Physically-Based Methods for Parametric Curve and Surface Reconstruction

by

Lian Fang

S. M., Naval Architecture Engineering
National Taiwan University
June, 1987

B. S., Naval Architecture Engineering
National Taiwan University
June, 1985

Submitted to the Department of Mechanical Engineering
in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy
in Mechanical Engineering

at the

Massachusetts Institute of Technology

May, 1994

Signature of Author _____
Department of Mechanical Engineering
March 1, 1994

Certified by _____
Professor David C. Gossard
Thesis Supervisor

Accepted by _____
Ain A. Sonin
Chairman, Department Committee

Physically-Based Methods for Parametric Curve and Surface Reconstruction

by

Lian Fang

Submitted to the Department of Mechanical Engineering
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Mechanical Engineering.

Abstract

The surface reconstruction schemes currently used in the automobile industry often require the user to identify connectivity information for data points and to ensure the surface's fairness. This slows down the design process. In view of these deficiencies, this thesis developed new methods for parametric curve and surface reconstruction using the variational principle and a finite element based geometry representation. The proposed methods simulate the deformations of an elastic beam and a plate under the application forces by minimizing energy functionals which are specifically designed to suit computer aided design requirements. The energy functional is composed of a geometric irregularity measurement and a closeness measurement that relates the given data points and the geometry. The irregularity of the geometry is assessed by both global and local measurements. The closeness measurement is a weighted sum of the squared minimal distances between the data points and the geometry. This significantly reduces the amount of connectivity information required as input. Successive quadratic programming is used to find the solution for both curve and surface reconstruction. The appeals of the proposed methods are: parametric geometries can be reconstructed from scattered and noisy data; the user's effort in identifying the data's connectivity information is greatly reduced; geometric fairness is built into the reconstruction process; general n-sided surfaces are modeled without using trim curves; and integration with downstream design and analysis software is highly feasible.

Thesis Advisor: Professor David C. Gossard
Thesis Committee: Professor Nicholas M. Patrikalakis
Professor John R. Williams

To my dear parents,
my loving wife
and
our lovely daughter

Ackoweldgements

I would like to express my sincere gratitude to many people who gave me invaluable support in spirit or in action throughout the duration of my Ph.D. study at MIT.

First and foremost, I thank my thesis advisor Professor David C. Gossard for his support, encouragement and guidance without which this thesis would not have been possible. Other thesis committee members, Professor Nicholas M. Patrikalakis and Professor John R. Williams, have also greatly contributed to the direction and quality of this work. I would like to express sincere gratitude to them for their encouragement, help and constructive criticism.

This research was supported by Ford Motor Co. Ltd. Their financial support is sincerely appreciated. I would also like to express my appreciation to the Computer-Aided Design Laboratory, which Professor Gossard founded and directs, for providing excellent facilities and an environment conducive to research, which make my study at MIT an exceptional experience.

My life at MIT was greatly enriched by friends at the CADLAB. Discussions with Kenji Shimada, Ashok Kumar, Barbara Balents and David Wallace were always helpful. In particular, I would like to thank Barbara Balents for her continuous help in the writing of this thesis and several related papers. I would also like to thank Buo-Yuan Hsu for his help in producing the slides for the thesis presentation.

Finally, I would like to express my appreciation to my parents for their continuous support; my wife, Hueiyun Lin, for her encouragement and patience; our little girl, Michelle Fang, for her corporation during the writing of the thesis and the joy she has brought us.

Table of Contents

List of Figures

List of Tables

CHAPTER ONE

Introduction

Free-form surfaces, differed from the traditional sweep surfaces or revolutional surfaces, are surfaces that are doubly curved, smoothly varying and aesthetically pleasing. The modeling of free-form surfaces has become a major research topic in science with direct applications in many engineering disciplines. For example, in applications such as the design of automobiles, ships and aircrafts, the modeling of molded and stamped shapes enables the shapes' information to be computed so that NC tool paths can be automatically generated in the making of dies and stamps. For computer graphics, in rendering an object the normal vectors are often required so that a realistic image can be generated by complicated rendering methods. The precise calculation of the normal vectors requires the existence of a mathematical description of the object being rendered. A complete mathematical description of an object is also necessary to generate meshes which will serve in engineering analysis, for example, the finite element method.

Two major applications requiring mathematically complete description of free-form surfaces are curve/surface reconstruction and free-form surface design. The curve/surface reconstruction problem is, given a set of discrete descriptions such as the positions and normal vectors of the target curve or surface, generate a complete mathematical description of that geometry. This process is also known as the curve or surface fitting problem. The resulting geometry need not interpolate the given data. An approximation scheme will be preferable under some conditions depending on the nature of the data

points and the application's purpose. Free-form surfaces design is a problem with more flexibility. Given a specific way to define the free-form surface, create a surface or modify an existing one to meet the users' requirements. Although surface fitting techniques are often used as surface defining tools, surface fitting systems and surface design systems are generally separate and independent due to their distinct functionalities and the different approaches to achieve the respective goals.

The bulk of this thesis is dedicated to developing new methods using the variational principle and finite element based geometry representation to reconstruct curves and surfaces from noisy and scattered data points with less information than is used in the traditional methods. The variational principle has been used in the surface design area, resulting in a methodology now known as "variational surface design". By using the same principle to govern the underlying mathematics of both the reconstruction and design, the curve/surface reconstruction and surface design systems, conventionally separate and independent, can be unified and integrated, which will facilitate the design process in the automobile industry.

1.1 MOTIVATION

This thesis was motivated by the deficiencies observed in the current design process of the automobile industry. As illustrated in figure 1.1, the design process starts with the making of a full-sized clay model from the prototype design. The clay model is then digitized by a three dimensional coordinate measuring machine. These data points are entered into the computer to create a mathematical model of the automobile's outer body, which is subject to further modification until satisfied. Once the shape of the outer panel is completed, the inner panel which provides structural strength to the outer panel is created by adding more features such as grooves and holes onto an offset surface of the outer panel. After finishing the geometric design, the outer and inner panels are divided into a finite number of elements, called a mesh, so that engineering analysis such as structural strength tests, vibration tests and aerodynamic tests can be performed via the finite element method (FEM) or boundary element method (BEM). When the design fails to satisfy these structural or aerodynamic requirements, the design team must return to the outer panel design and repeat the processes until the structural and aerodynamic performance are satisfactory.

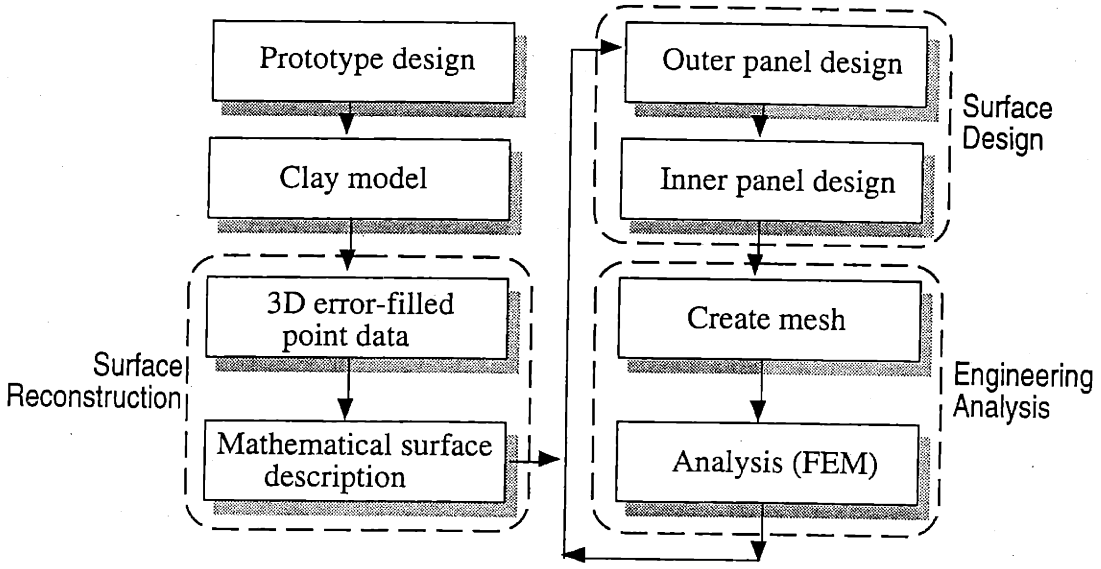


Figure 1.1: The typical design procedure in the automobile industry

At least two deficiencies, which make the entire design procedure time-consuming and expensive, are observed:

- *Slow surface reconstruction process*

Surface reconstruction is a process which converts the digitized data into a surface with a complete mathematical description. Conventionally, surfaces are created by blending a curve network which has been fitted from series of data points. The existing methods for reconstructing curves from data points all require full connectivity information for the data points. The fairness of the curve is afterwards achieved by adjusting the control points interactively. Both identifying connectivity information and curve fairing require a lot of human interference, and thus slow down the process. Moreover, a fair curve network does not guarantee the fairness of the surface being blended. Thus, designers are required to manipulate the control points controlling the surface's shape, which will be even more time-consuming.

- *Inefficient surface design methodology*

The design of a surface involves its creation and modification. Surfaces can be created using various techniques, for example, generating from a set of control points,

blending a curve network and surface fitting. How to modify a surface is often greatly restricted by the way that the surface is defined. The current dominant form in representing free-form surfaces is the control point method, typified by NURBS surfaces. One of the advantages of using NURBS surfaces is that the local control property enables users to change the shape locally. From another viewpoint, this local control property makes it very difficult to control the surface's global geometric properties, like convexity and fairness. Furthermore, since the control points generally do not lie on the surface, it is difficult to imagine what control points should be changed in order to achieve the desired modification. Using control points to change surface's shape might require one to manipulate most or even all of the control points to achieve a desired modification. In [Andersson 88], 170 hours had been reported for a skillful operator to form the surface description of a 700 series Volvo car hood. For designing surfaces as complicated as the inner panel, it might take up to several months.

1.2 METHODOLOGY REQUIREMENTS

In view of the deficiencies described above, the development of a methodology which enables the reconstruction and design of curves and surfaces through a unifying and efficient fashion is essential. Some desirable capabilities of such a methodology are addressed below.

- *Geometries should be represented in parametric form.*

Representing geometries (curves and surfaces) in parametric form is a basic requirement. The parametric representation has several advantages over the traditional explicit representation. First, it enables the representation of curves and surfaces with more complicated shapes than explicit representation. For example, curves in three dimensions can be given by a simple mathematical equation instead of being defined in terms of their projections onto two mutually perpendicular coordinate planes. A closed curve can be represented by a single equation and the problem of calculating the vertical tangent in a fixed coordinate system can also be avoided. Secondly, it enables coordinate transformations such as translation and rotation to be performed very simply. Finally, theories in differential geometry can be applied to calculated more detailed information like the normal vector of a surface for subsequent manufacturing purposes.

- *The reconstruction method should be able to reconstruct parametric geometries using less connectivity information and should ensure geometries' fairness as well.*

This requirement is a consequence of the previous one. All existing parametric curve reconstruction methods require full knowledge about data points' connectivity. For current parametric surface reconstruction methods, a lot of connectivity information must be identified also and the resulting surface is limited to be quadrilateral. For reconstructing parametric curves and surfaces free of such restraints, new methods should be devised.

- *The reconstruction method should be able to deal with noisy data.*

In some applications, errors occur in the process of gathering the data points. For example, digitizing an original clay model with worn out parts. When data points are subject to measurement errors, the reconstruction method should be able to deal with the noise so that the resulting surface will not follow the noise and appear to be undulatory.

- *The design method should be able to reduce user input and maintain fairness easily.*

The design process is slowed down partially due to the inefficient design method using existing technology. Users have to control many parameters to achieve a desired change or maintain fairness. An efficient design method should enable users to apply both local and global control on the surface while ensuring the fairness and keeping the user's input as minimal as possible provided the desired modification can be achieved.

1.3 THESIS GOAL

Given these requirements, what we need is a methodology using proper combination of governing principle and surface representation which enables a fair, generally nonquadrilateral parametric surface to be reconstructed from a set of scattered, noisy data with only partial connectivity information specified. Afterwards, this methodology should also enable the reconstructed surface to be modified via an efficient manner requiring less user input while maintaining fairness. The variational principle and finite-element based geometry representation have been chosen as the means for accomplishing this task.

The variational principle has been used in the design of free-form surfaces [Celniker 90] [Welch 92], resulting in a methodology now known as "variational surface design". This design paradigm is able to fulfill the fourth requirement listed above. However, none of the existing methods for reconstructing curves and surfaces is able to satisfy the rest of the requirements. Hence, the goal of this thesis is to develop methods for reconstructing parametric curves and surfaces, using the variational principle, from scattered and noisy data points with only partially specified connectivity information. The development of such a methodology will not only fulfill the above mentioned requirements but also enables the seamless integration with the variational surface design system, which will facilitate the design process of current automobile industry.

1.4 PROBLEM STATEMENT

Generally, the problem of curve and surface reconstruction can be stated as

"*Given a set of discrete descriptions of a target curve or a surface, generate a better description of that geometry, which enables more information to be explored.*"

The discrete descriptions of the target geometry could be the positions, the tangent directions, the normal vectors or curvatures. In many applications positional data are the easiest information to obtain, and it is often difficult or expensive to estimate additional information such as tangents or curvatures. A geometry's description is said to be a better one if it can be used to obtain more information than the given discrete ones. How *good* a description should be depends on the application's purpose. When visualizing the geometry's shape is the main goal, a polygonal representation with sufficient resolution is often enough. If the application expects to obtain more information, a complete mathematical description of that geometry is required.

In this thesis, the discrete descriptions of the geometry will be limited to positional data only. Measurement of inaccuracies which occur, for example, when digitizing an original model with worn out parts, are also considered. The geometry's representation is chosen to be a complete mathematical description in parametric form because of the necessity to obtain detailed information about geometry properties for future design and manufacture purposes. The curve and surface reconstruction problem that will be addressed in this thesis is then restated as

"Given a set of scattered, possibly error-filled data points, $\boldsymbol{P}_i(x_i, y_i, z_i),\ i = 0, \cdots, N-1$ *representing a curve or surface, generate a complete parametric representation of the geometry."*

1.5 APPROACH OVERVIEW

The variational principle, borrowed from statics and also known as principle of virtual work, can be stated as follows [Crandall 68]: *A geometrically admissible configuration of a mechanical system is in an equilibrium state if, and only if, the variational indicator*

$$\text{V.I.} = \sum_i \boldsymbol{f}_i \cdot \delta \boldsymbol{R}_i - \delta V \tag{1.5.1}$$

vanishes for arbitrary geometrically admissible variations of configuration. In the above equation, $\delta \boldsymbol{R}_i$ is the geometrically admissible vector displacement of each degree of freedom defining the system; $\boldsymbol{f}_i$ represents a vector force acting on the corresponding degree of freedom and V is the system's potential energy. To apply the variational principle on the creation of curves and surfaces, the degrees of freedom that define curves or surfaces are viewed as the configuration of a mechanical system. Curve's or surface's shape is decided by the admissible configuration in the equilibrium state, namely, the one making the variational indicator vanish. Speaking more precisely, if there are no external forces acting on the geometry, its shape is decided by minimizing the system's potential energy functional which should be designed depending on the application's characteristics.

Parametric curves and surfaces are mappings from their parametric space into the three dimensional object space. For curves the parametric space is a one dimensional interval and for surfaces a two dimensional bounded region. In a finite-element based parametric geometry representation, the geometry's parametric domain is divided into a finite number of elements. The geometry is represented as a composition of different mappings of these elements with a certain continuity. The collection of subdivisions of the parametric domain is often called a *mesh*. The parametric domain of a FE-based parametric curve is a one dimensional mesh characterized by the parametric length of each element. The parametric domain of a FE-based parametric surface is a two dimensional mesh characterized by its range, shape and the element's shape. In this

thesis, a triangular element is used since domains of any shape can be decomposed into triangles and any number of edges can meet at a node in any pattern of angles in a triangular mesh. Note that the term mesh or element is interchangeably used to indicate entities in parametric space as well as mappings in object space.

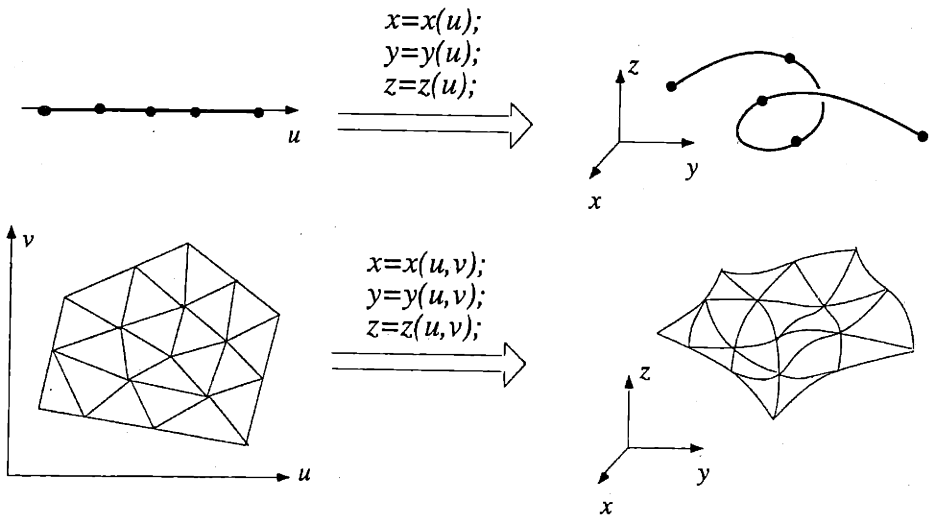


Figure 1.2: The finite element based representation for parametric curves and surfaces

Since the same principle is used in both the reconstruction and design of surfaces, the developed methods for curve and surface reconstruction can be used as design tools for designing free-form surfaces as well. The use of FE-based geometry representation also reduce the effort in generating a mesh for the engineering analysis in the later design stage. By the parametric surface reconstruction and design using the variational principle with a FE based geometry representation, users are allowed to

- use less connectivity information for reconstruction process.
- model surfaces with more complexity than traditional quadrilateral surfaces.
- maintain surface fairness simultaneously within the reconstruction or design process.
- reduce input for both reconstruction and design.
- analyze structural or aerodynamic performance with the existing mesh.

1.6 NOTATION AND SCOPE OF WORK

Throughout this thesis normal italic letters, such as f, $f(x)$, F and $F(x)$ are reserved for scalar values or functions. Bold italic letters, such as $\boldsymbol{w}$, $\boldsymbol{w}(u)$, $\boldsymbol{S}$ and $\boldsymbol{S}(u,v)$ stand for vectors or vector functions. Bold uppercase letters, such as $\mathbf{K}$ and $\mathbf{Q}$ will indicate matrices. The operator $|\cdot|$ represents the absolute value of a scalar value and $\|\cdot\|$ the norm of a vector.

The rest of this thesis is organized as follows. Chapter 2 will give a survey of previous work in curve reconstruction, surface reconstruction, surface design and triangular interpolants. In Chapter 3 we will develop a new method for reconstructing parametric curves from scattered and noisy data. The potential energy functional is composed of two terms: one that measures the closeness between data points and the parametric curve and the other that measures the irregularity of the curve in both the global and local senses. Interpretation and physical meanings of the functional are given in different viewpoints. By only assuming the knowledge of end points, the curve's shape is obtained by minimizing the energy functional via successive quadratic programming. The quality of the reconstructed curve are evaluated by the accuracy of fit and the distribution of curvature and curvature's derivative.

In Chapter 4, a new method for reconstructing general n-sided parametric surfaces from scattered and noisy data is presented. The given energy functional is also composed of a closeness measurement and an irregularity measurement. The input scattered and error-filled data are divided into two groups: boundary points and interior points. Boundary points are those points which are supposed to represent the surface's outer boundaries and inner boundaries, like holes and character curves. While the interior points are only used to indicate the surface's shape, the boundary points are also used to construct the surface's parametric space. Those points representing each boundary curve must be identified in advance, but within each series of points no connectivity information other than the end points are needed. The parametric curve reconstruction method is used to build the surface boundaries. A surface constrained by the existing boundaries would be found by minimizing the energy functional. Then this surface is sculpted by the spring forces coming from the interior points. Several methods for interrogating the quality of the reconstructed curve are also introduced.

In Chapter 5, some common technical issues encountered in the development of curve and surface reconstruction methods will be discussed. They are constrained nonlinear minimization, minimum distance computation, geometric constrains enforcement and bandwidth reduction. Results of the curve and surface reconstructed from unorganized data points by the proposed methods are presented in Chapter 6. Finally, Chapter 7 summarizes the thesis goal and approach and discuss some possible future work. Properties of the curvatures of parametric curves and surfaces are introduced in the appendix.

CHAPTER TWO

Previous Work

This chapter attempts to provide a survey of the previous research in or related to the different areas involved in this thesis: (1) curve reconstruction, (2) surface reconstruction, (3) surface design methods, and (4) triangular interpolants. In each area, various methods are categorized so that understanding can be gained in a more systematic manner. Though surface design is not the main goal of this thesis, since the developed methods for curve and surface reconstruction can be directly used as surface design tools, it is felt that a generally survey of the current surface design paradigms is still worthwhile. The immense literature in these areas makes it impossible to cover all the material here. Hence this chapter should be considered as an overview of each of the areas. Broadness rather than depth will be emphasized.

The rest of this chapter is organized as follows. Section 2.1 reviews methods used for the curve reconstruction. Methods are categorized into two major groups, parametrization methods and minimal-energy curves. Some other techniques which are applied after reconstruction to ensure the curve's fairness or to reduce data storage are also introduced. Section 2.2 reviews surface reconstruction methods categorized by surface's representation. They are explicit, parametric, implicit and simplicial representations. Section 2.3 reviews some free-form surface design methods including the nodal-basis method, the control point method, the lofting method, the transfinite method and the variational method. Section 2.4 introduces some triangular interpolants. Section 2.5

summarizes the limitations of the existing methods in the first three areas, which lead to the technical goals to be achieved in this thesis.

2.1 CURVE RECONSTRUCTION

The problem of reconstructing a parametric curve to a set of data points has been addressed for many years. The main purpose is to obtain a curve of good fairness which passes through or lies close enough to all data points. Although there is no formal definition for fairness, a curve's quality is generally judged according to its curvature distribution and many methods have been developed to generate curves with pleasing curvature distributions. Methods in curve reconstruction can be broadly divided into data interpolation and data approximation (or data smoothing). The data interpolation schemes generate curves exactly passing through all given data points. The data smoothing schemes only generate curves lying close enough to the data points. When data points are subject to error measurement, data smoothing is more appropriate.

The shape of a parametric curve fitted from data points is decided by how the curve being created is related to the data points, namely, the points' parametrization and the vectors which define the curve. Naturally, methods can be divided into two categories: how to determine the "optimal" parametrization and how to determine the shape-defining vectors. For methods of the first type, once the parametrization is made, curve's shape is simply determined by a least square fit or by introducing certain boundary conditions, like natural end conditions, to solve a linear equation set. For methods falling in the second category, specific routines are applied to determine the shape-defining vectors based on a pre-selected parametrization. There are still other methods used to improve fairness and reduce data storage for oversampled data without losing accuracy. They are treated as post processors since they are generally applied when the reconstruction process is done.

In the following, diverse parametrizations will be introduced and discussed in section 2.1.1, and the minimal-energy concept used to determine the shape-defining vectors will be discussed in section 2.1.2. Section 2.1.3 will discuss some "post-process" methods including fairing and merging.

2.1.1 Parametrization

Many methods have been developed to seek a good parametrization of data points based on the assumption that they are ordered. The simplest one is uniform or equidistant parametrization which assigns parametric values to each point uniformly

$$u_i = i \quad , 0 \le i \le (N-1) \tag{2.1.1}$$

This method is too simplistic to meet the requirements of most applications since it ignores the data points' distribution. A better choice which takes the effect of the data point distribution into consideration is chord length parametrization. It assigns parametric values by

$$u_i - u_{i-1} = \left\| \boldsymbol{P}_i - \boldsymbol{P}_{i-1} \right\| , \quad 1 \le i \le (N-1) , \; u_0 = 0 \tag{2.1.2}$$

Chord length parametrization usually produces better results than uniform parametrization, although not in all cases. It has been proved that curves with corners[1] cannot be produced by using chord length parametrization, but possibly occurs while using uniform parametrization [Epstein 76]. Chord length is actually an approximation of the true arc length of the fitted curve. Another alternative is circular arc parametrization in which the arc length is estimated by fitting a circle through each group of three consecutive points [Earnshaw 71]. The parametrization can even be improved by calculating the arc length of the fitted curve and doing the fitting iteratively as suggested in [Earnshaw 71] and [de Boor 78]. Marin proposed a method in which the "optimal" parametrization is found by minimizing the integral of the squared second derivative of the curve [Marin 84]. Hoschek proposed a so-called "intrinsic" parametrization by which a particular parametrization is used to fit the curve first, the parametrization is then improved by a Newton-like iterative approach so that the error vectors are orthogonal to the approximation curve [Hoschek 88]. This method generally produces very good result, but is only suitable for generating approximation curves.

---

[1] A corner is a point on a curve where the tangent direction changes discontinuously. When it changes in 180 degrees, it is called a cusp.

Another parametrization known as centripetal parametrization was derived from a "centripetal model" by Lee [Lee 89]. This parametrization is based on a physical analogy to a car driving along the interpolating curve. It assigns parametric values by

$$u_i - u_{i-1} = \|P_i - P_{i-1}\|^{1/2} \ , \ 1 \le i \le (N-1) \ , \ u_0 = 0 \qquad (2.1.3)$$

In most cases, this parametrization has better results than chord length parametrization. Actually, an equation similar to equation (2.1.3) and comments had appeared in [Hosaka 78] without any explanation about its good performance. The uniform, chord length and centripetal parametrizations can be written in a more general form stated below, with e=0.0, 1.0 and 0.5 respectively.

$$u_i - u_{i-1} = \|P_i - P_{i-1}\|^{e} \ , \ 1 \le i \le (N-1) \ , \ u_0 = 0 \qquad (2.1.4)$$

Lee also proved that only when e is taken between 0.5 and 1.0, then it is not possible for corners to appear in the interior of the interpolating curve [Lee 92] .

A more complicated parametrization method was developed by Foley and Nielson [Foley 89]. It sets

$$u_i - u_{i-1} = d_i \left[1 + \frac{3}{2} \frac{\hat{\Theta}_i d_{i-1}}{d_{i-1} + d_i} + \frac{3}{2} \frac{\hat{\Theta}_{i+1} d_{i+1}}{d_i + d_{i+1}} \right] ; \ 1 \le i \le (N-1), \ u_0 = 0 \qquad (2.1.5)$$

where $d_i = \|P_i - P_{i-1}\|$, $\hat{\Theta}_i = min(\pi - \Theta_i, \pi/2)$ and Θ_i is the angle formed by P_{i-1}, P_i and P_{i+1}. This parametrization is designed to be used for data points with very large variations. In [Farin 93], the centripetal parametrization is viewed as the "best" parametrization compared to uniform, chord-length and Foley's parametrizations since it compromises between the algorithm's computing cost and the result. In fact, there is probably no "best" parametrization since any method could be defeated by a suitably chosen data set. Cases where uniform parametrization gives better results than the other methods still exist. Uniform parametrization is also the only parametrization which is invariant under affine transformation of the data points.

Recently, nearly arc-length parametrization was proposed in [Wever 91] and [Wang 93]. Wever finds a cubic C^2 interpolating curve with unit tangent vector at discrete data

points by minimizing the integral of the squared second derivative of the cubic interpolating curve, constrained by the requirement of unit tangent vectors at the data sites. The variables are the tangent vectors at the data sites. Wang's approach is similar. The ordered data points are fitted by a cubic spline based on centripetal parametrization associated with natural end conditions. A quintic curve passing through the data points is formed. The first and second derivatives information at the data points are taken from the existing cubic curve and the parametrization is made by minimizing the deviation of the derivative's magnitude from 1. This technique generates a quintic interpolation curve which is nearly arc-length parametrized.

Almost all the parametrizations mentioned above are used to fit a Hermite-type curve to the data points, i.e., create one cubic or quintic polynomial between each two consecutive data points. If the interpolating curve is required to be a B-spline, the parametrization should depend on the knot vector. For the case of an open B-spline curve of order k, once the knot vector is selected, the parameters are assigned to data points by

$$u_i = \frac{1}{k-1}(t_{i+1} + t_{i+2} + \cdots + t_{i+k-1}) \; ; \; 0 \le i \le (N-1) \tag{2.1.6}$$

where the t_is are the knot values. This equation guarantees a nonsingular system matrix. From experience, when fitting a B-spline to data points, this equation almost always results in much better results than other methods. The selection of the knot vector can depend on the data points. Hartley and Judd proposed [Hartley 78, 80]

$$\begin{aligned} t_i &= 0 && ; \; 0 \le i \le k-1 \\ t_i &= t_{i-1} + \frac{\sum_{j=i}^{k+i-2} \|\boldsymbol{P}_{j+1} - \boldsymbol{P}_j\|}{\sum_{s=0}^{n-k} \sum_{j=s}^{k+s-2} \|\boldsymbol{P}_{j+1} - \boldsymbol{P}_j\|} && ; \; k \le i \le (N-1) \\ t_i &= 1 && ; \; N \le i \le N+k+1 \end{aligned} \tag{2.1.7}$$

This is probably the only parametrization method which takes into account the order of the resulting curve.

Some other research in parametrization methods includes [McConalogue 70] and [Cohen 88] which are listed here without discussion.

2.1.2 Minimal Energy Curves

In addition to methods finding the "optimal" parametrization, the minimal-energy spline has been used widely to obtain the "optimal" shape directly based on a specific parametrization by minimizing an energy functional associated with the curve. Data interpolation using minimal energy splines can track back to Schweikert's spline under tension [Schweikert 66] in which the curve's shape is determined by minimizing a functional representing the potential energy stored in the curve as

$$\int [f''(t)]^2 dt + s \int [f'(t)]^2 dt \tag{2.1.8}$$

where s is the tension parameter used to adjust the global tightness of the curve. Subsequent research using minimal-energy splines differed in the form of the energy functional to be minimized. For example, the ν-spline proposed by Nielson [Nielson 74] minimized

$$\int [f''(t)]^2 dt + \sum_{i=0}^{N-1} v_i [f'(t_i)]^2 \tag{2.1.9}$$

and the τ-spline generalized from the ν-spline by Hagen [Hagen 85] minimizes

$$\int [f'''(t)]^2 dt + \sum_{i=0}^{N-1} v_{1i} [f'(t_i)]^2 + \sum_{i=0}^{N-1} v_{2i} [f''(t_i)]^2 \tag{2.1.10}$$

Other research falling in this category includes [Foley 87a] [Pottmann 90] and [Celniker 90b].

Interestingly, much less work has been published in data smoothing using minimal energy splines than in data interpolation. To author's knowledge, the earliest work in this area is by Reinsch [Reinsch 67]. The problem of smoothing a set of given planar data points (x_i, y_i), $i = 0, \cdots, N-1$, was cast as a constrained minimization problem as

$$\begin{aligned} &\text{Minimize} \quad \int_{x_0}^{x_{N-1}} g''(x)^2 dx \\ &\text{Subject to} \quad \sum_{i=0}^{N-1} \left(\frac{g(x_i) - y_i}{\delta y_i} \right)^2 \le S \end{aligned} \tag{2.1.11}$$

where $g(x)$ is a C^2 function, $\delta y_i > 0$ controls the influence of each point and $S \geq 0$ controls the extent of smoothing. Choosing $S = 0$ leads back to the interpolation problem. Hosaka used a similar approach on smoothing three dimensional points and extended it to the smoothing of a rectangular curve network [Hosaka 78]. Nowacki applied Hosaka's method to generate fair curves in a rectangular curve network, a GC^1 surface is then generated by interpolating through the mesh [Nowacki 88]. Basically, the functional being minimized includes not only the curve's potential energy but also an error term measuring the distances between the approximating curve and the data. This is a more general formulation since the error term will vanish for data interpolation problems.

Note that in all the minimal-energy curves mentioned above, the energy functional is linearized from the actual energy stored in a deformed beam and is not invariant under different parametrizations. Nonlinear minimal energy curves which minimize the squared curvature had been used. Discussion of nonlinear splines begins with Birkhoff and de Boor [Birkhoff 65]. Properties of the nonlinear minimal energy splines was further investigated and explored by many researchers [Mehlum 69] [Jerome 75] [Golomb 82] [Horn 83] [Jou 92]. Another nonlinear spline minimizing the integral of the squared curvature's derivative with respect to the arc length of the curve was also proposed [Ohlin 87] Moreton applied this nonlinear spline to a network of quintic Hermite curves [Moreton 91]. Though good results were achieved, the computing cost increased significantly due to the presence of nonlinear terms.

2.1.3 Fairing and Merging

Fairing is a postprocess for improving the fairness of an existing curve after interpolation. The positions of the data points being interpolated are changed. The general procedure is to first interpolate the given ordered points based on a particular parametrization. Data points associated with large discontinuities in the high 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. The process can be automated by fairing the place where the largest discontinuity in curvature or the third derivative occurs. But it should be used with care since a peak in curvature may be a feature of the curve rather than an indication of unfairness. In this case, it should not be smoothed out. Readers are directed to [Renz 82] [Kjellander 83b] [Farin 87] and [Sapidis 90] for more details.

When fitting a curve to a set of data points, the data points are often oversampled. Consequently, a composite curve with too many pieces is created. For convenience in subsequent data storage and handling, it is desirable to represent the curve by a more concise representation. The merging process generally merges two consecutive curve elements into one with the same or even lower degree to avoid the unnecessary oscillations. The new curve with fewer elements after merging must not deviate from the original curve or data points more than a prespecified tolerance and must preserve position and tangent continuity. Readers are directed to [Kallay 91] and [Chou 93] in which methods are developed to merge composite cubic curves efficiently and successfully. However, the methods do not preserve the C^2 continuity of the original curve.

2.2 SURFACE RECONSTRUCTION

Three dimensional surface reconstruction is an important problem in many fields like geology, medical imaging, industry, computer vision, etc. Its many applications have stimulated the development of diverse methods for solving the problem since the early sixties. Different methods, developed to fulfill different application requirements, can be categorized by several criterion. As in curve reconstruction, data interpolation and data smoothing schemes differ in whether the data points lie exactly on the resulting surface. Methods can be also characterized as global or local depending on whether the resulting surface changes globally or locally in response to the change of a single data point. Another criterion that can be used to categorize surface reconstruction methods is the surface representation which generally is chosen to suit the application. For example, if visualization is the main goal as in medical applications, the surface is often reconstructed as a collection of small triangular facets, called simplicial surfaces. A full mathematical description of the surface is not necessary and may be redundant. Conversely, a full mathematical description of the surface is essential for producing contour plots for geological applications, rendering complicated objects in computer graphics and facilitating further design and analysis in industrial applications. For those techniques which generate a complete mathematical description of the surface, "function reconstruction" might be a more specific name, allowing it to be distinguished from the other techniques which could be called "shape reconstruction".

In the following, surface reconstruction methods are categorized by surface's representation. Subsection 2.2.1 to 2.2.4 will be devoted to the introduction of reconstruction methods for explicit surfaces, parametric surfaces, implicit surfaces and simplicial surfaces respectively.

2.2.1 Explicit Surface Representation

Reconstructing surfaces in explicit form has received an enormous amount of attention in the field of approximation theory since the sixties and even lately in computer vision research. This problem can be stated as "Given scattered data points (x_i, y_i, f_i), i=0,..., N-1, construct a smooth (at least C^1) function $F(x, y)$ so that $f_i = F(x_i, y_i)$ for i=0,..., N-1, Many basic ideas and different approaches have been described by Schumaker and Barnhill in survey papers [Schumaker 76] and [Barnhill 77]. Many developed methods have been tested and compared by Franke [Franke 79, 82, 86]. An extensive bibliography is listed in [Franke 87] Interested readers should refer to these excellent papers and the references therein. Here these methods are further divided into four different types: (1) Inverse distance weighted method, (2) Global basis function method, (3) Local basis blending method, and (4) Triangular finite element based method.

2.2.1.1 Inverse Distance Weighted Methods

In the literature, this method has become known as Shepard's method although the method was used before Shepard's original publication in 1968. The basic idea is to take the interpolant to be a weighted sum of the value of the ordinates. All methods in this category can be viewed as generalizations or variations of Shepard's method. The basic Shepard's method is

$$F(x,y) = \begin{cases} \sum_{i=0}^{N-1} w_i(x,y) f_i \Big/ \sum_{i=0}^{N-1} w_i(x,y) & \text{, when } d_i \neq 0 \text{ for all } i \quad (2.2.1a) \\ f_i & \text{, when } d_i = 0 \quad (2.2.1b) \end{cases}$$

where $w_i(x,y) = d_i^{\mu}$ and $d_i = ((x-x_i)^2 + (y-y_i)^2)^{1/2}$. The typical value of μ is taken as -2 but other values can be used and may even differ for different points. The function $F(x, y)$ constructed in this way is not a polynomial or rational function. Its behavior in the vicinity of (x_i, y_i) depends on the value of μ. It can be shown that for $0 < \mu \leq 1$, $F(x, y)$

has cusps at the data sites and for $\mu > 1$. $F(x, y)$ has "flat spots" at the data points, namely, the partial derivatives vanish there.

Shepard's method is a global method by which means changing one data point affects the shape of $F(x, y)$ globally. The common drawback of global methods is that as the number of data points increases, the computing cost will drastically increase correspondingly. A method to localize the data points' support is to replace the inverse-distance weighting functions by a new one which is zero outside a certain disk with radius R and centered at (x_i, y_i)

$$w_i(x,y) = \left(\frac{(R-d_i)_+}{Rd_i} \right)^2 \quad , \quad (R-d_i)_+ = \begin{cases} R-d_i & \text{if } (R-d_i) \geq 0 \\ 0 & \text{if } (R-d_i) < 0 \end{cases} \tag{2.2.2}$$

The deficiency of the "flat spots" can also be overcome by replacing f_i in equation (2.2.1a) by a local approximation using derivative information either given or generated from the data. The interpolation function would take the form as

$$F(x,y) = \sum_{i=0}^{N-1} w_i(x,y) \left[f_i + \left(\frac{\partial f}{\partial x} \right)_i (x-x_i) + \left(\frac{\partial f}{\partial y} \right)_i (y-y_i) \right] \Big/ \sum_{i=1}^{N} w_i(x,y) \tag{2.2.3}$$

2.2.1.2 *Global Basis Function Methods*

These methods can be characterized by the following idea. Select N functions $\phi_i(x,y)$ and then determine the coefficients A_i so that the interpolating function

$$F(x,y) = \sum_{i=0}^{N-1} A_i \phi_i(x,y) \tag{2.2.4}$$

satisfies $F(x_i, y_i) = f_i$, for i=0,..., N-1. The coefficients A_i can be easily calculated provided the system matrix is nonsingular. Notice that the basis functions $\phi_i(x,y)$ could be chosen to have local support, but since the coefficients have to be obtained through solving a linear equations set, the method is still considered as global.

Although the procedure described above is simple, the appropriate choice of the functions $\phi_i(x,y)$ is not particularly easy to make. For example, polynomial functions would be the first type of function to come to mind. But there is still some leeway in the

choice of which powers of x and y to use. Even if this can be determined, it is not easy to guarantee that the system matrix is nonsingular. When the number of data points is large, the system matrix often becomes ill-conditioned. Finally, polynomials of moderately large degree exhibit a considerable oscillatory character, and the resulting surface is often too undulatory to be acceptable even though it is C^{∞}.

A common way to select the basis functions is to associate each basis function with each data point and compose $\phi_i(x,y)$ as an univariate function of d_i, where $d_i = ((x-x_i)^2 + (y-y_i)^2)^{1/2}$. A famous method of this kind is Hardy's "multiquadratic" method [Hardy 71] in which the basis functions are taken to be $\phi_i(x,y) = (d_i^2 + r^2)^{1/2}$. Here r is a parameter to be specified by users. This method is quite stable and yields consistently pleasing and smooth surfaces. Methods of this sort taking the basis function as a rotation invariant function of variable d_i are known as the radial basis function methods. Nearly as good results can be obtained using the "reciprocal multiquadratic" method with $\phi_i(x,y) = (d_i^2 + r^2)^{-1/2}$. However, the choice of r is more crucial since smaller values of r will result in a surface of peaks and dips at each data point. A more general formulation of the radial basis function method is to incorporate a set of polynomial functions in addition to the original radial basis functions. This enables the resulting surface to retain polynomial precision.

Two methods which use rotation invariant basis functions similar to the multiquadric method are due to Duchon. One of the basis functions is $\phi_i = d_i^2 \log d_i$ and the other is $\phi_i = d_i^3$. Both of the interpolation functions include a linear polynomial function. The resulting interpolant from the former is proved to be the solution which minimizes the functional

$$\int_{R^2} \left(\left(\frac{\partial^2 F}{\partial x^2} \right)^2 + 2 \left(\frac{\partial^2 F}{\partial x \partial y} \right)^2 + \left(\frac{\partial^2 F}{\partial y^2} \right)^2 \right) dxdy \qquad (2.2.5)$$

in a certain Hilbert space, and is called a "thin plate spline". This thin plate spline was previously discovered by Harder and Desmarais [Harder 72] who used the name "surface spline".

2.2.1.3 *Local Basis Blending Methods*

If the number of data points is too large to be treated globally, a method that deals with data points locally may be desirable. The main idea of such methods is to blend local interpolants of the data points in such a way that a smooth global interpolant is achieved. The interpolation function will generally take the form

$$F(x,y)=\sum_{i=0}^{N-1} w_i(x,y)Q_i(x,y) \tag{2.2.6}$$

where $w_i(x,y)$ are a set of weighting functions which are nonnegative, have finite local support, and sum to one for all (x_i,y_i). The local interpolant $Q_i(x,y)$ has the property that $Q_i(x_i,y_i)=f_i$. For suitably chosen $w_i(x,y)$, only a few terms are nonzero in the sum, so the evaluation of $F(x,y)$ is fast. For example, $Q_i(x,y)$ can be the quadratic polynomial interpolating $f(x,y)$ at (x_i,y_i) and the five nearest neighbors to (x_i,y_i) from the data set, and the weighting function could be

$$w_i(x,y)=H_i(x,y)\Big/\sum_{i=0}^{N-1} H_i(x,y) \tag{2.2.7}$$

where

$$H_i(x,y)=\begin{cases}1-\left(\dfrac{d_i}{R_i}\right)^2\left(3-2\dfrac{d_i}{R_i}\right) & , d_i \le R_i,\\ 0 & , d_i > R_i\end{cases} \tag{2.2.8}$$

and R_i is the distance between (x_i,y_i) and its fifth closest neighbor. The Shepard's method with finite support weighting in equation (2.2.2) also falls in this category.

The support region need not be a circular disk centered at (x_i,y_i). A scheme using overlapping rectangles for the support regions with product cubic Hermite functions for the weighting functions has also been suggested. Use of rectangular support simplifies the problem of determining which terms are nonzero and thus results in a faster algorithm. Some methods, categorized as "triangle based blending methods" in [Franke 82], can be viewed as local blending methods with triangular support regions in which only three terms are nonzero. Triangulation of the convex hull of the point set $\{(x_i,y_i)\}$ is the first step, followed by a blending of the three local interpolants associated to each vertex of the triangle in which the interested point falls. The weighting functions are the typical finite element shape functions satisfying $w_i(x_j,y_j)=\delta_{ij}$.

2.2.1.4 Triangular Finite Element Based Methods

Methods of this sort can be generally accomplished in three steps: triangulation, partial derivatives estimation, and finally, assembly of discrete triangular surface elements. The first step, triangulation, is to triangulate the convex hull of the data points' projection on x-y plane. This topic has been treated extensively in the literature. Usually the Delaunay triangulation is thought to be the "optimal" one which created triangles as equilateral as possible. Once the triangulation is found, the interpolant is obtained by piecing together piecewise functions over each triangle with a certain continuity at the triangles' boundaries. This will usually necessitate the estimation of some partial derivatives at the data sites. Which derivatives are required depends on the triangular element used in the method. See section 2.4 for a brief survey of triangular interpolants.

Akima uses a 21 degrees of freedom (dof) quintic element which requires estimation of the first and second partial derivatives at the data points [Akima 78]. For evaluating the first derivatives at point $\boldsymbol{P}_0$, the nearest neighbors of the x-y planar projection of $\boldsymbol{P}_0$ are found. Two points, $\boldsymbol{P}_i$ and $\boldsymbol{P}_j$ $(i \neq j)$, are arbitrarily selected from the neighbors, and the cross product of vectors $\boldsymbol{P}_0\boldsymbol{P}_i$ and $\boldsymbol{P}_0\boldsymbol{P}_j$. are constructed. A vector sum of the cross product vectors is taken from all the possible combinations of $\boldsymbol{P}_i$ and $\boldsymbol{P}_j$. Then the first derivatives F_x, F_y at $\boldsymbol{P}_0$ are estimated as those of a plane that is normal to the resultant vector sum thus constructed. The same procedure is applied on the estimated first derivatives to obtain the second derivatives. Lawson's method [Lawson 77] is similar in spirit to Akima's except that the Clough-Tocher element is used and requires only the first partial derivatives which are obtained from a (local) quadratic approximation. The minimum norm network introduced by Nielson [Nielson 83] is an innovative way to estimate the partial derivatives. He uses a 9-dof cubic element with a rational correction to achieve a C^1 interpolant. The required first partial derivatives are obtained by assuming a cubic variation along each edge and minimizing the integral of squared second derivatives over all edges in the triangulation. This idea was generalized by Pottmann [Pottmann 91] to produce a C^2 surface. For methods in this category, the accuracy of the estimated derivatives are very important and have a pronounced effect on the visual aspects of the surfaces as well as the accuracy.

In the past decade, this problem also attracted the attention of the computer vision community. A *visual surface* is reconstructed from the range data of stereo image pairs.

