Proceedings of DETC '03
ASME 2003 Design Engineering Technical Conferences and Computers and Information in Engineering Conference
Chicago, Illinois USA, September 2-6, 2003

DETC2003/CIE-48173

DESTRUCTIVE MODELING BY VOLUME DECOMPOSITION AND ITS APPLICATIONS

Yoonhwan Woo
Graduate school of Automotive Engineering
Kookmin University
Seoul, 136-702, KOREA

Sang Hun Lee
Graduate school of Automotive Engineering
Kookmin University
Seoul, 136-702, KOREA

ABSTRACT

Adding simple volumes, which are often called primitives, is a natural way to construct complex solid models. Conversely, cell-based volume decomposition is the existing method to decompose a complex solid model into simpler volumes that are often the primitives used to create the model. One problem of this volume decomposition is that it generates a large number of cells, many of which are unnecessary for the decomposition. In this paper, a volume decomposition method that improves the performance by avoiding generating the unnecessary cells is presented. Some possible applications are also presented to attest the usefulness of this volume decomposition method.

Keywords: volume decomposition, solid modeling, feature recognition, local modification

INTRODUCTION

In solid modeling, adding simple volumes, which are often called primitives, is a natural way to construct complex solid models. Most of today's feature-based CAD systems utilize this methodology so that users can create solid models more intuitively and also modify the models more easily afterward.

Conversely, volume decomposition is the method to decompose a complex solid model into simpler volumes that are often the primitives used to create the model. For example, the model in Figure 1 can be decomposed into the three primitives with the method. There were several research efforts to decompose solid models into simpler volumes in the past. Most of them are found in the area of machining feature recognition. These methods were used to decompose the delta volume – the volumetric difference between the part and the stock - of machined parts into the volumes of simple machining features.

Shah et al. [1] proposed a method to decompose a solid model into maximum convex volumes by the method called halfspace partitioning. With the method, they attempted to decompose a solid model by intersecting the model and the halfspaces of the faces. But as known in the B-rep to CSG conversion problem [2], it is not always possible to decompose a solid model having curved faces by a set operation of the halfspaces. For example, the volume A in Figure 2(c) cannot be represented by a set operation of the halfspaces, because of the cylindrical faces.

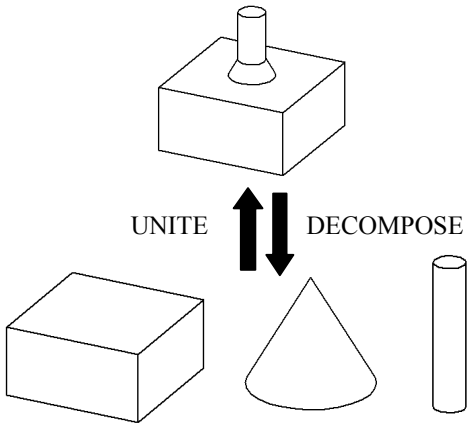

Figure 1. Union vs. decomposition

Sakurai et al. [3] introduced the concept of maximal volumes and developed the method to generate them. This method can decompose solid models with curved faces. A volume is a maximal volume of a solid model if the volume satisfies the following conditions:

1. The volume is contained in the solid model.
2. The volume does not have a concave edge.

3. For each face of the volume, there exists a face of the solid model that shares the same surface geometry and that is connected to or overlaps with the face of the solid model.
4. The volume is not contained in any other volume that satisfies the above three conditions.

The maximal volume decomposition comprises of three steps: cellular decomposition, cell collection, and volume generation. In the step of cellular decomposition, it extends each face of the model that has concave edges and intersects it with the model in order to create cells. Next, the cells satisfying the conditions for maximal volume are collected in the stage of cell collection, and finally the cells in each collection are composed into a maximal volume. The model in Figure 2(a), for example, is first decomposed into 10 cells as shown in Figure 2(b), and they are composed into two maximal volumes as shown in Figure 2(c).

