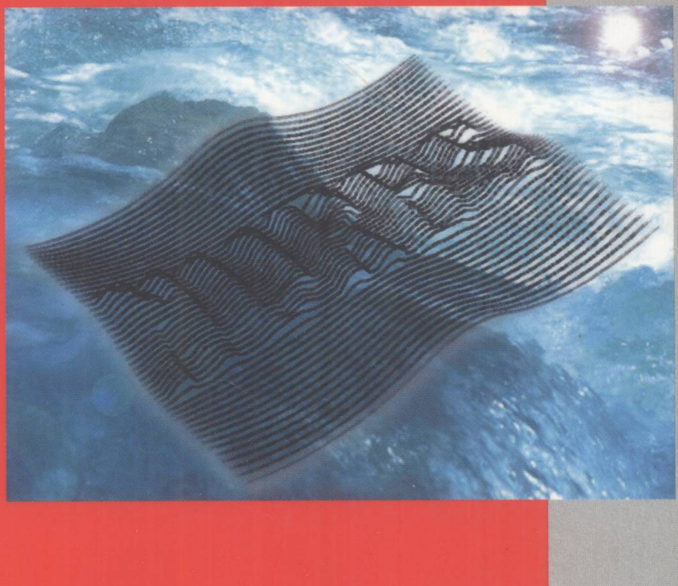
Les Piegl
Wayne Tiller

The *NURBS* Book

2nd Edition



MONOGRAPHS IN VISUAL COMMUNICATION

Springer

Les Piegl
Wayne Tiller

The *NURBS* Book

Second Edition
with 334 Figures in 578 Parts

Springer

Prof. Dr. Les Piegl
University of South Florida
Department of Computer Science and Engineering
Fowler Avenue, ENG 118
Tampa, FL 33620-5399, USA

Dr. Wayne Tiller
GeomWare, Inc.
3036 Ridgetop Road
Tyler, TX 75703-8906, USA

ISBN 3-540-61545-8 2nd ed. Springer-Verlag Berlin Heidelberg New York
ISBN 3-540-55069-0 1st ed. Springer-Verlag Berlin Heidelberg New York

Cip data applied for
Die Deutsche Bibliothek - CIP-Einheitsaufnahme
Piegl, Les: The NURBS book / Les Piegl ; Wayne Tiller. - 2. ed. -
Berlin ; Heidelberg ; New York ; Barcelona ; Budapest ; Hong Kong ; London ;
Milan ; Paris ; Tokyo : Springer, 1997
(Monographs in visual communications)
ISBN 3-540-61545-8
NE: Tiller, Wayne

Springer-Verlag Berlin Heidelberg New York
a member of Springer Science+Business Media

Cover design: Design & Production, Heidelberg
Typesetting: Nancy A. Rogers, NAR Assoc., Annapolis, MD, using $\TeX$
SPIN: 12543724 33/3180 - Printed on acid-free paper

To the memory of my mother Anna, and to my father János

L.P.

To my grandmother, Fern Melrose Bost, and to the memories of my grandparents, James Raney Bost, Pearl Weeks Tiller, and Albert Carroll Tiller

W.T.

FOREWORD

Until recently B-spline curves and surfaces (NURBS) were principally of interest to the computer aided design community, where they have become the standard for curve and surface description. Today we are seeing expanded use of NURBS in modeling objects for the visual arts, including the film and entertainment industries, art, and sculpture. NURBS are now also being used for modeling scenes for virtual reality applications. These applications are expected to increase. Consequently, it is quite appropriate for *The NURBS Book* to be part of the *Monographs in Visual Communication* Series.

B-spline curves and surfaces have been an enduring element throughout my professional life. The first edition of *Mathematical Elements for Computer Graphics*, published in 1972, was the first computer aided design/interactive computer graphics textbook to contain material on B-splines. That material was obtained through the good graces of Bill Gordon and Louie Knapp while they were at Syracuse University. A paper of mine, presented during the Summer of 1977 at a Society of Naval Architects and Marine Engineers meeting on computer aided ship surface design, was arguably the first to examine the use of B-spline curves for ship design.

