COMPUTER-AIDED DESIGN & APPLICATIONS, 2017
VOL. 14, NO. 4, 450–463
http://dx.doi.org/10.1080/16864360.2016.1257187

Boss recognition algorithm for application to finite element analysis

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

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

ABSTRACT
In finite element analysis (FEA), computer-aided design (CAD) models must be converted into solid meshes so that the solver can perform the desired analysis and simulation. Hexahedral and prism meshes are better than tetrahedral meshes, but are inherently more complex and difficult to generate. The purpose of this study was to propose an approach based on feature recognition for generating better quality solid meshes for FEA applications. Particularly, this study focused on the development of a boss recognition algorithm, the output of boss data for meshing, and the development of a process for automatic boss meshing. The proposed boss recognition method contains three parts: the preliminary functions, the data base, and the boss recognition. The first two parts provide a framework that can be used for other types of feature recognition. The core of the boss recognition is the tube recognition, which involves five main steps of validation. The output data of a boss includes the rib, tube, and hole data, which record not only feature data, but also the meshing data that can be used for automatic meshing. The meshing of a boss is divided into rib meshing and tube meshing. Both hexahedral meshes and prism meshes may be used depending on the regularity and orthogonality of the shape. This paper provides a detailed description of the proposed algorithm and presents several examples to illustrate its feasibility.

KEYWORDS
Boss recognition; Feature recognition; Meshing; B-rep model

1. Introduction

In finite element analysis (FEA), computer-aided design (CAD) models must be converted into solid meshes so that the solver can perform the desired analysis and simulation. Basic solid elements include tetrahedrons, pyramids, prisms, and hexahedrons, each composed of triangular and quadrilateral faces. Tetrahedral meshes are commonly used because they can be generated automatically, but they can provide a result with a low accuracy and a huge amount of meshes are required to describe a given object shape. In particular, tetrahedral meshes are unsuitable for representing thin areas or corners. By contrast, hexahedral and prism meshes are characterized by higher accuracy, convergence, and application specificity, rendering them more preferable to tetrahedral meshes [3]. However, hexahedral meshes are inherently more complex and difficult to generate.

The methods that are widely used for generating all-hexahedral meshes or the combination of hexahedral and prism meshes are mapping/submapping, meshing primitives, and sweeping. These all require the recognition and decomposition of a CAD model into some recognized patterns, followed by the conversion of each of them into meshes using a meshing algorithm [21]. Although considerable progress has been made on automatic decomposition and meshing, none of the methods is currently reliable enough for industrial applications because the CAD models are more complex, variable, and unpredictable. It is necessary to combine multiple meshing algorithms and mesh types to deal with practical problems.

Feature recognition has been considered an important tool that fully integrates CAD, computer aided manufacturing (CAM), and computer aided engineering (CAE). The aim of feature recognition is to convert the CAD data of a part into a set of features required by downstream finite element analysis or manufacturing applications, such as process planning, and NC code generation and inspection. However, features suitable for FEA applications may not be the same as those suitable for manufacturing applications. Depressions, e.g., holes, slots, and pockets, are typical machining features. Those used in FEA are more like design features, including both depression and protrusion types. Some examples of protrusion types include ribs and columns.

Industrial CAD models are usually more complex and variable in geometry and topology. Several common CAD model characteristics are often ignored in the

CONTACT Ming-Hsuan Wang a7512cs@hotmail.com

available feature-recognition algorithms. First, the fillet, a common feature in CAD models, can complicate the topological relationship of adjacent faces. When filleting is performed at the transition of multiple faces, the topological data become even more complex. Second, irrational geometric data frequently occurs in CAD models. Typical cases include improper continuity between adjacent smooth surfaces; unexpected subdivision of edges, planes, and surfaces; and erroneous data in geometry and topology. The feature recognition algorithm may fail unless these conditions are handled beforehand. Finally, virtual planes or surfaces, appearing frequently in CAD models, are seldom handled. A virtual plane or surface is one that extends to other features or part shapes. The existence of virtual planes or surfaces makes the feature recognition more difficult.

Many plastic products are designed as thin parts. Bosses on such products provide the function to lock two parts together with screws. In injection molding analysis, some tiny features may be suppressed before EFA is performed. Typical tiny features are such as tiny fillets because their size is usually much smaller than their neighboring faces. Bosses cannot be suppressed in the analysis because they belong to a functional design. A boss is a composite feature, often comprising a hole, ribs, and a tube. The hole is located on the center of the tube, and its shape can either be circular or polygonal. The tube is long and slender for holding the screw. Ribs are connected to the outer face of the tube to strengthen the structure. Most bosses generally have ribs, but some may not. If the constructing elements of a boss are processed independently, e.g., recognizing and meshing the tube, the hole, and the ribs separately, then the topology of the meshes at the transitions between features may become a problem. If fillets appear on the surface boundary, this problem may become even more severe. In order to recognize a boss, it is necessary to recognize the hole, ribs, and tube first, and then record all topological data related to these features. To mesh a boss, a decomposition algorithm must be developed, and the associated topological relations among them must be built. Each of the decomposed regions can then be converted into solid meshes, composed of either all-hexahedral meshes or the combination of hexahedral and prism meshes.

The purpose of the present study was to propose an approach based on feature recognition for generating better quality solid meshes for FEA applications emphasizing on plastic injection molding. Although feature recognition has been studied for a long time, its application in FEA has not been investigated extensively. Particularly, this study focused on the development of a boss recognition algorithm, the output of boss data for meshing, and the development of a process for automatic boss meshing. The approach emphasizes the recognition of all bosses in a CAD model, as well as the computation of all corresponding data needed to automatically generate solid meshes. The boss recognition process contains three parts: the preliminary functions, the data base, and the boss recognition. The first two parts provide a general framework that can be used for other types of feature recognition. The core of the boss recognition is the tube recognition, which involves five validation steps. The output data of a boss includes the rib, tube, and hole data, which record not only feature data, but also the meshing data that can be used for automatic meshing. The meshing of a boss is primarily divided into rib meshing and tube meshing. Either hexahedral meshes or prism meshes can be used on the ribs and tube, depending on the regularity and orthogonality of the shape. Once a boss can be recognized and meshed appropriately, similar algorithms can be developed for simpler features. Several results both in boss recognition and meshing are presented to demonstrate the feasibility of the proposed method. The primary contribution of the proposed method is to recognize various types of bosses, ranging from basic type to more complex ones with virtual faces or multiple base faces. The meshing algorithm is not the primary concern in this study although a simple concept for meshing will be described. More strict meshing strategies for different kinds of bosses must be developed.

2. Literature review

Most feature recognition studies have been related to CAD, CAM, and computer-aided process planning (CAPP). Shah et al. [27] gave an in-depth review of the development of feature recognition techniques between 1976 and 2000. They classified the techniques into six basic groups: topological, heuristic, symbolic, volumetric and process-centric, and hybrid. Each of these was further divided into several sub-groups. They emphasized the review of the following sub-groups: graph based and hint based methods, convex hull decomposition, and volume decomposition-recomposition techniques, because these seem to be more successful ones. Topological data structures are the foundation of various feature recognition algorithms and should be addressed first. Ansaldi et al. [2] proposed a face adjacency graph (FAG) to record the adjacent relationship of faces and edges. An edge is denoted by a node on the graph and its adjacent faces are denoted by the links between nodes. The feature recognition is performed by comparing the FAG data of the testing shape with that of the reference feature. Joshi et al. [12] added a property at the edge attribute, i.e., concave or convex angle between two neighboring faces, for improving the shape-comparison efficiency. The upgraded data structure is called the AAG.

Marefat et al. [23] proposed a data structure, called the extended attributed adjacency graph (EAAG), to deal with primitives that intersect with each other. The depressions in the part are represented as cavity graphs, in which the links reflect the concavity of the intersection between two faces, and the node labels reflect the relative orientation of the faces comprising the depression. Lu et al. [22] and Li et al. [19] proposed a hybrid data structure by combining topological data structures (e.g., AAG and EAAG) and heuristic rules (e.g., hint-based or rule-based) for feature recognition. They defined seed faces on the model and searched for the target features in accordance with the proposed hybrid data structure. Li et al. [18] further proposed the HAAG to describe the topological and geometric information between adjacent edges and faces. The main difference between the EAAG and the HAAG is that the HAAG provides more geometric information, such as the face normal, the angle between two faces, and the edge length, enabling a search for free-form surfaces.