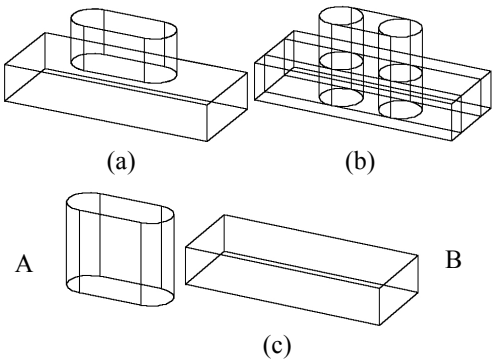

Figure 2. Maximal volume decomposition; (a) solid model; (b) cells; (c) maximal volumes.

In spite of the usefulness in solid modeling such as recognition of intersecting features, the scalability of the cell-based volume decomposition method has been a problem for practical applications - the method becomes very slow as the number of cells increases. There was an effort to improve the scalability of the maximal volume decomposition by recursively bisecting solid models and composing the maximal volumes of the bisected volumes [4], but the generation of a large number of unnecessary cells is still a problem to be solved so that it can be used for more practical applications. Though their method improves the speed of the decomposition using the divide-and-conquer paradigm, the nature of the problem - generation of a large number of *unnecessary* cells – has not been addressed. The generation of unnecessary cells is mainly caused by the global effect of local geometry. The effect of a single face in a solid model is often confined to the vicinity of the face. However, each of the faces having concave edges is infinitely extended and intersected with the whole model to create cells, which unavoidably generates a large number of unnecessary cells.

In this paper, a volume decomposition method to effectively address the problem of generating a large number of unnecessary cells is presented. This method reduces the time required to decompose a solid model into maximal volumes significantly. Hereafter, the Sakurai et al.'s method [3] is referred to as the 'original' method.

MAXIMAL VOLUME DECOMPOSITION WITH REDUCED NUMBER OF UNNECESSARY CELLS

When solid models are Boolean-operated, some portions of the models disappear in the resultant model. It is indispensable to extend the faces of the model in order to restore the disappearing portions. The first stage of the volume decomposition is to extend faces for cellular decomposition. It should be noted that, unlike the faces of the analytic surfaces, there is no unique way to extend faces of free-form surfaces. For this reason, the solid models whose free-form faces have concave edges are not handled in this paper.

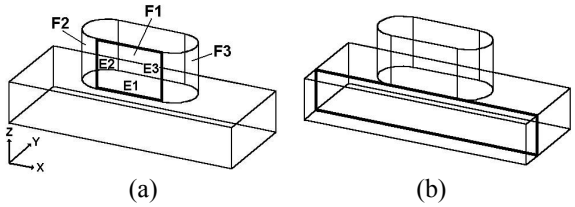

Figure 3. Face extension; (a) F1 to be extended; (b) F2 extended globally.

One problem in cellular decomposition is how to extend the faces to generate cells. It is clear that it would be more efficient to have a less number of cells that are sufficient to generate maximal volumes. The 'original' method for maximal volume decomposition extends each face having concave edges and intersects it with the whole model to generate cells. For example, the face F1 having the concave edge E1 in Figure 3(a) is extended infinitely and intersected with the model, resulting in the face denoted by the thick lines in Figure 3(b). The face is globally extended over the whole model. However, it is observed that some faces do not have to be extended in such a way. The effect of the face F1 for the extension is limited to the negative z-axes directions due to the non-concave edges E2 and E3 that are shared with the cylindrical faces F2 and F3. So it is not necessary to extend the face beyond E2 and E3. Extension of the face only beyond the concave edges will be sufficient.

To locally extend a face having concave edges, it is basically necessary to take into account the adjacent faces sharing the non-concave edges. We extend not only the face but also the faces sharing the non-concave edges, and intersect the extended faces with the copied model. Then only the newly created faces having the concave edges are selected and non-regularly united with the original model. By doing this, we can localize the face extension and avoid generating a large number of unnecessary cells that causes a combinatorial complexity later in the cell collection. The algorithm of the localized face extension is presented in the Extend_face_locally() pseude-code as follows:

```
BOOL Extend_face_locally(F_in)
INPUT: F_in, face
OUTPUT: modified model
```

```
{
   (1) If F_in does not have concave edges, return FALSE;
   (2) If F_in is a free-form face, return FALSE;
   (3) Copy the solid model owning F_in to S_copy;
   (4) Collect F_in;

       For each edge of F_in, e {
   (5)     If e is a concave edge, mark e;
           Else {
   (6)         If e is connected to a concave edge of F_in {
   (7)             Collect the other face sharing e;
               }
           }
       }

       For each of the collected faces {
   (8)     Make F_ext, the double-sided sheet body that is large
               enough to cover the bounding box of S_copy;
   (9)     Non-regularly subtract F_ext from S_copy;
       }
  (10) Select from S_copy the double-sided faces having the
           marked concave edges;
  (11) Non-regularly unite the selected faces with the solid
           model;
       Return TRUE;
}
```

