ELSEVIER

Computer-Aided Design 33 (2001) 925–934

COMPUTER-AIDED DESIGN

www.elsevier.com/locate/cad

Model simplification for meshing using face clustering

A. Sheffer*

Computational Science and Engineering Program, University of Illinois at Urbana-Champaign, 3244 DCL, 1304 W. Springfield av., Urbana, IL 61801, USA

Received 18 April 2000; revised 7 August 2000; accepted 11 August 2000

Abstract

In this work a new method for CAD model simplification is presented. The method is especially suited for simplifications performed in preparation for meshing the models. It uses a local analysis approach based on a clustering procedure.

The method provides a generic approach for model simplification. It is efficient and robust. It works on non-manifold models, free-form and linear faces, and models with large curved regions. It provides symmetric partitioning of details like blends and fillets, resulting in more symmetric simplified models.

Keywords: Simplification; Mesh; CAD models

1. Introduction

Real-life CAD models are often highly detailed with a large number of faces representing each feature. Such detailed descriptions are unnecessary for many engineering applications, including mesh generation and analysis. The complexity of the CAD description severely complicates the meshing procedure, often hindering the use of many available meshing techniques. It also requires a significant increase in the number of mesh elements created to represent the model, increasing the analysis time dramatically. Hence model simplification is essential, both to improve the quality of the generated meshes and to decrease significantly the number of mesh elements generated.

For analysis purposes it is sometimes possible to eliminate minor features after the mesh had been created, thus reducing the analysis time and improving mesh quality. However the existing algorithms for mesh-based simplification like Ref. [1] are limited to triangular or tetrahedral meshes. Moreover, since the original mesh reflects the detailed model, the meshing time and complexity remain high.

The need for model simplification is not limited to mesh generation. Complex models need to be simplified for graphical manipulation and display [2–4]. Simplification is an essential step in CAD model reconstruction from point sets [5,6]. Most of the algorithms suggested in this context, use polygonal models for both input and output and hence are not suitable for simplification of real-life CAD models which usually contain free-form surfaces.

One approach to simplifying complex CAD models is by explicit suppression of minor details. This approach was suggested in Refs. [7,8]. The recognition of details to suppress can be done either by feature recognition techniques [8] or by analysis of the medial axis of the model [7]. The main disadvantages of this approach are: inability to detect slivers and small faces, which do not form prominent features and the need to explicitly define the possible detail shapes.

An alternative approach based on local analysis of the model faces was used in Refs. [9,10]. The methods suggested in both papers are based on clustering of groups of faces into regions. Each region is then represented as a single model face. The clustering approach is commonly used in simplification of polygonal models for graphics purposes, and in those two works it is extended to handle models with non-linear faces. This approach is significantly more efficient and robust then the feature-based methods. It avoids the need to define specific detection and treatment strategies for each type of feature. It also handles slivers and other minor details, which do not form explicit features. While the approaches presented in Refs. [9,10] appear to be promising, each of the proposed algorithms has significant weaknesses.

The algorithm in Ref. [9] uses angle and distance criteria to cluster planar facets approximating the model faces into regions. Then, based on the resulting clusters analysis, the

* Tel.: +1-217-244-0360; fax: +1-217-333-1910.
E-mail address: sheffa@uiuc.edu (A. Sheffer).

0010-4485/01/$ - see front matter
PII: S0010-4485(00)00116-0

model faces are merged or collapsed, to achieve topological simplification. The algorithm uses virtual topology operators [11] to perform the merge and collapse operations. The method is based solely on computations of distances and angles between planes and/or planar facets and therefore is very efficient. The use of the collapse operator allows symmetric partitioning of such typical features as blends and fillets, resulting in more symmetric simplified models. Experience shows that preserving symmetric properties provides more accurate analysis results. The method as presented has three main disadvantages: the order in which the facets are merged is random, hence order changes can result in differences in the final simplified models; the use of approximating planes does not allow clustering of faces into highly non-planar regions which sometimes are desirable; and finally it requires conformity between facets on adjacent faces.

The clustering method in Ref. [10] starts from a state in which each model face is viewed as a cluster. Then it repeatedly merges pairs of adjacent clusters into new clusters. The order of merging is determined by weights on the links between clusters. The weights are computed based on a combination of *geometric indices* (measurements). After the clustering terminates, each region is meshed using a projected plane approximation. The method provides a systematic approach to clustering, avoiding the randomness of the previous approach and introducing several important geometric indices, which tend to improve the shape of the generated regions. Its major drawbacks compared to the method in Ref. [9] are higher complexity cost resulting from the use of analytical measurements of region curvature, such as Gauss map and normal vector variance and lack of symmetry in the simplified models, since the method has no equivalent to the collapse feature in Ref. [9]. Both methods do not address clustering of faces into smooth but non-planar regions such as cylinders.

1.1. Contribution

In this paper, we present a new algorithm for model simplification for meshing. It uses a local analysis similar to [9,10], but overcomes the disadvantages of the two works.

Similarly to Ref. [10], the algorithm starts from the state in which each model face is viewed as a cluster. Then it repeatedly merges pairs of adjacent clusters into new clusters. The choice of which pair to merge is based on weights assigned to links between clusters. The weights are computed based on a combination of geometric indices.

The majority of the indices and the way they are computed and used are new. The indices are specifically defined to handle such issues as detail suppression over non-planar regions, correct treatment of slivers and non-manifold topologies.