Essentially, the above-mentioned data structure records the adjacent information of lower level entities such as faces, edges, and vertices. Various methods are available for the extraction of higher level depression and protrusion features. Hariya et al. [9] developed a design support system for checking whether the CAD shape satisfies the design rules by recognizing the feature shapes of the CAD model. A rule-based process was developed for recognizing several common feature shapes, e.g., bosses, ribs, fillets, and holes. However, their algorithm is valid only for simple and isolated feature shapes. Flavien et al. [7] proposed a method to extract generative processes from a given B-rep shape as a high-level shape description. Any generative process of the construction graph can preserve the reliability of the corresponding volume. It can be used in FEA to obtain connections between idealized primitives of the construction graph.

Abu et al. [1] proposed an attribute-based technique for feature recognition based on different characteristics assigned to each feature, such as the total number of faces, the total number of edges, the types of faces, and the protrusion or depression attribute. However, the rules are too simple to be used for industrial CAD models. Jun et al. [13] studied the issue of extracting geometric features from a set of scanned points using reverse engineering. An edge detection algorithm was developed to detect feature edges. Then, an artificial neutral network was employed for extracting features such as pockets, steps, slots, bosses, holes and blocks. The features that can be extracted are isolated features only.

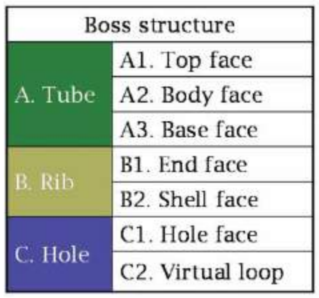
Zhang et al. [29] proposed a region-based method for recognizing protrusion and depression features. Symbolic computation was employed to characterize the curvature properties of the free-form surface, which can help to decompose the surface into regions. A rule-based approach was then employed to recognize protrusion and depression features in terms of specific geometrical and topological relations.

Cuillière et al. [6] proposed an approach of the automatic 3D mesh generation problem featuring a pre-optimization scheme to identify geometric features. More precise knowledge on how geometric features were identified and used was provided, which can be used to calculate a nodal density field across the parts that is more interesting for analysis. Li et al. [17] proposed an edge-based approach for recognizing small depression features during mesh generation. Edges are first classified as convex, concave, or smooth shapes. Convex inner loops are then formed with edges and used for recognizing depression features. Specifically, their algorithm can handle fillets and chamfers for simple depression types.

Lim et al. [20] proposed an algorithm for recognizing depression and protrusion features (DP features) on free-form 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. Ismail et al. [10,11] proposed a technique called edge boundary classification (EBC) for recognizing simple and interacting cylindrical and conical 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 was formed in terms of three test points. The first two points were on the edge loop and the third point was the midpoint of the two. A loop-up table in terms of the EBC pattern was provided to detect different feature types.

3. Overall method

The basic structure of a boss is depicted in Fig. 1, which primarily contains three components: a tube, a hole, and ribs. The tube, the main component of the boss, consists of the top face, body face, and base face. The base face is where the boss resides. The top and body faces determine the outer shape of the tube. The hole is located in the center of the tube and consists of a hole face and a loop, where the loop determines the boundary of the hole. Ribs are attached on the body and base faces. A rib is composed of end faces and shell faces, located on the side and top of



Boss structure	
A. Tube 🟩	A1. Top face
	A2. Body face
	A3. Base face
B. Rib	B1. End face
	B2. Shell face
C. Hole 🟦	C1. Hole face
	C2. Virtual loop

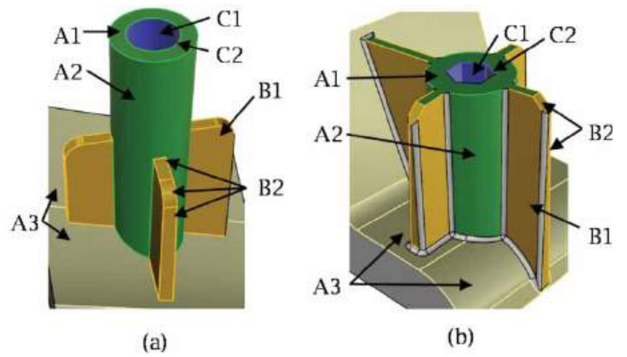


Figure 1. Basic structure of the boss in this study: (a) Case 1, (b) Case 2.

the rib, respectively. Some of the features as mentioned above are available in CAD feature tree during the design process. However, we still need to develop an independent boss recognition algorithm owing to the following reasons: (1) the most important issue here is not just the recognition of individual feature, but the extraction of specific attributes on each feature and the correlation of them so that they can be used for automatic mesh generation. Such information does not exist in CAD feature tree. (2) It is quite common that some of the features are not recognized as expected in CAD feature tree. For example, holes and ribs may be common features in CAD feature tree, but tubes may not be considered as one feature. Moreover, bosses are often not considered as a feature in CAD feature tree. (3) Some formats of the CAD models do not bring the feature tree, such as STEP, IGES, etc. The recognition process proposed in this study is based on a B-rep structure, which provides a high-level shape description to achieve the recognition of composite features.

Bosses on CAD models may be coupled with other shapes or features, becoming more complex. Fig. 2 depicts several forms of bosses, including a basic type, one with virtual faces on the ribs, another with multiple base faces, and a hybrid type. For the basic type, the height of each rib and the shape of the hole can be varied. For the type with virtual faces, some of the rib faces are coupled with other features or shapes. For the type with multiple base faces, the tube and ribs are located on multiple base faces. For the hybrid type, at least two of the aforementioned conditions appear simultaneously. Ismail et al. [10] proposed an algorithm for recognizing the basic type of bosses, but the more complex types were not considered in their algorithm.

Fig. 3 depicts the overall flowchart of the proposed feature recognition method for bosses. It is primarily

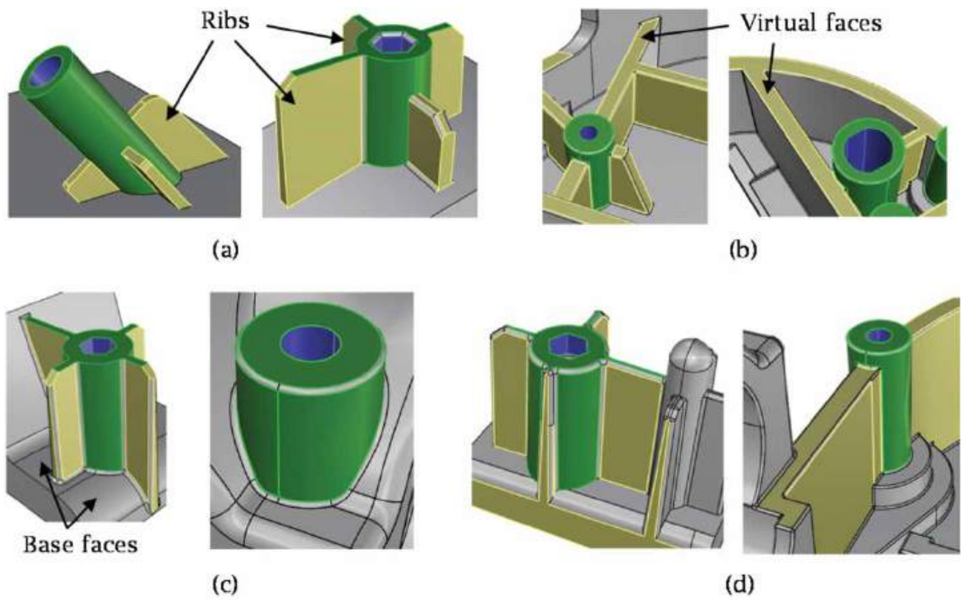


Figure 2. Classification of a boss: (a) Basic type, (b) Ribs with virtual faces, (c) Multiple base faces, (d) Hybrid.