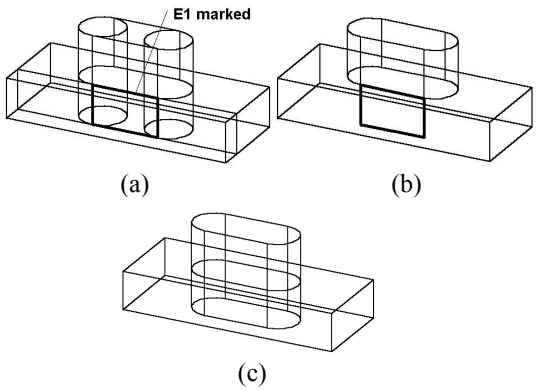

Figure 4. Localized face extension; (a) result of operations (1)-(9) on the copied body; (b) the face non-regularized united with the original model; (c) result of localized face extension.

For example, to locally extend F1 in Figure 3(a), the faces F2 and F3 are collected in addition to F1. Then they are extended to make the double-sided sheet bodies of F1, F2, and F3. Next, they are non-regularly subtracted from the copied model, resulting in the three double-sided faces originated from F1 as shown in Figure 4(a). The face denoted by the thick lines is the only double-sided face having E1. So the face is selected and united with the original solid model as shown in Figure 4(b). Likewise, other faces are locally extended in the same way, ending up with the interim non-manifold model as shown in Figure 4(c).

The cellular topology is then attached to the non-manifold model by forming cells each of which contains topological entities that enclose a void. Each cell points to a CSHELL that bounds the 3D region of the cell, and the CSHELL contains a list of CFACEs. A CFACE then points to a face in the model with a forward or reverse sense. Therefore, every double-sided face generated by non-regularized Boolean operations will be pointed by two CFACEs with opposite senses after the cellular topology is attached on it. With this convention, the information necessary for the next operation such as cell adjacency can be easily obtained.

For the solid model in Figure 2(a), the localized face extension generates only three cells as shown in Figure 4(c), while the original method generates ten cells as shown in Figure 2(b). Figure 5 shows another example. For the solid model in Figure 5(a), the localized face extension generates only 21 cells shown in Figure 5(b), but the original method generates 291 cells shown in Figure 5(c). Once the cells are generated, the collection of the cells and the generation of volumes from the cells are processed in the same way as the original method does.

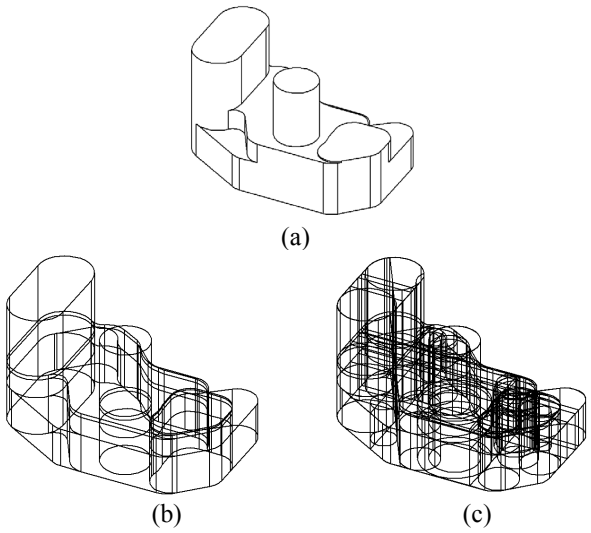

Figure 5. Example of cell generation; (a) a solid model; (b) cell generated with the localized face extension; (c) cell generated with the global face extension.

