Engineering with Computers (2014) 30:237–252
DOI 10.1007/s00366-013-0332-z

ORIGINAL ARTICLE

Geometric reasoning in sketch-based volumetric decomposition framework for hexahedral meshing

Jean Hsiang-Chun Lu · Inho Song · William Roshan Quadros · Kenji Shimada

Received: 12 February 2013 / Accepted: 4 September 2013 / Published online: 1 October 2013

Abstract This paper presents a sketch-based volumetric decomposition framework using geometric reasoning to assist in hex meshing. The sketch-based user interface makes the framework user-friendly and intuitive, and the geometric reasoning engine makes the framework smarter and improves the usability. The system first generates a database that contains both the B-rep and 3D medial object to capture the exterior and interior of the input model, respectively. Next, the geometric reasoning process determines sweeping direction and two types of sweepable regions and provides visual aids to assist the user in developing decomposition solutions. The user conducts decomposition via the sketch-based user interface, which understands the user's intent through freehand stroke inputs for smart decomposition. Imprint and merge operations are then performed on the decomposed model before passing it to the sweeping algorithm to create hex meshes. The proposed framework has been tested on industrial models.

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

J. H.-C. Lu (✉) · I. Song · K. Shimada
Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA, USA
e-mail: hsiangcl@andrew.cmu.edu

I. Song
e-mail: songphd@andrew.cmu.edu

K. Shimada
e-mail: shimada@andrew.cmu.edu

W. R. Quadros
Sandia National Laboratories, Albuquerque, NM 87185, USA
e-mail: wrquadr@sandia.gov

Keywords 3D medial object · Geometric reasoning · Hexahedral meshing · Sketch-based decomposition

1 Introduction

Hex mesh offers benefits over tetrahedral mesh [28], and usually preferred because it often yields more accurate solution [1, 4, 36]. Due to the geometric constraints of hex mesh elements, volumetric decomposition becomes critical to generate hex-meshable sub-domains. One survey [7] reported that the geometry decomposition task takes 31 % of the total time in the modeling and as compared to the simulation process takes only 4 % running the simulation. The statistic shows that one of the main bottlenecks of meshing comes from the decomposition task. Although many efforts have been made over the years towards fully automatic volumetric decomposition [6, 24, 25, 32], existing methods can only handle a limited class of shapes or end up generating poor quality hex elements. Semi-automatic methods [17] suggest cutting surfaces and allow user interaction to determine the desired one for decomposition. However, most of the suggested cutting surfaces fail to produce hex meshable sub-domains. The remaining non-meshable portions have to be manually decomposed. The user expertise on mesh generation is heavily required, and a great amount of user interaction has to be conducted. Unfortunately, the existing CAE packages do not provide suitable user interfaces (UIs) for manual decomposition operations. This makes the whole process tedious and time-consuming, especially to novice users.

The main bottleneck of manual decomposition comes from the existing user interface. It requires a series of complicated actions such as selecting menus or icons or entering parameters to define accurate cutting surfaces. The

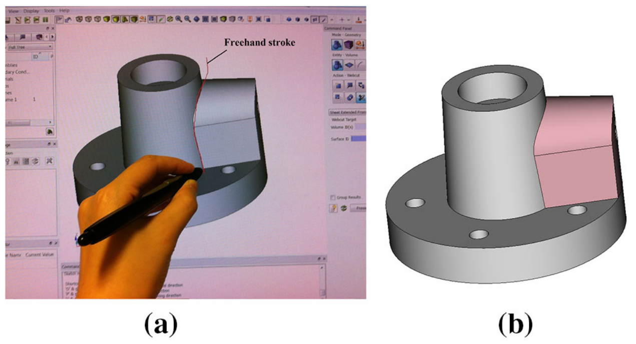


Fig. 1 The sketch-based decomposition. **a** A freehand stroke is created in the sketch-based UI. **b** The stroke decomposed the model into two sub-domains

user has to be familiar with a CAE package to conduct decomposition operations. To determine sweepable regions and ideal cutting positions also require special user expertise on volumetric decomposition. A sketch-based UI has been presented by Lu et al. [12] to support the manual decomposition process. Strokes are accepted as inputs to create cutting surface, which saves time spent on entering geometric information to define cutting surfaces. However, this approach does not provide the users with decomposition suggestions and relies on domain knowledge on decomposition and mesh generation.

This paper presents a sketch-based framework with a geometric reasoning engine to improve the usability and efficiency of volumetric decomposition to assist in hex-meshing. The system first generates a database that contains both the B-Rep and 3D medial object (MO) to capture the exterior and interior of the input model, respectively. Then, the geometric reasoning engine uses the database to detect the sweepable regions and sweeping directions from the given model and displays them to assist the users. The user follows the visual aids to conduct sketch-based operations (e.g., draw strokes to create cutting surfaces as shown in Fig. 1) through the sketch-based UI. The geometric reasoning engine infers the user's intents to create desired cutting surfaces and executes appropriate decomposition commands. Imprint and merge operations are then performed on the decomposed model before passing it to the sweeping algorithm to create hex meshes.

2 Related work

2.1 Volumetric decomposition for hex meshing

Much research has been done in developing decomposition-based approach for hex meshing. White et al. [35] proposed a virtual decomposition method to decompose the volume into mappable sub-domains. This method is based on the corner angles of the objects, and is only suitable for blocky regions or objects with well defined corners. Shin et al. [29, 30] describe swept volume decomposition method that works for geometries that can be decomposed into linearly swept volumes. Sheffer et al. [27] use embedded Voronoi graph for decomposing the object which results in sweepable sub-volumes. These automatic decomposition methods only work for specific types of geometries. The decomposition of a general solid is not always possible, and there are usually unmeshable sub-volumes remaining. Lu et al. [13] described a semi-automatic approach that uses local geometric information and feature recognition algorithms to create a list of cutting surface, and allows the users to select the desired one to subdivide meshable regions [17]. However, this approach does not always create reasonable cutting surfaces and leaves the remaining non-meshable pieces that needs further manual operations.

2.1.1 *Medial-based decomposition*

Medial axis transform (MAT) introduced by Blum [2] has been used to decompose the complex domain into simpler sub-domains. Tam et al. [31] use 2D medial axis to subdivide complex objects into simpler 2D regions, and uses sets of template to insert quadrilaterals into the sub-domains. In 3D domain, Quadros et al. [21] introduced an algorithm that couples medial axis decomposition with an advancing front method. In general, these methods produce high quality meshes but they are not robust and may require a great deal of user interaction, especially if the domain has non-manifold boundaries. Donaghy et al. [5] reduced model dimension for further analysis using MAT to classify several topology features such as end region of beams, concave and convex corners. Sampl [22] first meshed the medial faces, and then extruded the mesh on both sides of the medial until the mesh intersects the boundary of the object. Chong et al. [3] used medial surface to reduce the complex original model to recognize features by identifying edge types, and by using classified medial information to treat different geometry combinations. Price et al. [19, 20] described a medial surface subdivision technique to decompose objects into a collection of meshable primitives which can be meshed directly using a midpoint sub-division technique proposed by Li et al. [11]. This approach is less than reliable for general geometries and may create poor quality meshes for shapes that have degenerate medial axes. Makem et al. [15] applied shape metrics to automatically identify long, thin regions within a thick body, and partition the thick body into non-manifold slender sub-domains. Luo et al. [14] used medial surface points to identify thin regions within the volume, and subdivide the volume for mesh generation.

These approaches are only available for specific configuration and the remaining portions are not hex-meshable and will be filled with tetrahedral elements.

2.2 Sketch-based decomposition

Sketch-based or pen-based approaches have been researched and applied in the computer aided design (CAD) field. The key concept is to create 3D shapes using strokes extracted from user's freehand input or existing drawings. These approaches improve the methods of inputting data and creating freeform surfaces in the CAD software or similar modeling systems. Igarashi et al. [8] proposed a sketch-based system to create freeform 3D objects defined by closed strokes. Extrusion can be made on the objects. Varley et al. [34] converts a 2D sketch to B-Rep solid model instead of accepting direct stroke input from the user. Masry et al. [16] proposed optimization-based reconstruction algorithms to reconstruct sketches in a 3D sketching system for analysis. Kara et al. [10] presented a template-based approach for industrial design, which allows the user starting from modifying the templates to create 3D shapes.

Lu et al. [12] used a sketch-based UI to assist in geometric decomposition for hex meshing. The system takes user's freehand stroke as inputs to create accurate and well-aligned cutting surfaces. The system beautifies user's freehand strokes and determines the proper alignments for the stroke to existing boundaries. The stroke is then snapped and extruded to create the cutting surface. The automatic alignment of the cutting surface prevents the bad angles between the boundaries to ensure the mesh quality. This approach provides an intuitive way to conduce decomposition, and speeds up the cumbersome decomposition process by freeing the users from having to input details in traditional GUI or command line. However, its stroke snapping evaluation process does not consider sweepable regions from the model; therefore, the user expertise and domain knowledge are still required to recognize ideal cutting regions that subdivide hex-meshable sub-domains.

3 Framework overview

The framework shown in Fig. 2 is designed as an adjunct to automatic decomposition methods. It provides an intuitive way to operate on parts that an automatic method cannot handle. Three portions form the framework: (1) geometric reasoning database, (2) geometric reasoning engine, and (3) sketch-based UI/operations.