For many, B-splines, rational B-splines, and NURBS have been a bit mysterious. Consequently, for the last several years a thorough, detailed, clearly written, and easily understood book on B-splines and rational B-splines has been needed. Thus, it was with considerable anticipation that I awaited Les Piegl and Wayne Tiller's book. I was not disappointed: They have elegantly and fully satisfied that need with *The NURBS Book*. In developing the material for the book, they draw from their considerable academic and industrial experience with NURBS to present this rather complex subject in a straightforward manner: Their presentation style is clear and detailed. The necessary mathematics is presented with considerable attention to detail and more than adequate rigor. The algorithms (many of which are in C-like pseudocode) are well thought out and meticulously prepared. In the interests of accuracy, each and every illustration in the book was computer generated – a monumental task. They have created a book of lasting value.

B-spline curves and surfaces grew out of the pioneering work of Pierre Bézier in the early 1970s. Perhaps one can consider B-spline curves and surfaces the children of Bézier curves and surfaces, and nonuniform rational B-splines, or NURBS, the grandchildren. The timing is about right; they have certainly come of age.

Finally, it is only appropriate to acknowledge my pleasure in working with both Les Piegl and Wayne Tiller to bring this project to fruition.

David F. Rogers
Series Editor
Monographs in Visual Communication

PREFACE

Non-Uniform **R**ational **B-S**plines, commonly referred to as **NURBS,** have become the de facto industry standard for the representation, design, and data exchange of geometric information processed by computers. Many national and international standards, e.g., IGES, STEP, and PHIGS, recognize NURBS as powerful tools for geometric design. The enormous success behind NURBS is largely due to the fact that

- NURBS provide a unified mathematical basis for representing both analytic shapes, such as conic sections and quadric surfaces, as well as free-form entities, such as car bodies and ship hulls;
- designing with NURBS is intuitive; almost every tool and algorithm has an easy-to-understand geometric interpretation;
- NURBS algorithms are fast and numerically stable;
- NURBS curves and surfaces are invariant under common geometric transformations, such as translation, rotation, parallel and perspective projections;
- NURBS are generalizations of nonrational B-splines and rational and nonrational Bézier curves and surfaces.

The excellent mathematical and algorithmic properties, combined with successful industrial applications, have contributed to the enormous popularity of NURBS. NURBS play a role in the CAD/CAM/CAE world similar to that of the English language in science and business: "Want to talk business? Learn to talk NURBS".

The purpose of this book is basically twofold: to fill a large gap in the literature that has existed since the early seventies, and to provide a comprehensive reference on all aspects of NURBS. The literature on NURBS is sparse and scattered, and the available papers deal mainly with the mathematics of splines, which is fairly complex and requires a detailed understanding of spline theory. This book is aimed at the average engineer who has a solid background in elementary college mathematics and computing. No doctoral degree is required to understand the concepts and to implement the literally hundreds of algorithms that are introduced.

During the four years of writing this book, we have

- surveyed the available literature and presented important results;
- continued our research on NURBS and included the latest developments; in fact, about half of the book contains new material developed in the last few years;
- developed a comprehensive NURBS library, called *Nlib V1.0, V2.0*. This library is the result of over 20 man-years of experience in NURBS research and development, and it combines new and well-tried software practices applied in previous systems that we designed;

- tested every single formula and algorithm, and presented graphical illustrations precisely computed using the routines of *Nlib*. **This book does not contain any hand-drawn figures; each figure is precisely computed and hence is accurate.**

We are pleased to present all of the accomplishments to the reader: (1) the book as a comprehensive reference, (2) *Nlib* source code (to order please see page 639 of this volume), and (3) the illustrations to instructors who adopt the book to teach a course on NURBS. In order for the reader to appreciate the enormous amount of work that went into this reference book, we present some data. To generate the graphical illustrations and to build *Nlib*, we wrote exactly (not counting the hundreds of test programs)

- **1,524** programs, that required
- **15,001,600** bytes of storage, which is roughly equivalent to
- **350,000** lines of code.

It was no picnic!

Some years ago a few researchers joked about NURBS, saying that the acronym really stands for **N**obody **U**nderstands **R**ational **B-S**plines. We admit that our colleagues were right. In the last four years, we were largely influenced by this interpretation and tried to present the material in the book in an intuitive manner. We hope that this helps change the acronym NURBS to EURBS, that is, **E**verybody **U**nderstands **R**ational **B-S**plines. We welcome the reader's opinion on our job and suggestions on possible improvements.

