Journal of Zhejiang University SCIENCE A
ISSN 1673-565X (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Trimming self-intersections in swept volume solid modeling*

Zhi-qi XU†1, Xiu-zi YE[1], Zhi-yang CHEN[2], Yin ZHANG[1], San-yuan ZHANG†‡1

([1]*State Key Laboratory of CAD & CG, School of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China*)

([2]*College of Software, Zhejiang University of Technology, Hangzhou 310014, China*)

†E-mail: learlynn@zju.edu.cn; syzhang@cs.zju.edu.cn

Received July 2, 2007; revision accepted Dec. 12, 2007

Abstract: Swept volume solid modeling has been applied to many areas such as NC machining simulation and verification, robot workspace analysis, collision detection, and CAD. But self-intersections continue to be a challenging problem in the boundary representation of swept volume solids. A novel algorithm is presented in this paper to trim self-intersection regions in swept volume solids modeling. This trimming algorithm consists of two major steps: (1) roughly detecting self-intersection regions by checking intersections or overlapping of the envelop profiles; (2) splitting the whole envelop surfaces of the swept volume solid into separate non-self-intersecting patches to trim global self-intersections, and to trim local self-intersections, dividing local self-intersecting regions into patches and replacing self-intersecting patches with non-self-intersecting ones. Examples show that our algorithm is efficient and robust.

Key words: Self-intersection, Swept volume (SV), Solid modeling
doi:10.1631/jzus.A071357 **Document code:** A **CLC number:** TP391

INTRODUCTION

Swept volume (SV) is defined as the space occupied by sweeping a solid or a collection of surfaces along an arbitrary trajectory (Kim *et al*., 2004), and it has been applied to many areas such as geometric modeling, NC machining simulation and verification, robot workspace, and collision detection (Blackmore *et al*., 1997b; Abdel-Malek *et al*., 2006).

The problem of SV is developed with many disciplines including envelop theory (Wang and Wang, 1986; Martin and Stephenson, 1990; Weld and Leu, 1990; Pottmann and Peternell, 2000), sweep differential equations (SDE) (Blackmore and Leu, 1992; Blackmore *et al*., 1997a) and Jacobian rank-deficiency method (Abdel-Malek and Yeh, 1997). Envelop theory is one of the most fundamental formulations for SV problems, and states that the SV of a compact n-manifold in $\mathbb{R}^n$ is generated by combining the SVs of the boundaries of this manifold at each location in the sweep. The SV problem is also developed in parallel with many solid modeling techniques such as boundary representation (B-Rep), spatial partitioning and constructive solid geometry (Requicha and Voelcker, 1982). B-Rep is one of the most widely used modeling techniques, because it supports a variety of mathematical surfaces including Bezier, spline and NURBS. Based on both envelop theory and B-rep method, the envelop of the space occupied by the SV can be identified as a solid, called "SV solid".

However, one challenging problem may often arise during the SV solid modeling, which is that the SV envelop surfaces may self-intersect. It will result in non-manifold boundary surfaces of SV solids and is one of the most difficult aspects for analyzing and representing SVs. The problem of self-intersections in SV solid modeling has been largely overlooked in literature due to its difficulty, but it is still very desirable to remove all unwanted inner pieces of the envelop surfaces inside the SV, especially in the

‡ Corresponding author

* Project supported by the National Natural Science Foundation of China (No. 60473106), the Hi-Tech Research and Development Program (863) of China (Nos. 2007AA01Z311 and 2007AA04Z1A5), and the National Research Foundation for the Doctoral Program of Higher Education of China (No. 20060335114)

context of a boundary modeler which demands all handled objects be manifolds (Blackmore *et al*., 1999; Rossignac *et al*., 2007).

In this paper, a novel algorithm is proposed for trimming self-intersections in SV solid modeling of revolutions. In Section 2, we will briefly introduce the definition of SV problems and the procedures of modeling an SV solid. The new detection-and-trimming method of self-intersections in SV solid modeling will be presented in detail with examples in Sections 3 and 4. A conclusion will be given in Section 5.

SV SOLID MODELING

NC machining tools are or can be approximated by revolved solids in machining simulation and verification, and geometric modeling applications. So, in this paper we focus on the self-intersection-trimming of SVs generated by revolved solids along curved swept trajectories.

Let $M\subset\mathbb{R}^3$ be the revolution generator solid to be swept in $\mathbb{R}^3$. Let $\tau(t)$ be a composite curve as the sweeping trajectory curve in $\mathbb{R}^3$, and $\boldsymbol{R}(t)$ be a time-varying matrix from $\tau(t)$ which presents the transformation from the start reference frame to the reference frame at time t, where t is the single time variable normalized to vary in [0, 1]. Here, $\boldsymbol{R}(0)$ corresponds to the identity matrix. The sweeping of the generator solid M can be described by the following formula:

$$M(t)=\tau(t)+\boldsymbol{R}(t)\cdot M. \tag{1}$$

The SV solid of M can be defined as

$$SV(M)=\{\cup M(t)|t\in[0, 1]\}, \tag{2}$$

where $\cup$ represents the union of moving generators.

The envelop of the SV solid can be defined as the union of all envelop profiles of the moving generator at each position over the entire time interval as follows:

$$\partial SV(M)=\partial_{-}M(0)\cup\partial_{+}M(1)\cup\{\partial_0 M(t): t\in[0, 1]\}. \tag{3}$$

$\partial_{-}M(0)$ consists of the ingress surface patches of M at the start position of the motion; $\partial_{+}M(1)$ consists of the egress surface patches of M at the end position of the motion. The envelop profile $\partial_0 M(t)$ of SV is extracted from $M(t)$ at time t (Yu and Wang, 2003). In this paper, the envelop profile $\partial_0 M(t)$ is considered as a composite curve, which is named as $C(t)$. As presented in the envelop theory, the SV envelop surface is created through the envelop profile curve set $\{C(t)\}$, which means that the envelop surface and the intermediate generator solid $M(t)$ tangentially touch at the envelop profile curve $C(t)$ at a given time t. So if a point P lies on $C(t)$, it must also both lie on the envelop surface and $M(t)$, and its velocity vector at time t must be in the tangent plane of the SV envelop surface at this point. That is to say,

$$\boldsymbol{V}_P\cdot\boldsymbol{N}_P=0, \tag{4}$$

where $\boldsymbol{V}_P$ is the instantaneous velocity vector of P; $\boldsymbol{N}_P$ is the normal vector of the SV envelop surface at P and it can also be evaluated from the boundary surface of the moving generator solid $M(t)$ at P (Blackmore *et al*., 1997a; Piegl and Tiller, 1997; Mann and Bedi, 2002).

