CHAPTER

EIGHT

Construction of Common Surfaces

8.1 Introduction

Sections 2 through 5 of this chapter cover the NURBS representation of the four surface types: bilinear surfaces, general cylinders (extruded surfaces), ruled surfaces, and surfaces of revolution. As the reader surely realizes by now, these representations are not unique. Considerations such as parameterization, convex hull, continuity, software design, application area, data exchange, and even personal taste can influence the choice of representation method. Our presentation closely follows the constructions given by Piegl and Tiller [Pieg87a]. Section 6 uses nonuniform scaling to obtain additional surfaces such as the ellipsoid and elliptic paraboloid. Section 7 presents a method for constructing a three-sided patch on a sphere, whose boundary curves are circles whose radii are equal to the radius of the sphere. This patch is useful as a corner fillet surface.

In order to clearly indicate the degree in this chapter, we use the notation $R_{i,p;j,q}(u,v)$ for the i,jth rational basis function of degree $p \times q$ (compare this with Eq. [4.12]).

8.2 Bilinear Surfaces

Let $\mathbf{P}_{0,0}, \mathbf{P}_{1,0}, \mathbf{P}_{0,1}, \mathbf{P}_{1,1}$ be four points in three-dimensional space. We want to construct a NURBS representation of the surface obtained by bilinearly interpolating between the four line segments, $\mathbf{P}_{0,0}\mathbf{P}_{1,0}$, $\mathbf{P}_{0,1}\mathbf{P}_{1,1}$, $\mathbf{P}_{0,0}\mathbf{P}_{0,1}$, and $\mathbf{P}_{1,0}\mathbf{P}_{1,1}$. Clearly, the desired (nonrational) surface is given by

$$\mathbf{S}(u,v) = \sum_{i=0}^{1}\sum_{j=0}^{1} N_{i,1}(u) N_{j,1}(v) \mathbf{P}_{i,j} \tag{8.1}$$

with

$$U = V = \{0,0,1,1\}$$

Equation (8.1) represents a simple linear interpolation between the opposite boundary lines in each of the two directions. In Figure 8.1a the four points lie in a common plane, hence the surface is that portion of the plane bounded by the four line segments. Figure 8.1b shows a hyperbolic paraboloid obtained by linear interpolations between the nonparallel diagonals of a cube.

8.3 The General Cylinder

Let $\mathbf{W}$ be a vector of unit length and $\mathbf{C}(u) = \sum_{i=0}^{n} R_{i,p}(u)\mathbf{P}_i$ be a pth-degree NURBS curve on the knot vector, U, with weights w_i. We want the representation of the general cylinder, $\mathbf{S}(u,v)$, obtained by sweeping $\mathbf{C}(u)$ a distance d along $\mathbf{W}$. Denoting the parameter for the sweep direction by v, $0 \le v \le 1$, clearly $\mathbf{S}(u,v)$ must satisfy two conditions

- for fixed $\bar{u}$, $\mathbf{S}(\bar{u},v)$ is a straight line segment from $\mathbf{C}(\bar{u})$ to $\mathbf{C}(\bar{u}) + d\mathbf{W}$;
- for fixed $\bar{v}$

$$\mathbf{S}(u,\bar{v}) = \mathbf{C}(u) + \bar{v}d\mathbf{W} = \sum_{i=0}^{n} R_{i,p}(\mathbf{P}_i + \bar{v}d\mathbf{W})$$

(the translational invariance property).

The desired representation is

$$\mathbf{S}(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{1} R_{i,p;j,1}(u,v)\mathbf{P}_{i,j} \tag{8.2}$$

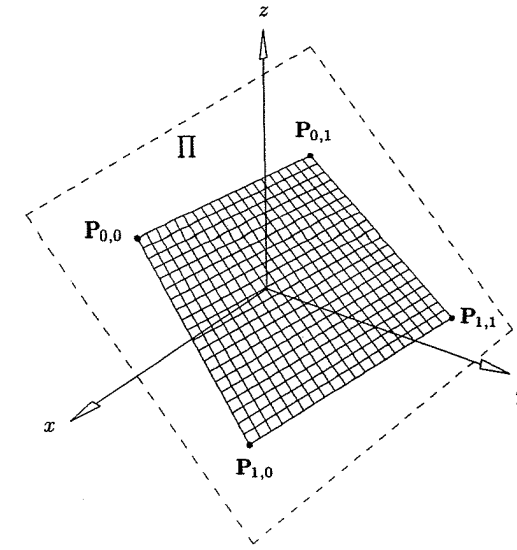
on knot vectors U and V, where $V = \{0,0,1,1\}$ and U is the knot vector of $\mathbf{C}(u)$. The control points are given by $\mathbf{P}_{i,0} = \mathbf{P}_i$ and $\mathbf{P}_{i,1} = \mathbf{P}_i + d\mathbf{W}$, and the weights are $w_{i,0} = w_{i,1} = w_i$. A general cylinder is shown in Figure 8.2.

A right circular cylinder is obtained by translating the NURBS circle a distance d along a vector normal to the plane of the circle. Any of the circle representations given in Chapter 7 will do. Figure 8.3 shows a right circular cylinder whose underlying circle is the quadratic, nine-point, square control polygon representation of Example `Ex7.2` and Figure 7.16a. The cylinder is defined by

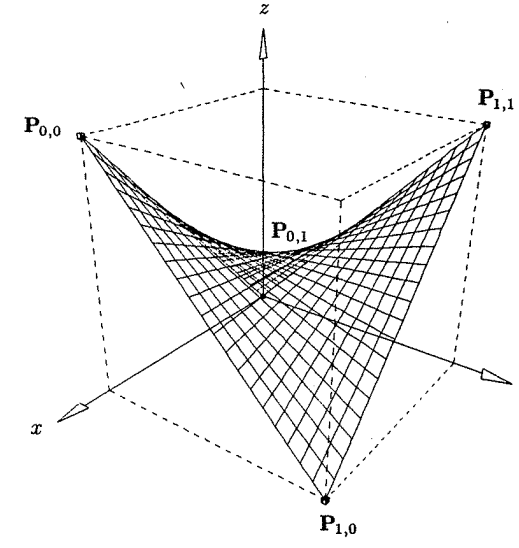
$$\mathbf{S}(u,v) = \sum_{i=0}^{8}\sum_{j=0}^{1} R_{i,2;j,1}(u,v)\mathbf{P}_{i,j} \tag{8.3}$$

where $V = \{0,0,1,1\}$ and U and the weights ($w_{i,0}$ and $w_{i,1}$) are those given for the circle in Example `Ex7.2`. Note that the control net forms the circumscribing box of the cylinder.

Cylinders with circular and conic cross sections of sweep angle less than 360° are obtained by translating the control points computed by Algorithms `A7.1` and `A7.3`. Figure 8.4 shows an elliptic cylinder.



(a)



(b)

Figure 8.1. Bilinear surfaces. (a) Straight line boundaries lying in the plane Π; (b) nonplanar straight line boundaries (defined by diagonals of a cube) yielding a hyperbolic paraboloid.

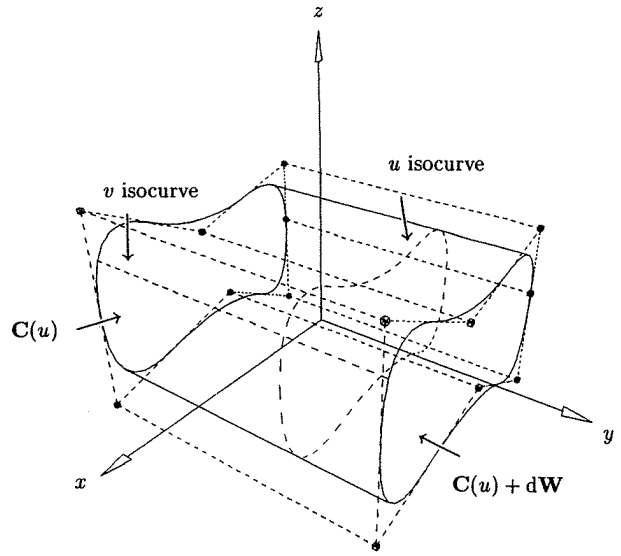


Figure 8.2. A general cylinder obtained by translating $\mathbf{C}(u)$ a distance d along $\mathbf{W}$.

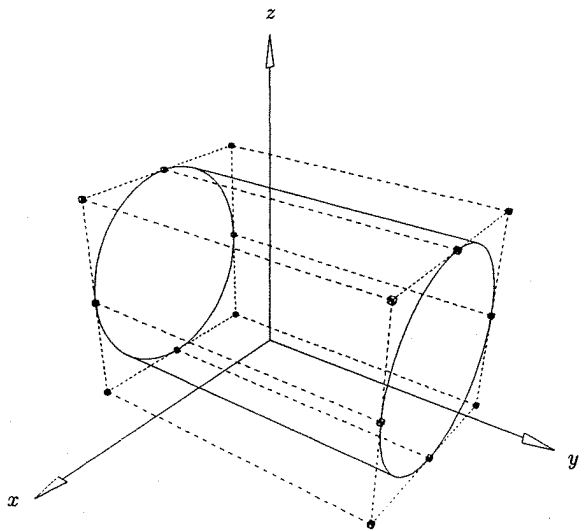


Figure 8.3. A right circular cylinder.

