Available online at www.sciencedirect.com

SCIENCE DIRECT®

Computer Aided Geometric Design 22 (2005) 1–26

COMPUTER AIDED GEOMETRIC DESIGN

www.elsevier.com/locate/cagd

Approximate swept volumes of NURBS surfaces or solids

Jingzhou Yang [*], Karim Abdel-Malek

Virtual Soldier Research Program, Center for Computer-Aided Design, The University of Iowa, 116 Engineering Research Facility, Iowa City, IA 52242-1000, USA

Received 11 July 2003; received in revised form 13 February 2004; accepted 2 August 2004

Available online 28 August 2004

Abstract

This paper presents a method of determining the approximate swept volume of Non-Uniform Rational B-Spline (NURBS) surfaces or solids. The method consists of (1) slicing the NURBS surfaces or solids by finding the intersection of plane/surface; (2) forming the sliced curves; (3) setting up the local moving coordinate system; (4) determining the characteristic (also called singular) points or curves by obtaining local maxima points at discrete frames during motion and with respect to a local moving coordinate system; (5) fitting each NURBS singular surface; (6) trimming the singular surfaces to obtain the boundary of the final approximate swept volumes by the surface/surface intersection and perturbation method. The local moving coordinate system is set up in reference to the motion direction of the rigid body as determined from its composite velocity vector. This work aims to develop a rigorous method for identifying and visualizing the approximate swept volume generated as a result of sweeping a NURBS surface or solid. The method and numerical algorithm are illustrated through examples.

Keywords: NURBS; Velocity vector; Deformation; Boundary envelope; Swept volume; Trimming

1. Introduction

The Non-Uniform-Rational-B-Spline (NURBS) representation has become the standard curve and surface description in the field of Computer-Aided Design (CAD). NURBS can be viewed as a unified

[*] Corresponding author. Tel: (319) 353-2249; Fax: (319) 384-0542.
E-mail address: jyang@engineering.uiowa.edu (J. Yang).

0167-8396/$ – see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.cagd.2004.08.002

method for representing all piecewise polynomial geometry used in CAD/CAM systems since they include Bézier and B-spline geometric entities as special cases (Farin, 1993).

The objective of this work is to obtain a systematic method for formulating the sweep operation of NURBS surfaces or solids and to develop a capability for visualizing the resulting volume. The resulting NURBS swept volume has many advantages because the swept volume can be viewed as manipulable free-form solids. Indeed, free-form volumes have also been called trivariate solids (Joy and Duchaineau, 1999). Because of their geometric structure, these solids can play an important role in the fields of visualization, virtual reality, and physics-based modeling.

Sweeping of geometric entities is important and interesting in its own right and has been shown to have a variety of applications such as robotic analysis, collision detection, manufacturing design and planning, numerically controlled (NC) machining verification and simulation (Roth et al., 2001), and ergonomic analysis (see the review by Abdel-Malek et al. (2001)).

The swept volume is defined as the geometric space occupied by a moving object along a trajectory in a given time interval. Note that the trajectory is not limited to a single parametric curve but could be a given surface (i.e., sweeping of a surface on another surface). The moving object, which can be a curve, surface, or solid, is called a generator. The motion of the generator along a path is a sweep.

Several mathematical methods have been developed to address the sweeping of geometric entities, such as the Jacobian Rank Deficiency (JRD) method (Abdel-Malek and Yeh, 1997; Abdel-Malek et al., 2000a, 2000b, 2000c, 2001; Abdel-Malek and Yang, 2000), the Sweep Differential Equation (SDE) method (Blackmore et al., 1992, 1994, 1997a, 1997b, 1999; Blackmore and Leu, 1992), and the trivariate solid visualization method (Joy, 1992; Joy and Duchaineau, 1999; Madrigal and Joy, 1999). General methods to determine the swept volume of parametric and B-spline entities are well-established. The JRD method has only been demonstrated in parametric and implicit sweeping with multiple parameters. However, a promising method by Joy (1992) extended the JRD method to B-spline trivariate solids using an approximation to the Jacobian determinant of the underlying B-spline function to determine the implicit boundary surface.

The SDE method (Blackmore et al., 1999) had been demonstrated for the determination of the sweep of planar parametric curves but has not been attempted for implementing the sweeps of free-form surfaces. The reader is referred to a recent comprehensive review (Abdel-Malek et al., 2001) addressing swept volumes, with particularly focus on the JRD, SDE and trivariate volume visualization methods.

Many researchers have demonstrated curve skinning for NURBS (Ball, 1974, 1975, 1977; Filip and Ball, 1989; Till, 1983; Woodward, 1987, 1988; Hohmeyer and Barsky, 1991), as well as the concept of swung NURBS (Woodward, 1987). To our knowledge, there are no reported techniques for determining and visualizing the NURBS solid generated by sweeping a NURBS surface. Sweeps of objects undergoing deformation (excluding free-form surfaces) beyond the application of the ray-casting engine to such sweeps (Marisam et al., 1992) were not addressed. Nevertheless, parametric deformable models were addressed by the SDE method. Perhaps the closest notable work to NURBS sweeping is that by Xia and Ge (2000), which presents a formulation for approximating the motion of ruled surfaces represented as NURBS; the work was limited to such surfaces. While considerable research that relates to swept volumes has been carried out over the past decade, it is evident that a consistent and systematic formulation does not exist. As for sweeping of NURBS curves, only a few algorithms have been reported (Bloomenthal and Riessenfeld, 1991; Coquillart, 1987; Piegl and Tiller, 1997) where the resulting swept geometry is indeed a NURBS surface.

In this paper, we propose a methodology to determine and visualize the approximate swept volume of NURBS surfaces or solids, subject to general translational and rotational motions. The method proposed here can be applied to parametric surfaces. Note, however, that we have presented an effective method (JRD) for the sweeping (and multiple sweeping) of parametric and implicit surfaces in closed form in previous works (Abdel-Malek and Yeh, 1997; Abdel-Malek and Othman, 1999). JRD method is not applicable to the NURBS surfaces because of the unique properties it exhibits. The method proposed in this paper uses existing utilities, especially plane/surface and surface/surface intersection, planar curve extremal point detection, curve and surface fitting to simplify the design of the algorithm.

2. Formulation

As a geometric entity is moving in space while experiencing general translational and rotational motions, a trace is made by edges (although arbitrary) of the entity on the boundary, thus characterizing and defining the envelope. An edge (in Fig. 1) for a geometric entity is a grazing curve constructed by grazing points on a moving surface at which the direction of motion lies in the tangent plane. An envelope (in Fig. 1) is the set of surfaces that reside on the boundary of an object. This envelope could be made of several surface patches that cover the object.

The goal is to determine these edges, obtain the boundary, and visualize it. In this section, we first define the problem, then give the outline of the proposed methodology, and, finally, show several examples to illustrate the algorithm.

2.1. Problem definition

Consider this sweep of a NURBS surface or solid $\mathbf{S}(u, v)$ in Fig. 2 as

$$^{A}\boldsymbol{\xi}(u, v, t) = \mathbf{R}(t)\mathbf{S}(u, v) + \boldsymbol{\Psi}(t), \tag{1}$$

where $\mathbf{S}(u, v)$ is a NURBS surface of degree (p, q) and is in the form of Eq. (A.6), the parameters u and v are limited to $0 \leqslant u, v \leqslant 1$, where $n = r - p - 1$, $m = s - q - 1$, and r and s are the number of elements in the knot vectors U and V, respectively. $\mathbf{R}(t)$ is a (3×3) rotation matrix with the rotation axis $\boldsymbol{\Gamma}$; $\boldsymbol{\omega}$ is the angular velocity vector, where $\boldsymbol{\omega} = [\omega_x\ \omega_y\ \omega_z]^{\mathbf{T}}$; and $\boldsymbol{\Psi}(t)$ is the path trajectory, and $t \in [0, t_{\max}]$. Point

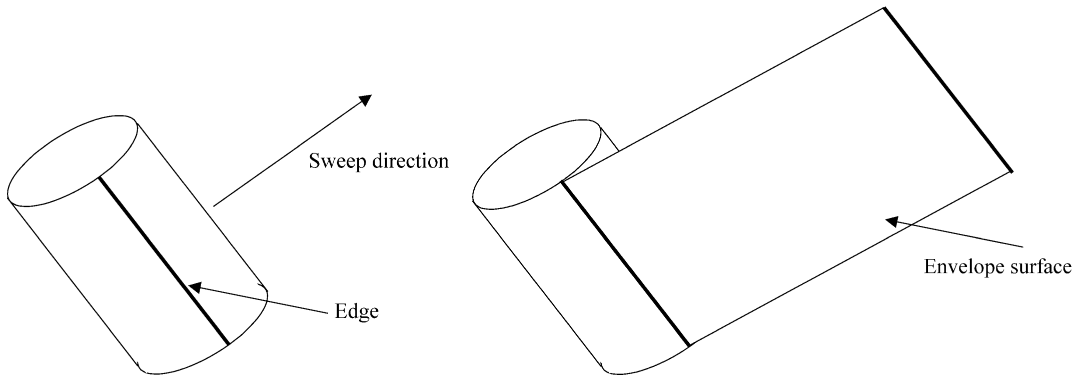


Fig. 1. Edge and envelope surface.

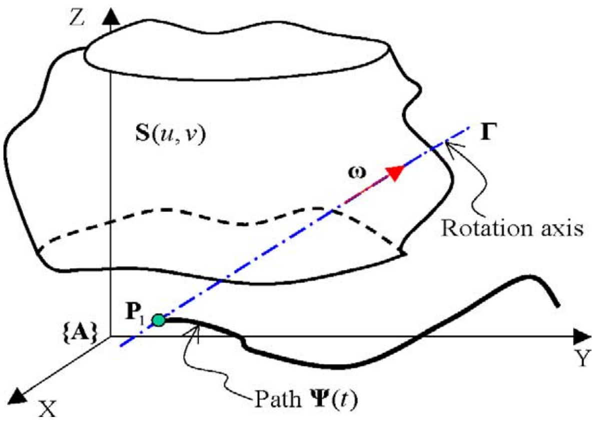


Fig. 2. Sweeping the NURBS surfaces or solids ($t = t_1 = 0$).

$\mathbf{P}_1$ is the instantaneous center of $\mathbf{S}(u, v)$ at $t = t_1 = 0$; $\{A\} = \{\mathbf{X}, \mathbf{Y}, \mathbf{Z}\}$ is the global coordinate system. Note that Eq. (1) indeed is a volume. The objective is to better understand the geometric make-up of this volume and to develop a systematic method to identify its envelope and to visualize it.

2.2. Methodology

The swept volume of NURBS surfaces or solids is computed by determining singular surfaces of the sliced NURBS curves from the original NURBS surfaces or solids where this process reduces the dimensionality. The algorithm for the approximate swept volume of NURBS surfaces and solids is summarized as follows:

1. Slice the object (NURBS surface or solid).
2. Build the planar curve for each slice.
3. Set up the local moving coordinate system to be embedded with each planar curve (slice).
4. Find the extremal points (maxima and minima).
5. Construct the singular surfaces.
6. Trim the singular surfaces to determine the boundary sub-surfaces and delete the internal sub-surfaces.

2.2.1. Slice the object (NURBS surface or solid)

The direction of the plane to slice the NURBS surface or solid is important. Later on we will show that this direction will not only affect the detection of the extreme points on each slice curve, but also the stability of the singular surface fitting. Note that we slice the object at the beginning of this sweeping.

In Fig. 3 the curve ${}^{A}\mathbf{C}^{i}(\lambda^{i}, t_1)$ is formed by intersection of plane Υ and surface $\mathbf{S}(u, v)$, and λ^i is the parameter of the sliced curve, $i = 1, 2, \ldots, \chi$, which represents the ith slice. The point $\mathbf{Q}_1^i$ is the geometric center of the curve ${}^{A}\mathbf{C}^{i}(\lambda^{i}, t_1)$, and $\mathbf{r}_1^i$ is the vector from point $\mathbf{P}_1$ to point $\mathbf{Q}_1^i$. $\mathbf{V}_{\mathbf{Q}_1^i}$ is the absolute velocity of point $\mathbf{Q}_1^i$. $\mathbf{n}$ is the normal vector of plane Υ. The criteria to choose the plane Υ are defined by

$$\mathbf{n} \cdot \mathbf{V}_{\mathbf{Q}_1^i} \neq 0. \tag{2}$$

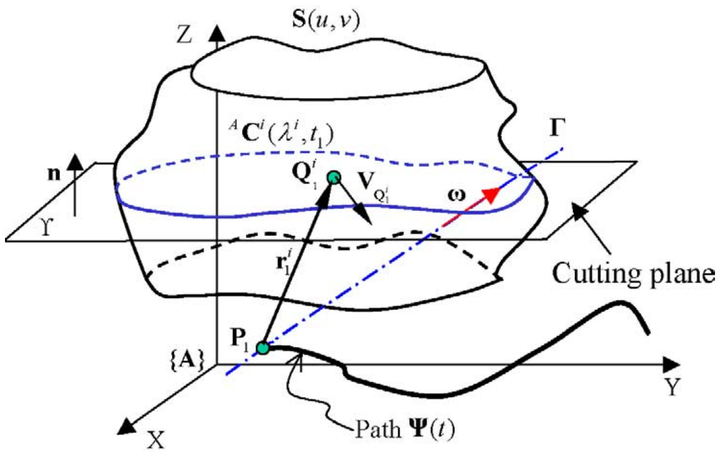


Fig. 3. Slicing the object to be swept.

