Laser Measurement and Reconstruction of Curved Plates

by

Kwang Hee Ko

Seoul National University, Republic of Korea

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
June 2001

© Massachusetts Institute of Technology 2001. All rights reserved.
Laser Measurement and Reconstruction of Curved Plates
by
Kwang Hee Ko

Submitted to the Department of Ocean Engineering
and the Department of Mechanical Engineering
on May 11, 2001, 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
An automated system has been developed which acquires range image data from
a curved plate, analyzes them and yields a suitable mathematical representation.
The plate is assumed to be topologically equivalent to a four sided free-form surface
representable in a parametric form, and all tangent planes of the surface form a small
angle with respect to a reference plane. The system consists of three subsystems:
a laser scanner, a robot and a workstation. Each subsystem is connected through
TCP/IP and a serial line so that all operations are synchronized. The limitation of
the scanner that the size of the rectangular array of points produced by the scanner
is too small to capture the whole shape is overcome by attaching the scanner to the
robot so that enough shots to cover the whole object can be taken by placing the
scanner at appropriate positions. The procedure starts with detecting the edges of
the plate, which is necessary to compute the number and position of the shots that
are needed to obtain the full model. Edges are detected by investigating changes of
tangent angles along each raster line of the scanner. The estimation of boundary is
followed by scanning of the plate that starts from a corner and sweeps through the
whole plate in a zig-zag manner. The data of each view are registered in a common
coordinate system to get an appropriate representation of the plate. Data points along
raster lines of digitization are approximated by B-spline curves, which are provided as
an input to the surface reconstruction algorithm which has been developed based on
the concept of a curve on a surface. This new approach creates a B-spline surface from
the input in the least squares sense. Theoretical and numerical complexity analysis
is performed and examples are presented. Comparison between the measured data
from the plate and the reconstructed B-spline surface is made.

Thesis Co-Supervisor: Nicholas M. Patrikalakis, Kawasaki Professor of Engineering,
Professor of Ocean and Mechanical Engineering

Thesis Co-Supervisor: Takashi Maekawa, Lecturer and Principal Research Scientist
Acknowledgments

First of all, I want to thank my family for their love, understanding, and support during my study at MIT and my fiancée, Suyeon, for her love and emotional support.

I would like to thank my thesis supervisors, Professor Nicholas M. Patrikalakis and Dr. Takashi Maekawa, for their expert advice on my research work and instructive guidance on my academic studies.

Thanks also go to Mr. Fred Baker, for providing a stable hardware environment for my thesis work and Design Laboratory fellows Dr. Wonjoon Cho, Mr. Robert Damus, Dr. Todd R. Jackson, Ms. Hongye Liu, Dr. Guoling Shen, Dr. Guoxin Yu and Mr. Pubudu Wariyapola for useful discussions and Ms. Kristin Gunst for making a good laboratory environment.

Funding for this research was obtained in part from the New Industry Research Organization (NIRO), and the Department of Ocean Engineering.
# Contents

Abstract ........................................ 3
Acknowledgments .................................. 4
Contents ......................................... 5
List of Figures .................................... 8
List of Tables ..................................... 10

## 1 Introduction ................................ 11
  1.1 Background and Motivation ............... 11
  1.2 Research Objectives ...................... 12
  1.3 Literature Review ......................... 12
  1.4 Thesis Organization ...................... 12

## 2 Theoretical Background ................. 14
  2.1 Introduction ................................ 14
  2.2 Review of Differential Geometry of Curves and Surfaces .... 14
      2.2.1 Basic Theory of Curves ............ 14
      2.2.2 Basic Theory of Surfaces .......... 17
  2.3 Bézier Curve and Surface Representation .......... 21
      2.3.1 Bernstein Polynomials ............ 21
      2.3.2 Bézier Curves and Surfaces ....... 22
      2.3.3 Degree Elevation .................. 26
      2.3.4 Subdivision ....................... 27
  2.4 B-spline Curve and Surface Representation ........ 27
      2.4.1 B-spline Basis Function .......... 27
      2.4.2 B-spline Curves and Surfaces ...... 28
      2.4.3 Knot Insertion .................... 32
  2.5 Coons Surface Patch ..................... 34
  2.6 Lofting .................................... 35

## 3 Surface Approximation Using the Concept of a Curve on a Surface .... 38
  3.1 Introduction ................................ 38
  3.2 Curves on a Bézier Surface Patch .......... 38
### Table of Contents

3.2.1 Formulation for Bézier Surface Patch ...................................... 38
3.3 Curves on a B-spline Surface Patch ........................................... 43
3.4 Boundary Condition .............................................................. 46
3.5 B-spline Surface Approximation ............................................... 47
  3.5.1 Least Squares Method ..................................................... 47
3.6 Complexity Analysis ............................................................ 48

4 System and Configuration .......................................................... 51
  4.1 Introduction ............................................................................. 51
  4.2 Hardware and Software ........................................................ 51
    4.2.1 Laser Scanner ............................................................... 51
    4.2.2 Robot ............................................................................. 52
    4.2.3 Host Computer ............................................................. 53
    4.2.4 Accuracy .................................................................... 53
  4.3 Configuration .......................................................................... 54
    4.3.1 TCP/IP .......................................................................... 54
    4.3.2 Serial Communication .................................................. 56
    4.3.3 Connection ................................................................. 57

5 System Operation Procedures ....................................................... 59
  5.1 Introduction ............................................................................. 59
  5.2 Initialization ........................................................................... 59
    5.2.1 Scanner and Network .................................................... 59
    5.2.2 Robot ............................................................................ 60
  5.3 Flow Chart of the System ........................................................ 60
    5.3.1 Overview ....................................................................... 60
    5.3.2 Boundary Construction ................................................. 60
    5.3.3 Surface Scanning .......................................................... 64
    5.3.4 Surface Fitting ............................................................. 65
  5.4 Complexity Analysis of the System .......................................... 67

6 Applications and Examples .......................................................... 69
  6.1 Introduction ............................................................................. 69
  6.2 Surface Reconstruction .......................................................... 69
    6.2.1 Artificial Data .................................................................. 69
    6.2.2 Measured Data .............................................................. 70
  6.3 Surface Creation ....................................................................... 70
  6.4 Engineering Example of the System ......................................... 70
  6.5 Experimental Complexity Analysis .......................................... 73
    6.5.1 Bézier Surface Approximation ....................................... 73
    6.5.2 B-spline Surface Approximation .................................... 74

7 Conclusions and Future Work ...................................................... 83
  7.1 Conclusions and Contributions ................................................ 83
  7.2 Future Work ........................................................................... 84
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-1</td>
<td>Derivation of the normal vector</td>
</tr>
<tr>
<td>2-2</td>
<td>Normal, tangent and binormal vectors</td>
</tr>
<tr>
<td>2-3</td>
<td>Definition of a regular surface [12]</td>
</tr>
<tr>
<td>2-4</td>
<td>Tangent plane and normal vector</td>
</tr>
<tr>
<td>2-5</td>
<td>Definition of positive normal curvature (a) $\kappa n \cdot N = \kappa_n$; (b) $\kappa n \cdot N = -\kappa_n$</td>
</tr>
<tr>
<td>2-6</td>
<td>A Bézier curve and its hodograph</td>
</tr>
<tr>
<td>2-7</td>
<td>The de Casteljau algorithm</td>
</tr>
<tr>
<td>2-8</td>
<td>A biquadratic Bézier surface</td>
</tr>
<tr>
<td>2-9</td>
<td>Degree elevation</td>
</tr>
<tr>
<td>2-10</td>
<td>B-spline curve</td>
</tr>
<tr>
<td>2-11</td>
<td>The de Boor algorithm</td>
</tr>
<tr>
<td>3-1</td>
<td>Parametrization</td>
</tr>
<tr>
<td>3-2</td>
<td>Diagram for Bézier surface approximation</td>
</tr>
<tr>
<td>3-3</td>
<td>(a) A bicubic B-spline surface patch and its control points. (b) After knot insertion, the B-spline surface has been split into 9 Bézier surface patches.</td>
</tr>
<tr>
<td>4-1</td>
<td>Data points captured by the scanner</td>
</tr>
<tr>
<td>4-2</td>
<td>The Adept SCARA type robot</td>
</tr>
<tr>
<td>4-3</td>
<td>Diagram of software modules of the system</td>
</tr>
<tr>
<td>4-4</td>
<td>A diagram for TCP/IP connection</td>
</tr>
<tr>
<td>4-5</td>
<td>A diagram for socket interface</td>
</tr>
<tr>
<td>4-6</td>
<td>Configuration of the system</td>
</tr>
<tr>
<td>5-1</td>
<td>Overview of the system</td>
</tr>
<tr>
<td>5-2</td>
<td>Data set containing an edge</td>
</tr>
<tr>
<td>5-3</td>
<td>Corner detection</td>
</tr>
<tr>
<td>5-4</td>
<td>Tangent vectors of extracted edge points at both ends</td>
</tr>
<tr>
<td>5-5</td>
<td>Corner is detected by the difference of the tangent vectors at both ends</td>
</tr>
<tr>
<td>5-6</td>
<td>The global coordinate system</td>
</tr>
<tr>
<td>5-7</td>
<td>Merged data points and reconstructed surface through lofting</td>
</tr>
<tr>
<td>5-8</td>
<td>Isoparametric lines of the reconstructed surface through lofting</td>
</tr>
<tr>
<td>6-1</td>
<td>Surface reconstruction of the model surface</td>
</tr>
<tr>
<td>6-2</td>
<td>Reconstructed surface based on measured data</td>
</tr>
<tr>
<td>Page</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>6-3</td>
<td>(a) Plot of elapsed time vs. the number of control points (b) Plot in log scale.</td>
</tr>
<tr>
<td>6-4</td>
<td>Plot of elapsed time vs. the number of control points</td>
</tr>
<tr>
<td>6-5</td>
<td>The left side of a car hood created from sketch curves</td>
</tr>
<tr>
<td>6-6</td>
<td>Interrogation of the reconstructed surface</td>
</tr>
<tr>
<td>6-7</td>
<td>Reconstruction of fuselage</td>
</tr>
<tr>
<td>6-8</td>
<td>The model plate</td>
</tr>
<tr>
<td>6-9</td>
<td>Detected boundary</td>
</tr>
<tr>
<td>6-10</td>
<td>Scanned data points and reconstructed surface</td>
</tr>
<tr>
<td>6-11</td>
<td>Interrogation of the reconstructed surface</td>
</tr>
<tr>
<td>6-12</td>
<td>Reconstructed surface and CMM data points</td>
</tr>
</tbody>
</table>
List of Tables

6.1 Elapsed time for each execution at various numbers of control points 73
6.2 RMS between two calculation methods of Bernstein polynomials . . . 75
6.3 Elapsed time for each execution at various numbers of control points 75
Chapter 1
Introduction

1.1 Background and Motivation

The laser forming process [45, 46, 47] is a manufacturing technology by which a plate is incrementally deformed to the desired shape by selectively heating it with a high power laser beam. It involves determining a trajectory for laser heating by assessing geometric and mechanical properties of the plate. However, because an exact computation of the heating paths for getting the desired deformation of the plate is too expensive, an iteration scheme is adopted; in this scheme, the shape of the plate which is produced through one phase during the line heating process is assessed and then is fed to an algorithm that computes the next phase’s line heating trajectory. The data for assessment is measured with a laser scanner. Given that the field of view of the scanner available to use is too small to capture the whole shape of the plate, multiple views need to be taken to achieve total coverage of the surface. The scanner is mounted on a robot arm whose range of mobility is large enough to digitize the plate. The data of each view must be registered in a common coordinate system to get an appropriate geometric representation of the plate.

The reconstructed surface is represented in terms of Non-Uniform Rational B-Spline (NURBS) surfaces or a special case of NURBS called Bézier surfaces [21, 31]. NURBS surface patches are the most widely used representation method in the CAD/CAM field because of their generality, geometric and computational properties and compatibility with international standards such as IGES (Initial Graphics Exchange Specification) (see [36]) and STEP (Standard for the Exchange of Product Models) (see [30]).

The lofting procedure, one of the existing methods for surface construction can be used to reconstruct the surface from the digitized data points. However, the technique requires to construct lines from the data points that will become iso-parametric lines of the resulting surface. The lines created from the data points in general may not be smooth so that the surface reconstructed from them may have inappropriate geometric properties which are revealed during shape interrogation. In order to overcome this problem of the lofting procedure, a novel method which contains the lofting as a special case has been developed and used for surface reconstruction as well as surface
1.2 Research Objectives

The main goal of this research is to develop an automated system and related algorithms which acquire image data from a plate, analyze them and create a suitable mathematical model. The effort is then extended to develop a novel method for surface construction which not only can be applied to two different areas such as reverse engineering and design but also produces better results than existing methods.

1.3 Literature Review

Many algorithms of surface reconstruction for reverse engineering have been proposed in recent years. Tiller [40] presents a method for lofting or skinning an array of planar NURBS curves of arbitrary shape. More discussion on it is made in the textbook by Piegl and Tiller [33]. Sapidis and Besl [38] have extended the region-growing technique developed by Besl [7] to automate the construction of surface patches from range data. Eck and Hoppe [13] present a procedure for reconstructing a tensor product B-spline surface from a set of scanned 3D points. They define a surface as a network of B-spline patches which is automatically constructed along with a parameterization of the data points over the patches. This scheme directly reconstructs a surface of arbitrary topological type. A new approach for reconstructing a smooth surface from a set of scattered points in 3D space has been proposed by Gregorski et al. [20]. The given point set is decomposed into a data structure called a strip tree which is used to fit a set of least squares quadratic surfaces to the data points. The quadratic surfaces are blended to form a B-spline surface. Lee et al. [24] present a hierarchical scheme to interpolate or approximate scattered 3D data points which are represented in a height function form.

Many efforts have been made to automate data acquisition and surface reconstruction. Milroy et al. [28] developed an automated scanning system using an intermediate data model that consists of three orthogonal cross-sections. Xi et al. [44] present CAD-based path planning for 3D laser scanning which provides an appropriate basis for automating data acquisition. A comprehensive survey on the automated visual inspection systems and techniques published from 1988 to 1993 was performed by Newman and Jain [29]. On the other hand, Várady et al. [42] reviews the process of reverse engineering of shapes. Various aspects of reverse engineering are discussed and the most important steps and strategies are outlined and compared.

1.4 Thesis Organization

The remainder of this thesis is organized as follows. Chapter 2 starts with discussion on basic differential geometry of curves and surfaces and continues brief review of Bézier and B-spline representations of curves and
surfaces. It also reviews useful algorithms for Bézier and B-spline representations and surface generation such as lofting and Coons surfaces.

Chapter 3 develops a novel approach for surface approximation. Bézier and B-spline surface approximation using the concept of a curve on a surface is developed in detail and a discussion of the least squares method is included. Complexity analysis of the new method is presented.

Chapter 4 describes the components of the system and how they are configured. Communication protocols such as TCP/IP and RS-232C and their applications in the system are presented.

In Chapter 5, initialization of the system is described and operations and algorithms of each step performed by the system are explained. The analysis of complexity of the flow of the system is presented.

Several practical examples and numerical complexity analysis of the new method are presented in Chapter 6. The thesis concludes with Chapter 7 which also includes recommendations for future work.
Chapter 2

Theoretical Background

2.1 Introduction

Basic differential geometry of curves and surfaces is reviewed. Bézier and B-spline curve and surface representation and useful algorithms for Bézier and B-spline representations are discussed. Two surface generation methods, Coons surface blending and lofting, are presented in detail.

2.2 Review of Differential Geometry of Curves and Surfaces

This section reviews the theory of curves and surfaces. It presents various fundamental formulae and properties such as derivatives, curvatures and Frenet-Serret formulae which provide the background necessary in the discussions of later chapters. The information in this section is based on classical texts on differential geometry and geometric modeling such as those by do Carmo [12], Faux and Pratt [16], Hoschek and Lasser [21] and Patrikalakis and Maekawa [31].