According to the above definitions of SV problems, the whole SV solid modeling can be presented in the following steps:

Step 1: Discretize the trajectory curve into m points according to the curvature variations, denote those points as P_0, P_1, P_2, …, P_{m-1}, and set up reference frames F_i (i=0, 1, 2, …, m−1) at each discrete point P_i (Fig.1). In order to avoid a twisted or self-intersecting SV solid at most, proper reference frames are important. So the modified Frenet Frames (Xu *et al*., 2007) are adopted in this paper.

(a) (b)

Fig.1 Step 1: (a) Discrete points P_0, P_1, P_2, …, P_{m-1} on the trajectory and corresponding reference frames on the points; (b) Magnified end of the trajectory

Step 2: Transform the generator within those reference frames. We use the following equation to describe the transform:

$$F_i=\boldsymbol{A}_i\cdot F_0, \tag{5}$$

where F_0 is the first frame set up at the start of the trajectory, F_i is the local frame at the ith discrete position P_i, and $\boldsymbol{A}_i$ is the transform matrix from F_0 to F_i (Peternell *et al*., 2005).

Then calculate the velocity vectors of the generator at each discrete position. According to Eq.(5), the sweeping of a point x on the generator M can be presented as

$$x_i=\boldsymbol{A}_i\cdot x, \tag{6}$$

where x_i is the point on the moving generator M_i transformed from the start position into the ith discrete position, and the motion of x can be taken as a trajectory. So the velocity vector $\boldsymbol{V}_{x,i}$ of x at the ith discrete position is just the tangent vector of this point on the corresponding sweeping trajectory (Mann and Bedi, 2002):

$$\boldsymbol{V}_{x,i}=\dot{x}_i=\dot{\boldsymbol{A}}_i\cdot x. \tag{7}$$

At last, extract and approximate the envelop profile curve $C(t)$. Firstly, we tessellate the rotation axis Ax_i of the moving revolved generator M_i into n points $\{Ap_j \mid j=0, 1, 2, \ldots, n-1\}$ at each local frame, make a plane through the point Ap_j and with the normal of Ax_i, intersect this plane with the boundary surfaces of M_i to get a circle SC_j, calculate the velocity vector $\boldsymbol{V}_j$ at the center point of a sphere which passes SC_j, make a plane through the spherical center point and with the normal of $\boldsymbol{V}_j$, and then intersect this plane with $\boldsymbol{V}_j$ to obtain the grazing points $\{G_{k,j} \mid j=0, 1, 2, \ldots, n-1; k=0, 1, 2\}$. The envelop profile curve $C(t)$ is the approximation of B-spline curves through these grazing points (Fig.2). The details refer to our earlier paper (Xu *et al*., 2007).

Step 3: Suppose the extracted envelop profile $C(t)$ is a composite of B-spline curves, we can approximate a B-spline envelop surface $S(u, v)$ (skinned surface) by lofting through the profile curve set $\{C(t)\}$ (Piegl and Tiller, 1997) (Fig.3a).

Based on B-splines, the definition of $S(u, v)$ is as follows.

It is known that $S(u, v)$ touches the moving generator at the profile curve $C(t)$ at any given time t, so points on the profile curve $C(t)$ must also lie on the surface $S(u, v)$. Suppose the composite profile curve $C(t)$ at time t can be represented as follows:

$$C_t(u)=\sum_{i=0}^{n} R_{i,p}(u)P_{i,t}. \tag{8}$$

The approximation of the envelop surface is a process of blending the profile curves together to create a surface. Let

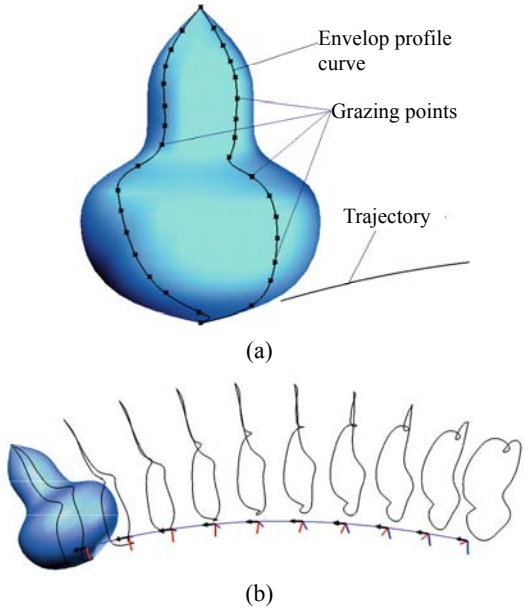

Fig.2 Step 2: (a) Grazing points and the approximated envelop profile curve on the generator at the start position of the trajectory; (b) Envelop profile curves at each position

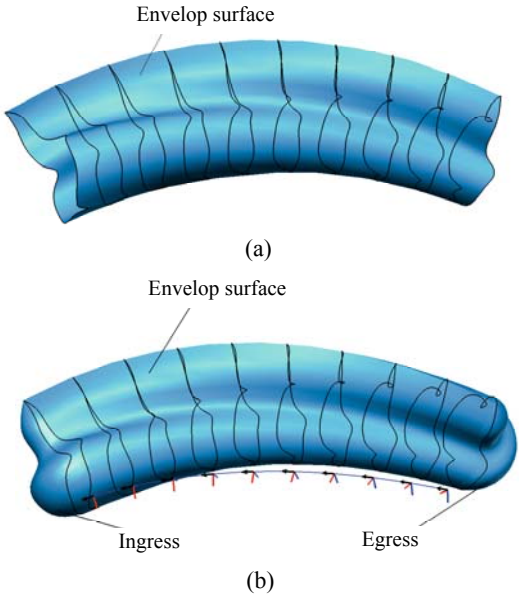

Fig.3 Step 3: (a) Envelop surface created by lofting all the profile curves; (b) The whole SV solid which combines the envelop surface with the ingress/egress surfaces

$$S(u,v)=\sum_{i=0}^{n}\sum_{j=0}^{m}R_{i,j,p,q}(u,v)\cdot P_{i,j}, \tag{9}$$

where $\{R_{i,j,p,q}(u, v)\}$ are the rational B-spline basis functions (Piegl and Tiller, 1997).

Let $S(u,v)$ be given by $C_t(u)$:

$$S(u, v_t)=C_t(u). \tag{10}$$

Here v_t is a set of unknown variables. From Eq.(9), we can get

$$S(u,v)=\sum_{i=0}^{n}R_{i,p}\left(\sum_{j=0}^{m}R_{j,q}(v)P_{i,j}\right), \tag{11}$$

if

$$Q_i(v)=\sum_{j=0}^{m}R_{j,q}(v)\cdot P_{i,j}, \tag{12}$$

and

$$Q_i(\overline{v}_j)=P_{i,j},\ i=0,1,\ldots,n;\ j=0,1,\ldots,m. \tag{13}$$

Eqs.(8)~(13) are used to generate m+1 interpolations across the control points of the swept profile curves, yielding the control points $Q_{i,j}$ of the envelope surface. Then, the V direction interpolations through $P_{i,t}$ are carried out in four-dimensional space. Finally, Eq.(11) can represent the approximated envelope surface (Weinert *et al*., 2004).

