An efficient curvature-based partitioning of large-scale STL models

Jingbin Hao
College of Mechanical and Electrical Engineering, China University of Mining and Technology, Xuzhou, China

Liang Fang
Department of Mechanical Engineering and Automatics, China University of Mining and Technology, Xuzhou, China, and

Robert E. Williams
Department of Industrial and Management Systems Engineering, University of Nebraska-Lincoln, Lincoln, Nebraska, USA

Abstract

Purpose – Rapid prototyping (RP) of large-scale solid models requires the stereolithographic (STL) file to be precisely partitioned. Especially, the selection of cutting positions is critical for the fabrication and assembly of sub-models. The purpose of this paper is to present an efficient curvature-based partitioning for selecting the best-fit loop and decomposing the large complex model into smaller and simpler sub-models with similar-shaped joints, which facilitate the final assembly.

Design/methodology/approach – The partition algorithm is benefited from curvature analysis of the model surface, including extracting the feature edges and constructing the feature loops. The efficiency enhancement is achieved by selecting the best-fit loop and constructing the similar-shape joints. The utility of the algorithm is demonstrated by the fabrication of large-scale rapid prototypes.

Findings – By using the proposed curvature-based partition algorithm, the reasonability and efficiency of STL model partition can be greatly improved, and the complexity of sub-models has been reduced. It is found that the large-scale model is efficiently partitioned and the sub-models are precisely assembled using the proposed partitioning.

Originality/value – The curvature-based partition algorithm is used in the RP field for the first time. Based on the curvature-based partitioning, the reasonability and efficiency of large-scale RP is addressed in this paper.

Keywords Rapid prototypes, Manufacturing systems, Modelling

Paper type Research paper

1. Introduction

The main benefit of rapid prototyping (RP) is its ability to build arbitrary geometries without conventional cutting tools. In recent years, RP technologies have begun to be used as direct manufacturing techniques in applications such as automobile and aerospace (Pham and Dimov, 2001). However, the current application of RP is limited in large-scale model fabrication, because of two aspects:

1 The maximum dimension of the RP machine is confined, normally $<600 \times 600 \times 600$ mm (e.g. the work volume of 3D systems' SLA-7000 is $508 \times 508 \times 600$ mm). The partition of the large-scale model is generally manual and experiential, so it is difficult to ensure the rationality and quality of model partition.
2 For large-scale solid models (particularly those with complex shapes), the amount and complexity of sub-models have direct impact on the efficiency and precision of RP. As the stereolithographic (STL) file format is the standard data format of RP, the partition of large-scale STL models has been considered a fundamental problem, which is a necessary pre-processing step for large-scale RP.

The planar division (Sun *et al.*, 2003) is the simplest method, which divides the model into a set of fit-size sub-models by certain planes. However, the cutting positions are selected by experience. The cut planes need to divide the original faces of boundaries and may pass through some regions which are no need to be divided. Moreover, the large-scale models usually have certain components that characterize their important features (such as wings or legs of animal models, functional structures of mechanical parts). If the cutting position is inappropriate, it may cause more complex structures, which

The authors would like to thank the Department of Industrial and Management Systems Engineering (IMSE) and the University of Nebraska-Lincoln for providing the research facility and support. The authors would like to thank Professor Tim Hemsath and the UNL College of Architecture for building the prototypes. This research is supported in part by the State Scholarship Fund of the China Scholarship Council (No. 2008642010), the Major State Basic Research Development Program of China (973 Program) (No. 2007CB607605) and China Postdoctorate Science Foundation (No. 20090461157).

Received: 19 October 2009
Revised: 16 February 2010, 11 April 2010
Accepted: 12 April 2010

Rapid Prototyping Journal
17/2 (2011) 116–127
© Emerald Group Publishing Limited [ISSN 1355-2546]
[DOI 10.1108/13552541111113862]

will affect the fabricating efficiency and the assembly quality of sub-models. So the planar division is inefficient for the complex large-scale model partition.

One common strategy for dealing with a large complex model is to decompose it into smaller and simpler sub-models. As STL files use triangle meshes to represent the surface of a solid model, the core issue is how to decompose and reconstruct triangle meshes. In computational geometry, several approaches have been discussed in the past for mesh decomposition. Convex decomposition schemes (Chazelle and Palios, 1992; Lien and Amato, 2004) are proposed to decompose the surface into pieces that are easy to process. However, the complex model has many small concavities which result in over-segmentation. Watershed decomposition (Mangan and Whitaker, 1999; Razdan and Bae, 2003) can resolve the over-segmentation problem, but it is less useful for planar decomposition. Face clustering (Garland *et al.*, 2001) and K-means clustering (Shlafman *et al.*, 2002; Katz and Tal, 2003) can distinguish meaningful components from the remaining object. But for huge triangle meshes, it will be a time-costly process because of multiple clustering iterations. In this paper, we use an efficient curvature-based partitioning that decomposes a large-scale model into several meaningful sub-models which are easier to fabricate. A meaningful sub-model refers to a component which can be perceptually distinguished from the remaining model (Biederman, 1987).

Another main issue of the partition strategy is the precision of sub-models' assembly. The planar division cannot ensure the precision of sub-models' assembly. Ladder-like division (Chen *et al.*, 2004) can construct the ladder-like splice surface to improve the assembly, but the construction of the ladder-like surface is more complex and this splice surface only restrict two degrees of freedom for assembly. The location joints can be added and subtracted to sub-models by Boolean operation (Chen, 2007), but the joint model needs to be first built according to the splice surface of sub-models. The key idea of our algorithm is to build the similar-shaped joint on the splice surface to facilitate the assembly. The similar-shaped joint can be directly built based on the cutting contour.

The remainder of the paper proceeds as follows. The next section describes some aspects of the literature that are related to our proposed partitioning. Section 3 discusses the curvature estimation of edges, whereas Section 4 describes the algorithm details of the curvature-based partitioning. Section 5 presents the building of similar-shaped joints. Section 6 shows some experimental cases and analyzes the performance of the proposed partitioning. Finally, Section 7 concludes and discusses future directions.

2. Related work

To efficiently partition the STL model, we first need to find the boundaries between the meaningful sub-models, which pass at regions of deep concavities. Consisting of an unordered list of triangular faces, an STL file represents the outside surface of an object. An STL model (denoted by M), consists of a set of n points (vertices $v_i \in \mathrm{E}^3, 0 \leq i < n$) and a set of planar convex polygons (faces) made up of these vertices. There are three major methods that can be used for the curvature estimation of an STL model: vertex-, edge- and face-based methods.

2.1 Vertex-based method

The definition of a sub-model is the one in which regions consist of connected vertices which have the same curvature value (within a tolerance). One typical vertex-based method is the watershed segmentation algorithm, which has been used previously in image segmenting (Mangan and Whitaker, 1999). The major drawback to this method is that no hard boundaries are created for the features or regions. Razdan and Bae (2003) proposed a hybrid approach for resolving the boundary problem. One problem with this approach is the dependency on the additional triangulation of the model.

2.2 Edge-based method

A feature edge is an edge shared by two faces whose normal vectors make an angle greater than a certain threshold. These obtained edges are integrated into curves, which can be classified as feature boundaries (Srikantiah, 2000). The edge-based method can find the meaningful cut plane, but the disadvantage is extracting many disconnected feature edges, thereby resulting in incomplete feature loops.