After the initial clustering is finished, a "collapsibility" check procedure is applied to detect faces which can be symmetrically divided between several clusters. The faces are then divided accordingly. This procedure provides more symmetric simplified models.

The algorithm uses the virtual topology merge and collapse operators [11] to modify the model. The virtual topology model representation separates the model geometry and connectivity information and allows topological (connectivity) entities to correspond to a part or a set of geometric entities (e.g. a face can correspond to a part of a model surface or to a set of adjacent surfaces). The merge operator replaces a set of adjacent model faces by a single face removing the shared edges. The new face corresponds to the set of surfaces of the original faces. This operator is used in the algorithm to merge faces belonging to a single cluster. The collapse operator is described in detail in Section 4.1. It is used to divide faces between several neighbor clusters.

The main advantages of the new algorithm are: (1) it provides a generic approach for suppression of minor details and model simplification; (2) it is very efficient and robust, due to the local analysis approach and the use of easy to compute geometric indices; (3) it can simplify models with large curved regions (like cylindrical or spherical parts) by allowing the construction of smooth non-planar regions; (4) it provides symmetrical partitioning of such typical features as blends and filets, creating more symmetrical simplified models, by using the collapse procedure, improving the accuracy of the analysis; (5) the model geometry is not changed and the mesh deviation from the original model is bounded by the mesh element size, thanks to the use of virtual topology operators; (6) it has no restrictions on geometric definition of the faces, and can handle any analytic surfaces as well as faceted or virtual topology [11] descriptions; and (7) it can be applied to any set of faces, including faces not forming a volume and non-manifold topologies.

The paper is organized as follows. Section 2 describes the clustering algorithm. The geometric indices used for merging the regions are described in Section 3. In Section 4 the algorithm for detecting collapsible faces and performing the actual collapse is described. Section 5 provides some implementation details and several examples of applying the simplification procedure. Section 6 summarizes the paper, discusses the advantages and drawbacks of the algorithm and suggests topics for future work.

2. The clustering algorithm

2.1. Adjacency graph

The first stage of the proposed simplification algorithm is a clustering of sets of faces into regions based on compatibility criteria.

The clusters connectivity is mapped to an adjacency graph to represent the face-clustering procedure. Each

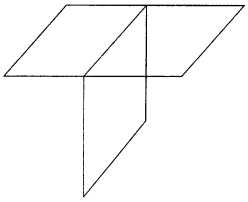


Fig. 1. Example of non-manifold face topology.

cluster is mapped to a graph node, and the adjacency of a pair of clusters is mapped to a graph arc between the respective nodes. Each face is defined as a cluster node in the initial graph, and an arc is created between each pair of adjacent faces. A merging of two clusters corresponds to an arc contraction operation in this graph.

We introduce the notion of *uncontractable* arcs to handle non-manifold topologies and adjacent faces which the user does not desire to be merged. When non-manifold topologies occur, i.e. when more than two faces share a common edge (Fig. 1), none of those faces can be merged together without violating the model connectivity. The adjacency arcs between each such pair are marked as *uncontractable*, and the method for handling these arcs, described below, prevents merging the faces.

Each arc is assigned a weight that corresponds to the improvement in the shape properties of the merged cluster versus the two original clusters. The weight computation is explained in the next section.

A contraction of an arc (a, b) (between the nodes a and b) is done as following (Fig. 2).

1. Remove (a, b).
2. Merge a and b into a new node ab.
3. Reassign all arcs incident on a or on b to the new node.
4. If an arc (a, c) (or (b, c)) was marked as *uncontractable*, then the arc (ab, c) is marked so as well.
5. If a and b had arcs to the same node n ((a, n) and (b, n)) unite the generated duplicate arcs (ab, n). If (a, n) or (b, n) was marked as *uncontractable*, so is the united arc (ab, n).

2.2. Graph contraction algorithm

The clustering procedure is a sequence of arc contractions performed as following:

1. Create the initial adjacency graph, with each face viewed as a cluster and an arc between each pair of adjacent faces.
2. Assign weights for each arc.
3. Place the arcs in a priority queue.
4. If the queue is empty, terminate the algorithm.
5. Remove the arc with highest weight from the queue.
6. If the weight is negative, terminate the algorithm (no more contractable arcs exist).
7. Contract the arc as described above. Update the priority queue by uniting duplicate arcs if necessary.
8. Recompute the weights for the arcs attached to the new node and update the queue accordingly.
9. Jump to step 4.

The algorithm can be applied to the whole model, or to a subset of the model faces. It can be repeated on different subsets or on the same set of faces with different geometrical parameters for the weight computation.

3. Merging conditions

3.1. Clustering considerations

The purpose of CAD model simplification in the context of mesh generation is to create a new model with the following properties:

- it is easier to mesh than the original model;
- it can be meshed with significantly less mesh elements, given a prescribed element size;
- its mesh is a close approximation of the original model;
- its mesh is likely to have better quality then a mesh of the original model.

To the author's knowledge, there are no exact measurements quantifying those properties, and heuristics based on experience are used when estimating a model's suitability for meshing. The following considerations are usually taken into account when the simplification is done manually.

- Boundary preservation — clear edge boundaries have to be maintained at sharp corners.
- Region size — the local feature size at any point within a region should be large relative to the desired element size. The local mesh size is bounded by the distance between opposite region boundaries, and hence, this distance should be large enough.
- Region smoothness — a region has to be smooth and its curvature must change gradually.