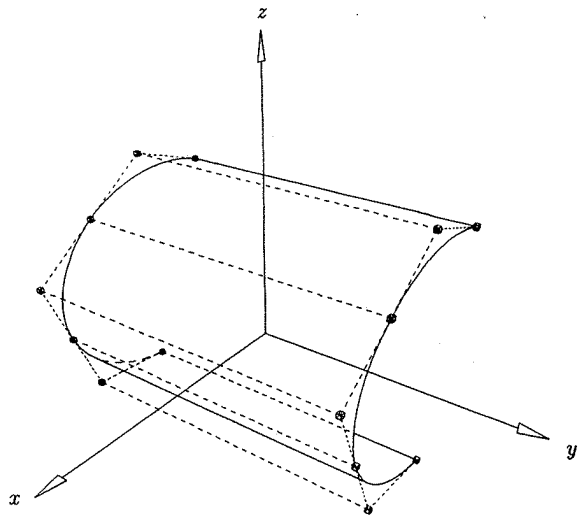


Figure 8.4. An elliptic cylindrical patch.

8.4 The Ruled Surface

Assume we have two NURBS curves

$$\mathbf{C}_k(u) = \sum_{i=0}^{n_k} R_{i,p_k}(u)\mathbf{P}_i^k \qquad k = 1,2$$

defined on knot vectors $U^k = \{u_0^k, \ldots, u_{m_k}^k\}$. We want a surface which is ruled in the v direction, i.e., it is a linear interpolation between $\mathbf{C}_1(u)$ and $\mathbf{C}_2(u)$. Furthermore, we require that the interpolation be between points of equal parameter value, i.e., for fixed $\bar{u}$, $\mathbf{S}(\bar{u}, v)$ is a straight line segment connecting the points $\mathbf{C}_1(\bar{u})$ and $\mathbf{C}_2(\bar{u})$. The desired surface form is

$$\mathbf{S}(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{1} R_{i,p;j,1}(u,v)\mathbf{P}_{i,j} \tag{8.4}$$

where $V = \{0,0,1,1\}$. n, U, p, $w_{i,j}$, and $\mathbf{P}_{i,j}$ must all be determined.

Because of the tensor product nature of the surface, the two boundary curves, $\mathbf{C}_k(u)$, must have the same degree and be defined on the same knot vector. Hence, the algorithm for determining n, U, p, $w_{i,j}$ and $\mathbf{P}_{i,j}$ is

1. ensure that the two curves are defined on the same parameter range, i.e., $u_0^1 = u_0^2$ and $u_{m_1}^1 = u_{m_2}^2$;

2. if $p_1 = p_2$, then set $p = p_1$; otherwise, set p to the larger of p_1, p_2, and raise the degree of the lower degree curve to p (Algorithm `A5.9`);
3. if the knot vectors U^1 and U^2 are not identical, merge them to obtain the knot vector U; more precisely, u_j is in U if it is in either U^1 or U^2. The maximum multiplicity of u_j in U^1 or U^2 is also carried over to U. For example, if $U^1 = \{0,0,0,1,2,2,4,4,4\}$ and $U^2 = \{0,0,0,1,2,3,4,4,4\}$, then $U = \{0,0,0,1,2,2,3,4,4,4\}$;
4. using U, apply knot refinement (Algorithm `A5.4`) to both curves, $\mathbf{C}_k(u)$; this step yields the final values of n, $w_{i,j}$, and $\mathbf{P}_{i,j}$.

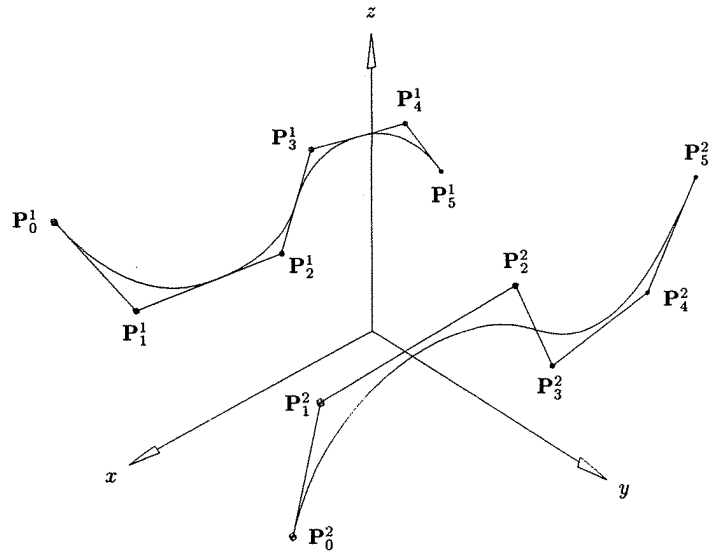
Figures 8.5a–8.5e show the process of constructing a ruled surface between a degree two and a degree three curve.

The general cone is also a ruled surface. Let $\mathbf{P}$ be the vertex point of the cone, and let

$$\mathbf{C}_1(u) = \sum_{i=0}^{n} R_{i,p}(u)\mathbf{P}_i^1$$

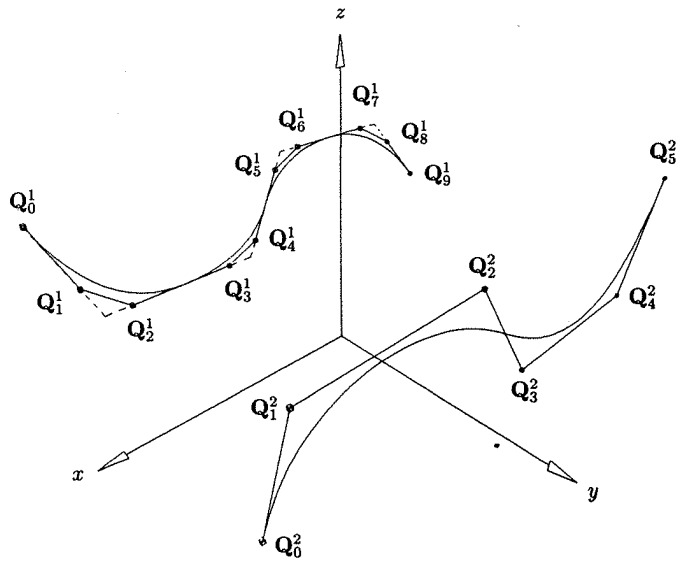
be its base curve. Define the curve

$$\mathbf{C}_2(u) = \sum_{i=0}^{n} R_{i,p}(u)\mathbf{P}_i^2$$

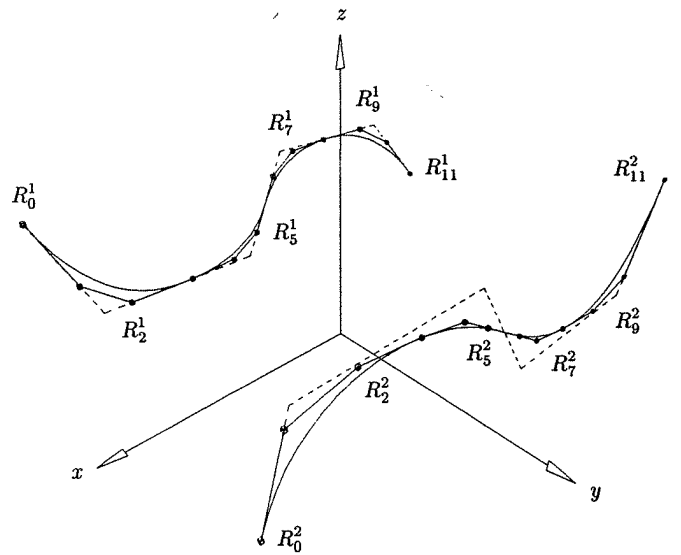


(a)

Figure 8.5. Construction of a ruled surface between two NURBS curves. (a) Input degree two and degree three curves, $U^1 = \{0,0,0,\frac{1}{4},\frac{1}{2},\frac{3}{4},1,1,1\}$ and $U^2 = \{0,0,0,0,\frac{3}{10},\frac{7}{10},1,1,1,1\}$.