2.3 Face-based method

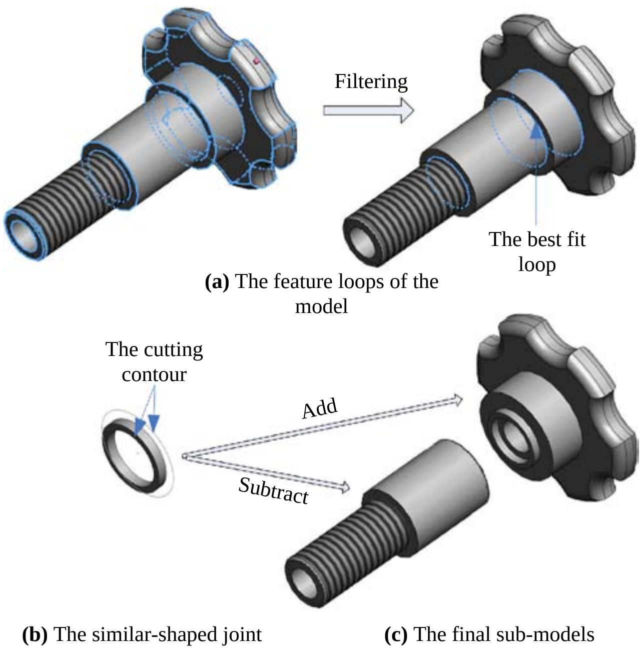
The face clusters, which are connected sets of faces, represent the aggregate properties of the original surface at different scales rather than providing geometric approximations of varying complexity (Garland *et al.*, 2001). The clusters may be well approximated with planar elements, but the boundaries of the cluster are not appropriate for meaningful partition.

Comparing these three methods, the vertex- and the face-based methods are inefficient for the large-scale model, because of multiple clustering of vertices and merging of triangle meshes (Li *et al.*, 2001; Kanungo *et al.*, 2002). In contrast, the edge-based method is the most effective way to find the feature boundaries of the large-scale model, where the model can be partitioned meaningfully. Obviously, the process of feature boundaries is faster than that of the whole model. The main challenges of the edge-based method include extracting and grouping the feature edges, selecting the best fit loop and constructing the cutting contour. The similar-shaped joint is achieved based on the cutting contour to ensure the assembly of sub-models. Our algorithm consists of four stages:

1 Estimating curvature at each edge of the model and extracting the feature edges.
2 Grouping and filtering the feature edges to approach the feature loops and selecting the best fit loop.
3 Constructing the cutting contour with the best fit loop to hierarchically partition the model.
4 Building the similar-shaped joints on the splice surfaces of sub-models.

For instance, we wish to partition the model in Figure 1. After estimating the curvature of the edges, the feature edges are extracted by certain thresholds. In Figure 1(a), the feature loops have been constructed by grouping the feature edges. Based on the partition condition – the build size of RP machine is $600 \times 600 \times 600$ mm and the original model is $800 \times 450 \times 450$ mm, X-axis is first selected as the cutting axis. One best fit loop is selected to partition the model into two sub-models. The similar-shaped joint has been built based on the cutting contour (Figure 1(b)) and added or subtracted to the paired sub-models (Figure 1(c)). Since the

Figure 1 The optimal partition of one model



(a) The feature loops of the model

(b) The similar-shaped joint

(c) The final sub-models

two sub-models are less than the build size, the partition has been ended.

3. Curvature estimation of edges

This section describes three curvature criteria to estimate and weight the edges. The feature edges with greater value than a certain threshold can be extracted and grouped as the feature loops.

3.1 Dihedral angle

The dihedral angle of the adjacent faces is the main curvature parameter, which has been used to estimate surface curvature (Dong *et al.*, 2008). We define $e(i, j)$ as the adjacent edge of faces i and j. n_i and n_j are the normal vectors. The dihedral angle of $e(i, j)$ is:

$$\theta(e(i,j)) = \arccos(\mathrm{n}_i \cdot \mathrm{n}_j) \tag{1}$$

Considering the various resolutions of triangulation (most modeling software choose 10° as default value, such as SolidWorks, Pro/E), the maximal dihedral angle α (under 10°) and the minimal dihedral angle β (over 10°) are calculated. The threshold value ε is given by:

$$\varepsilon = \frac{\alpha + \beta}{2} \tag{2}$$

If the dihedral angle θ is bigger than the given threshold value ε, the edge e is marked as the feature edge. For the regular shapes and the obvious curvature changes, the feature edges can be directly distinguished by the dihedral angle.

3.2 The perimeter ratio

In some regions which have gradual curvature changes, the boundary edges have smaller dihedral angles than the threshold value. For example, the boundary edges of the fillets have smaller dihedral angles (as shown in Figure 2), which could not be eligible as the feature edge.

In view of these boundary lines, there is a big perimeter difference between two adjacent faces. Our algorithm applies another criterion – the perimeter ratio – to extract these edges. We define C_i and C_j as the perimeters of faces i and j. The perimeter ratio r is given by:

$$r(e(i,j)) = \frac{C_{\max}}{C_{\min}} \tag{3}$$

where $C_{\max} = \max(C_i, C_j)$, $C_{\min} = \min(C_i, C_j)$. If $\theta > 0°$, and r is bigger than a certain threshold λ, this edge will be marked as the feature edge. After calculating the perimeter ratios of every edge, set R as the average of all perimeter ratios. Based on the experiments by Yang *et al.* (2008), the extraction of these feature edges is most efficient when $\lambda = 4R$.

3.3 Convexity

Since the convexity of a model surface is the useful information to determine the feature loop, we add the convexity parameter to edges. Based on the right-hand rule, the vertices of face $i : V_i = (a, b, c)$ and the vertices of face $j : V_j = (a, d, b)$. We define $e(i, j) = ab$ and $e(j, i) = ba$. The convexity of $e(i, j)$ is given by:

$$\eta(e(i,j)) = (\mathrm{n}_i \times \mathrm{n}_j) \cdot \mathrm{n}(e(i,j)) \tag{4}$$

As shown in Figure 3, if $\eta > 0$, $e(i,j)$ is convex. If $\eta > 0$, $e(i,j)$ is concave. Otherwise, $e(i, j)$ is planar. Note that $\eta(e(j, i))$ is the same as $\eta(e(i,j))$, because they denote the same feature of faces i and j. By the experiments of Lien and Amato (2004), a concave feature makes a better candidate for a cut plane. So, we check for the concave feature loops prior to the convex ones when selecting the best fit loop.

Using the above-mentioned three criteria, the whole edges have been weighted with these three parameters. The four experimental models in Figure 4(a) have different size, complexity and resolution (as shown in Table I), so the feature edges have been extracted by their own thresholds (Figure 4(b)). We suppose that a model has n faces, the number of edges is $3n/2$. The process of curvature estimation takes $O(3n/2)$ time complexity.

4. Algorithm – the curvature-based partition

This section describes the algorithm details of the curvature-based partitioning. For the large-scale model with certain

Figure 2 The boundary edges of the fillets

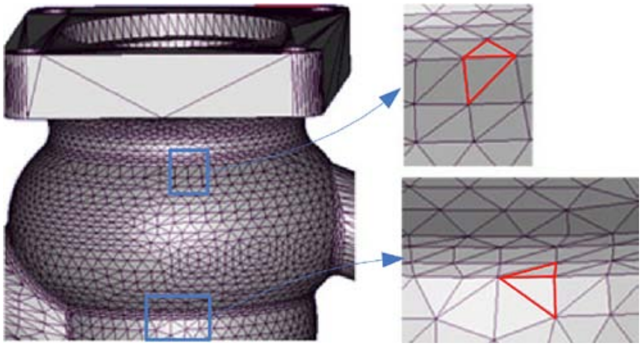
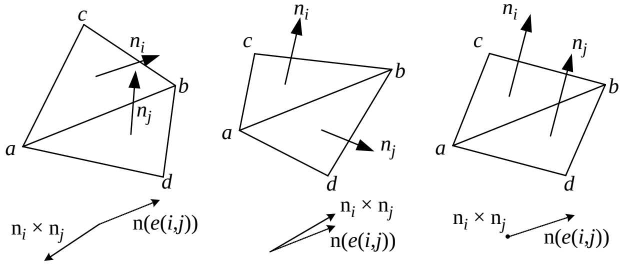


Figure 3 The convexity of $e(i, j)$



(a) The concave edge **(b)** The convex edge **(c)** The planar edge

feature components, our algorithm is to use the best fit loops to partition the model hierarchically. The extracted feature edges are independent in the edges array, so we group sets of similar and adjacent feature edges to be the feature loops. The best fit loop will be selected from the feature loops according to the weight of feature loop and the estimation of partition condition. The estimation of partition condition is also useful to the planar division of the large simple model.