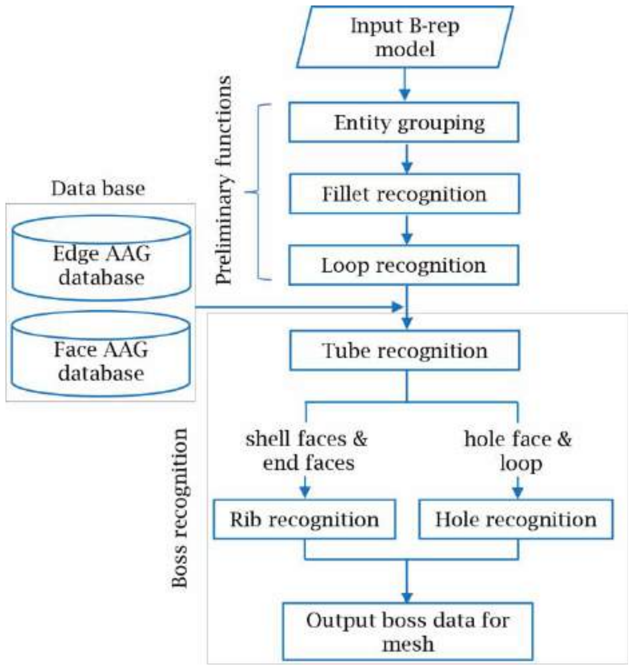


Figure 3. Overall flowchart of the proposed method for boss recognition.

divided into three parts: the preliminary functions, the data base, and the boss recognition. The preliminary functions are employed to deal with irrational geometry and to recognize fillets and loops. Three main functions in this stage are entity grouping, fillet recognition, and loop recognition. Entity grouping reorganizes irrational geometry by grouping the faces and edges with similar attributes. A face or edge may be divided into two or more pieces base on a different drawing process or conversion of files. Entity grouping can record those elements of similar attributes such that they can be processed simultaneously, without changing the geometric data. The entity grouping consists of plane grouping, edge grouping, and surface grouping.

Fillets, also called blend faces, are recognized using a fillet recognition algorithm. Fillets are generally long and narrow compared to their neighboring faces. They should be recognized first for providing the topological data of their neighboring faces and edges. The topological data can be used for searching the neighboring part faces. Moreover, when the diameter of a fillet is very small, it may be suppressed before the meshing in injection molding analysis. As a rule of thumb, if the radius of a fillet is smaller than one third of the distance between two mesh nodes, this fillet may be suppressed. Otherwise, it will result in poor tiny meshes. When a fillet needs to be meshed, it should be separated and meshed independently. A fillet recognition algorithm is employed to recognize various types of blend faces. Loops are used to help the recognition of other features. A loop may describe the boundary of a depression or protrusion feature, and hence can be employed for searching the target feature. A loop recognition algorithm is employed to recognize all types of loops. The data base is established for boss recognition, and is primarily composed of an edge AAG and a face AAG. However, additional topological information of edges and faces are also recorded in accordance with the needs of the proposed algorithm.

The proposed boss recognition process is composed of the following three steps: tube recognition, rib recognition, and hole recognition. First, the tube recognition records the relations between the ribs and hole, the body, and the base faces. It outputs shell and end faces to search for ribs, and a loop and hole face to search for the hole. Rib recognition and hole recognition are then implemented separately to identify the desired features. After these steps, all features and data related to a boss are now available. As we intend to employ hexahedral and prism elements for representing a boss, its meshing algorithm should be developed. The boss data can finally be analyzed in terms of the needs of the meshing algorithm.

4. Preliminarily functions and database

The entity grouping process consists of the following three types of grouping:

(1) *Plane grouping*: all adjacent planes with the same surface normal should be considered a group. Two neighboring planes of the same group must be positioned continuous and have the same surface normal. An algorithm is proposed to group all neighboring planes that satisfy the abovementioned conditions.
(2) *Edge grouping*: all adjacent edges on a face with G^1 continuity should be considered a group. For example, if a surface consists of four edges, and all adjacent edges are G^0 continuous, then each edge is individually considered a group. In Fig. 4, the edges e_2

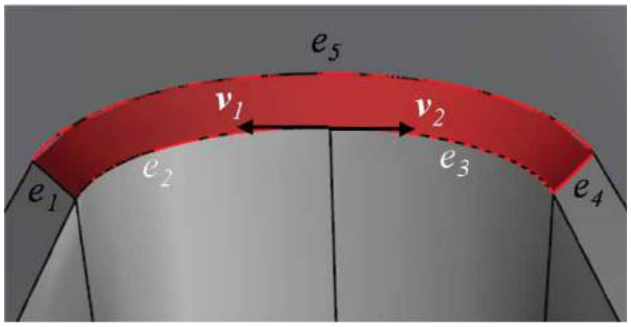


Figure 4. Edge grouping algorithm.

and e_3 are grouped because their tangent vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ at the junction are parallel to each other. The topological elements in a B-rep model include the vertex, trim, edge, loop, and face. Trims and edges represent the boundary profile of a face in the parametric and 3D domains, respectively. The difference between trims and edges is that edges are non-directional, whereas trims are directional. Therefore, an algorithm is employed to group trims and edges on the same face when they are G^1 continuous.

(3) *Surface grouping*: all adjacent surfaces with the same curvature should be considered a group. This is done by checking the curvature of the common edge of two neighboring surfaces. A particular application of this algorithm is when a fillet is divided into two surfaces in the CAD data. The neighboring information of a fillet cannot be found correctly if two fillet surfaces are not grouped.

Blending is a common function used in 3D CAD modeling to smooth sharp edges. Typical blend faces are edge blend faces (EBFs) and vertex blend faces (VBFs). If the feature before surface blending is an edge, the blend face is called an EBF, whereas if the feature before surface blending is a vertex, the blend face is called a VBF. A VBF can exist alone, as shown in Fig. 5(a), or as several VBFs connected to each other, as shown in Fig. 5(b). Other mixed blend faces may also exist [5],[16],[28], but are not considered here, as they will not affect the recognition of bosses. A fillet recognition algorithm was proposed to identify various types of fillets, with the topological relationships and attributes of edges and faces related to fillets recorded [15]. In this algorithm, a procedure with five conditions was proposed to detect general fillets from the B-rep model, and then put them on a list. A VBF detection algorithm with three criteria was then employed to extract all VBFs from the list; the faces left on the list were considered EBFs.

Three types of loops can be defined in a B-rep model: single, virtual, and multivirtual loops. A loop is essentially formed by trims. A single loop is the current loop recorded in the B-rep model. A virtual loop lies across faces that are at least G^1 continuous. Finally, a multivirtual loop lies across faces that are either G^0 or G^1 continuous. The proposed loop structure provides a more complete data structure for recognizing various types of features. There are two kinds of loops on each face boundary, the outer and inner loops, formed in a counterclockwise and a clockwise direction, respectively (Fig. 6(a)). The features that can be identified using the inner loop may be depression or protrusion features. For example, in Fig. 6(a), a hole is on the top face of the boss, and the yellow trim is an inner loop belonging to the top face. In Fig. 6(b), the pocket lies across two faces, and the yellow loop was not recorded in the B-rep model. Finally, in Fig. 6(c), a hole lies across two faces that are not connected smoothly. Using the yellow loop, the hole can be easily recognized and the topological data can be obtained accordingly. A loop recognition algorithm was proposed to identify various types of loops, with the topological relationships and attributes of edges and faces recorded [14]. In this algorithm, fillets were detected first. A clustering method was then employed for partitioning non-fillets into groups. All constructed edges of an inner loop within a group should be at least G^1 continuous, which formed a virtual loop. Some loops might be across multiple groups; accordingly, a procedure was proposed to search for multivirtual loops.

The edge and face AAG database is generated to provide the topological and geometric information among edges and faces, respectively. All groups, fillets, and loops as mentioned above have their own data structure. However, the corresponding attributes are also recorded in the edge and face AAG database. The edge AAG is shown in Tab. 1, and contains seven attributes. The first five attributes represent the topological and geometric

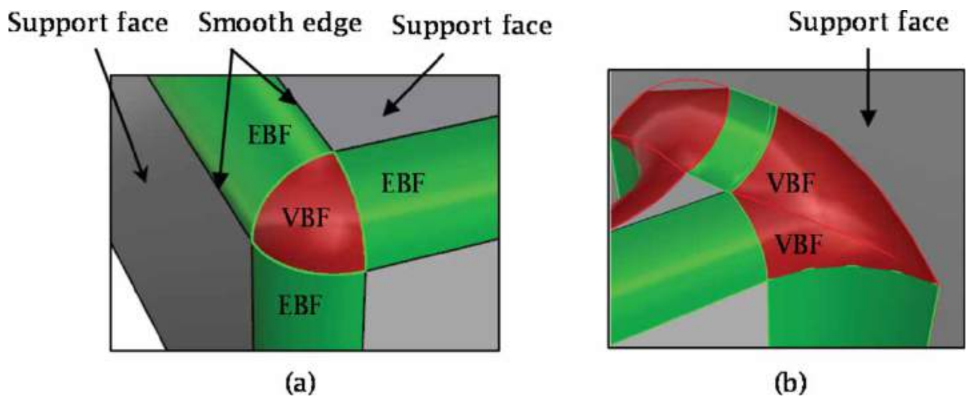


Figure 5. Various types of blend faces: (a) Edge blend faces and a single vertex blend face, (b) Multiple vertex blend faces.

