

Chapter 1

Introduction to Freeform Surface Modelling



Abstract The ability to use a variety of CAD tools is one of the most basic competences expected of a modern engineer. It represents a language and a tool, and is used to communicate with other engineers anywhere in the world. While the basic CAD operations are fairly easy to learn, understanding the functioning of CAD tools requires more detailed knowledge of mathematical and computer laws behind the individual CAD operations. In this chapter the reader will learn about the mathematical laws of CAD curves and surfaces that are the key to understanding freeform modelling. It begins with a computer description of the curves in space, which is followed by various procedures for creating freeform surfaces based on these curves. The chapter ends with an explanation of several definitions and descriptions of the transitions, i.e., the continuity between the curves and the surfaces. Understanding the latter is becoming increasingly important for the automotive, aerospace and consumer-goods industries, as properly executed transitions between surfaces have a significant effect on the strength, aerodynamic, ergonomic and aesthetic properties of a product. At the same time, these topics are often not covered by many regular engineering curricula due to a lack of time or because their importance is overlooked. As a result, engineers only deal with them when they face a concrete challenge for the first time—which is often on their first job.

1.1 The Role of Freeform Surfaces in Modern Modelling and Design

In order to represent objects or products, i.e., technical systems in general terms, a variety of tools has been used from early times. Together with analogue records in 3D space, solutions were being explored for the best possible digital records. The advent of the computer processor and its adapted mathematical relations brought about an opportunity for a high-quality digital reproduction of 3D space. Throughout the development of 3D modelling, different procedures and approaches evolved that seemed fairly important at the time. However, only with the passage of time did some of the methods get thoroughly tested and so were able to establish themselves as necessary or important. This is why it is vital to understand that some

of the development concepts and models were important and necessary in order to be able to create the clearest possible direction and the more general and intuitive use of modellers.

Historically, development was gradual and went from a simple representation of wire models, surface representation of 3D models and eventually to a solid model as the most faithful way of representing realistic models in space (Fig. 1.1).

Such developments and opportunities gradually brought solutions to the user that were comparable across different modellers. This makes sense, as modeller developers eventually meet the users, who generally have a similar approach to solving tasks and requirements. For this reason, the large number of providers was gradually reduced to a small number of suppliers of general 3D modelling equipment. At the same time there are also developers for specific needs and specific designs, and modellers include separated modules, adapted to different tasks.

The user interface is an important quality of a modeller, and it either is or should be standardised according to the input commands, as software providers will have to reach an agreement as to how the user can change from one modeller to another.

Geometric transformations are the next significant module. The level of their accuracy determines the actual use of a modeller. Of course it does not include simple transformations, but primarily the modelling of freeform surfaces that merge into a logically re-shaped new form that can be used in most existing technologies. This means that geometric transformations should be adapted to specific technologies in use, such as pressure casting, thermoplastic injection moulding, die casting, glassblowing, etc.

1.2 Topological Building Blocks of 3D Models

The geometry of bodies in sophisticated 3D modelling systems is characterised by descriptions of the individual basic geometric elements: points, straight lines, parts of a circle, general 3D curves and surfaces of any complexity. There are also alternative descriptions of bodies, based on the use of other elements: describing the surface of a 3D object exclusively by triangles, describing a 3D volume by elementary volume elements (voxels), grouping 3D primitives according to their function, etc. They will not be dealt with in this chapter.

- **Points** represent the coordinates of knots.
- **Edges** represent lines where edges are located. They are bounded by the corresponding knots.
- **Loops** represent the closed connection of edges, defined by points.
- **Surfaces** represent a similar data structure to the loops. They are added to the surface normal data only in order to define them in space.
- **Volume** represents a space, uniquely and in its entirety bounded by surfaces, which is why it is referred to as a closed volume. Normals to the surfaces that have been completely finished are oriented outwards.


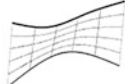
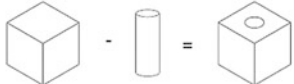
	 Wireframe model	 Surface model	 Solid model
1960	Development of wire and surface models		
1970	First use of wire and surface models	Development of solid models modelling	
1980	Specific use of wire and surface models	First use of solid models	
1990	General use of wire and surface models	Specific use of solid models	
2000	General use of wire, surface and solid models		
2010	Introducing special techniques of wire, surface and solid models for product modelling		

Fig. 1.1 Historical development of the theory of modelling and the use of modellers [1]

The basic building blocks, such as a point, a straight line, a circle and a circular arc, have been dealt with in the literature (e.g., [1, 2]), while this book will focus primarily on curves and surfaces.

First, however, it must be pointed out that 3D modelling should be understood as modelling in space and defined by coordinate systems. In everyday speech, an incorrect term *CAD: Computer Aided Design* is often used. Such a term is completely wrong, as there is no programming language for the purpose of designing, it is just for modelling. So, the correct term is 3D modeller. As of late, modelling software providers also tend to say “3D modellers” when referring to their products. In other words, design is much more than giving a product a shape that only serves its own purpose. It requires understanding and a consideration of all the physical, mathematical and other principles that define the shape and the standards, set by a cultural or technical environment, and the use of manufacturing, application and decomposition technologies. Design also requires that all the design solutions are clearly supported by the corresponding calculations.

1.2.1 Curve

In general terms a curve can be defined as the shortest connection between two end points. Curves are generally expected to be smooth, i.e., with no sharp transitions, referred to as tangency (a first-order continuity) in mathematics. When a continuous change of the radius of a curve curvature at each point is also expected, it requires a second-order continuity. In mathematical terms it means that a curve should be

differentiable twice over the required interval. High-school plane geometry provides plenty of examples for such curves, e.g., conic sections (ellipse, hyperbola, parabola; Fig. 1.2); polynomial functions (Fig. 1.3) or rational functions. A curve can also be presented in the form of a table of points, with the values of the coordinates x , y , z , whereas in between the points it is possible to use linear, parabolic or any other interpolation with a higher-order polynomial.

Curves can also be given parametrically. In this form, each point coordinate on a curve also depends separately on one coordinate more than the degree of the used curve. A new coordinate can then be introduced by adding another parameter, denoted as t , for example. Its value must be chosen over an interval between 0 and 1. For a more detailed numerical analysis, both extreme values are of special interest. They are the beginning $t = 0$ and the end $t = 1$. The parameter $t = 0$ is not necessarily used at the beginning of the curve. Figure 1.4 shows an example of a parametric curve, where the x and y coordinates are given separately with their polynomial of the type that depends only on the parameter t . The equation shows that such a curve will turn randomly across the space, i.e., it has no boundaries, similar to those in explicitly given functions of the $y = f(x)$ type (the multivalued problem). For this reason it is vital to be careful with the selection of functions. They need to be single-valued over the whole interval under consideration.

$$x(t) = a_n \cdot t^n + a_{n-1} \cdot t^{n-1} + \dots + a_1 \cdot t + a_0 \quad (1.1)$$

Over the past 20 years, B-splines and their upgraded NURBS (Non-Uniform Rational B-spline) have emerged as the most common way of describing curves. These are curves consisting of individual cubic parametric space polynomials, and

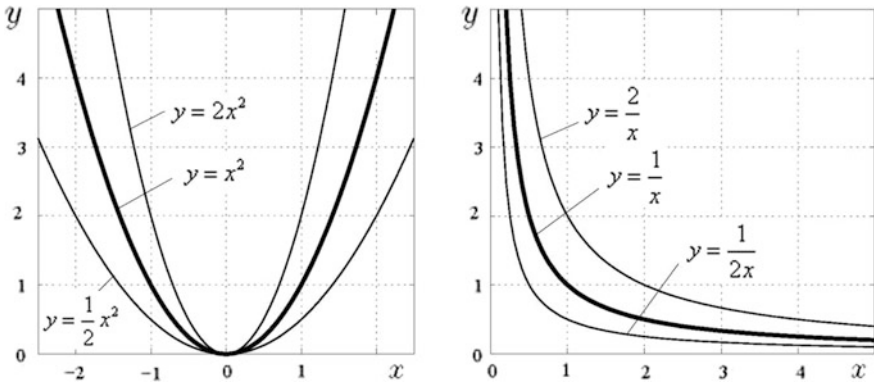


Fig. 1.2 Parabolas and hyperbolas as an example of curves [1]

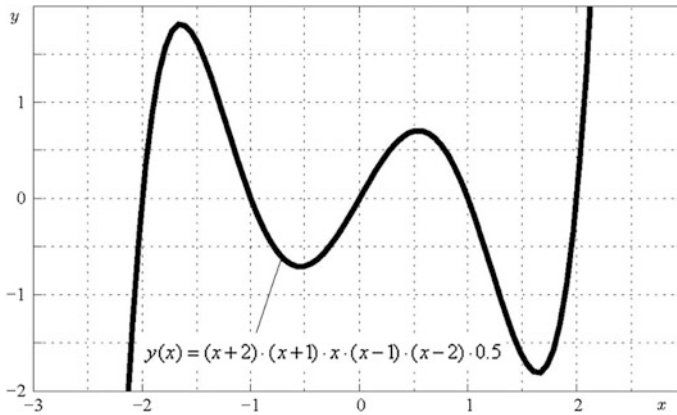
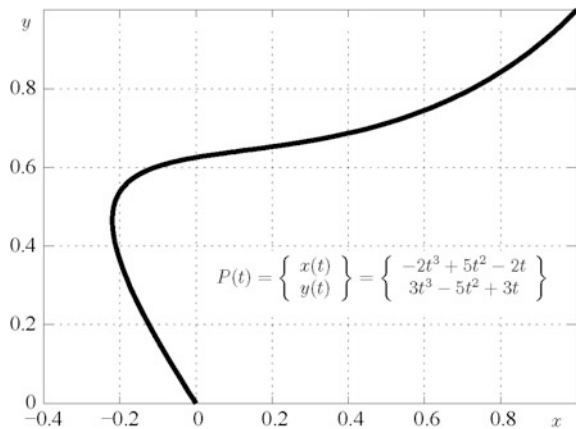


Fig. 1.3 A polynomial as an example of using curves [1]

Fig. 1.4 A parametric curve, determined by two third-degree polynomials [1]



they are also determined by control points (Fig. 1.5). Their key advantages over other types of curves are as follows:

- they are defined by means of a relatively small number of control points,
- they are very flexible and have an excellent ability to approximate the frequently used curves, such as circles, conic sections and polynomials,
- when interpolated through given points, they intuitively “bend” and do not “run far away” from the polyline, connecting the interpolation points,
- they are independent of the position in the coordinate system, which makes them easy to rotate into any position in space,
- if necessary, they exhibit locality, which means that modifying the position of one interpolation point affects only a limited curve area.

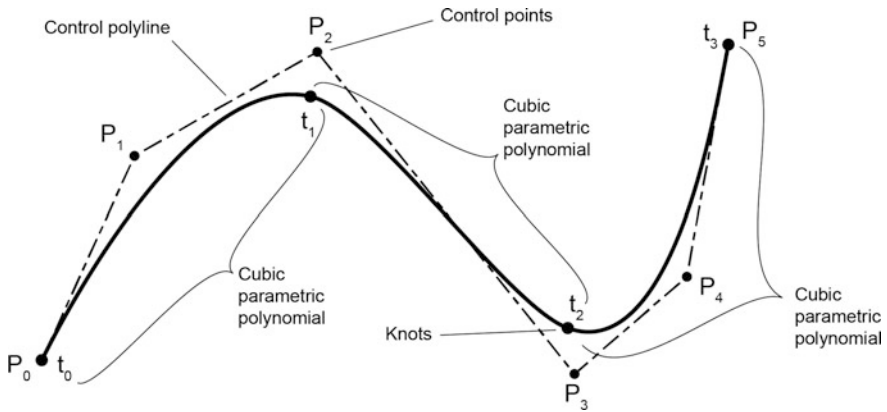


Fig. 1.5 B-spline, determined by control points [1]

Parametric curves, expressed in a mathematical form as B-splines or NURBS, are further discussed in Sect. 1.3.

1.2.2 Surface

Surfaces are shapes that are usually defined in 3D space. Because describing a surface can be a rather complex job, there are several approaches available to model objects using surfaces. Examples of highly complex surfaces can include modelling the organic parts of living organisms, such as the human skeleton or brain. Our focus will be on describing surfaces that can be created with conventional procedures and involving sensible effort and acceptable accuracy. The following are the most frequent types of describing surfaces:

- surfaces as parts of curved surfaces of regular geometric bodies,
- parametric surfaces,
- interpolate surfaces,
- polygon meshes (mostly triangles and quadrilaterals).

Modelling bodies in space often involves a surface that is a planar polygon, such as a rectangle. We also often deal with surfaces that are in part or in entirety parts of curved surfaces of regular geometric bodies, e.g., a cylinder, a triangle, a sphere or a torus. Such surfaces are relatively easy to describe with analytic equations, particularly in a parametric form (Figs. 1.6 and 1.7).

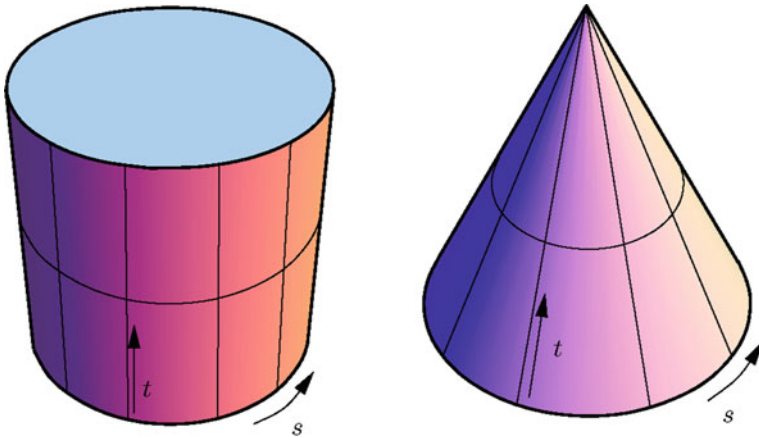


Fig. 1.6 Cylinder and cone surfaces [1]

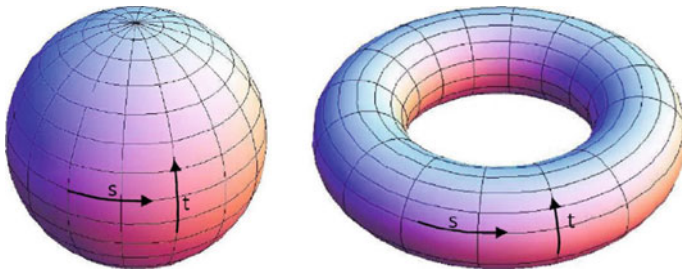


Fig. 1.7 Sphere and torus surfaces [1]

The equation of a cylinder can be parametrically written as:

$$P(s, t) = \begin{Bmatrix} x(s, t) \\ y(s, t) \\ z(s, t) \end{Bmatrix} = \begin{Bmatrix} R \cdot \cos(s) \\ R \cdot \sin(s) \\ t \end{Bmatrix} \quad (1.2)$$

where R is the radius of the cylinder, s is a parameter, interpreted geometrically as an angle over an interval $[0, \dots, 360]$, and t is a parameter over an interval $[0, \dots, H]$, where H is the height of the cylinder.