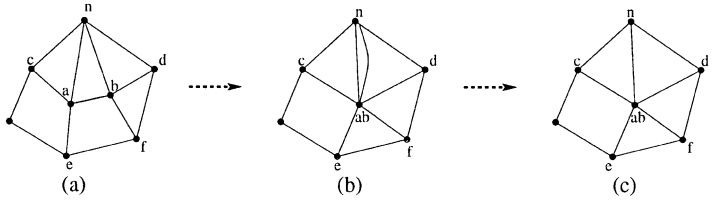


Fig. 2. Stages of arc contraction. (a) to (b) Steps 1–3. Joining the cluster nodes and reassigning the node arcs. (b) to (c) Step 5. Uniting duplicate arcs.

- Simple region boundary shape — the region boundary should be simple and smooth. Sharp corners and complicated boundaries tend to reduce local mesh quality.

We define below a set of geometric indices, which attempt to quantify the above qualities. The geometric indices $G_{a\to b}$ are defined with respect to a pair of clusters (a and b), and measure the *attraction* of cluster a to cluster b. The attraction represents the improvement in the shape properties of the model after merging the clusters. The attraction is *directional*, i.e. $G_{a\to b} \neq G_{b\to a}$. This notion of *directional* attraction becomes especially important when face collapse, rather then face merger, is considered (see Section 4). For some indices, the attraction value is *mutual*, i.e. $G_{a\to b} = G_{b\to a}$, in such cases the notation $G_{a,b}$ is used.

The index values are defined in $\{[0\ldots1], -1\}$, in order to compare them using a common scale. Higher value indicates bigger attraction. -1 indicates that based on this index the clusters are *repulsed*.

The different indices are described in Section 3.2. Section 3.3 explains how the indices are combined to assign weights to the arcs between the graph nodes representing the clusters.

3.2. Indices

3.2.1. Uncontractable arcs

This index U is used to detect clusters which cannot be merged because the boundary edge between them is *uncontractable*. This happens when an edge is non-manifold, i.e. when it is shared by more than two faces, or when a user defined an edge as *uncontractable*. The arc index U is *mutual* and is 0 or -1 according to whether it can be contracted or not.

3.2.2. Boundary preservation

To create good surface and volume meshes, model edges at sharp boundaries between faces must be retained. Merging faces with a sharp corner between them can lead to low mesh quality near the corner, and sometimes even cause the surface meshing algorithms to fail. The index for the edge angle between clusters is *mutual*, and is defined as:

$$E_{a,b} = \begin{cases} -1 & \text{if } \exists \alpha_i < E_{\min} \\ \left(\frac{1}{K}\sum_i^K \alpha_i - E_{\min}\right)/(\pi - E_{\min}) & \text{otherwise} \end{cases} \tag{1}$$

where K is the number of shared edges between the clusters, and α_i the maximal dihedral angle between the clusters along a shared edge i. (Often this value is unavailable within the CAD framework, in such case the angle at the midpoint of the edge is used instead). $E_{\min}$ is the minimal angle allowed, as set by the user. $E_{a,b}$ is visualized in Fig. 3(a).

3.2.3. Region size

To obtain a good quality mesh with a small number of elements, the local feature size at any point within the region should be large enough compared to the element size. To measure the effect of a merge operation on the local feature size, we define a *relative area* index, which is based on region area and perimeter length. The index is *directed*, with a smaller region attracted to the larger one. The index for relative area is defined for cluster a relative to b as:

$$A_{a\to b} = \begin{cases} -1 & \text{if } \dfrac{\text{Area(a)}}{l} > A_{\max} \\ \left(A_{\max} - \dfrac{\text{Area(a)}}{l}\right)/A_{\max} & \text{otherwise} \end{cases} \tag{2}$$

where l is the length of the shared boundary between a and b (if l/perimeter(a) $> 1/2$ we use $l = $ Perimeter(a)/2). $A_{\max}$ is the maximal relative area of regions that are permitted to be attracted to other regions. $A_{a\to b}$ is visualized in Fig. 3(b).

3.2.4. Boundary shape

The simpler the shape of a region, the easier it is to mesh with high-quality elements. Hence, when joining regions, it is better to join regions with longer shared boundaries and to join clusters when the contact angle is obtuse.

Two indices are defined to measure those conditions. The index R for shared boundary ratio is *directed* and defined for cluster a relative to cluster b as:

$$R_{a\to b} = \begin{cases} -1 & \text{if } \dfrac{l}{\text{Perimeter(a)}} < R_{\min} \\ \left(\dfrac{l}{\text{Perimeter(a)}} - R_{\min}\right)/(1 - R_{\min}) & \text{otherwise} \end{cases} \tag{3}$$

where l is the length of the shared boundary. $R_{\min}$ the minimum permissible ratio of the length of the shared boundary between regions a and b to the total perimeter of region a. Usually, $R_{\min}$ is 0 or very small. Fig. 4(a) visualizes R for different cluster relations.

