Proceedings of IDETC/CIE 2005
ASME 2005 International Design Engineering Technical Conferences
& Computers and Information in Engineering Conference
September 24-28, 2005, Long Beach, California USA

DETC2005-84912

A Divide-and-Conquer Algorithm for Machining Feature Recognition over Network

Shuming Gao Guangping Zhou Yusheng Liu Xiang Chen
State key Lab of CAD&CG
Zhejiang University, Hangzhou 310027, China
{smgao, gpzhou, ysliu, xchen}@cad.zju.edu.cn

ABSTRACT

In this paper, a divide-and-conquer algorithm for machining feature recognition over network is presented. The algorithm consists of three steps. First, decompose the part and its stock into a number of sub-objects in the client and transfer the sub-objects to the server one by one. Meanwhile, perform machining feature recognition on each sub-object using the MCSG based approach in the server in parallel. Finally, generate the machining feature model of the part by synthesizing all the machining features including decomposed features recognized from all the sub-objects and send it back to the client. With divide-and-conquer and parallel computing, the algorithm is able to decrease the delay of transferring a complex CAD model over network and improve the capability of handling complex parts. Implementation details are included and some test results are given.

Key words: Machining feature recognition; Network centric design and manufacturing; CAD/CAM

1. INTRODUCTION

Machining feature recognition (MFR) that recognizes all the machining features from a solid model is one of the important techniques of CAD/CAM[1,2]. It plays a key role in realizing seamless integration of CAD/CAPP/CAM. In last two decades, a great variety of MFR techniques have been developed, including graph based and “hint” based methods[3,4,5,6], convex hull decomposition[7,8], and volume decomposition-recomposition approaches[9,10,11,12].

As a new product development paradigm, network centric design and manufacturing has been rapidly developed in recent years[13]. For network centric design and manufacturing, it is desirable that designers or engineers without any MFR system to use can invoke remote MFR services to set up the machining feature model of the design they are working on so as to perform manufacturability evaluation or CAPP on the design. That is, network centric design and manufacturing requires network based MFR technique. In order to effectively realize MFR over network, two issues need to be addressed: one is the delay of transferring the huge solid model of a very complex product like an engine over network, which maybe very serious due to the high complexity of the solid model of a complex part; the other is the capability that MFR algorithm deals with complex parts since current MFR algorithms still have difficulty in handling them.

According to our knowledge, the research on network based MFR is still very little up to now. The following two works are comparatively related to this paper.

William C Regli, Satyandra K Gupta and Dana S Nau[14] present an approach for performing trace-based MFR using a multiprocessor architecture. Their approach divides the MFR problem into independent subtasks through the feature types and trace decomposition and performs the subtasks using distributed computing resources. According to their experiment results, the computation time of MFR is improved through the parallelization of their trace-based MFR algorithm. However, the problem decomposition in their approach is closely associated with their trace-based MFR algorithm.

Nagesh Belludi and Derek Yip-Hoi[15] propose a MFR method which recognizes machining features from the VRML model of a part. By using approximate but compressed VRML model of the part rather than its exact and complicated B-rep, their method can reduce the delay of transferring a complex part over network to certain extent. Specifically, the method reconstructs the surface model of the part from its VRML model first, then uses feature recognition algorithm based on graph-matching to recognize features from the surface model, at last, if there are the parameters of features that can not be extracted from VRML model, generates the STEP file of the corresponding local area in the B-rep of the part and converts it to SAT file of ACIS, which are used to extract the exact parameters in ACIS.

Yoonhwan Woo and Hiroshi Sakurai[16] adopt divide-and conquer idea locally in their feature recognition method based on volume decomposition-recomposition. Their method recognizes the machining features with the maximal volumes and conducts process planning based on them. To make the generating of the maximal volumes fast, the divide and conquer strategy is used. First the delta volume of the part is recursively bisected into smaller volumes; then each of the volumes is decomposed into maximal volumes; and finally the maximal volumes generated are composed into the maximal volumes of the delta volume recursively. In their method, the composition is performed upon maximal volumes through union operations. It may not be suitable for the composition of decomposed face-based features well.