4.1 Constructing feature loops

The feature loops are critical boundaries which contain sets of similar and adjacent feature edges. There are two kinds of feature loops:

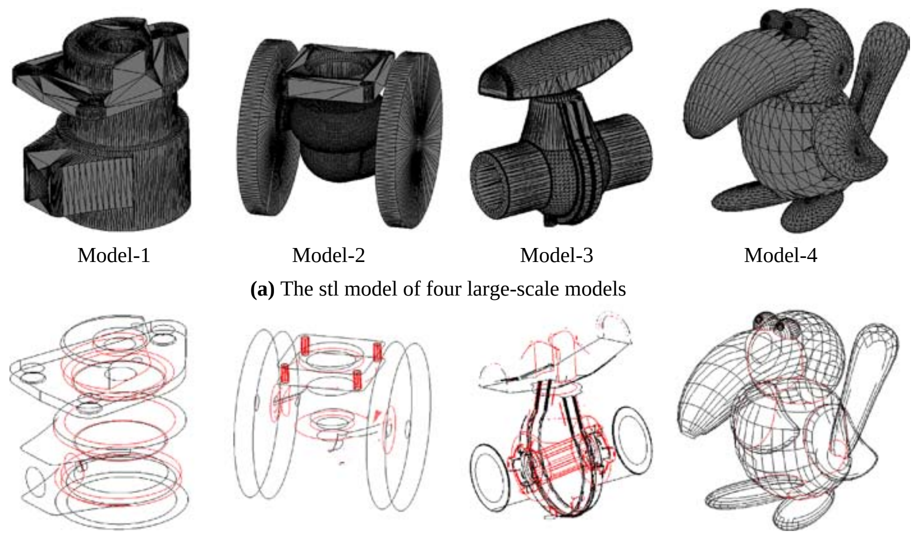
1. *The intersect boundary (as shown in the red plane of* Figure 5). This kind of feature loop passes the inside of the model, so that it can be used to decompose the model.
2. *The surface boundary (the blue plane)*. This kind of feature loop just shows the surface feature of the model. In this paper, we refer to the intersect loops as the feature loops.

4.1.1 Grouping adjacent feature edges

For a feature edge e_i, the adjacent feature edges have been searched. Let e_j be an adjacent feature edge which is not belonged to the same face and has similar dihedral angle or similar perimeter ratio with e_i. The plane $p_{i,j} : ax + by + cz + d = 0$ is defined by e_i, e_j.

If e_i, e_j are all concave, or if each two adjacent faces of e_i, e_j are on different sides of $p_{i,j}$, the plane $p_{i,j}$ is identified as the intersect plane and e_i, e_j are intersect feature edges. In Figure 6, $p_{i,j}$ in (a) and (b) are the intersect planes; others are surface planes (the gray side is the front of a face and the red side is the back).

Figure 4 The curvature estimation of four experimental models



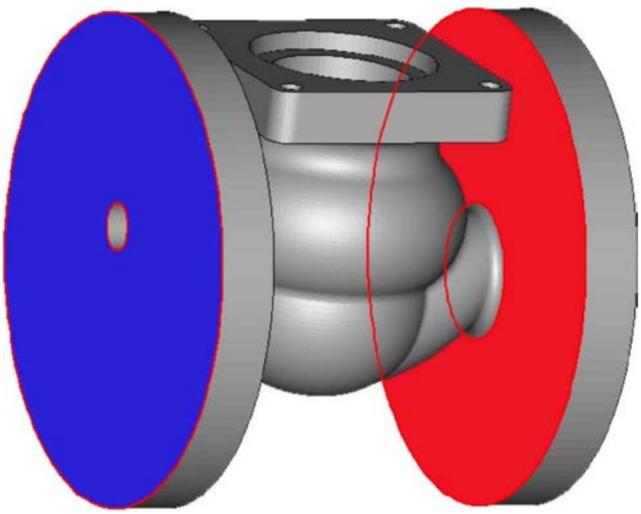
(a) The stl model of four large-scale models

(b) The feature edges of four large-scale models (the red edges are concave)

Table I The partition information of four experimental models

	Model			
Data	**Model-1**	**Model-2**	**Model-3**	**Model-4**
Size (mm) X/Y/Z	589/911/775	723/702/702	427/437/263	402/274/166
Triangle	8206	57574	48362	8052
Point	4093	28777	24180	4028
Feature edge	4101	3647	9398	4347
Threshold 1 – dihedral angle	9.204°	10.002°	10.001°	10.003°
Threshold 2 – perimeter ratio	9.245	5.774	10.520	6.292
Feature loop	28	17	52	116

Figure 5 The different feature loops in a model



Then $p_{i,j}$ is added into the plane set S_1. The edges e_i, e_j are added into the edge set E_1 and connected with the double linked list.

Searching for another feature edge which is similar and adjacent to e_j or e_i. Let e_k be another adjacent feature edge of e_i, if the planes $p_{i,k}$ and $p_{i,j}$ are similar with a tolerance $\delta < \varepsilon$ (ε is the threshold value of the dihedral angle), $p_{i,k}$ will be added into the plane set S_1. e_k will be added to the edge set E_1 and connected to the link. This process is repeated until the link is closed or there are no other similar and adjacent feature edges. The linked edge set E_1 can be marked as one feature loop. The plane P_1, which contains the maximum edges of E_1, will be set as the feature plane of E_1. In other words, the feature plane P_1 is identified as the most covered plane of S_1.

Other adjacent feature edges are grouped by the same way until all similar and adjacent feature edges have been processed. Suppose there are feature edges, this process takes $O(m \log m)$ time complexity.

4.1.2 Grouping isolated feature edges

Some of the isolated feature edges are data noise or surface feature edges which are useless for our approach. But others may be the disconnected feature edges in certain feature planes which need to be grouped for completing feature loops.

Each isolated feature edge has been checked. If an edge is almost in the nearest feature plane with a tolerance $\delta < \varepsilon$, it will be marked the nearest feature plane as the index. This process is repeated until all isolated edges have been processed. Finally, all useful adjacent and isolated feature edges have been filtered and grouped by the feature plane.

4.1.3 Closing incomplete feature loops

The case that requires more attention is the incomplete feature loop. There are three major causes:

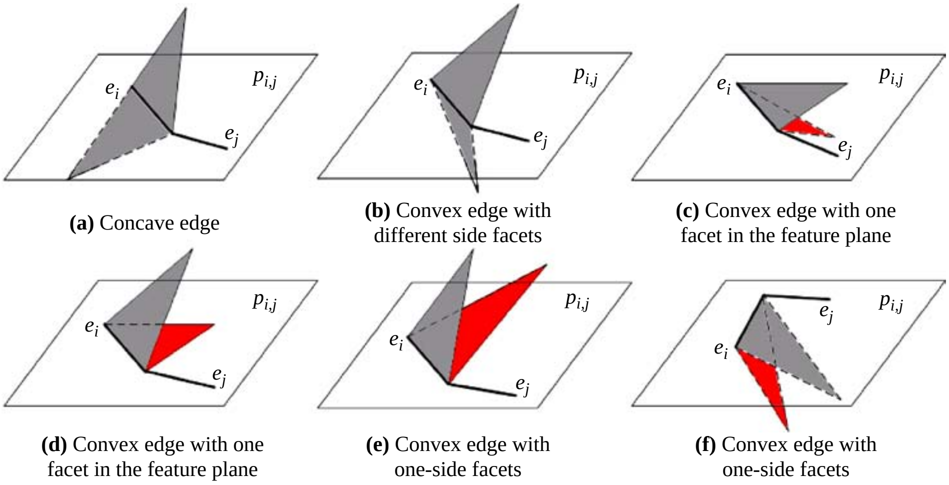
(a) the data noise;
(b) the incomplete feature; and
(c) the hybrid features (as shown in Figure 7).