A possible equation for the lateral surface of a cone can be parametrically written as:

$$P(s, t) = \begin{Bmatrix} x(s, t) \\ y(s, t) \\ z(s, t) \end{Bmatrix} = \begin{Bmatrix} R \cdot \left(1 - \frac{t}{H}\right) \cdot \cos(s) \\ R \cdot \left(1 - \frac{t}{H}\right) \cdot \sin(s) \\ t \end{Bmatrix} \quad (1.3)$$

where R is the radius of a cone, s is a parameter, interpreted geometrically as an angle over an interval $[0, \dots, 360]$, and t is a parameter over an interval $[0, \dots, H]$, where H is the height of the cone.

The equation of a sphere can be parametrically written as:

$$P(s, t) = \begin{Bmatrix} x(s, t) \\ y(s, t) \\ z(s, t) \end{Bmatrix} = \begin{Bmatrix} R \cdot \cos(s) \cdot \cos(t) \\ R \cdot \sin(s) \cdot \cos(t) \\ t \cdot \sin(s) \end{Bmatrix} \quad (1.4)$$

where R is the radius of a sphere, s is a parameter, interpreted geometrically as an angle over an interval $[0, \dots, 360]$, and t is a parameter, interpreted geometrically as an angle over an interval $[0, \dots, 360]$, and is orthogonal to the parameter s .

The equation of a torus can be parametrically written as:

$$P(s, t) = \begin{Bmatrix} x(s, t) \\ y(s, t) \\ z(s, t) \end{Bmatrix} = \begin{Bmatrix} \cos(s) \cdot (R + r \sin(t)) \\ \sin(s) \cdot (R + r \sin(t)) \\ r \cdot \sin(s) \end{Bmatrix} \quad (1.5)$$

where R is the major radius, r is the minor radius of the torus, and s is the parameter, interpreted geometrically as an angle over an interval $[0, \dots, 360]$, and t is a parameter, interpreted geometrically as an angle over an interval $[0, \dots, 360]$, and is orthogonal to the parameter s .

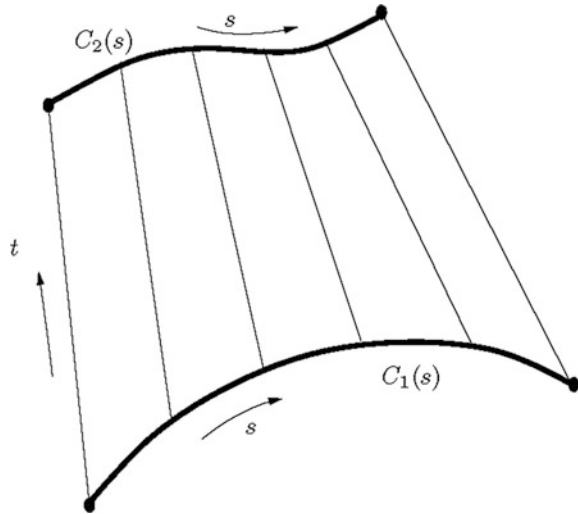
It should be noted that the presented shapes (a cylinder, a cone, a sphere and a torus) can be mathematically defined separately for each coordinate by defining the pitch for the coordinate s , e.g., 0.002 mm, which allows the tool travel via the controller using the same pitch. This means that inside an NC machine's controller, a mathematical operation, following the equation of a geometric body, can take place, and that the calculated coordinates are directly written into the command instructions in charge of moving the tool across, for example, a Cartesian coordinate system.

Interpolate surfaces can be designed by supposing a limited number of known points (edges), through which a surface runs. On the remaining part of the surface—that could be referred to as freeform surfaces—let us suppose a smooth, calm and intuitively foreseeable course (Fig. 1.8). Our description will focus on surface patches with four edges. Each of these edges can represent any curve. Such cases are typical of managing freeform surfaces when working on products. Some typical examples of interpolate surfaces include:

- linear interpolation between two opposite boundary curves,
- bilinear interpolation of the interior, when all four boundary curves are known,
- bi-polynomial interpolation, when an $n \times m$ mesh of orthogonal curves is available.

A more detailed description of the individual approaches can be found in Sect. 1.4.

Fig. 1.8 Interpolate surface defined by two boundary curves [1]



Freeform parametric surfaces are a generalised form of parametric curves. In Sect. 1.2.1 an individual coordinate of a 2D or 3D polynomial curve was defined as $x(t) = a_n \cdot t^n + a_{n-1} \cdot t^{n-1} + \dots + a_1 \cdot t^1 + a_0$, while parametric surfaces represent the tensor product of the individual coordinates in the direction of the parameter t and in the direction of the parameter s . In engineering graphics, third-degree parametric surfaces (i.e., cubic parametric surfaces) in particular have become established. Replacing a set of the so-called basis parametric functions $t^3, t^2, t, 1$ by basis functions will result in the so-called Bézier cubic parametric surface patch. Like with expanding the usability area of the parametric curves, Bézier's formulation of the surface patch can also be extended to surface splines and NURBS surfaces. This will be discussed in detail in Sect. 1.3. Using a relatively small number of available control points, this type of surface (Fig. 1.9) can result in the efficient shaping of randomly curved surfaces.

Polygon meshes are used to describe surfaces when they are too complex for an analytical description. A typical example from everyday use is capturing data on surfaces by means of the optical or mechanical contact scanning of 3D objects. **Optical scanners** usually capture from several thousand to several million dots in space. Using appropriate software, these dots are then filtered, reduced if necessary, and connected into a polygon mesh, usually consisting of triangles, as shown in Fig. 1.10. In special cases it is also possible to generate a quadrilateral mesh. Using a **mechanical contact scanner** yields significantly fewer contact dots; however, it improves the possibility of modifying the surfaces between the dots. Using one or the other measuring method is a question for the engineer and depends on the method of work and the approach to product development.

Another example of using polygon meshes is running computer simulations in the area of strength, heat transfer, fluid dynamics or injection moulding. In this simulation it is a reverse process. A computer modeller starts with an analytically

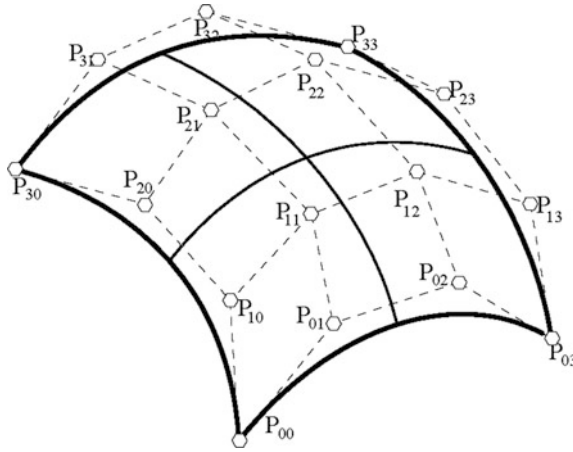


Fig. 1.9 Freeform parametric surface described with 16 control points [1]



Fig. 1.10 Point cloud of a scanned surface (left) and a triangle mesh (right)

described geometry. In the second step, a special pre-simulation part of the software divides the described surface geometry into a large number of triangles or quadrilaterals. The resulting elementary parts are then used to define the matrix-simulation equation, which is used to calculate the distributions, such as stress, deformation, temperature, pressure, etc.

1.3 B-spline Curves and NURBS Curves and Surfaces

The basic elements used for creating freeform surfaces are linear elements, generally referred to as curves. They represent an edge or a connection between two points. A surface can be formed by linear elements related to one another in some relationship. In the Cartesian coordinate system we normally use orthogonality at the intersections between linear elements (curves). The interpenetration of orthogonally connected linear elements (curves) is used to present a surface.

Let us first take a look at how a linear element (i.e., curve) is defined. From the curves used in space modelling the following properties are expected:

- zeroth-, first- and second-order continuity,
- good deformability; local and global control is possible,
- maintaining the shape when subject to geometric transformations (especially complex transformations, such as rotation in space),
- tendency to reduced variation,
- the simplest possible mathematical description,
- universality of the mathematical description for all types of standard shapes (straight lines, parts of circles, parabolas, ellipses, polynomials, etc., as well as flat faces, and cylindrical, spherical and interpolate surfaces).

A *Bézier* curve satisfies all these properties. Pierre Bézier was a French engineer (Fig. 1.11) who worked for the Renault car manufacturer between 1933 and 1975, where he dealt with the problem of transferring the prototype geometry of a car body (which was made out of wood or another material) to the drawing board, or accurately defining it in a mathematical form. In 1966 he published an article in the *Automatisme* journal, where he defined a novel method for describing parametric cubic curves, which earned him lasting fame, as his description became the basis for the mathematical description of curves and surfaces for all subsequent computer modellers. Paul de Casteljau, an engineer at Citroën, had detailed such a description before Bézier, but he only published it in the company's internal newsletter. Because the information was not accessible, Casteljau's notation passed into oblivion. Today, Bézier's description is established and B-curves are named after him.

Interpolation has been used for a long time to create new shapes. In our case, the problem is about guiding a smooth and 'appealing' curve through a set of given points. In the past, craftsmen and designers made use of thin wooden or metal splines. These metal splines were firmly tied together, which made them deformable, like a pile of boards, resting on two supports. Engineers used them for the purpose of flexible curves as late as 1990. They fixed the tied metal splines at support points (Fig. 1.12), and, if necessary, further weighted them with lead (Fig. 1.13). We can attempt to represent the shape $y = y(x)$ of such a wooden or metal spline using a differential equation:



Fig. 1.11 Pierre Bézier, French engineer and mathematician

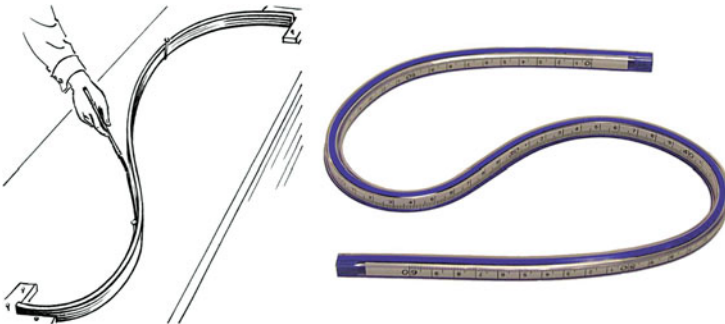


Fig. 1.12 An example of using a metal spline (left) and modern commercial bend ruler (right)

$$M(x) = E \cdot I \cdot \frac{y''}{(1 + y'^2)^{\frac{3}{2}}} \quad (1.6)$$

The product $E \cdot I$ represents the stiffness coefficient. It is constant if the material is homogenous and the spline has a constant cross-section. In the absence of any other outside forces between the support points and weights, the torque of the internal forces $M(x)$ linearly depends on the distance x .

When the deformations are small $y' \ll 1$, the term y' is negligible, and Eq. 1.6 can be approximated by $E \cdot I \cdot y'' = M(x)$ or $y^4 = 0$. Due to the linearity of the torque $M(x)$, the shape $y = y(x)$ is piecewise cubic and continuous. The first and second derivatives are continuous, too. The shapes of the curves that had for a long time been used for designing and shaping new products therefore originate in mechanics and are based on the deflection line of the loaded (and therefore deformed) spline or profile.

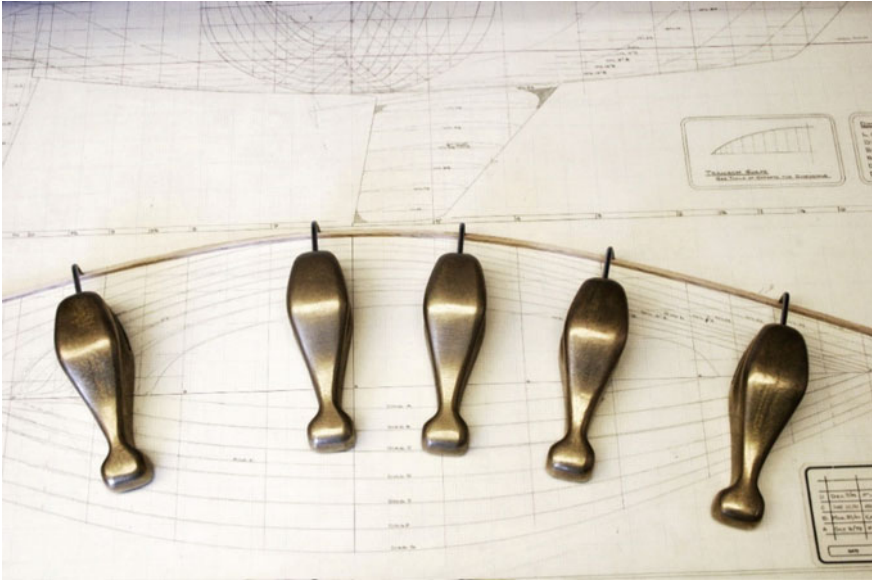


Fig. 1.13 Wooden spline, weighted with lead, set to outline a ship's hull

1.3.1 Cubic Parametric Curves

Three-dimensional space modelling requires a description of the space curves that need to be numerically manageable and not subject to the problems of numerical solutions. We also do not want oscillating curves, which happens with the use of higher-order polynomial curves. The simplest, non-problematically continuous curves are polynomials. If at least second-order continuity is required, parametric cubic polynomials will be sufficient.

$$\begin{aligned} x(t) &= a_1t^3 + b_1t^2 + c_1t + d_1 \\ y(t) &= a_2t^3 + b_2t^2 + c_2t + d_2 \\ z(t) &= a_3t^3 + b_3t^2 + c_3t + d_3 \end{aligned} \quad (1.7)$$

The parameter t can generally have a value over any interval. In our case, when the parameter t is used as part of a uniform space, the value over an interval $[0, \dots, 1]$ is used. The curve can be written in matrix form as:

$$P(t) = \begin{bmatrix} x(t) \\ y(t) \\ z(t) \end{bmatrix} = \begin{bmatrix} a_1 & b_1 & c_1 & d_1 \\ a_2 & b_2 & c_2 & d_2 \\ a_3 & b_3 & c_3 & d_3 \end{bmatrix} \cdot \begin{Bmatrix} t^3 \\ t^2 \\ t \\ 1 \end{Bmatrix} \quad (1.8)$$

If the coefficients a_1, a_2, a_3 are combined into the vector \mathbf{A} and the coefficients b_1, b_2, b_3 into \mathbf{B} and so on, it can be written as:

$$P(t) = [\{\mathbf{A}\} \quad \{\mathbf{B}\} \quad \{\mathbf{C}\} \quad \{\mathbf{D}\}] \cdot \begin{Bmatrix} t^3 \\ t^2 \\ t \\ 1 \end{Bmatrix} \quad (1.9)$$

We can opt for different shapes by choosing parametric cubic polynomials. Different authors were looking for the most suitable examples.

