COMPUTER-AIDED DESIGN & APPLICATIONS, 2016
VOL. 13, NO. 1, 95–107
http://dx.doi.org/10.1080/16864360.2015.1059200

Recognition of depression and protrusion features on B-rep models based on virtual loops

Jiing-Yih Lai[a], Ming-Hsuan Wang[a], Yu-Kai Chiu[a], Chia-Hsiang Hsu[b], Yao-Chen Tsai[b] and Chung-Yi Huang[b]

[a]National Central University, Jhongli, Taiwan; [b]CoreTech System (Moldex3D) Co., Ltd., Hsinchu, Taiwan

ABSTRACT
Loops are vital elements in B-rep models and are used to describe the boundary contours of faces. A loop is only defined on a single face, which does not reflect real situations in which features mostly lie across multiple faces. The objective of this study is to detect virtual loops across multiple faces and subsequently use them for recognizing depression and protrusion features in computer-aided design models. Three loop types are defined: single, virtual, and multivirtual loops; virtual and multivirtual loops lie across multiple faces with different boundary conditions across faces. The data of the detected loops are then used to develop a feature recognition algorithm for identifying various depression and protrusion types, ranging from simple circular holes on a face to complex irregular pockets on multiple faces with fillets. This paper provides a detailed description of the proposed algorithm and presents several examples to illustrate its feasibility.

KEYWORDS
virtual loop; loop recognition; feature recognition; blend-face recognition

1. Introduction

Mesh generation is one of the most essential steps in computer-aided engineering (CAE) analysis. One approach for automatic mesh generation involves digitizing all edges on a computer aided design (CAD) model, then generating surface meshes on each of the faces according to the nodes surrounding the face, and finally converting all surface meshes into solid meshes [14]. Tiny, irregular, and redundant meshes may occur if all edges are equally considered. To improve the quality of meshes, it is imperative to identify specific features in the CAD model and subsequently apply a specific digitization strategy for each feature. Typical features that should be assessed are fillets, chamfers, holes, extrusions, ribs, and bosses. Although extensive research has been performed in feature recognition [2],[9],[17],[20] its application in CAE analysis is still limited because of the difficulty of converting three-dimensional (3D) shapes into solid meshes, particularly in hexahedral elements. Nevertheless, feature recognition is still crucial for developing automatic mesh generation algorithms. This study proposes a loop-recognition algorithm and demonstrates its applications for recognizing various hole types and protrusions. This is part of a long-term research aimed at recognizing all critical features that should be meshed separately. The proposed loop data can be used as a basis for developing advanced feature recognition algorithms.

Boundary representation (B-rep) is a method for representing shapes in solid modeling and can be implemented in various types of data structure and used in different manners. Fig. 1 illustrates the B-rep data structure used in Rhino [16] and the current study. In this B-rep data structure (denoted as the B-rep model hereafter), loops are vital elements and are used to link all edges corresponding to faces. In B-rep models, a loop is defined only on a single face. However, features in 3D CAD models may lie across multiple faces; hence, these features are beyond the data structure of current B-rep models. If the concept of "loop" in current B-rep models can be extended to more general cases where a loop can lie across multiple faces and associated topological data can be recorded, then the restriction of current loop structures for feature recognition can be relieved. Therefore, to extend the applicability of loops in complex feature recognition, the definition of loops should be expanded and associated feature recognition algorithm should be developed.

CONTACT Jiing-Yih Lai jylai@ncu.edu.tw

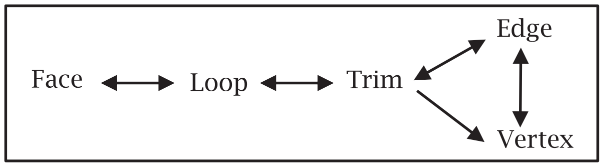


Figure 1. B-rep data structure used in Rhino.

Various feature types exist according to different classifications. One of the fundamental classifications is that a feature is either a concave or convex type. Either type can be divided further into several subtypes. For example, a concave feature includes features such as holes and pockets, whereas a convex feature includes features such as bosses, extrusions, and ribs. Among feature recognition methods, determining the contour surrounding the target feature is the most systematic and effective for accurately recognizing complex features. This is because a feature can generally be recognized by detecting its outline contour.

Most researchers have employed the topological relationships of adjacent entities, such as face adjacency graphs and attributed adjacency graphs (AAGs), for recognizing features [1],[8],[18]. Ansaldi et al. [1] presented a face adjacency graph structure for generating graphs of the topological relationship of adjacent faces. Joshi et al. [8] added rules according to edges and face conditions to an AAG method for recognizing feature types. Tian et al. [18] defined three types of edge and developed rules for identifying holes. These researchers have suggested that all fillets must be removed before performing feature recognition, which is practically difficult because various fillets in real cases are complex.

Several researchers have focused on the recognition of fillets because fillets are prevalent in CAD models and complicate the topological structure of faces [4],[7],[11],[19]. Venkataraman et al. [19] clarified edge blend face (EBF) as face blend face, and cliff blend face as a face-edge blend face. Cui et al. [4] proposed an algorithm that uses smooth edges, support faces, and span angles for recognizing EBFs and vertex blend faces (VBFs). Li et al. [11] introduced an algorithm that uses smooth edges, normal vectors, span angles, and the area of a target feature for automatically recognizing and screening BFs. Joshi et al. [7] emphasized the importance of recognizing and simplifying BFs on freeform surfaces. However, the form of fillets in real CAD models may vary considerably, and not all forms can be thoroughly recognized.

Some studies have also been conducted for recognizing depression or protrusion features [12],[21]. Lim et al. [12] proposed an algorithm for the identification of depression and protrusion features (DP-features) on freeform solids. This algorithm identifies the boundary edges of DP-features and then creates a surface patch to cover the depressions or isolate the protrusions. Their method lies in the use of G^1 continuity between edge segments to identify DP-feature boundaries that cross multiple faces and geometries. Zhang et al. [21] proposed a region-based method for recognizing protrusion and depression features. Symbolic computation was employed to characterize the curvature properties of the freeform surface and to help to decompose the surface into regions. A rule-based approach was then employed to recognize protrusion and depression features in terms of particular geometry and topology relations.

Regarding the application of loops for feature recognition, Li et al. [10] proposed an edge-based approach for recognizing small depression features for the purpose of mesh generation. Edges are classified as convex, concave and smooth shapes. Convex inner loops are then formed by edges and are used for recognizing depression features. In particular, their method can handle the existence of fillets and chamfers on surface boundaries. However, the depression types considered in this study are too simple. Chung et al. [3] proposed an approach that entails combining connected faces for searching loops across multiple faces. The running time is mainly influenced by the maximum number of faces to be connected. Joshi et al. [7] proposed an algorithm for simplifying holes according to edges across different faces for sheet metal parts. The edges across different faces are similar to one loop type addressed in the current study. Ismail et al. [5],[6] proposed a technique called edge boundary classification (EBC) for recognizing simple and interacting cylindrical- and conical-based features from B-rep models. They used edge loops to form the basis of the edge boundary classification technique. An edge loop is composed of a set of connected edges that form the closed boundary of a non-self-interacting face. An EBC pattern is formed in terms of three test points, two points are on the edge loop and the third is the midpoint of them. A loop-up table in terms of the EBC pattern is provided to detect different feature types. Lu et al. [13] proposed a feature-based decomposition technique for automatically generating hexahedral meshes. This technique comprises four stages: feature determination to extract decomposition features, cutting surface generation to form the cutting surfaces, body decomposition to derive the imprinted volumes, and mesh generation. Various loop types are defined according to concave and convex properties between connected faces. The loops are combined with rules for extracting protrusion and depression features. However, they did not evaluate the occurrence of fillets in the CAD model.