Since the feature loop is not the final cutting contour, we first close incomplete feature loops to weight them for selecting the best fit loop. In Section 4.3.1, we use the cut plane to complete the cutting contour for partitioning the model.

For the conditions (a) and (b), the incomplete loops and unconnected edges are in the same feature plane. The method is to search for two endpoints which have the same direction and the shortest Euclidean distance, then connect them without adding a new edge:

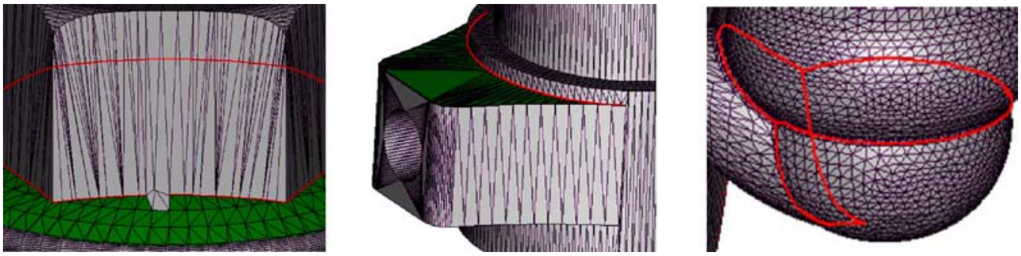
- The single incomplete feature loop can be closed directly by connecting the beginning edge and the end edge (as shown in Figure 8(a), set e_1 as the next adjacent edge of e_n).
- In Figure 8(b), there are several incomplete feature loops and unconnected feature edges in the same feature plane. To close these loops, the feature edges are first directed to distinguish between the inside loop and the outside loop (Bajaj and Dey, 1992). The solid region is on its left side (the arrowhead in Figure 8 shows the direction of the feature edge). The outside feature loop is counterclockwise and the inside one is clockwise. The inside feature loops and outside feature loops have been closed separately. For example, the end edge e_3 is used to find the next adjacent edge. Comparing the Euclidean distance between e_3 and the beginning edges of other feature loops (e_4, e_7) or unconnected feature edges (e_{16}, e_{17}). The edge e_{17} which has the shortest distance has been set as the next adjacent edge of e_3. This process is repeated until there are no end edges whose next adjacent edge is NULL.

Figure 6 The relationship of feature edges and planes



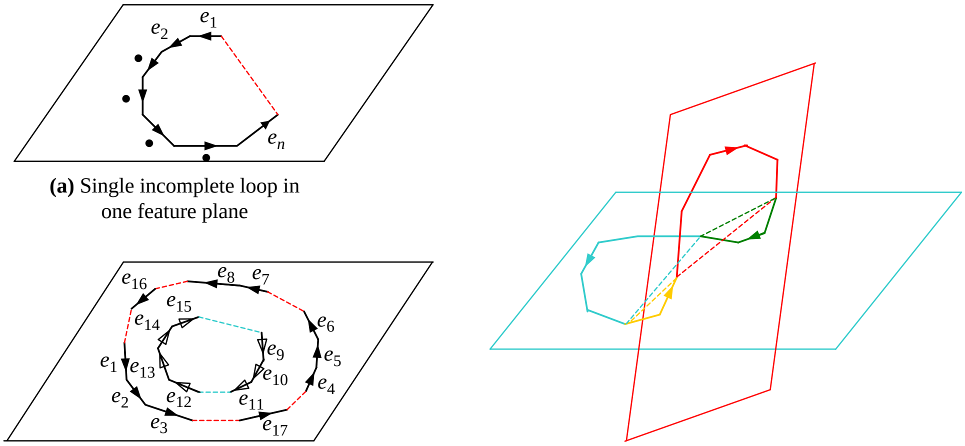
(a) Concave edge
(b) Convex edge with different side facets
(c) Convex edge with one facet in the feature plane
(d) Convex edge with one facet in the feature plane
(e) Convex edge with one-side facets
(f) Convex edge with one-side facets

Figure 7 Three kinds of incomplete feature loops



(a) Data noise **(b)** Incomplete feature **(c)** Hybrid features

Figure 8 Three kinds of incomplete feature loops



(a) Single incomplete loop in one feature plane

(b) Several incomplete loops and unconnected edges in one feature plane

(c) Incomplete loops in different feature planes

For the condition (c), the hybrid features are in different feature planes. The method is to close these loops separately based on the feature plane. In Figure 8(c), the complete loop belongs to four feature planes. Each incomplete loop is closed by the above method, and then four separate feature loops will be obtained.

4.2 Selecting the best fit loop

To efficiently use the RP machine's build-volume, we select the best-fit loop to partition the model hierarchically. For each level of partition, the cutting axis and times have been determined according to the estimation of partition condition. Then the best-fit loops can be selected by comparing the weight of feature loops.

4.2.1 The estimation of partition condition

For a large-scale model M, the partition result can be expressed as:

$$M = \sum_{j=1}^{n} m_j, (m_j \leq m_{\max}) \tag{5}$$

where n is the number of sub-models that are decomposed from the original model. Every sub-model, m_j needs to be smaller than the maximum working size $m_{\max}$. In most cases, the maximum working size is preset by the build size of RP machine and the bounding box of M is easy to calculate:

$$m_{\max} = x_{\max} \times y_{\max} \times z_{\max} \tag{6}$$

$$B(M) = [X_{\min}, X_{\max}] \times [Y_{\min}, Y_{\max}] \times [Z_{\min}, Z_{\max}] \tag{7}$$

Comparing the bounding size and the max size of three axes, the cutting times of each axis can be calculated as:

$$N_X = Int\left(\frac{X_{\max} - X_{\min}}{x_{\max}}\right); \quad N_Y = Int\left(\frac{Y_{\max} - Y_{\min}}{y_{\max}}\right); \quad N_Z = Int\left(\frac{Z_{\max} - Z_{\min}}{z_{\max}}\right) \tag{8}$$

To partition the model hierarchically, one axis with the most cutting times will be selected as the cutting axis. In Figure 1, X-axis is selected as the cutting axis to partition the model ($N_X = 1$, $N_Y = 0$, $N_Z = 0$). The approximate cutting positions can be obtained as:

$$X_j = X_{\min} + j \times \frac{X_{\max} - X_{\min}}{N_X + 1}, \quad (j = 1, 2, \ldots, N_X) \tag{9}$$

Thus, the partition scheme can be determined with the cutting axis, the cutting times and the approximate cutting positions. For the large-scale model with certain feature components, the best-fit loop can be selected according to the partition scheme. For the large-scale model with less important feature, the partition scheme can be used as the guideline of the planar division.