IMPLEMENTATION AND EXAMPLES

The method presented in this paper has been fully implemented using C++ on top of ACIS® running on Windows NT with the Intel 1Ghz processor. Various solid models have been tested and some of them are shown as examples.

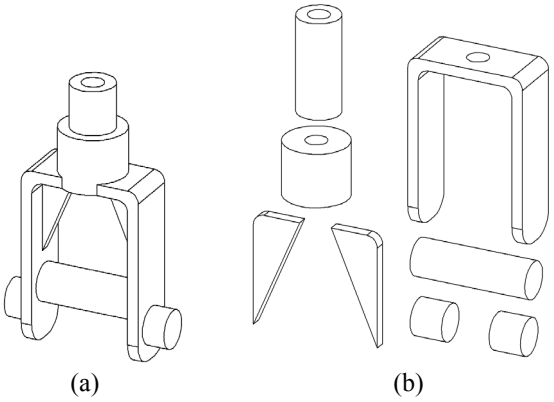

Figure 6. Example 1; (a) solid model of a fork; (b) maximal volumes of the model.

Figure 6(a) shows the solid model of a part. This model is decomposed into 7 maximal volumes shown in Figure 6(b). The number of cells generated by the new method presented in this paper is 50 while the original method generates 186 cells. In terms of the CPU time, the whole process for the maximal volume decomposition by the new method took 1.225 seconds, but it took 2.359 seconds with the original method. The performance discrepancies become bigger as models get more complex. The part in Figure 7(a) is taken from the NIST national design repository [5]. For the delta volume shown in Figure 7(b), the number of cells generated by the new method is 950, while the original method generates 2391 cells. The new method took 72.282 CPU seconds to decompose the delta volume into 74 maximal volumes, but the original method took 218.168 CPU seconds.

As seen with these examples, it has been verified that this localized face extension reduces the number of cells drastically for many cases and improves the overall performance of maximal volume decomposition significantly.

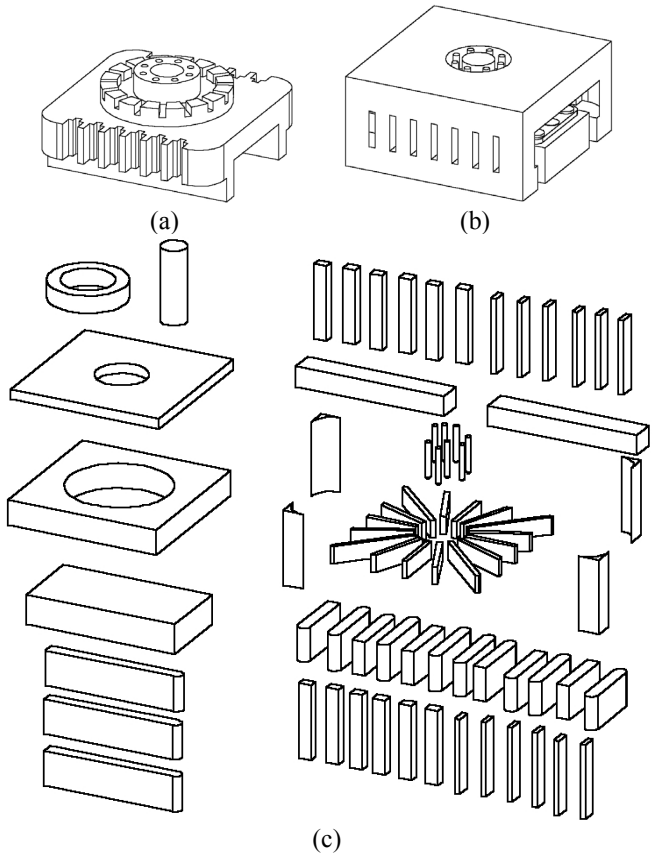

Figure 7. Example 2: (a) part; (b) delta volume; (c) maximal volumes of the delta volume.