It is our pleasure to acknowledge the help and support of many people and organizations. First and foremost, we are grateful to our spouses, Karen Piegl and LaVella Tiller, for their patience, support, and love. We owe special thanks to Nancy Rogers of NAR Associates for the beautiful typesetting job, and David Rogers for the editorial and technical discussions that led to many improvements in the manuscript. We also thank Jim Oliver and Tim Strotman for the many suggestions and technical correspondence that helped shape this book into its current form. Tiller also thanks the many past and present colleagues in industry who over the years contributed inspiring discussions, valuable insights, support, and collegial companionship: They know who they are. Piegl's research was supported in part by the National Science Foundation under grant CCR-9217768 awarded to the University of South Florida, and by various grants from the Florida High Technology and Industry Council.

March 1995

Les Piegl
Wayne Tiller

It is less than a year since the first printing of *The NURBS Book*. Due to its popularity, Springer-Verlag decided to publish a soft cover edition of the book. Apart from being significantly more affordable, the second printing corrects a number of errors; redesigns Algorithm `A3.5` to eliminate the use of a local array; and fixes minor bugs in the knot insertion algorithms, `A5.1` and `A5.3`, as well the degree elevation algorithm, `A5.9`. Apart from these corrections, this printing is identical to the first printing.

July 1996

Les Piegl
Wayne Tiller

CONTENTS

Chapter Five Fundamental Geometric Algorithms

Chapter Six Advanced Geometric Algorithms

Chapter Seven Conics and Circles

Chapter Eight Construction of Common Surfaces

Chapter Nine Curve and Surface Fitting

CHAPTER

ONE

Curve and Surface Basics

1.1 Implicit and Parametric Forms

The two most common methods of representing curves and surfaces in geometric modeling are implicit equations and parametric functions.

The implicit equation of a curve lying in the xy plane has the form $f(x,y) = 0$. This equation describes an implicit relationship between the x and y coordinates of the points lying on the curve. For a given curve the equation is unique up to a multiplicative constant. An example is the circle of unit radius centered at the origin, specified by the equation $f(x,y) = x^2 + y^2 - 1 = 0$ (Figure 1.1).

In parametric form, each of the coordinates of a point on the curve is represented separately as an explicit function of an independent parameter

$$\mathbf{C}(u) = \big(x(u), y(u)\big) \qquad a \le u \le b$$

Thus, $\mathbf{C}(u)$ is a vector-valued function of the independent variable, u. Although the interval $[a,b]$ is arbitrary, it is usually normalized to $[0,1]$. The first quadrant of the circle shown in Figure 1.1 is defined by the parametric functions

$$\begin{aligned} x(u) &= \cos(u) \\ y(u) &= \sin(u) \qquad 0 \le u \le \frac{\pi}{2} \end{aligned} \tag{1.1}$$

Setting $t = \tan(u/2)$, one can derive the alternate representation

$$\begin{aligned} x(t) &= \frac{1-t^2}{1+t^2} \\ y(t) &= \frac{2t}{1+t^2} \qquad 0 \le t \le 1 \end{aligned} \tag{1.2}$$

Thus, the parametric representation of a curve is not unique.

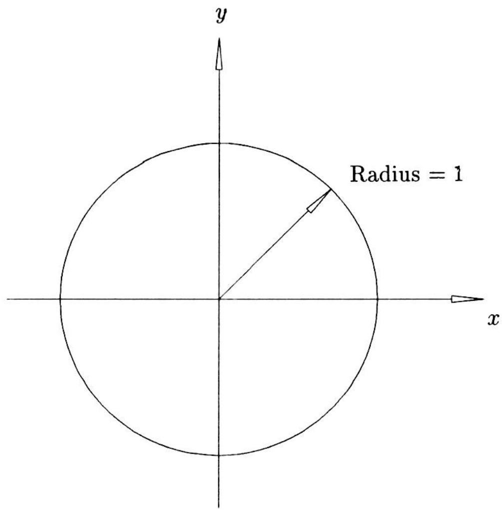


Figure 1.1. A circle of radius 1, centered at the origin.