Then the envelop surface is joined with the ingress and egress surfaces to form the SV solid.

ROUGH DETECTION OF SELF-INTERSECTIONS

One primary causation of self-intersections in SV solid modeling is the intersection or overlapping of the envelop profiles. So in this section a checking method is presented to roughly detect self-intersection.

As mentioned in the previous section, the trajectory curve is tessellated into m points as P_0, P_1, P_2, …, P_{m-1}, and a moving reference frame F_i is set up at the corresponding discrete position of P_i. Then all envelop profiles are extracted at each reference frame, and stored into the profile curve set $C=\{C_i|i=0, 1, 2, \ldots, m-1\}$ in order. The worst case of the computational complexity for this checking method is $O(m^2)$.

Let F_i be the ith frame and F_j the jth frame ($j\neq i$). Suppose the envelop profile curve C_i at frame F_i is approximated through the grazing point set $G_i=\{G_{k,l,i}|l=0, 1, \ldots, n-1; k=0, 1, 2\}$ (here n is the number of tessellation points on the generator rotation axis), and correspondingly the profile C_j at frame F_j is approximated through the grazing point set $G_j=\{G_{k,l,j}|l=0, 1, \ldots, n-1; k=0, 1, 2\}$. If either of these two envelop profiles intersects itself, there must be self-intersections in the SV envelop surfaces and this profile curve is marked and put into a special subset $X=\{C_i|i=0, 1, 2, \ldots\}$ in the position order and $X\subset C$. If neither of these two envelop profile curves intersects itself, they should also be checked whether they intersect or overlap. There are three cases in this checking:

Case 1 Both of the two envelop profile curves are planar curves.

Case 2 One of the two envelop profile curves is a planar curve, while the other is not.

Case 3 Neither of the two envelop profile curves is a planar curve.

In Case 1, suppose the plane bounded by C_i is α_i, and the plane bounded by C_j is α_j. If α_i and α_j have no intersections, there must be no self-intersections between these two profiles. If the two planes intersect, the intersection points between planes α_i and α_j are denoted as $Q_{i,j}=\{Q_{k,i,j}|k=0, 1, 2\}$, and the intersection curves are denoted as $L_{i,j}=\{L_{k,i,j}|k=0, 1, 2, \ldots\}$. If the count of $Q_{i,j}$ is non-zero, or the count of $L_{i,j}$ is positive, there must be self-intersections, as shown in Fig.4. If there is confirmed intersection or overlapping, these two profile curves are marked and put into the subset X in the order of their positions in the set C.

In Case 2, suppose the plane bounded by C_i is α_i, then the intersection points between the non-planar profile curve C_j and α_i are marked as $Q_j=\{Q_{k,j}|k=0, 1, 2\}$, and the intersection curves are marked as $L_j=\{L_{k,j}|k=0, 1, 2, \ldots\}$. If either the count of Q_j or the count of L_j is larger than zero, there must be self-intersections, as shown in Fig.5. These two profile curves are also marked and put into the subset X in the order of their positions in the set C.

In Case 3, as it is difficult to judge whether two non-planar curves intersect, a rough but fast method is proposed for the rough intersection checking. As introduced above, the envelop profile curve C_i is approximated through the grazing points G_i. Hence

under a given tolerance, C_i can be considered as a polyline PL_i and the polyline PL_j is obtained from C_j through the grazing points G_j. Two triangular facets through two pairs of grazing points are created as the intersection checking areas (Fig.6b). If those triangular facets from C_i have intersections with the polyline PL_j, there must be self-intersections. If there is no intersection, the intersections between triangular facets from C_j and the polyline PL_i will also be checked. If either intersection exists, both the envelop profile curves C_i and C_j will be marked and put into the subset X in order. The pseudo-code of checking intersections between C_i and PL_j is shown in Algorithm 1. Intersections between C_j and PL_i are checked in the same way. If there are n' points in PL_j, the computational complexity of this algorithm is $O(n\cdot(n'-1))$.

Almost all self-intersections can be detected by checking above three cases. However, as envelop profile curves are extracted at discrete frames with a provided tolerance, even all of them have passed the above checks, there might still be self-intersections in the envelop surfaces. So after the envelop surfaces are created by blending all the profile curves, a more exact self-intersection detection, which is represented particularly in (Andersson *et al*., 1998; Cohen and Ho, 2000), is performed on these surfaces.

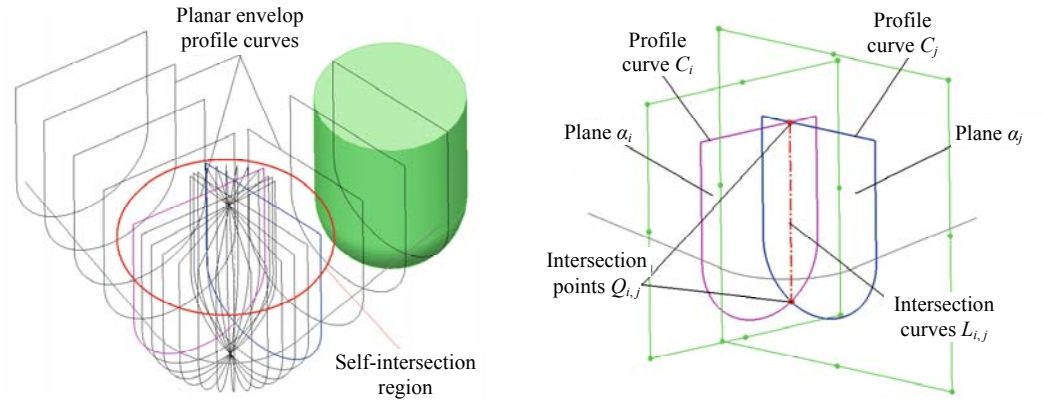

Fig.4 Case 1: (a) All the envelop profiles are planar and a self-intersection is detected; (b) A magnified part of the self-intersection region. As detected, there are intersection points and an intersection curve