2.2.1 Basic Theory of Curves

A curve is defined as an image of an open, or closed, or half-open finite or infinite interval $I$ mapped into $\mathbb{R}^2$ or $\mathbb{R}^3$ through a differentiable function. If the image is included in $\mathbb{R}^2$, it is a plane curve, while if it is in $\mathbb{R}^3$, it is a space curve. A point on the curve is given by $\mathbf{r}(t) = (x(t), y(t), z(t))$ where $t \in I$. The variable $t$ is the parameter of the curve. Namely, the mapping function $\mathbf{r}(t)$ is a curve represented in a parametric way.

Definition [12]
A parametrized differentiable curve $\mathbf{r} : I \rightarrow \mathbb{R}^3$ is said to be regular if $\dot{\mathbf{r}}(t) \neq 0$ for all $t \in I$.

Throughout this thesis, only regular parametrized differentiable curves are considered.
Arc length is defined as the length along a curve, \( r(t) \) between points \( r(t_0) \) and \( r(t) \) and (provided the function \( t \in [t_0, t] \rightarrow r(t) \) is one-to-one almost everywhere [31]) can be represented as:

\[
s(t) = \int_{t_0}^{t} \sqrt{\dot{x}^2(t) + \dot{y}^2(t) + \dot{z}^2(t)} dt
\]

(2.1)

\[
= \int_{t_0}^{t} \sqrt{\dot{r}(t) \cdot \dot{r}(t)} dt.
\]

Here the dot \( ' \) denotes the differentiation with respect to parameter \( t \). In addition the prime \( ' \) is used to denote the differentiation with respect to arc length \( s \). These notations are used throughout this thesis. Some useful formulae for the derivatives of arc length \( s \) with respect to parameter \( t \) and vice versa are as follows.

\[
\dot{s} = \frac{ds}{dt} = |\dot{r}| = \sqrt{\dot{r} \cdot \dot{r}}
\]

(2.2)

\[
\ddot{s} = \frac{d\dot{s}}{dt} = \frac{\dot{r} \cdot \ddot{r}}{\sqrt{\dot{r} \cdot \dot{r}}}
\]

\[
\dot{t}' = \frac{dt}{ds} = \frac{1}{|\dot{r}|} = \frac{1}{\sqrt{\dot{r} \cdot \dot{r}}}
\]

\[
\dot{t}'' = \frac{d\dot{t}'}{ds} = -\frac{\dot{r} \cdot \ddot{r}}{(\dot{r} \cdot \dot{r})^2}.
\]

The vector \( \frac{dr}{dt} = \dot{r} \) is called the tangent vector at a point on a curve \( r(t) \) and the unit tangent vector is defined by dividing the tangent vector by its magnitude as follows

\[
t = \frac{r}{|r|} = \frac{dr}{ds} = \frac{dr}{dt} = r'.
\]

(2.3)

If \( r(s) \) is a curve parametrized by arc length, \( r'(s) \) is a unit vector yielding

\[
r' \cdot r' = 1.
\]

(2.4)

Differentiating (2.4), we obtain

\[
r' \cdot r'' = 0
\]

(2.5)

implying that \( r'' \) is perpendicular to \( r' \), provided it is not a null vector.

The unit vector

\[
n = \frac{r''(s)}{|r''(s)|} = \frac{t'(s)}{|t'(s)|}
\]

(2.6)

is called the unit principal normal vector at \( s \).

The unit tangent vector \( t(s) \) and the unit normal vector \( n(s) \) form the osculating plane at \( s \). It is also well known that the plane through three consecutive points of
As in Figure 2-1: Derivation of the normal vector

the curve defines the osculating plane.

The curvature $\kappa$, which measures the deviation of a curve from a straight line, is defined by

$$\kappa = \left| r''(s) \right| = \frac{1}{\rho}$$

(2.7)

where the reciprocal of the curvature $\rho$, when $\kappa > 0$, is called the radius of curvature. Geometrically, the radius of curvature of a curve at a point is equivalent to the radius of the circle whose first and second derivatives are equal to those of the curve at a point (Hoschek [21]). The unit binormal vector $b$ is defined so that $(t, n, b)$ form a right handed screw i.e.: 

$$b = t \times n \quad t = n \times b \quad n = b \times t.$$  

(2.8)
The plane defined by \( \mathbf{n} \) and \( \mathbf{b} \) is the *normal plane* and the plane defined by \( \mathbf{b} \) and \( \mathbf{t} \) is the *rectifying plane*. The scalar \( \tau \) is defined by

\[
\mathbf{b}'(s) = -\tau \mathbf{n}
\]  

(2.9)

and is called the *torsion* of the curve at \( s \). The torsion is a measure of the deviation of a curve from being planar. It vanishes for planar curves. Since \( \mathbf{n} = \mathbf{b} \times \mathbf{t} \), the derivative of \( \mathbf{n} \) with respect to \( s \) becomes

\[
\mathbf{n}' = \mathbf{b}' \times \mathbf{t} + \mathbf{b} \times \mathbf{t}' = \tau \mathbf{b} - \kappa \mathbf{t}.
\]  

(2.10)

The set of differential equations

\[
\begin{align*}
\mathbf{t}' &= \kappa \mathbf{n} \\
\mathbf{n}' &= -\kappa \mathbf{t} + \tau \mathbf{b} \\
\mathbf{b}' &= -\tau \mathbf{n}
\end{align*}
\]  

(2.11)

is known as Frenet-Serret formulae.

The curvature \( \kappa \) and the torsion \( \tau \) are independent of the parametrization and describe completely the local behavior of the curve.

### 2.2.2 Basic Theory of Surfaces

A general parametric surface is defined as a vector-valued mapping from a two-dimensional parametric \( uv \) space to three dimensional space \( \mathbb{R}^3 \).

\[
\mathbf{r}(u,v) = \left[ x(u,v), \ y(u,v), \ z(u,v) \right]^T
\]  

(2.12)

where \( u_1 \leq u \leq u_2 \) and \( v_1 \leq v \leq v_2 \).

Roughly speaking, if a surface is smooth enough so that the usual notations of calculus can be extended to it, then it is called a regular surface. Mathematically, the definition of a regular surface is given by [12]

**Definition**

A subset \( S \subset \mathbb{R}^3 \) is a regular surface if, for each \( P \in S \), there exists a neighborhood \( V \) in \( \mathbb{R}^3 \) and a map \( \mathbf{r} : \mathbb{U} \to V \cap S \) of an open set \( \mathbb{U} \subset \mathbb{R}^2 \) onto \( V \cap S \subset \mathbb{R}^3 \) such that

- \( \mathbf{r} \) is differentiable. This means that the functions \( x(u,v) \), \( y(u,v) \) and \( z(u,v) \) have continuous partial derivatives of all orders in \( \mathbb{U} \).

- \( \mathbf{r} \) is a homeomorphism, which means that \( \mathbf{r} \) has an inverse \( \mathbf{r}^{-1} : V \cap S \to \mathbb{U} \) which is continuous.

- For each \((u_q, v_q) \in \mathbb{U}\), the differential \( d\mathbf{r}(u_q, v_q) : \mathbb{R}^2 \to \mathbb{R}^3 \) is one to one.

Additional geometric entities can be derived by introducing a curve on a surface represented as \( \mathbf{r}(u,v) \) with \( u = u(t) \) and \( v = v(t) \). Differentiating \( \mathbf{r}(t) \) with respect to
parameter \( t \) using the chain rule yields the tangent vector of the curve on the surface, which is also tangent to the surface.

\[
\mathbf{r}(t) = \mathbf{r}_u \dot{u} + \mathbf{r}_v \dot{v}. \tag{2.13}
\]

Provided \( \mathbf{r}_u \times \mathbf{r}_v \neq \mathbf{0} \), the two vectors \( \mathbf{r}_u \) and \( \mathbf{r}_v \) form a plane called the tangent plane to which the normal to the surface is orthogonal. The unit surface normal vector is given by

\[
\mathbf{N} = \frac{\mathbf{r}_u \times \mathbf{r}_v}{|\mathbf{r}_u \times \mathbf{r}_v|}. \tag{2.14}
\]

This definition of the normal vector is valid as long as \( \mathbf{r}_u \times \mathbf{r}_v \neq \mathbf{0} \).

**Definition** [12, 31]

A regular point \( \mathbf{p} \) on a parametric surface is defined as a point where \( \mathbf{r}_u \times \mathbf{r}_v \neq \mathbf{0} \). A point which is not a regular point is called a singular point.

The definition of arc length of a parametric curve is extended to a parametric curve \( u = u(t) \) and \( v = v(t) \) on a parametric surface \( \mathbf{r} = \mathbf{r}(u(t), v(t)) \) such that

\[
ds = \left| \frac{d\mathbf{r}}{dt} \right| dt = \left| \mathbf{r}_u \frac{du}{dt} + \mathbf{r}_v \frac{dv}{dt} \right| dt \tag{2.15}
\]

\[
= \sqrt{(\mathbf{r}_u \ddot{u} + \mathbf{r}_v \ddot{v}) \cdot (\mathbf{r}_u \ddot{u} + \mathbf{r}_v \ddot{v})} dt
\]

\[
= \sqrt{E \dot{u}^2 + 2F \dot{u} \dot{v} + G \dot{v}^2} dt
\]

where \( E = \mathbf{r}_u \cdot \mathbf{r}_u, \ F = \mathbf{r}_u \cdot \mathbf{r}_v \) and \( G = \mathbf{r}_v \cdot \mathbf{r}_v \). The first fundamental form is defined
$\mathbf{r}(t) = \mathbf{r}_u u + \mathbf{r}_v v$

Figure 2-4: Tangent plane and normal vector

as

$$I = ds^2 = d\mathbf{r} \cdot d\mathbf{r} = Edu^2 + 2Fdudv + Gdv^2$$

and $E$, $F$ and $G$ are called the first fundamental form coefficients. The first fundamental form provides a metric on a surface which is a fundamental concept for intrinsic geometry of the surface. The fact that the surface is regular guarantees that $EG - F^2$ is always positive and $E > 0, G > 0$, meaning $I$ is positive definite. The differential area of the surface can be approximated using the first fundamental form coefficients as $dA = \sqrt{EG - F^2} du dv$

The curvatures of the surface are quantified by introducing the second fundamental form. First, let us consider the curvature vector of a curve on the surface. The curvature vector $\mathbf{k}$ is given by

$$\mathbf{k} = \frac{d\mathbf{t}}{ds} = \mathbf{k}_n + \mathbf{k}_g = \kappa_n \mathbf{N} + \mathbf{k}_g$$

where $\mathbf{k}_n$ is the normal curvature vector and $\mathbf{k}_g$ is the geodesic curvature vector. $\kappa_n$ is called the normal curvature of the surface at $P$ in the direction $\mathbf{t}$. In other words, $\kappa_n$ is the magnitude of the projection of $\mathbf{k}$ onto the surface normal at $P$. The second fundamental form is given by

$$II = Ldu^2 + 2Mdudv + Ndvd^2$$

where $L$, $M$ and $N$ are called the second fundamental form coefficients and defined by

$$L = \mathbf{r}_{uu} \cdot \mathbf{N} \quad M = \mathbf{r}_{uv} \cdot \mathbf{N} \quad N = \mathbf{r}_{vv} \cdot \mathbf{N}.$$
Figure 2-5: Definition of positive normal curvature (a) $\kappa \mathbf{n} \cdot \mathbf{N} = \kappa_n$; (b) $\kappa \mathbf{n} \cdot \mathbf{N} = -\kappa_n$

The normal curvature $\kappa_n$ can be derived from differentiation of $\mathbf{N} \cdot \mathbf{t} = 0$ and expressed in terms of the first and second fundamental form coefficients as follows:

$$\frac{d\mathbf{t}}{ds} \cdot \mathbf{N} + \mathbf{t} \cdot \frac{d\mathbf{N}}{ds} = 0. \quad (2.20)$$

$$\kappa_n = \frac{d\mathbf{t}}{ds} \cdot \mathbf{N} = -\mathbf{t} \cdot \frac{d\mathbf{N}}{ds} = -\frac{d\mathbf{r}}{ds} \cdot \frac{d\mathbf{N}}{ds}$$

$$= \frac{dr \cdot dN}{\mathbf{r} \cdot d\mathbf{r}} = \frac{II}{I}$$

$$= \frac{L + 2M\lambda + N\lambda^2}{E + 2F\lambda + G\lambda^2} \quad (2.21)$$

where $\lambda = \frac{dv}{du}$. Sometimes the positive normal curvature is defined in the opposite direction, i.e. the center of curvature of the normal section curve is on the opposite side of the surface normal. In such cases, Equation (2.21) becomes

$$\kappa_n = -\frac{II}{I}. \quad (2.22)$$

The normal curvature $\kappa_n$ is not unique at a point $P$ on the surface. It depends on the direction of the tangent line to a curve passing through the point. $\kappa_n$ has at most two extreme values (maximum and minimum) depending on the direction, which are called the principal curvatures of the surface and are designated as $\kappa_1$ and $\kappa_2$, respectively. The extreme values of $\kappa_n$ occur when $\frac{d\kappa_n}{d\lambda} = 0$, resulting in the
following relations.

\[ \kappa_n^2 - 2H\kappa_n + K = 0 \]  
(2.23)

where

\[ K = \frac{LN - M^2}{EG - F^2} \]  
(2.24)

\[ H = \frac{EN + GL - 2FM}{2(EG - F^2)} \]

and \( K \) and \( H \) are called Gaussian curvature and mean curvature, respectively. Upon solving Equation (2.23) for the extreme values of curvature, we have

\[ \kappa_{\text{max}} = H + \sqrt{H^2 - K} \]  
(2.25)

\[ \kappa_{\text{min}} = H - \sqrt{H^2 - K} \]

and it can be easily shown that

\[ K = \kappa_{\text{max}}\kappa_{\text{min}} \]  
(2.26)

\[ H = \frac{1}{2}(\kappa_{\text{max}} + \kappa_{\text{min}}). \]

2.3 Bézier Curve and Surface Representation

This section reviews the representation of curves and surfaces in Bézier form and discusses various properties of Bernstein polynomials and the Bézier representation. Additionally, algorithms for degree elevation and subdivision are summarized. The descriptions are based on textbooks by Piegl and Tiller [33], Hoschek and Lasser [21], Rogers and Adams [37], Faux and Pratt [16], Farin [14] and Patrikalakis and Maekawa [31].

2.3.1 Bernstein Polynomials

Bernstein polynomials are defined by

\[ B_{i,n}(t) = \frac{n!}{i!(n-i)!}(1-t)^{n-i}t^i, \quad i = 0, \ldots, n \]  
(2.27)

and have several properties of interest. The property of non-negativity states that Bernstein polynomials are always non-negative in the interval \([0, 1]\):

\[ B_{i,n}(t) \geq 0, \quad 0 \leq t \leq 1 \text{ for all } i \text{ and } n. \]  
(2.28)
The partition of unity property can be derived from the binomial theorem such that
\[ \sum_{i=0}^{n} B_{i,n}(t) = (1 - t + t)^n = 1 \quad \text{for} \quad 0 \leq t \leq 1. \]  
(2.29)

The Bernstein polynomials exhibit a symmetry property:
\[ B_{i,n}(t) = B_{n-i,n}(1-t) \quad 0 \leq t \leq 1. \]  
(2.30)

From the definition of Bernstein polynomials, the following recurrence relation can be shown
\[ B_{i,n}(t) = (1 - t)B_{i,n-1}(t) + tB_{i-1,n-1}(t). \]  
(2.31)

The derivatives are given by
\[ \frac{d^p}{dt^p} B_{i,n}(t) = \frac{n!}{(n-p)!} \Delta^p B_{i,n}(t) \]  
(2.32)

where
\[ \Delta^0 B_{i,n}(t) = B_{i,n}(t), \quad \Delta^p B_{i,n}(t) = \Delta^{p-1} [B_{i-1,n-1}(t) - B_{i,n-1}(t)]. \]  
(2.33)

The linear precision property states that the monomial \( t \) can be expressed as the weighted sum of Bernstein polynomials of degree \( n \) with coefficients evenly spaced on the interval \([0, 1]\) such that
\[ t = \sum_{i=0}^{n} \frac{i}{n} B_{i,n}(t). \]  
(2.34)

2.3.2 Bézier Curves and Surfaces

Bézier Curves

Using the Bernstein polynomials as a basis, a Bézier curve of degree \( n \) is defined as a parametric curve
\[ \mathbf{r}(t) = \sum_{i=0}^{n} \mathbf{b}_i B_{i,n}(t), \quad 0 \leq t \leq 1 \]  
(2.35)

with \( \mathbf{b}_i \in \mathbb{R}^d, \quad d = 2, 3 \). The points \( \mathbf{b}_i \) are called control points or Bézier points and are defined as vectors in \( \mathbb{R}^2 \) or \( \mathbb{R}^3 \). They determine the shape of the curve and lines drawn between consecutive control points of the curve form the control polygon. A cubic Bézier curve together with its control polygon is shown in Figure 2-6.