Among all the parametric cubic polynomials, the most appropriate are those where simple functions ($t^3, t^2, t, 1$) (also referred to as basis functions) are replaced by *Bernstein polynomials* $\left((1-t)^3, 3 \cdot t \cdot (1-t)^2, 3 \cdot t^2 \cdot (1-t), t^3 \right)$ (Fig. 1.14). Using *Bernstein polynomials*, Eq. 1.9 can be written as:

$$P(t) = [\{P_0\} \quad \{P_1\} \quad \{P_2\} \quad \{P_3\}] \cdot \begin{Bmatrix} (1-t)^3 \\ 3t^2(1-t)^2 \\ 3t(1-t) \\ 1 \end{Bmatrix} \quad (1.10)$$

Equation 1.10 defines the Bézier cubic parametric polynomial and is a genius reformulation of the (Hermitian) shape of the cubic polynomial 1.8. It turns out that the vectors $\{P_0\}, \{P_1\}, \{P_2\}, \{P_3\}$ can be interpreted as $\{x, y, z\}$ coordinates and termed *control points*, and a combination of the four points $P_0 - P_1 - P_2 - P_3$ should be termed the *control polyline* (polygon) of the cubic parametric curve $P(t)$. Such an interpretation allows an intuitive determination of the shape of the entire curve if the position of only four control points is known.

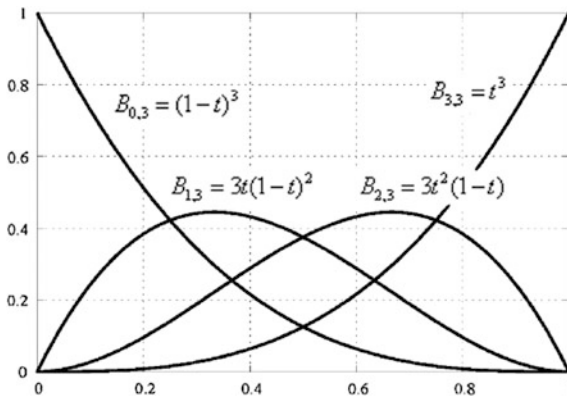


Fig. 1.14 Bernstein polynomial basis [1]

Such a defined curve has the following connection with the control points:

- the curve starts at the point P_0 and ends at the point P_3 ; it is parametric and defined for the parameter t over an interval $[0, \dots, 1]$; at the point P_0 , the parameter value is $t = 0$, at the point P_3 the parameter value is $t = 1$;
- the curve at the point P_0 and $P(t = 0)$ is tangential to the control polygon and the straight line $P_0 - P_1$; analogously, at the point P_3 and $P(t = 1)$ it is tangential to the line $P_2 - P_3$;
- the curve only approaches the points P_2 and P_3 .

A *Bézier* curve has other favourable properties:

- a plane convex control polygon results in a convex Bézier curve,
- using a Bernstein polynomial reduces the variations,
- with special geometric planning it is possible to reconstruct the shape of a curve from just the shape of the control polygon.

In geometric transformations, when describing curves using control points, it is possible to determine the curve in a transformed position as soon as you transform the control points.

The set of four Bernstein polynomials is a special case of a set of n -order blending functions. It can be written as:

$$B_{i,n}(t) = C(n, i) \cdot t^i \cdot (1 - t)^{n-i} \quad (1.11)$$

where the binomial coefficient is $C(n, i) = \frac{n!}{i!(n-i)!}$.

Bézier curves can also be defined by higher-order polynomials. It allows there to be any increase in the number of control points. Unfortunately, it turns out that increasing the order of polynomials also results in more complexity of the mathematical operations, taking place during operations among the geometric building blocks of computer models (e.g., calculating surface intersections). As a compromise between complexity and efficiency, cubic polynomials appeared in 3D modelling. A Bézier curve can also finally be formulated as:

$$P(t) = \sum_{i=0}^n P_i \cdot B_{i,n} \quad (1.12)$$

A third-order Bézier curve in parametric form is expressed as:

$$\begin{aligned} P(t) &= \sum_{i=0}^3 P_i \cdot B_{i,3} \\ &= P_0 \cdot (1 - t)^3 + P_1 \cdot 3t(1 - t)^2 + P_2 \cdot 3t^2(1 - t) + P_3 t^3 \end{aligned} \quad (1.13)$$

Example Let us determine the course of a Bézier curve, where a set of four points is known or we select them. In order to make the example simple, let all the points lie in the xy plane. The control points are:

$$P_0 = \{0, 0, 0\}^T \quad P_1 = \{1, 1, 0\}^T \quad P_2 = \{2, 0, 0\}^T \quad P_3 = \{3, 1, 0\}^T$$

Let us, for example, calculate 11 consecutive points on the curve at the parameter value $t = \{0.0; 0.1; 0.2; 1.0\}$:

$$\begin{aligned} P(0.0) &= P_0 = \{0, 0, 0\}^T \\ P(0.1) &= P_0 \cdot (1-t)^3 + P_1 \cdot 3t(1-t)^2 + P_2 \cdot 3t^2(1-t) + P_3 \cdot t^3 \\ &= \{0, 0, 0\} \cdot (1-0.1)^3 + \{1, 1, 0\} \cdot 3 \cdot 0.1 \cdot (1-0.1)^2 \\ &\quad + \{2, 0, 0\} \cdot 3 \cdot 0.1^2 \cdot (1-0.1) + \{3, 1, 0\} \cdot 0.1^3 \\ &= \{0.30, 0.24, 0\}^T \\ P(0.2) &= \{0.60, 0.39, 0\}^T \\ P(0.3) &= \{0.90, 0.47, 0\}^T \\ P(0.4) &= \{1.20, 0.50, 0\}^T \\ P(0.5) &= \{1.50, 0.50, 0\}^T \\ P(0.6) &= \{1.80, 0.50, 0\}^T \\ P(0.7) &= \{2.10, 0.53, 0\}^T \\ P(0.8) &= \{2.40, 0.61, 0\}^T \\ P(0.9) &= \{2.70, 0.67, 0\}^T \\ P(1.0) &= P_3 = \{3, 1, 0\}^T \end{aligned}$$

The control points, polyline and calculated points on a Bézier curve are shown in Fig. 1.15.

1.3.2 B-splines

Bézier curves are not suitable for direct use in 3D modelling. A Bézier curve has two major deficiencies: globality (poor definition of localities) and too few control points. If one control point moves, it to some extent affects the entire curve. Bézier curves can generally also not be drawn through an arbitrary number n of points without *merging* several segments, each described separately.

You can overcome both deficiencies by defining the B-splines. Like with a Bézier curve, a B-spline is also determined by a control polyline and only approaches the control points, despite now having an unlimited number of points. A B-spline is piecewise cubic on some interval of the parameter t . When t exceeds the interval boundary, the coefficients change accordingly.

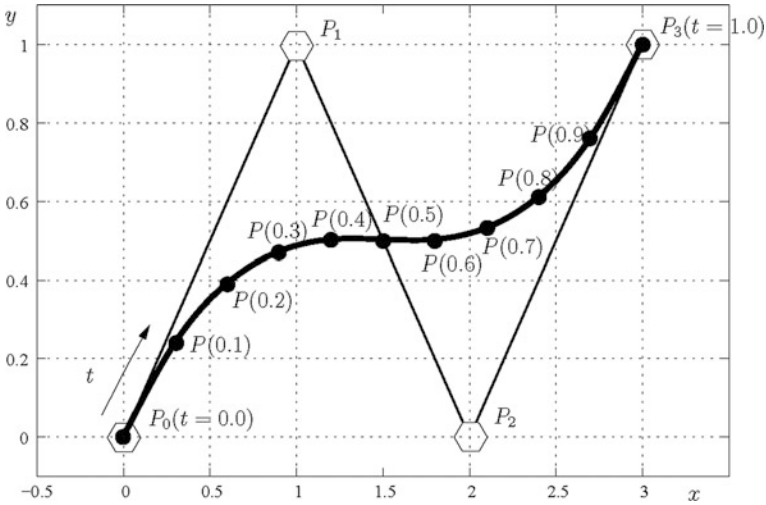


Fig. 1.15 Determining points on a Bézier curve

A B-spline can be expressed in a similar way to a Bézier curve with blending functions:

$$P(t) = \sum_{i=0}^n P_i \cdot N_{i,k} \tag{1.14}$$

Equation 1.14 includes a curve with $(n + 1)$ control points P_i , whereas the blending functions $N_{i,k}$ are of $(k - 1)$ order. In this case, the order (or degree) of the blending function is completely independent of the number of control points. Instead of Bézier (Bernstein) blending functions a slightly different basis was used. The key novelty is that it is possible to increase the number of control points in the B-spline's blending functions without increasing the order of the blending function. The blending functions for a B-spline can be defined by means of a recursive algorithm (Eqs. 1.15 and 1.16), published by *C. de Boor* in 1972 [3, 4]:

$$N_{i,1}(t) = \begin{cases} 1, & t_i \leq t < t_{i+1} \\ 0, & \text{elsewhere} \end{cases} \tag{1.15}$$

And, when $k > 1$:

$$N_{i,k}(t) = \frac{t-t_i}{t_{i+k-1}-t_i} N_{i,k-1}(t) + \frac{t_{i+k}-t}{t_{i+k}-t_{i+1}} N_{i+1,k-1}(t) \tag{1.16}$$

The variable t is a coordinate in the curve's parametric space, and t_i are the values of the knot vector T [5]. You can now imagine the curve as a set of polynomial sections (segments) with different values of the variable t . The point where the two segments *merge* is referred to as *knot* t_i , and a set of knots as a knot vector:

$$T = \{t_0, t_1, \dots, t_i, \dots, t_{n+k}\} \quad (1.17)$$

Formulating a Bézier curve, the variable t was assigned a value over the interval $[0, \dots, 1]$. Formulating a B-spline, the knots can in principle assume any value, the only condition being that they increase monotonically. The shape of the curve depends slightly on the selection of the value for the knot vector.

Some knots can even be repeated several times. When two knots coincide to form one, it is referred to as a knot having *twofoldness*.

A particularly interesting example of a B-spline is a knot vector where the first k and the last k knots are repeated. A k -order B-spline can have up to $k-1$ continuous derivatives. Each knot repetition reduces the number of possible continuous derivatives by 1. In the case of a k -fold repetition of knots, the curve begins at the first and ends at the last control point, whereas at these two points the curve is tangential to the control polyline (analogous to a Bézier curve). In the case of a cubic spline ($k = 4$) the first four and the last four values of the knot vector are repeated. In such cases, when (only) the first k and the last k knots of the knot vector are repeated (Eq. 1.8), this is referred to as the knot vector being *clamped*.

$$T = \left\{ \underbrace{t_0, t_1, \dots, t_{k-1}}_{k \text{ identical knots}}, \underbrace{t_k, t_{k+1}, \dots, t_{n-1}, t_n}_{n-k+1 \text{ internal knots}}, \underbrace{t_{n+1}, \dots, t_{n+k}}_{k \text{ identical knots}} \right\} \quad (1.18)$$

where

$$\begin{aligned} t_0 = t_1 = \dots = t_{k-1} & \quad k \text{ identical knots} \\ t_k, t_{k+1}, \dots, t_{n-1}, t_n & \quad n-k+1 \text{ internal knots} \\ t_{n+1} = t_{n+2} = \dots = t_{n+k} & \quad k \text{ identical knots} \end{aligned}$$

Clamped B-splines are a special example of curves that are very often used in computer graphics and modellers. Their practical value lies in the fact that the curve begins at the first and ends at the last control point (Fig. 1.16), and that the curve at these two points is tangential to the control polygon (analogous to a Bézier curve). When a different knot vector is used, the ends of the curve do not coincide with the first and the last control points. When knots are repeated, it yields a quotient $0/0$ in the *de Boor* algorithm. In programming the expression $0/0$ should be equalised with 0.

Example Let us take the control polygon in Fig. 1.16, where $(n+1) = 7$ control points are given. The blending functions of the B-spline are cubic, i.e., third-order, and $k = 4$. Therefore, there are 7 blending functions in total $N_{0,4}(t), N_{1,4}(t), \dots, N_{6,4}(t)$, and the number of knots is $n+1+k = 11$. The curve shown for the clamped B-spline therefore consists of 4 segments of cubic blending functions. The clamped-knot vector is therefore expressed as:

$$T = \{0, 0, 0, 0, 1, 2, 3, 4, 4, 4, 4\} \quad (1.19)$$

Figure 1.17 shows the blending functions $N_{1,4}(t)$, for a cubic $k = 4$ B-spline, defined by means of the knot vector 1.19.