A second index is used to avoid the creation of sharp concave corners at the contact vertices between the clusters. The index, $C_{a,b}$, is *mutual* and is defined as:

$$\Delta = \frac{1}{N}\sum_i^N \frac{(\phi_i - \alpha_i) + (\phi_i - \beta_i)}{2}$$

$$C_{a,b} = \begin{cases} -1 & \text{if } \Delta < \Delta_{\min} \\ 0 & \text{if } \Delta_{\min} \le \Delta < 0 \\ \dfrac{\Delta}{2\pi} & \text{otherwise} \end{cases} \tag{4}$$

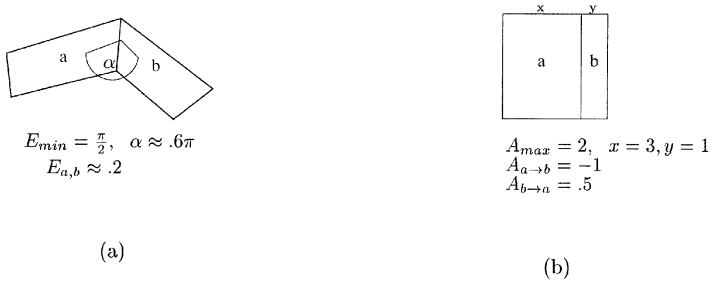


Fig. 3. Geometric indices: (a) boundary angle sharpness; (b) relative region size.

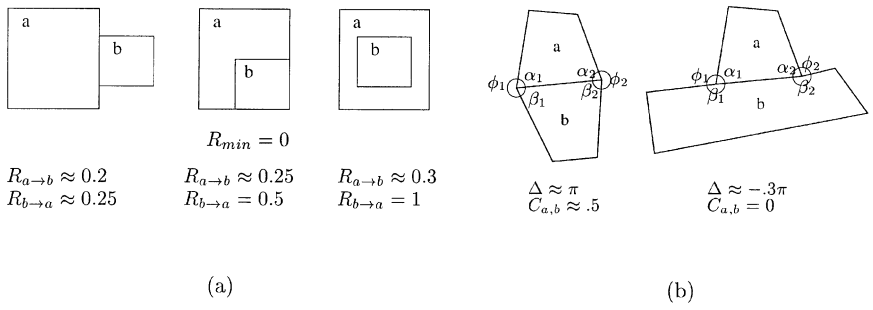


Fig. 4. Boundary shape measures: (a) shared boundary ratio; (b) contact angle.

where N is the number of end points of the discrete segments in the shared boundary (usually 2), α_i and β_i are interior angles of a and b, respectively, at the end point, and ϕ_i is the exterior angle of the joint cluster (see Fig. 4(b)). Attraction between clusters ceases when joining the clusters generates sharper angles than in the original clusters. $\Delta_{\min}$ is the bound on the sharpest angle to be created. Usually, the angle is not bounded.

3.2.5. Region curvature change

It is important to maintain the compatibility of curvature between the two clusters in a merge operation. The specific compatibility requirements depend on the application. Since the merge operation is virtual in our simplification procedure, i.e. only the model connectivity is changed, the merge does not change the model geometry. Hence, the mesh deviation from the original model remains bounded by the mesh element size, and no accuracy is lost during meshing. For most tetrahedral schemes, if there is a smooth boundary between two clusters (edge angle index $E \approx 1$), then the clusters can be merged with no negative effect on the mesh quality (Fig. 5(c)). However, for hexahedral meshing, face orthogonality has to be maintained to get a high-quality mesh (Fig. 5(e)), since most existing hexahedral algorithms construct semi-structured meshes [12]. Hence, the variation of curvature within a region must be considered. In previous works [9,10], the curvature indices were based on region planarity, so that only clusters with bounded curvature are formed. The difference between a planarity-based measure and one that allows curved regions can be seen in Fig. 11(b) and (c). In this work we introduce a new curvature index which allows construction and handling of curved regions when desired.

The index has to compare face orientations, not just at the common boundary, but across the entire cluster area. The index computation has to be efficient since it is repeated many times during the simplification procedure. This makes the use of analytic measurements impractical.

The region curvature index is defined as a *directed* index from cluster a to b. It is defined based on a set of sampling points in a. Given a point P_a on a the index $S_{a\to b}$ at P_a is computed as follows (Fig. 6(a)).

- Project P_a to the shared boundary to create point P_{ab}.
- Following the direction of the vector $\overline{P_a, P_{ab}}$ compute the point P'_a on b at the same distance from P_{ab} as P_a.
- Compute the angle ψ_a between $\overline{P_a, P_{ab}}$ and $\overline{P'_a, P_{ab}}$.

Given ψ_a there are two options to compute the curvature index:

- If the user requires nearly planar clusters, then the desired surface curvature is close to 0, and the index is measured as:

$$S^1_{a\to b} = \begin{cases} -1 & \text{if } \psi_a < S_{\min} \\ \dfrac{\psi_a - S_{\min}}{\pi - S_{\min}} & \text{otherwise} \end{cases} \tag{5}$$

where $S_{\min}$ is the lower bound on the acceptable angle between the segments.