After importing the model, a database (Sect. 3.3) is created for geometric reasoning. The geometric reasoning engine (Sect. 4) recognizes sweepable regions from the given model and provides the visual aids to assist the user

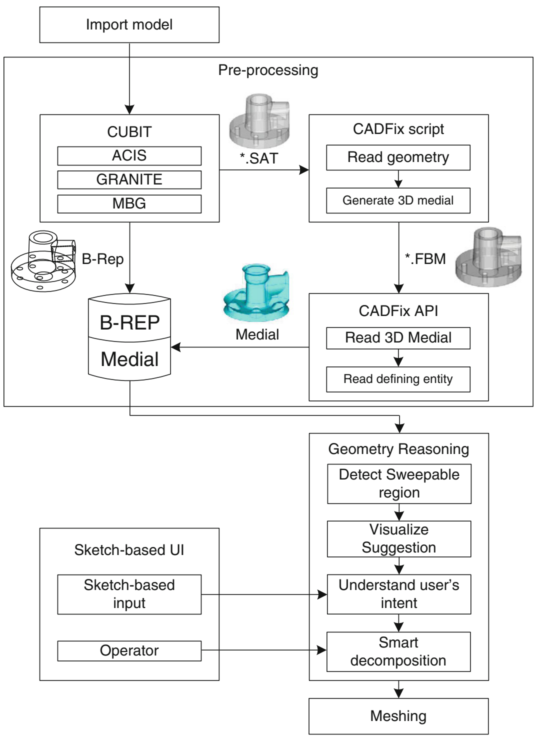


Fig. 2 The framework of the sketch-based decomposition with geometric reasoning

develop decomposition solutions. The sketch-based user interface (UI) [12] in the front end takes care of the user interaction and accepts sketch-based inputs. After accepting the inputs, the geometric reasoning engines infers user's intents and conducts decomposition operations. Finally, appropriate meshing algorithms are assigned to the sub-volumes to complete hex-meshing.

3.1 Sketch-based user interface and operation

The goal of the sketch-based approach is to make the decomposition process more user-friendly and intuitive. The key of sketch-based decomposition is to infer user's intent from the rough stroke inputs for accurate decomposition. It saves the user time from entering detail geometry information to define cutting surfaces or specify decomposition operations via graphic user interface (GUI) or command line using the existing CAD/CAE packages.

The sketch-based approach allows users to make inputs with stroke using stylus or other pointing devices. The system detects the stroke type, the location of the stroke to beautify, and aligns the stroke to existing entities; and the user extrudes the stroke to create an accurate cutting surface [12]. The advantage is that the sketch-based approach is more intuitive and user-friendly to create cutting surface, the user does not have to type in syntax or look up for certain icons from the manual. However, since the sketch-based approach does not recognize the sweepable region on the model and it is not able to provide decomposition suggestions to the user, domain knowledge on decomposition for hex-meshing is required.

3.2 Geometric reasoning engine

The goal of geometric reasoning is to figure out the desired cutting position, shape of the cutting surface, and the proper decomposition operation from user's rough inputs. The geometric reasoning engine brings the intelligence to the framework by detecting the sweepable regions and generating decomposition suggestions. The sweepable regions are highlighted as visual aids, and the user uses the sketch-based methods to conduct volumetric decomposition easily. The geometric reasoning engine also understands user's intents from rough sketch-based inputs, and returns a smart decomposition result. The proposed geometric reasoning engine has four steps: Step 1: detect sweepable region, Step 2: visualize decomposition suggestions, Step 3: understand user's intent, and Step 4: smart decomposition.

The user follows the visual aids to create cutting surfaces via the sketch-based UI. If the input is a freehand stroke, the geometric reasoning process searches for the best snapping candidate among the entities in the database for the input stroke. In the sketch-based UI, two types of inputs are accepted: freehand strokes, and picked entities. The freehand stroke is used to understand user's intent and then extruded to create a cutting surface. The stroke tells the desired shape of the cutting surface and the cutting position. If the input is a series of picked entities, the reasoning process evaluates the possible operation for decomposition using those picked entities. The picked entities can be used to infer how the user wants to create cutting surfaces/decompose the model (e.g., sweep entity 1 along entity 2).

3.3 Database

The geometric reasoning engine uses a database that contains B-Rep and 3D MO to detect sweepable regions and suggest potential decomposition. The 3D MO is the 3D equivalent of medial axis (MA), which is defined by the locus of the center of maximal inscribed sphere as it travels around the interior of the shape [2]. The MO is a skeleton representation of a 3D shape that contains rich geometric information and is of reduced dimension. The database also contains a map of the one-to-one correspondence between 3D MO and the given model.

The 3D MO used in the framework is generated by CADFix [9]. Since all the operations are conducted in the mesher CUBIT [23], a script is used to call CADFix to generate the 3D medial. Then, an application programming interface (API) is used to obtain the 3D MO.

3.3.1 Data structure

In order to generate a map between B-Rep and MO, a data structure is designed as shown in Fig. 3. The structure has an "entity" class that contains ID, bound box, and centre points as class member. The entity class has "medial group", "medial face", and "medial curve" as child classes.

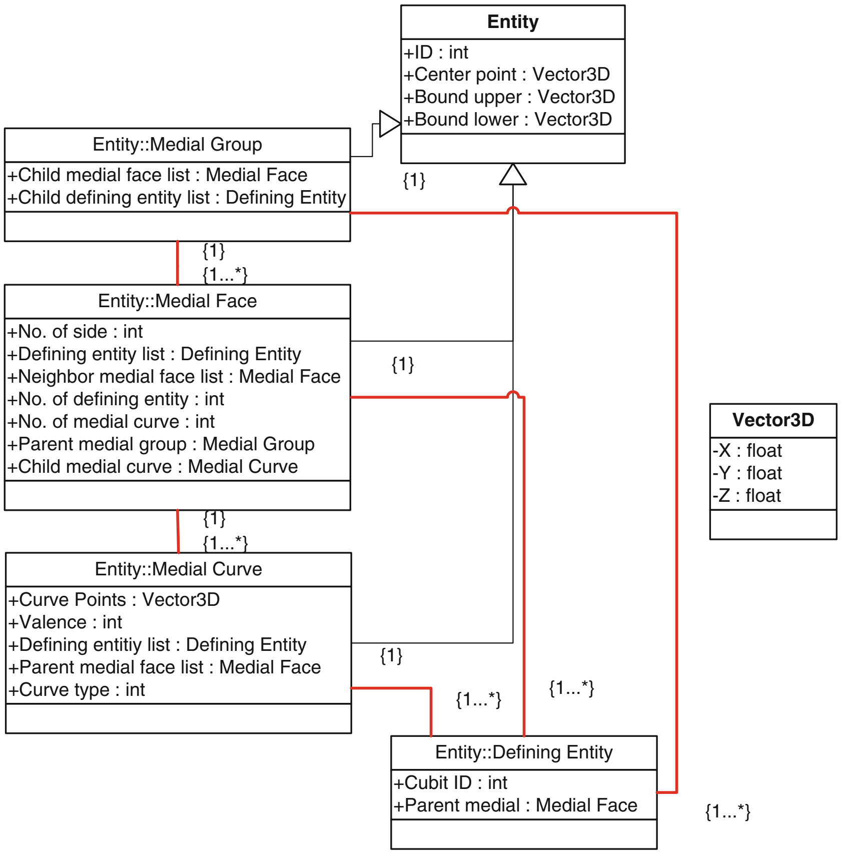
A medial group stores medial faces in the same patch, and their corresponding defining entity group. Given a medial face, its parent medial group and child medial curve can be retrieved. Given a medial curve, its defining entity list and parent the medial face list can be retrieved. Each defining entity is assigned a unique ID in CUBIT. With the unique CUBIT ID, the defining entities can be mapped to the B-Rep. CUBIT uses the common geometry module (CGM) [33] to represent B-Rep solid model [26]. This way, given a medial entity, the B-Rep that defines the medial entity can be retrieved.

3.4 Terminologies

The following section lists the terminologies used in this paper.

- Medial object: a set of entities defined as the locus of the centre of the maximal ball as it rolls inside the given model (Fig. 4b).
- Medial curve: a curve that connects two medial vertices (Fig. 4b).
- Medial face: a surface bounded by medial curves (Fig. 4b).
- Flap: a medial face that touches the boundary of the original model (Fig. 4b).
- Trimmed MO: MO without flaps (Fig. 4c).
- MO group: a set of connected medial faces that is the skeleton representation of a sweepable region. A MO group can be classified as a 2-manifold or a non-manifold group and depends on how the member medial faces connected to other faces.
- Non-manifold MO group: a set of connected non-manifold medial faces that is the skeleton representation of a sweepable region.
- 2-manifold MO group: a set of connected 2-manifold medial faces that is the skeleton representation of a sweepable region.
- Valence: a medial curve is incident to one or more medial faces. The number of medial faces that is incident on a medial curve determines the valence of a medial curve. If a medial curve is shared by two medial faces, the valence is two; if it is shared by three medial surfaces, the valence is three; and so on. In this paper, the term V2 is used to describe valence equals to 2; and V3 is used to describe valence equals to 3.