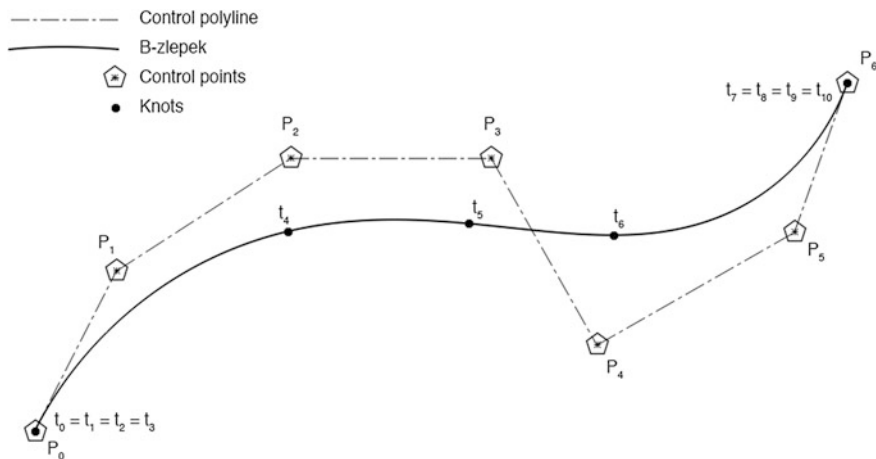


Fig. 1.16 A B-spline with seven control points, clamped to the first and last control points

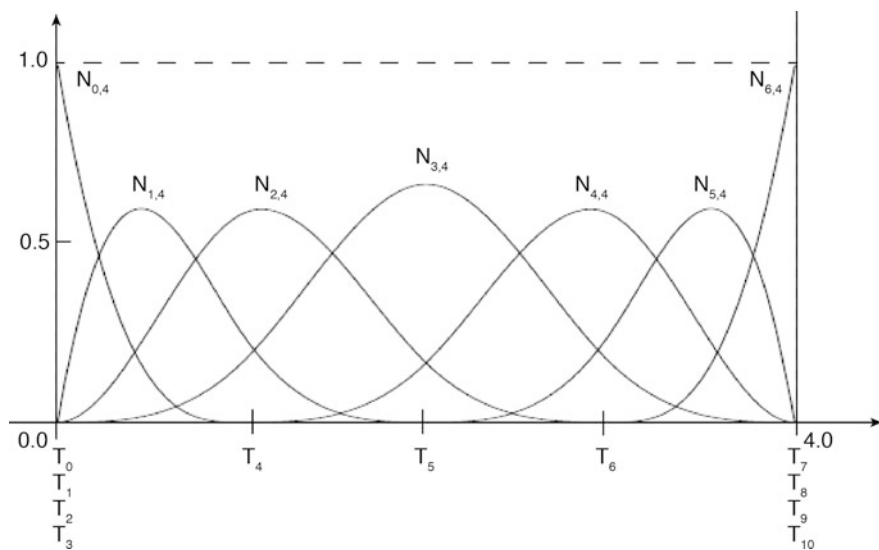


Fig. 1.17 Blending functions of a cubic B-spline defined by the knot vector $T = \{0, 0, 0, 0, 1, 2, 3, 4, 4, 4, 4\}$

All the examples of B-splines presented above had knots over an interval $[0, n + k]$. It is nevertheless advisable to normalise the knot vector so that it involves the interval $[0, 1]$. This improves the accuracy of numerical calculations with a floating point due to the higher density of decimal places over this interval [6].

When knots divide the whole parametric space of the curve $P(t)$ into uniform intervals (e.g., the one in Eq. 1.19 or another vector written in a normal form: $T = \{0, 0, 0, 0, 0.2, 0.4, 0.6, 0.8, 1, 1, 1, 1\}$), this is referred to as the curve being a uniform B-spline. Generally, these intervals do not need to be uniform; knots can be distributed unequally across the parametric space of the curve. The only important thing is that the order of knots is not falling, i.e., that the next knot is higher or equal to the preceding one.

It has been explained above what happens when knots at the beginning and the end of the knot vector are repeated k -times. A special example of non-uniform B-splines is the curves where multiple internal knots appear (e.g., $T = \{0, 0, 0, 0, 0.2, 0.2, 0.2, 0.8, 1, 1, 1, 1\}$). Increasing the multiplicity s of internal knots decreases the continuity level of the curve in a given knot as a result of the decreasing number of blending curves that do not equal 0 in this knot. When the multiplicity of internal knots is $s = k - 1$ (e.g., three repetitions of the internal knot in a $k = 4$ -order cubic spline), it leaves only one influential blending function, which means that only one control point affects that particular knot. This results in the curve passing through this particular control point; however, its transition from the left to the right of this point is no longer smooth (tangential), due to $C^{k-1-s} = C^0$ (see Sect. 1.5). A further increase in the multiplicity of internal knots to the value $k \leq s$ makes no sense, as this would remove the last remaining blending function at a given control point, which would eliminate any effect of the control point on the curve.

Some interesting properties can be achieved by increasing the multiplicity of internal knots. These include special tangent conditions or even sharp edges on the shape of the curve. Figure 1.18 shows a B-spline defined on the same set of control points. In the first case the multiplicity (repeatability of the same point of a spline) of the third control point equals one, in the second case it is two and three in the third case. The knot vectors for all three cases are as follows:

$$\begin{aligned} T_1 &= \{0, 0, 0, 0, 1, 2, 3, 4, 4, 4, 4\} \\ T_2 &= \{0, 0, 0, 0, 1, 2, 2, 3, 4, 4, 4, 4\} \\ T_3 &= \{0, 0, 0, 0, 1, 2, 2, 2, 3, 4, 4, 4, 4\} \end{aligned}$$

1.3.3 Increasing Curve Control—NURBS

Ordinary B-splines, as discussed above, are a powerful tool in computer geometry in their own right; however, they lack further flexibility due to the B-splines being unable to exactly describe the curves in the family of conic sections (i.e., parts of circles, ellipses, parabolas or hyperbolas). For this reason, the need arose to rationalise B-splines by adding to each control point of a B-spline a new parameter w_i (the fourth coordinate), referred to as the *weight*. It allows an accurate determination of the effect of the control point. In the Cartesian coordinate system, such an *enriched* control point can be written as:

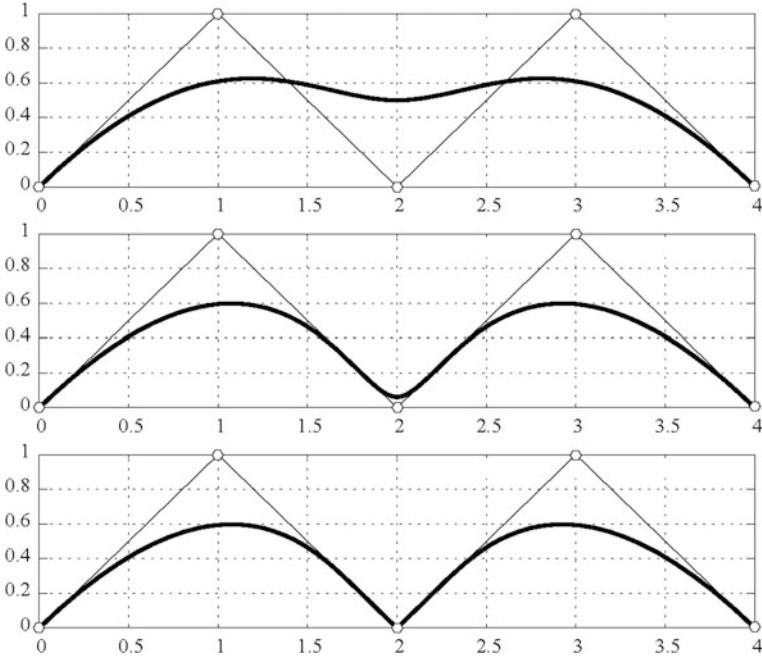


Fig. 1.18 Effect of n-multiplicity of control points on the shape of a B-spline. Above: $t_3 = 3$; Centre: $t_3 = t_4 = 3$; Below: $t_3 = t_4 = t_5 = 3$. Knot vectors have a different number of knots in each case, but they are all over an interval $[0, 4]$

$$P_i^w = w_i P_i = w_i \begin{Bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{Bmatrix} = \begin{Bmatrix} w_i x_i \\ w_i y_i \\ w_i z_i \\ w_i \end{Bmatrix} \tag{1.20}$$

A rational form of a non-uniform B-spline (NURBS) that introduces control-point weights is defined as:

$$P(t) = \frac{\sum_{i=0}^n P_i w_i N_{i,k}(t)}{\sum_{i=0}^n w_i N_{i,k}(t)} \text{ or } \tag{1.21}$$

$$P(t) = \sum_{i=0}^n P_i R_{i,k}(t) \text{ where } : R_{i,k}(t) = \frac{w_i N_{i,k}(t)}{\sum_{j=0}^n w_j N_{j,k}(t)} \tag{1.22}$$

It turns out that the weight w_i affects the curve only locally, around the point P_i . Increasing the value of the weight w_i pulls the curve closer to the control point, whereas decreasing it pushes it away from the control point. If all the weights are assigned a value of 1, it yields an ordinary non-rational B-spline. Another special case is when the value $w_i = 0$. In this case the control point P_i has no effect on the

curve. The effects of the value w_i are shown in Figs. 1.19 and 1.20. The latter shows how the different values of the weight w_i define different types of conic sections.

1.3.4 Describing Surfaces

Describing the curves from the previous sections can be simply generalised into two dimensions: a point on the surface $P(s, t)$ is given by a biparametric function of the parameters s and t . It consists of blending functions for each parameter.

Hence, a cubic Bézier surface is defined as:

$$P(s, t) = \sum_{i=0}^3 \sum_{j=0}^3 P_{ij} \cdot B_{i,3}(s) \cdot B_{j,3}(t) \tag{1.23}$$

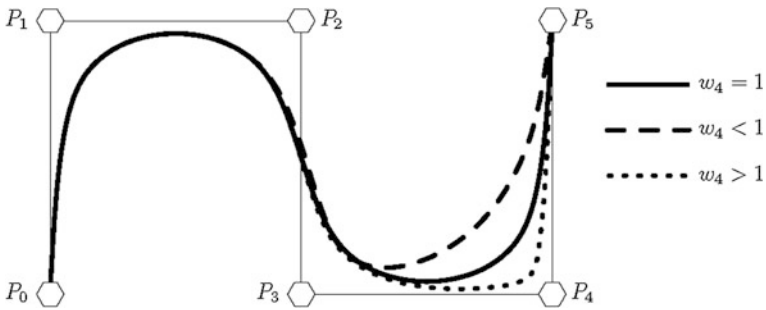


Fig. 1.19 Effect of the parameter w_i at the point P_i on the course of a NURBS curve [1]

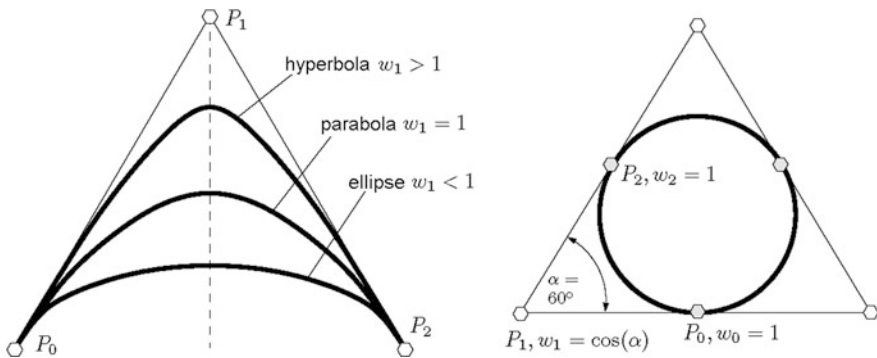
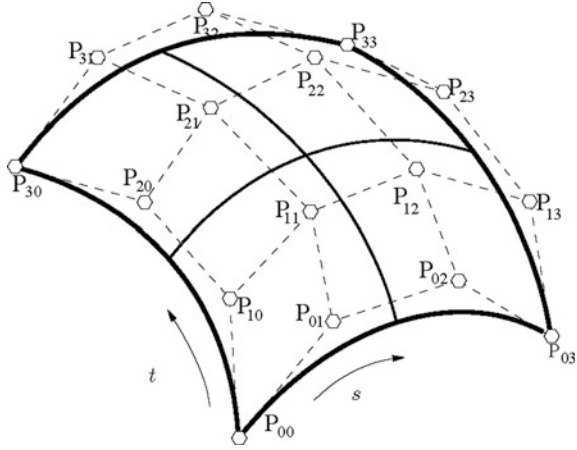


Fig. 1.20 Adjusting the parameter w_i allows an accurate description of different types of conic sections by means of NURBS. The circle consists of three circular arcs [1]

Fig. 1.21 Bézier surface with control points



A total of 16 control points make up a control polyhedron, as shown in Fig. 1.21.

Analogously to curves, the above definition can be extended to B-splines or NURBS, where a control polyhedron from $(n + 1) \times (m + 1)$ control points takes part:

$$P(s, t) = \sum_{i=0}^n \sum_{j=0}^m P_{ij} \cdot R_{i,k}(s) \cdot R_{j,l}(t) \tag{1.24}$$

1.3.5 B-spline Interpolation

When modelling complex curves or surfaces, presenting surfaces by means of B-spline interpolations is particularly interesting if we are given a set of points over which *the finest possible* curves or surfaces are to be stretched. Such a smooth surface is sought when we record or scan a surface that is to be used for further developments.

In the case of B-spline interpolations, the interpolation problem will be solved when a set of given points can serve as a basis for the calculation or determining such a set of control points that a B-spline will pass through. For cubic splines, however, the problem is not uniquely solvable. This is discussed in more detail in [4] and [7]. In our case we will only focus on some of the most useful examples.

The simplest option is for the knots to appear at exactly the same positions as the interpolation points X_i . In this case we need to choose a value of the parameter t at the knots—a procedure usually referred to as parameterisation. The method of parameterisation slightly defines the shape of the curve between the knots. For example, we can choose a uniform parameterisation (an example of a knot vector: $T = \{0, 0, 0, 0, 1, 2, \dots, 6, 6, 6, 6\}$). However, it turns out that such a

parameterisation will not yield an intuitively expected course of the curve. The reason lies in the fact that such a method takes no account of the spatial distribution of the points. Significantly better results are obtained if the parameterisation takes account of the distances between the points. One such method is cord-length parameterisation, where the differences between the values of the parameter of consecutive knots are proportional to the distance between the points.

$$\frac{\Delta_i}{\Delta_{i+1}} = \frac{\|\Delta X_i\|}{\|\Delta X_{i+1}\|} \quad (1.25)$$

The difference between two adjacent values of the parameter $t_{i+1} - t_i$ in the knot vector is written as Δ_i , whereas $\|\Delta X_{i+1} - \Delta X_i\|$ represents the chord length ('air distance') between two consecutive knots on the curve.

In some special cases a centripetal parameterisation yields even better results. This can be written using the following equation:

$$\frac{\Delta_i}{\Delta_{i+1}} = \left[\frac{\|\Delta X_i\|}{\|\Delta X_{i+1}\|} \right]^{1/2} \quad (1.26)$$

Figure 1.22 shows B-splines that interpolate the same data set. Three different types of parameterisation were used. We can see that the choice of parameterisation has a major effect on the shape of the curve between the interpolation points.

Our interpolation problem will be defined as follows. There is a given set of space (interpolation) points X_0, \dots, X_L and a corresponding parameterisation (a set of parameter t values in the knots t_0, \dots, t_L), and the number of control points is larger than the number of interpolation points. Let us find the cubic B-spline $P(t)$, defined by the same set of knots and unknown control points P_{-1}, \dots, P_{L+1} so that the following is true $P(t_i) = X_i$.