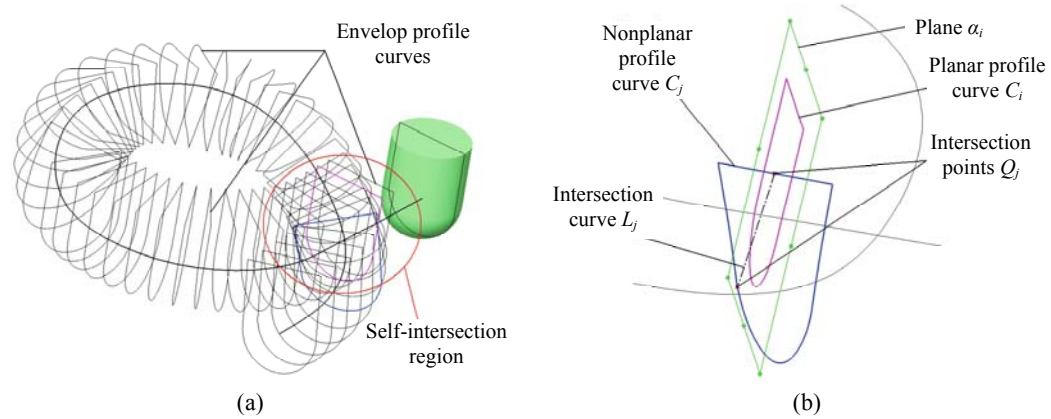

Fig.5 Case 2: (a) Self-intersection is detected by checking the intersection of two profile curves, one is planar and the other is not; (b) A magnified part of the self-intersection region. As detected, there are intersection points and an intersection curve

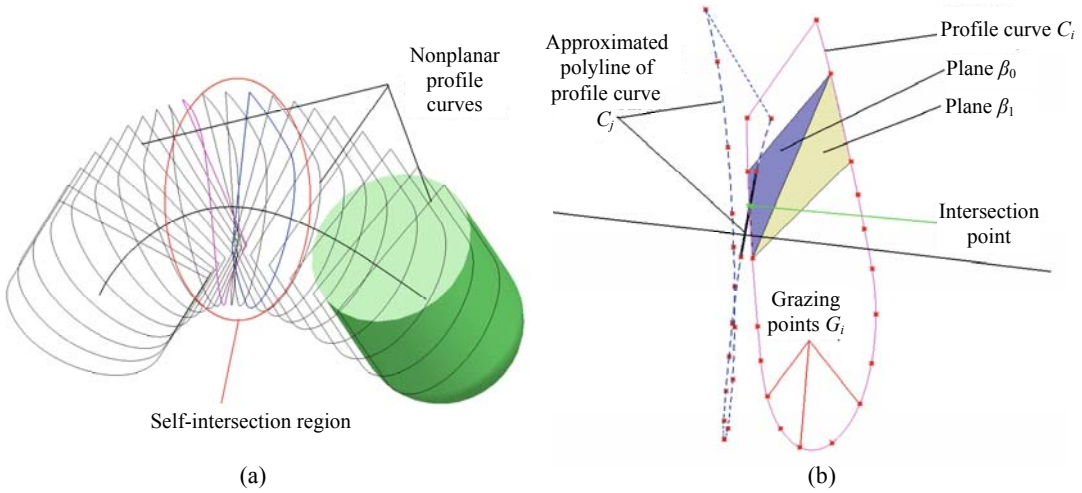

Fig.6 Case 3: (a) Self-intersection is detected by checking the intersection of two profile curves; (b) A magnified part of the self-intersection region. As detected, there are intersections between the triangular facets from C_i and the polyline PL_j

Algorithm 1 Fast check of intersections between C_i and PL_j (Intersections between C_j and PL_i are checked in the same way)

Input:

G_i: Grazing points of C_i, here $G_i=\{G_{k,j,i}|k=0, 1, 2; j=0, 1, \ldots, n-1\}$;

PL_j: Polyline from C_j.

Output:

iIntersected: A Boolean value, 'true' means the two profile curves intersect.

```
Begin
  For j=0 to j=n−2
    Set a subset G_{j,i}={G_{k,j,i}|k=0, 1, 2}, G_{j,i}⊂G_i;
    If the count of G_{j,i} is 0
      Continue;
    End If
    For t=j+1 to n−1
      Set a subset G_{t,i}={G_{k,t,i}|k=0, 1, 2}, G_{t,i}⊂G_i;
      If the count of G_{t,i} is 0
        Continue;
      End If
      Create a pair of triangular facets β0 and β1 through
        points G_{j,i} and G_{t,i};
      If β0 or β1 intersects with the polyline PL_j
        iIntersected=true;
        Break;
      EndIf
      j=t;
      Break;
    End
    If (iIntersected==true)
    Break;
  End
End
```

TRIMMING SELF-INTERSECTIONS

In the context of representing boundaries of SV solids, the envelop surfaces can be considered as a collection of properly-joined single-component patches. So trimming self-intersections out in SV solids can be taken as removing self-intersection patches (Andersson *et al*., 1998).

After the envelop surface is created according to Eq.(10), local and global self-intersection regions of SV envelop surfaces are trimmed in separate ways with a full utilization of the evaluated subset X. All surfaces referred to in this paper are presented as NURBS surfaces.

Global self-intersections

Global self-intersections in SV envelop surfaces always result from the intersections or overlapping of two discontinuous envelop profiles denoted as X_i and X_j (X_i, $X_j \in X$) as shown in Fig.7.

Suppose the profile curve X_i is extracted at the discrete position P_i and X_j is extracted at the position P_j. Here P_i and P_j are not in the given parameter interval named as Δ_t of the trajectory. Two smallest subsets Δ_i and Δ_j of the envelop profile curve set C are calculated around X_i and X_j,

$$\begin{cases} \Delta_i = \{C_k \mid k \in [i-\alpha, i+\beta],\ \ \alpha, \beta > 0\}, \\ \Delta_j = \{C_k \mid k \in [j-\delta, j+\eta],\ \ \delta, \eta > 0\}, \\ X_i \in \Delta_i;\ X_j \in \Delta_j;\ \ \Delta_i, \Delta_j \subseteq C. \end{cases} \quad (14)$$

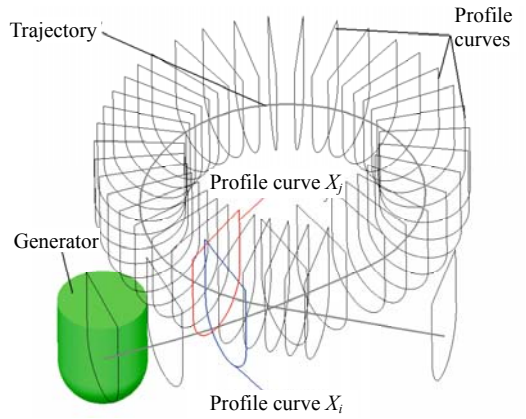

Fig.7 Two discontinuous envelop profile curves X_i (i=2) and X_j (j=38) intersect, which causes a global self-intersection in this sweeping