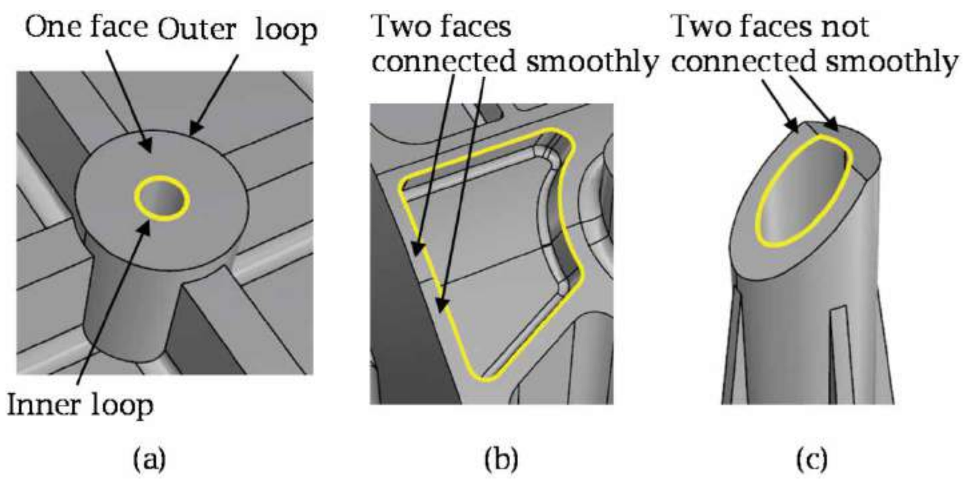

One face
Outer loop
Inner loop
Two faces connected smoothly
Two faces not connected smoothly
(a)
(b)
(c)

Figure 6. All types of loops, where inner loops are used for feature recognition, and outer loops are used for evaluating multivirtual loops: (a) Single loop, (b) Virtual loop, (c) Multivirtual loop.

Table 1. Edge AAG database.

No.	Attribute	Remark
1	Edge index	Index of the edge
4	Geometry type	Line or curve
2	Convexity property*	Convexity status of two faces neighboring the edge
3	Convexity angle	Angle of two faces neighboring the edge
5	Adjacent faces	Indices of two neighboring faces
6	Group index**	Index of the group to which the edge belongs
7	Loop index***	Index of the loop to which the edge belongs

*11 convexity properties: Non-smooth concave or convex edge, plane-surface concave or convex edge, surface-surface concave or convex edge, sphere concave or convex edge, plane edge, reflection edge, and free edge
**0: Does not belong to any group; Others: Group index
***Can be single, virtual or multivirtual loop

Table 2. Face AAG database.

No.	Attribute	Remark
1	Face index	Index of the face
2	Geometry type	Plane or surface
3	No. of loops	Number of loops on the face
4	No. of edges	Number of edges on the face
5	No. of convex edges	Number of convex edges on the face
6	No. of concave edges	Number of concave edges on the face
7	Group index*	Index of the group to which the face belongs
8	Blend-face type**	The type of blend face

*0: Does not belong to any group; Others: Group index
*0: Does not a blend face, 1: EBF, 2: Single VBF, 3: Multiple VBFs

properties of the edge itself. The sixth attribute records the group index of the edge, and the last attribute records the loop index of the edge. The second attribute in Tab. 1 represents the convexity property of the edge. It can further be divided into 11 sub-types. Fig. 7 depicts four examples of the convexity property and the convexity angle for each of them. The face AAG is shown in Tab. 2, which contains eight attributes. The first six attributes represent the topological and geometric properties of the face itself. The seventh attribute records the group index of the face, and the last attribute records the type of blend face, if it exists. For the second attribute, the geometric type, we only consider plane and surface types. Furthermore, for the third attribute, the loop number, we only record the number of loops on a face in the B-rep model. If a loop crosses multiple faces, the last attribute in the edge AAG database can be used to obtain the corresponding information.

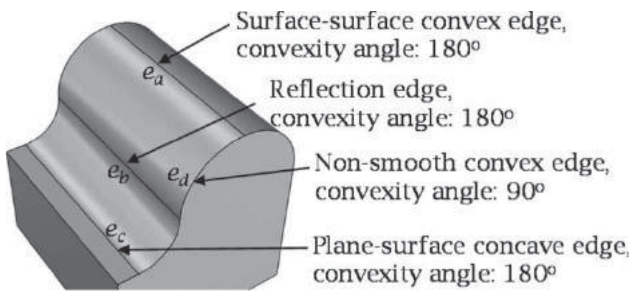

Surface-surface convex edge, convexity angle: 180°
Reflection edge, convexity angle: 180°
Non-smooth convex edge, convexity angle: 90°
Plane-surface concave edge, convexity angle: 180°
e_a
e_b
e_c
e_d

Figure 7. Four edges (e_a–e_d) showing different convexity properties and convexity angles.

5. Boss recognition algorithm

The boss recognition is performed loop by loop. For each loop, called a tested loop hereafter, the tube recognition is performed first. The overall flowchart of the tube recognition algorithm is shown in Fig. 8. Tube recognition involves five steps for validating the input loop data. If one of the five steps is not satisfied, no tube exists on the tested loop, and the check shifts to the next loop. By contrast, if all five steps are satisfied, a tube exists on the tested loop, and the data corresponding to the tube are recorded. The algorithm will yield a set of shell faces and end faces for rib recognition, and a loop and a hole face

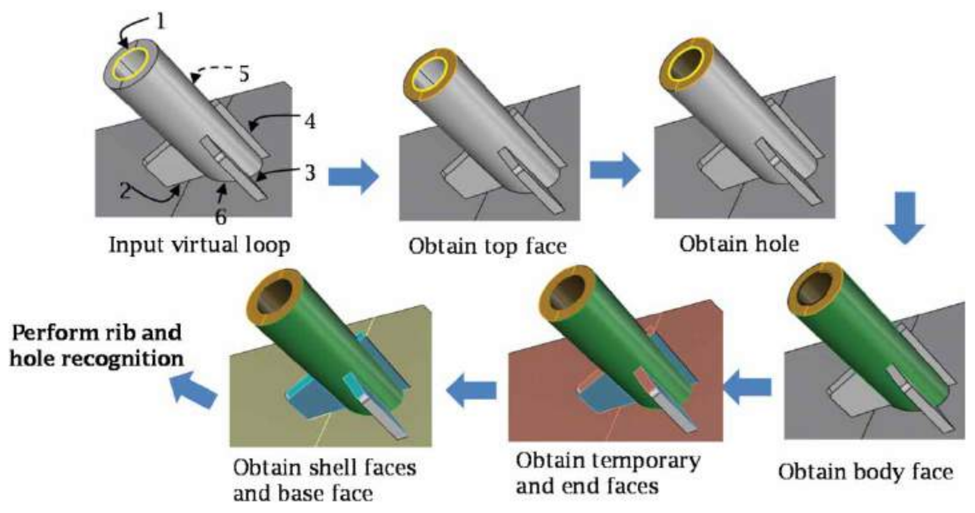

1
5
4
3
2
6
Input virtual loop
Obtain top face
Obtain hole
Obtain body face
Obtain temporary and end faces
Obtain shell faces and base face
Perform rib and hole recognition

Figure 8. Overall flowchart of the tube recognition algorithm.

for hole recognition. This procedure continues until all loops have been tested. For example, for the boss shown in Fig. 9, six loops (Fig. 9(a)) are recognized by the loop recognition algorithm and entered into the tube recognition algorithm. Among these loops, only Loop 1 satisfies all five steps, whereas Loops 2–6 do not satisfy, as highlighted in Fig. 9(b). Therefore, a tube exists on Loop 1. The five tube recognition steps are described as follows:

(1) Find all convex edges: First, the algorithm finds the face belonging to the tested loop and records it as the top face (green in Fig. 9(a)). Next, a hole is discovered by checking for convex edges on the tested loop, shown as the yellow loop in Fig. 9(a). Notably, the tube must have a hole at its center. If one of these edges is not convex, it does not satisfy the aforementioned condition, and all information recorded until that point is deleted. The process then moves to the next loop.