It is instructive to think of $\mathbf{C}(u) = \big(x(u), y(u)\big)$ as the path traced out by a particle as a function of time; u is the time variable, and $[a,b]$ is the time interval. The first and second derivatives of $\mathbf{C}(u)$ are the velocity and acceleration of the particle, respectively. Differentiating Eqs. (1.1) and (1.2) once yields the velocity functions

$$\mathbf{C}'(u) = \big(x'(u), y'(u)\big) = \big(-\sin(u), \cos(u)\big)$$

$$\mathbf{C}'(t) = \big(x'(t), y'(t)\big) = \left(\frac{-4t}{(1+t^2)^2}, \frac{2(1-t^2)}{(1+t^2)^2}\right)$$

Notice that the magnitude of the velocity vector, $\mathbf{C}'(u)$, is a constant

$$|\mathbf{C}'(u)| = \sqrt{\sin^2(u) + \cos^2(u)} = 1$$

i.e., the direction of the particle is changing with time, but its speed is constant. This is referred to as a *uniform parameterization*. Substituting $t = 0$ and $t = 1$ into $\mathbf{C}'(t)$ yields $\mathbf{C}'(0) = (0,2)$ and $\mathbf{C}'(1) = (-1,0)$, i.e., the particle's starting speed is twice its ending speed (Figure 1.2).

A surface is defined by an implicit equation of the form $f(x,y,z) = 0$. An example is the sphere of unit radius centered at the origin, shown in Figure 1.3 and specified by the equation $x^2 + y^2 + z^2 - 1 = 0$. A parametric representation (not unique) of the same sphere is given by $\mathbf{S}(u,v) = \big(x(u,v), y(u,v), z(u,v)\big)$, where

$$\begin{aligned} x(u,v) &= \sin(u)\cos(v) \\ y(u,v) &= \sin(u)\sin(v) \\ z(u,v) &= \cos(u) \qquad 0 \le u \le \pi, \ \ 0 \le v \le 2\pi \end{aligned} \tag{1.3}$$

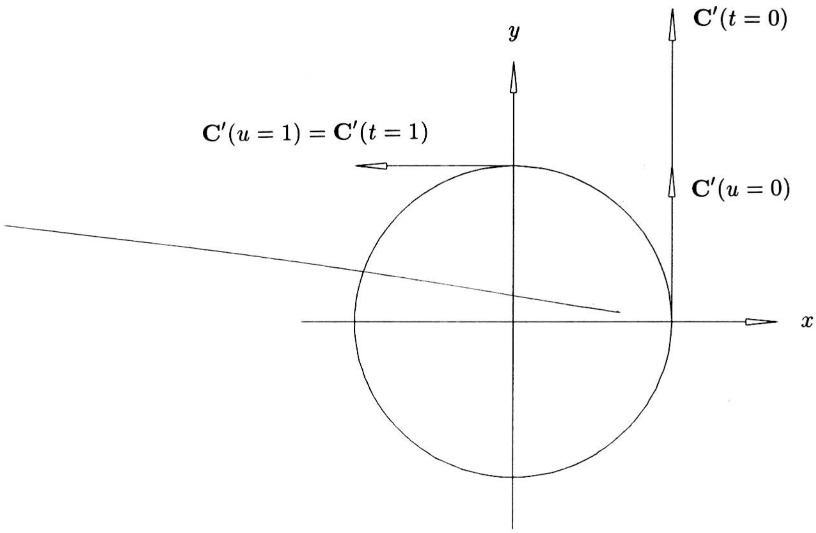


Figure 1.2. Velocity vectors $\mathbf{C}'(u)$ and $\mathbf{C}'(t)$ at $u, t = 0$, and 1.

Notice that two parameters are required to define a surface. Holding u fixed and varying v generates the latitudinal lines of the sphere; holding v fixed and varying u generates the longitudinal lines.