4.2.2 The weight of the feature loop

The weight of the feature loop determines the quality of the feature loop, which can be used to select the best fit loop. As the cutting axis has been determined (Support *X*-axis), our algorithm is to use the center position $X(E_i)$ and the bounding rectangle $BX(E_\mathrm{i})$ to weight the feature loop E_i:

$$X(E_i) = \frac{x_{\min}^{E_i} + x_{\max}^{E_i}}{2} \tag{10}$$

$$BX(E_i) = \left[y_{\min}^{E_i}, y_{\max}^{E_i}\right] \times \left[z_{\min}^{E_i}, z_{\max}^{E_i}\right] \tag{11}$$

where $x_{\min}^{E_i}, x_{\max}^{E_i}, y_{\min}^{E_i}, y_{\max}^{E_i}, z_{\min}^{E_i}, z_{\max}^{E_i}$ are calculated by comparing the feature edges of E_i. As the feature plane P_i and the center position $X(E_i)$ have been marked to the feature loop E_i, two kinds of feature loops can be filtered out before selecting the best fit loop:

1 The feature loops which are similarly parallel to the cutting axis are filtered out.
2 The feature loops which are mostly close to both ends of the model are filtered out.

4.2.3 The selection of the best fit loop

As the cutting times is N_X, N_X of the best fit loops will be selected as follows:

1 Checking the concave feature loops for the best fit loop prior to the convex ones.
2 Comparing the center position $X(E_i)$ and the approximate cutting position X_j, the closest feature loop will be selected as the best fit loop.
3 If some feature loops have same distance to X_j, the feature loop with the maximum bounding rectangle will be selected as the best fit loop.
4 If there is no feasible concave feature loop, the convex feature loops will be checking according to (2) and (3).

For surface decomposition in many applications (e.g. shape representation and simplification), the best fit loop can be directly used to split the surface at the feature edges. For the STL model with certain internal structures, there may be certain inside contours in the best fit loop, or the best fit loop may be the inside contour. So the final cutting contour needs be completed for obtaining the complete STL file of sub-models.

4.3 Partitioning with best fit loops

For the STL model partition, we define the feature plane of the best fit loop as the cut plane which can be used to complete the cutting contour *K* (as shown in Figure 9(b)). Then we divide the model into two open half models at the edges of *K*. The details of the partition algorithm are as follows.

4.3.1 Constructing the cutting contour

Since RP is a layer-by-layer fabrication, the slice contour of the STL model can be obtained with the given slice plane by many efficient methods (Kou and Tan, 2009). For the incomplete best fit loop, the contiguous intersect faces which are connected to the incomplete loops have been retrieved and divided by the cut plane. The new edges can be added and linked to complete the best fit loop.

For the simple model, the cutting contour can be obtained directly from the best fit loop. However, for a large STL model (as shown in Figure 9(a)), there may be several solid regions in the cut plane. Considering inside and outside surfaces of the solid model, if the best fit loop is the outside one, the internal region will be checked for the inside faces to construct the inside contour of *K*; otherwise, the external region will be checked for the outside faces to construct the first-hierarchy outside contour.

In Figure 9(b), the cutting region of the best fit loop has been distinguished for completing the cutting contour. Other remaining regions are simply neglected. All the intersect faces are retrieved and divided by the cut plane. Then the original intersect faces are removed from the face set and the new divided faces are added. The new edges are then linked and added to construct the cutting contour *K* (Figure 9(c)).

4.3.2 Building two face sets

The faces which are adjacent with the edges of the cutting contour *K* are classified to two face sets: F_k^+ and F_k^- (Figure 9(c)). Then, other faces are sorted to these two sets according to the adjacency relation. The edges of *K* are copied to each paired sub-model and reconstructed with only one adjacent face. Finally, the model *M* is divided to two open sub-models (Figure 9(d)):

$$M_\kappa^+ = \{f_i | f_i \in F_\kappa^+\}; \quad M_\kappa^- = \{f_j | f_j \in F_\kappa^-\} \tag{12}$$

where $M_\kappa^+ \cup M_\kappa^- = M$, $M_\kappa^+ \cap M_\kappa^- = K$. The cutting contour of each open model can be filled with triangle faces using the Delaunay triangulation method to achieve the closed sub-models (Shewchuk, 2002).

Each sub-model is recursively partitioned until at least one of the following conditions is met:

- the size of sub-model is smaller than the maximum working size;
- there are no feature loops, so the large sub-models cannot be partitioned by our method; and
- the remaining feature loops do not fit the requirement, so there are no eligible candidates for further partition.

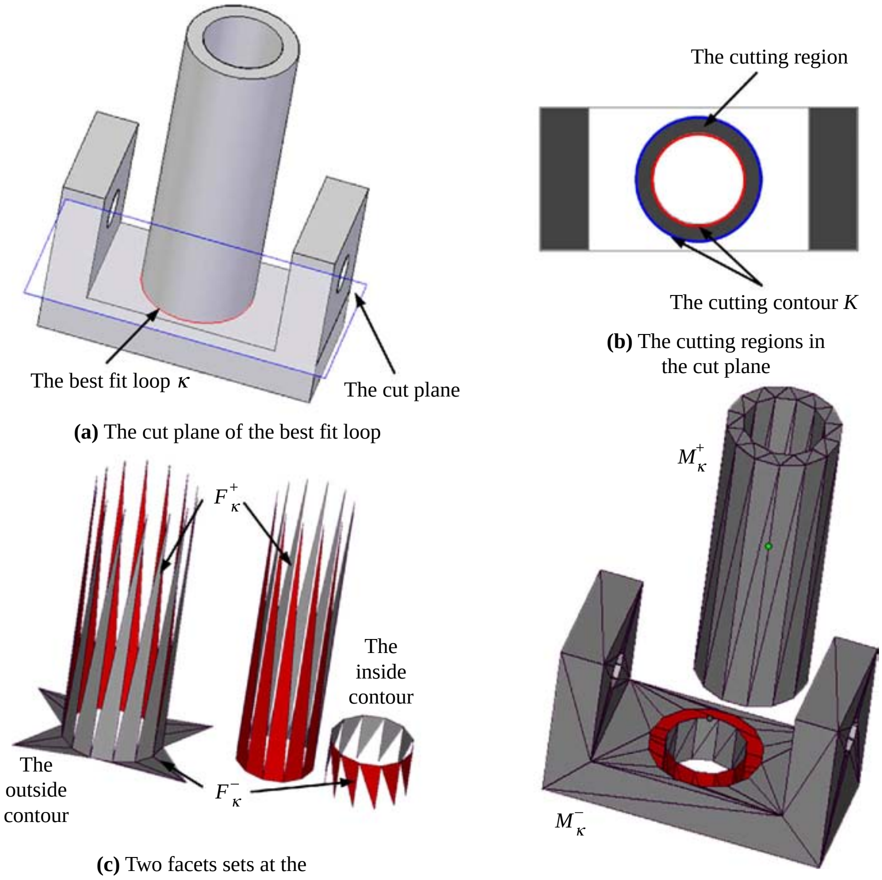
For the conditions (b) and (c), the large sub-models can be directly partitioned by the cut planes decided by the approximate cutting positions and the cutting axis.

Figure 10 shows the hierarchical curvature-based partition of Model-3. The first-level cutting axis is *X*-axis and the first-level cutting times is two (Figure 10(b)). Since the left and the right sub-models are smaller than the maximum working size, they do not need to be partitioned. The second-level cutting axis of the middle sub-model is *Y* axis and the second-level cutting times is one (Figure 10(c)). The upper sub-model of Figure 10(c) is larger than the maximum working size, the cutting axis is *X*-axis and the cutting times is one. But there has no feasible feature loop to be partitioned in *X*-axis, so the curvature-based partition has been stopped. Then this kind of sub-model can be partitioned by the planar division based on the estimation of partition condition.