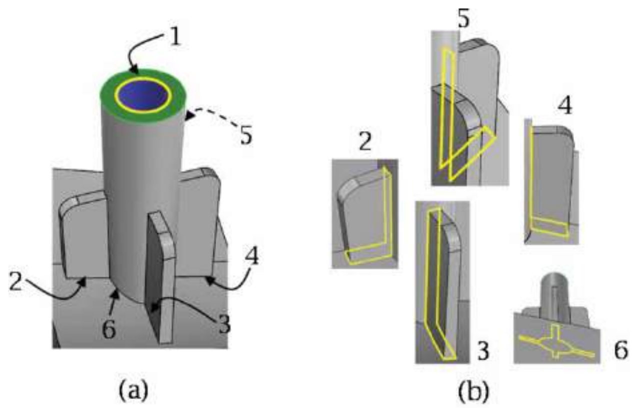


Figure 9. An example of the loop data input: (a) Six loops, (b) Highlight loops in 2–6 illustrate the loop recognition.

(2) Obtain a valid hole: The algorithm then checks for a hole and determines parameters such as the hole shape (circular or regular polygonal), radius, and center point. All edges on the hole boundary are composed of lines or arcs. If all edges are lines, vertices of the edges are averaged to determine the center point. If all vertices are equidistant from this point, the hole radius and faces, shown in blue in Fig. 9(a), are recorded. If all edges are arcs and can form a circle, the radius and hole faces are recorded. If the edges do not fit either of the two aforementioned conditions, all current information is deleted, and the process moves to the next loop. Moreover, if fillets exist on the hole boundary, the hole faces can be searched indirectly. As all fillets have been detected beforehand and two neighboring edges of a fillet are recorded, it is easy to find the opposite edge of an edge on a fillet. Therefore, whenever an edge on the hole boundary is detected, we can check whether its neighboring face is a fillet or not. If yes, the opposite edge on the fillet can be used to find the hole face. Fig. 10 depicts the method to search two neighboring faces (f_1 & f_2) separated by a fillet (f_f), where e_1 and e_2 are two edges of f_f adjacent to f_1 and f_2, respectively. Using the neighboring relationship of the edges on the fillet, it is easy to find f_2 starting from f_1.

(3) Obtain body faces: Body faces are found from the faces neighboring the outer loop of the top face. They should be cylindrical or conic shape. If a fillet exists between a top face and a body face, the body face can be detected indirectly using the neighboring relationship of the edges on the fillet, as mentioned previously in (2). The green faces in Fig. 1 are the top (A1) and the body (A2) faces. Subsequently, the

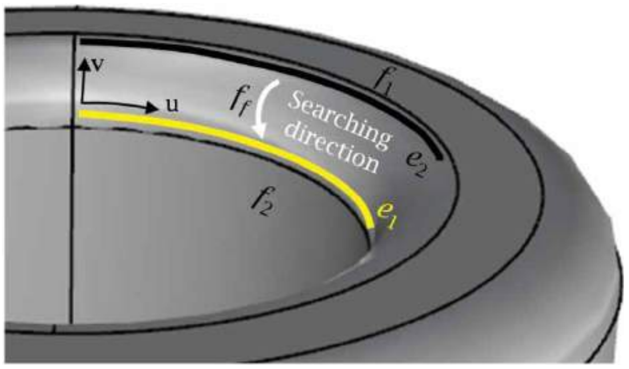


Figure 10. The search method for two faces separated by a fillet.

radius and center point of the tube are recorded. The center points of the hole and the tube should be inspected to ensure that their distance is smaller than the desired tolerance.

(4) Obtain temporary and end faces: The algorithm searches for faces neighboring the body faces and classifies them into vertical and horizontal connecting faces. If the vertical connecting faces are planar and connected to a concave edge, they are recorded as the end faces (B1 in Fig. 1). If the horizontal connecting faces are connected to a concave edge, they are recorded as the temporary faces. If fillets exist, the method used in (2) can be employed again to find the vertical or horizontal connecting faces. Each of the connecting faces can then be judged whether it belongs to a temporary face or an end face.

In addition, if a tube is connected to a convex edge, it represents a floating tube. In this case, it does not have any base face, and is thus not considered a tube. Subsequently, all recorded data is deleted, and the process moves to the next loop.

(5) Obtain shell and base faces: Once the end faces are determined, the temporary faces can be classified into two types: shell faces and base faces. If a temporary face connects to a neighboring end face with a concave edge, it is recorded as a base face (A3 in Fig. 1). If a temporary face connects to a neighboring end face with a convex edge, it is recorded as a shell face (B2 in Fig. 1). The shell face can change according to the height of the ribs. If the height of a rib is less than that of the tube, the shell face attached to the tube will be an independent face, as shown in Fig. 1(a). By contrast, if the height of a rib is equal to that of the tube, the shell face attached to the tube will merge with the top face into one single face, as shown in Fig. 1(b). When a fillet exists between an end face and a temporary face, the neighboring relationship of edges and faces on the fillet can be employed to detect the temporary face that neighbors to the same fillet as the end face.

In rib recognition, a set of end faces (f_{e1}, f_{e2} ...), shell faces (f_{s1}, f_{s2} ...), and fillets are input. One of the end faces, namely f_{e1}, is set as f_{end1}, and the system searches the shell face (f_{shell}) through the connecting convex edges. Using this edge, we can find the corresponding edges on f_{shell} and the corresponding end faces f_{end2}, where f_{end2} are composed of $f_{end11}, f_{end12}, f_{end13}$, etc. Ideally, f_{end1} and f_{end2} should be perpendicular to the common face f_{shell}, which can yield a rectangular cross section for the rib. However, a slight deviation in perpendicularity is allowed to yield a trapezoidal cross section for the rib. Consequently, one set of ribs will be recorded: $f_{end1}, f_{end2}, f_{shell}$, the fillets of the rib, and the relationship of this rib connecting with the tube. The limitation of the proposed algorithm is that f_{end} and f_{shell} should be planar.

With hole recognition, loops are the starting element. Based on the fillet and edge AAG information, each edge on loops can obtain the convexity property. Following the convex edge, the side faces of a hole can be found. However, another edge of this hole can be obtained by continuing the search downwards. If a convex edge is found, this hole is a through hole. By contrast, if it is a concave edge, this hole is a simple blind hole, where the blind hole lies on a single face. If a blind hole lies on multiple faces, the convex and concave conditions may become more complex. If it is necessary, additional algorithm must be developed to distinguish and separate each of the cases. Finally, one set of holes is also recorded, as well as the data for side faces, bottom faces, the hole fillets, and the relationship of the hole located at the tube. As mentioned above, the data corresponding to the rib and hole can be used in the boss data structure.

6. Output boss data for meshing

The data recorded for a boss includes three data sets: the rib data, tube data, and hole data, as listed in Tab. 3. The attributes in each data set can be divided into two types: feature data and meshing data, where the feature data records the faces on each of the constructing features, and the meshing data records additional information for meshing. For the rib, end faces (R1) and shell faces (R4) belong to the feature data, whereas edge pairs (R2), vertex pairs (R3), shell-shell edges (R5), and tube-rib edges (R6) belong to the meshing data. For the tube, hole (T1) and outer face (T2) belong to the feature data, whereas the edges (T3), neighboring faces (T4), face attributes (T5), and tube thickness (T6) belong to the feature data. For the hole, the geometry (H1), side face (H2), and bottom

Table 3. Data recorded for the rib, tube, and hole on a boss.

Code	Attribute		Remark
	Rib data		
R1	End face	End faces	Two end faces
R2		Edge pairs	Pairs of edges on two end faces
R3		Vertex pairs	Pairs of vertices on two end faces
R4	Shell face	Shell faces	Shell faces
R5		Shell-shell edges	Common edges of shell faces
R6		Tube-rib edges	Edges connecting the tube and the rib
	Tube data		
Code	Attribute		Remark
T1	Hole		Index of the hole
T2	Outer face	Outer face	Index of the outer face
T3		Edges	All edges on the outer faces
T4		Neighboring faces	All neighboring faces of the edges
T5		Face attributes	Attributes of all neighboring faces
T6	Tube thickness		Thickness of the tube
	Hole data		
Code	Attribute		Remark
H1	Geometry		Circular or noncircular hole
H2	Side face		Index of the side face
H3	Bottom face		Index of the bottom face
H4	Type of bottom face		Flat, sphere, bevel or through hole
H5	Depth		Depth of the hole

base (H3) belong to the feature data, whereas the type of bottom face (H4) and the depth (H5) belong to meshing data. The attribute definitions for the rib, tube, and hole are shown in Fig. 11. The definition of the abovementioned data structure for a boss enables the development of an automatic meshing algorithm.