The surface representation remains explicit since the scene perceived by visual systems is always an intensity map on a 2D domain. The most important requirement of surface reconstruction methods in computer vision is that they should be able to deal with surface position or orientation discontinuities which do not occur in approximation theory and computer aided design. Barrow and Tenebaum [Barrow 81] were among the first to introduce the surface reconstruction problem into computer vision research. Since then much work has been devoted to the surface reconstruction problem in computer vision. Many ideas now used widely in reconstructing visual surfaces from depth data or stereo-image pairs originated from the community of approximation theory discussed above. For example, the thin plate spline listed in equation (2.2.5) was used in Grimson's early work in computer vision [Grimson 81]. Terzopoulos's control-continuity stabilizing function [Terzopoulos 86, 88], used to control the smoothness and continuity of the reconstructed surfaces is an extension from Schweikert's spline in tension and Duchon's thin plate spline. The two-stage scheme proposed by Sinha [Sinha 91] for reconstructing surfaces preserving discontinuity is similar to the work due to Foley [Foley 87b] except that an outlier rejection scheme is included in the first stage. A survey of literature about surface reconstruction in computer vision is available in [Bolle 91]. Since the interest of these researchers was solving computer vision problems, the surface's representation, ranges of topology, and geometric constraints are inappropriate for CAD applications.

2.2.2 Parametric Surface Representation

Although many papers deal with explicit surface fitting, this single-valued surface representation greatly limits their application in industry due to poor surface modifiability and the inability to model complicated surfaces. Parametric surfaces are able to model complicated shapes, like quadratic surfaces and free-form surfaces, most of which explicit representation can not. Using a parametric representation also allows users to modify the surface efficiently after reconstruction. Another advantage is that the reconstructed surface is coordinate system invariant so that the surface can be transformed by transforming the data points. The greatest difference between explicit and parametric surface reconstruction is that the former has a default mapping domain, the x-y plane, and parametrization of each point is already known. For parametric surface reconstruction, the parametric values on the mapping domain, often called the u-v plane, is not known *a priori*, and hence must be inferred from the data points. This often requires assumptions about the connectivity of data points. Once the parametrization is

obtained, the parametric surface can be obtained by a least-squared fit or minimizing an energy functional associated to the surface, similar to the curve reconstruction techniques.

Probably due to the difficulty of the parametric surface reconstruction, the number of related published papers is significantly less than that in the area of explicit surface reconstruction. The most common method to determine the parametrization is to assume that the data points form a conceptual rectangular grid [Earnshaw 71] as illustrated in figure 2.1. Speaking more precisely, given data points $\boldsymbol{P}_{ij} = (x_{ij}, y_{ij}, z_{ij})$, $i = 0, \cdots, m-1$ and $j = 0, \cdots, n-1$, the parametrization of each point can be made by using any parametrization method depicted in section 2.1.1. For example, for data points in each row or column, chord length parametrization would result in

$$u_{i+1,j} = u_{i,j} + \frac{\left\|\boldsymbol{P}_{i+1,j} - \boldsymbol{P}_{ij}\right\|}{\sum_{k=1}^{m-1}\left\|\boldsymbol{P}_{k+1,j} - \boldsymbol{P}_{kj}\right\|} \quad ; \quad v_{i,j+1} = v_{i,j} + \frac{\left\|\boldsymbol{P}_{i,j+1} - \boldsymbol{P}_{ij}\right\|}{\sum_{k=1}^{n-1}\left\|\boldsymbol{P}_{i,k+1} - \boldsymbol{P}_{ik}\right\|} \tag{2.2.9}$$

with $u_{00} = v_{00} = 0$. In [Pratt 85], the assumption about data point connectivity is less restrictive---instead of forming a rectangular grid, data points are only required to be ordered along some "tracks" on the surface being measured. This is illustrated in figure 2.2. Chord length parametrization is used to assigned v-parameter values from 0 to 1 along each track first. The same method is used to assign u-parameter values to all those data points for which v=0 and v=1. Finally, u-parameter values are calculated for all the interior points on the original tracks by linear interpolation between known u-values at their end points. This method was also used in [Sarkar 91] to reconstruct surfaces.

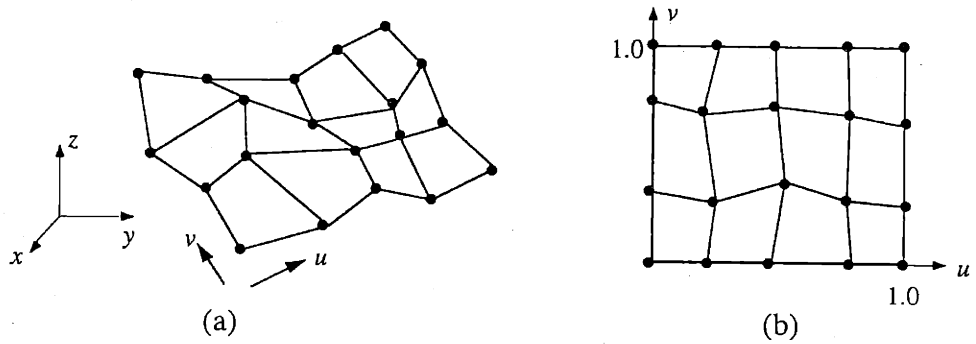


Figure 2.1: (a) Rectangular-gridded data points and (b) the parametrization

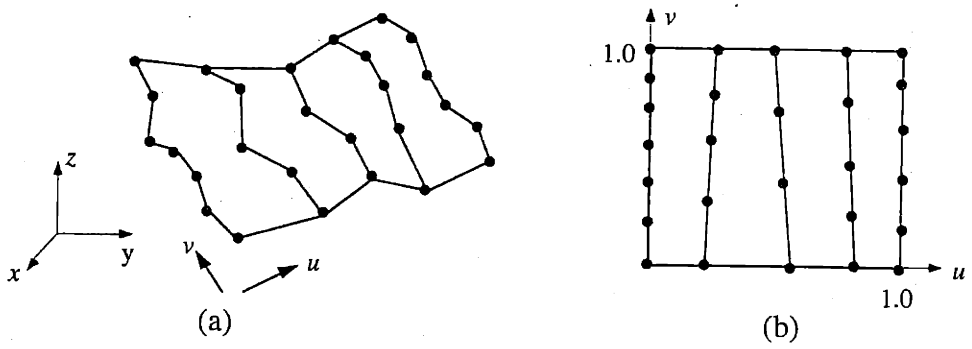


Figure 2.2: (a) Data points along some tracks and (b) the parametrization

Once the parametrization of the data points is determined, the surface's shape can be calculated by solving a set of linear equations with some boundary conditions or by minimizing an energy functional associated with the surface.

Another work in parametric surface reconstruction is done in [Andersson 88]. In this work, Bézier curves are fitted from ordered boundary points indicating the patches boundaries to form a curve network. The space enclosed by four Bézier curves are filled with a Bézier patch next. The entire surface is then improved by solving a linear programming problem with the Karmarkar algorithm to reduce the distances to the other interior points and ensure the surface's convexity. Note that some of the published papers use parametric interpolants, for example, Bézier patches or B-spline patches as the surface primitives but the x-y planar projection of the data points is taken as the (u, v) parametrization. Techniques falling in this category are not considered as parametric surface reconstruction methods since the algorithms do not produce results which are coordinate system invariant. [Mann 92] attempts to give a survey of the methods of parametric scattered data fitting using triangular interpolants. This paper presumes that the connectivity information for the data points is already known and a three dimensional triangular mesh has been established. The paper actually focuses on methods for generating smooth triangular interpolants from given triangular meshes.

Fairing methods also exist for improving surface's fairness. Readers can refer to [Kjellander 83a], [Lott 88] and [Alourdas 89]

2.2.3 Implicit Surface Representation

An implicit surface is a surface defined by $F(x,y,z)=0$. If $F(x,y,z)$ consists of only polynomials with real number coefficients, it is specifically called an "implicit algebraic surface". The primary advantage of using implicit definition for surfaces is the closure properties of the class of algebraic surfaces under modeling operators such as intersection, convolution, offset, blending, etc. Closure under modeling operators allow cascading repetitions without any need of approximation. Another advantage of using implicit surface representation is that the closeness measurement between data points and the surface (not necessarily the minimum distance) can be easily calculated by directly plugging the points' (x, y, z) components into the implicit function.

Quadric surfaces are one type of the implicit algebraic surfaces with degree not greater than 2. Many real world objects can be modeled by quadric surfaces and their Boolean operations. A more general implicit surface primitive is superquadrics which are extensions of the implicit quadric surfaces; however, they are not implicit algebraic surfaces. In contrast to the parametric surfaces that are defined by a set of space vectors, implicit surfaces are generally defined by some scalar parameters, for example, a quadric surface is completely defined by ten parameters. Most of the methods that reconstruct implicit surfaces attempt to extract these parameters which describe the global properties of the surface. For more detailed information in implicit surfaces reconstruction, readers are encouraged to refer to [Bolle 91] and the references therein.

2.2.4 Simplicial Surface Representation

When visualization is the major purpose for reconstructing surfaces, smoothness and fairness of the surfaces is not a major concern. For such applications, the surface can be represented by a collection of small triangular facets, called simplicial surfaces. The *marching cube* algorithm devised by Lorensen in 1987 [Lorensen 87] is the seminal work in this area. A subsequent method, called the *dividing cube* algorithm [Cline 88], is also used widely. This technique is very popular in medical imaging because of its ability to recreate very complex shapes such as a human skull, from X-ray or MRI scan pictures. Generally the amount of input data is required to be close to several thousand data points. A successful example using the marching cube method in reconstructing a simplicial surface from unordered points is demonstrated in [Hoppe 92, 93].

2.3 SURFACE DESIGN METHODS

The goal of a surface design system is to allow users to easily create and modify the shape of surfaces. In the past three decades, various methods have been devised for creating surfaces. However, only those for free-form surfaces are considered here; surfaces generated by revolution or sweeping operations will not be considered. Methods for creating free-forms surfaces are divided into two broad categories: surfaces from discrete points and surfaces from curves. Other techniques used to modify, rather than create, the surface are also discussed.

2.3.1 Surfaces from Discrete Points

2.3.1.1 Nodal Basis Method

Surfaces of this type are defined by interpolating a set of given space vectors, like the position vectors and tangent vectors, at a finite number of nodes on the surface. A good example is the bicubic Hermite surface which interpolates the position vector, the two tangent vectors and the twist vector at the four corner nodes of the surface. Other interpolants of this type can be found in finite element method literature. The strength of such surfaces is that the shape is explicitly represented through its geometric properties, which eases the enforcement of constraints on not only position but also tangent vectors. The weakness is the lack of intuition about the relation between the shape and the derivatives. Users are often confused about the effect of changing the values of tangent vectors and twist vectors, which makes it difficult to predict the shape of the surface under modification.

2.3.1.2 Control Point Method

The control point method is the current dominant method of defining free-form surfaces. The surface is defined by a set of points forming a conceptual rectangular (or triangular) mesh, called the control point mesh. The shape of the surface is modified by moving the control points. NURBS surfaces are one type of surface falling in this category. This scheme does not try to interpolate the control point positions. Instead the control points act to generally define the position of the surface.

One of the nice geometric properties engineered into surfaces of this type is that the surface's geometry depends only on the location of the control points in the immediate neighborhood. This local control property facilitates interactive shape design in a progressive manner, namely, users can modify the surface region-by-region. The portion that has been modified will not be altered during modification of other regions. However, it also makes it very difficult to control the surface's global geometric properties like convexity and fairness. Modifying a surface while preserving its global convexity requires the user to manipulate several or even all control points simultaneously. This task has proven to be very cumbersome. Andersson estimated that it would take 170 hours in modifying the control point locations on a 700 series Volvo car hood to achieve global geometric convexity [Andersson 88]. As opposed to the nodal basis method, it is difficult to enforce geometric constraints on surfaces which are not defined explicitly by their geometric properties. Imposing constraints on such geometric properties requires additional consideration of constraint equations.

Another disadvantage of the control point method is that when there are features lying in the diagonal to the parametric domain, many control points must be used to ensure that the feature can be described in enough detail. The extra control points are wasteful for other relatively flat regions.

2.3.2 Surfaces from Curves

2.3.2.1 Lofting Method

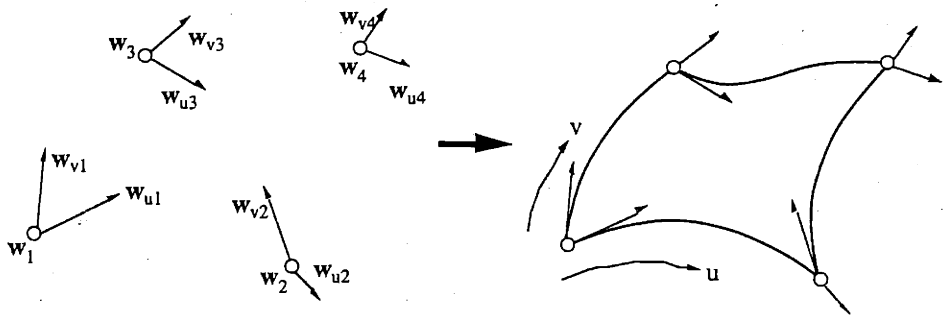
The lofting method is one of the oldest methods used to design surfaces in early CAGD applications. It derives its name from the large architectural space, the loft, where ship hull designers traditionally were located. However, recently the term skinning has been used more frequently. A simple lofting surface is to generate a smooth patch through two parametric curves that are to become embedded on two opposite sides of the parametric domain. The ruled surface which linearly blends these two boundary curves is probably the simplest form. Lofting can be generalized to interpolate a family of curves defined in a set of parallel planes [Tiller 83]. This method has been extended to allow non-planar curves with arbitrary orientation [Woodward 88] and can be viewed as a special case of generalized cylinders in which the surface is defined by a single curve which can change its shape as it is swept through space along a prescribed trajectory

[Coquillart 87]. An example of using this method to generate a NURBS surface for approximating a generalized cylinder is presented in [Bardis 90].

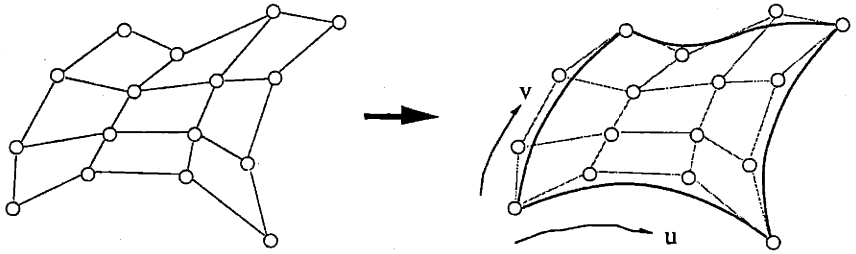
2.3.2.2 *Transfinite Method*

The transfinite schemes define surfaces by blending a set of space curves which become the boundary of the surface being blended. This scheme was originated by Coons's early work in which four boundary curves are bilinearly blended into a surface. [Coons 67] It can be generalized to achieve higher continuity between patches by using tangent or even higher order derivatives on the boundary curves and higher degree blending functions. One major problem arising in this scheme is the decision of the twist vectors. Twist vectors are not a geometric property of the surface but an artifact of the parametrization. In [Faux 79], a simple example shows that even a flat plane can possess nonzero twist. To cubically blend four curves into a rectangular patch, twist vectors need to be set at the four corners of the patch. Various methods have been developed to set the values for the twist vectors. Historically, the first method is due to Ferguson who set the twist vectors to be zero. Barnhill, et al. used the twist vectors of a bilinearly blended Coons patch as the desired twist vectors. Nowacki, et al. obtained the twist by minimizing an energy functional composed of the quadratic sum of the principal curvatures of the surface. This technique was applied to a generalized Coons patch by Hagen and Schulze [Hagen 87]. Kallay used a similar approach but replace the functional with the quadratic sum of the surface's second derivatives [Kallay 90]. A survey of methods for selecting twist vectors was given in [Barnhill 88], which review various methods and interrogate the resulting surfaces by their curvatures.

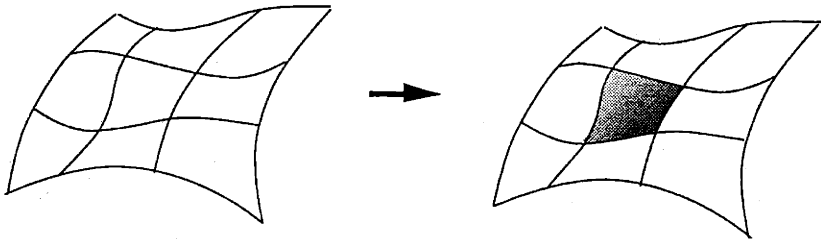
One of the advantages of defining surfaces from curves via the lofting or transfinite methods is that curves with different degrees and forms could be smoothly blended into a surface. However, control of such surfaces is generally laborious since it is done by controlling the boundary curves from which the surface is blended. For example, local control of the interior region of a transfinite surface is achieved by representing the surface as a set of patches, joined together with certain continuities and bounded by a curve network found from the original surface. The interior of the original surface is then modified by changing the shape of interior curves.



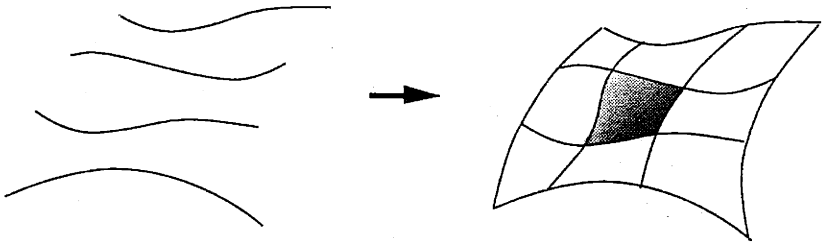
(a) surfaces from discrete points: the nodal basis method



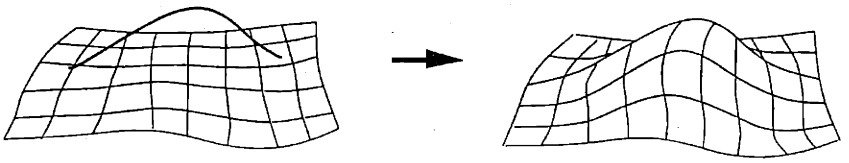
(b) surfaces from discrete points: the control point method



(c)surfaces from curves: transfinite method



(d) surfaces from curves: lofting



(e) variational method

Figure 2.3: Current free-form surface design paradigms

2.3.3 Variational Method

Energy minimization idea has been used extensively in the interpolation of ordered data points, which results in the so called minimal energy curves that were introduced in Chapter 2. This idea have also been applied to help to determine the twist vectors of the Coons-like surfaces [Hagen 87, Kallay 90]. Employing energy minimization concept in surface design is known as "variational surface design" now since it entails the surface's shape as the solution of a constrained variational optimization problem. In variational surface design, surface shape is modified by some typical operators including the sculpting forces and geometric constraints such as points and curves. Users specify points in space, and curves are generated by fitting groups of points. Points and curves could be freely added in this way and serve as the handles for manipulating the surface.

[Celniker 90a] applied the minimal energy idea associated with FE-based surface representation as a "shapewright" tool. Applying this idea to B-spine surface has been done by [Celniker 92], [Welch 92] and [Kallay 93]. Bloor and Wilson [Bloor 89] created blending surfaces by solving partial differential equation with boundary values by the finite difference method. This can be shown to be equivalent to the energy minimization principle.

2.3.4 Free-form Surface Modification

In addition to the methods described above, other techniques for the modification of free-form surfaces are worthy of note. Barr's [Barr 84] global deformation method uses an R^3 to R^3 transformation to change the shape of solid primitives. The basic deformations include tapering, twisting and bending. Sederberg introduced the FFD (free-form deformation) method which deforms the surface shape enclosed in a user-specified FFD lattice via an R^3 to R^3 mapping [Sederberg 86]. This work was extended by Coquillart who used a non-parallelepipedical 3D lattice [Coquillart 90]. This approach, though an improvement over manipulating control points, can still have difficulties, for example in creating a proper 3D lattice to enclose the region to be modified. Forsey presents a surface refinement scheme using the Oslo algorithm to obtain new control points of the refinement region [Forsey 88]. The new control points controlling the small portion of the surface are defined with respect to its parent surface's control points so that changing the parent surface will alter the child surfaces correspondingly. The final surface is a union of portions of various B-spline surfaces in different levels of overlay

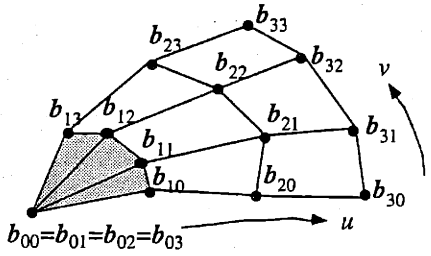
(i.e. the depth in the hierarchy). Surface shape is controlled by manipulating the control points of the overlays at different levels. However the refinement is confined to the isoparametric location of additional control points, and deformation of arbitrary shape cannot be achieved. Cavendish proposed a procedure to create a functional surface composed of a primary surface, a secondary surface and a smooth transition between them [Cavendish 92]. Though an elegant approach for creating complicated functional surfaces such as a contact lens cleaning kit, difficulties may arise when applying this technique to parametrically-described surfaces.

2.4 TRIANGULAR INTERPOLANTS

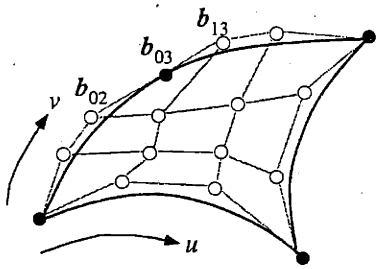
Triangular elements have always been important to the FEM community because the range of topological shapes that can be modeled is much larger than with rectangular elements. Any planar polygonal shape can be modeled with arbitrarily complex meshes where any number of edges can meet at a node in any pattern of angles. In this section, different triangular interpolants will be introduced in three categories: control points, nodal basis and transfinite.

2.4.1 Control Point Method

Triangular interpolants can be defined by a set of control points using methods similar to those used in quadrilateral patches. A triangular patch can be made by forcing all the control points on one row or column of a tensor product surface to be the same. Since this is a direct extension of the quadrilateral patch, no more new code needs to be added in the implementation. At such a degeneracy point, the normal vector is not defined unless the neighboring control points are coplanar as shown in figure 2.4a. This means when stitching several degenerate patches together, all the control points connected to each degenerate point must lie on the same plane. The capability for controlling the shape of a composite surface is thus greatly reduced due to the constraints around each degenerate point.

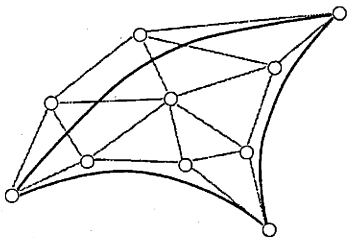


(a) A triangular patch formed by "collapsing" the control points. All the control points on the shaded region have to be coplanar to ensure a unique normal at the degenerate point.

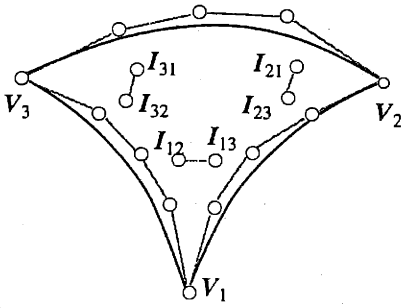


(b) A triangular patch formed by allowing the derivatives at a corner to be linearly dependent (control points b_{02}, b_{03} and b_{13} are collinear)

Figure 2.4: Two types of degenerate triangular patch



o Bézier ordinates
(a) A cubic triangular Bézier patch



(b) A quartic triangular Gregory patch

Figure 2.5: Two control-point type triangular interpolants

In [Faux 79], a formula is offered to calculate the normal vector of such a degenerate edge. A more general formulation for evaluating the unit normal and its derivatives on a degenerate edge is given in [Berger 92]. Another type of degenerate triangular patch can be formed by allowing the tangent vectors at one corner of a quadrilateral patch to be linearly dependent as in figure 2.4b. The condition to ensure the existence of normal vector at such a corner is discussed in [Farin 93].

Similar to the Bézier surface, the Bernstein basis can be used to define triangular Bézier patches as

$$w(L_1,L_2,L_3)=\sum_{i+j+k=n} b_{ijk} B^n_{ijk}(L_1,L_2,L_3) \quad ; \quad B^n_{ijk}(L_1,L_2,L_3)=\frac{n!}{i!j!k!}L_1^i L_2^j L_3^k \qquad (2.4.1)$$

where, L_1, L_2 and L_3 are the barycentric coordinates satisfying $L_1+L_2+L_3=1$ and B_{ijk}^n are the Bernstein polynomials of degree n. The coefficients b_{ijk} are the control points called Bézier ordinates (or Bézier points). They form a triangular net as illustrated in figure 2.5a. Note that there are a total of $(n+1)(n+2)/2$ terms in the summation of $w(L_1, L_2, L_3)$ and any straight line on the triangle is a curve of degree n on the triangular patch. The de Casteljau algorithm and degree elevation techniques can be applied to triangular Bézier patches. The continuity conditions that have to be satisfied when stitching two triangular Bézier patches or one triangular Bézier patch to another quadrilateral patch is discussed in [Farin 82], [Jensen 87], [Piper 87], [Shirman 87] and [Farin 93].

The triangular Gregory patch is a variant of the triangular Bézier patch. For a quartic triangular Gregory patch, six control points are used to replace the three interior control points of a quartic triangular Bézier patch. The surface is defined similarly as to a triangular Bézier patch except that the interior control points are blended from the new control points by

$$I_i = \frac{L_k(1-L_j)I_{ik} + L_j(1-L_k)I_{ij}}{L_k(1-L_j) + L_j(1-L_k)} \tag{2.4.2}$$

where, (i, j, k) are the cyclic indices, i.e., (i, j, k)=(1, 2, 3), (2, 3, 1) or (3, 1, 2). A quartic triangular Bézier patch will appear if all three pairs of control points overlap.

2.4.2 Nodal Basis Method

Triangular interpolants of this type interpolate a set of given space vectors, like the position vectors and tangent vectors at a finite number of nodes on the triangular surface. Most of the nodal-basis triangular interpolants used in CAD were drawn from the finite element community. Zienkiewicz published a 9 degrees of freedom triangular element interpolating the position and tangents at the triangle's vertices to solve the plate bending problem. This element is nonconforming, i.e., it does not ensure the continuity of cross boundary derivative. The nonconformity can be overcome by introducing extra degrees of freedom at the mid-edge nodes, resulting in a 12-dof element. This 12-dof element is composed of the original 9-dof interpolant and a rational correction function with singularities at the vertices. It has cubic variation in the shape of the edges and parabolic variation in the normal derivatives along the edges. However, the 12-dof triangular element presents some assembly difficulties as different numbers of degrees of freedom

are associated with the nodes. To avoid this difficulty the degrees of freedom at the mid-side nodes can now be constrained or condensed. For example, the normal derivatives at the mid-side nodes can be assumed to be the average of the normal derivatives at the end nodes of that edge. This reduces the degrees of freedom of the element from 12 back to 9 but still maintains the C^1 continuity between adjoining elements. The normal derivatives on the edges will vary linearly instead of parabolically. Unfortunately, this kind of element tends to be too flat in the center of the element, and when highly curved, will calculate surface normals in the wrong directions. In section 4.2, the 9-dof and 12-dof triangular element will be introduced in more detail.

Other nodal-bass type triangular interpolants are also available. Irons and Bell both produced 18 degree of freedom C^1 interpolants [Irons 69] [Bell 69]. Irons used the positions and tangents at the corner vertices but considered the position, the normal derivative and the mixed derivative at the three mid-edge points for interpolating the triangle. Bell used the position, the tangents, and the three 2nd parametric derivatives at each of the triangle vertices and the normal derivatives at the midedge nodes as degrees of freedom. The three degrees of freedom at the mid-edge nodes are condensed later by enforcing cubic variation on the normal derivative. Herron pointed out a major limitation of discrete interpolants which involved cross boundary derivatives to form a C^1 surface [Herron 85]. Requiring C^1 continuity between patches makes sense only if the patch domains are adjacent in the domain space. Herron proposed a G^1 triangle interpolant with 15 degrees of freedom to be used to make well behaved closed surfaces. The fifteen degrees of freedom include the position vector and two tangential derivatives in edge directions at each vertex node and two cross boundary derivatives per edge, taken at 1/3 and 2/3 points in the perpendicular direction. The values of the cross boundary derivatives on each common boundary are generated from the two edge-direction derivatives at the two vertices bounding the corresponding edge. So the actual input data is only 9 space vectors.

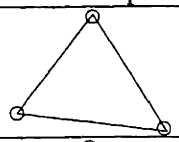
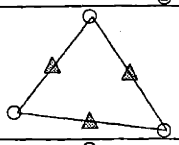
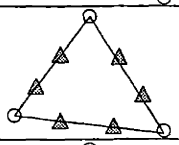
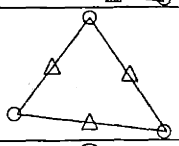
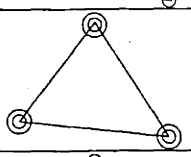
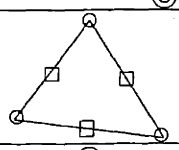
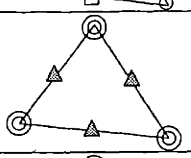
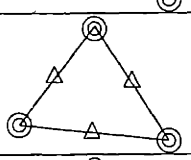
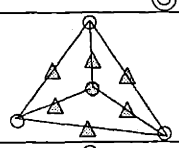
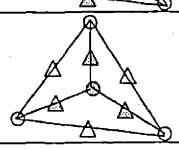
Zenisek was the first to give a general theorems about C^m triangular elements [Zenisek 70]. He shows that if a piecewise polynomial function defined over an arbitrarily triangulated domain is to be C^m, all derivatives up to the $(2m)$th order at the vertices of the triangles must be used. The resulting function is of degree $4m+1$. The simplest piecewise polynomial function defined over triangles which will be C^2 is of degree 9. It is determined by position and derivatives up to and including all fourth derivatives at the triangle vertices, first normal cross-boundary derivatives at the

midpoints of edges, second normal cross-boundary derivatives at points dividing the triangle's edges into three equal parts and positions at the centroid of the triangle. A Bernstein-Bézier form for representing this 55-dof interpolant is done by Whelan [Whelan 86]

A Clough-Tocher element is an alternative way to generate conforming triangular element without using rational correction functions. This element interpolates the position and tangents at vertex nodes and normal derivatives at midedge nodes. It is generated by first partitioning the triangle, called the "macro-triangle", into three subtriangles, called the "micro-triangles" at an internal point, often the centroid. Each of the micro-triangles is represented as a complete cubic expansion involving 10 terms. Thus there are 30 unknown coefficients to be calculated. For each micro-triangle, the nodal values must be satisfied, thus gives 7x3 equations. In addition, continuity of position and tangents at the internal point provides additional six equations and continuity of normal derivatives of internal mid-edges a further three. Thus, we have 30 equations to solve for the 30 unknowns. The Clough-Tocher element has C^2 discontinuity along the internal edges. Constraints of normal derivatives on exterior edges leads to an element with 9 degrees of freedom.

A summary of these nodal basis triangular interpolants is listed in table 2.1. The shape of the control-point type interpolant is decided purely by its control points. A composite surface composed of many single patches of this sort is a collection of many mappings from a 2D standard triangle to 3D space. Cross-boundary continuity is achieved by carefully constraining the control points of adjacent patches. Since there is no correlation between the parametric domain of each triangular patch, a surface composed of patches of this type is more capable of representing complicated shapes. However, this also makes it hard to maintain the topological information such as adjacency information between patches. In contrast, shapes of interpolants of the nodal-basis type are decided by both the given vectors representing geometric properties and the range and shape of the triangle. A composite surface composed of many single patches can be considered as a single mapping from a 2D triangulated domain to 3D space. Tangent plane continuity between patches is automatically satisfied as long as the properties of the boundary only depend on the geometric information on the boundary. The adjacency information for patches can be tracked by the adjacency of the corresponding parametric domain.

Table 2.1: Some nodal-basis triangular interpolants

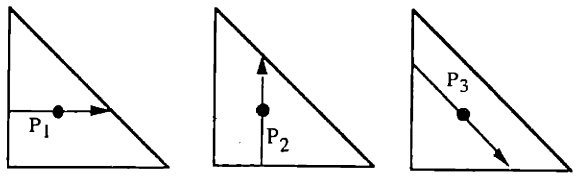
# of dof	dof map	w along edges	w_n along edges	Remarks
9		cubic	parabolic	non-conforming
9		cubic	linear	conforming (condensed 12-dof)
9		cubic	parabolic	conforming[Herron 85] (condensed 15-dof)
12		cubic	parabolic	conforming
18		quintic	quartic	nonconforming
18		quartic	cubic	conforming [Irons 69]
18		quintic	cubic	conforming [Bell 69] (condensed 21-dof)
21		quintic	quartic	conforming
9		cubic	linear	conforming (Clough-Tocher element)
12		cubic	parabolic	conforming (Clough-Tocher element)

○: (w, w_u, w_v) △: (w_n) □: (w, w_n, w_{tn}) ◎: $(w, w_u, w_v, w_{uu}, w_{vv}, w_{uv})$

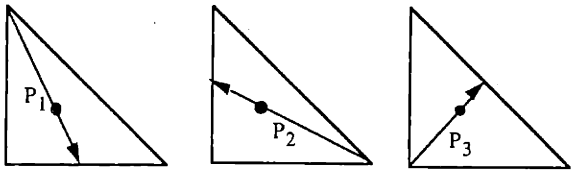
Shaded triangles or circles represent degrees of freedom which will be condensed.

2.4.3 Transfinite Method

The transfinite scheme, which probably originated from Coons patch, has been extended to the triangle. This type of interpolant matches data on the entire boundary of a triangle. Gordon called them transfinite interpolants in order to distinguish them from the discrete interpolants which interpolate a finite number of discrete points on the triangle. Generally, the data supplied consists of the position values, normal derivatives cross the boundary or tangent planes on the boundary of the triangle. Barnhill, Birkhoff and Gordon initiated the development of transfinite triangular interpolants in 1973 [Barnhill 73]. These interpolants, known as BBG interpolants, are built up from univariate Hermite operators applied along lines parallel to the edges of the triangle. As in the rectangular Coons patch, this interpolant may sometimes fail due to the incompatibility of the mixed derivatives at the vertices and considerable efforts have been made to correct this flaw. Nielson's radial projectors interpolate data on each vertex and the vertex's opposite edge and is thus named the "side-vertex method" [Nielson 79]. Both of these two schemes generate rational interpolants. The BBG interpolants is rational in its blending functions and the side-vertex scheme is rational in blending function's arguments.



(a) The BBG operators



(b) The radial projectors

Figure 2.6: The BBG and radial lofting operators

Gregory and Charrot [Gregory 80] made a C^1 triangulation that is compatible with a rectangular Coons grid. This work was then extended to support n-sided patches with C^2 continuity. The vertices in this method's meshes are restricted to existing at the simple

intersections of the bounding curve mesh. Each vertex is expected to have only four properly aligned edges. Gregory [Gregory 85] formalized the transfinite scheme for triangles by showing how blends could be made over a simplex of any order. Nielson used the side-vertex method to interpolate the position and outward surface's normals on the triangle's boundary [Nielson 87]. The resulting interpolant is G^1 instead of C^1. Hagen presented a curvature continuous transfinite triangular interpolant which interpolates the position, tangent planes and curvature elements on the boundary of the triangle [Hagen 88].

2.5 LIMITATIONS

In the first three sections, a vast amount of literature in curves reconstruction, surface reconstruction and surface design methodology has been reviewed. The following are the limitations of the existing methods.

Curve reconstruction

All the parametrization methods and minimal energy methods require the data points to be ordered. When the number of data points is small, the connectivity information can be identified with the aid of a graphical display such as a three dimensional plot. However, this task becomes more and more time-consuming as the number of points increases. Furthermore, if errors are embedded in the data points during the data inquiry stage, the connectivity of the data points is difficult to be determined. A method which is able to deal with unordered, error-filled data is important under these considerations.

Surfaces Reconstruction

The success of a surface reconstruction method depends heavily on the surface representation which in turn depends on the application's specific requirements. Methods which are perfect for one application may not be appropriate for other applications. There is still no method which is able to take the complex data set of a human skull and find mathematical equations that represent it, allowing its geometric properties to be analyzed, rendered graphically and modified. Before the development of a new method, the surface representation must be carefully selected.

This thesis will focus on industrial applications in which a mathematical description for the surface is required. For example, geometric properties such as normal vectors and principal curvatures must be generated for the manufacturing purposes. The representation must also enable the surface to be rendered and modified easily to facilitate the design process. Table 2.2 is a list of the suitabilities of the four different surface representations with respect to some common requirements found in most of the applications. An examination of these considerations led to the choice of parametric representation for our research.

Table 2.2: The suitabilities of different surface representations with respect to common industrial requirements

	explicit	parametric	implicit	simplicial
ability to represent complex shape	x	Δ	Δ	√
full mathematical description	√	√	√	x
easy to be modified	Δ	√	x	x
easy to be rendered	√	√	x	√
easy for object recognition	x	x	√	Δ

x: Poor; Δ: Moderate; √: Good

Existing parametric surface reconstruction methods require a lot of connectivity information about the data points and are not able to deal with data points with embedded errors. The methods developed are also only suitable for reconstructing quadrilateral surfaces. A general n-sided surface is achieved by trimming the reconstructed quadrilateral surface, which generally causes the boundary curves of the trimmed surface to be high degree curves. A conversion process to lower degree curves frequently is necessary in subsequent problems like assembling trimmed surfaces and blending surface generation.

Surface Design

The basic goal of a free-form surface design methodology is to enable users to create a surface and control its shapes effortlessly. Traditionally, the pursuit of this goal has taken the form of a search for an "optimal" surface definition and the shape is controlled

through the direct manipulations of the degrees of freedom by the users. The strengths and weaknesses of the different surface design schemes had been discussed in the previous section. When the controls provided to the users are closely tied to the degrees of freedom defining the surface, controlling the surface shape will become more and more inefficient and hard to be achieved as the degrees of freedom of the surface increases. This problem has been attacked by a scheme which utilizes the variational principle as a new methodology for designing surfaces. This scheme, now known as the variational surface design method, casts the design problem as an optimization problem by which the degrees of freedom are determined by minimizing an functional subject to some design constraints. This approach enables the creation of surfaces with minimal input and satisfy the fairness requirement automatically.

The goal of this thesis is to develop new methodology for curve and surface reconstruction which is able to overcome the limitations mentioned above and can be seamlessly integrated with variational surface design.

CHAPTER THREE

Curve Reconstruction

Reconstructing a smooth curve from 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 $\boldsymbol{P}_i$, i=0,..., N-1, taken from a target curve, reconstruct a curve which approximates the original one to a satisfactory extent and also possesses visually good appearance". Problems of this kind can be broadly categorized into interpolation problems and approximation (or smoothing) problems according to whether the resulting curve is required to pass through all the data points exactly or not. 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 unwanted undulations. Many papers have addressed this problem based on the assumption that the data points are ordered. However, the task of identifying the connectivity becomes more and more time-consuming and difficult as the number of points increases and measurement errors occur in the data inquiry stage. A curve reconstruction method which is able to deal with unordered, error-filled data using only partial connectivity information is important under these considerations.

In this chapter, 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 multiple-valued data. The problem is formulated through the variational principle which seeks the optimal

curve as the solution of a minimization problem which is then solved by successive quadratic programming. The objective function to be minimized is specially designed for parametric curves, since they are well-suited for industrial applications. It is composed of a global and a local irregularity measurement for the solution and a closeness measurement between the solution and the data points. Other than the end points of the curve, none of the data point connectivity information is required, thus eliminating the necessity of an initial parametrization. Both cubic and quintic Hermite polynomials are implemented as the curve basis. Geometric constraints can be incorporated into the minimization process if necessary.

The rest of this chapter is organized as follows. Section 3.1 introduces the energy functional to be minimized, its physical analogy and interpretation, the finite element solution and the minimization method. Section 3.2 introduces the cubic and quintic Hermite polynomials used as basis functions of the reconstructed curve. Section 3.3 shows the effect of weighting values on curve's shape. Section 3.4 describes some methods used to evaluate the quality of the fitted curve. Section 3.5 summarizes this chapter.

3.1 VARIATIONAL FORMULATION

3.1.1 Objective Function

A problem is called 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 plausibility of these solutions. In this regard, constraints such as the smoothness of the solution were introduced to act as "stabilizers" in the regularization theory pioneered by Tikhonov and others [Tikhonov 77]. Through regularization, the original ill-posed problem is reformulated through the variational principle which seeks the solution as the one minimizing

$$E_{curve}(f) = \sum_{p=1}^{m} \int_C \alpha_p(x) \left(\frac{d^p f(x)}{dx^p} \right)^2 dx + \sum_{i=1}^{N} \lambda_i (y_i - f(x_i))^2 \qquad (3.1.1)$$

where, $\alpha_p(x)$ are prespecified, nonnegative weighting functions, and λ_i are positive weighting factors. The first term in equation (3.1.1) is a global irregularity measurement of the solution. The higher the value is, the less likely that this solution will be accepted. It is also referred to as the stabilizing term in regularization theory since it acts as a stabilizer of the original ill-posed problem. The second term, referred to as the error term, is a closeness measurement between the solution and the given data points.

Since the thesis's interest is in CAD applications, the above-posed functional is not appropriate for the following reasons and should be modified.

(1) Inappropriate solution representation.

The explicit representation of the solution in the above equation represents a planar curve only. Since a curve of dimension higher than two cannot be represented by one simple explicit equation, the parametric form is a better choice in this regard. The

parametric curve provides consistent representation from 2D to 3D and even higher dimensions, and it also has many advantageous properties over explicit curves.

(2) Inappropriate closeness measurement

Since a parametric representation is essential for CAD applications, a parametrization of the data points, which is not known *a priori*, is required to calculate the error term. Even if the parametrization is obtained by some means (e.g., assuming connectivity information of data points), using $(y_i - f(x_i))^2$ as a measurement of error is somewhat inappropriate since it could cause nonzero measurements even when all the points lie exactly on the curve.

(3) Insufficient measurement of irregularity

In the CAD community it is very common to use low degree polynomials as the basis function when representing a curve. This avoids the unnecessary oscillations that appear in high degree polynomials. The resulting curve is a piecewise continuous one with discontinuities in high derivatives at the junctions of curve elements. These derivative discontinuities are not desirable properties and can be viewed as a local irregularity measurement of the curve, which can also be included in the functional.

From these reasons, the functional was modified into

$$E_{curve}(w) = \sum_{p=1}^{m} \int_C \alpha_p(u) \left\| w^{(p)}(u) \right\|^2 du + \sum_{p=0}^{m} \beta_p \sum_i \left\| \Delta w^{(p)}(u_i) \right\|^2 + \sum_{i=1}^{N} \lambda_i D_c(P_i, w(u))^2 \qquad (3.1.2)$$

where, $w(u)$ is a parametric curve. $w^{(p)}(u)$ is its pth derivative and $\Delta w^{(p)}(u_i)$ represents the discontinuity of $w^{(p)}(u)$ at u_i. $\alpha_p(u)$, β_p and λ_i are prespecified, nonnegative weighting functions or factors. $D_c(\boldsymbol{P}_i, w(u))$ is a distance metric which is defined as the shortest Euclidean distance from $\boldsymbol{P}_i$ to $w(u)$. The evaluation of $D_c(\boldsymbol{P}_i, w(u))$ results in a constrained single variable minimization problem which will be discussed later.

The choice of m depends on the application. For most applications in geometric modeling 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 *k*th order derivatives, only solutions with continuous derivatives up to the (*k*+1)th order need to be considered. Experience shows that discontinuities of C^0 and C^1 can be easily detected by a human's visual system, and C^2 discontinuity can be recognized by experienced users. Hence, derivatives will be taken up to the third order in this thesis. However, 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.

3.1.2 Physical Interpretation

Consider a system composed of a thin elastic beam constrained by some boundary conditions, with springs that have one end fixed at some position in space and the other end attached to the beam. These springs are allowed to slide freely along the beam so that they always maintain the shortest approach to the beam. The potential energy contained in this system can be expressed as a sum of the strain energy stored in the beam itself and that in the springs. The mechanical behavior of this system can be determined by applying the variational principle, and the equilibrium state of the beam is the one which makes the variational indicator vanish. For a stable equilibrium state, the system's potential energy is required to be at its minimum.

The curve reconstruction problem is simulated as the physical system depicted above. The parametric space curve is analogous to the elastic beam and the data points to the fixed positions at which the springs are fixed. Each spring is assumed to have one end anchored at a data point and the other end slides freely along the curve so that the shortest distance is maintained. This is illustrated in figure 3.1a. The squared first derivative term in the functional stands for the strain energy due to stretching of the beam and the squared second derivative term 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 functional to be minimized and the strain energy in material mechanics, sometimes we also refer it as an "energy functional". Minimizing the integral of the squared magnitude of the first and second derivatives makes the curve stretch and bend as little as possible.