5. Algorithm – the similar-shaped joint

Instead of directly triangulating the cutting region, it is possible to build the joint structures on the splice surfaces of sub-models to facilitate the assembly. Our approach is to

Figure 9 The solid regions of the best fit cut plane



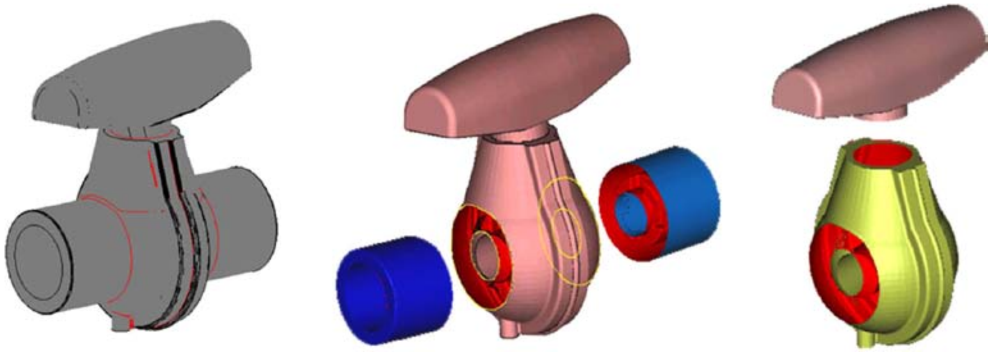
(a) The cut plane of the best fit loop

(b) The cutting regions in the cut plane

(c) Two facets sets at the cutting contour

(d) Two open half models

Figure 10 Hierarchical curvature-based partition of Model-3



(a) The feature loops **(b)** The first level partition **(c)** The second level partition

build the similar-shaped joint on the splice surface according to the cutting contour. The algorithm details are as follows.

5.1 Similar-shaped loop

First, the similar-shaped loop is defined according to the cutting contour. Set $L(o)$ – outside loop; $L(i)$ – inside loop; $L(s)$ – similar-shaped loop.

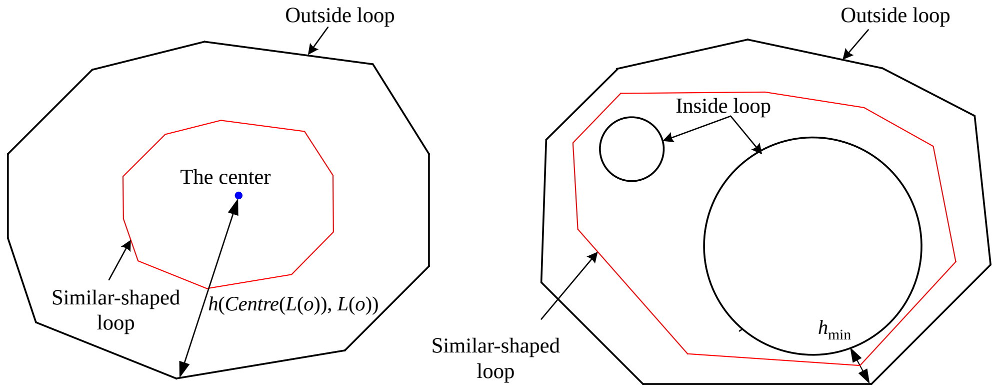
If the cutting contour only contains one outside loop (Figure 11(a)), the distance of $L(o)$ and $L(s)$ is given by:

$$H(L(s), L(o)) = \frac{1}{2} H(Centre(L(o)), L(o)) \tag{13}$$

where the centre of $L(o)$ is:

$$Centre\ x = \frac{\sum_{i=1}^{n} x_i}{n}; \quad Centre\ y = \frac{\sum_{i=1}^{n} y_i}{n}; \quad Centre\ z = \frac{\sum_{i=1}^{n} z_i}{n} \tag{14}$$

Figure 11 The similar-shaped loop



(a) One outside loop

(b) More than two loops

If the cutting contour contains more than two loops (Figure 11(b)), the distance of $L(o)$ and $L(s)$ is given by:

$$H(L(s), L(o)) = \frac{1}{2} H_{\min}(L(i), L(o)) \qquad (15)$$

For each vertex of the outside loop, the nearest vertex of inside loops has been found out. Then the corresponding vertex of the similar-shaped loop is computed by:

$$v_i(s) = \frac{1}{2}(v_i(o), v_{h_{\min}}(i)) \qquad (16)$$

The region of $L(o)$ and $L(s)$ has been triangulated by adding new faces. For the corresponding edges $v_i(s)v_{i+1}(s)$ and $v_i(o)v_{i+1}(o)$, two new faces f_i and f_{i+1} are constructed by linking the corresponding vertices (as shown in Figure 12):

$$\begin{aligned} f_i &= (v_i(o), v_{i+1}(s), v_i(s)); \\ f_{i+1} &= (v_i(o), v_{i+1}(o), v_{i+1}(s)) \end{aligned} \qquad (17)$$

5.2 The depth of similar-shaped joints

To ensure the joints are entirely inside of sub-models, the depth of joints must be restricted. If the joint is too long, it may pass through the solid and destroy the surface of the model; or if the joint is too short, it may be not able to guarantee the assembly of sub-models.

With the similar-shaped loop and the cut plane, the maximum depth of the similar-shaped joint is firstly computed. The projections of the similar-shaped loop to both sub-models have been made along the vertical direction of the cut plane (as shown in Figure 13). For each vertex v_i, the opposite two vertices v_i^+ and v_i^- are obtained. Comparing the projection distance, the shortest distance of $v_i v_i^+$ has been chosen as the maximum joint depth of M_κ^+, and the shortest distance of $v_i v_i^-$ has been chosen as the maximum joint depth of M_κ^-. If the distance between the vertex P and the cut plane Pp is shorter than the shortest distance of $v_i v_i^+$, the maximum depth of M_κ^+ will be restricted by Pp. If there are several faces in the projected loop (such as face, α in M_κ^-), the condition will be more complex. Sun *et al.* (2004) described the details of deciding the maximum joint depth.

Figure 12 Triangulation of $L(o)$ and $L(s)$

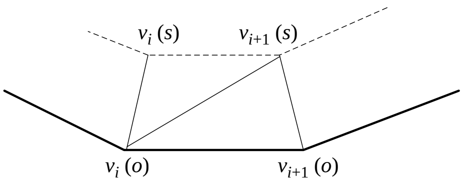
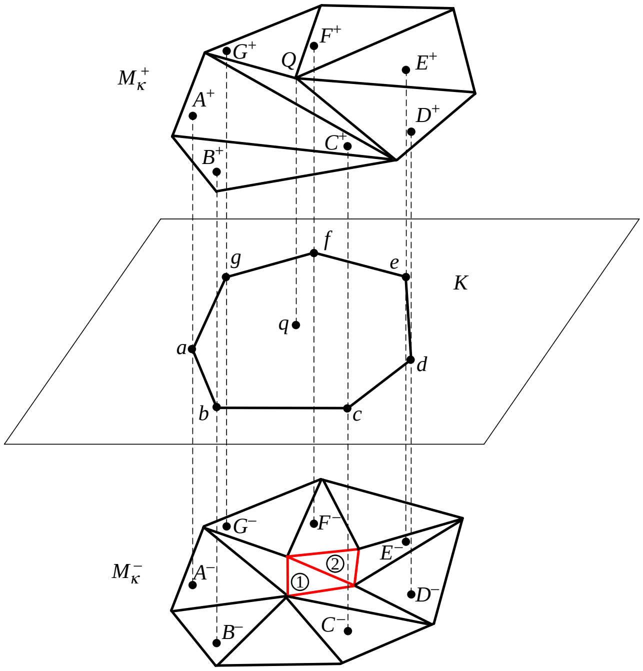
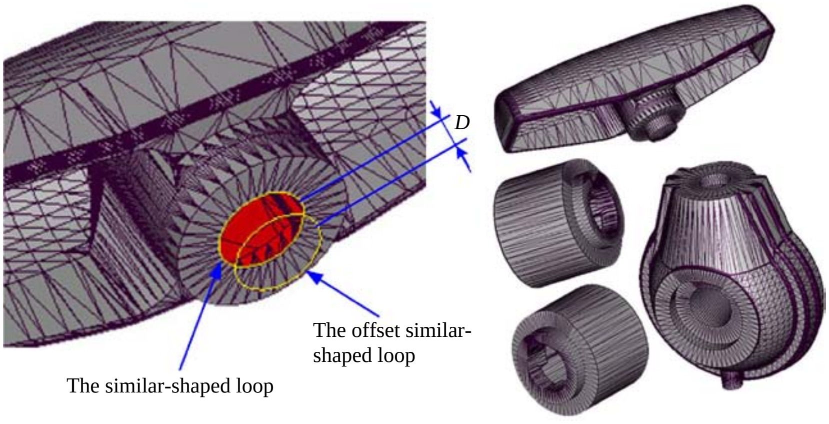