Real 3D models are generally complex because the features are typically filleted at the boundary and are composed of multiple faces. A feature may be affected by the following types of multiple face: (1) The base face at which the feature is located may be composed of multiple faces; (2) the feature itself may be composed of multiple faces; (3) the feature is generally filleted at its boundary, resulting in one or more transition faces (BFs) between the feature face and the base face; and (4) one or more of the faces related to the feature are virtual faces (a virtual face refers to a face that is extended to other part faces or features).

The complexity of a feature primarily depends on the status of the following four conditions: Do multiple base faces exist? Do multiple feature faces exist? Does a BF exist? Does a virtual face exist? In the simplest case, all the answers to these questions are “No”; however, in the most complex case, all the answers are “Yes.” Fig. 2 depicts different combinations of the mentioned types of multiple face in several CAD models. The first plot in Fig. 2 represents a simple case, because this case involves only multiple base faces; by contrast, the final plot in Fig. 2 represents a complex case because it illustrates a model of multiple base faces and feature faces with fillets in between. The complexity of the other three cases falls between the previous two cases. For all the mentioned cases, it is imperative to define and construct different loop types that represent all elements (edges, rims and faces) related to the target features, and to employ such loops for feature recognition.

This study presents a virtual loop concept to represent all loop types used in CAD models and proposes algorithms for recognizing various types of depression and protrusion features. Three loop types are defined in this study: single, virtual, and multivirtual loops. A single loop is the current loop recorded in a B-rep model. A virtual loop lies across faces that are at least G^1 continuous. A multivirtual loop, however, lies across faces that are either G^0 or G^1 continuous. A loop recognition algorithm is proposed for identifying and distinguishing the aforementioned loops in CAD models. A feature recognition algorithm is then developed using the available loop data to identify various types of depression and protrusion features, ranging from simple circular holes on a single face to complex irregular pockets on multiple faces with fillets on boundary edges.

2. Overall method

Figure 3 depicts an example demonstrating the integrated procedure of the proposed loop- and feature-recognition algorithm. The B-rep model and BFs serve as the inputs. Several algorithms can recognize blend faces (BFs) [10], [19], which are considered the input in the subsequent discussion. Three loop types are defined according to the recognition complexity. The single loop is exactly the same as the loop data recorded in the B-rep model. This loop describes a contour of trims surrounding the exterior or interior boundaries of a face. Both virtual and multivirtual loops lie across multiple faces. For the virtual loop, all faces contributing the loop are at least G^1 continuous with their adjacent faces along the loop direction. For the multivirtual loop, however, all faces contributing the loop may either be G^0 or G^1 continuous with their adjacent faces along the loop direction. Single loops in a CAD model can be acquired directly from the B-rep model. Virtual and multivirtual loops, however, are currently not recorded in the B-rep data structure. As shown in Fig. 4, a single and virtual loop can each be further divided into inner and outer loops. Inner loops are used to search for features, whereas outer loops are used to search for multivirtual loops. A multivirtual loop is another type of inner loop used to search for features across multiple faces.

At the loop recognition stage, all loop types, including single inner and outer loops, virtual inner and outer loops, and multivirtual loops (inner loop), must be recognized. Because single loops are already prerecorded in the B-rep model, the goal at this stage is to detect virtual and multivirtual loops. Because the difference between a virtual loop and a multivirtual loop is the continuity condition of the edges along the loop, all faces in the CAD

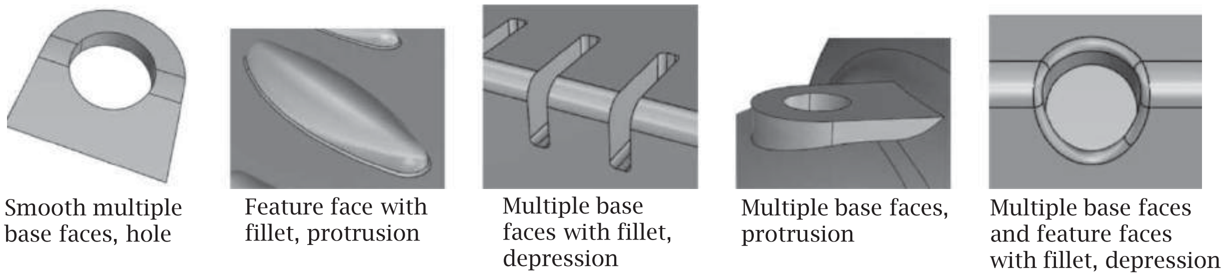

Smooth multiple base faces, hole
Feature face with fillet, protrusion
Multiple base faces with fillet, depression
Multiple base faces, protrusion
Multiple base faces and feature faces with fillet, depression

Figure 2. Several examples expressing the compositions of multiple base faces, multiple feature faces, blend faces, and virtual face on features.

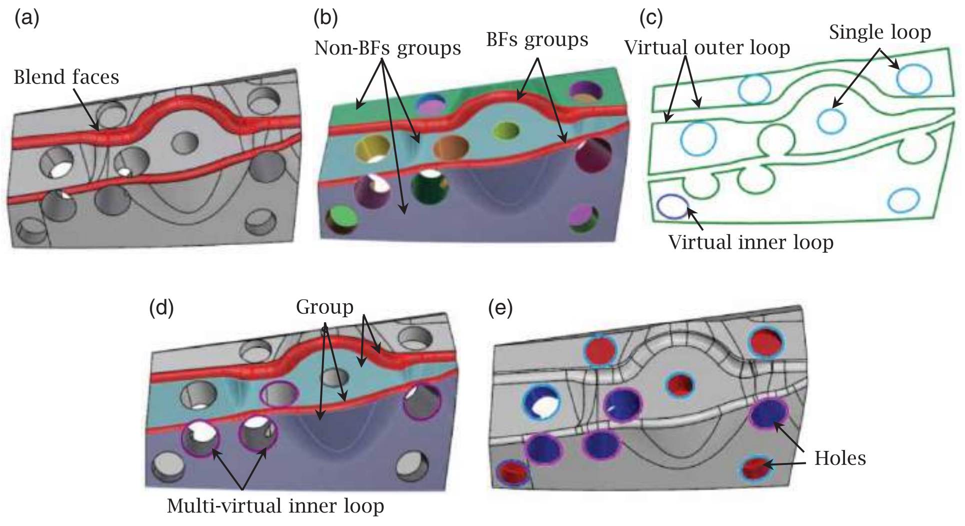


Figure 3. An example showing the integrated procedure of the proposed loops- and features- recognition method, (a) input B-rep model and blend faces, (b) blend faces and non-blend faces clustering, (c) determination of virtual inner and outer loops, (d) determination of multi-virtual loops, (e) feature recognition.

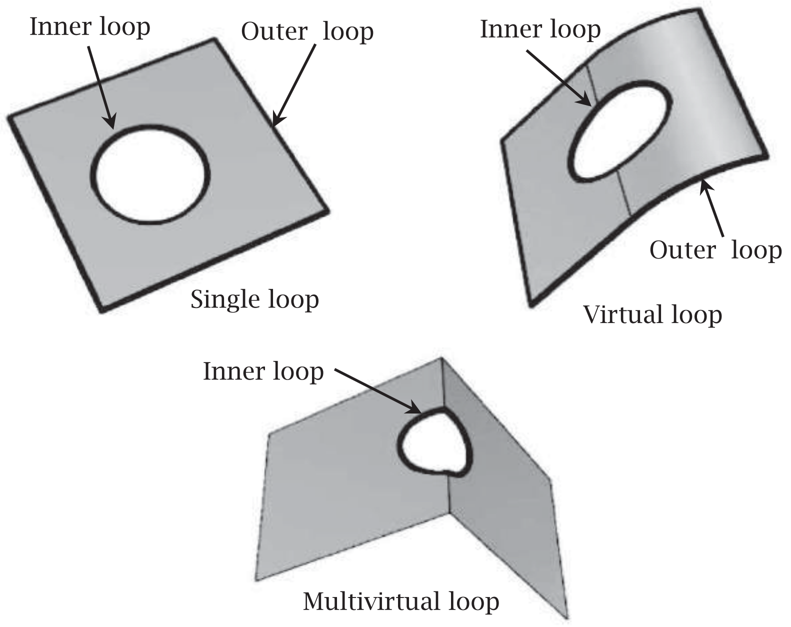


Figure 4. All types of inner and outer loops, where inner loops are used for feature recognition, and outer loops are used for evaluating multivirtual loops.

model are grouped separately; the groups are used to identify various loop types. The faces are first divided into BFs and non-BFs. All neighboring BFs are grouped individually. Similarly, all neighboring non-BFs are grouped individually. The boundary conditions of a group of non-BFs are either BFs or non-BFs having G^0 continuity with the group. Therefore, all faces in a group of non-BFs are at least G^1 continuous with their adjacent faces, and the loops in such a group are virtual loops.