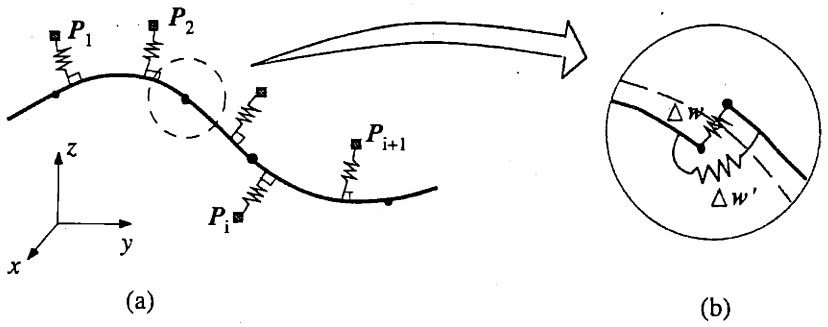


Figure 3.1: The physical model for the curve reconstruction. (a) Curves deformed by springs attached by the shortest approach. (b) The built-in springs which absorb the energy due to the discontinuity at the junction

Although the squared third derivative term in the energy functional does not correspond to any physical meaning, it does have a geometric meaning. The magnitude of the second derivative $\|w''\|$ is actually an approximation of the curvature κ when $\|w'\|$ is assumed to be small. Similarly, the third derivative term is treated as a rough estimate of the rate of change of curvature, $d\kappa/ds$. Since the integral of the squared magnitude of the derivative of curvature evaluates to zero for circular arcs and straight lines, introducing the third derivative term enables the curve to form a circular arc (approximately) when constraints allow. The local irregularity measurement can be viewed as the energy stored in small springs "built- in" at the curve element's junctions that serve as connections between two elements. Whenever discontinuities occur at the junctions, energy is stored in these built-in springs. This is illustrated in figure 3.1b.

From the viewpoint of optimization theory, the functional can also be viewed as an augmented objective function using exterior penalty functions method in an optimization problem stated as

$$\begin{aligned} &\text{Minimize} \quad \overline{E}_{curve}(w) = \sum_{p=1}^{m} \int_C \alpha_p(u) \|w^{(p)}(u)\|^2 du + \sum_{p=0}^{m} \beta_p \sum_i \|\Delta w^{(p)}(u_i)\|^2 \\ &\text{Subject to} \quad D_c(P_i, w(u))^2 \le \varepsilon_i \quad , i = 0, \cdots, N-1 \end{aligned} \tag{3.1.3}$$

The stabilizing term in equation (3.1.2) is a general form of the energy functionals that are used in many minimal-energy splines. The different energy functionals used for the splines under tension, (weighted) ν-splines, and (weighted) τ-splines emerge with properly chosen the weighting functions. Compared to the energy functionals used in

[Reinsch 67] and [Hosaka 78], the global irregularity measurement includes not only the integral of the squared second derivative but also that of the first and the third derivatives. A local irregularity measurement is also considered. Through the choice of different weighting values, the shape obtained from the minimization is more capable of fitting the given data points and possesses better quality in both the global and local senses. Another significant difference resides in the formulation of the error term measuring the closeness of the approximating curve to the data. In the above-mentioned references, the error term is taken as the sum of squared vector differences of data points and "nodes" defining the approximation curve. This kind of formulation still requires the data points to be ordered. Defining the closeness measurement as a sum of the shortest Euclidean distances between the data points and the curve eliminates the requirement for the data point connectivity information at the cost of increased computing time.

3.1.3 Nonlinear Minimization

In computer modeling, continuous models are approximated by superposing a finite series of continuous functions with weighting factors. This approximation for $w(u)$ is

$$w(u) \approx \sum_{i}^{n} \phi_i(u) q_i = \Phi^{\mathrm{T}} \mathbf{Q} \tag{3.1.4}$$

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

Substituting equation (3.1.4) into (3.1.2) and taking $m = 3$ results in

$$E_{curve}(\mathbf{Q}) = \mathbf{Q}^{\mathrm{T}} \mathbf{K} \mathbf{Q} + \sum_{i=1}^{N} \lambda_i D_i^2(P_i, \mathbf{Q}) \tag{3.1.5}$$

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

$$\mathbf{K}_1 = \int_c \alpha_1(u) \Phi' \Phi'^{\mathrm{T}} du, \ \mathbf{K}_2 = \int_c \alpha_2(u) \Phi'' \Phi''^{\mathrm{T}} du, \ \mathbf{K}_3 = \int_c \alpha_3(u) \Phi''' \Phi'''^{\mathrm{T}} du$$

and $\mathbf{K}_d$ is the matrix resulting from the discontinuity terms.

It can be shown that if the stabilizing term consists of only one squared derivative of order m, at least m constraints must be applied in order to find a stable solution. This can be interpreted more easily by the analogy of material mechanics mentioned in section 3.2. If only stretch energy is included, the system is like a beam under axial deformation, which requires at least one position constraint. If only bending energy is included, 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 the stabilizing term.

Inspired by the fact that the first term is quadratic, successive quadratic programming (SQP) is used to seek the solution. This method replaces the original objective function by its local quadratic approximation at the solution estimate and solve the resulting approximate subproblem. This replacement is repeated and the solution estimate is improved during iterations until a certain convergence criterion is achieved. 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. More details about the SQP technique will be discussed in Chapter 6.

3.2 THE CURVE PRIMITIVE

3.2.1 Selection of the Curve Primitive

The basis functions of a 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 **K** in equation (3.1.5) thus speeding up calculation and easing the imposition of complicated geometric constraints. In this work, Hermite polynomials are adopted as the curve basis. Hermite polynomials are not very popular for modeling curves in CAGD because loops and cusps are easy to appear due to an inappropriate setting of the tangent vectors' magnitudes. However, using the energy minimization concept to obtain the tangent vectors makes the Hermite polynomial well behaved. Hermite polynomials are selected as the curve basis because they are explicitly represented by the degrees of freedom that will be constrained. This makes it easier to impose geometric constraints on the curve than if the curves were defined by control points. Using the reduced transformation method to impose slope constraints is as easy as imposing position constraints. In fact, the Bernstein basis and B-spline basis could also be used as the curve primitive but constraining the derivatives would be more laborious.

3.2.2 Hermite Polynomials

Hermite polynomials are a set of polynomial functions defined in a fixed interval $0 \le u \le h$. A general form of Hermite polynomials is written as

$$H_i(m,h,u) = \frac{1}{i!}\sum_{j=0}^{m-i}(-1)^{i+j-1}\mathrm{C}_j^{2m+1-i}h^i\left(\frac{u}{h}\right)^{i+j}\left(\frac{u}{h}-1\right)^{2m+1-i-j}$$
$$G_i(m,h,u) = \frac{1}{i!}\sum_{j=0}^{m-i}(-1)^{j}\mathrm{C}_j^{2m+1-i}h^i\left(\frac{u}{h}\right)^{2m+1-i-j}\left(\frac{u}{h}-1\right)^{i+j} \qquad (3.2.1)$$

where the integer m decides the number of polynomials and their degrees. A set of Hermite polynomials of degree $2m+1$ includes $2(m+1)$ polynomial functions. In practical cases, the value of m is taken less than three. When $m=0$, it is the linear Hermite polynomial; $m=1$, the cubic Hermite polynomial; and $m=2$, the quintic polynomial. They have the following properties:

$$\begin{aligned}
&\frac{\partial^{(j)}H_i(m,h,0)}{\partial u^{(j)}} = \delta_{ij} \quad ; \quad \frac{\partial^{(j)}H_i(m,h,h)}{\partial u^{(j)}} = 0 \quad , i,j \le m \\
&\frac{\partial^{(j)}G_i(m,h,0)}{\partial u^{(j)}} = 0 \quad ; \quad \frac{\partial^{(j)}G_i(m,h,h)}{\partial u^{(j)}} = \delta_{ij} \quad , i,j \le m \\
&H_i(m,h,u) = (-1)^i G_i(m,h,h-u) \\
&H_0(m,h,u) + G_0(m,h,u) = 1 \\
&H_1(m,h,u) + G_1(m,h,u) + hG_0(m,h,u) = u
\end{aligned} \tag{3.2.2}$$

where $\delta_{ij} = 1$ for $i = j$ and $\delta_{ij} = 0$ otherwise.

A polynomial curve of degree $2m+1$ can be created by using Hermite polynomials of the same degree to interpolate the geometric properties at the curve's two ends as

$$w(u) = \sum_{i=0}^{m} \left\{ w_0^{(i)} H_i(m,h,u) + w_h^{(i)} G_i(m,h,u) \right\} \tag{3.2.3}$$

where $w_0^{(i)}$ and $w_h^{(i)}$ are the values of the ith derivatives of $w(u)$ at $u=0$ and $u=h$. A linear Hermite polynomial curve depends on the nodal positions only. Higher order interpolations can be achieved by supplying higher order derivatives of the curve at the nodes. Since a composite curve of multiple linear Hermite curves does not enable C^1 continuity at the junctions, linear Hermite curves are rarely used in CAD applications. Cubic and quintic Hermite polynomial curves are more popular, especially in the field of curve fitting. They were used as the curve primitives in this thesis.

The shape of a cubic Hermite polynomial curve is defined by its position and tangent vectors at the two nodes. For a quintic Hermite polynomial curve, additional information of the second derivative is required. Generally the shape of a cubic or quintic Hermite curve is described as

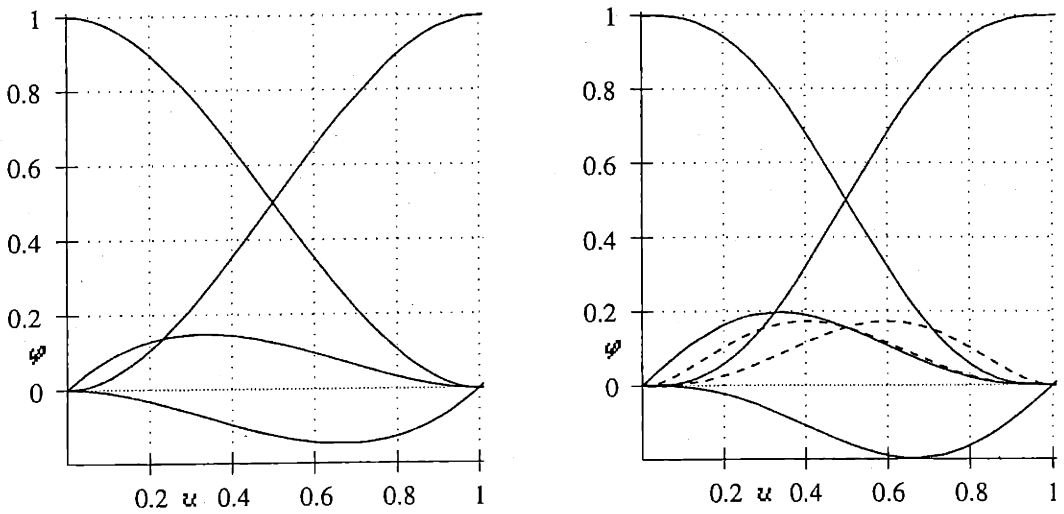
$$w_e(u) = \sum_{i=1}^{n} q_i \varphi_i \tag{3.2.4}$$

where, n=4 for cubic curves and 6 for quintic curves, $q_1 = w(0)$, $q_2 = w'(0)$, $q_3 = w(h)$ and $q_4 = w'(h)$ for cubic curves, and $q_1 = w(0)$, $q_2 = w'(0)$, $q_3 = w''(0)$, $q_4 = w(h)$, $q_5 = w'(h)$ and $q_6 = w''(h)$ for quintic curves. Shape functions, φ_i, for cubic and quintic Hermite curves are listed in table 3.1 and shown in figure 3.2.

Table 3.1: Shape functions for cubic and quintic Hermite curves

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

h is the element's parametric length.



(a) Cubic Hermite polynomials (b) Quintic Hermite polynomials

Figure 3.2: The cubic and quintic Hermite polynomials (the φ_3^q and φ_6^q functions, drawn with a dashed line, of quintic Hermite polynomials are magnified by a factor of ten)

A cubic Hermite curve is C^1 continuous along the entire curve but generally has C^2 discontinuities at the junctions. This discontinuity can be eliminated by imposing a compatibility equation on the degrees of freedom of adjacent elements. Similarly, a quintic Hermite curve is C^2 continuous along the entire curve but has C^3 discontinuities at the junctions. As for to its cubic cousin, the discontinuity can be eliminated by imposing a proper relationship on the degrees of freedom of adjacent elements.

3.2.3 Hermite FE Curve Stiffness Matrix

A finite element curve is a curve whose parametric space is divided by a series of nodes in a fixed interval $[a, b]$. Parametric values are assigned to each node monotonically so that $a = u_0 < u_1 < \cdots < u_{n-1} = b$. The segment between two nodes are often called an element. A curve element is the mapping of its parametric domain $[u_i, u_{i+1}]$ to the object space. The shape of a curve element is determined by the information at the two nodes and the interpolation functions. A curve element often can be reparametrized into $[0, h]$, $h = u_{i+1} - u_i$ without changing its shape. A FE curve is a composition of these curve elements each of which is C^∞ continuous. However, the entire curve's continuity is limited to the continuities at the nodes and depends on the degree of the basis functions used.

Hermite polynomial is a good basis for defining finite-element based curve since a FE curve can be constructed by piecing together Hermite polynomials with arbitrary continuity at the junctions provided the degree of the Hermite polynomials used is sufficiently high. Using cubic Hermite polynomials generally results in a C^1 curve with C^2 discontinuities and use of quintic Hermite polynomials produces a C^2 curve with C^3 discontinuities at the nodes unless further constraints are imposed. By using the piecewise Hermite curve, calculation of the matrices $\mathbf{K}_1$, $\mathbf{K}_2$ and $\mathbf{K}_3$ can be simplified into evaluating 4×4 (or 6×6 for quintic Hermite curve) submatrices of integrals for each curve element and assembling them together properly. These 4×4 (or 6×6) submatrices are called element stiffness matrices. When the weighting functions are constant over each element, the evaluation of these element matrices are simply exercise in calculus and can be done easily by software like MACSYMA which performs symbolic mathematical manipulations. They are given in table 3.2.

Table 3.2: Element matrices for cubic and quintic Hermite curves

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

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

Calculation of the matrix $\mathbf{K}_d$ can also be simplified into the evaluation of 6×6 (or 9×9 for quintic Hermite curve) submatrices. These submatrices are called "inter-element matrices" here since they are associated with the junction of each two adjacent elements rather than to any particular element. For a cubic Hermite curve, there is no first derivative discontinuity in the curve. The discontinuities in the 2nd derivative at the junctions can be calculated as

$$\begin{aligned} \Delta w_i'' &= w_i''(0) - w_{i-1}''(h_{i-1}) \\ &= \begin{bmatrix} -\frac{6}{h_i^2} & -\frac{4}{h_i} & \frac{6}{h_i^2} & -\frac{2}{h_i} \end{bmatrix} \begin{bmatrix} q_i \\ q_i' \\ q_{i+1} \\ q_{i+1}' \end{bmatrix} - \begin{bmatrix} \frac{6}{h_{i-1}^2} & \frac{2}{h_{i-1}} & -\frac{6}{h_{i-1}^2} & \frac{4}{h_{i-1}} \end{bmatrix} \begin{bmatrix} q_{i-1} \\ q_{i-1}' \\ q_i \\ q_i' \end{bmatrix} \\ &= \left(\overline{\mathbf{K}}_{d2}^e\right)^{\mathrm{T}} \mathbf{Q}_i^e \end{aligned} \tag{3.2.5}$$

with $\overline{\mathbf{K}}_{d2}^{e}=\left[-\frac{6}{h_{i-1}^2} \quad -\frac{2}{h_{i-1}} \quad (\frac{6}{h_{i-1}^2}-\frac{6}{h_i^2}) \quad (-\frac{4}{h_{i-1}}-\frac{4}{h_i}) \quad \frac{6}{h_i^2} \quad -\frac{2}{h_i}\right]^{\mathrm{T}}$ and $\mathbf{Q}_i^e=[q_{i-1} \quad q'_{i-1} \quad q_i \quad q'_i \quad q_{i+1} \quad q'_{i+1}]^{\mathrm{T}}$ is a column vector containing the position and tangent vectors at three consecutive nodes. h_{i-1} and h_i are the parametric lengths.

The discontinuity in the third derivative is

$$\Delta w_i''' = w_i'''(0) - w_{i-1}'''(h_{i-1}) = (\overline{\mathbf{K}}_{d3}^{e})^{\mathrm{T}}\mathbf{Q}_i^e \tag{3.2.6}$$

with

$$\overline{\mathbf{K}}_{d3}^{e}=\left[-\frac{12}{h_{i-1}^3} \quad -\frac{6}{h_{i-1}^2} \quad (\frac{12}{h_{i-1}^3}+\frac{12}{h_i^3}) \quad (\frac{6}{h_i^2}-\frac{6}{h_{i-1}^2}) \quad -\frac{12}{h_i^3} \quad \frac{6}{h_i^2}\right]^{\mathrm{T}}$$

The inter-element matrices for a cubic Hermite curve can be calculated by

$$\mathbf{K}_d^e = \beta_2\overline{\mathbf{K}}_{d2}^{e}(\overline{\mathbf{K}}_{d2}^{e})^{\mathrm{T}} + \beta_3\overline{\mathbf{K}}_{d3}^{e}(\overline{\mathbf{K}}_{d3}^{e})^{\mathrm{T}} \tag{3.2.7}$$

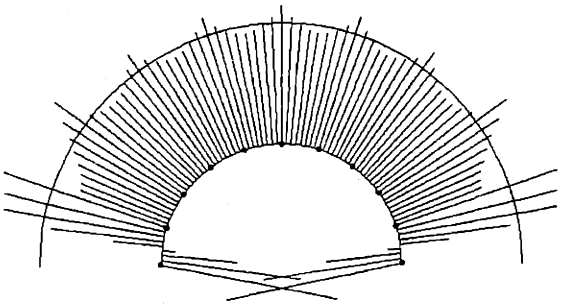
For a quintic Hermite curve, there are neither C^1 nor C^2 discontinuities. Its inter-element matrix is composed of only one term, the C^3 discontinuity, which can be calculated similarly. While the element matrices depend on the element's parametric length only, the inter-element matrices depend on both the parametric lengths of the adjacent elements. After the calculation of the element matrices and the inter-element matrices, they must be properly assembled according to the corresponding degrees of freedom to obtain the system matrix.

3.3 CHOOSE WEIGHTING VALUES

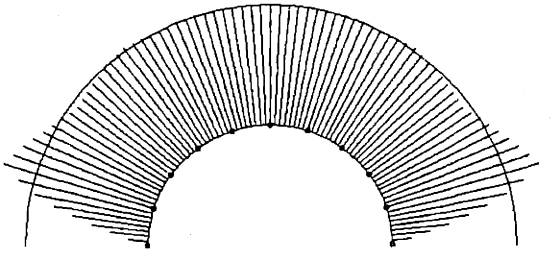
The choice of the weighting values is not a trivial problem. Setting different weighting values in the energy functional results in curves with totally different properties. A basic understanding about the effect of the weighting values is necessary so that proper values can be chosen to satisfy users' requirements. As mentioned previously, the integral of the squared first derivative represents the energy due to stretching and the integral of the squared second derivative represents the energy due to bending. While minimizing the weighted sum of these two terms causes the curve to stretch and bend as little as possible, minimizing the integral of the squared third derivative causes the curve to assume a circular shape when constraints allow. In the following, a simple example is used to show how different weighting values affect the curve's shape. This example shows a cubic curve interpolating eleven data points taken from a unit half circle. No errors are inserted in the data and knowledge of the sequencing of the data is assumed at this time. Different results based on different weighting values are shown in figures 3.3a to 3.3e.

We can see that minimizing the stretching energy only does not give a pleasant curvature distribution (figure 3.3a). When only the bending energy is minimized (figure 3.3b), the curvature distribution is much smoother and the discontinuities at the junctions are not recognizable. The curvature also vanishes at the two ends, which is far from what it should be. Actually, it can be proven that a cubic interpolating curve minimizing the integral of the squared second derivative is a C^2 curve with natural boundary conditions, i.e., zero curvature at the curve's ends. The curvature plots in figures 3.3c to 3.3e reveal that introducing the squared third derivative term makes the curve behave more like a circular arc. The large curvature discontinuity shown in figure 3.3c is due to the fact that the cubic Hermite curve has C^2 discontinuities at the junctions, and the presence of the α_3 term tends to minimize the curvature variation of each element, but adversely enlarges the discontinuities at the junctions. This enlarged discontinuity is reduced by introducing the β_2 term and totally eliminated by using a C^2 continuous curve as shown in figures 3.3d and 3.3e respectively.

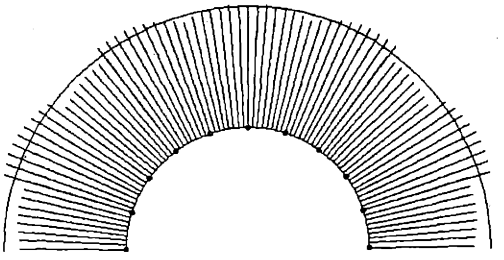
Some quantitative results about the accuracy of fit of these curves are listed in table 3.3.



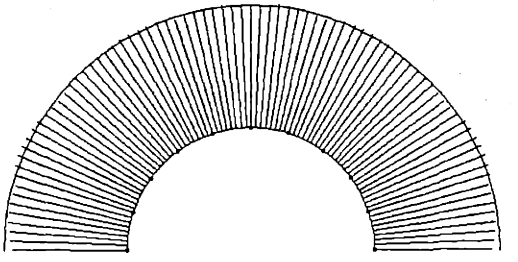
(a) C^1 cubic Hermite curve : α_1=1.0



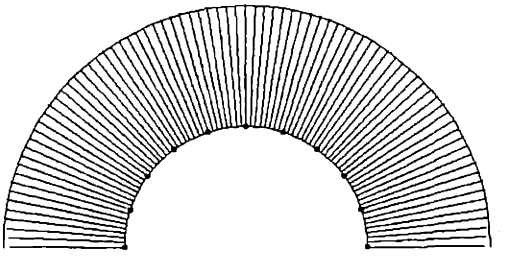
(b) C^1 cubic Hermite curve: α_2=1.0



(c) C^1 cubic Hermite curve: α_2=1.0, α_3=0.1



(d) C^1 cubic Hermite curve: α_2=1.0, α_3=0.1, β_2=0.1



(e) C^2 cubic Hermite curve: α_2=1.0, α_3=0.1

Figure 3.3: The effect of different weighting values on curve's shape

Table 3.3: The accuracy of fit of the interpolating curves to a unit half circle with different weighting values

Case	e_{max}*	κ_{max}	κ_{min}
exact unit circle	0.0	1.0	1.0
Figure 3.3a	0.00891	2.704854	-1.7385
Figure 3.3b	0.004835	1.285663	0.00
Figure 3.3c	0.001243	1.11008	0.864166
Figure 3.3d	0.00033	1.033248	0.932816
Figure 3.3e	0.000096	1.012692	0.944818

* compare to a unit circle

We can see that only minimizing the stretching energy does not guarantee a fair curve. Only minimizing the bending energy results in zero curvature at the curve's ends. Only minimizing the variation of curvature requires more constraints and the use of a C^2 curve. Eventually, a question will arise: is there an optimal combination of weighting values for curve reconstruction? The optimal curve is the one which is able to fulfill the user's expectation of the curve's shape. Since different users may have different expectations for the same data set, any curve shape resulting from a set of weighting values could be defeated by another. When only position data is available, the weighting values can be left to the user's intention. However, users must be aware of the basic effect of these weighting values so that the curve's shape can be tuned to fulfill their requirements.

Some authors use varied weighting values to eliminate the unwanted undulations, the so-called "Gibbs phenomena", in the interpolation of data points with large variations. The basic idea is that since the minimization formulation tends to evenly distribute the weighted stretching and bending energy along the curve, using smaller weighting values for some regions will force the bending energy, and thus the curvature, to be larger and more capable of representing large data variation. Because this technique requires additional information (such as the estimated second derivatives at the data sites) to evaluate the data variation, it generally only applies to explicit curves since such information can be easily obtained by finite difference estimation. Incorporating such a technique into the curve reconstruction method described in this chapter did not significantly improve the ability to represent data points with wild variations. The reason is that during the minimization process, regions of the curve with smaller weighting

values have less stiffness, so it is for them to be stretched and bent. This causes regions with large variations to be represented by even fewer elements than for the constant weighting values case, which reduces the ability to represent the large variation.

3.4 CURVE QUALITY ANALYSIS

The purpose of curve quality analysis is to analyze both the accuracy of the fit and the quality of the curve being fitted in quantitative and qualitative manners. In this section, indicators used to evaluate the accuracy of fit and the quality of the curve will be discussed.

3.4.1 Accuracy of Fit

To analyze the accuracy of the fitted curve, an analytic target curve defined either parametrically or explicitly must be chosen, from which the data points are taken and errors are inserted so that comparisons may be made between the analytic curve and the fitted curve. Indicators showing the accuracy of fit should be defined also. It should be noted that defining a scalar metric which is able to properly indicate the "difference" between two arbitrary parametric curves is not an easy task. The difference measurement can neither be defined as the maximum nor the minimum of the distance function between two parametric curves, $\boldsymbol{T}(u)$ and $\boldsymbol{C}(v)$, represented as a bivariate function

$$D(u,v) = \|\boldsymbol{T}(u) - \boldsymbol{C}(v)\| \quad ; \quad u_{\min} \le u \le u_{\max}, v_{\min} \le v \le v_{\max} \tag{3.4.1}$$

The maximum distance associated to the stationary points of the distance function cannot be used as the indicators either. In the following, several indicators measuring the accuracy of fit between two parametric curves: $\boldsymbol{T}(u)$, the target curve and $\boldsymbol{C}(v)$, the fitted curve are defined.

First, a pointwise error measurement is defined as the minimum distance between any particular point on one curve to the other curve. For example, the following equation

$$e(\boldsymbol{C}(v_0)) = \min_{u}\{\|\boldsymbol{C}(v_0) - \boldsymbol{P}\| \,\big|\, \boldsymbol{P} \in \boldsymbol{T}(u), u_{\min} \le u \le u_{\max}\} \tag{3.4.2}$$

defines the minimum distance between a point on $\boldsymbol{C}(v)$ with parameter v_0 and the curve $\boldsymbol{T}(u)$. The point $\boldsymbol{P}$ on the target curve, which satisfies (3.4.2) could be either in the interior of $\boldsymbol{T}(u)$ or at its ends. If $\boldsymbol{P}$ is in the interior of the curve, it is easy to show that the line segment connecting $\boldsymbol{C}(v_0)$ and $\boldsymbol{P}$ and the tangent vector at $\boldsymbol{P}$ are mutually orthogonal. A global error measurement can then be defined as the maximum of the pointwise error measurements as

$$e_{\max} = \max\{e(v) | \; v_{\min} \le v \le v_{\max}\} \tag{3.4.3}$$

This measurement of difference is called the *supmetric*. Another measurement which is only suitable for planar curves is the so-called "L_2 norm" which measures the area bracketed by the two curves.

Since $e(v)$'s value is obtained by solving a single variable constrained nonlinear minimization problem, the procedure to evaluate $e_{\max}$ is computationally expensive. A simplification is made by sampling points on $\boldsymbol{C}(v)$, computing the pointwise error as defined above and picking up the maximum. It should be noted that the above definition for the supmetric generally does not result in same global error measurements for sampling points from curve $\boldsymbol{T}(u)$ and from curve $\boldsymbol{C}(v)$. However, the difference is small since these two curve are expected to be fairly close. A more robust definition for the supmetric could be made by selecting the larger one from the two candidates.

3.4.2 Second Order Interrogation Schemes

In evaluating the quality of a curve, fairness is an important indicator. However, there is still no formal definition of the fairness of a curve. The fairness of a curve is stated qualitatively rather than quantitatively. Generally, a curve is said to be fair when it has continuous curvature or its curvature distribution is almost piecewise linear with as few spans as possible. Consequently, plotting the curvature becomes a good way to evaluate the curve's fairness. Using a curvature plot, inflection points and curvature discontinuities can be easily identified. This is particularly true for planar curves. The importance of the curvature plot can be seen in figure 3.3 in which curves with different weighting values are visually indistinguishable without the curvature plot. The curvature of a space curve is always positive, but for planar curves, negative signs can be assigned to the curvature.

The formulas for calculating the curvature of space curves and the signed curvature of planar curves are provided in the appendix.

A curvature plot can be drawn in at least two different ways. One is to draw the regular 2D plot with the curvature's magnitude as the y-axis and the parametric values or the cumulative arc length as the x-axis. The other way is to draw line segments emanating from a number of points on the curve, along the normal of the curve or its opposite direction, whose length is proportional to the magnitude of the curvature at that point. Such a depiction is usually called a "curvature comb" due to its appearance (see figure 3.3). Drawing a curvature plot in the first way can easily show where the inflection points are and the techniques are generally used when comparisons are being made with respect to another curve. The curvature comb obviously has the advantage that not only the magnitude but also the direction can be displayed. It should be noted that when comparing the curvature plots of two parametric curves, the choice of x-axis should be made carefully. If the parametric values are used as the x-axis, different parametrizations can cause drastic differences to appear even if the two curves are fairly close. Using cumulative arc length or normalized cumulative arc length is a better choice.

3.4.3 Third Order Interrogation Schemes

A third order scheme proposed in [Sapidis 90] was used to evaluate the fairness of an integral cubic B-spline. A local fairness indicator z_i is defined as

$$z_i = \left|\kappa_s(u_i^+) - \kappa_s(u_i^-)\right| \tag{3.4.4}$$

where κ_s is the curvature's derivative with respect to the arc length. The global fairness indicator ζ is defined as the sum of all local indicators. This scheme was implemented as an interrogation tool of curve fairness in [Alourdas 89] and [Hottel 91]. In this regard, drawing the κ_s plot is also helpful in visualizing the fairness of a curve. Note that only the derivative of the curvature's magnitude is concerned.

3.5 SUMMARY

This chapter described a new method for generating a smooth parametric curve to approximate a set of error-filled, unordered data points. Users only need to specify the end points of the curve. An energy functional composed of a global and local irregularity measurement of the curve and a closeness measurement between data points and the curve was given. Curves are represented by finite-element Hermite polynomials. The successive quadratic programming technique is applied to find the solution. The resulting curves generally have very good fairness. Even when a C^1 curve is used, the minimization tends to seek a shape which is C^3 continuous. Compared to existing data point interpolation and approximation schemes, this method obviously has the advantage that no connectivity information other than the end points is required. The interpolation problem can also be treated as a special case of our method in which the data points to be interpolated are considered as constraints in the minimization process. On the other hand, since this algorithm does not assume connectivity information, the initial estimate of the solution for the minimization is often a straight line connecting the two end points specified by the user. This "universal" initial guess works quite well for most cases, but for data points 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. Yet, this does not raise any difficulty since self-intersecting curves are rarely found in practical applications.

CHAPTER FOUR

Surface Reconstruction

The need to reconstruct a surface from scattered data points arises frequently in many applications such as medical imaging, machine vision, industrial engineering, geometric modeling, etc. This problem can be stated as "Given a set of data points, P_i, $i = 0, \cdots, N-1$, taken from a target surface, reconstruct a surface which approximates the original one to satisfactorily with a visually acceptable appearance". Diverse methods have been developed for different purposes based on various principles. A general survey of previous work in this area is given in Chapter 2.

Surface reconstruction methods can be categorized in several ways: interpolation vs approximation schemes, global vs local schemes, and by the surface representation. When data points are subject to error measurements, an approximation scheme is more appropriate than an interpolation scheme. The choice of surface representation is heavily influenced by an application's purpose. When the ultimate goal is visualizing a shape or plotting a contour map of a reconstructed surface as in medical imaging and geological applications, using an explicit surface representation or even a collection of polygonal facets is sufficient. For industrial applications, such as the reverse engineering, a computer-based representation of a physical object is frequently constructed from data points measured from a model made of clay or some other prototyping material. In such applications, the purpose of constructing a CAD model is to facilitate further modification of the prototype design and to allow the calculation of analytic geometric

properties which are often required in the downstream structural or aerodynamic analysis and manufacturing process. For these purposes, a parametric representation for the surface is preferable to an explicit or implicit representation due to the fact that parametric surfaces are coordinate system invariant, easy to trace and easy to manipulate.

Many papers have addressed the problem of generating a surface from measured data points. However, most methods proposed use explicit surfaces only. Current parametric surface reconstruction schemes are all subject to restrictions, for example, the data points must form a rectangular grid, and the reconstructed surface can not be a general n-sided surface. In this chapter, a method is presented for the reconstruction of a smooth n-sided parametric surface from a set of unordered and error-filled data points with only partial topology information assumed. This method simulates the deformation of an arbitrary-shaped thin plate, constrained at its perimeter, under the application of spring forces. Similar to the curve reconstruction method, this problem is formulated through the variational principle, which seeks the optimal shape as the solution of a constrained minimization problem. The general n-sided surface is modeled by a triangle-based FE surface composed of a finite number of triangular surfaces connected with C^1 continuity. The parametric domain is generated from the surface's boundaries reconstructed from corresponding groups of points which are assumed to be known.

The rest of this chapter is organized as follows. In section 4.1, we give the energy functional to be minimized, its physical interpretation and the minimization method. Section 4.2 introduces two different triangular surface primitive. Section 4.3 describes the surface reconstruction procedure including input data, the creation of the parametric domain and mesh generation. Section 4.4 introduces some methods used to evaluate the quality of the reconstructed surface. Finally, section 4.5 summarizes this chapter.

4.1 VARIATIONAL FORMULATION

The variational formulation for surface reconstruction is very similar to that for curve reconstruction. They are both formulated as minimization problems. Both of the functionals to be minimized mimic the behavior of their physical counterparts in the real world. The same minimization technique is applied to solve the problems. Considering these similarities, the description of common or similar issues is simplified in this section to avoid redundancy.

4.1.1 Objective Function

Reconstructing a surface from a set of data points is an ill-posed problem. Similar to the curve reconstruction, it can be reformulated into a well-posed problem through the variational principle. The reformulated problem seeks the solution which minimizes

$$E_{surface}(w)=E_{irregularity}(w)+E_{closeness}(w, \mathbf{P}) \tag{4.1.1a}$$

with

$$E_{irregularity}(w)=\sum_{p=1}^{m}\sum_{\substack{i+j=p\\k+l=p}}\int_{\Omega}\alpha_{ijkl}(u,v)\left\|\frac{\partial^p w(u,v)}{\partial u^i \partial v^j}\right\|\left\|\frac{\partial^p w(u,v)}{\partial u^k \partial v^l}\right\|dudv \tag{4.1.1b}$$

$$+\sum_{p=0}^{m}\sum_{i+j=p}\sum_{k}\beta_{ijk}\int_{C_k}\left\|\Delta\left(\frac{\partial^p w(u,v)}{\partial u^i \partial v^j}\right)\right\|^2 ds$$

$$E_{closeness}(w, \mathbf{P})=\sum_{i=0}^{N-1}\lambda_i D_s(\boldsymbol{P}_i, w(u,v))^2 \tag{4.1.1c}$$

where $w(u,v)$ is a parametric surface, $\alpha_{ijkl}(u,v)$, β_p and λ_i are all weighting functions or factors and $D_s(P_i,w(u,v))$ is a distance metric defined as the shortest Euclidean distance between the data points and the surface. The $E_{irregularity}(w)$ term measuring the irregularity of the solution consisits of a global and a local local irregularity measurements. The $E_{closeness}(w, \mathbf{P})$ term is the closeness measurement between the data points set $\mathbf{P}$ and the surface $w(u,v)$. As in curve reconstruction, the surface is represented parametrically instead of explicitly. The closeness measurement is defined based on the shortest distances between the points and the surface instead of the distances between point pairs with the same parameter values. The local irregularity measurement in addition to the global one is also included. It measures the integral of position and derivative discontinuities occurred in the interior of the surface.

The value of m is taken as 2 in this thesis since the surface primitive we use only guarantees a C^1 surface. The β_{ijk} are set to the same value for derivatives of the same order, i.e., $\beta_{ijk} = \beta_p$ for all k and ij satisfying $i+j=p$. Furthermore, in most geometric modeling and aesthetic surface design applications, surfaces are required to be at least C^1 continuous, thus only discontinuity in the second derivatives needs to be considered. Thus, equation (4.1.1) can be practically simplified into

$$E_{surface}(w) = \int_\Omega \left[\mathbf{W}_s^T \bar{\alpha}_s \mathbf{W}_s + \mathbf{W}_b^T \bar{\alpha}_b \mathbf{W}_b\right] dudv \qquad (4.1.2)$$
$$+\beta_2 \sum_k \int_{C_k} \left[\Delta \mathbf{W}_b^T \Delta \mathbf{W}_b\right] ds + \sum_{i=0}^{N-1} \lambda_i D_s(P_i, w(u,v))^2$$

where, $\mathbf{W}_s = [w_u \;\; w_v]^T$, $\mathbf{W}_b = [w_{uu} \;\; w_{vv} \;\; w_{uv}]^T$, $\Delta \mathbf{W}_b = [\Delta w_{uu} \;\; \Delta w_{vv} \;\; \Delta w_{uv}]^T$, $\bar{\alpha}_s$ is a 2×2 weighting matrix and $\bar{\alpha}_b$ a 3×3 weighting matrix.

4.1.2 Physical Interpretation

A physical model of the functional described by equation (4.1.2) is illustrated in figure 4.1. A perfectly elastic plate with arbitrary shape, constrained at its perimeter and at some interior curves, 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 surface). Each spring is assumed to have one end anchored at a data point and the other end slides on the surface so that the shortest distance is maintained. The squared first derivative term in the functional stands for the strain energy due to stretching of the plate and the squared second derivative term reflects the strain energy due to bending. The error term corresponds to the strain energy stored in the springs.

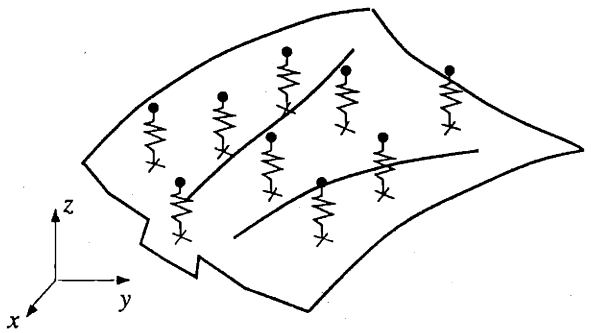


Figure 4.1: The physical model for surface reconstruction

The weighting matrices act to weight the stretching and bending parts in the overall energy of the shape. Increasing the $\overline{\alpha}_s$ values causes the shape to increasingly resist stretch and to find solutions of smaller area. Larger $\overline{\alpha}_b$ values will cause the surface to distribute its curvature over large regions resulting in flatter shapes.

4.1.3 Nonlinear Minimization

A continuous surface is approximated by superposing a finite series of continuous functions with weighting factors.

$$w(u,v) \approx \sum_{i}^{n} q_i \phi_i(u,v) = \Phi^{\mathrm{T}}(u,v)\mathbf{Q} \tag{4.1.3}$$

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

Substituting equation (4.1.3) into (4.1.2) results in an equation similar to equation (3.1.5)

$$E_{surface}(\mathbf{Q}) = \mathbf{Q}^{\mathrm{T}}\mathbf{K}\mathbf{Q} + \sum_{i=0}^{N-1} \lambda_i D_s^2(\boldsymbol{P}_i, \mathbf{Q}) \tag{4.1.4}$$

where $\mathbf{K} = \mathbf{K}_s + \mathbf{K}_b + \mathbf{K}_d$ is the system stiffness matrix,

$$\mathbf{K}_s = \int_{\Omega} \Phi_s^{\mathrm{T}} \overline{\alpha}_s \Phi_s \, du dv \quad ; \quad \mathbf{K}_b = \int_{\Omega} \Phi_b^{\mathrm{T}} \overline{\alpha}_b \Phi_b \, du dv \tag{4.1.5}$$
$$\text{with } \Phi_s = [\Phi_u \quad \Phi_v]^{\mathrm{T}} \text{ and } \Phi_b = [\Phi_{uu} \quad \Phi_{vv} \quad \Phi_{uv}]^{\mathrm{T}}$$

and $\mathbf{K}_d$ is a matrix resulting from the second order discontinuity if the $\phi_i(u,v)$ used are only C^1 continuous.

At least three constraints are necessary to guarantee that a stable solution can be obtained when minimizing the functional. These three constraints can be all position constraints or two position constraints plus one slope constraint in either the u or v direction. This is analogous to the necessary boundary conditions of a plate under lateral loading.

Again, successive quadratic programming is used to seek the solution.

4.2 THE SURFACE PRIMITIVE

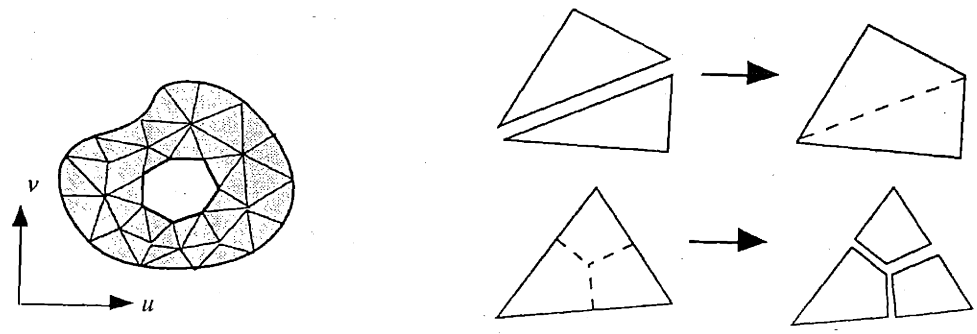
4.2.1 Selection of the Surface Primitive

All parametric surface schemes select a region in the parametric space for defining the domain of the mapping. The shape and the range of the mapping domain can have a significant effect on the shape of the object to be modeled. Many free-form surfaces such as B-spline and NURBS surfaces are based on quadrilateral patch which is defined on a rectangle in the parametric domain. However, when dealing with general n-sided surfaces, triangular patches have the following advantages over quadrilateral patches:

(1) Generality: A domain of any shape on the u-v plane, namely any surface topology, can be decomposed into triangles and any number of edges can meet at a vertex in any pattern of angles in a triangular mesh. Hence, the range of parametric shapes that can be modeled with triangular elements is much larger than is possible using rectangular elements.

(2) Uniformity: Representing general n-sided surfaces with triangular elements can reduce the effort required to develop algorithms for treating surfaces with different topologies.

(3) Flexibility: Triangular elements can be aggregated or decomposed into quadrilateral elements if necessary.



(a) A region of any shape can be decomposed into triangles

(b) Aggregation and decomposition of triangles into quadrilateral shapes.

Figure 4.2: Advantages of using triangular elements

For these reasons, the triangular patches were selected for the surface primitive. The next step was to select a triangular interpolant capable of fulfilling the application requirements. The nodal-basis type triangular interpolant was selected from the three different types of triangular interpolants introduced in section 2.4 because

- Continuity between triangular patches is easy to achieve as long as the interpolant's edge geometry is completely defined by the properties of the nodes on that edge.
- Topological information between patches is easy to record, namely,. That is, if two triangular patches' parametric domains are adjacent, they are also adjacent in three dimensional space and vise versa.
- Representing a surface as a weighted sum of its geometric properties at discrete nodes allows easy imposition of geometric constraints.

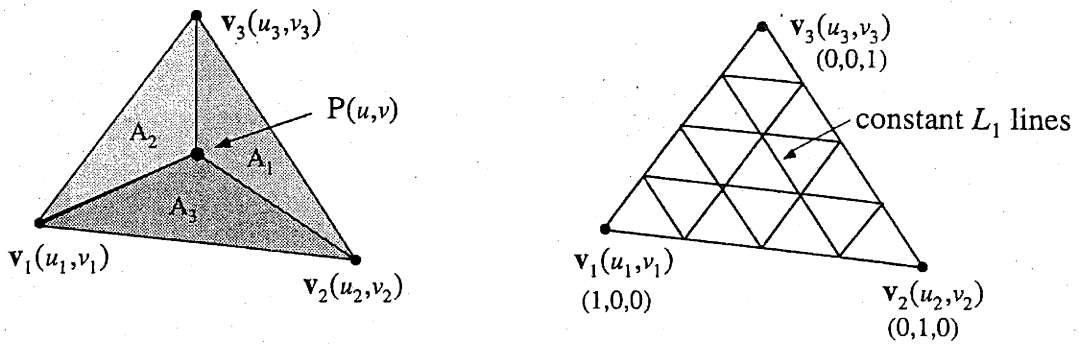
The following sections will introduce two nodal-basis type conforming triangular interpolants: a 12-dof interpolant with cubic boundary curves and a 21-dof interpolant with quintic boundary curves. These two interpolants were selected to match the cubic and quintic Hermite curves used for curve reconstruction. A surface reconstruction system described in this chapter can be implemented based on either the combination of cubic Hermite curve and 12-dof triangular interpolant or quintic Hermite curve and 21-dof triangular interpolant. The system that were actually implemented takes the former combination.

4.2.2 Twelve Degrees of Freedom Triangular Interpolant

4.2.2.1 Barycentric Coordinates

Barycentric coordinates are an alternative to Cartesian coordinates for defining locations in two dimensional or three dimensional space. Barycentric coordinates in 2 dimensions are a natural choice for defining functions that apply to triangular domains. Two-dimensional barycentric coordinates are defined with respect to a triangle. The barycentric coordinates of any point inside this triangle can be obtained by finding the area ratios as

$$(L_1, L_2, L_3) = (\frac{A_1}{A}, \frac{A_2}{A}, \frac{A_3}{A}) \;\; ; A = A_1 + A_2 + A_3 \tag{4.2.1}$$



(a) The coordinates defined via area ratios (b) The constant barycentric coordinates lines

Figure 4.3: The barycentric coordinates in two dimensions (the area coordinates)

where, A is the area of the triangle and A_1, A_2 and A_3 are the areas of the three subtriangles as illustrated in figure 4.3a. Lines of constant L_1 values will be parallel to the edge opposite the v_1 vertex. The line $L_1 = 0$ is coincident with the opposite edge as in figure 4.3b. The same is true for lines of constant L_2 and L_3. Two dimensional barycentric coordinates are also called area coordinates in the finite element community due to this relation.

