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.

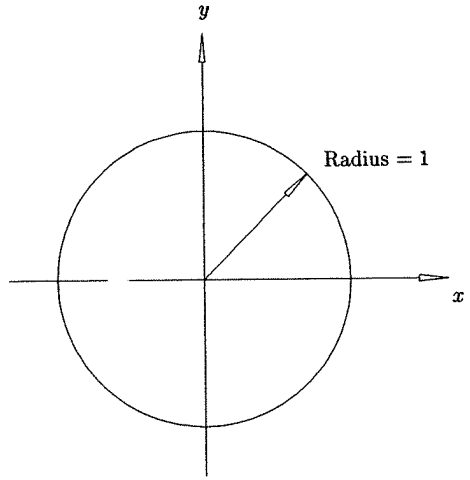


Figure 1.1. A circle of radius 1, centered at the origin.

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

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

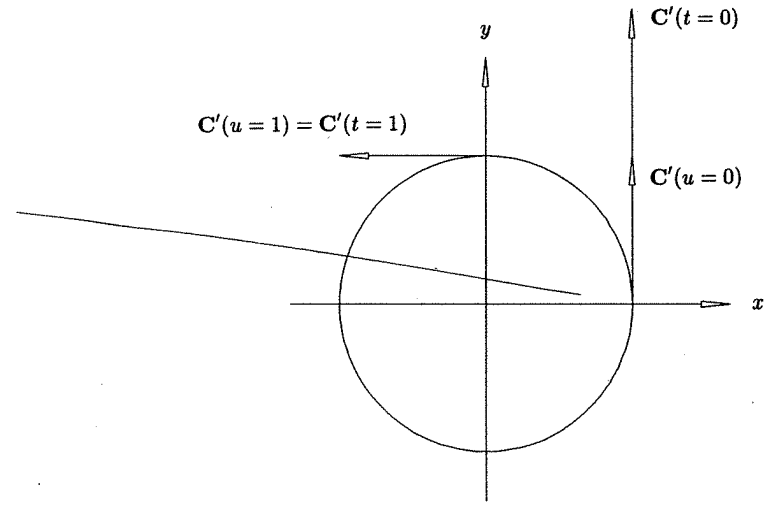


Figure 1.2. Velocity vectors $\mathbf{C}'(u)$ and $\mathbf{C}'(t)$ at $u, t = 0$, and 1.

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.

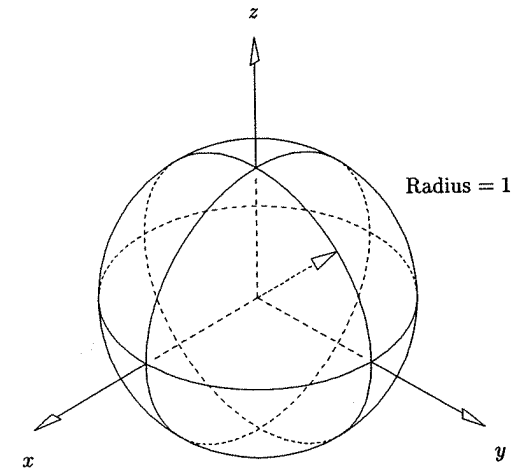


Figure 1.3. A sphere of radius 1, centered at the origin.

Denote the partial derivatives of $\mathbf{S}(u,v)$ by $\mathbf{S}_u(u,v) = \big(x_u(u,v), y_u(u,v), z_u(u,v)\big)$ and $\mathbf{S}_v(u,v) = \big(x_v(u,v), y_v(u,v), z_v(u,v)\big)$, 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, $0 \le v \le 2\pi$, $\mathbf{S}_v(0,v) = \mathbf{S}_v(\pi,v) = 0$, 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:

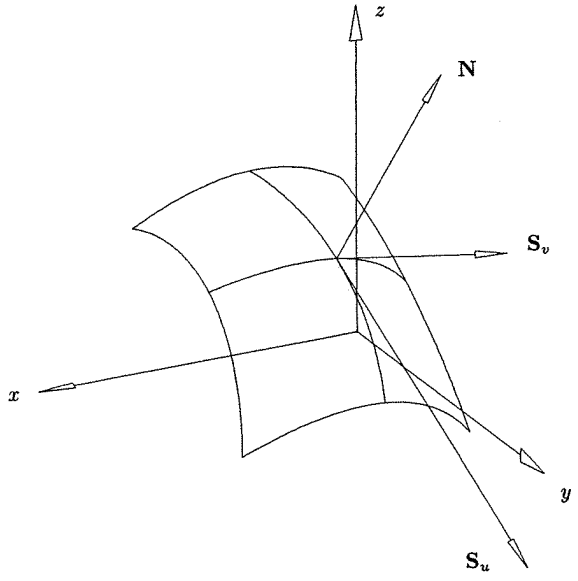


Figure 1.4. Partial derivative and unit normal vectors of $\mathbf{S}(u,v)$.

- 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 (from $\mathbf{C}(a)$ to $\mathbf{C}(b)$ if $a \le u \le b$); 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 ([Faux81; Mort85; Hoff89; Beac91]).

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

The general algorithm is

```
ALGORITHM A1.1
  Horner1(a,n,u0,C)
    { /*  Compute point on power basis curve.  */
      /*  Input:  a,n,u0  */
      /*  Output: C  */
    C = a[n];
    for (i=n-1; i>=0; i--)
      C = C*u0 + a[i];
    }
```

Examples

Ex1.1 $n = 1$. $\mathbf{C}(u) = \mathbf{a}_0 + \mathbf{a}_1 u$, $0 \le u \le 1$, is a straight line segment between the points $\mathbf{a}_0$ and $\mathbf{a}_0 + \mathbf{a}_1$ (Figure 1.5). The constant $\mathbf{C}'(u) = \mathbf{a}_1$ gives the direction of the line.

Ex1.2 $n = 2$. In general, $\mathbf{C}(u) = \mathbf{a}_0 + \mathbf{a}_1 u + \mathbf{a}_2 u^2$, $0 \le u \le 1$, is a parabolic arc between the points $\mathbf{a}_0$ and $\mathbf{a}_0 + \mathbf{a}_1 + \mathbf{a}_2$ (Figure 1.6). This is shown by

1. transforming $\mathbf{C}(u)$ into the xy plane ($\mathbf{C}(u)$ does lie in a unique plane);
2. setting $x = x_0 + x_1 u + x_2 u^2$ and $y = y_0 + y_1 u + y_2 u^2$, and then eliminating u and u^2 from these equations to obtain a second-degree implicit equation in x and y;
3. observing that the form of the implicit equation is that of a parabola.

Notice that the acceleration vector, $\mathbf{C}''(u) = 2\mathbf{a}_2$, is a constant. There are two special (degenerate) cases of interest, both occurring when the vector $\mathbf{a}_2$ is parallel to the initial tangent vector, $\mathbf{a}_1$ (when $x_1 y_2 = x_2 y_1$). In this case, the tangent vector does not turn, i.e., we get a straight line. The vector $\mathbf{a}_2$ can point in the same direction as $\mathbf{a}_1$ (Figure 1.7a), or in the opposite direction (Figure 1.7b). In Figure 1.7b, $\mathbf{a}_1 + 2\mathbf{a}_2 u_0 = 0$ for some $0 \le u_0 \le 1$ (velocity goes to zero, the particle stops), and a portion of the line segment is retraced in the opposite direction.

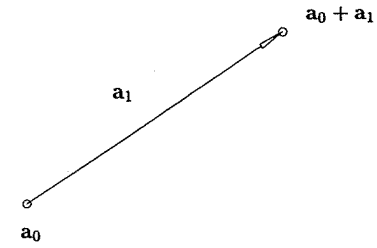


Figure 1.5. Straight line segment, $\mathbf{C}(u) = \mathbf{a}_0 + \mathbf{a}_1 u$.

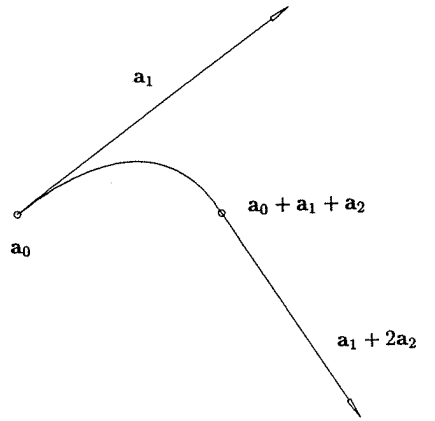
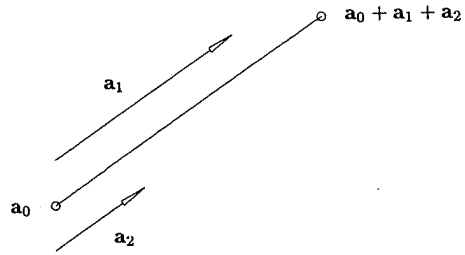


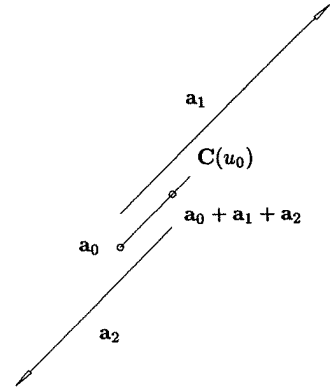
Figure 1.6. Parabolic arc, $\mathbf{C}(u) = \mathbf{a}_0 + \mathbf{a}_1 u + \mathbf{a}_2 u^2$.

Ex1.3 $n = 3$. The cubic, $\mathbf{C}(u) = \mathbf{a}_0 + \mathbf{a}_1 u + \mathbf{a}_2 u^2 + \mathbf{a}_3 u^3$, is a very general curve; it can be a truly *twisted* three-dimensional curve, not lying in a single plane (Figure 1.8a); it can have an inflection point (Figure 1.8b); a cusp (Figure 1.8c); or a loop (Figure 1.8d). A twisted curve results if $\mathbf{a}_0, \mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3$ do not lie in a unique plane. An inflection point on a planar curve is defined as a point where the curve is smooth (no cusp) and the tangent line at that point passes through the curve. This implies a change in the turning direction of the curve. At an inflection point, either $\mathbf{C}''(u) = 0$, or $\mathbf{C}'(u) \parallel \mathbf{C}''(u)$. A necessary (but not sufficient) condition for a cusp at $u = u_0$ is $\mathbf{C}'(u_0) = 0$ (velocity zero). Conditions for a loop to occur are also known (see [Ferg66, 67, 69, 93; Forr70, 80; Wang81; Ston89; Su89]).



(a)

Figure 1.7. $\mathbf{a}_1$ and $\mathbf{a}_2$ parallel. (a) Same direction; (b) opposite directions.



(b)

Figure 1.7. (*Continued.*)

1.3 Bézier Curves

Next we study another parametric polynomial curve, the Bézier curve. Since they both use polynomials for their coordinate functions, the power basis and Bézier forms are mathematically equivalent; i.e., any curve that can be represented in one form can also be represented in the other form. However, the Bézier method is superior to the power basis form for geometric modeling. Our presentation of Bézier curves is rather informal; for a more rigorous and complete treatment the reader should consult other references [Forr72; Bezi72, 86; Gord74a; Chan81; Fari93; Yama88; Hosc93; Roge90].

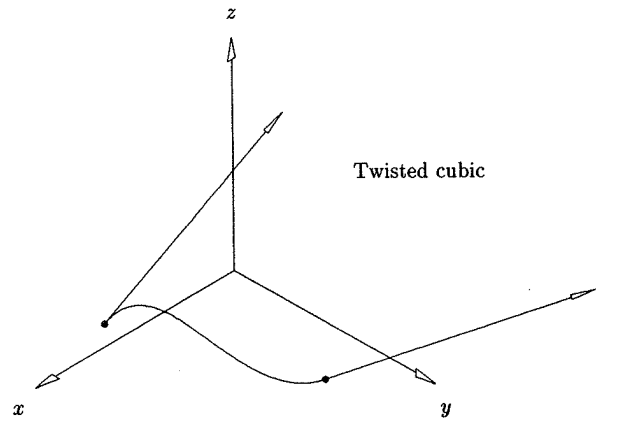
The power basis form has the following disadvantages:

- it is unnatural for interactive shape design; the coefficients $\{\mathbf{a}_i\}$ convey very little geometric insight about the shape of the curve. Furthermore, a designer typically wants to specify end conditions at both ends of the curve, not just at the starting point;
- algorithms for processing power basis polynomials have an algebraic rather than a geometric flavor (e.g., Horner's method);
- numerically, it is a rather poor form; e.g., Horner's method is prone to round-off error if the coefficients vary greatly in magnitude (see [Faro87, 88; Dani89]).

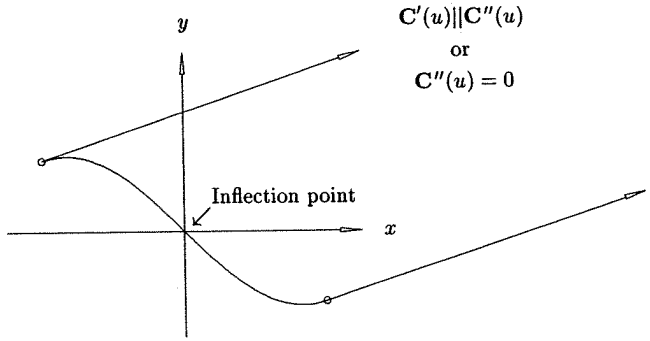
The Bézier method remedies these shortcomings.

