Spline Representations

Splines are another example of boundary representation modeling techniques. In drafting terminology, a spline is a flexible strip used to produce a smooth curve through a designated set of points. Several small weights are distributed along the length of the strip to hold it in position on the drafting table as the curve is drawn. The term *spline curve* originally referred to a curve drawn in this manner. We can mathematically describe such a curve with a piecewise cubic polynomial function whose first and second derivatives are continuous across the various curve sections. In computer graphics, the term **spline curve** now refers to any composite curve formed with polynomial sections satisfying any specified continuity conditions at the boundary of the pieces. A **spline surface** can be described with two sets of spline curves. There are several different kinds of spline specifications that are used in computer-graphics applications. Each individual specification simply refers to a particular type of polynomial with certain prescribed boundary conditions.

Splines are used to design curve and surface shapes, to digitize drawings, and to specify animation paths for the objects or the camera position in a scene. Typical computer-aided design (CAD) applications for splines include the design of automobile bodies, aircraft and spacecraft surfaces, ship hulls, and home appliances.

1 Interpolation and Approximation Splines

We specify a spline curve by giving a set of coordinate positions, called **control points,** which indicate the general shape of the curve. These coordinate positions are then fitted with piecewise-continuous, parametric polynomial functions in one of two ways. When polynomial sections are fitted so that all the control points are connected, as in Figure 1, the resulting curve is said to **interpolate** the set of control points. On the other hand, when the generated polynomial curve is plotted so that some, or all, of the control points are not on the curve path, the resulting curve is said to **approximate** the set of control points (Figure 2). Similar methods are used to construct interpolation or approximation spline surfaces.

FIGURE 1
A set of six control points interpolated with piecewise continuous polynomial sections.

Interpolation methods are commonly used to digitize drawings or to specify animation paths. Approximation methods are used primarily as design tools to create object shapes. Figure 3 shows the screen display of an approximation spline surface for a design application. Straight lines connect the control-point positions above the surface.

FIGURE 2
A set of six control points approximated with piecewise continuous polynomial sections.

A spline curve or surface is defined, modified, and manipulated with operations on the control points. By interactively selecting spatial positions for the control points, a designer can set up an initial shape. After the polynomial fit is displayed for a given set of control points, the designer can then reposition some of or all the control points to restructure the shape of the object. Geometric transformations (translation, rotation, and scaling) are applied to the object by transforming the control points. In addition, CAD packages sometimes insert extra control points to aid a designer in adjusting the object shapes.

A set of control points forms a boundary for a region of space that is called the **convex hull.** One way to envision the shape of a convex hull for a two-dimensional curve is to imagine a rubber band stretched around the positions of the control

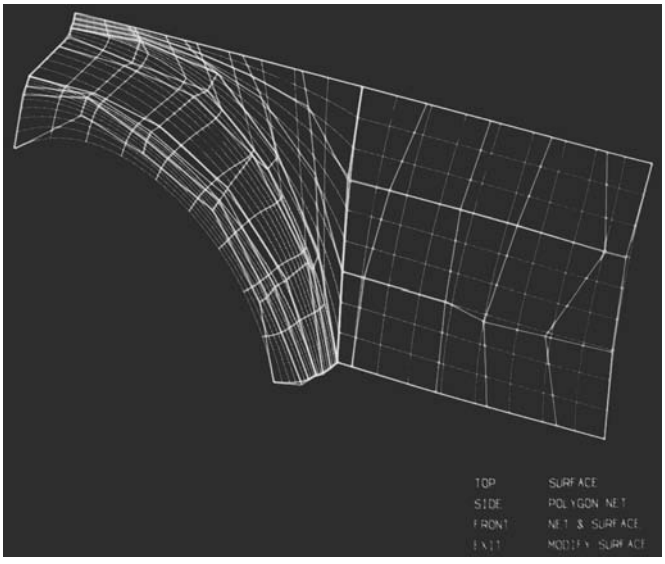

FIGURE 3
An approximation spline surface for a CAD application in automotive design. Surface contours are plotted with polynomial curve sections, and the surface control points are connected with straight-line segments. (*Courtesy of Evans & Sutherland.*)

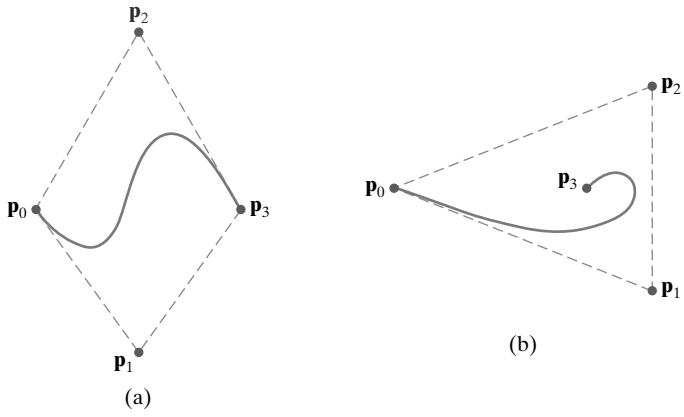

FIGURE 4
Convex-hull shapes (dashed lines) for two sets of control points in the *x y* plane.