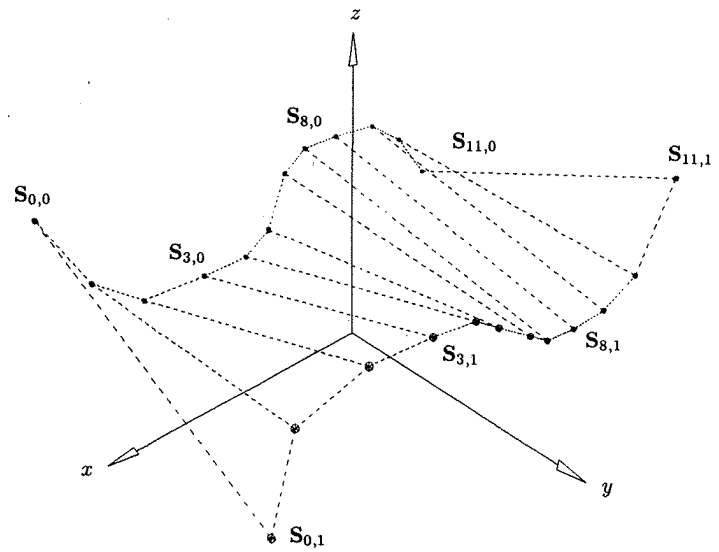


(b)



(c)

Figure 8.5. (*Continued.*) (b) degree elevation of degree two curve; (c) knot refined curves after merging the knot vectors; surface knot vector is $U = \{0,0,0,0,\frac{1}{4},\frac{1}{4},\frac{3}{10},\frac{1}{2},\frac{1}{2},\frac{7}{10},\frac{3}{4},\frac{3}{4},1,1,1,1\}$.



(d)

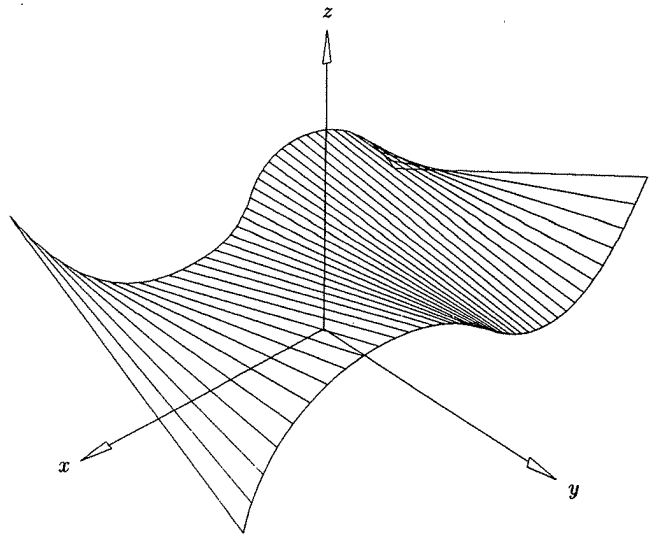
Figure 8.5. (*Continued.*) (d) control net of ruled surface.

with $\mathbf{P}_i^2 = \mathbf{P}$ and $w_i^2 = w_i^1$ for all i (a degenerate curve). The ruled surface between $\mathbf{C}_1(u)$ and $\mathbf{C}_2(u)$ is the desired cone. Figure 8.6 shows a right circular cone, and Figures 8.7a–8.7c show some exotic ruled surfaces defined by lines and circles.

We emphasize again that the rulings (straight line segments) on the NURBS surface are between points of equal parameter value, not points of equal relative arc length. In general, ruling according to relative arc length yields a geometrically different surface, and this cannot be achieved using NURBS. As a consequence, mathematically precise conversion between NURBS and an IGES ruled surface (Type 118), Form 0 [IGE93] is not possible.

8.5 The Surface of Revolution

Let $\mathbf{C}(v) = \sum_{j=0}^{m} R_{j,q}(v)\mathbf{P}_j$ be a qth-degree NURBS curve on the knot vector V. $\mathbf{C}(v)$ is called the *generatrix*, and it is to be revolved about an axis. At the end of this section we present an algorithm which can revolve $\mathbf{C}(v)$ through an arbitrary angle about an arbitrary axis. However, for the sake of simplicity let us now assume that $\mathbf{C}(v)$ lies in the xz plane, and that we revolve $\mathbf{C}(v)$ a full 360° about the z-axis. The required surface, $\mathbf{S}(u, v)$, has characteristics such that (see Figure 8.8):



(e)

Figure 8.5. (*Continued.*) (e) ruled surface.

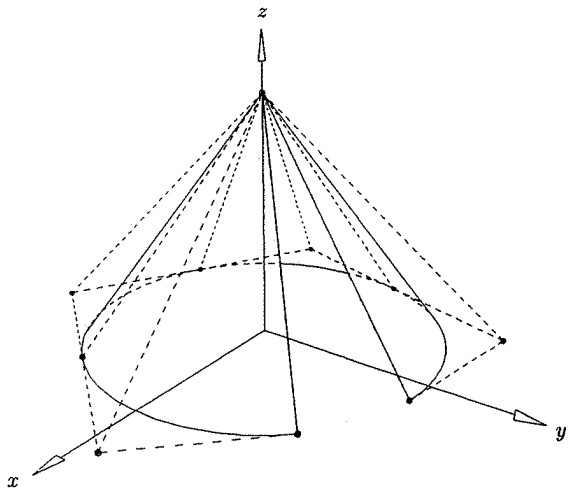
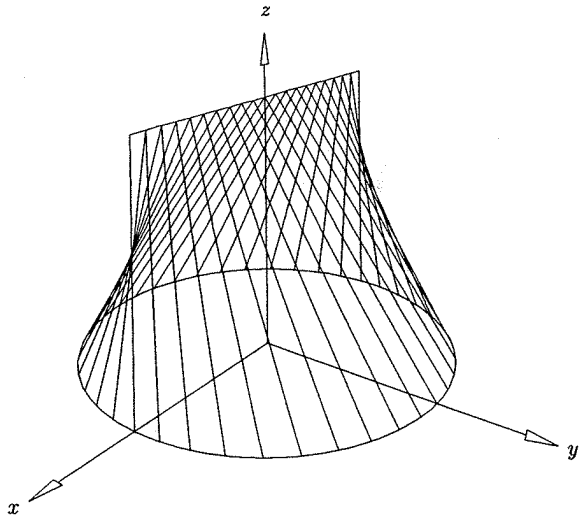
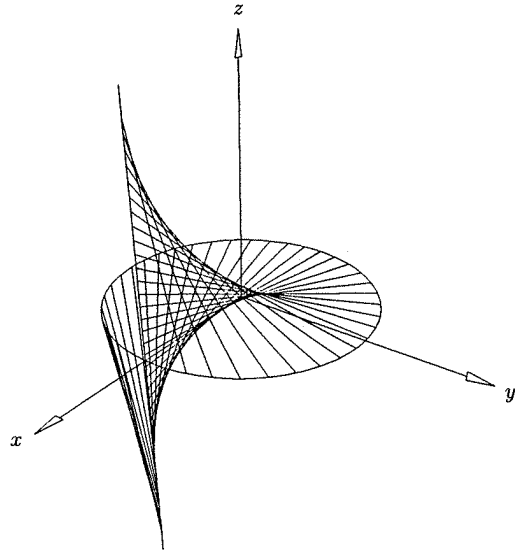


Figure 8.6. A right circular conical patch.

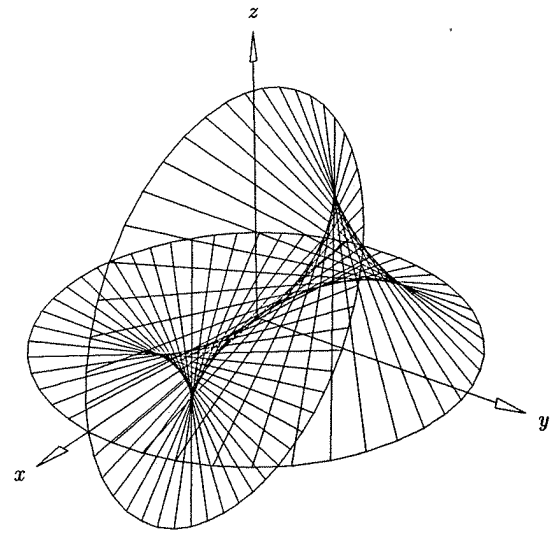


(a)



(b)

Figure 8.7. Ruled surfaces defined by circles and lines. (a) Degree two ruled surface (conoid); (b) degree three ruled surface.



(c)

Figure 8.7. (*Continued.*) (c) degree four ruled surface.

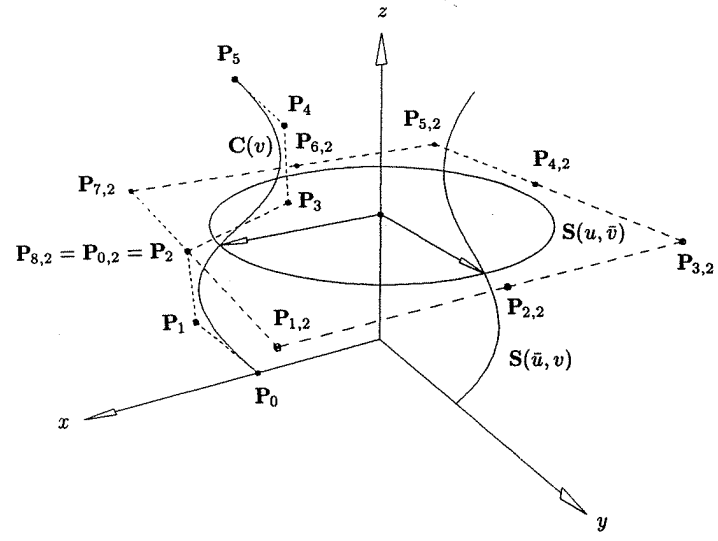


Figure 8.8. Definition of a surface of revolution.