An nth-degree Bézier curve is defined by

$$\mathbf{C}(u) = \sum_{i=0}^{n} B_{i,n}(u)\mathbf{P}_i \qquad 0 \le u \le 1 \tag{1.7}$$



(a)



(b)

Figure 1.8. Cubic curves. (a) Three-dimensional twisted; (b) inflection point; (c) cusp; (d) loop.

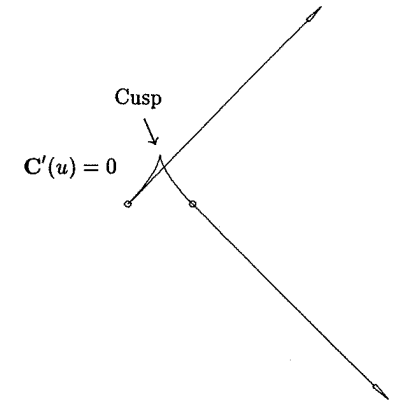
The basis (blending) functions, $\{B_{i,n}(u)\}$, are the classical nth-degree Bernstein polynomials ([Bern12; Lore86]) given by

$$B_{i,n}(u) = \frac{n!}{i!(n-i)!}\, u^i\,(1-u)^{n-i} \tag{1.8}$$

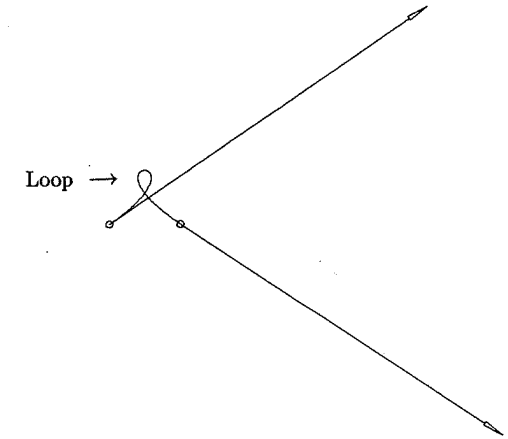
The geometric coefficients of this form, $\{\mathbf{P}_i\}$, are called *control points*. Notice that the definition, Eq. (1.7), requires that $u \in [0,1]$.

Examples

Ex1.4 $n = 1$. From Eq. (1.8) we have $B_{0,1}(u) = 1 - u$ and $B_{1,1}(u) = u$; and Eq. (1.7) takes the form $\mathbf{C}(u) = (1-u)\,\mathbf{P}_0 + u\mathbf{P}_1$. This is a straight line segment from $\mathbf{P}_0$ to $\mathbf{P}_1$ (see Figure 1.9).



(c)



(d)

Figure 1.8. (*Continued.*)

Ex1.5 $n = 2$. From Eqs. (1.7) and (1.8) we have $\mathbf{C}(u) = (1-u)^2\,\mathbf{P}_0 + 2u(1-u)\,\mathbf{P}_1 + u^2\,\mathbf{P}_2$. This is a parabolic arc from $\mathbf{P}_0$ to $\mathbf{P}_2$ (see Figure 1.10). Notice that

- the polygon formed by $\{\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2\}$, called the *control polygon*, approximates the shape of the curve rather nicely;
- $\mathbf{P}_0 = \mathbf{C}(0)$ and $\mathbf{P}_2 = \mathbf{C}(1)$;
- the tangent directions to the curve at its endpoints are parallel to $\mathbf{P}_1 - \mathbf{P}_0$ and $\mathbf{P}_2 - \mathbf{P}_1$ (this is derived later);
- the curve is contained in the triangle formed by $\mathbf{P}_0\mathbf{P}_1\mathbf{P}_2$.

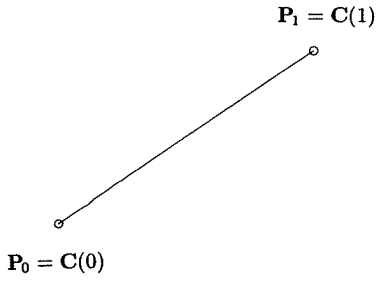


Figure 1.9. A first-degree Bézier curve.

Ex1.6 $n = 3$. We have $\mathbf{C}(u) = (1-u)^3\,\mathbf{P}_0 + 3u(1-u)^2\,\mathbf{P}_1 + 3u^2(1-u)\,\mathbf{P}_2 + u^3\,\mathbf{P}_3$. Examples of cubic Bézier curves are shown in Figures 1.11a to 1.11f. Notice that

- the control polygons approximate the shapes of the curves;
- $\mathbf{P}_0 = \mathbf{C}(0)$ and $\mathbf{P}_3 = \mathbf{C}(1)$;
- the endpoint tangent directions are parallel to $\mathbf{P}_1 - \mathbf{P}_0$ and $\mathbf{P}_3 - \mathbf{P}_2$;
- convex hull property: the curves are contained in the convex hulls of their defining control points (Figure 1.11c);
- variation diminishing property: no straight line intersects a curve more times than it intersects the curve's control polygon (for a three-dimensional Bézier curve, replace the words 'straight line' with the word 'plane'). This expresses the property that a Bézier curve follows its control polygon rather closely and does not wiggle more than its control polygon (Figure 1.11f);
- initially (at $u = 0$) the curve is turning in the same direction as $\mathbf{P}_0\mathbf{P}_1\mathbf{P}_2$. At $u = 1$ it is turning in the direction $\mathbf{P}_1\mathbf{P}_2\mathbf{P}_3$;
- a loop in the control polygon may or may not imply a loop in the curve. The transition between Figure 1.11e and Figure 1.11f is a curve with a cusp.

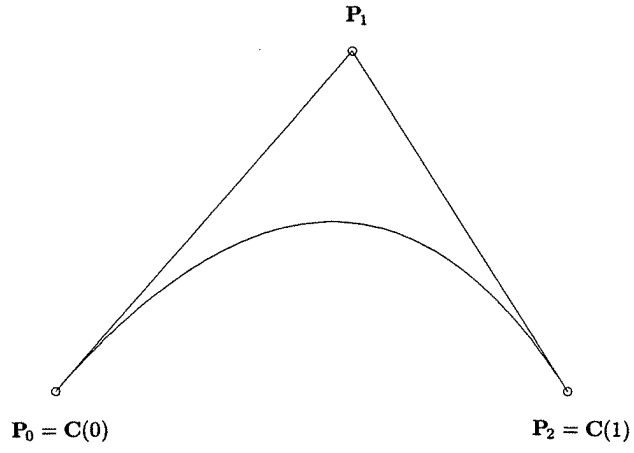
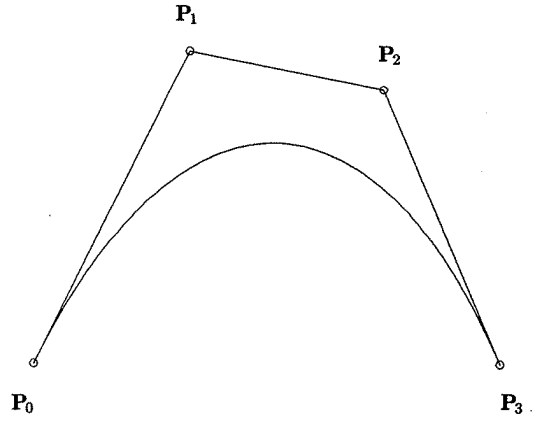
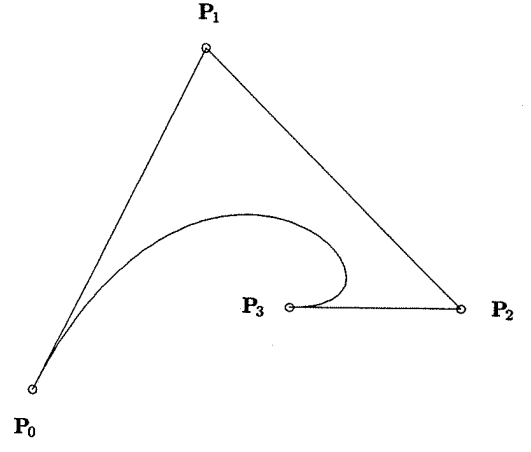


Figure 1.10. A second-degree Bézier curve.



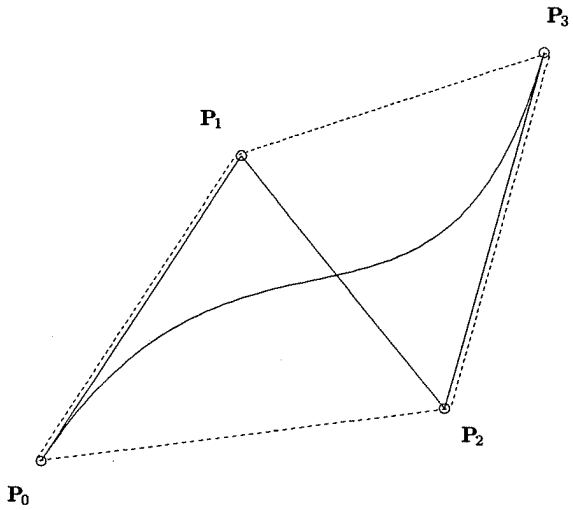
(a)

Figure 1.11. Cubic Bézier curves.

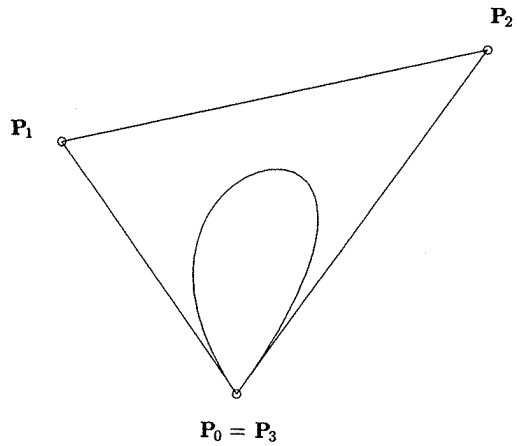


(b)

Figure 1.11. (*Continued.*)



(c)

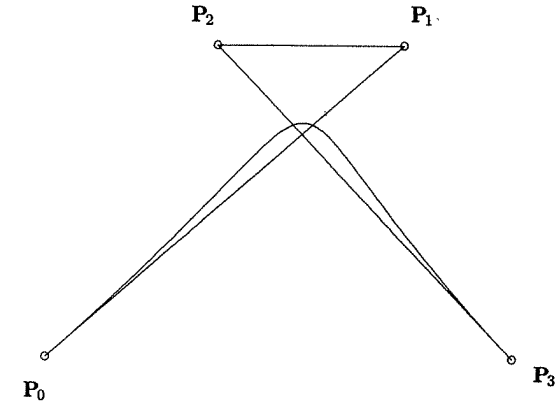


(d)

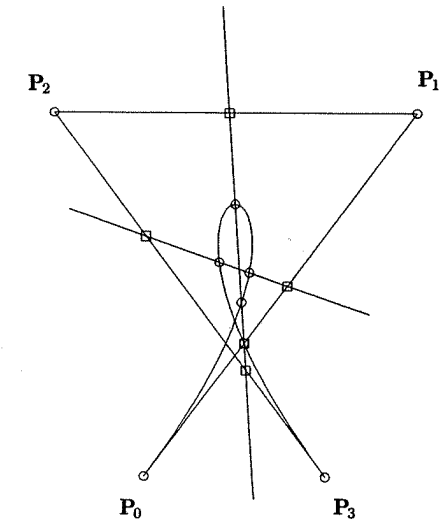
Figure 1.11. (*Continued.*)

Ex1.7 $n = 6$. Figure 1.12 shows a sixth-degree, closed Bézier curve. The curve is smooth at $\mathbf{C}(0)$ $(= \mathbf{C}(1))$ because $\mathbf{P}_1 - \mathbf{P}_0$ is parallel to $\mathbf{P}_6 - \mathbf{P}_5$. By smooth we mean that the tangent vectors at $u = 0$ and $u = 1$ have the same direction.

In addition to the previously mentioned properties, Bézier curves are invariant under the usual transformations such as rotations, translations, and scalings;



(e)



(f)

Figure 1.11. (*Continued.*)

that is, one applies the transformation to the curve by applying it to the control polygon. We present this concept more rigorously in Chapter 3 for B-spline curves (of which Bézier curves are a special case).

In any curve (or surface) representation scheme, the choice of basis functions determines the geometric characteristics of the scheme. Figures 1.13a–d show the basis functions $\{B_{i,n}(u)\}$ for $n = 1, 2, 3, 9$. These functions have these properties:

P1.1 nonnegativity: $B_{i,n}(u) \geq 0$ for all i, n and $0 \leq u \leq 1$;

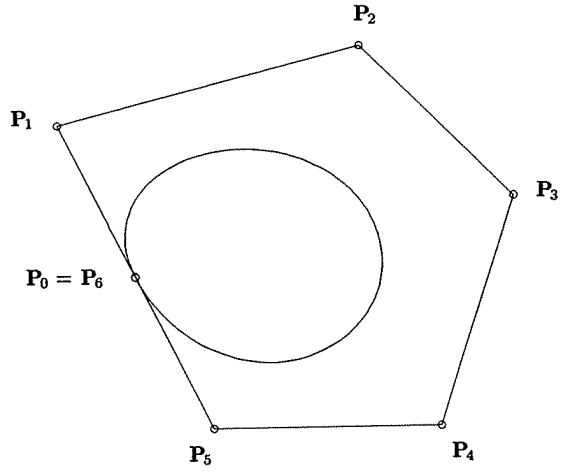


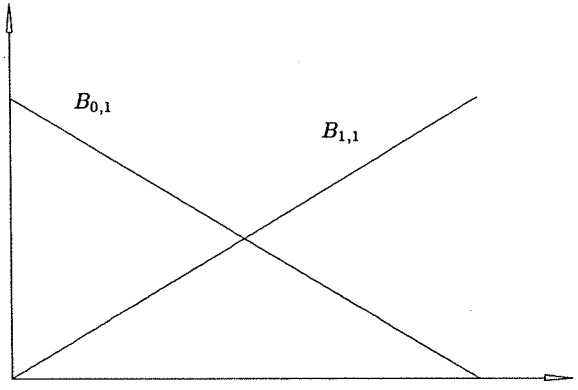
Figure 1.12. A smooth, closed, sixth-degree Bézier curve.