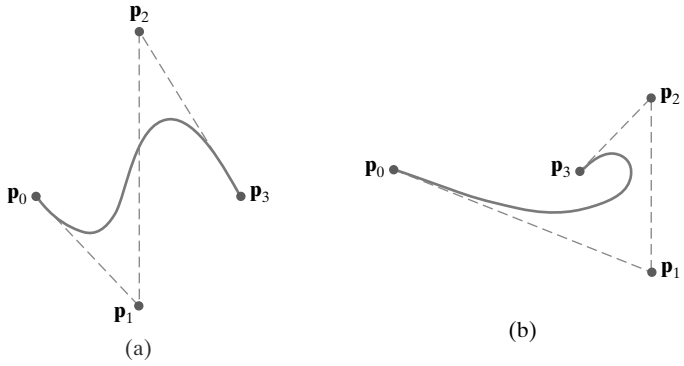

FIGURE 5
Control-graph shapes (dashed lines) for two sets of control points in the *x y* plane.

points so that each control point is either on the perimeter of this boundary or inside it (Figure 4). Thus, the convex hull for a two-dimensional spline curve is a convex polygon. In three-dimensional space, the convex hull for a set of spline control points forms a convex polyhedron. Convex hulls provide a measure for the deviation of a curve or surface from the region of space near the control points. In most cases, a spline is bounded by its convex hull, which ensures that the object shape follows the control points without erratic oscillations. Also, the convex hull provides a measure of the coordinate extents of a designed curve or surface, so it is useful in clipping and viewing routines.

A polyline connecting the sequence of control points for an approximation spline curve is usually displayed to remind a designer of the control-point positions and ordering. This set of connected line segments is called the **control graph** for the curve. Often the control graph is alluded to as the "control polygon" or the "characteristic polygon," even though the control graph is a polyline and not a polygon. Figure 5 shows the shape of the control graph for the control-point sequences in Figure 4. For a spline surface, two sets of polyline control-point connectors form the edges for the polygon facets in a quadrilateral mesh for the surface control graph, as in Figure 3.

2 Parametric Continuity Conditions

To ensure a smooth transition from one section of a piecewise parametric spline to the next, we can impose various **continuity conditions** at the connection points. If each section of a spline curve is described with a set of parametric coordinate

functions of the form

$$x = x(u), \qquad y = y(u), \qquad z = z(u), \qquad u_1 \le u \le u_2 \tag{1}$$

we set **parametric continuity** by matching the parametric derivatives of adjoining curve sections at their common boundary.

Zero-order parametric continuity, represented as C^0 continuity, means simply that the curves meet. That is, the values of x, y, and z evaluated at u_2 for the first curve section are equal, respectively, to the values of x, y, and z evaluated at u_1 for the next curve section. **First-order parametric continuity,** referred to as C^1 continuity, means that the first parametric derivatives (tangent lines) of the coordinate functions in Equation 1 for two successive curve sections are equal at their joining point. **Second-order parametric continuity,** or C^2 continuity, means that both the first and second parametric derivatives of the two curve sections are the same at the intersection. Higher-order parametric continuity conditions are defined similarly. Figure 6 shows examples of C^0, C^1, and C^2 continuity.

With second-order parametric continuity, the rates of change of the tangent vectors of connecting sections are equal at their intersection. Thus, the tangent line transitions smoothly from one section of the curve to the next [Figure 6(c)]. With first-order parametric continuity, however, the rate of change of tangent vectors for the two sections can be quite different [Figure 6(b)], so that the general shapes of the two adjacent sections can change abruptly. First-order parametric continuity is often sufficient for digitizing drawings and some design applications, while second-order parametric continuity is useful for setting up animation paths for camera motion and for many precision CAD requirements. A camera traveling along the curve path in Figure 6(b) with equal steps in parameter u would experience an abrupt change in acceleration at the boundary of the two sections, producing a discontinuity in the motion sequence. But if the camera was traveling along the path in Figure 6(c), the frame sequence for the motion would smoothly transition across the boundary.

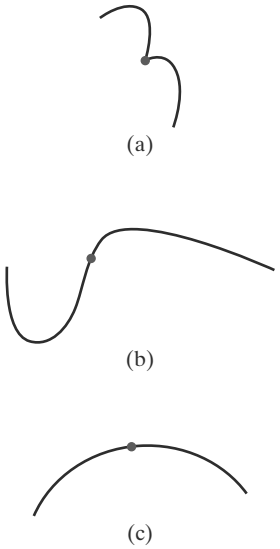

FIGURE 6
Piecewise construction of a curve by joining two curve segments using different orders of continuity: (a) zero-order continuity only, (b) first-order continuity, and (c) second-order continuity.

3 Geometric Continuity Conditions

Another method for joining two successive curve sections is to specify conditions for **geometric continuity.** In this case, we require only that the parametric derivatives of the two sections are proportional to each other at their common boundary, instead of requiring equality.

Zero-order geometric continuity, described as G^0 continuity, is the same as zero-order parametric continuity. That is, two successive curve sections must have the same coordinate position at the boundary point. **First-order geometric continuity,** or G^1 continuity, means that the parametric first derivatives are proportional at the intersection of two successive sections. If we denote the parametric position on the curve as $\mathbf{P}(u)$, the direction of the tangent vector $\mathbf{P}'(u)$, but not necessarily its magnitude, will be the same for two successive curve sections at their common point under G^1 continuity. **Second-order geometric continuity,** or G^2 continuity, means that both the first and second parametric derivatives of the two curve sections are proportional at their boundary. Under G^2 continuity, curvatures of two curve sections will match at the joining position.

A curve generated with geometric continuity conditions is similar to one generated with parametric continuity, but with slight differences in curve shape. Figure 7 provides a comparison of geometric and parametric continuity. With geometric continuity, the curve is pulled toward the section with the greater magnitude for the tangent vector.