Bézier curves have the following properties.

The geometric invariance property states that the shape of the Bézier curve doesn’t change under translation and rotation of its control points, which is assured by the
partition of unity property of Bernstein polynomials.

At its end points, the Bézier curve exhibits several important geometric properties. The first and last control points are the end point of the curve. In other words, \( b_0 = r(0) \) and \( b_n = r(1) \). Also the curve is tangent to the control polygon at the end points, which can be easily shown by taking the first derivative of the Bézier curve.

The first derivative of the Bézier curve is very important in the study of intersection and other interrogation problems such as singularities and inflection points. It is another Bézier curve called hodograph, whose degree is lower than the original curve by one and has control points \( \Delta b_i = n(b_{i+1} - b_i), i = 0, \ldots, n - 1 \).

The convex hull property states that the entire Bézier curve is enclosed within the convex hull of its control points.

The variation diminishing property states that the number of intersections of a straight line with a planar Bézier curve is no greater than the number of intersections of the line with its control polygon. The same property holds true for a plane with a space Bézier curve.

Due to the symmetry property of Bernstein polynomials, the Bézier curve satisfies
the following property:

\[
\sum_{i=0}^{n} b_i B_{i,n}(t) = \sum_{i=0}^{n} \mathbf{b}_i^* B_{i,n}(1 - t)
\]  

(2.36)

where \( \mathbf{b}_i^* = \mathbf{b}_i \), \( i = 0, 1, \ldots, n \).

**Bézier Curve Evaluation**

Two methods are available for evaluating Bézier curves. The first method uses the definition of a Bézier curve directly by plugging a specific parameter value \( t_0 \). However, such method is not as accurate as the second method, which uses the *de Casteljau algorithm*, given by the recursion:

\[
\mathbf{b}_i^k(t_0) = (1 - t_0)\mathbf{b}_{i-1}^k + t_0 \mathbf{b}_i^{k-1}
\]

(2.37)

for \( k = 1, 2, \ldots, n \) and \( i = k, \ldots, n \). This formula is applied recursively to obtain the new control points. This algorithm is illustrated graphically in Figure 2-7 and has the following properties.
- The vectors $b_i^0$ are the original control points of the curve.
- The position vector of the curve at parameter $t_0$ is $b_i^n$.

**Bézier Surface Patches**

A Bézier surface patch is a tensor product surface which is based on a Bézier curve scheme. Basis functions are represented as the product of two univariate functions or Bernstein polynomials of $u$ and $v$. Control points are arranged topologically in a bidirectional $m \times n$ net. A Bézier surface is defined by

$$
\mathbf{r}(u, v) = \sum_{i=0}^{m} \sum_{j=0}^{n} b_{ij} B_{i,m}(u) B_{j,n}(v), \ 0 \leq u, v \leq 1. \quad (2.38)
$$

Here the set of straight line segments drawn between consecutive control points $b_{ij}$

![Bézier Surface Patches](image)

Figure 2-8: A biquadratic Bézier surface

is called the *control net* and $B_{i,m}(u) B_{j,n}(v)$ are the bivariate basis functions. An example of a bi-quadratic Bézier surface with its control net is given in Figure 2-8.

A Bézier surface patch exhibits most of the properties of Bézier curves since it is based on the Bézier curve scheme.

- Bézier surface patches inherit the convex hull property of Bézier curves.
- The corners of the Bézier surface patch coincide with the corner points of the control net, and the derivatives in the $u$ and $v$ directions at the corners are tangent to the net.
The boundary isoparametric curves have the same control points as the corresponding boundary points on the control net. However, the variation diminishing property does not apply to Bézier surface patches.

### 2.3.3 Degree Elevation

Degree elevation is a fundamental algorithm useful in the Bézier curve and surface based geometric design. A Bézier curve of degree \( n \) can be expressed as a Bézier curve of higher degree without changing its shape.

For a single stage of degree elevation, the basis functions of degree \( n \) can be expressed in terms of basis functions of degree \( n + 1 \) as:

\[
B_{i,n}(t) = \left(1 - \frac{i}{n+1}\right)B_{i,n+1}(t) + \frac{i+1}{n+1}B_{i+1,n+1}(t), \quad i = 0, 1, \cdots, n. \tag{2.39}
\]

From expression (2.39), the new control points for degree \( n + 1 \) are obtained in terms of those of degree \( n \) as:

\[
P_{k,n+1} = \frac{k}{n+1}P_{k-1,n} + \left(1 - \frac{k}{n+1}\right)P_{k,n} \tag{2.40}
\]

for \( k = 0, 1, 2, \cdots, n + 1 \) and \( P_{-1,n} = P_{n+1,n} = 0 \). The expression (2.39) can be generalized to the case of degree elevation from degree \( n \) to degree \( n + r \) as:

\[
B_{k,n}(t) = \sum_{j=k}^{k+r} \binom{n}{k} \binom{r}{j-k} \binom{n+r}{j} B_{j,n+r}(t), \quad k = 0, 1, \cdots, n \tag{2.41}
\]

which can be derived by multiplication of \( B_{k,n}(t) \) by the expression of \( \{(1 - t) + t\}^r \)

or by induction on (2.39). The new coefficients of degree \( n + r \) are obtained by
substituting (2.41) into the definition of $B_{k,n}(t)$, i.e. $B_{k,n}(t) = \binom{n}{k} (1 - t)^{n-k} t^k$, $k = 0, 1, \ldots, n$ yielding

$$P_{k,n+r} = \sum_{j=\max(0,k-r)}^{\min(n,k)} \binom{r}{k-j} \binom{n}{j} \binom{n+r}{k} P_{j,n} \quad (2.42)$$

for $k = 0, 1, \ldots, n+r$. Since the degree of the Bézier curve determines the number of control points, degree elevation can add flexibility in curve design by increasing the number of control points of the Bézier curve.

2.3.4 Subdivision

The de Casteljau algorithm is also used to subdivide a Bézier curve without modifying its shape. The graphical illustration of the algorithm is given in Figure 2-7. Here the original curve is subdivided into two (independent) curves with control points $(b_0, b_1, \ldots, b_0)$ and $(b_n, b_{n-1}, \ldots, b_0)$, respectively. For surface subdivision, the algorithm is applied to the Bézier surface in $u$ direction and $v$ direction consecutively.

2.4 B-spline Curve and Surface Representation

Two characteristics limit the flexibility of the Bézier representation. First, the number of control points fixes the degree of the Bézier curve. Therefore, to increase the complexity of the shape of the Bézier curve by adding control points requires increasing the degree of the curve or satisfying the continuity conditions between consecutive segments of a composite curve. Second, the global nature of Bernstein polynomials makes a change in one control point propagate throughout the whole curve, meaning design of a specific section or local control is very difficult. These drawbacks can be avoided with the introduction of a new type of basis function, or B-spline basis.

2.4.1 B-spline Basis Function

A B-spline basis contains the Bernstein polynomial basis as a special case and in general demonstrates the non-global behavior. An order $k$ B-spline is formed by joining several pieces of polynomial of degree $k-1$ with $C^{k-2}$ continuity at the break points. The break points are given as a set of non-descending real values $t_i$, $t_0 \leq t_1 \leq \cdots \leq t_m$ which form the knot vector, $\mathbf{T}$. They also determine the parametrization of the basis functions

$$\mathbf{T} = (t_0, t_1, \cdots, t_m). \quad (2.43)$$
Given a knot vector $T$, the B-spline basis function is defined by recursion:

$$N_{i,1}(t) = \begin{cases} 
1 & \text{for } t_i \leq t < t_{i+1} \\
0 & \text{otherwise}
\end{cases} \quad (2.44)$$

and

$$N_{i,k}(t) = \frac{(t-t_i)}{t_{i+k-1}-t_i}N_{i,k-1}(t) + \frac{t_{i+k}-t}{t_{i+k}-t_{i+1}}N_{i+1,k-1}(t), \quad (2.45)$$

$$k \geq 1, \ i = 0, 1, \ldots, n.$$ 

The B-spline basis functions have the following properties:

- **Positivity:** $N_{i,k}(t) > 0$ for $t_i < t < t_{i+k}$.
- **Local Support:** $N_{i,k}(t) = 0$ for $t_0 \leq t \leq t_i$, and $t_{i+k} \leq t \leq t_{n+k}$.
- **Partition of Unity:** $\sum_{i=0}^{n} N_{i,k}(t) = 1$ for $t \in [t_0, t_m]$.
- **Continuity:** $N_{i,k}(t)$ has $C^{k-2}$ continuity at each simple knot.

### 2.4.2 B-spline Curves and Surfaces

**B-spline Curves**

A B-spline curve is defined as a linear combination of control points $P_i$ and B-spline basis functions $N_{i,k}(t)$ given by

$$\mathbf{r}(t) = \sum_{i=0}^{n} \mathbf{P}_i N_{i,k}(t), \quad n \geq k-1, \quad t \in [t_{k-1}, t_{n+1}]. \quad (2.46)$$

The basis function $N_{i,k}(t)$ is defined over a knot vector

$$T = (t_0, t_1, \ldots, t_{k-1}, t_k, t_{k+1}, \ldots, t_n, t_{n+1}, t_{n+k}). \quad (2.47)$$

Each knot span $t_i \leq t \leq t_{i+1}$ is mapped onto a polynomial curve between two successive joints $\mathbf{r}(t_i)$ and $\mathbf{r}(t_{i+1})$.

A B-spline curve has several properties. The **geometric invariance property** states that the shape of the B-spline curve does not change under translation and rotation, which is assured by the **partition of unity** for B-spline basis functions. The **convex hull property** for the B-spline curve, which is the same concept as that for the Bézier curve, applies locally so that a span lies within the convex hull of the control points that affect it. The **local support property**, which is an advantage over the Bézier representation, is that a single span of a B-spline curve is controlled only by $k$ control points and that any control point affects $k$ spans. In other words, changing $\mathbf{P}_i$ affects the curve in the parametric range $t_i < t < t_{i+k}$ and the curve at a point $t$ where $t_r < t < t_{r+1}$ is determined completely by the control points $\mathbf{P}_{r-(k-1)} \cdots \mathbf{P}_r$. The
variation diminishing property, which is the same as that for Bézier curves, also holds for B-spline curves.

The multiplicity of a knot is directly related to the continuity of the B-spline curve in such a way that increasing the multiplicity of a knot reduces the continuity of the curve at that knot. Specifically, the curve is $k - p - 1$ times continuously differentiable at a knot with multiplicity $p(\leq k)$, meaning it has $C^{(k-p-1)}$ continuity. This implies that the control polygon will coincide with the curve at a knot of multiplicity $k - 1$ and a knot with multiplicity $k$ indicates $C^{-1}$ continuity or a discontinuous curve.

Unlike Bézier curves, the end points of B-spline curves may not coincide with the end vertices of the control polygons. However, repeating the knots at the end $k$ times will force the end points to coincide with the end vertices of the control polygon. The knot vector for this case will be

$$T = (t_0, t_1, \ldots, t_{k-1}, t_k, t_{k+1}, \ldots, t_{n-1}, t_n, t_{n+1}, \ldots, t_{n+k})$$

where the first $k$ elements and the last $k$ elements are identical, i.e. $t_0 = t_1 = \cdots t_{k-1}$ and $t_{n+1} = t_{n+2} = \cdots = t_{n+k}$.

The B-spline curve with a knot vector $T$ as in (2.48) is tangent to the control polygon at its end points. This can be observed from the first derivative of the
B-spline curve given by

$$\hat{r}(t) = \sum_{i=1}^{n} (k-1) \left( \frac{P_i - P_{i-1}}{t_{i+k-1} - t_i} \right) N_{i,k-1}(t) \quad (2.49)$$

$$\hat{r}(0) = \frac{k-1}{t_k - t_1} (P_1 - P_0)$$

$$\hat{r}(1) = \frac{k-1}{t_{n+k-1} - t_n} (P_n - P_{n-1}).$$

**Evaluation**

When evaluating a B-spline curve at a specific parameter value $t_0$, the Cox-de Boor algorithm, which is a generalization of the de Casteljau algorithm, can be used. The repeated substitution of the recursive definition of the B-spline basis function and re-indexing yields the Cox-de Boor algorithm.

$$r(t) = \sum_{i=1}^{n+j} P_i^j N_{i,k-j}(t) \quad (2.50)$$

$$P_i^j = (1 - \alpha_i^j) P_i^{j-1} + \alpha_i^j P_i^{j-1}$$

with

$$\alpha_i^j = \frac{t - t_i}{t_{i+k-j} - t_i}, \quad P_j^0 = P_j. \quad (2.51)$$

For $j = k - 1$, the B-spline basis function reduces to $N_{i,1}$ for $t \in [t_i, t_{i+1}]$ and $P^{k-1}_i$ coincides with the curve $r(t) = P^{k-1}_i$. The graphical illustration of the Cox-de Boor algorithm for a cubic B-spline curve is given in Figure 2-11.

**B-spline Surface Patches**

A B-spline surface patch is a tensor product surface defined using the B-spline basis functions and a topologically rectangular net of control points $P_{ij}$, $(0 \leq i \leq m, \ 0 \leq j \leq n)$ and two knot vectors $U = (u_0, u_1, \ldots, u_{m+k})$ and $V = (v_0, v_1, \ldots, v_{n+l})$. The mathematical representation is given by

$$r(u, v) = \sum_{i=0}^{m} \sum_{j=0}^{n} P_{ij} N_{i,k}(u) N_{j,l}(v). \quad (2.52)$$

Some of the properties of B-spline curves also extend to B-spline surface patches such as:

- Geometric invariance property
Figure 2-11: The de Boor algorithm
• End point geometric property
• Convex hull property

No variation diminishing property, however, is known for B-spline surface patches.

2.4.3 Knot Insertion

Increasing the number of control points by elevating the degree of a Bézier curve is a useful way to give more flexibility in shape control of the Bézier curve without altering its original shape. The same idea is applicable to B-spline curves. Inserting a new knot into a given knot vector increases the number of control points without modifying the B-spline curve either geometrically or parametrically. Knot insertion is useful in subdivision of curves and surfaces and in evaluation of points and derivatives. The following description of knot insertion is based on the book by Piegl and Tiller [33].