Using the subdivision algorithm (Levin, 1999) and (Litke et al., 2001), we obtain the intersection points $\mathbf{g}_1^i, \mathbf{g}_2^i, \ldots, \mathbf{g}_l^i$, where l is the total number of the points, by cutting $\mathbf{S}(u, v)$ with plane Υ.

2.2.2. *Build the sliced planar curves*

We can calculate the control points $\mathbf{H}_\alpha^i$ of the ith slice from the intersection points $\mathbf{g}_1^i, \mathbf{g}_2^i, \ldots, \mathbf{g}_l^i$ in step 1, where $\alpha = 1, \ldots, k$, k is the total number of control points for this curve. One can obtain the knot vector U, base functions $N_{\alpha,p}(\lambda^i)$, and the weight vector ω_α by NURBS fitting algorithm (Piegl and Tiller, 1997), where p is the degree of the curve, λ^i is the parameter for this curve; then the initial ith slice at $t = t_1 = 0$ can be defined according to Eq. (A.7) by

$$ {}^{A}\mathbf{C}^i(\lambda^i, 0) = \frac{\sum_{\alpha=1}^{k} N_{\alpha,p}(\lambda^i)\omega_\alpha \mathbf{H}_\alpha^i}{\sum_{\alpha=1}^{k} N_{\alpha,p}(\lambda^i)\omega_\alpha}. \tag{3} $$

The curve ${}^{A}\mathbf{C}^i(\lambda^i, t_j)$ shown in Fig. 4 at time $t = t_j$, where $j \in 1, \ldots, \beta$, and $\beta - 1$ is the total number of intervals of time t, can be defined by

$$ {}^{A}\mathbf{C}^i(\lambda^i, t_j) = \mathbf{R}(t_j)\,{}^{A}\mathbf{C}^i(\lambda^i, 0) + \boldsymbol{\Psi}(t_j). \tag{4} $$

From Eq. (3), Eq. (4) can be simplified to

$$ {}^{A}\mathbf{C}^i(\lambda^i, t_j) = \mathbf{R}(t_j)\frac{\sum_{\alpha=1}^{k} N_{\alpha,p}(\lambda^i)\omega_\alpha \mathbf{H}_\alpha^i}{\sum_{\alpha=1}^{k} N_{\alpha,p}(\lambda^i)\omega_\alpha} + \boldsymbol{\Psi}(t_j). \tag{5} $$

2.2.3. *Set up the local moving coordinate system*

Consider the slice ${}^{A}\mathbf{C}^i(\lambda^i, t_j)$ as a rigid body, where this body is swept such that it translates along the path $\boldsymbol{\Psi}(t)$, and rotates at an absolute angular velocity $\boldsymbol{\omega}$ with respect to the rotation axis $\boldsymbol{\Gamma}$. If this rotation axis coincides with the X-axis, Y-axis, or Z-axis, the rotation matrix $\mathbf{R}(t)$ will be $\mathbf{R}_X(t)$, $\mathbf{R}_Y(t)$, $\mathbf{R}_Z(t)$, respectively. Otherwise, $\mathbf{R}(t) = \mathbf{R}_X(t) \bullet \mathbf{R}_Y(t) \bullet \mathbf{R}_Z(t)$. Fig. 4 depicts the local moving coordinate system $\{B\} = \{\mathbf{e}_{1j}^i, \mathbf{e}_{2j}^i, \mathbf{e}_{3j}^i\}$, where the unit vectors $\mathbf{e}_{1j}^i, \mathbf{e}_{2j}^i$ and $\mathbf{e}_{3j}^i$ will be defined in the following section. Point $\mathbf{P}_j$ is the instantaneous center, which is the intersection of the rotation axis $\boldsymbol{\Gamma}$ and the path

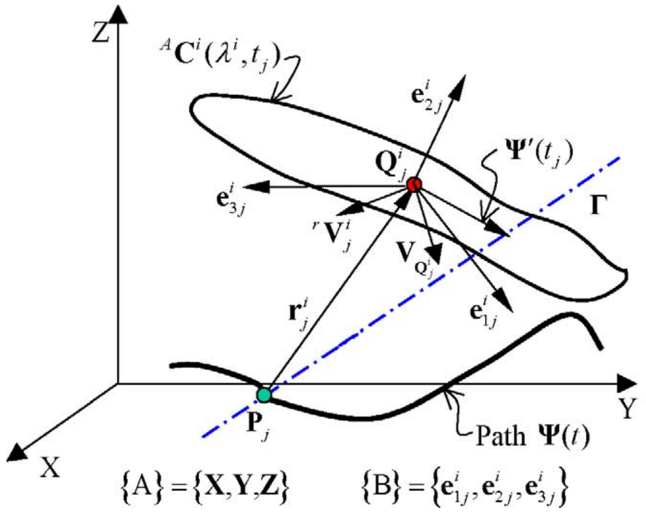

Z
${}^{A}\mathbf{C}^{i}(\lambda^{i},t_{j})$
$\mathbf{e}^{i}_{2j}$
$\mathbf{\Psi}'(t_{j})$
$\mathbf{Q}^{i}_{j}$
$\mathbf{e}^{i}_{3j}$
Γ
${}^{r}\mathbf{V}^{i}_{j}$
$\mathbf{V}_{\mathbf{Q}^{i}_{j}}$
$\mathbf{r}^{i}_{j}$
$\mathbf{e}^{i}_{1j}$
Y
$\mathbf{P}_{j}$
Path $\mathbf{\Psi}(t)$
X
$\{A\}=\{\mathbf{X},\mathbf{Y},\mathbf{Z}\}$
$\{B\}=\{\mathbf{e}^{i}_{1j},\mathbf{e}^{i}_{2j},\mathbf{e}^{i}_{3j}\}$

Fig. 4. Local moving coordinate system.

$\mathbf{\Psi}(t)$, while point $\mathbf{Q}^i_j$ is the geometric center of the slice ${}^A\mathbf{C}^i(\lambda^i, t_j)$ at $t = t_j$. $\mathbf{r}^i_j$ is a position vector from the instantaneous center $\mathbf{P}_j$ to point $\mathbf{Q}^i_j$; ${}^r\mathbf{V}^i_j$ is the relative velocity of point $\mathbf{Q}^i_j$ with respect to $\mathbf{P}_j$. $\mathbf{V}_{\mathbf{Q}^i_j}$ is the absolute velocity of point $\mathbf{Q}^i_j$. Fig. 4 shows that at $t = t_j$ the vectors $\mathbf{r}^i_j$, $\mathbf{V}_{\mathbf{Q}^i_j}$, ${}^r\mathbf{V}^i_j$, $\mathbf{\Psi}'(t_j)$, $\mathbf{e}^i_{1j}$, $\mathbf{e}^i_{2j}$ and $\mathbf{e}^i_{3j}$ are related.

Note that the rigid body ${}^A\mathbf{C}^i(\lambda^i, t_j)$ undergoes translational and rotational motion. Therefore, $\mathbf{V}_{\mathbf{Q}^i_j}$ is defined by

$$\mathbf{V}_{\mathbf{Q}^i_j} = \mathbf{V}_{\mathbf{p}_j} + {}^r\mathbf{V}^i_j, \quad \text{where } \mathbf{V}_{\mathbf{p}_j} = \mathbf{\Psi}'(t_j),\ {}^r\mathbf{V}^i_j = \boldsymbol{\omega} \times \mathbf{r}^i_j \text{ and } \mathbf{r}^i_j = \mathbf{Q}^i_j - \mathbf{P}_j = \mathbf{Q}^i_j - \mathbf{\Psi}(t_j). \tag{6}$$

At $t = t_1 = 0$ we can define point $\mathbf{Q}^i_j|_{t=0} = \mathbf{Q}^i_1$, then at $t = t_j$, $\mathbf{Q}^i_j = \mathbf{R}(t_j)\mathbf{Q}^i_1 + \mathbf{\Psi}(t_j)$. Therefore, from Eq. (6) $\mathbf{V}_{\mathbf{Q}^i_j}$ will be

$$\mathbf{V}_{\mathbf{Q}^i_j} = \mathbf{\Psi}'(t_j) + \begin{bmatrix} w_x \\ w_y \\ w_z \end{bmatrix} \times \left(\mathbf{R}(t_j)\mathbf{Q}^i_1\right). \tag{7}$$

We can obtain the normal vector of the planar curve ${}^A\mathbf{C}^i(\lambda^i, t_j)$ by choosing three different points on this curve in clockwise direction, $\mathbf{F}^i_{1j} = {}^A\mathbf{C}^i(\lambda^i_1, t_j)$, $\mathbf{F}^i_{2j} = {}^A\mathbf{C}^i(\lambda^i_2, t_j)$, $\mathbf{F}^i_{3j} = {}^A\mathbf{C}^i(\lambda^i_3, t_j)$, where $\lambda^i_1 < \lambda^i_2 < \lambda^i_3$; therefore, two unparallel vectors in this curve's plane $\mathbf{F}^i_{1j} - \mathbf{F}^i_{2j}$ and $\mathbf{F}^i_{3j} - \mathbf{F}^i_{2j}$ can be generated, and the unit normal vector can be defined by

$$\mathbf{e}^i_{2j} = \frac{(\mathbf{F}^i_{1j} - \mathbf{F}^i_{2j}) \times (\mathbf{F}^i_{3j} - \mathbf{F}^i_{2j})}{\|(\mathbf{F}^i_{1j} - \mathbf{F}^i_{2j}) \times (\mathbf{F}^i_{3j} - \mathbf{F}^i_{2j})\|}. \tag{8}$$

From Eq. (5) we can get

$$\mathbf{F}^i_{1j} - \mathbf{F}^i_{2j} = \mathbf{R}(t_j)\left(\frac{\sum_{\alpha=1}^{k} N_{\alpha,p}(\lambda^i_1)\omega_\alpha \mathbf{H}^i_\alpha}{\sum_{\alpha=1}^{k} N_{\alpha,p}(\lambda^i_1)\omega_\alpha} - \frac{\sum_{\alpha=1}^{k} N_{\alpha,p}(\lambda^i_2)\omega_\alpha \mathbf{H}^i_\alpha}{\sum_{\alpha=1}^{k} N_{\alpha,p}(\lambda^i_2)\omega_\alpha}\right), \tag{9}$$

$$\mathbf{F}^i_{3j} - \mathbf{F}^i_{2j} = \mathbf{R}(t_j)\left(\frac{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_3)\omega_\alpha \mathbf{H}^i_\alpha}{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_3)\omega_\alpha} - \frac{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_2)\omega_\alpha \mathbf{H}^i_\alpha}{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_2)\omega_\alpha}\right) \tag{10}$$

and from Eqs. (8)–(10)

$$\mathbf{e}^i_{2j} = \frac{\mathbf{R}(t_j)\left(\frac{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_2)\omega_\alpha \mathbf{H}^i_\alpha}{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_2)\omega_\alpha} - \frac{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_1)\omega_\alpha \mathbf{H}^i_\alpha}{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_1)\omega_\alpha}\right) \times \mathbf{R}(t_j)\left(\frac{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_3)\omega_\alpha \mathbf{H}^i_\alpha}{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_3)\omega_\alpha} - \frac{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_1)\omega_\alpha \mathbf{H}^i_\alpha}{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_1)\omega_\alpha}\right)}{\left\|\mathbf{R}(t_j)\left(\frac{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_2)\omega_\alpha \mathbf{H}^i_\alpha}{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_2)\omega_\alpha} - \frac{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_1)\omega_\alpha \mathbf{H}^i_\alpha}{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_1)\omega_\alpha}\right) \times \mathbf{R}(t_j)\left(\frac{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_3)\omega_\alpha \mathbf{H}^i_\alpha}{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_3)\omega_\alpha} - \frac{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_1)\omega_\alpha \mathbf{H}^i_\alpha}{\sum_{\alpha=1}^k N_{\alpha,p}(\lambda^i_1)\omega_\alpha}\right)\right\|}. \tag{11}$$

Eq. (11) reduces to a numerical value when carried out. Then, $\mathbf{e}^i_{3j}$ can be obtained by

$$\mathbf{e}^i_{3j} = \frac{\mathbf{V}_{\mathbf{Q}^i_j} \times \mathbf{e}^i_{2j}}{\|\mathbf{V}_{\mathbf{Q}^i_j} \times \mathbf{e}^i_{2j}\|}. \tag{12}$$