It turns out that the set of control points can be defined by solving the following (tri-diagonal) system of equations:

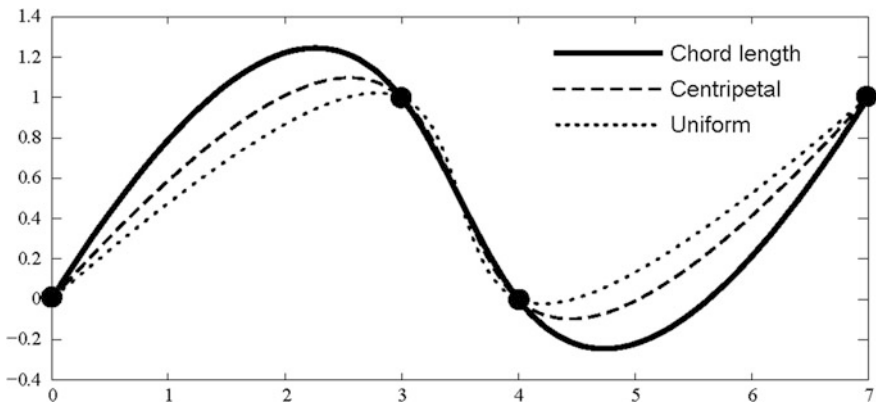


Fig. 1.22 Effect of parameterisation on the shape of a curve

where δ_0 and ε_0 are defined by the equations:

$$\delta_0 = \frac{t_2 - t_1}{t_2 - t_0}, \text{ and } \varepsilon_0 = 1.0 - \delta_0 \quad (1.31)$$

The second-to-last control point P_L is determined by an analogous rearrangement of Eqs. 1.30 and 1.31: (Fig. 1.23)

$$P_L = \left(\frac{\delta_L + 1}{3}\right)X_L + \frac{1}{3\delta_L}X_{L-1} - \frac{\varepsilon_L^2}{3\delta_L}X_{L-2} \quad (1.32)$$

where δ_L and ε_L are defined by the equations:

$$\delta_L = \frac{t_2 - t_1}{t_2 - t_0}, \text{ and } \varepsilon_L = 1.0 - \delta_L \quad (1.33)$$

An example of parameterizing a curve and defining control points: As an example, let's take the following (planar) set of points, through which a curve shall pass (Fig. 1.23):

$$\begin{aligned} X_0 &= \{0, 0, 0\}^T & X_1 &= \{1, 1, 0\}^T & X_2 &= \{2, 1, 0\}^T & X_3 &= \{2.5, 0.5, 0\}^T \\ X_4 &= \{4, 2, 0\}^T \end{aligned}$$

With such defined and selected points we can approach parameterization. For this purpose, let's calculate the distance between the points:

$$d_1 = 1.4142, \quad d_2 = 1.0000, \quad d_3 = 0.7071, \quad d_4 = 2.1213.$$

The sum of distances is 5.2426. If the parameter t is to have value 0 at the beginning of the curve and value 1 at its end, the distances should be normalized.

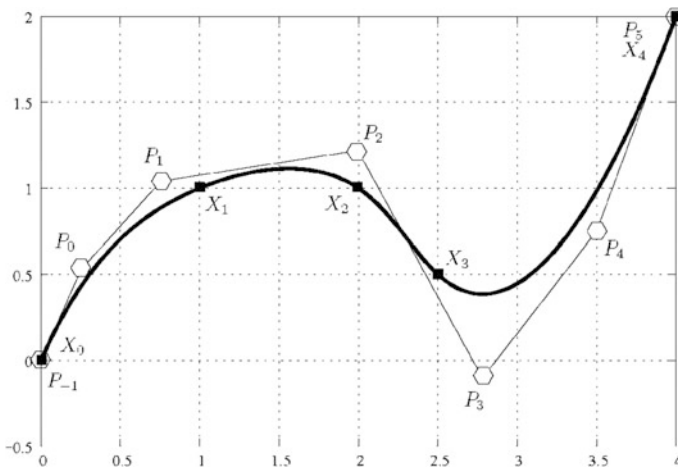


Fig. 1.23 Example of B-spline interpolation

This can be achieved by dividing them with the cumulative length. Parameterization now corresponds to the cumulative normalized distance:

$$t_0 = 0.0000, \quad t_1 = 0.2698, \quad t_2 = 0.6405, \quad t_3 = 0.5954 \quad t_4 = 1.0000$$

Let's calculate the differences Δ_i :

$$\Delta_{-1} = 0, \quad \Delta_0 = 0.2698, \quad \Delta_1 = 0.1907, \quad \Delta_2 = 0.1349, \quad \Delta_3 = 0.4046, \\ \Delta_4 = 0.0$$

Let's calculate the coefficients $\alpha_i, \beta_i, \gamma_i$ for an index of choice, e.g. $i = 2$:

$$\alpha_2 = \frac{(\Delta_2)^2}{\Delta_0 + \Delta_1 + \Delta_2} = 0.0306 \quad (1.34a)$$

$$\beta_2 = \left(\frac{\Delta_2(\Delta_0 + \Delta_1)}{\Delta_0 + \Delta_1 + \Delta_2} \right) + \left(\frac{\Delta_1(\Delta_2 + \Delta_3)}{\Delta_1 + \Delta_2 + \Delta_3} \right) = 0.2452 \quad (1.34b)$$

$$\gamma_2 = \frac{(\Delta_1)^2}{\Delta_1 + \Delta_2 + \Delta_3} = 0.0498. \quad (1.34c)$$

The system of equations now reads:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0.0790 & 0.2593 & 0.1222 & 0 & 0 \\ 0 & 0.0306 & 0.2452 & 0.0498 & 0 \\ 0 & 0 & 0.2242 & 0.2816 & 0.0337 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix} = \begin{bmatrix} \left\{ \begin{array}{c} 0.25 \\ 0.53 \\ 0 \end{array} \right\} \\ \left\{ \begin{array}{c} 0.46 \\ 0.46 \\ 0 \end{array} \right\} \\ \left\{ \begin{array}{c} 0.65 \\ 0.33 \\ 0 \end{array} \right\} \\ \left\{ \begin{array}{c} 1.35 \\ 0.27 \\ 0 \end{array} \right\} \\ \left\{ \begin{array}{c} 3.50 \\ 0.75 \\ 0 \end{array} \right\} \end{bmatrix} \quad (1.35)$$

The point \mathbf{P}_0 was defined by means of Bessel boundary condition 1.30:

$$\delta = \frac{t_2 - t_1}{t_2 - t_0} = \frac{0.4605 - 0.2698}{0.4605 - 0.0000} = 0.4141, \text{ and } \epsilon = 1.0 - \delta = 0.5859$$

$$\mathbf{P}_0 = 0.4714 \cdot \mathbf{X}_0 + 0.8049 \cdot \mathbf{X}_1 - 0.2763 \cdot \mathbf{X}_2$$

$$= 0.4714 \cdot \{0, 0, 0\} + 0.8049 \cdot \{1, 1, 0\}^T - 0.2763 \cdot \{2, 1, 0\}^T \quad (1.36)$$

Analogously, the second to last point \mathbf{P}_4 is added:

The solution of the system of equations:

$$\mathbf{P}_0 = \{0.2525, 0.5286, 0\}^T$$

$$\mathbf{P}_1 = \{0.7583, 0.0405, 0\}^T$$

$$\mathbf{P}_2 = \{1.9959, 1.2189, 0\}^T$$

$$\mathbf{P}_3 = \{2.7816, -0.1024, 0\}^T$$

$$\mathbf{P}_4 = \{3.500, 0.7500, 0\}^T$$

An example of surface interpolation as a bicubic B-spline: Similarly to defining curve interpolation through space points, surface interpolation can be defined as a bicubic B-spline (tensor product of a B-spline). The interpolation problem can be formulated similarly to what we did for curves.

A set of space points \mathbf{X}_{ij} is given. They are organized in a rectangular (orthogonal) scheme with dimensions $(K + 1) \times (L + 1)$. Also given is a corresponding parameterization (a set of the parameter s values at knots s_0, \dots, s_K , and the parameter t with knots t_0, \dots, t_L). The objective is to find the B-spline $\mathbf{P}(s, t)$, defined by the same set of knots and unknown control points \mathbf{P}_{ij} , so that:

$$\mathbf{P}(s_i, t_j) = \mathbf{X}_{ij} \quad (1.37)$$

Our algorithm is shown in Fig. 1.24. For each line of the data points \mathbf{X}_i , two boundary conditions shall be prescribed (Bessel, for example) and the interpolation problem shall be solved, as was done for individual curves. This is followed by solving the interpolation problem by columns, where additional points appear, defining boundary conditions from the preceding phase. The resulting rectangular scheme of control points defines the B-spline that interpolates the data set \mathbf{X}_{ij} .

The B-spline is defined by means of parameterization in the s - and t -direction. Each line (s -direction) therefore requires the same parameterization. When interpolation points in individual lines (columns) in space are very unevenly distributed, it is difficult to define the parameterization that would yield a satisfactory form of interpolation in all parts of the bicubic spline.

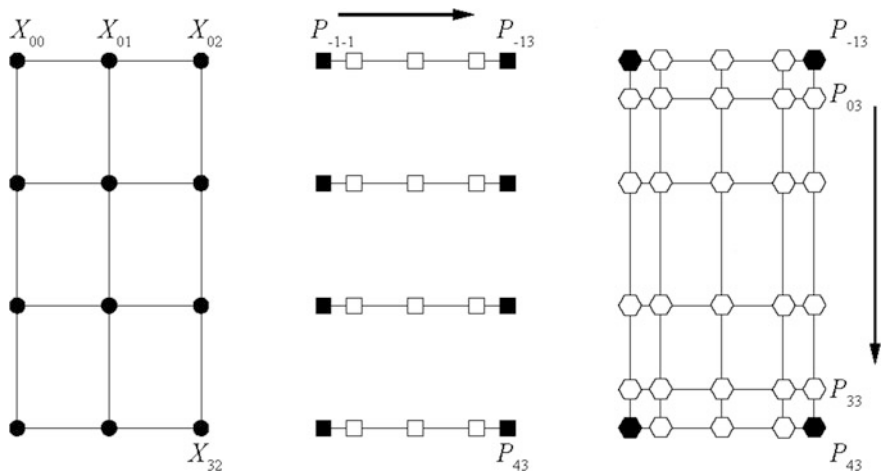


Fig. 1.24 Diagram of the algorithm for defining a bi-cubic B-spline

Table 1.1 Interpolation points X_{ij}

Index	i = 0	i = 1	i = 2	i = 3	i = 4
j = 0	0.0	1.0	2.0	2.5	4.0
	0.0	1.0	1.0	0.5	2.0
	0.0	0.0	0.0	0.0	0.0
j = 1	0.0	0.9	1.8	2.5	3.7
	0.5	2.5	2.0	1.5	2.0
	2.0	2.0	2.0	2.0	2.0
j = 2	0.0	1.0	2.0	3.0	4.0
	0.0	0.8	1.0	0.7	0.0
	4.0	4.0	4.0	4.0	4.0

As an example, let's take a (space) set of points (Table 1.1), through which a surface shall pass (Fig. 1.25).

Define the first-phase parameterisation. The procedure is similar to that for a curve:

1. Calculating the distance between the points in lines;
2. Normalising the distances so that the parameter t assumes the value 0 at the beginning of the curve and the value 1 at the end;
3. Calculating the mean value of the parameterisation according to the columns.

The results of the first-phase parameterisation are collected in Table 1.2.

Continue by following the scheme in Fig. 1.24. Calculate the control points for three curves in the s -direction following the procedure described in the previous example. The control points are given in Table 1.3.

The calculated control points are used as the input data for the algorithm used to define the control points in the t -direction. The parameterisation in the second phase is defined by the numerical values given in Table 1.4.

Table 1.4 Parameterisation of a bi-cubic B-spline in the t -direction, second phase

Index	$j = 0$	$j = 1$	$j = 2$
$i = 0$	0.000	0.500	1.000
$i = 1$	0.000	0.490	1.000
$i = 2$	0.000	0.483	1.000
$i = 3$	0.000	0.499	1.000
$i = 4$	0.000	0.514	1.000
$i = 5$	0.000	0.461	1.000
$i = 6$	0.000	0.416	1.000
Average:	0.000	0.480	1.000

Table 1.5 Control points $P_{i,j}$ after the second phase

Index	$i = 0$	$i = 1$	$i = 2$	$i = 3$	$i = 4$	$i = 5$	$i = 6$
$j = 0$	0.000	0.214	0.660	2.107	2.495	3.367	4.000
	0.000	0.538	1.020	1.207	0.042	0.997	2.000
	0.000	0.000	0.000	0.000	0.000	0.000	0.000
$j = 1$	0.000	0.201	0.634	1.938	2.474	3.259	3.808
	0.321	1.332	2.291	1.737	0.729	1.454	2.296
	0.691	0.691	0.691	0.691	0.691	0.691	0.691
$j = 2$	0.000	0.186	0.622	1.723	2.589	3.216	3.599
	0.668	2.164	3.625	2.299	1.559	1.833	2.296
	2.079	2.079	2.079	2.079	2.079	2.079	2.079
$j = 3$	0.000	0.200	0.698	1.792	3.030	3.550	3.792
	0.347	1.285	2.220	1.731	1.142	1.017	1.103
	3.362	3.362	3.362	3.362	3.362	3.362	3.362
$j = 4$	0.000	0.214	0.760	1.903	3.347	3.825	4.000
	0.000	0.365	0.751	1.130	0.588	0.271	0.000
	4.000	4.000	4.000	4.000	4.000	4.000	4.000

At the end you will obtain the control points that are defined after the second phase of the algorithm and given in Table 1.5. Figure 1.25 shows the data, the calculated control points and the course of the surface in three-dimensional space.

1.4 Surface Interpolation with Boundary Curves

We have explained one of the methods for creating curves and surfaces, useful for 3D modelling. Below, a slightly different method will be presented. Instead of describing the surfaces using control polygons, the definitions will *fill* the space between the boundary curves. The techniques used in this process are named after their authors, *S. Coons* and *W. Gordon*. Before describing them we shall introduce an important part of their definition. In both cases it is based on a ruled surface.