Let \( r(t) \) be a B-spline curve defined on \( \mathbf{U} = \{u_0, \ldots, u_m\} \).

\[
r(t) = \sum_{i=0}^{n} N_{i,p}(t) P_i.
\] (2.53)

Let \( \bar{u} \in [u_k, u_{k+1}] \) and insert \( \bar{u} \) into \( \mathbf{U} \) to yield the new knot vector.

\[
\mathbf{U} = \{\bar{u}_0 = u_0, \ldots, \bar{u}_k = u_k, \bar{u}_{k+1} = \bar{u}, \bar{u}_{k+2} = u_{k+1}, \ldots, \bar{u}_{m+1} = u_m\}.
\] (2.54)

The curve has a representation on knot vector \( \mathbf{U} \) of the form

\[
r(t) = \sum_{i=0}^{n+1} N_{i,p}(t) Q_i.
\] (2.55)

where \( N_{i,p}(t) \) is the \( p \)th-degree basis function on \( \mathbf{U} \). The curve must be the same before and after insertion, meaning the following equation must hold.

\[
r(t) = \sum_{i=0}^{n} N_{i,p}(t) P_i = \sum_{i=0}^{n+1} N_{i,p}(t) Q_i.
\] (2.56)

Substituting \( n + 2 \) suitable values of \( t \) yields a system of linear equations and the new \( Q_i \) can be calculated by solving the system. However, a more efficient solution can be obtained by using the local support property of B-spline basis functions. The property states that in any given knot span, \([u_j, u_{j+1}]\) at most \( p + 1 \) of the \( N_{i,p}(t) \), namely the functions \( N_{j-p,p}(t), \ldots, N_{j,p}(t) \) are nonzero. From this property and the fact that \( \bar{u} \in [u_k, u_{k+1}] \), the following relation can be derived from Equation (2.56).

\[
\sum_{i=k-p}^{k} N_{i,p}(t) P_i = \sum_{i=k-p}^{k+1} N_{i,p}(t) Q_i.
\] (2.57)
for all \( t \in [u_k, u_{k+1}) \) and

\[
N_{i,p}(t) = \overline{N}_{i,p}(t), \quad i = 0, \ldots, k - p - 1 \quad (2.58)
\]
\[
N_{i,p}(t) = \overline{N}_{i+1,p}(t), \quad i = k + 1, \ldots, n. \quad (2.59)
\]

Equations (2.57) and (2.59) imply that

\[
P_i = Q_i, \quad i = 0, \ldots, k - p - 1 \quad (2.60)
\]
\[
P_i = Q_{i+1}, \quad i = k + 1, \ldots, n. \quad (2.61)
\]

The \( N_{i,p}(t) \) for \( i = k - p, \ldots, k \) can be expressed in terms of \( \overline{N}_{i,p}(t) \) when \( i = k - p, \ldots, k + 1 \).

\[
N_{i,p}(t) = \frac{\bar{u} - u_i}{u_{i+p+1} - u_i} \overline{N}_{i,p}(t) + \frac{u_{i+p+2} - \bar{u}}{u_{i+p+2} - u_{i+1}} \overline{N}_{i+1,p}(t). \quad (2.62)
\]

Substituting (2.62) into (2.56) and comparing the coefficients of both sides yields

\[
Q_i = \alpha_i P_i + (1 - \alpha_i) P_{i-1} \quad (2.63)
\]

where,

\[
\alpha_i = \begin{cases} 
1 & \text{if } i \leq k - p \\
\frac{\bar{u} - u_i}{u_{i+p+1} - u_i} & k - p + 1 \leq i \leq k \\
0 & \text{if } i \geq k + 1
\end{cases}
\]

Inserting a knot multiple times can be handled through a generalization of Equation (2.63). Let us assume that the inserted knot value is \( \bar{u} \in [u_k, u_{k+1}) \) and initially has multiplicity \( s \). When it is inserted \( r \) times (where \( r + s \leq p \)), the ith new control point on the \( r \)th insertion is given by

\[
Q_{i,r} = \alpha_{i,r} Q_{i,r-1} + (1 - \alpha_{i,r}) Q_{i-1,r-1} \quad (2.64)
\]

where,

\[
\alpha_i = \begin{cases} 
1 & \text{if } i \leq k - p + r - 1 \\
\frac{\bar{u} - u_i}{u_{i+p+r+1} - u_i} & k - p + r \leq i \leq k - s \\
0 & \text{if } i \geq k - s + 1
\end{cases}
\]

A B-spline curve can be subdivided into Bézier segments by knot insertion at each internal knot until the multiplicity of each internal knot becomes \( p \). The new control points of each Bézier segment are obtained by Equation (2.64).
2.5 Coons Surface Patch

A Coons surface patch can be defined by suitable “blending” from four boundary curves as described in [21]. If a bilinear blending function is used, the resulting surface patch is called a bilinear Coons surface patch. Let us assume that four boundary curves \( P(u, 0) \), \( P(u, 1) \), \( P(0, v) \), and \( P(1, v) \) are given. A Coons surface patch is defined as [21]

\[
\mathbf{r}(u, v) = \left[ \begin{array}{c} 1 - u \\ u \end{array} \right] \left[ \begin{array}{cc} P(0, v) \\ P(1, v) \end{array} \right] + \left[ \begin{array}{cc} P(u, 0) \\ P(u, 1) \end{array} \right] \left[ \begin{array}{c} 1 - v \\ v \end{array} \right]
\]

(2.65)

When the four boundary curves are given in the B-spline form, a standard B-spline surface can be extracted by linearly blending the control points of the boundary curves, as in [25].

Let us assume that we have four boundary curves in B-spline form, \( \mathbf{r}(u, 0) \), \( \mathbf{r}(u, 1) \), \( \mathbf{r}(0, v) \) and \( \mathbf{r}(1, v) \) and opposite boundary curves are defined over the same knot vectors. This can always be achieved by merging knot vectors of opposite curves and inserting knots. A B-spline surface defined over the same two knot vectors of the boundaries is given by

\[
\mathbf{r}(u, v) = \sum_{i=0}^{n_1} \sum_{j=0}^{n_2} \mathbf{R}_{ij} N_{i,M}(u) N_{j,N}(v), \quad 0 \leq u, v \leq 1
\]

(2.66)

where

\[
\mathbf{r}(u, 0) = \sum_{i=0}^{n_1} \sum_{j=0}^{n_2} \mathbf{R}_{ij} N_{i,M}(u) N_{j,N}(0)
\]

(2.67)

\[
\mathbf{r}(u, 1) = \sum_{i=0}^{n_1} \sum_{j=0}^{n_2} \mathbf{R}_{ij} N_{i,M}(u) N_{j,N}(1)
\]

\[
\mathbf{r}(0, v) = \sum_{i=0}^{n_1} \sum_{j=0}^{n_2} \mathbf{R}_{ij} N_{i,M}(0) N_{j,N}(v)
\]

\[
\mathbf{r}(1, v) = \sum_{i=0}^{n_1} \sum_{j=0}^{n_2} \mathbf{R}_{ij} N_{i,M}(1) N_{j,N}(v)
\]

and \( M \) and \( N \) are the B-spline orders in the \( u \) and \( v \) directions, respectively. The
inner control points of the surface can be bilinearly blended as (see [25])

\[
R_{ij} = \begin{bmatrix}
1 - u_i & u_i \\
1 - v_j & v_j
\end{bmatrix}
\begin{bmatrix}
R_{ij0} \\
R_{i,jn}
\end{bmatrix} + \begin{bmatrix}
1 - u_i & u_i \\
1 - v_j & v_j
\end{bmatrix}
\begin{bmatrix}
R_{0j} \\
R_{0n,2}
\end{bmatrix}
\begin{bmatrix}
1 - v_j \\
v_j
\end{bmatrix}
\] (2.68)

Here, \( u_i \) and \( v_j \) are defined as

\[
u_i = (i - 1)/(n_1 - 1), \quad i = 1, 2, \ldots, n_1 \\
v_j = (j - 1)/(n_2 - 1), \quad j = 1, 2, \ldots, n_2.
\] (2.69)

### 2.6 Lofting

Given a family of \( m \) B-spline curves \( C^k(v), \) \( 0 \leq k \leq m - 1 \) defined over the same knot vector, we can find a B-spline surface using a standard lofting procedure \([33, 41]\). The curves become isoparametric lines of the resulting surface after lofting. The chord length parametrization is used for parametrization in \( u \)-direction. Applying the chord length parametrization to the end points of each curve, i.e. \( C^k(0), \) \( 0 \leq k \leq m - 1 \) and \( C^k(1), \) \( 0 \leq k \leq m - 1 \) and averaging them

\[
u_k = \frac{u_{0k} + u_{1k}}{2}, \quad 0 \leq k \leq m - 1
\] (2.70)

we can locate the parametric values of the curves in \( u \)-direction.

The lofted surface is a B-spline surface patch, defined in terms of parameter variables \((u, v)\) normalized to be in the range \([0, 1]\), given by:

\[
r(u, v) = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} R_{ij} N_i M(u) N_j N(v)
\] (2.71)

with knot vectors in the \( u \) and \( v \) parametric directions

\[
S = (s_0, s_1, \ldots, s_{m+M-1}), \quad T = (t_0, t_1, \ldots, t_{n+N-1})
\] (2.72)

where \( M \) and \( N \) are the B-spline orders in the \( u \) and \( v \) directions, respectively.

**Interpolation**

Since \( R(u, v) \) interpolates a family of curves which become isoparametric lines of the final surface, \( R(u_k, v), \) \( 0 \leq k \leq m - 1 \), the following condition must hold:

\[
\sum_{i=0}^{m-1} \sum_{j=0}^{n-1} R_{ij} N_i M(u_k) N_j N(v) = R(u_k, v) = \sum_{j=0}^{n-1} C_j(u_k) N_j N(v)
\] (2.73)
where \(0 \leq k \leq m\) and \(C_j(u_k)\) is the \(j + 1\)th control point of the \(k + 1\)th B-spline curves to be interpolated. Equation (2.73) is satisfied if and only if
\[
\sum_{i=0}^{m-1} R_{i,j} N_{i,M}(u_k) = C_j(u_k) \tag{2.74}
\]
where \(0 \leq j \leq n - 1\) and \(0 \leq k \leq m - 1\), since \(N_{j,N}(v)\) is a basis for B-splines of order \(N\) over the knot vector \(T\). Equation (2.74) is a set of \(3n\) linear systems of \(m\) equations each, all with the same system matrix. It can be rewritten in the matrix form as follows:
\[
[N] [R] = [C] \tag{2.75}
\]
where
\[
[N] = [N_{i,M}(u_k)]_{0 \leq k < m-1, 0 \leq i < n-1}
\]
\[
[R] = [r_0 \cdots r_{n-1}].
\]
\(r_j\) are submatrices of dimension \(m\) such that
\[
[r_j] = [R_{0,j} \cdots R_{m-1,j}]_{0 \leq j \leq n-1} \tag{2.77}
\]
and each \(R_{i,j}\) is a row vector of dimension 3.
\[
R_{i,j} = [x_{ij} y_{ij} z_{ij}]_{0 \leq i \leq m-1}. \tag{2.78}
\]
Further,
\[
[C] = [c_0 \cdots c_{n-1}] \tag{2.79}
\]
c\(_j\) are submatrices of dimension \(m\) such that
\[
[c_j] = [C_j(u_0) \cdots C_j(u_{m-1})]_{0 \leq j \leq n-1} \tag{2.80}
\]
and \(C_j(u_k)\) are row vectors with three elements each
\[
C_j(u_k) = [x_j(u_k) y_j(u_k) z_j(u_k)]_{0 \leq k \leq m-1}. \tag{2.81}
\]
The matrix \([N]\) is banded so that a specialized approach may be taken to solve the matrix equation (2.75) in order to save time and storage requirements.

**Approximation**

If curves are placed very closely, lofting them by interpolation may result in oscillations of the interpolated surface. Even though the deviation between two consecutive curves is not large, it may induce a large tangent value which undermines the smoothness of the resulting surface. This can be avoided by adopting an approximation.
scheme instead of interpolation.

Equation (2.74) implies that the number of lofted curves fixes the number of control points in the lofting direction. The idea for lofting through approximation is to use less control points in the lofting direction of the final surface, which leads to the following equation,

$$
\sum_{i=0}^{q-1} R_{ij} N_{i,m}(u_k) = C_j(u_k)
$$

(2.82)

where $0 \leq j \leq n - 1$, $0 \leq k \leq m - 1$ and $q < m$. Equation (2.82) is an overdetermined system and its solution can be found by using the least squares method [11, 35].
Chapter 3

Surface Approximation Using the Concept of a Curve on a Surface

3.1 Introduction

The concept of a curve on a surface is used in developing the theory of surface differential geometry [12, 31]. On the other hand, Hu and Sun [22] used the concept for matching two trimmed B-spline surfaces. In this thesis the concept is used to approximate a free-form surface in B-spline form from unorganized curves, which is called 2D to 3D conversion. The development of this algorithm is divided into two sections. First, the formulation of Bézier surface approximation is developed in detail. Second, this formulation is extended to B-spline surface approximation.

3.2 Curves on a Bézier Surface Patch

3.2.1 Formulation for Bézier Surface Patch

The measured \( m \) stripes, which are made of a series of points lined up in a sequential manner, are approximated in the least squares sense as Bézier curves of the form

\[
d(t) = \sum_{\mu=0}^{n_{3}} d_{\mu} B_{\mu,n_{3}}(t), \quad t \in [0, 1], \quad \xi = 1, \ldots, m
\]  

(3.1)

where the vector coefficients \( d_{\mu} \) are the control points of the \( \xi \)th curve and \( B_{\mu,n_{3}} \) are the Bernstein polynomials of degree \( n_{3} \). The resulting Bézier curves are considered as curves lying on a Bézier surface \( r(u, v) \) given by:

\[
r(u, v) = \sum_{i=0}^{n_{1}} \sum_{j=0}^{n_{2}} r_{ij} B_{i,n_{1}}(u) B_{j,n_{2}}(v), \quad (u, v) \in [0, 1]^{2}
\]  

(3.2)

where \( r_{ij} \) are the control points and \( B_{i,n_{1}}(u) \) and \( B_{j,n_{2}}(v) \) are the Bernstein polynomials of degree \( n_{1} \) and \( n_{2} \), respectively. Let us denote \( I = [0, 1] \) and \( U = [0, 1]^{2} \).
If we denote Equation (3.1) as $d_\xi(t_\xi) : I \rightarrow \mathbb{R}^3$ where $t_\xi \in I$ and Equation (3.2) as $r : U \rightarrow \mathbb{R}^3$, where $(u_\xi(t_\xi), v_\xi(t_\xi)) \in U$ the concept of a curve on a surface, as shown in Figure 3-1, can be stated that there exist unique functions $u_\xi$ and $v_\xi$ on a parametric domain $U$ such that

$$d_\xi(t_\xi) = r(u_\xi(t_\xi), v_\xi(t_\xi))$$

Since the stripes are approximately 3D planar curves, they can be assumed to be straight lines in parametric space $U$. Therefore, with the two end points of each stripe parameterized, the functions $u_\xi(t_\xi)$ and $v_\xi(t_\xi)$ are assumed to be

$$u_\xi(t_\xi) = (u_\xi^b - u_\xi^a)t_\xi + u_\xi^a$$
$$v_\xi(t_\xi) = (v_\xi^b - v_\xi^a)t_\xi + v_\xi^a$$

where $\xi = 1, \cdots, m$, $t_\xi \in [0, 1]$, and $(u_\xi^a, v_\xi^a)$ and $(u_\xi^b, v_\xi^b)$ are the parameterized two end points of each stripe. Then $r_\xi(t_\xi) = r(u_\xi(t_\xi), v_\xi(t_\xi))$, $\xi = 1, \cdots, m$, $t_\xi \in [0, 1]$ represent curves lying on the surface $r = r(u, v)$ which must match with the corresponding approximated stripes $d_\xi(t_\xi)$, $\xi = 1, \cdots, m$, $t_\xi \in [0, 1]$. Accordingly for each stripe the following relation must hold:

$$\sum_{i=0}^{n_2} \sum_{j=0}^{n_2} r_{ij} B_{i,n_1}((u_\xi^b - u_\xi^a)t_\xi + u_\xi^a) B_{j,n_2}((v_\xi^b - v_\xi^a)t_\xi + v_\xi^a) = \sum_{\mu=0}^{n_3} d_\xi^\mu B_{\mu,n_3}(t_\xi)$$

for $\xi = 1, \cdots, m$.

It is apparent that the left hand side of Equation (3.5) reduces to a degree $n_1 + n_2$ polynomial in $t_\xi$, while the right hand side is a degree $n_3$ polynomial in $t_\xi$. The coefficients of the polynomials of the left hand side in Bernstein form are evaluated using the arithmetic operations in Bernstein form. Also the degree of the right hand side can be raised to that of the left hand side through degree elevation. Since
Bernstein basis polynomials are linearly independent, each set of coefficients must be equal in order to satisfy Equation (3.5), yielding $n_1 + n_2 + 1$ linear vector equations with $(n_1 + 1)(n_2 + 1)$ vector unknowns (i.e. $r_{ij}$).