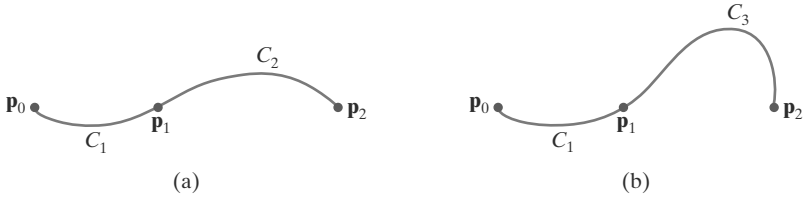

FIGURE 7
Three control points fitted with two curve sections joined with (a) parametric continuity and (b) geometric continuity, where the tangent vector of curve C_3 at point $\mathbf{P}_1$ has a greater magnitude than the tangent vector of curve C_1 at $\mathbf{P}_1$.

4 Spline Specifications

There are three equivalent methods for specifying a particular spline representation, given the degree of the polynomial and the control-point positions: (1) We can state the set of boundary conditions that are imposed on the spline; or (2) we can state the matrix that characterizes the spline; or (3) we can state the set of *blending functions* (or *basis functions*) that determine how specified constraints on the curve are combined to calculate positions along the curve path.

To illustrate these three equivalent specifications, suppose we have the following parametric cubic polynomial representation for the x coordinate along the path of a spline-curve section:

$$x(u) = a_x u^3 + b_x u^2 + c_x u + d_x, \qquad 0 \le u \le 1 \tag{2}$$

Boundary conditions for this curve can be set for the endpoint coordinate positions $x(0)$ and $x(1)$ and for the parametric first derivatives at the endpoints: $x'(0)$ and $x'(1)$. These four boundary conditions are sufficient to determine the values of the four coefficients a_x, b_x, c_x, and d_x.

From the boundary conditions, we can obtain the matrix that characterizes this spline curve by first rewriting Equation 2 as the following matrix product:

$$\begin{aligned} x(u) &= [u^3 \quad u^2 \quad u \quad 1] \begin{bmatrix} a_x \\ b_x \\ c_x \\ d_x \end{bmatrix} \\ &= \mathbf{U} \cdot \mathbf{C} \end{aligned} \tag{3}$$

where $\mathbf{U}$ is the row matrix of powers of parameter u and $\mathbf{C}$ is the coefficient column matrix. Using Equation 3, we can write the boundary conditions in matrix form and solve for the coefficient matrix $\mathbf{C}$ as

$$\mathbf{C} = \mathbf{M}_{\text{spline}} \cdot \mathbf{M}_{\text{geom}} \tag{4}$$

where $\mathbf{M}_{\text{geom}}$ is a four-element column matrix containing the geometric constraint values (boundary conditions) on the spline, and $\mathbf{M}_{\text{spline}}$ is the 4 by 4 matrix that transforms the geometric constraint values to the polynomial coefficients and provides a characterization for the spline curve. Matrix $\mathbf{M}_{\text{geom}}$ contains control-point coordinate values and other geometric constraints that have been specified. Thus, we can substitute the matrix representation for $\mathbf{C}$ into Equation 3 to obtain

$$x(u) = \mathbf{U} \cdot \mathbf{M}_{\text{spline}} \cdot \mathbf{M}_{\text{geom}} \tag{5}$$

The matrix $\mathbf{M}_{\text{spline}}$, characterizing a spline representation, sometimes called the *basis matrix,* is particularly useful for transforming from one spline representation to another.

Finally, we can expand Equation 5 to obtain a polynomial representation for coordinate x in terms of the geometric constraint parameters g_k, such as the control-point coordinates and slope of the curve at the control points:

$$x(u) = \sum_{k=0}^{3} g_k \cdot \mathrm{BF}_k(u) \tag{6}$$

The polynomials $\mathrm{BF}_k(u)$, for $k = 0, 1, 2, 3$, are called **blending functions** or **basis functions** because they combine (blend) the geometric constraint values to obtain coordinate positions along the curve. In subsequent sections, we explore the features of the various spline curves and surfaces that are useful in computer-graphics applications, including the specification of their matrix and blending-function representations.

5 Spline Surfaces

The usual procedure for defining a spline surface is to specify two sets of spline curves using a mesh of control points over some region of space. If we denote the control-point positions as $\mathbf{p}_{k_u,k_v}$, then any point position on the spline surface can be computed as the product of the spline-curve blending functions as follows:

$$\mathbf{P}(u, v) = \sum_{k_u,k_v} \mathbf{p}_{k_u,k_v} \mathrm{BF}_{k_u}(u)\mathrm{BF}_{k_v}(v) \tag{7}$$

Surface parameters u and v often vary over the range from 0 to 1, but this range depends on the type of spline curves we use. One method for designating the three-dimensional control-point positions is to select height values above a two-dimensional mesh of positions on a ground plane.

6 Trimming Spline Surfaces

In CAD applications, a surface design may require some features that are not implemented just by adjusting control-point positions. For instance, a section of a spline surface may need to be snipped off to fit two design pieces together, or a hole may be needed so that a conduit can pass through the surface. For these applications, graphics packages often provide functions to generate **trimming curves** that can be used to take out sections of a spline surface, as illustrated in Figure 8. Trimming curves are typically defined in parametric uv surface coordinates, and often they must be specified as closed curves.

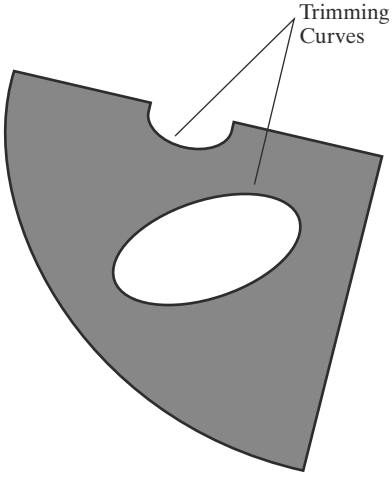

FIGURE 8
Modification of a surface section using trimming curves.