Simplify Eq. (12) from Eq. (7)

$$\mathbf{e}^i_{3j} = \frac{(\boldsymbol{\Psi}'(t_j) + \boldsymbol{\omega} \times (\mathbf{R}(t_j)\mathbf{Q}^i_1)) \times \mathbf{e}^i_{2j}}{\|\boldsymbol{\Psi}'(t_j) + \boldsymbol{\omega} \times (\mathbf{R}(t_j)\mathbf{Q}^i_1) \times \mathbf{e}^i_{2j}\|}. \tag{13}$$

As a result, the $\mathbf{e}^i_1$ can be computed by

$$\mathbf{e}^i_{1j} = \mathbf{e}^i_{2j} \times \mathbf{e}^i_{3j}, \tag{14}$$

$$\mathbf{e}^i_{1j} = \mathbf{e}^i_{2j} \times \frac{(\boldsymbol{\Psi}'(t_j) + \boldsymbol{\omega} \times (\mathbf{R}(t_j)\mathbf{Q}^i_1)) \times \mathbf{e}^i_{2j}}{\|\boldsymbol{\Psi}'(t_j) + \boldsymbol{\omega} \times (\mathbf{R}(t_j)\mathbf{Q}^i_1) \times \mathbf{e}^i_{2j}\|}, \tag{15}$$

where $\mathbf{e}^i_{2j}$ has a numerical solution in Eq. (12). From the above, we create a local moving coordinate system $\{B\} = \{\mathbf{e}^i_{1j}, \mathbf{e}^i_{2j}, \mathbf{e}^i_{3j}\}$. If Eq. (12) equals zero or close to zero, the velocity vector $\mathbf{V}_{\mathbf{Q}^i_j}$ will parallel $\mathbf{e}^i_{2j}$ and it will cause a problem in step 4. Therefore, we have to jump back to step 1 to adjust the direction of the cutting plane Υ.

2.2.4. *Find the extremal points (maxima and minima)*

To find the extreme points on the curve ${}^A\mathbf{C}^i(\lambda^i, t_j)$ with respect to the local moving coordinate system obtained from step 3, we need to transform the curve ${}^A\mathbf{C}^i(\lambda^i, t_j)$ from the global system $\{A\}$ to this local moving system $\{B\}$. The transformation matrix from $\{B\}$ to $\{A\}$ can be defined by

$${}^A\mathbf{T}_B(t_j) = \begin{bmatrix} \mathbf{X}\cdot\mathbf{e}^i_{1j} & \mathbf{X}\cdot\mathbf{e}^i_{2j} & \mathbf{X}\cdot\mathbf{e}^i_{3j} \\ \mathbf{Y}\cdot\mathbf{e}^i_{1j} & \mathbf{Y}\cdot\mathbf{e}^i_{2j} & \mathbf{Y}\cdot\mathbf{e}^i_{3j} \\ \mathbf{Z}\cdot\mathbf{e}^i_{1j} & \mathbf{Z}\cdot\mathbf{e}^i_{2j} & \mathbf{Z}\cdot\mathbf{e}^i_{3j} \end{bmatrix}, \tag{16}$$

where $\mathbf{X} = [1\ 0\ 0]^T$, $\mathbf{Y} = [0\ 1\ 0]^T$, $\mathbf{Z} = [0\ 0\ 1]^T$.

Therefore, Eq. (16) can be simplified to

$${}^A\mathbf{T}_B(t_j) = [\mathbf{e}^i_{1j}\ \mathbf{e}^i_{2j}\ \mathbf{e}^i_{3j}] \tag{17}$$

from Eqs. (12)–(15), Eq. (17) can be derived as

$${}^A\mathbf{T}_B(t_j) = \left[\mathbf{e}^i_{2j} \times \frac{(\boldsymbol{\Psi}'(t_j) + \boldsymbol{\omega} \times (\mathbf{R}(t_j)\mathbf{Q}^i_1)) \times \mathbf{e}^i_{2j}}{\|\boldsymbol{\Psi}'(t_j) + \boldsymbol{\omega} \times (\mathbf{R}(t_j)\mathbf{Q}^i_1) \times \mathbf{e}^i_{2j}\|} \quad \mathbf{e}^i_{2j} \quad \frac{(\boldsymbol{\Psi}'(t_j) + \boldsymbol{\omega} \times (\mathbf{R}(t_j)\mathbf{Q}^i_1)) \times \mathbf{e}^i_{2j}}{\|\boldsymbol{\Psi}'(t_j) + \boldsymbol{\omega} \times (\mathbf{R}(t_j)\mathbf{Q}^i_1) \times \mathbf{e}^i_{2j}\|}\right]. \tag{18}$$

With the transformation matrix, the slice i curve ${}^{A}\mathbf{C}^{i}(\lambda^{i}, t_{j})$ can be represented in the local coordinate system $\{B\}$ by

$$ {}^{B}\mathbf{C}^{i}(\lambda^{i}, t_{j}) = ({}^{A}\mathbf{T}_{B})^{T\,A}\mathbf{C}^{i}(\lambda^{i}, t_{j}). \tag{19} $$

Substitute Eq. (5) into the above equation

$$ {}^{B}\mathbf{C}^{i}(\lambda^{i}, t_{j}) = \left[\mathbf{e}^{i}_{2j} \times \frac{(\boldsymbol{\Psi}'(t_j) + \boldsymbol{\omega} \times (\mathbf{R}(t_j)\mathbf{Q}^{i}_{1})) \times \mathbf{e}^{i}_{2j}}{\|\boldsymbol{\Psi}'(t_j) + \boldsymbol{\omega} \times (\mathbf{R}(t_j)\mathbf{Q}^{i}_{1}) \times \mathbf{e}^{i}_{2j}\|} \quad \mathbf{e}^{i}_{2j} \quad \frac{(\boldsymbol{\Psi}'(t_j) + \boldsymbol{\omega} \times (\mathbf{R}(t_j)\mathbf{Q}^{i}_{1})) \times \mathbf{e}^{i}_{2j}}{\|\boldsymbol{\Psi}'(t_j) + \boldsymbol{\omega} \times (\mathbf{R}(t_j)\mathbf{Q}^{i}_{1}) \times \mathbf{e}^{i}_{2j}\|} \right]^{T} \left(\mathbf{R}(t_j) \frac{\sum_{\alpha=1}^{k} N_{\alpha,p}(\lambda^{i})\omega_{\alpha}\mathbf{H}^{i}_{\alpha}}{\sum_{\alpha=1}^{k} N_{\alpha,p}(\lambda^{i})\omega_{\alpha}} + \boldsymbol{\Psi}(t_j) \right). \tag{20} $$

From the above, the curve ${}^{B}\mathbf{C}^{i}(\lambda^{i}, t_{j})$ lies in the plane $\mathbf{e}^{i}_{1j}\mathbf{Q}^{i}_{j}\mathbf{e}^{i}_{3j}$ and the absolute velocity $\mathbf{V}_{\mathbf{Q}^{i}_{j}}$ of point $\mathbf{Q}^{i}_{j}$ lies in the plane $\mathbf{e}^{i}_{1j}\mathbf{Q}^{i}_{j}\mathbf{e}^{i}_{2j}$. To determine the singular points at $t = t_j$, it is necessary to determine all the extreme points (for $\mathbf{e}^{i}_{3j}$ coordinates) with respect to local coordinate $\{B\}$. In the local coordinate system $\{B\}$, especially in the plane $\mathbf{e}^{i}_{1j}\mathbf{Q}^{i}_{j}\mathbf{e}^{i}_{3j}$, we can derive the following equation

$$ \frac{d\,{}^{B}\mathbf{C}^{i}(\lambda^{i}, t_{j})_{3}}{d\,{}^{B}\mathbf{C}^{i}(\lambda^{i}, t_{j})_{1}} = 0, \tag{21} $$

where ${}^{B}\mathbf{C}^{i}(\lambda^{i}, t_{j}) = [{}^{B}\mathbf{C}^{i}(\lambda^{i}, t_{j})_{1}\ {}^{B}\mathbf{C}^{i}(\lambda^{i}, t_{j})_{2}\ {}^{B}\mathbf{C}^{i}(\lambda^{i}, t_{j})_{3}]^{T}$, and Eq. (21) represents the derivative of ${}^{B}\mathbf{C}^{i}(\lambda^{i}, t_{j})_{3}$ with respect to ${}^{B}\mathbf{C}^{i}(\lambda^{i}, t_{j})_{1}$ equals zero.

Eq. (21) is equal to the following one

$$ \frac{\frac{d\,{}^{B}\mathbf{C}^{i}(\lambda^{i}, t_{j})_{3}}{d\lambda^{i}}}{\frac{d\,{}^{B}\mathbf{C}^{i}(\lambda^{i}, t_{j})_{1}}{d\lambda^{i}}} = 0. \tag{22} $$

Solving Eq. (22), we obtain a set $\lambda^{i} = {}^{\tau}\lambda^{i},\ i = 1, \ldots, \eta$, and ($\tau = 1$ or $\tau = 1, 2$, or $\tau = 2$) that characterize singular points on the boundary of the entity at this particular location in space at $t = t_j$. Note here that $\tau = 1$ means the singular points are at maximum, $\tau = 1, 2$ means the singular points are at minimum and maximum, and $\tau = 2$ means the singular points are at minimum. When substituting the set ${}^{\tau}\lambda^{i}$ into the initial curves ${}^{A}\mathbf{C}^{i}(\lambda^{i}, t_{j})$ with $i = 1, \ldots, \chi$ we can calculate the singular points ${}^{A}C^{i}({}^{\tau}\lambda^{i}, t_{j})$, where $i = 1, \ldots, \eta$ and ($\tau = 1$ or $\tau = 1, 2$, or $\tau = 2$). For $j \in 1, \ldots, \beta$ we can find all of the extremal points ${}^{A}C^{i}({}^{\tau}\lambda^{i}, t_{j})$ for each slice $i = 1, \ldots, \eta$.

2.2.5. Construct the singular surfaces

From step 4, we have the singular points for maxima and minima, and the singular surfaces $\boldsymbol{\Phi}(\mu, \nu)$, $0 \leqslant \mu, \nu \leqslant 1$, can be formed through these extremal points (maxima or minima, respectively). If we define the general singular sets as Ω, then Type I singular sets are these surfaces obtained from the extremal points for maxima and minima points, respectively, which are defined by

Type I.

$$ \Omega^{1} = \left\{ \boldsymbol{\Phi}^{1}(\mu, \nu) \cup \boldsymbol{\Phi}^{2}(\mu, \nu) \right\}, \tag{23} $$

where $\boldsymbol{\Phi}^{1}(\mu, \nu)$ is fitted by maxima points ${}^{A}C^{i}({}^{1}\lambda^{i}, t_{j})$ with i and j indices; $\boldsymbol{\Phi}^{2}(\mu, \nu)$ is fitted by minima points ${}^{A}C^{i}({}^{2}\lambda^{i}, t_{j})$ with i and j indices.

Type II singularity sets are formed by setting any one of the parameters u, v, t to its limit in Eq. (1), and which are defined by

Type II.

$$\Omega^2 = \{{}^A\boldsymbol{\xi}(u, v, t) \mid u \to \text{limit}\} \cup \{{}^A\boldsymbol{\xi}(u, v, t) \mid v \to \text{limit}\} \cup \{{}^A\boldsymbol{\xi}(u, v, t) \mid t \to \text{limit}\}. \tag{24}$$

Therefore, the final swept volume is $\Omega = \Omega^1 \cup \Omega^2$.

There are two types of surface fitting: interpolation and approximation (Piegl and Tiller, 1997). In the interpolation method, we construct a surface that satisfies the given data precisely, e.g., the surface passes through the given points and assumes the given derivatives at the prescribed points. But if we choose a slicing direction improperly, i.e., where the value of Eqs. (2) and (12) is close to zero, then this interpolation method will be unstable. In the approximation method, it is often desirable to specify a maximum bound on the deviation of the surface from the given data, and to specify certain constraints, i.e., data that are to be satisfied precisely. This approximation method of surface fitting will solve the unstable problem. In this paper we use the surface interpolation method.

The so-called set of characteristic equations (22) is highly nonlinear, but is readily solvable using well-known numerical techniques. The parametric set $({}^\tau\lambda^i, t_j)$ represents points on the edge of the surface with respect to the direction of motion (i.e., with respect to the velocity vector). These points, referred to as singular points, always fall on the 'highest' point of the surface, although not all of them necessarily fall on the boundary of the resulting swept volume. Therefore, we must further identify those singular points that exist on the boundary of the swept volume.