Here $C_{i-\alpha}$ and $C_{i+\beta}$, $C_{j-\delta}$ and $C_{j+\eta}$ are the corresponding interval ends, while $C_{i-\alpha}$ and $C_{i+\beta}$ have no intersections or overlapping with any other profiles in Δ_i, and vice versa for $C_{j-\delta}$ and $C_{j+\eta}$. If the envelop surface denoted as S_e without the ingress or egress patches has global self-intersections only, according to Eqs.(10)~(13), it can be split into five segments $S_{0,i-\alpha}$, $S_{i-\alpha,i+\beta}$, $S_{i+\beta,j-\delta}$, $S_{j-\delta,j+\eta}$ and $S_{j+\eta,m-1}$ at the above interval ends $C_{i-\alpha}$ and $C_{i+\beta}$, $C_{j-\delta}$ and $C_{j+\eta}$. We put the indices $i-\alpha$, $i+\beta$, $j-\delta$, $j+\eta$ into the set I in ascending order, $I=\{I_k|0<k<m-1\}$, which collects all the positional indices of these interval ends of profile curves in the set C, then S_e can be presented as a union of segments after trimming (Fig.8):

$$S_e=\{S_{i,j}\,|\,i,j\in I\cup\{0\}\cup\{m-1\},i<j\}, \qquad (15)$$

here $S_{i,j}$ is the separated surface segment.

After the segmentation, each envelop surface segment can create a corresponding solid segment with solid modeling techniques and the entire SV solid could be modeled by combining these conterminous solid segments. Fig.9 shows three more examples of the global self-intersection trimming.

Local self-intersections

Local self-intersections usually originate from high curvatures of the trajectory. Suppose the SV envelop surface denoted as S_e without the ingress or egress patches has a local intersection between two envelop profiles X_i and X_j ($i<j$) (Fig.10), which are extracted at the discrete positions P_i and P_j, respectively. Here P_i and P_j are in the same interval Δ_t of the trajectory.

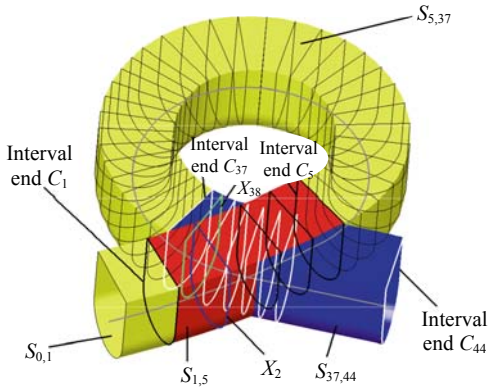

Fig.8 Representation of S_e with the example in Fig.7. In this example, Δ_2={C_1, C_2, C_3, C_4, C_5}, Δ_{38}={C_{37}, C_{38}, …, C_{43}, C_{44}}, and S_e={$S_{0,1}$, $S_{1,5}$, $S_{5,37}$, $S_{37,44}$}

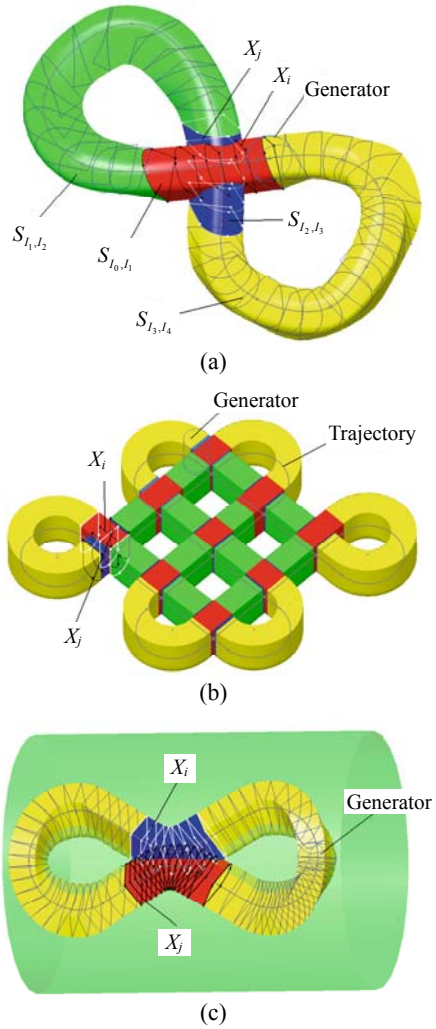

Fig.9 Examples of the global self-intersection trimming. The trajectory is: (a) a 3D closed spline curve; (b) a 2D closed composite curve; (c) a 3D closed curve on a cylindrical face. The self-intersection regions are split according to the corresponding segmental intervals

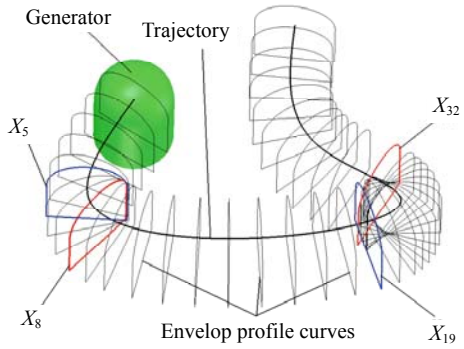

Fig.10 Two continuous envelop profile curves X_i (i=5) and X_j (j=8) intersect. So are X_i (i=19) and X_j (j=32). These cause two local self-intersections in this sweeping

One ordered subsets $\Delta_{i,j}$ of the envelop profiles C are calculated around X_i and X_j, and

$$\begin{cases} \Delta_{i,j} = \{C_k \mid k \in [i-\alpha, j+\beta],\ \ \alpha, \beta > 0\}, \\ X_i, X_j \in \Delta_{i,j},\ \ \Delta_{i,j} \subseteq C, \end{cases} \tag{16}$$

where $C_{i-\alpha}$ and $C_{j+\beta}$ are the closest pair of envelop profiles around X_i and X_j, while they do not intersect or overlap with any other profiles in $\Delta_{i,j}$ (Fig.11). The whole envelop surface S_e is firstly split in the same way as in the global self-intersection trimming at the profile curves $C_{i-\alpha}$ and $C_{j+\beta}$. Then all positional indices (such as $i-\alpha$ and $j+\beta$) of the interval end profiles in C will be collected into the set I in ascending order, $I=\{I_k|0<k<m-1\}$, and S_e is represented as

$$\begin{aligned} S_e = \{S_{i,j} \mid i, j \in I \cup \{0\} \cup \{m-1\}, i < j\} \cup \\ \{S_{i^*,j^*} \mid i^*, j^* \in I \cup \{0\} \cup \{m-1\}, i^* < j^*\}, \end{aligned} \tag{17}$$

where the set $\{S_{i,j}\}$ contains surface segments without self-intersections and the set $\{S_{i^*,j^*}\}$ are those with self-intersections.