7 Cubic-Spline Interpolation Methods

This class of splines is most often used to set up paths for object motions or to provide a representation for an existing object or drawing, but interpolation splines are also used sometimes to design object shapes. Cubic polynomials offer a reasonable compromise between flexibility and speed of computation. Compared to higher-order polynomials, cubic splines require less calculations and storage space, and they are more stable. Compared to quadratic polynomials and straight-line segments, cubic splines are more flexible for modeling object shapes.

Given a set of control points, cubic interpolation splines are obtained by fitting the input points with a piecewise cubic polynomial curve that passes through every control point. Suppose that we have $n + 1$ control points specified with coordinates

$$\mathbf{p}_k = (x_k, y_k, z_k), \qquad k = 0, 1, 2, \ldots, n$$

A cubic interpolation fit of these points is illustrated in Figure 9. We can describe the parametric cubic polynomial that is to be fitted between each pair of control points with the following set of equations:

$$\begin{aligned} x(u) &= a_x u^3 + b_x u^2 + c_x u + d_x \\ y(u) &= a_y u^3 + b_y u^2 + c_y u + d_y, \qquad (0 \le u \le 1) \\ z(u) &= a_z u^3 + b_z u^2 + c_z u + d_z \end{aligned} \tag{8}$$

For each of these three equations, we need to determine the values for the four coefficients a, b, c, and d in the polynomial representation for each of the n curve sections between the $n + 1$ control points. We do this by setting enough boundary conditions at the control-point positions between curve sections so that we can obtain numerical values for all the coefficients. In the following sections, we discuss common methods for setting the boundary conditions for cubic interpolation splines.

Natural Cubic Splines

One of the first spline curves to be developed for graphics applications is the **natural cubic spline.** This interpolation curve is a mathematical representation of the original drafting spline. We formulate a natural cubic spline by requiring that two adjacent curve sections have the same first and second parametric derivatives at their common boundary. Thus, natural cubic splines have C^2 continuity.

If we have $n + 1$ control points, as in Figure 9, then we have n curve sections with a total of $4n$ polynomial coefficients to be determined. At each of the $n - 1$ interior control points, we have four boundary conditions: The two curve sections on either side of a control point must have the same first and second parametric derivatives at that control point, and each curve must pass through that control point. This gives us $4n - 4$ equations to be satisfied by the $4n$ polynomial coefficients. We obtain an additional equation from the first control point $\mathbf{p}_0$, the position of the beginning of the curve, and another condition from control point $\mathbf{p}_n$, which must be the last point on the curve. However, we still need

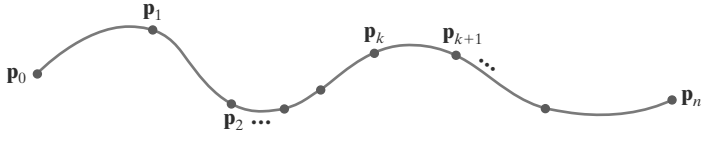

FIGURE 9
A piecewise continuous cubic-spline interpolation of $n + 1$ control points.

two more conditions to be able to determine values for all the coefficients. One method for obtaining the two additional conditions is to set the second derivatives at $\mathbf{p}_0$ and $\mathbf{p}_n$ equal to 0. Another approach is to add two extra control points (called *dummy points*), one at each end of the original control-point sequence. That is, we add a control point labeled $\mathbf{p}_{-1}$ at the beginning of the curve and a control point labeled $\mathbf{p}_{n+1}$ at the end. Then all the original control points are interior points, and we have the necessary $4n$ boundary conditions.

Although natural cubic splines are a mathematical model for the drafting spline, they have a major disadvantage. If the position of any of the control points is altered, the entire curve is affected. Thus, natural cubic splines allow for no "local control," so that we cannot restructure part of the curve without specifying an entirely new set of control points. For this reason, other representations for a cubic-spline interpolation have been developed.

Hermite Interpolation

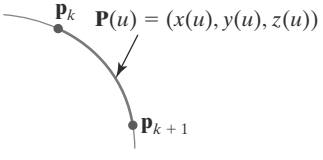

FIGURE 10
Parametric point function $\mathbf{P}(u)$ for a Hermite curve section between control points $\mathbf{p}_k$ and $\mathbf{p}_{k+1}$.

A **Hermite spline** (named after the French mathematician Charles Hermite) is an interpolating piecewise cubic polynomial with a specified tangent at each control point. Unlike the natural cubic splines, Hermite splines can be adjusted locally because each curve section depends only on its endpoint constraints.

If $\mathbf{P}(u)$ represents a parametric cubic point function for the curve section between control points $\mathbf{p}_k$ and $\mathbf{p}_{k+1}$, as shown in Figure 10, then the boundary conditions that define this Hermite curve section are

$$\begin{aligned} \mathbf{P}(0) &= \mathbf{p}_k \\ \mathbf{P}(1) &= \mathbf{p}_{k+1} \\ \mathbf{P}'(0) &= \mathbf{D}\mathbf{p}_k \\ \mathbf{P}'(1) &= \mathbf{D}\mathbf{p}_{k+1} \end{aligned} \tag{9}$$

with $\mathbf{D}\mathbf{p}_k$ and $\mathbf{D}\mathbf{p}_{k+1}$ specifying the values for the parametric derivatives (slope of the curve) at control points $\mathbf{p}_k$ and $\mathbf{p}_{k+1}$, respectively.