Figure 13 The projection of the similar-shaped loop



According to the maximum joint depth, the depth of the similar-shaped joint D can be selected from 1/5 to 1/10 of the maximum joint depth of subtracted sub-model. As shown in Figure 14(a), the similar-shaped loop is copied and offset with H (Qu and Stucker, 2003). Then the surface of the similar-shaped joint also can be triangulated by the Delaunay triangulation method. After the new faces have been added to one sub-model, all of them are reversed and added to the paired sub-model. Thus, the sub-models with the similar-shaped joint have been achieved (Figure 14(b) shows the second level sub-models of Model-3 with the similar-shaped joint).

Figure 14 The similar-shaped joints of the second-level sub-models of Model-3



(a) The construction of the similar-shaped joint

(b) The second level sub-models with the similar-shaped joint

6. Case study

In this section, we describe the partition result of the four experimental models (Figure 4) to study the performance of the proposed algorithm. The proposed algorithm has been successfully implemented in 3DEPS Partition – an independent partitioning module of 3DEPS system developed by the present authors in China University of Mining and Technology. The prototype of Model 4 is fabricated using the ZPrinter-450 (www.zcorp.com/en/Products/3D-Printers/ZPrinter-450/spage.aspx).

In Figure 4, the four experimental models have different size, complexity and resolution (as shown in Table I). The thresholds of dihedral angle and perimeter ratio are accordingly different. As the build size of ZPrinter-450 is $203 \times 254 \times 203\,\text{mm}$, the max working size is set $m_{\max} = 200 \times 250 \times 200\,\text{mm}$. First, the program translates and rotates the longest direction of model to the Y-axis. Then the hierarchical partition scheme has been determined according to the estimation of partition condition.

Comparing the time complexity of the curvature estimation, the vertex- and the face-based methods are about equal – $O(n)$; our algorithm is a little more – $O(3n/2)$. However, the curvature estimation of edges is easier and more accurate than that of faces and vertices. The curvature of a face is decided by three neighboring faces; the curvature of a vertex is decided by more than three neighboring vertices or three adjacent faces; the curvature of an edge is just decided by two adjacent faces. Moreover, the face-based method uses the multiple merging of faces to decompose the surface regions. The vertex-based method uses the multiple clustering of vertices to divide the vertex set of the model. If the data size of the model is huge, the calculated amounts of these two methods will obviously rise. By comparison, the data size of feature loops is apparently smaller (as shown in Table I), and these feature loops can be directly used to distinguish the meaningful components. So the efficiency of our algorithm is higher for the large complex model.

Since Model-1, 2 and 3 are the mechanical parts, their structure are relatively simple and easy to distinguish. Based on the estimation of partition condition, they have been efficiently partitioned with best fit loops. Comparing the planar division, the curvature-based partition makes the best of the original edges to decompose the model. This method can keep the original data information and reduce face dividing, so the precision and efficiency of this method is naturally higher. Certainly, the planar division is useful to partition the large sub-models without feasible feature loops.

Model-4 is a crow model, which is more complex than mechanical parts. Using the curvature-based partitioning, Models-4 has been partitioned into four sub-models: “break”, “head”, “body” and “tail” (as shown in Figure 15(a)). The size of each sub-model is fit for the working size of ZPrinter-450. Meanwhile, we try to partition the “body” into a set of simpler sub-models to improve the processing efficiency of the RP machine. As shown in Figure 15(b), the wings and feet of the crow model have been separated from the body for simplifying the structure of the original body. The wings and feet can be fabricated in the same platform with the tail.

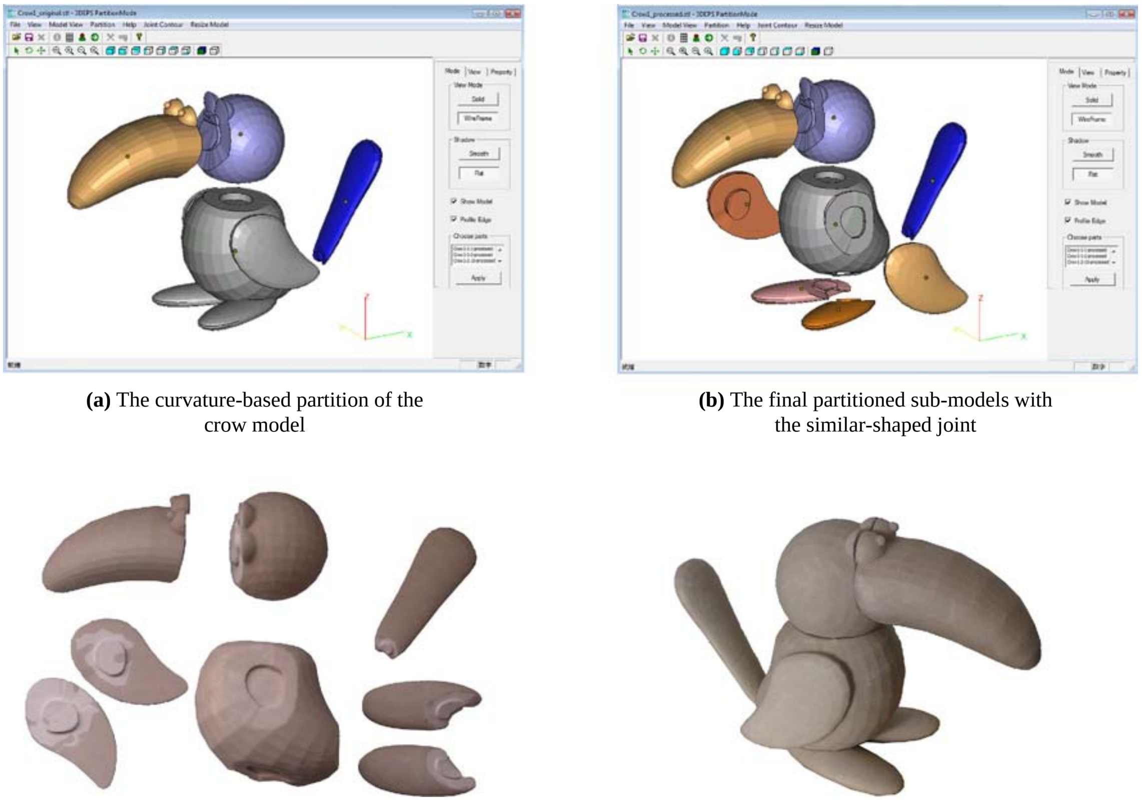
Considering the post-processing and the assembly ration, the practical fitting allowance should be calculated. The similar-shaped joint of the paired sub-models need to be rescaled before the fabrication of sub-models (Delebecque *et al.*, 2008). Each sub-model is saved as a separate STL file, thus those sub-models can be fabricated separately or simultaneously (Figure 15(c)). The final assembled model is as shown in Figure 15(d). The experimental result demonstrates that the sub-models can be assembled accurately and firmly.

The limitations of our algorithm are as follows:

- The curvature-based partition is useless for the large models with less important feature. The upper sub-model of Model 3 is larger than the build size, which need to be partitioned by the planar division.
- Our approach is using the feature plane as the cut plane to construct the incomplete cutting contour.

