Robust Interrogation of Differential Properties for Design and Manufacture

by

George D. Margetis

B. S. E. Mechanical Engineering, June 1989
B. S. E. Naval Architecture and Marine Engineering, December 1989
M. S. E. Mechanical Engineering, December 1990
The University of Michigan
Ann Arbor

Submitted to the Department of Ocean Engineering and the Department of Mechanical Engineering in partial fulfillment of the requirements for the degrees of

Master of Science in Naval Architecture and Marine Engineering

and

Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 1994

Author .... [illegible]
Department of Ocean Engineering
June 27, 1994

Certified by .... [illegible]
Professor Nicholas M. Patrikalakis
Thesis Supervisor, Dept. of Ocean Engineering

Certified by ....
Professor David C. Gossard
Thesis Reader, Dept. of Mechanical Engineering

Accepted by ....
Professor A. Douglas Carmichael
Chairman, Ocean Eng. Departmental Graduate Committee

Accepted by ....
Professor Ain A. Sonin
Chairman, Mechanical Eng. Departmental Graduate Committee

Robust Interrogation of Differential Properties for Design and Manufacture

by

George D. Margetis

Submitted to the Department of Ocean Engineering and the Department of Mechanical Engineering on June 27, 1994, in partial fulfillment of the requirements for the degrees of
Master of Science in Naval Architecture and Marine Engineering
and
Master of Science in Mechanical Engineering

Abstract

The areas of Computer Aided Design (CAD) and Computer Aided Manufacturing (CAM) have experienced enormous advances in recent years. Nevertheless, the linkage between them is still weak. Robustness and accuracy are still the major problems of the area, particularly when dealing with complex sculptured or free-form objects, such as marine propellers. Robust interrogation of surfaces could be a major contribution to the automation of the manufacturing and machining processes. This thesis addresses the development of robust and accurate tools for the interrogation of complex surfaces described mathematically by B-splines. B-spline curves and surfaces are represented by piecewise polynomial equations. This allows for the formation of nonlinear polynomial equations that govern shape interrogation, ie. the extraction of important differential and global geometric properties. In this thesis we develop a continuous decomposition of non-uniform B-spline surface patches into a set of trimmed patches each with a specified range of curvature (Gaussian, mean, maximum principal, minimum principal and root mean square curvature). The original B-spline surface is subdivided into several Bézier patches. The formulation and solution of the nonlinear polynomial equations is performed on each separate Bézier patch, in order to achieve computational efficiency. Solutions to those nonlinear systems are the stationary points of each curvature function, and the surface umbilics. In order to render the methodology numerically robust, rounded interval arithmetic was implemented. Unlike double precision floating point arithmetic that suffers because of the existence of numerical error, rounded interval arithmetic guarantees that no solution of a system of nonlinear polynomial equations is missed. In addition, the solver for nonlinear polynomial systems is based in global Bernstein subdivision methods, which unlike local methods, do not require a first approximation for the root. The nature of the problem renders parallel processing very attractive, since the original problem is subdivided into several smaller subproblems. The implementation of parallel processing was made possible through Parallel Virtual Machine (PVM), a software system that enables a collection of heterogeneous computers to be used as a coherent and flexible concurrent computational resource. The introduction of parallel processing to the methodology, allows for faster continuous decomposition of complex B-splines surfaces. Detailed results for the parallel processing speed-up are included.

Thesis Supervisor: Nicholas M. Patrikalakis, Associate Professor of Ocean Engineering

Thesis Reader: David C. Gossard, Professor of Mechanical Engineering

To my mother Vassilia and my sister Maria

In Memory of my father Dimitris and my cousin George A. Tsaklas

Στη μητέρα μου Βασιλεία και στην αδελφή μου Μαρία

Στη Μνήμη του πατέρα μου Δημητρίου και του εξαδέλφου μου Γεωργίου Α. Τσάκλα

Acknowledgments

First and foremost, I would like to acknowledge my late father, Dimitris, my mother Vassilia and my sister Maria for their love, leadership, support and motivation. All that I have achieved, I owe to them. I could not have done it without them.

I would like to extend my deepest gratitude to my advisor Professor N. M. Patrikalakis for his support and leadership during the last year. He was to me everything a student would ever expect from his advisor. I truly enjoyed our interaction.

I would like to extend my deepest thanks to the director of the Ocean Engineering Design Laboratory, Professor Chryssostomos Chryssostomidis, for his support throughout my stay at MIT. My meetings with him always lifted my spirits, even during the hardest times I had to go through.

I would also like to thank Professor David C. Gossard, my thesis reader for his patience and support.

Dr. Takashi Maekawa supported this work with his experience and background. He laid the base for the research presented here. His help was enormous. Also, Dr. Seamus T. Tuohy was always available for help through his software engineering expertise and advice. I am deeply thankful to both of them.

I also would like to mention Dr. Nikiforos Papadakis who help me build problem solving skills and taught me how to do research during his years at MIT. His help during my qualifiers will be remembered for ever. I am deeply grateful to him.

Michael Drooker, manager of the Ocean Engineering Design Laboratory has helped me in several occasions. All of this thesis was developed in the Design Laboratory and his help is deeply appreciated. Stephen Abrams also had some valuable advice for my work.

I am grateful to my uncle Antonis Tsaklas who motivated me to come to MIT. His influence in my life has been enormous. I am also grateful to my cousin Dr. John Aggelidis whose advice and lengthy discussions I cherish.

I would also like to thank Zahra-el-Hayat Tazir to whom I owe so much of the advice and support during my years in Boston. I thank her for cheering me up during periods of stress and I hope she forgives me for I have been impatient several times.

I would like to thank all my friends at MIT and particularly Thanassis Tjavaras for his help and support all these years. He was there whenever I needed him even at 3:00 am. I would also like to mention my friends from Michigan, Tony Mercieca and Guy Fasulo.

I would like to thank all my friends in Greece and particularly Dimitris Koutas and John Evangelopoulos. Our friendship goes back many years and was cherished during my years in the U.S.A. Thank you for being my friends.

Finally I would like to extent my dearest thanks to my whole family in Greece and the U.S. for their love and support.

This work was supported, in part, by the MIT Sea Grant College Program, the Office of Naval Research, National Science Foundation and the Naval Sea Systems Command under grant numbers NA90AA-D-SG-424, N00014-91-J-1014, DDM-9215411 and N000249WR10564.

George D. Margetis
Cambridge, June 27, 1994

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Objective and Motivation

Although, in recent years the advances made in the area in Computer Aided Design (CAD) and Computer Aided Manufacturing (CAM) are enormous, these advances are mostly intended to support rather than replace human labor. In addition, CAD and CAM have been developed independently with little attention focused to the automated linkage between them, [18].

In the area of CAD, many objects are represented by means of free-form surfaces, also known as sculptured surfaces, represented by parametric equations. The parametric representation of the surfaces provides an efficient way to generate data points explicitly and avoids axis dependence. Free-form surfaces arise in the bodies of ships, aircrafts and automobiles and in general in every major industrial part. Since the shape of parts such as propeller or turbine blades, significantly affects their performance, major attention needs to be given to such forms during the design process.

The interrogation of free-form surfaces plays an important role in the analysis, design and manufacturing processes. We refer to *interrogation* as the extraction of important differential and global geometric properties that the free-form surface might exhibit. Such properties and their importance will be discussed in Chapter 2.

One of the obstacles, that prevents the implementation of a fully automated way to manufacture free-form surfaces, is the lack of robustness in the interrogation tools. Robustness is one of the key elements in achieving linkage between CAD and CAM. If robustness is not guaranteed, the need to manually or visually verify the results, leads to suboptimal procedures, commonly appearing in the form of large safety factors, [18].

The objective of this thesis is to develop a methodology that will robustly interrogate complex free-form objects in order to support automation of design and manufacturing. We give special attention to the robustness of this methodology, requiring that all the differential and global geometric properties of the free-form object are extracted automatically without failure.

One of the most important properties of a free-form object is the curvature of its surfaces at any given point. Definitions and details on the different kinds of curvatures are presented in Chapter 3.

In this thesis we limit our investigation of the differential properties to the examination of the curvature. The main information obtained from the curvature functions are the stationary points (local and global maxima, minima and saddle points of the curvature) as well as contour plots. This information is of vital interest during the machining process. Indeed, numerically controlled (NC) machining is used extensively in the industry today. The user must then know the exact range of curvature to select the optimal combination of tool path and cutter size for NC machining.

In addition, in this thesis we examine the capabilities of parallel programming for a more efficient evaluation of complex surfaces.

1.2 Literature Review

Extensive work on this subject, was done by Maekawa in his PhD Thesis [18]. Maekawa developed a robust interrogation tool that creates contour plots of the different kinds of curvature, of free-form surfaces described by a single Bézier patch. In his thesis, he distinguishes between interrogation algorithms based on local and discrete methods where the computation involves numerical uncertainty, and global and continuous methods where the computation involves numerical uncertainty.

More specifically, since curves and surfaces are usually represented by parametric piecewise polynomial equations, the governing equations for interrogation reduce to systems of nonlinear polynomial equations, frequently involving also square roots of polynomial equations. These systems of equations have been solved in the past using local numerical techniques such as Newton type methods that require good initial approximation to all roots and hence cannot provide full assurance that all roots will be found. On the other hand, global techniques described in [13], [18], [19] and [20] find all roots without requiring initial approximation.

Discrete color coded maps are used in existing commercial systems to estimate the range of principal curvatures but are not sufficient to provide detailed machining information, nor permit automation of the machining process or of fairing algorithms. *Continuous* decomposition of surfaces on the basis of curvature provides the exact range of curvatures and is able to supply detailed machining information. The degrees of some of the governing equations for interrogation are relatively high. In addition, if floating point arithmetic is employed for the computation, there exists substantial *numerical uncertainty* in the formulation and solution process. If however, the computation is conducted with interval arithmetic, one can obtain the results with *numerical certainty.*

The computational time required to robustly obtain color maps of curvature can be substantial, especially if the computation is conducted with intervals. For that reason, the methodology developed by Maekawa [18] was limited to simple surfaces described mathematically by a single Bézier patch. More complex surfaces, usually described by B-splines, were not handled. In Chapter 2 we present a brief overview of the free-form representation using Bézier patches and B-splines as well as the relationship between them. It is this relationship that allows us to decompose a B-spline patch, into several Bézier patches and apply separately, over each patch, a methodology similar to that of Maekawa's, [18], [20].

Subsequently, the patches are assembled to obtain the final result on the entire surface. It is noteworthy that the nature of the decomposition makes parallel programming appealing.

The implementation of the parallel programming was made possible through Parallel Virtual Machine (PVM), a software system that enables a collection of heterogeneous computers to be used as a coherent and flexible concurrent computational resource. More details about parallel programming and PVM will be discussed later, in Chapter 4.

1.3 Thesis Organization

This thesis is organized as follows.

Chapter 2 provides a brief review of the basic properties of Bézier and B-spline curves and surfaces, as well as the algorithms used for their transformation. A description of the methodology used to solve systems of nonlinear polynomial equations is included as well as an overview of the interval arithmetic procedures.

Chapter 3 presents the mathematical model used to produce the contour maps. It includes an introduction to the differential geometry of surfaces, mathematical methods of finding the critical points of the curvature on a free-form surface as well as the algorithm used for contouring.

Chapter 4 introduces the parallel processing procedure used in the computational aspect of this thesis. It describes the way "Parallel Virtual Machine" works and how it is implemented in our case.

Chapter 5 describes implementation issues for obtaining the contour maps of the curvatures on B-splines patches. It includes the particular algorithms and data structures used.

Chapter 6 presents selected examples created with the methodology described earlier. Several color coded curvature contour maps are included. Also, information on the performance of parallel programming and its impact on lowering the computational time.

Chapter 6 also summarizes the contribution of this thesis and draws conclusions. Directions for further research are also suggested.

Chapter 2

Theoretical Background

In this chapter, some topics necessary for further understanding of the thesis are presented. These topics have been developed by many other researchers and are widely available in the literature. Herein, the relevant references are given separately for each subject.

2.1 Review of Shape Representation Using Bézier and B-Spline Surfaces

A number of books and articles related to the area of Computer Aided Design include the theoretical background for the mathematical representation of free-form curves and surfaces using Bézier and B-spline representation. A small portion of the literature includes [8], [9], [1] and [33]. A very brief review is included here.

2.1.1 Bézier Curves and Surfaces

Bézier curves can be defined in two ways: via a recursive algorithm, which was developed by de Casteljau, or via an explicit representation using Bernstein polynomials. We will express

Bézier curves in terms of **Bernstein** polynomials defined by the following formula [8]:

$$B_{i,n}(t) = \begin{pmatrix} n \\ i \end{pmatrix} t^i(1-t)^{n-i} \quad i = 0, 1, \ldots, n \tag{2.1}$$

One of the important properties of the Bernstein polynomials is that they satisfy the following recursion,

$$B_{i,n}(t) = (1-t)B_{i,n-1}(t) + tB_{i-1,n-1}(t) \tag{2.2}$$

where,

$$B_{0,0}(t) \equiv 1 \quad \text{and} \quad B_{j,n}(t) \equiv 0 \text{ for } j \notin \{0, \ldots, n\} \tag{2.3}$$

Another important property is that Bernstein polynomials form a *partition of unity*:

$$\sum_{j=0}^{n} B_{j,n}(t) \equiv 1 \tag{2.4}$$

The equation describing the Bézier curve is given by,

$$\mathbf{R}(t) = \sum_{j=0}^{n} \mathbf{b_j} B_{j,n}(t) \tag{2.5}$$

where $\mathbf{b_0}, \mathbf{b_1}, \ldots, \mathbf{b_n} \in \mathcal{R}^3$ and $t \in [0, 1]$. The degree of the Bézier curve is n, and $\mathbf{b_j}$ are the control points.

The above could be extended to the formulation of the Bézier surfaces. We can consider a surface to be the locus of a curve that is continuously moving through space, and thereby changing its shape. In order to formulate the mathematical description of a Bézier surface, we first assume that the moving curve is a Bézier curve. At any time, the moving curve is then determined by a set of control points, which in turn, move through space on a curve. A second assumption is that this next curve is also a Bézier curve and that the curves on which the control points move are of the same degree. The formal definition of a Bézier patch is thus,

$$\mathbf{r}(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} \mathbf{P}_{ij} B_{i,m}(u) B_{j,n}(v) \tag{2.6}$$

where m, n are the degrees of the surface in the u, v parametric directions, and $\mathbf{P}_{ij}$ are the control points, $0 \leq u, v \leq 1$ and $B_{i,m}(u)$, $B_{j,n}(v)$ are the Bernstein basis functions.

2.1.2 B-Spline Curves and Surfaces

Although the Bézier representation for curves and surfaces, provides a powerful tool in design, it has some limitations. For instance, complicated shapes require high degree Bézier curves and surfaces, whereas for efficiency and accuracy, the degree should not exceed 10. Such complex curves and surfaces are modeled using piecewise polynomial curves and surfaces, known as B-splines. We define the B-spline curve and surface by using the B-spline basis.

The B-Spline Basis: The B-spline basis is defined recursively. The following recursion formula relates B-splines of degree n to B-splines of degree n-1:

$$N_l^n(u) = \frac{u - u_{l-1}}{u_{l+n-1} - u_{l-1}} N_l^{n-1}(u) + \frac{u_{l+n} - u}{u_{l+n} - u_l} N_{l+1}^{n-1}(u) \tag{2.7}$$

where

$$N_i^0(u) = \begin{cases} 1 & if \quad u_i \leq u \leq u_{i+1} \\ 0 & else \end{cases}$$

and u_i are the components of the *knot vector.* The knot vector is a non-decreasing sequence, in which the range of parameter is defined. Given integers n, L , the knot vector is defined as follows, [8]:

$$u_0, \ldots, u_{L+2n-2}$$

n will be the order of the B-spline, L will be the number of the polynomial segments of the B-spline curve and u_i the knots. Not all of the u_i have to be distinct. If $u_i = u_{i+1} = \ldots = u_{i+r-1}$, i.e., r successive knots coincide, we say that u_i has multiplicity r. If a knot does not coincide with any other knot, we say that it is simple.

$$\underbrace{u_0, \ldots, u_{n-2}}_{n-1\ knots}, \quad \underbrace{u_{n-1}, \ldots, u_{L+n-1}}_{domain\ of\ parameter}, \quad \underbrace{u_{L+n}, \ldots, u_{L+2n-2}}_{n-1\ knots} \tag{2.8}$$

The domain of B-spline curve is the range of parameter $u \in [u_{n-1}, \ldots, u_{L+n-1}]$. Although the knots could be any nondecreasing sequence, they usually go from zero to one by convention.

By using the basis functions, the B-spline curve can be defined as follows:

Given n, L, and $\mathbf{P}_0, \ldots, \mathbf{P}_{n+L-1} \in \mathcal{R}^3$

$$\mathbf{R}(u) = \sum_{i=0}^{L+n-1} \mathbf{P}_i N_i^n(u) \tag{2.9}$$

Practically, it is desirable to have u_0 and u_{L+n-1} both of full multiplicity n. This condition places the first and last control points $\mathbf{P}_0$ and $\mathbf{P}_{L+n-1}$ on the endpoints of the curve. If the end knots are allowed to be of lower multiplicity, then the first and last control points do not lie on the curve. The name "B-spline" is derived from the *Basis* spline functions.

In a similar way with the Bézier surface, the B-spline surface is defined as,

$$\mathbf{R}(u, v) = \sum_{i=0}^{m} \sum_{j=0}^{n} \mathbf{P}_{i,j} N_i^m(u) N_j^n(v) \tag{2.10}$$

2.2 Interval Arithmetic

2.2.1 Definition

An *interval* is a set of real numbers defined below [23]:

$$[a, b] = \{x | a \leq x \leq b\} \tag{2.11}$$

Two intervals $[a, b]$ and $[c, d]$ are said to be *equal* if $a = c$ and $b = d$. The *intersection* of two intervals is *empty* or $[a, b] \cap [c, d] = \emptyset$, if either $a > d$ or $c > b$. Otherwise, $[a, b] \cap$

$[c,d] = [max(a,c), min(b,d)]$. The *union* of the two intersecting intervals is $[a,b] \cup [c,d] = [min(a,c), max(b,d)]$. An *order* of intervals is defined by $[a,b] < [c,d]$ if and only if $b < c$. The width of an interval $[a,b]$ is $b - a$ and the *absolute value* is $|[a,b]| = max(|a|,|b|)$.

2.2.2 Interval Arithmetic and its Algebraic Properties

The interval arithmetic operations are defined by [23]

$$[a,b] \circ [c,d] = \{x \circ y \mid x \in [a,b] \ and \ y \in [c,d]\}. \tag{2.12}$$

where $\circ$ represents an arithmetic operation $\circ \in \{+,-,\cdot,/\}$. Using the end points of the two intervals, we can rewrite equation (2.12) more explicitly as follows,

$$\begin{aligned}
&[a,b] + [c,d] = [a+c, b+d] \\
&[a,b] - [c,d] = [a-d, b-c] \\
&[a,b] \cdot [c,d] = [min(ac,ad,bc,bd), max(ac,ad,bc,bd)] \\
&[a,b]/[c,d] = [min(a/c,a/d,b/c,b/d), max(a/c,a/d,b/c,b/d)]
\end{aligned} \tag{2.13}$$

provided $0 \notin [c,d]$ in the division relation.

Interval arithmetic is *commutative* and *associative.*

$$\begin{aligned}
&[a,b] + [c,d] = [c,d] + [a,b] \\
&[a,b] \cdot [c,d] = [c,d] \cdot [a,b] \\
&[a,b] + ([c,d] + [e,f]) = ([a,b] + [c,d]) + [e,f] \\
&[a,b] \cdot ([c,d] \cdot [e,f]) = ([a,b] \cdot [c,d]) \cdot [e,f]
\end{aligned}$$

But it is not *distributive*, however, it is *subdistributive.*

$$[a,b] \cdot ([c,d] + [e,f]) \subseteq [a,b] \cdot [c,d] + [a,b] \cdot [e,f]$$

2.2.3 Rounded Interval Arithmetic and its Implementation

If floating point arithmetic is used to evaluate the interval arithmetic equations (2.13), there is no guarantee that the roundings of the bounds are conducted conservatively. Floating numbers are represented in the computer by a fixed length. The number of bytes to represent a floating point number depends on the precision of the variable. For example, the IEEE standard for a *double-precision number* has 64 bits, 8 bytes wordsize, and is stored in a binary form $(\pm)m \cdot 2^{exp}$, where m is the *mantissa* $(0.5 \leq m < 1)$ and *exp* is the *exponent*. Figure (2-1) illustrates how the information is stored in the binary form, a single bit for sign, 11 bits for exponent and 52 bits for mantissa. Since the mantissa is restricted to the range $0.5 \leq m < 1$, the bit for 2^{-1} is not used. The exponent is 1022 biased to ensure the stored exponent is always positive. For example the number -0.125 is stored as $1011111111000\cdots0$. Most left bit represents the sign $-$, next 11 bits 01111111100 is the biased exponent which is 1020-1022 = -2 and the rest of 52 bits which are all zero represents the mantissa 0.5. Hence $-0.5 \cdot 2^{-2} = -0.125$. If x and x' are consecutive positive double-precision numbers, they differ by an amount ϵ called *ulp* (one Unit in the Last Place), so that $\epsilon = 2^{-53} \cdot 2^{exp} = 2^{exp-53}$. Now it is possible to carry out the operation of interval arithmetic with rounding, so that the computed end points always contain the exact interval as follows

$$
\begin{aligned}
&[a,b] + [c,d] \equiv [a + c - \epsilon, b + d + \epsilon] \\
&[a,b] - [c,d] \equiv [a - d - \epsilon, b - c + \epsilon] \\
&[a,b] \cdot [c,d] \equiv [min(ac, ad, bc, bd) - \epsilon, max(ac, ad, bc, bd) + \epsilon] \\
&[a,b]/[c,d] \equiv [min(a/c, a/d, b/c, b/d) - \epsilon, max(a/c, a/d, b/c, b/d) + \epsilon]
\end{aligned} \tag{2.14}
$$

Each ϵ in the equations can be obtained by $\epsilon = 2^{exp-53}$ where *exp* is extracted from *each* computed lower or upper bound. We refer to the definitions given in equations (2.14) as *rounded interval arithmetic.*

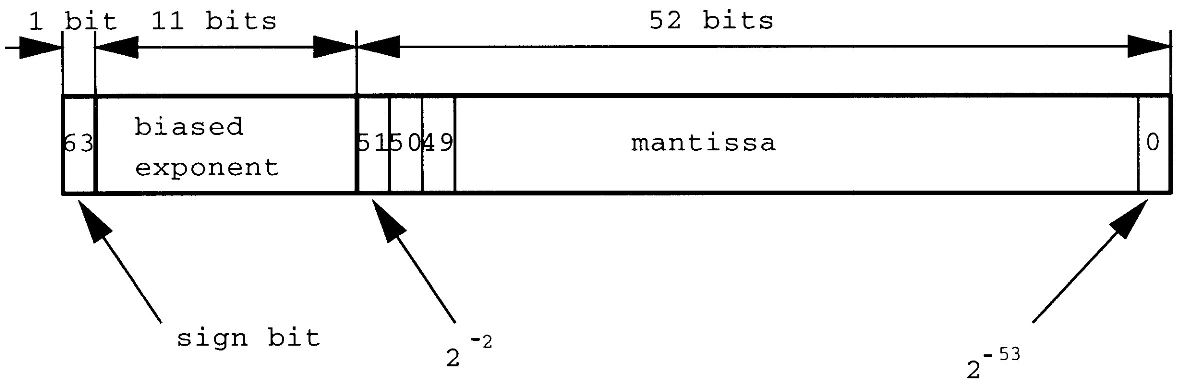


Figure 2-1: IEEE Format for Binary Representation of Double-Precision Floating-Point Number, adapted from [18]

2.3 Review of Computation of Real Roots of Nonlinear Polynomial Systems

The solution of systems of nonlinear polynomial equations can be computed using local numerical techniques which employ some variation of Newton methods. These methods though require good initial approximations to the roots and do not guarantee that all the possible roots are found. On the other hand, global numerical techniques are designed to compute all roots in some area of interest. Among the global methods, the Bernstein subdivision-based technique has been favored in recent CAD related research. Details of the method can be found in [18], [27], [29], and [30].

We will demonstrate this method with a single univariate polynomial equation $f(u) = 0$ of degree m over the range $a \leq u \leq b$. By making the affine parameter transformation $u = a + t(b - a)$ so that $0 \leq t \leq 1$, we can write $f(t)$ in Bernstein basis as:

$$f(t) = \sum_{i=0}^{m} \bar{f}_i B_{i,m}(t) \tag{2.15}$$

Using the linear precision property,

$$t = \sum_{i=0}^{m} \frac{i}{m} B_{i,m}(t) \tag{2.16}$$

we can rewrite the Bézier function $f(t)$ as a parametric Bézier curve $\mathbf{f}(t)$.

$$\mathbf{f}(t) = \begin{pmatrix} t \\ f(t) \end{pmatrix} = \sum_{i=0}^{M} \begin{pmatrix} \frac{i}{M} \\ \bar{f}_i \end{pmatrix} B_{i,M}(t) \tag{2.17}$$

Now the problem of finding roots of the univariate polynomial has been transformed into a problem of finding the intersection of the Bézier curve with the parameter axis which can be solved using the recursive de Casteljau subdivision algorithm. Figure (2-2) shows how the regions which do not contain the intersection points are discarded in the case of a quadratic Bézier curve. The large triangle is the convex hull of the quadratic Bézier curve. This triangle intersects the axis at two points $t = a$ and $t = b$. Applying de Casteljau subdivision algorithm to the Bézier curve with the control points being the vertices of the large triangle at these parameter values, we obtain a small triangle, (shaded in the figure), which also intersects the axis at two points. Such a recursive subdivision process, using the convex hull property, can be continued until the interval width becomes as small as required. But when there are more than one roots in the interval, the interval will not be reduced to arbitrarily small size. In such cases binary subdivision may be introduced [25]. Binary subdivision is applied when the box size did not reduce more than 20% from the previous step, in accordance with [25]. An extension of this algorithm to n-dimensions is described in [29], [30]. Moreover, if floating point arithmetic is employed, accuracy in a subdivision method could be lost for high degree polynomials. Consequently, in order to guarantee a robust and numerically verifiable solution, the Bernstein subdivision method, coupled with rounded interval arithmetic has been developed in [18], [20], [19]

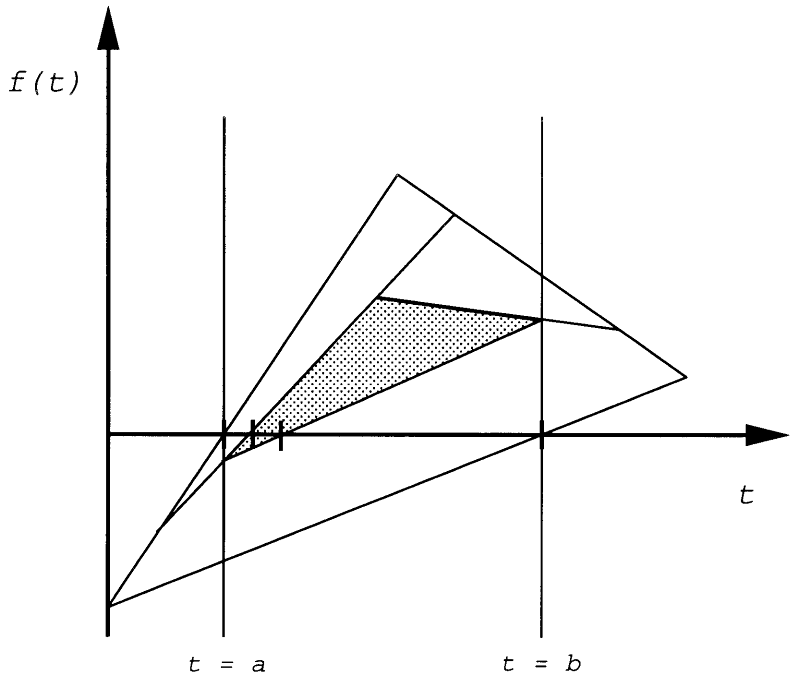


Figure 2-2: de Casteljau Algorithm Applied to the Quadratic Bézier Curve, adapted from [18]

Chapter 3

Differential Geometry of Surfaces

3.1 Introduction

In this chapter, we present the mathematical basis for the development of decomposing a surface into specific range of curvature, with particular emphasis on B-splines and Bezier patches. This chapter draws its contents to a significant degree from [18]. The differential geometry of curves and surfaces is fundamental in CAGD. The curves and surfaces treated in differential geometry are defined by functions which can be differentiated a certain number of times. A book by Hilbert and Cohn-Vossen [12] and a recent book by Koenderink [15] provide intuitive access to the extensive mathematical literature on three-dimensional shape analysis. The books by Struik, [31], doCarmo, [6], and Banchoff et al, [2] offer firm theoretical basis to the differential geometry aspects of three-dimensional shape description. In this section, we summarize the relevant definitions employed in this work.