We can write the vector equivalent of Equations 8 for this Hermite curve section as

$$\mathbf{P}(u) = \mathbf{a}\,u^3 + \mathbf{b}\,u^2 + \mathbf{c}\,u + \mathbf{d}, \qquad 0 \le u \le 1 \tag{10}$$

where the x component of $\mathbf{P}(\mathrm{u})$ is $x(u) = a_x\,u^3 + b_x\,u^2 + c_x\,u + d_x$, and similarly for the y and z components. The matrix equivalent of Equation 10 is

$$\mathbf{P}(u) = [u^3 \quad u^2 \quad u \quad 1] \cdot \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \end{bmatrix} \tag{11}$$

and the derivative of the point function can be expressed as

$$\mathbf{P}'(u) = [3u^2 \quad 2u \quad 1 \quad 0] \cdot \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \end{bmatrix} \tag{12}$$

Substituting endpoint values 0 and 1 for parameter u into the preceding two equations, we can express the Hermite boundary conditions 9 in the matrix form

$$\begin{bmatrix} \mathbf{p}_k \\ \mathbf{p}_{k+1} \\ \mathbf{D}\mathbf{p}_k \\ \mathbf{D}\mathbf{p}_{k+1} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \end{bmatrix} \tag{13}$$

Solving this equation for the polynomial coefficients, we get

$$\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}^{-1} \cdot \begin{bmatrix} \mathbf{p}_k \\ \mathbf{p}_{k+1} \\ \mathbf{D}\mathbf{p}_k \\ \mathbf{D}\mathbf{p}_{k+1} \end{bmatrix}$$

$$= \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{p}_k \\ \mathbf{p}_{k+1} \\ \mathbf{D}\mathbf{p}_k \\ \mathbf{D}\mathbf{p}_{k+1} \end{bmatrix}$$

$$= \mathbf{M}_H \cdot \begin{bmatrix} \mathbf{p}_k \\ \mathbf{p}_{k+1} \\ \mathbf{D}\mathbf{p}_k \\ \mathbf{D}\mathbf{p}_{k+1} \end{bmatrix} \tag{14}$$

where $\mathbf{M}_H$, the Hermite matrix, is the inverse of the boundary constraint matrix. Equation 11 can thus be written in terms of the boundary conditions as

$$\mathbf{P}(u) = [u^3 \quad u^2 \quad u \quad 1] \cdot \mathbf{M}_H \cdot \begin{bmatrix} \mathbf{p}_k \\ \mathbf{p}_{k+1} \\ \mathbf{D}\mathbf{p}_k \\ \mathbf{D}\mathbf{p}_{k+1} \end{bmatrix} \tag{15}$$

Finally, we can determine expressions for the polynomial Hermite blending functions, $H_k(u)$ for $k = 0, 1, 2, 3$, by carrying out the matrix multiplications in Equation 15 and collecting coefficients for the boundary constraints to obtain the polynomial form

$$\begin{aligned} \mathbf{P}(u) &= \mathbf{p}_k(2u^3 - 3u^2 + 1) + \mathbf{p}_{k+1}(-2u^3 + 3u^2) + \mathbf{D}\mathbf{p}_k(u^3 - 2u^2 + u) \\ &\qquad + \mathbf{D}\mathbf{p}_{k+1}(u^3 - u^2) \\ &= \mathbf{p}_k H_0(u) + \mathbf{p}_{k+1} H_1 + \mathbf{D}\mathbf{p}_k H_2 + \mathbf{D}\mathbf{p}_{k+1} H_3 \end{aligned} \tag{16}$$

Figure 11 shows the shape of the four Hermite blending functions.

Hermite polynomials can be useful for some digitizing applications, where it may not be too difficult to specify or approximate the curve slopes. But for most problems in computer graphics, it is more useful to generate spline curves without requiring input values for curve slopes or other geometric information, in addition to control-point coordinates. Cardinal splines and Kochanek-Bartels splines, discussed in the following two sections, are variations on the Hermite splines that do not require input values for the curve derivatives at the control points. Procedures for these splines compute parametric derivatives from the coordinate positions of the control points.

Cardinal Splines

As with Hermite splines, the **cardinal splines** are interpolating piecewise cubic polynomials with specified endpoint tangents at the boundary of each curve section. The difference is that we do not input the values for the endpoint tangents. For a cardinal spline, the slope at a control point is calculated from the coordinates of the two adjacent control points.