P1.2 partition of unity: $\sum_{i=0}^{n} B_{i,n}(u) = 1$ for all $0 \le u \le 1$;

P1.3 $B_{0,n}(0) = B_{n,n}(1) = 1$;

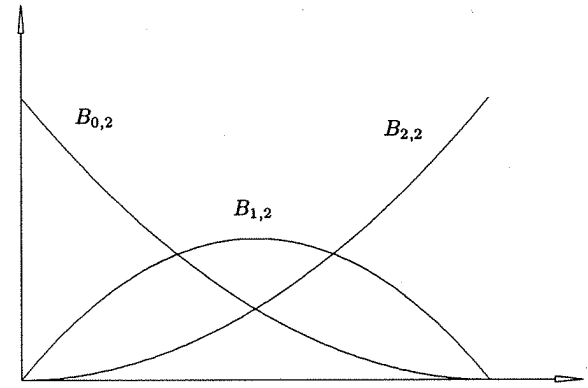
P1.4 $B_{i,n}(u)$ attains exactly one maximum on the interval $[0, 1]$, that is, at $u = i/n$;

P1.5 symmetry: for any n, the set of polynomials $\{B_{i,n}(u)\}$ is symmetric with respect to $u = 1/2$;



(a)

Figure 1.13. The Bernstein polynomials for (a) $n = 1$; (*b*) n = 2; (c) $n = 3$; (*d*) n = 9.



(b)

Figure 1.13. (*Continued.*)

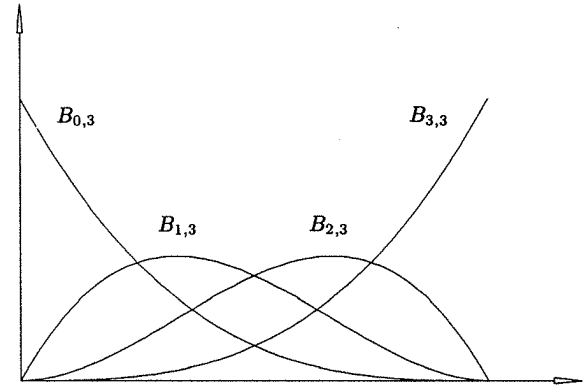
P1.6 recursive definition: $B_{i,n}(u) = (1-u)B_{i,n-1}(u) + uB_{i-1,n-1}(u)$ (see Figure 1.14); we define $B_{i,n}(u) \equiv 0$ if $i < 0$ or $i > n$;

P1.7 derivatives:

$$B'_{i,n}(u) = \frac{dB_{i,n}(u)}{du} = n\big(B_{i-1,n-1}(u) - B_{i,n-1}(u)\big)$$

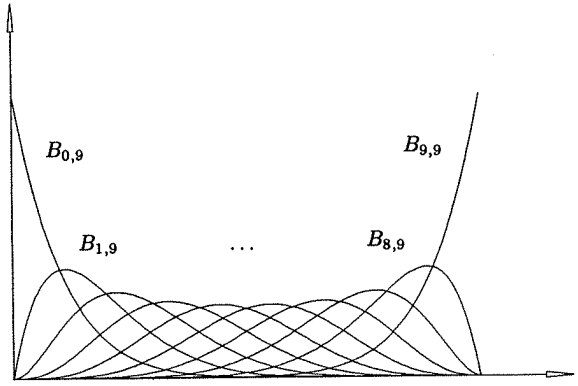
with $$B_{-1,n-1}(u) \equiv B_{n,n-1}(u) \equiv 0$$

Figure 1.15a shows the definition of $B'_{2,5}$, and Figure 1.15b illustrates all the cubic derivative functions.



(c)

Figure 1.13. (*Continued.*)



(d)

Figure 1.13. (*Continued.*)

From Eq. (1.8) we have $B_{0,0}(u) = 1$. Using property `P1.6`, the linear and quadratic Bernstein polynomials are

$$B_{0,1}(u) = (1-u)B_{0,0}(u) + uB_{-1,0}(u) = 1-u$$
$$B_{1,1}(u) = (1-u)B_{1,0}(u) + uB_{0,0}(u) = u$$
$$B_{0,2}(u) = (1-u)B_{0,1}(u) + uB_{-1,1}(u) = (1-u)^2$$
$$B_{1,2}(u) = (1-u)B_{1,1}(u) + uB_{0,1}(u) = (1-u)u + u(1-u) = 2u(1-u)$$
$$B_{2,2}(u) = (1-u)B_{2,1}(u) + uB_{1,1}(u) = u^2$$

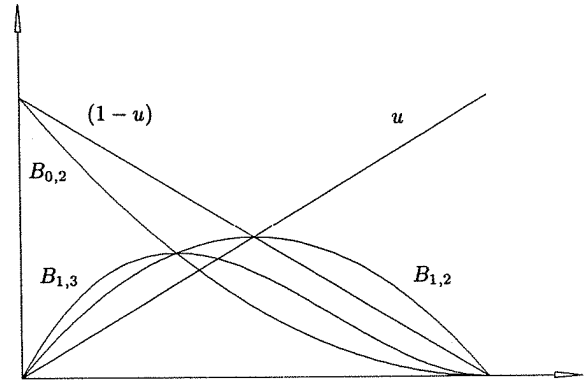
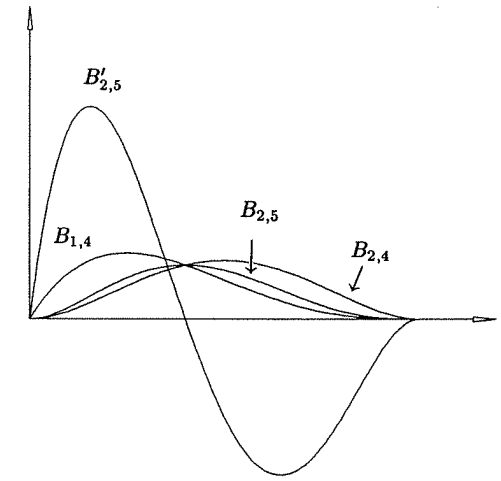
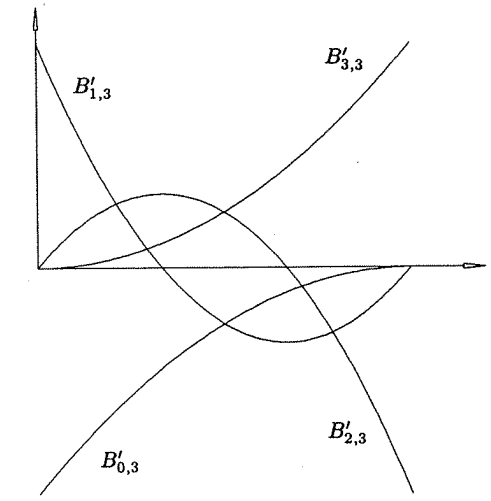


Figure 1.14. The recursive definition of the Bernstein polynomial, $B_{1,3}(u)$.



(a)



(b)

Figure 1.15. Derivatives. (a) The derivative $B'_{2,5}(u)$ in terms of $B_{1,4}(u)$ and $B_{2,4}(u)$; (b) the derivatives of the four cubic Bernstein polynomials, $B'_{0,3}(u)$; $B'_{1,3}(u)$; $B'_{2,3}(u)$; $B'_{3,3}(u)$.

Property `P1.6` yields simple algorithms to compute values of the Bernstein polynomials at fixed values of u. Algorithm `A1.2` computes the value $B_{i,n}(u)$ for fixed u. The computation of $B_{1,3}$ is depicted in Table 1.1.

```
ALGORITHM A1.2
  Bernstein(i,n,u,B)
    { /*  Compute the value of a Bernstein polynomial.  */
      /*  Input:  i,n,u  */
      /*  Output: B  */
    for (j=0; j<=n; j++)  /* compute the columns */
      temp[j] = 0.0;       /* of Table 1.1 */
    temp[n-i] = 1.0;      /* in a temporary array */
    u1 = 1.0-u;
    for (k=1; k<=n; k++)
      for (j=n; j>=k; j--)
        temp[j] = u1*temp[j] + u*temp[j-1];
    B = temp[n];
    }
```

Algorithm A1.3 computes the $n+1$ nth-degree Bernstein polynomials which are nonzero at fixed u. It avoids unnecessary computation of zero terms. The algorithm is depicted in Table 1.2 for the cubic case.

```
ALGORITHM A1.3
  AllBernstein(n,u,B)
```

Table 1.1. The computation of $B_{1,3}$.

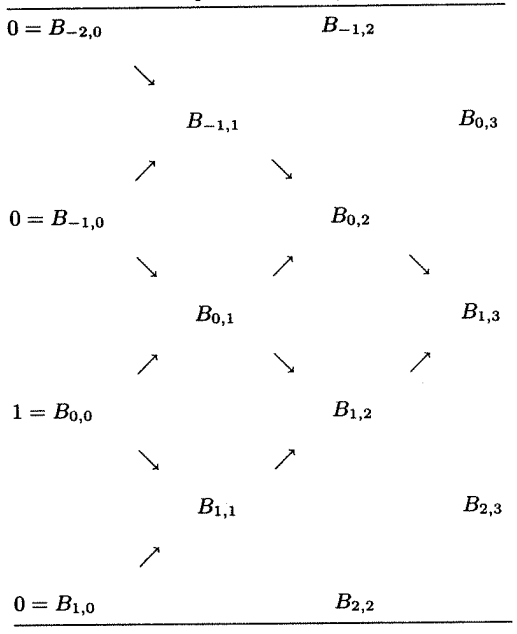
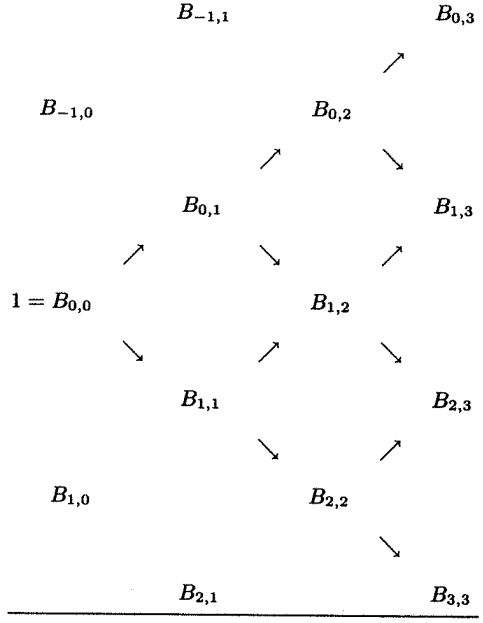


Table 1.2. Computation of all the cubic Bernstein polynomials.




```
{  /*  Compute all nth-degree Bernstein polynomials.  */
   /*  Input:  n,u  */
   /*  Output: B (an array, B[0],...,B[n])  */
B[0] = 1.0;
u1 = 1.0-u;
for (j=1; j<=n; j++)
  {
  saved = 0.0;
  for (k=0; k<j; k++)
    {
    temp = B[k];
    B[k] = saved+u1*temp;
    saved = u*temp;
    }
  B[j] = saved;
  }
}
```

Algorithm A1.4 combines A1.3 and Eq. (1.7) to compute the point on an nth-degree Bézier curve at a fixed u value.

```
ALGORITHM A1.4
  PointOnBezierCurve(P,n,u,C)
    { /*  Compute point on Bezier curve.  */
      /*  Input:  P,n,u  */
      /*  Output: C (a point)  */
    AllBernstein(n,u,B)   /* B is a local array */
    C = 0.0;
    for (k=0; k<=n; k++)   C = C + B[k]*P[k];
    }
```

Using property P1.7, it is easy to derive the general expression for the derivative of a Bézier curve

$$\mathbf{C}'(u) = \frac{d\left(\sum_{i=0}^{n} B_{i,n}(u)\,\mathbf{P}_i\right)}{du} = \sum_{i=0}^{n} B'_{i,n}(u)\,\mathbf{P}_i$$

$$= \sum_{i=0}^{n} n\big(B_{i-1,n-1}(u) - B_{i,n-1}(u)\big)\,\mathbf{P}_i$$

$$= n\sum_{i=0}^{n-1} B_{i,n-1}(u)(\mathbf{P}_{i+1} - \mathbf{P}_i) \qquad (1.9)$$

From Eq. (1.9) we easily obtain formulas for the end derivatives of a Bézier curve, e.g.

$$\mathbf{C}'(0) = n(\mathbf{P}_1 - \mathbf{P}_0) \qquad \mathbf{C}''(0) = n(n-1)(\mathbf{P}_0 - 2\mathbf{P}_1 + \mathbf{P}_2)$$
$$\mathbf{C}'(1) = n(\mathbf{P}_n - \mathbf{P}_{n-1}) \qquad \mathbf{C}''(1) = n(n-1)(\mathbf{P}_n - 2\mathbf{P}_{n-1} + \mathbf{P}_{n-2}) \qquad (1.10)$$

Notice from Eqs. (1.9) and (1.10) that

- the derivative of an nth-degree Bézier curve is an $(n-1)$th-degree Bézier curve;
- the expressions for the end derivatives at $u = 0$ and $u = 1$ are symmetric (due, of course, to the symmetry of the basis functions);
- the kth derivative at an endpoint depends (in a geometrically very intuitive manner) solely on the $k+1$ control points at that end.