2.2.6. Trim the singular surfaces to obtain the boundary

Several analytical techniques have been devised for addressing the trimming problem, and these have led to a reasonably satisfactory solution for two-dimensional swept volumes (Ahn et al., 1993; Jiang et al., 1994; Ling and Chase, 1996) and to substantial progress towards a solution for three-dimensional swept volumes (Akman and Arslan, 1992; Elber, 1997). Hu et al. (1997) used a slicing procedure to reduce the dimension of the trimming problem, and Blackmore et al. (1999) developed a 3-D trimming algorithm to determine the boundary surface of a three-dimensional swept volume.

In this paper, we use a two-step trimming algorithm: (1) Apply surface/surface intersection to divide the singular surfaces into sub-surfaces; (2) Use a perturbation method to determine the boundary of the swept volume.

Consider ns NURBS singular entities

$$\boldsymbol{\chi}^\kappa(\mathbf{u}^\kappa), \quad \text{for } \kappa = 1, \ldots, ns. \tag{25}$$

Intersections between these singular surfaces may exist. The intersection curves, also called singular curves, partition a singular surface into a number of sub-regions called sub-surfaces, denoted by $\boldsymbol{\psi}^\kappa$. Singular curves are considered as higher-order singularities. It is necessary to determine these curves, such that every sub-surface can be defined, as shown in Fig. 5.

To compute the intersection curves between two NURBS singular surfaces defined by

$$\boldsymbol{\chi}^1(u, v), \quad u_1 \leqslant u \leqslant u_2,\ v_1 \leqslant v \leqslant v_2, \tag{26}$$

$$\boldsymbol{\chi}^2(s, w), \quad s_1 \leqslant s \leqslant s_2,\ w_1 \leqslant w \leqslant w_2 \tag{27}$$

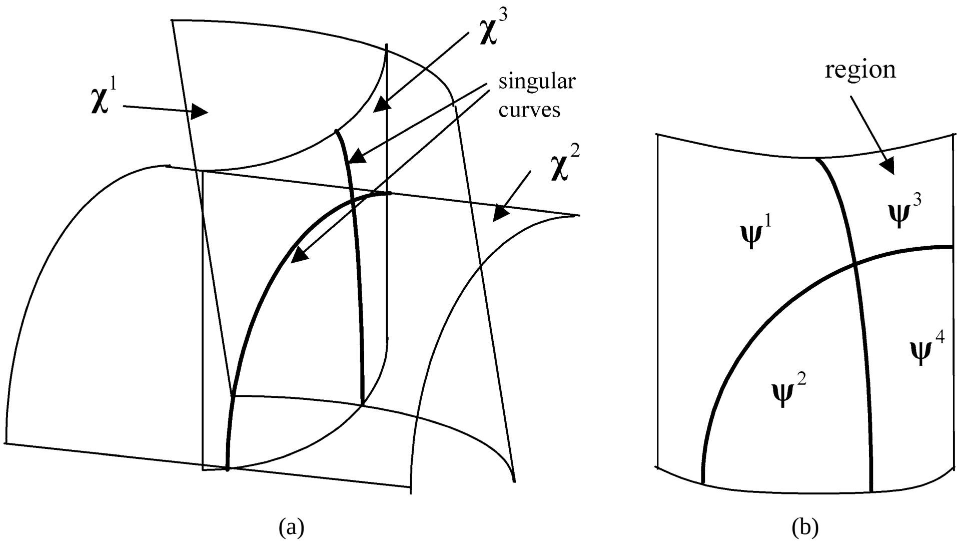


Fig. 5. The intersection and partition of singular surfaces.

the subdivision method proposed by Cohen et al. (2001) is employed.

$$\chi^1(\mathbf{u}^1) - \chi^2(\mathbf{u}^2) = \mathbf{0}. \tag{28}$$

Determining the intersections between all singular surfaces allows each sub-surface to be identified. Sub-surfaces may exist inside the swept volume or on its outer boundaries. To determine whether sub-surface ψ^κ is internal or on the boundary, a perturbation of a selected point on the sub-surface is carried out. The idea is that the points, perturbed along the normal direction, on both sides of the sub-surface should satisfy the equation of constraints, if the sub-surface is inside the swept volume, rather than on the boundary. Any point on the sub-surface can be selected, provided the point is not on an intersection curve.

For a sub-surface $\psi^\kappa(\mathbf{u}^\kappa) = \psi^\kappa(s, w)$, the variables s, w are the parameters. The normal vector of a singular surface can be calculated as

$$\mathbf{n}^0 = \left(\frac{\partial\psi^\kappa}{\partial s} \times \frac{\partial\psi^\kappa}{\partial w}\right) \Big/ \left\|\frac{\partial\psi^\kappa}{\partial s} \times \frac{\partial\psi^\kappa}{\partial w}\right\|. \tag{29}$$

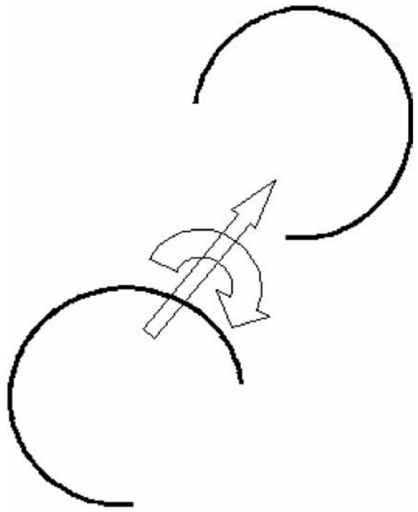
For a small perturbation $\partial\varepsilon$ about this point and along the normal vector $\mathbf{n}^0$, the coordinates of the perturbed point can be calculated as

$$\mathbf{x}^\mathbf{p} = \psi^\kappa(s^0, w^0) \pm \partial\varepsilon\mathbf{n}^0. \tag{30}$$

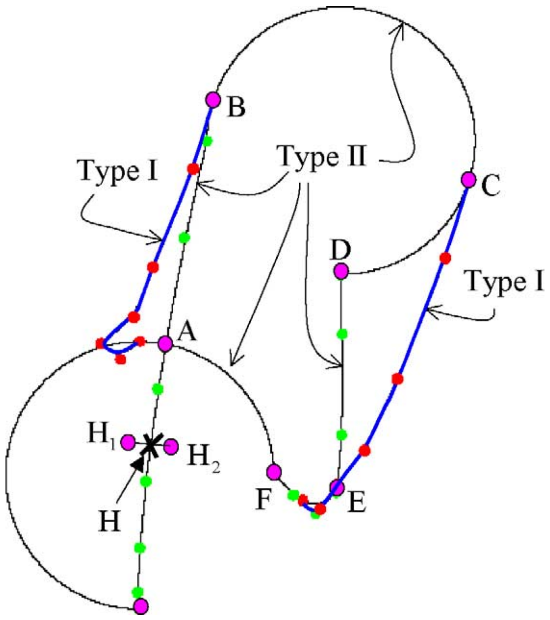
For the perturbed point to exist within the swept volume, an admissible vector $\mathbf{v} = [u \ v \ t]^T$ should exist, which satisfies Eq. (1), such that a solution to the following equation exists.

$$\begin{bmatrix} \mathbf{x}^\mathbf{p} - {}^A\boldsymbol{\xi}(\mathbf{v}) \\ u - 0.5 - 0.5\sin\lambda_1 \\ u - 0.5 - 0.5\sin\lambda_2 \\ t - 0.5t_{\max} - 0.5t_{\max}\sin\lambda_3 \end{bmatrix} = \mathbf{0}, \tag{31}$$

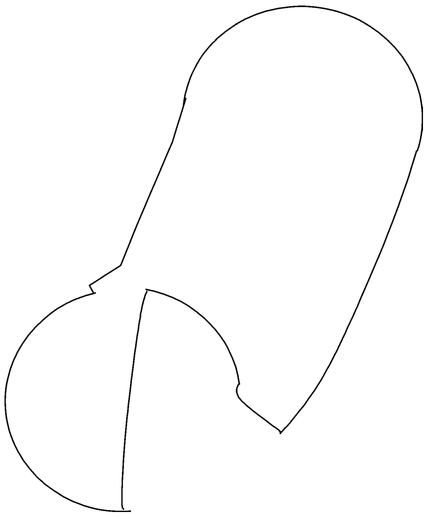
where $\boldsymbol{\lambda} = [\lambda_1\ \lambda_2\ \lambda_3]^T$ are the slack variables. The modified Newton–Raphson method with the generalized inverse is used to determine whether this nonlinear system of equations has a solution. The sub-surface $\boldsymbol{\psi}^{\kappa}$ is an internal surface if and only if solutions of Eq. (31) exist for both perturbations of $\pm\partial\varepsilon$; otherwise, it will be on the boundary of the swept volume. Note that we have used a perturbation of 0.1 the minimum dimension of the swept volume as a rule of thumb and there are no rigorous rules for selecting a $\partial\varepsilon$.



(a)



(b)



(c)

Fig. 6. (a) Sweeping of a planar curve. (b) Singular curves. (c) Boundary envelope of the swept area.

To illustrate the algorithm we proposed, consider the sweep of a simple free-form planar curve as illustrated in Fig. 6(a). The objective is to sweep the planar curve along a given path while changing the orientation. The analysis of this sweep operation yields the curves shown in Fig. 6(b). After identifying which curve segments remain on the boundary, the envelope of the swept area is depicted in Fig. 6(c). Determination of the envelope is a two-step process:

(1) Identification of curve segments (and surface regions) that intersect;
(2) Determination of curve segments (or surface regions) that lie on the boundary to the envelope.

To establish whether a curve segment is inside or outside the boundary, we use the perturbation method for each point on the segment. From the perturbation, we obtain two points (on each side of the curve) and determine whether both points satisfy the sweep equation. Numerically, this method is not intensive and is only required for one point on each curve segment. Consider, for example, the point H on the curve segment AG. A perturbation of that point in the normal direction to the tangent would yield two points H_1 and H_2 (depicted in Fig. 6(b)). Since only one of those points satisfies the sweep equation (i.e., belongs to the workspace swept by the geometric entity), then this curve segment is identified as a boundary segment. Note that the resulting boundary envelope of the swept area is readily determined and shown in Fig. 6(c). If both points of a perturbation satisfy the sweep equation, then the curve segment (or surface patch) is considered an internal branch and is clipped (i.e., removed).

2.3. Computational error and cost

We will show the results for the computational errors by the examples in the following section (Table 3). Considering all of the six steps discussed above it is not difficult to show that the computational cost is $O([\mathrm{Max}(\eta, \beta)]^4)$. Therefore, when choosing the values i and j we have consider both computational error and cost. Values for η and β in step 1 can be made to be selected by the user such that more accurate swept volumes are calculated and less cost is made.

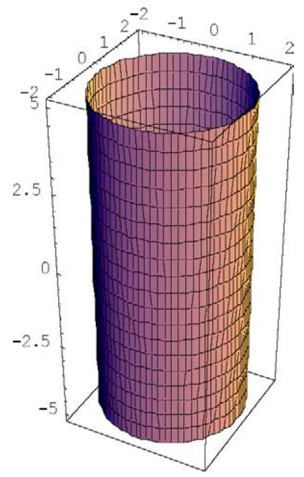
3. Examples

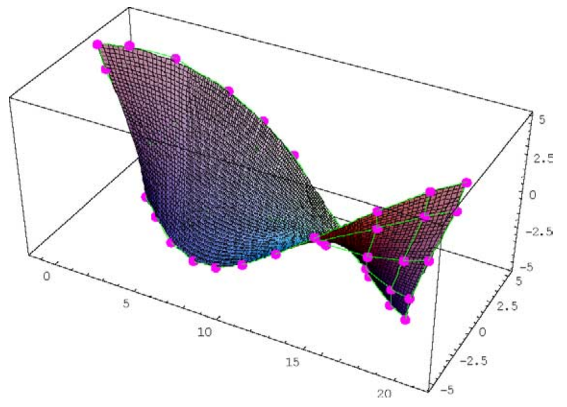
Example 1. Consider the NURBS cylindrical surface in Fig. 7(a), which has path $\boldsymbol{\Psi}(t) = [10t\ 0\ 0]^T$ and rotation matrix

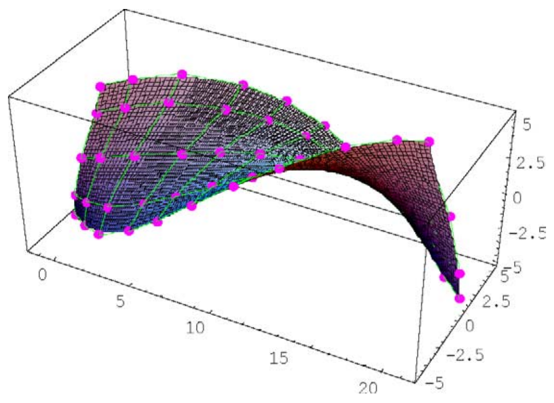
$$\mathbf{R}(t) = \mathbf{R}_X(t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\left(-\frac{\pi}{2}t\right) & -\sin\left(-\frac{\pi}{2}t\right) \\ 0 & \sin\left(-\frac{\pi}{2}t\right) & \cos\left(-\frac{\pi}{2}t\right) \end{bmatrix}, \quad 0 \leqslant t \leqslant 2 \quad \text{and} \quad \boldsymbol{\omega} = \left[-\tfrac{\pi}{2} \quad 0 \quad 0\right]^{\mathbf{T}}.$$

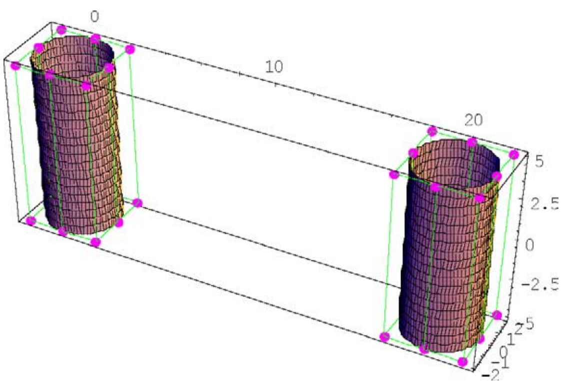
In Eq. (23), let $i = 5$ and $j = 11$, then we get $\tau = 1$ and is entered into Table 1. As a result, a singular surface s_1 is identified and shown in Fig. 7(b).