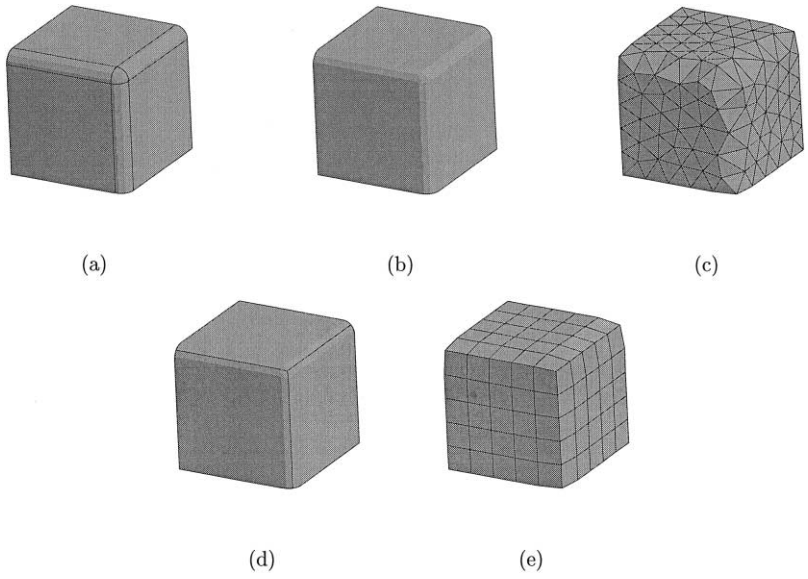


Fig. 5. The effect of the region-curvature measure on the model and its mesh: (a) original model; (b) the model simplified with no region-curvature constraint; (c) tetrahedral mesh of the model (873 elements), a good hexahedral mesh cannot be built; (d) model simplified with a region-curvature constraint (after applying the face-collapse procedure); (e) hexahedral mesh of the model (125 elements).

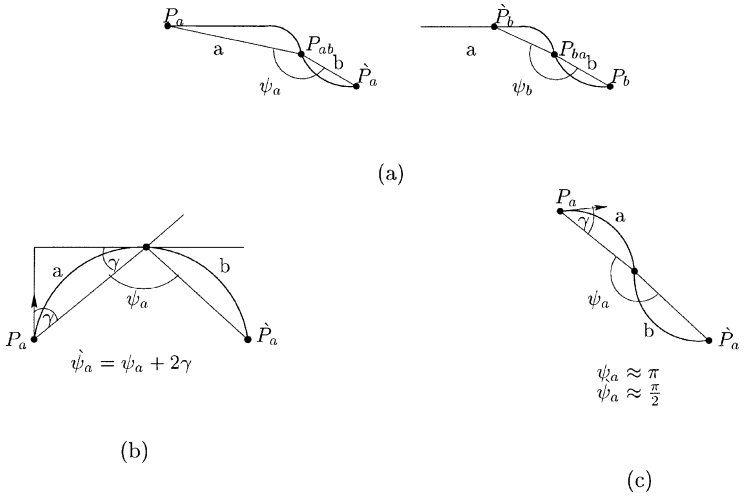


Fig. 6. Region curvature measurements: (a) measuring ψ_a and ψ_b; (b) measure $\acute{\psi}_a$ by taking into account the change in the face normal over the distance. For a cylindrical surface this gives $\acute{\psi}_a = \pi \rightarrow S_{a\rightarrow b} = 1$; (c) the difference between ψ_a and $\acute{\psi}_a$ on a complex surface.

- If the user requires clusters, which are smooth but not necessarily planar, then the desired surface curvature should change gradually. The index is then measured as:

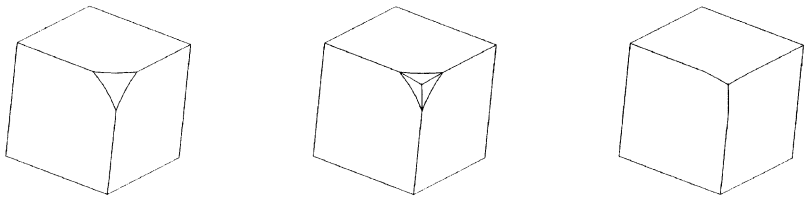
$$\psi_a = \psi_a + 2\gamma$$

$$\textit{if } \acute{\psi}_a > \pi \qquad \acute{\psi}_a = 2\pi - \acute{\psi}_a$$

$$S^2_{a\rightarrow b} = \begin{cases} -1 & \text{if } \acute{\psi}_a < S_{\min} \\ \dfrac{\acute{\psi}_a - S_{\min}}{\pi - S_{\min}} & \text{otherwise} \end{cases} \tag{6}$$

where γ is the angle between $\overline{P_a, P_{ab}}$ and the tangent of a at P_a in its direction. Using this formulation, parts of a cylinder have a perfect match with $S = 1$ (Fig. 6(b)).

The difference between the two measures is shown in Fig. 6(c). As can be seen from this example, the choice of which measure to use depends on the desired behavior of the algorithm and on the shape of the faces and face clusters involved. It is possible to measure both, using $\max\{S^1_{a\rightarrow b}, S^2_{a\rightarrow b}\}$ as the measure, or to choose the measure based on the cluster properties. That is, if a cluster is



(a) (b) (c)

Fig. 7. The stages of a face collapse between all neighbors: (a) a brick with a cut corner; (b) splitting the corner face into three parts; (c) the final model after the merge of the parts with adjacent faces.

nearly planar, then use $S^1_{a\to b}$ for comparing it with adjacent clusters; if it is curved, use $S^2_{a\to b}$.

The above definition applies to a single sample point P_a on a. To get a correct measure of curvature difference, a set of points must be sampled. The worst point is used to determine if $S_{a\to b}$ is set to -1. Otherwise, the average over the sample points is used for the overall index. The size of the sample point set depends on how accurate the index should be. An easy and effective way to get good sample points is to use the nodes from the face triangulations used for a shaded object display. Such triangulations are computed routinely by CAD applications, so no extra overhead is incurred in their computation. They provide a good curvature-based approximation of the model.

3.3. Weight combination from indices

After all the different indices are computed, the total weight of the link is computed. Here we again use the notion of relative attractivity defining, in addition to the overall weight, the weight or overall attraction of each cluster to the other. The weights are defined as follows.