1.4.1 Ruled Surface

Ruled surfaces are simple and form the basis for creating surface models. Their boundary curves $C_1(s)$ and $C_2(s)$ are given in advance (Fig. 1.26). Let both be defined over the same interval of the parameter $s \in [0, \dots, 1]$. The surface $P(s, t)$ should be found such that the following is true:

$$P(s, t = 0) = C_1(s), P(s, t = 1) = C_2(s) \quad (1.38)$$

Of course the problem has many solutions. Let us try a *linear interpolation* between both curves:

$$P(s, t) = (1 - t) \cdot C_1(s) + t \cdot C_2(s) \quad (1.39)$$

To do this, *linear blending functions* $f_1 = t$ and $f_2 = 1 - t$ were used. The interpolation was performed by taking into account the whole space between the entire C_1 and C_2 curves, not only the space between the discrete points. Some authors call this process *transfinite interpolation*. It is also important that this definition does not restrict the type of boundary curves. The first one can be a B-spline, for example, and the other one a polyline or a section of an analytically presented function. The shape of a ruled surface depends on the parameterisation of both curves (Sect. 1.4).

1.4.2 Coons Patch

A ruled surface can only define an interpolation with two boundary curves, which places a major restriction on defining freeform surfaces, which are usually defined by four edges. For this reason we will expand our problem to four boundary curves, as follows: $C_1(s)$, $C_2(s)$, $C_3(s)$ and $C_4(s)$ (Fig. 1.27).

Fig. 1.26 Ruled surface

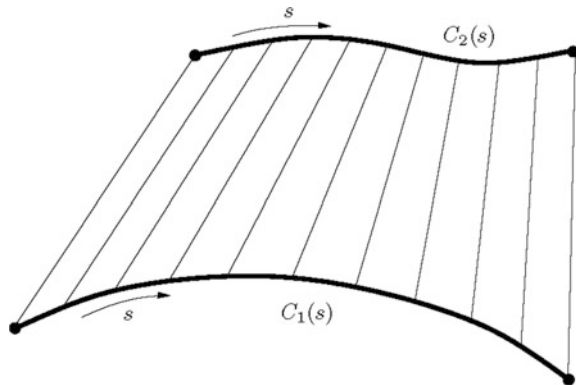
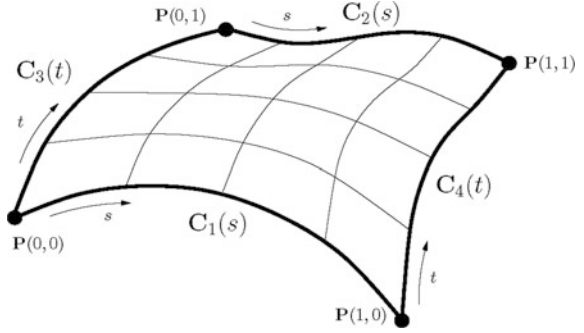


Fig. 1.27 Coons patch



Let all of them be defined over the same interval of the parameters $s \in [0, \dots, 1]$ and $t \in [0, \dots, 1]$. The surface $\mathbf{P}(s, t)$ should be found so that the following is true:

$$\mathbf{P}(s, t = 0) = \mathbf{C}_1(s), \mathbf{P}(s, t = 1) = \mathbf{C}_2(s) \quad (1.40)$$

$$\mathbf{P}(s = 0, t) = \mathbf{C}_3(t), \mathbf{P}(s = 1, t) = \mathbf{C}_4(t) \quad (1.41)$$

Ruled surfaces were introduced in Sect. 1.4.1. Let us now try to use them as a constituent part of a new definition:

$$\mathbf{P}_{\text{ruled, direction } t}(s, t) = (1 - t) \cdot \mathbf{P}(s, 0) + t \cdot \mathbf{P}(s, 1) \quad (1.42)$$

$$\mathbf{P}_{\text{ruled, direction } s}(s, t) = (1 - s) \cdot \mathbf{P}(0, t) + s \cdot \mathbf{P}(1, t) \quad (1.43)$$

However, it turns out that simply adding up two interpolate surfaces, defined by two opposite curves, is not sufficient and wrong. Subtracted from the sum should be the bilinear interpolation $\mathbf{P}_{biv}(s, t)$, defined by the corners of the surface under consideration:

$$\mathbf{P}_{biv}(s, t) = [1 - s, s] \begin{bmatrix} \mathbf{P}(0, 0) & \mathbf{P}(0, 1) \\ \mathbf{P}(1, 0) & \mathbf{P}(1, 1) \end{bmatrix} \begin{Bmatrix} 1 - t \\ t \end{Bmatrix} \quad (1.44)$$

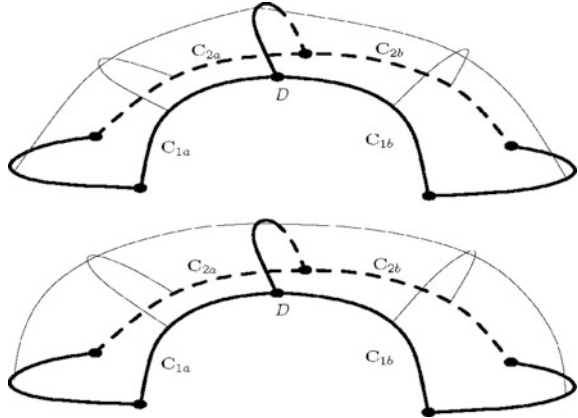
Finally, a Coons patch can be put together in the form of an equation:

$$\mathbf{P}(s, t) = \mathbf{P}_{\text{ruled, direction } t}(s, t) + \mathbf{P}_{\text{ruled, direction } s}(s, t) - \mathbf{P}_{biv}(s, t) \quad (1.45)$$

or in expanded form:

$$\begin{aligned} \mathbf{P}(s, t) = & [1 - s, s] \begin{Bmatrix} \mathbf{P}(0, t) \\ \mathbf{P}(1, t) \end{Bmatrix} + [\mathbf{P}(s, 0) \quad \mathbf{P}(s, 1)] \begin{Bmatrix} 1 - t \\ t \end{Bmatrix} \\ & - [1 - s, s] \begin{bmatrix} \mathbf{P}(0, 0) & \mathbf{P}(0, 1) \\ \mathbf{P}(1, 0) & \mathbf{P}(1, 1) \end{bmatrix} \begin{Bmatrix} 1 - t \\ t \end{Bmatrix} \end{aligned} \quad (1.46)$$

Fig. 1.28 Adjacent *Coons patches* using: a set of linear blending functions (above), and a set of cubic blending functions (below)



To create a ruled surface we used one of the simplest sets of blending functions. These are linear functions $f_1 = t$ and $f_2 = 1 - t$. Other sets could also be used; the cubic one, for example, $f_1 = (1 - t)^3 + 3t(1 - t)^2$ and $f_2 = 3t^2(1 - t) + t^3$. The advantage of the latter is shown in Fig. 1.28. Despite the curves C_{1a} and C_{1b} having a continuous derivative to the point D , this is not true of the adjacent *Coons patches*, when a set of linear blending functions is used (Fig. 1.28a). Figure 1.28b shows adjacent *Coons patches* with this cubic set of blending functions.

1.4.3 Gordon Patch

A *Gordon patch* is a generalisation of a *Coons patch*. Its concept was developed by William J. Gordon around 1970, while working for General Motors Research Labs. He was dealing with the problem that often not only four boundary curves are given, but also a mesh $(n + 1) \times (m + 1)$ of curves. Such a mesh can usually be obtained by means of 3D-recording or mechanical measuring with the use of contact or non-contact measuring devices (Fig. 1.29).

A Gordon patch is therefore aimed at defining freeform surfaces when a data structure with the $(n + 1) \times (m + 1)$ system of curves is given. The problem can be formulated with given curves:

$$C_{s,0}(s), C_{s,1}(s), \dots, C_{s,n}(s) \text{ and } C_{t,0}(t), C_{t,1}(t), \dots, C_{t,m}(t)$$

Let both sets be defined over the same interval of the parameter $s \in [0, \dots, 1]$ and $t \in [0, \dots, 1]$. The surface $P(s, t)$ should be found such that the following is true:

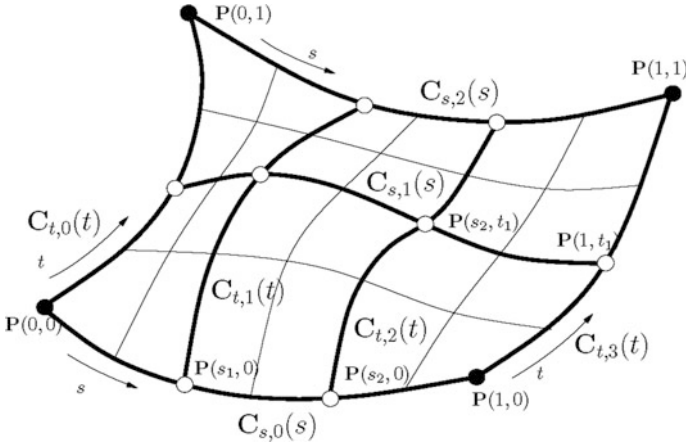


Fig. 1.29 Gordon patch interpolating a mesh of curves

$$P(s, t_i) = C_{s,i}(s) \text{ where } i = 0, \dots, n \quad (1.47)$$

$$P(s_i, t) = C_{t,i}(t) \text{ where } i = 0, \dots, m \quad (1.48)$$

The idea is similar to that for a Coons patch: first, find the constituent part $P_{directt}(s, t)$ that allows the interpolation of one family of iso-parametric curves in the direction t . This is followed by finding the other constituent part $P_{directs}(s, t)$. At the end, add the two parts together and subtract the interpolant that includes the corners $P_{iv}(s, t)$.

To define a ruled surface, *linear blending functions* $f_1 = t$ and $f_2 = 1-t$ were used. With more edges, a higher-order polynomial interpolation such as a *Lagrange* polynomial can be used:

$$L_i^m(s) = \frac{\prod_{j=0, j \neq i}^m (s-s_j)}{\prod_{j=0, j \neq i}^m (s_i-s_j)} \quad (1.49)$$

The first and the second constituent parts can now be generalised into:

$$P_{\text{direction } s}(s, t) = \sum_{i=0}^m P(s_i, t) L_i^m(s) \quad (1.50a)$$

$$P_{\text{direction } t}(s, t) = \sum_{j=0}^n P(s, t_j) L_j^n(t) \quad (1.50b)$$

The third constituent part is:

$$P_{iv}(s, t) = \sum_{i=0}^m \sum_{j=0}^n P(s_i, t_j) L_i^m(s) L_j^n(t) \quad (1.51)$$

And finally, the Gordon patch is:

$$\mathbf{P}(s, t) = \mathbf{P}_{\text{smers}}(s, t) + \mathbf{P}_{\text{smert}}(s, t) - \mathbf{P}_{iv}(s, t) \quad (1.52)$$

The Gordon patch concludes the presentation of mathematical formulations that present a significant basis for the modelling of bodies in space. A detailed study of special surfaces and the shapes of bodies requires additional skills in advanced mathematics. Below, we would like to present the application of the above-presented modelling and designing knowledge in practical use, as top designing and shaping can only be achieved through a simultaneous understanding of the mathematical background of modelling, physical, technological and ergonomic properties, and the rules of materials and the environment.

1.5 Continuity of Curves and Surfaces

So far, enough knowledge has been acquired to explain a few examples, referring to the continuity of curves and geometries in engineering graphics (modellers). An accurate description of the continuity of curves and surfaces is important not only for the reasons of how a surface or a curve looks, but also because of the design requirements. We have seen that curves follow certain laws of physics (load, torque, fluid flow, etc.). If our products are to successfully perform their function on the basis of the selected physical conditions, such curves should be selected so that they correspond to the relevant laws of physics. It is of utmost importance to provide adequate transitions between the curves and the surfaces. In connection with transitions, we will come across several terms that can cause considerable confusion among the users of modellers with no prior theoretical knowledge. In this part of the chapter we will come across a variety of definitions, such as *curve curvature*, *parametric continuity* and the *geometric continuity* of curves.

1.5.1 Normal Vector and Curvature

Understanding the definitions of the normal vector and curve curvature is essential for an understanding of curve continuity.

Normal vector: Let us suppose a fixed point $\mathbf{P}(t)$, a variable point \mathbf{T} on its left, and a variable point \mathbf{Q} on its right. All of them are on a parametric curve and they uniquely define a plane. When \mathbf{T} and \mathbf{Q} are approaching $\mathbf{P}(t)$, the plane approaches its final position, where it becomes a tangent plane to the curve at the point $\mathbf{P}(t)$. On the tangent plane at point $\mathbf{P}(t)$ also lies the curve tangent at this point. The tangent plane is the plane that passes through $\mathbf{P}(t)$ and involves both $\mathbf{P}'(t)$ and $\mathbf{P}''(t)$. In general terms, it can be said that the following equation is true for a given point on this plane, where p and q are parameters.

$$P(t) + pP'(t) + qP''(t) \tag{1.53}$$

A *bi-normal vector* $B(t)$ is a unit vector obtained from the vector product $P'(t)$ and $P''(t)$:

$$B(t) = \frac{P'(t) \times P''(t)}{|P'(t) \times P''(t)|} \tag{1.54}$$

The equation above makes it clear that the bi-normal vector $B(t)$ is orthogonal to both $P'(t)$ and $P''(t)$ and consequently also orthogonal to the tangent plane. The line where the points $P(t) + tB(t)$ lie is therefore a *bi-normal line* at point $P(t)$.

A *normal vector* is a vector that is orthogonal to both the tangent and the bi-normal vector. Its direction can be defined by the right-handed system. The unit normal vector $N(t)$ can be defined with the following equation:

$$N(t) = \frac{B(t) \times P'(t)}{|B(t) \times P'(t)|} \tag{1.55}$$

The mutual positions of the tangent, bi-normal and normal vectors at a given point on the curve are shown in Fig. 1.30. It should be pointed out that the tangent, normal and $P''(t)$ vectors all lie on the same plane.