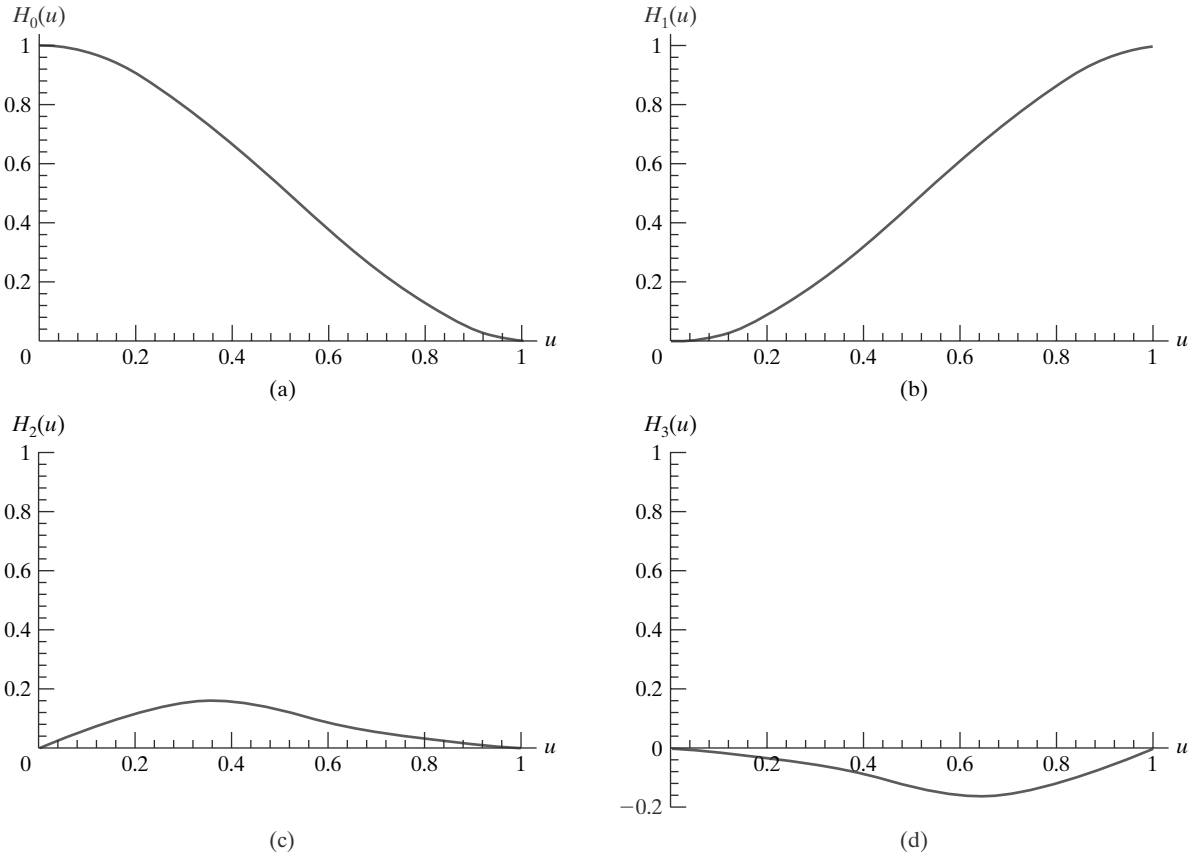

FIGURE 11
The Hermite blending functions.

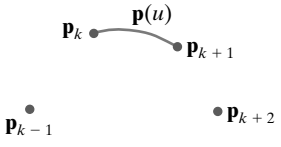

FIGURE 12
Parametric point function $\mathbf{P}(u)$ for a cardinal-spline section between control points $\mathbf{p}_k$ and $\mathbf{p}_{k+1}$.

A cardinal spline section is completely specified with four consecutive control-point positions. The middle two control points are the section endpoints, and the other two points are used in the calculation of the endpoint slopes. If we take $\mathbf{P}(u)$ as the representation for the parametric cubic point function for the curve section between control points $\mathbf{p}_k$ and $\mathbf{p}_{k+1}$, as in Figure 12, then the four control points from $\mathbf{p}_{k-1}$ to $\mathbf{p}_{k+1}$ are used to set the boundary conditions for the cardinal-spline section as

$$\begin{aligned} \mathbf{P}(0) &= \mathbf{p}_k \\ \mathbf{P}(1) &= \mathbf{p}_{k+1} \\ \mathbf{P}'(0) &= \frac{1}{2}(1-t)(\mathbf{p}_{k+1} - \mathbf{p}_{k-1}) \\ \mathbf{P}'(1) &= \frac{1}{2}(1-t)(\mathbf{p}_{k+2} - \mathbf{p}_k) \end{aligned} \tag{17}$$

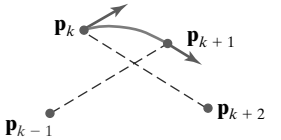

FIGURE 13
Tangent vectors at the endpoints of a cardinal-spline section are parallel to the chords formed with neighboring control points (dashed lines).

Thus, the slopes at control points $\mathbf{p}_k$ and $\mathbf{p}_{k+1}$ are taken to be proportional, respectively, to the chords $\overline{\mathbf{p}_{k-1}\mathbf{p}_{k+1}}$ and $\overline{\mathbf{p}_k\mathbf{p}_{k+2}}$ (Figure 13). Parameter t is called the **tension** parameter because it controls how loosely or tightly the cardinal spline fits the input control points. Figure 14 illustrates the shape of a cardinal curve for very small and very large values of tension t. When $t = 0$, this class of curves is referred to as **Catmull-Rom splines,** or **Overhauser splines.**

Using methods similar to those for Hermite splines, we can convert the boundary conditions 17 into the matrix form

$$\mathbf{P}(u) = [u^3 \quad u^2 \quad u \quad 1] \cdot \mathbf{M}_C \cdot \begin{bmatrix} \mathbf{p}_{k-1} \\ \mathbf{p}_k \\ \mathbf{p}_{k+1} \\ \mathbf{p}_{k+2} \end{bmatrix} \tag{18}$$

where the cardinal matrix is

$$\mathbf{M}_C = \begin{bmatrix} -s & 2-s & s-2 & s \\ 2s & s-3 & 3-2s & -s \\ -s & 0 & s & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{19}$$

with $s = (1 - t)/2$.

Expanding Equation 18 into polynomial form, we have

$$\begin{aligned} \mathbf{P}(u) &= \mathbf{p}_{k-1}(-s\,u^3 + 2s\,u^2 - s\,u) + \mathbf{p}_k[(2-s)u^3 + (s-3)u^2 + 1] \\ &\quad + \mathbf{p}_{k+1}[(s-2)u^3 + (3-2s)u^2 + s\,u] + \mathbf{p}_{k+2}(s\,u^3 - s\,u^2) \\ &= \mathbf{p}_{k-1}\,\mathrm{CAR}_0(u) + \mathbf{p}_k\,\mathrm{CAR}_1(u) + \mathbf{p}_{k+1}\,\mathrm{CAR}_2(u) + \mathbf{p}_{k+2}\,\mathrm{CAR}_3(u) \end{aligned} \tag{20}$$

where the polynomials $\mathrm{CAR}_k(u)$ for $k = 0, 1, 2, 3$ are the cardinal-spline blending (basis) functions. Figure 15 gives a plot of the basis functions for cardinal splines with $t = 0$.