Multivirtual loops essentially lie across multiple groups of blend and non-BFs. Two types of multivirtual loop exist: The first type crosses BFs whereas the second type does not. Virtual outer loops are employed to search for multivirtual loops. Consider one of them as a seed loop. All neighboring loops to the seed are located and divided into two types: G^0 and G^1 continuity at the common edge. When the faces at the common edge are G^0 continuous, the multivirtual loop lies across two groups of non-BFs; by contrast, when they are G^1 continuous, the multivirtual loop lies across two groups of non-BFs, with BFs in between. Two procedures are developed to evaluate both multivirtual loop types separately.

The feature recognition stage involves the recognition of depressions and protrusions. The face on which a feature is located is called a base face; the contour at the transition of the base face and the feature is called a loop, and the feature faces adjacent to the loop are called side faces. The feature recognition algorithm is implemented systematically by loop. For each loop, a set of angles θ's between the base face and the side faces is evaluated. Each angle θ is evaluated at the vertex of an edge. When all angles are convex, the corresponding side faces are considered a protrusion. By contrast, when at least one angle among all angles is concave, the corresponding side faces are considered a depression. Both protrusions and depressions can further be divided into several subtypes; only holes are introduced hereafter. Blind holes and through holes can be distinguished in the proposed algorithm; furthermore, circular holes and noncircular holes can be separated. Noncircular and blind holes are commonly called pockets in computer aided manufacturing.

3. Loop recognition

The loop recognition procedures can be divided into four steps (Fig. 5).

(1) Cluster BFs and non-BFs: The faces are separated into groups so that the search can be implemented according to the boundary conditions of each group.
(2) Determine virtual outer and inner loops within each group of faces: The faces in each group of non-BFs are G^1 continuous with their adjacent faces. For each group of non-BFs, an algorithm is implemented to detect virtual outer and inner loops.
(3) Determine multivirtual inner loops across multiple face groups: The edges on each multivirtual inner loop are either G^0 or G^1 continuous with their adjacent faces.
(4) Record all topological information: The topological information of all new loops is regenerated in the B-rep model and such topological data are recorded in the data structure.

These steps are detailed as follows.

3.1. Cluster blend faces and nonblend faces

All faces are clustered so that the entire CAD model can be partitioned into small face groups. The partitioning concept is outlined as follows: (1) All BFs that connect to each other in sequence are considered as a group, and (2) all non-BFs that are G^1 continuous with their adjacent

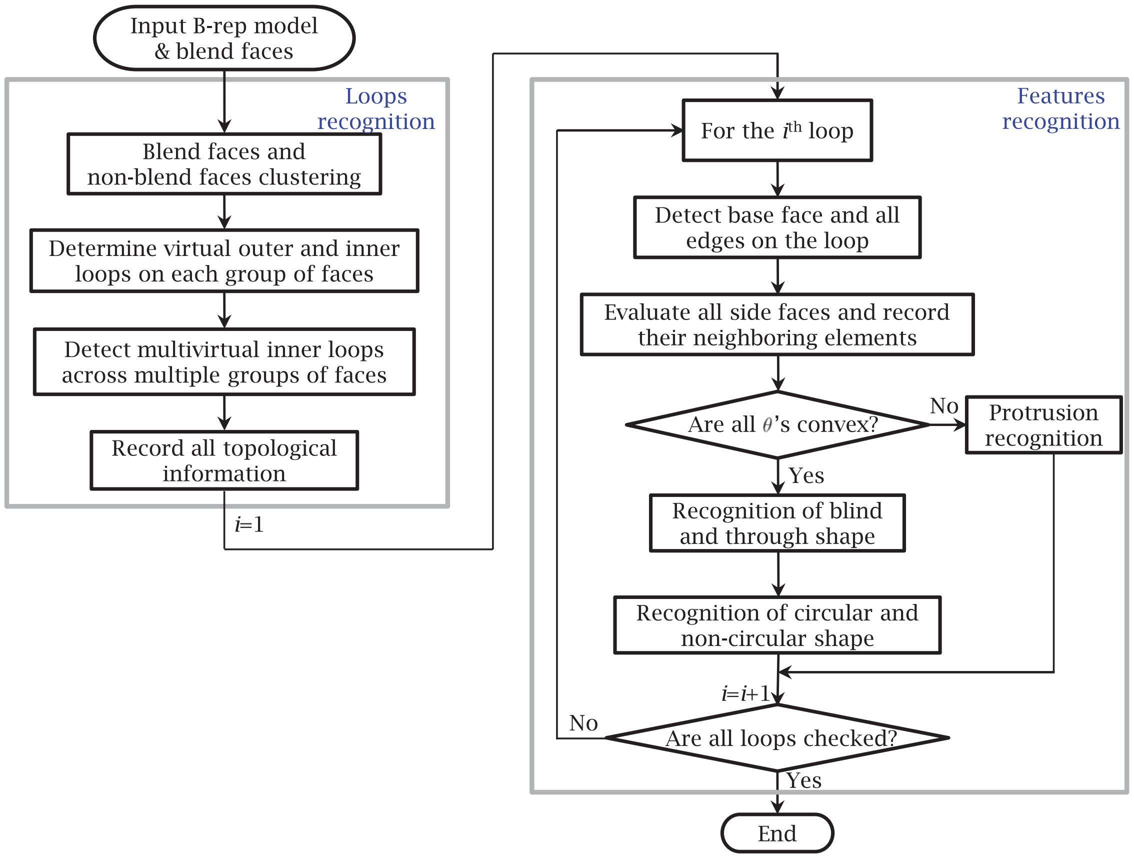

Input B-rep model & blend faces
Loops recognition
Blend faces and non-blend faces clustering
Determine virtual outer and inner loops on each group of faces
Detect multivirtual inner loops across multiple groups of faces
Record all topological information
i=1
Features recognition
For the ith loop
Detect base face and all edges on the loop
Evaluate all side faces and record their neighboring elements
Are all θ's convex?
No
Protrusion recognition
Yes
Recognition of blind and through shape
Recognition of circular and non-circular shape
i=i+1
No
Are all loops checked?
Yes
End

Figure 5. Overall flowchart of the proposed method for the recognition of various kinds of loops and features.

faces are considered as a group. Therefore, after the clustering process, a set of groups can be derived, and these groups are saved in an array. The groups of BFs are saved at the front of the array, whereas the groups of non-BFs are saved after those of BFs. In this type of arrangement, different group types can be easily recognized.

The process of clustering BFs is straightforward and simple. The topological data of edges and trims in the B-rep model can be used to evaluate the neighboring faces of a BF. Whenever a neighboring BF is located, the search can advance from the new BF. A group of BFs can thus be obtained when the search can no longer advance. This clustering algorithm is continued until all BFs have been examined. A group can contain a single BF or multiple BFs. In addition, a formed group of BFs contains an outer loop but no inner loop. This is a vital property applied in a subsequent step to search for virtual and multivirtual loops.