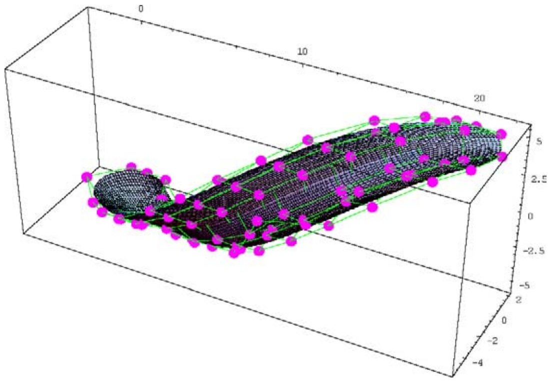
Type I singular set II $\tau = 2$ is shown in Table 2 and the singular surface is s_2. It is plotted in Fig. 7(c). Type II singularities are $s_3 = \{t = 0\}$, $s_4 = \{t = 2\}$, $s_5 = \{v = 0\}$, $s_6 = \{v = 2\}$, $s_7 = \{u = 0\}$, $s_8 = \{u = 2\pi\}$. But s_7, s_8 are inside the swept volume, so are not plotted. Singular surfaces s_3, s_4, s_5, s_6 are shown in Figs. 7(b), (d)–(f), respectively, and the final swept volume is shown in Fig. 7(g).


(a)


(b)


(c)


(d)


(e)

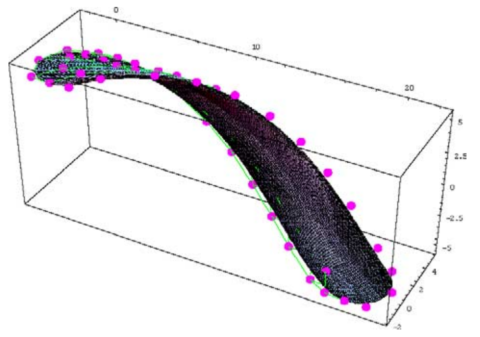
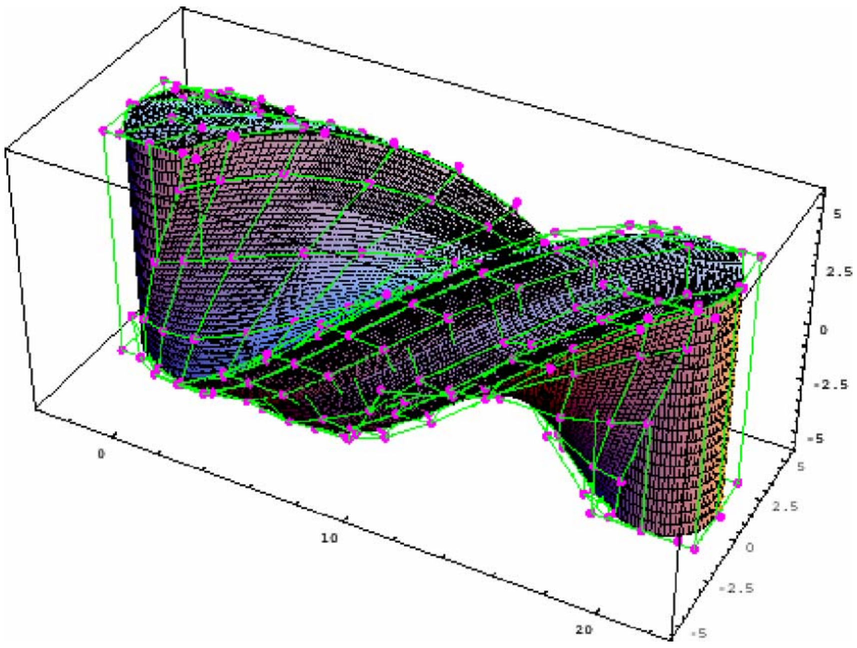

(f)

Fig. 7. (a) Initial surface. (b) Type I singularity s_1. (c) Type I singularity s_2. (d) Type II singular sets s_3, s_4. (e) Type II singular set s_5. (f) Type II singular set s_6. (g) Final swept volume.



(g)

Fig. 7. (*continued*)

Table 1
Singular points set I (Type I)

i / *j*	1	2	3	4	5
1	(1.235, 1.573, −5.000)	(0.853, 1.809, −3.000)	(0.000, 2.000, 0.000)	(−0.853, 1.809, 3.000)	(−1.235, 1.573, 5.000)
2	(3.235, −0.049, −5.241)	(2.853, 0.794, −3.412)	(2.000, 1.902, −0.618)	(1.147, 2.648, 2.294)	(0.765, 3.041, 4.269)
3	(5.235, −1.666, −4.970)	(4.853, −0.299, −3.490)	(4.000, 1.618, −1.176)	(3.147, 3.227, 1.364)	(2.765, 4.211, 3.121)
4	(7.235, −3.121, −4.211)	(6.853, −1.364, −3.227)	(6.000, 1.176, −1.618)	(5.147, 3.491, 0.299)	(4.765, 4.969, 1.666)
5	(9.235, −4.269, −3.041)	(8.853, −2.294, −2.647)	(8.000, 0.618, −1.902)	(7.147, 3.412, −0.794)	(6.765, 5.241, 0.049)
6	(11.235, −5.000, −1.573)	(10.853, −3.000, −1.809)	(10.000, 0.000, −2.000)	(9.147, 3.000, −1.809)	(8.765, 5.000, −1.573)
7	(13.235, −5.241, 0.049)	(12.853, −3.412, −0.794)	(12.000, −0.618, −1.902)	(11.147, 2.294, −2.648)	(10.646, 4.269, −3.041)
8	(15.235, −4.967, 1.666)	(14.853, −3.491, 0.299)	(14.000, −1.176, −1.618)	(13.147, 1.364, −3.227)	(12.765, 3.121, −4.211)
9	(17.235, −4.211, 3.121)	(16.853, −3.227, 1.364)	(16.000, −1.618, −1.176)	(15.147, 0.2997, −3.491)	(14.765, 1.666, −4.969)
10	(19.235, −3.041, 4.269)	(18.853, −2.648, 2.294)	(18.000, −1.902, −0.618)	(17.147, −0.794, −3.412)	(16.765, 0.049, −5.241)
11	(21.235, −1.573, 5.000)	(20.853, −1.809, 3.000)	(20.000, −2.000, 0.000)	(19.147, −1.809, −3.000)	(18.765, −1.573, −5.000)

Example 2. Consider the modeling of a coil (a mechanical spring) in a Computer-Aided Design system. Although the spring can be modeled using parametric or implicit methods, the need for deformable volumetric models such as those generated by the sweeping of NURBS surfaces has recently emerged in applications for virtual reality and physics-based modeling. Using free-form NURBS sweeping, we not only are able to represent the coil, but can also deform or modify it by changing the control nets or weights and can superimpose physics-based models (Terzopoulos and Qin, 1994).

Table 2
Singular points set II (Type I)

i j	1	2	3	4	5
1	(−1.235, −1.573, −5.000)	(−0.853, −1.809, −3.000)	(0.000, −2.000, 0.000)	(0.853, −1.809, 3.000)	(1.235, −1.573, 5.000)
2	(0.765, −3.041, −4.269)	(1.147, −2.648, −2.294)	(2.000, −1.902, −0.618)	(2.853, −0.793, 3.412)	(3.235, 0.049, 5.241)
3	(2.765, −4.211, −3.121)	(3.147, −3.227, −1.364)	(4.000, −1.618, −1.176)	(4.853, 0.299, 3.491)	(5.235, 1.666, 4.969)
4	(4.765, −4.969, −1.666)	(5.147, −3.491, −0.299)	(6.000, −1.176, −1.618)	(6.853, 1.364, 3.227)	(7.235, 3.121, 4.211)
5	(6.765, −5.241, −0.049)	(7.147, −3.412, 0.794)	(8.000,−0.618, −1.902)	(8.853, 2.294, 2.648)	(9.235, 4.269, 3.041)
6	(8.765, −5.000, 1.573)	(9.147, −3.000, 1.809)	(10.000, 0.000, 2.000)	(10.853, 3.000, 1.809)	11.235, 5.000, 1.573)
7	(10.765, −4.269, 3.041)	(11.147, −2.294, 2.648)	(12.000, 0.618, 1.902)	(12.853, 3.411, 0.794)	(13.235, 5.241, −0.049)
8	(12.765, −3.121, 4.211)	(13.147, −1.364, 3.227)	(14.000, 1.176, 1.618)	(14.853, 3.491, −0.299)	(15.235, 4.969, −1.666)
9	(14.765, −1.666, 4.969)	(15.147, −0.299, 3.491)	(16.000, 1.618, 1.176)	(16.853, 3.227, −1.364)	(17.235, 4.211, −3.121)
10	(16.765, −0.049, 5.241)	(17.147, 0.794, 3.412)	(18.000, 1.902, 0.618)	(18.853, 2.648, −2.294)	(19.235, 3.041, −4.269)
11	(18.765, 1.573, 5.000)	(19.147, 1.809, 3.000)	(20.000, 2.000, 0.000)	(20.853, 1.809, −3.000)	(21.235, 1.573, −5.000)

Consider the spherical surface shown in Fig. 8(a) represented by NURBS and the path which is in Fig. 7(b),

$$\mathbf{R}(t)=\mathbf{R}_Y(t)=\begin{bmatrix}\cos\left(\frac{\pi}{2}t\right) & 0 & \sin\left(\frac{\pi}{2}t\right)\\ 0 & 1 & 0\\ -\sin\left(\frac{\pi}{2}t\right) & 0 & \cos\left(\frac{\pi}{2}t\right)\end{bmatrix},\quad \boldsymbol{\omega}=\begin{bmatrix}0 & \frac{\pi}{2} & 0\end{bmatrix}^{\mathbf{T}}$$

and the final swept volume, in Fig. 8(g). The deformed volume is shown in Fig. 8(h).

Example 3. The surface shown in Fig. 9(a) rotates along X direction, therefore the rotation matrix is

$$\mathbf{R}(t)=\mathbf{R}_X(t)=\begin{bmatrix}1 & 0 & 0\\ 0 & \cos\left(-\frac{\pi}{3}t\right) & -\sin\left(-\frac{\pi}{3}t\right)\\ 0 & \sin\left(-\frac{\pi}{3}t\right) & \cos\left(-\frac{\pi}{3}t\right)\end{bmatrix},\quad \boldsymbol{\omega}=\begin{bmatrix}-\frac{\pi}{3} & 0 & 0\end{bmatrix}^{\mathbf{T}}.$$

The path is

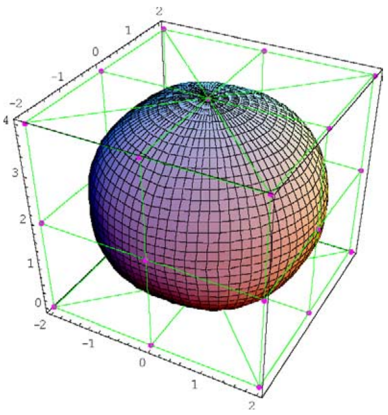
$$\boldsymbol{\Psi}(t)=\begin{bmatrix}0\\ 10t\\ -3t^2\end{bmatrix}.$$

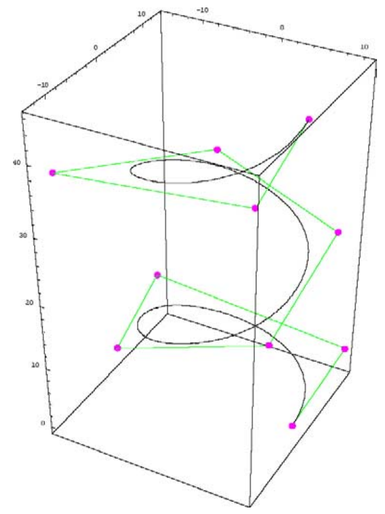
The Type I singular sets are in Fig. 9(b) and (c). The type I singular sets are in Fig. 8(b) and (c). The Type II singular sets are in Fig. 9(d)–(g) in Fig. 9(h) ($t=0$ and $t=1$). Fig. 9(k)–(m) are the different views of the final result of the NURBS surface sweeping.