Define the set of indices:

$$I_{a\to b} = \{U_{a,b}, E_{a,b}, A_{a\to b}, R_{a\to b}, C_{a,b}, S_{a\to b}\},\ \text{s.t. } I_{a\to b,i} \in I_{a\to b},\ \ i = 1\ldots 6.$$

$$W_{a\to b} = \begin{cases} -1 & \text{if } \exists I_{a\to b,i} = -1 \\ \sum_i w_i I_{a\to b,i} & \text{otherwise} \end{cases} \tag{7}$$

where w_i are the coefficients (weights) assigned to each index. $W_{b\to a}$ is defined similarly, and the total link weight is defined as

$$W_{a,b} = \begin{cases} -1 & \text{if } W_{a\to b} = -1 \textit{ or } W_{b\to a} = -1 \\ \sum_i w_i \max(I_{a\to b,i}, I_{b\to a,i}) & \text{otherwise} \end{cases} \tag{8}$$

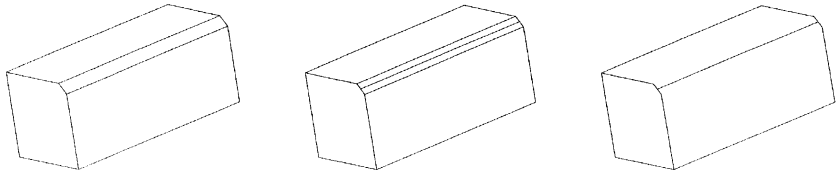
The choice of the coefficients w_i is rather heuristic. In general, once the link passes the mergeability test, (i.e. none of the indices is negative), the main influence of the weights is on the shape of the region. Hence, shape and size indices become more important in determining the arc weight and therefore the order of cluster merging. Section 5 lists the particular coefficients used for the examples displayed.

4. Symmetrical face-collapse procedure

4.1. The collapse operator

Ref. [11] introduced the collapse operator as one of the virtual topology operators. The collapse operator splits a face f between a set of adjacent neighbor faces $\{s_i,\ i = 1\ldots n\}$. The collapse is done in two steps as shown in Figs. 7 and 8. The steps are:

1. split the face f into a set of faces $\{f_i,\ i = 1\ldots n\}$ such that



(a) (b) (c)

Fig. 8. The stages of a face collapse between a sub-set of the neighbors: (a) a brick with a chamfered face; (b) intermediate state after splitting the chamfer into halves; (c) the brick after the collapse of the chamfer between two neighbor faces.

Table 1
Simplification examples data. (NA stands for unset constraint value). The only threshold value which depends on the model or mesh size is A_{max}. For the dreidel model (the only one where A_{max} is set), A_{max} is four times the mesh size used in Fig. 9(c)

Model	E_{min} (°)	A_{max}	R_{min}	S_{min} (°)	Collapse	# faces before	# faces after
Cube Fig. 5(b)	170°	NA	NA	NA	Yes	10	4
Cube Fig. 5(d)	170	NA	NA	135	Yes	10	6
Dreidel	150	20	0.2	120	Yes	47	23
Pump housing	150	NA	NA	130	Yes	127	46
Half sphere (use S^1)	150	NA	NA	135	No	94	8
Half sphere (use S^2)	150	NA	NA	130	No	94	3

each face f_i is adjacent to the shared boundary with the neighbor face s_i;
2. merge each f_i with s_i (for $i = 1...n$).

Both the split and merge operations modify the model connectivity only, keeping the original geometry.

4.2. Collapse detection algorithm

After the face-clustering algorithm has terminated, an algorithm for detecting collapsible faces is applied. Faces which should be collapsed to improve the model are those which can belong to several clusters and have equal attraction to those clusters.

The conditions for a face to be collapsed (split) between a set of clusters are:

- it shares boundary edges with all the clusters;
- it is attracted to all the clusters;
- the attraction to the clusters is nearly identical, i.e. the face is "symmetric" with regard to them.

Given that these conditions are satisfied, the algorithm for detecting collapsed faces acts to create overlaps between clusters by allowing a face to belong to more than one cluster. The faces, which belong to more than one cluster, are later collapsed between the clusters.

The procedure for constructing overlapping regions is:
For each cluster c:

1. For each boundary edge b:
   1.1 Get face f, which contains b but is not in c. f belongs to cluster d.
   1.2 If no such face, continue to next edge.
   1.3 Create clusters $f_c = \{f\}$ and $d_f = d \setminus f_c$.
   1.4 Compute $W_{f_c \to c}$, $W_{f_c \to d_f}$.
   1.5 If $W_{f_c \to c} \approx W_{f_c \to d_f}$,
      1.5.1 $c = c \cup f_c$.
      1.5.2 goto 1 (restart the check on c),
   1.6 else continue to next edge.

After the overlapping clusters are built, the actual merge and collapse are performed. First for each cluster, the faces which belong only to this cluster (*core* faces) are merged together into a *region face*. Then each face belonging to a set of clusters is collapsed between the region faces corresponding to those clusters. Notes: (1) in a case where a cluster contains no core faces, all its faces are removed from other clusters and merged (avoiding a dispersal of a full cluster); (2) the order of collapsing faces is determined by choosing at each stage a face to collapse which shares edges with all the region faces that it is collapsed between.

With the merge and collapse procedure complete, the model simplification procedure terminates.

5. Examples and results

The algorithm was implemented as part of the Gambit CFD preprocessor [13] by Fluent Inc. The implementation of the virtual topology operators in Gambit was used to perform the merge and collapse operators used by the simplification algorithm.

The algorithm is demonstrated below on several examples representing different geometries. The examples and the specific algorithm parameters used in them are summarized in Table 1.