The objective of this work is to extend the traditional MFR methods to realize MFR over network and make it fast and be able to deal with very complicated parts to meet the requirements of network centric design and manufacturing. In order to achieve this objective, divided-and-conquer strategy and parallel computing are adopted and incorporated with the traditional MFR methods.

2. OVERVIEW OF THE ALGORITHM

In order to effectively realize the network based MFR, we developed a divide-and-conquer based MFR algorithm. The key idea of the algorithm is decomposing a complex part into a number of sub-objects, taking them as the objects to be transferred over network and performing MFR on each received sub-object in the server in parallel with the transferring of the remainder sub-objects so as to enormously reduce the delay of transferring a complex part over network indirectly and the complexity of the model that the MFR algorithm has to deal with.

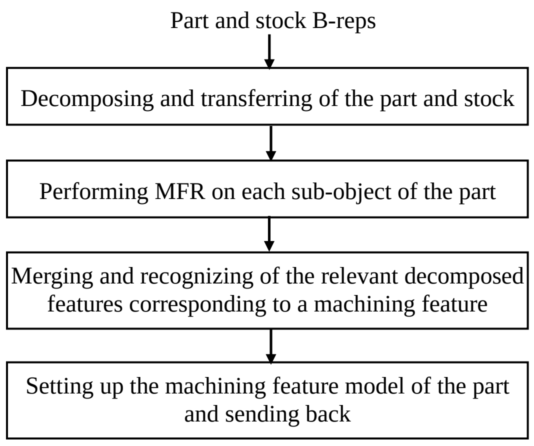

Part and stock B-reps
Decomposing and transferring of the part and stock
Performing MFR on each sub-object of the part
Merging and recognizing of the relevant decomposed features corresponding to a machining feature
Setting up the machining feature model of the part and sending back

Fig. 1. Overview of the algorithm

Fig. 1 gives the overview of the algorithm that consists of four major steps.

1). Decomposing the complex part that needs to be recognized as well as its stock into a number of sub-objects in the client and transferring them one by one to the server where the MFR service is provided;

2). Performing MFR on each received sub-object in the server in parallel with the transferring of the remainder sub-objects;

3). Merging and recognizing all the relevant decomposed features corresponding to a machining feature of the part, in which all the vertices, edges and faces that are lost during the decomposition of the part are reconstructed;

4). Setting up the machining feature model of the part and sending it back to the client in XML format.

3. DECOMPOSING AND TRANSFERRING OF A PART

In order to reduce the degree of complexity of the solid model that needs to be transferred over network each time, the complex part that needs to be performed MFR over network is decomposed into a number of sub-objects first. Decomposing and transferring of a complex part are divided into the following four steps:

1) Constructing three splitting faces. In this work, there are two ways to construct splitting faces: automated and manual. Under the automated way, the algorithm automatically generates the three middle-splitting faces of the enclosing box of the part in the directions of axes X, Y and Z, and takes them as three splitting faces. In view that these three splitting faces are respectively perpendicular to axes X, Y and Z, they are called X-splitting face, Y-splitting face and Z-splitting face accordingly. Fig. 2 shows an example of the three splitting faces constructed automatically for a part. The manual way is based on the automated way, and it allows users to translate the three automatically generated splitting faces along axes X, Y and Z to their desirable positions.

2) Decomposing the complex part and its stock. We decompose the part into 8 sub-objects by splitting it with the three splitting faces. As an example, the result of the decomposition on the part in Fig.2 is shown in Fig. 3. In the meantime, the stock of the part is also decomposed into 8 sub-objects by the same splitting faces, and the corresponding relationship between each sub-object of the part and that of the stock is established.

3) For each sub-object of the part, check if its degree of complexity is less than the given threshold. If so, the decomposition is stopped for this sub-object; otherwise, the above process is repeated for this sub-object. In this work, the degree of complexity of an object refers to the number of the faces the object contains and the threshold is set as 500 by default.

4) Transferring all the sub-objects and splitting faces to the server providing MFR service. Here, the three splitting faces are transferred first. Then, for each time, one sub-object of the part as well as its corresponding stock is transferred.