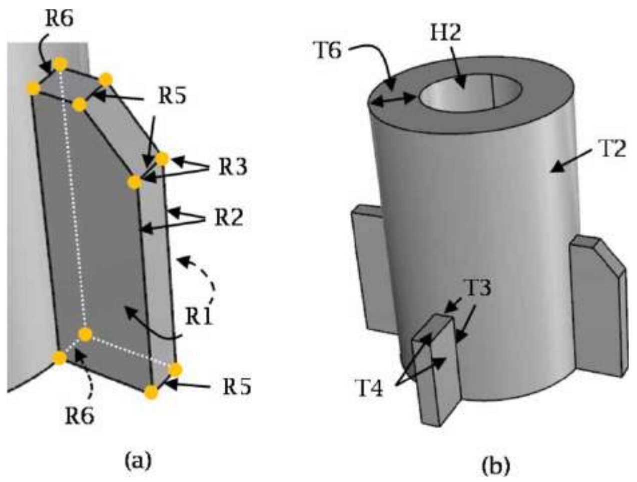


Figure 11. Attributes used in rib data, tube data, and hole data: (a) Rib, (b) Tube and hole.

For meshing, a boss can primarily be decomposed into two parts: the ribs and tube. If fillets exist on the surface boundaries, the meshing process is more difficult, and the meshing data is more complex. Therefore, we only discuss the meshing algorithm for bosses with all fillets suppressed. The meshing algorithm with all fillets included will be discussed elsewhere. For rib meshing, as depicted in Fig. 12, node seeding on one end face is performed first. If the end face is rectangular, it is meshed using rectangular elements. By contrast, if the end face is non-rectangular, it is meshed using triangular elements. A mapping process can finally be implemented along the second end face to generate the solid mesh for the rib.

For tube meshing, as depicted in Fig. 13, the outer faces of the tube are obtained first. It may be one or more faces. The surface meshes from the ribs and all their boundary edges are used to check the regularity of the surface meshes. If all surface meshes from the ribs intersect the tube orthogonally, the outer faces of the tube are meshed using rectangular elements. Otherwise, the outer faces of the tube are meshed using triangular elements. By combining the surface meshes from the ribs and the meshes from the outer faces of the tube, a mapping process can be implemented to generate solid meshes for the tube. If the ribs intersect the tube orthogonally, the tube can be meshed using all hexahedral meshes; while for ribs that do not intersect the tube orthogonally, the tube is meshed using both hexahedral and prism meshes. The meshing strategies shown in Figs 12 and 13 are only used to demonstrate the concept of recording specific data on a rib and a tube, respectively, to aid the generation of better type of meshes. When the shape of the hole, tube or ribs is changed, the meshing strategy should be modified also. In addition, when fillets appear on the top or bottom of the tube, additional meshing strategy should be proposed for the region near fillets. For example, different mesh templates may be designed for the meshing of different types of fillets. A thoughtful study on different meshing strategies for the bosses shown in Fig. 2 and the corresponding data that should be recorded will be provided elsewhere.

7. Results and discussion

The boss recognition algorithm proposed herein was employed to test more than 30 models [8], and all bosses were recognized successfully. A program based on the proposed algorithm was written in C++ and was based on the Rhino CAD platform [26] and the open-NURBS function [25]. Fig. 14 illustrates examples of the boss recognition results obtained using the proposed algorithm. Tubes, ribs, and holes are shown in green, blue, and yellow, respectively. Shape complexity ranged from a simple individual boss to several interconnected bosses. Examples of bosses recognized successfully in this study include those located on several surfaces or non-regular faces, and with many virtual faces. Virtual face is ribs and tube faces sharing the same face. There is no restriction

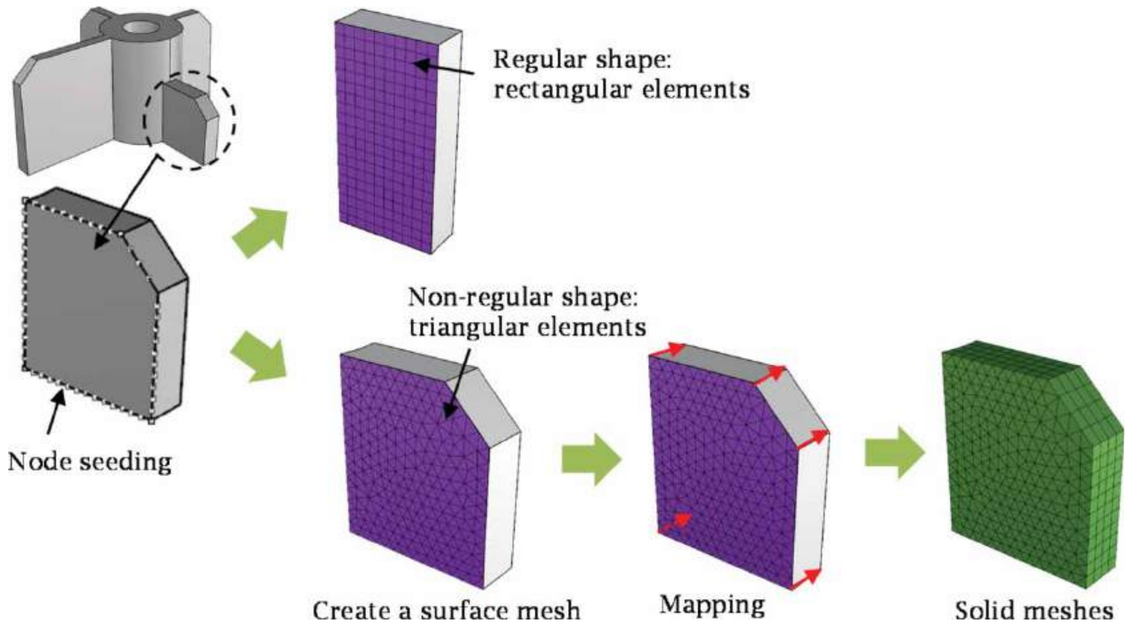


Figure 12. Generation of solid meshes for the rib.

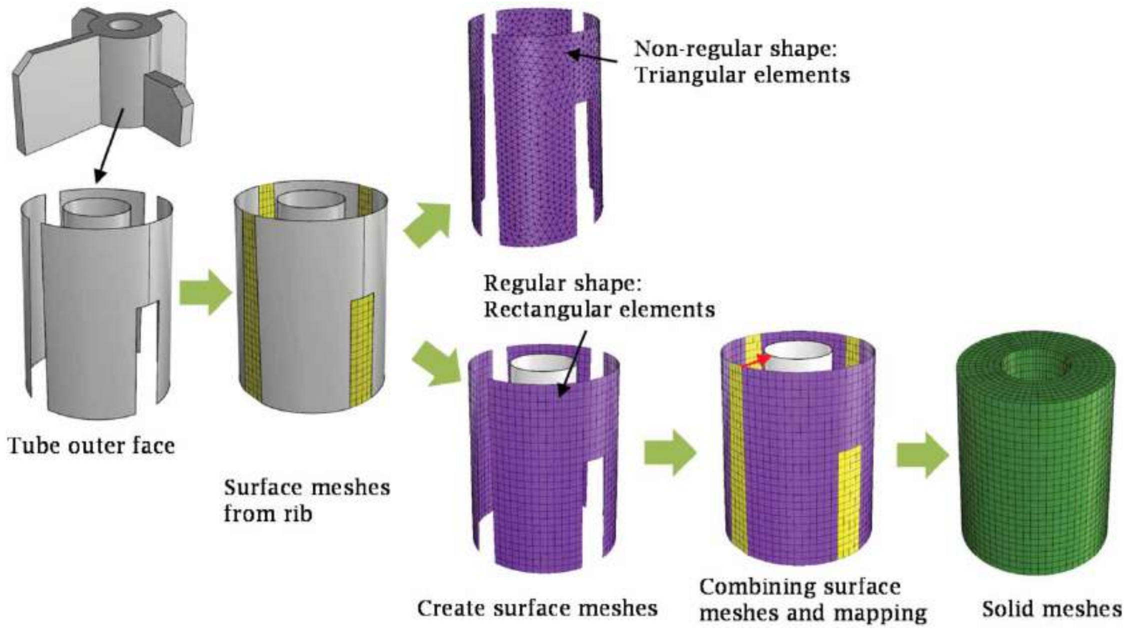


Figure 13. Generation of solid meshes for the tube.