For all the example models the coefficient values w used for computing the arc weights from the geometric indices were set to $w = \{0, .2, .2, .2, .3, .1\}$, where the values of w correspond to the indices $\{U, E, A, R, C, S\}$ respectively. This choice of weights reflects the fact that after the thresholds (Table 1) for the various indices are satisfied, the main consideration in merging the faces is the size and boundary shape of the generated region.

In Fig. 9 a dreidel model is simplified. The initial model is shown in Fig. 9(a) and the simplified model in Fig. 9(b). The simplification process reduced the number of model faces from 47 to 23, including collapse of 12 faces on the four corners of the dreidel handle. The algorithm chose to merge the faces around the handle top with the side faces rather than collapse them between the sides and the top. This is due to the fact that the top face is concave and hence the blended faces have higher attraction to the sides than the top.

Fig. 10 shows the simplification of a typical mechanical part (a pump housing). The model contains numerous blends and fillets, which are detected and handled by the algorithm.

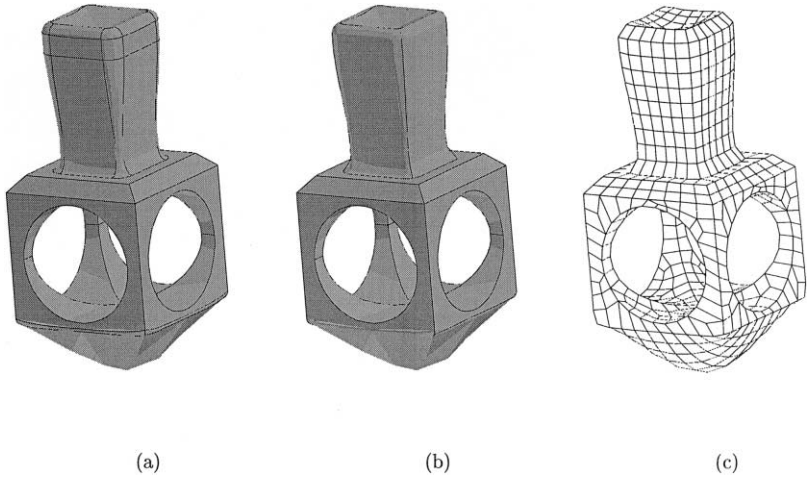


Fig. 9. Simplification of a dreidel shape: (a) initial model with 47 faces; (b) simplified model with 23 faces; (c) quadrilateral surface mesh of the simplified model.

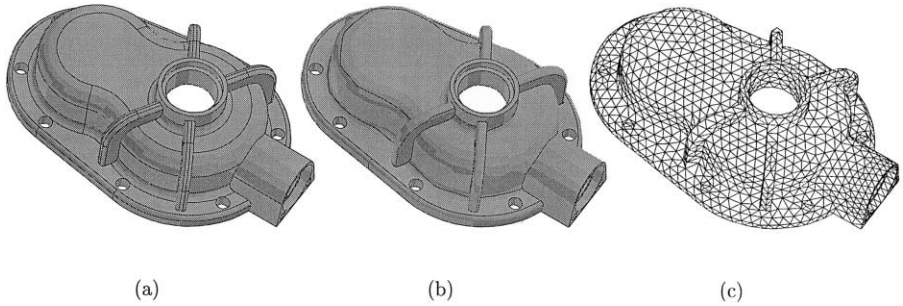


Fig. 10. Simplification of a pump housing model: (a) initial model with 127 faces; (b) simplified model with 46 faces; (c) tetrahedral mesh of the simplified model (7491 elements).

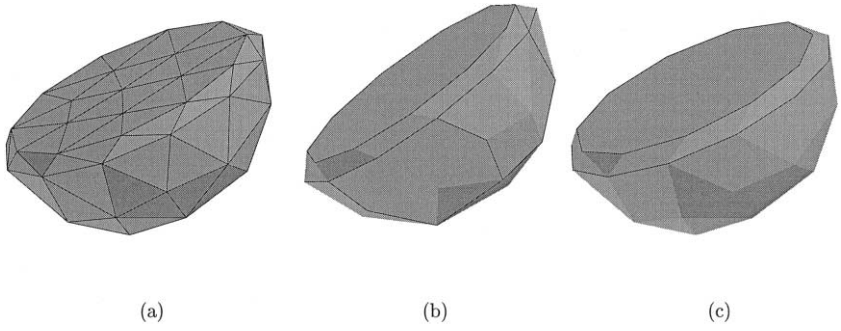


Fig. 11. A faceted model simplification: (a) original model (94 faces); (b) model simplified using a planarity-based measure (use S^1 in S computation) (8 faces); (c) model simplified using a curvature-based measure (use S^2 in S computation) (3 faces).

A fine size tetrahedral mesh of the original model contained 8716 elements. A mesh of the simplified model with similar resolution (Fig. 10(c)) contains 7491 elements and since the original geometry is maintained (only the connectivity changes) there is no loss of accuracy compared to the original mesh.

Fig. 11 demonstrates the simplification of a faceted model — a blended half sphere. Faceted models are often used as input to mesh generation algorithms. To generate a reasonably sized mesh of such a model, the number of faces in the model needs to be dramatically reduced. This model demonstrates the difference between the two region-curvature measures defined in the paper. In Fig. 11(b), a simplification using the planar measure S^1 (using the angle ψ) is applied, resulting in eight relatively planar clusters. In Fig. 11(c), the curvature-based measure S^2 (using ψ) is used for the simplification. This results in more significant simplification with three clusters, which is much more suitable for the model involved. As a result of the

simplification, the number of mesh elements in a tetrahedral mesh of the model is reduced as follows: for element size equal to about half of the original model edges, the number of elements is reduced from 798 to 575; for element size equal to about 1.5 the length of the original model edges the number reduces from 160 to 71.