For a group of non-BFs, the boundary edges of the group are either adjacent to the BFs or G^0 continuous with their adjacent faces. Therefore, starting from a seed face, a region growing is performed along each boundary edge to determine the face that is G^1 continuous with the seed face. When the growing process reaches an edge neighboring the BF or an edge exhibiting G^0 continuity with its adjacent face, it stops in that direction. This procedure is implemented for all faces in the face list except for when a face is a BF or when it already belongs to a group. Once the growing is completed, a set of groups all composed of non-BFs can be derived.

3.2. Determine virtual outer and inner loops within each group of faces

For each group of non-BFs, at least one loop exists- in the virtual outer loop. This loop represents the external boundary of the entire group of faces. In addition, virtual inner loops may exist if depressions or protrusions are present. These loops are called "inner loops" because they appear within the group of faces; the term "virtual" is used to distinguish these loops from traditional loops in the B-rep model. Fig. 6 illustrates a schematic definition of virtual loops. In this example, the group comprises four faces, where t_i denotes the trim on each edge and l_j denotes the loop corresponding to each face in the B-rep model. The outer and inner dark profiles (i.e., the dark outer and inner lines) represent the virtual outer and inner loops, respectively.

The B-rep data structure essentially comprises five elements, namely the vertex, edge, trim, loop, and face. The data recorded in each element include geometric and topological data [16]. A brief description of some of the topological information related to this study is provided as follows. An edge is essentially a 3D element and is nondirectional. A trim, however, is a two-dimensional element and is directional. An edge is generally mapped onto two trims. A loop is essentially composed of trims. The problem of identifying a loop for a group of faces is equivalent to evaluating the trims surrounding the target loop. When an edge is located on the boundary of a group, its two trims should belong to two different groups. By contrast, when an edge is located in a group, its two trims should belong to the same group. This property is employed in the proposed algorithm for removing trims corresponding to inner edges.

Fig. 7 depicts the flowchart of the process involved in evaluating virtual outer and inner loops within a group of faces; the input is all groups G_i, where i denotes the group index. Because the first n_b groups belong to BFs, each of these groups can be processed first to yield a virtual outer loop. In general, no virtual inner loop exists in the group of BFs because the radii of BFs are short.

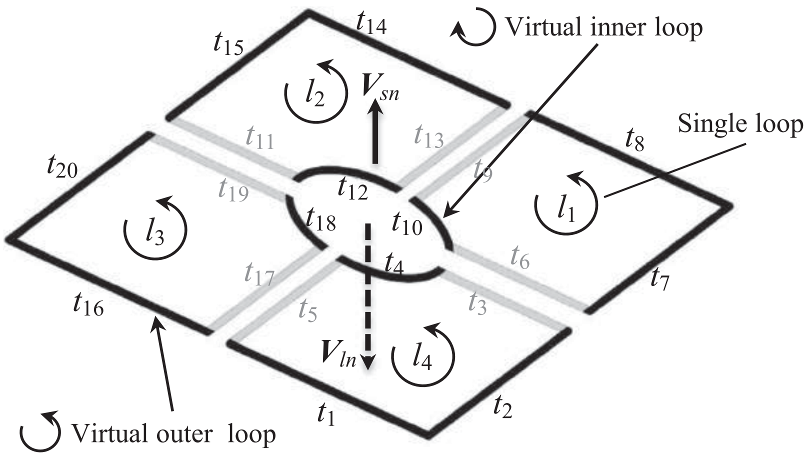


Figure 6. Redundant trims removed to form virtual inner and outer loops, where t_i indicates trims and l_j indicates loops.

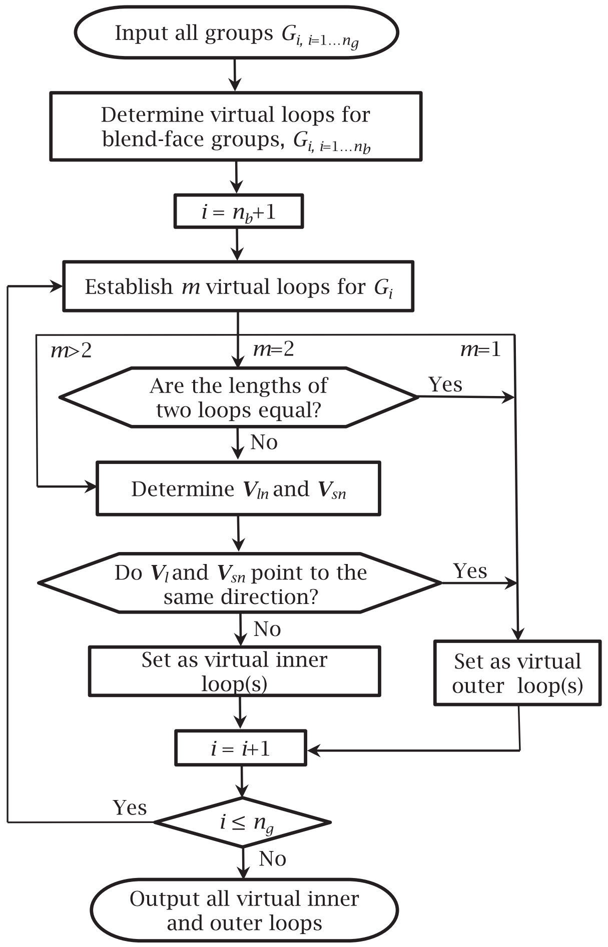

Input all groups $G_{i,\ i=1...n_g}$
Determine virtual loops for blend-face groups, $G_{i,\ i=1...n_b}$
$i = n_b+1$
Establish m virtual loops for G_i
m>2
m=2
m=1
Are the lengths of two loops equal?
Yes
No
Determine V_{ln} and V_{sn}
Do V_l and V_{sn} point to the same direction?
Yes
No
Set as virtual inner loop(s)
Set as virtual outer loop(s)
$i = i+1$
Yes
$i \le n_g$
No
Output all virtual inner and outer loops

Figure 7. Flowchart for determining virtual inner and outer loops on each group of faces.

However, in some cases, virtual inner loops may still exist, such as tiny holes on BFs. Furthermore, a search must be performed to determine whether any inner loop exists in the group of BFs. The remaining other groups essentially comprise non-BFs. Each of the groups is examined individually to detect the virtual outer loops and virtual inner loops. For each of these groups, the redundant trims, such as the gray profiles shown in Fig. 6, are removed first. The remaining trims can easily form individual trim chains, where m denotes the number of chains found. A chain refers to a series of trims forming a closed contour.

As illustrated in Fig. 7, when $m = 1$, only the outer loop exists and can thus be considered a virtual outer loop. If $m = 2$ and the lengths of both profiles are equal, then the two loops belong to two boundaries of a cylindrical surface; hence, these loops are considered virtual outer loops. If $m > 2$ or $m = 2$, but with different lengths, then two vectors $\boldsymbol{V}_{ln}$ and $\boldsymbol{V}_{sn}$ (Fig. 6) are evaluated, where $\boldsymbol{V}_{ln}$, a loop vector, represents the surface normal of a plane lying on the loop, and $\boldsymbol{V}_{sn}$ denotes the surface normal of the non-BFs. If $\boldsymbol{V}_{ln}$ and $\boldsymbol{V}_{sn}$ point in the same direction, then this loop is considered a virtual outer loop; however, if $\boldsymbol{V}_{ln}$ and $\boldsymbol{V}_{sn}$ point in reverse directions, then this loop is considered a virtual inner loop. A cylindrical surface with different lengths on both profiles and all other cases belong to this loop type. This procedure is repeated for all non-BF groups until all virtual outer and inner loops are derived.

Fig. 6 indicates two virtual loops, where the dark lines represent the trims of the loops and gray lines represent the removed trims (because these trims belong to inner edges). The remaining trims can form two virtual loops; the first loop (t_1-t_2-t_7-t_8-t_{14}-t_{15}-t_{20}-t_{16}) and second loop (t_4-t_{18}-t_{12}-t_{10}). For the first loop, the trims are arranged counterclockwise; thus, $\boldsymbol{V}_{ln}\cdot\boldsymbol{V}_{sn} = 1$, indicating a virtual outer loop. For the second loop, the trims are arranged clockwise; thus, $\boldsymbol{V}_{ln}\cdot\boldsymbol{V}_{sn} = -1$, indicating a virtual inner loop.