A general parametric surface can be defined as a vector-valued mapping from two-dimensional parametric uv-space to a set of three-dimensional coordinates

$$\mathbf{r}(u,v) = [x(u,v),\ y(u,v),\ z(u,v)]^T \tag{3.1}$$

The shape of a surface is completely characterized by two important geometric structures:

the first and second fundamental forms. The first fundamental form I provides metrical properties of surfaces such as measurement of lengths, areas and angles between two curves on the surface. It is defined as the dot product of infinitesimal displacement $d\mathbf{r}$ with itself.

$$\begin{aligned} I &= d\mathbf{r} \cdot d\mathbf{r} = (\mathbf{r}_u du + \mathbf{r}_v dv) \cdot (\mathbf{r}_u du + \mathbf{r}_v dv) \\ &= E du^2 + 2F du dv + G dv^2 \\ &= d\mathbf{q}[\Gamma] d\mathbf{q}^T \end{aligned} \tag{3.2}$$

where subscripts denote partial derivatives, and

$$E = \mathbf{r}_u \cdot \mathbf{r}_u, \quad F = \mathbf{r}_u \cdot \mathbf{r}_v, \quad G = \mathbf{r}_v \cdot \mathbf{r}_v \tag{3.3}$$

$$d\mathbf{q} = [du \;\; dv] \tag{3.4}$$

$$[\Gamma] = \begin{pmatrix} E & F \\ F & G \end{pmatrix} \tag{3.5}$$

The second fundamental form II permits the analysis of the surface curvature at a given point and is defined as the dot product of infinitesimal displacement $d\mathbf{r}$ and infinitesimal variation $d\mathrm{N}$ of the surface unit normal vector $\mathbf{N}$.

$$\begin{aligned} II &= -d\mathbf{r} \cdot d\mathrm{N} = -(\mathbf{r}_u du + \mathbf{r}_v dv) \cdot (\mathrm{N}_u du + \mathrm{N}_v dv) \\ &= L du^2 + 2M du dv + N dv^2 \\ &= d\mathbf{q}[\Delta] d\mathbf{q}^T \end{aligned} \tag{3.6}$$

where

$$\mathrm{N} = \frac{\mathbf{r}_u \times \mathbf{r}_v}{|\mathbf{r}_u \times \mathbf{r}_v|} \tag{3.7}$$

$$L = \mathrm{N} \cdot \mathbf{r}_{uu}, \quad M = \mathrm{N} \cdot \mathbf{r}_{uv}, \quad N = \mathrm{N} \cdot \mathbf{r}_{vv} \tag{3.8}$$

$$[\Delta] = \begin{pmatrix} L & M \\ M & N \end{pmatrix} \tag{3.9}$$

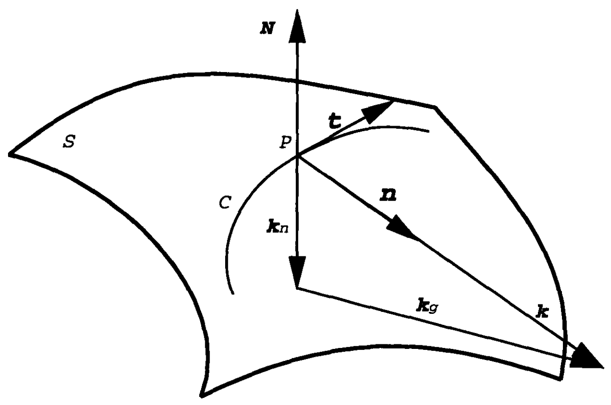


Figure 3-1: Definition of Normal Curvature, adapted from [18]

and $\frac{\partial}{\partial u}(\mathbf{r}_u \cdot \mathrm{N}) = 0$, $\frac{\partial}{\partial v}(\mathbf{r}_v \cdot \mathrm{N}) = 0$ are used in the derivation . In order to quantify the curvatures of a surface S, we consider a curve C on S which passes through point P as shown in Fig. (3-1). $\mathbf{t}$ is the unit tangent vector and $\mathbf{n}$ is the unit normal vector of the curve C at point P. If $\mathbf{k}$ is the curvature vector of the curve C on the surface S at P, which can be obtained by $\mathbf{k} = \frac{d\mathbf{t}}{ds}$, we can represent $\mathbf{k}$ as sum of a normal and a tangential component $\mathbf{k}_n$ and $\mathbf{k}_g$. $\mathbf{k}_n$ is called the normal curvature vector and $\mathbf{k}_g$ is called the geodesic curvature vector. The normal curvature vector can be expressed as a multiple of the unit surface normal vector N namely

$$\mathbf{k}_n = \kappa \mathrm{N} \tag{3.10}$$

where κ is the normal curvature and can be obtained by differentiating the equation $\mathrm{N}\cdot\mathbf{t} = 0$ along C with respect to the arc length.

$$\begin{aligned} \kappa &= -\frac{d\mathbf{t}}{ds}\cdot \mathrm{N} = \mathbf{t}\cdot\frac{d\mathrm{N}}{ds} = \frac{d\mathbf{r}}{ds}\cdot\frac{d\mathrm{N}}{ds} \\ &= -\frac{II}{I} = -\frac{Ldu^2 + 2Mdudv + Ndv^2}{Edu^2 + 2Fdudv + Gdv^2} = -\frac{L + 2M\lambda + N\lambda^2}{E + 2F\lambda + G\lambda^2} \end{aligned} \tag{3.11}$$

where $\lambda = \frac{dv}{du}$ specifies the direction of the curve. The sign convention used in equation (3.11) ensures that positive κ is on the side of the surface opposite to the direction of the normal. At any given point (u, v), κ in general varies with each direction λ. The extreme values of κ can be obtained by evaluating $\frac{d\kappa}{d\lambda} = 0$ which gives:

$$(E + 2F\lambda + G\lambda^2)(\lambda N + M) - (L + 2M\lambda + N\lambda^2)(\lambda G + F) = 0 \tag{3.12}$$

Since

$$\begin{aligned} E + 2F\lambda + G\lambda^2 &= (E + F\lambda) + \lambda(F + G\lambda), \\ L + 2M\lambda + N\lambda^2 &= (L + M\lambda) + \lambda(M + N\lambda) \end{aligned}$$

equation (3.12) can be reduced to

$$(E + F\lambda)(M + \lambda N) = (L + M\lambda)(F + \lambda G) \tag{3.13}$$

Using equation (3.13), equation (3.11) can be rewritten as:

$$\kappa = -\frac{L + 2M\lambda + N\lambda^2}{E + 2F\lambda + G\lambda^2} = -\frac{M + \lambda N}{F + \lambda G} = -\frac{L + M\lambda}{E + \lambda F} \tag{3.14}$$

Therefore κ satisfies the two simultaneous equations

$$\begin{aligned} (L + \kappa E)du + (M + \kappa F)dv = 0 \\ (M + \kappa F)du + (N + \kappa G)dv = 0 \end{aligned} \tag{3.15}$$

These equations can be simultaneously satisfied if and only if

$$\begin{vmatrix} L + \kappa E & M + \kappa F \\ M + \kappa F & N + \kappa G \end{vmatrix} = 0 \tag{3.16}$$

This quadratic equation in κ gives the upper and lower bounds of the normal curvature, which are the maximum principal curvature κ_{max} and the minimum principal curvature κ_{min}. The corresponding directions λ define directions in the uv-plane and the corresponding directions in the tangent plane are called *principal directions of curvature* and are in general orthogonal. The two roots are given by

$$\kappa_{max} = H + \sqrt{H^2 - K} \tag{3.17}$$

$$\kappa_{min} = H - \sqrt{H^2 - K} \tag{3.18}$$

where K is the *Gaussian curvature* and H is the *mean curvature* defined by

$$K = \frac{LN - M^2}{EG - F^2} \tag{3.19}$$

$$H = \frac{2FM - EN - GL}{2(EG - F^2)} \tag{3.20}$$

From equations (3.17), (3.18), it is readily seen that

$$K = \kappa_{max}\kappa_{min} \tag{3.21}$$

$$H = \frac{\kappa_{max} + \kappa_{min}}{2} \tag{3.22}$$

We can also define the *root mean square curvature* as,

$$\kappa_{rms} = \sqrt{\kappa_{max}^2 + \kappa_{min}^2} \tag{3.23}$$

If κ_{max} and κ_{min} have the same sign the Gaussian curvature is positive and the point is called *elliptic* point of the surface. Any patch on an ellipsoid is an elliptic region. If either of κ_{max} or κ_{min} is zero, the Gaussian curvature is zero and the point is called *parabolic. Developable surfaces* have zero Gaussian curvature at their regular points. Finally, if κ_{max} and κ_{min} have different signs the Gaussian curvature is negative and the point is called *hyperbolic.* Any point on a hyperbolic paraboloid is a hyperbolic point. When κ_{max} and κ_{min} are identical, the point approximates a sphere and is called an *umbilical point.* In the special case, where the identical principal curvatures vanish, the surface becomes locally *flat.* Note that at the *flat* point, $K = H = 0$. A spherical umbilic occurs at an elliptic point, but never at a hyperbolic point. From equation (3.11) it is apparent that at an umbilic I and II are proportional because κ = *constant*, and consequently, we have the following relation at the umbilic

$$\frac{L}{E} = \frac{M}{F} = \frac{N}{G} \tag{3.24}$$

The net of lines, that have as tangents the principal curvature directions at all of their points, form two sets of curves intersecting at right angles. They are called *lines of curvature.* The lines of curvature depend only on the shape of the surface, and not upon the parametrization. Lines of curvature provide a method to describe the variation of principal curvatures across a surface. At umbilical points only, the principal directions are indeterminate and the net of lines of curvature may have singular properties, [1], [18], [21]. Lines of curvature are obtained by integrating equations (3.15).

3.2 Stationary Points of Curvature

To subdivide the surface into regions of specific range of curvature, we need to determine the following, [18].

1. Locations of all the stationary points of the curvature and the associated values of curvature, so as to provide a correct topological decomposition of the surface on the basis of curvature.

2. Global maximum and minimum of the curvature to find the overall range of curvature.

The surface of interest is an integral B-spline patch (with non-uniform knots). A B-spline patch is a piece-wise polynomial, and a powerful generalization of polynomial Bézier patches. Therefore, we subdivide the B-spline surface into Bézier patches by inserting knots, so we can deal with polynomials [3], [5], [16]. An integral Bézier patch can be defined as,

$$\mathbf{r}(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n}\mathbf{P}_{ij}B_{i,m}(u)B_{j,n}(v) \tag{3.25}$$

where m, n are the the degrees of the patch in u, v parametric directions, and $\mathbf{P}_{ij}$ are the control points, $0 \leq u, v \leq 1$ and $B_{i,m}(u)$, $B_{j,n}(v)$ are the Bernstein basis functions [33]. An additional assumption is that the surface is regular, i.e. its Jacobian has full rank and therefore $\mathbf{r}_u \times \mathbf{r}_v \neq \mathbf{0}$. Points where $\mathbf{r}_u \times \mathbf{r}_v = \mathbf{0}$ correspond to either singularities of the parametrizations or intrinsic degeneracies of the surface such as ridges and cusps.

Gaussian, mean and principal curvatures can be evaluated in terms of parametric derivatives of $\mathbf{r}(u, v)$ [6]. Let the curvature in question be denoted by $C(u, v)$, then, to locate all the stationary points of curvature and to find the global maximum and minimum values of the curvature to provide a correct topological decomposition of the surface, the following need to be evaluated, for each separate Bézier patch, [18]:

1. The four values of curvature at the parameter domain corners

$$C(0,0),\ C(0,1),\ C(1,0),\ C(1,1) \tag{3.26}$$

2. Stationary points along parameter domain boundaries (roots of the 4 equations)

$$\begin{aligned} &C_u(u,0) = 0,\ C_u(u,1) = 0,\ 0 \leq u \leq 1 \\ &C_v(0,v) = 0,\ C_v(1,v) = 0,\ 0 \leq v \leq 1 \end{aligned} \tag{3.27}$$

3. Stationary points within the parameter domain (roots of the 2 simultaneous equations)

$$C_u(u,v) = 0,\ C_v(u,v) = 0,\ 0 \leq u,\ v \leq 1 \tag{3.28}$$

The curvature values at the parameter domain corners are readily computed. The computation of stationary points of the Gaussian, mean and principal curvatures along the boundary and within the parameter domain are further discussed in sections 3.2.1, 3.2.2 and 3.2.3 respectively.

3.2.1 Gaussian Curvature K

The governing equation for computing the stationary points of Gaussian curvature along the boundary, are obtained by substituting equation (3.19) into (3.27) and expressing the equation such that the denominator and the numerator include exclusively polynomials, [18].

$$K_u(u,0) = \frac{\hat{A}(u,0)}{S^6(u,0)} = 0, \quad K_u(u,1) = \frac{\hat{A}(u,1)}{S^6(u,1)} = 0, \quad 0 \le u \le 1 \tag{3.29}$$

$$K_v(0,v) = \frac{\bar{A}(0,v)}{S^6(0,v)} = 0, \quad K_v(1,v) = \frac{\bar{A}(1,v)}{S^6(1,v)} = 0, \quad 0 \le v \le 1 \tag{3.30}$$

where

$$S = |\mathbf{S}| = |\mathbf{r}_u \times \mathbf{r}_v| \tag{3.31}$$

$$\hat{A} = A_u S^2 - 4(\mathbf{S} \cdot \mathbf{S}_u)A \tag{3.32}$$

$$\bar{A} = A_v S^2 - 4(\mathbf{S} \cdot \mathbf{S}_v)A \tag{3.33}$$

$\hat{A}$, $\bar{A}$ are polynomials of degree $(10m-7, 10n-6)$, $(10m-6, 10n-7)$ in u and v. Polynomial A and its partial derivatives and partial derivatives of $\mathbf{S}$ are given in appendix A. Since we are assuming a regular surface, $S \neq 0$, we need only set the numerators of equations (3.29) and (3.30) to zero, resulting in

$$\hat{A}(u,0) = 0, \quad \hat{A}(u,1) = 0, \quad 0 \le u \le 1 \tag{3.34}$$

$$\bar{A}(0,v) = 0, \quad \bar{A}(1,v) = 0, \quad 0 \le v \le 1 \tag{3.35}$$

Therefore, for stationary points of Gaussian curvature along the domain boundary we need to solve four univariate polynomial equations (3.34) of degree $10m-7$ in u and (3.35) of degree $10n-7$ in v.

For the stationary points within the domain, we substitute equation (3.19) into (3.28) which yields

$$K_u(u,v) = \frac{\hat{A}(u,v)}{S^6(u,v)} = 0, \quad K_v(u,v) = \frac{\bar{A}(u,v)}{S^6(u,v)} = 0, \quad 0 \leq u,v \leq 1 \tag{3.36}$$

As $S \neq 0$, equations (3.36) are satisfied if

$$\hat{A}(u,v) = 0, \quad \bar{A}(u,v) = 0, \quad 0 \leq u,v \leq 1 \tag{3.37}$$

which are two simultaneous bivariate polynomial equations of degree $(10m-7, 10n-6)$, $(10m-6, 10n-7)$ in u and v.

3.2.2 Mean Curvature H

Similarly to the Gaussian curvature, we have the following equations to evaluate the stationary points of mean curvature H along the boundary, [18].

$$H_u(u,0) = \frac{\hat{B}(u,0)}{2S^5(u,0)} = 0, \quad H_u(u,1) = \frac{\hat{B}(u,1)}{2S^5(u,1)} = 0, \quad 0 \leq u \leq 1 \tag{3.38}$$

$$H_v(0,v) = \frac{\bar{B}(0,v)}{2S^5(0,v)} = 0, \quad H_v(1,v) = \frac{\bar{B}(1,v)}{2S^5(1,v)} = 0, \quad 0 \leq v \leq 1 \tag{3.39}$$

where

$$\hat{B} = B_u S^2 - 3(\mathbf{S} \cdot \mathbf{S}_u) B \tag{3.40}$$

$$\bar{B} = B_v S^2 - 3(\mathbf{S} \cdot \mathbf{S}_v) B \tag{3.41}$$

$\hat{B}$, $\bar{B}$ are polynomials of degree $(9m-6, 9n-5)$, $(9m-5, 9n-6)$ in u and v. Polynomial B and its partial derivatives are given in appendix A. As $S \neq 0$, we need only set the numerators of equations (3.38) and (3.39) to zero, resulting in

$$\hat{B}(u,0) = 0, \quad \hat{B}(u,1) = 0, \quad 0 \leq u \leq 1 \tag{3.42}$$

$$\bar{B}(0,v) = 0, \quad \bar{B}(1,v) = 0, \quad 0 \leq v \leq 1 \tag{3.43}$$

Therefore, for the stationary points of mean curvature along the domain boundary we need to solve four univariate polynomial equations (3.42) of degree $9m-6$ in u and (3.43) of degree $9n-6$ in v.

For the stationary points within the domain, we have

$$H_u(u,v) = \frac{\hat{B}(u,v)}{2S^5(u,v)} = 0, \quad H_v(u,v) = \frac{\bar{B}(u,v)}{2S^5(u,v)} = 0, \quad 0 \le u,v \le 1 \tag{3.44}$$

Since $S \neq 0$, equations (3.44) reduce to two simultaneous bivariate polynomial equations

$$\hat{B}(u,v) = 0, \quad \bar{B}(u,v) = 0, \quad 0 \le u,v \le 1 \tag{3.45}$$

3.2.3 Principal Curvature κ

To obtain the stationary points of principal curvature κ along the domain boundaries, we substitute equations (3.17) and (3.18) into (3.27) and express the equations such that the denominator and the numerator only include polynomials and square root of polynomials, [18].

$$\begin{aligned}
\kappa_u(u,0) &= \frac{f_1(u,0) \pm f_2(u,0)\sqrt{f_3(u,0)}}{2S^5(u,0)\sqrt{f_3(u,0)}} = 0, \quad 0 \le u \le 1 \\
\kappa_u(u,1) &= \frac{f_1(u,1) \pm f_2(u,1)\sqrt{f_3(u,1)}}{2S^5(u,1)\sqrt{f_3(u,1)}} = 0, \quad 0 \le u \le 1 \qquad (3.46) \\
\kappa_v(0,v) &= \frac{g_1(0,v) \pm g_2(0,v)\sqrt{f_3(0,v)}}{2S^5(0,v)\sqrt{f_3(0,v)}} = 0, \quad 0 \le v \le 1 \\
\kappa_v(1,v) &= \frac{g_1(1,v) \pm g_2(1,v)\sqrt{f_3(1,v)}}{2S^5(1,v)\sqrt{f_3(1,v)}} = 0, \quad 0 \le v \le 1 \qquad (3.47)
\end{aligned}$$

The plus and minus signs correspond to the maximum and minimum principal curvatures, and $f_1(u,v)$, $f_2(u,v)$, $f_3(u,v)$, $g_1(u,v)$ and $g_2(u,v)$ are polynomials of degree $(14m-9, 14n-8)$, $(9m-6, 9n-5)$, $(10m-6, 10n-6)$, $(14m-8, 14n-9)$, $(9m-5, 9n-6)$ in u and v

parameters and are given by

$$f_1(u,v) = (BB_u - 2A_uS^2)S^2 + (8AS^2 - 3B^2)(\mathbf{S}\cdot\mathbf{S}_u) \quad (3.48)$$

$$f_2(u,v) = B_uS^2 - 3(\mathbf{S}\cdot\mathbf{S}_u)B \quad (3.49)$$

$$f_3(u,v) = B^2 - 4AS^2 \quad (3.50)$$

$$g_1(u,v) = (BB_v - 2A_vS^2)S^2 + (8AS^2 - 3B^2)(\mathbf{S}\cdot\mathbf{S}_v) \quad (3.51)$$

$$g_2(u,v) = B_vS^2 - 3(\mathbf{S}\cdot\mathbf{S}_v)B \quad (3.52)$$

First we assume that $f_3 \neq 0$ and also $S \neq 0$, then equations (3.46), (3.47) become

$$f_1(u,0) \pm f_2(u,0)\sqrt{f_3(u,0)} = 0, \quad f_1(u,1) \pm f_2(u,1)\sqrt{f_3(u,1)} = 0, \quad 0 \leq u \leq 1 \quad (3.53)$$

$$g_1(0,v) \pm g_2(0,v)\sqrt{f_3(0,v)} = 0, \quad g_1(1,v) \pm g_2(1,v)\sqrt{f_3(1,v)} = 0, \quad 0 \leq v \leq 1 \quad (3.54)$$

Consequently, for the stationary points of principal curvatures along the boundary we need to solve four univariate irrational equations involving polynomials and square roots of polynomials (which arise from the analytic expressions of principal curvatures).

When $f_3 = 0$ (or equivalently $H^2 - K = 0$ if $S \neq 0$), equations (3.46), (3.47) become singular. This condition is equivalent to the point where the two principal curvatures are identical, i.e. an umbilical point. If the umbilical point coincides with a local maximum or minimum of the curvature, we cannot use equations (3.53) and (3.54) to locate such a point. In this case we need to locate the umbilical point first. To locate umbilical points along the domain boundaries, we need to solve the following equations

$$H^2(u,0) - K(u,0) = \frac{f_3(u,0)}{4S^6(u,0)} = 0, \quad H^2(u,1) - K(u,1) = \frac{f_3(u,1)}{4S^6(u,1)} = 0, \quad 0 \leq u \leq 1 \quad (3.55)$$

$$H^2(0,v) - K(0,v) = \frac{f_3(0,v)}{4S^6(0,v)} = 0, \quad H^2(1,v) - K(1,v) = \frac{f_3(1,v)}{4S^6(1,v)} = 0, \quad 0 \leq v \leq 1 \quad (3.56)$$

Since $S \neq 0$, we need to solve $f_3(u,0) = 0$, $f_3(u,1) = 0$, $f_3(0,v) = 0$, $f_3(1,v) = 0$. Then we use the criterion (see Appendix B) at the umbilic to check if the point is a local extremum of the principal curvatures [18], [21].

In the case of stationary points of principal curvature κ within the domain, the simultaneous bivariate equations (3.28) become

$$\kappa_u(u,v) = \frac{f_1(u,v) \pm f_2(u,v)\sqrt{f_3(u,v)}}{2S^5(u,v)\sqrt{f_3(u,v)}} = 0, \quad 0 \leq u,v \leq 1$$
$$\kappa_v(u,v) = \frac{g_1(u,v) \pm g_2(u,v)\sqrt{f_3(u,v)}}{2S^5(u,v)\sqrt{f_3(u,v)}} = 0, \quad 0 \leq u,v \leq 1 \tag{3.57}$$

Assuming $f_3 \neq 0$ and $S \neq 0$, we obtain

$$f_1(u,v) \pm f_2(u,v)\sqrt{f_3(u,v)} = 0, \quad g_1(u,v) \pm g_2(u,v)\sqrt{f_3(u,v)} = 0, \quad 0 \leq u,v \leq 1 \tag{3.58}$$

These are two simultaneous bivariate irrational equations involving polynomials and square roots of polynomials (which arise from the analytical expressions of principal curvatures).

At the umbilics, equations (3.57) become singular and similarly to the univariate case for the domain boundaries, we need to locate the umbilical points first by finding the roots of the bivariate polynomial equation $f_3(u,v) = 0$. Let

$$W(u,v) = H^2(u,v) - K(u,v) \tag{3.59}$$

then $W(u,v) = \frac{f_3(u,v)}{4S^6(u,v)}$ is a non-negative function, therefore $W(u,v)$ has a global minimum at the umbilic, see Appendix B. The condition for global minimum at the umbilic implies that $\nabla W = \mathbf{0}$ or equivalently (given that $f_3(u,v) = 0$)

$$W_u = \frac{f_{3u}}{4S^6} = 0, \quad W_v = \frac{f_{3v}}{4S^6} = 0 \tag{3.60}$$

Therefore, the locations of umbilics are the solutions of the following three equations, assuming $S \neq 0$

$$f_{3u}(u,v) = 0, \quad f_{3v}(u,v) = 0, \quad f_3(u,v) = 0 \quad 0 \leq u,v \leq 1 \tag{3.61}$$

These equations can be reduced to :

$$\begin{aligned} BB_u - 2A_uS^2 - 4A(\mathbf{S}\cdot\mathbf{S}_u) &= 0, \\ BB_v - 2A_vS^2 - 4A(\mathbf{S}\cdot\mathbf{S}_v) &= 0, \\ B^2 - 4AS^2 &= 0 \end{aligned} \tag{3.62}$$

with $0 \leq u, v \leq 1$. Since $f_3(u,v) = 0$ at the umbilics, equations (3.58) reduce to $f_1(u,v) = 0$, $g_1(u,v) = 0$. If we substitute the first equation of (3.62) into equation (3.48) and use the fact $f_3 = B^2 - 4AS^2 = 0$, we obtain $f_1(u,v) = 0$. Similarly by substituting the second equation of (3.62) into equation (3.51), we obtain $g_1(u,v) = 0$. Consequently, the solutions of equation (3.58) include not only the locations of extrema of principal curvatures but also the locations of the umbilical points. Then we use the criterion in Appendix B at the umbilical points to check if the umbilical point is a local extremum of principal curvatures.

3.3 Contouring

The constant curvature lines divide the surface into regions of specific range of curvature. The contouring levels should be determined to faithfully represent the curvature distribution. To do so, the following properties should be determined, [18]:

- Global maximum and minimum curvature values in the entire domain to find the range of curvature values.
- Locations of all the local maxima and minima of curvature inside the domain around which loops may be formed.
- Locations of all the saddle points of the curvature where the contour lines cross or exhibit more complex behavior.

Classification of stationary points of functions is briefly reviewed in Appendix B.

3.3.1 Finding Starting Points

If the original, B-spline surface is subdivided along the isoparametric line which contain the local maxima and minima of curvature inside the domain, and if the contouring levels of curvature are chosen such that the contour lines avoid saddle points, each sub-patch will contain simple contour branches without any loops or singularities. Therefore, we can find all the starting points of the various levels of contour lines along the parameter domain boundary of each sub-patch by finding the roots of following equations. These equations are formulated for the corresponding Bézier patches. Solutions to these equations yield local parametric coordinates, on the Bézier patch and must be translated to global parametric coordinates, on the B-spline surface. Starting with Gaussian curvature, [18]

$$K(u,0) = \frac{A(u,0)}{S^4(u,0)} = C_K, \quad K(u,1) = \frac{A(u,1)}{S^4(u,1)} = C_K \quad 0 \le u \le 1 \tag{3.63}$$

$$K(0,v) = \frac{A(0,v)}{S^4(0,v)} = C_K, \quad K(1,v) = \frac{A(1,v)}{S^4(1,v)} = C_K \quad 0 \le v \le 1 \tag{3.64}$$

where C_K is the constant Gaussian curvature value. These equations can be rewritten as follows

$$C_K S^4(u,0) - A(u,0) = 0, \quad C_K S^4(u,1) - A(u,1) = 0 \quad 0 \le u \le 1 \tag{3.65}$$

$$C_K S^4(0,v) - A(0,v) = 0, \quad C_K S^4(1,v) - A(1,v) = 0 \quad 0 \le v \le 1 \tag{3.66}$$

Equations (3.65), (3.66) are univariate polynomials of degree $8m-4$ in u and $8n-4$ in v respectively.

Similarly for mean curvature

$$H(u,0) = \frac{B(u,0)}{2S^3(u,0)} = C_H, \quad H(u,1) = \frac{B(u,1)}{2S^3(u,1)} = C_H \quad 0 \le u \le 1 \tag{3.67}$$

$$H(0,v) = \frac{B(0,v)}{2S^3(0,v)} = C_H, \quad H(1,v) = \frac{B(1,v)}{2S^3(1,v)} = C_H \quad 0 \le v \le 1 \tag{3.68}$$

where C_H is the constant mean curvature value. These equations can be rewritten as follows

$$B(u,0) - 2C_H\sqrt{S^2(u,0)}S^2(u,0) = 0, \quad B(u,1) - 2C_H\sqrt{S^2(u,1)}S^2(u,1) = 0 \quad 0 \le u \le 1 \quad (3.69)$$

$$B(0,v) - 2C_H\sqrt{S^2(0,v)}S^2(0,v) = 0, \quad B(1,v) - 2C_H\sqrt{S^2(1,v)}S^2(1,v) = 0 \quad 0 \le v \le 1 \quad (3.70)$$

Equations (3.69), (3.70) are the univariate irrational functions involving polynomials and square roots of polynomials which come from the normalization of the normal vector of the surface, see equation (3.7). $B(u,v)$ is a polynomial of degree $(5m-3, 5n-3)$ and $S^2(u,v)$ is a polynomial of degree $(4m-2, 4n-2)$.

Finally for the principal curvatures

$$\kappa(u,0) = \frac{B(u,0) \pm \sqrt{f_3(u,0)}}{2S^3(u,0)} = C_\kappa, \quad \kappa(u,1) = \frac{B(u,1) \pm \sqrt{f_3(u,1)}}{2S^3(u,1)} = C_\kappa, \quad 0 \le u \le 1 \quad (3.71)$$

$$\kappa(0,v) = \frac{B(0,v) \pm \sqrt{f_3(0,v)}}{2S^3(0,v)} = C_\kappa, \quad \kappa(1,v) = \frac{B(1,v) \pm \sqrt{f_3(1,v)}}{2S^3(1,v)} = C_\kappa, \quad 0 \le v \le 1 \quad (3.72)$$

where C_κ is the constant value of principal curvature and $f_3(u,v)$ is a polynomial function defined in equation (3.50). Equations (3.71) and (3.72) can be rewritten as follows

$$\begin{aligned}
B(u,0) &\pm \sqrt{f_3(u,0)} - 2C_\kappa S^2(u,0)\sqrt{S^2(u,0)} = 0 \quad 0 \le u \le 1 \\
B(u,1) &\pm \sqrt{f_3(u,1)} - 2C_\kappa S^2(u,1)\sqrt{S^2(u,1)} = 0 \quad 0 \le u \le 1 \qquad (3.73) \\
B(0,v) &\pm \sqrt{f_3(0,v)} - 2C_\kappa S^2(0,v)\sqrt{S^2(0,v)} = 0 \quad 0 \le v \le 1 \\
B(1,v) &\pm \sqrt{f_3(1,v)} - 2C_\kappa S^2(1,v)\sqrt{S^2(1,v)} = 0 \quad 0 \le v \le 1 \qquad (3.74)
\end{aligned}$$

Equations (3.73), (3.74) are the univariate irrational functions involving polynomials and two square roots of polynomials which come from the analytical expression of the principal curvature and normalization of the normal vector of the surface.

Since non-loop contour lines must start from a domain boundary and must end at a domain boundary point, the starting points for contour lines of curvature occur in pairs.

3.3.2 Mathematical Formulation of Contouring

Contour lines for constant curvature satisfy the following equation

$$C(u, v) = constant \tag{3.75}$$

where $C(u, v)$ is a curvature at the given point (u, v) on the B-spline surface. Curvature contouring takes place on the original B-spline surface and not on the Bézier patches. We now consider a space curve which lies on the surface represented by the parametric form $\mathbf{r}(t) = \mathbf{r}[u(t), v(t)]$. Differentiating the equation (3.75) with respect to t yields

$$C_u \dot{u} + C_v \dot{v} = 0 \tag{3.76}$$

where $\dot{u}$, $\dot{v}$ are the first derivatives with respect to t. ($\dot{u}$, $\dot{v}$) gives the direction of the contour line in parameter space. The solutions to the equation (3.76) are

$$\dot{u} = \zeta C_v, \quad \dot{v} = -\zeta C_u \tag{3.77}$$

where ζ is an arbitrary non zero factor that can be chosen to provide arc-length parametrization as follows

$$\zeta = \pm \frac{1}{\sqrt{C_u^2 + C_v^2}} \tag{3.78}$$

C_u and C_v are evaluated on the B-spline surface.

Contour lines of Gaussian curvature for K=0 separates a patch into elliptic (concave and convex) and hyperbolic (saddle) regions [24]. This information is useful for $3D$ and $5D$ machining. Also the union of contour lines of maximum principal curvature for κ_{max}=0 and minimum principal curvature for κ_{min}=0 separate the region in a way similar to the contour lines $K = 0$.