- for fixed $\bar{u}$, $\mathbf{S}(\bar{u}, v)$ is the curve $\mathbf{C}(v)$ rotated some angle about the z-axis;
- for fixed $\bar{v}$, $\mathbf{S}(u, \bar{v})$ is a full circle which lies in a plane perpendicular to the z-axis, with its center on the z-axis.

Let us use the nine-point circle representation, with $U = \{0, 0, 0, 1/4, 1/4, 1/2, 1/2, 3/4, 3/4, 1, 1, 1\}$ and weights $w_i = \{1, \sqrt{2}/2, 1, \sqrt{2}/2, 1, \sqrt{2}/2, 1, \sqrt{2}/2, 1\}$. Then the required surface has the form

$$\mathbf{S}(u, v) = \sum_{i=0}^{8} \sum_{j=0}^{m} R_{i,2;j,q}(u, v) \mathbf{P}_{i,j} \tag{8.5}$$

The knot vectors are U and V. The control points and weights are determined as follows (see Figure 8.8): For $i = 0$, $\mathbf{P}_{i,j} = \mathbf{P}_{0,j} = \mathbf{P}_j$. Because of the circular nature of $\mathbf{S}(u, \bar{v})$, the $\mathbf{P}_{i,j}$ for fixed j, $0 \le i \le 8$, all lie in the plane $z = z_j$. They lie on the square of width $2x_j$, with center on the z-axis. The weights are defined by taking the products of the w_j and the circle weights, w_i, i.e., for fixed j, $w_{0,j} = w_j$, $w_{1,j} = (\sqrt{2}/2)w_j$, $w_{2,j} = w_j$, $w_{3,j} = (\sqrt{2}/2)w_j, \ldots, w_{8,j} = w_j$. Figures 8.9a and 8.9b show the control net and the corresponding surface of revolution generated by Eq. (8.5) and the curve of Figure 8.8.

It is not entirely obvious that Eq. (8.5) represents the desired surface of revolution, hence we sketch a proof here. Fixing $u = \bar{u}$ and using the unprojected form of Eq. (8.5), we write

$$\mathbf{S}^w(\bar{u}, v) = \sum_{j=0}^{m} N_{j,q}(v) \left(\sum_{i=0}^{8} N_{i,2}(\bar{u}) \mathbf{P}^w_{i,j} \right) = \sum_{j=0}^{m} N_{j,q}(v) \mathbf{Q}^w_j$$

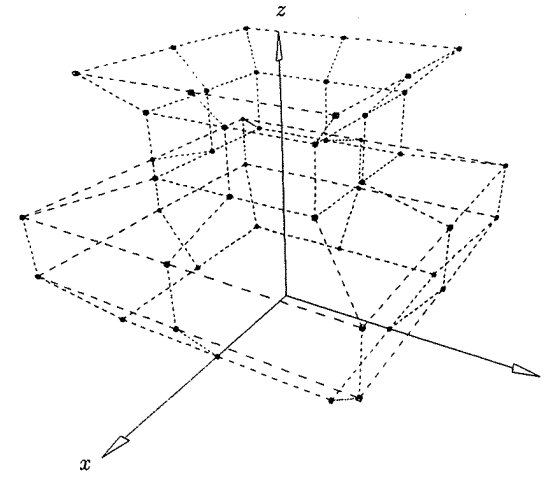
with

$$\mathbf{Q}^w_j = \sum_{i=0}^{8} N_{i,2}(\bar{u}) \mathbf{P}^w_{i,j} \qquad 0 \le j \le m$$

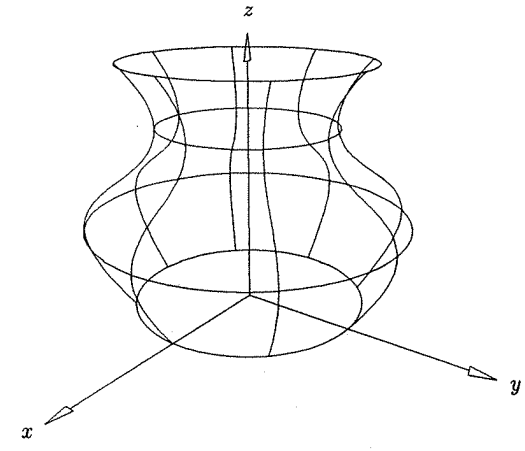
But this says that $\mathbf{Q}_j$ is just $\mathbf{P}_{0,j}$ $(= \mathbf{P}_j)$ rotated about the z-axis through some fixed angle, θ, corresponding to $\bar{u}$. Hence, from the rotational invariance of NURBS curves, it follows that $\mathbf{S}(\bar{u}, v)$ is just $\mathbf{C}(v)$ rotated about the z-axis through θ. This completes the proof.

Several common surfaces are surfaces of revolution. A torus is obtained by revolving a full circle about the z-axis (Figure 8.10). A sphere is obtained by revolving about the z-axis a half circle whose endpoints lie on the z-axis (see Figure 8.11). Notice that the control points at the north and south poles of the sphere are repeated nine times: $\mathbf{P}_{0,0} = \cdots = \mathbf{P}_{8,0}$ and $\mathbf{P}_{0,4} = \cdots = \mathbf{P}_{8,4}$. Hence, the partial derivatives at these poles with respect to u are identically zero. However, the normal vectors clearly exist there.

We now present an algorithm which constructs a NURBS surface of revolution through an arbitrary angle, `theta`, about an arbitrary axis (see Figure 8.12). The axis is specified by a point, `S`, and a unit length vector, `T`. For convenience, we separate weights and three-dimensional control points in this algorithm; `m,Pj[],wj[]` define the generatrix curve, and `n,U,Pij[][],wij[][]` are computed (see Eq. [8.5]). We assume four utility functions:



(a)



(b)

Figure 8.9. (a.) A control net for a surface of revolution; (b) a surface of revolution.

`PointToLine()` : projects a point to a line;
`VecNormalize()` : normalizes a vector and returns its magnitude;
`VecCrossProd()` : computes the vector cross product;
`Intersect3DLines()` : intersects two three-dimensional lines.

Note the similarity between this algorithm and Algorithm `A7.1`.

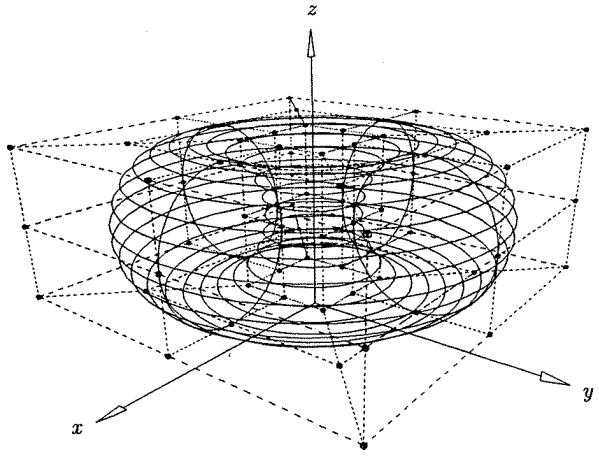


Figure 8.10. A torus as a surface of revolution.

```
ALGORITHM A8.1
  MakeRevolvedSurf(S,T,theta,m,Pj,wj,n,U,Pij,wij)
    { /*  Create NURBS surface of revolution  */
      /*  Input:  S,T,theta,m,Pj,wj  */
      /*  Output: n,U,Pij,wij  */
    if (theta <= 90.0)   narcs = 1;
      else
        if (theta <= 180.0)
          {  narcs = 2;   U[3] = U[4] = 0.5;  }
          else
            if (theta <= 270.0)
              {
              narcs = 3;
              U[3] = U[4] = 1.0/3.0;
              U[5] = U[6] = 2.0/3.0;
              }
              else
              {
              narcs = 4;
              U[3] = U[4] = 0.25;
              U[5] = U[6] = 0.5;
              U[7] = U[8] = 0.75;
              }
    dtheta = theta/narcs;
    j = 3 + 2*(narcs-1);   /* load end knots */
    for (i=0; i<3; j++,i++)
```

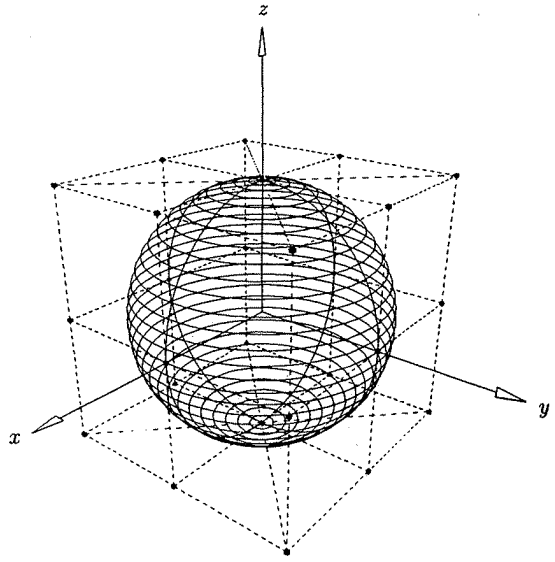



Figure 8.11. A sphere as a surface of revolution.