To evaluate the $n_1 + n_2 + 1$ Bernstein coefficients of the degree $n_1 + n_2$ polynomial of the left hand side, we start by rewriting the term $B_{i,n_1}((u_k^b - u_k^a)t_k + u_k^a)$ with the help of the binomial theorem as follows:

$$B_{i,n_1}((u_k^b - u_k^a)t_k + u_k^a) = \sum_{k=0}^{i} \sum_{l=k}^{n_1-i+k} B_{k,l}(u_k^b)B_{i-k,n_1-l}(u_k^a)B_{l,n_1}(t_k) \quad (3.6)$$

Similarly we can rewrite $B_{j,n_2}((v_k^b - v_k^a)t_k + v_k^a)$. Then using the formulae for multiplication of polynomials in Bernstein form (see [15]), we obtain

$$B_{i,n_1}((u_k^b - u_k^a)t_k + u_k^a)B_{j,n_2}((v_k^b - v_k^a)t_k + v_k^a) = \sum_{k=0}^{i} \sum_{l=k}^{n_1-i+k} \sum_{p=0}^{n_2-j+p} \sum_{q=p}^{n_2} \binom{n_1}{k} \binom{n_2}{l} W(t_k) \quad (3.7)$$

where

$$W(t_k) = B_{k,l}(u_k^b)B_{i-k,n_1-l}(u_k^a)B_{j-p,n_2-q}(v_k^b)B_{l+q,r}(t_k) \quad (3.8)$$

and $r = n_1 + n_2$. Now we raise the degree of the right hand side polynomial from $n_3$ to $r = n_1 + n_2$.

$$\sum_{\mu=0}^{n_3} d_{i,\mu} B_{\mu,n_3}(t_k)$$

$$= \sum_{\mu=0}^{r} \left( \sum_{j=\max(0,\mu-r+n_3)}^{\min(n_3,\mu)} \binom{r-n_3}{\mu-j} \binom{n_3}{j} \frac{1}{(r+j)!} \right) B_{\mu,r}(t_k), \quad t_k \in [0, 1]. \quad (3.9)$$

Finally by substituting Equations (3.7) and (3.9) into (3.5) we have

$$\sum_{i=0}^{n_1} \sum_{j=0}^{n_2} \sum_{k=0}^{n_1-i+k} \sum_{l=k}^{n_2-j+p} \sum_{p=0}^{n_1-i+k} \sum_{q=p}^{n_2-j+p} \binom{n_1}{l} \binom{n_2}{q} W(t_k)$$

$$= \sum_{\mu=0}^{r} \left( \sum_{j=\max(0,\mu-r+n_3)}^{\min(n_3,\mu)} \binom{r-n_3}{\mu-j} \binom{n_3}{j} \frac{1}{(r+j)!} \right) B_{\mu,r}(t_k), \quad t_k \in [0, 1] \quad (3.10)$$

where $W(t_k) = B_{k,l}(u_k^b)B_{i-k,n_1-l}(u_k^a)B_{j-p,n_3-q}(v_k^b)B_{l+q,r}(t_k)$ or in the matrix
form

\[
\begin{pmatrix}
B_{0,r}(t_\xi) & B_{1,r}(t_\xi) & \cdots & B_{r,r}(t_\xi)
\end{pmatrix}
\begin{bmatrix}
\alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n_1n_2} \\
\alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n_1n_2} \\
\vdots & \vdots & \ddots & \vdots \\
\alpha_{r,0} & \alpha_{r,1} & \cdots & \alpha_{r,n_1n_2}
\end{bmatrix}
\begin{pmatrix}
r_0 \\
r_1 \\
r_2 \\
\vdots \\
r_{n_1n_2-1} \\
r_{n_1n_2}
\end{pmatrix}
= \begin{pmatrix}
g_{\xi_0} \\
g_{\xi_1} \\
\vdots \\
g_{\xi_r}
\end{pmatrix}
\]

(3.11)

where

\[
g_{\xi_\mu} = \sum_{j=\max(0,\mu-r+n_3)}^{\min(n_2,\mu)} \binom{r-n_3}{\mu-j} \binom{n_3}{j} d_{\xi_j}.
\]

(3.12)

Since the Bernstein basis polynomials are linearly independent, Equation (3.11) reduces to

\[
\begin{bmatrix}
\alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n_1n_2} \\
\alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n_1n_2} \\
\vdots & \vdots & \ddots & \vdots \\
\alpha_{r,0} & \alpha_{r,1} & \cdots & \alpha_{r,n_1n_2}
\end{bmatrix}
\begin{pmatrix}
r_0 \\
r_1 \\
r_2 \\
\vdots \\
r_{n_1n_2-1} \\
r_{n_1n_2}
\end{pmatrix}
= \begin{pmatrix}
g_{\xi_0} \\
g_{\xi_1} \\
\vdots \\
g_{\xi_r}
\end{pmatrix}
\]

(3.13)

For \(m\) stripes we have \(m(n_1 + n_2 + 1)\) linear vector equations with \((n_1 + 1)(n_2 + 1)\) unknown control points \(r_{ij}\) of the Bézier surface, or briefly

\[
[\alpha]r = g
\]

(3.14)

where \([\alpha]\) is an \(m(n_1 + n_2 + 1) \times (n_1 + 1)(n_2 + 1)\) matrix, \(g\) is an \(m(n_1 + n_2 + 1) \times 1\) vector, and \(r\) is an \((n_1 + 1)(n_2 + 1) \times 1\) vector which can be determined using the least squares method [11, 35].

Figure 3-2 illustrates the process for Bézier surface approximation. The data points in Figure 3-2(a) are approximated with Bézier curves which are projected onto the base surface generated from the boundary curves (see Figure 3-2(b)) to estimate the parametric values of the end points of each curve. In the uv parametric space as shown in Figure 3-2(c), the Bézier curves are parameterized as straight lines by using Equation (3.4). The curves are provided as input to the reconstruction process which leads us to Equation (3.14). By solving the system of equations, we can calculate the control net of the Bézier surface approximating the given data (see Figure 3-2(d)).
Figure 3-2: Diagram for Bézier surface approximation
3.3 Curves on a B-spline Surface Patch

In this section, the development of Bézier surface patch reconstruction is extended to B-spline surface patch reconstruction. For the sake of simplicity we assume that the approximated stripe curves are cubic B-spline curves

\[ e_\xi(t_\xi) = \sum_{\mu=0}^{n_3} e_{\xi,\mu} N_{\mu,4}(t_\xi), \quad t_\xi \in [0,1], \quad \xi = 1, \ldots, m \]  

(3.15)

and assume that the surface to be reconstructed is a bicubic B-spline surface defined by

\[ q(u, v) = \sum_{i=0}^{m_1} \sum_{j=0}^{m_2} q_{ij} N_{i,4}(u)N_{j,4}(v), \quad (u, v) \in [0,1]^2 \]  

(3.16)

where \( q_{ij}, 0 \leq i \leq m_1, 0 \leq j \leq m_2 \) are control points, and \( N_{i,4} \) and \( N_{j,4} \) are the B-spline basis functions defined over knot vectors

\[ U_q = (u_0, u_1, \ldots, u_3, u_4, u_5, \ldots, u_{m_1-1}, u_{m_1}, u_{m_1+1}, \ldots, u_{m_1+4}) \]  

(3.17)

\[ V_q = (v_0, v_1, \ldots, v_3, v_4, v_5, \ldots, v_{m_2-1}, v_{m_2}, v_{m_2+1}, \ldots, v_{m_2+4}) \]  

(3.18)

associated with each parameter \( u, v \). If we insert knots 3 times at each internal knot in \( u \) and \( v \) directions, the bicubic B-spline surface will be split into \((m_1-2) \times (m_2-2)\) Bézier surface patches.

Let us denote the same surface defined on

\[ U_p = (u_0, u_1, \ldots, u_3, u_4, u_5, \ldots, u_{m_1-8}, u_{m_1-7}, u_{m_1-6}, u_{m_1-5}, \ldots, u_{m_1-2}) \]  

4 equal knots 3 equal knots

\[ V_p = (v_0, v_1, \ldots, v_3, v_4, v_5, \ldots, v_{m_2-8}, v_{m_2-7}, v_{m_2-6}, v_{m_2-5}, \ldots, v_{m_2-2}) \]  

4 equal knots 3 equal knots

as

\[ p(u, v) = \sum_{k=0}^{3m_1-6} \sum_{l=0}^{3m_2-6} p_{kl} N_{k,4}(u)N_{l,4}(v), \quad (u, v) \in [0,1]^2 \]  

(3.19)

where \( p_{kl}, 0 \leq k \leq 3m_1 - 6, 0 \leq l \leq 3m_2 - 6 \) are the control points. We want to express the control points \( p_{kl} \) in terms of \( q_{ij} \). As a first step we insert 2 knots, using Boehm's knot insertion algorithm [9, 34], at each internal knot \( u = u_p \), \( 4 \leq p \leq m_1 \) of \( U_q \) (3.17) leading to the following intermediate control points \( q_{kl}, 0 \leq k \leq 3m_1 - 6, \ldots \)
\[0 \leq j \leq m_2.\]

\[
\begin{align*}
\bar{q}_{0j} &= q_{0j} & (3.20) \\
\bar{q}_{1j} &= q_{1j} & (3.21) \\
\bar{q}_{(3p-10)j} &= \frac{u_p - u_p - 2}{u_p - u_p - 2} q_{(p-2)j} + \frac{u_{p+1} - u_p}{u_{p+1} - u_p - 2} q_{(p-3)j} & (3.22) \\
\bar{q}_{(3p-8)j} &= \frac{u_p - u_p - 1}{u_p - u_p - 1} q_{(p-1)j} + \frac{u_{p+1} - u_p}{u_{p+1} - u_p - 1} q_{(p-2)j} & (3.23) \\
\bar{q}_{(3p-9)j} &= \frac{u_p - u_p - 1}{u_p - u_p - 1} q_{(3p-8)j} + \frac{u_{p+1} - u_p}{u_{p+1} - u_p - 1} q_{(3p-10)j} & (3.24) \\
\bar{q}_{(3m_1-7)j} &= q_{(m_1-1)j} & (3.25) \\
\bar{q}_{(3m_1-6)j} &= q_{m_1j} & (3.26)
\end{align*}
\]

The next step is to insert 2 knots at each internal knot \(v = v_q, 4 \leq q \leq m_2\) of \(V_q\) (3.18) leading to

\[
\begin{align*}
p_{k0} &= \bar{q}_{k0} & (3.27) \\
p_{k1} &= \bar{q}_{k1} & (3.28) \\
p_{k(3q-10)} &= \frac{v_q - v_q - 2}{v_q - v_q - 2} \bar{q}_{k(q-2)} + \frac{v_{q+1} - v_q}{v_{q+1} - v_q - 2} \bar{q}_{k(q-3)} & (3.29) \\
p_{k(3q-8)} &= \frac{v_q - v_q - 1}{v_q - v_q - 1} \bar{q}_{k(q-1)} + \frac{v_{q+1} - v_q}{v_{q+1} - v_q - 1} \bar{q}_{k(q-2)} & (3.30) \\
p_{k(3q-9)} &= \frac{v_q - v_q - 1}{v_q - v_q - 1} \bar{q}_{k(3q-8)} + \frac{v_{q+1} - v_q}{v_{q+1} - v_q - 1} \bar{q}_{k(3q-10)} & (3.31) \\
p_{k(3m_2-7)} &= \bar{q}_{k(m_2-1)} & (3.32) \\
p_{k(3m_2-6)} &= \bar{q}_{km_2} & (3.33)
\end{align*}
\]

where \(0 \leq k \leq 3m_1 - 6\). Finally we are able to express the control points of the Bézier surface patches in terms of those of a B-spline surface (3.16) using Equations (3.20) to (3.33). The control points of a Bézier surface patch \(r_{ij}\) corresponding to the area bounded by the lattice \((u_p, v_q)-(u_{p+1}, v_q)-(u_{p+1}, v_{q+1})-(u_p, v_{q+1})\) where \(3 \leq p \leq m_1, 3 \leq q \leq m_2\) are given by

\[
r_{ij} = p_{(3(p-3)+i)(3(q-3)+j)}, \quad 0 \leq i \leq 3, \quad 0 \leq j \leq 3. \quad (3.34)
\]

Figures 3-3 (a) and (b) show a bicubic B-spline surface on

\[
\begin{align*}
U_q &= (0, 0, 0, 0, \frac{1}{3}, \frac{2}{3}, 1, 1, 1, 1) & (3.35) \\
V_q &= (0, 0, 0, 0, \frac{1}{3}, \frac{2}{3}, 1, 1, 1, 1) & (3.36)
\end{align*}
\]

with \(6 \times 6\) control points and its 9 split Bézier surface patches. In this case, the 16 control points of the center Bézier patch, whose parameter domain corresponds to
the area bounded by the lattice \((u_3, v_3)-(u_4, v_3)-(u_4, v_4)-(u_3, v_4)\), are given by

\[
P_{33} = \frac{1}{36} q_{33} + \frac{7}{72} q_{32} + \frac{1}{24} q_{31} + \frac{7}{72} q_{23} + \frac{49}{144} q_{22} + \frac{7}{48} q_{21} + \frac{1}{24} q_{13} + \frac{7}{48} q_{12} + \frac{1}{16} q_{11}
\]

\[
P_{34} = \frac{1}{18} q_{33} + \frac{1}{9} q_{32} + \frac{7}{36} q_{23} + \frac{7}{18} q_{22} + \frac{1}{12} q_{13} + \frac{1}{6} q_{12}
\]

\[
P_{35} = \frac{1}{9} q_{33} + \frac{1}{18} q_{32} + \frac{7}{18} q_{23} + \frac{7}{36} q_{22} + \frac{1}{6} q_{13} + \frac{1}{12} q_{12}
\]

\[
P_{36} = \frac{1}{24} q_{34} + \frac{1}{72} q_{33} + \frac{1}{36} q_{32} + \frac{7}{48} q_{24} + \frac{49}{144} q_{23} + \frac{7}{72} q_{22} + \frac{1}{16} q_{14} + \frac{7}{48} q_{13} + \frac{1}{24} q_{12}
\]

\[
P_{43} = \frac{1}{18} q_{33} + \frac{7}{36} q_{32} + \frac{1}{12} q_{31} + \frac{1}{9} q_{23} + \frac{7}{18} q_{22} + \frac{1}{6} q_{21}
\]

\[
P_{44} = \frac{1}{9} q_{33} + \frac{2}{9} q_{32} + \frac{2}{9} q_{23} + \frac{4}{9} q_{22}
\]

\[
P_{45} = \frac{1}{9} q_{33} + \frac{2}{9} q_{32} + \frac{4}{9} q_{23} + \frac{2}{9} q_{22}
\]

\[
P_{46} = \frac{1}{12} q_{34} + \frac{1}{36} q_{33} + \frac{1}{18} q_{32} + \frac{1}{6} q_{24} + \frac{7}{18} q_{23} + \frac{1}{9} q_{22}
\]

\[
P_{53} = \frac{1}{9} q_{33} + \frac{7}{18} q_{32} + \frac{1}{6} q_{31} + \frac{1}{18} q_{23} + \frac{7}{36} q_{22} + \frac{1}{12} q_{21}
\]

\[
P_{54} = \frac{2}{9} q_{33} + \frac{1}{9} q_{32} + \frac{1}{9} q_{23} + \frac{2}{9} q_{22}
\]

\[
P_{55} = \frac{4}{9} q_{33} + \frac{1}{9} q_{32} + \frac{9}{9} q_{23} + \frac{1}{9} q_{22}
\]

\[
P_{56} = \frac{1}{6} q_{34} + \frac{7}{18} q_{33} + \frac{1}{9} q_{32} + \frac{1}{12} q_{24} + \frac{7}{36} q_{23} + \frac{1}{18} q_{22}
\]

\[
P_{63} = \frac{1}{24} q_{43} + \frac{1}{48} q_{42} + \frac{1}{16} q_{41} + \frac{7}{72} q_{33} + \frac{49}{144} q_{32} + \frac{7}{48} q_{31} + \frac{1}{36} q_{23} + \frac{7}{72} q_{22} + \frac{1}{24} q_{21}
\]

\[
P_{64} = \frac{1}{12} q_{43} + \frac{1}{6} q_{42} + \frac{7}{36} q_{33} + \frac{7}{18} q_{32} + \frac{1}{18} q_{23} + \frac{1}{9} q_{22}
\]

\[
P_{65} = \frac{1}{18} q_{22} + \frac{7}{36} q_{32} + \frac{1}{12} q_{42} + \frac{1}{9} q_{23} + \frac{7}{18} q_{33} + \frac{1}{6} q_{43}
\]

\[
P_{66} = \frac{1}{16} q_{44} + \frac{1}{48} q_{43} + \frac{1}{24} q_{42} + \frac{7}{48} q_{34} + \frac{49}{144} q_{33} + \frac{7}{72} q_{32} + \frac{1}{24} q_{44} + \frac{7}{72} q_{23} + \frac{1}{36} q_{22}
\]

Once we have the control points for each Bézier surface patch expressed by those of the B-spline surface patch to be reconstructed, we use the same procedure that we used for the Bézier surface patch reconstruction described in Section 3.2.1 to reconstruct a B-spline surface patch. We also assume that the pre-images of the stripes in the \(uv\)-parameter space are straight lines. Generally the straight lines in the \(uv\)-parameter space extend over several Bézier patches and we need to compute the intersections with the lattice of isoparametric lines at each internal knot. The approximated cubic B-spline curves are first subdivided into Bézier curves and further subdivided into smaller Bézier curves at these intersection points using Boehm’s algorithm [9, 34]. For each Bézier stripe in each Bézier patch we apply Equation (3.13)
Figure 3-3: (a) A bicubic B-spline surface patch and its control points. (b) After knot insertion, the B-spline surface has been split into 9 Bézier surface patches.

where control points of a Bézier patch $r_{ij}$ are replaced by the linear combination of $q_{ij}$ through the relations from (3.20) to (3.34).