Conversion between barycentric and Cartesian coordinates of any point on a plane can be done by

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} u_1 & u_2 & u_3 \\ v_1 & v_2 & v_3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} L_1 \\ L_2 \\ L_3 \end{bmatrix} \tag{4.2.2}$$

and

$$\begin{bmatrix} L_1 \\ L_2 \\ L_3 \end{bmatrix} = \frac{1}{2\Delta} \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix} \begin{bmatrix} 1 \\ u \\ v \end{bmatrix} \; ; \; \frac{1}{2\Delta} \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix} \equiv \mathbf{L} \tag{4.2.3}$$

where the point is given as (u,v) in Cartesian coordinates and (L_1, L_2, L_3) in barycentric coordinates. The mapping triangle is defined by three vertices (u_1,v_1), (u_2,v_2) and (u_3,v_3). Δ stands for the triangle's area. The $\mathbf{L}$ matrix completely characterizes the shape

of the triangle up to rigid body translations. Elements of the **L** matrix can be calculated by

$$\begin{aligned} a_i &= u_j v_k - u_k v_j \Rightarrow \textstyle\sum_i a_i / 2\Delta = 1 \; ; \\ b_i &= v_j - v_k \quad\;\; \Rightarrow \textstyle\sum_i b_i = 0 \quad\; ; \\ c_i &= u_k - u_j \quad\;\; \Rightarrow \textstyle\sum_i c_i = 0 \quad\; ; \end{aligned} \tag{4.2.4}$$

where the (i, j, k) are cyclic indices, namely, (i, j, k)=(1, 2, 3), (2, 3, 1) or (3, 1, 2).

Since the barycentric mapping is linear, differentiating barycentric functions in terms of the Cartesian coordinates is very simple. Applying the chain rule for first order derivatives yields

$$\begin{bmatrix} \frac{\partial}{\partial u} \\ \frac{\partial}{\partial v} \end{bmatrix} = \begin{bmatrix} \frac{\partial L_1}{\partial u} & \frac{\partial L_2}{\partial u} & \frac{\partial L_3}{\partial u} \\ \frac{\partial L_1}{\partial v} & \frac{\partial L_2}{\partial v} & \frac{\partial L_3}{\partial v} \end{bmatrix} \begin{bmatrix} \frac{\partial}{\partial L_1} \\ \frac{\partial}{\partial L_2} \\ \frac{\partial}{\partial L_3} \end{bmatrix} = \frac{1}{2\Delta} \begin{bmatrix} b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{bmatrix} \begin{bmatrix} \frac{\partial}{\partial L_1} \\ \frac{\partial}{\partial L_2} \\ \frac{\partial}{\partial L_3} \end{bmatrix} = \mathbf{J}_1 \nabla_{\mathrm{L}} \tag{4.2.5}$$

Applying the chain rule twice yields the relationship for second order differentials;

$$\begin{bmatrix} \frac{\partial^2}{\partial u^2} \\ \frac{\partial^2}{\partial v^2} \\ \frac{\partial^2}{\partial u \partial v} \end{bmatrix} = \frac{\begin{bmatrix} b_1^2 & 2b_1b_2 & b_2^2 & 2b_2b_3 & b_3^2 & 2b_3b_1 \\ c_1^2 & 2c_1c_2 & c_2^2 & 2c_2c_3 & c_3^2 & 2c_3c_1 \\ b_1c_1 & b_1c_2 + b_2c_1 & b_2c_2 & b_2c_3 + b_3c_2 & b_3c_3 & b_3c_1 + b_1c_3 \end{bmatrix}}{4\Delta^2} \begin{bmatrix} \frac{\partial^2}{\partial L_1 \partial L_1} \\ \frac{\partial^2}{\partial L_1 \partial L_1} \\ \frac{\partial^2}{\partial L_2 \partial L_2} \\ \frac{\partial^2}{\partial L_2 \partial L_3} \\ \frac{\partial^2}{\partial L_3 \partial L_3} \\ \frac{\partial^2}{\partial L_3 \partial L_1} \end{bmatrix} \tag{4.2.6}$$

$$= \mathbf{J}_2 \nabla_{\mathrm{L}}^2$$

In addition to the Cartesian derivatives, derivatives of the barycentric functions can be taken in the directions tangent or normal to the edges of the mapping triangle. The direction of the tangent vector, t, to an edge is defined so that its associated normal, n, has a positive v component and the cross product, $t \times n$, is positive. The angle between the u axis and a tangent vector, γ_{ij}, for edge ij will always obey the relation,

$$0 \le \gamma_{ij} < \frac{\pi}{2} \text{ or } \frac{3\pi}{2} \le \gamma_{ij} < 2\pi \tag{4.2.7}$$

Figure 4.4 shows the relation between the mapping triangle, positive normals and the tangent angles. The orthonormal transformation from the Cartesian coordinates to the tangent-normal coordinates is given by

$$\begin{bmatrix} t_{ij} \\ n_{ij} \end{bmatrix} = \begin{bmatrix} \cos(\gamma_{ij}) & \sin(\gamma_{ij}) \\ -\sin(\gamma_{ij}) & \cos(\gamma_{ij}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \tag{4.2.8}$$

Again, by applying the chain rule, derivatives in the tangent and the normal of edge ij can be related to the derivatives in the Cartesian coordinates or barycentric coordinates. The relation of the first derivatives in different coordinates is

$$\begin{aligned} \begin{bmatrix} \frac{\partial}{\partial t_{ij}} \\ \frac{\partial}{\partial n_{ij}} \end{bmatrix} &= \begin{bmatrix} \frac{\partial u}{\partial t_{ij}} & \frac{\partial v}{\partial t_{ij}} \\ \frac{\partial u}{\partial n_{ij}} & \frac{\partial v}{\partial n_{ij}} \end{bmatrix} \begin{bmatrix} \frac{\partial}{\partial u} \\ \frac{\partial}{\partial v} \end{bmatrix} = \begin{bmatrix} \cos(\gamma_{ij}) & \sin(\gamma_{ij}) \\ -\sin(\gamma_{ij}) & \cos(\gamma_{ij}) \end{bmatrix} \begin{bmatrix} \frac{\partial}{\partial u} \\ \frac{\partial}{\partial v} \end{bmatrix} = \mathbf{J}_{\gamma_{ij}} \begin{bmatrix} \frac{\partial}{\partial u} \\ \frac{\partial}{\partial v} \end{bmatrix} \\ &= \mathbf{J}_{\gamma_{ij}} \mathbf{J}_{\mathrm{I}} \nabla_{\mathrm{L}} = \mathbf{T}^{(1)}_{(tn)_{ij}} \nabla_{\mathrm{L}} \end{aligned} \tag{4.2.9}$$

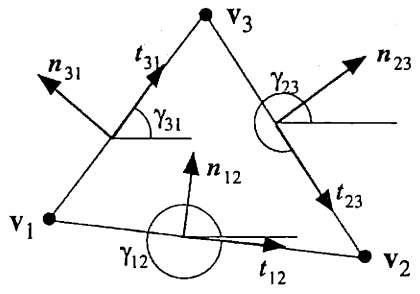
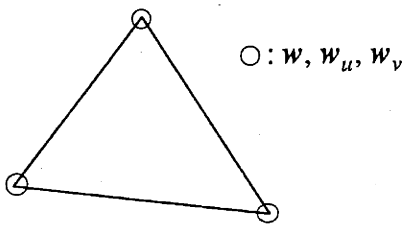
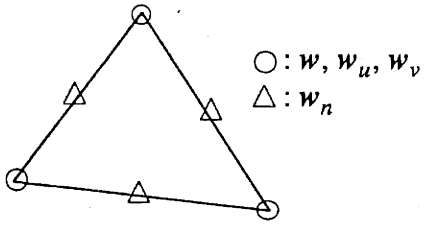


Figure 4.4: Edge tangents and normals of a barycentric mapping triangle



(a) The 9-dof triangular element



(b) The 12-dof triangular element

Figure 4.5: Degree of freedom maps of two triangular elements

and that for the second derivatives is

$$\begin{bmatrix} \dfrac{\partial^2}{\partial t_{ij}^2} \\ \dfrac{\partial^2}{\partial n_{ij}^2} \\ \dfrac{\partial^2}{\partial t_{ij}\partial n_{ij}} \end{bmatrix} = \begin{bmatrix} c^2 & s^2 & 2sc \\ s^2 & c^2 & -2sc \\ -sc & sc & c^2 - s^2 \end{bmatrix} \begin{bmatrix} \dfrac{\partial^2}{\partial u^2} \\ \dfrac{\partial^2}{\partial v^2} \\ \dfrac{\partial^2}{\partial u \partial v} \end{bmatrix} = \mathbf{J}_{\gamma_{ij}}^{(2)} \begin{bmatrix} \dfrac{\partial^2}{\partial u^2} \\ \dfrac{\partial^2}{\partial v^2} \\ \dfrac{\partial^2}{\partial u \partial v} \end{bmatrix} \tag{4.2.10}$$

$$= \mathbf{J}_{\gamma_{ij}}^{(2)} \mathbf{J}_2 \nabla_{\mathrm{L}}^2 = \mathbf{T}_{(tn)_{ij}}^{(2)} \nabla_{\mathrm{L}}^2$$

where $c = \cos(\gamma_{ij})$ and $s = \sin(\gamma_{ij})$.

4.2.2.2 9-dof Cubic Triangular Element

The 9-dof cubic triangular element is an interpolant which interpolates the position and tangent vectors in the u and v directions at nodes located at the triangle's vertices as described by the following equation and illustrated in figure 4.5a.

$$w(u,v) = \mathbf{N}^9(L_1, L_2, L_3, \mathbf{L})\, \mathbf{a}_9^{(e)} \tag{4.2.11}$$

where $\mathbf{a}_9^{(e)}$ contains the ordered degrees of freedom for one triangle element as

$$\mathbf{a}_9^{(e)} = \begin{bmatrix} w_1 & w_{u1} & w_{v1} & w_2 & w_{u2} & w_{v2} & w_3 & w_{u3} & w_{v3} \end{bmatrix}^{\mathrm{T}} \tag{4.2.12}$$

where $w_1 = w(u_1, v_1)$, $w_{u1} = w_u(u_1, v_1), \cdots$, etc

and $\mathbf{N}^9(L_1, L_2, L_3, \mathbf{L})$ is a set of interpolation functions, called shape functions, defined in barycentric coordinates over the triangle defined by three vertices (u_i, v_i), i=1 to 3.

One of the advantages of defining the degrees of freedom of the interpolant by its geometric properties is that it simplifies the generation of the associated set of shape functions. The *i*th shape function is required to be 1 for the geometric property at the node corresponding to the *i*th degree of freedom for the triangle and be zero for all other geometric properties. Zienkiewicz first published these functions for such a triangle in 1969 as the preferred element for solving linear plate bending problems. They are based on the "f" functions shown in figure 4.6 and are expressed as

$$\mathbf{N}^9(L_1,L_2,L_3,\mathbf{L}) = \begin{bmatrix} \mathbf{N}_1^9 & \mathbf{N}_2^9 & \mathbf{N}_3^9 \end{bmatrix} \tag{4.2.13}$$

where $\mathbf{N}_1^9$, $\mathbf{N}_2^9$ and $\mathbf{N}_3^9$ are three 1×3 matrices each of which is associated to one of the nodes of the triangle.

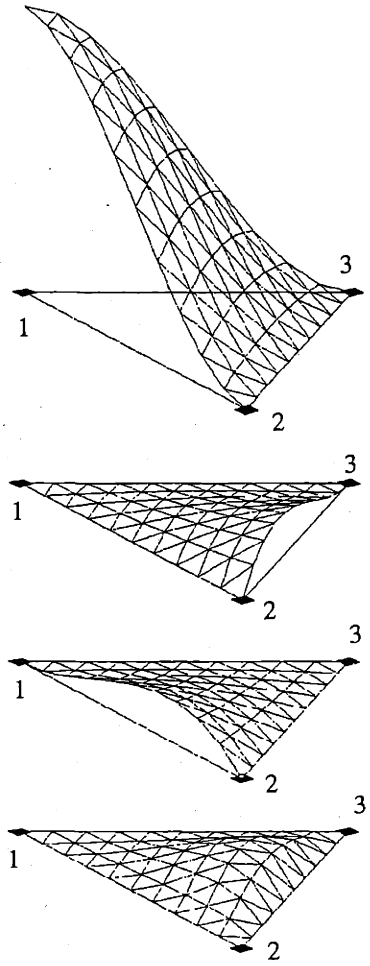
$$\begin{aligned} \mathbf{N}_1^{9\mathrm{T}} &= \begin{bmatrix} L_1 + L_1^2L_2 + L_1^2L_3 - L_1L_2^2 - L_1L_3^2 \\ c_3(L_1^2L_2 + \frac{1}{2}L_1L_2L_3) - c_2(L_1^2L_3 + \frac{1}{2}L_1L_2L_3) \\ -b_3(L_1^2L_2 + \frac{1}{2}L_1L_2L_3) + b_2(L_1^2L_3 + \frac{1}{2}L_1L_2L_3) \end{bmatrix} \\ &= \begin{bmatrix} \varphi_1 \\ \varphi_2 \\ \varphi_3 \end{bmatrix} = \begin{bmatrix} f_1 \\ c_3f_{12} - c_2f_{13} \\ -b_3f_{12} + b_2f_{13} \end{bmatrix} \end{aligned} \tag{4.2.14}$$

The definition of $\mathbf{N}_2^9$ and $\mathbf{N}_3^9$ can be completed by using the symmetry of the functions defined over a triangle in barycentric coordinates. To generate $\mathbf{N}_2^9$ use the above equations but replace (1, 2, 3) indices with (2, 3, 1) indices for all the subscripts. Replacing the (1, 2, 3) indices with (3, 1, 2) indices will generate $\mathbf{N}_3^9$.

By carefully examining the components of the f functions, it can be found that they are composed of three types of basic functions. The properties of the f functions actually come from the combination of these basic functions.

(1) $\bar{f}(L_1,L_2,L_3) = L_i \quad ; \; i \in \{1, 2, 3\}$

This shape is a flat plane. Its value at vertex i is 1 and zero at the other vertices. The shape along the edge opposite to vertex i is also zero.



$$f_1 = L_1 + L_1^2 L_2 + L_1^2 L_3 - L_1 L_2^2 - L_1 L_3^2$$

$$f_{23} = L_2^2 L_3 + \tfrac{1}{2} L_1 L_2 L_3$$

$$f_{21} = L_2^2 L_1 + \tfrac{1}{2} L_1 L_2 L_3$$

$$e_{23} = \frac{L_1 L_2^2 L_3^2 (1 + L_1)}{(L_1 + L_2)(L_1 + L_3)}$$

Figure 4.6: The "f" and "e" functions used to build the shape functions. (adapted from [Celniker 90a])

(2) $\bar{f}(L_1, L_2, L_3) = L_i^2 L_j \quad ; \; i, j \in \{1, 2, 3\}, \; i \neq j$

This shape and its normal and tangent derivatives are zero at all three vertices except that the normal derivative at vertex i is non-zero. The shape geometry along edge ki and edge kj ($k \in \{1, 2, 3\}, \; k \neq i, k \neq j$) is zero implying that the tangent derivatives along those edges are zero. The value of $\nabla_L \bar{f}(L_i = 0) = 0$ along edge kj implying that the normal derivative is zero along that entire edge as well.

(3) $\bar{f}(L_1, L_2, L_3) = L_1 L_2 L_3$

This shape is zero along all three edges which implies that the tangent derivatives are zero on all the edges. The normal derivatives are zero at the vertices but vary along each

edge. This function is known as the bubble function, since its addition to an existing shape will only change the interior geometry and will not affect the edge geometry or the surface tangents at the vertex locations.

The properties of the $\mathbf{N}^9$ functions can be examined by looking only at the $\mathbf{N}_1^9$ functions and relying on symmetry to extend the results to the two other nodes. Substituting $L_1 = 0$ and $L_2 = 1 - L_3$ into equation (4.2.14) and the yielded cubic equation in L_3 reveals that the shape along edge 23 varies cubically and is completely described by the information at its end nodes, i.e., node 2 and node 3.

To find the behavior of the derivatives in the tangent and normal directions along the edges, one has to compute $\nabla_L \mathbf{N}^9$, the derivatives in barycentric coordinates of the shape functions. Only the formulas for $\mathbf{N}_1^9$ are given below. The others can be found similarly by changing the indices of the subscripts on both sides of the equal sign.

$$\nabla_L \mathbf{N}_1^9 = \begin{bmatrix} \partial \mathbf{N}_1^9 / \partial L_1 \\ \partial \mathbf{N}_1^9 / \partial L_2 \\ \partial \mathbf{N}_1^9 / \partial L_3 \end{bmatrix} = \begin{bmatrix} \mathbf{N}_{1L_1}^9 \\ \mathbf{N}_{1L_2}^9 \\ \mathbf{N}_{1L_3}^9 \end{bmatrix} \tag{4.2.15}$$

where

$$\begin{aligned}
\mathbf{N}_{1L_1}^{9T} &= \begin{bmatrix} 1 + 2L_1L_2 + 2L_1L_3 - L_2^2 - L_3^2 \\ c_3(2L_1L_2 + \frac{1}{2}L_2L_3) - c_2(2L_1L_3 + \frac{1}{2}L_2L_3) \\ -b_3(2L_1L_2 + \frac{1}{2}L_2L_3) + b_2(L_1L_3 + \frac{1}{2}L_2L_3) \end{bmatrix} \\
\mathbf{N}_{1L_2}^{9T} &= \begin{bmatrix} L_1^2 - 2L_1L_2 \\ c_3(L_1^2 + \frac{1}{2}L_1L_3) - c_2(\frac{1}{2}L_1L_3) \\ -b_3(L_1^2 + \frac{1}{2}L_1L_3) + b_2(\frac{1}{2}L_1L_3) \end{bmatrix} \\
\mathbf{N}_{1L_3}^{9T} &= \begin{bmatrix} L_1^2 - 2L_1L_3 \\ c_3(\frac{1}{2}L_1L_2) - c_2(L_1^2 + \frac{1}{2}L_1L_2) \\ -b_3(\frac{1}{2}L_1L_2) + b_2(L_1^2 + \frac{1}{2}L_1L_2) \end{bmatrix}
\end{aligned} \tag{4.2.16}$$

Checking the geometry along edge 23, by substituting $L_1 = 0$ and $L_2 = 1 - L_3$ into equations (4.2.16) and (4.2.15), reveals that both the tangent and normal derivatives vary

parabolically in L_3. The tangent derivative depends on the properties at nodes 2 and 3 only, but the normal derivative depends on the properties at all three nodes.

Since the normal derivatives across the triangle's edges depend on the properties of all the three nodes, C^1 continuity between triangles is not guaranteed. In general, any property that must be continuous across a finite element boundary must be computed only by the data of the nodes on that boundary.

4.2.2.3 12-dof Triangular Element

As described in the previous section, the shape of the 9-dof triangular element varies cubically along each edge and the normal derivative varies parabolically. It will require at least 7 degrees of freedom to be able to define both the shape and the normal derivative geometry along an edge: four degrees of freedom to define the shape of a cubic function and three to fix the shape of a parabolic function. The 9-dof triangle has only six degrees of freedom on an edge and is incapable of supporting C^1 continuity cross the boundary. The 12-dof triangular element is defined by adding three degrees of freedom, taken as the normal derivative at the mid-side of each edge, to the 9-dof triangle as shown in figure 4.5b.

The $\mathbf{N}^{12}$ shape functions can be built based on the 9-dof shape functions. First adjust the equations so that the modified 9-dof shape functions have zero normal derivatives at the mid-nodes. The second step is to add in proper functions so that the degrees of freedom at the new nodes directly set the mid-edge normal derivatives. Note that the added functions must not change the shape and the tangent derivative along the edges but must have a unit normal derivative at the mid-edge node. The e functions introduced by Zienkiewicz have a shape value of zero along all three edges and the only non-zero shape edge derivative is in the normal direction of one of the three edges. Along that edge, the normal derivative varies parabolically with its maximum at the center of the edge. The e functions can be used to define the $\mathbf{N}^{12}$ shape functions of the 12-dof triangle.

The effect of the 9-dof shape functions on the normal derivative at the mid-edges can be found by evaluating the functions at the new nodes as

$$\begin{bmatrix} w_{n12}(\frac{1}{2},\frac{1}{2},0) \\ w_{n23}(0,\frac{1}{2},\frac{1}{2}) \\ w_{n31}(\frac{1}{2},0,\frac{1}{2}) \end{bmatrix} = \mathbf{Z}\mathbf{a}_9^{(e)} \tag{4.2.17}$$

The matrix $\mathbf{Z}$ is constant depending only on the terms of the $\mathbf{L}$ matrix defining the barycentric mapping. A new set of shape functions can be written that independently set their 9 degrees of freedom while leaving the mid-side normal derivatives at zero:

$$\overline{w} = \left(\mathbf{N}^9 - 4 \begin{bmatrix} \frac{e_{12}}{d_{12}} & \frac{e_{23}}{d_{23}} & \frac{e_{31}}{d_{31}} \end{bmatrix} \mathbf{Z} \right) \mathbf{a}_9^{(e)} \tag{4.2.18}$$

$$\text{with} \quad d_{12} = (-b_3 \sin(\gamma_{12}) + c_3 \cos(\gamma_{12}))/2\Delta \ ;$$
$$d_{23} = (-b_1 \sin(\gamma_{23}) + c_1 \cos(\gamma_{23}))/2\Delta \ ;$$
$$d_{31} = (-b_2 \sin(\gamma_{31}) + c_2 \cos(\gamma_{31}))/2\Delta \ ;$$

where e_{12}, e_{23} and e_{31} are the e functions. The e_{23} function is listed in figure 4.6. The others can be found by changing the subscripts cyclically. The d_{ij}s are elements of matrices $\mathbf{T}^{(1)}_{(tn)_{ij}}$ which converts the derivatives in barycentric coordinates into tangent-normal coordinates. The 12-dof shape functions can now be generated by adding the appropriately weighted e functions to set the mid-side normal derivatives to equal the values of the mid-side nodal degrees of freedom as

$$w = \left\{ \left[\mathbf{N}^9 \mid 0 \quad 0 \quad 0 \right] + 4 \begin{bmatrix} \frac{e_{12}}{d_{12}} & \frac{e_{23}}{d_{23}} & \frac{e_{31}}{d_{31}} \end{bmatrix} \left[-\mathbf{Z} \mid \begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{matrix} \right] \right\} \left[\begin{array}{c} \mathbf{a}_9^{(e)} \\ \hline w_{n12} \\ w_{n23} \\ w_{n31} \end{array} \right] \tag{4.2.19}$$

$$= \mathbf{N}^{12}\mathbf{a}_{12}^{(e)}$$

where $\mathbf{a}_{12}^{(e)} = \begin{bmatrix} w_1 & w_{u1} & w_{v1} & w_2 & w_{u2} & w_{v2} & w_3 & w_{u3} & w_{v3} & w_{n12} & w_{n23} & w_{n31} \end{bmatrix}^T$.

This builds a set of shape functions that has the required 7 degrees of freedom per edge so that both the shape and the normal derivative can be set independently in terms of the properties at the nodes on that edge. These 12-dof triangular shape functions support C^1 continuity between adjoining elements. The triangle shape varies cubically and the normal derivative varies parabolically along each edge.

The e functions are rational polynomials and have singularities in the evaluation of the cross derivative w_{uv} at the triangle vertices. This singularity is required of a triangle that enforces C^1 continuity without using the second order derivatives at the vertex nodes. It should be noted that other rational functions can be used to achieve the cross-boundary continuity as long as they have same normal derivatives distribution as the interpolant to be corrected along the boundary and present no effect on the shape and tangent along the boundary. Though the surface's cross derivatives are not defined at the triangle's vertices, the singularity of the cross derivative does not present problems [Celniker 90a]. The actual surfaces that have been generated are continuous and appear well behaved in the neighborhood of the element vertices.

4.2.3 Twenty-one Dof Triangular Interpolant

Requiring that the geometry along an edge be defined in terms of the nodal information on that edge guarantees inter-element continuities in position and derivatives. However, it also raises the problem that the cross derivative at the common vertex of two edges is not unique [Zienkiewicz 67]. This problem can be resolved in two ways. The first is to use functions which have singularities at the common nodes as the e functions for the 12-dof triangular elements. The second is to introduce the cross derivative as one of the degrees of freedom at the common node. The bicubic Hermite patch is an example. However, this requires that the angles at all the edge intersections be the same as in rectangular or parallel line grids. For triangular meshes in which any number of edges can meet in any number of angles, the specification of all second derivatives at the node becomes necessary. This will result in a 21-dof triangular element.

This 21-dof triangular interpolant is defined by a complete fifth order polynomial expansion in u and v containing the following twenty-one items:

$$
\begin{array}{ccccccc}
 & & & 1 & & & \\
 & & & u & v & & \\
 & & u^2 & uv & v^2 & & \\
 & u^3 & u^2v & uv^2 & v^3 & & \\
u^4 & u^3v & u^2v^2 & uv^3 & v^4 & &
\end{array}
$$

$$
u^5 \quad u^4v \quad u^3v^2 \quad u^2v^3 \quad uv^4 \quad v^5
$$

It is the triangular interpolant, defined in polynomial form and of minimal degree, which achieves C^1 continuity cross the boundaries. It interpolates the position and derivatives up

to the second order at the triangle's three vertices and normal derivatives at the midedge nodes as described by the following equation and illustrated in figure 4.7.

$$w(u,v) = \mathbf{N}^{21}(u,v)\mathbf{a}_{21}^{(e)} = \mathbf{UC}\,\mathbf{a}_{21}^{(e)} \tag{4.2.20}$$

where $\mathbf{a}_{21}^{(e)}$ contains the ordered degrees of freedom of one element as

$$\begin{aligned} \mathbf{a}_{21}^{(e)} &= \begin{bmatrix} \mathbf{w}_1 & \mathbf{w}_2 & \mathbf{w}_3 & \mathbf{w}_n \end{bmatrix}^{\mathrm{T}} \\ \mathbf{w}_i &= \begin{bmatrix} w & w_u & w_v & w_{uu} & w_{vv} & w_{uv} \end{bmatrix} \text{ at vertex } i,\ i \in \{1,2,3\}, \text{ and} \\ \mathbf{w}_n &= \begin{bmatrix} w_{n12} & w_{n23} & w_{n31} \end{bmatrix} \end{aligned} \tag{4.2.21}$$

and $\mathbf{N}^{21}(u,v)$ is a set of shape functions written in matrix form $\mathbf{N}^{21} = \mathbf{UC}$, with $\mathbf{U} = \begin{bmatrix} 1 & u & v & u^2 & \cdots & uv^4 & v^5 \end{bmatrix}$ and $\mathbf{C}$, a 21×21 coefficient matrix.

Note that the shape functions should satisfy the requirement that the ith function has a unit value for the geometric property at the node corresponding to the ith degree of freedom and zero for all other geometric properties. Thus given a triangle defined by three noncollinear vertices (u_1,v_1), (u_2,v_2) and (u_3,v_3), the coefficient matrix which completely specifies the twenty-one shape functions can be determined by finding the inverse of a 21×21 matrix as

$$\overline{\mathbf{U}}\mathbf{C} = \mathbf{I} \quad \text{or} \quad \mathbf{C} = \overline{\mathbf{U}}^{-1} \tag{4.2.22}$$

where $\overline{\mathbf{U}} = \begin{bmatrix} \mathbf{V}_1 & \mathbf{V}_2 & \mathbf{V}_3 & \mathbf{V}_n \end{bmatrix}$, with $\mathbf{V}_i = \begin{bmatrix} \mathbf{U}^{\mathrm{T}} & \mathbf{U}_u^{\mathrm{T}} & \mathbf{U}_v^{\mathrm{T}} & \mathbf{U}_{uu}^{\mathrm{T}} & \mathbf{U}_{vv}^{\mathrm{T}} & \mathbf{U}_{uv}^{\mathrm{T}} \end{bmatrix}\Big|_{(u_i,v_i)}$

$$\mathbf{V}_n = \left[\mathbf{U}_{n12}^{\mathrm{T}}\Big|_{\left(\frac{1}{2}(u_1+u_2),\frac{1}{2}(v_1+v_2)\right)} \quad \mathbf{U}_{n23}^{\mathrm{T}}\Big|_{\left(\frac{1}{2}(u_2+u_3),\frac{1}{2}(v_2+v_3)\right)} \quad \mathbf{U}_{n31}^{\mathrm{T}}\Big|_{\left(\frac{1}{2}(u_3+u_1),\frac{1}{2}(v_3+v_1)\right)} \right],$$

and $\mathbf{I}$ is a 21×21 identity matrix.

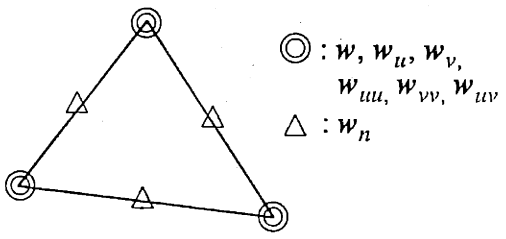


Figure 4.7: Triangular element with 21 degrees of freedom

The shape along the edges of this 21-dof triangular interpolant varies quintically and the normal derivatives vary quartically along the edge. Unlike the 12-dof element described above, it has no singularities in the cross derivatives at vertices, however, still only C^1 continuity is ensured across the triangle's boundary.

4.2.4 Triangular FE Surface Stiffness Matrix

A triangular FE surface is a surface defined over a triangulated finite domain on the parametric plane. Each triangle in the domain, also called an element, corresponds to a triangular surface patch in space. The continuity between triangular patches are determined according to the triangular interpolant used, for example, the above-mentioned 12-dof and 21-dof interpolant ensure C^1 continuity across the patches' boundaries.

By using a piecewise continuous FE surface, calculation of $\mathbf{K}_s$ and $\mathbf{K}_b$ in (4.1.5) can be simplified to the evaluation of elementary stiffness matrices as

$$\mathbf{K}^e = \mathbf{K}_s^e + \mathbf{K}_b^e \tag{4.2.23}$$

$$\text{with } \mathbf{K}_s^e = \int_\Omega \left(\mathrm{N}_\mathrm{s}^\mathrm{T}\overline{\alpha}_s\mathrm{N}_\mathrm{s}\right) dudv, \quad \mathbf{K}_b^e = \int_\Omega \left(\mathrm{N}_\mathrm{b}^\mathrm{T}\overline{\alpha}_b\mathrm{N}_\mathrm{b}\right) dudv,$$
$$\mathrm{N}_\mathrm{s} = [\mathrm{N}_u \quad \mathrm{N}_v]^\mathrm{T} \quad \text{and} \quad \mathrm{N}_\mathrm{b} = [\mathrm{N}_{uu} \quad \mathrm{N}_{vv} \quad \mathrm{N}_{uv}]^\mathrm{T}$$

where N contains the shape functions, i.e., $\mathrm{N}^{12}(L_1, L_2, L_3)$ for the 12-dof interpolant and $\mathrm{N}^{21}(u, v)$ for the 21-dof interpolant. For the 12-dof interpolant, the integral are evaluated approximately with the Gaussian-Legendre quadrature because the actual shape functions are too complicated to be evaluated analytically due to the rational "e" functions. For the 21-dof interpolant, evaluation of the elementary stiffness matrices can be analytically precalculated once and used for each triangular element provided all the elements of the weighting matrices are constant.

Because a FE surface using the 12-dof (or the 21-dof) triangular elements does not ensure C^2 continuity across triangle's boundaries,. the $\mathbf{K}_d$ matrix are resulted from the C^2 discontinuity along each common edge of two adjacent triangles. For two adjacent triangles denoted by α and β, the second derivatives discontinuity along the common edge pq is represented as

$$\Delta \mathbf{W}_b = \begin{bmatrix} \Delta w_{uu}|_{\text{edge } pq} \\ \Delta w_{vv}|_{\text{edge } pq} \\ \Delta w_{uv}|_{\text{edge } pq} \end{bmatrix} = \mathbf{T}_\alpha \, \mathbf{a}_\alpha^e - \mathbf{T}_\beta \, \mathbf{a}_\beta^e \quad ; \quad \mathbf{T}_i = \begin{bmatrix} \mathbf{N}_{uu}^{(i)}|_{\text{edge } pq} \\ \mathbf{N}_{vv}^{(i)}|_{\text{edge } pq} \\ \mathbf{N}_{uv}^{(i)}|_{\text{edge } pq} \end{bmatrix} \quad ; \quad i \in (\alpha, \beta) \tag{4.2.24}$$

Integrating the squared discontinuity in the second derivatives along the triangle's boundary results in

$$\int_0^l \Delta \mathbf{W}_b^{\mathrm{T}} \Delta \mathbf{W}_b \, dL = \mathbf{a}_\alpha^{e\,\mathrm{T}} \mathbf{K}_{d_{\alpha\alpha}}^e \mathbf{a}_\alpha^e + \mathbf{a}_\beta^{e\,\mathrm{T}} \mathbf{K}_{d_{\beta\beta}}^e \mathbf{a}_\beta^e - \mathbf{a}_\alpha^{e\,\mathrm{T}} \mathbf{K}_{d_{\alpha\beta}}^e \mathbf{a}_\beta^e - \mathbf{a}_\beta^{e\,\mathrm{T}} \mathbf{K}_{d_{\beta\alpha}}^e \mathbf{a}_\alpha^e \tag{4.2.25}$$

$$\text{with } \mathbf{K}_{d_{ij}}^e = \int_0^l \mathbf{T}_i^{\mathrm{T}} \mathbf{T}_j \, dL \quad ; \quad i, j \in (\alpha, \beta)$$

The $\mathbf{K}_d$ matrix can be constructed from the four $\mathbf{K}_{d_{ij}}^e$ matrices. The assembly into the global stiffness matrix should follow the numbering of the degrees of freedom in elements α and β.

4.3 PROCEDURE

The procedure of the proposed surface reconstruction method can be broadly divided into three stages: boundary and characteristic curves reconstruction, topology generation and energy minimization. Listed below and depicted in figure 4.8 is an overview of the entire method. More details of the procedure will be discussed in following sections.

Stage 1: Boundary and characteristic curves reconstruction.
Stage 2: Topology generation.
- Step 1: Create the parametric domain polygon.
- Step 2: Triangulate the parametric domain.

Stage 3: Creation of the preliminary surface.
- Step 1: Create a minimal-energy surface constrained by the boundary curves.
- Step 2: If characteristic curves exist, create the mesh constraints of the parametric domain from the characteristic curves. Otherwise, go to Stage 4.
- Step 3: Triangulate the domain subject to mesh constraints.
- Step 4: Create a new minimal-energy surface constrained by both the boundary curves and the characteristic curves.

Stage 4: Begin the energy minimization until some convergence criterion is satisfied.

4.3.1 Input Data

The data points to be fitted are divided into boundary points, characteristic points and interior points. The boundary points represent the surface's boundaries which enclose the surface. The characteristic points represent characteristic curves lying on the surface. Here the characteristic curves are defined as curves lying on the surface other than the boundary curves. The styling lines common on car hoods are a good example. The interior points indicate some positions in space. They provide the information about the surface's interior shape. Besides indicating boundaries of the surface, boundary points are also used to create the mapping domain. The interior points can be totally unorganized, while the boundary points and characteristic points must be identified by the user. Both the boundary points and characteristic points are divided into several *point groups* each of which represents a single boundary curve or a characteristic curve. For each group of points, the end points must be specified by the user; the others can remain unorganized. The data structure used for the input data can be described as

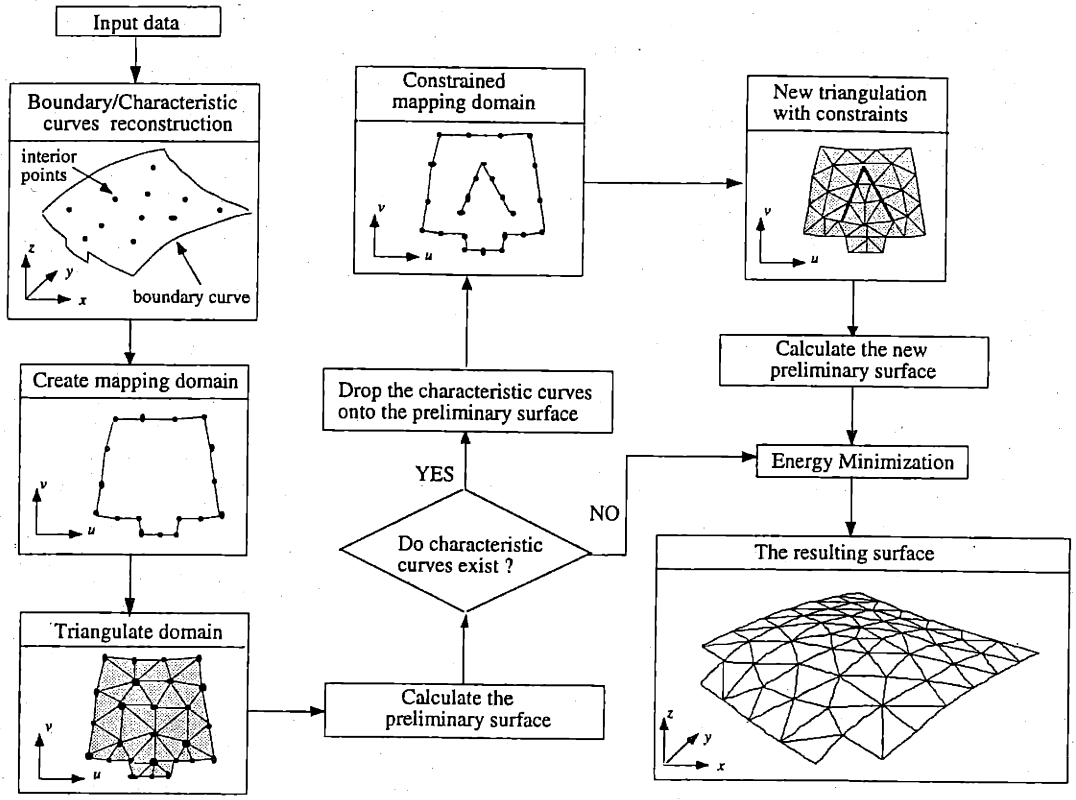


Figure 4.8: Flowchart of the surface reconstruction

SurfaceReconstructionData {

- *BoundaryPointGroups*
- *CharacteristicPointGroups*
- Scattered interior points
- Other parameters such as weighting factors, convergence tolerance, etc.

}

Each *BoundaryPointGroup* or *CharacteristicPointGroup* is a point group described by the following structure

PointGroupData {

- Scattered data points
- two end points
- Other parameters such as weighting factors, convergence tolerance, etc.

}

4.3.2 Boundary and Characteristic Curves Reconstruction

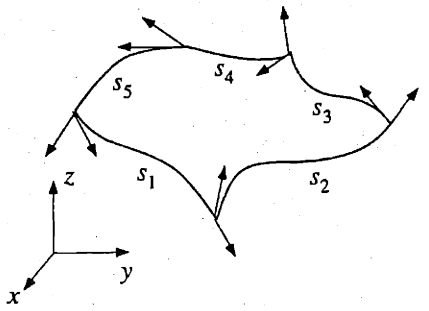
The boundary curves indicated by the boundary points and the characteristic curves indicated by characteristic points are reconstructed by parametric curves using the scheme described in Chapter 3. Then the boundary curves are used to create the mapping domain which will be discussed in the following section. The reconstructed characteristic curves are used to create the constraints on the mapping domain.

4.3.3 Topology Generation

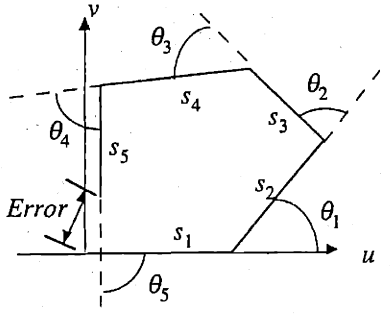
Conventionally, a unit square in the parametric plane is used to define a surface. However, for defining an n-sided surface, not only the range of the domain needs to be decided but also the shape of the domain. Using the x-y plane projection of the data points as the mapping domain is the simplest way but is only suitable for an explicitly-described surface and causes the algorithm to be coordinate system dependent. When the surface is not explicitly-described and even a projection to another plane will not always work, a new method for defining the parametric domain is necessary. This section will discuss a new method for creating the parametric domain. This method consists of two steps: create the parametric domain polygon, namely, the boundary of the parametric domain, and triangulate the domain.

4.3.3.1 Creation of the Parametric Domain Polygon

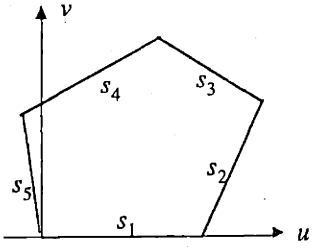
The creation of the parametric domain polygon requires the surface's boundary curves as input. These boundary curves form a closed loop in space as shown in figure 4.9a. At each intersection of two curves, two tangent vectors exist each of which belongs to one of the intersecting boundary curves. The angle of the two tangent vectors is called the "turning angle". Once the values of these turning angles and the arc length of each boundary curve are calculated, a polygon can be created on the parametric plane by arbitrarily selecting the origin as the starting point, the u direction as the first direction, and then starting to "march" as required by the arc lengths and the turning angles. This "marching path" might not form a closed polygon as shown in figure 4.9b. The error vector is a function of arc lengths, s_i and turning angles, θ_i. By fixing s_i and varying θ_i the error vector can be forced to be zero. In general, we prefer the summation of the relative deviations of θ_i to be minimum. That means we need to minimize relative deviation with equality constraints stated as



(a) The curve network formed by the boundary curves. The arrows indicate the tangent vectors at the two ends of each curve.



(b) The unclosed polygon with edges lengths equivalent to the arc lengths of the boundary curves.



(c) The polygon is forced to be closed by minimizing the sum of relative deviations of the turning angles with constraints.

Figure 4.9: Topology generation from existing boundaries

$$\begin{aligned} &\text{Minimize} \quad F(\theta_i) = \sum_i \frac{\left|\theta_i - \theta_i^{(0)}\right|}{\theta_i^{(0)}} \;;\; \theta_i^{(0)} \text{ are the original turning angles} \\ &\text{Subject to} \quad \text{Error}_x(\theta_i) = 0.0 \\ &\qquad\qquad \text{Error}_y(\theta_i) = 0.0 \\ &\qquad\qquad \sum_i \theta_i = 2\pi \end{aligned} \tag{4.3.1}$$

The first two constraints are to force the polygon to be closed. The last constraint is to prevent the polygon from intersecting itself. The method of multipliers (MOM) and the conjugate gradient method are used to find the solution.

4.3.3.2 Domain Triangulation

After defining the u-v plane, we have a set of 2D points which form a closed polygon on the parametric plane. An automatic mesh algorithm is needed to create the triangular mesh on which the surface is defined. This algorithm must be able to make dense meshes where the surface is highly curved and sparse meshes where the surface is relatively flat, i. e., a mesh algorithm with the capability of controlling the mesh density is needed.

Recently, a new physically-based mesh technique has been developed. The method was inspired by observing the behavior of soap bubbles floating in liquid. A computational model reproducing the physics of bubbles includes a prescribed density function controlling the size of bubbles, mass and viscous damping associated with each bubble and a modified Van der Waals force field between two bubbles. Fixed bubbles are placed on the boundary and the characteristic lines first, then free bubbles are injected into the interior domain. Soap bubbles attract and repel each other until achieving an equilibrium configuration, then, the mesh is completed by connecting the center of each bubble with Delaunay triangulation. Note that the Delaunay triangulation used here has been modified so that it is able to triangulate an arbitrary nonself-intersecting polygonal domain specified by a series of 2D planar vertices. This modified Delaunay triangulation is also capable of performing the triangulation subject to some prespecified mesh constraints. Figure 4.10 is an illustration for this soap bubble mesh method. Details of the soap bubble mesh method can be found in [Shimada 93].

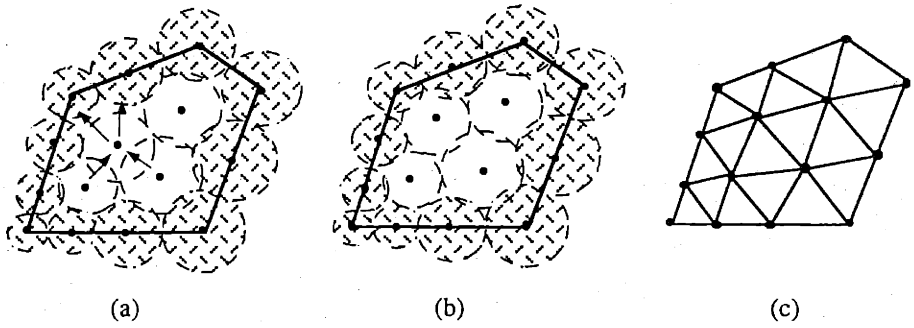


Figure 4.10: The soap-bubble mesh generation scheme: (a)Bubbles are repelled and attracted by each other, (b) Bubbles achieved the equilibrium, (c) Bubbles' centers are connected by Delaunay triangulation.

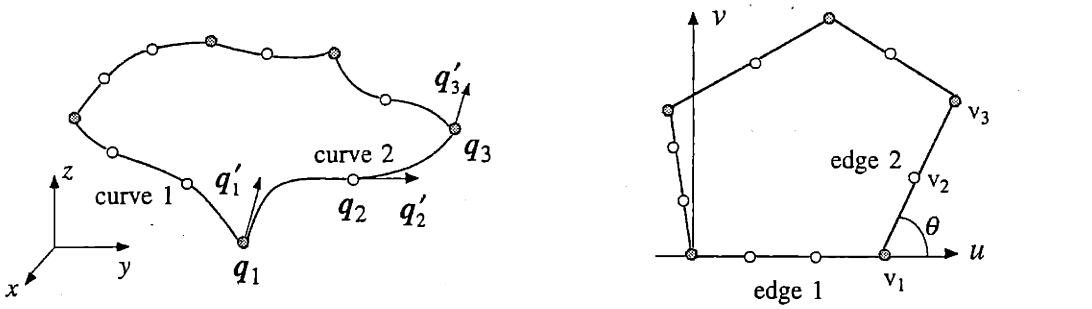
In the bubble mesh method, a density function is used to control mesh size so that the triangular elements are smaller in high density regions. Generally, the density is related to surface curvature such as maximum curvature and mean curvature. However, the surface is not available before the mesh is created. Note that the many developed gradient estimation methods that estimate the derivatives of z in the x and y directions are only appropriate when the surface is explicitly described. One solution to this chicken-egg problem is to form an approximation surface by assuming a constant density function first, then to change the mesh according to the surface's curvature, and recalculate the surface. When the surface is deformed, its corresponding mesh is changed based on the curvature of the deformed surface, and the surface's shape is recalculated based on the new mesh. In the implementation the mesh is allowed to change according to the surface's curvature every n iterations in the minimization. The value of n is decided by the user. From experience, the surface approaches the minimal energy shape during the first several iterations in the minimization. Hence, performing the remesh once after the first several iterations is generally sufficient.