Let $n = 2$ and $\mathbf{C}(u) = \sum_{i=0}^{2} B_{i,2}(u)\,\mathbf{P}_i$. Then

$$\mathbf{C}(u) = (1-u)^2\,\mathbf{P}_0 + 2u(1-u)\,\mathbf{P}_1 + u^2\,\mathbf{P}_2$$
$$= (1-u)\underbrace{\big((1-u)\,\mathbf{P}_0 + u\,\mathbf{P}_1\big)}_{\text{linear}} + u\underbrace{\big((1-u)\,\mathbf{P}_1 + u\,\mathbf{P}_2\big)}_{\text{linear}}$$

Thus, $\mathbf{C}(u)$ is obtained as the linear interpolation of two first-degree Bézier curves; in particular, any point on $\mathbf{C}(u)$ is obtained by three linear interpolations.

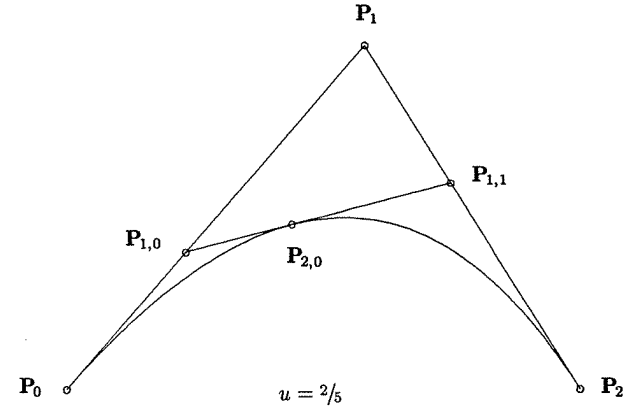


Figure 1.16. Obtaining a point on a quadratic Bézier curve by repeated linear interpolation at $u_0 = 2/5$.

Assuming a fixed $u = u_0$ and letting $\mathbf{P}_{1,0} = (1-u_0)\,\mathbf{P}_0 + u_0\mathbf{P}_1$, $\mathbf{P}_{1,1} = (1-u_0)\,\mathbf{P}_1 + u_0\mathbf{P}_2$, and $\mathbf{P}_{2,0} = (1-u_0)\,\mathbf{P}_{1,0} + u_0\mathbf{P}_{1,1}$, it follows that $\mathbf{C}(u_0) = \mathbf{P}_{2,0}$. The situation is depicted in Figure 1.16, and the cubic case is shown in Figure 1.17.

Denoting a general nth-degree Bézier curve by $\mathbf{C}_n(\mathbf{P}_0, \ldots, \mathbf{P}_n)$, we have

$$\mathbf{C}_n(\mathbf{P}_0, \ldots, \mathbf{P}_n) = (1-u)\mathbf{C}_{n-1}(\mathbf{P}_0, \ldots, \mathbf{P}_{n-1}) + u\mathbf{C}_{n-1}(\mathbf{P}_1, \ldots, \mathbf{P}_n) \qquad (1.11)$$

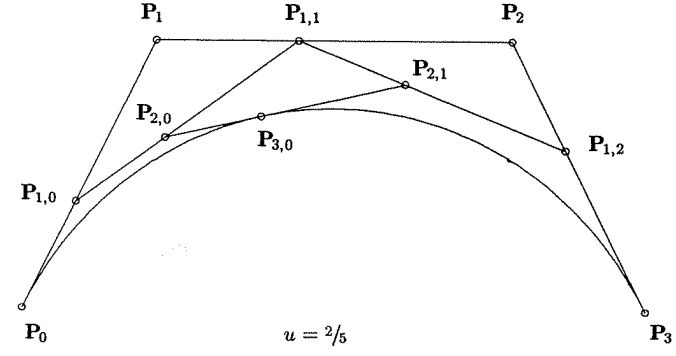


Figure 1.17. A point on a cubic Bézier curve by repeated linear interpolation at $u_0 = 2/5$.

This follows from the recursive definition of the basis functions (see P1.6). Fixing $u = u_0$ and denoting $\mathbf{P}_i$ by $\mathbf{P}_{0,i}$, Eq. (1.11) yields a recursive algorithm for computing the point $\mathbf{C}(u_0) = \mathbf{P}_{n,0}(u_0)$ on an nth-degree Bézier curve, i.e.

$$\mathbf{P}_{k,i}(u_0) = (1-u_0)\,\mathbf{P}_{k-1,i}(u_0) + u_0\,\mathbf{P}_{k-1,i+1}(u_0) \quad \text{for} \begin{cases} k = 1,\ldots,n \\ i = 0,\ldots,n-k \end{cases} \tag{1.12}$$

Equation (1.12) is called the *deCasteljau Algorithm* (see [Boeh84; deCa86, 93]). It is a corner cutting process (see Figures 1.16 and 1.17) which yields the triangular table of points shown in Table 1.3.

```
ALGORITHM A1.5
  deCasteljau1(P,n,u,C)
    { /*  Compute point on a Bézier curve */
      /*  using deCasteljau  */
      /*  Input:  P,n,u  */
      /*  Output: C (a point)  */
    for (i=0; i<=n; i++)    /* Use local array so we do not */
      Q[i] = P[i];          /* destroy control points */
    for (k=1; k<=n; k++)
      for (i=0; i<=n-k; i++)
        Q[i] = (1.0-u)*Q[i] + u*Q[i+1];
    C = Q[0];
    }
```

We conclude this section with a comparison of the Bézier and power basis methods. Clearly, the Bézier form is the more geometric of the two. Equation (1.10), together with the convex hull and variation diminishing properties, makes

Table 1.3. Points generated by the deCasteljau algorithm.

$\mathbf{P}_0$						
	$\mathbf{P}_{1,0}$					
$\mathbf{P}_1$		$\mathbf{P}_{2,0}$				
	$\mathbf{P}_{1,1}$					
$\mathbf{P}_2$		$\vdots$				
$\vdots$	$\vdots$	$\vdots$		$\mathbf{P}_{n-1,0}$		
$\vdots$	$\vdots$	$\vdots$	$\cdots$		$\mathbf{P}_{n,0}$ =	$\mathbf{C}(u_0)$
$\vdots$	$\vdots$	$\vdots$		$\mathbf{P}_{n-1,1}$		
$\mathbf{P}_{n-2}$		$\vdots$				
	$\mathbf{P}_{1,n-2}$					
$\mathbf{P}_{n-1}$		$\mathbf{P}_{2,n-2}$				
	$\mathbf{P}_{1,n-1}$					
$\mathbf{P}_n$						

Bézier curves more suitable for interactive curve design. The control points give the designer a more intuitive handle on curve shape than do the power basis coefficients. Furthermore, the deCasteljau algorithm is less prone to round-off error than Horner's algorithm. This is intuitively clear when one considers that the deCasteljau algorithm is simply repeated linear interpolation between points, all of which lie in the vicinity of the curve. The only disadvantage of the Bézier form is that point evaluation is less efficient (see Algorithms A1.1, A1.4, and A1.5, and Exercise 1.13 later in the chapter).

1.4 Rational Bézier Curves

Next we introduce the concepts of rational curves and homogeneous coordinates. To illustrate these concepts we give a brief introduction to rational Bézier curves. These curves are special cases of rational B-spline curves and as such are treated more completely and rigorously in subsequent chapters.

Although polynomials offer many advantages, there exist a number of important curve and surface types which cannot be represented precisely using polynomials, e.g., circles, ellipses, hyperbolas, cylinders, cones, spheres, etc. As an example, we give a proof that the unit circle in the xy plane, centered at the origin, cannot be represented using polynomial coordinate functions. To the contrary, let us assume that

$$x(u) = a_0 + a_1 u + \cdots + a_n u^n$$
$$y(u) = b_0 + b_1 u + \cdots + b_n u^n$$

Then $x^2 + y^2 - 1 = 0$ implies that

$$\begin{aligned} 0 &= (a_0 + a_1 u + \cdots + a_n u^n)^2 + (b_0 + b_1 u + \cdots + b_n u^n)^2 - 1 \\ &= (a_0^2 + b_0^2 - 1) + 2(a_0 a_1 + b_0 b_1)u + (a_1^2 + 2a_0 a_2 + b_1^2 + 2b_0 b_2)u^2 \\ &\quad + \cdots + (a_{n-1}^2 + 2a_{n-2}a_n + b_{n-1}^2 + 2b_{n-2}b_n)u^{2n-2} \\ &\quad + 2(a_n a_{n-1} + b_n b_{n-1})u^{2n-1} + (a_n^2 + b_n^2)u^{2n} \end{aligned}$$

This equation must hold for all u, which implies that all coefficients are zero. Starting with the highest degree and working down, we show in n steps that all $a_i = 0$ and $b_i = 0$ for $1 \le i \le n$.

Step

1. $a_n^2 + b_n^2 = 0$ implies $a_n = b_n = 0$.
2. $a_{n-1}^2 + 2a_{n-2}a_n + b_{n-1}^2 + 2b_{n-2}b_n = 0$ and Step 1 imply that $a_{n-1}^2 + b_{n-1}^2 = 0$ which implies that $a_{n-1} = b_{n-1} = 0$.

$\vdots$

n. $a_1^2 + 2a_0a_2 + b_1^2 + 2b_0b_2 = 0$ and Step $n-1$ imply that $a_1^2 + b_1^2 = 0$, which implies that $a_1 = b_1 = 0$.

Thus, $x(u) = a_0$ and $y(u) = b_0$, which is an obvious contradiction.

It is known from classical mathematics that all the conic curves, including the circle, can be represented using *rational functions*, which are defined as the ratio of two polynomials. In fact, they are represented with rational functions of the form

$$x(u) = \frac{X(u)}{W(u)} \qquad y(u) = \frac{Y(u)}{W(u)} \tag{1.13}$$

where $X(u)$, $Y(u)$, and $W(u)$ are polynomials, that is, each of the coordinate functions has the same denominator.

Examples

Ex1.8 Circle of radius 1, centered at the origin

$$x(u) = \frac{1-u^2}{1+u^2} \qquad y(u) = \frac{2u}{1+u^2}$$

Ex1.9 Ellipse, centered at the origin; the y-axis is the major axis, the x-axis is the minor axis, and the major and minor radii are 2 and 1, respectively

$$x(u) = \frac{1-u^2}{1+u^2} \qquad y(u) = \frac{4u}{1+u^2}$$

Ex1.10 Hyperbola, center at $\mathbf{P} = (0, 4/3)$; the y-axis is the transverse axis

$$x(u) = \frac{-1+2u}{1+2u-2u^2} \qquad y(u) = \frac{4u(1-u)}{1+2u-2u^2}$$

The lower branch (with vertex at $\mathbf{P} = (0, 2/3)$) is traced out for

$$u \in \left(\frac{1-\sqrt{3}}{2}, \frac{1+\sqrt{3}}{2}\right)$$

Ex1.11 Parabola, vertex at the origin; the y-axis is the axis of symmetry

$$x(u) = u \qquad y(u) = u^2$$

Notice that the parabola does not require rational functions. The reader should sketch these functions. For the circle equations it is easy to see that for any u, $(x(u), y(u))$ lies on the unit circle centered at the origin

$$(x(u))^2 + (y(u))^2 = \left(\frac{1-u^2}{1+u^2}\right)^2 + \left(\frac{2u}{1+u^2}\right)^2$$

$$= \frac{1-2u^2+u^4+4u^2}{(1+u^2)^2} = \frac{(1+u^2)^2}{(1+u^2)^2} = 1$$

Define an nth-degree *rational Bézier curve* by (see [Forr68; Pieg86; Fari83, 89])

$$\mathbf{C}(u) = \frac{\sum_{i=0}^{n} B_{i,n}(u) w_i \mathbf{P}_i}{\sum_{i=0}^{n} B_{i,n}(u) w_i} \qquad 0 \le u \le 1 \tag{1.14}$$

The $\mathbf{P}_i = (x_i, y_i, z_i)$ and $B_{i,n}(u)$ are as before; the w_i are scalars, called the *weights*. Thus, $W(u) = \sum_{i=0}^{n} B_{i,n}(u) w_i$ is the common denominator function. Except where explicitly stated otherwise, we assume that $w_i > 0$ for all i. This ensures that $W(u) > 0$ for all $u \in [0,1]$. We write

$$\mathbf{C}(u) = \sum_{i=0}^{n} R_{i,n}(u) \mathbf{P}_i \qquad 0 \le u \le 1 \tag{1.15}$$

where

$$R_{i,n}(u) = \frac{B_{i,n}(u) w_i}{\sum_{j=0}^{n} B_{j,n}(u) w_j}$$

The $R_{i,n}(u)$ are the rational basis functions for this curve form. Figure 1.18a shows an example of cubic basis functions, and Figure 1.18b a corresponding cubic rational Bézier curve.

The $R_{i,n}(u)$ have properties which can be easily derived from Eq. (1.15) and the corresponding properties of the $B_{i,n}(u)$:

P1.8 nonnegativity: $R_{i,n}(u) \ge 0$ for all i, n and $0 \le u \le 1$;

P1.9 partition of unity: $\sum_{i=0}^{n} R_{i,n}(u) = 1$ for all $0 \le u \le 1$;

P1.10 $R_{0,n}(0) = R_{n,n}(1) = 1$;

P1.11 $R_{i,n}(u)$ attains exactly one maximum on the interval $[0,1]$;

P1.12 if $w_i = 1$ for all i, then $R_{i,n}(u) = B_{i,n}(u)$ for all i; i.e., the $B_{i,n}(u)$ are a special case of the $R_{i,n}(u)$.