APPLICATIONS

Recongnition of intersecting machining features

In spite of the wide use of feature-based design systems, there still exists a need for feature recognition. The reason why feature recognition is still necessary is that features are domain-dependent. For example, the feature-based model for machining needs to be represented with only subtractive features since machining is a process that can only remove material. But designers use both additive features such as extrudes and subtractive features such as cuts, resulting in models with both additive and subtractive features.

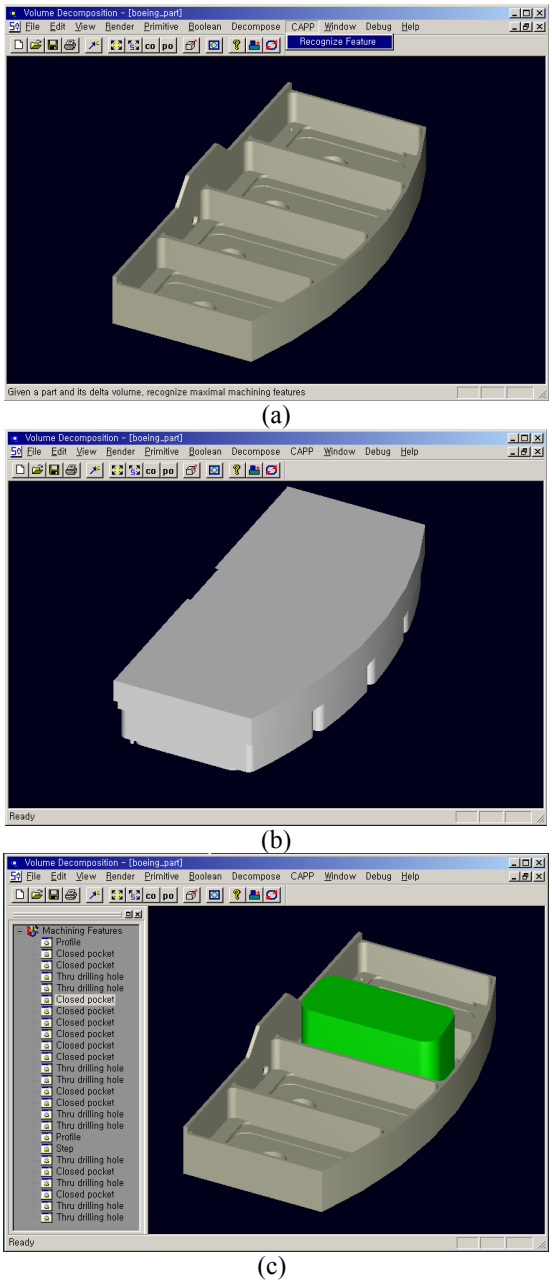

Figure 8. Machining feature recognition by maximal volume decomposition: (a) part; (b) the delta volume of the part; (c) 25 machining features recognized by maximal volume decomposition of the delta volume (a closed pocket displayed).

A machining feature in solid model can be considered as a volume associated with machining information on top of the

geometry and topology of the volume. Delta volume is the difference between a part and the stock and can be viewed as the sum of machining features each of which will be removed by a machining operation. Decomposing the delta volumes into machining features of simpler volumes has been considered as a method for recognizing intersecting features, and maximal volume decomposition is an effective method for this purpose [6]. Most of the maximal volumes of delta volume correspond to machining features. However, it is not always true that maximal volumes are the same as features of interest. Some may not be even machining features. Therefore, maximal volumes are to be processed in the context of machining to generate the features of interest.

Among many methods for feature recognition, some are more efficient to recognize certain types of machining features and some may be more efficient for recognizing other types of machining features. It is difficult to say that a single method is the best for machining feature recognition. This volume decomposition method is useful for recognizing intersecting features but may be expensive to recognize simple isolated features. Thus, along with other feature recognition method, this volume decomposition method can provide a feasible solution to the development of a commercial feature recognizer.

The solid model of a machined part in Figure 8(a) is taken from the National Design Repository. 25 machining features are recognized for the part by decomposing the delta volume shown in Figure 8(b) in several seconds. Figure 8(c) shows the tree of the recognized features and a closed pocket. In this case, every maximal volume corresponds to a machining feature.