We used the Trip Algorithm introduced by Preusser [28] to polygonize the area between contour lines. The points of the contour lines are computed successively by integrating the initial value problem for a system of coupled nonlinear differential equations (3.77) using the variable stepsize and variable order Adams method [26]. Starting points were computed by the method described in section (3.3.1). Accuracy of the contour line depends on the number of points used to represent the contour line by straight line segments. Note that for principal curvatures, C_u and C_v become singular at umbilical point, therefore, we avoid the contour level which is equivalent to the curvature value at the umbilics.

Chapter 4

Parallel Processing - Parallel Virtual Machine

4.1 Introduction and Motivations

In the area for CAD, several algorithms require a specific task to be repeated several times. Examples include the repeated intersection of parametric surfaces, [4] and, in our case, the repeated extraction of differential properties from several Bézier patches. The nature of the algorithms, make parallel processing very appealing. In the MIT Ocean Engineering Design Laboratory that the present work was implemented, no single computer has the ability to process in parallel. Under certain circumstances though, the available computers can be linked to create a pseudo or virtual parallel computer. In order to do so, Parallel Virtual Machine was used.

Parallel Virtual Machine (PVM) is a software system that enables a collection of heterogeneous computers to be used as a coherent and flexible concurrent computational resource, [10], [22], [7]. The individual computers may be shared- or local-memory multiprocessors, vector supercomputers, specialized graphics engines, or scalar workstations, and may be interconnected by a variety of networks, such as ethernet. PVM support software executes on each machine in a user-configurable pool, and presents a unified, general, and powerful computational environment for concurrent applications. User programs written in C, C++ or Fortran, are provided access to PVM through the use of calls to PVM library routines for functions such as process initiation, message transmission and reception, and synchronization via barriers or rendezvous. Users may optionally control the execution location of specific application components. The PVM system transparently handles message routing, data conversion for incompatible architectures, and other tasks that are necessary for operation in a heterogeneous, network environment.

PVM is ideally suited for concurrent applications composed of many interrelated parts. PVM is particularly effective for heterogeneous applications that exploit specific strengths of individual machines on a network. As a loosely coupled concurrent supercomputer environment, PVM is a viable scientific computing platform. PVM system has been used for a number of applications such as molecular dynamics simulations, superconductivity studies, distributed fractal computations, matrix algorithms, and in the classroom as the basis for teaching concurrent computing.
The system is composed of,

- Pvmd daemon program
- Libpvm programming library
- Application Components

The Pvmd daemon program is mainly a message router, but is also a source and sink of messages. It runs on each host of virtual machine and provides inter-host point of contact. It authenticates tasks and provides fault detection. In general, the Pvmd daemon is more robust than application components. The Libpvm programming library is linked with each application component and provides the low-level PVM "syscalls". It provides the functions for data transferring and implementation of the parallel processing. The application components are written by the user in PVM message-passing calls and are executed as PVM "tasks".

The PVM system uses its own terminology. For instance, a *Host* is a physical machine (Unix workstation). One or more hosts compose a *Virtual Machine*. A *Process* is a program, data or stack like a Unix process or a node program. A *Task* is a PVM process, a small unit of computation. Finally, a *Message* is an ordered list of data sent between tasks.

4.2 Using PVM

In order to initialize PVM, we create a file, named *hostfile*, that contains the names and addresses of all the hosts that are going to be used. All the hosts listed in the hostfile will be automatically added on the virtual machine, unless they are preceded by an ampersand (&). In this case, the host will not be added automatically, but the user may elect to do so manually through the PVM console. The user will be prompted for the passwords on each machine. After the PVM initialization, all the available hosts are interconnected and form the *Virtual Machine.*

The PVM system provides a console for monitoring the process. Through the console, the user can add or delete hosts, monitor individual processes running and terminate either these processes, or the whole system.

There are usually two different categories of application components, the master and the slave processes. The master process is equivalent to the main program in regular programming, while the slave processes are equivalent to subroutines. In conventional programming, variables are passed from the main program to the subroutines and the results from the subroutines to the main program. Similarly, in PVM programming, variables that are needed for a *Task*, are passed from the master process to the slave processes and the results are passed back from the slaves to the master process. The passing of the information is done through an ordered list of data (*Messages*) and with the aid of the Libpvm "syscalls". Unfortunately, "syscalls" allow the passing of integers, floats and characters which prevents the user to pass directly from the master process to the slaves, whole structures and classes. Structures and classes must be decomposed to simple elements (floats and integers), passed with the aid of "syscalls" and then recomposed again on the other side of the system. This is one of the drawbacks of using PVM. Each *Task* is uniquely identified (per virtual machine), and consequently the right data are send to the right *Task*. The tasks are similar in nature to each other. They have the same input and output variables. In other words, they behave like different calls to the same subroutine.

4.3 Programming Procedure

From the programming point of view, the master process must first establish contact with the PVM daemons. It will then know how many hosts are available. The number of tasks is preset by the needs of the algorithm. If the number of tasks is smaller than the number of hosts, some hosts, the last ones in the hostfile list, will be idle. For this reason, it is important to list the hosts in descending computational power in the hostfile in order to maximize performance. This excludes the home machine which in any case must be listed first.

If the number of hosts is smaller than the number of tasks, each host will be assigned a task and the remaining tasks will wait for the first host to finish its previous assignment. It is important to note here that as soon as one host terminates its task, it is assigned a new task. There is no need for all hosts to terminate their tasks before new tasks can be assigned.

After a task has been assign to a host, the data transfer needs to take place. The variables needed to perform the task are transferred from the master program to the slave program. This later program runs on the particular host selected. The data must be transferred as integers, floats and characters. These data are received from the slave program in the exact same order as they were send from the master program. The slave program executes the task and the results are send back to the master program. The master program receives the results and signals that the particular host has terminated its task and that it is therefore free to be assigned a new one. This process is repeated until no more tasks remain.

Chapter 5

Implementation

5.1 Introduction

The interrogation procedures described in the previous chapters were implemented in a C++ computer code. Basic input to this program is the B-Spline surface, described mathematically by the orders of the spline (degree + 1) in each parametric direction, their knot vectors and control points. Output of the program is the color coded contour maps of the surface's curvatures (Gaussian, mean, maximum principal, minimum principal and root mean square curvatures).

Since the B-spline surface is a piecewise polynomial and a powerful generalization of polynomial Bézier surfaces, it is preferable to first subdivide it into several Bézier patches, and perform the computations in polynomials, and then assemble the results back to the original form.

5.2 Methodology

5.2.1 Major Libraries and Basic Classes

The computer code is supported by several libraries. The most important one is the *Geometry Library* that defines classes related to B-splines and Bézier curves and surfaces and supports their operations. The library provides a number of routines, including operators to read and write the surfaces in terms of their mathematical representation, and functions to evaluate the derivative and integral Functions for subdividing B-splines to Bézier patches are also available. The *Geometry Library* also provides means for representing polynomial equations and supports their operations. The *Geometry Library* was assembled by the author, in part from pieces of pre-existing code, [17], [32]. Modifications were needed to be made in order to generalize the functions and make them fit the needs of the program.

In the course of this work, other minor libraries were also developed. Their description will come latter, as the methodology for the computer coding unfolds. In addition, several other, pre-existing libraries were used, and will be mentioned later.

In addition to the classes residing in the *Geometry Library* used to describe mathematically free-form surfaces, the most important class developed is called *solution.* It refers to any point on the surface, (u, v) in parametric space, along with its properties. This class is referred to as such, because most of the points that belong to it are solutions to a single or several polynomial equations. The fields of this class include, the local (withing the particular Bézier patch) and global (on the whole B-spline surface) coordinates in parametric space, (u, v) and their classification as a maximum, minimum, saddle, umbilic or regular point. Whenever we refer to a point on the surface in question, we refer to the particular values of this class.

5.2.2 The Main Program

The main program, reads a B-spline surface from a file. The B-spline surface is represented by a class defined in the *Geometry Library.* It is then subdivided to Bézier patches using the Oslo algorithm, [3], [5], [16] which also resides in the *Geometry Library.* If the surface in question is a Bézier surface, the decomposition will yield one Bézier patch.

The Bézier patches obtained by the Oslo decomposition, [16], are dynamically allocated in an array, and then each is checked for stationary points (local maximum, minimum and saddle points). Each Bézier patch is examined, one at a time, as part of a loop. The mathematical representation of the Bézier surface yields the system of polynomial equations necessary for the evaluation of the stationary points within the patch, Eq. (3.28), and the single equation for the evaluation of the stationary points on the boundaries, Eq. (3.27). The mathematical formulation is presented in chapter 3 and appendix A. The formulated equations were solved by means of a library function developed by Hu, [13], [14], which was itself based on an earlier implementation based on Maekawa [18] and Sherbrooke, [29]. This function is based on the Bernstein subdivision method, and is reviewed in Chapter 2. Solutions to these equations yield the locations for the stationary points in terms of local coordinates, parametric u, v values of the particular Bézier patch examined. A transformation takes place to convert these locations in parametric u, v values of the original B-spline.

Once all the stationary points have been found, the global maximum and minimum curvatures of the B-spline, and thus the range of values of curvature can be found. The range of curvature values is then subdivided to a preset number of increments. These increments represent the number of color coded bands required for the interrogation of the surface. After this procedure is completed, the values of curvature for the constant value contours is set, and the program proceeds to find their respective starting points.

Starting points for constant curvature of height value contours are found along the boundaries of the B-spline surface as well as on isoparametric lines passing through local maxima and minima. Either $u = const$ or $v = const$ is selected for the isoparametric lines. Note that isoparametric lines need not pass through the saddle or umbilic points (that are not extrema). Computations are again performed on Bézier patches. Each Bézier patch containing a boundary of the original surface or containing part of an isoparametric line, leads to certain nonlinear equations, (see section (3.3.1)), that are solved for the particular starting points, Fig. (5-1). Patches that do not contain a B-spline boundary and an isoparametric line, are not involved in the computation. Solutions to these equations yield the locations of the starting points in terms of local (Bézier patch) coordinates. A similar transformation to the one performed earlier for the stationary points, is required (to bring

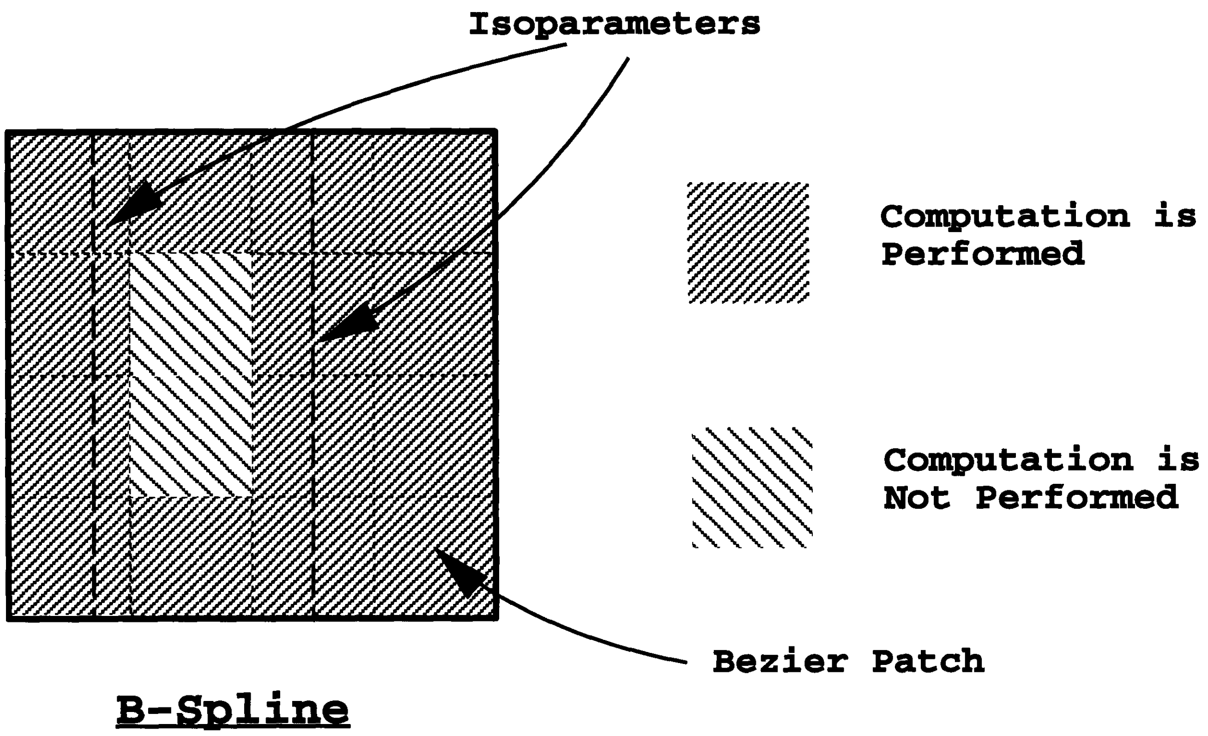


Figure 5-1: B-Spline Decomposition - Finding Starting Points

the solutions to the u-v parameter space of the B-spline surface patch). The mathematical formulation for the equations corresponding to the starting points, is presented in chapter 3 and appendix A. Each starting point is represented by the class *solution*, and includes information on the value of the curvature.

The last major computational procedure involves tracing of each constant value contour, from one starting point to another. The isoparametric lines abstractly subdivide the original B-spline surface into several regions. Each region is treated independently. The starting points on each region are sorted with respect to their counterclockwise distance from the lower left corner, Fig. (5-2). This is necessary for the polygonization process. Corner points are also allocated. A starting point on an isoparameter line will be allocated for both regions the isoparameter belongs to. For each region, all the starting points are loaded in a doubly linked list of *solutions*, so that the neighbors of every single point are known. The first

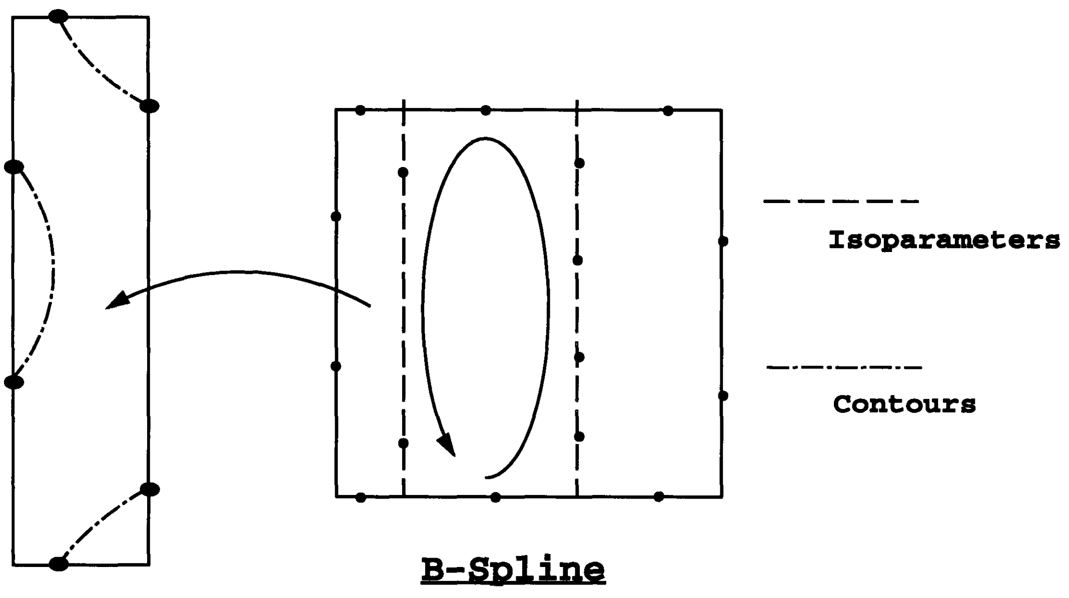


Figure 5-2: Tracing of the Contour Lines

and last points are identical, the lower left corner point. The integration is performed by using the Adams method, implemented by a NAG routine [26]. The integration step is an important parameter. Depending on the problem, the integration step takes values 15 to 50 times smaller than the width of the region (in the parameter space). Integration is initiated at one starting point and ends at another. Care is taken so that each contour is traced only once. For each particular region, points representing the constant value contours, including the starting points, are stored in a linked list of doubly linked lists of *solutions*. This data structure is appropriate for the polygonization.

After the integration part is completed, the constant curvature contours are available. To obtain the color coded contour regions we proceed as follows. Polygonization takes place, using a variation of the Trip algorithm [28]. The polygonization converts the complicated data structure and creates polygons with particular color code, according to their respective values of height or curvature. The output is written in a file and visualized on the screen in the form of color coded contour regions, examples of which are found in the next chapter.

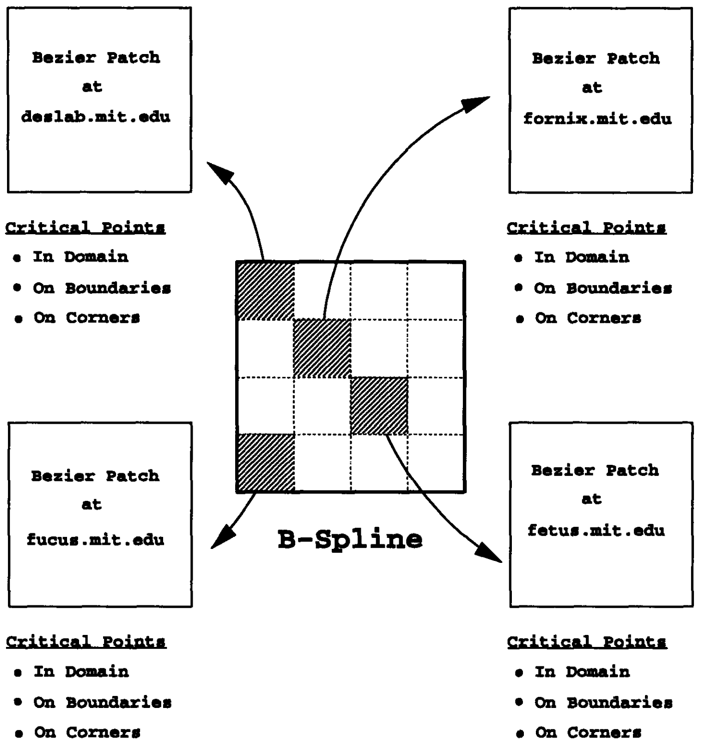


Figure 5-3: B-Spline Decomposition - Finding Stationary Points using Parallel Programming; deslab, fornix, fucus and fetus.mit.edu are names for various MIT Design Laboratory workstations

5.2.3 The Master Program - Parallel Programming

The decomposition of the original B-spline to several Bézier patches, allows the implementation of parallel programming. The program described above has *three* major computational parts: *finding the stationary points, finding the starting points and integrating for the contours.* All three parts can be performed in parallel.

To find the starting points in parallel, each Bézier patch that has been loaded on the dynamically allocated array, is send to a different *host* as part of an independent *task*, Fig. (5-3). Each *task* returns the stationary points of the corresponding Bézier patch. A transformation takes place so as to get the stationary points in terms of global coordinates on the original B-spline surface patch.

Starting points are also found using parallel processing. Each Bézier patch that needs

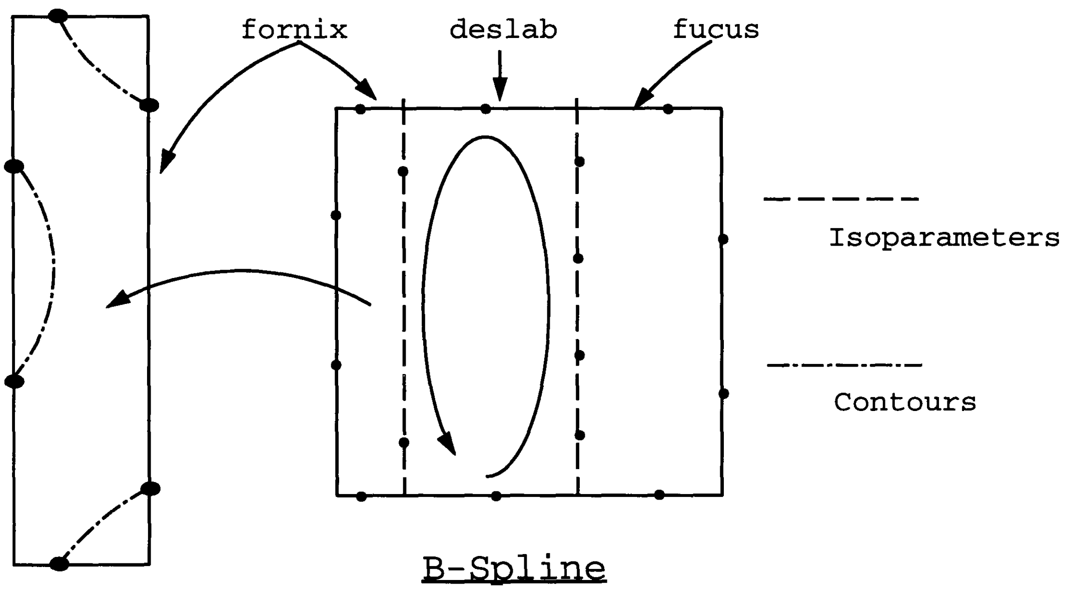


Figure 5-4: Tracing of the Contour Lines - Parallel Programming

to be examined, Fig. (5-1), is assigned to a different *host* as part of a separate *task*. Results yield the starting points on each corresponding Bézier patch, so a transformation is also needed.

Finally, the integration process can be performed in parallel. This time it is not the Bézier patches that provide the subdivision to the large problem, but rather each different region resulting by the tracing of isoparameters going through the maxima and minima of the B-spline surface curvature, Fig. (5-4). Each region is treated independently and assigned to a different *host*.

5.2.4 Implementation of Rounded Interval Arithmetic

The implementation described so far, was initially developed using double precision floating point arithmetic. Floating point arithmetic is associated with uncontrollable numerical error. Due to this fact, inaccuracies in the formulation and solution of the interrogation equations, grow and cause possible loss of a root. Maekawa [18] explains how solutions to simple polynomial equations using floating point arithmetic, can be incomplete in terms of the number of roots found. For this reason, the same implementation was developed using rounded interval arithmetic.

An introduction to the features and advantages of rounded interval arithmetic is presented in chapter 2. In short, rounded interval arithmetic, coupled with the Bernstein subdivision method for the solution of non-linear systems of polynomial equations, guarantees not to miss any roots.

There are two important tasks that need to be performed in rounded interval arithmetic: the formulation of the equations, and the process of their solution. For this reason, a new class called *isolution* was created. It has the exact same features of the class *solution* but instead of using fields expressed as floats, it uses fields expressed as rounded intervals. Similarly, a *Geometry Library* using rounded interval arithmetic was created. Interval arithmetic was used to formulate the system of equations for the stationary points and the equations for the starting points. In both cases an interval arithmetic solver based on the Bernstein subdivision method was used. After the starting points were found, the intervals were transformed into floats and integrations were performed as previously presented.

This method guarantees that no roots will be missed. However its major drawback is the extensive computational time required. The same equation could take as much as 20 times more of cpu time to solve in interval arithmetic than in floating point arithmetic.

Chapter 6

Examples and Applications - Conclusions and Recommendations

6.1 Introduction

To illustrate continuous surface decomposition, we used four different B-spline surfaces.

A wireframe representation of these surfaces is shown in Figs. (6-1) to (6-4). The knot vectors and control points for these surfaces are given in Appendix C. Figure (6-1) shows a sinusoidal surface which is a single bicubic Bézier patch. Figure (6-2) shows a surface which is also a single bicubic Bézier patch. Figure (6-3) shows a hat-like surface which consists of four bicubic Bézier patches, two in the u-direction and two in the v direction (2x2). Finally, Fig. (6-4) shows a sinusoidal surface which consists of 49 (7x7) bicubic Bézier patches.

The results presented here involve the creation of color coded curvature maps for the Gaussian, mean, maximum principal, minimum principal and root mean square curvatures, and some information about the computational speed required, with particular attention to the speed-up using parallel processing.

6.2 Contours and Color Coded Curvature Maps

For each surface, we present here the Gaussian, mean, maximum principal, minimum principal and root mean square curvatures. In each case, in addition to the color coded contour maps, a schematic figure showing the starting points, the isoparametric subdivision and the constant curvature contour lines, is presented. A table showing the stationary points for the particular curvature is also included. Global maxima and minima in the tables are shown with bold characters.

For the color coded curvature contour map, the following convention is applied: the global minimum takes the color blue, the global maximum the color yellow and everything in between is colored proportionally. The range of curvature is divided equally to a preset number of increments.

Figures (6-5) to (6-9) and Tables (6.1) to (6.5) present the results for the single Bézier patch sinusoidal surface shown in Fig. (6-1). The upper part of Fig. (6-5) presents the constant Gaussian curvature contours. We can distinguish the two maxima that reside within the domain. All stationary points are shown in Table (6.1). Isoparametric lines (at constant u) pass through those maxima, dividing the surface into three regions. Since the surface is a single Bézier patch, parallel processing is not needed for the computation of the stationary and starting points. The integration however is done in parallel, assigning different regions (in this case three) to different hosts. We can also identify the steep curvature gradient at the two lower corners of the surface. This type of situation could lead to complications, since starting points are very close to one another and could be considered identical. Care is also taken regarding the selection of the integration step. Indeed, large integration step will cause the program to fail while trying to connect two starting points very close to each other. The lower part of Fig. (6-5) shows the color coded Gaussian curvature maps. The large curvature gradient at the two lower corners is evident.

The upper part of Fig. (6-6) presents the constant mean curvature contours. Inside the domain, this surface shows one maximum and one minimum which subdivide it into three regions for parallel processing during integration. All the stationary points for the mean curvature for this surface are shown in Table (6.2). The lower part of Fig. (6-6) shows the color coded contour map.

Figure (6-7) shows the curvature contours and the color coded curvature maps for the maximum principal curvature. Table (6.3) presents all the stationary points for this case. We can see that for the maximum principal curvature, the surface contains two extrema within the domain and thus, two isoparameters divide the patch into three different regions. The lower right corner presents a steep maximum principal curvature gradient. The global maximum is located at the two lower corners.

Figure (6-8) shows the corresponding results for the minimum principal curvature. Table (6.4) presents all the stationary points for this case. The observations that can be made are very similar to the ones for the maximum principal curvature. This time the global minimum is located at the two lower corners. Two extrema points can be found inside the domain, subdividing the surface patch into three regions.

Figure (6-9) shows the corresponding results for the root mean square curvature. Table (6.5) presents all the stationary points for this case. Three local extrema are distinguished, one minimum (global) and two maxima. The global maximum is located at the two lower corners. The results for this case were produced by using rounded interval arithmetic only. The solver using floating point arithmetic kept loosing the global minimum at the middle, due to numerical error as was previously discussed. The usefulness of rounded interval arithmetic is evident.

Table (6.6) presents all the umbilical point for this surface. None of those umbilical points is a maximum or a minimum according to the criterion in Appendix B.

Figure (6-2) shows a single Bézier patch surface developed for demonstration purposes. Figures (6-10) to (6-14) show the curvature contours and color coded curvature maps for the Gaussian, mean, maximum principal, minimum principal and root mean square curvatures for that surface. Tables (6.7) to (6.11) present the stationary points for the same curvatures for this surface. Figure (6-10) shows the constant Gaussian curvature contour lines. Table

(6.7) presents all the stationary points for this case. We can distinguish the five extrema as well as the isoparameters associated with them, subdividing the surface into six regions. Each region will be treated as a *task* in parallel processing during the integration stage. Note the importance of assigning the integration step to be proportional to the width (in the u-direction) of each region, since in this example, the regions have wide ranges of widths. The color coded curvature map for this curvature is shown on the lower part of Fig. (6-10). Although the curvature range is not significant, several details can be noticed.

Figure (6-11) shows the results for the mean curvature. Table (6.2) presents all the stationary points for this case. There is only one minimum within the boundary, therefore the surface is subdivided into two regions only. Again the curvature range is not considerable.

Figure (6-12) shows the results for the maximum principal curvature. Table (6.3) presents all the stationary points for this case. We can distinguish three extrema points within the domain, two minima (one of the global) and a maximum (global). Table (6.9) presents all the stationary points in detail.

Figure (6-13) shows the results for the minimum principal curvature. Table (6.4) presents all the stationary points for this case. A local maximum and a local minimum exist within the boundary. The global minimum is located on the boundary $u = 0$ and the global maximum on the two lower corners.

Figure (6-14) shows the results for the root mean square curvature. Table (6.11) presents all the stationary points for this case. Three extrema exist within the boundary, one minimum and two maxima. Both global minimum and maximum are located on the boundary.

The next two examples involve multipatch cases. Figure (6-3) shows a wireframe representation of a hat-like surface. The surface is decomposable into four Bézier patches. Figures (6-15) to (6-17) and Tables (6.12) to (6.14) show the results for the Gaussian, mean and root mean square curvatures for this surface. The principal curvatures were not computed for this case. The program was terminated after 24 hours of computing the stationary points (for each Bézier patch) on a 150 MHz workstation. The reason for the delay is attributed to the characteristics of the surface, which is particularly flat at the center of the surface. This causes an excessive number of binary subdivisions leading to tremendous memory requirements which can exhaust computer resources even if the requested accuracy is not very strict.

In the upper part of Fig. (6-15) we can distinguish how the B-spline patch was subdivided into four Bézier patches. The two black lines, at $u = 0.5$ and $v = 0.5$, subdivide the B-spline into four Bézier patches. Finding the stationary and starting points can be performed in parallel, with each Bézier patch sent to a different *host*. Table (6.12) presents all the stationary points for the Gaussian curvature. The surface has a local (and, as it turns out, global) maximum in the center, and four (identical) local (and global) minima. Three isoparameters subdivide the original B-spline into four regions (red lines). Note that the isoparameter $u = 0.5$ is identical to the Bézier patch subdivision line, and that in two cases, one isoparameter line passes through two minima. Therefore, although we have five extrema within the domain, only three isoparameter lines are needed to subdivide the surfaces to four regions. Integration in the four different regions is performed in parallel. The lower part of Fig. (6-15) shows the color coded Gaussian curvature map.

Figure (6-16) shows the results for the mean curvature. Table (6.13) presents all the stationary points for this case. Similar observation with the results of the Gaussian curvature can be made. The global maximum is located at the center of the surface. Again, integration is performed in parallel on two different *hosts*. Note that although the lower part of Fig. (6-16) indicates the existence of eight minima located on the boundary, only four were found using the floating point arithmetic solver. Nevertheless, in this particular case, the correct curvature map was developed, and this omission did not influence the final result.

Figure (6-17) shows the results for the root mean square curvature. Table (6.14) presents all the stationary points for this case. This surface is very rich in properties. Nine extrema exist within its domain, although only five isoparameters $u = const$ are needed to subdivide it. Four are global maxima and four are global minima whereas the last one (in the middle) is a local minimum. Twelve stationary points are found on the boundary.