These yield the following geometric properties of rational Bézier curves:

P1.13 convex hull property: the curves are contained in the convex hulls of their defining control points (the $\mathbf{P}_i$);

P1.14 transformation invariance: rotations, translations, and scalings are applied to the curve by applying them to the control points;

P1.15 variation diminishing property: same as for polynomial Bézier curves (see previous section);

P1.16 endpoint interpolation: $\mathbf{C}(0) = \mathbf{P}_0$ and $\mathbf{C}(1) = \mathbf{P}_n$;

P1.17 the kth derivative at $u = 0$ ($u = 1$) depends on the first (last) $k+1$ control points and weights; in particular, $\mathbf{C}'(0)$ and $\mathbf{C}'(1)$ are parallel to $\mathbf{P}_1 - \mathbf{P}_0$ and $\mathbf{P}_n - \mathbf{P}_{n-1}$, respectively;

P1.18 polynomial Bézier curves are a special case of rational Bézier curves.

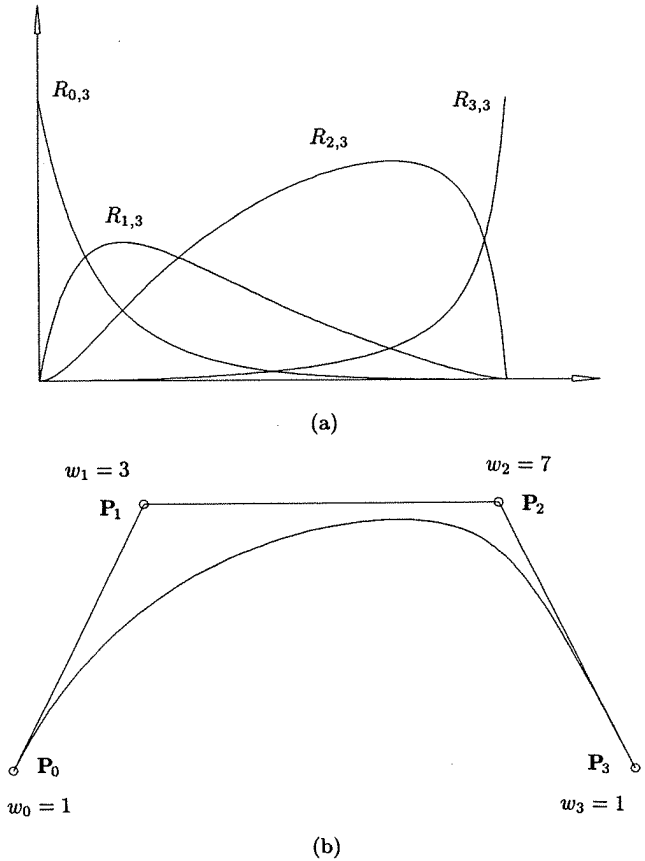


Figure 1.18. Rational cubic. (a) Basis functions; (b) Bézier curve.

Example

Ex1.12 Let us consider the rational Bézier circular arc.

$$\mathbf{C}(u) = \big(x(u), y(u)\big) = \left(\frac{1-u^2}{1+u^2}, \frac{2u}{1+u^2}\right) \qquad 0 \le u \le 1$$

represents one quadrant of the unit circle, as shown in Figure 1.19a. We now derive the quadratic rational Bézier representation of this circular arc. Clearly, from P1.16 and P1.17, $\mathbf{P}_0 = (1,0)$, $\mathbf{P}_1 = (1,1)$, and $\mathbf{P}_2 = (0,1)$. For the weights we have

$$W(u) = 1 + u^2 = \sum_{i=0}^{2} B_{i,2}(u) w_i = (1-u)^2 w_0 + 2u(1-u)w_1 + u^2 w_2$$

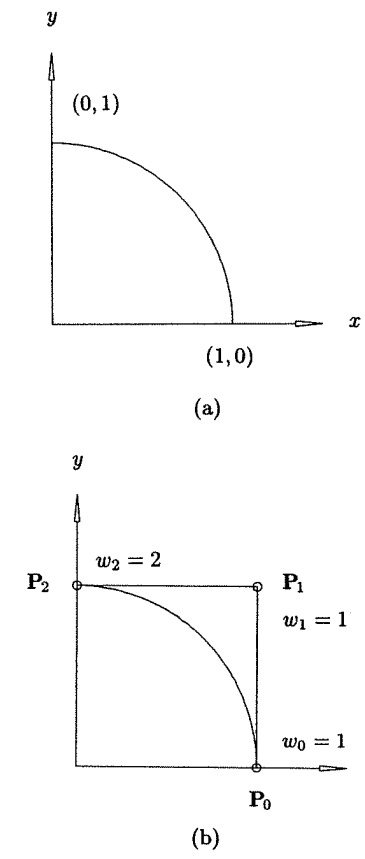


Figure 1.19. Representation of the unit circle. (a) $x(u) = (1-u^2)/(1+u^2)$ and $y(u) = (2u)/(1+u^2)$ for one quadrant; (b) the Bézier representation corresponding to Figure 1.19a ($w_0 = 1$, $w_1 = 1$, $w_2 = 2$).

Substituting $u = 0$ yields $w_0 = 1$, and $u = 1$ yields $w_2 = 2$. Finally, substituting $u = 1/2$ yields $5/4 = 1/4 w_0 + 1/2 w_1 + 1/4 w_2$, and using $w_0 = 1$ and $w_2 = 2$ yields $w_1 = 1$ (see Figure 1.19b).

Rational curves with coordinate functions in the form of Eq. (1.13) (one common denominator) have an elegant geometric interpretation which yields efficient processing and compact data storage. The idea is to use *homogeneous coordinates* to represent a rational curve in n-dimensional space as a polynomial curve in $(n+1)$-dimensional space (see [Robe65; Ries81; Patt85]). Let us start with a point in three-dimensional Euclidean space, $\mathbf{P} = (x, y, z)$. Then $\mathbf{P}$ is written as $\mathbf{P}^w = (wx, wy, wz, w) = (X, Y, Z, W)$ in four-dimensional space, $w \neq 0$. Now

$\mathbf{P}$ is obtained from $\mathbf{P}^w$ by dividing all coordinates by the fourth coordinate, W, i.e., by mapping $\mathbf{P}^w$ from the origin to the hyperplane $W = 1$ (see Figure 1.20 for the two-dimensional case, $\mathbf{P} = (x, y)$). This mapping, denoted by H, is a perspective map with center at the origin

$$\mathbf{P} = H\{\mathbf{P}^w\} = H\{(X, Y, Z, W)\} = \begin{cases} \left(\frac{X}{W}, \frac{Y}{W}, \frac{Z}{W}\right) & \text{if } W \neq 0 \\ \text{direction } (X, Y, Z) & \text{if } W = 0 \end{cases} \tag{1.16}$$

Notice that for arbitrary x, y, z, w_1, w_2, where $w_1 \neq w_2$

$$\begin{aligned} H\{\mathbf{P}^{w_1}\} &= H\{(w_1 x, w_1 y, w_1 z, w_1)\} = (x, y, z) \\ &= H\{(w_2 x, w_2 y, w_2 z, w_2)\} = H\{\mathbf{P}^{w_2}\} \end{aligned}$$

Now for a given set of control points, $\{\mathbf{P}_i\}$, and weights, $\{w_i\}$, construct the weighted control points, $\mathbf{P}_i^w = (w_i x_i, w_i y_i, w_i z_i, w_i)$. Then define the *nonrational* (polynomial) Bézier curve in four-dimensional space

$$\mathbf{C}^w(u) = \sum_{i=0}^{n} B_{i,n}(u)\, \mathbf{P}_i^w \tag{1.17}$$

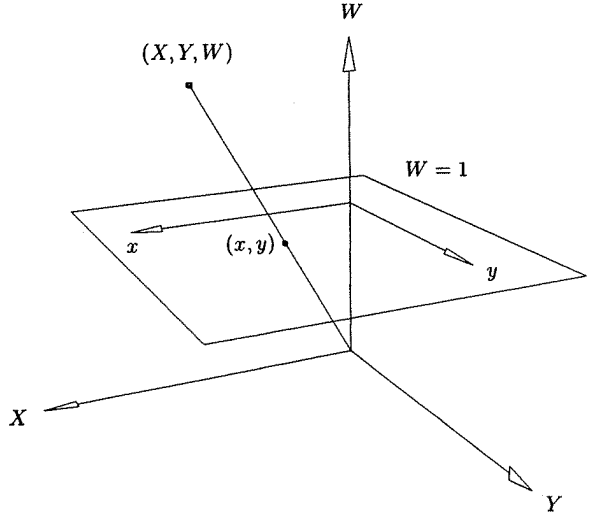


Figure 1.20. A representation of Euclidean points in homogeneous form.

Then, applying the perspective map, H, to $\mathbf{C}^w(u)$ yields the corresponding rational Bézier curve of Eq. (1.14) (see Figure 1.21), that is, writing out the coordinate functions of Eq. (1.17), we get

$$X(u) = \sum_{i=0}^{n} B_{i,n}(u) w_i x_i \qquad Y(u) = \sum_{i=0}^{n} B_{i,n}(u) w_i y_i$$

$$Z(u) = \sum_{i=0}^{n} B_{i,n}(u) w_i z_i \qquad W(u) = \sum_{i=0}^{n} B_{i,n}(u) w_i$$

Locating the curve in three-dimensional space yields

$$x(u) = \frac{X(u)}{W(u)} = \frac{\sum_{i=0}^{n} B_{i,n}(u) w_i x_i}{\sum_{i=0}^{n} B_{i,n}(u) w_i}$$

$$y(u) = \frac{Y(u)}{W(u)} = \frac{\sum_{i=0}^{n} B_{i,n}(u) w_i y_i}{\sum_{i=0}^{n} B_{i,n}(u) w_i}$$

$$z(u) = \frac{Z(u)}{W(u)} = \frac{\sum_{i=0}^{n} B_{i,n}(u) w_i z_i}{\sum_{i=0}^{n} B_{i,n}(u) w_i}$$

Using vector notation, we get

$$\begin{aligned} \mathbf{C}(u) = (x(u), y(u), z(u)) &= \frac{\sum_{i=0}^{n} B_{i,n}(u) w_i (x_i, y_i, z_i)}{\sum_{i=0}^{n} B_{i,n}(u) w_i} \\ &= \frac{\sum_{i=0}^{n} B_{i,n}(u) w_i \mathbf{P}_i}{\sum_{i=0}^{n} B_{i,n}(u) w_i} \end{aligned} \tag{1.18}$$

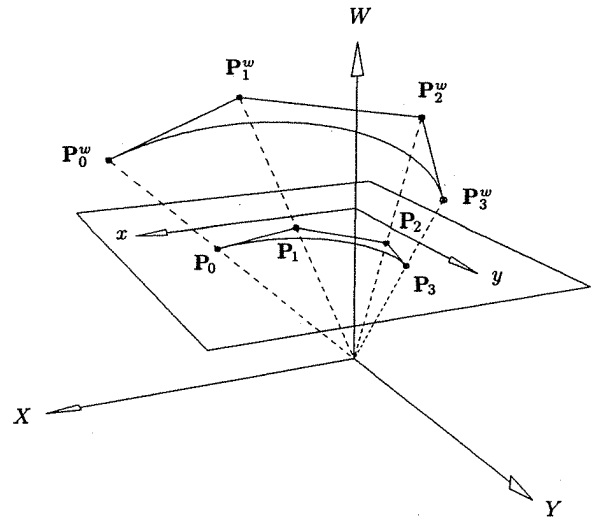


Figure 1.21. A geometric construction of a rational Bézier curve.

For algorithms in this book we primarily use the form given by Eq. (1.17), and an analogous form for rational B-spline curves. Thus, nonrational forms are processed in four-dimensional space, and the results are located in three-dimensional space using the map H. We refer interchangeably to either $\mathbf{C}^w(u)$ or $\mathbf{C}(u)$ as the rational Bézier (or B-spline) curve, although strictly speaking, $\mathbf{C}^w(u)$ is not a rational curve.

Examples

Ex1.13 Let us return to the circular arc of Figure 1.19b. We have $\mathbf{P}_0 = (1,0)$, $\mathbf{P}_1 = (1,1)$, $\mathbf{P}_2 = (0,1)$, and $w_0 = 1$, $w_1 = 1$, $w_2 = 2$. Hence, for Eq. (1.17) the three-dimensional control points are $\mathbf{P}_0^w = (1,0,1)$, $\mathbf{P}_1^w = (1,1,1)$, and $\mathbf{P}_2^w = (0,2,2)$. Then $\mathbf{C}^w(u) = (1-u)^2\,\mathbf{P}_0^w + 2u(1-u)\,\mathbf{P}_1^w + u^2\,\mathbf{P}_2^w$ is a parabolic arc (nonrational), which projects onto a circular arc on the $W = 1$ plane (see Figure 1.22).