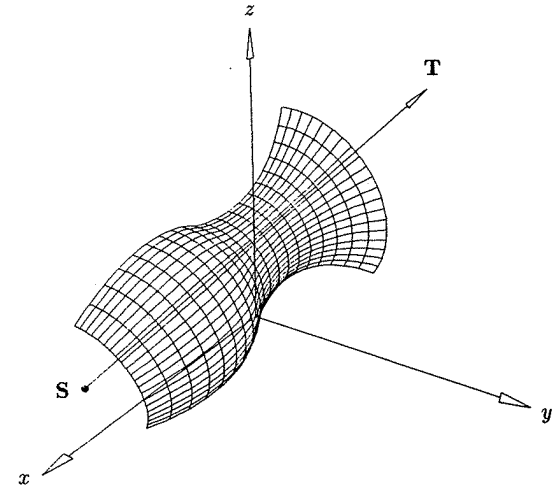


Figure 8.12. A general surface of revolution through an arbitrary angle about an arbitrary axis.

```
  { U[i] = 0.0;   U[j] = 1.0;  }
n = 2*narcs;
wm = cos(dtheta/2.0);   /* dtheta/2 is base angle */
angle = 0.0;   /* Compute sines and cosines only once */
for (i=1; i<=narcs; i++)
  {
  angle = angle + dtheta;
  cosines[i] = cos(angle);
  sines[i] = sin(angle);
  }
for (j=0; j<=m; j++)
  /* Loop and compute each u row of ctrl pts and weights */
  {
  PointToLine(S,T,Pj[j],O);
  X = Pj[j]-O;
  r = VecNormalize(X);    VecCrossProd(T,X,Y);
  Pij[0][j] = P0 = Pj[j];   /* Initialize first */
  wij[0][j] = wj[j];        /* ctrl pt and weight */
  T0 = Y;   index = 0;   angle = 0.0;
  for (i=1; i<=narcs; i++)   /* compute u row */
    {
    P2 = O + r*cosines[i]*X + r*sines[i]*Y;
    Pij[index+2][j] = P2;
    wij[index+2][j] = wj[j];
    T2 = -sines[i]*X + cosines[i]*Y;
    Intersect3DLines(P0,T0,P2,T2,Pij[index+1][j]);
    wij[index+1][j] = wm*wj[j];
    index = index + 2;
    if (i < narcs)   {  P0 = P2;   T0 = T2;  }
    }
  }
}
```

Figure 8.13 shows a general toroidal patch as a surface of revolution used to round off a sharp corner.

8.6 Nonuniform Scaling of Surfaces

Let $\mathbf{P} = (p_i)$, $i = 1, 2, 3$, be a point in three-dimensional space, and let $F = (f_i)$, $i = 1, 2, 3$, be three real numbers (*scale factors*), one for each coordinate. Scaling $\mathbf{P}$ about the origin by the factors F yields the point

$$\bar{\mathbf{P}} = (\bar{p}_i) \qquad \bar{p}_i = f_i p_i, \;\; i = 1, 2, 3 \tag{8.6}$$

If $\mathbf{C} = (c_i)$ is an arbitrary point, then $\mathbf{P}$ is scaled about $\mathbf{C}$ by the equation

$$\bar{\mathbf{P}} = (\bar{p}_i) \qquad \bar{p}_i = f_i p_i + (1 - f_i) c_i, \;\; i = 1, 2, 3 \tag{8.7}$$

The scaling is called *uniform* if the three f_i are equal; otherwise, it is *nonuniform*.

Several interesting surfaces can be obtained by applying a nonuniform scaling to a surface of revolution. Figure 8.14 shows an ellipsoid, centered at the origin, which has an implicit equation of the form

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1 \tag{8.8}$$

The three coordinate planes cut the surface in three ellipses. The ellipsoid is constructed by first producing the unit radius sphere, centered at the origin, and then scaling it about the origin with the factors $F = (a, b, c)$.

Figure 8.15b shows an elliptic paraboloid. The xz and yz planes cut the surface in parabolas; the xy plane cuts it in an ellipse. It has the implicit equation

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = cz \tag{8.9}$$

The elliptic paraboloid of Figure 8.15b is obtained from the paraboloid of revolution shown in Figure 8.15a by scaling about the origin with the factors $F = (a, b, 1)$.

A nonuniform scaling is performed on a NURBS surface by applying the scale factors to the three-dimensional control points. If the control points are stored in

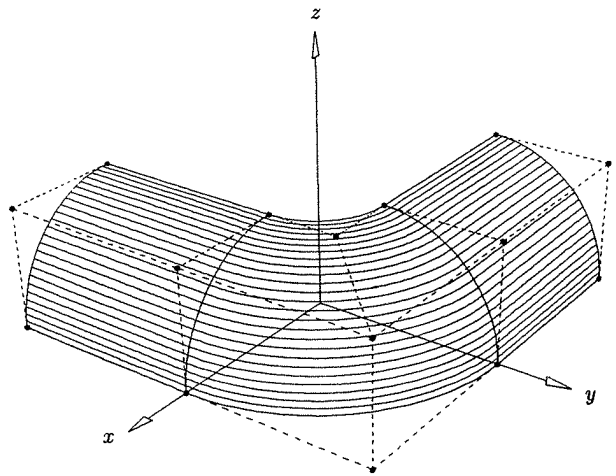


Figure 8.13. A toroidal patch as a surface of revolution used to round a sharp corner.

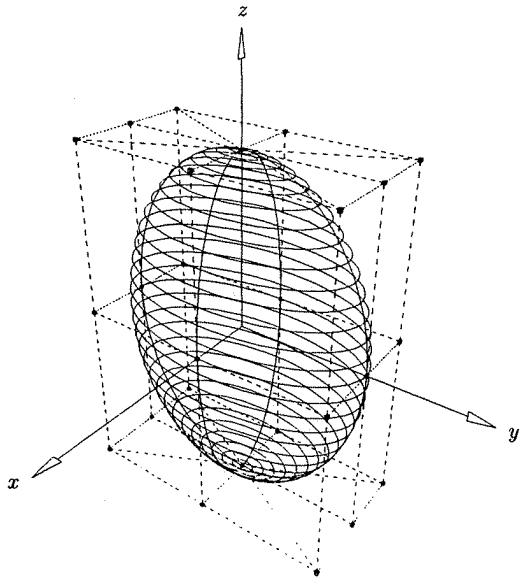
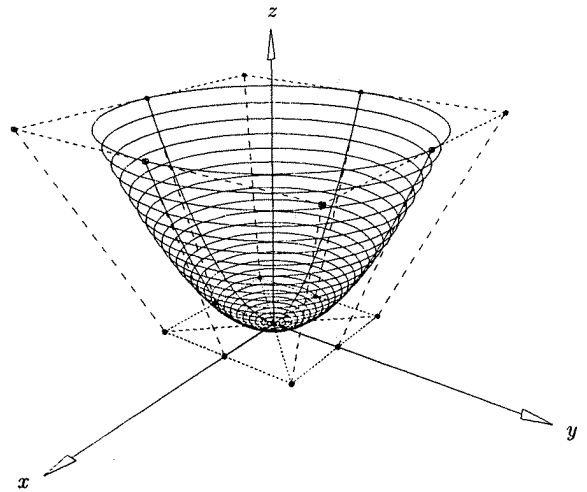
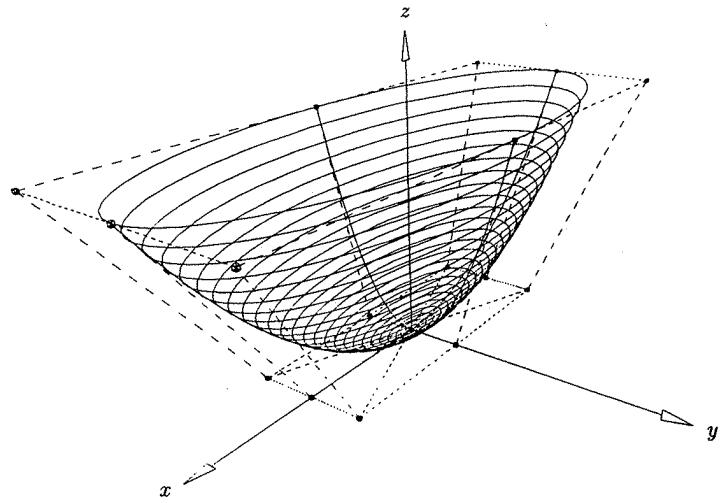


Figure 8.14. An ellipsoid obtained by scaling a sphere.



(a)

Figure 8.15. Paraboloids. (a) A paraboloid of revolution; (b) an elliptic paraboloid.