If we denote the total number of stripes in all the Bézier surface patches by $m_T$, we have a global system

$$[\beta]q = h$$  \hspace{1cm} (3.37)

where $[\beta]$ is a $7m_T \times (m_1 + 1)(m_2 + 1)$ matrix, $h$ is a $7m_T \times 1$ vector, and $q$ is an $(m_1 + 1)(m_2 + 1) \times 1$ unknown vector which is determined in the least square sense [17, 18].

### 3.4 Boundary Condition

A boundary condition for the surface reconstruction is considered in Equation (3.14) and (3.37). Boundary curves in Bézier or B-spline form are used as the boundary condition of reconstruction and the resulting surface interpolates them.

In Equation (3.14), the control points of the resulting Bézier surface that need to be determined are

$$r_{ij}(0 \leq i \leq n_1, \ 0 \leq j \leq n_2).$$  \hspace{1cm} (3.38)

However, the control points which correspond to the boundary curves are known and we have to find the rest of the inner control points as given in the following.

$$r_{ij}(1 \leq i \leq n_1 - 1, \ 1 \leq j \leq n_2 - 1).$$  \hspace{1cm} (3.39)

Therefore, we can rewrite Equation (3.13) as

$$[\alpha]r_1 = g - [\alpha]r_2 + [\alpha]r_3$$  \hspace{1cm} (3.40)
where \( \mathbf{r}_1 \) contains unknown control points with zeros in the rows corresponding to known boundary control points, \( \mathbf{r}_2 \) is the control points of the given boundary curves, i.e. \( \mathbf{r}_{00} \cdots \mathbf{r}_{0n_2}, \mathbf{r}_{00} \cdots \mathbf{r}_{n_10}, \mathbf{r}_{n_10} \cdots \mathbf{r}_{n_1n_2}, \mathbf{r}_{0n_2} \cdots \mathbf{r}_{n_1n_2} \), with zeros for the rest rows and \( \mathbf{r}_3 \) are four control points at the corners of the control net, i.e. \( \mathbf{r}_{00}, \mathbf{r}_{n_10}, \mathbf{r}_{0n_2}, \mathbf{r}_{n_1n_2} \), given by the boundary curves with zeros for the rest rows. Similarly, for the B-spline surface approximation, the same discussion is applied, which leads us to the following equation.

\[
[\beta]\mathbf{q}_1 = \mathbf{h} - [\beta]\mathbf{q}_2 + [\beta]\mathbf{q}_3
\]  

(3.41)

where \( \mathbf{q}_1, \mathbf{q}_2 \) and \( \mathbf{q}_3 \) are the same as those for the Bézier formulation. Equation (3.40) or (3.41) is reduced to a linear system of equations whose solution is obtained in the least squares sense.

### 3.5 B-spline Surface Approximation

#### 3.5.1 Least Squares Method

The linear system (3.37) is overdetermined, i.e. it has more equations than unknowns. In this case, no exact solution vector \( \mathbf{q} \) to the system is generally available. Instead, the best compromise solution must be sought, namely, the one that satisfies all equations as closely as possible simultaneously. The sum of the squares of the difference between the left and the right hand sides of Equation (3.37) is chosen as an indicator for the best solution.

The overdetermined linear system (3.37), reduces to the normal equations [11, 35] by premultiplying the transpose of \([\beta]\) as follows:

\[
[\beta]^T[\beta]\mathbf{q} = [\beta]^T\mathbf{h}
\]  

(3.42)

Study of the form of matrix \([\beta]\) indicates that after multiplying the transpose of the matrix \([\beta]\), the resulting matrix \([\beta]^T[\beta]\) is sparse, banded and symmetric. A wise use of such characteristics must be made to minimize computation time and memory consumption.

A routine which is specialized for finding the solution of a set of real symmetric positive-definite band linear equations [27] is used. This routine uses the Cholesky decomposition and the time taken is approximately proportional to \(N \times M1 \times IR\), where \(N\) is the order of the coefficient matrix such as \([\alpha]^T[\alpha]\) or \([\beta]^T[\beta]\), \(M1\) is the number of lines of the matrix on either side of the diagonal and \(IR\) is the number of columns of right-hand side of the equation [27]. \(N\) is equal to \(n_u \times n_v\) so that the running time becomes \(O(n_u n_v M1)\) where \(IR\) is a fixed value of 3.
3.6 Complexity Analysis

The calculation of coefficient matrix in Equation (3.14) involves nested summations (see Equation (3.10)), which is implemented with nested for-loops as follows:

1 : for (i = 0; i <= n1; i++)
2 : for (j = 0; j <= n2; j++)
3 : for (k = 0; k <= i; k++)
4 : for (l = k; l <= n1 - i + k; l++)
5 : for (p = 0; p <= j; p++)
6 : for (q = p; q <= n2 - j + p; q++)
7 : Operations for computing coefficients

We can divide the nested for-loops into two groups: for-loops for n1 and for-loops for n2. The indices related to n1 are i, k and l and the indices related to n2 are j, p and q. These two groups can be regarded independent of each other in that any indices in one group are not used in the other group. Therefore, we can analyze each group separately and multiply their results to get the complexity of the algorithm.

The first group related to n2 contains lines 2, 5 and 6. To simplify the analysis, let us assume that the operations for computing coefficients take a constant time, i.e. O(1). Then we have three nested for-loops, i.e. lines 2, 5 and 6. When j = 0, then p = 0 and q runs from 0 to n2. The next step is when j = 1. This time, p varies from 0 to 1 with q running from p to n2 - 1 + p and so on. By induction, the total number of operations, N, can be found as follows

\[ N = \sum_{j=0}^{n2} (j + 1)(n2 - j + 1). \]  \hspace{1cm} (3.43)

This equation can be expressed in closed form as follows

\[
N = \sum_{j=0}^{n2} n2j - j^2 + n2 + 1 \\
= n2 \sum_{j=0}^{n2} j - \sum_{j=0}^{n2} j^2 + \sum_{j=0}^{n2} n2 + \sum_{j=0}^{n2} 1 \\
= \frac{n2^2(n2 + 1)}{2} - \frac{1}{6} n2(n2 + 1)(2n2 + 1) + n2(n2 + 1) + (n2 + 1) \\
= \frac{1}{6} (n2 + 1)(n2 + 2)(n2 + 3) \]  \hspace{1cm} (3.44)

where the formulae \(\sum_{j=0}^{n} j = \frac{n(n+1)}{2}\) and \(\sum_{j=0}^{n} j^2 = \frac{1}{6} n(n+1)(2n+1)\) [10] have been used. Similarly, the second group, which consists of lines 1, 3 and 4, yields the number of executions in the same form as Equation (3.44). So, with the first and the second
groups considered altogether, the total number of operations, $N_T$ is

\[ N_T = \frac{1}{36} (n_1 + 1)(n_1 + 2)(n_1 + 3)(n_2 + 1)(n_2 + 2)(n_2 + 3) \]  

(3.45)

and if Big O notation is introduced [10], the time complexity of Equation (3.45) becomes $O(n_1^3 n_2^3)$ with the assumption that each operation takes a constant time.

Inside the operations for computing coefficients, line 7, there exist several routines which involve calculation of Bernstein polynomials and other for-loops which are dependent on $n_1$ and $n_2$. There are two different approaches to calculate Bernstein polynomials: one is to use the definition given in Equation (2.27) and the other is to use the recurrence relation given in Equation (2.31). Equation (2.27) requires $O(n)$ time, whereas Equation (2.31) takes $O(n^2)$ time where $n$ is the degree of Bernstein polynomials. In this algorithm, the recurrence relation, Equation (2.31) has been used because, in general, it gives more accurate results than Equation (2.27) when a floating point calculation is involved. Therefore, line 7 takes $O(n_1^3 + n_2^3)$ and together with the outer for-loops, the asymptotic behavior of the routine becomes $O(n_1 n_2^3 (n_1^2 + n_2^2))$. The routine is applied to each input curve. So if $n_I$ is the number of input curves, then the total operation takes $O(n_1 n_2^3 (n_1^2 + n_2^2))$ time.

When a bicubic B-spline surface is used for reconstruction, subdivision into cubic Bézier surface patches is included as an intermediate step. The subdivision requires $O(n_s)$ time where $n_s$ is the number of subdivided cubic Bézier surface patches. The number $n_s$ is determined by the number of internal knots in $u$ and $v$ directions which are related to the number of control points of boundary curves. Let us assume that the numbers of control points in $u$ and $v$ directions are $n_u$ and $n_v$, respectively. Because the total number of cubic Bézier surface patches is proportional to the product of the numbers of control points in both directions, the required time for subdivision reduces to $O(n_u n_v)$.

After the subdivision, a system of equations for each Bézier surface patch is obtained to determine its 16 control points that produce a surface approximating the input curve segments lying on the Bézier surface patch. This takes $O(n_i n_1^3 n_2^3 (n_1^2 + n_2^2))$ time where $n_i$ is the number of the input curve segments lying on each Bézier surface patch and $n_1$ and $n_2$ are degrees of the subdivided Bézier surface patch in $u$ and $v$ directions. However, the subdivided Bézier surface patch has a degree 3 in $u$ and $v$ directions which fixes $n_1$ and $n_2$ so that the time reduces to $O(n_i)$. This process is applied to every Bézier surface patch. Therefore, the total operation takes \( \sum_{l=0}^{n_u} n_v n_u n_v \) where $n_u$ and $n_v$ are the numbers of control points in $u$ and $v$ directions respectively and $n_{il}$ is the number of the input curve segments lying on the $l$th Bézier surface patch. It can be written as $O((\sum_{l=0}^{n_u} n_{il}) n_u n_v)$, which reduces to $O(n_I n_u n_v)$ where $n_I$ is $\sum_{l=0}^{n_u} n_{il}$.

The linear conversion between the control points of a subdivided Bézier surface patch and those of the reconstructed B-spline surface shown in page 45 is performed in $O(n_I n_u n_v)$ where $n_i$ is the number of the input curve segments lying on the Bézier surface patch and $n_u$ and $n_v$ are the numbers of control points in $u$ and $v$ directions respectively. This conversion is applied to every Bézier surface patch so that the total
execution time is \( \sum_{l=1}^{n_u n_v} O(n_d n_u n_v) \) which can be rewritten as \( O((\sum_{l=1}^{n_u n_v} n_d) n_u n_v) \) where \( n_d \) is the number of the input curve segments lying on the \( l \)th Bézier surface patch. Therefore, it becomes \( O(n_l n_u n_v) \) where \( n_l \) is \( \sum_{l=0}^{n_u n_v} n_d \).

Equation (3.37) is an overdetermined system so that no unique solution can be found. The best compromised solution to satisfy the set of equations in the least squares sense is calculated. In general, the number of rows of \( [\beta] \) denoted as \( N_r \) is proportional to the number of the subdivided Bézier segments \( n_l \) and the number of the subdivided Bézier surface patches \( n_g \). Normalizing the equation by pre-multiplying the transpose of \( [\beta] \) results in a banded and symmetric matrix which enables us to use a specialized technique such as Cholesky decomposition. However, the multiplication time complexity is of \( O(N_r^2 n_u n_v) \) [10] which grows dramatically as \( N_r \) increases. Here, \( N_r \) is proportional to \( 7 \times n_l \). Therefore this is rewritten as \( O(n_l^2 n_u n_v) \).

Once the normalized coefficient matrix \( [\beta]^T [\beta] \) is calculated, the change of the shapes of any input curves can be reflected much faster. After finding the set of control points of subdivided Bézier surface patches, we can update the right hand side of Equation (3.37) and just pre-multiply \( [\beta]^T \) to it. Then we can get a new system of equations for the new family of input curves without evaluating \( [\beta]^T [\beta] \) again. Therefore, the time complexity for the subsequent phase reduces to \( O(N_r^2) \), because the number of columns of \( [h] \) is always 3.
Chapter 4

System and Configuration

4.1 Introduction

This chapter describes three components that comprise the system such as a host computer, a robot and a laser scanner. The hardware and software is explained and the configuration of the system is described.

4.2 Hardware and Software

4.2.1 Laser Scanner

Hardware

The sensor system 4D Imager (4DI) [19], which is manufactured by Intelligent Automation Systems, is attached to the robot for adaptive sensing. The sensor is a flexible non-contact laser sensor that captures three dimensional information from an object in real time. 4DI system consists of three essential components: the 4DI sensor head, a PC equipped with an image processing board and 3D triangulation software. The sensor head has a light source and three CCD video cameras (768 × 494 pixels). The light source is a visible red laser (with wavelength of 768 nm) combined with a diffraction grating that generates a fixed number of stripes, also called laser planes, which enable the system to acquire 50,000 points in a scene with a single image frame. The range measurements are calculated using triangulation between the binary stereo images and the calibrated laser plane locations. The sensor is able to acquire data in 0.1 milliseconds and process them in 100 milliseconds and has a measurement standard deviation of 1/2000 of the field of view. The laser scanner system is controlled by a PC which processes a set of data points through the image processing board. The PC is operated on Windows 95 and utility programs for the scanner system are provided by the manufacturer.
Software

A program has been developed to support various operations for the scanner. It provides functions for connection between the scanner and the host computer through the internet line (TCP/IP connection) and for control of the scanner with the library functions for the basic operations supplied by the manufacturer. Visual C++ 6.0 is used for software development and a socket interface is used for TCP/IP network connection.

4.2.2 Robot

Hardware

The Adept One 5-axis direct-drive SCARA configuration manipulator [2] has high accuracy, speed and reliability. Joints 1, 2, 4, and 5 are rotational, while Joint 3 is translational as shown in Figure 4-2. The robot controller has its own operating system and a script language for various operations of the robot [3, 4].

Software

A program for the robot has been developed in V+, which is a computer based control system and programming language designed specifically for use with the robot. It provides all the functionality necessary for connection and control of the robot. The program consists of two parts: one controls the communication with the host computer (communication part) and the other controls the robot movements (control part). The communication part reads commands and data from the host computer and sends information to the requester. The interpretation of commands and data is performed in the control part. It also generates commands for motion control of the robot.
4.2.3 Host Computer

An SGI machine (200MHz MIPS R5000, 128Mbyte RAM), operating on UNIX (IRIX 6.5), is designated as the host computer of the entire system. It is connected with the robot controller and the PC for the scanner and a stand alone program running in the host computer analyzes 3D data coming from the scanner, predicts the next position of the robot, issues various commands to synchronize every process of each unit and produces CAD surface models. The programs are written in the C and C++ languages based on X Window System and Open Inventor [43] is used for visualization.