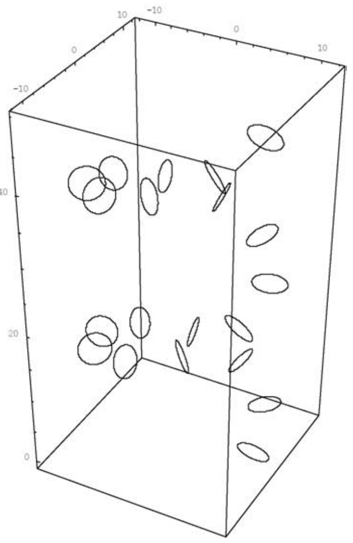
The computational errors for the above three examples are shown in Table 3.

4. Conclusions

A systematic method for characterizing and visualizing the approximate volume generated as a result of sweeping a NURBS surface (or solid) in space was presented. The NURBS surface (or solid) experiences translational and rotational motions where the envelope of every point touched by this surface denotes the swept volume. A rigorous method based upon calculating a local moving coordinate


(a)


(b)


(c)

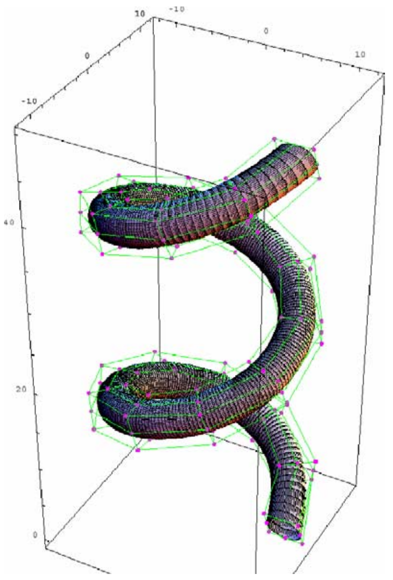
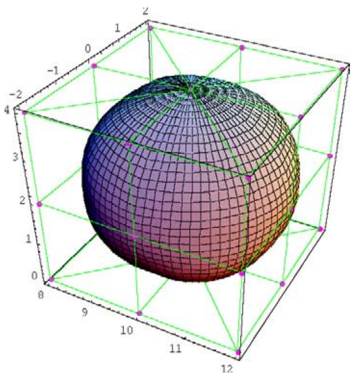
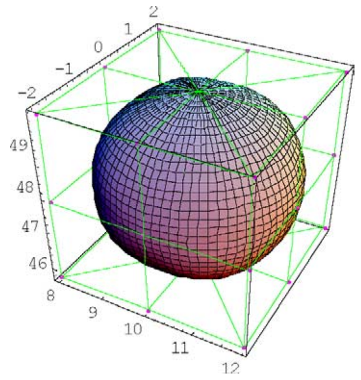

(d)

Fig. 8. (a) Solid sphere. (b) Path of NURBS sweeping. (c) Characteristic curves. (d) Type I singular surface. (e) Type II singular surface $(t = 0)$. (f) Type II singular surface $(t = 2)$. (g) Final swept volume. (h) Modified swept volume.

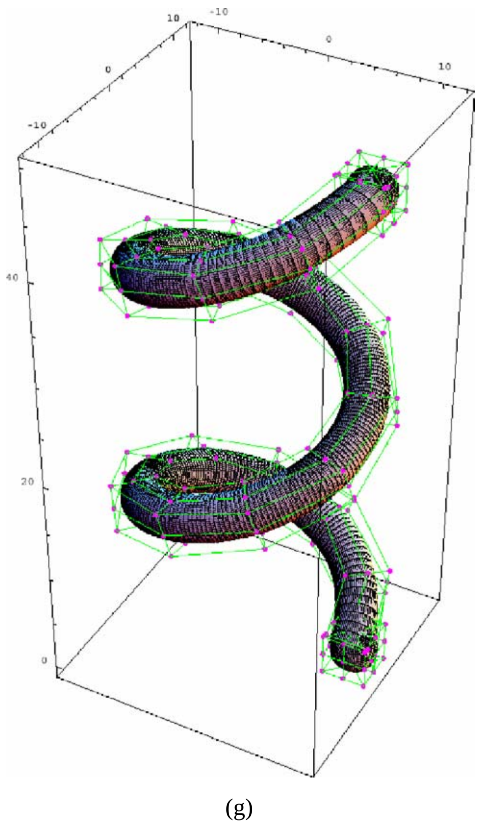
system from the velocity vector along the path was established. Local maxima points (also called singular or grazing points) were determined and used to formulate singular surfaces. It was shown that these points always exist on the edge of the rigid body in the direction of motion (although not necessarily on the boundary of the swept volume). It was shown that the direction of motion is identified as a result of combining a rigid body velocity vector (instantaneous center) and the relative velocity vector.



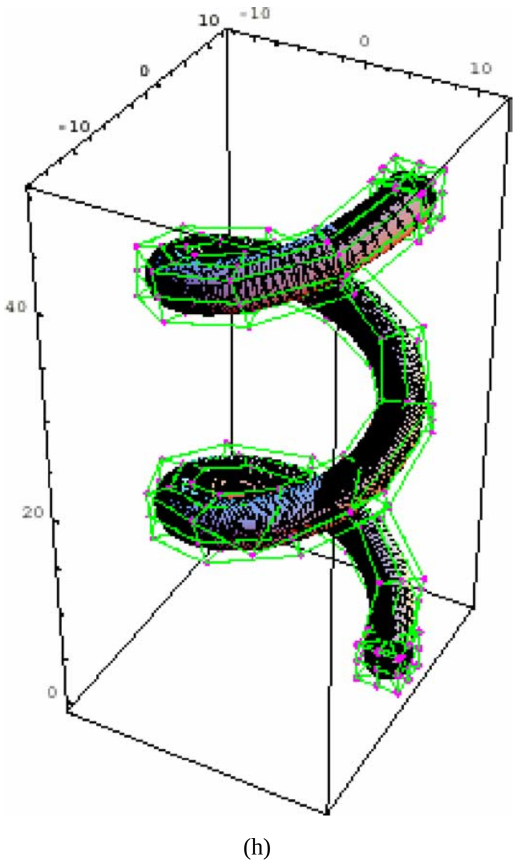
(e)



(f)



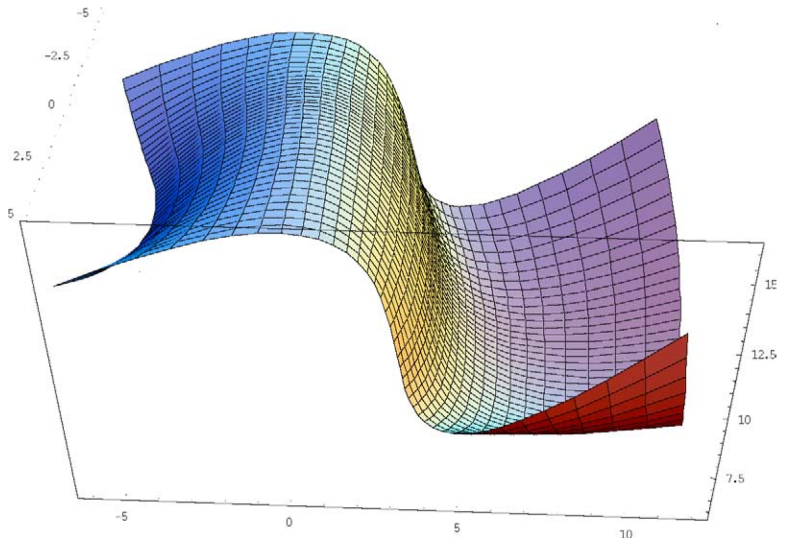
(g)



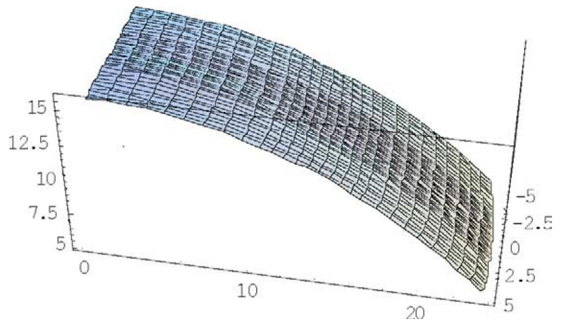
(h)

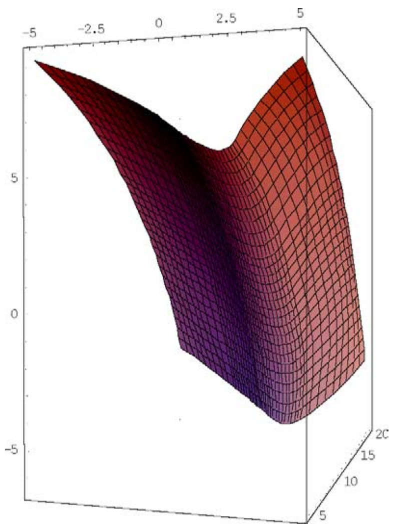
Fig. 8. (*continued*)

Trimming of curve segments (or surface patches) as one way to divide all singular curves and surfaces that are interested into sub-curves or sub-surfaces and remove the internal sub-curves or sub-surfaces to delineate those segments and patches that are on the boundary. It was shown that the proposed perturba-



(a)



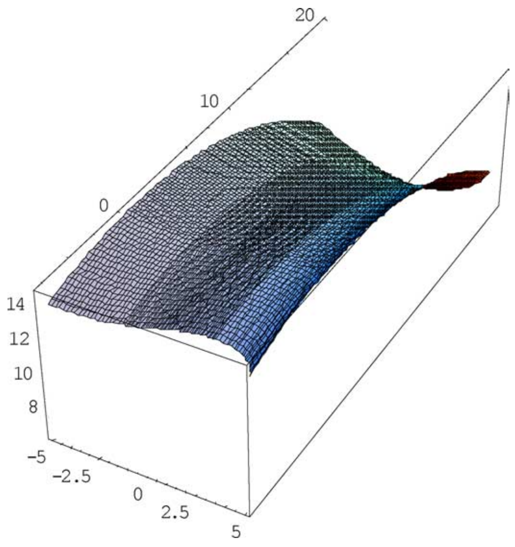


(b) (c)

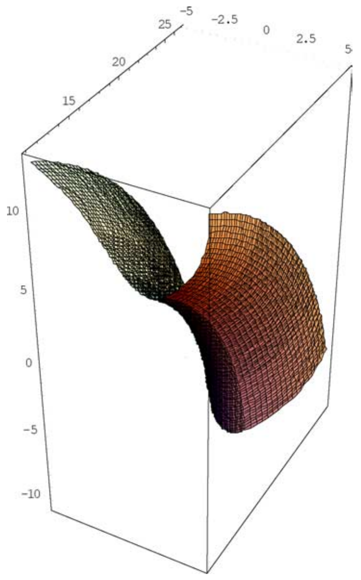
Fig. 9. Sweeping an arbitrary NURBS surface. (a) Initial surface. (b) Type I singularity (top). (c) Type I singularity (bottom). (d) Type II singularity $(v = 0)$. (e) Type II singularity $(v = 1)$. (f) Type II singularity $(u = 0)$. (g) Type II singularity $(u = 1)$. (h) Type II singularity ($t = 0$ and $t = 1$). (i) One view of the combination of (b), (c), (d), (e), (h). (j) Another view of the combination of (b), (c), (d), (e), (h). (k) One view of the final result. (l) One view of the final result. (m) One view of the final result.

tion technique can be applied to establish whether a segment (or surface patch) exists on the boundary of the swept volume.

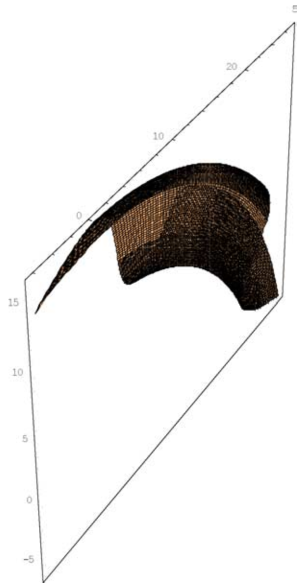
The presented method for sweeping of NURBS surfaces (or solids) was shown to be valid for any type of sweeping. The illustrative examples show the procedure of the method and the computational cost and approximation errors are discussed.