(b)

Figure 8.15. (*Continued.*)

three-dimensional (unweighted) form, then Eq. (8.6) or Eq. (8.7) can be applied directly. The weights do not change. If the control points are stored in four-dimensional form

$$\mathbf{P}^w = (p_1^w, p_2^w, p_3^w, w) = (p_i^w, w) \qquad i = 1, 2, 3$$

then only the first three coordinates change. Equation (8.6) becomes

$$\bar{\mathbf{P}}^w = (\bar{p}_i^w, w) \qquad \bar{p}_i^w = f_i p_i^w, \ \ i = 1, 2, 3 \tag{8.10}$$

and Eq. (8.7) becomes

$$\bar{\mathbf{P}}^w = (\bar{p}_i^w, w) \qquad \bar{p}_i^w = f_i p_i^w + (1 - f_i) w c_i, \ \ i = 1, 2, 3 \tag{8.11}$$

8.7 A Three-sided Spherical Surface

When three surfaces intersect, the result is three edge curves meeting at a corner point (see Figure 8.16a). Since sharp edges and corners are often undesirable in manufactured parts, surfaces are required which "smooth" or "round off" the sharp edges and corners. These are called *fillet* surfaces. A common conceptual method of obtaining fillet surfaces is the rolling marble model. If a marble with radius R is rolled along all the edge curves and into the corner, the surfaces $\mathbf{S}_4$ $\mathbf{S}_7$ of Figure 8.16b are obtained. $\mathbf{S}_4$–$\mathbf{S}_6$ are *edge fillets*, and $\mathbf{S}_7$ is a *corner*

fillet. Cross sections of $\mathbf{S}_4$–$\mathbf{S}_6$ perpendicular to the directions of the respective edge curves are circular arcs of radius R, whose arc length varies as the angle between the surfaces varies along an edge. In general, these surfaces are quite complex and cannot be represented precisely in NURBS form. Using NURBS, they are usually approximated to within some user-specified tolerance. The corner fillet, $\mathbf{S}_7$, is a three-sided *patch*, lying on a sphere of radius R. We use the term patch to mean a subsurface of some larger surface. The three boundary curves are circular arcs of radius R, whose centers coincide with the center of the sphere. Furthermore, assuming the surfaces do not meet tangentially, the arcs have sweep angles less than 180° and therefore can each be represented with one Bézier segment with all positive weights. The corner fillet is precisely representable as a (4×2)-degree NURBS surface, with one degenerate boundary.

In addition to obtaining a corner fillet surface, the method presented here is a simple and specific case of a more general technique to construct tensor product patches on quadric surfaces. A *quadric surface* is any surface having a second-order implicit equation of the form

$$S(x,y,z) = ax^2 + by^2 + cz^2 + dxy + exz + fyz + gx + hy + iz + j = 0 \qquad (8.12)$$

Spheres, ellipsoids, hyperboloids, paraboloids, and circular cylinders and cones are examples of quadric surfaces. All of these, and many other important surfaces such as tori and a surface of revolution whose generating curve is quadratic, are generated by two families of curves of maximum degree 2. For example, a sphere is generated by two families of circles (circles of latitude and longitude), and the right circular cylinder by a family of straight lines and a family of circles. The techniques of Sections 8.2–8.6 can be used to construct patches on these surfaces if the patch boundary curves are portions of curves comprising the two generating families. The resulting surface patch is biquadratic. If the patch boundaries do not follow the generating families (as in the case of the general corner fillet), then the situation is more complex, and the patch may not even be representable as a NURBS surface. Some results and algorithms have been obtained for the case that the underlying surface is quadric and the patch boundaries are conic sections, e.g., see [Geis90; Boeh91; Hosc92a].

We now show how to construct a corner fillet. Although the details are rather messy, the method is conceptually quite simple; we simply compose two mappings. Without loss of generality we assume that the underlying sphere has radius R and is centered at $(0,0,R)$ (see Figure 8.17). The well-known *stereographic projection* (see [Coxe67; DoCa76]) yields a rational parametric representation of the sphere $\mathbf{S}_1(u,v) = \big(x(u,v), y(u,v), z(u,v)\big)$, where

$$\begin{aligned} x(u,v) &= \frac{4R^2u}{u^2+v^2+4R^2} \\ y(u,v) &= \frac{4R^2v}{u^2+v^2+4R^2} \\ z(u,v) &= \frac{2R(u^2+v^2)}{u^2+v^2+4R^2} \end{aligned} \qquad (8.13)$$

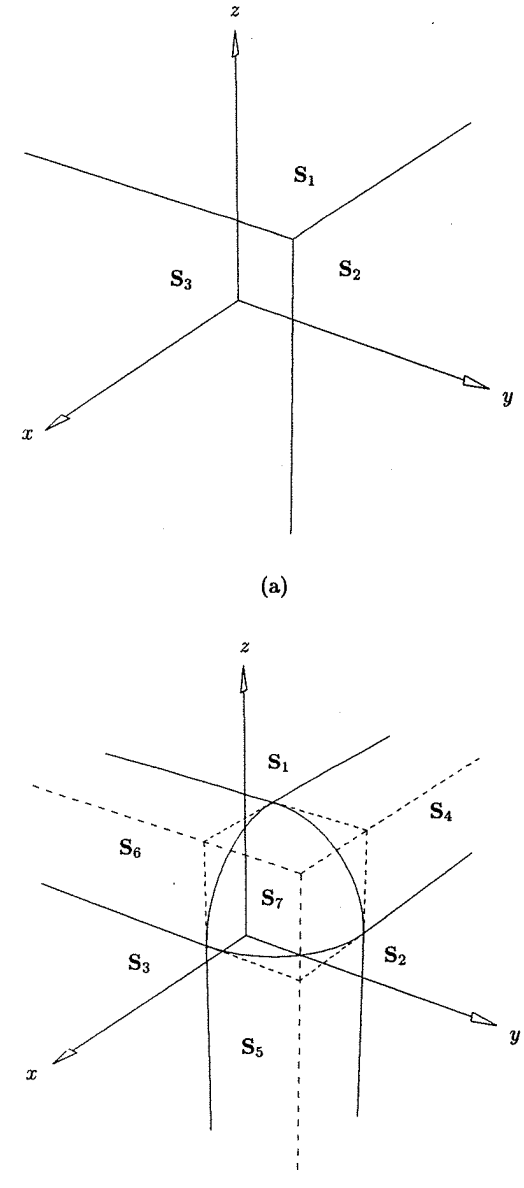


Figure 8.16. Rounding edges and a corner. (a) Three edges and a corner formed by three intersecting planes; (b) three edge fillet surfaces and one corner fillet surface formed by rolling a marble; the rounded corner and the three edges are shown dashed.

that is, a (u, v) point maps into the (x, y, z) point which is at the intersection of the sphere and the line segment from $(u, v, 0)$ to $(0, 0, 2R)$ (see Figure 8.17). Under this projection we have the following correspondences:

- straight lines passing through the origin of the xy plane map to longitudinal circles on the sphere;
- all other circles on the sphere correspond to circles in the xy plane.

The three boundaries of the fillet surface are circular arcs of radius R, with centers at $(0, 0, R)$. Denote these arcs by $\mathbf{C}_1(t), \mathbf{C}_2(s), \mathbf{C}_3(t)$, with $0 \le s, t \le 1$. Without loss of generality we assume that the patch boundaries are positioned and oriented as (see Figure 8.18):

- $\mathbf{C}_1(t)$ and $\mathbf{C}_3(t)$ start at the origin: $\mathbf{C}_1(0) = \mathbf{C}_3(0) = (0, 0, 0)$;
- $\mathbf{C}_1(t)$ lies in the xz plane;
- $\mathbf{C}_1(1) = \mathbf{C}_2(0)$ and $\mathbf{C}_2(1) = \mathbf{C}_3(1)$.

Then the projection of the fillet patch yields a flat surface, $\mathbf{S}_2(s, t) = \big(u(s, t), v(s, t), 0\big)$ in the xy plane. The $s = 0$ and $s = 1$ boundaries of $\mathbf{S}_2(s, t)$ are straight lines, the $t = 0$ boundary is degenerate, and the $t = 1$ boundary is a circular arc. Hence, $\mathbf{S}_2(s, t)$ has degree $(2, 1)$ and the composition mapping $\mathbf{S}(s, t) = \mathbf{S}_1 \circ \mathbf{S}_2$ yields a (4×2)-degree rational Bézier representation of the fillet surface.

Clearly, general formulas can be derived for computing the control points and weights of $\mathbf{S}(s, t)$. However, as most terms are zero anyway, we specifically derive the formulas for each coordinate and weight separately. The computation is:

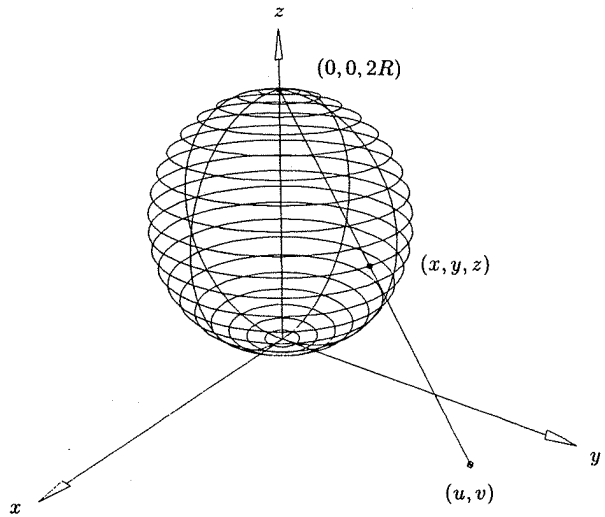


Figure 8.17. Stereographic projection.