3.3. Detect multivirtual inner loops across multiple groups of faces

A loop may frequently lie across BFs and multiple groups of non-BFs. The process of recognizing multivirtual loops is more difficult than that of recognizing virtual loops, because the variation in the neighboring conditions can be highly complex. Nevertheless, multivirtual loops have the following properties: (1) A multivirtual

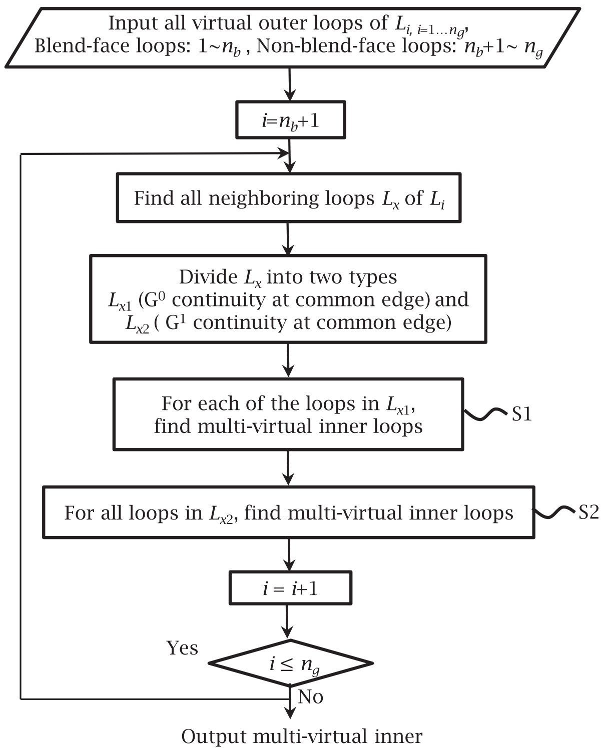

Input all virtual outer loops of $L_{i,\ i=1...n_g}$, Blend-face loops: $1\sim n_b$, Non-blend-face loops: $n_b+1\sim n_g$
$i=n_b+1$
Find all neighboring loops L_x of L_i
Divide L_x into two types L_{x1} (G^0 continuity at common edge) and L_{x2} (G^1 continuity at common edge)
For each of the loops in L_{x1}, find multi-virtual inner loops
S1
For all loops in L_{x2}, find multi-virtual inner loops
S2
$i = i+1$
Yes
$i \le n_g$
No
Output multi-virtual inner

Figure 8. Flowchart for determining multi-virtual inner loops across multiple groups of faces.

loop is composed of several outer loop segments from different groups. That is, each trim in the loop should originate from a virtual outer loop; (2) most trims in the loop originate from non-BFs, whereas some of the trims may originate from BFs; and (3) the trims in the loop may exhibit G^0 and G^1 continuity at the connecting points.

The concept of the process of recognizing multivirtual loops is described as follows. All virtual outer loops are considered the input to this process because of the first property mentioned in the preceding paragraph. The loop input can be divided into BF loops and non-BF loops because of the second property. The algorithm for recognizing multivirtual loops identifies a seed loop and then determines all loops neighboring this seed loop. The search is then divided into two tracks. If the common edge between the seed loop and a neighboring loop is G^0 continuous, then a subprocess is implemented to detect multivirtual loops. By contrast, if the common edge between the seed loop and a neighboring loop is G^1 continuous, then another subprocess is implemented to detect multivirtual loops. This is because of the mentioned third property. These procedures are repeated until all input virtual loops have been tested.

Fig. 8 shows the flowchart of the proposed algorithm for evaluating multivirtual inner loops across multiple groups. The input of this algorithm is the virtual outer loops $L_{i,i=1\ldots ng}$, where L_i represents the virtual outer loop of the ith group (obtained in the procedures presented in the previous section), and n_g is the number of groups. The first n_b loops comprise BFs, whereas the remaining loops comprise non-BFs. Starting from the first non-BF loop L_i, where $i = n_b + 1$, all of its neighboring loops L_x are derived. The loops L_x are divided into two groups (i.e., L_{x1} and L_{x2}) according to the continuity conditions between L_x and L_i. If L_x and L_i are G^0 continuous at the common edge, then L_x is attributed to L_{x1}; by contrast, if L_x and L_i are G^1 continuous at the common edge, then L_x is attributed to L_{x2}. For each loop in L_{x1}, S1 is implemented to identify the multivirtual inner loops. For all loops in L_{x2}, S2 is implemented to identify additional multivirtual inner loops. Subsequently, i is incremented by 1 to shift to the next non-BF loop, and the entire procedure is repeated. The process is completed when all non-BFs have been tested.

In S1, each of the loops in L_{x1} is merged with L_i to obtain an integrated loop L_A. In each L_A, all trims that form the inner edges are removed after the merging process. The remaining trims can thus be connected sequentially to form loops. If more than one loop is formed, then the longest loop is considered a multivirtual outer loop, whereas the remaining other loops are considered multivirtual inner loops. Once all L_{x1} have been tested, all the derived multivirtual inner loops are recorded. In S2, BFs exist between two non-BF groups, and L_i belongs to one of the groups. Therefore, when a BF is neighboring L_i, the loop in another group of non-BFs must be determined to search for multivirtual loops. Each of the loops in L_{x2} is examined to determine whether it has been merged previously; if not, then this loop must be a BF; this loop and its neighboring loops are recorded in a stack. Once all L_{x2} are examined, all loops recorded in the stack are merged as an integrated loop L_B, where all trims forming inner edges are removed. The remaining trims can then be connected sequentially to form loops. If more than one loop remains, then the longest loop is considered a multivirtual outer loop, whereas the other loops are considered multivirtual inner loops.

Fig. 9 depicts the computation of multivirtual loops. Assume L_5 (Fig. 9(a)) represents the seed loop. Its neighboring loops are L_1–L_4 and L_7, where L_3, L_4, and L_7 are G^0 continuous with L_5; L_1 and L_2 are G^1 continuous with L_5; and L_6 neighbors L_1 and L_2. Here, L_3, L_4, and L_7 are assigned to L_{x1}. Each loop in L_{x1} is merged individually with L_5 to determine whether a multivirtual

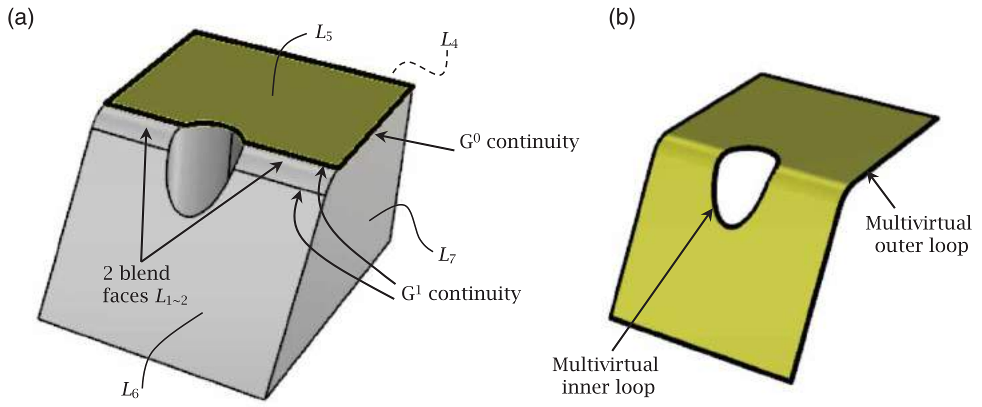


Figure 9. Detection of loops across multiple groups, (a) seed loop L5 and its neighboring loops, (b) multivirtual inner and outer loops detected.

inner loop exists. In this example, no inner loop exists between L_3 and L_5, L_4 and L_5, and L_7 and L_5. Subsequently, the loops L_1 and L_2 are assigned to L_{x2}. Each loop in L_{x2} is examined to identify its neighboring loop. In this example, all loops L_1 and L_2 can lead to L_6. Here, L_5, L_1, and L_2 are merged first, and L_6 is subsequently added to the resulting loop chain. Subsequently, all trims forming inner edges are removed from the loops. Fig. 9(b) indicates that after the removal of inner edges, one multivirtual outer loop and one multivirtual inner loop are clearly distinguishable.