Suppose the local self-intersections in S_e occur in the surface segment S_{i^*,j^*}, which can be considered as a collection of surface patches. As well known, corresponding UV parameters of the self-intersection positions can be evaluated by checking self-intersections in the B-spline surface (Andersson *et al*., 1998; Cohen and Ho, 2000). Given the parameters of self-intersection positions in S_{i^*,j^*}, this surface segment is further partitioned into several patches. If there is only one self-intersection position in S_{i^*,j^*} and let its UV parameter be $Sp(u_0, v_0)$ in the unitary parameter space of S_{i^*,j^*}, the surface segment S_{i^*,j^*} can be split with an evaluated area around $Sp(u_0, v_0)$, as shown in Fig.12. If there are n self-intersection positions in S_{i^*,j^*} (Fig.13), suppose the U parameters of self-intersection positions are ascending sorted as $u_0, u_1, u_2, \ldots, u_{n-1}$. So are the V parameters, which are named as $v_0, v_1, v_2, \ldots, v_{n-1}$. The partition of S_{i^*,j^*} is in the same way as there is only one self-intersection (Fig.13). After the partition, self-intersections only exist in the surface patches with self-intersection positions in.

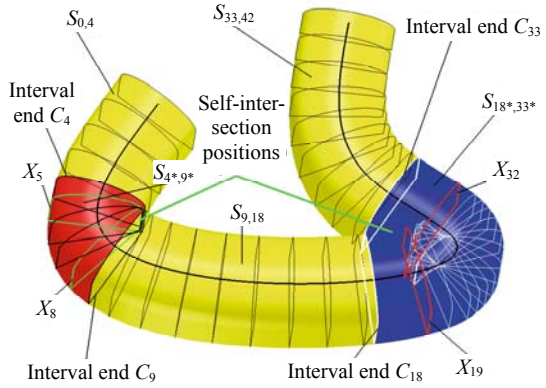

Fig.11 Representation of S_e with the example in Fig.10. In this example, there are two local self-intersection regions, $\Delta_{5,8}=\{C_4, C_5, \ldots, C_8, C_9\}$, $\Delta_{18,32}=\{C_{18}, C_{19}, \ldots, C_{32}, C_{33}\}$, and $S_e=\{S_{0,4}, S_{4^*,9^*}, S_{9,18}, S_{18^*,33^*}, S_{33,42}\}$

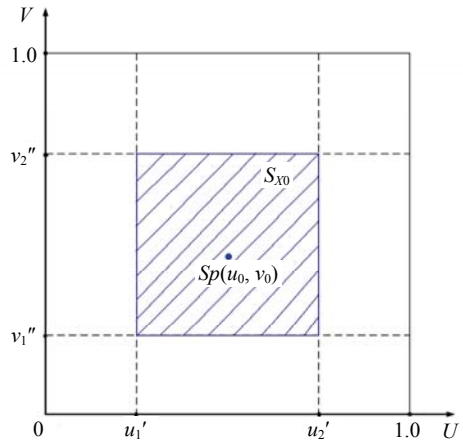

Fig.12 Unitary parameter space of the surface S_{i^*,j^*} is split into patches around one self-intersection position $Sp(u_0, v_0)$ at u_1', u_2', v_1'', v_2''. The surface patch with the self-intersection position is denoted as S_{X0}. Here $u_1' = k_1' \cdot u_0$, $u_2' = k_2' \cdot (1-u_0)$, $v_1'' = k_1'' \cdot v_0$, $v_2'' = k_2'' \cdot (1-v_0)$, while $k_1', k_2', k_1'', k_2'' \in [0, 1.0]$ and they are initialized as 0.5 but kept changing until there are only self-intersections in S_{X0}

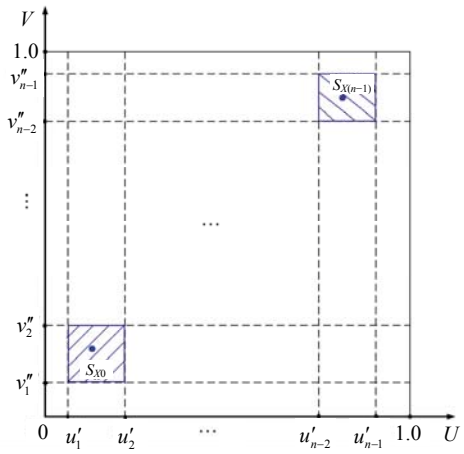

Fig.13 If there are n self-intersection positions in S_{i^*,j^*} and the U and V parameters are ascending sorted while they are denoted as u_0, u_1, u_2, …, u_{n-1} and v_0, v_1, v_2, …, v_{n-1}, respectively. So the parameters of the partition can be evaluated as: $u'_1 = k'_1 \cdot u_0$, …, $u'_i = k'_i \cdot (u_{i-1} - u_{i-2})$, …, $u'_{n-1} = k'_{n-1} \cdot (1.0 - u_{n-1})$ (i=2, 3, …, n−2); $v''_1 = k''_1 \cdot v_0$, …, $v''_j = k''_j \cdot (v_{j-1} - v_{j-2})$, …, $v''_{n-1} = k''_{n-1} \cdot (1.0 - v_{n-1})$ (j=2, 3, …, n−2), while $k'_i, k''_j \in [0, 1.0]$ (i, j= 0, 1, 2, …, n−1) and they are initialized as 0.5 but kept changing until there are only self-intersections in S_{Xi}, i=0, 1, 2, …

Among the surface patches, those local self-intersecting patches which contain the self-intersection positions in S_{i^*,j^*} are denoted as S_{Xi}, i=0, 1, 2, …, and other non-self-intersecting patches in S_{i^*,j^*} are denoted as S_{uXj}, j=0, 1, 2, … (Fig.14).

So trimming local self-intersections can be taken as recreating a new surface patch without self-intersections instead of S_{Xi} (Fig.15). The pseudo-code of this recreation algorithm is listed in Algorithm 2, and if the numbers of rows and columns of the control points net of S_{Xt} are n and m respectively, the computational complexity of this algorithm is $O(m \cdot n)$.