4.2.4 Accuracy

The specification of the scanner indicates that accuracy of digitization is $\pm 0.025\text{mm}$ for height and the performance specification of the robot says that each joint has its own accuracy and the robot in X-Y plane which involves simultaneous movements of several joints operates with the precision of $\pm 0.076\text{mm}$. However, a further consideration is needed on accuracy of the system. The fact that the size of the field of view of the scanner is too small to capture the whole shape of the plate requires a multiple scanning scheme, which implies that the scanner needs to be mounted on the robot arm, resulting in the interaction of the accuracy of each component. Since the data from different shots have to be merged into a common coordinate system, the precision with which the scanner is positioned for each shot has a direct influence on the accuracy with which the data points are referenced. Two aspects of the movement of the robot and the scanner have to be investigated to assess the overall precision of the system: accuracy and repeatability. Accuracy makes reference to the precision with which the robot arm is positioned, whereas repeatability means the difference in position when the arm is made to go to a certain position from different origins or after moving through different paths. A series of tests should be run in order to
assess accuracy and repeatability of the system.

4.3 Configuration

This section illustrates how the components are configured to behave as one integrated system. Two different connections, i.e. TCP/IP and RS-232C are explained and their application to the system is described.

4.3.1 TCP/IP

The origin of TCP/IP dates back to the ARPANET sponsored by the Advanced Research Project Agency (ARPA) of the Department of Defense (DoD) [39]. In 1984, the DoD divided the ARPANET into two networks: the ARPANET for experimental research and the MILNET for military purposes. In the early 1980s, a new family of protocols was specified as the standard for the ARPANET and associated DoD networks. Although the accurate name for this family of protocols is the “DARPA Internet Protocol Suite”, it is commonly referenced to as the TCP/IP protocol suite or just TCP/IP, (Transmission Control Protocol/Internet Protocol).

TCP/IP has been implemented on everything from personal computers to the largest supercomputers. It allows us to connect computers all over the world or to
have a network consisting of only two personal computers in the same room connected by the same TCP/IP protocol.

**IP**

IP stands for the *Internet Protocol*, which is the key tool for building large networks by interconnecting smaller networks. A service model for IP is defined to provide an addressing scheme and data delivery service which serves as a foundation for higher-level protocols.

**TCP**

TCP is the acronym of *Transmission Control Protocol*. It runs on top of IP on the hosts, which is clearly shown in Figure 4-4, and defines additional operations to provide better services. It guarantees the reliable, in-order delivery of a stream of bytes. It is a full-duplex protocol, meaning that each TCP connection supports a pair of byte streams, one flowing in each direction. It also includes a flow-control mechanism for each of these byte streams that allows the receiver to limit how much data the sender can transmit at a given time. Since TCP uses IP, the entire protocol suite is often called the TCP/IP protocol family.

**Layer Model for TCP/IP Connection**

A simplified 4 layer model connecting two systems is shown in Figure 4-4. A user on one system is allowed to send and receive files or data to and from another system. An application uses TCP, which in turn uses IP to exchange data with other host. The layers connected with dashed lines in Figure 4-4 indicate that they communicate virtually with each other using the indicated protocols. The solid lines between layers in Figure 4-4 mean the actual flow of data.
Berkeley Socket and Application

TCP/IP is implemented as a part of the operating system (OS). The interface to TCP/IP that an OS provides is called the network application programming interface (API). The API for TCP/IP, often called the socket interface and originally provided by the Berkeley distribution of UNIX, is supported in virtually all popular operating systems such as Windows 98/98/NT and UNIX. The API defines a family of functions of services supported by TCP/IP. An example of a sequence and the related functions for connection are illustrated in Figure 4-5.

The `socket()` operation in the server and client sides creates a socket which specifies the type of communication protocol desired. The `bind()` operation in the server side binds the newly created socket to the specified address and the `listen()` defines how many connections can be pending on the specified socket. The `accept()` operation accepts connections from clients. The client performs an active open, namely, it initiates communication with the server by invoking `connect()` operation. Once a connection is established, the `read()` and `write()` operations are invoked to receive and send data [39].

4.3.2 Serial Communication

A serial connection or RS-232C communication is another way for connecting any two components of the system. Several serial ports are available for general use and
each port is configured and set up when an OS is loaded and initialized. Specific
application programming interfaces (API’s) for serial input and output are supplied
and proper use of them make it possible to communicate with other devices equipped
with serial ports through the serial cable.

4.3.3 Connection

An extensive study of the flow of data, resources available for each piece of equipment
and the operational environment have lead us to reach the the current configuration
as shown in Figure 4-6.

The host computer and the laser scanner communicate with each other via the
internet line, i.e. through TCP/IP connection. Each component has a unique internet
address so that the distance between them is of no concern as long as they are hooked
up to the network. The services which TCP/IP provides enable the two devices to
exchange data and commands fast (10Mbps) and safely through the internet line.
Lack of TCP/IP services of the robot controller forces us to use an alternative for the
connection between the robot controller and the host computer. RS-232C or serial
connection is provided by both machines. Each unit supplies functions to handle
I/O operations through serial ports and systematic use of them enables us to achieve
the bi-directional communication between the robot controller and the host computer.
They are connected physically with a serial cable through which commands and values
for the position of the robot and the scanner are transmitted.
Figure 4-6: Configuration of the system
Chapter 5

System Operation Procedures

5.1 Introduction

This chapter provides a complete description on the operation procedures of the system. The system consists of the scanner, the robot and the host computer and is configured as described in Chapter 4. It is initialized and begins the loop for automatic surface reconstruction. Initialization of each component and the flow of the system are discussed.

5.2 Initialization

5.2.1 Scanner and Network

Scanner

The scanner and the image processing board which is installed on the PC must be initialized for the system to work properly. This initialization is performed through a function provided by the manufacturer whenever the scanner is turned on.

Network

The scanner controller or PC, which is operated on Windows 95, supports TCP/IP network protocol and communicates with the host computer through the internet line. TCP/IP connection requires at least one side to be a server and others to be clients. Normally, the server is configured to respond to client’s requests. Before starting its operation, the network must be initialized for the connection between the host computer and the scanner in such a way that the scanner, which is set up as a server, is waiting for any request from the host computer. The host computer is designated as a client because it is the host computer that requests various operations for the scanner.
5.2.2 Robot

During the boot process, the robot initializes itself by moving to predefined sequential positions to calibrate its movements. This calibration stage should be performed whenever the robot system is turned on. After initialization, the control program in the robot controller begins and gets ready to respond to any request from the host computer.

5.3 Flow Chart of the System

5.3.1 Overview

The flow chart of the system is depicted in Figure 5-1. One assumption is made for this procedure that the object of interest is topologically equivalent to a four sided free-form surface representable in an explicit, smooth, non-parametric form, whose tangent planes form a small angle with an appropriate reference plane. Before beginning the process, the scanner is placed in the middle of the plate so that the field of view of the scanner is positioned entirely inside the surface of the plate. After starting the boundary extraction loop shown in Figure 5-1, the robot moves in the x direction (see Figure 5-6 for the coordinate system) until it reaches one of the boundary curves. When the system detects an edge, it begins the boundary construction process. Once the locus of the scanner forms a closed loop, the iteration stops and the whole boundary is constructed. Based on the boundary information, the scanner digitizes the whole surface of the plate following a zig-zag path and a surface is reconstructed. The CAD representation of the surface is forwarded to the line heating subsystem for further analysis.

5.3.2 Boundary Construction

An estimate of the boundary of the plate must be made to allow computation of the size and shape of the plate from which the number and position of the scanner shots are determined in order to digitize the whole shape. It is used for extracting information of the boundary of the plate to limit the range that the robot may sweep over and approximating the surface in later stages.

Edge Detection

The data set produced by the scanner is organized in such a way that it shows many stripes which are almost parallel to each other and each stripe is made of a series of points lined up in a sequential manner as shown in Figure 5-2. The difference of the height resulting from the thickness of the plate ensures the existence of an edge in the current shot as shown in Figure 5-2. Tangent vectors along each stripe (approximated by taking the difference of the position vectors of two consecutive points) are computed to locate edge points in the current digitized data set by checking an abrupt change in the tangent vector values.
Figure 5-1: Overview of the system

Figure 5-2: Data set containing an edge
Boundary Tracking

An assumption that the plate is “approximately rectangular” reduces many special cases to be handled in the boundary tracking algorithm. More research needs to be performed in the future to treat more general cases. In the meantime, the assumption is adopted in algorithm development.

Once the scanner reaches one of the boundary edges, it starts to track the boundary lines of the plate in the counterclockwise direction. There are two cases that the algorithm should differentiate: a boundary line and a corner. First, the edge points are extracted from the raster lines by the edge detection algorithm as explained in the previous section. Three consecutive points from the left and three from the right edge points are taken to find the average left and right tangent vectors, respectively as shown in Figure 5-4. Each case can be identified by checking the difference of the angle of the tangent vectors of both ends. Figure 5-5 illustrates the two separate cases where the algorithm recognizes a corner or an edge. Figure 5-5(a) shows that the scanner captures an edge and Figure 5-5(b) shows that the scanner captures a corner. When the scanner detects a side, it moves to the next position which is estimated from the left tangent vector and a user-defined distance. The direction of the left tangent vector and a half of the width of the field of view of the scanner are used to determine the next position. When the scanner captures a corner, it rotates by the amount of angle difference of the two tangent vectors at both ends of the extracted edge points and moves to the next scanning position determined from the left tangent vector and a user-defined distance. The corner points of the plate should be identified for boundary construction. However, it often happens that a corner point is positioned between two extracted edge points. In that case, the corner point is extrapolated based on the edge points in the neighborhood.

The scanner is always positioned with rotation around Joint 4 (see Figure 4-2) so that an acute angle (less than 50 degrees) is formed between the boundary lines of the plate and the raster lines of the scanner to avoid the case that the raster lines lie parallel to any of the boundary lines near the corner of the plate. If the raster lines are parallel to the boundary line around the corner, it happens that a boundary line lies between any two adjacent raster lines as shown in Figure 5-3(a). In such case, the boundary line cannot be captured in the data set. Therefore, in order to detect the missing boundary line, decision must be made that the current data set contains a corner by investigating the relative positions of the raster lines and one more shot for the same region needs to be taken after rotation to recognize the missing edge line. But disturbances coming from the environment (such as light condition, surface condition of the plate, etc) sometimes introduce unexpected errors in the positions of the raster lines, which lead to failure in detection of the existence of the corner.

Once the corner is recognized, the scanner rotates by the amount of angle difference of the two tangent vectors at both ends of the extracted edge points and moves to the next scanning position determined from the left tangent vector.
Figure 5-3: Corner detection

Figure 5-4: Tangent vectors of extracted edge points at both ends
Figure 5-5: Corner is detected by the difference of the tangent vectors at both ends.

5.3.3 Surface Scanning

From the geometric information of the plate which comes from the construction of the boundary, the number of camera shots which is sufficient to capture the whole plate is calculated. A simple approach for this calculation involves calculation of the area of the 2D bounding box that contains the boundaries followed by division of this entire area by the area of the field of view of the scanner. Depending on the smoothness of the plate, the size of the overlap region between any two consecutive shots should also be considered. For a complicated shape, overlapping portions are necessary to capture all geometric features of the surface and to improve fine tuning of registration of digitized data points. In the merging process, the redundant data are discarded. However, for a smoothly varying surface such as an intermediate shape during the metal forming process by line heating, overlapping is not necessary.

The scanner is positioned such that it begins to scan at the left top corner of the plate. It scans in the negative $x$ direction (see Figure 5-6) and reaching the opposite edge, moves in the negative $y$ direction. These successive movements form a zig-zag path which the scanner follows to cover the entire surface. During digitization of the surface, the measured 3D data points together with the position of the scanner are sent to the host computer and stored on the disk for later use.
5.3.4 Surface Fitting

Data Preparation

Data points from each scan are based on the local coordinate system whose origin lies at the center of the field of view of the scanner. Therefore all the data points must be registered to a common coordinate system to get an appropriate geometric representation of the plate in a single coordinate frame of reference. The robot reads the position of the tip of its arm with respect to the global coordinate system which is fixed at the base of the robot body (see Figure 5-6). The distance between the tip position and the origin of the local coordinate system and the angle made around Joint 4 (see Figure 4-2) are used in the registration to get the coordinates of the data points with respect to the global coordinate system. Let us assume that \( P \) is a data point with respect to the local coordinate system, \( \alpha \) is the angle made for the boundary tracking process and \( d_x \) and \( d_y \) are the physical distances between the origin of the local coordinate system and the tip position of the robot arm. The registered point, \( P' \), is given by

\[
P' = RP + T \tag{5.1}
\]

where

\[
R = \begin{bmatrix} \cos \alpha & \sin \alpha & 0 \\ -\sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad T = \begin{bmatrix} d_x \\ d_y \\ 0 \end{bmatrix}
\]

and \( R, T \) are a rotation matrix and a translation vector. Here, the translation is performed in \( xy \) plane and the rotation is applied around Joint 4. All the registered data points based on the global coordinate system are merged to represent the digitized surface of the plate as shown in Figure 5-7.

An issue may arise whether the registered data points do reflect the real surface of the plate or not, because the translation and rotation operations given in Equation (5.1) are influenced by not only inaccurate installation of the scanner and the robot but also errors originating in the robot movements. Such uncertainty must be resolved through the thorough investigation of the accuracy and repeatability of the system as suggested in section 4.2.4.

Surface Reconstruction

Two methods are examined for surface reconstruction. One is lofting and the other is surface reconstruction using the concept of a curve on a surface. In order to use the lofting method, the data points have to be processed in such a way that they form isoparametric lines of the reconstructed surface. This step includes selection of stripes in the family of data stripes. The strategy is to choose two stripes in the two consecutive shots (one stripe from each shot), which give the minimum distance and connect those selected stripes implicitly. They are treated as one stripe after connection, which runs from an edge to the other facing edge and is used as an
Figure 5-6: The global coordinate system

Figure 5-7: Merged data points and reconstructed surface through lofting
isoparametric line in the lofting stage. In general, however, the connected lines are not suitable for lofting because the stripes after the registration may not align smoothly, which results in "non-smooth" iso-parametric lines (i.e. isoparametric lines with rapid change of tangent vector as in Figure 5-8) on the resulting surface as shown in Figure 5-7 and Figure 5-8.

The second method does not require that the data points have to be processed so that they are considered to be on iso-parametric lines. Any set of curves can be used for surface reconstruction. Lofting can be treated as a special case of this approach. The detailed explanation of this more general approach is given in Chapter 3.

5.4 Complexity Analysis of the System

The analysis of the system can be categorized into two phases: the first phase includes data acquisition/registration and the second phase includes a surface creation. The time for the data acquisition/registration phase depends on the size of the plate, moving velocity of the robot arm and digitization speed of the scanner system, meaning that it is proportional only to the size of the plate because the other parameters are fixed machine specific-values.

The second phase contains algorithms for surface reconstruction and creation. As explained in section 3.6, setting up Equation (3.14) requires $O(n_1n_u^3n_v^3(n_u^2 + n_v^2))$ for the Bézier surface patch case and for the B-spline surface patch case, setting up
Equation (3.37) takes $O(n_I n_u n_v)$ approximately where $n_u$ and $n_v$ are the numbers of control points in $u$ and $v$ directions, respectively and $n_I$ (after subdivision into Bézier curves) is the number of input curves. The last step in constructing a surface is to solve the linear system of equations such as Equation (3.14) or Equation (3.37).
Chapter 6

Applications and Examples

6.1 Introduction

The performance of the new surface approximation method based on the concept of a curve on a surface is demonstrated with some examples. Two applications of the new approach are examined: one is surface reconstruction and the other is surface creation. Artificial and real examples show that it can be used for both purposes. The system is tested with a plate produced by the line heating process to assess its performance.