3.4. Record all topological information

Any repeated loops and trims that overlap the same path are deleted, and only the loops where all trims are derived from BFs are filtered. After these procedures, all loop types in a CAD model are detected, and all the adjacent edges and faces related to each loop are recorded.

4. Feature recognition

The procedures involved in recognizing depression and protrusion features are described as follows (Fig. 5). For each inner loop obtained in the procedures highlighted in the preceding sections (whether it is a single, virtual, or multivirtual loop), the base face and all edges of the loop are detected. All side faces are then evaluated according to these edges. A side face may directly or indirectly neighbor an edge with one or several BFs in between. If the adjacent edge is not smooth, then the side face is adjacent to the base face and can therefore be directly obtained from the B-rep data; by contrast, if the adjacent edge is smooth, then a BF exists between the side face and the base face. The BF is recorded as the neighboring face of the base face, and the other face adjacent to the BF is recorded as the side face.

Fig. 10(a) depicts a loop with four edges e_0–e_3 on the base face B_F, where edges e_1 and e_3 are nonsmooth edges; e_0 and e_2 are smooth edges; f_0 and f_2 are BFs; and f_1, f_3, f_4, and f_5 are side faces. To determine whether an edge is smooth or nonsmooth, two surface normal vectors on both sides of the candidate edge, respectively, are compared. If both surface normal vectors point in the same direction, then this candidate edge is considered smooth and its adjacent face is considered a top fillet. The neighboring face of this top face can be obtained and considered the side face of this candidate edge. By contrast, if an angle exists between the two surface normal vectors, then this candidate edge is considered nonsmooth and its

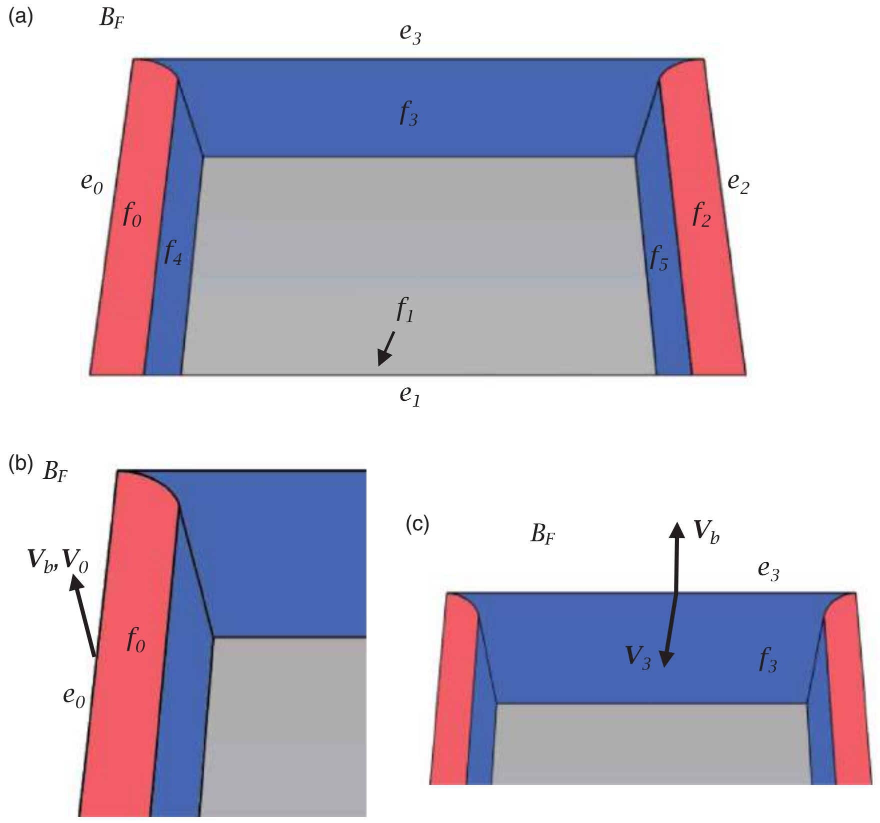


Figure 10. Topological data recorded on a loop, (a) a hole with two smooth edges and two non-smooth edges, (b) judgment of smooth edge, (c) judgment of non-smooth edge.

adjacent face is considered a side face. Fig. 10(b) indicates that the surface normal vectors $\boldsymbol{V}_b$ and $\boldsymbol{V}_0$ are identical; hence, e_0 is a smooth edge and its neighboring face f_0 is a top fillet. The neighboring face of f_0 is f_4, which is considered the side face of e_0. Fig. 10(c) indicates that the surface normal vectors $\boldsymbol{V}_b$ and $\boldsymbol{V}_0$ are different; therefore, e_3 is a nonsmooth edge and f_3 is a side face. The properties of the other two edges e_2 and e_1 can be determined in a similar manner. Two arrays are defined to record the mentioned topological data. The first array records the BFs of the loop and the corresponding edge for each of them, whereas the second array records all side faces of the loop and the corresponding edge for each of them. All side faces corresponding to the base face can be obtained using this data structure.

To determine whether the side faces inside a loop belong to a depression or a protrusion, a directional vector $\boldsymbol{v}_{bs}$ on one vertex of the loop is compared with the loop tangent vector $\boldsymbol{v}_l$. As illustrated in Fig. 11, the directional vector $\boldsymbol{v}_{bs}$ is defined as $\boldsymbol{v}_{bs} = \boldsymbol{v}_b \times \boldsymbol{v}_s$, where $\boldsymbol{v}_b$ is the surface normal of the base face and $\boldsymbol{v}_s$ is the surface normal of the side face. Let $d = \boldsymbol{v}_{bs} * \boldsymbol{v}_l$. If $d > 0$, then the corner is convex, indicating that the shape is a depression, whereas if $d < 0$, then the corner is concave, indicating that the shape is a protrusion.

When the feature is determined as a hole, additional steps must be executed to distinguish various hole types (i.e., blind holes, through holes, circular holes, and noncircular holes). First, the other edge on the side face must be examined to determine whether the bottom face and the BF exist. Blind holes can thus be distinguished from through holes by using such information. Similarly, circular and noncircular holes can be distinguished in this step. When the hole for a loop is determined, the system shifts to the next loop, and the entire search procedure is repeated. The rules for the determining protrusions are similar to those used for determining depressions. Additional rules may be required to classify and recognize different protrusions.