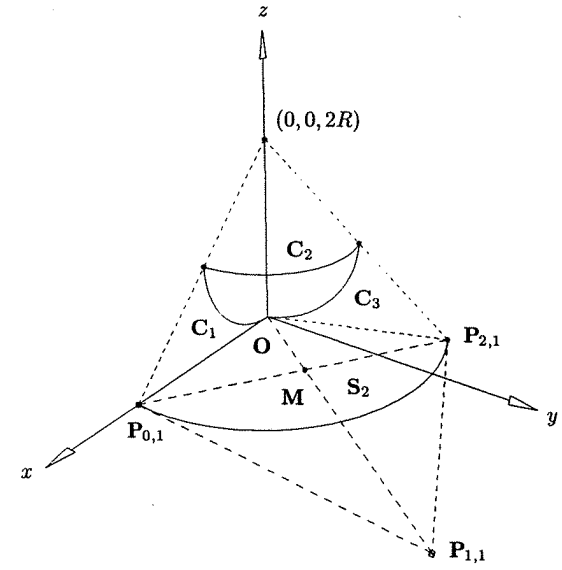


Figure 8.18. The flat surface $\mathbf{S}_2$ obtained by projecting the fillet surface onto the xy plane.

1. obtain the Bézier representation of $\mathbf{S}_2(s, t)$;
2. convert $\mathbf{S}_2(s, t)$ to power basis form;
3. substitute the power basis coefficients of $\mathbf{S}_2(s, t)$ into $\mathbf{S}_1(u, v)$ (Eq. [8.13]) to obtain the power basis form of the fillet patch, $\mathbf{S}(s, t)$;
4. convert $\mathbf{S}(s, t)$ to Bézier form.

The Bézier representation of $\mathbf{S}_2(s, t)$ requires six four-dimensional control points, $\mathbf{P}^w_{i,j}$, $0 \le i \le 2$, $0 \le j \le 1$. Clearly, $\mathbf{P}_{0,0} = \mathbf{P}_{1,0} = \mathbf{P}_{2,0} = (0, 0, 0)$, with $w_{0,0} = w_{2,0} = 1$. $w_{1,0}$ is set equal to $w_{1,1}$, which must still be computed later. $\mathbf{P}^w_{i,1}$, $0 \le i \le 2$, are the control points of the circular arc which is the projection of $\mathbf{C}_2(s)$. They are obtained as follows:

1. project the three points $\mathbf{C}_2(0)$, $\mathbf{C}_2(1/2)$, and $\mathbf{C}_2(1)$ to the circular arc in the xy plane. Let (x, y, z) be the coordinates of one of these points; then the uv coordinates of the projection are computed by the formula

$$\alpha = \frac{-2R}{z - 2R} \qquad u = \alpha x, \;\; v = \alpha y \tag{8.14}$$

   This yields $\mathbf{P}^w_{0,1}$ and $\mathbf{P}^w_{2,1}$ $(w_{0,1} = w_{2,1} = 1)$.

2. compute the center ($\mathbf{O}$ in Figure 8.18) and the radius, r, of the projected arc; for example, $\mathbf{O}$ is found by intersecting the perpendicular bisectors

of two of the chords formed by the projections of the three points, $\mathbf{C}_2(0)$, $\mathbf{C}_2(1/2)$, and $\mathbf{C}_2(1)$;

3. then compute $\mathbf{P}^w_{1,1}$ by (see Figure 8.18):
   - compute $\mathbf{M} = 1/2(\mathbf{P}_{0,1} + \mathbf{P}_{2,1})$;
   - $\mathbf{P}_{1,1}$ is the inverse of $\mathbf{M}$ with respect to the projected circular arc (see [Coxe67]), hence $\mathbf{OM} \cdot \mathbf{OP}_{1,1} = r^2$; from this we get

$$\mathbf{P}_{1,1} = (1-\alpha)\mathbf{O} + \alpha\mathbf{M}$$

     where $$\alpha = \frac{r^2}{|\,\mathbf{OM}\,|^2};$$

   - from Eq. (7.33), $w_{1,1} = \cos \angle \mathbf{P}_{1,1}\mathbf{P}_{2,1}\mathbf{P}_{0,1}$.

Let $[\,\mathbf{a}^w_{i,j}\,]$ denote the power basis coefficients of $\mathbf{S}^w_2(s,t)$. From Eq. (6.91) it follows that

$$[\,\mathbf{a}^w_{i,j}\,] = M_2\,[\,\mathbf{P}^w_{i,j}\,]\,M_1^T$$

where M_2 and M_1 are the second- and first-order Bézier matrices, respectively (Eq. [6.80]). Applying the matrix multiplications yields

$$\begin{aligned}
\mathbf{a}^w_{0,0} &= \mathbf{P}^w_{0,0}\\
\mathbf{a}^w_{0,1} &= \mathbf{P}^w_{0,1} - \mathbf{P}^w_{0,0}\\
\mathbf{a}^w_{1,0} &= 2(\mathbf{P}^w_{1,0} - \mathbf{P}^w_{0,0})\\
\mathbf{a}^w_{1,1} &= 2(\mathbf{P}^w_{1,1} + \mathbf{P}^w_{0,0} - \mathbf{P}^w_{0,1} - \mathbf{P}^w_{1,0})\\
\mathbf{a}^w_{2,0} &= \mathbf{P}^w_{0,0} - 2\mathbf{P}^w_{1,0} + \mathbf{P}^w_{2,0}\\
\mathbf{a}^w_{2,1} &= \mathbf{P}^w_{0,1} + \mathbf{P}^w_{2,1} - \mathbf{P}^w_{0,0} - \mathbf{P}^w_{2,0} + 2(\mathbf{P}^w_{1,0} - \mathbf{P}^w_{1,1})
\end{aligned} \tag{8.15}$$

Now let

$$\mathbf{a}^w_{i,j} = (u_{i,j}, v_{i,j}, h_{i,j}) \tag{8.16}$$

(note that the $u_{i,j}$ and $v_{i,j}$ are weighted). The z coordinate is zero, and to avoid a conflict later we use h instead of w for a weight. Denote the control points of the circular arc by $\mathbf{P}^w_{i,1} = (\alpha_i, \beta_i, \gamma_i)$, $0 \le i \le 2$ (α_i and β_i weighted). Assuming corner weights to be 1, and considering the fact that $\mathbf{C}_1(t)$ lies in the xz plane and $\mathbf{C}_1(0) = (0,0,0)$, it follows that the only nonzero $u_{i,j}$, $v_{i,j}$, and $h_{i,j}$ are

$$\begin{aligned}
u_{0,1} &= \alpha_0\\
u_{1,1} &= 2(\alpha_1 - \alpha_0)\\
u_{2,1} &= \alpha_2 + \alpha_0 - 2\alpha_1\\
v_{1,1} &= 2\beta_1\\
v_{2,1} &= \beta_2 - 2\beta_1\\
h_{0,0} &= 1\\
h_{1,0} &= 2(\gamma_1 - 1)\\
h_{2,0} &= 2(1 - \gamma_1)
\end{aligned} \tag{8.17}$$

Using Eqs. (8.16) and (8.17) and the power basis form

$$\mathbf{S}^w_2(s,t) = \begin{bmatrix}1 & s & s^2\end{bmatrix}[\,\mathbf{a}^w_{i,j}\,]\begin{bmatrix}1\\ t\end{bmatrix} \tag{8.18}$$

we obtain

$$\mathbf{S}_2(s,t) = \big(u(s,t), v(s,t)\big)$$

with

$$u(s,t) = \frac{U(s,t)}{H(s,t)} \qquad v(s,t) = \frac{V(s,t)}{H(s,t)} \tag{8.19}$$

and

$$\begin{aligned}
U(s,t) &= u_{0,1}t + u_{1,1}st + u_{2,1}s^2t\\
V(s,t) &= v_{1,1}st + v_{2,1}s^2t\\
H(s,t) &= h_{0,0} + h_{1,0}s + h_{2,0}s^2
\end{aligned} \tag{8.20}$$