Figure (6-4) shows a more complicated sinusoidal surface. This surface is decomposable into 49 Bézier patches, 7 in each direction. Figures (6-18) to (6-22) and Tables (6.15) to (6.20) show the results for this specific case. The upper part of Fig. (6-18) shows the constant Gaussian curvature contours. We can distinguish the subdivision boundaries of the 49 Bézier patches. It can be noted that not all the patches are of the same size. This surface is very rich in differential geometry properties, having six local minima and two local maxima. Table (6.15) presents all the stationary points for this case. Five $u = const.$ isoparameter lines are needed. They subdivide the surface into six regions for parallel integration. From the lower part of Fig. (6-18) we can see that although the range of the curvature is large, on most of the surface the Gaussian curvature is around zero.

Figure (6-19) shows the corresponding results for the mean curvature. Table (6.16) presents all the stationary points for this case. The mean curvature appears to be more complicated than the Gaussian. Six extrema have been found: three minima and three maxima. Three isoparameter lines divide the surface into four regions for parallel integration.

Figure (6-20) shows the corresponding results for the maximum principal curvature. Table (6.17) presents all the stationary points found for this case. Note that Table (6.17) does not contain the global minimum. This is due to the fact that the global minimum was missed by the solver operating in floating point arithmetic. No interval arithmetic calculation was performed for that surface. Nevertheless, the global minimum is very close to the corner of one of the Bézier patches, whose curvature is automatically evaluated. In that way, the full range of the curvature was captured, and the results shown in Fig. (6-20) are accurate. Seven local maxima are found within the domain. Seven $u = const$ isoparameters subdivide the surface.

Figure (6-21) shows the corresponding results for the minimum principal curvature. Table (6.18) presents all the stationary points found for this case. This time the global maximum is missing, for exactly the same reason as above. The results are very similar with the maximum principal curvature, but completely opposite.

Figure (6-22) shows the corresponding results for the root mean square curvature. Several extrema are found within the domain and on the boundary, see Tables (6.19) and (6.20). The global maximum is located very close to the boundary, whereas the global minimum in the center of the surface.

u	v	Gaussian Curvature	Classification
0.805	0.374	10.297	**M A X I M U M**
0.195	0.374	10.297	**M A X I M U M**
0.500	0.440	0.000	S A D D L E
0.500	0.000	-20.250	M I N I M U M
0.500	1.000	-7.290	M I N I M U M
0.000	0.440	0.000	M A X I M U M
1.000	0.440	0.000	M A X I M U M
0.789	0.000	0.000	M A X I M U M
0.211	0.000	0.000	M A X I M U M
0.789	1.000	0.000	M A X I M U M
0.211	1.000	0.000	M A X I M U M
0.000	0.000	-81.000	**M I N I M U M**
1.000	0.000	-81.000	**M I N I M U M**

Table 6.1: Stationary Points - Gaussian Curvature - Sinusoidal Surface - 1 Bézier Patch

u	v	Mean Curvature	Classification
0.810	0.414	-4.056	**M I N I M U M**
0.190	0.414	4.056	**M A X I M U M**
0.000	0.440	0.607	M I N I M U M
1.000	0.861	-1.155	M I N I M U M
1.000	0.089	-1.155	M I N I M U M
0.884	0.000	-0.539	M I N I M U M
0.681	0.000	-0.539	M I N I M U M
0.211	0.000	0.524	M I N I M U M
0.789	1.000	-0.121	M I N I M U M
0.000	0.861	1.155	M A X I M U M
0.000	0.089	1.155	M A X I M U M
1.000	0.440	-0.607	M A X I M U M
0.789	0.000	-0.524	M A X I M U M
0.319	0.000	0.539	M A X I M U M
0.116	0.000	0.539	M A X I M U M
0.211	1.000	0.121	M A X I M U M

Table 6.2: Stationary Points - Mean Curvature - Sinusoidal Surface - 1 Bézier Patch

u	v	Max. Principal Curvature	Classification
0.789	0.303	-1.665	**M I N I M U M**
0.187	0.440	6.607	M A X I M U M
0.378	0.851	2.470	S A D D L E
0.082	0.802	4.504	S A D D L E
0.321	0.157	3.276	S A D D L E
0.114	0.184	5.127	S A D D L E
0.211	0.000	1.047	M I N I M U M
0.211	1.000	0.242	M I N I M U M
0.789	0.000	0.000	M I N I M U M
0.789	1.000	0.000	M I N I M U M
0.478	0.000	4.569	M A X I M U M
0.491	1.000	2.704	M A X I M U M
0.000	0.043	7.952	M A X I M U M
0.000	0.908	5.232	M A X I M U M
0.000	0.000	9.000	**M A X I M U M**
1.000	0.000	9.000	**M A X I M U M**

Table 6.3: Stationary Points - Maximum Principal Curvature - Sinusoidal Surface - 1 Bézier Patch

u	v	Min. Principal Curvature	Classification
0.813	0.440	-6.607	M I N I M U M
0.211	0.303	1.665	**M A X I M U M**
0.918	0.802	-4.504	S A D D L E
0.622	0.851	-2.470	S A D D L E
0.886	0.184	-5.127	S A D D L E
0.679	0.157	-3.276	S A D D L E
0.509	1.000	-2.704	M I N I M U M
0.522	0.000	-4.569	M I N I M U M
1.000	0.043	-7.952	M I N I M U M
1.000	0.908	-5.232	M I N I M U M
0.211	0.000	0.000	M A X I M U M
0.211	1.000	0.000	M A X I M U M
0.789	0.000	-1.047	M A X I M U M
0.789	1.000	-0.242	M A X I M U M
0.000	0.000	-9.000	**M I N I M U M**
1.000	0.000	-9.000	**M I N I M U M**

Table 6.4: Stationary Points - Minimum Principal Curvature - Sinusoidal Surface - 1 Bézier Patch

u	v	κ_{rms} Curvature	Classification
0.500	0.500	0.000	**M I N I M U M**
0.188	0.433	6.771	M A X I M U M
0.812	0.433	6.771	M A X I M U M
0.086	0.794	4.515	S A D D L E
0.110	0.176	5.159	S A D D L E
0.325	0.150	3.282	S A D D L E
0.351	0.812	2.552	S A D D L E
0.649	0.812	2.552	S A D D L E
0.675	0.150	3.282	S A D D L E
0.890	0.176	5.159	S A D D L E
0.914	0.794	4.515	S A D D L E
0.000	0.440	1.213	M I N I M U M
0.211	0.000	1.047	M I N I M U M
0.211	1.000	0.242	M I N I M U M
0.789	0.000	1.047	M I N I M U M
0.789	1.000	0.242	M I N I M U M
1.000	0.440	1.213	M I N I M U M
0.000	0.000	12.728	**M A X I M U M**
1.000	0.000	12.728	**M A X I M U M**

Table 6.5: Stationary Points - RMS Curvature - Sinusoidal Surface - 1 Bézier Patch

u	v	Max. & Min. Principal Curvature	Classification
0.789	0.984	-0.267	U M B I L I C
0.789	0.052	-1.197	U M B I L I C
0.500	0.440	0.000	U M B I L I C
0.211	0.984	0.267	U M B I L I C
0.211	0.052	1.197	U M B I L I C

Table 6.6: Umbilical Points - Sinusoidal Surface - 1 Bézier Patch

u	v	Gaussian Curvature	Classification
0.770	0.404	-0.115	**M I N I M U M**
0.132	0.950	-0.098	M I N I M U M
0.324	0.032	-0.086	M I N I M U M
0.618	0.098	0.098	**M A X I M U M**
0.351	0.417	0.038	M A X I M U M
0.516	0.746	-0.050	S A D D L E
0.429	0.348	0.035	S A D D L E
0.000	0.944	-0.094	M I N I M U M
0.000	0.039	-0.015	M I N I M U M
1.000	0.385	-0.052	M I N I M U M
0.975	0.000	-0.008	M I N I M U M
0.326	0.000	-0.085	M I N I M U M
0.145	1.000	-0.096	M I N I M U M
0.000	0.377	-0.011	M A X I M U M
0.644	0.000	0.085	M A X I M U M
0.899	1.000	-0.016	M A X I M U M

Table 6.7: Stationary Points - Gaussian Curvature - 1 Bézier Patch

u	v	Mean Curvature	Classification
0.596	0.042	-0.476	M I N I M U M
0.940	0.979	-0.058	S A D D L E
0.319	0.719	-0.036	S A D D L E
0.466	0.211	-0.450	S A D D L E
0.000	0.449	-0.678	**M I N I M U M**
1.000	0.955	-0.061	M I N I M U M
0.607	0.000	-0.473	M I N I M U M
0.507	1.000	-0.216	M I N I M U M
1.000	0.256	0.165	**M A X I M U M**
0.009	0.000	0.019	M A X I M U M
0.948	1.000	-0.057	M A X I M U M

Table 6.8: Stationary Points - Mean Curvature - 1 Bézier Patch

u	v	Max. Principal Curvature	Classification
0.811	0.437	0.497	**M A X I M U M**
0.634	0.134	-0.130	**M I N I M U M**
0.430	0.717	0.179	S A D D L E
0.465	0.355	-0.056	S A D D L E
0.465	0.354	-0.056	S A D D L E
0.464	0.356	-0.056	S A D D L E
0.464	0.355	-0.056	S A D D L E
0.396	0.516	-0.108	U M B I L I C
0.000	0.584	0.021	M I N I M U M
0.000	0.334	0.011	M I N I M U M
0.670	0.000	-0.106	M I N I M U M
0.670	1.000	0.068	M I N I M U M
1.000	0.969	0.088	M I N I M U M
0.000	0.986	0.320	M A X I M U M
0.223	0.000	0.228	M A X I M U M
1.000	0.197	0.414	M A X I M U M

Table 6.9: Stationary Points - Maximum Principal Curvature - 1 Bézier Patch

u	v	Min. Principal Curvature	Classification
0.583	0.019	-0.859	M I N I M U M
0.396	0.517	-0.109	M A X I M U M
0.910	0.885	-0.209	S A D D L E
0.657	0.568	-0.211	S A D D L E
0.527	0.112	-0.853	S A D D L E
0.086	0.855	-0.312	S A D D L E
0.000	0.426	-1.346	**M I N I M U M**
0.482	1.000	-0.524	M I N I M U M
0.589	0.000	-0.858	M I N I M U M
1.000	0.780	-0.197	M I N I M U M
0.000	0.884	-0.310	M A X I M U M
0.937	1.000	-0.197	M A X I M U M
1.000	0.000	-0.042	**M A X I M U M**

Table 6.10: Stationary Points - Minimum Principal Curvature - 1 Bézier Patch

u	v	κ_{rms} Curvature	Classification
0.411	0.533	0.123	M I N I M U M
0.585	0.022	0.863	M A X I M U M
0.794	0.433	0.544	M A X I M U M
0.522	0.121	0.855	S A D D L E
0.583	0.715	0.331	S A D D L E
0.703	0.263	0.350	S A D D L E
0.000	0.781	0.401	M I N I M U M
0.000	0.039	0.175	M I N I M U M
0.928	0.000	0.115	**M I N I M U M**
0.932	1.000	0.214	M I N I M U M
0.000	0.948	0.434	M A X I M U M
0.000	0.449	1.364	**M A X I M U M**
0.475	1.000	0.533	M A X I M U M
0.592	0.000	0.862	M A X I M U M
1.000	0.323	0.454	M A X I M U M

Table 6.11: Stationary Points - RMS Curvature - 1 Bézier Patch

u	v	Gaussian Curvature	Classification
0.143	0.143	-15.157	**M I N I M U M**
0.143	0.857	-15.157	**M I N I M U M**
0.857	0.143	-15.157	**M I N I M U M**
0.857	0.857	-15.157	**M I N I M U M**
0.500	0.500	2.250	**M A X I M U M**
0.000	0.787	-8.818	M I N I M U M
0.787	0.000	-8.818	M I N I M U M
1.000	0.787	-8.818	M I N I M U M
0.787	1.000	-8.818	M I N I M U M
0.000	0.500	-1.843	M A X I M U M
0.500	0.000	-1.843	M A X I M U M
0.500	1.000	-1.843	M A X I M U M
1.000	0.500	-1.843	M A X I M U M

Table 6.12: Stationary Points - Hat-like Surface - Gaussian Curvature - 4 Bézier Patches

u	v	Mean Curvature	Classification
0.500	0.500	1.500	**M A X I M U M**
0.152	0.152	-0.404	S A D D L E
0.152	0.848	-0.404	S A D D L E
0.848	0.152	-0.404	S A D D L E
0.848	0.848	-0.404	S A D D L E
0.000	0.688	-1.103	**M I N I M U M**
0.688	0.000	-1.103	**M I N I M U M**
1.000	0.688	-1.103	**M I N I M U M**
0.688	1.000	-1.103	**M I N I M U M**
0.000	0.500	-0.816	M A X I M U M
0.500	0.000	-0.816	M A X I M U M
0.500	1.000	-0.816	M A X I M U M
1.000	0.500	-0.816	M A X I M U M

Table 6.13: Stationary Points - Hat-like Surface - Mean Curvature - 4 Bézier Patches

u	v	κ_{rms} Curvature	Classification
0.189	0.500	1.138	**M I N I M U M**
0.500	0.189	1.138	**M I N I M U M**
0.500	0.811	1.138	**M I N I M U M**
0.811	0.500	1.138	**M I N I M U M**
0.500	0.500	2.121	M I N I M U M
0.144	0.144	5.561	**M A X I M U M**
0.144	0.856	5.561	**M A X I M U M**
0.856	0.144	5.561	**M A X I M U M**
0.856	0.856	5.561	**M A X I M U M**
0.371	0.371	1.845	S A D D L E
0.371	0.629	1.845	S A D D L E
0.629	0.371	1.845	S A D D L E
0.629	0.629	1.845	S A D D L E
0.000	0.500	2.520	M I N I M U M
0.500	0.000	2.520	M I N I M U M
0.500	1.000	2.520	M I N I M U M
1.000	0.500	2.520	M I N I M U M
0.000	0.247	4.502	M A X I M U M
0.000	0.753	4.502	M A X I M U M
0.247	0.000	4.502	M A X I M U M
0.247	1.000	4.502	M A X I M U M
0.753	0.000	4.502	M A X I M U M
0.753	1.000	4.502	M A X I M U M
1.000	0.247	4.502	M A X I M U M
1.000	0.753	4.502	M A X I M U M

Table 6.14: Stationary Points - Hat-like Surface - RMS Curvature - 4 Bézier Patches

u	v	Gaussian Curvature	Classification
0.172	0.500	-76.141	M I N I M U M
0.241	0.007	-163.018	**M I N I M U M**
0.241	0.993	-163.018	**M I N I M U M**
0.759	0.007	-163.018	**M I N I M U M**
0.759	0.993	-163.018	**M I N I M U M**
0.828	0.500	-76.141	M I N I M U M
0.500	0.180	47.763	**M A X I M U M**
0.500	0.820	47.763	**M A X I M U M**
0.156	0.155	-1.521	S A D D L E
0.158	0.257	-1.544	S A D D L E
0.158	0.743	-1.544	S A D D L E
0.156	0.845	-1.521	S A D D L E
0.500	0.500	0.000	S A D D L E
0.844	0.155	-1.521	S A D D L E
0.842	0.257	-1.544	S A D D L E
0.842	0.743	-1.544	S A D D L E
0.844	0.845	-1.521	S A D D L E
0.000	0.086	-0.521	M I N I M U M
0.000	0.914	-0.521	M I N I M U M
0.000	0.500	-3.092	M I N I M U M
0.248	0.000	-156.371	M I N I M U M
0.248	1.000	-156.371	M I N I M U M
0.752	0.000	-156.371	M I N I M U M
0.752	1.000	-156.371	M I N I M U M
1.000	0.086	-0.521	M I N I M U M
1.000	0.500	-3.092	M I N I M U M
1.000	0.914	-0.521	M I N I M U M
0.000	0.166	-0.125	M A X I M U M
0.000	0.834	-0.125	M A X I M U M
0.500	0.000	-0.090	M A X I M U M
0.500	1.000	-0.090	M A X I M U M
1.000	0.166	-0.125	M A X I M U M
1.000	0.834	-0.125	M A X I M U M

Table 6.15: Stationary Points - Sinusoidal Surface - Gaussian Curvature - 49 Bézier Patches

u	v	Mean Curvature	Classification
0.063	0.363	-2.742	M I N I M U M
0.500	0.823	-7.307	**M I N I M U M**
0.937	0.363	-2.742	M I N I M U M
0.063	0.637	2.742	M A X I M U M
0.500	0.177	7.307	**M A X I M U M**
0.937	0.637	2.742	M A X I M U M
0.165	0.030	-4.061	S A D D L E
0.165	0.970	4.061	S A D D L E
0.155	0.500	0.000	S A D D L E
0.835	0.030	-4.061	S A D D L E
0.835	0.970	4.061	S A D D L E
0.845	0.500	0.000	S A D D L E
0.000	0.140	-5.322	M I N I M U M
0.000	0.334	-2.473	M I N I M U M
0.000	0.735	2.106	M I N I M U M
0.063	1.000	-0.249	M I N I M U M
0.259	0.000	-5.804	M I N I M U M
0.500	1.000	-0.119	M I N I M U M
0.741	0.000	-5.804	M I N I M U M
1.000	0.140	-5.322	M I N I M U M
1.000	0.334	-2.473	M I N I M U M
1.000	0.735	2.106	M I N I M U M
0.937	1.000	-0.249	M I N I M U M
0.063	0.000	0.249	M A X I M U M
0.000	0.265	-2.106	M A X I M U M
0.000	0.666	2.473	M A X I M U M
0.000	0.860	5.322	M A X I M U M
0.259	1.000	5.804	M A X I M U M
0.500	0.000	0.119	M A X I M U M
0.741	1.000	5.804	M A X I M U M
0.937	0.000	0.249	M A X I M U M
1.000	0.265	-2.106	M A X I M U M
1.000	0.666	2.473	M A X I M U M
1.000	0.860	5.322	M A X I M U M

Table 6.16: Stationary Points - Sinusoidal Surface - Mean Curvature - 49 Bézier Patches

u	v	Max. Principal Curvature	Classification
0.111	0.605	6.806	M A X I M U M
0.181	0.486	8.812	M A X I M U M
0.234	0.011	8.609	M A X I M U M
0.500	0.174	9.720	M A X I M U M
0.766	0.011	8.609	M A X I M U M
0.819	0.486	8.812	M A X I M U M
0.889	0.605	6.806	M A X I M U M
0.047	0.904	9.642	S A D D L E
0.197	0.214	0.731	S A D D L E
0.305	0.060	6.594	S A D D L E
0.354	0.354	4.982	S A D D L E
0.418	0.283	5.273	S A D D L E
0.500	0.799	-5.080	S A D D L E
0.582	0.283	5.273	S A D D L E
0.646	0.354	4.982	S A D D L E
0.695	0.060	6.594	S A D D L E
0.803	0.215	0.731	S A D D L E
0.953	0.904	9.642	S A D D L E
0.000	0.701	4.843	M I N I M U M
0.500	0.000	0.442	M I N I M U M
0.500	1.000	0.204	M I N I M U M
1.000	0.701	4.843	M I N I M U M
0.000	0.875	10.311	M A X I M U M
0.254	1.000	19.517	**M A X I M U M**
0.746	1.000	19.517	**M A X I M U M**
1.000	0.875	10.311	M A X I M U M
0.373	0.203	1.849	U M B I L I C
0.373	0.797	-1.849	U M B I L I C
0.500	0.072	1.994	U M B I L I C
0.500	0.247	4.661	U M B I L I C
0.500	0.500	0.000	U M B I L I C
0.500	0.753	-4.661	U M B I L I C
0.500	0.970	-0.815	U M B I L I C
0.627	0.203	1.848	U M B I L I C
0.627	0.797	-1.848	U M B I L I C

Table 6.17: Stationary Points - Sinusoidal Surface - Max. Principal Curvature - 49 Bézier Patches

u	v	Min. Principal Curvature	Classification
0.111	0.395	-6.806	M I N I M U M
0.181	0.514	-8.812	M I N I M U M
0.234	0.989	-8.609	M I N I M U M
0.500	0.826	-9.720	M I N I M U M
0.766	0.989	-8.609	M I N I M U M
0.819	0.514	-8.812	M I N I M U M
0.889	0.395	-6.806	M I N I M U M
0.047	0.096	-9.642	S A D D L E
0.197	0.786	-0.731	S A D D L E
0.305	0.940	-6.594	S A D D L E
0.354	0.646	-4.982	S A D D L E
0.418	0.717	-5.273	S A D D L E
0.500	0.201	5.080	S A D D L E
0.582	0.717	-5.273	S A D D L E
0.646	0.646	-4.982	S A D D L E
0.695	0.940	-6.594	S A D D L E
0.803	0.786	-0.731	S A D D L E
0.953	0.096	-9.642	S A D D L E
0.000	0.125	-10.311	M I N I M U M
0.254	0.000	-19.517	**M I N I M U M**
0.746	0.000	-19.517	**M I N I M U M**
1.000	0.125	-10.311	M I N I M U M
0.000	0.299	-4.843	M A X I M U M
0.500	0.000	-0.204	M A X I M U M
0.500	1.000	-0.442	M A X I M U M
1.000	0.299	-4.843	M A X I M U M
0.373	0.203	1.849	U M B I L I C
0.373	0.797	-1.849	U M B I L I C
0.500	0.072	1.994	U M B I L I C
0.500	0.247	4.661	U M B I L I C
0.500	0.500	0.000	U M B I L I C
0.500	0.753	-4.661	U M B I L I C
0.500	0.970	-0.815	U M B I L I C
0.627	0.203	1.848	U M B I L I C
0.627	0.797	-1.848	U M B I L I C

Table 6.18: Stationary Points - Sinusoidal Surface - Min. Principal Curvature - 49 Bézier Patches

u	v	κ_{rms} Curvature	Classification
0.279	0.171	0.978	M I N I M U M
0.286	0.291	1.416	M I N I M U M
0.286	0.810	1.416	M I N I M U M
0.279	0.829	0.978	M I N I M U M
0.572	0.002	0.477	M I N I M U M
0.572	0.536	0.000	**M I N I M U M**
0.572	0.998	0.477	M I N I M U M
0.844	0.171	0.978	M I N I M U M
0.837	0.291	1.416	M I N I M U M
0.837	0.810	1.416	M I N I M U M
0.844	0.829	0.978	M I N I M U M
0.172	0.536	12.340	M A X I M U M
0.261	0.002	21.120	M A X I M U M
0.261	0.998	21.120	M A X I M U M
0.572	0.176	10.877	M A X I M U M
0.572	0.824	10.877	M A X I M U M
0.862	0.002	21.120	**M A X I M U M**
0.862	0.998	21.120	**M A X I M U M**
0.828	0.536	12.340	M A X I M U M
0.046	0.096	9.646	S A D D L E
0.046	0.904	9.646	S A D D L E
0.391	0.070	6.741	S A D D L E
0.368	0.257	1.443	S A D D L E
0.368	0.845	1.443	S A D D L E
0.391	0.930	6.741	S A D D L E
0.486	0.386	5.443	S A D D L E
0.358	0.380	4.987	S A D D L E
0.358	0.703	4.987	S A D D L E
0.486	0.716	5.443	S A D D L E
0.653	0.386	5.443	S A D D L E
0.781	0.380	4.987	S A D D L E
0.781	0.703	4.987	S A D D L E
0.653	0.716	5.443	S A D D L E
0.732	0.070	6.741	S A D D L E
0.755	0.257	1.443	S A D D L E
0.755	0.845	1.443	S A D D L E
0.732	0.930	6.741	S A D D L E
0.954	0.096	9.646	S A D D L E
0.954	0.904	9.646	S A D D L E

Table 6.19: Stationary Points - Sinusoidal Surface - RMS Curvature - 49 Bézier Patches

u	v	κ_{rms} Curvature	Classification
0.000	0.002	0.745	M I N I M U M
0.000	0.313	4.439	M I N I M U M
0.000	0.536	2.487	M I N I M U M
0.000	0.789	4.439	M I N I M U M
0.000	0.998	0.745	M I N I M U M
0.572	0.000	0.486	M I N I M U M
0.572	1.000	0.486	M I N I M U M
1.000	0.002	0.745	M I N I M U M
1.000	0.313	4.439	M I N I M U M
1.000	0.536	2.487	M I N I M U M
1.000	0.789	4.439	M I N I M U M
1.000	0.998	0.745	M I N I M U M
0.000	0.139	10.667	M A X I M U M
0.000	0.355	5.396	M A X I M U M
0.000	0.728	5.396	M A X I M U M
0.000	0.861	10.667	M A X I M U M
0.269	0.000	21.080	M A X I M U M
0.269	1.000	21.080	M A X I M U M
0.854	0.000	21.080	M A X I M U M
0.854	1.000	21.080	M A X I M U M
1.000	0.139	10.667	M A X I M U M
1.000	0.355	5.396	M A X I M U M
1.000	0.728	5.396	M A X I M U M
1.000	0.861	10.667	M A X I M U M

Table 6.20: Stationary Points - Sinusoidal Surface - RMS Curvature - 49 Bézier Patches (Continue)