Fig. 3 The UML data structure to manage the medial information and the relationship between the B-Rep


Entity
+ID : int
+Center point : Vector3D
+Bound upper : Vector3D
+Bound lower : Vector3D
Entity::Medial Group
+Child medial face list : Medial Face
+Child defining entity list : Defining Entity
{1}
{1}
{1...*}
Entity::Medial Face
+No. of side : int
+Defining entity list : Defining Entity
+Neighbor medial face list : Medial Face
+No. of defining entity : int
+No. of medial curve : int
+Parent medial group : Medial Group
+Child medial curve : Medial Curve
{1}
Vector3D
-X : float
-Y : float
-Z : float
{1}
{1...*}
Entity::Medial Curve
+Curve Points : Vector3D
+Valence : int
+Defining entitiy list : Defining Entity
+Parent medial face list : Medial Face
+Curve type : int
{1}
{1...*}
{1...*}
Entity::Defining Entity
+Cubit ID : int
+Parent medial : Medial Face
{1...*}

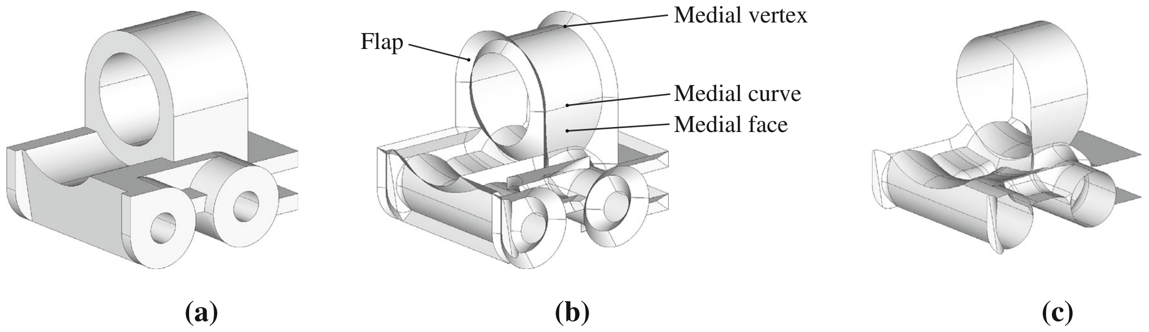


Fig. 4 **a** The original model. **b** The 3D MO of the model: a medial vertex, medial curve, medial face, and flap are labeled. **c** The trimmed MO

– Junction: a medial curve shared by more than two medial faces, in other words, a medial curve with valence greater than 1 (see Fig. 5).
– Touching site: a location on the boundary of the model which defines a particular section of MO. (See Fig. 5) A curve on the defining entity constructed by the touch sites is called touching curve.
– Defining entity: boundaries of the original model where the maximal sphere touches. These entities define the MO and hence are referred as defining entities (Fig. 5). The face entity is called defining surface; the curve entity is called defining curve; and the vertex is called defining vertex.

4 Sweepable region detection

The geometric reasoning engine detects the sweepable region using the 3D MO of the given model. The algorithm takes the 3D MO as input and returns the grouped sweepable medial faces as sweepable MO subsets. Generally, a sweepable medial face group is the skeleton representation of a sweepable volume. There is a one-to-one correspondence between the model boundaries and its MO. The sweepable region on the model can be detected by mapping the sweepable MO subsets to the model boundaries. The process is shown in Fig. 6. The MO and trimmed MO of the given model (Fig. 6a) are shown in Fig. 6b and c,

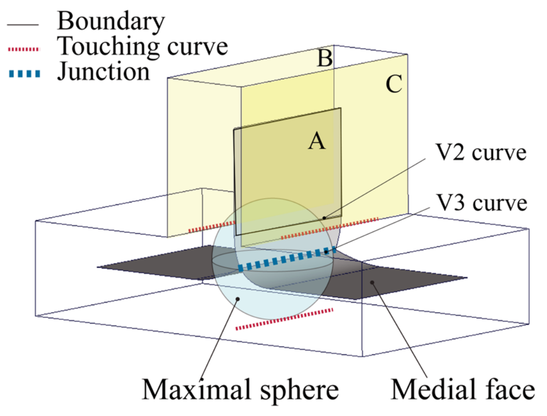


Fig. 5 The illustration of the medial terminology: a junction, V2 curve, V3 curve and the touching sites of the junction. The defining entities of medial face *A* is surface *B* and *C*

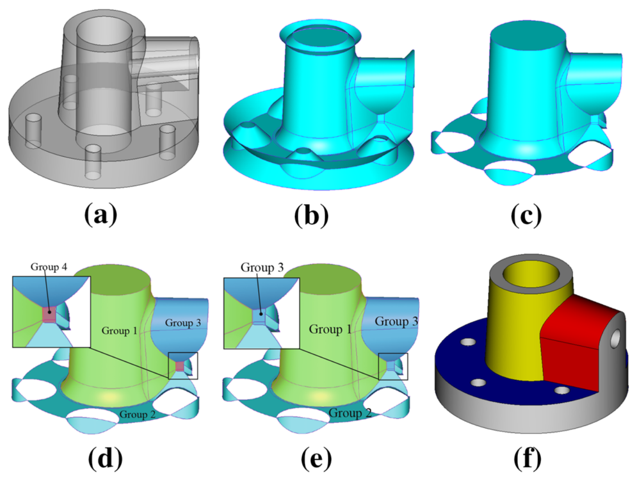


Fig. 6 **a** Original model. **b** MO. **c** Trimmed MO. **d** Four 2-manifold groups. **e** Non-manifold patch obtained by uniting groups 3 and 4. **f** Different *colors* are assigned to each sweepable sub-volume on the original model

respectively. Fig. 6d and g show the result of sweepable region detection. Four sweepable regions are detected as marked on Fig. 6f.

The MO can be break down into two types of sweepable geometries based on how each medial faces are connected to their neighbors:

– 2-manifold medial face group: a single medial face which has two defining surfaces is the skeleton representation of a sweepable region as shown in Fig. 7a. The two defining surfaces are the sweeping source and target surfaces.
  When two medial faces are 2-manifold—are connected via a V2 junction, they are a skeleton of a sweepable region (Fig. 7b).
– Non-manifold medial face group: a non-manifold medial face group as shown in Fig. 8 is sweepable from its 1D medial curves. The non-manifold medial face group has a corresponding 3D sub-volume on the original model.

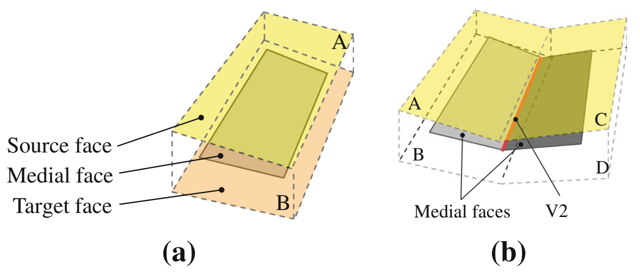


Fig. 7 **a** A MO face represents a sweepable volume. The two defining entities are assigned as sweep source and target. **b** Two medial faces, connected through a V2 curves represent two sweepable volumes, are united. Surfaces *A* and *C* are the sweep source and surface *B* and *D* are the target

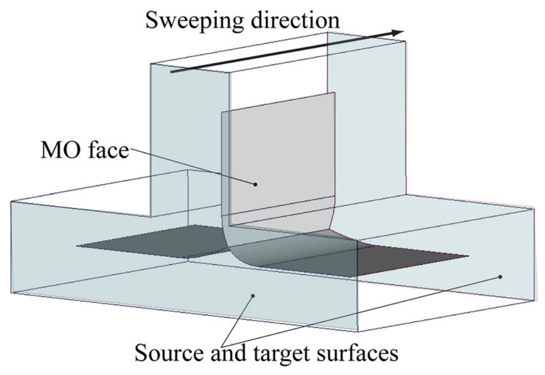


Fig. 8 The medial face group can be made sweepable if two end surfaces are assigned as the source and target surfaces of a sweeping operation

4.1 2-manifold grouping

Figure 9 demonstrates our breadth-first traversal MO segmentation algorithm on a MO. At the initial state (Fig. 9a), we start from the largest MO face (*L*). In state 2, *L* is assigned to a group (yellow) and its neighbor medial faces are detected (Fig. 9b). In state 3, the neighbor face across a V2 joins the current yellow patch (Fig. 9c); other neighbor faces across a V3 remain unvisited (painted in white). The neighbors of the newly added faces are detected. In state 4, one more neighbor faces across a V2 joins the yellow patch (Fig. 9d); the other two neighbors across a V3 remain unvisited. In state 5, the yellow patch has no more neighbors across V2 curves to visit (Fig. 9e). The yellow patch is now complete. Then we start over again from the largest unvisited MO face (painted in green). One neighbor face across a V2 is detected and joins the green group. In state 6 (Fig. 9f), the green group has no more neighbors across V2s to visit. The green patch is now complete. Search for the largest unvisited MO face (painted in grey) and repeat