4.3.4 Creation of the Preliminary Surface

4.3.4.1 Surface Boundary Conformity

An important requirement in creating the preliminary surface is that the surface boundary should conform to the reconstructed boundary curves. Before describing how to generate the preliminary surface, the compatibility equations which impose constraints on the surface to achieve the boundary conformity are derived.

Recall that the first step for domain triangulation is to place fixed bubbles on the edges of the domain polygon. The fixed bubble placement has a critical influence on the surface boundary conformity. Figure 4.11 shows a Hermite curve network and the corresponding parametric domain polygon created by the method described in the previous section. Each edge of the polygon is divided into the same numbers of segments as the number of curve elements of the corresponding curve. Each line segment is expected to be mapped to a space curve which is identical to the corresponding Hermite curve element. The following discussion assumes that the cubic Hermite curve and the 12-dof triangular interpolant are used as the curve and surface primitives.



(a) The Hermite curve network (b) The corresponding parametric domain polygon

Figure 4.11: The placement of vertices on the domain polygon edges. (The filled circles represent the junctions of two curves and the empty circles the junctions of the curve elements in each curve).

As illustrated in this figure, the curve of interest (curve 2) is specified by position vectors $\boldsymbol{q}_i$, slope vectors $\boldsymbol{q}_i'$ (i=1 to 3) and parametric length h_j (j=1 to 2). The corresponding edge (edge 2) of the domain polygon is also divided into three segments. The surface shape along segment $\overline{v_1v_2}$ is a cubic curve which can be represented as a cubic Hermite curve with unit parametric length. Requiring this curve to be identical to the curve specified by $\boldsymbol{q}_1$, $\boldsymbol{q}_2$, $\boldsymbol{q}_1'$, $\boldsymbol{q}_2'$ and h_1 results in four compatibility equations, one position and one slope compatibility equations for each of the two vertices of the segment:

$$\begin{aligned} \boldsymbol{w}_1 &= \boldsymbol{q}_1 \\ \boldsymbol{w}_2 &= \boldsymbol{q}_2 \\ (u_2-u_1)\boldsymbol{w}_{u1}+(v_2-v_1)\boldsymbol{w}_{v1} &= h_1\boldsymbol{q}_1' \\ (u_2-u_1)\boldsymbol{w}_{u2}+(v_2-v_1)\boldsymbol{w}_{v2} &= h_1\boldsymbol{q}_2' \end{aligned} \qquad (4.3.2)$$

where (u_1,v_1) and (u_2,v_2) are the parametric coordinates of the vertices 1 and 2 and $\boldsymbol{w}_i$, $\boldsymbol{w}_{ui}$ and $\boldsymbol{w}_{vi}$ are the position and derivative vectors at vertex i. Similar compatibility equations can be obtained for other segments. Generally speaking, two position and two slope compatibility equations will be generated for each vertex. The two position compatibility equations are identical. However, the two slope compatibility equations are different and could pose some problems. For example, the two slope compatibility equations for vertex 2 are

$$(u_2-u_1)\boldsymbol{w}_{u2}+(v_2-v_1)\boldsymbol{w}_{v2} = h_1\boldsymbol{q}_2' \qquad (4.3.3a)$$

$$(u_3-u_2)\boldsymbol{w}_{u2}+(v_3-v_2)\boldsymbol{w}_{v2} = h_2\boldsymbol{q}_2' \qquad (4.3.3b)$$

Dividing equation (4.3.3a) by the length of segment $\overline{v_1v_2}$ and equation (4.3.3b) by the length of segment $\overline{v_2v_3}$ results in

$$\cos\theta\, w_{u2} + \sin\theta\, w_{v2} = (h_1/\ell_1)q_2' \tag{4.3.4a}$$

$$\cos\theta\, w_{u2} + \sin\theta\, w_{v2} = (h_2/\ell_2)q_2' \tag{4.3.4b}$$

where $\ell_1 = ((u_2-u_1)^2 + (v_2-v_1)^2)^{1/2}$, $\ell_2 = ((u_3-u_2)^2 + (v_3-v_2)^2)^{1/2}$ and θ is the angle between the line segment and the u axis as illustrated in figure 4.11.

To constrain the surface boundary to conform exactly to the reconstructed curves, all of the compatibility equations must be enforced exactly in the minimization. However, it is clear that the two slope compatibility equations at each vertex cannot be satisfied simultaneously unless the equation

$$\frac{h_1}{\ell_1} = \frac{h_2}{\ell_2} \tag{4.3.5}$$

is satisfied. This condition applies to every vertex on the edges of the domain polygon except those that are at the corners. This also means that the placement of the subdividing nodes (the empty circles) on the polygon's edges should be taken according to the parametric lengths of the curve elements of the corresponding curve.

When a higher triangulation density is desired, more vertices can be placed into each segment. The placement of these vertices should also follow the rule described by (4.3.5).

4.3.4.2 Preliminary Surface Generation

Once the parametric domain has been created and the constraints properly set, a preliminary surface is created by minimizing the irregularity measurement subject to the geometric constraints which are required to ensure the surface boundary to match the existing boundary curves. Since the functional composed of the irregularity measurement only is quadratic and all the constraints are linear, the minimization can be done by imposing the linear constraints by the reduced transformation method and then solving a reduced set of linear equations.

In addition to the boundary curves, the preliminary surface is also required to pass through the characteristic curves if they exist. Several steps are taken to accomplish this task. First, the two ends of the characteristic curve are "dropped" onto the preliminary surface. The word "drop" represents a procedure to project the point onto the surface in such a way that the line connecting the point and its projected point aligns with the normal vector at the projected point. Second, subdividing nodes are placed on the line segment defined by the corresponding parametric locations of the two projected points. This line segment serves as a constraint of the subsequent Delaunay triangulation since the mesh lines are required to align with it. Then free bubbles are injected and the modified Delaunay triangulation is performed. Finally the proper constraint equations are set and a minimal-energy surface constrained by both the boundary curves and characteristic curves is created. This surface will be used as the initial surface shape in the energy minimization.

4.3.5 The Energy Minimization

Once the parametric domain is defined and the preliminary surface is created, the optimal shape is found by minimizing the energy functional described in equation (4.1.2) by successive quadratic programming. This minimization is a nested minimization problem in which minimization subproblems are involved in each iteration. The minimization subproblems include computation of the minimum distance between a point and the parametric surface and the optimization of the node configuration when remesh is required.

4.4 SURFACE QUALITY ANALYSIS

The purpose of surface quality analysis is to analyze both the accuracy of the fit and the quality of the surface that was reconstructed in a quantitative or a qualitative manner. This section discusses indicators used to evaluate the accuracy of fit and interrogation schemes used to judge the quality of the surface. In [Alourdas 89] and [Hottel 91], surface interrogation schemes are categorized by the order of the information required to be calculated. Same categorization is used in this section.

4.4.1 Accuracy of Fit

Error measurements similar to those defined for parametric curves can be defined for parametric surfaces. The pointwise error and global error measurement between two parametric surfaces can be defined as

$$e(S(u_0,v_0)) = \min\{\|S(u_0,v_0) - P\| \mid P \in T(s,t),(s,t) \in \Omega_T\} \tag{4.4.1}$$

$$e_{max} = \max\{e(u,v) \mid (u,v) \in \Omega_S\} \tag{4.4.2}$$

where $T(s,t)$ is the target surface and $S(u,v)$, the reconstructed surface. Ω_T and Ω_S are the parametric domains of $T(s,t)$ and $S(u,v)$ respectively.

The evaluation of the pointwise errors between two parametric surfaces requires the solution of a two-variable constrained nonlinear minimization. For evaluating the supmetric, a simplification is made by sampling points on either the reconstructed surface or the target surface, calculating the pointwise errors with respect to one another and selecting the maximum. The computation of minimum distance from a given point to a parametric surface will be described in Chapter 5. Different global error measurements could be obtained for sampling points from the reconstructed surface and from the target surface. However, the difference will be negligible provided the two surfaces are fairly close. A better definition can be made by using the larger one as the global error measurement.

4.4.2 Zero Order Interrogation Schemes

4.4.2.1 Wireframe

The wireframe of a surface is produced by displaying its boundary curves and a number of curves lying on the surface. For quadrilateral surfaces such as tensor-product surfaces, isoparametric curves are the natural choice. For surfaces represented by connected triangular patches where shape functions are expressed in barycentric coordinates, drawing the isoparametric curves in barycentric coordinates for each triangular patch is a more direct way. A triangular grid instead of a rectangular grid will be formed. Drawing the isoparametric curves in the u-v coordinates requires more effort in calculating the intersecting points of the constant u and v lines with the polygon enclosing the parametric domain, and determining which element the point of interest falls. The wireframe of a surface only provides a rough idea of the underlying shape. Judging the surface quality merely from its wireframe made of isoparametric curves might mislead the designer. For example, a simple plane can be parametrized in such a way that its isoparametric curves are not straight lines in the space, which will appear to represent a curvy surface by human visual system's interpretation. Furthermore, when surface is highly curved, the surface's real shape is often obscured by the overlapping curves. Though the hidden line removal schemes might be helpful in this matter, the wireframe picture is still not an appropriate means to judge the natural of the surface and its fairness.

4.4.2.2 Contouring

A better indication of the surface shape is normally obtained using contour curves. There are different types of contour curves. The simplest contouring method is to intersect the surface with a family of user-defined parallel planes. More complex contouring methods involve intersections with a series of coaxial cylinders or cones of different radii. Propeller and turbine blades are normally contoured using such methods. The calculation of the contour curves is actually a surface-surface intersection problem between the surface under interrogation and the flat planes. In [Hoschek 93], techniques used for calculating the intersections of surfaces are categorized into algebraic methods, subdivision methods, embedding methods, discretization methods and tracing methods. In this thesis, the grid method, one of the discretization methods, is used to ease implementation.

4.4.3 First Order Interrogation Schemes

4.4.3.1 Shading

A shaded image of a surface usually gives a more realistic visualization of the surface in comparison to a wireframe model. Different shading techniques exist. The two most popular shading models are the Gouraud shading model and the Phong shading model. The Gouraud shading model evaluates the normal vectors at the polygon's vertices by averaging the normal vectors of the facets sharing that vertex. The intensity at the vertex is obtained by applying the normal thus calculated to an intensity equation. The interior pixel's intensity is then calculated by bilinear interpolation from the intensities at the polygon's vertices. The Gouraud shading model is computationally less expensive than the Phong shading model but the latter is able to overcome some of the disadvantages of Gouraud shading such as smoothing out the corrugations. In the Phong shading model, the attributes being interpolated are the vertex normals, rather than vertex intensities. Vertex normals are calculated by averaging the normal vectors of the surfaces that share the vertex. The intensity of each pixel is then calculated from the interpolated normal. More complicated shading models which take into account the material's properties, the incidence angle and the wavelength of the incidence light also exist. Readers are directed to [Watt 89].

4.4.3.2 Isophotes

Isophotes are curves of constant light intensity on a surface with respect to a light source with direction $\boldsymbol{L}$ [Poeschl 84]. These curves can be used to detect the surface's irregularities, especially, the continuity conditions between surface's patches. Mathematically, the isophotes are defined as the collection of points which satisfy

$$N(u,v)\cdot \boldsymbol{L} = c \quad , 0 \le c \le 1 \tag{4.4.3}$$

where $N(u,v)$ is the unit normal vector of the surface $S(u,v)$. Silhouettes are special isophotes , c=0 , with respect to the light source.

If the surface is C^M continuous, then the isophotes will be C^{M-1} continuous curves. Since visualizing low degree discontinuities is always easier than high degree discontinuities, the surface's discontinuity in higher degrees can be detected by

visualizing the continuity of its isophotes. In [Hagen 92], isophotes are used to detect the continuity of three different surfaces which appear to be identical even in the shaded images but which are actually C^0, C^1 and C^2 continuous respectively.

4.4.3.3 Reflection Lines

Two reflection lines have been defined. In [Klass 80], the reflection line is defined as the reflected image of a straight line, the so-called light line

$$\boldsymbol{L}(t) = \boldsymbol{L}_0 + t\,\boldsymbol{s},\ t \in \mathrm{R} \tag{4.4.4}$$

on a surface $\boldsymbol{S}(u,v)$, perceived from a fixed point $\boldsymbol{A}$, the point of vision. Points on such a reflection line are called reflection points.

As illustrated in figure 4.12a, $\boldsymbol{P}$ represents the reflection point on the surface of the light point $\boldsymbol{L}$ with respect to the view point $\boldsymbol{A}$. Their relation can be formulated from the geometric dependency as

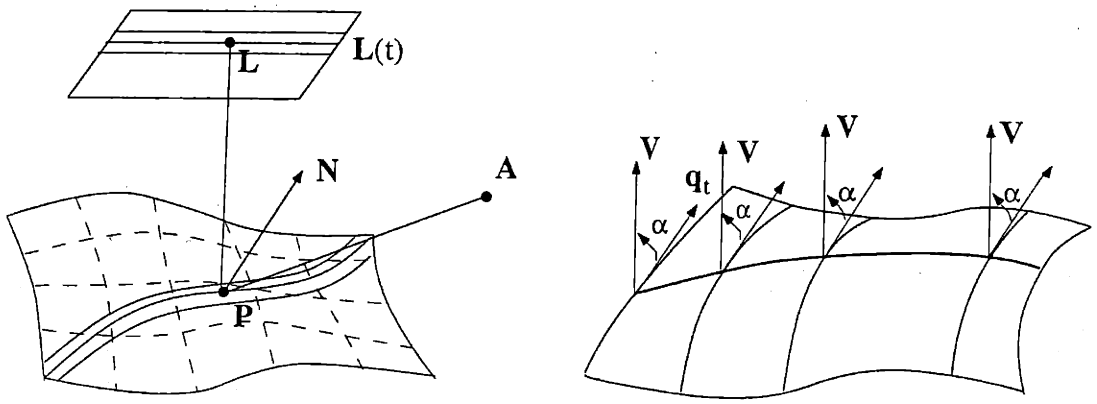
$$\frac{\boldsymbol{a}}{\|\boldsymbol{a}\|} + \frac{\boldsymbol{b}}{\|\boldsymbol{b}\|} = 2\boldsymbol{N}(\boldsymbol{N} \cdot \frac{\boldsymbol{b}}{\|\boldsymbol{b}\|}) = 2\boldsymbol{N}(\boldsymbol{N} \cdot \frac{\boldsymbol{a}}{\|\boldsymbol{a}\|}) \tag{4.4.5}$$

where $\boldsymbol{a} = \boldsymbol{PA} = \boldsymbol{A} - \boldsymbol{P}$ and $\boldsymbol{b} = \boldsymbol{PL} = \boldsymbol{L} - \boldsymbol{P}$. If we move the light point $\boldsymbol{L}$ along the light line, the trajectory of reflection point $\boldsymbol{P}$ will be a curve on the surface, the reflection line. If a family of parallel light lines are selected, which form the light plane, we can obtain a family of reflection lines on the surface, which can be used to judge the surface's quality.

Given a light point, a fixed view point and a surface, the reflection point can be obtained by solving the nonlinear system of equations for the unknown parametric values

$$\boldsymbol{b} + \lambda \boldsymbol{a} = 2\boldsymbol{N}(\boldsymbol{N} \cdot \boldsymbol{b}) \quad \text{with } \lambda = \frac{\|\boldsymbol{b}\|}{\|\boldsymbol{a}\|} \tag{4.4.6}$$

These three nonlinear equations can be reduced to two by eliminating λ, but the existence and unambiguity of solutions must be ensured by an appropriate choice of the view point $\boldsymbol{A}$.



(a) The reflection lines proposed in [Klass 80] (adapted from [Hagen 92])

(b) The reflection lines proposed in [Kaufmann 88]

Figure 4.12: Two kinds of reflection lines

Another type of reflection line is proposed in [Kaufmann 88]. This kind of reflection line is defined using a family of curves $q(t)$ on the surface. They can be either the isoparametric curves or the intersection curves of the surface with a family of parallel planes. For this family of curves, a reflection line is defined by connecting the point on each curve, on which the curve's tangent vector forms an angle α with a given vector V as illustrated in figure 4.12b. This can be expressed as

$$\frac{q_t}{\|q_t\|} \cdot \frac{V}{\|V\|} = \cos\alpha = c \quad ; \quad 0 \le c \le 1 \tag{4.4.7}$$

For every value of the angle α, a reflection line is obtained.

The Kaufmann-type reflection line does not have significant physical meaning as the Klass-type reflection line previously introduced. However, it is easier to be implemented and costs less computation time. In [Alourdas 89] and [Hottel 91], the Kaufmann-type reflection line was used as a tool for interrogating surfaces fairness. In these two work, the family of curves are the isoparametric curves of the surface being interrogated and the curve's tangent vectors are simply the surface's derivatives in one of the parametric directions. However, from the author's experiences, surface interrogation schemes should not depend on surface parametrization because doing so will cause different interrogation results for geometrically identical surfaces with different parametrizations. In this

regards, the intersection curves of the surface with a family of parallel planes is a better choice for generating the Kaufmann-type reflection lines.

4.4.4 Second Order Interrogation Schemes

The second order interrogation of a parametric surface involves using the surface's curvatures to detect the irregularities such as unwanted inflection points. All the different curvatures of a parametric surface, such as Gaussian curvature, mean curvature, etc., can be defined by the surface's first and second fundamental coefficients. The formulas used to calculate a variety of surface's curvatures are provided in the appendix. Variations of the surface curvature can be displayed as a contour map of constant curvatures or a coded-color map [Dill 81] [Beck 86] [Seidenberg 92].

A surface inflection exists on a surface at a point $\boldsymbol{P}$ if the surface crosses the tangent plane at $\boldsymbol{P}$ which is then called an inflection point. The curvature maps can give us the information about the inflections of the surface. If the Gaussian curvature is positive at every point of a region of a surface, then there are no inflections in that region. If the Gaussian curvature changes signs in a region then there is an inflection in the region. If the surface is developable, i.e., Gaussian curvature is everywhere zero, or the surface's Gaussian curvature is very close to zero, then Gaussian curvature alone cannot provide sufficient information about the surface shape. In this case, the mean curvature is also needed. If the Gaussian curvature is zero in a region then the surface has an inflection point in that region only if the mean curvature changes signs. Generally, the Gaussian curvature and the mean curvature will provide sufficient information to identify the surface's inflections.

Other second order interrogation tools including curvature lines and geodesics will not be discussed here. Readers are referred to [Beck 86] [Munchmeyer 87] [Alourdas 89] and [Hottel 91].

4.5 SUMMARY

This chapter described a new method for generating a smooth parametric surface to approximate a set of error-filled, unordered data points. The surface is not limited to quadrilateral surfaces as in traditional methods. Users only need to specify each group of points forming the surface's boundary curves and the two end points in each group. An energy functional composed of a global and local irregularity measurement and a closeness measurement between data points and the surface is given. Surfaces are represented by finite-element triangular patches. The successive quadratic programming technique is applied to find the solution. The resulting surface generally has good fairness even when only a C^1 surface is implemented. In comparison to existing parametric surface interpolation methods, this method uses more general surface representation and requires less connectivity information. The interpolation problem may also be treated as a special case of our method in which the data points to be interpolated are considered as constraints in the minimization process.

CHAPTER FIVE

Implementations

In this chapter, we discuss several technical issues encountered in the development of curve and surface reconstruction methods and give a brief analysis about the computational complexity of the proposed methods. The technical issues include: (1) constrained nonlinear minimization, (2) minimum distance computation between a point and a parametric curve or a surface, (3) enforcement of geometric constraints, and (4) reduction of system matrix's bandwidth in solving a set of linear equations.

The rest of this chapter is organized as follows. In Section 5.1, we introduce the minimization methods used in this work for solving the many problems, formulated as constrained nonlinear minimizations, arising in the development of curve and surface reconstruction methods. Section 5.2 discusses the minimum distance computation problem. In particular, a roots-finding method utilizing the conversion property between power basis and Bernstein basis is introduced. Section 5.3 introduces the reduced matrix transformation technique which enforces linear constraints in a quadratic programming problem. Section 5.4 introduces the basic concept in bandwidth reduction algorithms which have drastic impacts on the computing time for solving a set of linear equations. Section 5.5 gives a brief analysis regarding the computational complexity of the proposed curve and surface reconstruction methods.

5.1 CONSTRAINED NONLINEAR MINIMIZATION

The proposed curve and surface reconstruction method treat the solutions as ones that minimize certain energy functionals. The computation of the minimum distance between a point and a parametric curve or a surface and the creation of the parametric domain's boundary are also formulated as constrained nonlinear minimizations. This section attempts to give a brief description of the minimization methods used in this work. Since optimization methods are not our main research topic but numerical tools used to find the solution, only introductory material is covered. Details of these methods can be found in most optimization text books such as [Reklaitis 83] and [Bazaraa 93]

5.1.1 Successive Quadratic Programming

For a general constrained nonlinear minimization problem of size n, stated as

$$\begin{aligned} &\text{Minimize} && f(\boldsymbol{x}) && , \boldsymbol{x}=[x_1,x_2,\cdots,x_n]^{\mathrm{T}} \\ &\text{Subject to} && g_j(\boldsymbol{x})\geq 0.0 && ;\ j=1,\cdots,J \\ & && h_k(\boldsymbol{x})=0.0 && ;\ k=1,\cdots,K \\ & && x_i^{(l)}\leq x_i\leq x_i^{(u)} \end{aligned} \tag{5.1.1}$$

successive quadratic programming (SQP) replaces the original NLP problem by a quadratic subproblem which consists of the quadratic approximation of the objective function and linear approximation of the constraints at the solution estimate. This subproblem is then solved by typical quadratic programming methods. This replacement is repeated and the solution estimate is improved during iterations until a certain convergence criterion is achieved. The advantage of using such quadratic approximations is that the subproblem with the quadratic objective function and linear constraints can generally be solved in a finite number of iterations. A typical solution strategy using SQP would consist of the following steps:

Step 1. Formulate the quadratic subproblem at the solution estimate $x^{(t)}$:

$$\begin{aligned} &\text{Minimize} && \nabla f(\boldsymbol{x}^{(t)})^{\mathrm{T}}\boldsymbol{d}+\tfrac{1}{2}\boldsymbol{d}^{\mathrm{T}}\nabla^2 f(\boldsymbol{x}^{(t)})\boldsymbol{d} \\ &\text{Subject to} && g_j(\boldsymbol{x}^{(t)})+\nabla g_j(\boldsymbol{x}^{(t)})^{\mathrm{T}}\boldsymbol{d}\geq 0.0 && ;\ j=1,\cdots,J \\ & && h_k(\boldsymbol{x}^{(t)})+\nabla h_k(\boldsymbol{x}^{(t)})^{\mathrm{T}}\boldsymbol{d}=0.0 && ;\ k=1,\cdots,K \end{aligned} \tag{5.1.2}$$

Step 2. Solve the quadratic subproblem, and set $\boldsymbol{x}^{(t+1)} = \boldsymbol{x}^{(t)} + \boldsymbol{d}$.
Step 3. Check for convergence. If not converged, go to step 1.

The nonlinear functionals listed in equations (3.1.5) and (4.1.4) are in the same form and can be described by

$$E = E_{geometry} + E_{spring} = \mathbf{Q}^{\mathrm{T}}\mathbf{K}\mathbf{Q} + \sum_{i=0}^{N-1} \lambda_i D_i^2(\mathbf{Q}) \tag{5.1.3}$$

where $\mathbf{Q} = [q_1 \quad q_2 \quad \cdots \quad q_n]^{\mathrm{T}}$ is a column vector containing n degrees of freedom each of which is a 3 dimensional vector, $\mathbf{K}$ is the system stiffness matrix, and $D(\cdot)$ is a distance operator measuring the distance between a point and a curve or a surface as described previously. Note that the size of this minimization problem, i. e., the number of variables, is $3n$ instead of n, and some geometric equality constraints have to be enforced in the minimization. The use of SQP to minimize the functional was inspired by the fact that the first term is quadratic. The local quadratic approximation of the original problem is taken as

$$\overline{E}(\mathbf{d};\ \mathbf{Q}^{(t)}) = \nabla E(\mathbf{Q}^{(t)})^{\mathrm{T}}\mathbf{d} + \mathbf{d}^{\mathrm{T}}\mathbf{K}\mathbf{d} \tag{5.1.4}$$

The initial solution estimate is obtained by finding the solution of $\overline{E} = \mathbf{Q}^{\mathrm{T}}\mathbf{K}\mathbf{Q}$ subject to geometric constraints. An illustration of the SQP is shown in figure 5.1.

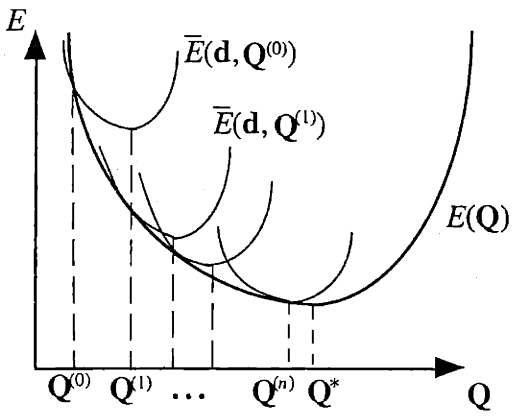


Figure 5.1: Successive quadratic programming

It should be noted that the x, y and z components in either $\mathbf{Q}^T\mathbf{K}\mathbf{Q}$ or $\mathbf{d}^T\mathbf{K}\mathbf{d}$ of the subproblem are decoupled. If the geometric constraints and the $\nabla E(\mathbf{Q}^{(n)})^T$ can be made to be x-y-z decoupled as well, the quadratic subproblem can be further separated into three items each of which is also a quadratic function in either x, y or z. Minimizing the subproblem is the same as minimizing the x, y and z quadratic functions separately. Thus, the size of the subproblem, originally $3n$, can be reduced by a factor of 3. Because minimizing a quadratic function with linear equality constraints can be achieved by solving a set of linear equations, which generally causes computing time to be $O(n^2)$, reducing the problem size by a factor of 3 is equivalent to reducing the computing time by a factor of 9. Making the geometric constraints x-y-z decoupled can be done by restricting the constraints to be linear combinations of the explicit degrees of freedom q_is. However, it is more difficult to evaluate the $\nabla E(\mathbf{Q}^{(n)})^T$ due to the presence of the error term. The error term cannot be represented as an explicit function of the degrees of freedom. It is not even continuous with respect to the degrees of freedom in the sense that the attachment point can jump in response to a small change in the degrees of freedom.

Knowledge from the applied mechanics can be used to find ∇E_{spring}. Recall that the minimization formulation is a result of applying the variational principle to the physical model. Thus, the variation of the spring's energy is equivalent to the variation of the work done by the forces acting on the geometry. This can be expressed as

$$\delta E_{spring} = \int_\Omega f \cdot \delta w ds \tag{5.1.5}$$

or

$$\nabla E_{spring}^T \delta\mathbf{Q} = \int_\Omega f \cdot \Phi^T \delta\mathbf{Q}\, ds = \mathbf{F}^T \delta\mathbf{Q} \text{ with } \mathbf{F}^T = \int_\Omega f \cdot \Phi^T ds \tag{5.1.6}$$

where the integral is taken over the domain on which the geometry is defined. ∇E_{spring} turns out to be the *generalized forces* acting in the directions of the *generalized coordinates*, the degrees of freedom which completely and independently specify the system's configuration. Though the evaluation of ∇E_{spring} still cannot be evaluated by direct differentiation with respect to q_is, it can be obtained by approximating the spring forces as distributed forces along the curve and performing the integral described by equation (5.1.6). The forms of $f(u)$ (or $f(u,v)$ for surfaces) are somewhat arbitrary as long as they provide a reasonable approximation of the spring forces acting on the geometry.

5.1.1.1 Resolving the Spring Forces

For curve reconstruction, two methods for resolving the spring forces into distributed forces are used. The first method resolves each of the concentrated spring forces into a force linearly distributed on the curve element the attachment point falls on. The second one resolves the concentrated force into a force linearly distributed on the element that the attachment point falls on and its adjacent elements. The linear distributed force can be specified by the force densities, denoted by p_a and p_b, at the two ends of the curve element that the attachment point falls on. They are illustrated in figure 5.2.

The value of p_a and p_b can be uniquely determined by requiring that the magnitude of the total force and the application point remain unchanged. This results in

$$F = \frac{(p_a + p_b)h}{2} \tag{5.1.7a}$$

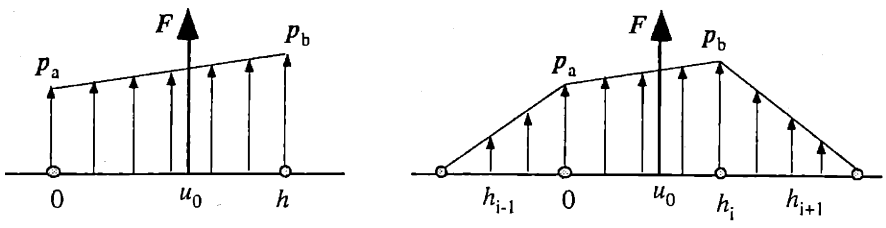
$$u_0 = \frac{(p_a + 2p_b)h}{3(p_a + p_b)} \tag{5.1.7b}$$

for the first method, and

$$F = \frac{p_a h_{i-1}}{2} + \frac{(p_a + p_b)h_i}{2} + \frac{p_b h_{i+1}}{2} \tag{5.1.8a}$$

$$Fu_0 = -\frac{p_a h_{i-1}}{2}\frac{h_{i-1}}{3} + \frac{(p_a + p_b)h_i}{2}\bar{u} + \frac{p_b h_{i+1}}{2}\left(\frac{h_{i+1}}{3} + h_i\right) \tag{5.1.8b}$$

with $\bar{u} = \frac{1}{3}h_i(p_a + 2p_b)/(p_a + p_b)$ for the second method.



(a) (b)

Figure 5.2: Two different ways for resolving the spring forces acting on a deformable curve.

After some algebra manipulations, the solutions for p_a and p_b can be calculated as

the first method:

$$p_a = \frac{F(4-6u)}{h} \quad ; \quad p_b = \frac{F(6u-2)}{h} \tag{5.1.9a}$$

the second method:

$$p_a = \frac{F(2h_{i+1}+4h_i-6u_0)}{(h_{i-1}+h_i)(h_{i-1}+h_i+h_{i+1})} \quad ; \quad p_b = \frac{F(6u_0+2h_{i-1}-2h_i)}{(h_i+h_{i+1})(h_{i-1}+h_i+h_{i+1})} \tag{5.1.9b}$$

Once all the springs forces are resolved into linear distributed forces, the matrix **F** can be obtained by performing the integration element by element and assembling them together as for the system stiffness matrix. The element force is

$$\mathbf{F}_i^e = \int_0^h f_i^e(u)\Psi(u)du = \int_0^h \left(p_a + \frac{u}{h}(p_b - p_a)\right)\Psi(u)du \tag{5.1.10}$$

where $\Psi(u)$ is same as that used in table 3.2. The results for cubic and quintic Hermite elements are shown in table 5.1.

Both these two methods have been implemented and the results show that the second method results in smoother curvature distributions. This is because the spring forces resolved using the second method have a smoother distributed force than the first method.

Table 5.1: The forces matrices for cubic and quintic Hermite element

Cubic element	Quintic element
$\mathbf{F}_i^e = \frac{h}{60}\begin{bmatrix} 21 & 9 \\ 3h & 2h \\ 9 & 21 \\ -2h & -3h \end{bmatrix}\begin{bmatrix} p_a \\ p_b \end{bmatrix}$	$\mathbf{F}_i^e = \frac{h}{840}\begin{bmatrix} 300 & 120 \\ 52h & 32h \\ 4h^3 & 3h^2 \\ 120 & 300 \\ -32h & -52h \\ 3h^2 & 4h^3 \end{bmatrix}\begin{bmatrix} p_a \\ p_b \end{bmatrix}$

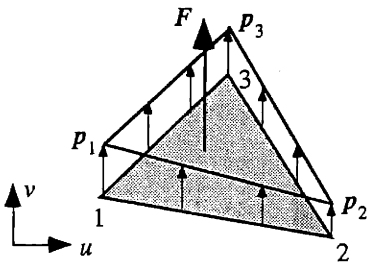


Figure 5.3: The equivalent linear distributed force for surface reconstruction

For surface reconstruction, each of the spring forces are resolved into a linear distributed force over the surface element that the attachment point falls on. This linear distributed force is characterized by the force densities, denoted by p_1, p_2 and p_3, at the triangle's vertices as illustrated in figure 5.3.

Now, there are three unknowns but only two equations (the equivalent magnitude and the same application point requirements) can be found. It is obvious that infinite solutions for p_1, p_2 and p_3 exist. Under such a condition, we simply use the following equations

$$p_1 = \frac{3F}{\Delta}\bar{L}_1 \quad ; \quad p_2 = \frac{3F}{\Delta}\bar{L}_2 \quad ; \quad p_3 = \frac{3F}{\Delta}\bar{L}_3 \tag{5.1.11}$$

where, $(\bar{L}_1, \bar{L}_2, \bar{L}_3)$ is the barycentric coordinate of the application point of the force F. The linear distributed force calculated by these equations satisfies the equivalent magnitude requirement and also ensures that the distributed force is always in the same direction of the spring force.

The calculation of the force matrix **F** for the surface reconstruction is carried in a way similar to that described above. However, due to the more complicated shape functions of the surface, the integration has to rely on numerical techniques such as Gauss-Lengendre integration method.

It is easier to gain insight about the nonlinear minimization used for curve and surface reconstruction by considering their physical analogies. Using the SQP technique in the minimization can be viewed as a simulation of the deforming process of a constrained elastic beam or thin plate under the application of spring forces. Starting with an initial shape which has a large energy functional value, in each iteration, springs anchored on

the fixed data points apply restoring forces on the geometry, and the geometry deforms to a more appropriate shape with a smaller energy functional value. The iteration stops when the system's potential energy achieves a stable equilibrium state, i.e., at the minimum.

5.1.2 Method of Multiplier

A standard approach in solving constrained minimization problems is to transform the constrained minimization problem into an unconstrained one by using penalty functions. By changing the penalty parameters from iteration to iteration, the solution estimate will be forced to converge to the minimum which satisfies the constraints. The most commonly used penalty function is the exterior penalty function which takes the following form

$$P(x)=f(x)+R\sum_{j=1}^{J}\langle g_j(x)\rangle^2+R\sum_{k=1}^{K}h_k^2(x) \tag{5.1.12}$$

However, the change in penalty parameters forces the transformed subproblems to be progressively ill-conditioned, which limits the utility of the method for practical applications. Method of multiplier (MOM) is a method devised to remedy this flaw.

For a general constrained nonlinear minimization involving the objective function $f(x)$, inequality constraints $g_i(x)\geq 0,\ i=1,\cdots,J$ and equality constraints $h_i(x)=0$, $i=1,\cdots,K$, the MOM uses a penalty function taking the following form

$$P(x,\sigma,\tau)=f(x)+R\sum_{j=1}^{J}\left\{\langle g_j(x)+\sigma_j\rangle^2-\sigma_j^2\right\}+R\sum_{k=1}^{K}\left\{(h_k(x)+\tau_k)^2-\tau_k^2\right\} \tag{5.1.13}$$

where R is a constant scale factor which may vary from constraint to constraint but remains constant from iteration to iteration. The bracket operator is defined as

$$\langle a\rangle=\begin{cases}a & \text{if } a\leq 0\\ 0 & \text{if } a>0\end{cases} \tag{5.1.14}$$

The parameters σ_j and τ_k are used to bias the penalty terms in a way that forces convergence of the iterations under rather mild conditions on the function. The σ_j and τ_k parameters are updated in each iteration according to the following rules

$$\sigma_j^{(t+1)} = \left\langle g_j(x^{(t)}) + \sigma_j^{(t+1)} \right\rangle \qquad j = 1, \cdots, J$$
$$\tau_k^{(t+1)} = h_k(x^{(t)}) + \tau_k^{(t)} \qquad k = 1, \cdots, K \tag{5.1.15}$$

Because the bracket operator, σ_js are always negative, whereas the τ_ks can take either sign. These updating rules tend to change penalty term in a way that increases the penalty on violated constraints in successive stages, thus forcing the stationary points toward feasibility. The choice for $\sigma_j^{(0)}$ and $\tau_k^{(0)}$ could be arbitrary. Setting them to zero at the beginning results in a first minimization stage which is identical to the first unconstrained minimization using standard exterior penalty terms.

The Method of Multiplier in conjunction with the conjugate gradient method, a gradient-based unconstrained minimization method, is used to solve the problem arising in the creation of parametric domain's boundary, which is described in section 4.3.3.1.

5.1.3 Conjugate Gradient Method

The conjugate gradient method is a gradient-based unconstrained multi-variable nonlinear minimization method that uses only first-order information about the objective function. It was devised to overcome the difficulties in Cauchy's method (the steepest descend method) and Newton's method. Cauchy's method tends to be effective far from the minimum but becomes less so as the minimum is approached, whereas Newton's method can be unreliable far from the minimum but is very efficient as the minimum is approached. The conjugate gradient method tends to exhibit the positive characteristics of both the Cauchy and Newton methods.

Almost all the gradient based unconstrained minimization methods employ a similar procedure:

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} s(x^{(k)}) \tag{5.1.16}$$

where $x^{(k)}$ is current estimate of the solution containing n variables, $\alpha^{(k)}$ is a step-length parameter to be decided, and $s(x^{(k)}) = s^{(k)}$ is the search direction in n dimensional space. That is, looking for an adequate value of α which minimizes the objective function in a certain search direction. Methods differ in the manner in which $s^{(k)}$ is determined.

The conjugate gradient method employs the concept of conjugate directions: The optimum of a quadratic function given by $f(\boldsymbol{x}) = a + \boldsymbol{b}^T\boldsymbol{x} + \frac{1}{2}\boldsymbol{x}^T\mathbf{C}\boldsymbol{x}$ can be found by exactly N single variable line searches, one along each of the conjugate directions with respect to matrix **C**. The conjugate gradient method uses the gradient information to find the search directions so that the search directions in two consecutive searches are mutually conjugate and the gradients at the two consecutive solution estimates are mutually orthogonal. The two most important conjugate gradient methods are the Fletcher-Reeves method and the Polak-Ribiere method. Both these two methods employ similar search direction updating rules as

$$\boldsymbol{s}^{(k)} = -\nabla f(\boldsymbol{x}^{(k)}) + \gamma^{(k)}\boldsymbol{s}^{(k-1)} \quad \text{with } \boldsymbol{s}^{(0)} = -\nabla f(\boldsymbol{x}^{(0)}) \tag{5.1.17}$$

but with a subtle difference in the constant $\gamma^{(k)}$:

The Fletcher-Reeves method:
$$\gamma^{(k)} = \frac{\left\|\nabla f(\boldsymbol{x}^{(k)})\right\|^2}{\left\|\nabla f(\boldsymbol{x}^{(k-1)})\right\|^2} \tag{5.1.18a}$$

The Polak-Ribiere method:
$$\gamma^{(k)} = \frac{\left(\nabla f(\boldsymbol{x}^{(k)}) - \nabla f(\boldsymbol{x}^{(k-1)})\right)^{\mathrm{T}} \nabla f(\boldsymbol{x}^{(k)})}{\left\|\nabla f(\boldsymbol{x}^{(k-1)})\right\|^2} \tag{5.1.18b}$$

Another class of gradient-based unconstrained minimization methods are variable metric methods, also known as quasi-Newton methods. Methods in this class also utilize only the first-order information but require a storage matrix of size $n \times n$ for updating the search direction. The search direction updating formula is also more complicated than that of the conjugate gradient method. There is no significant advantage held by the variable metric method over the conjugate gradient method except that the former was developed earlier, more widely propagated, and has been integrated into many sophisticated optimization software programs. Because of the simpler search direction update rule and smaller storage requirements needed to advance the search, the conjugate gradient method is used here as the unconstrained minimization method for the problem of computing the minimum distance between a point and a parametric surface and the problem of creating the parametric domain's boundary.

5.2 THE MINIMUM DISTANCE COMPUTATION

Finding the minimum distance between two geometries is a typical problem in geometric modeling. This section only addresses the computation of the minimum distance from a point to a parametric curve or to a parametric surface. This computation is necessary when calculating the spring forces in the curve and surface reconstructions and evaluating the accuracy of fit of the reconstructed curves and surfaces. Distance computations between other geometries such as curve to curve, curve to surface, and surface to surface are not addressed here. Readers are directed to [Zhou 93] and references therein.

5.2.1 Minimum Distance Between a Point And a Curve

Mathematically, to find the minimum distance from a given point $\boldsymbol{P}$ to a parametric curve $w(u)$, $u=[a,b]$, one has to solve a single variable constrained minimization problem expressed as

$$\begin{aligned} &\text{Minimize} \quad G(u) = \|\boldsymbol{P} - \boldsymbol{w}(u)\|^2 \\ &\text{Subject to} \quad a \le u \le b \end{aligned} \tag{5.2.1}$$

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

Typically, this problem can be solved by finding the roots of $g(u)$, the derivative of $G(u)$, selecting those falling in the interval of interest and comparing the corresponding objective function values. Geometrically, this is the same as finding a vector $\boldsymbol{QP}$ from point $\boldsymbol{P}$ to the curve that is orthogonal to the tangent vector of $w(u)$ at point $\boldsymbol{Q}$. Figure 5.4 shows the vector geometry. Note that some point-curve relationships can produce solutions of $g(u)$ that cause maximum distances in the interval. Hence, all the roots of $g(u)$ in $[a,b]$ along with $u=a$ and $u=b$ should be examined to find the real minimum distance. To find the minimum distance between a given point and a piecewise continuous curve, the minimum distance between the given point and each curve element, defined between 0 and h, should be calculated first, and then compare them to obtain the global minimum.

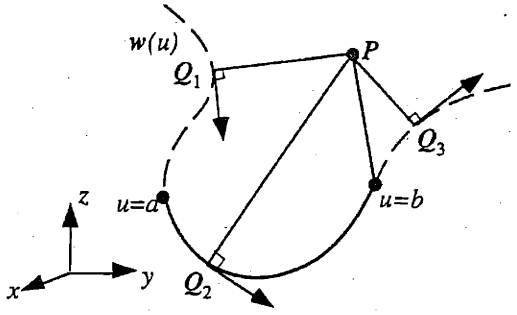


Figure 5.4 Vector geometry of the minimum distance between a given point P and a parametric curve $w(u)$. Points Q_1, Q_2 and Q_3 are all roots of $g(u)$. Q_3 results in the global minimum but the minimum distance between P and the curve segment occurs at $u = b$.

So far this problem is converted into a problem of finding roots of $g(u)$ in a certain range. However, this is still not an easy task. Problems still exist such as how many roots exist in this interval and how to bracket these roots so that other algorithms such as 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 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 points, a fast and robust root-finding algorithm is in demand.

One of the beauties of using polynomials as curve basis is that the derivative of the squared distance function in equation (5.2.1) is also a polynomial function. It is well-known that the number of real roots of a polynomial function cannot exceed the function's degree. Another advantage is that a polynomial function represented in power basis can be converted to 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 Bernstein basis of degree M using the following identity

$$t^k = \sum_{i=0}^{M} b_{i,k} B_{i,M}(t) \qquad ; b_{i,k} = \frac{C_k^i}{C_k^M} \ , \ b_{i,k} = 0 \text{ if } i < k \tag{5.2.2}$$

and then,

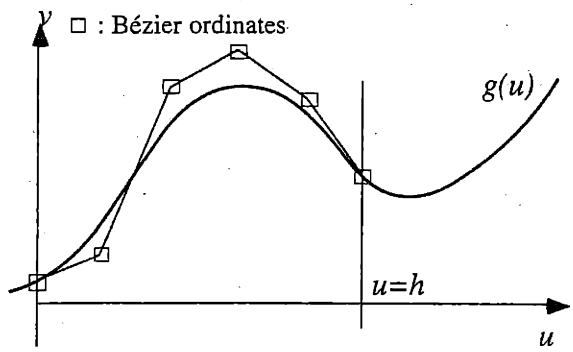
$$g(u)=\sum_{i=0}^{M}a_i u^i=\sum_{i=0}^{M}\bar{a}_i t^i=\sum_{i=0}^{M}\bar{a}_i\sum_{j=0}^{M}b_{j,i}B_{j,M}(t)=\sum_{j=0}^{M}g_j B_{j,M}(t) \tag{5.2.3}$$

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

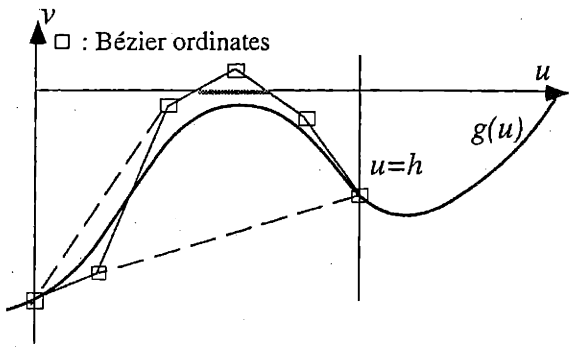
Once the original polynomial is transformed into the Bézier form, we can rewrite $g(u)$ as an explicit Bézier curve by

$$g(t)=\begin{pmatrix} t \\ g(t)\end{pmatrix}=\sum_{i=0}^{M}\begin{pmatrix} \frac{i}{M} \\ g_i\end{pmatrix}B_{i,M}(t) \tag{5.2.4}$$

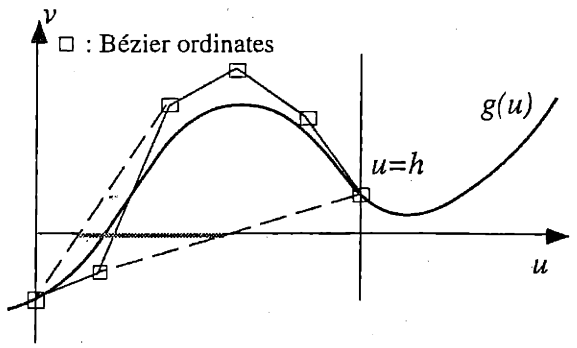
Now, finding the roots of an univariate polynomial function $g(u)$ inside the range $[0, h]$ is equivalent to the problem of finding roots of $g(t)$ inside [0, 1]. The latter is then converted into a problem of finding the intersection of the Bézier curve $g(t)$ with the parameter axis. Thanks to the 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 the Bézier ordinates is larger than zero, there is no guarantee that roots must exist or how many roots exist in $[0, h]$. In such a case, a subinterval, indicating the only region that the roots can exist, can be located by finding the intersection of the convex hull of the Bézier ordinates with the u axis and the de Casteljau algorithm can be used to subdivide the curve. This technique can be used to recursively refine the interval until the Bézier ordinates have no intersection or only one intersection with the u axis. Then either no roots exist in the subinterval or other algorithms such as the Newton-Raphson method can be used to achieve a faster convergence. In figures 5.5 subintervals are indicated by shaded lines. If more than one root exists, the refinement of the interval will not be significant. In such a case, bisecting the interval and performing same refinement scheme on two subintervals is required.