Curvature: The easiest way to explain the notion of the curvature of a curve is with the following interpretation: Let us suppose that $P(t)$ is any fixed point on a curve and the points T and Q are variable points to the left and to the right of $P(t)$ on the curve. These three points uniquely define a circle as long as they are lying on the same line (i.e., they are not collinear). When the points T and Q get close to the point $P(t)$, the circle defined by these three points reaches its final radius r and the centre position S (Fig. 1.31, dotted line). Such a circle is referred to as *the circle of curvature* at point $P(t)$. Consequently, the more the curve is bent—the larger the *curvature* and the smaller the radius of the circle of curvature at the point $P(t)$. Mathematically this is written as:

Fig. 1.30 Mutual positions of the tangent, bi-normal and normal vectors at a given point on the curve

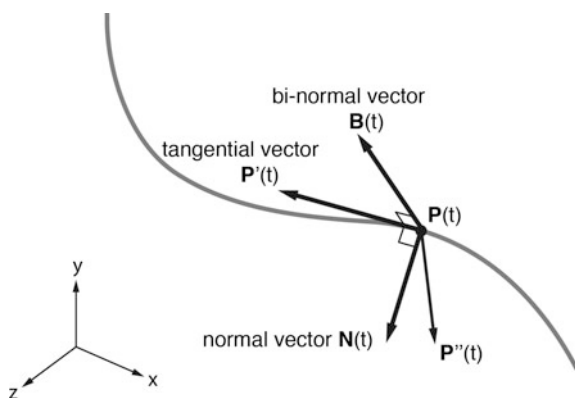
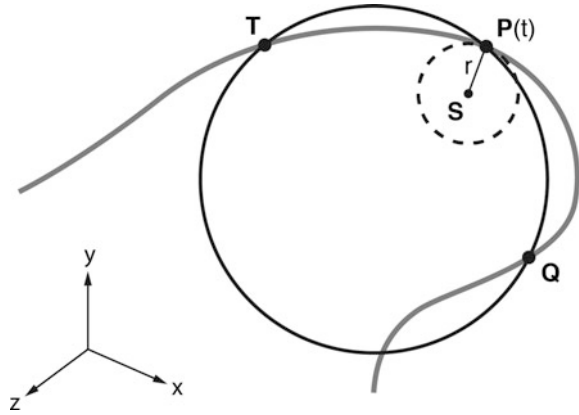


Fig. 1.31 When the points T and Q move close to the point $P(t)$, the circle that is uniquely defined by the points approaches its final size and position, the circle of curvature, defined by its centre S and radius r



$$\kappa_{P(t)} = \frac{1}{r_{P(t)}} \quad (1.56)$$

Analogously, the larger the radius of curvature at the point $P(t)$ the smaller the curvature. Hence, you can imagine a straight line (line segment) as a curve with an infinitely large radius of the circle of curvature.

Since three non-collinear points in space uniquely define a plane in space, the circle of curvature at the point $P(t)$ also lies on this plane. It is also true that this plane is tangential to the curve at the point $P(t)$. As we know from the definition of the tangent plane, the circle of curvature lies on the tangent plane. Because the circle of curvature is tangent to the curve and as such also to the curve tangent, it means that the centre of the circle of curvature is in the direction of the normal to the curve.

Hence, the curvature κ at the parameter t on the curve $P(t)$ can be calculated using the equation:

$$\kappa(t) = \frac{|\mathbf{P}'(t) \times \mathbf{P}''(t)|}{|\mathbf{P}'(t)|^3} \quad (1.57)$$

1.5.2 Parametric Continuity

Parametric continuity refers to parametric curves and describes the *smoothness* of a parameter along the curve.

A curve has the parametric continuity C^n when the n -th derivative of the function describing the curve is continuous along the entire curve.

$$\frac{d^n \mathbf{P}(t)}{dt^n} \quad (1.58)$$

The most characteristic orders of parametric continuity are:

- C^{-1} Curves are discontinuous
- C^0 Curves are continuous—they share common end-start points
- C^1 First derivatives are continuous.
- C^2 First and second derivatives are continuous
- C^n First through n -th derivatives are continuous

Problems with parametric continuity: Parametric continuity C^n generally allows an excellent definition of the continuity of two curves at a contact point, but not in all cases. Let us take an example of the two straight lines in Fig. 1.32, described by the Eq. 1.55, where T_1 , T_2 and T_3 are collinear points in space.

$$f(u) = T_1 + u(T_2 - T_1) \quad (1.59a)$$

$$g(v) = T_2 + v(T_3 - T_2) \quad (1.59b)$$

When a normalized parameter u (and analogously v) travels from 0 to 1 it is obvious that the curves have continuity C^0 (they are in contact at the point T_2). The question arises whether the curves also provide continuity C^1 :

$$f'(u) = T_2 - T_1 \quad (1.60a)$$

$$g'(v) = T_3 - T_2 \quad (1.60b)$$

Consequently, $f'(u) = T_2 - T_1$ generally does not equal $g'(v) = T_3 - T_2$. As a result, these two curves do not provide C^1 continuity at the point T_2 ; however, Fig. 1.32 can raise doubts that this is not true in this case. A solution is possible if the direction vectors $T_1 - T_2$ and $T_2 - T_3$ are replaced by their unit vectors and change the area of the parameters u and v . It results in the following two equations:

$$F(u) = T_1 + u \frac{(T_2 - T_1)}{|T_2 - T_1|} \quad (1.61a)$$

$$G(v) = T_2 + v \frac{(T_3 - T_2)}{|T_3 - T_2|} \quad (1.61b)$$

where the parameter u runs over an interval 0 to $|T_2 - T_1|$ and the parameter v over an interval 0 to $|T_3 - T_2|$. Because $F'(u) = G'(v) = \text{unitvector}$ in the direction

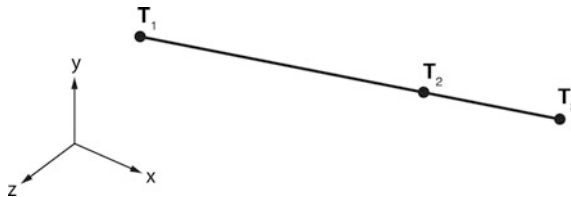


Fig. 1.32 Continuity of two coaxial lines. In order to prove their C^1 and C^2 continuity at the common point, the equations of the lines should be re-parameterised accordingly

from T_1 to T_3 , it means that the curves are C^1 continuous. The process of parameterisation that was used, therefore makes solving this problem possible.

Let us take a look at another example where the parameters u and v (Eq. 1.58) run over an interval $[0, 1]$ as shown in Fig. 1.33.

$$\mathbf{f}(u) = \left(-\cos\left(u^2 \frac{\pi}{2}\right), \sin\left(u^2 \frac{\pi}{2}\right), 0\right) \quad (1.62a)$$

$$\mathbf{g}(v) = \left(\sin\left(v^2 \frac{\pi}{2}\right), \cos\left(v^2 \frac{\pi}{2}\right), 0\right) \quad (1.62b)$$

The function $\mathbf{f}(u)$ over an interval 0 to 1 describes the left part of the circular arc in Fig. 1.33. Similarly, the function $\mathbf{g}(v)$ over an interval from 0 to 1 describes the right part of the circular arc. The curves are in contact at the point $T_1 = (0, 1, 0) = \mathbf{f}(1) = \mathbf{g}(0)$. The calculation follows:

$$\mathbf{f}'(u) = \left(\pi u \sin\left(u^2 \frac{\pi}{2}\right), \pi u \cos\left(u^2 \frac{\pi}{2}\right), 0\right)$$

$$\mathbf{f}''(u) = \left(\pi^2 u^2 \cos\left(u^2 \frac{\pi}{2}\right), -\pi^2 u^2 \sin\left(u^2 \frac{\pi}{2}\right), 0\right)$$

$$\mathbf{f}'(u) \times \mathbf{f}''(u) = (0, 0, -\pi^3 u^3)$$

$$|\mathbf{f}'(u)| = \pi u$$

$$|\mathbf{f}'(u) \times \mathbf{f}''(u)| = \pi^3 u^3$$

$$\kappa(u) = 1$$

$$\mathbf{g}'(v) = \left(\pi v \cos\left(v^2 \frac{\pi}{2}\right), -\pi v \sin\left(v^2 \frac{\pi}{2}\right), 0\right)$$

$$\mathbf{g}''(v) = \left(-\pi^2 v^2 \sin\left(v^2 \frac{\pi}{2}\right), -\pi^2 v^2 \cos\left(v^2 \frac{\pi}{2}\right), 0\right)$$

$$\mathbf{g}'(v) \times \mathbf{g}''(v) = (0, 0, -\pi^3 v^3)$$

$$|\mathbf{g}'(v)| = \pi v$$

$$|\mathbf{g}'(v) \times \mathbf{g}''(v)| = \pi^3 v^3$$

$$\kappa(v) = 1$$

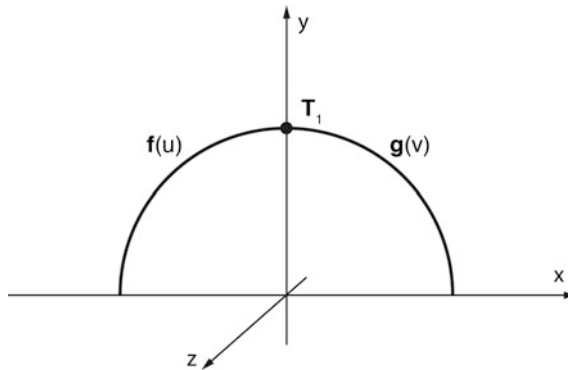


Fig. 1.33 Continuity of two sections of a circular arc

It should be noted here that both $\mathbf{g}'(0)$ and $\mathbf{g}''(0)$ are zero vectors, and as such their direction at this point is not clearly defined. Consequently, it is not possible to make conclusions about the curve's continuity properties at the point of contact, although Fig. 1.33 makes it seem that there is some continuity between both curves at the point of contact, and that they obviously have a common tangent.

Let us try to solve the problem again by means of parameterisation, i.e., by modifying their parametric equations, without changing their forms, the same as we did in the previous case. In the equation $\mathbf{f}(u)$ replace $u^2 = p$, and in the equation $\mathbf{g}(v)$ replace $v^2 = q$. This results in:

$$\begin{aligned}\mathbf{f}(p) &= \left(-\cos\left(p\frac{\pi}{2}\right), \sin\left(p\frac{\pi}{2}\right), 0\right) \\ \mathbf{g}(q) &= \left(\sin\left(q\frac{\pi}{2}\right), \cos\left(q\frac{\pi}{2}\right), 0\right)\end{aligned}$$

Their derivatives are as follows:

$$\begin{aligned}\mathbf{f}'(p) &= \left(\frac{\pi}{2}\sin\left(p\frac{\pi}{2}\right), \frac{\pi}{2}\cos\left(p\frac{\pi}{2}\right), 0\right) \\ \mathbf{f}''(p) &= \left(\left(\frac{\pi}{2}\right)^2\cos\left(p\frac{\pi}{2}\right), -\left(\frac{\pi}{2}\right)^2\sin\left(p\frac{\pi}{2}\right), 0\right) \\ \mathbf{g}'(q) &= \left(\frac{\pi}{2}\cos\left(q\frac{\pi}{2}\right), -\frac{\pi}{2}\sin\left(q\frac{\pi}{2}\right), 0\right) \\ \mathbf{g}''(q) &= \left(-\left(\frac{\pi}{2}\right)^2\sin\left(q\frac{\pi}{2}\right), -\left(\frac{\pi}{2}\right)^2\cos\left(q\frac{\pi}{2}\right), 0\right) \\ \mathbf{f}'(p) \times \mathbf{f}''(p) &= \mathbf{g}'(q) \times \mathbf{g}''(q) = \left(0, 0, -\left(\frac{\pi}{2}\right)^3\right) \\ |\mathbf{f}'(p) \times \mathbf{f}''(p)| &= |\mathbf{g}'(q) \times \mathbf{g}''(q)| = \left(\frac{\pi}{2}\right)^3 \\ |\mathbf{f}'(p)| &= |\mathbf{g}'(q)| = \frac{\pi}{2} \\ \kappa(p) &= \kappa(q) = 1\end{aligned}$$

After changing the variables, the following is true $\mathbf{f}'(1) = \mathbf{g}'(0) = (\pi/2, 0, 0)$. And so it can be said that the curves have continuity C^1 . Let us take a look at their other derivatives: $\mathbf{f}''(1) = \mathbf{g}''(0) = (0, -(\pi/2)^2, 0)$, which means that they also provide continuity C^2 . What is more, due to the continuity along the entire curve 1, it is possible to claim that the curves also have a continuous curvature (which makes sense as they are part of a circle).

These two examples have shown that the choice of parameterisation (re-parameterisation or changing the variables) has a significant effect on the continuity of curves. Consequently, a mathematical determination of the continuity can

always throw the calculated results into doubt. The solution, shown in previous subchapters, is to use the arc length.

Let a curve have an arc length s . The curve can be so parameterised that the point $f(u)$ is on the curve that is separated by the arc length u from the initial point of the curve $f(0)$, with u lying in the interval from 0 to s . By means of parameterisation using the arc length, where u travels from 0 to s , $f(u)$ travels along the curve from $f(0)$ to $f(s)$ at the same ‘speed’. As a result, the tangent vector measuring the speed is of unit-length. Many equations can be simplified in this manner.

Despite re-parameterisation by the arc length (an alternative term is *natural parameterisation*) looking, at least in theory, a simple and elegant method, it is often demanding and unpractical because defining arc lengths requires the integration of functions and the use of square roots.

1.5.3 Geometric Continuity

It has been shown that in some cases it is difficult to determine the parametric continuity of curves. Many of the C^1 continuous curves have continuous curvature, but lack C^2 continuity at the point of contact, and some of them are not double differentiable. However, these curves appear *smooth* at the points of contact and also at the transitions from one section to another. After changing a variable, some of these curves even become C^2 continuous at the points of contact, as shown earlier in this chapter. The problem is that it is sometimes difficult to find an appropriate re-parameterisation procedure that makes this possible. This was the reason for defining the *geometric continuity*, as it slightly relaxes the requirements for C^2 :

Two sections of naturally parameterised curves are G^k *geometric continuous* at the joining point if and only if all the left and right i -th derivatives match for each $i \leq k$.

Although the above definition requires parameterisation by the arc length, the one below, which is an equivalent definition, does not require it.

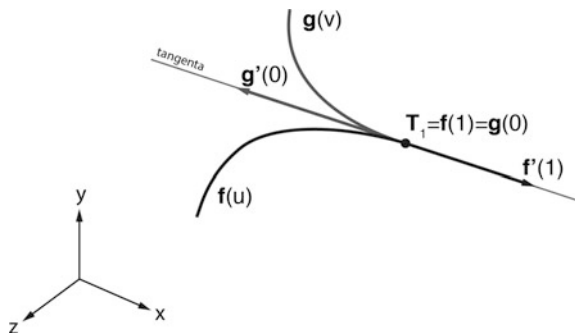
Two sections of curves are G^k *geometric continuous* at the joining point if and only if two parameterisations exist, one for each curve segment that all the left and right i -th derivatives match for each $i \leq k$.

This definition is already better, but still not good enough, as we often do not know how to find an appropriate parameterisation. Fortunately, designing and modelling is mainly only about the cases when $k = 1$ and $k = 2$.

Let us first define the continuity G^1 :

Two G^0 continuous curve sections are G^1 *geometric continuous* if and only if the vectors $f'(u)$ and $g'(v)$ have the same direction at the joining point, and $f'(u)$ and $g'(v)$ are evaluated at the joining point.