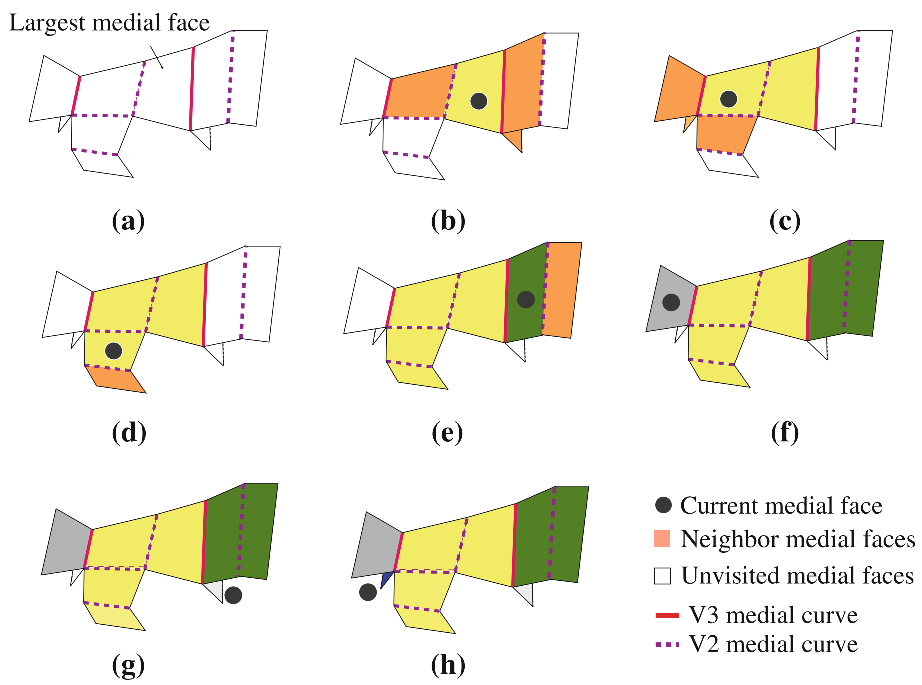


Fig. 9 A detailed 2-manifold grouping example. The *black dot* represents the current medial face in the stack for Breadth-first search (BFS). The faces marked in *orange* are the 2-manifold neighbors of the stack member

the grouping procedure. In state 7 (Fig. 9g), search for the largest unvisited MO face (paint in dark blue). This MO face has no unvisited neighbor faces, and forms a patch. In the final state, only one unvisited face remains (painted in dark blue). This face forms another patch. Fig. 9h shows segmentation result for this trimmed MO. The algorithm is shown in Algorithm 1.

Algorithm 1 2-manifold grouping algorithm

```
Input: The original model and its trimmed MO list $TrimmedMOFaceList$;
Output: MO groups;
 1: while $TrimmedMOFaceList$ is not empty do
 2:   Search for largest MO face ($L$) from $TrimmedMOFaceList$
 3:   Append $L$ to patch $S_i$
 4:   Append $L$ to $CurrentMOFaceList$
 5:   Remove $L$ from $TrimmedMOFaceList$
 6:   while The number of $CurrentMOFaceList$'s neighbor medial faces $! = 0$ do
 7:     for each current MO face $C$ do
 8:       if the valence of $C$'s child MO edge $== 2$ then
 9:         for each child MO edge do
10:           Search for parent medial faces ($P$)
11:           Append $P$ to $S_i$
12:           Append $P$ to $CurrentMOFaceList$
13:           Remove $P$ from $TrimmedMOFaceList$
14:         end for
15:       end if
16:       Remove $C$ from $CurrentMOFaceList$
17:     end for
18:     Remove $L$ from $CurrentMOFaceList$
19:   end while
20:   Remove $P$ from $CurrentMOFaceList$
21:   $i++$
22: end while
```

4.2 Non-manifold grouping

A non-manifold medial face group is a skeleton representation of a sweepable volume if these medial faces full fill these requirements:

- The junction shared by the non-manifold medial faces shares the same sets of end entities.
- The non-manifold medial faces are four-sided.

By grouping 2-manifold groups into a single non-manifold group can prevent non-manifold cutting surfaces and reduces the amount of cuts. In addition, the mesh quality of this region is better than sweeping it in the radius direction of the medial group.

4.2.1 Four-sided medial faces

If a junction curve connects more than two four-sided medial faces, the connected medial faces represented a sub-domain which is sweepable along the junction curve. The four-sided medial guaranteed the existence of linking surfaces. In Figs. 8 and 10, both cases have a

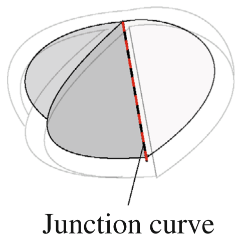


Fig. 10 A non-manifold medial face group that contains two-sided medial faces

junction curves shared by three medial faces. Medial faces in the first case are four-sided, and in the second case are two-sided. The corresponding volume in the case shown in Fig. 10 is not sweepable along the junction curve because there is no linking surfaces; and the corresponding volume case Fig. 8 can be swept along the junction curve. The algorithm to group four-sided medial faces shared by a medial curve with valence greater than 2 is shown in Algorithm 2.

Algorithm 2 non-manifold medial face grouping algorithm

```
Input: A list of junction curves with valance greater than 2
    medial faces $ListV3$ of the MO;
Output: Non-manifold medial face groups in
    $ListNonManifoldGroup$;
1: for Each medial curve from $ListV3$ do
2:     Append parent medial face to $ListPface$
3:     for Each medial face $F_4$ in $ListPface$ do
4:         if $F_4$ contains four child edge then
5:             Append $F_4$ to $ListNonManifoldGroup$
6:         end if
7:     end for
8: end for
```

The sweepable region detection process first groups the 2-manifold medial faces and then combines the four-sided medial faces that share the same junction to non-manifold groups. Next, maps each medial group to their corresponding sub-volumes on the model. Fig. 11 demonstrates an example of sweepable region detection. Starting with the 2-manifold grouping, each 2-manifold grouping stops when it encounters a junction as shown Fig. 11d. This step results in nine 2-manifold groups (M1, M5, M8, M9).

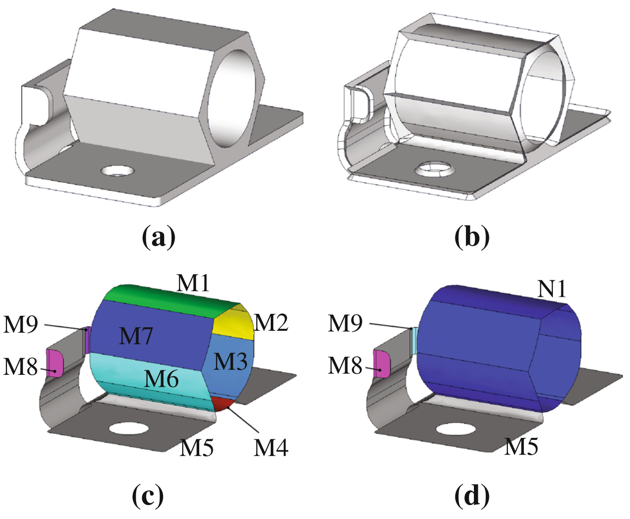


Fig. 11 **a** Original model. **b** 3D MO. **c** Nine 2-manifold groups (M1, …, M9). **d** A non-manifold group (N1) obtained by grouping (M1, M2, M3, M4, M6, M7)

Then, the non-manifold grouping algorithm groups M1, M2, M3, M4, M6, M7 into a non-manifold group N1, and results four sweepable groups as shown in Fig. 11d. These groups represent sweepable regions on the model. Groups M4, M5, M8, M9 are swept along the radius direction of their member medial faces, and N1 is sweepable along its junctions. Fig. 12 shows the sweepable region detection results created by the proposed process. Different colors are applied to each MO group.

5 Visual aids to assist in decomposition

The framework provides three types of visual aid to assist the user conduct volumetric decomposition: (1) sweepable region, (2) sweeping direction, and (3) ideal cutting region. The surfaces from the given model are color coded to represent different sweepable regions. The sweeping

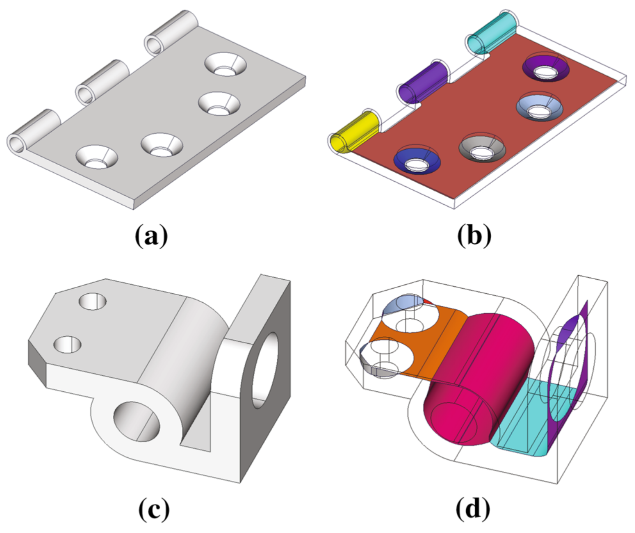


Fig. 12 Display the sweepable region detection result on the MO. **a**, **c** The original model. **b**, **d** Each sweepable MO groups are *color* coded

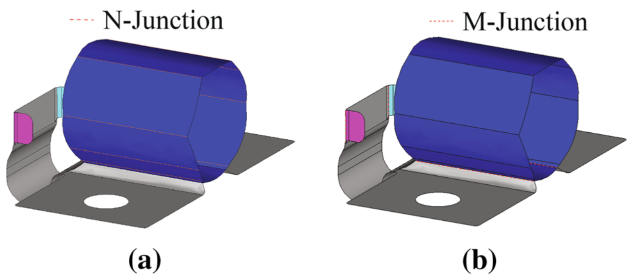


Fig. 13 Two types of junction. **a** N-Junction. N-Junctions exist in the non-manifold group (N1 as shown in Fig. 11). **b** M-Junction. M-Junctions split each sweepable groups and locate on the intersection of each group

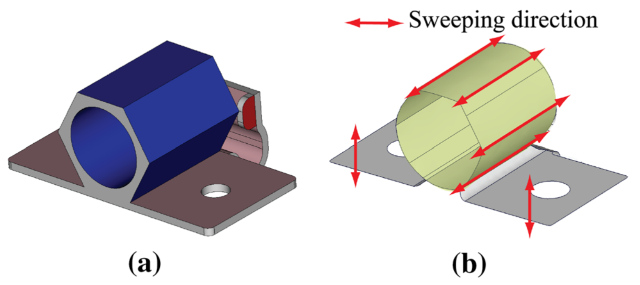


Fig. 14 **a** The four sweepable regions are visualized on the model. **b** The sweeping directions for each region can be identified using the junctions

directions are displays using arrows, and the ideal cutting regions are highlighted on the model surfaces.