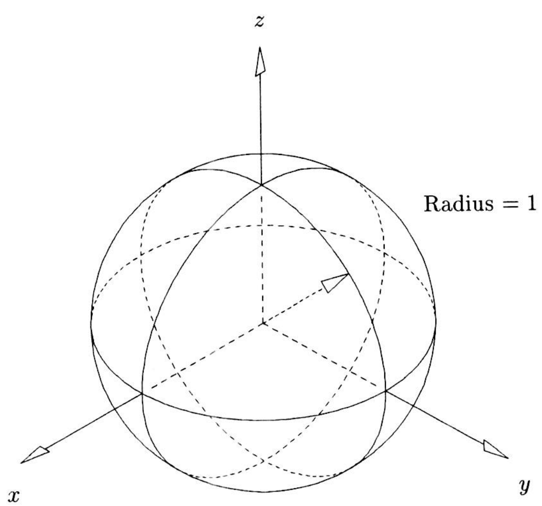


Figure 1.3. A sphere of radius 1, centered at the origin.

Denote the partial derivatives of $\mathbf{S}(u,v)$ by $\mathbf{S}_u(u,v) = (x_u(u,v), y_u(u,v), z_u(u,v))$ and $\mathbf{S}_v(u,v) = (x_v(u,v), y_v(u,v), z_v(u,v))$, i.e., the velocities along latitudinal and longitudinal lines. At any point on the surface where the vector cross product $\mathbf{S}_u \times \mathbf{S}_v$ does not vanish, the unit normal vector, $\mathbf{N}$, is given by (Figure 1.4)

$$\mathbf{N} = \frac{\mathbf{S}_u \times \mathbf{S}_v}{|\mathbf{S}_u \times \mathbf{S}_v|} \tag{1.4}$$

The existence of a normal vector at a point, and the corresponding tangent plane, is a geometric property of the surface independent of the parameterization. Different parameterizations give different partial derivatives, but Eq. (1.4) always yields $\mathbf{N}$ provided the denominator does not vanish. From Eq. (1.3) it can be seen that for all v, $0 \leq v \leq 2\pi$, $\mathbf{S}_v(0,v) = \mathbf{S}_v(\pi,v) = 0$, that is, $\mathbf{S}_v$ vanishes at the north and south poles of the sphere. Clearly, normal vectors do exist at the two poles, but under this parameterization Eq. (1.4) cannot be used to compute them.

Of the implicit and parametric forms, it is difficult to maintain that one is always more appropriate than the other. Both have their advantages and disadvantages. Successful geometric modeling is done using both techniques. A comparison of the two methods follows:

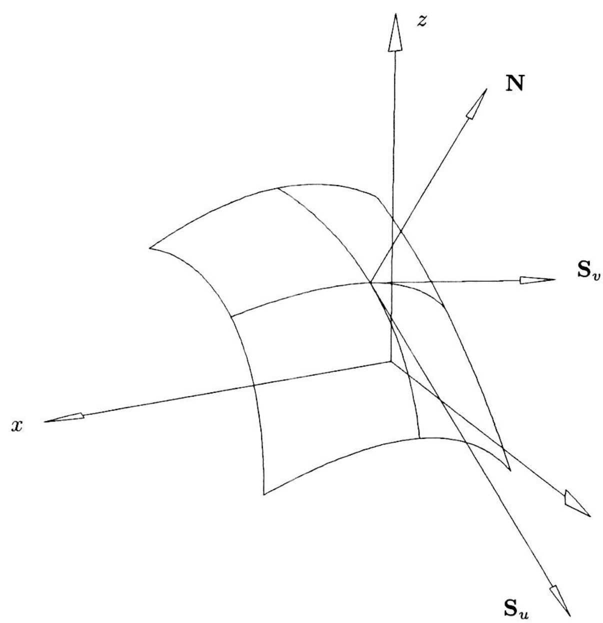


Figure 1.4. Partial derivative and unit normal vectors of $\mathbf{S}(u,v)$.