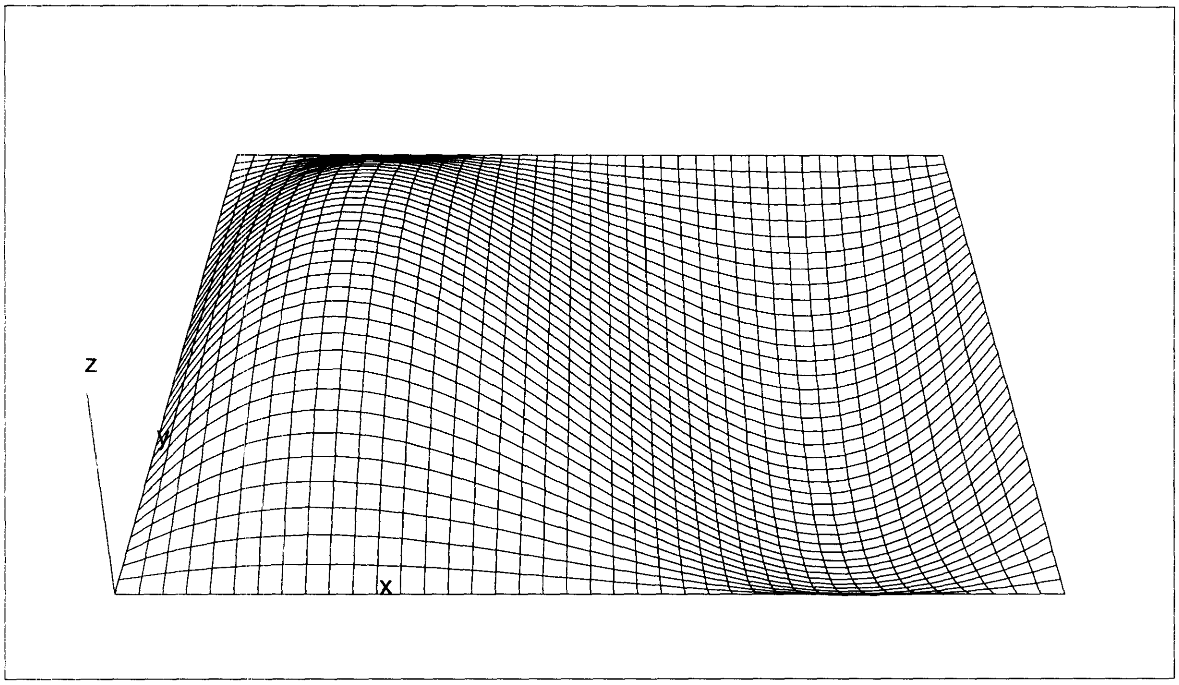


Figure 6-1: Sinusoidal Surface - 1 Bézier Patch

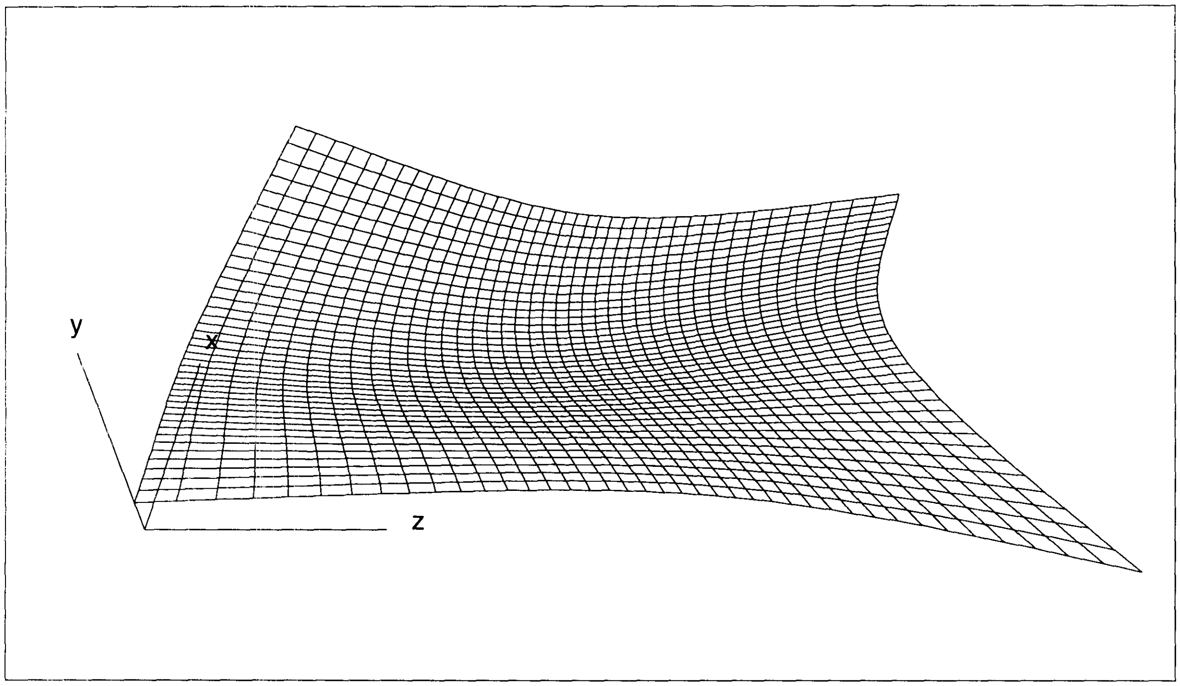


Figure 6-2: Random Data Surface - 1 Bézier Patch

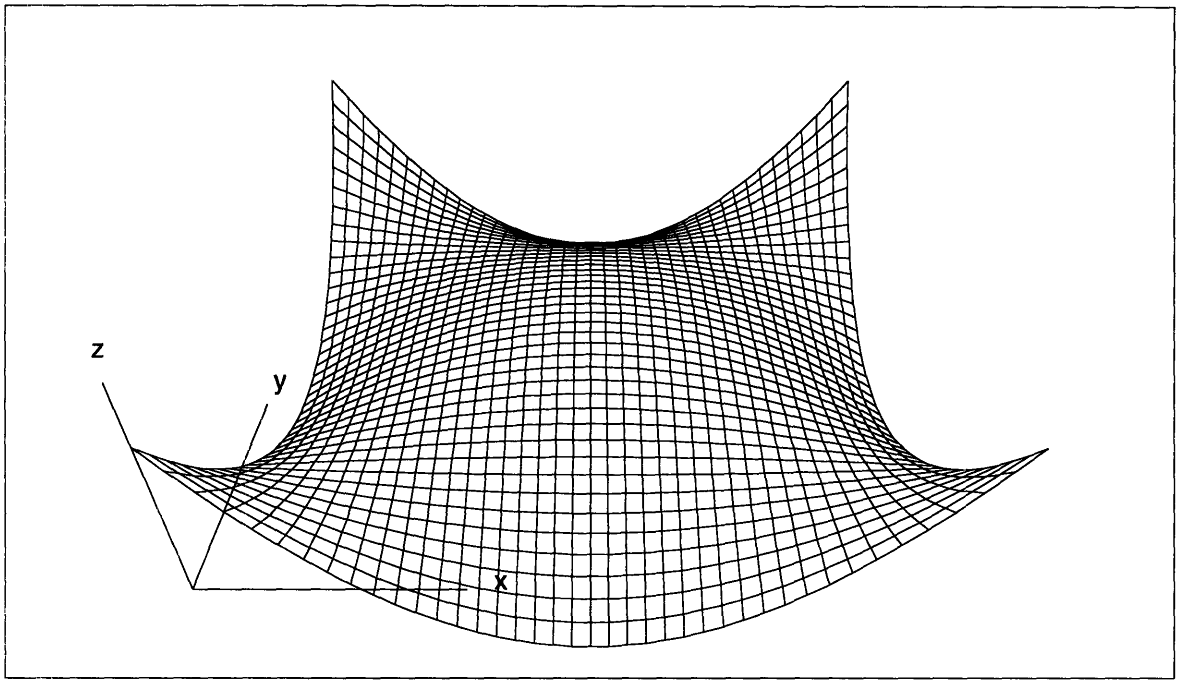


Figure 6-3: Hat-like Surface - 2 x 2 Bézier Patches

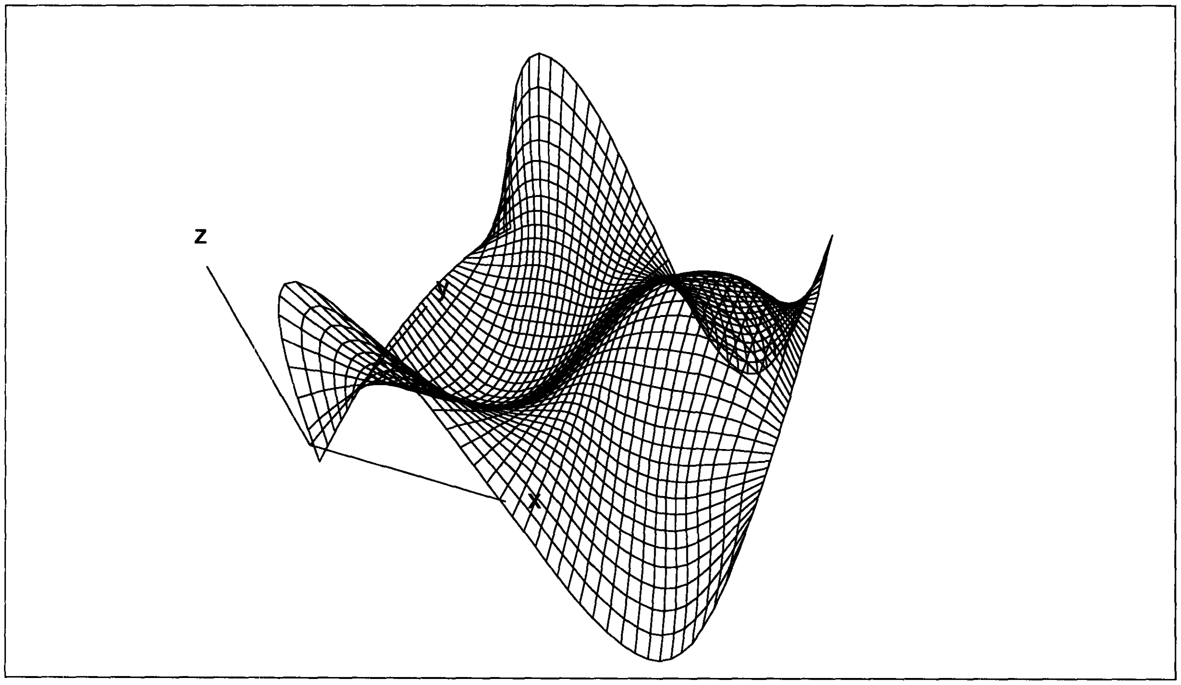
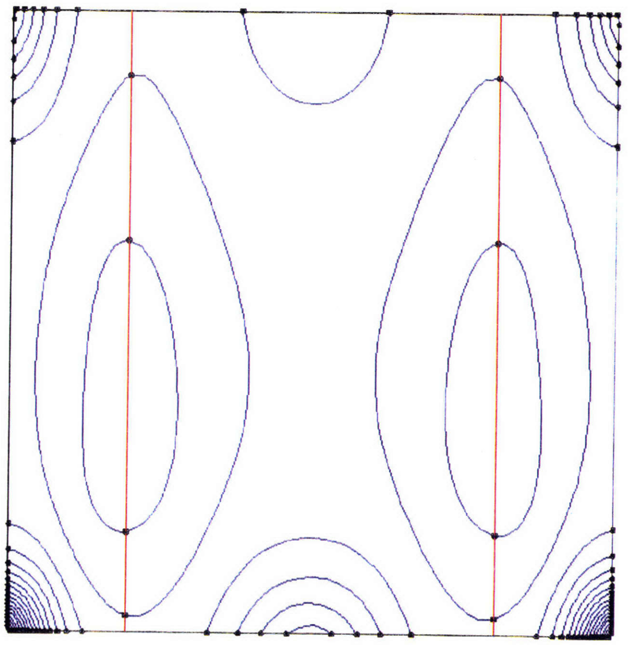


Figure 6-4: Sinusoidal Surface 7 x 7 Bézier Patches



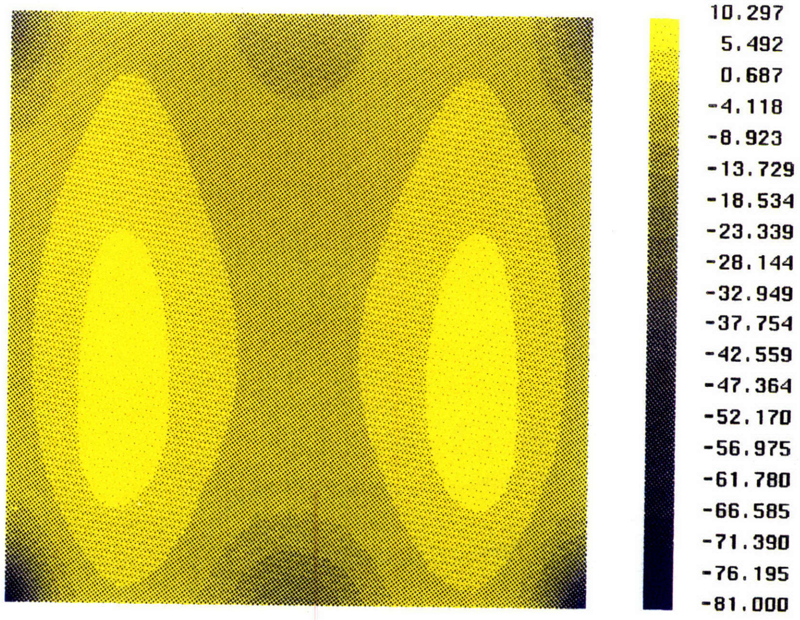
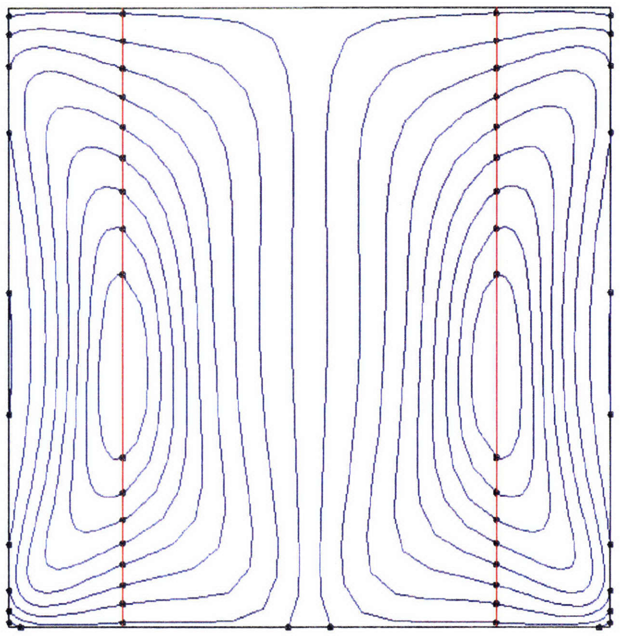


Figure 6-5: Sinusoidal Surface (1 Bézier Patch) - Gaussian Curvature



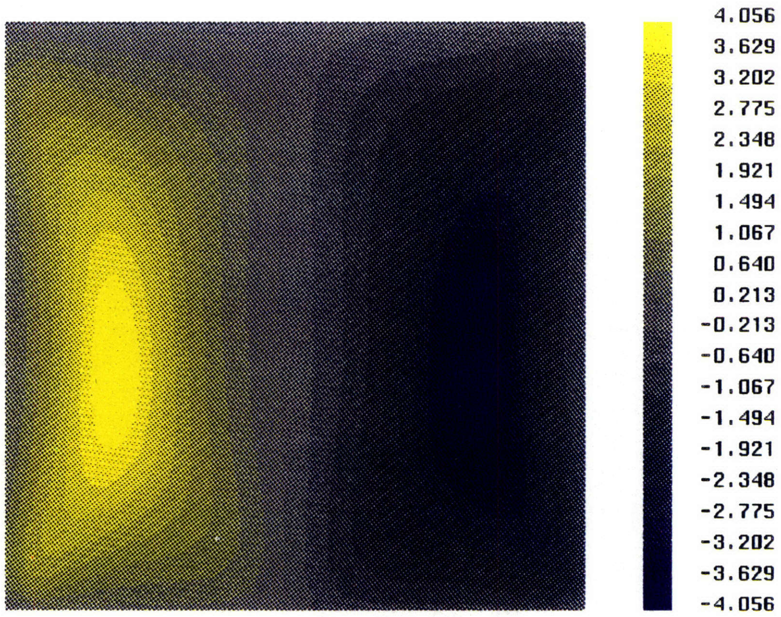
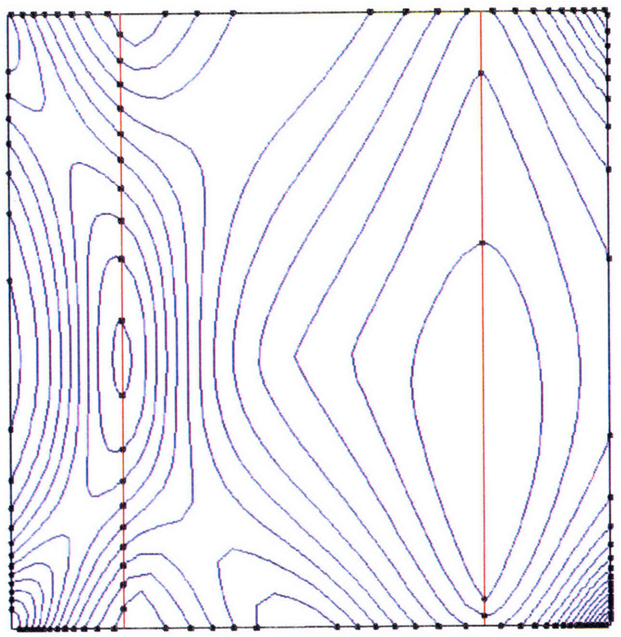


Figure 6-6: Sinusoidal Surface (1 Bézier Patch) - Mean Curvature



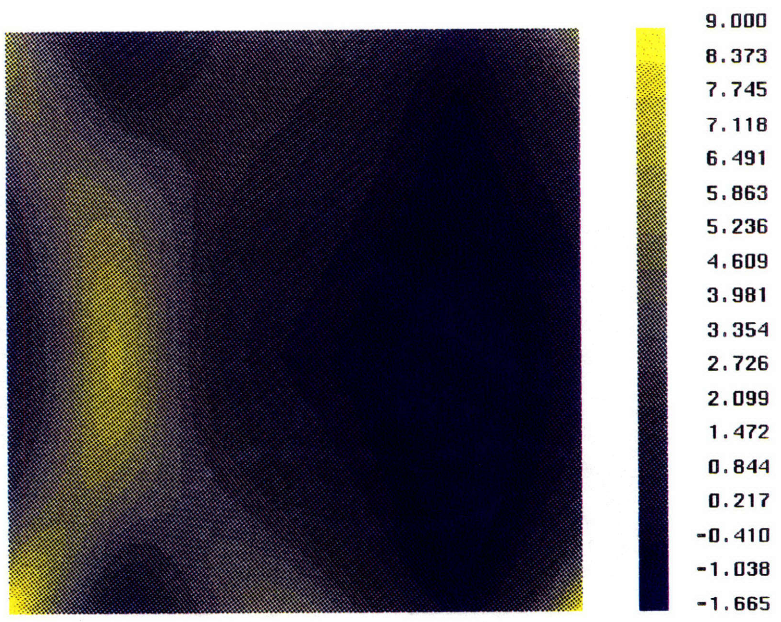
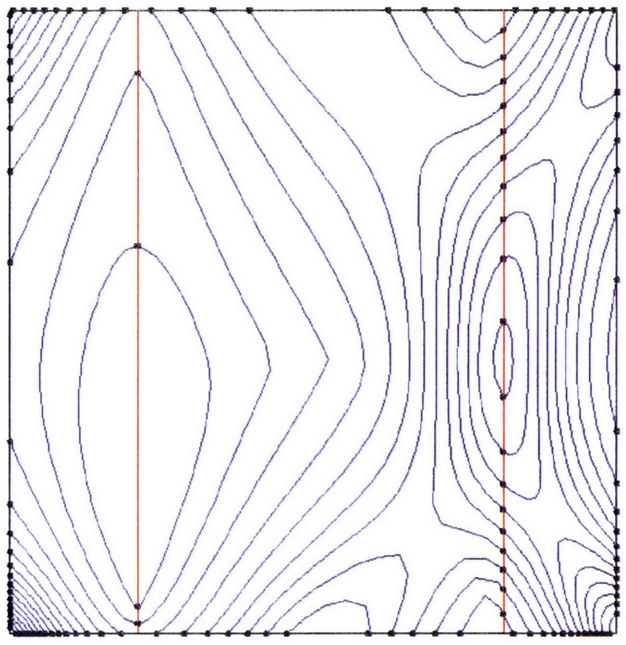


Figure 6-7: Sinusoidal Surface (1 Bézier Patch) - Maximum Principal Curvature



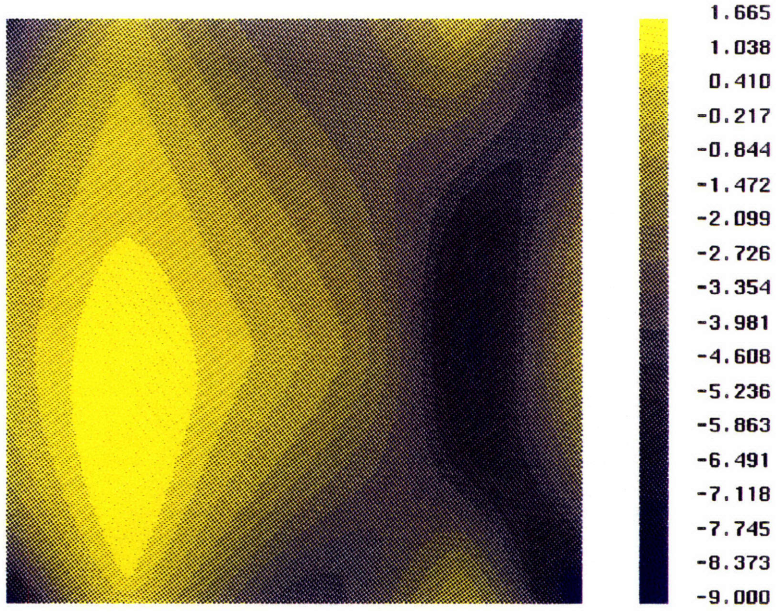
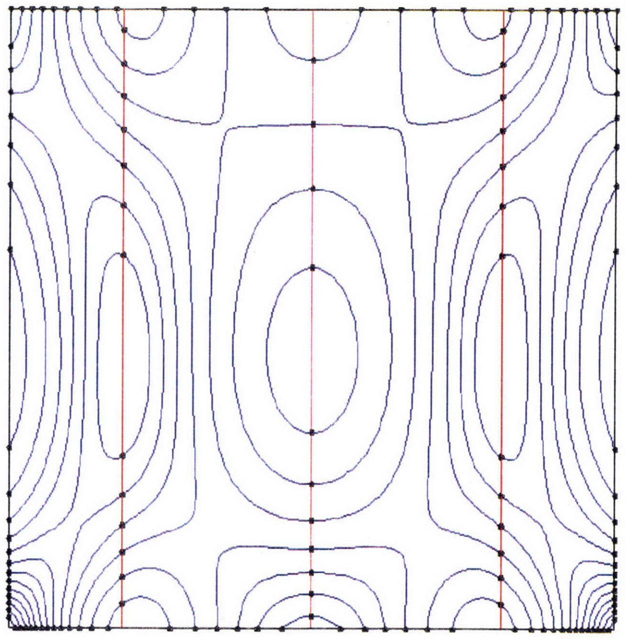


Figure 6-8: Sinusoidal Surface (1 Bézier Patch) - Minimum Principal Curvature



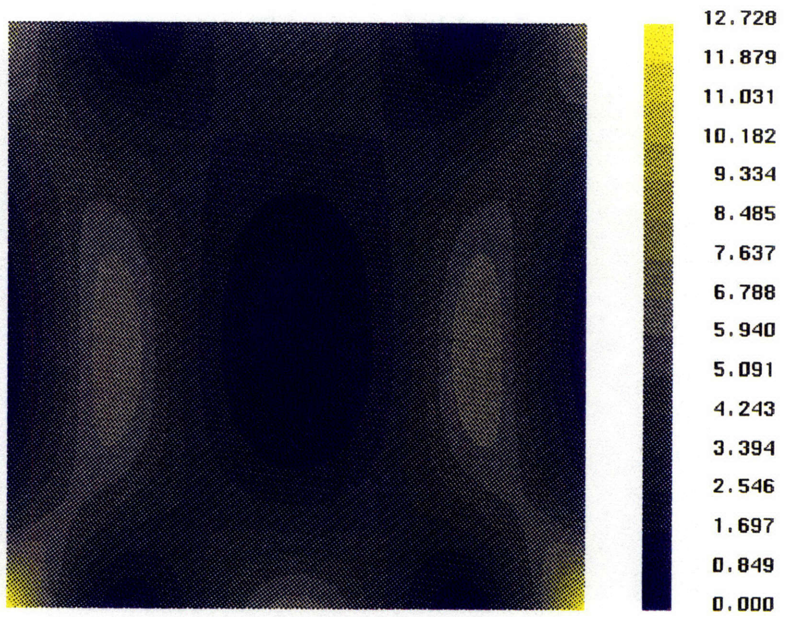
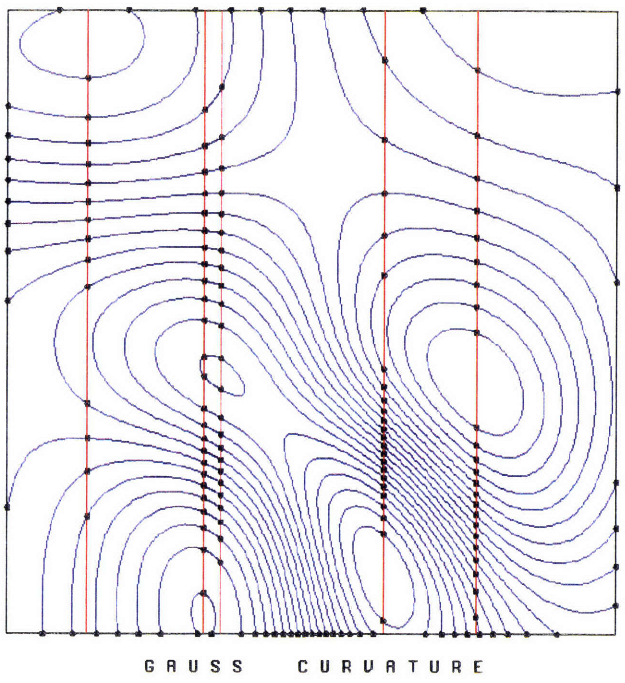


Figure 6-9: Sinusoidal Surface (1 Bézier Patch) - R M S Curvature



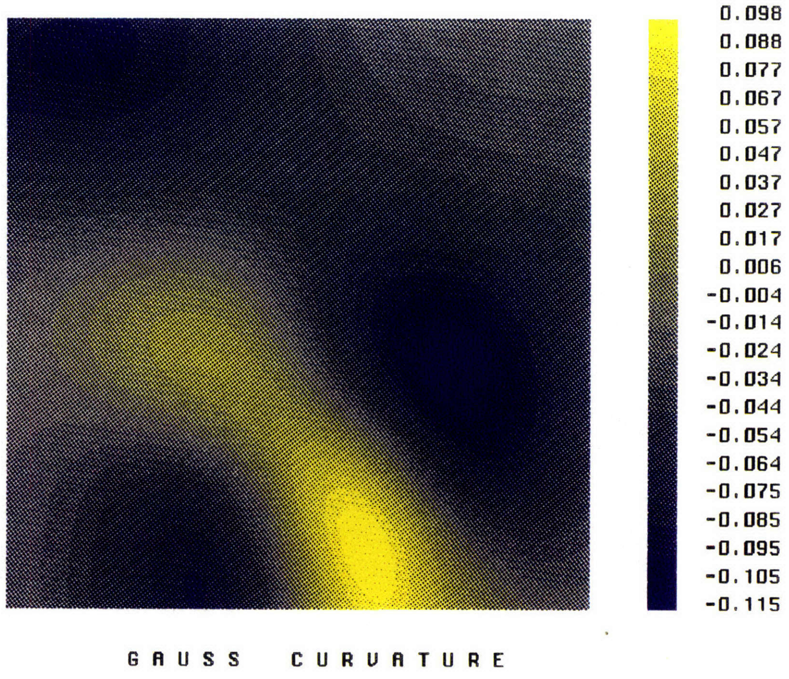
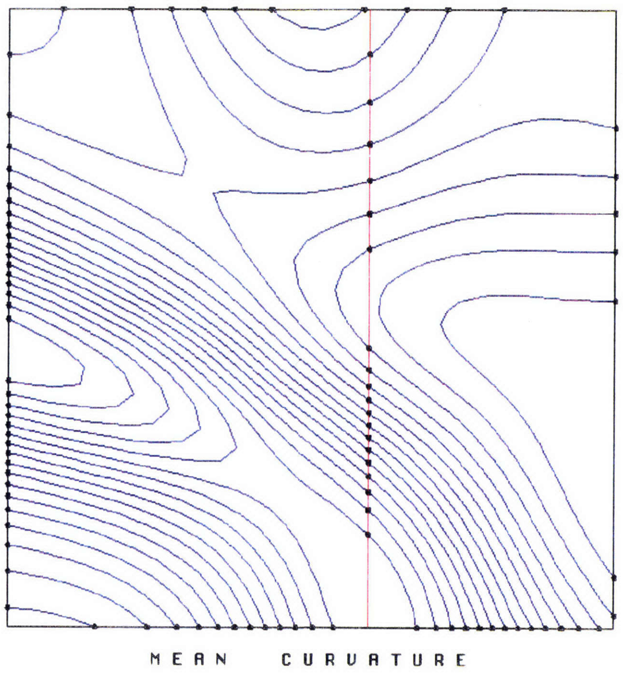


Figure 6-10: Data Surface (1 Bézier Patch) - Gaussian Curvature



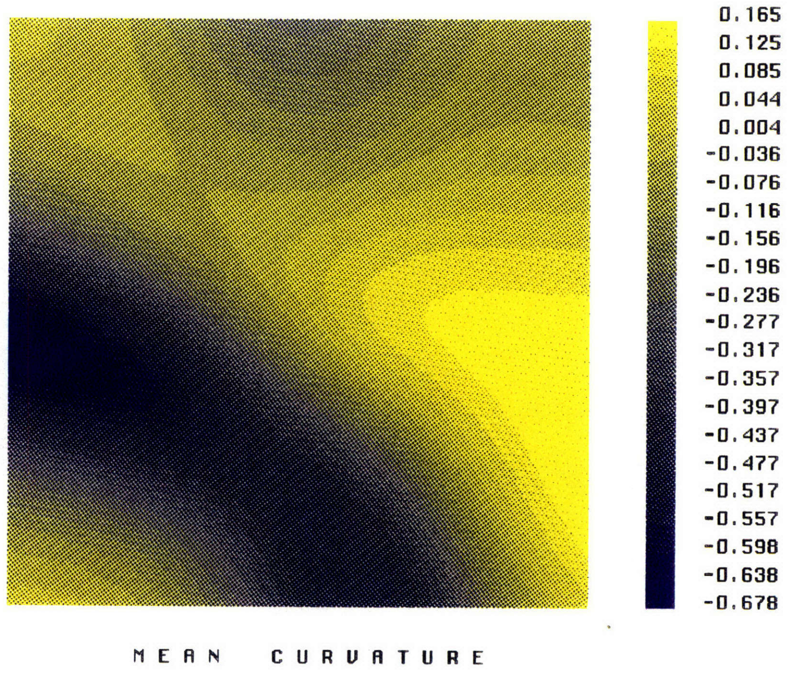
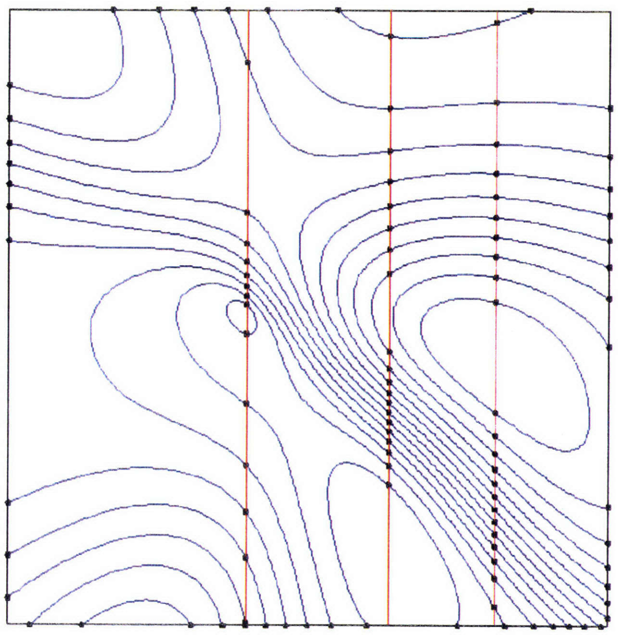


Figure 6-11: Data Surface (1 Bézier Patch) - Mean Curvature



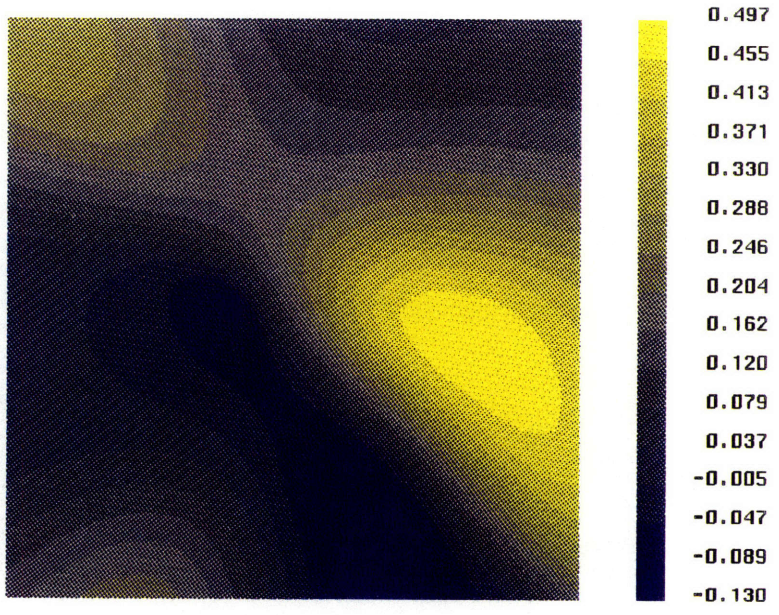
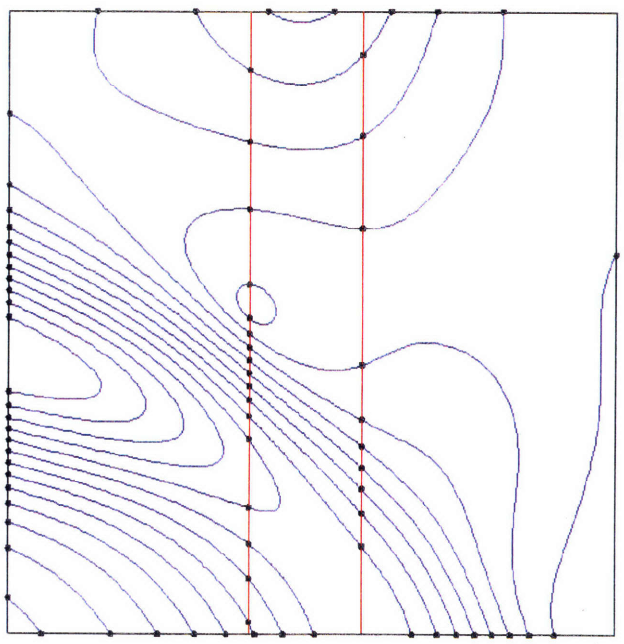