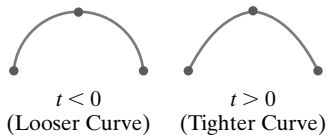

FIGURE 14
Effect of the tension parameter on the shape of a cardinal-spline section.

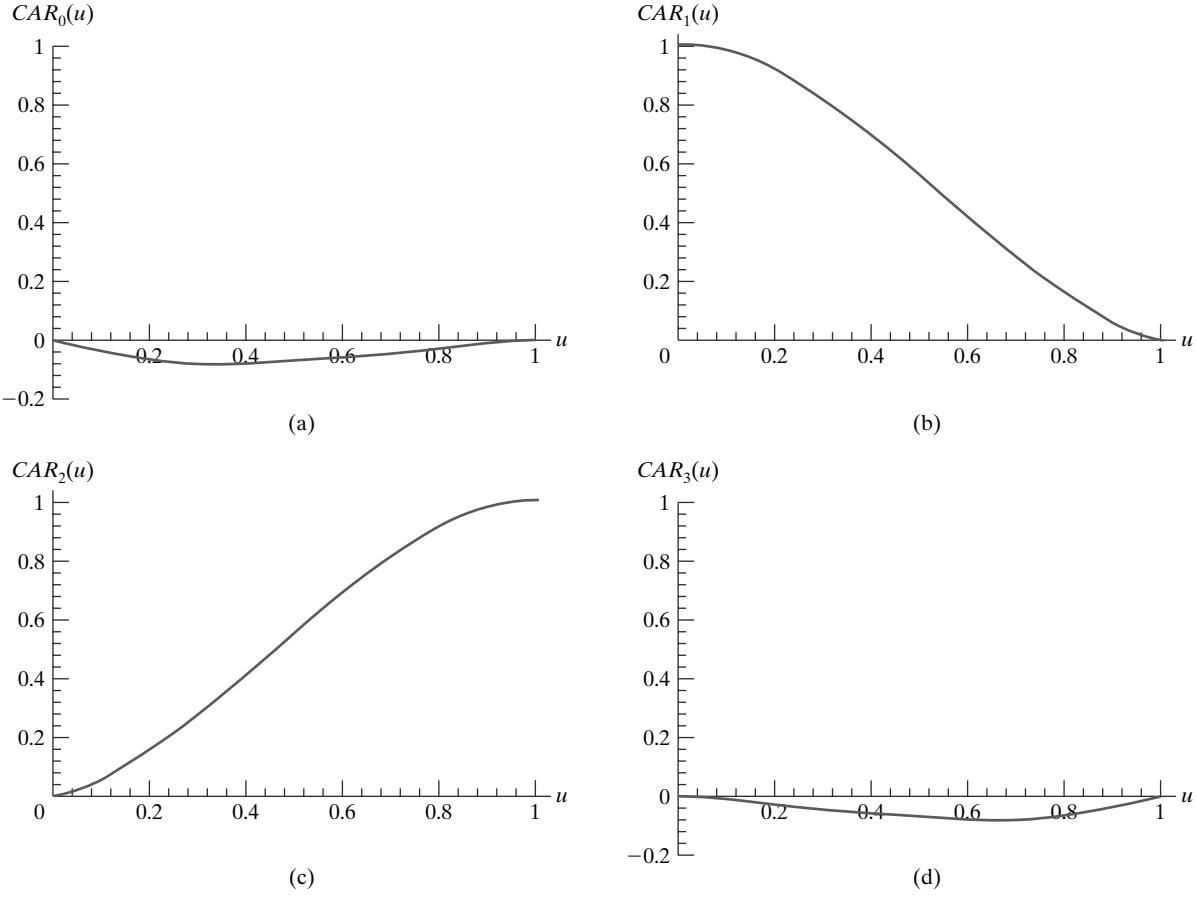

FIGURE 15
The cardinal-spline blending functions for $t = 0$ ($s = 0.5$).

Examples of curves produced with the cardinal-spline blending functions are given in Figures 16, 17, and 18. In Figure 16, four cardinal-spline sections are plotted to form a closed curve. The first curve section is generated using the control-point set $\{\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3\}$, the second curve is produced with the control-point set $\{\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_0\}$, the third curve section has control points $\{\mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_0, \mathbf{p}_1\}$, and the final curve section has control points $\{\mathbf{p}_3, \mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2\}$. In Figure 17, a closed curve is obtained with a single cardinal-spline section by setting the position of the third control point to the coordinate position of the

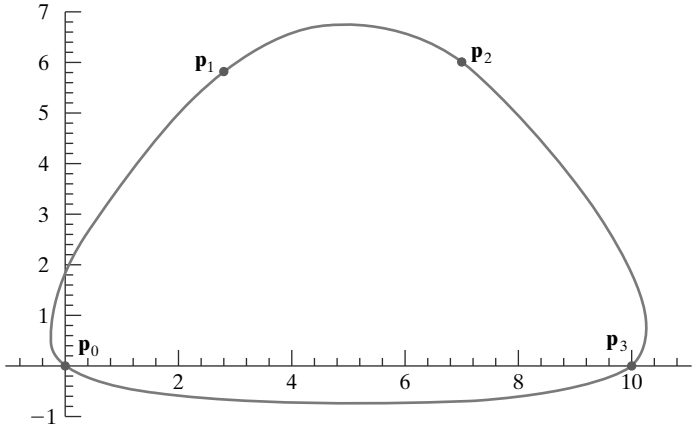

FIGURE 16
A closed curve with four cardinal-spline sections, obtained with a cyclic permutation of the control points and with tension parameter $t = 0$.