Let u_0 be fixed. Since $\mathbf{C}^w(u)$ is a polynomial Bézier curve, we use the deCasteljau algorithm to compute $\mathbf{C}^w(u_0)$; subsequently, $\mathbf{C}(u_0) = H\{\mathbf{C}^w(u_0)\}$. Thus, we apply Eq. (1.12) to the $\mathbf{P}_i^w$

$$\mathbf{P}_{k,i}^w(u_0) = (1-u_0)\,\mathbf{P}_{k-1,i}^w + u_0\,\mathbf{P}_{k-1,i+1}^w \quad \text{for} \quad \begin{cases} k = 1,\ldots,n \\ i = 0,\ldots,n-k \end{cases} \tag{1.19}$$

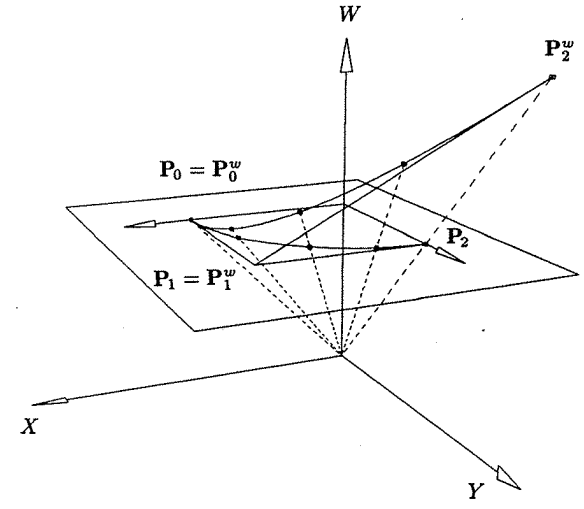


Figure 1.22. A homogeneous representation of a circular arc.

Ex1.14 Let us apply Eq. (1.19) to compute the point at $u = 1/2$ on the rational Bézier circular arc of Example 1.13. The arc is given by $\mathbf{C}^w(u) = (1-u)^2\,\mathbf{P}_0^w + 2u(1-u)\,\mathbf{P}_1^w + u^2\,\mathbf{P}_2^w$, where $\mathbf{P}_0^w = (1,0,1)$, $\mathbf{P}_1^w = (1,1,1)$, $\mathbf{P}_2^w = (0,2,2)$. The triangular set of generated points is shown in Table 1.4. Then $\mathbf{C}(1/2) = H\{\mathbf{C}^w(1/2)\} = H\{(3/4, 1, 5/4)\} = (3/5, 4/5)$.

Now let us compute the point using the other representations we have developed. Let

$$\mathbf{C}(u) = \left(\frac{(1-u^2)}{(1+u^2)}, \frac{(2u)}{(1+u^2)}\right)$$

Table 1.4. Generation of the point $\mathbf{C}^w(1/2)$ on the circular arc.

$(1,0,1)$			
	$\left(1, \frac{1}{2}, 1\right)$		
$(1,1,1)$		$\left(\frac{3}{4}, 1, \frac{5}{4}\right)$	$= \mathbf{C}^w\left(\frac{1}{2}\right)$
	$\left(\frac{1}{2}, \frac{3}{2}, \frac{3}{2}\right)$		
$(0,2,2)$			

Then
$$\mathbf{C}\left(\tfrac{1}{2}\right) = \left(\frac{1-\left(\frac{1}{2}\right)^2}{1+\left(\frac{1}{2}\right)^2}, \frac{2\left(\frac{1}{2}\right)}{1+\left(\frac{1}{2}\right)^2}\right) = \left(\tfrac{3}{5}, \tfrac{4}{5}\right)$$

Using Eq. (1.17)

$$\mathbf{C}^w\left(\frac{1}{2}\right) = \sum_{i=0}^{2} B_{i,2}(\tfrac{1}{2})\,\mathbf{P}_i^w =$$

$$= \left(1-\left(\frac{1}{2}\right)\right)^2 (1,0,1) + 2\left(\frac{1}{2}\right)\left(1-\left(\frac{1}{2}\right)\right)(1,1,1)$$

$$+ \left(\frac{1}{2}\right)^2 (0,2,2)$$

$$= \frac{1}{4}(1,0,1) + \frac{1}{2}(1,1,1) + \frac{1}{4}(0,2,2) = \left(\frac{3}{4}, 1, \frac{5}{4}\right)$$

Projecting yields $(3/5, 4/5)$. Equations (1.18) and (1.15) yield the same result.

Finally, we note that $\mathbf{C}(1/2) = (3/5, 4/5)$ is not the midpoint of the circular arc in the first quadrant; i.e., the parameterization is not uniform (see Section 1.1). The point $(3/5, 4/5)$ is more than half the arc length from the starting point. This is intuitively correct, since by differentiating $\mathbf{C}(u)$ one can see that the starting speed is twice the end speed.

1.5 Tensor Product Surfaces

The curve $\mathbf{C}(u)$ is a vector-valued function of one parameter. It is a mapping (deformation) of a straight line segment into Euclidean three-dimensional space. A surface is a vector-valued function of two parameters, u and v, and represents a mapping of a region, $\mathcal{R}$, of the uv plane into Euclidean three-dimensional space. Thus it has the form $\mathbf{S}(u,v) = \big(x(u,v), y(u,v), z(u,v)\big)$, $(u,v) \in \mathcal{R}$. There are many schemes for representing surfaces (see [Hosc93; Pieg89a, 93] and the many references cited in [Pieg89a]). They differ in the coordinate functions used and the type of region $\mathcal{R}$. Probably the simplest method, and the one most widely used in geometric modeling applications, is the *tensor product* scheme. This is the method we use in the remainder of this book.

The tensor product method is basically a bidirectional curve scheme. It uses basis functions and geometric coefficients. The basis functions are bivariate functions of u and v, which are constructed as products of univariate basis functions.

The geometric coefficients are arranged (topologically) in a bidirectional, $n \times m$ net. Thus, a tensor product surface has the form

$$\mathbf{S}(u,v) = \big(x(u,v), y(u,v), z(u,v)\big) = \sum_{i=0}^{n}\sum_{j=0}^{m} f_i(u) g_j(v) \mathbf{b}_{i,j} \tag{1.20}$$

where
$$\begin{cases} \mathbf{b}_{i,j} = (x_{i,j}, y_{i,j}, z_{i,j}) \\ 0 \le u, v \le 1 \end{cases}$$

Note that the (u,v) domain of this mapping is a square (a rectangle, in general). Note also that $\mathbf{S}(u,v)$ has a matrix form

$$\mathbf{S}(u,v) = [\,f_i(u)\,]^T\,[\,\mathbf{b}_{i,j}\,]\,[\,g_j(v)\,]$$

where $[\,f_i(u)\,]^T$ is a $(1) \times (n+1)$ row vector, $[\,g_j(v)\,]$ is a $(m+1) \times (1)$ column vector, and $[\,\mathbf{b}_{i,j}\,]$ is a $(n+1) \times (m+1)$ matrix of three-dimensional points.

As an example we consider the power basis surface

$$\mathbf{S}(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} \mathbf{a}_{i,j} u^i v^j = [\,u^i\,]^T\,[\,\mathbf{a}_{i,j}\,]\,[\,v^j\,] \qquad \begin{cases} \mathbf{a}_{i,j} = (x_{i,j}, y_{i,j}, z_{i,j}) \\ 0 \le u, v \le 1 \end{cases} \tag{1.21}$$

We have $f_i(u) = u^i$ and $g_j(v) = v^j$, and the basis functions are the products, $\{u^i v^j\}$. If we fix $u = u_0$, then

$$\mathbf{C}_{u_0}(v) = \mathbf{S}(u_0, v) = \sum_{j=0}^{m}\left(\sum_{i=0}^{n} \mathbf{a}_{i,j}\, u_0^i\right) v^j = \sum_{j=0}^{m} \mathbf{b}_j(u_0)\, v^j \tag{1.22}$$

where
$$\mathbf{b}_j(u_0) = \sum_{i=0}^{n} \mathbf{a}_{i,j}\, u_0^i$$

is a power basis curve lying on the surface, $\mathbf{S}(u,v)$. Similarly, $\mathbf{C}_{v_0}(u)$ is a power basis curve lying on $\mathbf{S}(u,v)$; and the curves $\mathbf{C}_{u_0}(v)$ and $\mathbf{C}_{v_0}(u)$ intersect at the surface point, $\mathbf{S}(u_0, v_0)$. These curves are called *isoparametric curves* (or *isocurves*). $\mathbf{C}_{u_0}(v)$ is called a v *curve*, $\mathbf{C}_{v_0}(u)$ a u *curve* (see Figure 1.23).

Equation (1.21) can be written as

$$\begin{aligned} \mathbf{S}(u,v) = {} & \underbrace{\{\mathbf{a}_{0,0} + \mathbf{a}_{0,1}v + \mathbf{a}_{0,2}v^2 + \cdots + \mathbf{a}_{0,m}v^m\}}_{\mathbf{b}_0} \\ & + u\underbrace{\{\mathbf{a}_{1,0} + \mathbf{a}_{1,1}v + \mathbf{a}_{1,2}v^2 + \cdots + \mathbf{a}_{1,m}v^m\}}_{\mathbf{b}_1} \\ & + u^2\underbrace{\{\mathbf{a}_{2,0} + \mathbf{a}_{2,1}v + \mathbf{a}_{2,2}v^2 + \cdots + \mathbf{a}_{2,m}v^m\}}_{\mathbf{b}_2} \\ & \vdots \\ & + u^n\underbrace{\{\mathbf{a}_{n,0} + \mathbf{a}_{n,1}v + \mathbf{a}_{n,2}v^2 + \cdots + \mathbf{a}_{n,m}v^m\}}_{\mathbf{b}_n} \\ = {} & \mathbf{b}_0 + \mathbf{b}_1 u + \mathbf{b}_2 u^2 + \cdots + \mathbf{b}_n u^n \end{aligned}$$

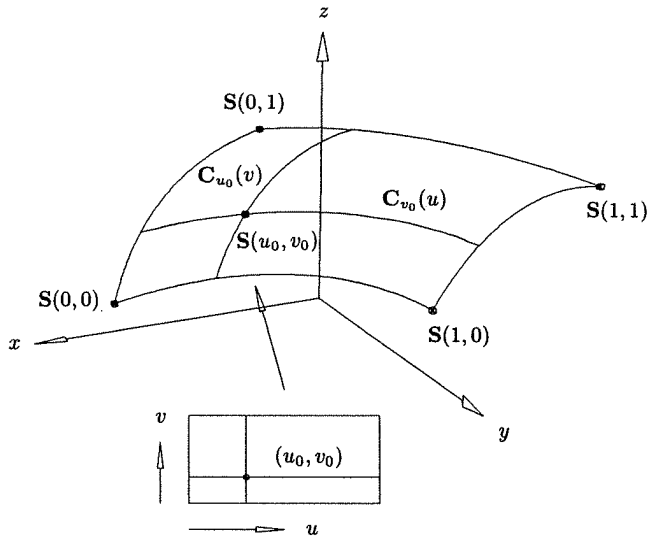


Figure 1.23. A tensor product surface showing isoparametric curves.

The terms in the braces are simple polynomials that can be evaluated by the Horner Algorithm (A1.1), yielding $\mathbf{b}_0, \mathbf{b}_1, \ldots, \mathbf{b}_n$. Using the $\mathbf{b}$s and reapplying the algorithm, we obtain the point on the surface. Thus we have Algorithm A1.6.

```
ALGORITHM A1.6
  Horner2(a,n,m,u0,v0,S)
    { /* Compute point on a power basis surface.  */
      /* Input:  a,n,m,u0,v0  */
      /* Output: S  */
    for (i=0; i<=n; i++)
      Horner1(a[i][],m,v0,b[i]); /* a[i][] is the ith row */
    Horner1(b,n,u0,S);
    }
```

Algorithm A1.6 is typical of the algorithms for tensor product surfaces. They can usually be obtained by extending from the curve algorithms, often by processing the n (or m) rows of coefficients (as curves) in one direction, then processing one or more rows in the other direction.

Differentiating Eq. (1.21), we obtain

$$\mathbf{S}_u(u,v) = \sum_{i=1}^{n}\sum_{j=0}^{m} i\,\mathbf{a}_{i,j}\,u^{i-1}v^j \qquad \mathbf{S}_v(u,v) = \sum_{i=0}^{n}\sum_{j=1}^{m} j\,\mathbf{a}_{i,j}\,u^i v^{j-1}$$

Notice that for fixed (u_0, v_0), $\mathbf{S}_u(u_0,v_0) = \mathbf{C}'_{v_0}(u_0)$ and $\mathbf{S}_v(u_0,v_0) = \mathbf{C}'_{u_0}(v_0)$. The normal vector, $\mathbf{N}$, is computed using Eq. (1.4).

Nonrational Bézier surfaces are obtained by taking a bidirectional net of control points and products of the univariate Bernstein polynomials

$$\mathbf{S}(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} B_{i,n}(u)B_{j,m}(v)\mathbf{P}_{i,j} \qquad 0 \le u,v \le 1 \tag{1.23}$$

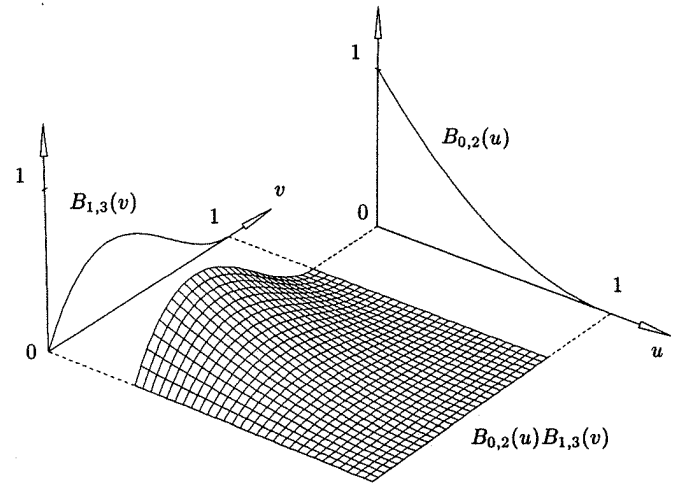
The basis function $B_{0,2}(u)B_{1,3}(v)$ is shown in Figure 1.24a, and Figure 1.24b shows a quadratic × cubic Bézier surface.