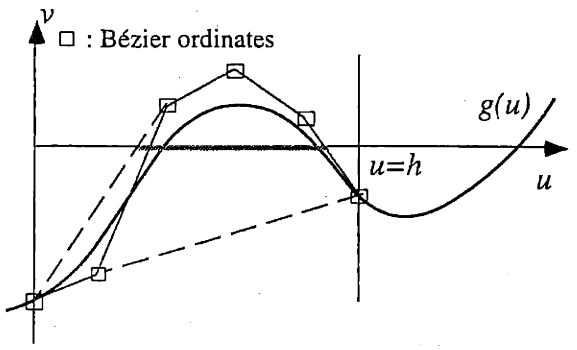
(a) The control point polygon has no intersection with u axis, which indicates that $g(u)$ has no roots in $[0, h]$



(b) The control point polygon has an intersection with u axis, which does not necessarily guarantee roots in $[0, h]$



(c) Both the control point polygon and the curve has one intersection with u axis in $[0, h]$



(d) Both the control points polygon and the curve has two intersections with u axis in $[0, h]$

Figure 5.5: Converting a polynomial from power basis to Bernstein basis

5.2.2 Minimum Distance Between a Point And a Surface

To find the minimum distance between a given point $\boldsymbol{P}$ and a piecewise continuous surface, the minimum distance between the given point and each surface element should be calculated first, and then compared them to obtain the global minimum. Hence the problem becomes a multi-variable constrained minimization problem stated as

$$\begin{aligned} &\text{Minimize} \quad G(u,v) = \|\boldsymbol{P} - \boldsymbol{w}(u,v)\|^2 \\ &\text{Subject to} \quad (u,v) \in \Omega_S \end{aligned} \tag{5.2.5}$$

where Ω_S is the region on the parametric domain over which the surface patch $\boldsymbol{w}(u,v)$ is defined. For a quadrilateral patch, Ω_S is often a unit square. For the triangular FE-based surface description used in this thesis, Ω_S is a triangle defined by three vertices on the u-v plane. Rather than stating the constraints in u-v coordinates, which will result in three linear inequality constraints, the (u, v) coordinates are converted into the area coordinates (L_1, L_2, L_3) and the problem is rewritten as

$$\begin{aligned} &\text{Minimize} \quad G(L_1,L_2,L_3) = \|\boldsymbol{P} - \boldsymbol{w}(L_1,L_2,L_3)\|^2 \\ &\text{Subject to} \quad 0 \le L_1 \le 1 \\ &\qquad\qquad\quad 0 \le L_2 \le 1 \\ &\qquad\qquad\quad 0 \le L_3 \le 1 \\ &\qquad\qquad\quad L_1 + L_2 + L_3 = 1 \end{aligned} \tag{5.2.6}$$

This problem can be further reduced to a two-variable problem by substituting the equality constraint into the object function as

$$\begin{aligned} &\text{Minimize} \quad G(L_1,L_2) = \|\boldsymbol{P} - \boldsymbol{w}(L_1,L_2,1-L_1-L_2)\|^2 \\ &\text{Subject to} \quad 0 \le L_1 \le 1 \\ &\qquad\qquad\quad 0 \le L_2 \le 1 \\ &\qquad\qquad\quad 0 \le L_1 + L_2 \le 1 \end{aligned} \tag{5.2.7}$$

in which the constraints can be treated more directly since they are simply upper and lower bounds of the barycentric coordinates.

As mentioned before, most the multi-variable gradient-based minimization methods employ the a single-variable minimization scheme on a specific direction. Due to these variable bounds, the step parameter $\alpha^{(k)}$ to be decided in each of the line search should be restricted in a certain range. Substituting each component of $\boldsymbol{x}^{(k+1)}=[L_1^{(k+1)}\ L_2^{(k+1)}]^T$ into

the three constraints in equation (5.2.7), we can obtained three sets of upper and lower bounds of $\alpha^{(k)}$, from which the real bound of $\alpha^{(k)}$ can be found by taking their intersection.

The normal way to find the minimum distance between a data point and a surface is to do the constrained multi-variable minimization for each element and then choose the global minimum from all the candidates. If we have N points and the surface is composed of M elements, then it is necessary to do the same minimization $N \times M$ times so as to be certain to find the minimum distance for all the data points. This approach would increase the computing time tremendously. Therefore we assume that during each energy minimization iteration, the surface shape will not change so much that the attachment points for interior data points only move a small distance. That means the current attachment points usually fall on the same elements as in the previous iteration. Hence, we can use the attachment point and the corresponding element from the previous iteration as the starting point for the search. If the minimum falls on an element boundary, it means the real minimum could move to the adjacent element, so we continue searching on the adjacent element until the minimum falls within the element. This strategy only requires $c \times N$ times minimization for the M interior points. c is an integer value indicating how many elements we have performed minimizing search on. From experience, the value of c is often less than 3. Thus, by using the exhaustive element-by-element search for the first iteration only and this strategy for subsequent iterations, computing time is greatly reduced.

It should be noted that the conversion from power basis to Bernstein basis can be applied to the bivariate or even multivariate distance functions also. The bounding box of the solution can be iteratively refined by the polygon projection method. However, due to the complexity of the e function used in the surface primitive, this method was not implemented. Details of this method and its various applications can be found in [Sherbrooke 93], [Maekawa 93] and [Zhou 93].

5.3 GEOMETRIC CONSTRAINTS ENFORCEMENT

General geometric constraints can be imposed by augmenting the original energy functional with Lagrange multipliers or penalty functions. However, the resulting equations often lose their linearity, complicating the problem. If we restrict the class of constraints by considering only those composed of linear combination of the degrees of freedom of the problem, the constraint imposition can be achieved easily and exactly 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.

5.3.1 Reduced Transformation Method

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

$$\begin{aligned} &\text{Minimize} \quad F = \tfrac{1}{2}\mathbf{Q}^{\mathrm{T}}\mathbf{K}\mathbf{Q} + \mathbf{F}^{\mathrm{T}}\mathbf{Q} \\ &\text{Subject to} \quad h_k(\mathbf{Q}) = 0.0 \quad ; \quad k = 1,\cdots,m \end{aligned} \tag{5.3.1}$$

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

Since each linear constrain 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:

$$\mathbf{AQ} = \mathbf{AP}^{-1}\overline{\mathbf{Q}} = [\mathbf{A}_1|\mathbf{A}_2]\left\{\begin{matrix}\mathbf{Q}_{indep} \\ \mathbf{Q}_{dep}\end{matrix}\right\} = \mathbf{B} \tag{5.3.2}$$

or

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

where, $\mathbf{P}$ is a permutation matrix that $\mathbf{PQ} = \overline{\mathbf{Q}} = \left[\mathbf{Q}_{indep}|\mathbf{Q}_{dep}\right]^{\mathrm{T}}$

$\mathbf{Q}_{indep}$ is a reduced column vector containing only $(n-m)$ independent variables.
$\mathbf{Q}_{dep}$ is a column vector of dimension m containing dependent variables.

The dependent variables can be represented by the independent variables as

$$\mathbf{Q}_{dep} = \mathbf{A}_2^{-1}\mathbf{B} - \mathbf{A}_2^{-1}\mathbf{A}_1\mathbf{Q}_{indep} = \mathbf{D}_0 + \mathbf{D}_1\mathbf{Q}_{indep} \tag{5.3.3}$$

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

$$\mathbf{Q} = \mathbf{P}^{-1}\left\{\frac{\mathbf{Q}_{indep}}{\mathbf{Q}_{dep}}\right\} = \mathbf{P}^{-1}\left(\left[\frac{\mathbf{I}}{\mathbf{D}_1}\right]\mathbf{Q}_{indep} + \left\{\frac{\mathbf{Z}}{\mathbf{D}_0}\right\}\right) = \mathbf{D}_3\mathbf{Q}_{indep} + \mathbf{D}_2 \tag{5.3.4}$$

in which $\mathbf{I}$ is an identity matrix and $\mathbf{Z}$ is a zero vector. Substituting equation (5.3.4) into the objective function in equation (5.3.1) and taking $\partial F/\partial \mathbf{Q}_{indep} = 0$, we will get a reduced set of unconstrained linear equations of number $n-m$

$$(\mathbf{D}_3^{\mathrm{T}}\mathbf{K}\mathbf{D}_3)\mathbf{Q}_{indep} = \mathbf{D}_3^{\mathrm{T}}\mathbf{F} - \mathbf{D}_3^{\mathrm{T}}\mathbf{K}\mathbf{D}_2 \tag{5.3.5}$$

which automatically enforce the linear constraints while trying to find the solution which minimizes the quadratic functional of the subproblem. The new system matrix $\mathbf{D}_3^{\mathrm{T}}\mathbf{K}\mathbf{D}_3$ retains the symmetry, positive definiteness of the original matrix $\mathbf{K}$ and usually remains banded. In practice when the constraints are only to 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.

5.3.2 An Example : Constraining a C^1 Cubic Hermite Curve

An example for constraining a C^1 cubic Hermite curve into a C^2 curve using the reduced transformation method is demonstrated in the following. For a cubic Hermite curve composed of $n+1$ nodes marked from 0 to n, the C^2 continuity at the $(i+1)$th junction of a cubic Hermite curve requires

$$w_i''(u = h_i) = w_{i+1}''(u = 0) \tag{5.3.6}$$

where $w_i(u)$ and $w_{i+1}(u)$ are the ith and $(i+1)$th curve element, and h_i is the parametric length of the ith element.

Substituting the cubic Hermite representations into above equation, we obtain

$$h_{i+1}q_i' + 2(h_{i+1}+h_i)q_{i+1}' + h_i q_{i+2}' = 3\left(h_i/h_{i+1}\right)(q_{i+2}-q_{i+1}) + 3\left(h_{i+1}/h_i\right)(q_{i+1}-q_i) \tag{5.3.7}$$

Applying the C^2 continuity condition at each junction, the resulting compatibility equations can be written in matrix form as

$$\begin{bmatrix} 2(h_1+h_0) & h_0 & 0 & \cdots & 0 \\ h_2 & 2(h_2+h_1) & h_1 & \cdots & 0 \\ 0 & \cdots & \cdots & \cdots & 0 \\ \vdots & \cdots & h_{n-2} & 2(h_{n-2}+h_{n-3}) & h_{n-3} \\ 0 & \cdots & 0 & h_{n-1} & 2(h_{n-1}+h_{n-2}) \end{bmatrix} \begin{bmatrix} q_1' \\ q_2' \\ \vdots \\ q_{n-2}' \\ q_{n-1}' \end{bmatrix} \tag{5.3.8}$$

$$= \begin{bmatrix} -h_1 & -3\frac{h_1}{h_0} & 3(\frac{h_1}{h_0}-\frac{h_0}{h_1}) & 3\frac{h_0}{h_1} & 0 & \cdots & \cdots & 0 \\ 0 & 0 & -3\frac{h_2}{h_1} & 3(\frac{h_2}{h_1}-\frac{h_1}{h_2}) & 3\frac{h_1}{h_2} & 0 & \cdots & \vdots \\ \vdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & \cdots & \cdots & 0 & -3\frac{h_{n-1}}{h_{n-2}} & 3(\frac{h_{n-1}}{h_{n-2}}-\frac{h_{n-2}}{h_{n-1}}) & -3\frac{h_{n-2}}{h_{n-1}} & -h_{n-2} \end{bmatrix} \begin{bmatrix} q_0' \\ q_0 \\ q_1 \\ \vdots \\ q_n \\ q_n' \end{bmatrix}$$

or

$$[\mathbf{H}_1]_{(n-1)\times(n-1)}\{\mathbf{Q}_{dep}\}_{(n-1)} = [\mathbf{H}_2]_{(n-1)\times(n+3)}\{\mathbf{Q}_{indep}\}_{(n+3)} \tag{5.3.9}$$

The converting matrix $\mathbf{D}_1$ can be found by

$$\mathbf{D}_1 = \mathbf{H}_1^{-1}\mathbf{H}_2 \tag{5.3.10}$$

5.4 BANDWIDTH REDUCTION

Solving a set of linear equations, $\mathbf{Ax} = \mathbf{b}$, is a frequently encountered problem in science and engineering. Various solution methods exist such as Gaussian elimination and Cholesky decomposition followed by a back substitution. If the system matrix is banded, i.e., the matrix is sparse and has nonzero elements clustering around the diagonal elements, taking advantage of the bandwidth not only reduces the storage for the matrix but also reduces computing time drastically. The typical computing time for solving a set of n linear equations by Cholesky decomposition followed by a back substitution is in the order of $O(n(n+1))$. When the matrix is banded the solution time becomes $O(2\beta n)$ where β is the bandwidth of the system. Narrow bandwidth greatly reduces the cost of computations.

Formally, the bandwidth β of a square matrix $\mathbf{A}$ is defined by $\beta = \max\{|i-j| a_{ij} \neq 0\}$, where a_{ij} are the elements of the square matrix $\mathbf{A}$. A sparse square matrix's bandwidth can often be reduced by permuting the rows and columns in a proper way. In our application, this problem arises in each iteration of the SQP method, minimizing a local quadratic approximation of the original objective function by solving a set of linear equations. The system matrix is the global stiffness matrix $\mathbf{K}$. Its bandwidth depends on how the numbering of the global degrees of freedom are assigned. Speaking more precisely, if we do not consider the discontinuity term, the bandwidth of the stiffness matrix $\mathbf{K}$ is the maximum difference of the numbering of the degrees of freedom belonging to the same element. How to reduce the bandwidth of the stiffness matrix by properly numbering the degrees of freedoms has been studied in the research of finite element analysis. In this section, only the basic concepts of the bandwidth reduction methods is introduced. Readers are directed to [Cuthill 69] and [Gibbs 76] for details.

5.4.1 Basic Concepts From Graph Theory

5.4.1.1 Basic Definitions

Significant insight can be gained by using a graph to represent the relationship between components of $\mathbf{x}$, imposed by the system matrix $\mathbf{A}$. Given a set of points $\{v_1, v_2, \cdots, v_n\}$ on the same plane, a graph $G = \langle V, E \rangle$ can be defined, where $V = \{v_1, v_2, \cdots, v_n\}$ and $E = \{(v_i, v_j) : v_i, v_j \in V, i \neq j\}$. The elements of V and E are called

vertices and *edges*. If $(v_1,v_2) \in E$, then v_1 and v_2 are said to be *adjacent*. The *degree* of a vertex is the number of vertices adjacent to it. A *path* of length t is a sequence of connecting edges (v_0,v_1), (v_1,v_2),..., (v_{t-1},v_t). A graph is *connected* if there is a path connecting each pair of vertices. The *distance* between vertices v_1 and v_2 of a connected graph is the length of the shortest path from one to another. The *diameter* of graph G is the shortest path connecting two vertices between which the distance is maximum among those of all possible combinations of vertex pairs in V. The term diameter will also be used to indicate the length of such a path. A one-to-one map, f, from V onto the set $\{1,2,\cdots,n\}$ is called a *numbering* of G. For each numbering f, we define $\beta_f(G)$, the *bandwidth* of G *relative to* f, as

$$\beta_f(G) = \max\{|f(v_i) - f(v_j)| : (v_i,v_j) \in E\} \tag{5.4.1}$$

The minimum of $\beta_f(G)$ over all numbering of G is called the *bandwidth* of G and denoted as $\beta(G)$.

5.4.1.2 The Level Structure

An important concept in many bandwidth reduction algorithms is that of level structure. A *level structure*, $L(G)$ with depth k, of a graph G is a partition of the set V into levels $L_1,L_2,\cdots,L_k$ such that

1. all vertices adjacent to vertices in level L_1 are in either level L_1 or L_2,
2. all vertices adjacent to vertices in level L_k are in either level L_k or L_{k-1}, and
3. all vertices adjacent to vertices in level L_i $(1<i<k)$ are in either L_{i-1}, L_i or L_{i+1}.

A level structure can be established with respect to any particular vertices in V. It is called the *level structure rooted at* v $(v \in V)$ and denoted as $L_v(G)$. Its levels are determined by

1. $L_1 = \{v\}$ and
2. for $i>1$, L_i is the set of those vertices adjacent to vertices of level L_{i-1} not yet assigned to a level.

In any level structure, the number of vertices in level L_i is called the *width of level i* and is denoted as $w_i(L)$. $w(L) = \max\{w_i\}$ is the *width* of the level structure. It can be easily shown that for any level structure, L, a numbering f_L that assigns consecutive

integers level by level, first to the vertices of level L_1, then to those of L_2, and so forth, will yield a bandwidth β_{f_L} satisfying

$$\beta_{f_L} \leq 2w(L) - 1 \tag{5.4.2}$$

5.4.2 Bandwidth Reduction Algorithm

Typically, a bandwidth reduction algorithm will involve two steps: finding a starting vertex to generate a rooted level structure and the numbering.

A. *Finding a starting vertex* : It is obvious that increasing the number of levels usually decreases the average number of vertices in each level and tends to decrease the width of the level structure, thus the bandwidth as well. Ideally, the level structure should be generated to root at end vertices of a diameter. However, often we generate a pseudo-diameter instead of the real diameter since finding the real diameter is computationally costly. A pseudo diameter can be found by the following steps

1. Pick an arbitrary vertex with minimal degree and call it v.
2. Generate the level structure rooted at v, called L_v. Let S be the set of vertices which are in the last level of L_v.
3. Generate level structures rooted at vertices $s \in S$ selected in order of increasing degree. If for some $s \in S$ the depth of L_s is greater than the depth of L_v, replace v by s and return to step 2.
4. From v and vertices in S, choose the vertex for which the associated rooted level structure has the smallest width as the starting vertex.

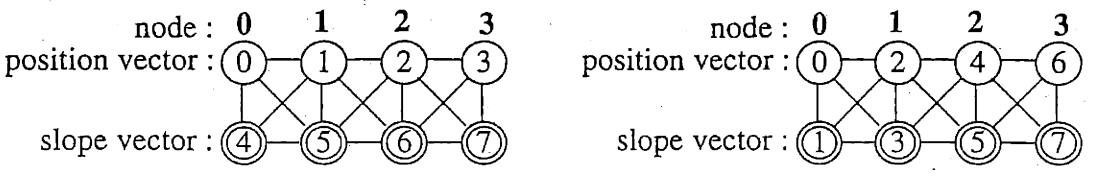
B. *Numbering*: once the rooted level structure is generated, consecutive integers can be assigned to the vertices level by level according to the following procedures.

1. Assign the number 1 (or the smallest positive integer appropriate to the application, like number 0) to the root vertex.
2. For each successive level beginning with level 2, sort the vertices in two keys: first the lowest number of their adjacent numbered vertices in the preceding level, and second the degree of the vertices in ascending order. Number the vertices according to the sorting order. The procedure terminates when all the vertices on all levels have been numbered.

The procedures described above are just the basic concepts in various methods. Variations can be made to improve the algorithm's performance. For example, the level structure need not root at a single vertex, i.e., level 1 might contain more than one vertex. The actual procedure implemented in this thesis is a combination of Cuthill-McKee's and Gibb's algorithms. Gibb's method for finding the starting vertex and Cuthill-McKee's method for numbering vertices were adopted to seek a balance between computing cost, implementation complexity and performance. Details of various methods can be found in the above-mentioned references and references therein. The Following sections are two examples of the bandwidth reduction algorithms applied on FE curves and surfaces.

5.4.3 Example 1: FE Curves

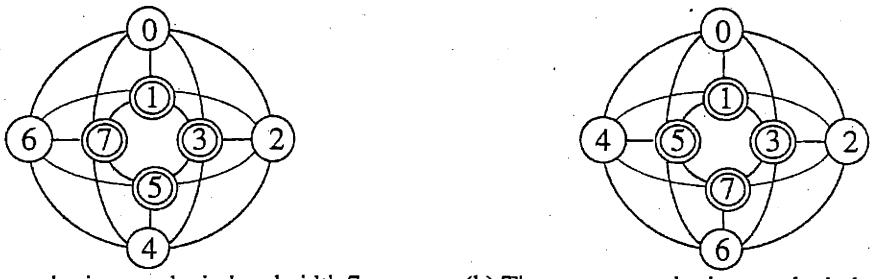
Before applying the bandwidth reduction algorithm, we should represent the degrees of freedom of the FE curve by a graph. Figure 5.6 shows the graph representation of the degrees of freedom of a cubic Hermite curve. The circle represents a position vector of a node and the double circle the slope vector. The integer in a (double) circle(s) is the numberings of the corresponding degrees of freedom. For open cubic Hermite curves, if the bandwidth reduction algorithm is not considered, though the numbering of the global degree of freedom could be arbitrary, two different numberings could naturally arise. The first one is to number all the position vectors at the nodes first, and then all the slope vectors. The other is to number the degrees of freedom node by node. These two different numberings are illustrated in figures 5.6a and b. The former results in a longer bandwidth which is 5 in this example compared to the bandwidth 3 that resulted from the latter numbering. As the number of nodes increases, the bandwidth resulted from the first numbering method increases correspondingly while the other remains the same. Obviously the numbering illustrated in figure 5.6b is the better choice which is also the result after applying the bandwidth reduction algorithm.



(a) A numbering results in bandwidth 5

(b) The proper numbering results in bandwidth 3

Figure 5.6: The graph representations of two different numbering of an open cubic Hermite curve with four nodes



(a) A numbering results in bandwidth 7 (b) The proper numbering results in bandwidth 5

Figure 5.7: The graph representations of two different natural numbering of a closed cubic Hermite curve with four nodes

When applying the above-mentioned numbering rule on closed cubic Hermite curves, a bandwidth as large as possible will arise in the last curve element connecting the first and the last nodes. This is illustrated in figure 5.7a. The proper numbering is shown in figure 5.7b.

5.4.4 Example 2: FE Surfaces

It is easy to number the degrees of freedom of a FE curve with the minimum bandwidth without using any bandwidth reduction algorithm. Using the bandwidth reduction algorithm is more essential for finding a better numbering for the degrees of freedom of FE surfaces. The original numbering of the vertices in the triangular mesh used as the parametric domain of a FE surface always first numbers the triangle's vertices on the domain's boundary in a counterclockwise order, then the other interior vertices based on the element's numbering, and finally all the midedge nodes. This is done purely to ease the implementation of the domain Delaunay triangulation algorithm which needs the information about the domain's boundary to exclude the triangles outside the boundary. The numbering of degrees of freedom is then assigned according to the nodes' numbering, three degrees of freedom for vertex nodes and one for mid-edge nodes. The bandwidth of the system matrix generated by this numbering is generally very large.

For the 12-dof triangular element, each triangle in the mesh will result in 12 vertices and 66 (C_2^{12}) edges in the graph representation. For the 21-dof element, 21 vertices and 210 edges. However, in the implementation, we did not generate the graph representation of all the degrees of freedom to perform the bandwidth reduction algorithm. This is because doing so will make the degrees of freedom associated to the same node have

nonsuccessive numbering, which would be unnatural and raise difficulties in the programming. Instead, the graph representation is generated based on the vertex nodes and midedge nodes. Figure 5.8 illustrates the graph representation of a 12-dof triangular element, which contains 6 vertices and 15 edges. A proper numbering of the nodes is generated at first, and the numbering of the degrees of freedom is assigned according to the nodes' numbering.

The parametric domain of the Gaussian surface in figure 6.7e is used as an example. This triangular mesh is composed of 54 vertex nodes, 129 mid-edge nodes and 76 triangles. The original numbering of all the nodes are shown in figure 5.9a. It results in a bandwidth of 281 (the total degrees of freedom is 291). The new numbering after applying the bandwidth reduction algorithm is shown in figure 5.9b in which the bandwidth is reduced to 74.

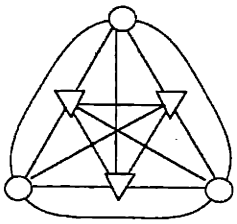
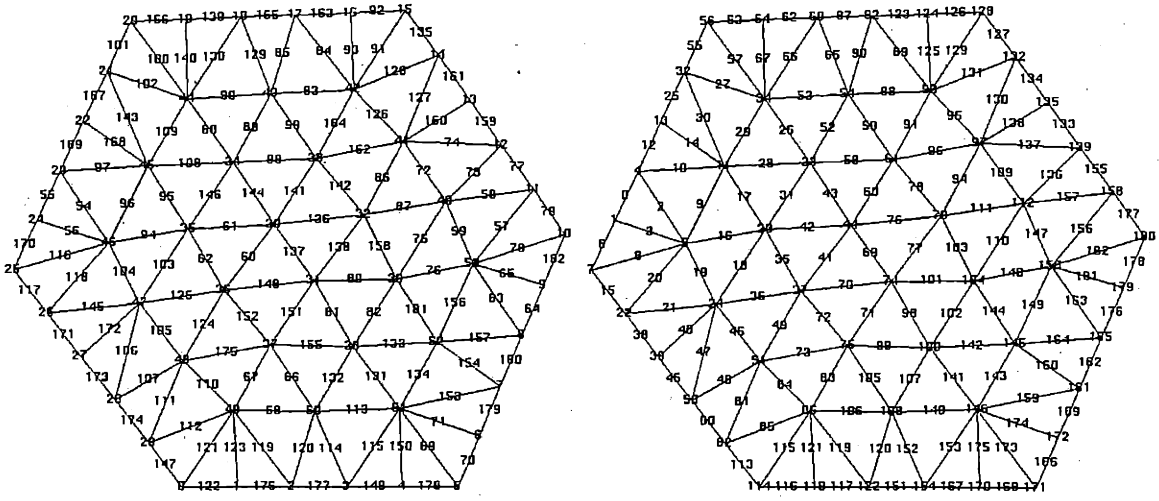


Figure 5.8: Graph representation of a 12-dof triangular element



(a) Before the bandwidth reduction: bandwidth= 281 (b) After the bandwidth reduction: bandwidth= 74

Figure 5.9: An example of bandwidth reduction for FE surfaces

5.5 COMPLEXITY ANALYSIS

Two main processes are involved in each it
eration of the energy minimization using SQP for curve and surface reconstruction: finding the attachment points for each data point and solving a set of linear equations to update the solution estimate. To find the attachment points, since the computation of minimum distance is performed over each element for each data point, the computing time is $O(NM)$, where N designates the number of data points and M the number of elements used to represent the curve or surface. However, it is reasonable to assume that the attachment points only move a small distance during two consecutive iterations. Thus, the search can be performed on the elements that the attachment points fall on in the previous iteration and their adjacent elements. This will reduce the computing time to $O(N)$. To solve a set of linear equations, the computing cost is generally $O(N_{\text{dof}}^2)$ where N_{dof} is the number of the unconstrained degrees of freedom which can be deemed as linearly proportional to the number of elements used. If we take advantage of the fact that the system matrix is banded, the computing cost will be $O(N_{\text{dof}})$.

For surface reconstruction, some additional stages are required before entering the energy minimization stage. Compared to the computing cost of the energy minimization, most of the computing time of these stages is negligible except for the triangulation of the parametric domain using the soap bubble method. The considerable computing cost of the soap bubble method, $O(n \log n)$ with n indicating the number of floating bubbles, is due to the dynamic simulation of the soap bubbles' behavior [Shimada 93]. If the parametric domain is remeshed very frequently according to the surface's curvature, the computing cost of the energy minimization process will be significantly increased.

CHAPTER SIX

Results

This chapter presents some results for curve and surface reconstruction using the methods described in Chapter 3 and Chapter 4. Section 6.1 gives the results for curve reconstruction. Two examples showing curves reconstructed from error-filled data points are given. Data points were randomly generated from analytic target curves with random perturbations with a maximum error of 5% . The reconstructed curves are interrogated by the accuracy of fit and the curvature and curvature variation plot. Comparisons to existing curve interpolating methods are also given. Section 6.2 gives the results for surface reconstruction. The examples used are a rotation invariant Gaussian surface and a hood of the Ford Taurus car. The reconstructed surfaces are interrogated by various methods described in Chapter 4.

All the figures shown in this chapter are generated on a Silicon Graphics Indigo^2 Extreme graphic workstation. This workstation uses a R4000 CPU running at 100 MHz and is capable of rendering 1.2M 3D vectors or 205K unlighted, Gouraud-shaded polygons per second.

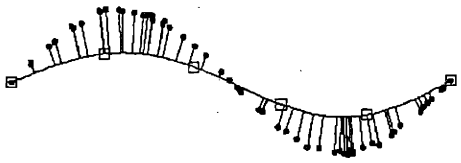
6.1 RESULTS FOR CURVE RECONSTRUCTION

The first example is a sine curve defined within $[0, 2\pi]$. Sixty points were generated randomly with errors in the y direction only. Figure 6.1 shows the input data points, the curve during deformation, the final reconstructed curve and its curvature distribution. The second example is an airfoil defined by a fourth order B-spline. Errors are inserted in the normal direction. The original airfoil, the reconstructed one and their curvature distributions are shown in figure 6.3. The data used in these two examples were also used in [Fang 92a], however, instead of cubic Hermite curves, the examples use quintic Hermite curves with fewer elements. The plots of curvature κ and curvature variation $d\kappa/ds$ with respect to the normalized arc length for both the reconstructed curve and the original curve are shown in figure 6.2 and figure 6.4.

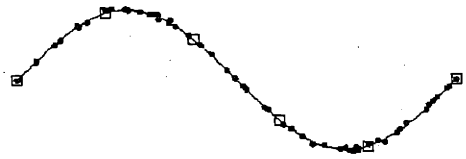
Comparisons with other existing methods are also made. Figure 6.5 and 6.6 show comparisons of the approximating curve produced by the proposed method and interpolating curves using uniform, chord-length and centripetal parametrizations. The data points used have no errors. Though the proposed method only produces an approximation curve, the errors between the points and the curve can be sufficiently small. The results show that the proposed method generally produces fairer curves than other methods. Another advantage of the proposed method is that oversampled data points can be reconstructed by a C^2 curve with only a few elements, in contrast to traditional methods in which an interpolating curve is generated followed by a merging process which eliminates the original C^2 continuity.

Quantitative results about the accuracy of fit of the two examples are given in table 6.1. The statistics on the pointwise errors are obtained by sampling points on the original curve, calculating the minimum distances to the reconstructed curve and then computing the statistics based on these minimum distances. The L_2 norm is calculated by performing trapezoidal integration from the minimal distances and the arc length.

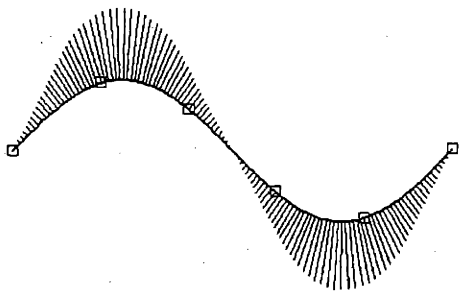
(a) The input of unordered, error-filled data



(b) The deforming curve with springs attached to it

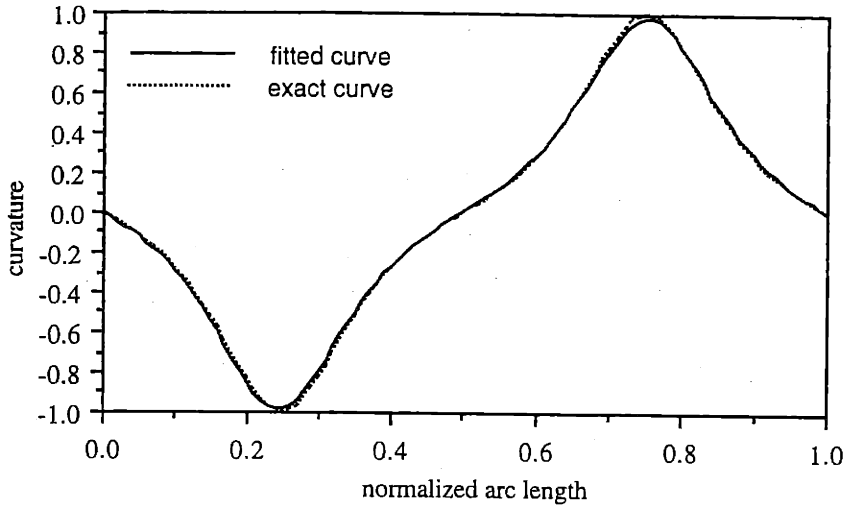


(c) The reconstructed curve with five quintic Hermite elements

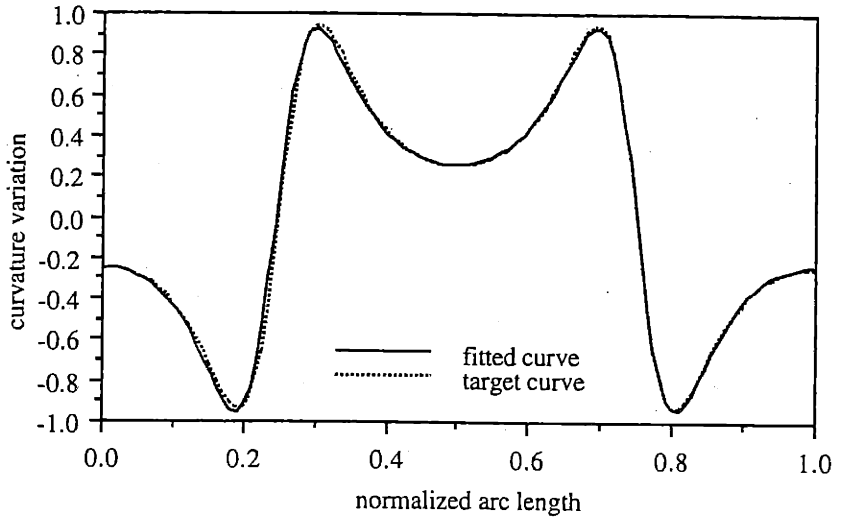


(d) The reconstructed curve with curvature comb

Figure 6.1: Example 1: sine curve (no. of data points = 60)

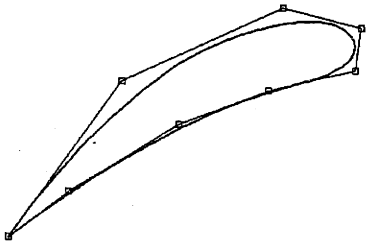


(a) Curvature plot of the reconstructed sine curve compared to the exact sine curve

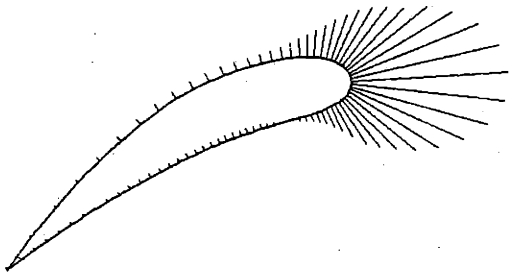


(b) Curvature variation plot of the reconstructed sine curve compared to the exact sine curve

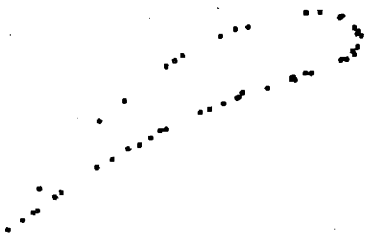
Figure 6.2: The plots of the curvature κ and curvature variation $d\kappa/ds$ of the reconstructed sine curve and the exact sine curve.



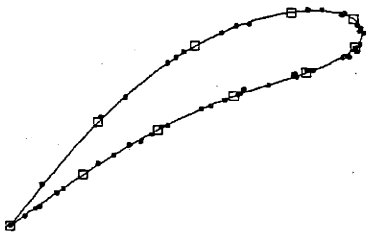
(a) The original airfoil defined by B-spline (Control points are represented by empty squares)



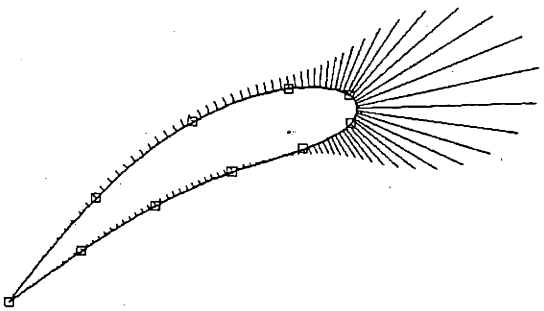
(b) The curvature comb of the original curve



(c) The input of unordered, error-filled data

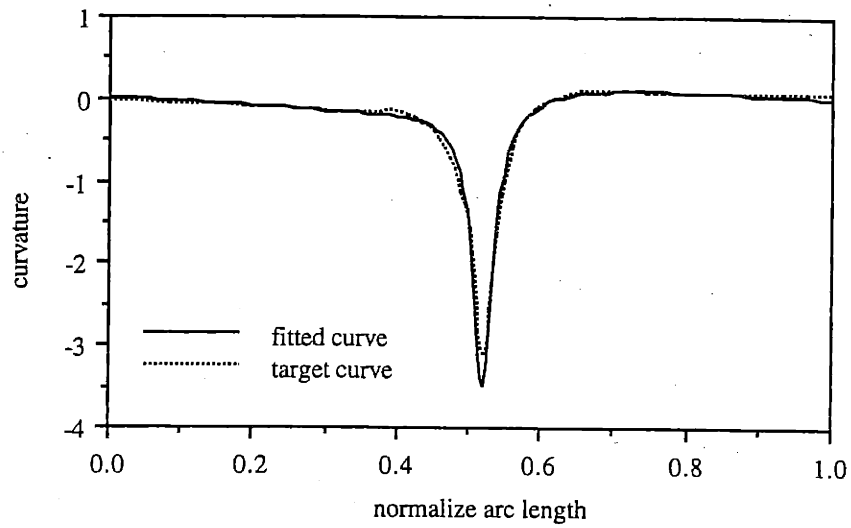


(d) The reconstructed curve using ten quintic Hermite element

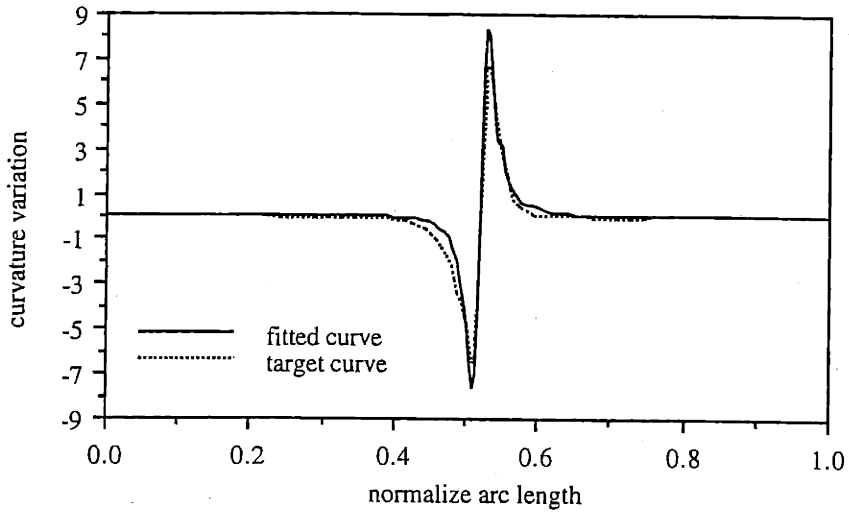


(e) The reconstructed curve with its curvature comb

Figure 6.3: Example 2: an airfoil defined by a fourth order B-spline. (no. of data points = 55)

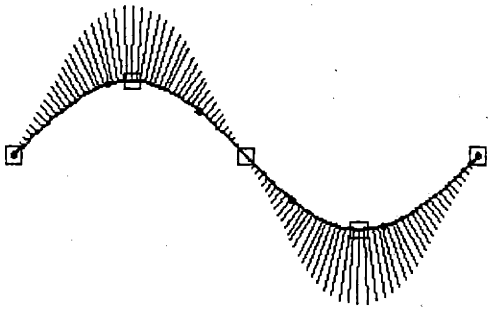


(a) Curvature plot of the reconstructed airfoil compared to the target curve

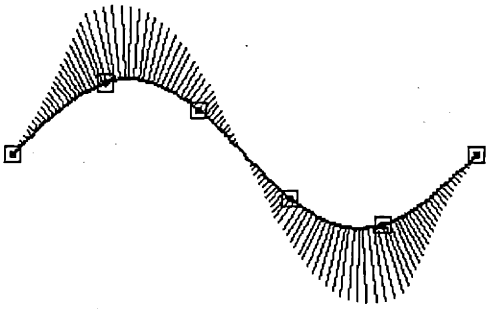


(b) Curvature variation plot of the reconstructed airfoil compared to the target curve

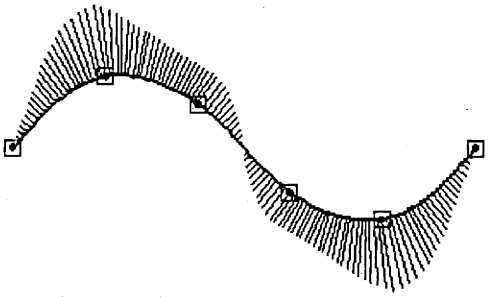
Figure 6.4: The plots of the curvature κ and curvature variation $d\kappa/ds$ of the reconstructed airfoil and the exact one.



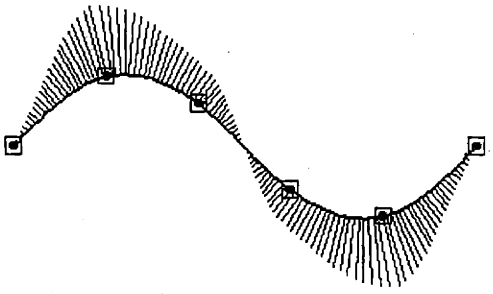
(a) An approximation curve generated by the proposed method



(b) An interpolating curve with uniform parametrization

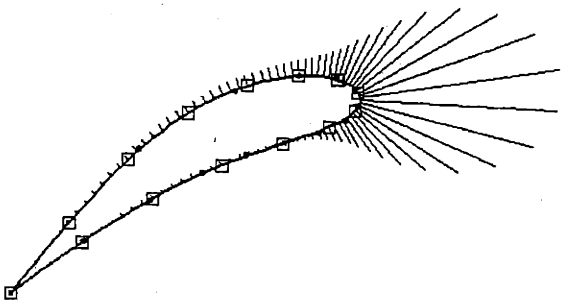


(c) An interpolating curve with chord-length parametrization

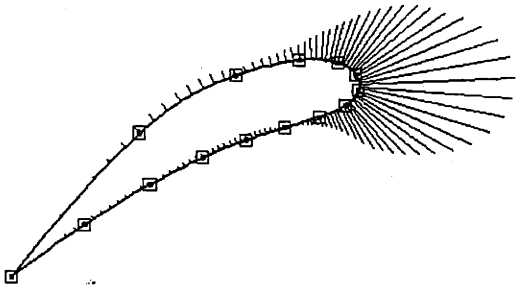


(d) An interpolating curve with centripetal parametrization

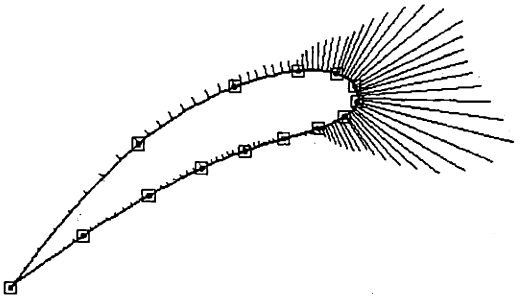
Figure 6.5: Comparison of the sine curve reconstructed by different methods



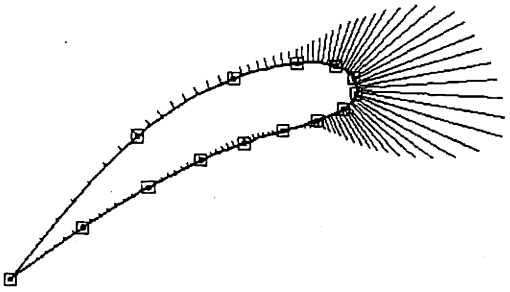
(a) An approximation curve generated by the proposed method



(b) An interpolating curve with uniform parametrization



(c) An interpolating curve with chord-length parametrization



(d) An interpolating curve with centripetal parametrization

Figure 6.6: Comparison of the airfoils reconstructed by different methods

Table 6.1: The accuracy of fit of the sine curve and the airfoil. (Values in the parenthesis are the values of the target curve.)