- By adding a z coordinate, the parametric method is easily extended to represent arbitrary curves in three-dimensional space, $\mathbf{C}(u) = \big(x(u), y(u), z(u)\big)$; the implicit form only specifies curves in the xy (or xz or yz) plane;
- It is cumbersome to represent bounded curve segments (or surface patches) with the implicit form. However, boundedness is built into the parametric form through the bounds on the parameter interval. On the other hand, unbounded geometry (e.g., a simple straight line given by $f(x, y) = ax + by + c = 0$) is difficult to implement using parametric geometry;
- Parametric curves possess a natural direction of traversal $\big($from $\mathbf{C}(a)$ to $\mathbf{C}(b)$ if $a \leq u \leq b\big)$; implicit curves do not. Hence, it is easy to generate ordered sequences of points along a parametric curve. A similar statement holds for generating meshes of points on surfaces;
- The parametric form is more natural for designing and representing shape in a computer. The coefficients of many parametric functions, e.g., Bézier and B-spline, possess considerable geometric significance. This translates into intuitive design methods and numerically stable algorithms with a distinctly geometric flavor;
- The complexity of many geometric operations and manipulations depends greatly on the method of representation. Two classic examples are:
  - compute a point on a curve or surface – difficult in the implicit form;
  - given a point, determine if it is on the curve or surface – difficult in the parametric form;
- In the parametric form, one must sometimes deal with parametric anomalies which are unrelated to true geometry. An example of this is the unit sphere (see Eq.[1.3]). The poles are parametric critical points which are algorithmically difficult, but geometrically the poles are no different than any other point on the sphere.

We are concerned almost exclusively with parametric forms in the remainder of this book. More details on implicit and parameteric forms can be found in standard texts $\big($[Faux81; Mort85; Hoff89; Beac91]$\big)$.

1.2 Power Basis Form of a Curve

Clearly, by allowing the coordinate functions $x(u)$, $y(u)$, and $z(u)$ to be arbitrary, we obtain a great variety of curves. However, there are trade-offs when implementing a geometric modeling system. The ideal situation is to restrict ourselves to a class of functions which

- are capable of precisely representing all the curves the users of the system need;
- are easily, efficiently, and accurately processed in a computer, in particular:
  - the computation of points and derivatives on the curves is efficient;

  - numerical processing of the functions is relatively insensitive to floating point round-off error;
  - the functions require little memory for storage;
- are simple and mathematically well understood.

A widely used class of functions is the polynomials. Although they satisfy the last two criteria in this list, there are a number of important curve (and surface) types which cannot be precisely represented using polynomials; these curves must be approximated in systems using polynomials. In this section and the next, we study two common methods of expressing polynomial functions, power basis and Bézier. Although mathematically equivalent, we will see that the Bézier method is far better suited to representing and manipulating shape in a computer.

An nth-degree power basis curve is given by

$$\mathbf{C}(u) = \big(x(u), y(u), z(u)\big) = \sum_{i=0}^{n} \mathbf{a}_i u^i \qquad 0 \le u \le 1 \tag{1.5}$$

The $\mathbf{a}_i = (x_i, y_i, z_i)$ are vectors, hence

$$x(u) = \sum_{i=0}^{n} x_i u^i \quad y(u) = \sum_{i=0}^{n} y_i u^i \quad z(u) = \sum_{i=0}^{n} z_i u^i$$

In matrix form Eq. (1.5) is

$$\mathbf{C}(u) = [\mathbf{a}_0 \quad \mathbf{a}_1 \quad \cdots \quad \mathbf{a}_n] \begin{bmatrix} 1 \\ u \\ \vdots \\ u^n \end{bmatrix} = [\,\mathbf{a}_i\,]^T [\,u^i\,] \tag{1.6}$$

(We write a row vector as the transpose of a column vector.)

Differentiating Eq. (1.5) yields

$$\mathbf{a}_i = \frac{\mathbf{C}^{(i)}(u)|_{u=0}}{i!}$$

where $\mathbf{C}^{(i)}(u)|_{u=0}$ is the ith derivative of $\mathbf{C}(u)$ at $u = 0$. The $n+1$ functions, $\{u^i\}$, are called the basis (or blending) functions, and the $\{\mathbf{a}_i\}$ the coefficients of the power basis representation.

Given u_0, the point $\mathbf{C}(u_0)$ on a power basis curve is most efficiently computed using Horner's method

- for degree $= 1: \ \mathbf{C}(u_0) = \mathbf{a}_1 u_0 + \mathbf{a}_0$
- degree $= 2: \ \mathbf{C}(u_0) = (\mathbf{a}_2 u_0 + \mathbf{a}_1)\, u_0 + \mathbf{a}_0$
- $\vdots$
- degree $=$ n $: \ \mathbf{C}(u_0) = \big((\cdots(\mathbf{a}_n u_0 + \mathbf{a}_{n-1})\, u_0 + \mathbf{a}_{n-2}\big)\, u_0 + \cdots + \mathbf{a}_0$