For the convenience of description and without losing generality, the algorithm given below will just take it into account that the part is decomposed into 8 sub-objects only.

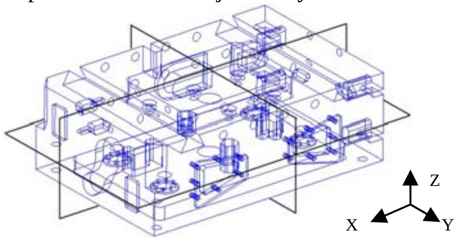


Fig. 2.Three splitting faces of a part

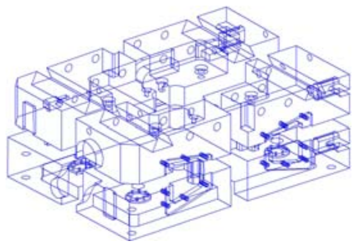


Fig. 3. The decomposed 8 sub-objects

4. MACHINING FEATURE RECOGNITION UPON EACH SUB-OBJECT

In the server, once a sub-object of the part together with its corresponding sub-object of the stock is received, a MFR thread is created and used perform MFR on the received sub-object of the part.

In this work, the approach to MFR based on minimal condition sub-graph[17] is adopted as the basis of the divide-and-conquer algorithm for MFR over network. The minimal condition sub-graph (MCSG) refers to the maximal sub EAAG (Extended Attributed Adjacency Graph) of a feature that remains in the EAAG of the part. In the MCSG based approach, MCSGs are taken as hints to recognize interacting machining features, and they are generated by graph decomposition and completed by adding virtual links, corresponding to the entities lost by machining feature interaction. Fig. 4 gives the overview of the MCSG based MFR algorithm.

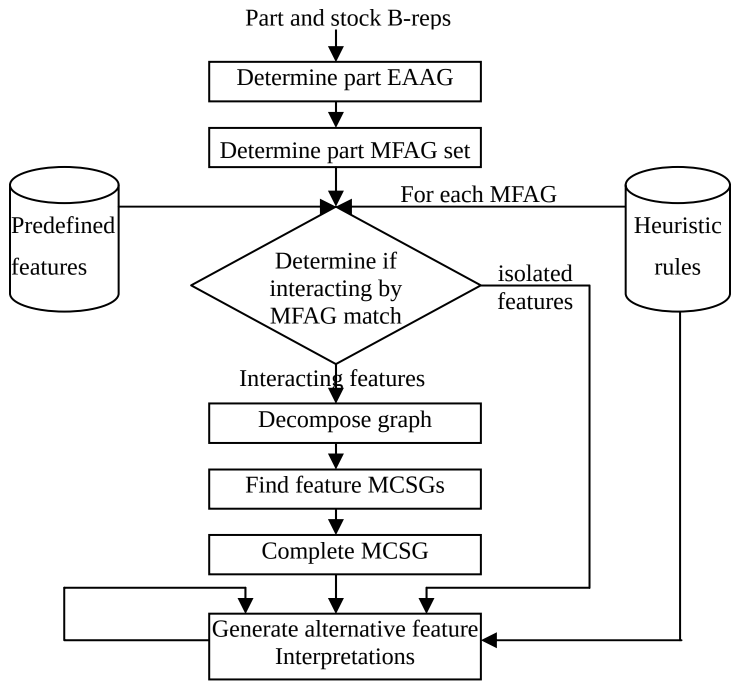
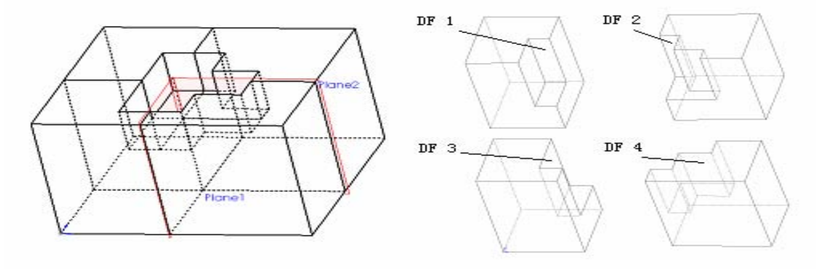