6.2 Surface Reconstruction

6.2.1 Artificial Data

To demonstrate the performance of the new surface reconstruction algorithm, numerical experiments with artificial data have been carried out. The wave-like surface is used as a model surface whose peak height is 0.1333 and trough height is −0.1333 as shown in Figure 6-1. The model surface is a bicubic B-spline with a 11 × 11 control net and uniform knots \( U = V = (0, 0, 0, \frac{1}{8}, 1, \frac{1}{4}, 1, \frac{3}{8}, 1, \frac{5}{8}, 1, \frac{7}{8}, 1, 1, 1, 1) \). Therefore the surface is decomposed into 64 Bézier patches during the approximation process.

For input to the approximation, 16 stripes and 4 boundary curves are extracted from the model surface. Each stripe is constructed from 81 data points that are sampled uniformly along the pre-image of the stripe in the parametric domain of the model surface as illustrated in Figure 6-1(a) and 6-1(b). The stripes are approximated by cubic B-spline curves of 20 control points with 16 uniform knots. The 16 B-spline stripes are first subdivided into 272 Bézier curves and further subdivided into 426 Bézier curves at the intersections with the lattice of isoparametric lines at each internal knot of the surface. Therefore the size of matrices \([\beta_1]\) and \([\beta_2]\) shown in section 3.5 are 2982 × 40, respectively. The reconstructed B-spline surface has the same number of control points and the same knot vectors as those of the model surface.

The maximum, minimum, average and standard deviation between the model surface and the reconstructed surface are evaluated on a 50 × 50 grid. Figure 6-1(e)
shows the signed difference on those grid points (for the sake of clarity, the number of grid points are reduced for visualization).

### 6.2.2 Measured Data

A bicubic Bézier patch from the measured data points from a plate formed by line heating has been reconstructed as shown in Figure 6-2. The data points form stripes which are approximated into Bézier curves also shown in Figure 6-2, which are used as an input to the reconstruction process. Comparison between Figure 6-2 and 5-7(b) shows that the new approach produces a better result than the lofting procedure in that the isoparametric lines of the surface are smooth.

### 6.3 Surface Creation

A car hood is chosen as an example for demonstrating surface creation for the new method. Four boundary curves and 17 stripes are generated (as if they are sketched) and then given as input to the surface creation method. Due to the symmetry of the hood, only the left part of the hood is considered. The knot vector of the 4 boundary curves is chosen to be \((0, 0, 0, 0, \frac{1}{5}, \frac{2}{5}, \frac{3}{5}, 1, 1, 1, 1, 1)\). Consequently, the reconstructed surface has the same knot vector in both \(u\) and \(v\) directions, resulting in 25 Bézier patches. The dotted lines in Figure 6-5(a) show the subdivided 25 Bézier patches. The 17 curves also shown in Figure 6-5(a) are subdivided into 37 Bézier curve segments and provided as input to the algorithm for the creation of the final surface. The base surface shown in Figure 6-5(b) is created by bilinearly blending the four boundary curves based on Coons surface interpolation. Figure 6-5(c) depicts the shaded image of the created surface.

### 6.4 Engineering Example of the System

Two practical examples which demonstrate the performance of the system are presented: a part of a fuselage and a rectangular plate.

A part of a fuselage has been reconstructed as shown in Figure 6-7. The input curves and boundary curves are given as shown in Figure 6-7(a). The four boundary curves are represented as cubic B-spline curves with 5 control points and a knot vector \((0, 0, 0, 0, 0.5, 1, 1, 1, 1, 1, 1, 1, 1, 1)\) from which the base surface of Figure 6-7(b) has been obtained through Coons surface interpolation and shows four Bézier surface patches that are approximated and combined into a B-spline surface patch with 25 control points and a knot vector \((0, 0, 0, 0, 0.5, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)\) in u and v directions. The 7 input curves are subdivided into 57 Bézier curve segments and provided as input to the algorithm. Figure 6-7(c) depicts the final reconstructed surface and Figure 6-7(d) shows isophotes of the approximated surface [1].

A plate that has been produced by line heating process in the MIT Ocean Engineering Fabrication Laboratory [47, 45, 46, 5] is chosen for verification of the system.
Figure 6-1: Surface reconstruction of the model surface
Before forming, the plate is a rectangular, flat and smooth steel plate, whose dimensions are $30.5\,cm \times 30.5\,cm \times 0.6\,cm$ as shown in Figure 6-8. Its shape after forming satisfies the geometric and topological assumptions made in developing the system. To maximize the digitization performance of the scanner, the surface of the plate has been sprayed with gray paint.

The detected boundary of the plate is shown in Figure 6-9. From the boundary information, the number of shots, that need to be taken to capture the entire surface of the plate, is estimated. In this example, 9 shots were taken to sweep the plate in a zig-zag manner. The surface is smoothly varying so that it was digitized with no overlap region between any two consecutive shots. The data points were registered with respect to the global coordinate system to represent the surface of the plate. The data were processed by using Equation (5.1) with $d_x = 140 \times \sin(\alpha)$, $d_y = 140 \times \cos(\alpha)$ and $\alpha$ less than 45 degrees and we have the digitized surface of the plate as shown in Figure 6-10.

Each stripe can be approximated in either a B-spline or a Bézier form depending on the shape of the stripe. In this example, the stripes are simple and smooth enough to be well represented in a cubic Bézier form. Fifty-four cubic Bézier curves are selected for the input to the approximation in order to reduce computational overhead. The four boundary curves are cubic B-spline curves with 8 control points and uniform knots $T = (0, 0, 0, 0, 0.2, 0.4, 0.6, 0.8, 1, 1, 1, 1, 1)$ and the stripes are cubic Bézier curves.

A bicubic B-spline surface is generated by linearly blending the four boundary
curves based on Coons surface interpolation, which is decomposed into 25 Bézier patches during the approximation procedure.

The approximated surface is given in Figure 6-10, which shows a better result over the surface from the lofting procedure shown in Figure 5-8 in that it has smooth isoparametric lines.

The plate was also measured with a coordinate measuring machine (CMM) device which is a contact measurement machine. The measured data points from the device are regarded as a reference for assessing the quality of the reconstructed surface. A number of 1169 data points have been sampled as shown in Figure 6-12.

Localization [1, 6] is done before comparison, which is the preprocessing of verification of shape conformance of the reconstructed surface to the plate. The result of localization is given in Figure 6-12 along with the reconstructed surface. After localization, the maximum distance between the surface and the data points is calculated. The given example shows the maximum distance is 1.49mm.

### 6.5 Experimental Complexity Analysis

#### 6.5.1 Bézier Surface Approximation

The Bézier surface approximation given in Equation (3.10) is tested with an real example of Figure 6-2 to verify that the numerical results follow the theoretical analysis in section 3.6 and compare the effect when Equation (2.27) is used.

The time complexity for Bézier surface approximation is $O(n_1 n_2^3 n^2(n_1^2 + n_2^2))$ when Equation (2.31) is used (case (2)) as given in section 3.6. On the other hand when Equation (2.27) is used (case (1)), then the time complexity becomes $O(n_1 n_2^4 n^3(n_1 + n_2))$. In this experiment, we used a fixed number of input curves $n_1 = 54$ so that the time complexity depends only on $n_1$ and $n_2$ where $n_1$ and $n_2$ are degrees of the Bézier surface in $u$ and $v$ directions. For Bézier representation, the number of control points is proportional to the degree. Let us denote the numbers of control points in $u$ and $v$ directions as $n_u$ and $n_v$. For simplicity, the same number of control points is used in both directions for each case, i.e. $n_u = n_v = n$. The numbers of control points vary from 64 to 625 and the average execution times are recorded in Table 6.1. An SGI machine (200MHz MIPS R5000 CPU, 128Mbyte RAM) was used for this numerical experiment.

<table>
<thead>
<tr>
<th>Control Points</th>
<th>case (1) Eqn (2.27)</th>
<th>case (2) Eqn (2.31)</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>15.2</td>
<td>16.8</td>
</tr>
<tr>
<td>100</td>
<td>37.5</td>
<td>47.8</td>
</tr>
<tr>
<td>225</td>
<td>335.8</td>
<td>607.3</td>
</tr>
<tr>
<td>400</td>
<td>1983.1</td>
<td>4664.6</td>
</tr>
<tr>
<td>625</td>
<td>8304.1</td>
<td>24003.4</td>
</tr>
</tbody>
</table>

Table 6.1: Elapsed time for each execution at various numbers of control points

Let us assume that the total number of control points $n^2 (= n_u n_v)$ is denoted as $x$
and the elapsed time as $t$. The plot of $x$ vs. $t$ is given in Figure 6-3(a) and Figure 6-3(b) is drawn in log scale. In those figures, the dashed line is the case where Equation (2.27) is used and the solid line is the case where Equation (2.31) is used.

![Figure 6-3](image)

(a)  

(b)

Figure 6-3: (a) Plot of elapsed time vs. the number of control points (b) Plot in log scale.

The data points in Figure 6-3 (b) are well approximated with a straight line

$$Y = \alpha + \beta X$$

where $Y$ is $\log t$ and $X$ is $\log x$. The slopes of the lines $\beta$ are 2.7853 and 3.2123 from the least squares method. It means that the elapsed time $t$ is proportional to $x^{2.7853}$ and $x^{3.2123}$, i.e. $n^{5.5766}$ and $n^{6.4246}$. The experimental results turn out to be better than the theoretical analysis. The reason is that the numbers of control points used for the experiment are not big enough to reflect the asymptotic behavior. From the theoretical analysis, however, it can be expected that as the number of control points increases, the time complexity will be bounded by $O(n^8)$ (generally $O(n_u^3n_v^3(n_u^2 + n_v^2))$) for case (2) and by $O(n^7)$ (generally $O(n_u^3n_v^3(n_u + n_v))$) for case (1) asymptotically.

Table 6.2 shows the difference between the two calculation methods of Bernstein polynomials, i.e. Equations (2.27) and (2.31). After getting the resulting surfaces using both methods, the distances between the corresponding control points of both reconstructed surfaces were calculated and the root mean square (RMS) values as well as the maximum values of these distances are recorded for each case. The table indicates that as the degree of the Bézier surface increases, the RMS and maximum values also increase, which means that depending on the methods for calculating Bernstein polynomials, the results may become different.

### 6.5.2 B-spline Surface Approximation

B-spline surface approximation is tested with a real example to verify that the numerical results follow the theoretical analysis.
The complexity for B-spline surface approximation is $O(rinrin)$ as discussed in section 3.6. Seventy four input curves are used for this numerical experiment. Therefore, the time complexity becomes $O(nu_nv_v)$. The surface for the test is the plate shown in Figure 6-10. Table 6.3 shows the elapsed time of the program for various numbers of control points.

<table>
<thead>
<tr>
<th>Control Points</th>
<th>Elapsed Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>11.7</td>
</tr>
<tr>
<td>121</td>
<td>14.1</td>
</tr>
<tr>
<td>196</td>
<td>19.2</td>
</tr>
<tr>
<td>289</td>
<td>21.5</td>
</tr>
<tr>
<td>400</td>
<td>25.1</td>
</tr>
<tr>
<td>529</td>
<td>29.1</td>
</tr>
</tbody>
</table>

Table 6.3: Elapsed time for each execution at various numbers of control points

The data points are well approximated with a straight line whose slope is 0.0368 as shown in Figure 6-4. This means that the elapsed time is proportional to the number of control points of the reconstructed B-spline surface patch, which is consistent with the result obtained by the theoretical analysis given in section 3.6.
Figure 6-5: The left side of a car hood created from sketch curves
Figure 6-6: Interrogation of the reconstructed surface
Figure 6-7: Reconstruction of fuselage
Figure 6-8: The model plate

Figure 6-9: Detected boundary
Figure 6-10: Scanned data points and reconstructed surface
Figure 6-11: Interrogation of the reconstructed surface
Figure 6-12: Reconstructed surface and CMM data points
Chapter 7

Conclusions and Future Work

7.1 Conclusions and Contributions

In this thesis, an automated system, which digitizes the surface of a topologically rectangular plate manufactured from line heating process and produces a suitable CAD representation, has been developed. It consists of three subsystems: a scanner, a robot and a UNIX workstation. The scanner digitizes the surface of a plate. The limitation that the field of view of the scanner is much smaller than the size of the plate is overcome by attaching the scanner to the robot which has a sufficiently large working envelope and moving the robot to appropriate positions to cover the whole surface. The workstation analyzes data coming from the scanner and the robot and issues various commands for real time operations of the scanner and the robot. The computer and the laser scanner communicate with each other via the internet line (TCP/IP) and the computer and the robot communicate through a serial cable (RS-232C).

The process of the system starts with boundary estimation. The scanner tracks the boundary edges of the plate and extracts the boundary lines. The assumption made for developing the system that the plate is topologically rectangular simplifies the boundary extraction algorithm. After the whole boundary is estimated, the system scans the entire surface following a zig-zag path planned based on the boundary information. The scanned data points are registered with respect to a common coordinate system and provided to the surface reconstruction algorithm. The reconstructed surface is given to the line heating algorithm for further analysis.

2D to 3D conversion algorithm, which is a novel method for surface creation, has been proposed. The concept of a curve on a surface has been extended to surface reconstruction. Curves in either Bézier or B-spline form are used for input. When B-spline curves are provided, they are subdivided into Bézier curve segments through knot insertion and then given as input. The input curves are parameterized as straight lines in the parametric domain and processed to satisfy the condition of a curve on a surface. This leads to a system of equations for control points of the resulting
Bézier surface patch, which is solved in the least squares sense. This process requires \( O(n_1n_2^3n_3^2(n_1^2+n_2^2)) \) time for setting up the system of equations and it takes \( O(n_1n_2^3n_3^2 + n_1^3n_2^3) \) time to solve the system of equations, where \( n_f \) is the number of input curves, \( n_1 \) and \( n_2 \) are degrees of Bézier surface patch in \( u \) and \( v \) directions, respectively. Unlike lofting, this new approach does not require that input curves have to be isoparametric lines. Therefore, it is a general scheme which contains lofting as a special case. B-spline surface approximation involves additional steps. A base surface in B-spline form is generated from the boundary lines. Then it is subdivided into cubic Bézier surface patches by knot insertion in \( O(n_2n_3) \). Setting up the global system of equations for finding the control points of the resulting B-spline surface takes \( O(n_fn_un_v) \) time where \( n_f \) is the number of subdivided input curves and \( n_u \) and \( n_v \) are the numbers of control points of the resulting B-spline surface in \( u \) and \( v \) directions. Solving the system of equations requires \( O(n_2^3n_3n_v) \) for normalization and \( O(n_fn_un_v) \) for the Cholesky decomposition and calculation of the solution where \( n_f \) is the number of the subdivided input curves. If the same input curves are used with small variations, the normalization of the system of equations reduces to \( O(n_2^2) \) in the subsequent phase so that we can find the result faster.

Examples given in Chapter 6 demonstrate the feasibility of the new method for surface creation as well as surface reconstruction. The proposed system uses the new scheme for surface reconstruction and produces a better result than the lofting procedure in that the isoparametric lines of the reconstructed surface are smoother.

### 7.2 Future Work

A thorough investigation of accuracy and repeatability of the system needs to be performed. They have a direct impact on the quality of the resulting surface. A systematic test of the system needs to be done and errors involved in each motion and algorithm have to be analyzed.

An assumption has been made to develop the system that the plate of interest is topologically rectangular. This simplifies the edge detection algorithm. On the other hand, this simplification prevents the system from handling more general cases. Further research needs to be directed toward improving the algorithms for edge detection and digitization. First, the edge detection algorithm has to be extended to cover objects of more general shapes and at the same time positioning of the scanner needs to be improved. Second, the path planning for surface scanning has to be optimized by assessing the geometric properties conveyed by the data points from the previous scan and determining the best position for the next view of the scanner. An adaptive path planning scheme, which improves the surface digitization process would be advantageous.

The parameterization method which is included in the surface approximation needs more investigation. The current method involves approximation of the stripes in 3D space as straight lines in the parametric domain because the surface of interest is smooth so that the digitized stripes almost seem to be straight lines. However, in general, such assumption may not be applicable to the case where each stripe shows
significant three-dimensionality. The current algorithm needs to be extended to deal with such a more general case.

The stability of computation is another aspect worthy of study. The influence of the distribution of the input stripes and the impact of path selection on the stability of computation need to be studied in some detail.
Bibliography