5.1 Types of junction

Junctions play an important role in sweepable region detection. Two types of junctions are defined based on the grouping result (Fig. 13). To create the visual aids, each type of junction has different usages: M-Junction can be used to recognize ideal cutting position; N-Junction represents the sweeping direction of the non-manifold groups.

– N-Junction: a junction in a non-manifold medial face group.
– M-Junction: a junction that splits different sweepable medial face groups.

These two types of junctions are critical on detecting ideal cutting region and sweeping direction.

5.2 Sweepable region

The sweepable region can be identified by mapping each MO groups' defining surface to the original model.

A model shown in Fig. 13a contains four MO groups. The defining entities of each patch are color coded to visualize each sweepable region as shown in Fig. 14a. The suggested sweeping direction is displayed with arrows in Fig. 14b.

5.3 Sweeping direction

If a MO group does not contain any N-Junctions, it has a corresponding sub-volume which is sweepable along the radius direction of the child medial faces from the patch. If a medial patch contains a N-Junction, the patch has a corresponding sub-volume sweepable along the N-Junction. Fig. 15 shows with red arrows the sweepable direction on the non-manifold MO groups.

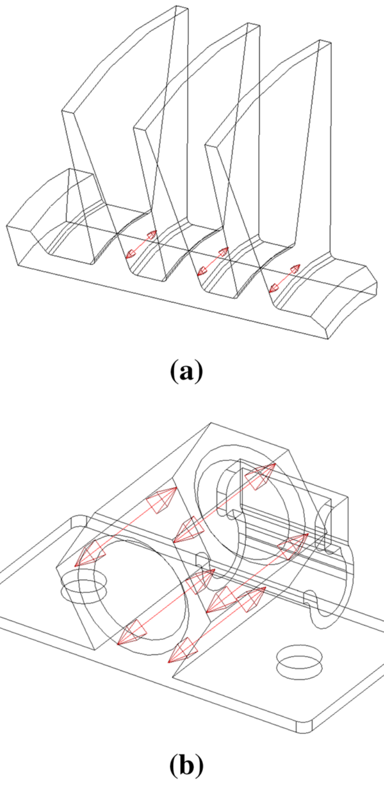
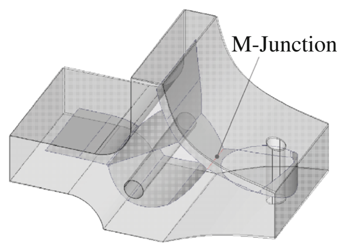


Fig. 15 Sweeping direction visualization. The *red arrows* demonstrate the sweeping direction. **a** The model is a non-manifold MO group, which is sweepable along the N-Junctions. **b** The model has one non-manifold MO group (N1 in Fig. 11d) and is sweepable along the N-Junctions

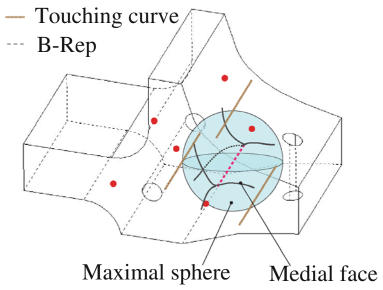
5.4 Ideal cutting position

The user's intent is to partition the model into sweepable sub-domains by creating cutting surfaces using freehand strokes or existing entities. Conduct decomposition at the ideal cutting position produces sweepable sub-domains and enhances mesh quality. For MO groups which do not contain any N-Junctions, the two defining surfaces of its member medial face are the sweeping source and target. Therefore, the ideal cutting position locates on the boundaries of the defining surfaces. However, in some cases the ideal cutting position does not overlap with any existing entities on the model. As shown in Fig. 16a, there is a cutting region which splits the different sweepable regions.

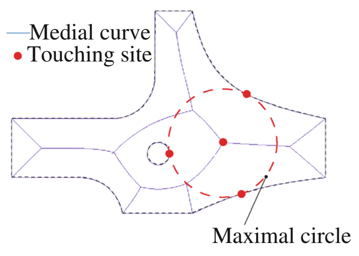
In step 1, the 3D medial is used to detect sweepable regions: each medial segment and group has a corresponding sweepable volume, and is split by M-Junctions. This means the touching sites of the M-Junctions represent the ideal cutting position on the boundaries of the model. The touching sites could be a vertex (the tangent point of the maximal sphere) or a curve. If the curve matches a B-Rep edge, the edge is the defining edge of the



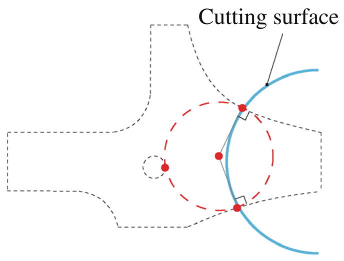
(a)



(b)



(c)



(d)

Fig. 16 An illustration of ideal cutting position. **a** A M-Junction splits two sweepable MO groups. **b** The touching curves of the M-Junction. **c** The *front view* of the medial and the B-Rep. The maximal sphere touches the boundary at three tangent points. **d** A cutting surface goes through the tangent points is perpendicular to the boundaries

M-Junction. Otherwise, the curve is the touching curve formed by a series of tangent points.

An example of using a touching curve for decomposition is shown in Fig. 16. The model has two medial patches split by an M-Junction. The touching curves shown in Fig. 16b are curves on the B-Rep that defines the M-Junction, which indicate the partition line on the boundary. The 2D medial on the front surface is shown in Fig. 16c. The touching site in the front view is the tangent point of the maximal circle. A cutting surface cut through the tangent points that is perpendicular to the boundaries is shown in Fig. 16d. This makes the cutting area to always have ideal hex element. By avoiding the bad angles at the cutting area, mesh quality is ensured.

6 Sketch-based decomposition

The sketch-based decomposition uses sketch-based operations to conduct decomposition following the geometric reasoning results described in the section above. The UI and basic sketch-based operations were introduced in [12]. The UI first takes user's freehand input as sequences of screen coordinates. The freehand inputs are then resampled and smoothed. The processed input point sets are then identified as one of the three types: lines, circles, or splines; and then are fitted to appropriate 1D geometry according to the type they are identified as. Cutting surfaces are created by sweeping the 1D geometry entity in a given sweeping direction, and are used to decompose the model.

6.1 Stroke alignment and snapping

In order to use the freehand stroke on accurate decomposition, the stroke is snapped to B-Rep edges and medial touching curves. The sketch-based UI evaluates the alignment type using the method proposed in our previous paper on the pen-based UI [12]. The alignment types include offset, overlap, perpendicular and concentric. However, the alignment evaluation algorithm does not handle the gaps between the stroke's end points to the boundaries after snapping, which makes the stroke unable to cut through the body. Stroke extension and vertex snapping functions are provided to solve this problem. For a stroke $S(p_0, \ldots, p_n)$, if its end points are located on the model surface and are not connected to any boundaries after snapping, we first search if there are any B-Rep vertices near the end points within a pre-defined distance. If so, we snap the end point to the closest vertex. Otherwise, we use the intersections of the boundaries and $\overrightarrow{p_1p_0}$ or $\overrightarrow{p_{n-1}p_n}$ as the new end points of the stroke.

6.2 Smart decomposition operation

The geometric reasoning engine intelligently determines the appropriate decomposition command. It detects the different purposes of the same type of input. The current