Local modification for solid model

Another possible application of maximal volume decomposition is the local modifications for translated CAD models. When CAD data is transferred to an incompatible CAD system, the data has to be translated for the system. The interoperability has helped heighten the status of the concurrent engineering and reduce the time for the product development cycle. However, the interoperability is not complete in the sense that only topology and geometry of the models are transferred. The true interoperability should have the ability to manipulate not only geometry and topology, but also feature information on top of them. But currently the precious feature information is being lost during the CAD data translation for some reasons. Until now, little progress had been made to share and transfer the feature information.

Once feature information is lost, the advantage of feature-based modeling is not valid any longer. Now modifications for the models are purely dependent on the geometric and topological manipulations. The present methods for the local modifications such as moving and tapering faces basically make use of 'tweaking', which is the technique to replace the surface of a face with different one [7]. When the surface of a face is changed, the geometry of the related entities such as edges and vertices also need to be changed. The tweaking operation calculates the curves of the new edges by intersecting the face's new surface with the surfaces of adjacent faces, and the positions of the new vertices by intersecting each new curve with its neighbors. As long as the topology of the entities remains the same, it is straightforward to calculate their new geometry. But the cases are not always so. In many cases, local modifications cause the topological changes and they fail.

In Figure 9(a), for example, suppose a user needs to move the faces of the protrusive feature on the translated model by 2 in the x-axis direction. This modification does not involve topological changes, and the geometry of new edges and vertices can be calculated with the adjacent faces. So the move by tweaking succeeds as shown in Figure 9(b). However, what if the user wants to move the faces of the feature by 5 in the x-axis direction as shown in Figure 9(c)? In this case, the modification fails[1]. The feature moves beyond the side face of the model, so the corresponding topological changes should be made. But as tweaking only involves the faces to be changed and the faces adjacent to them, the information for the topological manipulations cannot not obtained. In order to get the model as shown in Figure 9(c), it is actually necessary to consider all the faces of the model, not only the feature faces and their adjacent faces. However, this approach will be very complicated, and the local modification will not be 'local' any more. It is becoming more like a Boolean operation requiring a 'global' reasoning for the intersections of all the faces of a model.

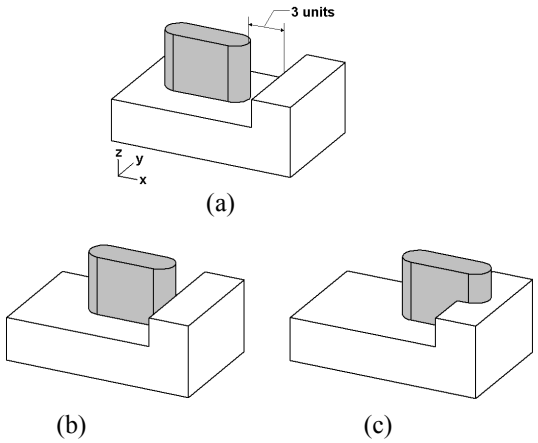

Figure 9. An example of move operation; (a) a protrusive feature on a translated model; (b) move the boss along x-axis by 2 units – no topological changes necessary: tweaking succeeds; (c) move the boss along x-axis direction by 5 units – topological changes necessary: tweaking fails.

Now, suppose we were able to separate the volume of the feature from the model, as shown in Figure 10(a). In this case, it will be straightforward to modify the separated feature without concerning about the topological changes. The move operation can be simply done by translating the separated feature by 5 as shown in Figure 10(b). After that, it is united with the remaining volume by a Boolean operation as shown in Figure 10(c). By doing this, the local modification can be achieved without considering all the faces of the model for tweaking.

Separation of feature can be achieved by applying the maximal volume decomposition to only a set of selected faces. Maximal volume decomposition extends all the faces having concave edges to create cells to decompose a solid model into

[1] With the ACIS 3D modeling kernel, the operation results in an invalid model.

maximal volumes. For the purpose of separation of a specific feature, however, it is not necessary to extend all the faces to create cells. Only a set of faces sufficient to separate the feature from the model are used. Those faces will be the faces of a feature that user selects and their adjacent faces. By extending only these faces and intersecting them with the model, the cells that can be used for the feature separation can be generated. The cell collection process will be the same as the regular maximal volume decomposition except that the concave edges on non-selected faces are treated as "non-concave".