6. Summary

This work presents a new algorithm for CAD model simplification. The method is especially well suited to simplify models for mesh generation and analysis applications needs, but can be used for other purposes as well. The main step of the algorithm is a procedure for clustering model faces into regions based on a combination of geometric indices that were defined to suit the needs of meshing and analysis. After the clusters are generated virtual topology operators of merge and collapse are used to modify the model connectivity based on the clustering results.

The method is generic and can be applied to any type of model. It handles both faceted and free-form models, closed volumes, sets of faces and non-manifold topologies. It is very efficient and robust, due to the local analysis approach and the use of easy to compute geometric indices. This is the first method that can successfully be used to simplify models with large curved regions (like cylindrical or spherical parts), due to the introduction of a new curvature measure index. By using the collapse procedure, the method provides symmetrical partitioning of such typical features as blends and filets, creating more symmetrical simplified models. Since the model geometry is not changed, the mesh deviation from the original model is bounded by the mesh element size, hence the accuracy of the analysis is not affected by the simplification.

The approach presented in this paper has some drawbacks, whose removal should be investigated in the future. The setting methods for the threshold values of the indices and the coefficients used in computing the arc weights based on the indices are currently very heuristic. A more systematic setting would be advantageous. For this purpose more research is required to investigate the influence of the different indices and especially the region-curvature measures on the simplification results.

A potential limitation of the proposed approach is that the clustering procedure is a greedy type algorithm. That is, the procedure selects the best pair of clusters to merge at a given time. This does not always lead to the optimal solution. A more complex graph search algorithm, which includes retraction of previously performed mergers, might be able to provide better results. However, such an algorithm will necessarily be less efficient.

Acknowledgements

The work reported here was supported, in part, by the Center for Process Simulation and Design (NSF DMS 98-73945) and the Center for Simulation of Advanced Rockets (DOE LLNL B341494). I would like to thank the CUBIT group at Sandia National Laboratories [14] for providing the dreidel (Fig. 9) and pump housing (Fig. 10) models.

References

[1] Shephard MS, Beall MW, O'Bara RM. Revisiting the elimination of the adverse effects of small model features in automatically generated meshes. 7th International Meshing Roundtable, Sandia National Laboratories, SAND98-2250, 1998. p. 347–64.
[2] Klein R, Liebich G, Strasser W. Mesh reduction with error control. Visualization 1996;96:311–8.
[3] Garland M, Heckbert PS. Surface simplification using quadric error metrics. Proceedings of SIGGRAPH '97, 1997. p. 209–21.
[4] Kalvin AD, Taylor RH. Superfaces: polygonal mesh simplification with bounder error. IEEE Computer Graphics and Applications 1996;16(3):64–77.
[5] Eck M, Hoppe H. Automatic reconstruction of b-spline surface of arbitrary topological type. SIGGRAPH '96 Proceedings, 1996.
[6] Volpin O, Sheffer A, Bercovier M. Mesh simplification with smooth surface reconstruction. Computer-Aided Design (CAD) 1998;30:875–82.
[7] Armstrong CG, Robinson DJ, McKeag RM, Li TS, Bridgett SJ, Donaghy CA, McGleenan RJ. Medials for meshing and more. 4th International Meshing Roundtable, Sandia National Laboratories, SAND95-2130, 1995.p. 277–88.
[8] Dabke P, Vallury P, Sheppard S. Using features to support finite element idealizations. ASME Computers in Engineering Conference, Minneapolis, MN, Sept. 1994.
[9] Sheffer A, Blacker TD, Bercovier M. Clustering: automated detail suppression using virtual topology. Trends in Unstructured Mesh Generation, ASME, AMD 1997;220:57–64.
[10] Inoue K, Itoh T, Yamada A, Furuhata T, Shimada K. Clustering large number of faces for 2-dimensional mesh generation. 8th International Meshing Roundtable, Sandia National Laboratories, SAND99-2288, 1999. p. 281–92.
[11] Sheffer A, Blacker TD, Bercovier M. Virtual topology operators for meshing. 6th International Meshing Roundtable, Sandia National Laboratories, SAND97-2399, 1997. p. 49–66 (to appear in International Journal of Computational Geometry and Applications).
[12] Owen SJ. A survey of unstructured mesh generation technology. 7th International Meshing Roundtable, Sandia National Laboratories, SAND98-2250, 1998. p. 239–67.
[13] Fluent Inc. GAMBIT 1.0 CFD preprocessor, User's Guide, 1998.
[14] http://endo.sandia.gov/SEACAS/CUBIT/Cubit.html. Cubit Mesh Generation Toolkit.

Alla Sheffer is a postdoctoral research associate in the Computational Science and Engineering Program, The University of Illinois at Urbana-Champaign (UIUC) since 1999, where she conducts research in geometric modeling, CAD and mesh generation. She obtained her PhD in Computer Science in 1999 from the Hebrew University of Jerusalem. She received a BSc degree in Computer Science and Mathematics in 1991 and an MSc degree in Computer Science in 1995, both from the Hebrew University of Jerusalem. Her research interests include geometric modeling, mesh generation, CAGD and computational geometry.