For fixed $u = u_0$

$$\begin{aligned}\mathbf{C}_{u_0}(v) = \mathbf{S}(u_0,v) &= \sum_{i=0}^{n}\sum_{j=0}^{m} B_{i,n}(u_0)B_{j,m}(v)\mathbf{P}_{i,j} \\ &= \sum_{j=0}^{m} B_{j,m}(v)\left(\sum_{i=0}^{n} B_{i,n}(u_0)\mathbf{P}_{i,j}\right) \\ &= \sum_{j=0}^{m} B_{j,m}(v)\,\mathbf{Q}_j(u_0)\end{aligned} \tag{1.24}$$

where $\mathbf{Q}_j(u_0) = \sum_{i=0}^{n} B_{i,n}(u_0)\mathbf{P}_{i,j} \qquad j = 0, \ldots, m$

is a Bézier curve lying on the surface. Analogously, $\mathbf{C}_{v_0}(u) = \sum_{i=0}^{n} B_{i,n}(u)\,\mathbf{Q}_i(v_0)$ is a Bézier u isocurve lying on the surface.



(a)

Figure 1.24. (a) The Bézier tensor product basis function, $B_{0,2}(u)B_{1,3}(v)$; (b) a quadratic × cubic Bézier surface.

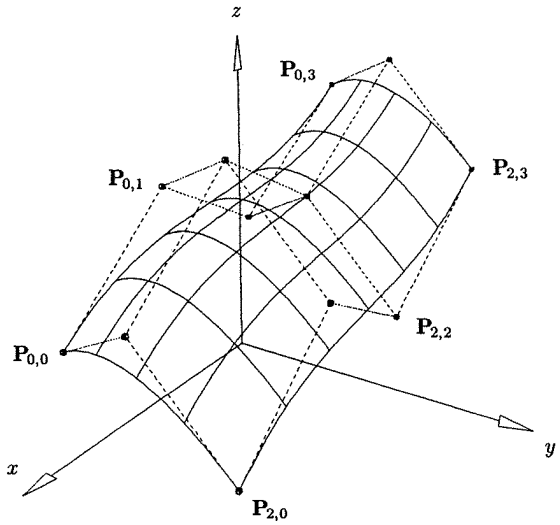
As is the case for curves, because of their excellent properties Bézier surfaces are better suited for geometric modeling applications than power basis surfaces. In particular,

- nonnegativity: $B_{i,n}(u)B_{j,m}(v) \geq 0$ for all i, j, u, v;
- partition of unity: $\sum_{i=0}^{n} \sum_{j=0}^{m} B_{i,n}(u)B_{j,m}(v) = 1$ for all u and v;
- $\mathbf{S}(u, v)$ is contained in the convex hull of its control points;
- transformation invariance;
- the surface interpolates the four corner control points;
- when triangulated, the control net forms a planar polyhedral approximation to the surface.

It is interesting to note that there is no known variation diminishing property for Bézier surfaces (see [Prau92]).

The deCasteljau algorithm (`A1.5`) is also easily extended to compute points on a Bézier surface. Refer to Eq. (1.24) and Figure 1.25. Let (u_0, v_0) be fixed. For fixed j_0, $\mathbf{Q}_{j_0}(u_0) = \sum_{i=0}^{n} B_{i,n}(u_0)\mathbf{P}_{i,j_0}$ is the point obtained by applying the deCasteljau algorithm to the j_0 row of control points, i.e., to $\{\mathbf{P}_{i,j_0}\}$, $i = 0, \ldots, n$. Therefore, applying the deCasteljau Algorithm $(m+1)$ times yields $\mathbf{C}_{u_0}(v)$; and applying it once more to $\mathbf{C}_{u_0}(v)$ at $v = v_0$ yields $\mathbf{C}_{u_0}(v_0) = \mathbf{S}(u_0, v_0)$. This process requires

$$\frac{n(n+1)(m+1)}{2} + \frac{m(m+1)}{2} \tag{1.25}$$



(b)

Figure 1.24. (*Continued.*)

linear interpolations (see Exercise 1.21). By symmetry, we can compute $\mathbf{C}_{v_0}(u)$ first $(n+1$ applications of deCasteljau) and then compute $\mathbf{C}_{v_0}(u_0) = \mathbf{S}(u_0, v_0)$. This requires

$$\frac{m(m+1)(n+1)}{2} + \frac{n(n+1)}{2} \tag{1.26}$$

linear interpolations. Thus, if $n > m$ compute $\mathbf{C}_{v_0}(u)$ first, then $\mathbf{C}_{v_0}(u_0)$; otherwise, compute $\mathbf{C}_{u_0}(v)$ first, then $\mathbf{C}_{u_0}(v_0)$.

```
ALGORITHM A1.7
  deCasteljau2(P,n,m,u0,v0,S)
    { /*  Compute a point on a Bézier surface */
      /*  by the deCasteljau.  */
      /*  Input:  P,n,m,u0,v0  */
      /*  Output: S
    if (n <= m)
      {
      for (j=0; j<=m; j++) /* P[j][] is jth row */
        deCasteljau1(P[j][],n,u0,Q[j]);
      deCasteljau1(Q,m,v0,S);
      }
    else
      {
      for (i=0; i<=n; i++)
```

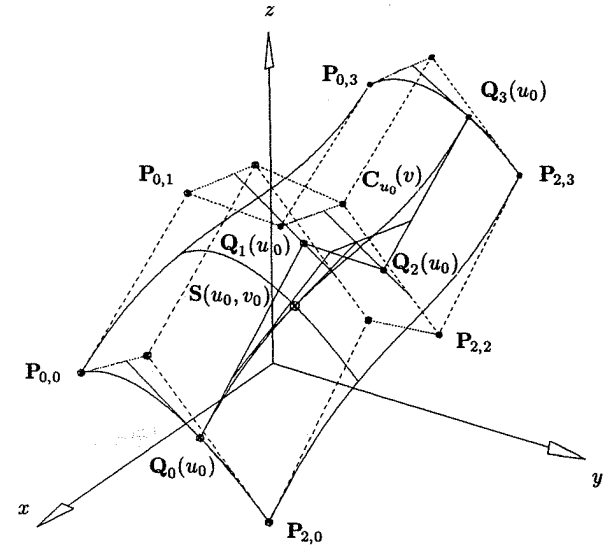


Figure 1.25. The deCasteljau algorithm for a Bézier surface.

```
        deCasteljau1(P[][i],m,v0,Q[i]);
      deCasteljau1(Q,n,u0,S);
      }
}
```

We define a *rational Bézier surface* to be the perspective projection of a four-dimensional polynomial Bézier surface (see [Pieg86; Fari89])

$$\mathbf{S}^w(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} B_{i,n}(u)B_{j,m}(v)\mathbf{P}^w_{i,j} \tag{1.27}$$

and

$$\mathbf{S}(u,v) = H\{\mathbf{S}^w(u,v)\} = \frac{\displaystyle\sum_{i=0}^{n}\sum_{j=0}^{m} B_{i,n}(u)B_{j,m}(v)w_{i,j}\mathbf{P}_{i,j}}{\displaystyle\sum_{i=0}^{n}\sum_{j=0}^{m} B_{i,n}(u)B_{j,m}(v)w_{i,j}}$$

$$= \sum_{i=0}^{n}\sum_{j=0}^{m} R_{i,j}(u,v)\mathbf{P}_{i,j} \tag{1.28}$$

where

$$R_{i,j}(u,v) = \frac{B_{i,n}(u)B_{j,m}(v)w_{i,j}}{\displaystyle\sum_{r=0}^{n}\sum_{s=0}^{m} B_{r,n}(u)B_{s,m}(v)w_{r,s}}$$

Notice that the $R_{i,j}(u,v)$ are rational functions, but they are not products of other basis functions. Hence, $\mathbf{S}(u,v)$ is not a tensor product surface, but $\mathbf{S}^w(u,v)$ is. As with curves, we generally work with Eq. (1.27) and project the results. Figure 1.26a shows a rational basis function, and Figure 1.26b depicts a quadratic × cubic rational Bézier surface. Compare these figures with Figures 1.24a and 1.24b.

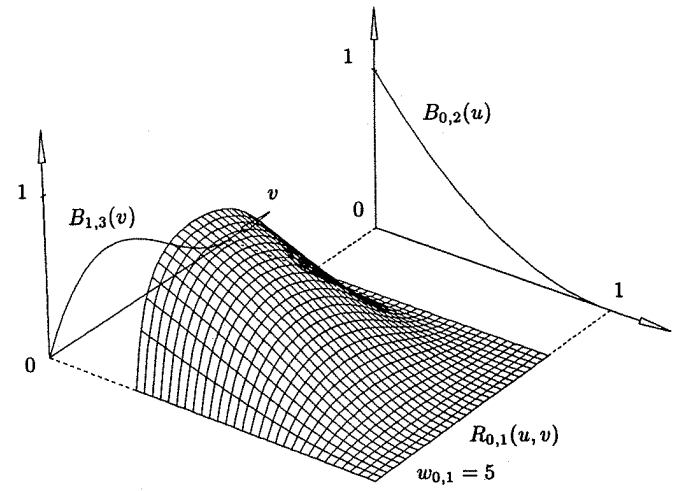
Assuming $w_{i,j} > 0$ for all i and j, the properties listed previously for nonrational Bézier surfaces (and the product functions $B_{i,n}(u)B_{j,m}(v)$) extend naturally to rational Bézier surfaces. Furthermore, if $w_{i,j} = 1$ for all i and j, then $R_{i,j}(u,v) = B_{i,n}(u)B_{j,m}(v)$, and the corresponding surface is nonrational.

Example

Ex1.15 Let us construct a cylindrical surface patch. From Section 1.4 we know that

$$\mathbf{C}^w(u) = \sum_{i=0}^{2} B_{i,2}(u)\mathbf{P}^w_i$$

for $\{\mathbf{P}^w_i\} = \{(0,1,0,1),(0,1,1,1),(0,0,2,2)\}$, is a circular arc in the yz plane. Using translation (P1.14, Section 1.4)



(a)

Figure 1.26. (a) The rational basis function $R_{0,1}(u,v)$ (with $w_{0,1} = 5$ and all other weights equal to one); (b) a quadratic × cubic rational Bézier surface.

$$\mathbf{C}^w_0(u) = \sum_{i=0}^{2} B_{i,2}(u)\mathbf{P}^w_{i,0} \quad \text{and} \quad \mathbf{C}^w_1(u) = \sum_{i=0}^{2} B_{i,2}(u)\mathbf{P}^w_{i,1}$$

where $$\{\mathbf{P}^w_{i,0}\} = \{(1,1,0,1),(1,1,1,1),(2,0,2,2)\}$$

and $$\{\mathbf{P}^w_{i,1}\} = \{(-1,1,0,1),(-1,1,1,1),(-2,0,2,2)\}$$

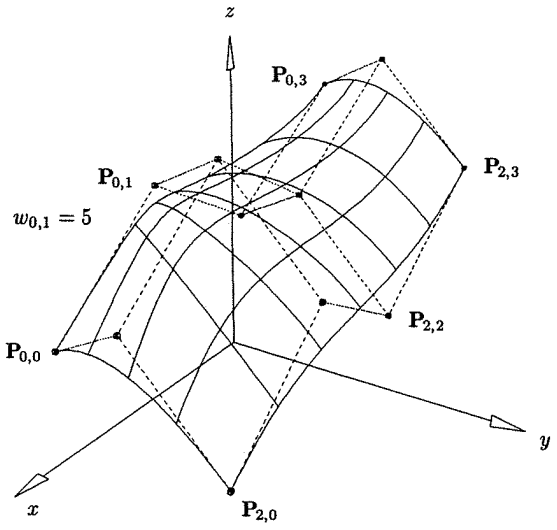
are circular arcs in the $x = 1$ and $x = -1$ planes, respectively (see Figure 1.27). A linear interpolation between $\mathbf{C}^w_0$ and $\mathbf{C}^w_1$ yields a cylindrical surface, i.e.

$$\mathbf{S}^w(u,v) = \sum_{i=0}^{2}\sum_{j=0}^{1} B_{i,2}(u)B_{j,1}(v)\mathbf{P}^w_{i,j}$$

For fixed $u = u_0$, $\mathbf{C}^w_{u_0}(v) = \sum_{j=0}^{1} B_{j,1}(v)\mathbf{Q}^w_j(u_0)$ is a straight line segment from $\mathbf{C}^w_0(u_0)$ to $\mathbf{C}^w_1(u_0)$ parallel to the x-axis. For fixed $v = v_0$, $\mathbf{C}^w_{v_0} = \mathbf{S}^w(u,v_0) = \sum_{i=0}^{2} B_{i,2}(u)\mathbf{Q}^w_i(v_0)$ is a circular arc in the plane $x = (1-v_0)(1) + v_0(-1) = 1 - 2v_0$. Now let us compute the point $\mathbf{S}(1/2, 1/2)$, using Algorithm A1.7. Note that $n > m$. First obtain $\mathbf{C}^w_{v_0=1/2}(u)$

$$(1,1,0,1) \qquad\qquad (0,1,0,1) = \mathbf{Q}^w_0(v_0)$$

$$(-1,1,0,1)$$



(b)