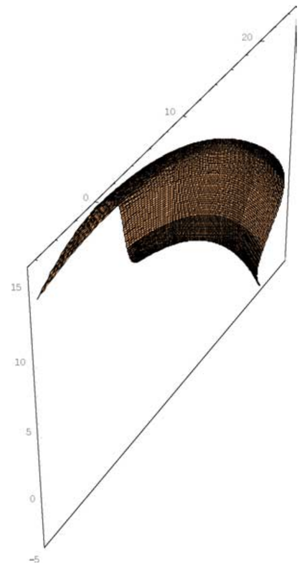
(d)



(e)



(f)

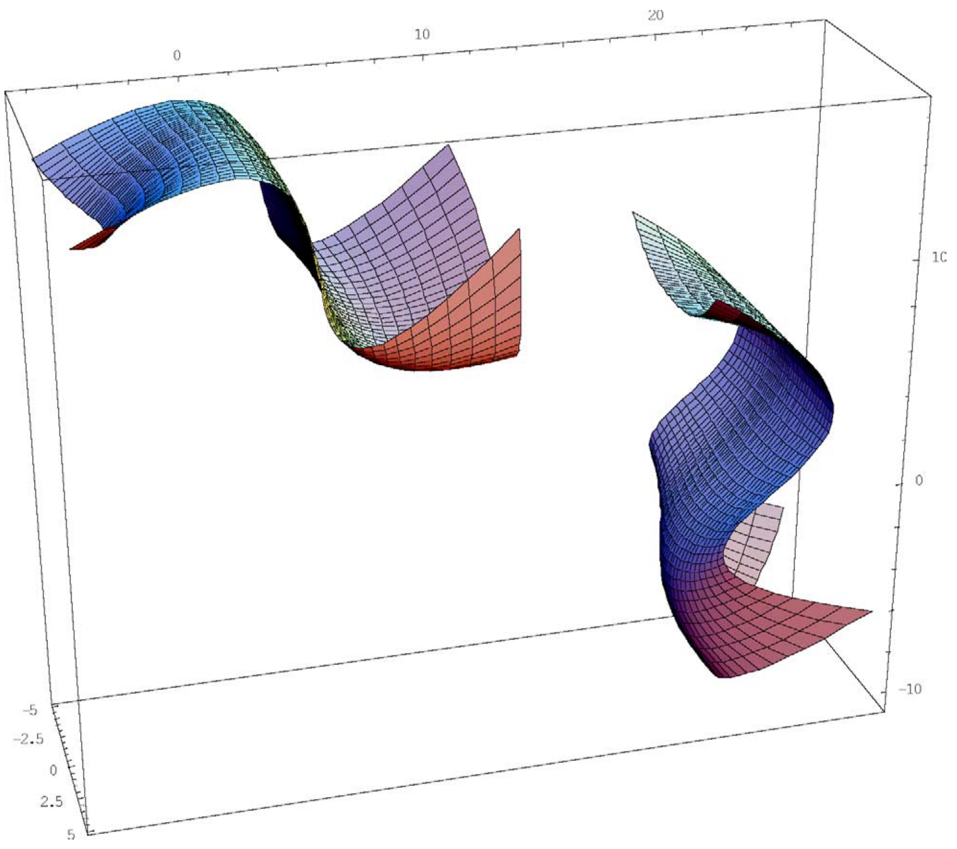


(g)

Fig. 9. (*continued*)

Acknowledgements

This research was funded by a research initiation grant from the Society of Manufacturing Engineers, a Collaborative Linkage grant from NATO, and the Iowa Space Grant Consortium. The authors appreciate the two anonymous reviewers' comments that make the improvement.



(h)

Fig. 9. (*continued*)

Appendix A

In this appendix, we summarize some basic definitions related to NURBS (Piegl and Tiller, 1997). A surface is defined by a NURBS form $\mathbf{S}(u,v)$ of degree (p,q) using the following data:

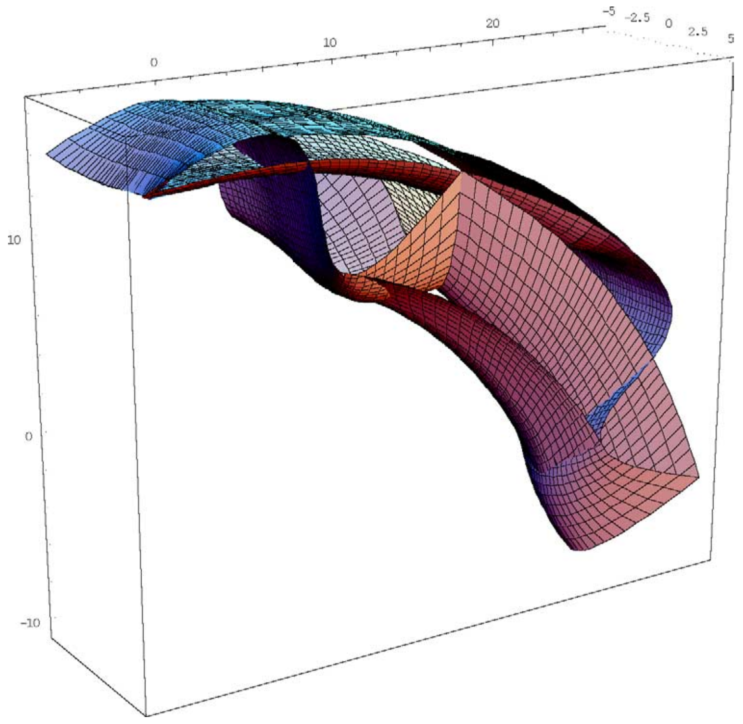
- Basis function $N_{i,p}(u)$(p is the degree), defined by a recursive Cox–deBoor algorithm

$$N_{i,0}(u)=\begin{cases}1, & u\in[u_i,u_{i+1})\\ 0, & \text{otherwise}\end{cases}$$
$$N_{i,p}(u)=\frac{u-u_i}{u_{i+p}-u_i}N_{i,p-1}(u)+\frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}}N_{i+1,p-1}(u). \tag{A.1}$$

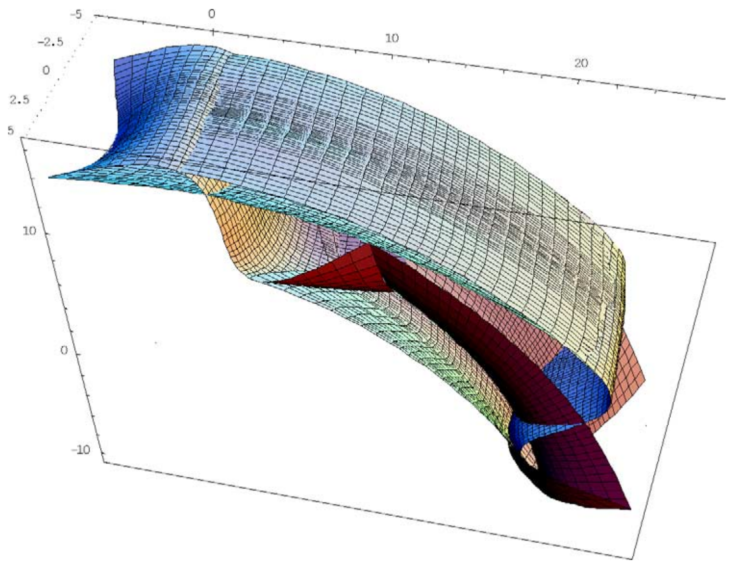
- The control points net $\mathbf{P}$, called the control polygon, where

$$\mathbf{P}=\{P_{i,j}\in\mathbb{R}^3;\ i=1,\dots,n;\ j=1,\dots,m\}, \tag{A.2}$$

  where n and m are the numbers of vertices along both directions.
- The knot vectors $\mathbf{U}$ and $\mathbf{V}$, where

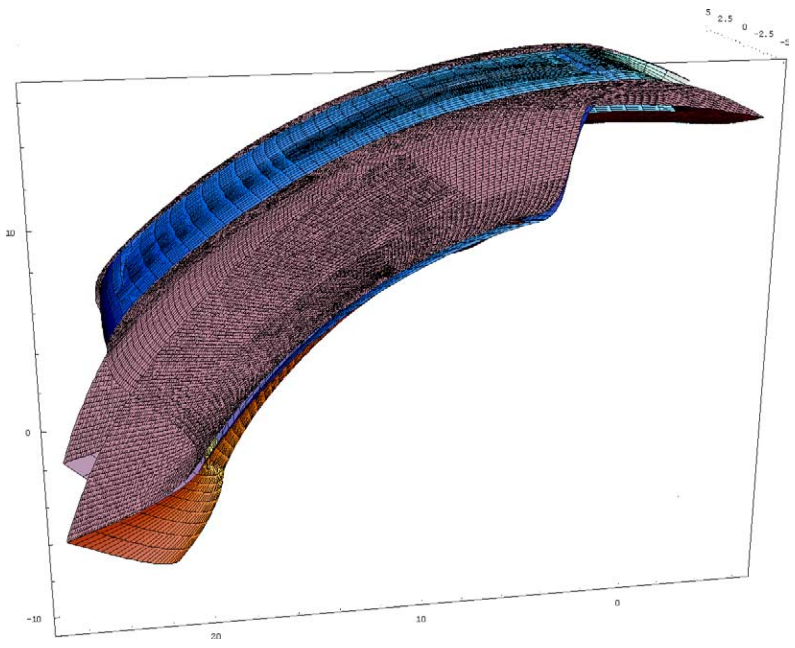


(i)

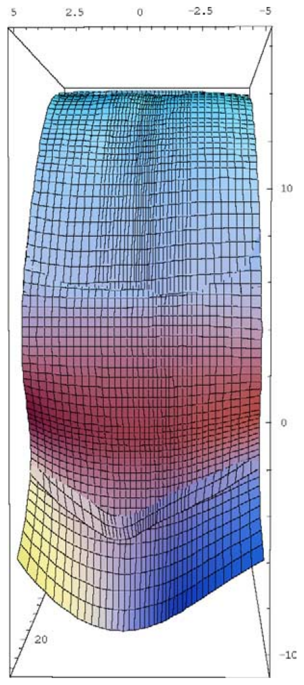


(j)

Fig. 9. (*continued*)

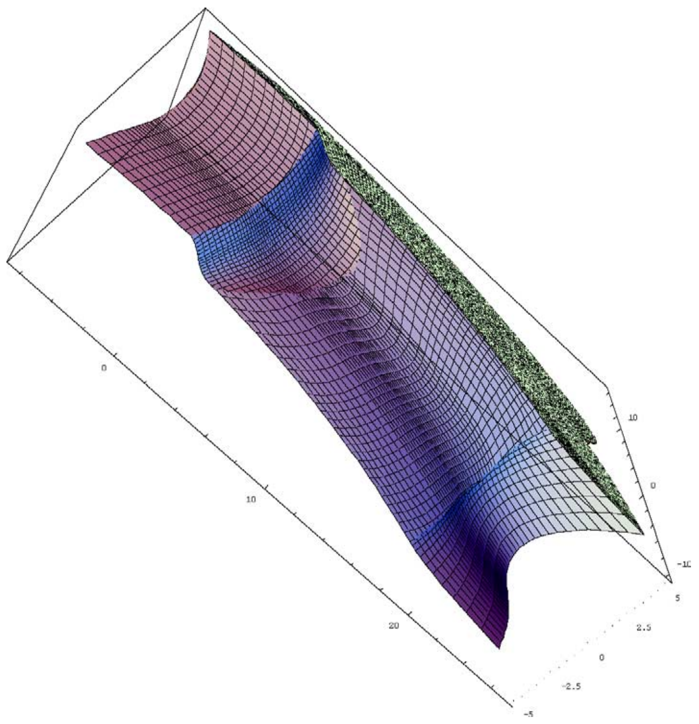


(k)



(l)

Fig. 9. (*continued*)



(m)

Fig. 9. (*continued*)

Table 3
Computational errors

Examples	η	β	Error
Example 1	5	11	0.15748
Example 2	7	20	0.36523
Example 3	10	15	0.57624

$$\mathbf{U} = \{\underbrace{e, \ldots, e}_{p+1}, u_{p+2}, \ldots, u_{r-p-1}, \underbrace{f, \ldots, f}_{p+1}\}, \tag{A.3}$$

$$\mathbf{V} = \{\underbrace{e, \ldots, e}_{q+1}, v_{q+2}, \ldots, v_{s-q-1}, \underbrace{f, \ldots, f}_{q+1}\}, \tag{A.4}$$

where p and q are the degree of the NURBS along both directions, $r = n + p + 1$ and $s = m + q + 1$.