Figure 6-12: Data Surface (1 Bézier Patch) - Maximum Principal Curvature



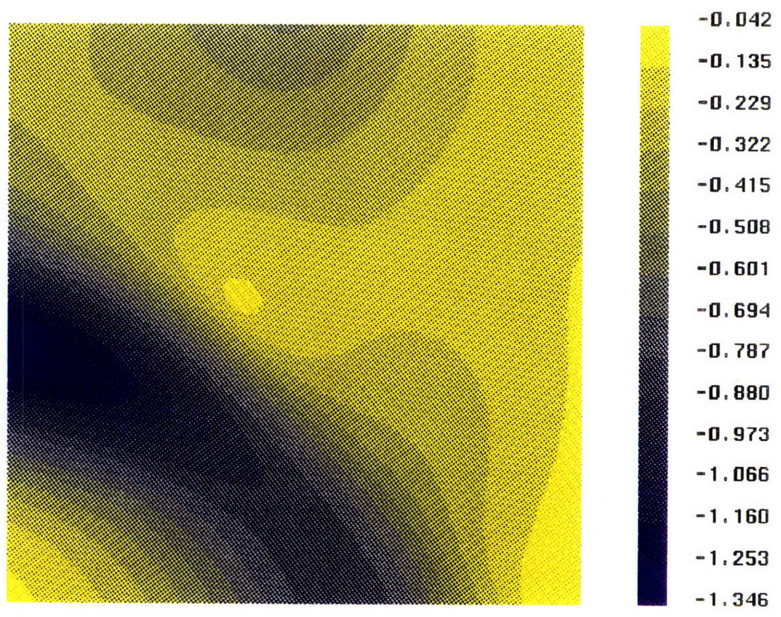


Figure 6-13: Data Surface (1 Bézier Patch) - Minimum Principal Curvature

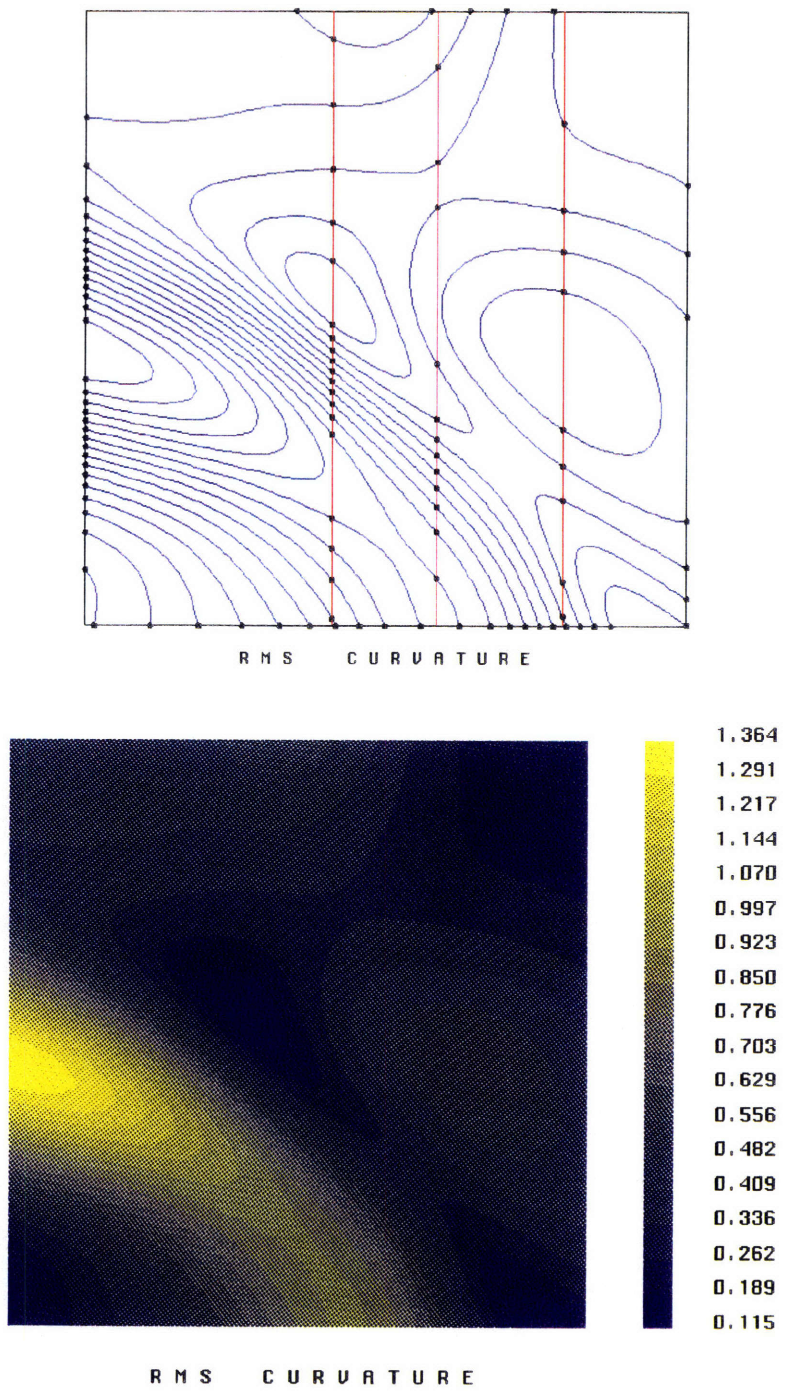
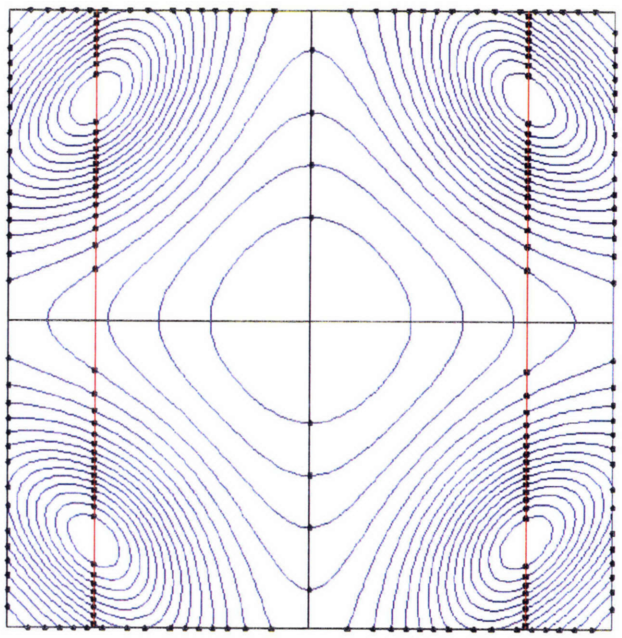


Figure 6-14: Data Surface (1 Bézier Patch) - R M S Curvature



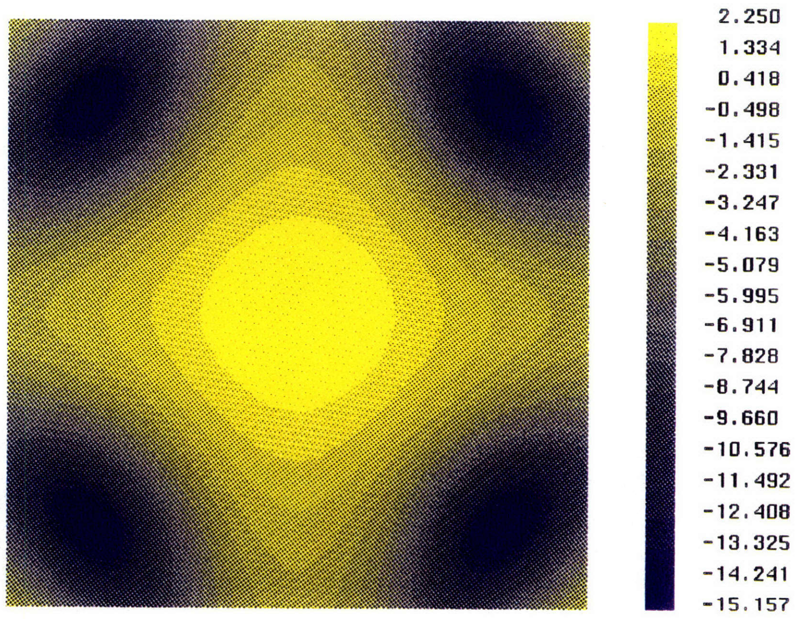
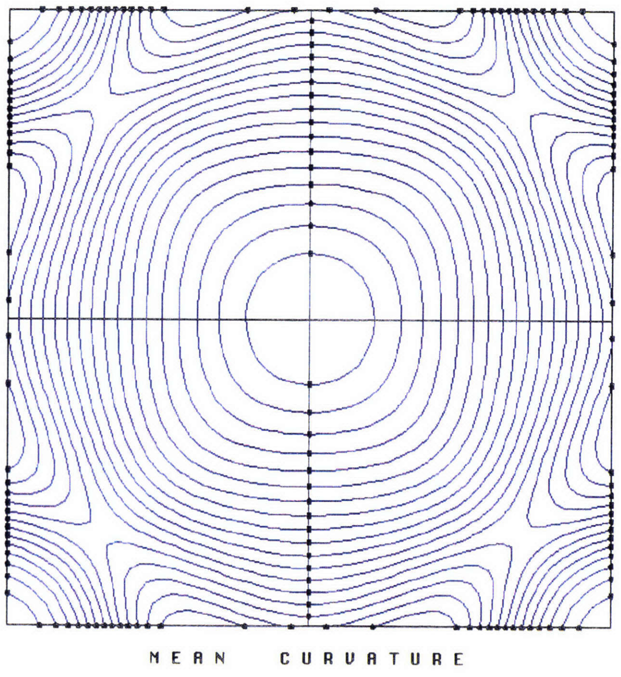


Figure 6-15: Hat-like Surface (2 x 2 Bézier Patch) - Gaussian Curvature



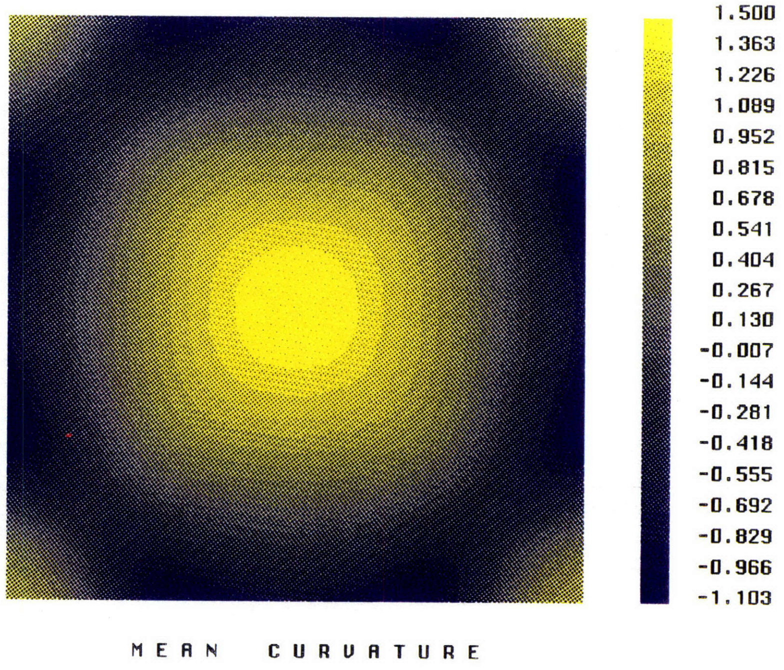
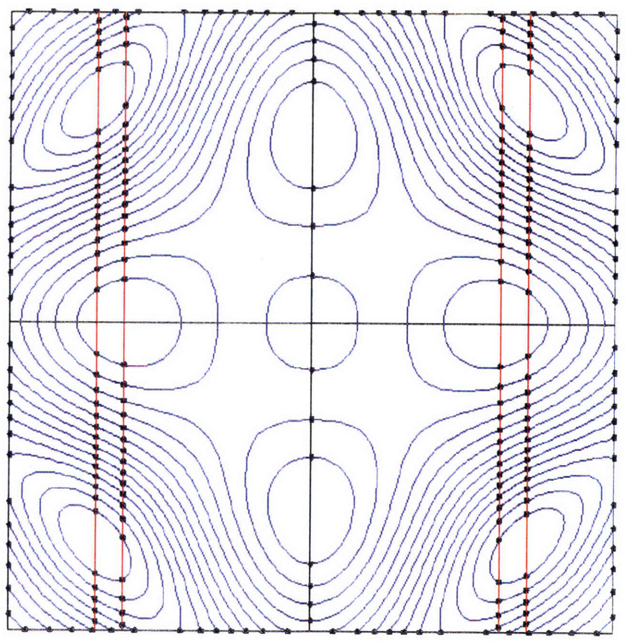


Figure 6-16: Hat-like Surface (2 x 2 Bézier Patch) - Mean Curvature



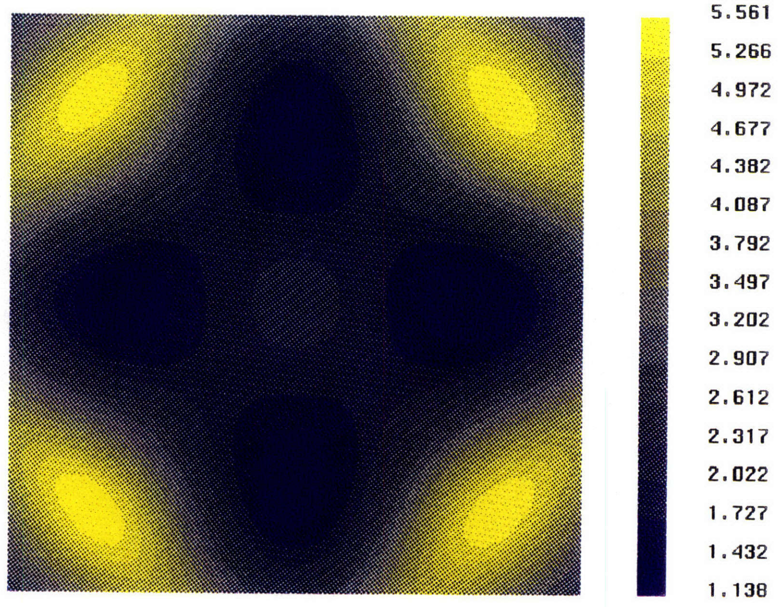
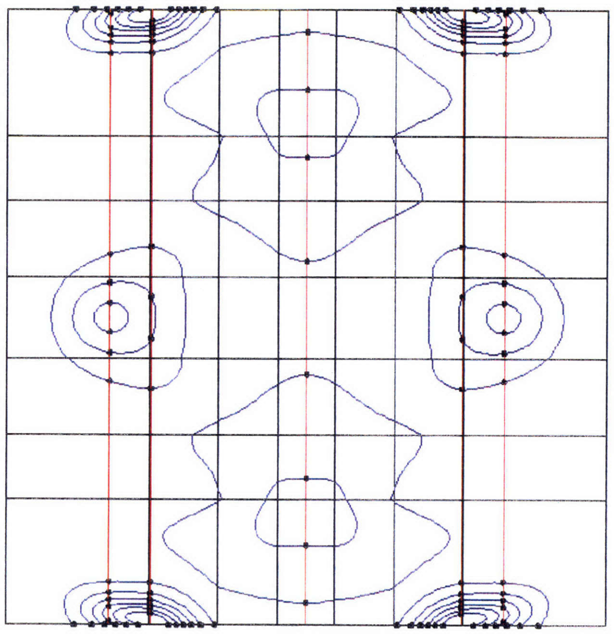


Figure 6-17: Hat-like Surface (2 x 2 Bézier Patch) - R M S Curvature



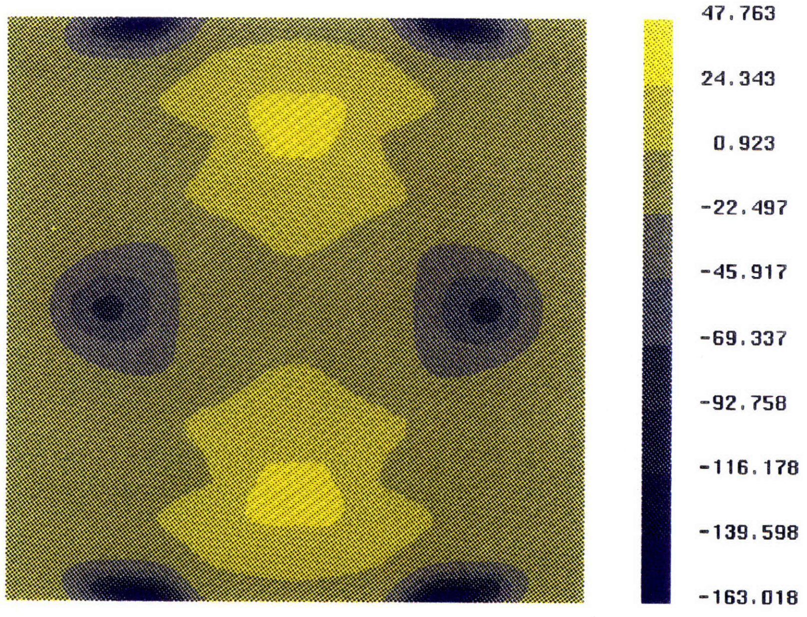
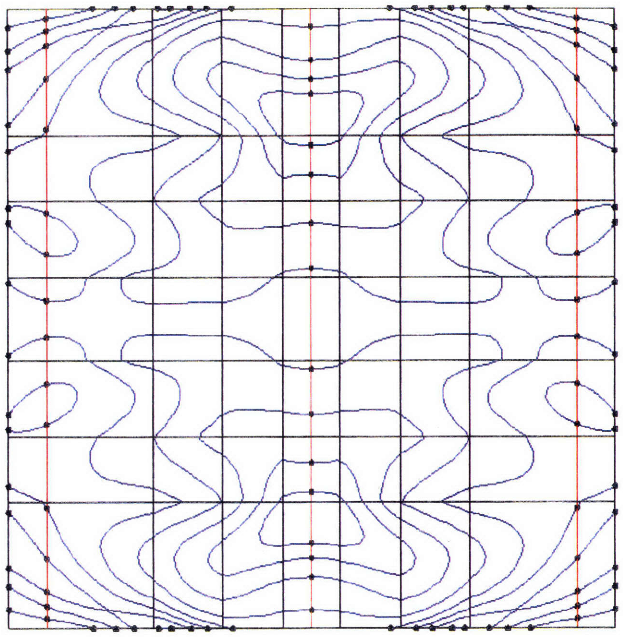


Figure 6-18: Sinusoidal Surface (7 x 7 Bézier Patch) - Gaussian Curvature



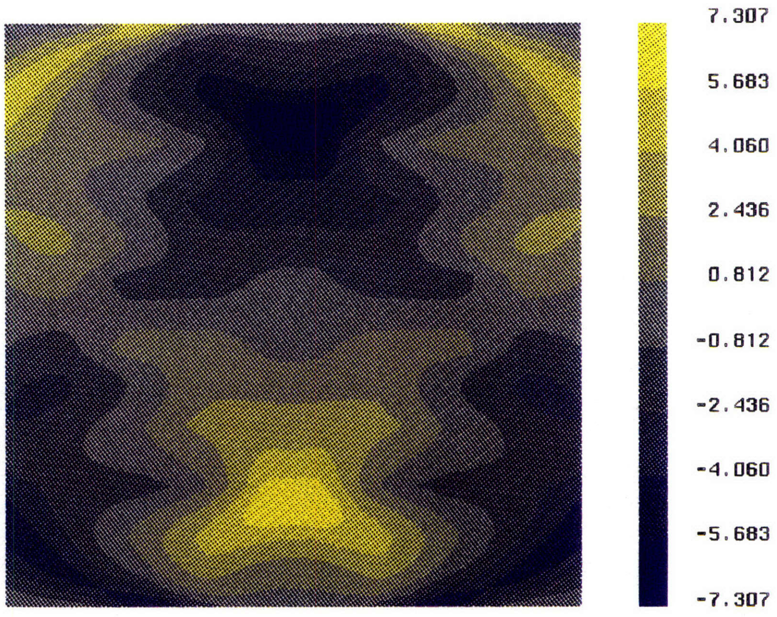
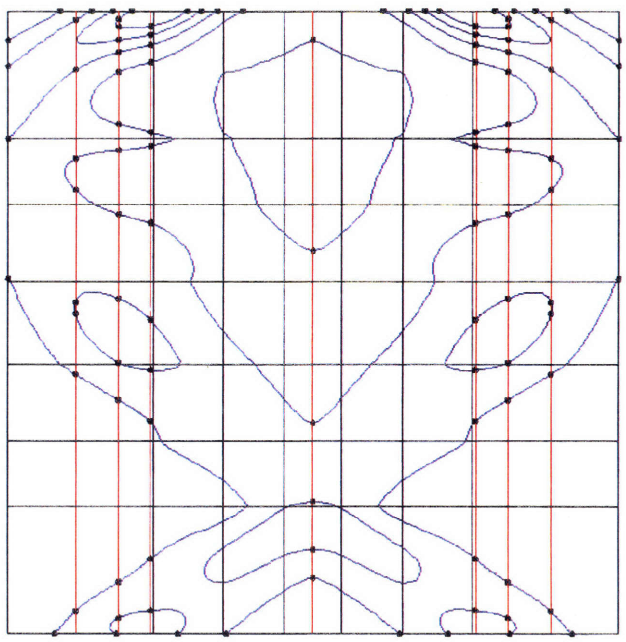


Figure 6-19: Sinusoidal Surface (7 x 7 Bézier Patch) - Mean Curvature



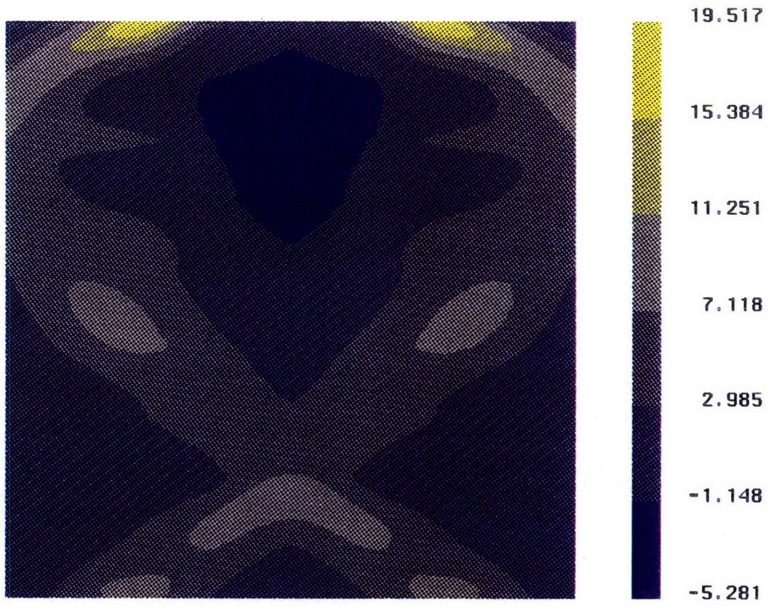
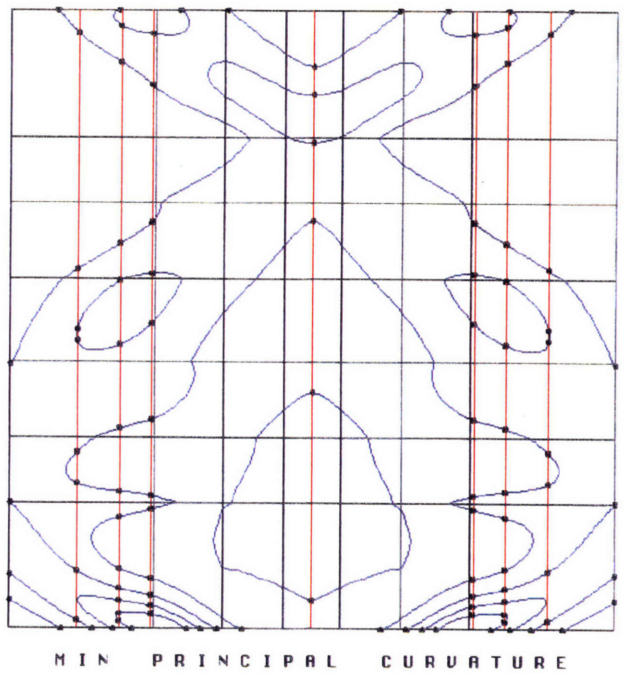


Figure 6-20: Sinusoidal Surface (7 x 7 Bézier Patch) - Max Principal Curvature



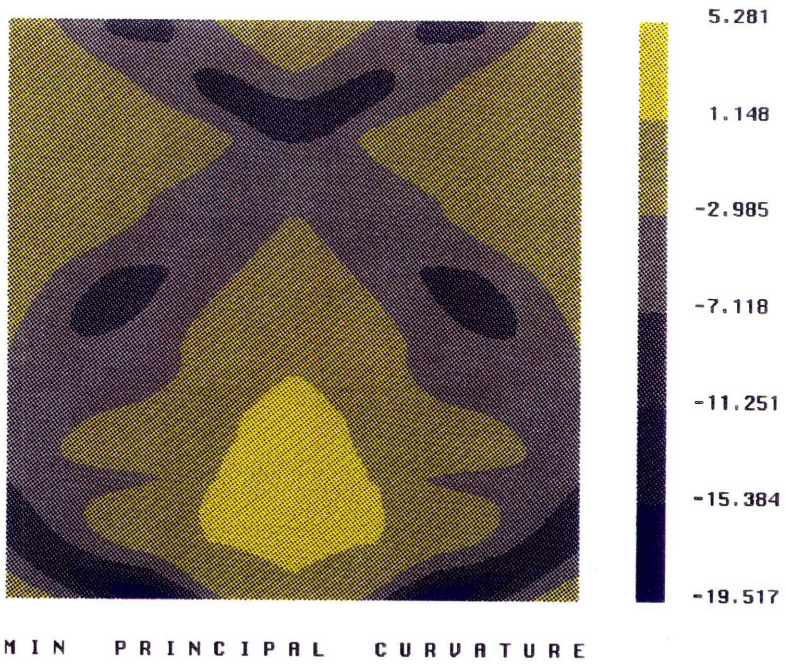
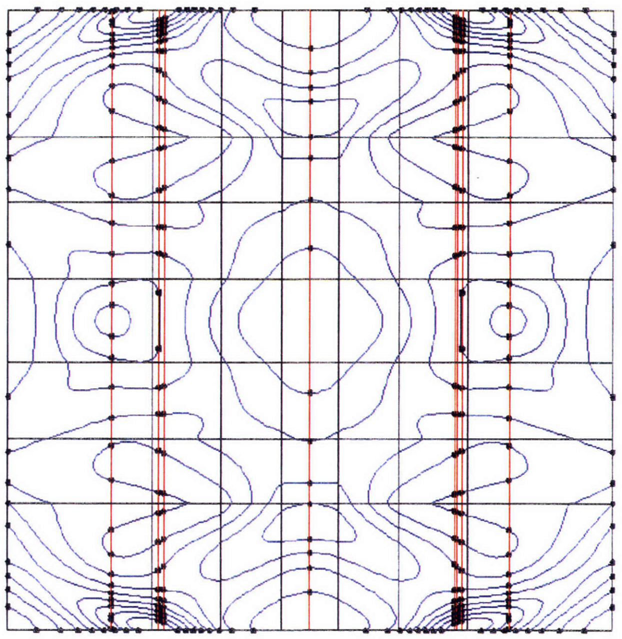


Figure 6-21: Sinusoidal Surface (7 x 7 Bézier Patch) - Min Principal Curvature



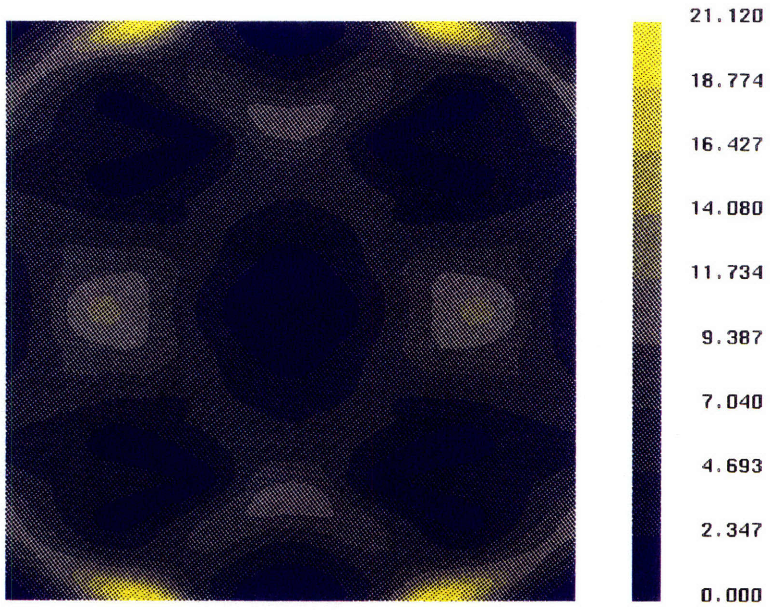


Figure 6-22: Sinusoidal Surface (7 x 7 Bézier Patch) - R M S Curvature

6.3 Parallel Processing and Performance Benchmarks

In order to develop the color coded curvature maps mentioned above, extensive computation is required. In this section, we examine the influence of parallel processing, on the computational time required for the process to take place.

All the runs of the program took place in the MIT Ocean Engineering Design Laboratory which is equipped with six Silicon Graphics workstations. Table (6.21) lists all the computers available in the Design Laboratory, along with their major characteristics.

It is customary to evaluate the performance of parallel processing by examining the computational time required as a function of the number of equivalent processors involved [4]. Nevertheless, because of the wide range of performances that the available *host* processors possess, it is not possible to do so. The reason for this is that while running a process in parallel on a very fast computer with a very slow machine, long idle time on the fast computer (while it is waiting for the slow one to finish) will appear. The situation worsens as the number of total *tasks* decreases. In order to overcome this problem, a number of possible computer configurations were created. Table (6.22) describes the different cases we tried. On the top of the list is what we would expect to be the slowest process (Fireb running by itself) and on the bottom of the list what we would expect to be the fastest (all the machines running in parallel). Note that Fireb and Famly are identical machines, therefore, we can compare computational time as a function of available processors. Note

Name	Type	CPU	RAM (MB)	Clock Speed (MHz)
fornix	Iris ONYX2-RE2	MIPS R4400 (2)	64	150
deslab	Iris 4DRPC50	MIPS R4000	32	100
fucus	Iris INDY-SC	MIPS R4000	16	100
fetus	Iris 4DRPC	MIPS R3000	80	33
fireb	Iris 4D35TG	MIPS R3000	16	36
famly	Iris 4D35TG	MIPS R3000	16	36

Table 6.21: Design Lab Computers

Number of Case	*Hosts*
1	Fireb
2	Fireb + Famly
3	Deslab
4	Fornix
5	Deslab + Fucus
6	Deslab + Fornix
7	Deslab + Fornix + Fucus
8	Deslab + Fornix + Fucus + Fetus
9	Deslab + Fornix + Fucus + Fetus + Famly
10	Deslab + Fornix + Fucus + Fetus + Famly + Fireb

Table 6.22: Possible Configurations for Parallel Processing Testing

also that although Fornix is the fastest available machine, the master process usually runs on Deslab. This is due to the fact that Fornix, the newest computer in the Laboratory, currently has some software incompatibility with the others and is unable to initiate the PVM Daemons on other machines. When this problem is overcome, final computational time will be even smaller.