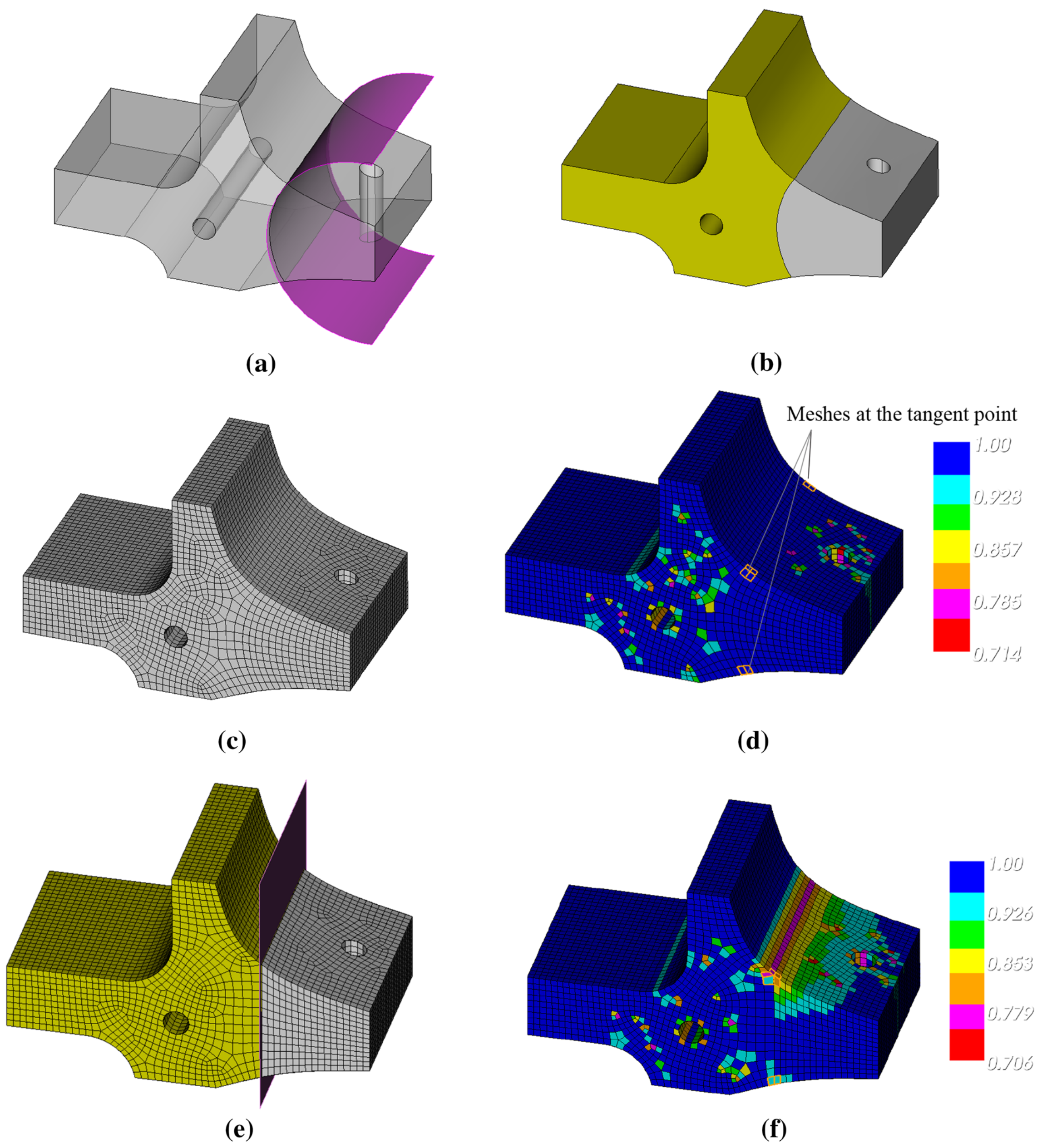


Fig. 17 **a** The cutting surface pass through the touching curve. **b** The decomposition result. **c** The all hex mesh. **d** The hex mesh quality using Scaled Jacobian. **e** Decompose the model with a planar surface. **f** The hex mesh quality using scaled Jacobian

implementation is able to create a cutting surface without manually specifying the operation by the following methods with the associated input type: (1) extending a picked surface. (2) Sweep a picked surface along another picked curve. (3) Revolve a picked surface by the axis of another picked periodic curve. (4) Revolve a picked curve by the axis of another picked periodic curve. (5) Fit a surface on a closed loop if the loop curves are picked. The automatic selection of one of the five operations is based on the number and type of picked geometric entities. If a user selects only one surface, the picked surface will be extended and the volume will be decomposed. If two entities are picked with the second one periodic, the first picked entity is assigned as a profile, and revolved about the axis defined by the second picked entity. If one or more curves are picked, we first check if the curves form(s) a closed loop. If so, then the picked curves are used as the bounding curves to create a cutting surface.

7 Results and discussion

Figure 17a shows a non-hex-meshable model. An arc cutting surface is the ideal cutting surface that subdivides the volume through the touching site of the M-Junction as shown in 16d. As discussed in Sect. 5.3, a surface cutting through the tangent points (touching site of the M-Junction) orthogonally ensures the generation of the ideal hex mesh at the cutting region. Figure 17c is the mesh generated using the decomposition solution obtained via

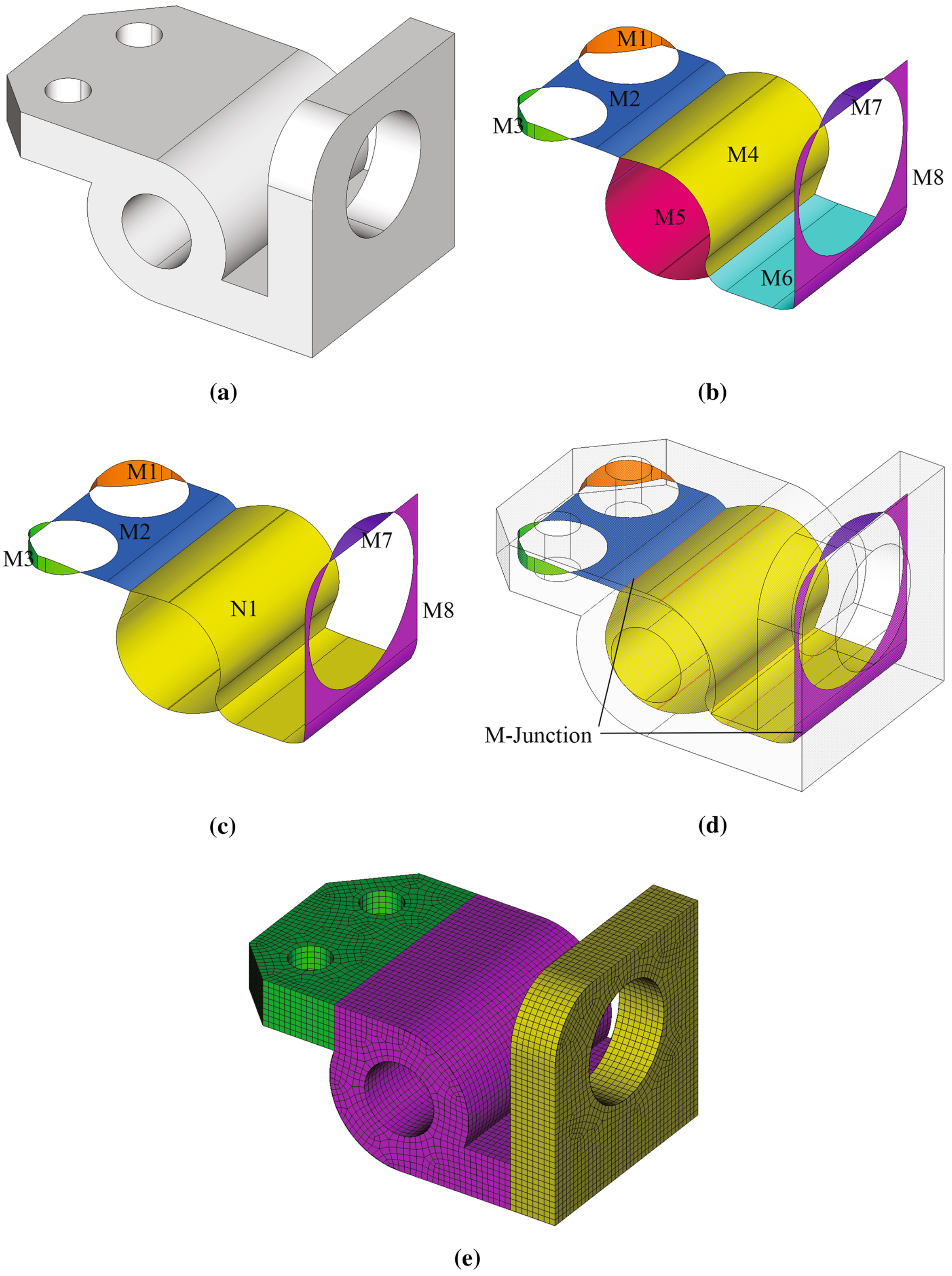


Fig. 18 Apply the proposed framework to an industrial model. **a** The original model. **b** The 2-manifold grouping result on the trimmed MO. Each 2-manifold groups are marked in *different colors*. **c** The non-manifold grouping result. **d** The *red curves* in the non-manifold group (N1) represent the sweeping direction visualization. **e** The model is decomposed into three regions and can be all hex-meshed