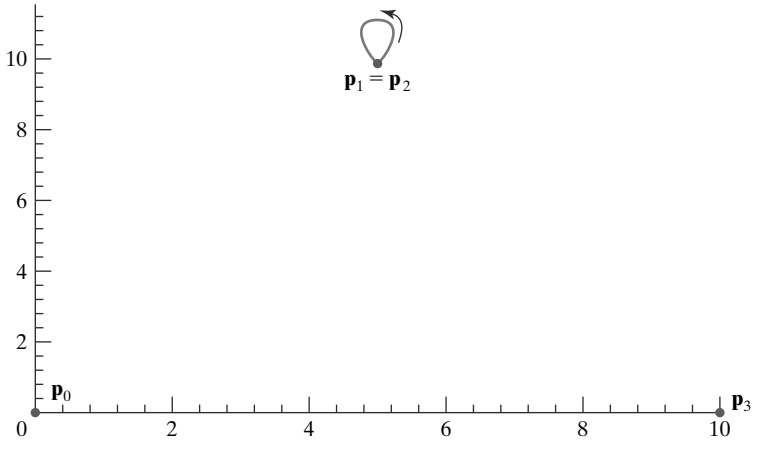

FIGURE 17
A cardinal- spline loop produced with curve endpoints at the same coordinate position. The tension parameter is set to the value 0.

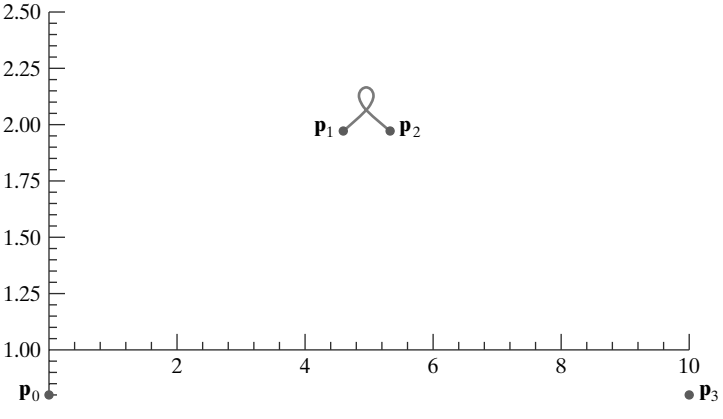

FIGURE 18
A self-intersecting cardinal-spline curve section produced with closely spaced curve endpoint positions. The tension parameter is set to the value 0.

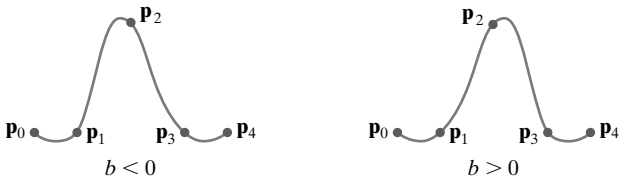

FIGURE 19
Effect of the bias parameter on the shape of a Kochanek-Bartels spline section.

second control point. In Figure 18, a self-intersecting cardinal-spline section is produced by setting the position of the third control point very near the coordinate position of the second control point. The resulting self-intersection is due to the constraints on the curve slope at the endpoints $\mathbf{p}_1$ and $\mathbf{p}_2$.

Kochanek-Bartels Splines

These interpolating cubic polynomials are extensions of the cardinal splines. Two additional parameters are introduced into the constraint equations defining **Kochanek-Bartels splines** to provide further flexibility in adjusting the shapes of curve sections.

Given four consecutive control points, labeled $\mathbf{p}_{k-1}$, $\mathbf{p}_k$, $\mathbf{p}_{k+1}$, and $\mathbf{p}_{k+2}$, we define the boundary conditions for a Kochanek-Bartels curve section between $\mathbf{p}_k$ and $\mathbf{p}_{k+1}$ as

$$\begin{aligned}
\mathbf{P}(0) &= \mathbf{p}_k \\
\mathbf{P}(1) &= \mathbf{p}_{k+1} \\
\mathbf{P}'(0)_{\text{in}} &= \frac{1}{2}(1-t)[(1+b)(1-c)(\mathbf{p}_k - \mathbf{p}_{k-1}) \\
&\qquad + (1-b)(1+c)(\mathbf{p}_{k+1} - \mathbf{p}_k)] \\
\mathbf{P}'(1)_{\text{out}} &= \frac{1}{2}(1-t)[(1+b)(1+c)(\mathbf{p}_{k+1} - \mathbf{p}_k) \\
&\qquad + (1-b)(1-c)(\mathbf{p}_{k+2} - \mathbf{p}_{k+1})]
\end{aligned} \tag{21}$$

where t is the **tension** parameter, b is the **bias** parameter, and c is the **continuity** parameter. In the Kochanek-Bartels formulation, parametric derivatives might not be continuous across section boundaries.

Tension parameter t has the same interpretation as in the cardinal spline formulation; that is, it controls the looseness or tightness of the curve sections. Bias, b, is used to adjust the curvature at each end of a section so that curve sections can be skewed toward one end or the other (Figure 19). Parameter c controls the continuity of the tangent vector across the boundaries of sections. If c is assigned a nonzero value, there is a discontinuity in the slope of the curve across section boundaries.

Kochanek-Bartels splines were designed to model animation paths. In particular, abrupt changes in the motion of an object can be simulated with nonzero values for parameter c. These motion changes are used in cartoon animations, for example, when a cartoon character stops quickly, changes direction, or collides with some other object.

8 Bézier Spline Curves

This spline approximation method was developed by the French engineer Pierre Bézier for use in the design of Renault automobile bodies. **Bézier splines** have a number of properties that make them highly useful and convenient for curve and