5. Results and discussion

Tab. 1 lists the number of faces, various types of detected loops, and computing time for seven CAD models. All types of outer and inner loops are detected and recorded in the table, but only the inner loops are used in feature recognition. The computing time shown in Tab. 1 indicates that the overall computational speed of the proposed algorithm is extremely fast; the CPU used for the computation is Intel Core i7 with 2.6 GHz clock speed. In this proposed algorithm, fillets must be recognized before loops are recognized. Therefore, the computing time listed in Tab. 1 includes the total time required for recognizing fillets and loops. The ratio shown in Tab. 1 indicates the average computing time per face. All ratio values listed in Tab. 1 are of the same order, indicating

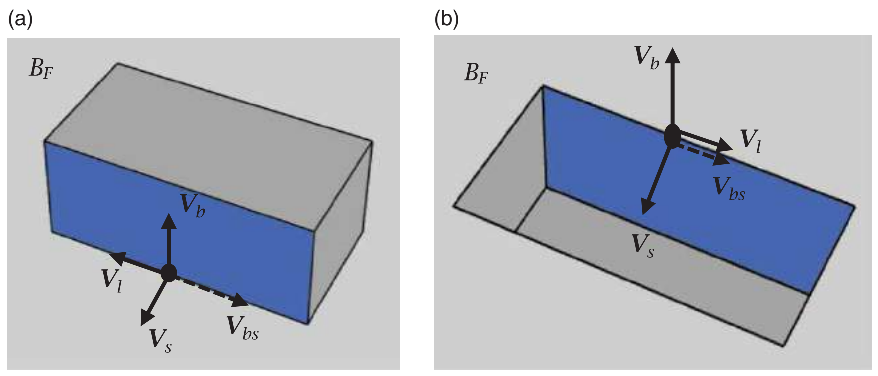


Figure 11. Distinction of extrusion and hole, (a) $\boldsymbol{V}_{bs}$ and $\boldsymbol{V}_l$ are in the opposite direction for an extrusion, (b) $\boldsymbol{V}_{bs}$ and $\boldsymbol{V}_l$ are in the same direction for a hole.

Table 1. Various loop types recognized and computing time required for seven CAD models.

		Single loop		Virtual loop				
Case	No. faces	Inner loop	Outer loop	Inner loop	Outer loop	Multivirtual inner loop	Computing time(s)	Ratio*(x10^{-5})
1	103	7	103	1	24	6	0.187	1.82
2	152	20	152	0	34	0	0.189	1.24
3	223	152	223	12	89	0	0.481	2.16
4	364	27	364	0	96	5	0.761	2.09
5	416	162	416	0	388	32	0.414	1.00
6	1275	393	1275	27	421	0	2.317	1.82
7	640	21	640	9	150	6	0.994	1.55

CPU:Intel Core i7 2.6 GHz, RAM: 6G *Ratio = Computing time/No. faces that the proposed algorithm runs in O(n) time, where n represents the number of faces in the CAD model.

Fig. 12 depicts the loops identified for the first six cases, where the numbers beneath each plot represent the number of single inner loops, virtual inner loops, and multivirtual inner loops, respectively. The results shown in Fig. 12 indicate that all loops can lie on any type of face and across multiple faces. Moreover, almost all faces are filleted at their boundaries. Therefore, these results show that the proposed loop recognition method can be implemented in real CAD models to extract all loop types and obtain the topological relationships among edges and faces neighboring all loops.

Fig. 13 illustrates the results of depression- and protrusion-recognition process obtained using the proposed algorithm for Case 7 in Tab. 1. Four hole types are identified, namely circular through holes, circular blind holes, noncircular through holes, and noncircular blind holes; these holes are shown in deep blue, red, aqua blue, and yellow, respectively. Moreover, the detected protrusion (shown in deep blue) is a cylindrical surface lying on multiple base faces. This type of simple protrusion, extruded from either a single base face or multiple base faces, can be detected easily using the proposed loop data structure.

The proposed loop- and hole-recognition algorithms can also be used for boss recognition. A boss, a type of protrusion, is generally more complex in shape and more difficult to recognize than holes and loops. Fig. 14 depicts the basic structure of a boss, which comprises a tube and hole as well as multiple ribs. In boss recognition, the proposed loop- and hole-recognition algorithms are used to recognize the tube and hole first. A rib recognition algorithm can then be applied to detect all ribs of the target boss. The composition of a boss can be highly complex (e.g., ribs with virtual faces, multiple base

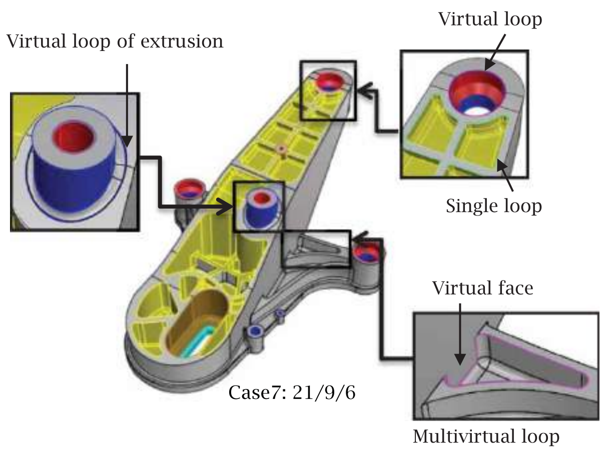


Figure 13. Combination of loop and hole/protrusion recognition algorithms.

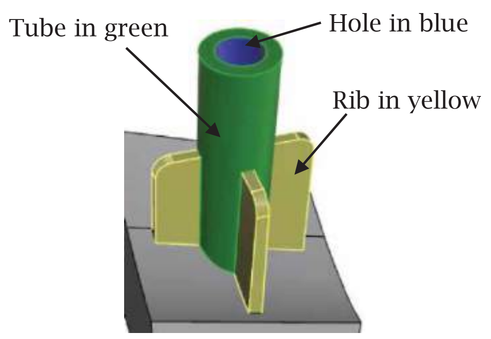


Figure 14. Basic structure of boss.

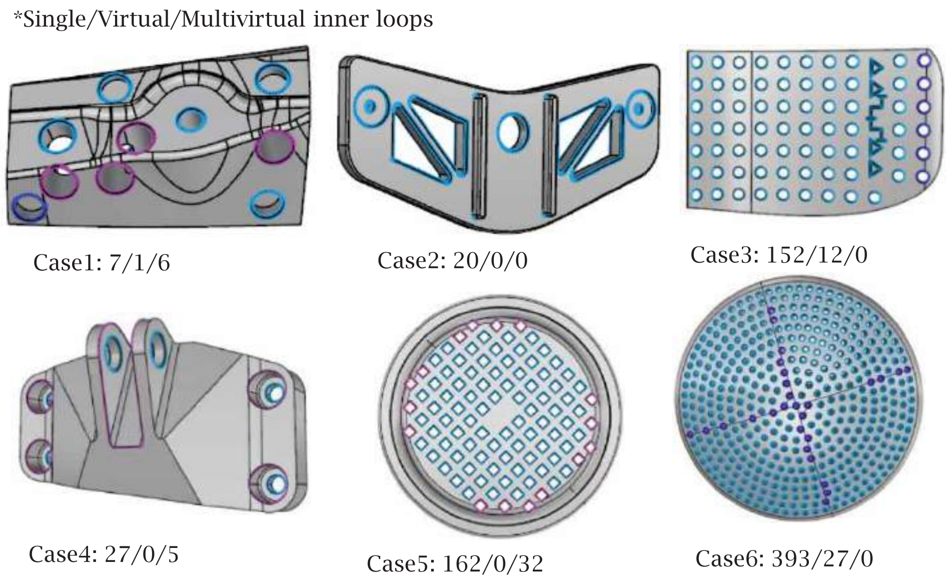
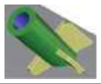
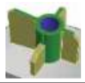
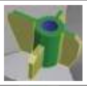
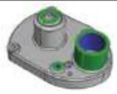


Figure 12. Various types of inner loop recognized, single, virtual and multivirtual inner loops are printed in water blue, blue and purple, respectively.

Table 2. Various boss types recognized for nineteen CAD models.