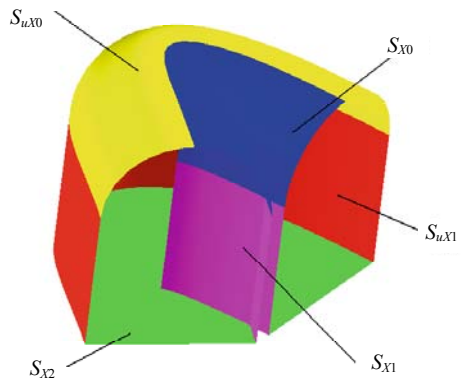

Fig.14 Self-intersecting surface segment $S_{18^*,33^*}$ is partitioned into five surface patches, where three are self-intersecting and two are not

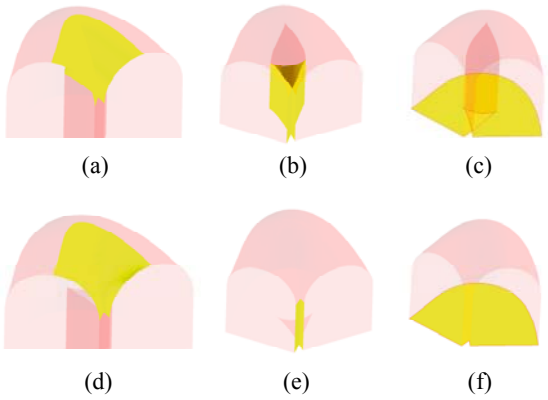

Fig.15 Three self-intersecting patches S_{X0} in (a), S_{X1} in (b), S_{X2} in (c) in Fig.14 are recreated with Algorithm 2. The newly created surface patches are (d)~(f), which correspond to (a)~(c) one to one

Algorithm 2 Substitution of a self-intersecting surface patch with a non-self-intersecting patch

Input:

S_{Xt}: A surface patch with local self-intersections.

$Cp_{i,j}$: The control points net of S_{Xt}, i=0, 1, 2, …, n−1 and j= 0, 1, 2, …, m−1, here i and j correspond to the row and column of the control points net, respectively.

Output:

S_{uXt}^*: A new surface patch without local self-intersections.

Begin

Create a new empty control point net of $Cp_{i,j}$, marked as $Cp^u_{i,j}$.

For i=1 to n−2

Set the control points of $Cp^u_{i,j}$ as follows:

$$\begin{cases} Cp^u_{i,0} = Cp_{i,0}, \\ Cp^u_{i,1} = Cp_{i,0} + \left(Cp_{i,1} - Cp_{i,0}\right) \cdot \dfrac{k_m - k_1}{k_4 - k_1}, \\ Cp^u_{i,m-2} = Cp_{i,m-1} + \left(Cp_{i,m-1} - Cp_{i,m-2}\right) \cdot \dfrac{k_{m+2} - k_3}{k_{m+2} - k_{m-1}}, \\ Cp^u_{i,m-1} = Cp_{i,m-1}. \end{cases} \tag{18}$$

Here $\{k_0, k_1, k_2, k_3, k_m, k_{m+1}, k_{m+2}, k_{m+3}\}$ is the knot list whose values are from the corresponding knot values of K_i= $\{Cp_{i,j} | j$=0, 1, 2, …, m−1$\}$;

Insert the left knots $\{k_4, k_5, \ldots, k_{m-2}, k_{m-1}\}$ from K_i back into the new control point net $Cp^u_{i,j}$, so as to make $Cp^u_{i,j}$ have the same number of control points as $Cp_{i,j}$;

End

Create a new copy control point net of $Cp_{i,j}$, marked as $Cp^v_{i,j}$;

For j=1 to m−2

Set the control points of $Cp^v_{i,j}$ as follows:

$$\begin{cases} Cp^v_{0,j} = Cp_{0,j}, \\ Cp^v_{1,j} = Cp_{0,j} + \left(Cp_{1,j} - Cp_{0,j}\right)\cdot\dfrac{k_n - k_1}{k_4 - k_1}, \\ Cp^v_{n-2,j} = Cp_{n-1,j} + \left(Cp_{n-1,j} - Cp_{n-2,j}\right)\cdot\dfrac{k_{n+2} - k_3}{k_{n+2} - k_{n-1}}, \\ Cp^v_{n-1,j} = Cp_{n-1,j}. \end{cases} \quad (19)$$

Here $\{k_0, k_1, k_2, k_3, k_n, k_{n+1}, k_{n+2}, k_{n+3}\}$ is the knot list whose values are from the corresponding knot values of $K_j = \{Cp_{i,j} | i=0, 1, 2, \ldots, n-1\}$;

Insert the left knots $\{k_4, k_5, \ldots, k_{n-2}, k_{n-1}\}$ from K_j into the new control point net $Cp^v_{i,j}$, so as to make $Cp^v_{i,j}$ have the same number of control points as $Cp_{i,j}$;

End

Create a new control point net $Cp^*_{i,j}$, let it be (see Fig.16):

$$Cp^*_{i,j} = (1-\varepsilon-\varphi)\cdot Cp_{i,j} + \varepsilon\cdot Cp^u_{i,j} + \varphi\cdot Cp^v_{i,j}, \quad (20)$$

$$\varepsilon, \varphi \in [0, 1];\ 0 \le \varepsilon+\varphi \le 1.$$

Make the new non-self-intersecting surface patch S^*_{uXt} through the new control point net $Cp^*_{i,j}$;

End

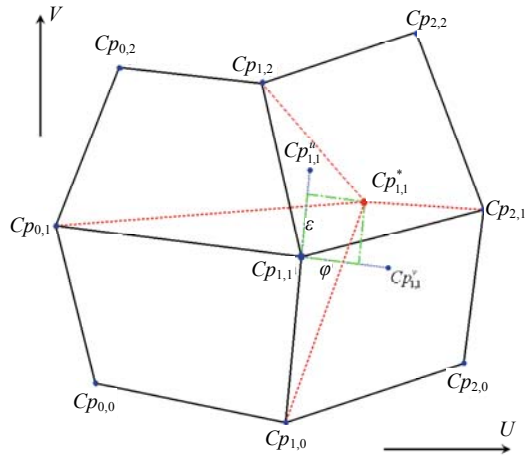

Fig.16 Taking the resetting of $Cp_{1,1}$ for example, the control point net is reset according to Eq.(20)

After the trimming of local self-intersections, the SV envelop surfaces S_e can be presented as

$$S_e = \{S_{i,j} \mid i, j \in I \cup \{0\} \cup \{m-1\}, i < j\} \cup \{S^*_{uXi} \mid i = 0, 1, 2, \ldots\} \cup \{S_{uXj} \mid j = 0, 1, 2, \ldots\}. \quad (21)$$

Sheets are created on non-self-intersecting surface segments and patches, then the entire SV solid can be modeled from these sheets by the techniques of solid modeling. Three trimming results of local self-intersections are shown in Fig.17.