Figure 1.26. (b) (*Continued.*)

$$\begin{matrix}(1,1,1,1)\\(-1,1,1,1)\end{matrix} \qquad (0,1,1,1) = \mathbf{Q}_1^w(v_0)$$

$$\begin{matrix}(2,0,2,2)\\(-2,0,2,2)\end{matrix} \qquad (0,0,2,2) = \mathbf{Q}_2^w(v_0)$$

Now $\mathbf{C}^w_{v_0=1/2}(u) = \sum_{i=0}^{2} B_{i,2}(u)\,\mathbf{Q}_i^w(v_0)$ is the circular arc in the yz plane. Then

$$\begin{matrix}(0,1,0,1)\\ \\(0,1,1,1)\\ \\(0,0,2,2)\end{matrix} \quad \begin{matrix}\left(0,1,\frac{1}{2},1\right)\\ \\ \left(0,\frac{3}{4},1,\frac{5}{4}\right) = \mathbf{S}^w\left(\frac{1}{2},\frac{1}{2}\right)\\ \\ \left(0,\frac{1}{2},\frac{3}{2},\frac{3}{2}\right)\end{matrix}$$

And projecting yields

$$\mathbf{S}\left(\frac{1}{2},\frac{1}{2}\right) = H\left\{\left(0,\frac{3}{4},1,\frac{5}{4}\right)\right\} = \left(0,\frac{3}{5},\frac{4}{5}\right)$$

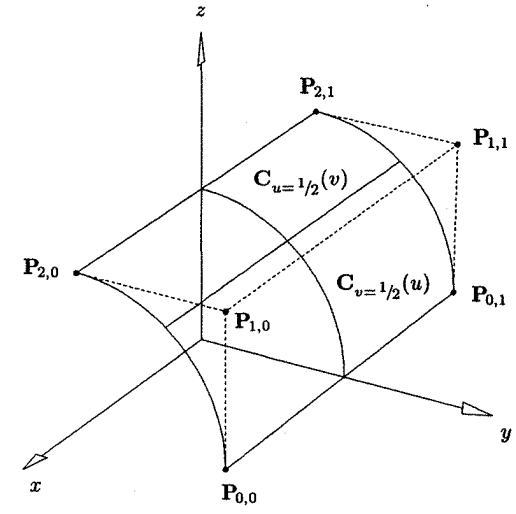


Figure 1.27. A cylindrical surface patch as a rational Bézier surface.

Exercises

1.1. Consider the two parametric representations of the circular arc given by Eqs. (1.1) and (1.2). Using Eq. (1.1), compute the curve point at $u = \pi/4$ and, using Eq. (1.2), the point at $t = 1/2$. Explain the results.

1.2. Compute the acceleration vector, $\mathbf{C}''(u)$, for Eq. (1.1). Explain the result.

1.3. Using trigonometric functions, give a parametric definition of the bounded surface of

- a right circular cone, with apex at the origin and axis of symmetry along the z-axis;
- the cone is opening upward, and is bounded above at $z = 2$ by the circle with radius = 1.

Modify Eq. (1.2) to get another representation of the same cone. Compute the first partial derivatives, $\mathbf{S}_u$ and $\mathbf{S}_v$, of the trigonometric representation. What are the values of these derivatives at the apex of the cone?

1.4. Consider the parabolic arc $\mathbf{C}(u) = (x(u), y(u)) = (-1 - u + 2u^2, -2u + u^2)$, $0 \le u \le 1$. Sketch this curve. The curve is rotated and translated by applying the transformations to the functions $x(u)$ and $y(u)$. Apply the two transformations

(1) 90° rotation about the origin. The rotation matrix (applied from the left) is

$$\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

(2) translation with the vector $(-1, -1)$.

The implicit equation of the underlying parabola is $x^2 - 4xy + 4y^2 - 4x - y - 5 = 0$. Sketch this curve. Apply the previous rotation and translation to this equation. Hint: let $\bar{x}, \bar{y}$ be the transformed coordinates. Find expressions $x = f(\bar{x}, \bar{y})$ and $y = g(\bar{x}, \bar{y})$ and substitute these into the implicit equation to obtain the implicit equation of the transformed parabola.

1.5. Determine formulas for the number of additions and multiplications necessary to compute a point on an nth-degree three-dimensional power basis curve.

1.6. Construct a cubic power basis curve with a loop. Hint: think about what endpoints and end derivatives, $\mathbf{C}'(0)$ and $\mathbf{C}'(1)$, are necessary.

1.7. Construct a cubic power basis curve with a cusp. Hint: think about $\mathbf{C}'(u)$ and $\mathbf{C}''(u)$. Sketch what $x'(u)$, $y'(u)$, $x''(u)$, and $y''(u)$ need to look like as functions of u. Determine a suitable $\mathbf{C}''(u)$, and then integrate to obtain $\mathbf{C}'(u)$ and $\mathbf{C}(u)$.

1.8. Construct a cubic power basis curve with an inflection point.

1.9. Let $\mathbf{C}(u) = (x(u), y(u)) = (1 + u - 2u^2 + u^3, 1 - 2u + u^3)$, $-1 \le u \le 1$. Let $u = 2v - 1$. Derive the curve $\mathbf{C}(v)$ by substituting $2v - 1$ for u in $\mathbf{C}(u)$. What degree is the curve $\mathbf{C}(v)$? Compute $\mathbf{C}(u)$ for $u = -1, 0, 1$. Compute $\mathbf{C}(v)$ for $v = 0, 1/2, 1$. What can you say about the curves $\mathbf{C}(u)$ and $\mathbf{C}(v)$? $\mathbf{C}(v)$ is called a *reparameterization* of $\mathbf{C}(u)$.

1.10. Check the property `P1.7` of the Bernstein polynomials for the cases $n = 2$ and $n = 3$.

1.11. It is sometimes necessary to reverse a curve, i.e., given $\mathbf{C}_1(u)$, $0 \le u \le 1$, produce $\mathbf{C}_2(v)$, $0 \le v \le 1$, such that the two curves are the same geometrically, but $\mathbf{C}_1(0) = \mathbf{C}_2(1)$ and $\mathbf{C}_1(1) = \mathbf{C}_2(0)$. How would you do this using the Bézier form? The power basis form?

1.12. Consider the xy planar cubic Bézier curve given by the control points $\mathbf{P}_0 = (0, 6)$, $\mathbf{P}_1 = (3, 6)$, $\mathbf{P}_2 = (6, 3)$, $\mathbf{P}_3 = (6, 0)$. Compute the point $\mathbf{C}(1/3)$ using the deCasteljau algorithm. Compute the same point by using Eqs. (1.7) and (1.8) directly, i.e., evaluate the basis functions at $u = 1/3$ and multiply by the appropriate control points.

1.13. Determine formulas for the number of additions and multiplications necessary to compute a point on an nth-degree three-dimensional Bézier curve using the deCasteljau algorithm, `A1.5`, and algorithm `A1.4`. Compare the results with Exercise 1.5 (the Horner algorithm).

1.14. For given n, row vector $[B_{0,n}(u), \ldots, B_{n,n}(u)]$ can be written as $[1, u, \ldots, u^n]M$, where M is an $(n+1) \times (n+1)$ matrix. Thus, a Bézier curve can be written in matrix form, $\mathbf{C}(u) = [u^i]^T M [\mathbf{P}_i]$. Compute the matrices M for $n = 1, 2, 3$. Notice that setting $[\mathbf{a}_i] = M[\mathbf{P}_i]$ yields the conversion of a Bézier curve to power basis form. Assuming $0 \le u \le 1$, $[\mathbf{P}_i] = M^{-1}[\mathbf{a}_i]$ gives the conversion from power basis to Bézier form.

1.15. Compare this with Exercise 1.9. It is also possible to define a Bézier curve on a parameter interval other than $[0, 1]$. This is equivalent to reparameterizing a Bézier curve. Let

$$\mathbf{C}(u) = \sum_{i=0}^{n} B_{i,n}(u)\mathbf{P}_i \qquad u \in [0, 1]$$

Let $v \in [a, b]$. Then $u = (v - a)/(b - a)$. Substitute this equation into Eq. (1.8) and derive this expression for the reparameterized curve

$$\mathbf{C}(v) = \frac{1}{(b-a)^n} \sum_{i=0}^{n} \frac{n!}{i!(n-i)!} (v-a)^i (b-v)^{n-i} \mathbf{P}_i$$

It is interesting to note that the control points do not change, only the basis functions. Reparameterization of the power basis form changes the geometric coefficients but not the basis functions.

1.16. Consider the circle

$$\mathbf{C}(u) = \left(\frac{1 - u^2}{1 + u^2}, \frac{2u}{1 + u^2} \right)$$

Determine which ranges of the parameter u yield which quadrants of the circle. Do these equations yield the entire circle? What can you say about the parameterization?

1.17. Consider the following rational cubic Bézier curve in the xy plane: $\mathbf{P}_0 = (0, 6)$, $\mathbf{P}_1 = (3, 6)$, $\mathbf{P}_2 = (6, 3)$, $\mathbf{P}_3 = (6, 0)$, $w_0 = 4$, $w_1 = 1$, $w_2 = 1$, $w_3 = 4$. Compute the point $\mathbf{C}(2/3)$ by expanding the deCasteljau table.

1.18. What characteristic is it of the rational functions we are using that allows us to use the homogeneous coordinate representation? Why is this representation advantageous?

1.19. Find the rational Bézier representation of the circular arc in the second quadrant, i.e., determine the $\mathbf{P}_i$ and w_i. Hint: use symmetry and check your result by showing that $(x(u))^2 + (y(u))^2 = 1$ for all $u \in [0, 1]$.

1.20. The circular arc in the first quadrant is also given by the equation

$$\mathbf{C}(u) = \left(\frac{1 + (\sqrt{2} - 2)u + (1 - \sqrt{2})u^2}{1 + (\sqrt{2} - 2)u + (2 - \sqrt{2})u^2}, \frac{\frac{\sqrt{2}}{2} u\Big((\sqrt{2} - 2)u + 2\Big)}{1 + (\sqrt{2} - 2)u + (2 - \sqrt{2})u^2} \right)$$

Determine the rational Bézier representation corresponding to these equations. Hint: the $\mathbf{P}_i$ must be the same as before – $(1, 0)$, $(1, 1)$, $(0, 1)$; Why? Compute the weights w_i by equating polynomials and substituting $u = 0, 1/2, 1$, as done previously. Compute the point $\mathbf{C}(1/2)$, using any method. What is interesting about $\mathbf{C}(1/2)$?

1.21. Derive Eqs. (1.25) and (1.26). Hint: use the formula $1 + 2 + \cdots + n = n(n+1)/2$.

1.22. For the cylindrical surface example (`Ex1.15`) compute the control points $\mathbf{Q}_j^w(u_0)$ for the isocurve $\mathbf{C}^w_{u_0 = 1/3}(v)$.

1.23. Let $n = 3$ and $m = 2$. Consider the nonrational Bézier surface defined by the control net

$$\{\mathbf{P}_{i,0}\} = \{(0, 0, 0), (3, 0, 3), (6, 0, 3), (9, 0, 0)\}$$

$$\{\mathbf{P}_{i,1}\} = \{(0, 2, 2), (3, 2, 5), (6, 2, 5), (9, 2, 2)\}$$

$$\{\mathbf{P}_{i,2}\} = \{(0, 4, 0), (3, 4, 3), (6, 4, 3), (9, 4, 0)\}$$

a. sketch this surface;

b. use the deCasteljau algorithm to compute the surface point $\mathbf{S}(1/3, 1/2)$;

c. fix $u_0 = 1/2$ and extract the Bézier representation (control points) of the curve $\mathbf{C}_{u_0 = 1/2}(v)$.

1.24. Let

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

and assume that $\mathbf{P}_{0,0} = \mathbf{P}_{1,0} = \cdots = \mathbf{P}_{n,0}$. How does this affect $\mathbf{S}(u,v)$, the derivatives $\mathbf{S}_u(u,v)$ and $\mathbf{S}_v(u,v)$, and the curves $\mathbf{C}_{v_0}(u)$? Assume that $\mathbf{P}_{i,0} = (1,0,0)$ for $i = 0,1,2$ in Example `Ex1.15`, with $w_{0,0} = 1$, $w_{1,0} = 1$, and $w_{2,0} = 2$. What type of surface do you get?

1.25. The prerequisite for this problem is Exercise 1.14. The rational Bézier surface (Eq. [1.27]) has a matrix form

$$\mathbf{S}^w(u,v) = [\, B_{i,n}(u) \,]^T \, [\, \mathbf{P}^w_{i,j} \,] \, [\, B_{j,m}(v) \,] = [\, u^i \,]^T \, M_n \, [\, \mathbf{P}^w_{i,j} \,] \, M_m^T \, [\, v^j \,]$$

where $[\, u^i \,]^T$ and $[\, v^j \,]$ are vectors, M_n is an $(n+1) \times (n+1)$ matrix, M_m^T is a $(m+1) \times (m+1)$ matrix, and $[\, \mathbf{P}^w_{i,j} \,]$ is an $(n+1) \times (m+1)$ matrix of four-dimensional points. Write this form down explicitly for the cylindrical surface example, `Ex1.15`. Using this matrix form, compute the point $\mathbf{S}^w(1/2, 1/2)$, and then project to obtain $\mathbf{S}(1/2, 1/2)$. There is no direct matrix form for $\mathbf{S}(u,v)$; why not?