No.	Case	No. Boss	Success, Failure
1		1	1, 0*
			1, 0*
2		1	1, 0
			1, 0
3		1	1, 0
			0, 1
4		1	1, 0
			0, 1
5		1	1, 0
			0, 1
6		1	1, 0
			0, 1
7		1	1, 0
			1, 0
8		1	1, 0
			1, 0
9		1	1, 0
			0, 1
10		6	6, 0
			6. 0
11		4	4, 0
			0, 4
12		2	0, 2
			0, 2
13		4	0, 4
			0, 4
14		4	4, 0
			0, 4
15		1	1, 0
			1, 0
16		2	2, 0
			0, 2
17		1	1, 0
			0, 1
18		1	1, 0
			0, 1
19		16	16 ,0
			12, 4

* 1st and 2nd lines on each case represent results from our study and CADdoctor, respectively.

faces, or hybrid type). We propose a boss recognition algorithm according to the proposed loop data structure for addressing different boss types. Tab. 2 shows a comparison of the boss recognition results of the proposed method with CADdoctor, a well-known commercial CAD system, for 19 CAD models. The parameters "success" and "failure" shown in Tab. 2 indicate the number of bosses detected successfully and erroneously, respectively. The values shown on the first and second lines of each case represent the results of the proposed method and CADdoctor, respectively. The proposed method demonstrates superior performance to that of CADdoctor because CADdoctor registers erroneous detections in 13 cases (i.e., Cases 3–6, 9, 11–14, and 16–19), whereas the proposed method registers erroneous detections in only two cases (i.e., Cases 12 and 13).

In CADdoctor, several restrictions are imposed on the definition of a boss; for example, the tube must be cylindrical and the height of the ribs must be lower than that of the tube. This is why CADdoctor fails in a higher number of cases compared with the proposed method. Regarding the proposed method, the primary reason for the failure in Cases 12 and 13 is that real CAD models are generally complex and variable. The examples in Cases 12 and 13 have chamfers and unexpected tiny faces, which are not consistent with the definition of boss in the proposed method. In summary, the proposed virtual and multivirtual loop data enable detecting various boundary contours on a single face or across multiple faces. The concept is simple and direct, and the proposed loop data can serve as the basis for developing various types of feature recognition algorithms.

6. Conclusion

This study proposes a virtual loop recognition method for detecting all loop types in a B-rep model and for enabling the recognition of depression and protrusion features lying across multiple faces. The most noteworthy contribution of the proposed method is that it can be used to detect loops of G^0 or G^1 continuity across multiple faces, which is beyond the capability of current B-rep data structures. A feature recognition algorithm is also developed using the data of the detected loops for recognizing several types of depression and protrusion. The integrated flowchart of the proposed algorithm should be as follows. Chamfers and fillets should be recognized first, followed by loop recognition. The loop data can then be used for recognizing holes, pockets, ribs, protrusions, and bosses. Future studies should consider integrating the proposed method with other feature recognition algorithms to ensure a highly complete detection of general feature types.

ORCID

Jiing-Yih Lai http://orcid.org/[0000-0002-0495-0826]
Ming-Hsuan Wang http://orcid.org/[0000-0003-4947-4218]
Yu-Kai Chiu http://orcid.org/[0000-0001-7379-7872]
Chia-Hsiang Hsu http://orcid.org/[0000-0001-5763-4766]
Yao-Chen Tsai http://orcid.org/[0000-0002-3408-8835]
Chung-Yi Huang http://orcid.org/[0000-0002-0848-0139]

References

[1] Ansaldi, S.; De Floriani, L.; Falcidieno, B.: Geometric modeling of solid objects by using a face adjacency graph representation, Computer Graphics (ACM), 19(3), 1985, 131–139. http://dx.doi.org/10.1145/325334.325218
[2] Babic, B.; Nesic, N.; Miljkovic, Z.: A review of automated feature recognition with rule-based pattern recognition, Computers in Industry, 59(4), 2008, 321–337. http://dx.doi.org/10.1016/j.compind.2007.09.001
[3] Chung, K.; Lee, K.; Kim, T.: Recognition of pass features for automatic parting surface generation in injection moulds, Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering, 216(5), 2002, 783–796. http://dx.doi.org/10.1243/0954405021520292
[4] Cui, X.; Gao, S.; Zhou, G.: An efficient algorithm for recognizing and suppressing blend features, Computer-Aided Design and Applications, 1(1–4), 2004, 421–428. http://dx.doi.org/10.1080/16864360.2004.10738284
[5] Ismail, N.; Bakar, N. A.; Juri, A. H.: Recognition of cylindrical and conical features using edge boundary classification, International Journal of Machine Tools & Manufacture, 45(6), 2005, 649–655. http://dx.doi.org/10.1016/j.ijmachtools.2004.10.008
[6] Ismail, N.; Abu Bakar, N.; Juri, A. H.: Recognition of cylindrical-based features using edge boundary technique for integrated manufacturing, Robotics and Computer-Integrated Manufacturing, 20(5), 2004, 417–422. http://dx.doi.org/10.1016/j.rcim.2004.03.004
[7] Joshi, N.; Dutta, D.: Feature simplification techniques for freeform surface models, Transactions of ASME, Journal of Computing and Information Science in Engineering, 3, 2003, 177–186. http://dx.doi.org/10.1115/1.1603307
[8] Joshi, S.; Chang, T. C: Graph-based heuristics for recognition of machined features from a 3D solid model, Computer-Aided Design, 20(2), 1988, 58–66. http://dx.doi.org/10.1016/0010-4485(88)90050-4
[9] Kyprianou, L. K.: Shape classification in computer aided design, Ph.D. Thesis, University of Cambridge, Cambridge, 1980.
[10] Li, J.; Sun, L.; Peng, J.; Du, J.; Fan, L.: Automatic small depression feature recognition from solid B-rep models for meshing, International Conference Electrical and Control Engineering (ICECE), 2011, 4386–4389. http://dx.doi.org/10.1109/ICECENG.2011.6057432
[11] Li, J.; Tong, G.; Shi, D.; Geng, M.; Zhu, H.; Hagiwara, I.: Automatic small blend recognition from B-rep models for analysis, Engineering with Computers, 25(3), 2009, 279–285. http://dx.doi.org/10.1007/s00366-009-0127-4
[12] Lim, T.; Medellin, H.; Torres-Sanchez, C.; Corney, J. R.; Ritchie, J. M.; Davies, J. B. C.: Edge-based identification of DP-features on free-form solids, IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(6), 2005, 851–860. http://dx.doi.org/10.1109/TPAMI.2005.118
[13] Lu, Y.; Gadh, R.; Tautges, T. J.: Feature based hex meshing methodology: feature recognition and volume decomposition, Computer-Aided Design, 33(3), 2001, 221–232. http://dx.doi.org/10.1016/S0010-4485(00)00122-6
[14] Modex3D, https://moldex3d.app.box.com/s/88ar5t9mcdhu1dnolpe3, accessed on 24-03-2015.
[15] Requicha, A. A. G., Voelcker, H. B.: Solid modeling: current status and research directions, Computer Graphics and Application, 3(7), 1983, 25–36. http://dx.doi.org/10.1109/MCG.1983.263271
[16] Rhinoceros, http://www.rhino3d.com, accessed on 24-03-2015.
[17] Subrahmanyam, S.; Wozny, M.: An overview of automatic feature recognition techniques for computer-aided process planning, Computers in Industry, 26(1), 1995, 1–21. http://dx.doi.org/10.1016/0166-3615(95)80003-4
[18] Tian, F.; Tian, X.; Geng, J.; Li, Z.; Zhang, Z.: A hybrid interactive feature recognition method based on lightweight model, 2010, 113–117. http://dx.doi.org/10.1109/icmtma.2010.428
[19] Venkataraman, S.; Sohoni, M.: Blend recognition algorithm and applications, The Sixth ACM Symposium on Solid Modeling and Applications, 2001, 99–108. http://dx.doi.org/10.1145/376957.376970
[20] Verma, A. K.; Rajotia, S.: A review of machining feature recognition methodologies, International Journal of Computer Integrated Manufacturing, 23(4), 2010, 353–368. http://dx.doi.org/10.1080/09511921003642121
[21] Zhang, C.; Zhou, X.; Li, C.: Feature extraction from freeform molded parts for moldability analysis, International Journal of Advanced Manufacturing Technology, 48(1–4), 2010, 273–282. http://dx.doi.org/10.1007/s00170-009-2273-7