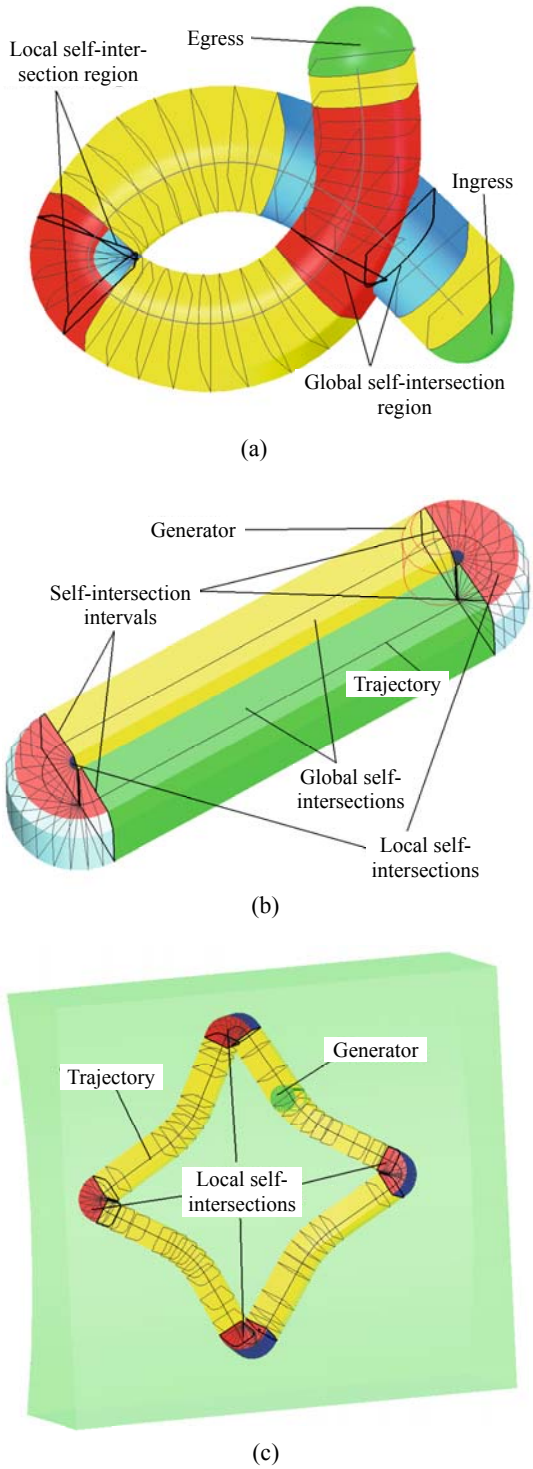

Fig.17 Three examples of the local self-intersection trimming. The trajectory in each example is: (a) an open 3D composite curve; (b) a closed 2D composite curve; (c) a closed 3D composite curve on a base surface. In (a) and (b) there are both global and local self-intersections, while in (c) there are local self-intersections only. The self-intersection patches are reconstructed according to Algorithms 1 and 2

CONCLUSION

We have presented a novel and efficient algorithm for trimming self-intersections in the modeling of swept volume (SV) solids. One of the most satisfying features of this algorithm is that no matter if both local self-intersections, global self-intersections only, or local and global self-intersections occur in the SV envelop surfaces, our algorithm can perform trimming well.

Next, we will improve the approximation errors and the computational cost of the algorithm, and in the future extend our algorithm to the general sweeps.

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. [doi:10.1016/S0010-4485(96)00097-8]

Abdel-Malek, K., Yang, J., Blackmore, D., Ken, J., 2006. Swept volumes: foundations, perspectives and applications. *Int. J. Shape Modeling*, **12**(1):87-127. [doi:10.1142/S0218654306000858]

Andersson, L.E., Peters, T.J., Stewart, N.F., 1998. Self-intersection of composite curves and surfaces. *Computer Aided Geometric Design*, **15**(5):507-527. [doi:10.1016/S0167-8396(98)00005-3]

Blackmore, D., Leu, M.C., 1992. Analysis of swept volume via Lie group and differential equations. *Int. J. Rob. Res.*, **11**(6):516-537. [doi:10.1177/027836499201100602]

Blackmore, D., Leu, M.C., Wang, L.P., 1997a. The sweep-envelop differential equation algorithm and its application to NC machining verification. *Computer-Aided Design*, **29**(9):629-637. [doi:10.1016/S0010-4485(96)00101-7]

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

Blackmore, D., Samulyak, R., Leu, M.C., 1999. Trimming swept volumes. *Computer-Aided Design*, **31**(3):215-223. [doi:10.1016/S0010-4485(99)00017-2]

Cohen, E., Ho, C.C., 2000. Surface Self-intersection. *In*: Lyche, T., Schumaker, L.L. (Eds.), Mathematical Methods for Curves and Surfaces, p.183-194.

Kim, Y.J., Varadhan, G., Leu, M.C., Dinesh, M., 2004. Fast swept volume approximation of complex polyhedral models. *Computer-Aided Design*, **36**:1013-1027. [doi:10.1016/j.cad.2004.01.004]

Mann, S., Bedi, S., 2002. Generalization of the imprint method to general surfaces of revolution for NC machining. *Computer-Aided Design*, **34**(5):373-378. [doi:10.1016/S0010-4485(01)00103-8]

Martin, R.R., Stephenson, P.C., 1990. Sweeping of three-dimensional objects. *Computer-Aided Design*, **22**(4):223-234. [doi:10.1016/0010-4485(90)90051-D]

Peternell, M., Pottmann, H., Steiner, T., Zhao, H., 2005. Swept volumes. *Computer Aided Design and Applications*, **2**:95-104.

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

Pottmann, H., Peternell, M., 2000. Envelops—Computational Theory and Applications. Proc. Spring Conf. on Computer Graphics and its Applications, p.3-23.

Requicha, A.A.G., Voelcker, H.B., 1982. Solid modeling: a historical summary and contemporary assessment. *IEEE Computer Graph. Appl.*, **2**(2):9-24. [doi:10.1109/MCG.1982.1674149]

Rossignac, J., Kim, J.J., Song, S.C., Suh, K.C., Joung, C.B., 2007. Boundary of the volume swept by a free-form solid in screw motion. *Computer-Aided Design*, **39**(9):745-755. [doi:10.1016/j.cad.2007.02.016]

Wang, W.P., Wang, K.K., 1986. Geometric modeling for swept volume of moving solids. *IEEE Computer Graph. Appl.*, **6**(12):8-17.

Weinert, K., Du, S.J., Damm, P., Stautner, M., 2004. Swept volume generation for the simulation of machining process. *Int. J. Machine Tools Manufacture*, **44**(6):617-628. [doi:10.1016/j.ijmachtools.2003.12.003]

Weld, J., Leu, M., 1990. Geometric representation of swept volume with application to polyhedral objects. *Int. J. Rob. Res.*, **9**(5):105-117. [doi:10.1177/027836499000900507]

Xu, Z.Q., Chen, Z.Y., Ye, X.Z., Zhang, S.Y., 2007. Approximate the Swept Volume of Revolutions along Curved Trajectories. Proc. ACM Symp. on Solid and Physical Modeling, p.309-314. [doi:10.1145/1236246.1236290]

Yu, H.B., Wang, Y.X., 2003. Swept volume and its application to mechanical design. *J. Eng. Graph.*, **24**(1):63-70 (in Chinese).