Table (6.23) shows the total computational time it takes to complete the computations involving the Gaussian and mean curvatures of the 2x2 Hat-like surface and the 7x7 Sinusoidal surface on the different configuration cases presented in Table (6.22). The data are presented only for the multipatch surfaces mentioned above, since these are the ones parallel processing is intended for. Note that for the Hat-like surface, cases 9 and 10 are inapplicable since the number of Bézier patches in that surface are four and this is the maximum number of *hosts* that could be used.

These results are also presented graphically in Figs. (6-23) to (6-26). We can make several observations concerning the efficiency of parallel programming, from these figures and table (6.23).

Figures (6-23) and (6-24) present computational time for the Hat-like surface which is

Number of Case	Total Computational Time (sec)			
	2x2 Hat-like Surface		7x7 Sinusoidal Surface	
	Gaussian	Mean	Gaussian	Mean
1	440	671	847	1321
2	315	425	562	979
3	244	362	505	839
4	120	169	298	460
5	208	221	299	550
6	153	194	234	360
7	159	152	181	301
8	146	219	173	286
9	n/a	n/a	180	281
10	n/a	n/a	179	250
Ratio	3.01	4.41	4.90	5.29

Table 6.23: Total Computational Time for Gaussian and Mean Curvatures

decomposable into four Bézier patches, and thus 4 PVM *processes*. The low number of processes makes the efficiency of parallel programming moderately low. We can definitely see a decreasing trend as we go from case 1 to case 8. There is one major observations to be made. Because of the size of the problem, case 4 (Fornix running by itself) can achieve equal or better results than any other combination. Indeed, the addition of other machines effectively delays the program. This effect could have been reduced if the master process for cases 5 and above, was running on Fornix. Nevertheless, the parallel programming efficiency is apparent when examining cases 1 and 2 (involving identical computers).

Figures (6-25) and (6-26) present computational time for the sinusoidal surface, which is decomposable into 49 Bézier patches. It is evident that the larger the size of the problem, the more efficient parallel processing is. Case 4 behaves more evenly in the case of the 49 patch problem. The last line of Table (6.23) shows the ratio of the largest to the smallest computational time for each problem. We can see that the ratio increases from the simplest (Gaussian curvature) and smallest (2x2 case) to the more complex (mean curvature) and larger (7x7 case) problem.

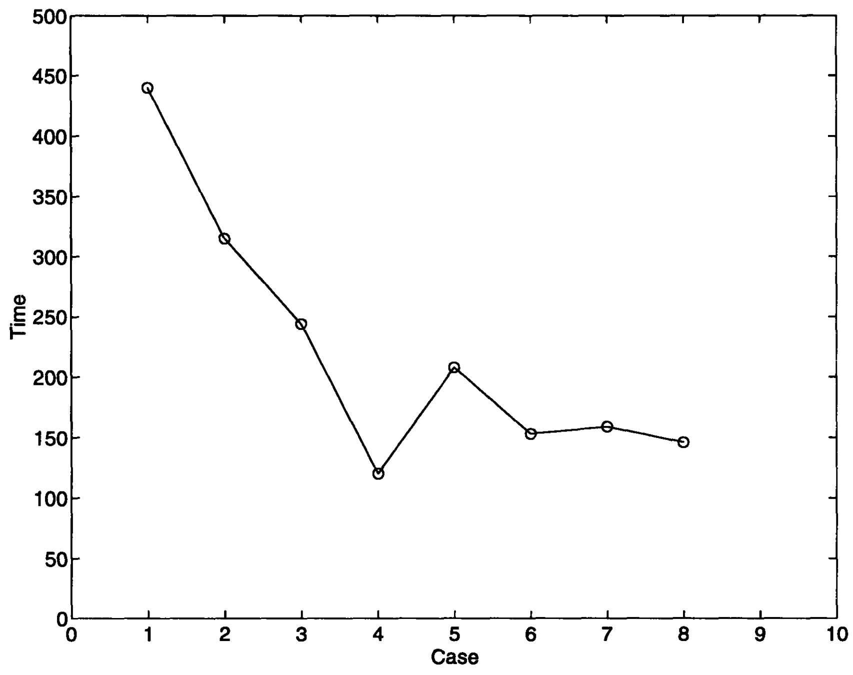


Figure 6-23: Parallel Programming - 2x2 Hat-like Surface, Gaussian Curvature

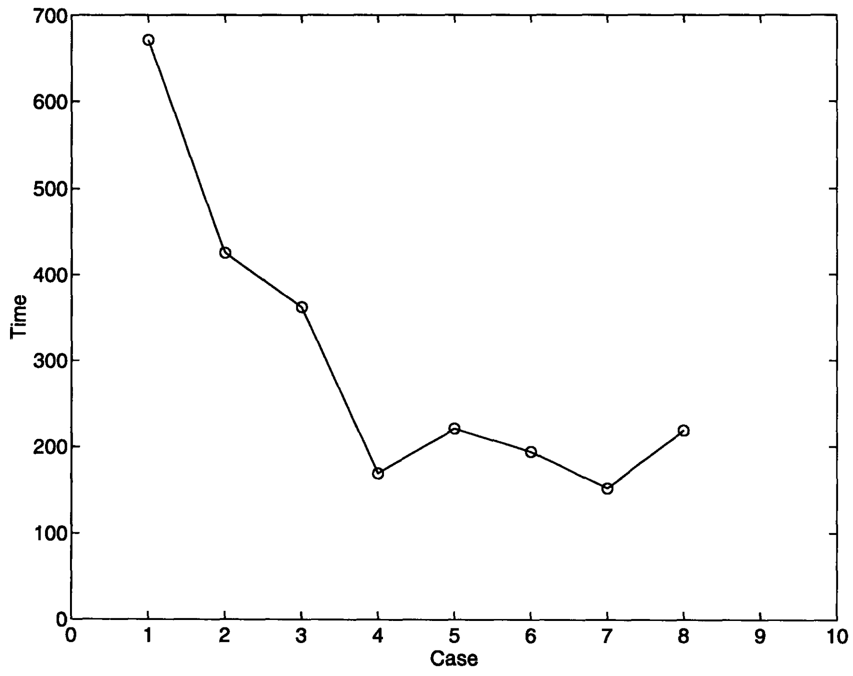


Figure 6-24: Parallel Programming - 2x2 Hat-like Surface, Mean Curvature

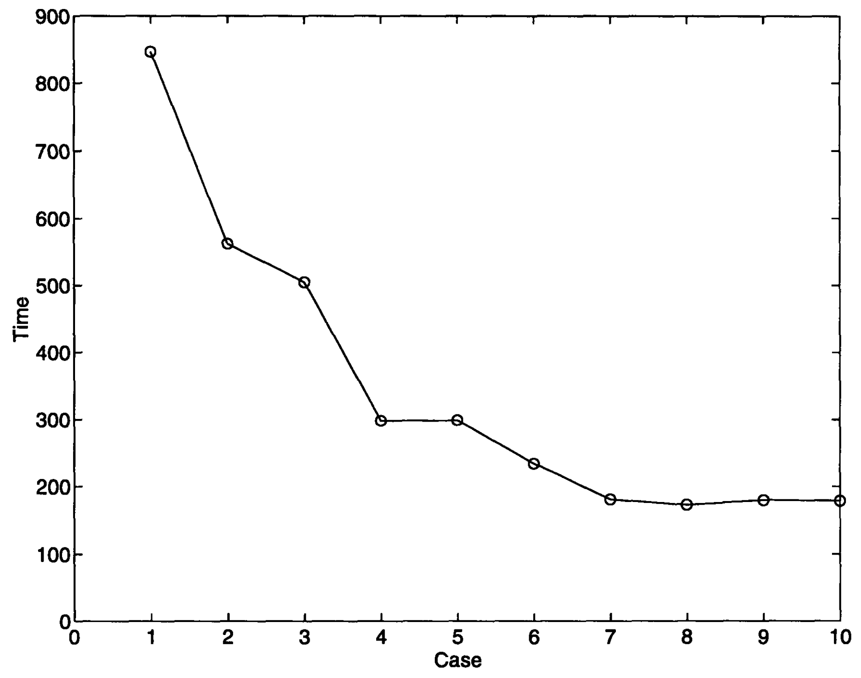


Figure 6-25: Parallel Programming - 7x7 Sinusoidal Surface, Gaussian Curvature

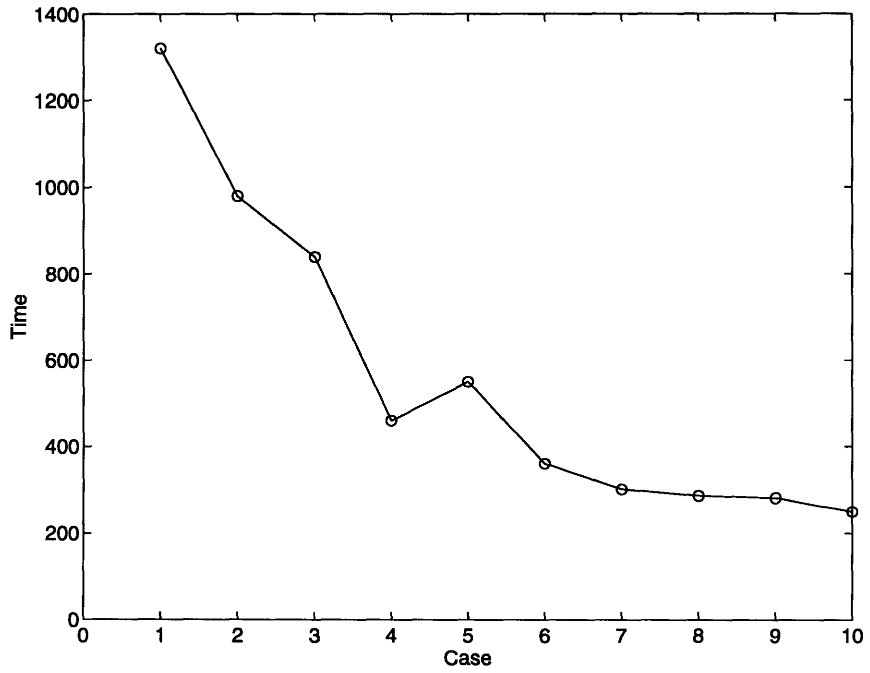


Figure 6-26: Parallel Programming - 7x7 Sinusoidal Surface, Mean Curvature

6.4 Conclusions and Recommendations

In this section we review the contribution of this thesis, and conclude and recommend topics for further investigation.

We have developed a robust tool to accurately interrogate complex non-uniform (integral) B-spline surfaces. Given any such B-spline surface, the methodology developed allows for the development of constant curvature contour maps for the Gaussian, mean, maximum principal, minimum principal and root mean square curvature. Important features of this methodology include:

- The use of both floating point arithmetic and rounded interval arithmetic for the formulation of the governing equations.
- The implementation of rounded interval arithmetic to the nonlinear polynomial equation solver, based on the Bernstein subdivision method.
- The subdivision of the B-spline surface into several Bézier patches. This allowed for more computational efficiency, since we were able to readily formulate and solve nonlinear polynomial systems of equations. It also allows the subdivision of a large problem into several smaller subproblems.
- The use of parallel processing to distribute the small subproblems to different processors, leading to great computational time efficiency.
- The implementation of the whole methodology in C++ allowing for easy modification and addition of features.

Major observations that can be made from using this methodology include:

- Floating point arithmetic is adequate for only about 95% of the cases examined. It could lead to wrong results if the nonlinear solver misses a root of the system of polynomials that govern the interrogation problem.

- Interval arithmetic guarantees a correct solution.
- The implementation using interval arithmetic is substantially slower than the corresponding using floating point arithmetic, by a factor that can reach the value of 40.
- Parallel processing substantially reduces the total computational time, when all processors involved have similar speed. Around 98% of the computation is performed in parallel.
- Large variations of speeds of the processors used might have a negative influence on the parallel processing speed-up.
- The existence of square roots in the governing equations and thus the implementation of the auxiliary variable method, increase the computational time. Curvature maps for Gaussian, mean and root mean square curvatures are computed faster than the corresponding ones for principal curvatures.

Considering the above, recommendations for future investigations are:

- Interval arithmetic should be used on complex surfaces with no indication of what the curvature range might be.
- When using floating point arithmetic, an approximate method of estimating the possible range of the curvature, before the actual computation is performed, is needed so as to check possible misses of the solver. This method may be the evaluation of the particular curvature on points of a dense grid, and the extraction of an approximate maximum and minimum.
- Global methods for computing the roots of nonlinear polynomial equations do not need initial approximations and when coupled with rounded interval arithmetic are robust, but slow. A combination of this method, with low required accuracy, that

will efficiently yield a good first approximation of all roots, with an interval Newton method for root refining in order to accelerate convergence, should be investigated.

- The implementation of a truly variable step integration method for contour tracing is needed, so that the program automatically varies the integration step according to the location of the starting points and the particular value of the curvature at any point.

Appendix A

Formulas for Curvature Partial Derivatives

The following is adapted from Maekawa, [18].

Surface normal

$$\mathbf{S} = \mathbf{r}_u \times \mathbf{r}_v \tag{A.1}$$

$$\mathbf{S}_u = \mathbf{r}_{uu} \times \mathbf{r}_v \ + \ \mathbf{r}_u \times \mathbf{r}_{uv}, \quad \mathbf{S}_v = \mathbf{r}_{uv} \times \mathbf{r}_v \ + \ \mathbf{r}_u \times \mathbf{r}_{vv} \tag{A.2}$$

$$\mathbf{S}_{uu} = \mathbf{r}_{uuu} \times \mathbf{r}_v \ + \ 2\mathbf{r}_{uu} \times \mathbf{r}_{uv} \ + \ \mathbf{r}_u \times \mathbf{r}_{uuv} \tag{A.3}$$

$$\mathbf{S}_{uv} = \mathbf{r}_{uuv} \times \mathbf{r}_v \ + \ \mathbf{r}_{uu} \times \mathbf{r}_{vv} \ + \ \mathbf{r}_u \times \mathbf{r}_{uvv} \tag{A.4}$$

$$\mathbf{S}_{vv} = \mathbf{r}_{uvv} \times \mathbf{r}_v \ + \ 2\mathbf{r}_{uv} \times \mathbf{r}_{vv} \ + \ \mathbf{r}_u \times \mathbf{r}_{vvv} \tag{A.5}$$

Scalar magnitude of surface normal

$$S = |\mathbf{r}_u \times \mathbf{r}_v| = \sqrt{det[\Gamma]} = \sqrt{EG - F^2} \tag{A.6}$$

$$S_u = \frac{\mathbf{S} \cdot \mathbf{S}_u}{S}, \quad S_v = \frac{\mathbf{S} \cdot \mathbf{S}_v}{S} \tag{A.7}$$

$$S_{uu} = \frac{\mathbf{S} \cdot \mathbf{S}_{uu} + \mathbf{S}_u \cdot \mathbf{S}_u - S_u^2}{S} \tag{A.8}$$

$$S_{uv} = \frac{\mathbf{S} \cdot \mathbf{S}_{uv} + \mathbf{S}_u \cdot \mathbf{S}_v - S_u S_v}{S} \tag{A.9}$$

$$S_{vv} = \frac{\mathbf{S} \cdot \mathbf{S}_{vv} + \mathbf{S}_v \cdot \mathbf{S}_v - S_v^2}{S} \tag{A.10}$$

The coefficients of first fundamental form

$$E = \mathbf{r}_u \cdot \mathbf{r}_u \tag{A.11}$$

$$E_u = 2\mathbf{r}_u \cdot \mathbf{r}_{uu}, \quad E_v = 2\mathbf{r}_u \cdot \mathbf{r}_{uv} \tag{A.12}$$

$$E_{uu} = 2(\mathbf{r}_{uu} \cdot \mathbf{r}_{uu} + \mathbf{r}_u \cdot \mathbf{r}_{uuu}) \tag{A.13}$$

$$E_{uv} = 2(\mathbf{r}_{uv} \cdot \mathbf{r}_{uu} + \mathbf{r}_u \cdot \mathbf{r}_{uuv}) \tag{A.14}$$

$$E_{vv} = 2(\mathbf{r}_{uv} \cdot \mathbf{r}_{uv} + \mathbf{r}_u \cdot \mathbf{r}_{uvv}) \tag{A.15}$$

$$F = \mathbf{r}_u \cdot \mathbf{r}_v \tag{A.16}$$

$$F_u = \mathbf{r}_{uu} \cdot \mathbf{r}_v + \mathbf{r}_u \cdot \mathbf{r}_{uv}, \quad F_v = \mathbf{r}_{uv} \cdot \mathbf{r}_v + \mathbf{r}_u \cdot \mathbf{r}_{vv} \tag{A.17}$$

$$F_{uu} = \mathbf{r}_{uuu} \cdot \mathbf{r}_v + 2\mathbf{r}_{uu} \cdot \mathbf{r}_{uv} + \mathbf{r}_u \cdot \mathbf{r}_{uuv} \tag{A.18}$$

$$F_{uv} = \mathbf{r}_{uuv} \cdot \mathbf{r}_v + \mathbf{r}_{uu} \cdot \mathbf{r}_{vv} + \mathbf{r}_{uv} \cdot \mathbf{r}_{uv} + \mathbf{r}_u \cdot \mathbf{r}_{uvv} \tag{A.19}$$

$$F_{vv} = \mathbf{r}_{uvv} \cdot \mathbf{r}_v + 2\mathbf{r}_{uv} \cdot \mathbf{r}_{vv} + \mathbf{r}_u \cdot \mathbf{r}_{vvv} \tag{A.20}$$

$$G = \mathbf{r}_v \cdot \mathbf{r}_v \tag{A.21}$$

$$G_u = 2\mathbf{r}_v \cdot \mathbf{r}_{uv}, \quad G_v = 2\mathbf{r}_v \cdot \mathbf{r}_{vv} \tag{A.22}$$

$$G_{uu} = 2(\mathbf{r}_{uv} \cdot \mathbf{r}_{uv} + \mathbf{r}_v \cdot \mathbf{r}_{uuv}) \tag{A.23}$$

$$G_{uv} = 2(\mathbf{r}_{vv} \cdot \mathbf{r}_{uv} + \mathbf{r}_v \cdot \mathbf{r}_{uvv}) \tag{A.24}$$

$$G_{vv} = 2(\mathbf{r}_{vv} \cdot \mathbf{r}_{vv} + \mathbf{r}_v \cdot \mathbf{r}_{vvv}) \tag{A.25}$$

The coefficients of second fundamental form multiplied by S

$$\tilde{L} = SL = \mathbf{S} \cdot \mathbf{r}_{uu} \tag{A.26}$$

$$\tilde{L}_u = \mathbf{S}_u \cdot \mathbf{r}_{uu} + \mathbf{S} \cdot \mathbf{r}_{uuu}, \quad \tilde{L}_v = \mathbf{S}_v \cdot \mathbf{r}_{uu} + \mathbf{S} \cdot \mathbf{r}_{uuv} \tag{A.27}$$

$$\tilde{L}_{uu} = \mathbf{S}_{uu} \cdot \mathbf{r}_{uu} + 2\mathbf{S}_u \cdot \mathbf{r}_{uuu} + \mathbf{S} \cdot \mathbf{r}_{uuuu} \tag{A.28}$$

$$\tilde{L}_{uv} = \mathbf{S}_{uv} \cdot \mathbf{r}_{uu} + \mathbf{S}_u \cdot \mathbf{r}_{uuv} + \mathbf{S}_v \cdot \mathbf{r}_{uuu} + \mathbf{S} \cdot \mathbf{r}_{uuuv} \tag{A.29}$$

$$\tilde{L}_{vv} = \mathbf{S}_{vv} \cdot \mathbf{r}_{uu} + 2\mathbf{S}_v \cdot \mathbf{r}_{uuv} + \mathbf{S} \cdot \mathbf{r}_{uuvv} \tag{A.30}$$

$$\tilde{M} = SM = \mathbf{S} \cdot \mathbf{r}_{uv} \tag{A.31}$$

$$\tilde{M}_u = \mathbf{S}_u \cdot \mathbf{r}_{uv} + \mathbf{S} \cdot \mathbf{r}_{uuv}, \quad \tilde{M}_v = \mathbf{S}_v \cdot \mathbf{r}_{uv} + \mathbf{S} \cdot \mathbf{r}_{uvv} \tag{A.32}$$

$$\tilde{M}_{uu} = \mathbf{S}_{uu} \cdot \mathbf{r}_{uv} + 2\mathbf{S}_u \cdot \mathbf{r}_{uuv} + \mathbf{S} \cdot \mathbf{r}_{uuuv} \tag{A.33}$$

$$\tilde{M}_{uv} = \mathbf{S}_{uv} \cdot \mathbf{r}_{uv} + \mathbf{S}_u \cdot \mathbf{r}_{uvv} + \mathbf{S}_v \cdot \mathbf{r}_{uuv} + \mathbf{S} \cdot \mathbf{r}_{uuvv} \tag{A.34}$$

$$\tilde{M}_{vv} = \mathbf{S}_{vv} \cdot \mathbf{r}_{uv} + 2\mathbf{S}_v \cdot \mathbf{r}_{uvv} + \mathbf{S} \cdot \mathbf{r}_{uvvv} \tag{A.35}$$

$$\tilde{N} = SN = \mathbf{S} \cdot \mathbf{r}_{vv} \tag{A.36}$$

$$\tilde{N}_u = \mathbf{S}_u \cdot \mathbf{r}_{vv} + \mathbf{S} \cdot \mathbf{r}_{uvv}, \quad \tilde{N}_v = \mathbf{S}_v \cdot \mathbf{r}_{vv} + \mathbf{S} \cdot \mathbf{r}_{vvv} \tag{A.37}$$

$$\tilde{N}_{uu} = \mathbf{S}_{uu} \cdot \mathbf{r}_{vv} + 2\mathbf{S}_u \cdot \mathbf{r}_{uvv} + \mathbf{S} \cdot \mathbf{r}_{uuvv} \tag{A.38}$$

$$\tilde{N}_{uv} = \mathbf{S}_{uv} \cdot \mathbf{r}_{vv} + \mathbf{S}_u \cdot \mathbf{r}_{vvv} + \mathbf{S}_v \cdot \mathbf{r}_{uvv} + \mathbf{S} \cdot \mathbf{r}_{uvvv} \tag{A.39}$$

$$\tilde{N}_{vv} = \mathbf{S}_{vv} \cdot \mathbf{r}_{vv} + 2\mathbf{S}_v \cdot \mathbf{r}_{vvv} + \mathbf{S} \cdot \mathbf{r}_{vvvv} \tag{A.40}$$

The determinant of second fundamental matrix multiplied by S^2

$$A = \tilde{L}\tilde{N} - \tilde{M}^2 \tag{A.41}$$

$$A_u = \tilde{L}_u\tilde{N} + \tilde{L}\tilde{N}_u - 2\tilde{M}\tilde{M}_u, \quad A_v = \tilde{L}_v\tilde{N} + \tilde{L}\tilde{N}_v - 2\tilde{M}\tilde{M}_v \tag{A.42}$$

$$A_{uu} = \tilde{L}_{uu}\tilde{N} + \tilde{L}\tilde{N}_{uu} + 2(\tilde{L}_u\tilde{N}_u - \tilde{M}_u^2 - \tilde{M}\tilde{M}_{uu}) \tag{A.43}$$

$$A_{uv} = \tilde{L}_{uv}\tilde{N} + \tilde{L}_u\tilde{N}_v + \tilde{L}_v\tilde{N}_u + \tilde{L}\tilde{N}_{uv} - 2(\tilde{M}_v\tilde{M}_u + \tilde{M}\tilde{M}_{uv}) \tag{A.44}$$

$$A_{vv} = \tilde{L}_{vv}\tilde{N} + \tilde{L}\tilde{N}_{vv} + 2(\tilde{L}_v\tilde{N}_v - \tilde{M}_v^2 - \tilde{M}\tilde{M}_{vv}) \tag{A.45}$$

The numerator of the mean curvature, equation (3.20) multiplied by S

$$B = 2F\tilde{M} - E\tilde{N} - G\tilde{L} \tag{A.46}$$

$$B_u = 2(F\tilde{M}_u + F_u\tilde{M}) - (E_u\tilde{N} + E\tilde{N}_u) - (G_u\tilde{L} + G\tilde{L}_u) \tag{A.47}$$

$$B_v = 2(F\tilde{M}_v + F_v\tilde{M}) - (E_v\tilde{N} + E\tilde{N}_v) - (G_v\tilde{L} + G\tilde{L}_v) \tag{A.48}$$

$$\begin{aligned} B_{uu} &= 2(F\tilde{M}_{uu} + 2F_u\tilde{M}_u + F_{uu}\tilde{M}) - (E_{uu}\tilde{N} + 2E_u\tilde{N}_u + E\tilde{N}_{uu}) \\ &\quad - (G_{uu}\tilde{L} + 2G_u\tilde{L}_u + G\tilde{L}_{uu}) \end{aligned} \tag{A.49}$$

$$\begin{aligned} B_{uv} &= 2(F\tilde{M}_{uv} + F_v\tilde{M}_u + F_u\tilde{M}_v + F_{uv}\tilde{M}) - (E_{uv}\tilde{N} + E_u\tilde{N}_v + E_v\tilde{N}_u + E\tilde{N}_{uv}) \\ &\quad -(G_{uv}\tilde{L} + G_u\tilde{L}_v + G_v\tilde{L}_u + G\tilde{L}_{uv}) \end{aligned} \tag{A.50}$$

$$\begin{aligned} B_{vv} &= 2(F\tilde{M}_{vv} + 2F_v\tilde{M}_v + F_{vv}\tilde{M}) - (E_{vv}\tilde{N} + 2E_v\tilde{N}_v + E\tilde{N}_{vv}) \\ &\quad -(G_{vv}\tilde{L} + 2G_v\tilde{L}_v + G\tilde{L}_{vv}) \end{aligned} \tag{A.51}$$

The Gaussian curvature

$$K = \frac{det[\Delta]}{det[\Gamma]} = \frac{LN - M^2}{EG - F^2} = \frac{\tilde{L}\tilde{N} - \tilde{M}^2}{S^4} = \frac{A}{S^4} \tag{A.52}$$

$$K_u = \frac{A_uS^2 - 4(\mathbf{S}\cdot\mathbf{S}_u)A}{S^6} = \frac{\hat{A}}{S^6} \tag{A.53}$$

$$K_v = \frac{A_vS^2 - 4(\mathbf{S}\cdot\mathbf{S}_v)A}{S^6} = \frac{\bar{A}}{S^6} \tag{A.54}$$

$$K_{uu} = \frac{\hat{A}_uS^2 - 6(\mathbf{S}\cdot\mathbf{S}_u)\hat{A}}{S^8} \tag{A.55}$$

$$K_{uv} = \frac{\hat{A}_vS^2 - 6(\mathbf{S}\cdot\mathbf{S}_v)\hat{A}}{S^8} \tag{A.56}$$

$$K_{vv} = \frac{\bar{A}_vS^2 - 6(\mathbf{S}\cdot\mathbf{S}_v)\bar{A}}{S^8} \tag{A.57}$$

where

$$\hat{A} = A_u S^2 - 4(\mathbf{S}\cdot\mathbf{S}_u)A \tag{A.58}$$

$$\hat{A}_u = A_{uu} S^2 - 2A_u(\mathbf{S}\cdot\mathbf{S}_u) - 4(\mathbf{S}_u\cdot\mathbf{S}_u + \mathbf{S}\cdot\mathbf{S}_{uu})A \tag{A.59}$$

$$\hat{A}_v = A_{uv} S^2 + 2A_u(\mathbf{S}\cdot\mathbf{S}_v) - 4(\mathbf{S}_u\cdot\mathbf{S}_v + \mathbf{S}\cdot\mathbf{S}_{uv})A - 4(\mathbf{S}\cdot\mathbf{S}_u)A_v \tag{A.60}$$

$$\bar{A} = A_v S^2 - 4(\mathbf{S}\cdot\mathbf{S}_v)A \tag{A.61}$$

$$\bar{A}_v = A_{vv} S^2 - 2A_v(\mathbf{S}\cdot\mathbf{S}_v) - 4(\mathbf{S}_v\cdot\mathbf{S}_v + \mathbf{S}\cdot\mathbf{S}_{vv})A \tag{A.62}$$

$$\tag{A.63}$$

The mean curvature

$$H = \frac{2FM - EN - GL}{2(EG - F^2)} = \frac{2F\tilde{M} - E\tilde{N} - G\tilde{L}}{2S^3} = \frac{B}{2S^3} \tag{A.64}$$

$$H_u = \frac{B_u S^2 - 3(\mathbf{S}\cdot\mathbf{S}_u)B}{2S^5} = \frac{\hat{B}}{2S^5} \tag{A.65}$$

$$H_v = \frac{B_v S^2 - 3(\mathbf{S}\cdot\mathbf{S}_v)B}{2S^5} = \frac{\bar{B}}{2S^5} \tag{A.66}$$

$$H_{uu} = \frac{\hat{B}_u S^2 - 5(\mathbf{S}\cdot\mathbf{S}_u)\hat{B}}{2S^7} \tag{A.67}$$

$$H_{uv} = \frac{\hat{B}_v S^2 - 5(\mathbf{S}\cdot\mathbf{S}_v)\hat{B}}{2S^7} \tag{A.68}$$

$$H_{vv} = \frac{\bar{B}_v S^2 - 5(\mathbf{S}\cdot\mathbf{S}_v)\bar{B}}{2S^7} \tag{A.69}$$

where

$$\hat{B} = B_u S^2 - 3(\mathbf{S}\cdot\mathbf{S}_u)B \tag{A.70}$$

$$\hat{B}_u = B_{uu} S^2 - B_u(\mathbf{S}\cdot\mathbf{S}_u) - 3(\mathbf{S}_u\cdot\mathbf{S}_u + \mathbf{S}\cdot\mathbf{S}_{uu})B \tag{A.71}$$

$$\hat{B}_v = B_{uv} S^2 + 2.0B_u(\mathbf{S}\cdot\mathbf{S}_v) - 3(\mathbf{S}_u\cdot\mathbf{S}_v + \mathbf{S}\cdot\mathbf{S}_{uv})B - 3(\mathbf{S}\cdot\mathbf{S}_u)B_v \tag{A.72}$$

$$\bar{B} = B_v S^2 - 3(\mathbf{S}\cdot\mathbf{S}_v)B \tag{A.73}$$

$$\bar{B}_v = B_{vv} S^2 - B_v(\mathbf{S}\cdot\mathbf{S}_v) - 3(\mathbf{S}_v\cdot\mathbf{S}_v + \mathbf{S}\cdot\mathbf{S}_{vv})B \tag{A.74}$$

The principal curvatures

$$\kappa = H \pm \sqrt{H^2 - K} = \frac{B \pm \sqrt{B^2 - 4AS^2}}{2S^3} \tag{A.75}$$

$$\kappa_u = \frac{2H_u\kappa - K_u}{2(\kappa - H)} \tag{A.76}$$

$$= \frac{(BB_u - 2A_uS^2)S^2 + (8AS^2 - 3B^2)(\mathbf{S}\cdot\mathbf{S}_u) \pm (B_uS^2 - 3(\mathbf{S}\cdot\mathbf{S}_u)B)\sqrt{B^2 - 4AS^2}}{2S^5\sqrt{B^2 - 4AS^2}} \tag{A.77}$$

$$\kappa_v = \frac{2H_v\kappa - K_v}{2(\kappa - H)} \tag{A.78}$$

$$= \frac{(BB_v - 2A_vS^2)S^2 + (8AS^2 - 3B^2)(\mathbf{S}\cdot\mathbf{S}_v) \pm (B_vS^2 - 3(\mathbf{S}\cdot\mathbf{S}_v)B)\sqrt{B^2 - 4AS^2}}{2S^5\sqrt{B^2 - 4AS^2}} \tag{A.79}$$

$$\kappa_{uu} = \frac{2H_{uu}\kappa + 4H_u\kappa_u - 2\kappa_u^2 - K_{uu}}{2(\kappa - H)} \tag{A.80}$$

$$\kappa_{uv} = \frac{2H_{uv}\kappa + 2(H_u\kappa_v + H_v\kappa_u) - 2\kappa_u\kappa_v - K_{uv}}{2(\kappa - H)} \tag{A.81}$$

$$\kappa_{vv} = \frac{2H_{vv}\kappa + 4H_v\kappa_v - 2\kappa_v^2 - K_{vv}}{2(\kappa - H)} \tag{A.82}$$

The root mean curvature

$$\kappa_{rms} = \sqrt{\kappa_{max}^2 + \kappa_{min}^2} = \sqrt{4H^2 - 2K} = \frac{\sqrt{B^2 - 2AS^2}}{S^3} \tag{A.83}$$

$$\frac{\partial \kappa_{rms}}{\partial u} = \frac{4HH_u - K_u}{\sqrt{4H^2 - 2K}} = \frac{(BB_u - A_uS^2)S^2 + (4AS^2 - 3B^2)\mathbf{S}\cdot\mathbf{S}_u}{S^5\sqrt{B^2 - 2AS^2}} \tag{A.84}$$

$$\frac{\partial \kappa_{rms}}{\partial v} = \frac{4HH_v - K_v}{\sqrt{4H^2 - 2K}} = \frac{(BB_v - A_vS^2)S^2 + (4AS^2 - 3B^2)\mathbf{S}\cdot\mathbf{S}_v}{S^5\sqrt{B^2 - 2AS^2}} \tag{A.85}$$

$$\frac{\partial^2 \kappa_{rms}}{\partial u^2} = \frac{8(H_u^2 + HH_{uu} - K_{uu})(2H^2 - K) - (4HH_u - K_u)^2}{(4H^2 - 2K)^{\frac{3}{2}}} \tag{A.86}$$

$$\frac{\partial^2 \kappa_{rms}}{\partial u \partial v} = \frac{8(H_uH_v + HH_{uv} - K_{uv})(2H^2 - K) - (4HH_u - K_u)(4HH_v - K_v)}{(4H^2 - 2K)^{\frac{3}{2}}} \tag{A.87}$$

$$\frac{\partial^2 \kappa_{rms}}{\partial v^2} = \frac{8(H_v^2 + HH_{vv} - K_{vv})(2H^2 - K) - (4HH_v - K_v)^2}{(4H^2 - 2K)^{\frac{3}{2}}} \tag{A.88}$$

Appendix B

Classification of Stationary Points of Functions

In this appendix we review some relevant material from the extrema theory of functions necessary in the classification of stationary points of curvature [18], [11].