Substituting Eqs. (8.19) and (8.20) into Eq. (8.13), we have

$$\mathbf{S}(s,t) = \big(x(s,t), y(s,t), z(s,t)\big) \tag{8.21}$$

with

$$x(s,t) = \frac{X(s,t)}{W(s,t)} \quad y(s,t) = \frac{Y(s,t)}{W(s,t)} \quad z(s,t) = \frac{Z(s,t)}{W(s,t)} \tag{8.22}$$

and

$$\begin{aligned}
X(s,t) &= 4R^2U(s,t)H(s,t)\\
&= 4R^2(u_{0,1}t + u_{1,1}st + u_{2,1}s^2t)(h_{0,0} + h_{1,0}s + h_{2,0}s^2)\\
&= 4R^2[u_{0,1}h_{0,0}t + (u_{1,1}h_{0,0} + u_{0,1}h_{1,0})st\\
&\quad + (u_{2,1}h_{0,0} + u_{1,1}h_{1,0} + u_{0,1}h_{2,0})s^2t\\
&\quad + (u_{2,1}h_{1,0} + u_{1,1}h_{2,0})s^3t + u_{2,1}h_{2,0}s^4t]
\end{aligned} \tag{8.23}$$

$$\begin{aligned}
Y(s,t) &= 4R^2V(s,t)H(s,t)\\
&= 4R^2(v_{1,1}st + v_{2,1}s^2t)(h_{0,0} + h_{1,0}s + h_{2,0}s^2)\\
&= 4R^2[v_{1,1}h_{0,0}st + (v_{2,1}h_{0,0} + v_{1,1}h_{1,0})s^2t\\
&\quad + (v_{1,1}h_{2,0} + v_{2,1}h_{1,0})s^3t + v_{2,1}h_{2,0}s^4t]
\end{aligned} \tag{8.24}$$

$$\begin{aligned}
Z(s,t) &= 2R\,[U(s,t)^2 + V(s,t)^2]\\
&= 2R\,[u_{0,1}^2t^2 + 2u_{0,1}u_{1,1}st^2 + (u_{1,1}^2 + 2u_{0,1}u_{2,1} + v_{1,1}^2)s^2t^2\\
&\quad + 2(u_{1,1}u_{2,1} + v_{1,1}v_{2,1})s^3t^2 + (u_{2,1}^2 + v_{2,1}^2)s^4t^2]
\end{aligned} \tag{8.25}$$

$$\begin{aligned}
W(s,t) &= U(s,t)^2 + V(s,t)^2 + 4R^2H(s,t)^2\\
&= 4R^2h_{0,0}^2 + 8R^2h_{0,0}h_{1,0}s + 4R^2(h_{1,0}^2 + 2h_{0,0}h_{2,0})s^2 + u_{0,1}^2t^2\\
&\quad + 8R^2h_{1,0}h_{2,0}s^3 + 2u_{0,1}u_{1,1}st^2 + 4R^2h_{2,0}^2s^4\\
&\quad + (u_{1,1}^2 + 2u_{0,1}u_{2,1} + v_{1,1}^2)s^2t^2\\
&\quad + 2(u_{1,1}u_{2,1} + v_{1,1}v_{2,1})s^3t^2 + (u_{2,1}^2 + v_{2,1}^2)s^4t^2
\end{aligned} \tag{8.26}$$

Denote by

$$\mathbf{S}^w(s,t) = [\,s^i\,]\,[\,\mathbf{b}^w_{i,j}\,]\,[\,t^j\,]^T \qquad 0 \le i \le 4 \quad 0 \le j \le 2 \tag{8.27}$$

the power basis representation of $\mathbf{S}^w(s,t)$, and let $\mathbf{b}^w_{i,j} = (x_{i,j}, y_{i,j}, z_{i,j}, w_{i,j})$. Most of the coordinates of the $\mathbf{b}^w_{i,j}$ are zero. The only nonzero elements are

$$\begin{aligned}
x_{0,1} &= 4R^2 u_{0,1} h_{0,0} \\
x_{1,1} &= 4R^2 (u_{1,1} h_{0,0} + u_{0,1} h_{1,0}) \\
x_{2,1} &= 4R^2 (u_{2,1} h_{0,0} + u_{1,1} h_{1,0} + u_{0,1} h_{2,0}) \\
x_{3,1} &= 4R^2 (u_{2,1} h_{1,0} + u_{1,1} h_{2,0}) \\
x_{4,1} &= 4R^2 u_{2,1} h_{2,0}
\end{aligned} \tag{8.28}$$

$$\begin{aligned}
y_{1,1} &= 4R^2 v_{1,1} h_{0,0} \\
y_{2,1} &= 4R^2 (v_{2,1} h_{0,0} + v_{1,1} h_{1,0}) \\
y_{3,1} &= 4R^2 (v_{1,1} h_{2,0} + v_{2,1} h_{1,0}) \\
y_{4,1} &= 4R^2 v_{2,1} h_{2,0}
\end{aligned} \tag{8.29}$$

$$\begin{aligned}
z_{0,2} &= 2R u^2_{0,1} \\
z_{1,2} &= 4R u_{0,1} u_{1,1} \\
z_{2,2} &= 2R (u^2_{1,1} + 2u_{0,1} u_{2,1} + v^2_{1,1}) \\
z_{3,2} &= 4R (u_{1,1} u_{2,1} + v_{1,1} v_{2,1}) \\
z_{4,2} &= 2R (u^2_{2,1} + v^2_{2,1})
\end{aligned} \tag{8.30}$$

$$\begin{aligned}
w_{0,0} &= 4R^2 h^2_{0,0} \\
w_{1,0} &= 8R^2 h_{0,0} h_{1,0} \\
w_{2,0} &= 4R^2 (h^2_{1,0} + 2h_{0,0} h_{2,0}) \\
w_{3,0} &= 8R^2 h_{1,0} h_{2,0} \\
w_{4,0} &= 4R^2 h^2_{2,0} \\
w_{0,2} &= u^2_{0,1} \\
w_{1,2} &= 2u_{0,1} u_{1,1} \\
w_{2,2} &= u^2_{1,1} + 2u_{0,1} u_{2,1} + v^2_{1,1} \\
w_{3,2} &= 2(u_{1,1} u_{2,1} + v_{1,1} v_{2,1}) \\
w_{4,2} &= u^2_{2,1} + v^2_{2,1}
\end{aligned} \tag{8.31}$$

Equations (8.27)–(8.31) give the power basis form of the fillet surface $\mathbf{S}^w(s,t)$. The Bézier (NURBS) representation is obtained by applying Eq. (6.100), that is

$$[\,\mathbf{P}^w_{i,j}\,] = M_4^{-1}\,[\,\mathbf{b}^w_{i,j}\,]\,(M_2^{-1})^T$$

Algorithm A6.2 is used to compute M_4^{-1} and M_2^{-1}.

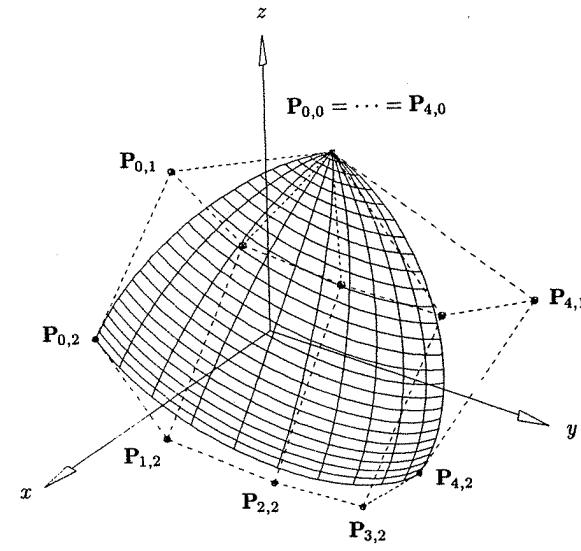


Figure 8.19. A corner fillet surface – an arbitrary spherical patch whose boundaries are great circles of the sphere.

We now summarize the complete algorithm. Input to the algorithm are three boundary arcs, $\mathbf{C}_1$, $\mathbf{C}_2$, and $\mathbf{C}_3$, arbitrarily positioned and oriented in space but joined at their endpoints. Output is a (4×2)-degree rational Bézier fillet patch.

```
ALGORITHM A8.2
  MakeCornerFilletSurf(C1,C2,C3,Pwij)
    { /* Create NURBS corner fillet surface  */
      /* Input:  C1,C2,C3  */
      /* Output: Pwij  */
    1. Compute the common radius and center point of the three
       boundary arcs.
    2. Using translation, rotation and possibly curve reversal
       (see Chapter 6), position and orient the boundary arcs
       as in Figure 8.18.  Note that, in fact, only C2 must
       actually be positioned and oriented, as only it is
       required in order to compute the three control points
       P^w_{i,1} = (α_i, β_i, γ_i).
    3. Compute the (α_i, β_i, γ_i) as described above.
    4. Convert S^w_2(s,t) from Bézier to power basis form
       (Eq. [8.17]).
    5. Compute the power basis form of the fillet patch
       S^w = S^w_1 ∘ S^w_2  (Eqs. [8.27]–[8.31]).
```

6. Convert $\mathbf{S}^w(s,t)$ to rational Bézier form (Eq. [8.32] and Algorithm A6.2).
7. Rotate/translate $\mathbf{S}(s,t)$ back to the correct location in three-dimensional space (the inverse transformation of Step 2).

}

Figure 8.19 shows a corner fillet patch constructed by Algorithm A8.2.