- The weight vector $\mathbf{W}$ attributed to each control point $P_{i,j}$, where

$$\mathbf{W} = \{w_{i,j} \in \mathbb{R};\ i = 1, \ldots, n;\ j = 1, \ldots, m\}. \tag{A.5}$$

Then we define a parametric surface on the domain $u \in [e, f]$, $v \in [e, f]$ and the surface is expressed by

$$\mathbf{S}(u,v) = \frac{\sum_{i=1}^{n} \sum_{j=1}^{m} N_{i,p}(u) N_{j,q}(v) w_{i,j} \mathbf{P}_{i,j}}{\sum_{l=1}^{n} \sum_{k=1}^{m} N_{l,p}(u) N_{k,q}(v) w_{l,k}}. \tag{A.6}$$

In the same way, a curve $\mathbf{C}(u)$ of degree p defined on a knot vector $\mathbf{U}$ and by a polygon $\mathbf{P}$ is expressed by

$$\mathbf{C}(u) = \frac{\sum_{i=1}^{n} N_{i,p}(u) \omega_i \mathbf{P}_i}{\sum_{i=1}^{n} N_{i,p}(u) \omega_i}. \tag{A.7}$$

A.1. Derivative of NURBS

First consider the derivative of a NURBS $\mathbf{C}(u)$. Let $\mathbf{C}(u) = \Re(u)/\hbar(u)$, where $\Re(u) = \sum_{i=1}^{n} N_{i,p}(u) w_i \mathbf{P}_i$ and $\hbar(u) = \sum_{i=1}^{n} N_{i,p}(u) w_i$, therefore

$$\mathbf{C}'(u) = \frac{\Re'(u) - \hbar'(u)\mathbf{C}(u)}{\hbar(u)} \tag{A.8}$$

where

$$\Re'(u) = \sum_{i=1}^{n-1} N_{i,p-1}(u) p \frac{\mathbf{P}_{i+1} w_{i+1} - \mathbf{P}_i w_i}{u_{i+p+1} - u_{i+1}} \quad \text{and} \quad \hbar'(u) = \sum_{i=1}^{n-1} N_{i,p-1}(u) p \frac{w_{i+1} - w_i}{u_{i+p+1} - u_{i+1}},$$

and where $\{N_{i,p-1}\}$ is defined in $\mathbf{U}' = \{\underbrace{e, \ldots, e}_{p}, u_{p+2}, \ldots, u_{r-p-1}, \underbrace{f, \ldots, f}_{p}\}$.

The NURBS surface partial derivative can be defined as follows: let $\mathbf{S}(u,v) = \mathbf{A}(u,v)/\omega(u,v)$, where $\mathbf{A}(u,v) = \sum_{i=1}^{n} \sum_{j=1}^{m} N_{i,p}(u) N_{j,q}(v) w_{i,j} \mathbf{P}_{i,j}$, $\omega(u,v) = \sum_{l=1}^{n} \sum_{k=1}^{m} N_{l,p}(u) N_{k,q}(v) w_{l,k}$, then the partial derivative of a NURBS $\mathbf{S}(u,v)$ can be expressed by

$$\mathbf{S}_\alpha(u,v) = \frac{\mathbf{A}_\alpha(u,v) - \omega_\alpha(u,v)\mathbf{S}(u,v)}{\omega(u,v)}, \tag{A.9}$$

where α is u or v.

$$\mathbf{A}_u(u,v) = \sum_{i=1}^{n-1} \sum_{j=1}^{m} N_{i,p-1}(u) N_{j,q}(v) \mathbf{P}w_{i,j}^{(1,0)}, \tag{A.10}$$

where $\mathbf{P}w_{i,j}^{(1,0)} = p \frac{w_{i+1,j}\mathbf{P}_{i+1,j} - w_{i,j}\mathbf{P}_{i,j}}{u_{i+p+1} - u_{i+1}}$, $\mathbf{U}^{(1)} = \{\underbrace{e, \ldots, e}_{p}, u_{p+2}, \ldots, u_{r-p-1}, \underbrace{f, \ldots, f}_{p}\}$, $\mathbf{V}^{(0)} = \mathbf{V}$.

$$\mathbf{A}_v(u,v) = \sum_{i=1}^{n} \sum_{j=1}^{m-1} N_{i,p}(u) N_{j,q-1}(v) \mathbf{P}w_{i,j}^{(0,1)}, \tag{A.11}$$

where $\mathbf{P}w_{i,j}^{(0,1)} = q \frac{w_{i,j+1}\mathbf{P}_{i,j+1} - w_{i,j}\mathbf{P}_{i,j}}{v_{j+q+1} - v_{j+1}}$, $\mathbf{U}^{(0)} = \mathbf{U}$, $\mathbf{V}^{(1)} = \{\underbrace{e, \ldots, e}_{q}, v_{q+2}, \ldots, v_{s-q-1}, \underbrace{f, \ldots, f}_{q}\}$.

$$\omega_u(u,v) = \sum_{i=1}^{n-1} \sum_{j=1}^{m} N_{i,p-1}(u) N_{j,q}(v) w_{i,j}^{(1,0)}, \tag{A.12}$$

where $w_{i,j}^{(1,0)} = p\frac{w_{i+1,j}-w_{i,j}}{u_{i+p+1}-u_{i+1}}$, $\mathbf{U}^{(1)} = \{\underbrace{e,\ldots,e}_{p}, u_{p+2},\ldots,u_{r-p-1}, \underbrace{f,\ldots,f}_{p}\}$, $\mathbf{V}^{(0)} = \mathbf{V}$.

$$\omega_v(u,v) = \sum_{i=1}^{n}\sum_{j=1}^{m-1} N_{i,p}(u)N_{j,q-1}(v)w_{i,j}^{(0,1)}, \tag{A.13}$$

where $w_{i,j}^{(0,1)} = q\frac{w_{i,j+1}-w_{i,j}}{v_{j+q+1}-v_{j+1}}$, $\mathbf{U}^{(0)} = \mathbf{U}$, $\mathbf{V}^{(1)} = \{\underbrace{e,\ldots,e}_{q}, v_{q+2},\ldots,v_{s-q-1}, \underbrace{f,\ldots,f}_{q}\}$.

References

Abdel-Malek, K., Yeh, H.J., 1997. Geometric representation of the swept volume using Jacobian rank-deficiency conditions. Computer-Aided Design 29 (6), 457–468.

Abdel-Malek, K., Othman, S., 1999. Multiple sweeping using the Ddenavit–Hartenberg representation method. Computer-Aided Design 31, 567–583.

Abdel-Malek, K., Yang, J., 2000. Method and code for the visualization of multivariate solids. In: Proceedings of the 26th ASME Design Automation Conference, Baltimore, MD.

Abdel-Malek, K., Yang, J., Blackmore, D., 2000a. Closed-form swept volume of implicit surfaces. In: Proceedings of the 26th ASME Design Automation Conference, Baltimore, MD.

Abdel-Malek, K., Blackmore, D., Joy, K.I., 2000b. Swept volumes: foundations, perspectives and applications. Internat. J. Shape Modeling, submitted for publication.

Abdel-Malek, K., Yeh, H.J., Othman, S., 2000c. Interior and exterior boundaries to the workspace of mechanical manipulators. Robotics and Computer-Integrated Manufacturing 16 (5), 365–376.

Abdel-Malek, K., Yang, J., Blackmore, D., 2001. On swept volume formulations: implicit surfaces. Computer-Aided Design 33 (1), 113–121.

Ahn, J.C., Kim, M.S., Lim, S.B., 1993. Approximate general sweep boundary of a 2D curved object. CVGIP: Computer Vision Graph and Image Process. 55, 98–128.

Akman, V., Arslan, A., 1992. Sweeping with all graphical ingredients in a topological picturebook. Computers and Graph 16, 273–281.

Ball, A.A., 1974. CONSURF. Part 1: Introduction to conic lofting tile. Computer-Aided Design 7 (4), 243–249.

Ball, A.A., 1975. CONSURF. Part 2: Design of the algorithms. Computer-Aided Design 9 (1), 237–242.

Ball, A.A., 1977. CONSURF. Part 3: How the program is used. Computer-Aided Design 9 (1), 9–12.

Blackmore, D., Leu, M.C., 1992. Analysis of swept volume via Lie groups and differential equations. Internat. J. Robotics Res. 11 (6), 516–537.

Blackmore, D., Jiang, H., Leu, M.C., 1992. Modeling of swept solids using sweep differential techniques. In: Proc. 4th IFIP WG5.2 Workshop Geometric Modeling in Computer-Aided Design, Rensselaerville, New York.

Blackmore, D., Leu, M.C., Frank, S., 1994. Analysis and modeling of deformed swept volumes. Computer-Aided Design 25 (4), 315–326.

Blackmore, D., Leu, M.C., Wang, L.P., Jiang, H., 1997a. Swept volumes: a retrospective and prospective view. Neural Parallel and Scientific Computations 5, 81–102.

Blackmore, D., Leu, M.C., Wang, L.P., 1997b. Sweep-envelope differential equation algorithm and its application to NC machining verification. Computer-Aided Design 29 (9), 629–637.

Blackmore, D., Samulyak, R., Leu, M.C., 1999. Trimming swept volumes. Computer-Aided Design 31 (3), 215–224.

Bloomenthal, M., Riessenfeld, R.F., 1991. Approximation of sweep surfaces by tensor product NURBS. Curves and Surfaces in Computer Vision and Graphics II, SPIE Proc. 1610, 132–144.

Cohen, E., Riesenfeld, R., Elber, G., 2001. Geometric Modeling with Splines: An Introduction. A.K. Peters.

Coquillart, S., 1987. A control-point-based sweeping technique. IEEE Comput. Graph. Appl. 7 (11), 36–45.

Elber, G., 1997. Global error bounds and amelioration of sweep surfaces. Computer-Aided Design 29, 441–447.

Farin, G., 1993. Curves and Surfaces for CAGD, A Practical Guide. Academic Press.

Filip, D., Ball, T., 1989. Procedurally representing lofted surfaces. IEEE Comput. Graph. Appl. 9 (6), 27–33.

Hohmeyer, M., Barsky, B., 1991. Skinning rational B-spline curves to construct an interpolatory surface. Comput. Vis. Graph. Image Processing: Graphical Models and Image Processing 53 (6), 511–521.

Hu, Z.J., Ling, Z.K., Shene, C.K., 1997. Modeling of a general sweep operation and its implementation. Preprint.

Jiang, H., Blackmore, D., Leu, M.C., 1994. The flow approach to CAD/CAM modeling of swept volumes. In: Manufacturing Systems: Design, Modeling and Analysis. Elsevier, Amsterdam, pp. 341–346.

Joy, K.I., 1992. Visualization of swept hyperpatch solids. In: Kunii, T.L. (Ed.), Visual Computing: Integrating Computer Graphics with Computer Vision (Proceedings of Computer Graphics International 92). Springer-Verlag, Tokyo, pp. 567–582.

Joy, K.I., Duchaineau, M.A., 1999. Boundary determination for trivariate solids. In: Proceedings of Pacific Graphics ’99, pp. 82–91.

Levin, A., 1999. Combined subdivision schemes and their applications to 3D modeling, PhD Thesis, Tel-Aviv University, Israel.

Ling, Z.K., Chase, T., 1996. Generating the swept area of a body undergoing planar motion. J. Mech. Design 118, 221–233.

Litke, N., Levin, A., Schroder, P., 2001. Trimming for dubdivision surfaces. Computer Aided Geometric Design 18 (5), 463–481.

Madrigal, C., Joy, K.I., 1999. Generating the envelope of a swept trivariate solid. In: Proceedings of the IASTED International Conference on Computer Graphics and Imaging, Palm Springs, CA, pp. 5–9.

Marisam, R., Menon, J., Voelcker, H.B., 1992. CAD/CAM applications of ray representations. In: Proc. 1992 NSF Design & Manufacturing Systems Conf., pp. 113–119.

Piegl, L., Tiller, W., 1997. The NURBS Book. Springer-Verlag.

Roth, D., Bedi, S., Ismail, F., Mann, S., 2001. Surface swept by a toroidal cutter during 5-axis machining. Computer-Aided Design 33, 57–63.

Terzopoulos, D., Qin, H., 1994. Dynamic NURBS with geometric constraints for interactive sculpting. ACM Trans. Graph. 13 (2), 103–136.

Till, W., 1983. Rational B-splines for curve and surface representation. IEEE Comput. Graph. Appl. 3 (6), 61–69.

Woodward, C., 1987. Cross-sectional design of B-spline surfaces. Computer and Graph. 11 (2), 193–201.

Woodward, C., 1988. Skinning techniques for interactive B-spline surface interpolation. Computer-Aided Design 20 (8), 441–451.

Xia, J., Ge, J., 2000. Kinematic approximation of ruled surfaces using NURBS motions of a cylindrical cutter. In: Proceedings of the ASME Design Automation Conference, Baltimore, MD.

Further reading

Gray, A., 1997. Modern Differential Geometry of Curves and Surfaces with Mathematica, second ed. CRC Press.