Curve		sine curve	airfoil
pointwise	e_{max}	0.009799	0.029417
	e_{mean}	0.002613	0.012217
	e_{stddev}	0.002837	0.008005
L_2 norm		0.01925	0.2234
$\lvert\kappa\rvert_{max}$		0.9983 (1.0)	3.52 (3.11)
$\lvert d\kappa/ds\rvert_{max}$		0.9559 (0.943)	8.312 (6.63)
arc length		7.64545 (7.640)	16.848373 (16.843323)

• chord length of the airfoil = 7.943

6.2 RESULTS FOR SURFACE RECONSTRUCTION

The first example used is the Gaussian surface which is an explicit function expressed as

$$F(x,y)=\exp\left\{-\frac{1}{2}\left[\frac{(x-\bar{x})^2}{\sigma_x^2}+\frac{(y-\bar{y})^2}{\sigma_y^2}\right]\right\} \tag{6.2.1}$$

This is actually the two-variable normal distribution function with a unit value when $(x,y)=(\bar{x},\bar{y})$. The parameters σ_x and σ_y control the decreasing rate from the center $(\bar{x},\bar{y})$. The parameter values used are $\sigma_x=\sigma_y=1.0$, which results in a rotation invariant function with respect to the vertical axis passing $(\bar{x},\bar{y})$.

Figure 6.7 illustrates the whole surface reconstruction process. Figure 6.7a shows the distribution of the data points on x-y plane: boundary points are sampled along edges of a hexagon centered at $(\bar{x},\bar{y})$ and interior points are sampled evenly distributed inside the hexagon. Figure 6.7b shows the data points scattered in the three dimensional space and the reconstructed boundary curves. Figure 6.7c and 6.7d show the initial unclosed polygon on $u-v$ plane and the closed one after the constrained minimization. The initial triangular mesh resulted from the use of constant density function and the corresponding approximation surface constrained by boundary curves are shown in figure 6.7e and figure 6.7f respectively. This mesh changes according to the maximum curvature of the reconstructed surface in the minimization. The final mesh and the final reconstructed surface are shown in figure 6.7g and 6.7h. Note that the triangles around the center is smaller than the others.

The results of applying various interrogation schemes to the reconstructed Gaussian surface are shown in figure 6.8 and 6.9. Figure 6.8 displays the interrogation results using the zero order schemes including the wireframes composed of isoparametric curves in both barycentric coordinates and (u,v) coordinates and the contour curves created by intersecting the reconstructed surface with a family of parallel planes of constant x, y and z values. Note that the nearly-circle z contour curves coincides with the z contour curves of the exact surface. The results of applying the first order interrogation schemes are shown in figure 6.9. Figure 6.9a shows the isophotes with respect to the light direction

indicated by the arrows. The C^0 continuous isophotes proves that the surface is C^1 continuous. Figure 6.9b and 6.9c show the configuration of the light lines and the view point and the corresponding Klass-type reflection lines.

The second example used is a hood of the Ford Taurus car. The original surface is only a half hood defined by a NURBS surface with 29×45 control points. A description of the whole hood is obtained by flipping the control points with respect to the center line, thus resulting in a NURBS surface with 57×45 control points. The reconstructed Taurus hood consists of 156 triangular elements with 543 degrees of freedom, only 21% of the numbers of degrees of freedom used for the whole Taurus hood defined in NURBS.

The results of applying various interrogation schemes to the reconstructed Taurus hood are shown in figure 6.10 to 6.14. Figure 6.10 shows the results of applying the zero order schemes. We can see that the u-v isoparametric wireframe indicates a concave region around the middle of the hood. However, from the x and y contour maps, the surface is actually convex. This is an example that the isoparametric wireframe may sometimes mislead the surface's shape. Comparisons with the original Taurus hood are made by applying the interrogation schemes to both the reconstructed and the original surfaces. Figure 6.11 and figure 6.12 show the comparisons by their z contour curves and the isophotes respectively. The light direction used for generating the isophotes is indicated by the arrows. In figure 6.13 and figure 6.14, two different configurations of the light lines and view point are used to produce the Klass-type reflection lines for both the reconstructed and the original Taurus hood. The comparisons of the Gaussian curvatures and mean curvatures of these two surfaces are shown in figure 6.15. All these figures show a good match between the reconstructed surface and the original surface except that the curvature at the front end of the reconstructed hood is smaller than the original one This is probably because insufficient elements are used in the reconstruction.

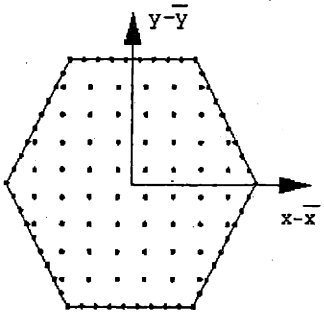
For verifying the effect of the inclusion of the local irregularity measurement in the energy functional, we reconstructed the above-mentioned Gaussian surface using mesh with constant density by minimizing different energy functionals. Figure 6.16 displays the coded-color maps of the Gaussian curvature and the mean curvature of two reconstructed surfaces. The energy functional used in figure 6.16a includes the local irregularity measurement, i.e., the C^2 discontinuity term, while the one used in figure 6.16b does not. It is obvious that the curvature discontinuity across the triangular

element's boundary is reduced in figure 6.16a and results in a smoother curvature distribution.

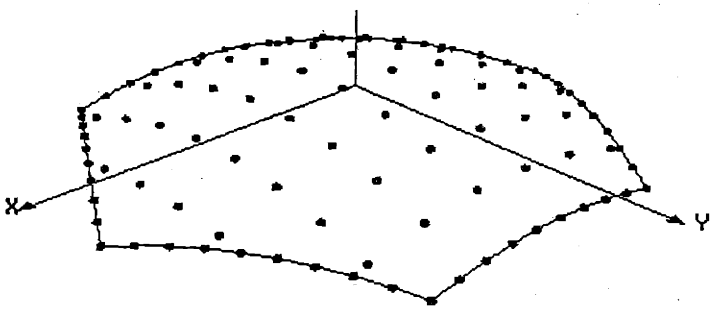
For verifying the accuracy of fit, additional points are sampled on the original surface. Note that the data points used as the input of the surface reconstruction cannot be used to measure the accuracy of fit since errors are embedded. The minimum distances between these additional points and the reconstructed surface are calculated and the statistics are then obtained. These pointwise error measurements for the reconstructed Gaussian surface and Taurus hood are shown in table 6.2.

Table 6.2: Pointwise errors of the reconstructed Gaussian surface and Taurus hood

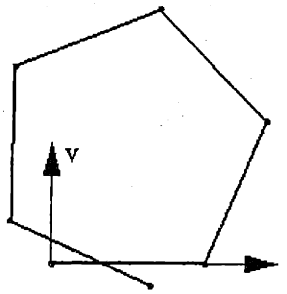
	Gaussian surface	Taurus hood
number of additionally sampled data	616	899
maximum error e_{max}	0.006794	0.006343
mean error e_{mean}	0.001159	0.001852
standard deviation e_{stddev}	0.000658	0.001216



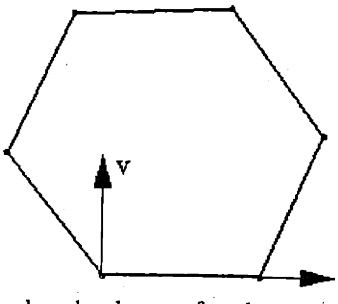
(a) The distribution of data points on (x, y) plane



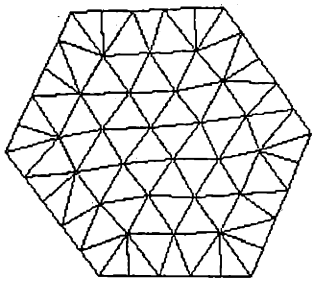
(b) The scattered data points in three dimensional space and the reconstructed boundary curves



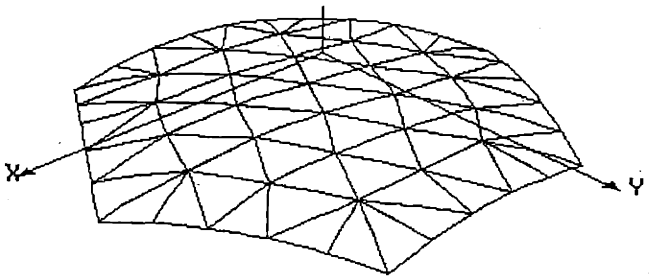
(c) The unclosed polygon on u-v plane



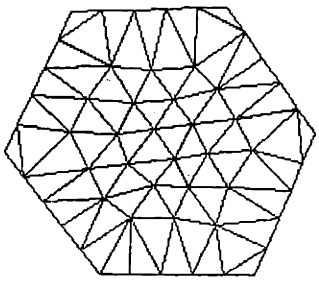
(d) The closed polygon after the constrained minimization



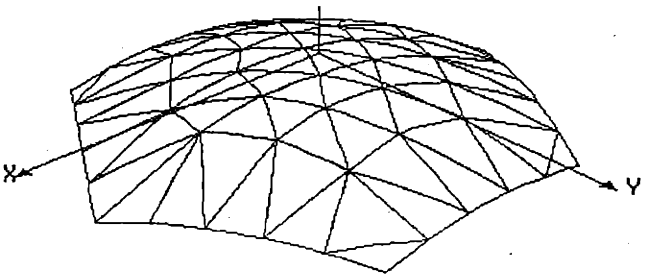
(e) The triangular mesh using constant density function



(f) The first approximation surface constrained by boundary curves

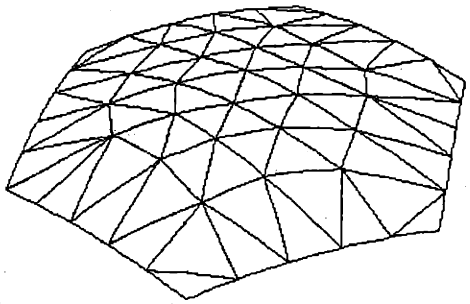


(g) The triangular mesh with variable size

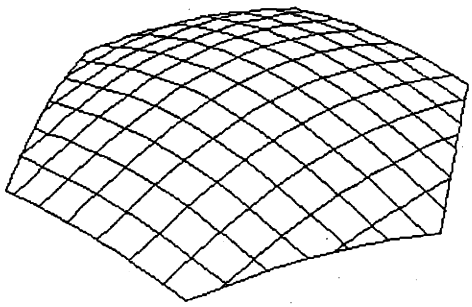


(h) The final reconstructed surface

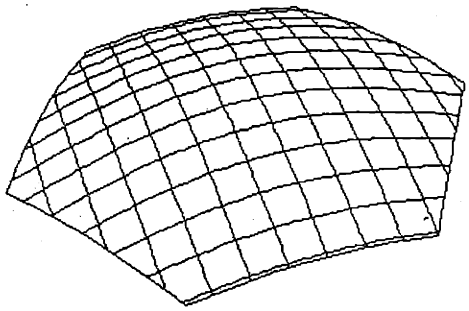
Figure 6.7: The surface reconstruction process



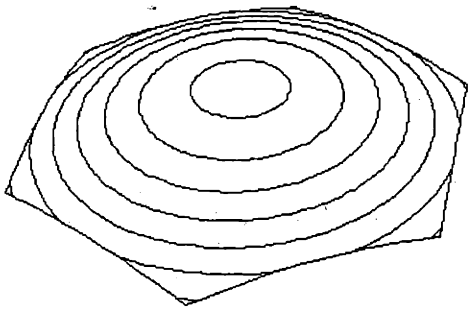
(a) The isoparametric wireframe in barycentric coordinates



(b) The isoparametric wireframe in u-v coordinates

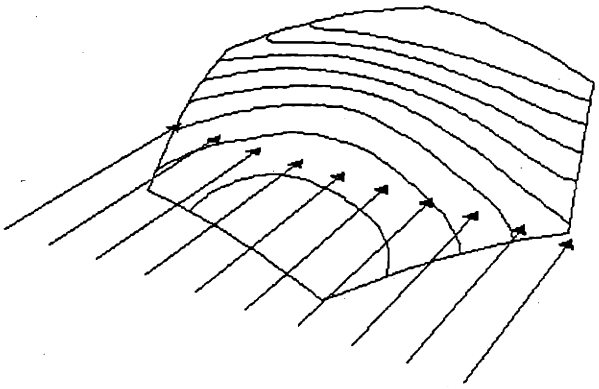


(c) The constant x and y contour curves

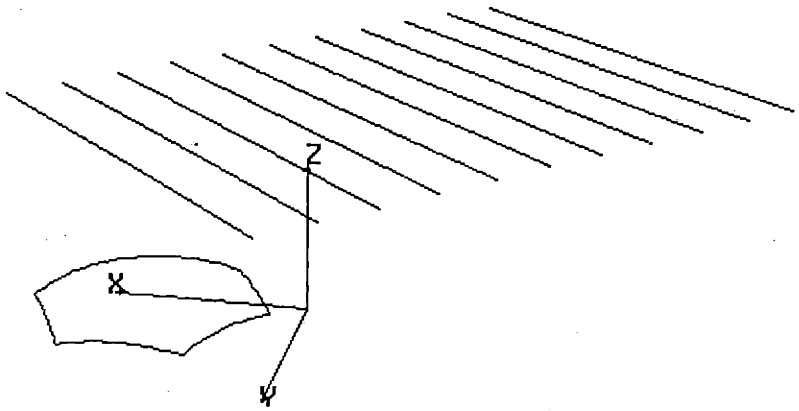


(d) The constant z contour curves

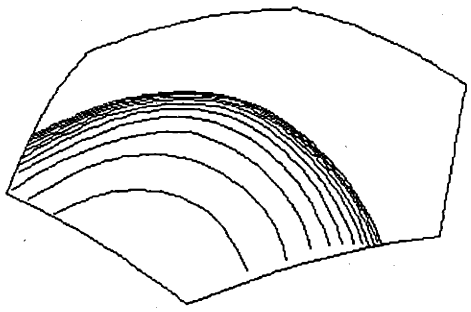
Figure 6.8: Interrogation of the reconstructed Gaussian surface using the zero order schemes



(a) The isophotes (arrows indicate the light direction)

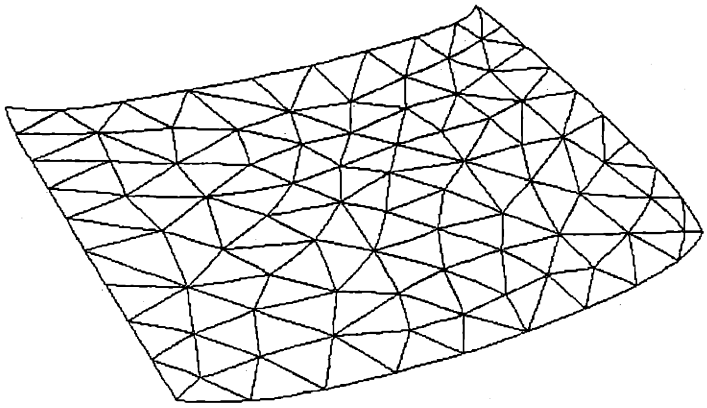


(b) Configuration of the light lines and the view point
(the view point is represented by the black dot)

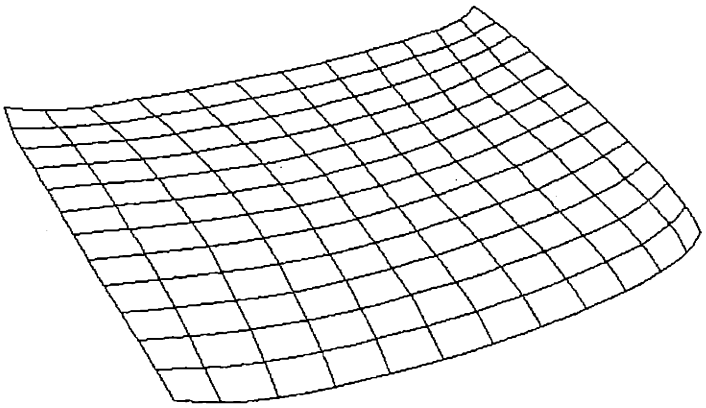


(c) The corresponding Klass-type reflection lines

Figure 6.9: Interrogation of the reconstructed Gaussian surface using the first order schemes

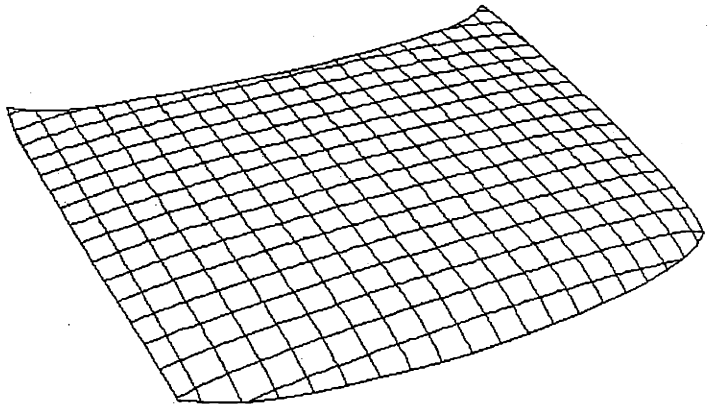


(a) The isoparametric wireframe in barycentric coordinates

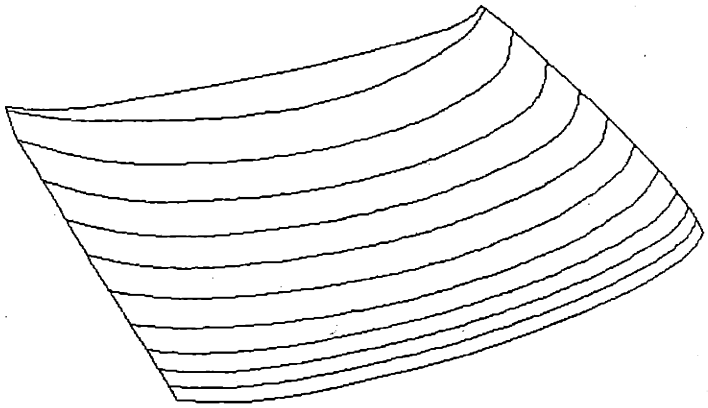


(b) The isoparametric wireframe in u-v coordinates

Figure 6.10: Interrogation of the reconstructed Taurus hood using the zero order schemes

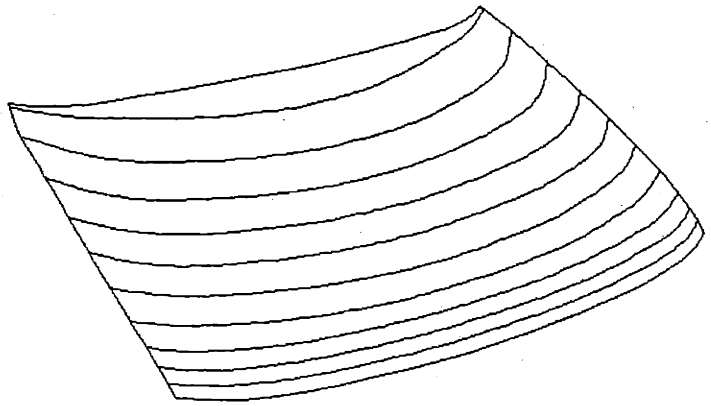


(c) The constant x and y contour curves

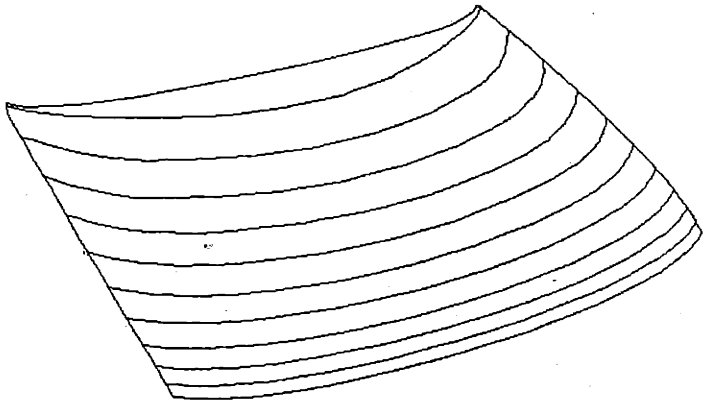


(d) The constant z contour curves

Figure 6.10 (cont'd): Interrogation of the reconstructed Taurus hood using the zero order schemes

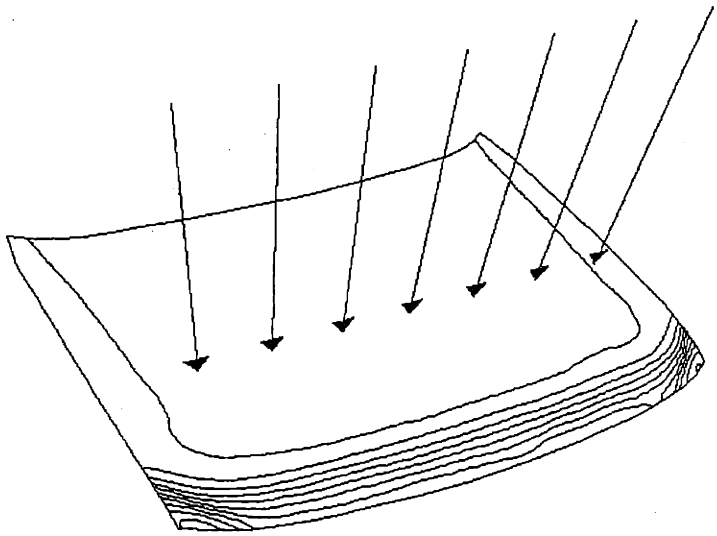


(a) Constant z contour curves of the reconstructed Taurus hood

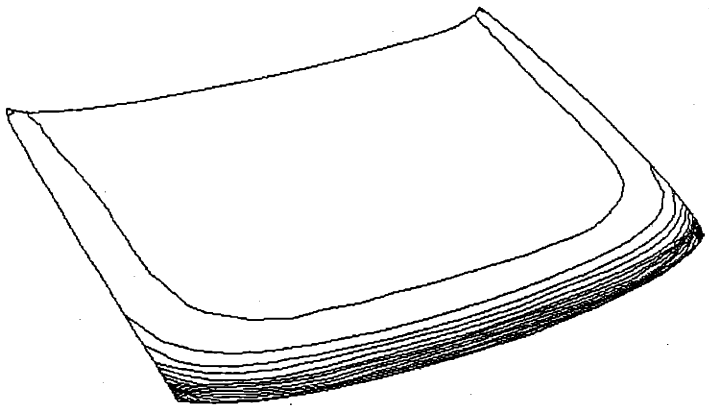


(b) Constant z contour curves of the original Taurus hood

Figure 6.11 Comparisons of the reconstructed and original Taurus hoods using constant z contour curves

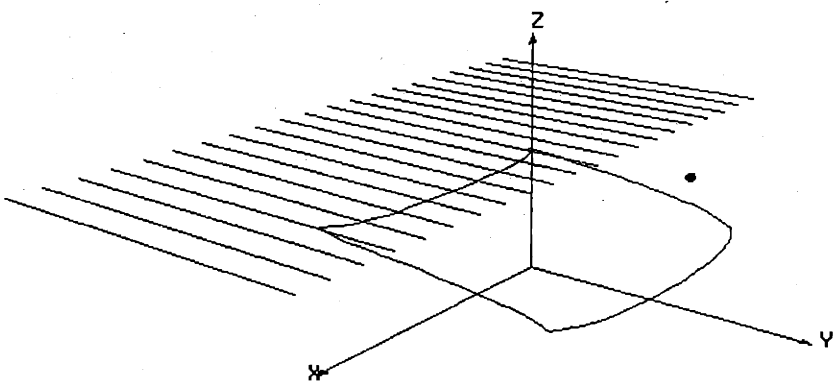


(a) Isophotes of the reconstructed Taurus hood (arrows indicate the light direction)

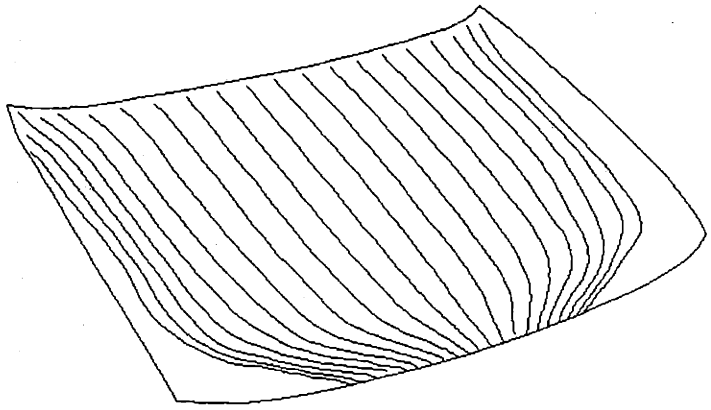


(b) Isophotes of the original Taurus hood (use the same light direction as in (a))

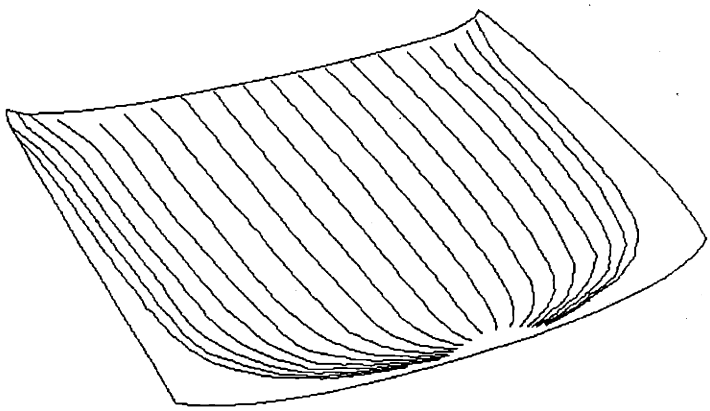
Figure 6.12 Comparisons of the reconstructed and original Taurus hoods using isophotes



(a) Configuration of light lines and the view point.
(the view point is represented by the black dot)

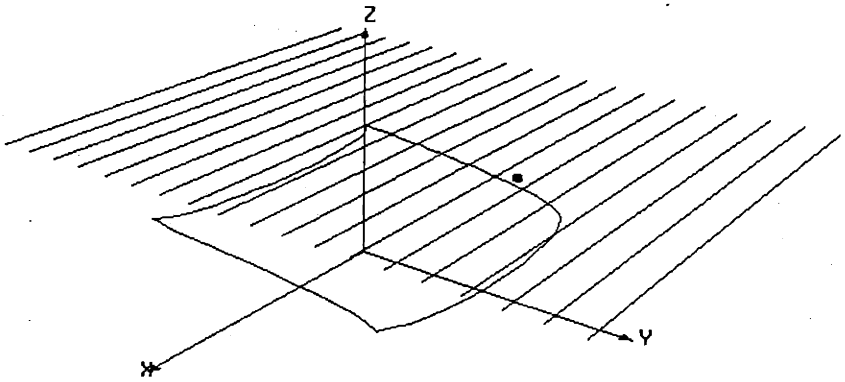


(b) Reflection lines of the reconstructed Taurus hood

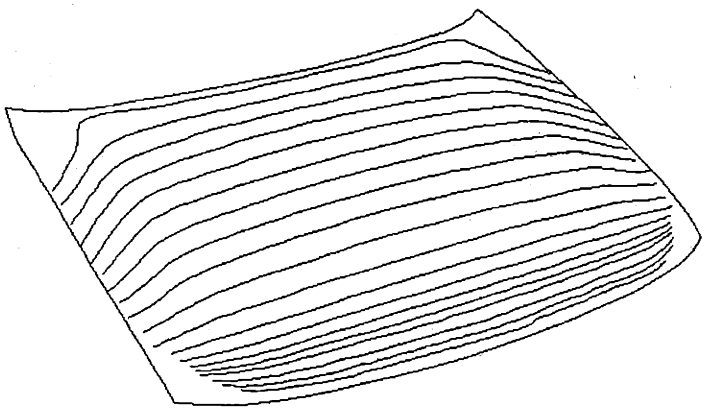


(c) Reflection lines of the original Taurus hood

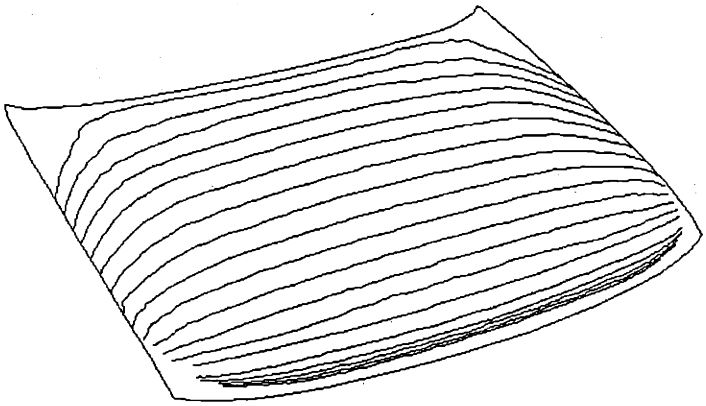
Figure 6.13 Comparisons of the reconstructed and original Taurus hoods using Klass-type reflection lines (light lines configuration no.1).



(a) Another configuration of the light lines and the view point.
(the view point is represented by the black dot)

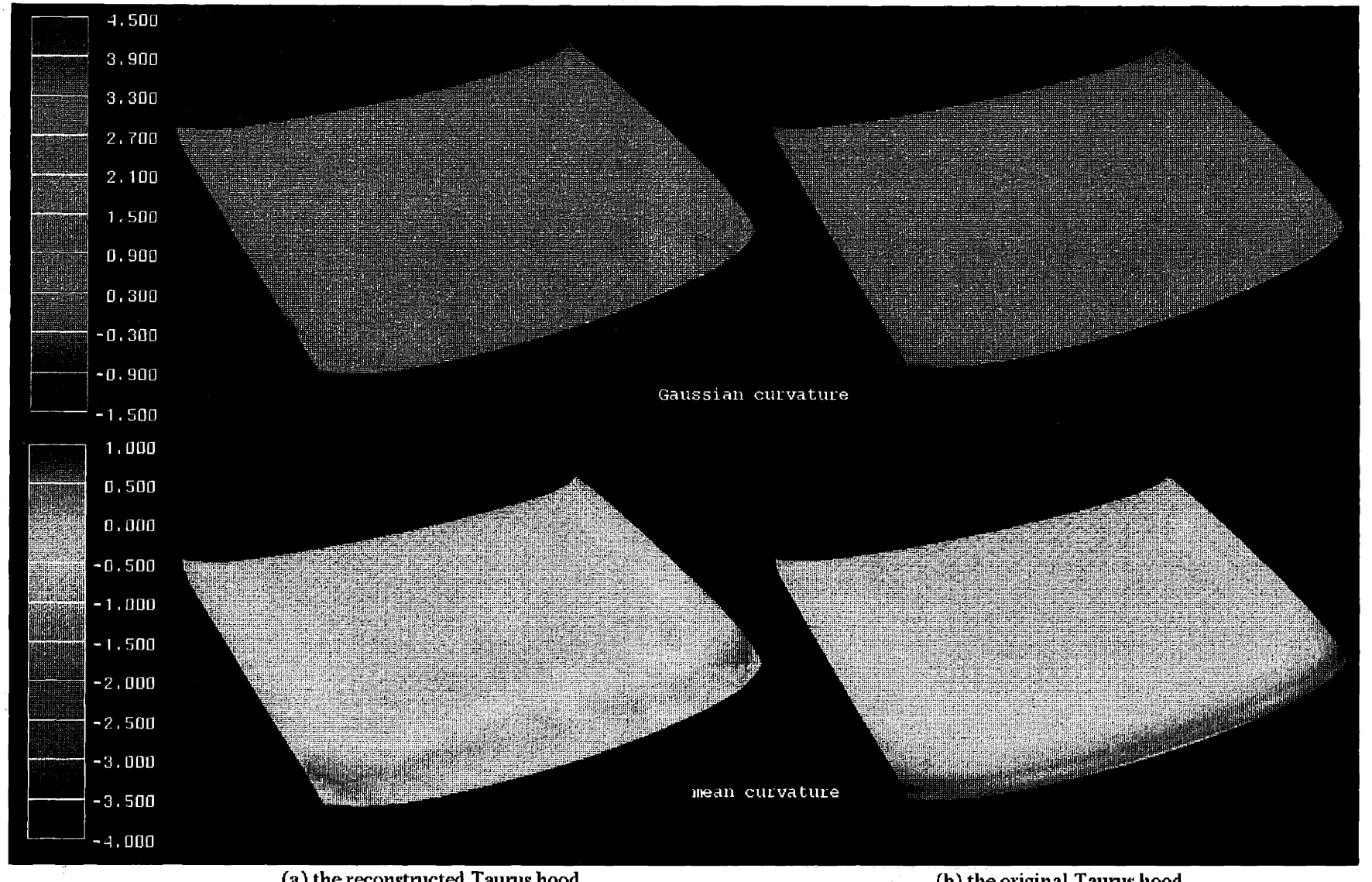


(b) Corresponding reflection lines of the reconstructed Taurus hood



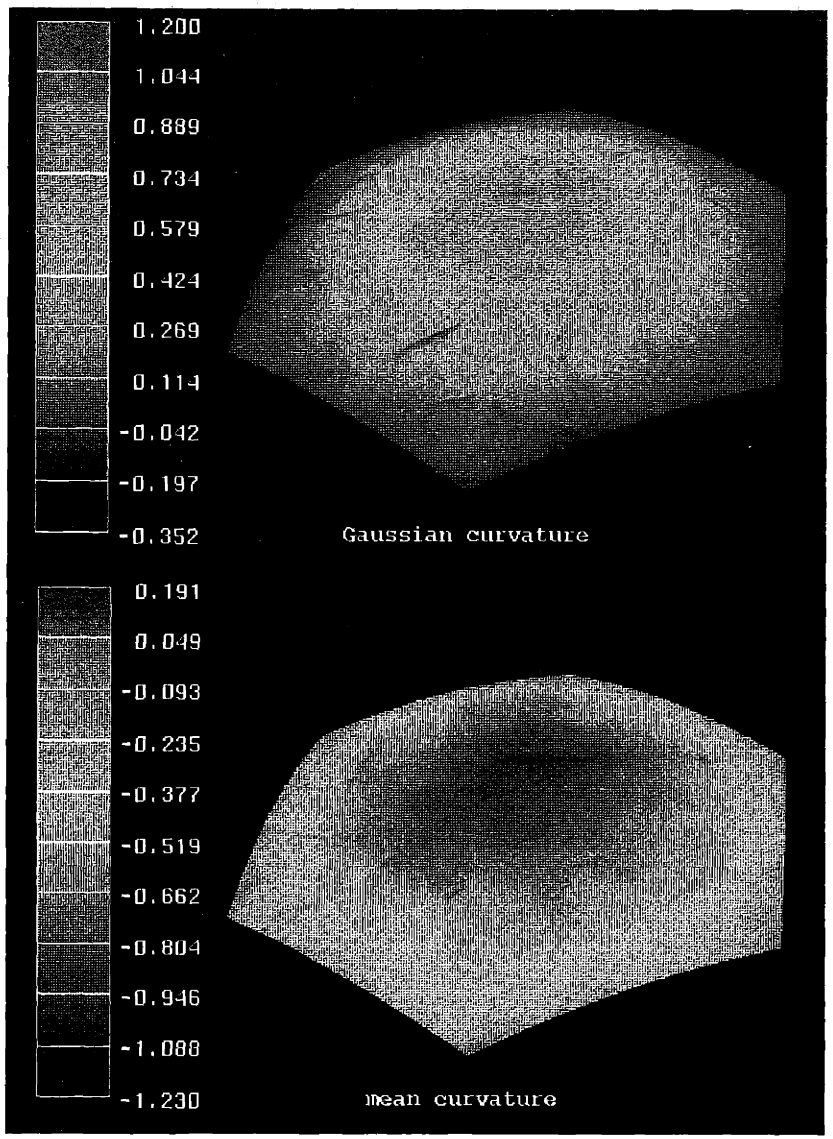
(c) Corresponding reflection lines of the original Taurus hood

Figure 6.14 Comparisons of the reconstructed and original Taurus hoods using Klass-type reflection lines (light lines configuration no.2).

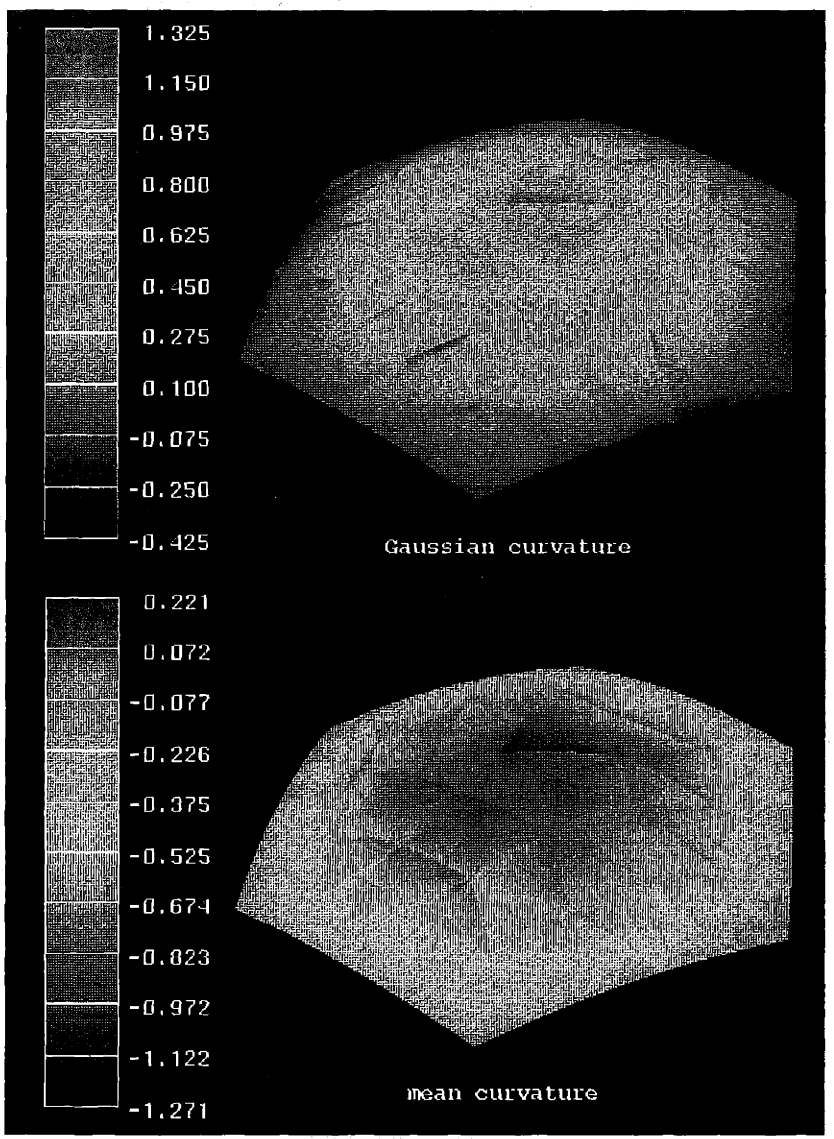


(a) the reconstructed Taurus hood (b) the original Taurus hood

Figure 6.15: Comparisons of the reconstructed and original Taurus hood using coded–color curvature maps



(a) Surface reconstructed with the 2nd derivatives discontinuity minimized



(b) Surface reconstructed without minimizing the 2nd derivatives discontinuity

Figure 6.16: The coded–color curvature maps of surfaces reconstructed by minimizing different energy functionals

CHAPTER SEVEN

Conclusions

7.1 THESIS SUMMARY

In view of the deficiencies in the design process currently used in the automobile industry, curve and surface reconstruction methods have been developed with the following capabilities using the variational principle and a finite element based geometry representation

- reconstruct parametric geometries,
- handle scattered and noisy data,
- reduce users' effort in identifying data's connectivity information,
- ensure fairness in the reconstruction process.
- model general n-sided surfaces without using trim curves, and
- integration with downstream design software.

The original ill-posed problems are reformulated into new ones that seek the solution that minimizes specially designed energy functionals. The energy functional is composed of an irregularity measurement of the geometry and a closeness measurement between the given data points and the geometry. The irregularity of the geometry is assessed by not only global measurements such as the integral of squared derivatives of the geometry but

also local measurements such as the squared derivatives discontinuities. The closeness measurement is a weighted sum of the squared minimal distances between the data points and the geometry. This enables the reduction of the input of the connectivity information for the data to a large extent. Successive quadratic programming is used to find the solution for both curve and surface reconstruction.

The novelty of the proposed method is that only partial connectivity information needs to be specified in contrast to the full knowledge of connectivity information assumed in the traditional methods. Using variational principles in curve and surface reconstruction enables the fairness of the geometry to be inherent and allows a seamless integration into a variational surface design system. Using a FE-based parametric geometry representation enables the modeling of more complicated surfaces.

7.2 CONTRIBUTION

The major contribution of the thesis is the development of novel physically-based methods for curve and surface reconstruction. It was shown that the approach can overcome most of the weaknesses of previous approaches.

- *Able to reconstruct parametric curves and surfaces.*

The major distinction between methods that are able to reconstruct parametric geometries and those that cannot is that the resulting geometry is coordinate system invariant. This means that when a coordinate transformation is applied to the data points, the reconstructed surface can be obtained by applying the same transformation to the surface reconstructed from the data before the transformation. Methods developed based on explicit geometry representation do not possess such a property. They are only invariant with respect to a combination of translation and rotation on the x-y plane.

- *Able to reconstruct geometries from scattered and noisy data points with only partial specification of connectivity information.*

This is one of the major contributions of this work. Traditional parametric curve and surface reconstruction methods all require a lot of connectivity information about the

given data points. By using a closeness measurement which is parametrization independent, the requirement for the data's connectivity has been greatly reduced.

- *Able to model general n-sided surfaces without trim curves.*

Using a finite-element mesh as the surface's representation adds the versatility of synthesis of free-form surfaces. A multi-sided surface can be modeled without defining trim curves on the parametric domain. A dense mesh is used for regions with important features and a coarse mesh for plain regions, thus reducing use of computer storage for the degree of freedoms that describes the surface.

- *Incorporate both interpolation and approximation schemes.*

Traditionally the choice of whether to use an interpolation or an approximation scheme depends on the number of unknowns and equations at hand. If the number of equations is not less than the number of unknowns, an approximation scheme is used and it reduces to the interpolation scheme only when the number of the unknowns is the same as the number of the equations. If the number of unknowns is more than the number of equations, infinite solutions exist. Applying the variational principle to curve and surface reconstruction enables the interpolation and approximation schemes be incorporated simultaneously. The interpolation conditions are treated as constraints in the minimization. As long as these constraints can be explicitly expressed as linear combination of the system's degrees of freedom, they can be satisfied exactly. When the number of unknowns is more than the number of equations, the minimization enables one unique solution to be selected from the infinite solutions.

- *Able to be integrated into a variational surface design system.*

A surface design system based on the variational principle has been proven to be able to reduce the users' input for creating and changing the surface shape, sustain both local and global control of the shape and ensure surface's fairness. Using the same principle in curve and surface reconstruction enables the developed method to be easily and seamlessly integrated into the variational surface design system.

- *The existing mesh can be directly used in the finite element analysis.*

The existing mesh can be used directly in engineering analysis using the finite element method or boundary element method. Here, the mesh indicates the collection of triangular surface patches which are mapped from a two dimensional triangulated parametric domain. All the information about the mesh necessary to perform the finite element analysis can be obtained from the surface's underlying mathematical representation. The need to decompose the surface is eliminated.

7.3 CURRENT LIMITATIONS

- *Surface primitives are only C^1 continuous*

The surface primitive used in this work only achieves a C^1 continuous surface. When considering the aerodynamic performance of the surface, a C^2 continuous surface would be better since the C^2 discontinuity could possibly increase the drag coefficient.

- *Inability to model C^1 closed surface without introducing degeneracies.*

The FE-based surface is a mapping from the R^2 domain to the R^3 domain. This mapping can be deemed as a process of stretching, compressing and bending a flat membrane to a curvy one. To form a closed surface, widely separated edges on the parametric domain (the flat membrane) would be mapped (stretched, compressed and bend) to the same curve or even the same point on the closed surface. This would introduce some serious degeneracy problems. One of the typical degeneracy problems is that one of the surface derivatives vanishes or the surface derivatives are linear dependent at some points, which results in difficulties in calculating the normal vectors. Actually this is a common problem that will arise when attempting to represent a closed surface by a single R^2-to-R^3 mapping. This problem can be resolved by using more than one single surface in representing the closed surface. In such a representation, each surface is a single mapping from its own parametric space to the object space. No degeneracies will be caused. However, the degrees of freedom of the adjacent surfaces should be carefully manipulated in order to achieve the continuity at the surfaces' junctions.

7.4 FUTURE DIRECTIONS

This section suggests some possible refinements and extensions of the proposed methods which will increase the depth of the proposed physically-based approach for surface reconstruction and extend the applicability of the variational principle.

7.4.1 Refinements

- *C^2 triangular interpolant*

Though a C^2 triangular interpolant of nodal-basis type is already available: the 55-dof interpolant using information up to the fourth derivatives, the large number of degrees of freedom per element and the high order boundary curve (to the 9th degree) makes its use infeasible. One possible alternative is to use the analogy that was used to develop the 12-dof interpolant: use a lower degree conforming interpolant, for example, the 21-dof interpolant, with a rational correction function which has zero value and zero normal derivatives along the boundary. This correction function probably has singularities in the third derivatives at the vertices, but this singularity should not cause any problems in practical applications.

- *Convexity preservation*

Convexity is an important feature of many surfaces such as automobile and aircraft bodies. Currently the convexity preserving property has not been built into the proposed curve and surface reconstruction methods. However, once the surface is reconstructed, a post process can be applied by imposing the reconstructed surface a small perturbation to satisfy the convexity condition.