Fig. 1.34 Curves with a common tangent, but tangent vectors in the opposite direction, which makes the curve not G^1 continuous



Let us point out again here that it is not enough if the curve sections share the same tangent at their common point; their tangent vectors also need to have the same direction. To be more precise, two curve sections sharing a common tangent are not yet G^1 continuous at a common point. For a better understanding, let us take a look at Fig. 1.34.

In Fig. 1.34 the curves $f(u)$ and $g(v)$ are in contact at the common point $f(1) = g(0)$ and share a common tangent. However, the tangent vectors of both curves point to the opposite direction and, consequently, the curves are not G^1 continuous at this point. In contrast to that, the examples in Fig. 1.32 and 1.33—according to the same definition—exhibit G^1 continuity at the points of contact, despite the initial calculations not showing C^1 continuity.

Gregory Nielson gave a simple definition of G^2 continuity:

Two C^1 continuous curve sections are G^2 geometric continuous at the joining point if and only if the vector $f''(u) - g''(v)$ is parallel to the tangent vector at the joining point, and $f''(u)$ and $g''(v)$ are evaluated at the point of contact.

An advantage of such a definition of G^2 continuity is its independence from the selected parameterisation; it is only necessary to verify the C^1 continuity in advance.

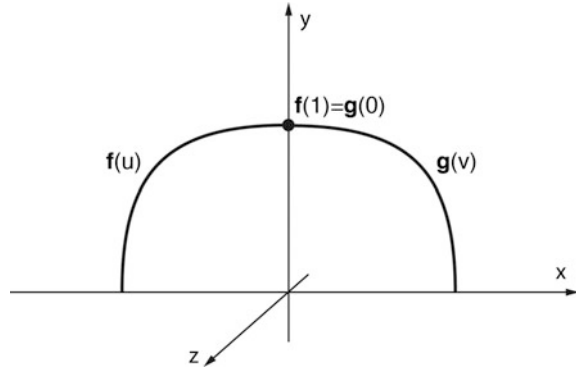
Let us take a look at another example of Nielson's definition: Let us take two parabolas with a joining point at $(0, 1, 0)$. The parabolas are described by the following equations as shown on Fig. 1.35:

$$f(u) = (-1 + u^2, 2u - u^2, 0)$$

$$g(v) = (2u - u^2, 1 - u^2, 0)$$

Both curves are within the normalized interval $[0, 1]$, while the joining point is $f(1) = g(0) = (0, 1, 0)$.

Fig. 1.35 Continuity of two parabolas with joining point at $(0, 1, 0)$



The basic calculations follow:

$$\begin{aligned}
 \mathbf{f}'(u) &= (2u, 2 - 2u, 0) \\
 \mathbf{f}''(u) &= (2, -2, 0) \\
 \mathbf{f}'(u) \times \mathbf{f}''(u) &= (0, 0, -4) \\
 |\mathbf{f}'(u)| &= 2\sqrt{1 - 2u + 2u^2} \\
 |\mathbf{f}'(u) \times \mathbf{f}''(u)| &= 4 \\
 \kappa(u) &= \frac{1}{2(1 - 2u + 2u^2)^{1.5}} \\
 \mathbf{g}'(v) &= (2 - 2v, -2v, 0) \\
 \mathbf{g}''(v) &= (-2, -2, 0) \\
 \mathbf{g}'(v) \times \mathbf{g}''(v) &= (0, 0, -4) \\
 |\mathbf{g}'(v)| &= 2\sqrt{1 - 2v + 2v^2} \\
 |\mathbf{g}'(v) \times \mathbf{g}''(v)| &= 4 \\
 \kappa(v) &= \frac{1}{2(1 - 2v + 2v^2)^{1.5}}
 \end{aligned}$$

Calculating $\mathbf{f}'(1) = \mathbf{g}'(0) = (2, 0, 0)$ shows that the curves are C^1 continuous at the joining point. Since $\mathbf{f}''(1) = (2, -2, 0)$ is not equal to $\mathbf{g}''(0) = (-2, -2, 0)$, they are not C^2 continuous; however, as the values of both curves' functions of curvature at the point of contact are identical $k_f(u) = k_g(v) = 1$, we can see that they have continuous curvature at the said point.

Let us verify now G^2 : with C^1 continuity at the joining point having been confirmed, we can move on. Let us first calculate $\mathbf{f}''(1) - \mathbf{g}''(0) = (4, 0, 0)$. The tangent vector can be calculated from $\mathbf{f}'(1) = (2, 0, 0)$. We can see that both vectors are parallel at the joining point and referring to Nielson's definition the said curves are G^2 continuous at this point.

According to the definition of the G^k order of continuity, explained above, it would be possible to find two such parameterisations of both curves for which C^2 continuity could be proved.

Due to all the above-described characteristics, geometrical continuity is very suitable for defining the continuity of implicit functions, such as conic sections (i.e., the curves generated when a plane cuts through a cone: a circle, an ellipse, a parabola, a hyperbola), and the continuity of other curve shapes that cannot be or are difficult to differentiate several times, such as a line (which can also be referred to as a circle with an indefinite radius).

To sum up some of the most characteristic orders of the geometric continuity of curves that we come across in design and modelling:

- G^0 curves meet at the point of contact
- G^1 curves have the same tangent direction at the point of contact
- G^2 curves have the same tangent direction and centre of curvature at the point of contact.

1.5.4 Practical Aspect of Continuity for Designing and Modelling

It should be clear by now that continuous, *smooth* transitions require at least G^1 continuity. For practical, physical and aesthetic reasons—for example, in aeronautical, car and nautical industries, and elsewhere—higher-order continuities should be provided. General command settings for edge blending between different CAD model surfaces, e.g., fillets, provide G^1 continuity. In order to have G^2 continuity, it usually needs to be explicitly specified in the settings. G^2 transitions can be edited later and they are usually based on cubic curves, explained earlier in this chapter.

Visual control can be of great assistance in the CAD modelling of suitable transitions. The surface of a car body, for example, will not look smooth without having at least G^2 continuous transitions. This makes the visual control of surface transitions in modellers an important part of providing appropriate continuity of transitions, and as such, it is supported in modellers by a variety of visualisation tools for a better impression. Throughout this book, the reader will be made familiar with these tools. The most typical ones are shown in Fig. 1.36. For curve curvature control, *curvature combs* are used. By means of line segments, orthogonal to a curve, they visualize the relative value of the curvature k at individual points on the curve. Also by means of visualisation tools, it is possible to define the position of the inflection point, as well as the point on a curve with the smallest circle of curvature.

Visual inspection of surface continuity of CAD objects can be performed in the first step by inspecting the model itself, especially its edges and the transitions

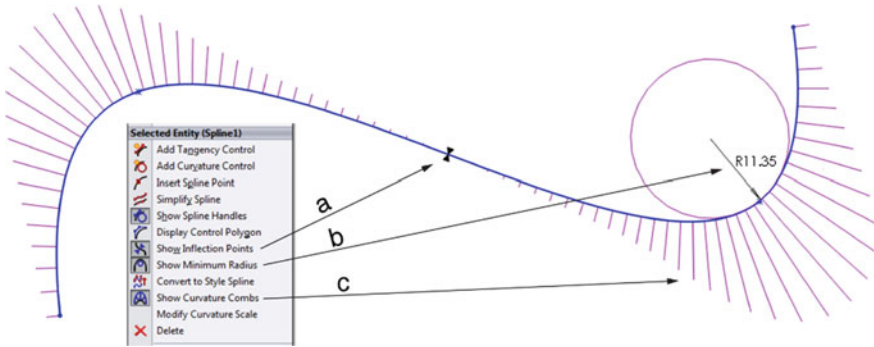


Fig. 1.36 Visualisation tools for curve curvature and curve continuity control in the SolidWorks modeller: **a** showing the inflection point, **b** position and size of the largest curvature, **c** *curvature combs*

between surfaces. Creating high-resolution renders can also be of some assistance. Figure 1.37 shows a model with three different transitions between three surfaces. On the left you can see a model where only surfaces meet, i.e., at a G^0 continuity. At the centre there is a model with a constant radius blending, i.e., at a G^1 continuity between the blending and the nearby upper surfaces. On the right is a model where the transition between two surfaces is achieved by the cubic function that allows G^2 curvature continuity between two surfaces. Although a screen image in the modeller displays almost no difference between the models at the centre and on the right, the difference—especially with objects from nature with smooth reflecting surfaces—can be very obvious. For this reason, advanced modellers feature tools that simulate reflections of different, usually zebra patterns from the environment. Figure 1.37 shows different reflections of the same zebra pattern for different transitions. Zebra stripes on the surfaces of G^0 transitions do not meet at the contact between two surfaces. In G^1 transitions the stripes meet; however, they form sharp transitions. Only G^2 transitions provide *smooth* transitions from one surface to the other. This tool allows an excellent overview of surface continuity on the edges; however, it provides no information about the size of curvature at any point on the surface. When such information is required, the tool for colour mapping of the radius of curvature on the surface should be applied. Using a colour scale, it colours the CAD model's surfaces according to their radius of curvature at a given point. Figure 1.37 shows that flat surfaces are coloured black, whereas other parts are coloured according to their respective curvatures. Hence, constant radius blendings (centre below) are coloured evenly with one colour, whereas the blendings with continuous curvature are shading from the colour of the first adjacent surface via a variety of green intensities into the colour of the next surface. Besides pattern- and colour-mapping of the curvature, modern modellers also allow collecting information on the curvature κ and the radius of curvature r at any given discrete point on the surface.

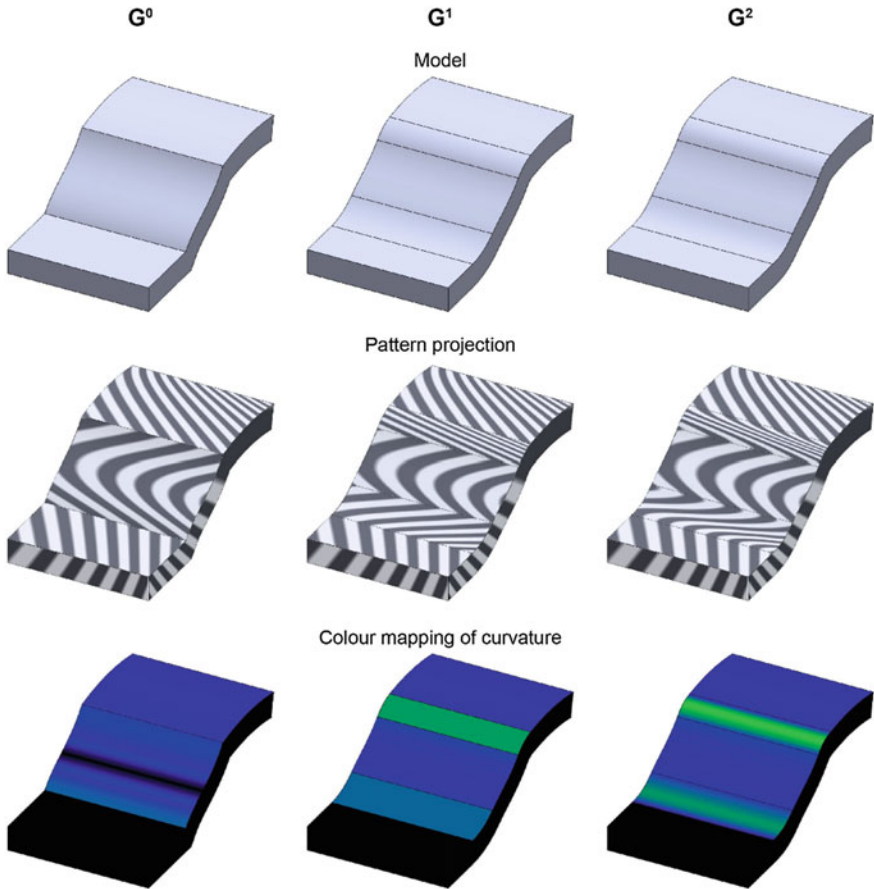


Fig. 1.37 CAD model with G^0 , G^1 and G^2 transitions between a model's upper surfaces. Transitions can be analysed by means of projecting patterns or colour mapping the surfaces

1.5.5 Surface Classes

Any engineer or experienced CAD modeller user will sooner or later come across the terms *Class A*, *Class B* and *Class C* surfaces. The term originated in the car industry and different car manufacturers' internal standards. Consequently, there is no standard definition of the classes; instead, there are several different interpretations.

In the broadest sense, the definition says:

- **Class A:** Visible surfaces of a product, requiring the highest aesthetic level, such as a car bonnet, headlight glass, etc.

- **Class B:** Visible surfaces of a product, not requiring the highest aesthetic level, such as car wheel nut holes, door openings, etc.
- **Class C:** Non-visible surfaces and parts of a product, such as a car seat mounting, spare wheel space, etc.

This allows industrial and mechanical designers to define what surfaces need to remain unchanged during detailed design and which ones can be adjusted in order for the product to achieve its full functionality and visual appeal. This is particularly important when a parent company leaves a product or a subassembly to a sub-contractor for final development and manufacturing. Companies can of course lay out additional details and define, by prescribed tolerances, various surface classes, so that they are in accordance with their aesthetic, design and technological standards.

Because of the inconsistent classification of surface classes it is recommended to use the terms *Class A*, *Class B* and *Class C* only for communication within a company and under the condition that everyone involved is familiar with the internal standardization of surfaces inside the particular company. When outsourcing development, it is recommended to set out in advance the common rules that apply to individual surfaces.

For general use, it is strongly discouraged to use surface classes. Instead, it is recommended to define surfaces by referring to parametric C^k or geometric G^k continuities, as they are mathematically accurately defined.

References

1. Duhovnik J, Kljajin M, Opalić M (2009) Inženirska grafika, Fakulteta za strojništvo, Ljubljana
2. Duhovnik J, Demšar I, Drešar P (2015) Space modeling with SolidWorks and NX. Springer, Heidelberg. <https://doi.org/10.1007/978-3-319-03862-9>
3. Watt A (1990) Fundamentals of three-dimensionals computer graphics. Addison Wesley, Boston
4. Farin GE (2002) Curves and surfaces for computer-aided geometric design. Academic Press, Cambridge
5. Jankauskas K (2010) Time-efficient nurbs curve evaluation algorithms. In: Proceedings of the 16th international conference on Information and Software Technologies, pp 60–69
6. Patrikalakis NM, Maekawa T (2002) Intersection problems. In: Farin GE, Kim M-S (eds) Handbook of computer aided geometric design. Elsevier, New York. <https://doi.org/10.1016/b978-044451104-1/50026-5>
7. Bu-Qing S, Ding-Yuan L (1989) Computational geometry: curve and surface modeling. Academic Press, Cambridge
8. Petrišič J (1999) Interpolacija. Fakulteta za strojništvo, Ljubljana