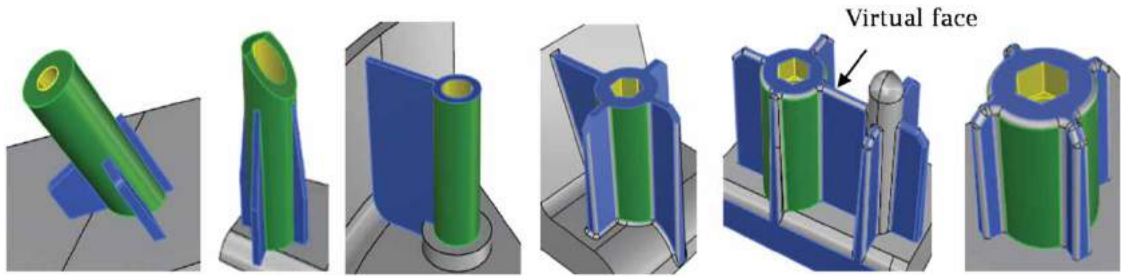


Figure 14. Results of the proposed boss recognition algorithm for several types of bosses.

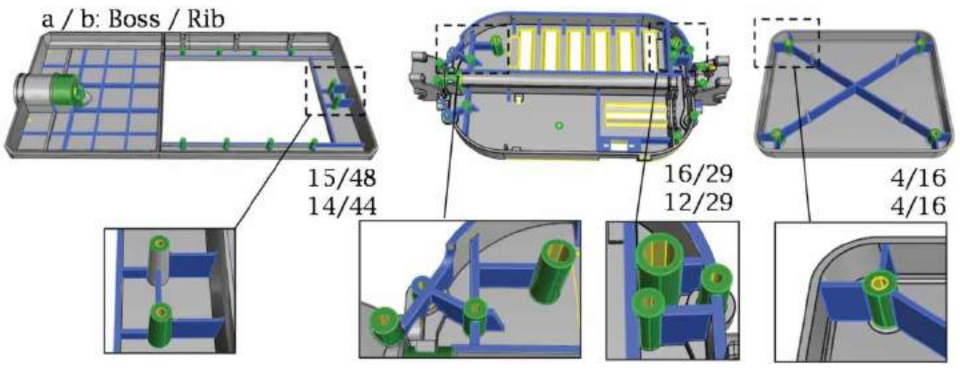


Figure 15. Boss recognition results for several CAD models, where the first and second rows of values are obtained by the proposed method and CADdoctor^{TM}, respectively.

on the height and thickness of the tube, but the hole should be kept on the center of the tube.

In addition, Fig. 15 shows recognition results for three real examples that combined rib, hole, and boss recognition. Some of the bosses are rounded at the edges either on the top or bottom of the holes, which results in fillets, as shown on the expanded plots. The proposed method can handle the existing of fillets. We compared all of the results to those obtained using CADdoctor^{TM} [4] under the same conditions, including the radius, width, and height. The upper values in each plot indicate our result, whereas the lower values indicate the results obtained using CADdoctor^{TM}. The first and second values in each row indicate the number of bosses and ribs recognized, respectively. As indicated by the results, the boss definitions in our study and in CADdoctor^{TM} are not completely identical. For example, the rib height can be the same as the tube height in the proposed algorithm, whereas this is not the case in CADdoctor^{TM}. This indicates that our algorithm recognizes more bosses than CADdoctor^{TM}. However, it still has limitations. For

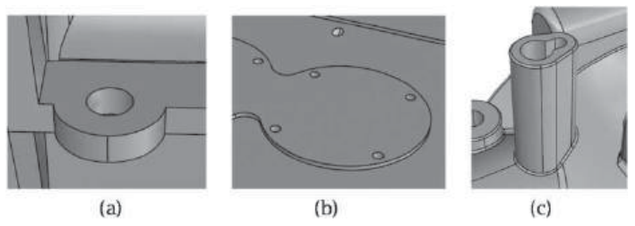


Figure 16. Three typical cases that cannot be processed using the proposed algorithm, (a) without a base face, (b) hole not on the center, (c) not circular or polygonal hole.

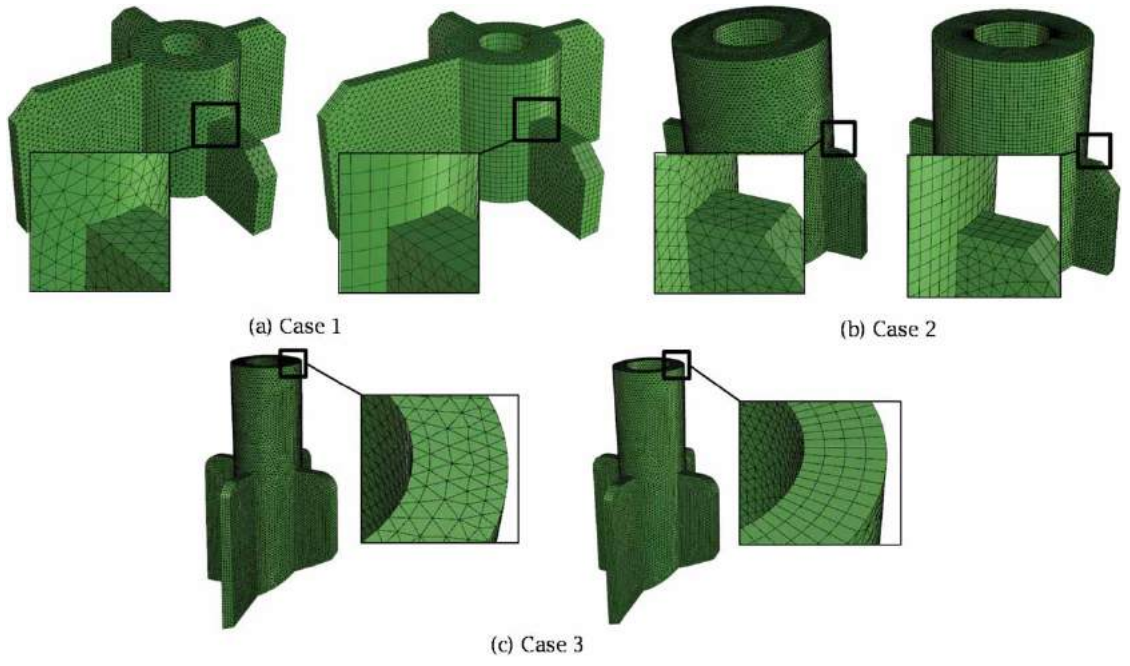


Figure 17. Comparison of conventional tetrahedral mesh (left) and hybrid mesh (right) obtained by the proposed method: (a) Case 1, (b) Case 2, (c) Case 3.

Table 4. Quality table of three boss cases.

			Aspect ratio[2]		Orthogonality[3]		Skewness[4]	
Case	Mesh element type	Number of meshes	Average	Min.	Average	Max.	Average	Min.
1	Full tetrahedral meshes	366193	0.82	0.38	17.68	56.31	0.89	0.64
	Hybrid meshes[1]	80004	1.00	0.81	2.76	19.74	0.99	0.83
2	Full tetrahedral meshes	79654	0.82	0.40	17.41	55.28	0.89	0.63
	Hybrid meshes	24328	0.99	0.73	1.58	16.83	0.99	0.81
3	Full tetrahedral meshes	269333	0.82	0.34	17.37	53.77	0.90	0.61
	Hybrid meshes	76784	0.97	0.34	9.87	53.69	0.99	0.77

1: include hexahedral and prism meshes, obtained by the proposed method
2: range: 0–1, 1 is the best
3: range: 0–180, 0 is the best
4: range: -∞–1, 1 is the best

instance, the following cases cannot be recognized by the proposed algorithm: (1) when a boss has no base face (Fig 16(a)), (2) when a hole is not located at the center of the tube (Fig 16(b)), or (3) when the cross-sectional shape of a hole is not circular or regular polygonal (Fig 16(c)).