The next step is to find out the volume of the feature from those generated by this selective volume decomposition. It can be found by comparing the selected faces with the faces of a volume. If a volume contains a selected face, it is the volume of the feature. Other volumes are united together. By doing this, we have only two volumes: the volume of feature and the remaining volume with the feature separated.

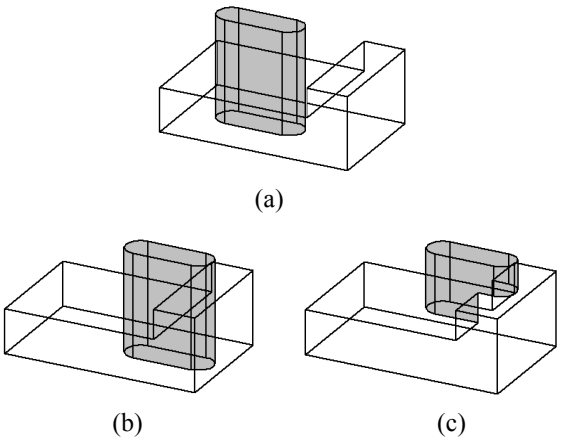

Figure 10. Volumetric approach to local modification; (a) separation of feature volume by volume decomposition; (b) move the feature volume by 5; (c) union of the feature and the model.

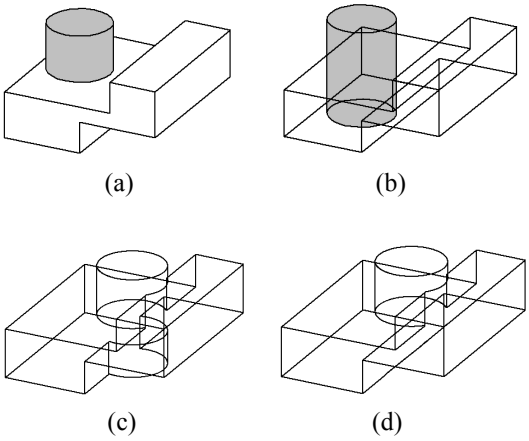

Figure 11. Which is the user's intention? (a) separated feature to be moved; (b) result 1; (c) result 2.

An issue to be addressed for the local modification for solid models using the volume decomposition is that there may be multiple choices based on user's intention. For example, when we say a user wants to move the cylindrical boss in Figure 11(a), what would be the result the user wants? If the result 1 in Figure 11(c) is the user's intention, the modification can be achieved by simply uniting the feature with the remaining volume. However, a simple Boolean operation will not work if the result 2 in Figure 11(d) is the user's intention. This issue will be further examined as a future work.

CONCLUSIONS

A volume decomposition method to effectively address the problem of generating a large number of unnecessary cells has been presented. This method reduces the time required to decompose a solid model into maximal volumes significantly. The practical applications of the method to machining feature recognition and local modification of translated model have been presented to attest the usefulness of the method.

REFERENCES

1. Shah J, Shen Y, Shirur A, Determination of machining volumes from extensible sets of design features. In: Shah J, Mantyla M, Nau D, editors. Adavances in feature based manufacturing, Elsevier, 1994. p.129-157.
2. Shapiro V, Vossler DL, Separation for boundary to CSG conversion. ACM Transactions on graphics, 1993;12(1);35-55.
3. Sakurai H, Dave P. Volume decomposition and feature recognition, Part II: curved objects. Computer Aided Design 1996;28(6/7):519-37.
4. Woo Y, Sakurai, H, Recognition of maximal features by volume decomposition. Computer Aided Design 2002;34:195-207.
5. National Design Repository, http://repos.mcs.drexel.edu.
6. Han J, Regli W, Rosen D, Special panel session for feature recognition, Proceedings of 17th International Computers in Engineering Conference, 1997, Sacramento, CA USA.
7. ACIS 3D modeling kernel online help, Spatial Corp, http://www.spatial.com.