Single Variable: Let $f(x)$ be a continuous, sufficiently differentiable, function of one variable x, then a necessary condition that f is a maximum or a minimum at $x = a$ is

$$f'(x) = 0 \;\; at \;\; x = a \tag{B.1}$$

The function $f(x)$ has a maximum, a minimum or neither maximum nor minimum according to

- if $f''(a) < 0$; maximum
- if $f''(a) > 0$; minimum
- if $f''(a) = 0$;
  - if $f'''(a) \neq 0$; neither maximum nor minimum
  - if $f'''(a) = 0$ and $f^{(iv)}(a) < 0$; maximum

- if $f'''(a) = 0$ and $f^{(iv)}(a) > 0$; minimum

In general, if $f^{(n)}(a)$ is the first derivative function that does not vanish then

- if n is odd; neither maximum nor minimum;
- if n is even;
  - if $f^{(n)}(a) < 0$; maximum
  - if $f^{(n)}(a) > 0$; minimum

Two Variables: Let $f(x, y)$ be a continuous function of two variables x and y. A necessary condition that f has an extremum at (x_0, y_0) is

$$f_x = f_y = 0 \quad at \quad (x_0, y_0) \tag{B.2}$$

Let H^* denote the Hessian matrix $\begin{pmatrix} f_{xx} & f_{xy} \\ f_{xy} & f_{yy} \end{pmatrix}$, then $f(x, y)$ has a maximum, a minimum or a saddle point according to

- if $f_{xx} < 0$ and $det[H^*] > 0$ at (x_0, y_0) : maximum
- if $f_{xx} > 0$ and $det[H^*] > 0$ at (x_0, y_0) : minimum
- if $det[H^*] < 0$ at (x_0, y_0) : saddle point
- if $det[H^*] = 0$: higher-order partial derivatives must be considered

For degenerate case *i.e.* $det[H^*] = 0$, there is a theorem by Scheeffer to classify the extrema, see [11].

The above relations are valid for the classification of the Stationary points for all the different curvatures discussed, except at the umbilical points for the principal curvatures. In the case of an umbilical point, a separate criterion exist, [18], [21].

Theorem (Criterion for extrema of principal curvature functions at umbilics): If we denote $W(u, v) = H^2(u, v) - K(u, v)$ and assume that $W(u, v)$ is at least C^2 smooth and at least one of the second order partial derivatives of $W(u, v)$ does not vanish then:

1. If $\nabla H = \mathbf{0}$ at the umbilic, then κ_{max} has a local minimum and κ_{min} has a local maximum.

3. If $\nabla H \neq \mathbf{0}$ at the umbilic, then κ_{max} has a local minimum and κ_{min} has a local maximum if and only if $(2HH_{uv} - K_{uv})^2 - (2HH_{uu} - K_{uu})(2HH_{vv} - K_{vv}) < 0$. In the case the previous relation is zero, additional evaluation of higher order terms is necessary.

Appendix C

Knot Vectors and Control Points for the Surfaces Used

We present here the knot vectors and control points for the surfaces used as examples in this thesis.

For the 1 Bézier patch sinusoidal surface, Fig. (6-1):

Knot Vector - u : $[0,0,0,0,1,1,1,1]$

Knot Vector - v : $[0,0,0,0,1,1,1,1]$

Control Points :

$$\begin{pmatrix} \mathbf{P}_{00} & \mathbf{P}_{10} & \mathbf{P}_{20} & \mathbf{P}_{30} \\ \mathbf{P}_{01} & \mathbf{P}_{11} & \mathbf{P}_{21} & \mathbf{P}_{31} \\ \mathbf{P}_{02} & \mathbf{P}_{12} & \mathbf{P}_{22} & \mathbf{P}_{32} \\ \mathbf{P}_{03} & \mathbf{P}_{13} & \mathbf{P}_{23} & \mathbf{P}_{33} \end{pmatrix} = \begin{pmatrix} (0,\ 0,\ 0) & (\frac{1}{3},\ 0,\ 0) & (\frac{2}{3},\ 0,\ 0) & (1,\ 0,\ 0) \\ (0,\ \frac{1}{3},\ 0) & (\frac{1}{3},\ \frac{1}{3},\ 1) & (\frac{2}{3},\ \frac{1}{3},\ -1) & (1,\ \frac{1}{3},\ 0) \\ (0,\ \frac{2}{3},\ 0) & (\frac{1}{3},\ \frac{2}{3},\ \frac{3}{5}) & (\frac{2}{3},\ \frac{2}{3},\ -\frac{3}{5}) & (1,\ \frac{2}{3},\ 0) \\ (0,\ 1,\ 0) & (\frac{1}{3},\ 1,\ 0) & (\frac{2}{3},\ 1,\ 0) & (1,\ 1,\ 0) \end{pmatrix}$$

For the 1 Bézier patch surface in Fig. (6-2):

Knot Vector - u : $[0,0,0,0,1,1,1,1]$

Knot Vector - v : $[0,0,0,0,1,1,1,1]$

Control Points :

$$\begin{array}{lcl}
\mathbf{P}_{00} & = & (\ 0.00000000000000,\ \ 0.10000000000000,\ \ 0.00000000000000) \\
\mathbf{P}_{01} & = & (\ 0.76166807045299,\ -0.17871973643107,\ -0.13653427756501) \\
\mathbf{P}_{02} & = & (\ 0.65723271184842,\ -0.15502141067231,\ -0.16038769731139) \\
\mathbf{P}_{03} & = & (\ 1.60000000000000,\ \ 0.10000000000000,\ \ 0.10000000000000) \\
\mathbf{P}_{10} & = & (\ 0.27852446657714,\ -0.15693717074725,\ \ 1.21583433259486) \\
\mathbf{P}_{11} & = & (\ 0.57935996328779,\ -0.11426636286158,\ \ 1.04018643363983) \\
\mathbf{P}_{12} & = & (\ 0.43919924825324,\ -0.08662265922664,\ \ 0.78854102569545) \\
\mathbf{P}_{13} & = & (\ 1.38042813814110,\ -0.13906075428985,\ \ 1.01172029862168) \\
\mathbf{P}_{20} & = & (\ 0.32975054541603,\ -0.17658556930683,\ \ 1.37478487401770) \\
\mathbf{P}_{21} & = & (\ 0.69080030605031,\ -0.13624558726510,\ \ 1.24026710894905) \\
\mathbf{P}_{22} & = & (\ 0.52367956768821,\ -0.10328459558212,\ \ 0.94021750966775) \\
\mathbf{P}_{23} & = & (\ 1.31296957939178,\ -0.13507415315986,\ \ 0.99321070380470) \\
\mathbf{P}_{30} & = & (-0.20000000000000,\ \ 0.10000000000000,\ \ 2.30000000000000) \\
\mathbf{P}_{31} & = & (\ 0.74243450440050,\ -0.00388300211731,\ \ 1.75989287977052) \\
\mathbf{P}_{32} & = & (\ 0.75124557906142,\ \ 0.01730029662848,\ \ 1.84436023893511) \\
\mathbf{P}_{33} & = & (\ 1.50000000000000,\ -0.10000000000000,\ \ 2.10000000000000)
\end{array}$$

For the 2x2 Bézier patch, Hat-like surface, Fig. (6-3):

Knot Vector - u : $[0,0,0,0,0.5,1,1,1,1]$

Knot Vector - v : $[0,0,0,0,0.5,1,1,1,1]$

Control Points :

$$\begin{pmatrix} \mathbf{P}_{00} & \mathbf{P}_{10} & \mathbf{P}_{20} & \mathbf{P}_{30} & \mathbf{P}_{40} \\ \mathbf{P}_{01} & \mathbf{P}_{11} & \mathbf{P}_{21} & \mathbf{P}_{31} & \mathbf{P}_{41} \\ \mathbf{P}_{02} & \mathbf{P}_{12} & \mathbf{P}_{22} & \mathbf{P}_{32} & \mathbf{P}_{42} \\ \mathbf{P}_{03} & \mathbf{P}_{13} & \mathbf{P}_{23} & \mathbf{P}_{33} & \mathbf{P}_{43} \\ \mathbf{P}_{04} & \mathbf{P}_{14} & \mathbf{P}_{24} & \mathbf{P}_{34} & \mathbf{P}_{44} \end{pmatrix} = \begin{pmatrix} (0,\ 0,\ \frac{1}{4}) & (\frac{1}{6},\ 0,\ 0) & (\frac{1}{2},\ 0,\ -\frac{1}{4}) & (\frac{5}{6},\ 0,\ 0) & (1,\ 0,\ \frac{1}{4}) \\ (0,\ \frac{1}{6},\ 0) & (\frac{1}{6},\ \frac{1}{6},\ 0) & (\frac{1}{2},\ \frac{1}{6},\ 0) & (\frac{5}{6},\ \frac{1}{6},\ 0) & (1,\ \frac{1}{6},\ 0) \\ (0,\ \frac{1}{2},\ -\frac{1}{4}) & (\frac{1}{6},\ \frac{1}{2},\ 0) & (\frac{1}{2},\ \frac{1}{2},\ \frac{1}{4}) & (\frac{5}{6},\ \frac{1}{2},\ 0) & (1,\ \frac{1}{2},\ -\frac{1}{4}) \\ (0,\ \frac{5}{6},\ 0) & (\frac{1}{6},\ \frac{5}{6},\ 0) & (\frac{1}{2},\ \frac{5}{6},\ 0) & (\frac{5}{6},\ \frac{5}{6},\ 0) & (1,\ \frac{5}{6},\ 0) \\ (0,\ 1,\ \frac{1}{4}) & (\frac{1}{6},\ 1,\ 0) & (\frac{1}{2},\ 1,\ -\frac{1}{4}) & (\frac{5}{6},\ 1,\ 0) & (1,\ 1,\ \frac{1}{4}) \end{pmatrix}$$

For the 7x7 Bézier patch, sinusoidal surface, Fig. (6-4):

Knot Vector - u : $[0,0,0,0,0.23835,0.35382,0.45330,0.54670,0.64176,0.76164,1,1,1,1]$

Knot Vector - v : $[0, 0, 0, 0, 0.20612, 0.31053, 0.43342, 0.56658, 0.68947, 0.79388, 1, 1, 1, 1]$

Control Points :

$\mathbf{P}_{00} = (0.00000000000000, 0.00000000000000, -0.06221300000000)$

$\mathbf{P}_{01} = (0.06405088105174, 0.00000000000000, 0.29135091715567)$

$\mathbf{P}_{02} = (0.16535046273978, 0.00000000000000, 0.64727762755190)$

$\mathbf{P}_{03} = (0.37220662017482, 0.00000000000000, 0.36063096405809)$

$\mathbf{P}_{04} = (0.45131622491635, 0.00000000000000, 0.12518026234675)$

$\mathbf{P}_{05} = (0.54868377508365, 0.00000000000000, -0.12518026234675)$

$\mathbf{P}_{06} = (0.62779337982518, 0.00000000000000, -0.36063096405809)$

$\mathbf{P}_{07} = (0.83464953726022, 0.00000000000000, -0.64727762755190)$

$\mathbf{P}_{08} = (0.93594911894826, 0.00000000000000, -0.29135091715567)$

$\mathbf{P}_{09} = (1.00000000000000, 0.00000000000000, 0.06221300000000)$

$\mathbf{P}_{10} = (0.00000000000000, 0.13583253485821, -0.01659031675964)$

$\mathbf{P}_{11} = (0.06145704960053, 0.10898676587787, 0.16987645738421)$

$\mathbf{P}_{12} = (0.16081849604788, 0.09162968474461, 0.29464539620041)$

$\mathbf{P}_{13} = (0.34867661129242, 0.09944550790955, 0.19686984290714)$

$\mathbf{P}_{14} = (0.44820516807221, 0.13154829414632, 0.04098895298656)$

$\mathbf{P}_{15} = (0.55179483192779, 0.13154829414633, -0.04098895298656)$

$\mathbf{P}_{16} = (0.65132338870758, 0.09944550790955, -0.19686984290714)$

$\mathbf{P}_{17} = (0.83918150395212, 0.09162968474461, -0.29464539620041)$

$\mathbf{P}_{18} = (0.93854295039947, 0.10898676587787, -0.16987645738421)$

$\mathbf{P}_{19} = (1.00000000000000, 0.13583253485821, 0.01659031675964)$

$\mathbf{P}_{20} = (0.00000000000000, 0.20526948831241, -0.00229202238419)$

$\mathbf{P}_{21} = (0.14894930522778, 0.13955288675633, 0.07522978969975)$

$\mathbf{P}_{22} = (0.20330364746607, 0.16436407536845, 0.11761919945117)$

$\mathbf{P}_{23} = (0.32867576616622, 0.15523791750164, 0.08060530429953)$

$\mathbf{P}_{24} = (0.43452525244251, 0.20227632087823, -0.01288762447697)$

$$
\begin{array}{lclll}
\mathbf{P}_{25} & = & (0.56547474755749, & 0.20227632087823, & 0.01288762447697) \\
\mathbf{P}_{26} & = & (0.67132423383379, & 0.15523791750164, & -0.08060530429953) \\
\mathbf{P}_{27} & = & (0.79669635253393, & 0.16436407536845, & -0.11761919945117) \\
\mathbf{P}_{28} & = & (0.85105069477222, & 0.13955288675633, & -0.07522978969975) \\
\mathbf{P}_{29} & = & (1.00000000000000, & 0.20526948831241, & 0.00229202238419) \\
\mathbf{P}_{30} & = & (0.00000000000000, & 0.36201976999900, & 0.02241372373392) \\
\mathbf{P}_{31} & = & (0.09870897349313, & 0.30258105125496, & -0.14752083539221) \\
\mathbf{P}_{32} & = & (0.18687359443761, & 0.28714446682970, & -0.09716096566139) \\
\mathbf{P}_{33} & = & (0.33262511044888, & 0.29794762830283, & -0.11059319065992) \\
\mathbf{P}_{34} & = & (0.44948459495528, & 0.36211148266435, & -0.06422486518159) \\
\mathbf{P}_{35} & = & (0.55051540504472, & 0.36211148266435, & 0.06422486518159) \\
\mathbf{P}_{36} & = & (0.66737488955112, & 0.29794762830283, & 0.11059319065992) \\
\mathbf{P}_{37} & = & (0.81312640556239, & 0.28714446682970, & 0.09716096566139) \\
\mathbf{P}_{38} & = & (0.90129102650687, & 0.30258105125496, & 0.14752083539221) \\
\mathbf{P}_{39} & = & (1.00000000000000, & 0.36201976999900, & -0.02241372373392) \\
\mathbf{P}_{40} & = & (0.00000000000000, & 0.45273715439131, & 0.02692166188480) \\
\mathbf{P}_{41} & = & (0.08630427975619, & 0.44102447024057, & -0.17119160466027) \\
\mathbf{P}_{42} & = & (0.17579438354938, & 0.40908095888274, & -0.26476773050818) \\
\mathbf{P}_{43} & = & (0.34558813204131, & 0.43171959569505, & -0.18193122404545) \\
\mathbf{P}_{44} & = & (0.44825912355741, & 0.45449790128174, & -0.06098474674169) \\
\mathbf{P}_{45} & = & (0.55174087644259, & 0.45449790128174, & 0.06098474674169) \\
\mathbf{P}_{46} & = & (0.65441186795869, & 0.43171959569505, & 0.18193122404545) \\
\mathbf{P}_{47} & = & (0.82420561645062, & 0.40908095888274, & 0.26476773050818) \\
\mathbf{P}_{48} & = & (0.91369572024381, & 0.44102447024057, & 0.17119160466027) \\
\mathbf{P}_{49} & = & (1.00000000000000, & 0.45273715439131, & -0.02692166188480)
\end{array}
$$

$\mathbf{P}_{50}$ = (0.00000000000000, 0.54726284560869, 0.02692166188480)

$\mathbf{P}_{51}$ = (0.08630427975619, 0.55897552975943, -0.17119160466027)

$\mathbf{P}_{52}$ = (0.17579438354938, 0.59091904111726, -0.26476773050818)

$\mathbf{P}_{53}$ = (0.34558813204131, 0.56828040430495, -0.18193122404545)

$\mathbf{P}_{54}$ = (0.44825912355741, 0.54550209871826, -0.06098474674169)

$\mathbf{P}_{55}$ = (0.55174087644259, 0.54550209871826, 0.06098474674169)

$\mathbf{P}_{56}$ = (0.65441186795869, 0.56828040430495, 0.18193122404545)

$\mathbf{P}_{57}$ = (0.82420561645062, 0.59091904111726, 0.26476773050818)

$\mathbf{P}_{58}$ = (0.91369572024381, 0.55897552975943, 0.17119160466026)

$\mathbf{P}_{59}$ = (1.00000000000000, 0.54726284560869, -0.02692166188480)

$\mathbf{P}_{60}$ = (0.00000000000000, 0.63798023000100, 0.02241372373392)

$\mathbf{P}_{61}$ = (0.09870897349313, 0.69741894874504, -0.14752083539221)

$\mathbf{P}_{62}$ = (0.18687359443761, 0.71285553317030, -0.09716096566139)

$\mathbf{P}_{63}$ = (0.33262511044888, 0.70205237169717, -0.11059319065992)

$\mathbf{P}_{64}$ = (0.44948459495528, 0.63788851733565, -0.06422486518159)

$\mathbf{P}_{65}$ = (0.55051540504472, 0.63788851733564, 0.06422486518159)

$\mathbf{P}_{66}$ = (0.66737488955112, 0.70205237169717, 0.11059319065992)

$\mathbf{P}_{67}$ = (0.81312640556239, 0.71285553317030, 0.09716096566139)

$\mathbf{P}_{68}$ = (0.90129102650687, 0.69741894874504, 0.14752083539221)

$\mathbf{P}_{69}$ = (1.00000000000000, 0.63798023000100, -0.02241372373392)

$\mathbf{P}_{70}$ = (0.00000000000000, 0.79473051168759, -0.00229202238419)

$\mathbf{P}_{71}$ = (0.14894930522778, 0.86044711324367, 0.07522978969975)

$\mathbf{P}_{72}$ = (0.20330364746607, 0.83563592463156, 0.11761919945117)

$\mathbf{P}_{73}$ = (0.32867576616622, 0.84476208249836, 0.08060530429953)

$\mathbf{P}_{74}$ = (0.43452525244250, 0.79772367912177, -0.01288762447697)

$\mathbf{P}_{75}$	=	(0.56547474755750,	0.79772367912177,	0.01288762447697)
$\mathbf{P}_{76}$	=	(0.67132423383379,	0.84476208249836,	-0.08060530429953)
$\mathbf{P}_{77}$	=	(0.79669635253393,	0.83563592463156,	-0.11761919945117)
$\mathbf{P}_{78}$	=	(0.85105069477222,	0.86044711324367,	-0.07522978969975)
$\mathbf{P}_{79}$	=	(1.00000000000000,	0.79473051168759,	0.00229202238419)
$\mathbf{P}_{80}$	=	(0.00000000000000,	0.86416746514179,	-0.01659031675964)
$\mathbf{P}_{81}$	=	(0.06145704960053,	0.89101323412213,	0.16987645738421)
$\mathbf{P}_{82}$	=	(0.16081849604788,	0.90837031525539,	0.29464539620041)
$\mathbf{P}_{83}$	=	(0.34867661129242,	0.90055449209045,	0.19686984290714)
$\mathbf{P}_{84}$	=	(0.44820516807221,	0.86845170585367,	0.04098895298656)
$\mathbf{P}_{85}$	=	(0.55179483192779,	0.86845170585368,	-0.04098895298656)
$\mathbf{P}_{86}$	=	(0.65132338870758,	0.90055449209045,	-0.19686984290714)
$\mathbf{P}_{87}$	=	(0.83918150395211,	0.90837031525539,	-0.29464539620041)
$\mathbf{P}_{88}$	=	(0.93854295039947,	0.89101323412213,	-0.16987645738421)
$\mathbf{P}_{89}$	=	(1.00000000000000,	0.86416746514179,	0.01659031675964)
$\mathbf{P}_{90}$	=	(0.00000000000000,	1.00000000000000,	-0.06221300000000)
$\mathbf{P}_{91}$	=	(0.06405088105174,	1.00000000000000,	0.29135091715567)
$\mathbf{P}_{92}$	=	(0.16535046273978,	1.00000000000000,	0.64727762755190)
$\mathbf{P}_{93}$	=	(0.37220662017482,	1.00000000000000,	0.36063096405809)
$\mathbf{P}_{94}$	=	(0.45131622491635,	1.00000000000000,	0.12518026234675)
$\mathbf{P}_{95}$	=	(0.54868377508365,	1.00000000000000,	-0.12518026234675)
$\mathbf{P}_{96}$	=	(0.62779337982518,	1.00000000000000,	-0.36063096405809)
$\mathbf{P}_{97}$	=	(0.83464953726022,	1.00000000000000,	-0.64727762755190)
$\mathbf{P}_{98}$	=	(0.93594911894826,	1.00000000000000,	-0.29135091715567)
$\mathbf{P}_{99}$	=	(1.00000000000000,	1.00000000000000,	0.06221300000000)

Bibliography

[1] P. G. Alourdas. Shape creation, interrogation and fairing using B-splines. Engineer's thesis, Massachusetts Institute of Technology, Department of Ocean Engineering, Cambridge, Massachusetts, 1989.

[2] T. Banchoff, T. Gaffney, and C. McCrory. *Cusps of Gauss Mappings.* Pittman, Boston, 1982.

[3] W. Boehm. Inserting new knots into B-spline curves. *Computer Aided Design*, 12(4):199–201, 1980.

[4] L. Chang, W. Bein, and A. Angel. Surface intersection using parallelism. *Computer Aided Geometric Design*, 11:39–69, 1994.

[5] E. Cohen, T. Lyche, and R. F. Riesenfeld. Discrete B-splines and subdivision techniques in computer-aided geometric design and computer graphics. *Computer Graphics and Image Processing*, 14:87–111, 1980.

[6] P. M. do Carmo. *Differential Geometry of Curves and Surfaces.* Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1976.

[7] J. J. Dongarra, G. A. Geist, R. Manchek, and V. S. Sunderam. Integrated PVM framework supports heterogeneous network computing. *Computers in Physics*, 7(2):166–175, April 1993.

[8] G. Farin. *Curves and Surfaces for Computer Aided Geometric Design: A Practical Guide.* Academic Press, San Diego, CA, 3rd edition, 1993.

[9] I. D. Faux and M. J. Pratt. *Computational Geometry for Design and Manufacture.* Ellis Horwood, Chichester, England, 1981.

[10] A. Geist, A. Beguelin, J. Dongarra, W. Jiang, R. Manchek, and V. Sunderam. PVM 3 user's guide and reference manual. Technical Report ORNL/TM-12187, Oak Ridge National Laboratory, Oak Ridge, Tennessee, May 1993.

[11] H. Hancock. *Theory of Maxima and Minima.* Dover, New York, 1960.

[12] D. Hilbert and S. Cohn-Vossen. *Geometry and the Imagination.* Chelsea, New York, 1952.

[13] C.-Y. Hu. *Robust Algorithms for Sculptured Shape Visualization.* Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, July 1993.

[14] C.-Y. Hu, T. Maekawa, and N. M. Patrikalakis. Robust 2D interval solid modeller. Design Laboratory Memorandum 94-6, MIT, Department of Ocean Engineering, Cambridge, MA, June 1994. In preparation.

[15] J. J. Koenderink. *Solid Shape.* MIT Press, Cambridge, MA, 1990.

[16] T. Lyche, E. Cohen, and K. Mørken. Knot line refinement algorithms for tensor product B-spline surfaces. *Computer Aided Geometric Design*, 2:133–139, 1985.

[17] T. Maekawa. Personal communication, December 1993.

[18] T. Maekawa. *Robust Computational Methods for Shape Interrogation.* PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, June 1993.

[19] T. Maekawa and N. M. Patrikalakis. Computation of singularities and intersections of offsets of planar curves. *Computer Aided Geometric Design*, 10(5):407–429, October 1993.

[20] T. Maekawa and N. M. Patrikalakis. Interrogation of differential geometry properties for design and manufacture. *The Visual Computer*, 10(4):216–237, March 1994.

[21] T. Maekawa, F.-E. Wolter, and N. M. Patrikalakis. Umbilics and lines of curvature for shape interrogation. Design Laboratory Memorandum 93-4, MIT Department of Ocean Engineering, Cambridge, MA, April 1993. Revised March 1994.

[22] R. Manchek. An introduction to PVM (Parallel Virtual Machine). Slides from the PVM Workshop, University of Tennessee, Knoxville, August 1993.

[23] R. E. Moore. *Interval Analysis.* Prentice-Hall, Englewood Cliffs, NJ, 1966.

[24] F. C. Munchmeyer. Shape interrogation: A case study. In G. Farin, editor, *Geometric Modeling*, pages 291–301. SIAM, Philadelphia, PA, 1987.

[25] T. Nishita, T. W. Sederberg, and M. Kakimoto. Ray tracing trimmed rational surface patches. *ACM Computer Graphics*, 24(4):337–345, August 1990.

[26] Numerical Algorithms Group, Oxford, England. *NAG Fortran Library Manual, Volumes 1-8*, mark 15 edition, 1991.

[27] N. M. Patrikalakis, T. Maekawa, E. C. Sherbrooke, and J. Zhou. Computation of singularities for engineering design. In T. L. Kunii and Y. Shinagawa, editors, *Modern Geometric Computing for Visualization*, pages 167–191. Tokyo: Springer-Verlag, June 1992.

[28] A. Preusser. Computing area filling contours for surface defined by piecewise polynomials. *Computer Aided Geometric Design*, 3:267–279, 1986.

[29] E. C. Sherbrooke. *Computation of the Solutions of Nonlinear Polynomial Systems.* Master's thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, October 1993.

[30] E. C. Sherbrooke and N. M. Patrikalakis. Computation of the solutions of nonlinear polynomial systems. *Computer Aided Geometric Design*, 10(5):379–405, October 1993.

[31] D. J. Struik. *Lectures on Classical Differential Geometry.* Addison-Wesley, Cambridge Mass., 1950.

[32] S. T. Tuohy. Personal communication, December 1993.

[33] F. Yamaguchi. *Curves and Surfaces in Computer Aided Geometric Design.* Springer-Verlag, NY, 1988.