Regarding computational efficiency, the face number in the CAD model is the main factor affecting the computational time of the proposed algorithm, but the average computing time for each process is only a few seconds. Fig. 17 illustrates the meshing result of three cases (Case 1, Case 2, and Case 3) involving the removal of all small fillets. In Fig. 17(a), the left plot represents conventional tetrahedral mesh and the right plot represents the hybrid mesh, including hexahedral and prism meshes, obtained by the proposed algorithm. In this case, two of the ribs are of the same height as that of the tube. For these two solid meshes, the same node seeding size was used and four layers were employed in each rib. Fig. 17(b) shows the same comparison for Case 2, where all ribs are shorter than the tube. Fig. 17(c) shows the same comparison for Case 3, where the bottom face of the tube is not parallel to its top face, which results in a case of non-regular shape. The left plot represents conventional tetrahedral mesh on the tube and the right plot represents the prism mesh on the tube. Tab. 4 shows a comparison of the quality of each pair of solid meshes for three cases, comprising the conventional tetrahedral mesh and the hybrid mesh obtained by the proposed algorithm. For example, consider Case 1; the average orthogonality with tetrahedral elements is 17.68, and the maximum orthogonality is 56.31. However, with hybrid elements, the average orthogonality is 2.76, and the maximum orthogonality is 19.74, which represents a dramatic decrease. Other indices, such as the mesh number, aspect ratio, and skewness, improved considerably as well. Both the minimum aspect ratio and minimum skewness are also improved significantly. As mentioned, satisfactory solid meshes should satisfy the following criteria: contain fewer meshes, have an aspect ratio and skewness of 1, and have an orthogonality of 0.

8. Conclusion

This study focused on the development of a boss recognition algorithm for small and thin features, and verified the feasibility of the proposed algorithm by testing several CAD models. A procedure for generating hexahedral and prism meshes for bosses was also presented. The main contribution of our method is that we propose an approach based on feature recognition for generating better quality solid meshes for FEA applications. An algorithm for the recognition of bosses was proposed. The output data of a boss for meshing was developed. The process for automatic meshing of bosses was also presented. The proposed method can not only be used for bosses but can also be expanded to other simpler features. This method can reduce the necessity of manual operations, hence decreasing the overall operational time for meshing. Furthermore, the meshing results indicate that all quality indices of the meshes generated using the proposed method improved considerably. Notably, the proposed meshing algorithm was feasible only for recognized features, and small fillets on a CAD model should first be removed. The meshing algorithm presented in this paper is only a simple concept to demonstrate how the data of the constructing elements are generated in order to help the generation of meshes. This simple concept is indeed not suitable for more complex situations on realistic CAD models. For example, fillets may exist on the top or bottom face of the tube, ribs are not aligned with the tube regularly, the shape of ribs may change, etc. In future studies, developing additional feature recognition and feature decomposition algorithms for converting most CAD model features into hexahedral meshes is imperative.

ORCID

Ming-Hsuan Wang http://orcid.org/0000-0003-4947-4218
Jiing-Yih Lai http://orcid.org/0000-0002-0495-0826
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] Abu, R.; Tap, M. Md.: Attribute based feature recognition for machining Features, Jurnal Teknologi, 46, 2007, 87–103. http://dx.doi.org/10.11113/jt.v46.284

[2] Ansaldi, S.; Floriani, L. D.; Falcidieno, B.: Geometric modeling of solid objects by using a face adjacency graph representation, ACM SIGGRAPH Computer Graphics, 19(3), 1985, 131–139. http://dx.doi.org/10.1145/325165.325218

[3] Benzley, S. E.; Perry, E.; Merkley, K.; Clark, B.; Sjaardema, G.: A comparison of all hexagonal and all tetrahedral finite element meshes for elastic and elasto-plastic analysis, In Proceedings, 4th International Meshing Roundtable, 1995, 179-191.

[4] CADdoctor, http://elysiuminc.com/products/caddoctor/

[5] 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

[6] Cuillière, J. C.; Maranzana, R.: Automatic and a priori refinement of three-dimensional meshes based on feature recognition techniques, Advances in Engineering Software, 30(8), 1999, 563–573. http://dx.doi.org/10.1016/S0965-9978(99)00013-7.

[7] Flavien, B.; Jean-Claude, L.; Stefanie, H.; Lionel, F.: Extraction of generative processes from B-Rep shapes and application to idealization transformations, Computer-Aided Design, 46, 2014, 79–89. http://dx.doi.org/10.1016/j.cad.2013.08.020

[8] GrabCAD, https://grabcad.com

[9] Hariya, M.; Nonaka, N.; Shimizu, Y.; Konishi, K.; Iwasaka, T.: Technique for checking design rules for three-dimensional CAD data, 2010 3rd IEEE International Conference on Computer Science and Information Technology, 1, 2010, 296–300. http://dx.doi.org/10.1109/ICCSIT.2010.5565010

[10] Ismail, N.; Abu Bakar, N.; Juri, A. H.: Recognition of cylindrical and conical features using edge boundary classification, (2005) International Journal of Machine Tools and Manufacture, 45(6), 2005, 649–655. http://dx.doi.org/10.1016/j.ijmachtools.2004.10.008

[11] 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

[12] 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

[13] Jun, Y.; Raja, V.; Park, S.: Geometric feature recognition for reverse engineering using neural networks, International Journal of Advanced Manufacturing Technology, 17(6), 2001, 462–470. http://dx.doi.org/10.1007/s001700170164

[14] Lai, J.-Y.; Wang, M.-H.; Chiu, Y.-K.; Hsu, C.-H.; Tsai, Y.-C.; Huang, C.-Y.: Recognition of depression and protrusion features on B-rep models based on virtual loops, 13(1), 2015, 95–107. http://dx.doi.org/10.1080/16864360.2015.1059200

[15] Lai, J.-Y.; You, Z.-W.; Chiu, Y.-K.; Wang, M.-H.; Hsu, C.-H.; Tsai, Y.-C.; Huang, C.-Y.: On the development of blend faces and holes recognition algorithm for CAE applications, Key Engineering Materials, 656-657, 2015, 789–794. http://dx.doi.org/10.4028/www.scientific.net/KEM.656-657.789

[16] Li, B.; Liu, J.: Detail feature recognition and decomposition in solid model, Computer-Aided Design, 34(5), 2002, 405–414. http://dx.doi.org/10.1016/S0010-4485(01)00118-X

[17] Li, J.; Sun, L.; Peng, J.; Du, J.; Fan, L.: Automatic small depression feature recognition from solid B-rep models for meshing, 2011 International Conference on Electrical and Control Engineering (ICECE), 2011, 4386–4389. http://dx.doi.org/10.1109/ICECENG.2011.6057432

[18] Li, Y.; Wang, W.; Liu, X.; Ma, Y.: Definition and recognition of rib features in aircraft structural part, International Journal of Computer Integrated Manufacturing 27(1), 2014, 1–19. http://dx.doi.org/10.1080/0951192X.2013.799784

[19] Li, Y.-G.; Ding, Y.-F.; Mou, W.-P.; Guo, H.: Feature recognition technology for aircraft structural parts based on a holistic attribute adjacency graph, Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 224(2), 2010, 271–278. http://dx.doi.org/10.1243/09544054JEM1634

[20] 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, (2005) IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(6), 2005, 851–860. http://dx.doi.org/10.1109/TPAMI.2005.118

[21] 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

[22] Lu, Y.; Li, Y.-G.: A feature recognization technology of complex structural parts based on re-extended attributed adjacency graph, Machinery Design & Manufacture, 219(5), 2009, 240–242.

[23] Marefat, M.; Kashyap, R.: Geometric reasoning for recognition of three-dimensional object features, Pattern Analysis and Machine Intelligence, IEEE, 12(10), 1990, 949–965. http://dx.doi.org/10.1109/34.58868

[24] Moldex3D, http://www.moldex3d.com/en/

[25] openNURBS, http://www.rhino3d.com/tw/opennurbs

[26] Rhinoceros, http://www.rhino3d.com

[27] Shah, J. J.; Anderson, D.; Kim, Y. S.; Joshi, S.: A discourse on geometric feature recognition from CAD models, Journal of Computing and Information Science in Engineering, 1(1), 2001, 41–51. http://dx.doi.org/10.1115/1.1345522

[28] Venkataraman, S.; Sohoni, M.: Blend recognition algorithm and applications, Proceedings of the sixth ACM Symposium on Solid Modeling and Applications, 2001, 99–108. http://dx.doi.org/10.1145/376957.376970

[29] Zhang, C.-J.; Zhou, X.-H.; Li, C.-X.: Automatic recognition of intersecting features of freeform sheet metal parts, Journal of Zhejiang University-Science A, 10(10), 2009, 1439–1449. http://dx.doi.org/10.1631/jzus.A0820705