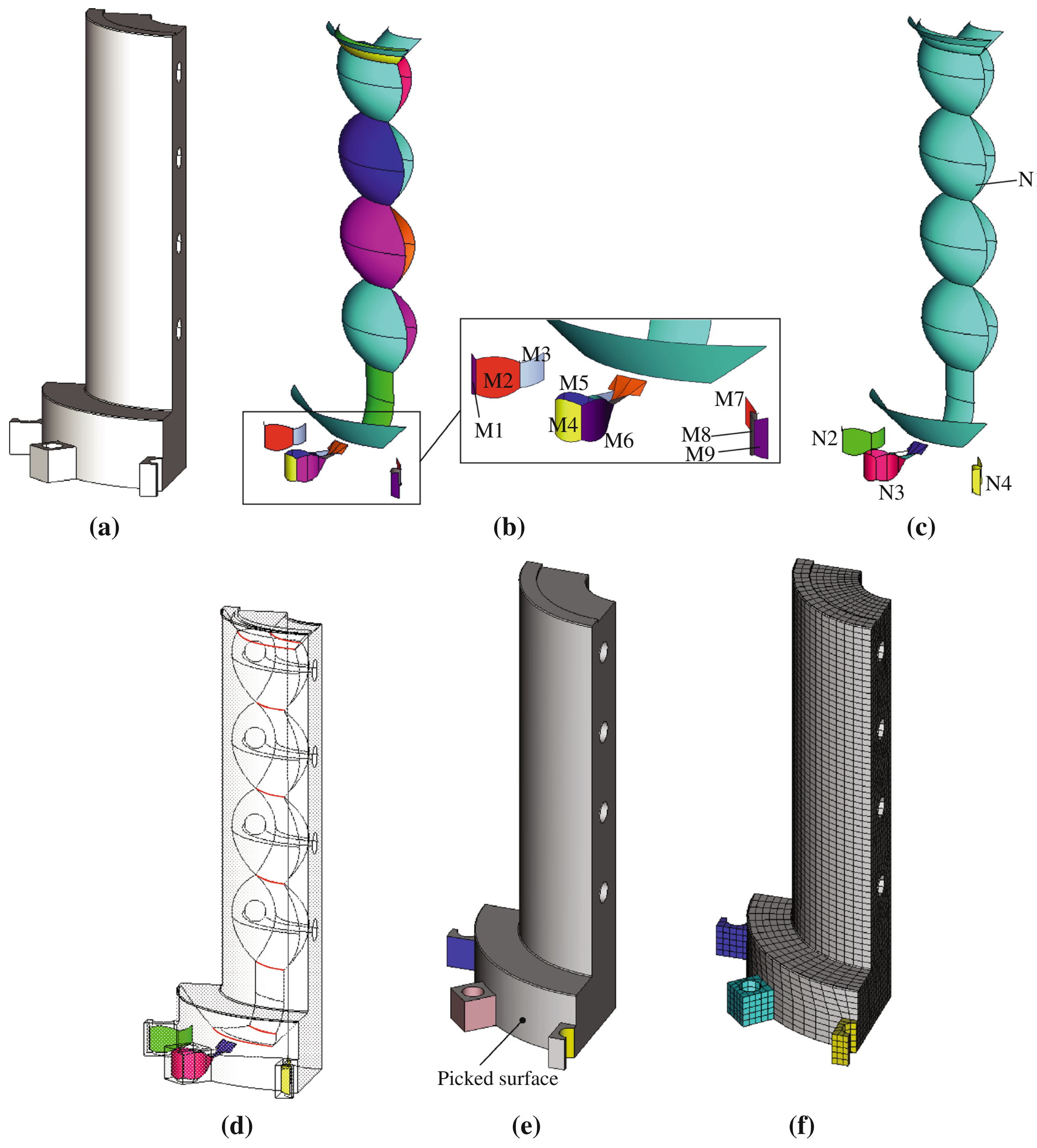


Fig. 19 An example of using the proposed framework on an industrial model. **a** The original model. **b** The 2-manifold grouping result on the trimmed MO. Each 2-manifold groups are marked in *different colors*. Part of the groups at the bottom are zoomed and labeled. **c** The non-manifold grouping result. **d** The *red curves* represent the sweeping direction visualization of N1. **e** The sweepable regions are visualized on the model. The user can pick the surface on the model following the visual aids to decompose the model. **f** The all hex-mesh result

geometric reasoning, and the hex elements highlighted in Fig. 17d have a min Scaled Jacobian of 0.969 at the tangent points. The framework intelligently creates the cutting surface that does not deteriorate the mesh quality. Using random planar cutting surface (commonly used manual solution) shown in Fig. 17e to produce sweepable subdomains; however, the mesh quality at the cutting region has a min Scaled Jacobian of 0.746. An arc that is perpendicular to the boundaries could be created using the proposed sketch-based UI very easily; however, it does not guarantee a 90 ° intersection angle. When the geometric reasoning via a medial touching curve is used in combination with a user-friendly sketch UI, the ideal cutting surface can be created.

A non-hex-meshable industrial model is shown in Fig. 18a. The 2-manifold grouping result on the trimmed MO is shown in Fig. 18b: (M1, M2,…, M7). The non-manifold grouping result is shown in Fig. 18c. The

2-manifold groups (M4, M5, M6) are grouped to a non-manifold group N1. The red curves shown in Fig. 18d are the N-Junctions of N1, which represent the sweeping direction of the N1's corresponding sub-volumes on the original model. Two M-Junctions are the medial curves that connect group M2, N1, and M8. The touching suites of these curves on the model could help the user making decision on where to conduct the decomposition. The corresponding cuts referring to these two M-Junctions split the model into three regions, and produce an all hex-meshed result as shown in Fig. 18d.

Fig. 19a shows an industrial model that requires decomposition for generation of hex mesh. The 2-manifold grouping result on the trimmed MO is shown in Fig. 19b. Each 2-manifold groups are marked in different colors. Part of the groups at the bottom is zoomed and labeled. The non-manifold grouping result is shown in Fig. 19c. 2-manifold groups (M1, M2, M3) are grouped to a non-manifold group N2; (M4, M5, M6) are grouped to N3; (M7, M8, M9) are grouped to N4. The rest of the 2-manifold groups which forms the biggest piece is grouped to N1. The red curves shown in Fig. 19d represent the

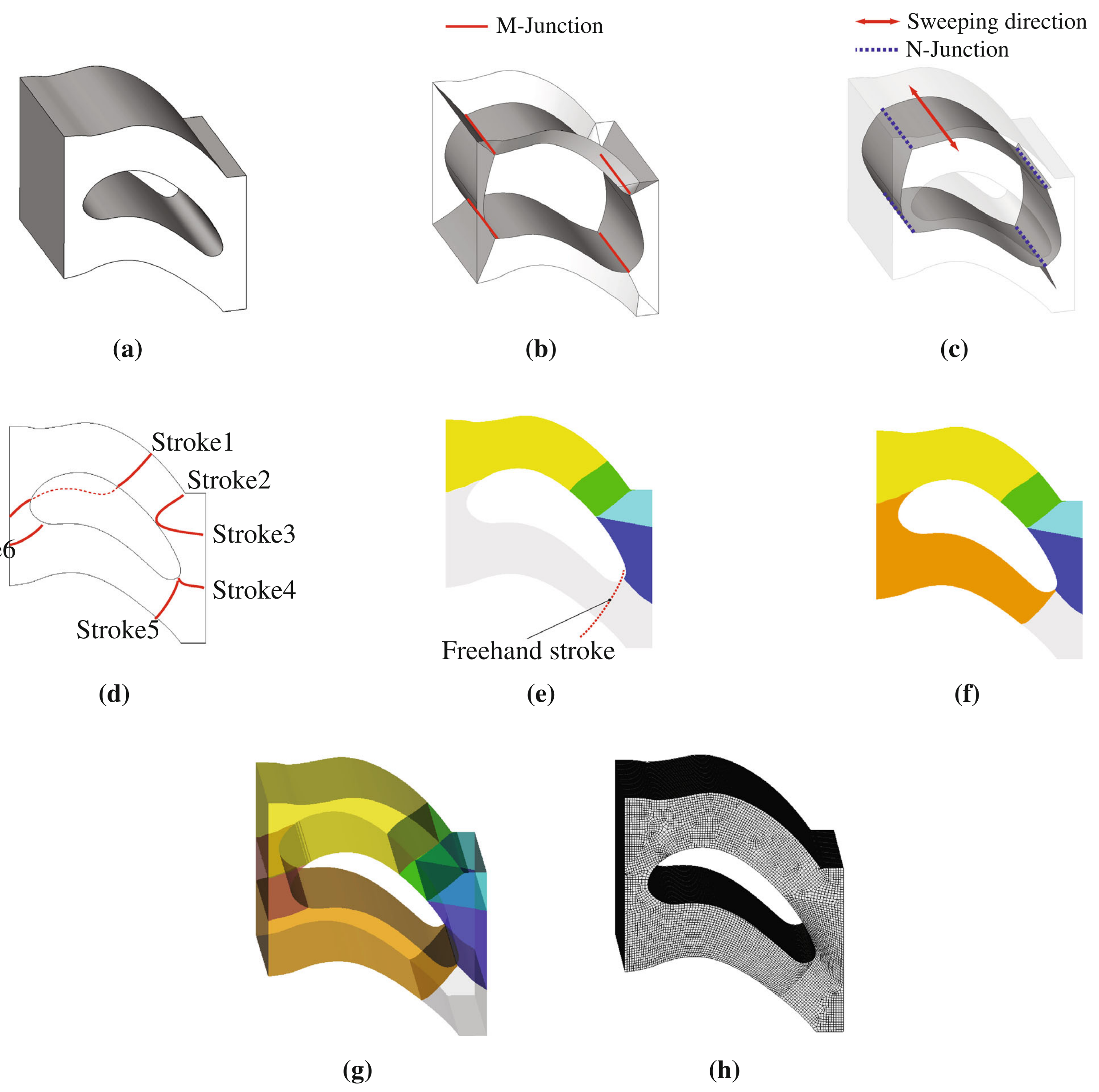


Fig. 20 Apply the proposed framework to CFD blocking. **a** A flow field around a turbine. (Reproduced from [18]) **b** Four M-junctions split many medial segments. **c** The segments have been grouped into one single patch, which is sweepable along the N-junctions. **d** The illustration of strokes to decompose the model. **e** The user draws stroke 5. **f** The decomposition result after conducting five cuts. **g** The final decomposition result. **h** The all hex mesh output

N-Junction in the non-manifold group N1. These curves are the sweeping direction of the corresponding sub-volumes of N1 on the original model. Fig. 19e shows the sweepable region visualization. The smart decomposition in the sketch-based UI allows the user to pick the specific surface to decompose the model. The all hex-meshed result is shown in Fig. 19f.

Appropriate topology must be chosen to ensure the mesh quality; therefore, the volume has to be decomposed into blocks which have appropriate topology for mapping/sub-mapping. In some cases, the mesh size and orientation are constrained to meet computational fluid dynamics (CFD) requirements.