7.4.2 Extensions

- *Quadrilateral mesh generation*

One of the advantages of using finite element based geometry representation is that the existing mesh of the final surface can be used in the structural and aerodynamic analysis using FEM or BEM. However, in both of these two popular engineering analysis

methods, quadrilateral elements are generally preferred over triangular elements since empirically they produce better analysis results when the degrees of freedom used in the analysis are approximately the same. Another advantage of using quadrilateral elements is that they are easier to be converted to NURBS surfaces for data exchange to different CAD systems or for the input to cutting path generation software which had been well developed for quadrilateral surfaces.

In representing n-sided surfaces by quadrilateral finite element surfaces, a quadrilateral mesh generation method which is able to control the element's size is essential and needs to be developed.

- *Reconstruction of closed surfaces*

Reconstructing closed surfaces has applications in the medical imaging in which human organs are reconstructed from X-ray or MRI scan pictures. For this kind of applications, a complete mathematical description might not be important. Using a simpler surface representation such as simplicial surfaces and the same variational approach would be an appealing approach. For applications such as representing a function which estimates the pressure, temperature, rainfall, etc. over a closed surface, a mathematical description for the surface is still required.

- *Incorporation of structural and aerodynamic analysis*

The goal of most structural and aerodynamic analysis is to minimize some specific objective functions or to reduce their values under some threshold. For example, in structural analysis, a common goal is to minimize the weight to strength ratio. Minimizing the drag coefficient is very common in aerodynamic analysis. As long as the objective function can be represented as a function of the degrees of freedom (or variables) defining the geometry's shape, the variational principle can come into play. We can use the variational principle as a unifying principle to integrate CAD model construction, aesthetic design and engineering analysis. The functional will become a weighted sum of the measurement of the satisfaction extent of geometric design constraints, aesthetic appearance and structural or aerodynamic performance. Since the evaluation of such an objective function would involve the FEM or BEM analysis, optimization methods which only utilize the function value would be more appropriate.

APPENDIX

Curvature of Curves and Surfaces

This section reviews the basic properties of curvature of parametric curves and surfaces, especially those used in this thesis, and provides the equations for calculating the curvature values. More details of the differential properties of the parametric curves and surfaces can be found in text books such as [do Carmo 76] [Faux 79] [Mortenson 85] [Rogers 90] and [Farin 93].

A.1 CURVATURE OF PARAMETRIC CURVES

A parametric curve in R^3 is a mapping from a one dimensional parametric space to the three dimension object space as

$$w(u) = \begin{bmatrix} x(u) & y(u) & z(u) \end{bmatrix}^T \quad ; \quad u \in [a,b] \subset R \tag{A.1.1}$$

A parametric curve is said to be regular if its tangent vector or the first derivative vector defined by

$$w'(u) = \frac{dw(u)}{du} = \begin{bmatrix} \frac{dx}{du} & \frac{dy}{du} & \frac{dz}{du} \end{bmatrix}^T \tag{A.1.2}$$

has nonzero magnitude for $u \in [a,b]$. The unit tangent vector t is defined as

$$t = \frac{dw}{ds} = \frac{w'}{\|w'\|} \quad ; \quad s = s(u) = \int_a^u \|w'(u)\| du \tag{A.1.3}$$

The derivative of the unit tangent vector with respect to the arc length is called the curvature vector

$$\kappa = \frac{dt}{ds} = \kappa n \tag{A.1.4}$$

where n is the normal vector and κ is the curvature, the magnitude of the curvature vector. The binormal vector b and the torsion vector τ are defined as

$$b = t \times n \quad ; \quad \tau = \frac{db}{ds} = -\tau n \quad , \quad \tau = \text{torsion} \tag{A.1.5}$$

The unit tangent vector, the normal vector and the binormal vector construct a orthogonal frame, called Frenet frame, associated at any point on the curve provided the curvature vector is nonzero.

The curvature vector and its magnitude of a regular space curve can be calculated by applying the chain rule to equation (A.1.3) as

$$\kappa = \kappa n = -\frac{(w' \cdot w'')}{\|w'\|^4} w' + \frac{1}{\|w'\|^2} w'' \quad ; \quad \kappa = \|\kappa\| = \frac{\|w'' \times w'\|}{\|w'\|^3} \tag{A.1.6}$$

The curvature of a space curve is always positive, However, for planar curves, negative signs can be assigned to the curvature when the cross product vector of the first and the second derivative points downward, i.e., to the -z direction. The signed curvature and the normal direction for planar curves are given by

$$\kappa = \frac{x'y'' - x''y'}{(x'^2 + y'^2)^{3/2}} \quad \text{and} \quad n = \frac{(-y', x')}{(x'^2 + y'^2)^{1/2}} \tag{A.1.7}$$

Note that the unit normal vector of planar curves only depends on its first derivative.

The derivative of the curvature vector with respect to arc length can be also calculated by applying the chain rule again to equation (A.1.6)

$$\frac{d\boldsymbol{\kappa}}{ds} = \frac{du}{ds}\frac{d\boldsymbol{\kappa}}{du} = \frac{1}{\|\boldsymbol{w}'\|}\frac{d}{du}\left(-\frac{(\boldsymbol{w}'\cdot\boldsymbol{w}'')}{\|\boldsymbol{w}'\|^4}\boldsymbol{w}' + \frac{1}{\|\boldsymbol{w}'\|^2}\boldsymbol{w}''\right) \tag{A.1.8}$$
$$= c_1\boldsymbol{w}' + c_2\boldsymbol{w}'' + c_3\boldsymbol{w}'''$$

where $c_1 = -\dfrac{\boldsymbol{w}''\cdot\boldsymbol{w}'' + \boldsymbol{w}'\cdot\boldsymbol{w}'''}{\|\boldsymbol{w}'\|^5} + \dfrac{4(\boldsymbol{w}'\cdot\boldsymbol{w}'')^2}{\|\boldsymbol{w}'\|^7}$, $c_2 = -\dfrac{3(\boldsymbol{w}'\cdot\boldsymbol{w}'')^2}{\|\boldsymbol{w}'\|^5}$ and $c_3 = \dfrac{1}{\|\boldsymbol{w}'\|^3}$.

Note that the magnitude of $d\boldsymbol{\kappa}/ds$ is not the same as $d\kappa/ds$. When only the derivative of the curvature is concerned, for space curves, it can be calculated by taking the inner product of $d\boldsymbol{\kappa}/ds$ and the unit normal vector. This can be easily proven in the following

$$\frac{d\boldsymbol{\kappa}}{ds}\cdot\boldsymbol{n} = \frac{d(\kappa\boldsymbol{n})}{ds}\cdot\boldsymbol{n} = \left(\frac{d\kappa}{ds}\boldsymbol{n} + \kappa\frac{d\boldsymbol{n}}{ds}\right)\cdot\boldsymbol{n} \tag{A.1.9}$$
$$= \frac{d\kappa}{ds}\boldsymbol{n}\cdot\boldsymbol{n} + \kappa(\tau\boldsymbol{b} - \kappa\boldsymbol{t})\cdot\boldsymbol{n} = \frac{d\kappa}{ds}$$

For planar curves, $d\kappa/ds$ can be calculated by

$$\frac{d\kappa}{ds} = \frac{x'y''' - x'''y'}{(x'^2 + y'^2)^2} - 3\kappa\frac{x'x'' - y'y''}{(x'^2 + y'^2)^{3/2}} \tag{A.1.10}$$

A.2 CURVATURE OF PARAMETRIC SURFACES

A parametric surface in R^3 is a mapping from a two dimensional parametric space to the three dimension object space as

$$w(u,v)=\begin{bmatrix}x(u,v) & y(u,v) & z(u,v)\end{bmatrix}^{T} \quad ; \quad \boldsymbol{u}=\begin{bmatrix}u & v\end{bmatrix}^{T} \in \Omega_s \subset R^2 \tag{A.2.1}$$

where Ω_s denotes a finite region in R^2. A parametric surface is said to be regular if its tangent plane is everywhere uniquely defined. This is equivalent to saying that the unit normal vector

$$\boldsymbol{n}=\frac{\boldsymbol{w}_u \times \boldsymbol{w}_v}{\left\|\boldsymbol{w}_u \times \boldsymbol{w}_v\right\|} \tag{A.2.2}$$

is everywhere uniquely defined. The surface's normal is indeterminate when $\left\|\boldsymbol{w}_u \times \boldsymbol{w}_v\right\|=0$. This occurs for points with true geometric discontinuity like a point lying on a cusp, a ridge or a self-intersection of the surface, and for points where the two parametric derivatives are parallel or one of them vanishes. In general, the latter cases can be remedied by using Taylor expansion on that point.

A regular curve $\boldsymbol{u}=\boldsymbol{u}(t)$ in the u-v plane defines a regular curve $\boldsymbol{w}(\boldsymbol{u}(t))$ on the surface. The squared arc length of the surface curve is given by

$$ds^2=\left\|\boldsymbol{w}_t\right\|^2 dt^2=\begin{bmatrix}u_t & v_t\end{bmatrix}\begin{bmatrix}\boldsymbol{w}_u \cdot \boldsymbol{w}_u & \boldsymbol{w}_u \cdot \boldsymbol{w}_v \\ \boldsymbol{w}_u \cdot \boldsymbol{w}_v & \boldsymbol{w}_v \cdot \boldsymbol{w}_v\end{bmatrix}\begin{bmatrix}u_t \\ v_t\end{bmatrix}dt^2 \tag{A.2.3}$$

This equation is known as the first fundamental form of the surface. The curvature of the surface curve is defined as its second derivative with respect to the arc length as

$$\frac{d^2\boldsymbol{w}}{ds^2}=\kappa \boldsymbol{m}=\boldsymbol{w}_{uu}\left(\frac{du}{ds}\right)^2+2\boldsymbol{w}_{uv}\left(\frac{du}{ds}\right)\left(\frac{dv}{ds}\right)+\boldsymbol{w}_{vv}\left(\frac{dv}{ds}\right)^2+\boldsymbol{w}_u\left(\frac{d^2u}{ds^2}\right)+\boldsymbol{w}_u\left(\frac{d^2v}{ds^2}\right) \tag{A.2.4}$$

where $\boldsymbol{m}$ is the unit normal vector of the surface curve. Note that the surface normal vector $\boldsymbol{n}$ which is orthogonal to $\boldsymbol{w}_u$ and $\boldsymbol{w}_u$ will not generally be aligned with the surface

curve's normal vector. Taking the inner product of $\kappa \boldsymbol{m}$ with the surface normal $\boldsymbol{n}$ and using $\boldsymbol{w}_u \cdot \boldsymbol{n} = \boldsymbol{w}_v \cdot \boldsymbol{n} = 0$ yields

$$\kappa \boldsymbol{m} \cdot \boldsymbol{n} = \boldsymbol{n} \cdot \boldsymbol{w}_{uu}\left(\frac{du}{ds}\right)^2 + 2\boldsymbol{n} \cdot \boldsymbol{w}_{uv}\left(\frac{du}{ds}\right)\left(\frac{dv}{ds}\right) + \boldsymbol{n} \cdot \boldsymbol{w}_{vv}\left(\frac{dv}{ds}\right)^2 \tag{A.2.5}$$

or

$$\kappa \boldsymbol{m} \cdot \boldsymbol{n} ds^2 = \begin{bmatrix} du & dv \end{bmatrix} \begin{bmatrix} \boldsymbol{n} \cdot \boldsymbol{w}_{uu} & \boldsymbol{n} \cdot \boldsymbol{w}_{uv} \\ \boldsymbol{n} \cdot \boldsymbol{w}_{uv} & \boldsymbol{n} \cdot \boldsymbol{w}_{vv} \end{bmatrix} \begin{bmatrix} du \\ dv \end{bmatrix} \tag{A.2.6}$$

This expression is known as the second fundamental form of the surface.

When a surface curve is generated by intersecting the surface with a plane containing the surface's normal vector at $\boldsymbol{P}_0$, the surface curve's unit normal vector at $\boldsymbol{P}_0$ will align with the surface's normal vector. Such a curve is called a normal section curve and the curvature at $\boldsymbol{P}_0$ is called normal curvature. The normal curvature at a point $\boldsymbol{P}_0$ lying on the surface $w(u,v)$ in the direction specified by (u_t, v_t) is given by

$$\kappa = -\frac{Lu_t^2 + 2Mu_t v_t + Nv_t^2}{Eu_t^2 + 2Fu_t v_t + Gv_t^2} \tag{A.2.7}$$

where L, M, N, E, F, and G are the first and second fundamental coefficients of $w(u,v)$ and are defined as

$$\begin{aligned} E &= \boldsymbol{w}_u \cdot \boldsymbol{w}_u \quad , \quad F = \boldsymbol{w}_u \cdot \boldsymbol{w}_v \quad , \quad G = \boldsymbol{w}_v \cdot \boldsymbol{w}_v \\ L &= \boldsymbol{n} \cdot \boldsymbol{w}_{uu} \quad , \quad M = \boldsymbol{n} \cdot \boldsymbol{w}_{uv} \quad , \quad N = \boldsymbol{n} \cdot \boldsymbol{w}_{vv} \end{aligned} \tag{A.2.8}$$

The convention is used that the normal curvature is positive when the center of curvature and the surface's normal lie on opposite sides of the surface. By rotating the intersecting plane about the surface's normal, infinite number of normal section curves and thus the normal curvatures can be generated. The normal curvature attains extremum values when $\partial\kappa/\partial u_t = 0$ and $\partial\kappa/\partial v_t = 0$. These conditions can be written as

$$\begin{aligned} (L + \kappa E)u_t + (M + \kappa F)v_t &= 0 \\ (M + \kappa F)u_t + (N + \kappa G)v_t &= 0 \end{aligned} \tag{A.2.9}$$

For these two equations to possess a consistent solution, κ must satisfy

$$\kappa^2 - 2H\kappa + K = 0 \tag{A.2.10}$$

where the coefficient K and H are defined as the Gaussian curvature and mean curvature, and are given as

$$K = \frac{LN - M^2}{EG - F^2} \quad , \quad H = \frac{2FM - (EN + GL)}{2(EG - F^2)} \tag{A.2.11}$$

The two solutions of equation (A.2.10) are the extremum values of the curvature which represent the upper and lower bounds of the curvature at a given point on the surface. They are called the principal curvatures and are given as

$$\begin{aligned} \kappa_{max} &= H + \sqrt{H^2 - K} \\ \kappa_{min} &= H - \sqrt{H^2 - K} \end{aligned} \tag{A.2.12}$$

It is easy to verify that

$$K = \kappa_{min}\kappa_{max} \quad \text{and} \quad H = \frac{\kappa_{min} + \kappa_{max}}{2} \tag{A.2.13}$$

Positive Gaussian curvature indicates that the two principal curvatures have the same sign. This happens for elliptic surfaces like spheres. Negative Gaussian curvature indicates hyperbolic surfaces such as a saddle point where the surface is convex in one direction and concave in another. The surface's shape is called parabolic when the Gaussian curvature is zero as for cylinders and ruled surfaces. Surfaces that have zero Gaussian curvature everywhere are called developable which means they can be laid flat on a plane without stretching or tearing.

Other curvatures that have been used are absolute curvature, primary and secondary curvatures. Absolute curvature is the sum of the absolute values of the two principal curvatures. The primary and secondary curvatures are the normal curvatures in the direction of the isoparametric curves. The primary curvature is the normal curvature in the u direction and the secondary curvature in the v direction. They can be calculated by

$$\kappa_u = -\frac{L}{E} \quad , \quad \kappa_v = -\frac{N}{G} \tag{A.2.14}$$

A summary of formula for calculating various surface curvatures are listed below.

Table A.1: Summary of the various surface curvatures

Curvature	Formula
maximum curvature	$\kappa_{max} = H + \sqrt{H^2 - K}$
minimum curvature	$\kappa_{min} = H - \sqrt{H^2 - K}$
Gaussian curvature	$K = \frac{LN - M^2}{EG - F^2} = \kappa_{min}\kappa_{max}$
mean curvature	$H = \frac{2FM - (EN + GL)}{2(EG - F^2)} = \frac{\kappa_{min} + \kappa_{max}}{2}$
absolute curvature	$A = \frac{\lvert\kappa_{min}\rvert + \lvert\kappa_{max}\rvert}{2}$
primary curvature	$\kappa_u = -\frac{L}{E}$
secondary curvature	$\kappa_v = -\frac{N}{G}$

The directions in which the normal curvature takes its maximum and minimum values are called the principal directions. The principal directions can be obtained by substituting κ_{max} and κ_{min} into equation (A.2.12) and finding the ratio of u_t and v_t or solving the following equation directly

$$\det\begin{bmatrix} \lambda^2 & -\lambda & 1 \\ E & F & G \\ L & M & N \end{bmatrix} = 0 \quad ; \quad \lambda = \frac{v_t}{u_t} \tag{A.2.15}$$

If $\frac{L}{E} = \frac{M}{F} = \frac{N}{G} = c$, then the above equation has an infinite number of solutions for λ. This means that the normal curvature κ is independent of the direction. At such a point, the two principal curvatures have the same magnitude as $\kappa_{max} = \kappa_{min} = -c$. Such a point is called an umbilical, or spherical point since the surface locally approximates a sphere at that point. If the principal curvatures are not equal, it can be shown that the two principal directions are orthogonal. A curve lying on the surface whose tangent is always in a principal direction is called a curvature line or line of curvature.

References

[Akima 78] Akima, H., *A method of bivariate interpolation and smooth surface fitting for irregularly distributed data points*, ACM Transactions on Mathematical Software, **4**(2): pp. 148-159, 1978.

[Alfeld 85] Alfeld, P., *Derivative generation from multivariate scattered data by functional minimization*, Computer Aided Geometric Design, **2**: pp. 281-296, 1985.

[Alfeld 88] Alfeld, P., *Scattered data interpolation in three or more variables.* in Mathematical Methods in Computer Aided Geometric Design, T. Lyche and L. L. Schumaker ed., pp. 1-33, Norway: Academic Press, Inc., 1988.

[Alourdas 89] Alourdas, P. G., *Shape creation, interrogation and fairing using B-splines*. 1989, Massachusetts of Technology, Ocean Engineering Department.

[Andersson 88] Andersson, E., *et al., Automatic construction of surface with prescribed shape*, Computer-Aided Design, **20**, 1988.

[Bajaj 92] Bajaj, C. L., *Surface fitting using implicit algebraic surface patches,* in *Topics in Surface Modeling,* H. Hagen, Editor(s). 1992, pp. 23-52.

[Bardis 90] Bardis, L. and Patrikalakis, N. M., *Surface approximation with rational B-splines*, Engineering with Computers, **6**(4): pp. 223-235, 1990.

[Barnhill 73] Barnhill, R. E., Birkhoff, G., and Gordon, W. J., *Smooth interpolation in triangles*, Journal of Approximation Theory, **8**: pp. 114-128, 1973.

[Barnhill 77] Barnhill, R. E., *Representation and approximation of surfaces,* in *Mathematical Software III.* 1977.

[Barnhill 88] Barnhill, R. E., *et al., Twists, curvatures and surface interrogation*, Computer-Aided Design, **20**(6): pp. 341-346, 1988.

[Barr 84] Barr, A. H., *Global and local deformations of solid primitives.* in ACM, SIGGRAPH, Computer Graphics, Vol. 18, pp. 21-30, 1984.

[Barrow 81] Barrow, H. G. and Tenenbaum, J. M., *Computational vision.* in Proceedings of IEEE, Vol. 69, pp. 572-595, 1981.

[Bazaraa 93] Bazaraa, M. S., Sherali, H. D. and Shetty, C. M., *Nonlinear Programming Theory and Algorithms*, 1993, New York: John Wiley & Sons.

[Beck 86] Beck, J. M., Farouki, R. T. and Hinds, J. K., *Surface analysis methods*, IEEE Computer Graphics and Applications, 1986.

[Bell 69] Bell, K., *A refined triangular plate bending finite element*, International Journal for Numerical Methods in Engineering, **1**: pp. 101-122, 1969.

[Berger 92] Berger, D. N. and Kallay, M., *Computing the unit normal on a degenerate edge*, Computer-Aided Design, **24**(7): pp. 395-396, 1992.

[Birkhoff 65] Birkhoff, G. and de Boor, C. R., *Nonlinear interpolation by splines, pseudosplines and elastica*. 1965, GM Research Laboratory.

[Bloor 89] Bloor, M. I. G. and Wilson, M. J., *Generating blend surfaces using partial differential equations*, Computer-Aided Design, **21**(3): pp. 165-171, 1989.

[Boehm 88] Boehm, W., *Visual continuity*, Computer-Aided Design, **20**(6), 1988.

[Böhm 84] Böhm, W., Farin, G. and Kahmann, J., *A survey of curve and surface methods in CAGD*, Computer Aided Geometric Design, **1**(1): pp. 1-60, 1984.

[Bolle 91] Bolle, R. M. and Vemuri, B. C., *On three-dimensional surface reconstruction methods*, IEEE Transactions on Pattern Analysis and machine Intelligence, **13**(1): pp. 1-13, 1991.

[Cavendish 92] Cavendish, J. C. and Marin, S. P., *A procedural feature-based approach for designing functional surfaces,* in *Topics in Surface Modeling*. 1992, pp. 145-168.

[Celniker 90a] Celniker, G., *ShapeWright: finite element based free-form shape design*. 1990, Mechanical Engineering Department, M.I.T.

[Celniker 90b] Celniker, G. and Gossard, D. C., *Continuous deformable curves and their application to fairing 3D geometry*. in Proceedings of the IFIP TC S/WG 5.2 Working conference on geometric modeling, pp. 16-21, Zensselarville, N. Y.: 1990.

[Celniker 92] Celniker, G. and Welch, W., *Linear constraints for deformable B-spline surface*. in Symposium on Interactive 3D Graphics, 1992.

[Chan 90] Chan, A. K., Chui, C. K. and Guan, L. T., *Radial basis function approach to interpolation of large reflecting surfaces*. in SPIE Vol. 1251, Curves and Surfaces in Computer Vision and Graphics, pp. 62-72, 1990.

[Chou 93] Chou, J. J. and Blake, M. W., *Planar cubics through a point in a direction*, Computer-Aided Design, **25**(6): pp. 348-354, 1993.

[Cline 88] Cline, H. E., *et al.*, *Two algorithms for the three-dimensional reconstruction of tomogram*, Medical Physics, **15**(3): pp. 320-327, 1988.

[Cohen 88] Cohen, E. and O'Dell, C. L., *A data dependent parametrization for spline approximation*. in Mathematical Methods in Computer Aided Geometric Design, T. Lyche and L. L. Schumaker ed., pp. 155-166, Norway: Academic Press, Inc., 1988.

[Coons 67] Coons, S. A., *Surfaces for the computer-aided design of space forms*. 1967, M.I.T.

[Coquillart 87] Coquillart, S., *A control-point based sweeping technique*, IEEE Computer Graphics and Applications, **7**(11), 1987.

[Coquillart 90] Coquillart, S., *Extended free-Form deformation: A sculpturing tool for 3D geometric modeling*. in SIGGRAPH, Computer Graphics, Vol. 24, 1990.

[Crandall 68] Crandall, S. H., *et al.*, *Dynamics of Mechanical and Electromechanical Systems*, 1968, New York: McGraw-Hill.

[Cuthill 69] Cuthill, E. and McKee, J., *Reducing the bandwidth of sparse symmetric matrices*. in ACM National Conference, pp. 157-172, New York: 1969.

[de Boor 78] de Boor, C., *A practical guide to splines*, 1978, Springer-Verlag, Berlin, FRG.

[Dill 81] Dill, J. C., *An application of color graphics to the display of surface curvature*. in Computer Graphics, Vol. 15, pp. 153-161, 1981.

[do Carmo 76] do Carmo, M. P., *Differential Geometry of Curves and Surfaces*, 1976, Prentice Hall.

[Dyn 83] Dyn, N., Levin, D. and Rippa, S., *Surface interpolation and smoothing by "thin plate" splines,* in *Approximation Theory IV.* 1983, pp. 445-449.

[Earnshaw 71] Earnshaw, J. L. and Yuille, I. M., *A method of fitting parametric equations for curves and surfaces to sets of points defining them approximately*, Computer-Aided Design, **3**: pp. 19-22, 1971.

[Epstein 76] Epstein, M. P., *On the influence of parametrization in parametric interpolation*, SIAM, J. of Numerical Analysis, **13**(2), 1976.

[Fang 92a] Fang, L. and Gossard, D. C., *Fitting 3D curves to unorganized data points using deformable curves*. in the Tenth International Conference on Computer Graphics: Visual Computing, T. L. Kunii ed., Tokyo: Springer-Verlag, 1992.

[Fang 92b] Fang, L. and Gossard, D. C., *Reconstruction of smooth parametric surfaces from unorganized data points*. in SPIE Conference on "Curves and Surfaces in Computer Vision and Graphics III", Vol. 1830, J. D. Warren ed., pp. 226-236, Boston: Springer-Verlag, 1992.

[Fang 94] Fang, L. and Gossard, D. C., *Multi-dimensional curve fitting to unorganized data points by nonlinear minimization*, Computer-Aided Design, **26**, 1994. (to appear)

[Farin 85] Farin, G., *A modified Clough-Tocher interpolant*, Computer Aided Geometric Design, **2**: pp. 19-27, 1985.

[Farin 93] Farin, G., *Curves and Surfaces for Computer Aided Geometric Design, A Practical Guide*, 3rd ed. 1993, Academic Press, Inc.

[Farin 87] Farin, G., *et al.*, *Fairing cubic B-spline curves*, Computer Aided Geometric Design, **4**: pp. 91-103, 1987.

[Faux 79] Faux, I. D. and Pratt, M. J., *Computational Geometry for Design and Manufacture*, 1979, Ellis Horwood Ltd.

[Foley 86] Foley, T. A., *Scattered data interpolation and approximation with error bounds*, Computer Aided Geometric Design, **3**: pp. 163-177, 1986.

[Foley 87a] Foley, T. A., *Interpolation with interval and point tension controls using cubic weighted n-splines*, ACM Transactions on Mathematical Software, **13**: pp. 68-96, 1987.

[Foley 87b] Foley, T. A., *Weighted bicubic spline interpolation to rapidly varying data*, ACM Transactions on Graphics, **6**(1): pp. 1-18, 1987.

[Foley 89] Foley, T. A. and Nielson, G. M., *Knot selection for parametric spline interpolation*. in Mathematical Methods in Computer Aided Geometric Design, T. Lyche and L. L. Schumaker ed., pp. 261-271, Norway: Academic Press, Inc., 1989.

[Forsey 88] Forsey, D. R. and Bartels, R. H., *Hierarchical B-spline refinement*. in ACM SIGGRAPH: Computer Graphics, Vol. 22, pp. 205-212, Atlanta: 1988.

[Franke 79] Franke, R., *A critical comparison of some methods for interpolation of scattered data*. 1979, Naval Postgraduate School.

[Franke 82] Franke, R., *Scattered data interpolation: test of some methods*, Mathematics of Computation, **38**(157): pp. 181-200, 1982.

[Franke 86] Franke, R., *Recent advances in the approximation of surfaces from scattered data,* in *Topics in Multivariable Approximation*. 1986, pp. 79-98.

[Franke 87] Franke, R. and Schumaker, L. L., *A bibliography of multivariate approximation,* in *Topics in Multivariate Approximation,* C. K. Chui, L. L. Schumaker, and F. I. Utreras, Editor(s). 1987, Academic Press: pp. 275-335.

[Franke 91] Franke, R. and Nielson, G. M., *Scattered data interpolation and applications: a tutorial and survey,* in *Geometric Modeling, methods*

and Applications, H. Hagen and D. Roller, Editor(s). 1991, Springer-Verlag: pp. 131-160.

[Gibbs 76] Gibbs, N. E., William G. Poole, J. and Stockmeyer, P. K., *An algorithm for reducing the bandwidth and profile for a sparse matrix*, SIAM J. Numer. Anal., **13**(2): pp. 236-250, 1976.

[Golomb 82] Golomb, M. and Jerome, J., *Equilibria of the curvature functional and manifolds of nonlinear interpolating spline curves*, SIAM J. Math. Anal., **w13**(3): pp. 421-458, 1982.

[Gregory 80] Gregory, J. A. and Charrot, P., *A C^1 triangular interpolant patch for computer-aided geometric design*, Computer Graphics and Image Processing, **13**: pp. 80-87, 1980.

[Gregory 85] Gregory, J. A., *Interpolation to boundary data on the simplex*, Computer Aided Geometric Design, **2**: pp. 43-52, 1985.

[Grimson 81] Grimson, W. E. L., *A computer implementation of a theory of human stereo vision*, Phil. Trans. Roy. Soc. London B, **298**: pp. 395-427, 1981.

[Grossman 77] Grossman, M., *Parametric curve fitting*, Computer Journal, **14**(2), 1977.

[Gürsoy 92] Gürsoy, H. N. and Patrikalakis, N. M., *An automatic coarse and fine surface mesh generation scheme based on medial axis transformation: part I Algorithms*, Engineering with Computer, **8**: pp. 121-137, 1992.

[Gürsoy 92] Gürsoy, H. N. and Patrikalakis, N. M., *An automatic coarse and fine surface mesh generation scheme based on medial axis transformation: part II Implementation*, Engineering with Computer, **8**: pp. 179-196, 1992.

[Hagen 85] Hagen, H., *Geometric spline curves*, Computer Aided Geometric Design, **2**: pp. 223-227, 1985.

[Hagen 87] Hagen, H. and Schulze, G., *Automatic smoothing with geometric surface patches*, Computer Aided Geometric Design, **4**: pp. 231-235, 1987.

[Hagen 88] Hagen, H. and Pottmann, H., *Curvature continuous triangular interpolants.* in Mathematical Methods in Computer Aided Geometric Design, T. Lyche and L. L. Schumaker ed., pp. 373-384, Norway: Academic Press, Inc., 1988.

[Hagen 92] Hagen, H., *et al., Surface interrogation algorithms*, IEEE Computer Graphics and Applications, **12**(5): pp. 53-60, 1992.

[Han 92] Han, W. and Jou, E., *On the computation of minimal-energy splines: convergence analysis*, Applied Mathematics and Computation, **47**: pp. 1-13, 1992.

[Harder 72] Harder, R. L. and Desmarais, R. N., *Interpolating using surface splines*, Journal of Aircraft, **9**: pp. 189-197, 1972.

[Hardy 71] Hardy, R. L., *Multiquadric equations of topography and other irregular surfaces*, J. Geophys. Res., **76**: pp. 1905-1915, 1971.

[Hartley 78] Hartley, P. and Judd, C., *Parametrization of Bézier-type B-spline curves*, Computer-Aided Design, **10**(2): pp. 130-134, 1978.

[Hartley 80] Hartley, P. and Judd, C., *Parametrization and shape of B-spline curves*, Computer-Aided Design, **12**(5): pp. 235-238, 1980.

[Hayes 87] Hayes, J. G., *Fitting surfaces to data,* in *The Mathematics of Surfaces II*. 1987, pp. 17-37.

[Hayes 73] Hayes, J. G. and Halliday, J., *The least-squares fitting of cubic spline surfaces to general data sets*, Journal of the Institute of Mathematics and it's Applications, **14**: pp. 89-103, 1973.

[Herron 85] Herron, G., *Smooth closed surfaces with discrete triangular interpolants*, Computer Aided Geometric Design, **2**: pp. 297-306, 1985.

[Hoppe 92] Hoppe, H., *et al., Surface reconstruction from unorganized points*. in ACM, SIGGRAPH, 1992.

[Hoppe 93] Hoppe, H., *et al., Mesh optimization*. 1993, University of Washington, Dept. of Computer Science and Engineering.

[Horn 83] Horn, B. K. P., *The curve of least energy*, ACM Transactions on Mathematical Software, **9**(4): pp. 441-460, 1983.

[Hosaka 69] Hosaka, M., *Theory of curve and surface synthesis and their smooth fitting*, Information Processing in Japan, **9**: pp. 60-68, 1969.

[Hosaka 78] Hosaka, M. and Kimura, F., *Synthesis methods of curves and surfaces in interactive CAD*. in IEEE Conf. Proc.: Interactive Techniques in Computer Aided Design, pp. 151-155, 1978.

[Hoschek 88] Hoschek, J., *Intrinsic parametrization for approximation*, Computer Aided Geometric Design, **5**, 1988.

[Hoschek 93] Hoschek, J. and Lasser, D., *Fundamentals of Computer Aided Geometric Design*, 1993, A K Peters, Ltd.

[Hottel 91] Hottel, G. R., *et al., Praxiteles: a geometric modeling and interrogation system*. in Marine Computers, 1991, Burlington, MA: 1991.

[Irons 69] Irons, B. M., *A conforming quartic triangular element for plate bending*, International Journal for Numerical Methods in Engineering, **1**: pp. 29-45, 1969.

[Jackson 88] Jackson, I. R. H., *Radial basis functions: a survey and new results,* in *Mathematics of Surfaces III*. 1988, pp. 115-133.

[Jensen 87] Jensen, T., *Assembling triangular and rectangular patches and multivariate splines,* in *Geometric Modeling: Algorithms and New Trends,* G. Farin, Editor(s). 1987, pp. 203-220.

[Jerome 75] Jerome, J. W., *Smoothing interpolating curves of prescribing length and minimum curvature.* in Proceeding of the American Mathematical Society, Vol. 51, pp. 62-66, 1975.

[Jou 92] Jou, E. and Han, W., *Minimal-energy splines with various end constraints,* in *Curve and Surface Design,* H. Hagen, Editor(s). 1992, pp. 23-40.

[Kallay 90] Kallay, M. and Ravani, B., *Optimal twist vectors as a tool for interpolating a network of curves with a minimum energy surface*, Computer Aided Geometric Design, **7**: pp. 465-473, 1990.

[Kallay 91] Kallay, M., *Approximating a composite cubic curve by one with fewer pieces*, Computer-Aided Design, **23**, 1991.

[Kallay 93] Kallay, M., *Constrained optimization in surface design.* in IFIP TC5/WG5.10 Working Conference on Modeling in Computer Graphics, 1993.

[Kass 88] Kass, M., Witkin, A. and Terzopoulos, D., *Snakes: active contour models*, International Journal of Computer Vision, : pp. 321-331, 1988.

[Kaufmann 88] Kaufmann, E. and Klass, E., *Smoothing surfaces using reflection lines for families of splines*, Computer-Aided Design, **20**(6): pp. 312-316, 1988.

[Kjellander 83a] Kjellander, J. A. P., *Smoothing of bicubic parametric surfaces*, Computer-Aided Design, **15**(5): pp. 288-293, 1983.

[Kjellander 83b] Kjellander, J. A. P., *Smoothing of cubic parametric splines*, Computer-Aided Design, **15**(3): pp. 175-179, 1983.

[Klass 80] Klass, R., *Correction of local surface irregularities using reflection lines*, Computer-Aided Design, **12**(2): pp. 73-77, 1980.

[Lancaster 86] Lancaster, P. and Salkauskas, K., *Curve and Surface Fitting, An Introduction*, 1986, Academic Press.

[Lawson 77] Lawson, C. L., *Software for C^1 surface interpolation,* in *Mathematical Software III*. 1977.

[Lee 89] Lee, E. T. Y., *Choosing nodes in parametric curve interpolation*, Computer-Aided Design, **21**(6), 1989.

[Lee 92] Lee, E. T. Y., *Corners, cusps, and parametrizations: variations on a theorem of Epstein*, SIAM Journal of Numerical Analysis, **29**, 1992.

[Lorensen 87] Lorensen, W. E., *Marching cubes: a high resolution 3D surface construction algorithm*, ACM Computer Graphics, **21**(4): pp. 163-169, 1987.

[Lott 88] Lott, N. J. and Pullin, D. I., *Method for fairing B-spline surfaces*, Computer-Aided Design, **20**(10): pp. 597-604, 1988.

[Maekawa 93] Maekawa, T. and Patrikalakis, N. M., *Computation of singularities and intersections of offsets of planar curves*, Computer Aided Geometric Design, **10**(5): pp. 407-429, 1993.

[Mann 92] Mann, S., *et al.*, *A survey of parametric scattered data fitting using triangular interpolants,* in *Curve and Surface Design,* H. Hagen, Editor(s). 1992,

[Marin 84] Marin, S. P., *An approach to data parametrization in parametric cubic spline interpolation problems*, Journal of Approximation Theory, **41**: pp. 64-86, 1984.

[McConalogue 70] McConalogue, D., *A quasi-intrinsic scheme for passing a smooth curve through a discrete set of points.*, The Computer Journal, **13**: pp. 392-396, 1970.

[Mehlum 69] Mehlum, E., *Curve and surface fitting based on variational criteria for smoothness*. 1969, Central Institute for Industrial Research, Oslo, Norway.

[Meyers 92] Meyers, D., Skinner, S. and Sloan, K., *Surfaces from contours*, ACM Transactions on Graphics, **11**(3): pp. 228-258, 1992.

[Moreton 91] Moreton, H. P. and Séquin, C. H., *Surface design with minimum energy networks*. in Proceeding of the Symposium on Solid Modeling Foundations and CAD/CAM Applications, pp. 291-301, Austin, Texas: ACM, New York, 1991.

[Moreton 92] Moreton, H. and Séquin, C., *Functional optimization for fair surface design*. in ACM SIGGRAPH: Computer Graphics, Vol. 26, pp. 167-176, Chicago: 1992.

[Mortenson 85] Mortenson, M. E., *Geometric Modeling*, 1985, John Wiley & Sons, Inc.

[Munchmeyer 87] Munchmeyer, F., *On surface imperfections,* in *Mathematics of Surfaces II,* R. Martin, Editor(s). 1987, Oxford University Press: pp. 459-474.

[Nielson 74] Nielson, G. M., *Some piecewise polynomial alternatives to splines under tension,* in *Computer Aided Geometric Design,* R. E. Barnhill and R. F. Riesenfeld, Editor(s). 1974, Academic Press.

[Nielson 79] Nielson, G. M., *The side-vertex method for interpolation in triangle*, Journal of Approximation Theory, **25**: pp. 318-336, 1979.

[Nielson 83] Nielson, G. M., *A method for interpolating scattered data based upon a minimum norm network*, Mathematics of Computation, **40**(161): pp. 253-271, 1983.

[Nielson 87] Nielson, G. M., *A transfinite visually continuous triangular interpolant,* in *Geometric Modeling: Algorithms and New Trends.* 1987.

[Nowacki 88] Nowacki, H., Liu, D. and Lu, X., *Mesh fairing GC^1 surface generation method.* in Theory and Practice of Geometric Modeling, W. Straβer and H.-P. Seidel ed., pp. 93-108, 1988.

[Nowacki 92] Nowacki, H., P. D. Kaklis, and J. Weber, *Curve mesh fairing and GC^2 surface interpolation*, Mathematical Modeling and Numerical Analysis, **26**(1): pp. 113-135, 1992.

[Ohlin_1987] Ohlin, Ir. S. C., *Splines for engineers.* in Eurographics'87, Proceedings of the European Computer Graphics COnference and Exhibition, G. Maréchal ed., pp. 555-565, Amsterdam, The Netherlands: 1987.

[Piper 87] Piper, B. R., *Visually smooth interpolation with triangular Bézier patches,* in *Geometric Modeling: Algorithms and New Trends,* G. Farin, Editor(s). 1987, pp. 221-233.

[Poeschl 84] Poeschl, T., *Detecting surface irregularities using isophotes*, Computer Aided Geometric Design, **1**(2): pp. 163-168, 1984.

[Pottmann 90] Pottmann, H., *Smooth curves under tension*, Computer-Aided Design, **22**(4): pp. 241-245, 1990.

[Pottmann 91] Pottmann, H., *Scattered data interpolation based upon generalized minimum norm network*, Constructive Approximation, **7**: pp. 247-256, 1991.

[Pratt 85] Pratt, M. J., *Smooth parametric surface approximations to discrete data*, Computer Aided Geometric Design, **2**: pp. 165-171, 1985.

[Reinsch 67] Reinsch, C. H., *Smoothing by spline functions*, Numerische Mathematik, **10**: pp. 177-183, 1967.

[Reklaitis 83] Reklaitis, G. V., Ravindran, A. and Ragsdell, K. M., *Engineering Optimization*, 1983, John Wiely and Sons.

[Renz 82] Renz, W., *Interactive smoothing of digitized point data*, Computer-Aided Design, **14**(5), 1982.

[Riesenfeld 89] Riesenfeld, R. F., *Design tools for shaping spline modes,* in *Mathematical Methods in Computer Aided Geometric Design,* T. Lyche and L. L. Schumaker, Editor(s). 1989, pp. 499-519.

[Rogers 90] Rogers, D. F. and Adams, J. A., *Mathematical Elements for Computer Graphics*, 2nd ed. 1990, McGraw-Hill.

[Sachs 91] Sachs, E., Roberts, A. and Stoops, D., *3-Draw: A tool for designing 3D shapes*, IEEE Computer Graphics and Applications, 1991.

[Sapidis 88] Sapidis, N. S. and Kaklis, P. D., *An algorithm for constructing convexity and monotonicity-preserving splines in tension*, Computer Aided Geometric Design, **5**: pp. 127-137, 1988.

[Sapidis 90] Sapidis, N. and Farin, G., *Automatic fairing algorithm for B-spline curves*, Computer-Aided Design, **22**(2): pp. 121-129, 1990.

[Sarkar 91] Sarkar, B. and Menq, C.-H., *Smooth-surface approximation and reverse engineering*, Computer-Aided Design, **23**(9): pp. 623-628, 1991.

[Schumaker 76] Schumaker, L. L., *Fitting surfaces to scattered data,* in *Approximation Theory II*. 1976, pp. 203-289.

[Schweikert 66] Schweikert, D. G., *An interpolation curve using a spline in tension*, J. Math. Phys., **45**: pp. 312-317, 1966.

[Sederberg 86] Sederberg, T. W. and Parry, S. R., *Free-form deformation of solid geometric models*. in ACM SIGGRAPH: Computer Graphics, Vol. 20, pp. 151-160, Dallas: 1986.

[Seidenberg 92] Seidenberg, L. R., Jerard, R. B. and Magewick, J., *Surface curvature analysis using color*. in Visualization 92 Conference, pp. 19-23, Boston: 1992.

[Sherbrooke 93] Sherbrooke, E. C. and Patrikalakis, N. M., *Computation of the solutions of nonlinear polynomial systems*, Computer Aided Geometric Design, **10**(5): pp. 379-405, 1993.

[Shimada 93] Shimada, K., *Physically-based Mesh generation: automated triangulation of surfaces and volumes via bubble packing*. 1993, MIT.

[Shirman 87] Shirman, L. A. and Séquin, C. H., *Local surface interpolation with Bézier patches*, Computer Aided Geometric Design, **4**: pp. 279-295, 1987.

[Sinha 91] Sinha, S. S., *Surface Reconstruction in Computer Vision*. 1991, Univ. of Michigan.

[Szeliski 89] Szeliski, R., *Fast surface interpolation using hierarchical basis functions*. in IEEE Proc. Conf. Computer Vision and Pattern Recognition, pp. 222-228, 1989.

[Tanaka 92] Tanaka, H. T. and Kishino, F., *Recovering and visualizing complex shapes from range data*. in The Tenth International Conference on Computer Graphics, T. L. Kunii ed., pp. 331-348, Tokyo: Springer-Verlag, 1992.

[Terzopoulos 86] Terzopoulos, D., *Regularization of inverse visual problems involving discontinuities*, IEEE Transactions on Pattern Analysis and machine Intelligence, **8**(4): pp. 413-424, 1986.

[Terzopoulos 87] Terzopoulos, D., *et al.*, *Elastic deformable models*. in ACM SIGGRAPH, Computer Graphics, Vol. 21, pp. 205-214, Anaheim: 1987.

[Terzopoulos 88] Terzopoulos, D., *The computation of visible-surface representation*, IEEE Transactions on Pattern Analysis and machine Intelligence, **10**(4): pp. 417-438, 1988.

[Terzopoulos 91] Terzopoulos, D. and M. Vasilescu, *Sampling and reconstruction with adaptive meshes*. in IEEE Proc. Conf. Computer Vision and Pattern Recognition, pp. 70-75, 1991.

[Tikhonov 77] Tikhonov, S. V. and Aresnin, V. A., *Solutions of Ill-posed Problems*, 1977, Washington, D. C.: Winston & Sons.

[Tiller 83] Tiller, W., *Rational B-splines for curve and surface representation*, IEEE Computer Graphics and Applications, : pp. 61-69, 1983.

[Wang 93] Wang, F.-C. and Yang, D. C. H., *Nearly arc-length parametrized quintic-spline interpolation for precision machining*, Computer-Aided Design, **25**(5): pp. 281-288, 1993.

[Watt 89] Watt, A., *Fundamentals of Three-Dimensional Computer Graphics*, 1989, Addison-Wesley Publishing Company.

[Welch 92] Welch, W. and Witkin, A., *Variational surface modeling*. in ACM SIGGRAPH, Computer Graphics, Vol. 26, pp. 157-166, 1992.

[Wever 91] Wever, U., *Optimal parametrization for cubic splines*, Computer-Aided Design, **23**: pp. 641-644, 1991.

[Whelan 86] Whelan, T., *A representation of a C^2 interpolant over triangles*, Computer Aided Geometric Design, **3**, 1986.

[Woodward 88] Woodward, C. D., *Skinning techniques for interactive B-spline surface interpolation*, Computer-Aided Design, **20**(8): pp. 441-451, 1988.

[Zenisek 70] Zenisek, A., *Interpolation polynomials on the triangle*, Numerische Mathematik, **15**: pp. 283-296, 1970.

[Zhou 93] Zhou, J., Sherbrooke, E. C. and Patrikalakis, N. M., *Computation of stationary points of distance functions*, Engineering with Computers, **9**(4): pp. 231-246, 1993.

[Zienkiewicz 67] Zienkiewicz, O. C., *The Finite Element Method*, 1967, McGraw-Hill Book Co., U. K.