For the hybrid feature, our approach may separate the non-planar feature loops and construct the cutting contour by the feature plane. In Figure 15, the “eyes” of the crow model are broken by the feature loop of the “break” and the “head”.

Figure 15 Prototypes fabricated with ZPrinter-450



(a) The curvature-based partition of the crow model

(b) The final partitioned sub-models with the similar-shaped joint

(c) The rapid prototype of sub-models

(d) The final assembled model

7. Conclusions and future work

We have presented a curvature-based algorithm for efficiently partitioning the large-scale STL models. The improvements of the proposed partitioning are as follows:

- The processing data size is smaller than that of the vertex- and the face-based methods. After the curvature estimation of edges, the feature edges can be efficiently extracted and grouped for constructing the feature loops, which will be processed for partitioning the model. The data size of feature loops is apparently smaller than that of the whole model.
- The precision and efficiency of the large-scale model partition have been improved. Since the feature loops are composed of the original edges of the model, this partitioning does not need to divide the original faces to decompose the model.
- The feature loops also can be used to decompose the complex model into simpler sub-models for improving the efficiency of sub-models' fabrication. For example, the fused deposition modeling system can reduce support structures with reasonable partitioning of complex models. Laminated object modeling system can make the most of each layer material by fabricating sub-models synchronously in one platform.
- The sub-models with the similar-shaped joint can be precisely assembled to be the original model. The similar-shaped joint has been efficiently built on the splice surface according to the cutting contour.

As the STL file format is the standard data interface between CAD software and the RP machines, the presented algorithm can be applied to all kinds of RP systems. This curvature-based partitioning also can be applied to other manufacturing technologies to widen the machining capacity.

Several enhancements can be added to our algorithm:

- The planar partition of the large simple model can be improved by using the original edges as the cutting contour.
- The large simple model can be partitioned into a set of the same parts (such as the standard cell model) by the matrix partition or the cellular partition.
- The hybrid features can be appropriately partitioned by the non-planar feature loops, which are constructed in the curved cut plane.
- The decomposition of the complex model can be used to improve the fabrication efficiency of RP machines.

We believe that this curvature-based partitioning of the large-scale model will expand the application of RP technology.

References

Bajaj, C.L. and Dey, T.K. (1992), "Convex decomposition of polyhedral and robustness", *Siam Journal on Computing*, Vol. 21 No. 2, pp. 339-64.

Biederman, I. (1987), "Recognition-by-component: a theory of human image understanding", *Psychological Review*, Vol. 94, pp. 115-47.

Chazelle, B. and Palios, L. (1992), "Decomposing the boundary of a nonconvex polyhedron", *SWAT'92 Proceedings of the Third Scandinavian Workshop on Algorithm Theory*, pp. 364-75.

Chen, Y. (2007), "Robust and accurate Boolean operations on polygonal models", paper presented at ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conferences, DETC2007-35731, Las Vegas, NV.

Chen, Y., Wang, C. and Huang, S. (2004), "An arithmetic of ladderlike dividing based on STL file format", *Journal of Huazhong University of Science and Technology* (*Nature Science*), Vol. 32 No. 8, pp. 61-3.

Delebecque, B., Houtmann, Y., Lauvaux, G. and Barlier, C. (2008), "Automated generation of assembly features in layered manufacturing", *Rapid Prototyping Journal*, Vol. 14 No. 4, pp. 234-45.

Dong, F., Zhang, R. and Liu, Y. (2008), "A hybrid approach to surface segmentation of sparse triangle meshes", *International Conference on Computer Science and Information Technology*, pp. 669-74.

Garland, M., Willmott, A. and Heckbert, P. (2001), "Hierarchical face clustering on polygonal surfaces", *Proceedings of ACM Symposium on Interactive 3D Graphics, Chapel Hill, NC, USA*, pp. 45-58.

Kanungo, T., Mount, D.M., Netanyahu, N.S., Piatko, C.D., Silverman, R. and Wu, A.Y. (2002), "An efficient k-means clustering algorithm: analysis and implementation", *IEEE Transactions on Computer Vision and Pattern Recognition*, Vol. 24 No. 7, pp. 881-92.

Katz, S. and Tal, A. (2003), "Hierarchical mesh decomposition using fuzzy clustering and cuts", *ACM Trans. Graph*, Vol. 22 No. 3, pp. 954-61.

Kou, X.Y. and Tan, S.T. (2009), "Robust and efficient algorithms for rapid prototyping of heterogeneous objects", *Rapid Prototyping Journal*, Vol. 15 No. 1, pp. 5-18.

Li, X., Toon, T., Tan, T. and Huang, Z. (2001), "Decomposing polygon meshes for interactive applications", *Proceedings of the 2001 Symposium on Interactive 3D Graphics, Chapel Hill, NC, USA*, pp. 35-42.

Lien, J.-M. and Amato, N.M. (2004), "Approximate convex decomposition of polygons", *Proc. 20th Annual ACM Symp. Computat. Geom.* (*SoCG*), *Polytechnic University in Brooklyn, New York, NY*, pp. 17-26.

Mangan, A. and Whitaker, R. (1999), "Partitioning 3D surface meshes using watershed segmentation", *IEEE Transactions on Computer Vision and Pattern Recognition*, Vol. 5 No. 4, pp. 70-7.

Pham, D.T. and Dimov, S.S. (2001), *Rapid Manufacturing: The Technologies and Applications of Rapid Prototyping and Rapid Tooling*, Springer, New York, NY.

Qu, X. and Stucker, B. (2003), "A 3D surface offset method for STL-format models", *Rapid Prototyping Journal*, Vol. 9 No. 3, pp. 133-41.

Razdan, A. and Bae, M. (2003), "A hybrid approach to feature segmentation of triangle meshes", *Computer-Aided Design*, Vol. 35 No. 9, pp. 783-9.

Shewchuk, J.R. (2002), "Delaunay refinement algorithms for triangular mesh generation", *Computational Geometry: Theory and Applications*, Vol. 22 Nos 1-3, pp. 21-74.

Shlafman, S., Tal, A. and Katz, S. (2002), "Metamorphosis of polyhedral surfaces using decomposition", *Eurographics*, Vol. 2002, pp. 219-28.

Srikantiah, R. (2000), "Multi-scale surface segmentation and description", MS thesis, Ohio State University, Columbus, OH.

Sun, K., Ceng, Q. and Sun, W. (2004), "Research of algorithm of direct creation of STL model with assembling location system", *Machinery*, Vol. 42 No. 3, pp. 24-6.

Sun, K., Ma, L., Fang, L. and Lu, B. (2003), "Research on the algorithm for sectional fabrication of rapid prototype based on STL data", *Machinery*, Vol. 41 No. 9, pp. 11-13.

Yang, S., Shu, S. and Zhu, S. (2008), "Feature line extraction from triangular meshes based on STL files", *Computer Engineering and Application*, Vol. 44 No. 4, pp. 14-19.

Further reading

Chazelle, B., Dobkin, D., Shourhura, N. and Tal, A. (1997), "Strategies for polyhedral surface decomposition: an experimental study", *Computational Geometry: Theory and Applications*, Vol. 7 Nos 4/5, pp. 327-42.

Lien, J.-M. and Amato, N.M. (2008), "Approximate convex decomposition of polyhedra and its applications", *Computer Aided Geometric Design*, Vol. 25 No. 7, pp. 503-22.

Sun, K., Ceng, Q. and Sun, W. (2004), "Research of algorithm of direct creation of STL model with assembling location system", *Machinery*, Vol. 42 No. 3, pp. 24-6.

Corresponding author

Jingbin Hao can be contacted at: jingbinhao@cumt.edu.cn