Figure 20a shows a fluid field around a turbine blade. The 3D MO is split into many 2-manifold MO groups by M-Junctions (Fig. 20b). After the non-manifold grouping process, Fig. 20c shows that the 2-manifold MO groups are grouped as one single non-manifold group, which is sweepable along the N-Junction, which indicates the whole volume is sweepable along this direction.

Using the proposed sketch-based UI, the end points of the freehand strokes are automatically snapped to the corner if they are very close. Five freeform cutting surfaces can be easily created with freehand strokes following the pattern shown in Fig. 20d. Note that the first stroke must cut through the whole volume to generate two sub-domains. When drawing the strokes, the end points are snapped to vertices if they are close to corners. Therefore, we can obtain an accurate shape for the blocks. Figure 20e and f demonstrate the decomposition using stroke 5. After one more cut with stroke 6, seven sub-domains are generated as shown in Fig. 20g. By conducting imprinting and merging, the volume can be all hex-meshed as shown in Fig. 20h by sweeping along the N-junction.

The geometric reasoning result tells the volume is sweepable in the N-junction direction, and the sketch-based UI allows the user to conduct accurate decomposition in a user-friendly and intuitive way. By using the proposed framework on this example, the block structure can be generated easily, and the mesh can be oriented along the block boundaries.

It is possible for the inscribed sphere to touch one surface at infinite points when traveling through the volume. In this over-constrained situation, the medial surface could degenerate to a line—as the inscribed sphere rolls along the axis of a cylinder); or a point—as the inscribed sphere rolls in another sphere. An important extension of the proposed method is handling volumes with a degenerate MO. In the degenerate case, Price et al. [19, 20] created primitives that are not part of the pre-defined set for non-degenerate volumes. Applying midpoint subdivision to these regions may result in poor mesh quality. In contrast, the proposed method can be applied to general geometries except for shapes whose MO itself is degenerate. The sweepable region detection algorithm does a medial face search so regions that contain degenerate medial axes are excluded from the medial face grouping routine.

The robustness of 3D MO generation may affect the result of the method proposed in this paper. It has been known that 3D MO generation is difficult for shapes that contain small features and exclusively complex shapes. Removing small, irrelevant features prior to the MO generation using graph-based feature recognition techniques could be potential solutions.

8 Conclusion

The paper presents a sketch-based volumetric decomposition framework using geometric reasoning to speed up the challenging and time-consuming decomposition process. The geometric reasoning approach infers user's intent and returns an accurate decomposition result from the rough sketch-based inputs. One of the main contributions is that the geometric reasoning brings the smartness aspect to the framework by: (1) detecting sweepable regions and sweeping direction; (2) providing visual aids for decomposition; (3) understanding user's intent by skeletal/MO-based snapping candidates, and determining ideal cutting position and alignment/snapping types; and (4) conducting smart decomposition. The proposed method has been tested on industrial models by generating hex meshes using sweeping algorithms.

Acknowledgments The authors would like to thank Dr. Geoffrey Butlin, Mr. Henry Bucklow, Mr. Robin Fairey, Mr. Mark Gammon, Mr. Mike Field, and Mr. John Lamont for assisting with the medial related work in CADFIX.

References

1. Steven EB, Ernest P, Karl M, Brett C, Greg S (1995) A comparison of all-hexahedral and all-tetrahedral finite element meshes for elastic and elasto-plastic analysis. In: Proceedings of the 4th International Meshing Roundtable, pp 179–191
2. Blum H (1967) A transformation for extracting new descriptors of shape. In: Walthem-Dunn (ed) Models for the perception of speech and visual form. MIT Press, Cambridge, MA, pp 362–380
3. Chong CS, Senthil Kumar A, Lee KH (2004) Automatic solid decomposition and reduction for non-manifold geometric model generation. Comput Aided Des 36(13):1357–1369
4. Cifuentes AO, Kalbag A (1992) A performance study of tetrahedral and hexahedral elements in 3-d finite element structural analysis. Finite Elem Anal Des 12:313–318
5. Donaghy RJ, Armstrong CG, Price MA (2000) Dimensional reduction of surface models for analysis. Eng Comput 16:24–35
6. Folwell NT, Mitchell SA (1998) Reliable whisker weaving via curve contraction. In: Proceedings of the 7th International Meshing Roundtable, pp 365–378

7. Hardwick M (2005) In DART system analysis presented to simulation sciences seminar
8. Igarashi T, Matsuoka S, Tanaka H (1999) Teddy: a sketching interface for 3D freeform design. In: Proceeding of the 26th annual conference on computer graphics and interactive, pp 409–416
9. ITI TranscenData (2013) CAD Translation-CADFix. In: http://www.cadfix.com
10. Kara LB, Shimada K (2006) Construction and modification of 3D geometry using a sketch-based interface. In: Proceeding of the EUROGRAPHICS Workshop on sketch-based interfaces and modeling, pp 59–66
11. Li TS, McKeag RM, Armstrong CG (1995) Hexahedral meshing using midpoint subdivision and integer programming. Comput Methods Appl Mech Eng 124(1-2):171–193
12. Lu JHC, Song I, Quadros WR, Shimada K (2010) Pen-based user interface for geometric decomposition for hexahedral mesh generation. In: Proceedings of the 19th International Meshing Roundtable, pp 263–278
13. Lu Y, Gadh R, Tautges TJ (2001) Feature based hex meshing methodology: feature recognition and volume decomposition. Comput Aided Des 33(3):221–232
14. Luo X-J, Shephard MS, Yin L-Z, OBara RM, Nastasi R, Beall MW (2010) Construction of near optimal meshes for 3d curved domains with thin sections and singularities for p-version method. Comput Methods Appl Mech Eng 26:215–229
15. Makem JE, Armstrong CG, Robinson TT (2012) Automatic decomposition and efficient semi-structured meshing of complex solids. In: Proceeding of the 20th International Meshing Roundtable. Springer, Berlin, Heidelberg, pp 199–215
16. Masry M, Kang D, Lipson H (2005) A freehand sketching interface for progressive construction of 3D objects. Comput Gr 29(4):563–575
17. Owen SJ, Clark B, Melander DJ, Brewer ML, Shepherd J, Merkley KG, Ernst C, Morris R (2007) An immersive topology environment for meshing. In: Proceeding of the 16th International Meshing Roundtable, Sandia National Laboratories, pp 553–578
18. Pointwise Inc (2011) Multi-block grids for axial turbines. In: http://www.pointwise.com/theconnector/March-2011/Gridding-an-Axial-Turbine-Video.shtml
19. Price MA, Armstrong CG (1997) Hexahedral mesh generation by medial surface subdivision: part II. Solids with flat and concave edges. Int J Numer Methods Eng 40(1):111–136
20. Price MA, Armstrong CG, Sabin MA (1995) Hexahedral mesh generation by medial surface subdivision: part I. Solids with convex edges. Int J Numer Methods Eng 38(19):3335–3359
21. Quadros WR, Ramaswami K, Prinz FB, Gurumoorthy B (2004) LayTracks: a new approach to automated geometry adaptive quadrilateral mesh generaton using medial axis transform. Int J Numer Meth Eng 61:209–237
22. Sampl P (2000) Semi-structured mesh generation based on medial axis. In: Proceeding of the 9th International Meshing Roundtable, pp 21–32
23. Sandia National Laboratories (2013) Cubit: Geometry and meshing toolkit. In: https://www.cubit.sandia.gov
24. Schneiders R (1995) Automatic generation of hexahedral finite element meshes. In: Proceedings of the 4th International Meshing Roundtable, pp 103–114
25. Schneiders R (1996) A grid-based algorithm for the generation of hexahedral element meshes. Eng Comput 12:168–177
26. Schoof L, Yarberry V (1995) Exodus II a finite element data model. SAND92-2137, Sandia National Laboratories
27. Sheffer A, Etzion M, Bercovier M (1999) Hexahedral mesh generation using the embedded voronoi graph. In: Proceedings of the 7th International Meshing Roundtable, pp 347–364
28. Shepherd JF, Johnson CR (2008) Hexahedral mesh generation constraints. Eng Comput 24(3):195–213
29. Shih BY, Sakurai H (1996) Automated hexahedral mesh generation by swept volume decomposition and recomposition. In: Proceeding of the 5th International Meshing Roundtable, pp 273–280
30. Shih BY, Sakurai H (1997) Shape recognition and shape-specific meshing for generating all hexahedral meshes. In: Proceeding of the 6th International Meshing Roundtable, pp 197–209
31. Tam T, Armstrong CG (1991) 2D finite element mesh generation by medial axis subdivision. Adv Eng Softw 13(5–6):313–324
32. Tautges TJ, Blacker T, Mitchell SA (1996) The whisker weaving algorithm: a connectivity-based method for constructing all-hexahedral finite element meshes. J Numer Methods Eng 39: 3327–3349
33. Timothy JT (2000) The common geometry module (CGM): a generic, extensible geometry interface. In: Proceeding of the 9th International Meshing Roundtable, Sandia National Laboratories, pp 337–348
34. Varley PAC, Suzuki H, Mitani J, Martin RR (2000) Interpretation of Single Sketch Input for Mesh and Solid Models. Int J Shape Model 6:207–240
35. White D, Mingwu L, Benzley SE, Sjaardema GD (1995) Automated hexahedral mesh generation by virtual decomposition. In: Proceeding of the 4th International Meshing Roundtable, Sandia National Laboratories, pp 165–176
36. Yamakawa S, Gentilini I, Shimada K (2011) Subdivision templates for converting a non-conformal hex-dominant mesh to a conformal hex-dominant mesh without pyramid elements. Eng Comput 27:51–65