Fig. 4. Overview of MFR based on MCSG

The MFR on each sub-object of the part is not the same as that on the whole part. The main difference lies in that the former needs to recognize decomposed features. When a complex part is decomposed into a number of sub-objects by three splitting faces, some machining feature of the part may be accordingly decomposed into several parts, each of which is in a sub-object of the part and is called a decomposed feature in this work. An example of decomposed features is shown in Fig.5. In order to enable the MCSG based MFR algorithm to recognize decomposed features, we extend it in following two aspects:



(a) A pocket feature in a part (b) Four decomposed features of the pocket

Fig. 5. Illustration of decomposed features

1) During determining whether a part concave adjacency graph (PCAG) forms a machining feature or not, if the PCAG is determined not matching any machining feature in the library, it is further checked whether there exists any node of the PCAG whose corresponding face lies on a splitting face. If there is such a node, the PCAG is recognized as a decomposed feature.

2) For each recognized machining feature, it is further checked whether the feature contains at least one face that lies on a splitting face, if so, the machining feature is marked as a decomposed feature instead of a real machining feature of the part. This is because any face of a real machining feature of the part is impossible to lie on any splitting face.

After all the machining features including decomposed features are recognized using the extended MCSG based MFR algorithm, in order to support the merging of the decomposed features corresponding to a real machining feature, the recognized decomposed features are classified and their shell bodies are generated as follows:

1) All the decomposed features are classified into the following seven types according to the related relationships between a decomposed feature and the three splitting faces. Here a decomposed feature is called related to a splitting face if it was generated completely or partially by this splitting face during the part decomposition.

(1) The decomposed feature is just related to X-splitting-face;

(2) The decomposed feature is just related to Y-splitting-face;

(3) The decomposed feature is just related to Z-splitting-face;

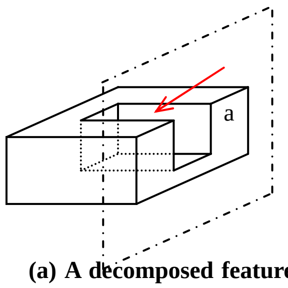
(4) The decomposed feature is related to both X-splitting-face and Y-splitting-face;

(5) The decomposed feature is related to both X-splitting-face and Z-splitting-face;

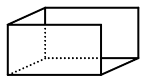
(6) The decomposed feature is related to both Y-splitting-face and Z-splitting-face;

(7) The decomposed feature is related to all the three splitting faces;

2) For each decomposed feature, a shell body independent of the sub-object it belongs to is created by invoking a shell body generation procedure with all the current faces of the decomposed feature as input, and its pointer is put into the data structure of the decomposed feature. Fig.6 illustrates the constructing of the shell body of a decomposed feature. Later, the EAAG of the machining feature restored through the merging of the decomposed features can be generated completely based on the shell bodies of the decomposed features rather than the sub-objects they belong to, so that it becomes simpler and the demand of storage space can be largely reduced.



(a) A decomposed feature in a sub-object



(b) Shell body of the decomposed feature

Fig. 6. Constructing of a decomposed feature's shell body

Finally, seven link lists are established, each of which is used to record a type of decomposed features recognized during the MFR on the sub-object.

5. MERGING AND RE-RECOGNIZING OF THE DECOMPOSED FEATURES

After the MFR upon all the sub-objects has been completed, the relevant decomposed features recognized are merged so as to correctly restore the real machining features of the part that are destroyed during the decomposing of the part. Here the relevant decomposed features refer to those decomposed features that belong to a common machining feature of the part. The algorithm of merging and re-recognizing of the relevant decomposed features consists of three parts.

5.1 Merging of the shell bodies of the relevant decomposed features

In order to merge the relevant decomposed features, we first get their shell bodies merged as follows.

1) Determining the relevant decomposed features related to the X-splitting face and getting each pair of the decomposed features that have the common edge on the X-splitting face merged

(1) Find out all the decomposed features with types 1, 4, 5 and 7 from all the recognized decomposed features. According to the classification, they are all related to the X-splitting face;

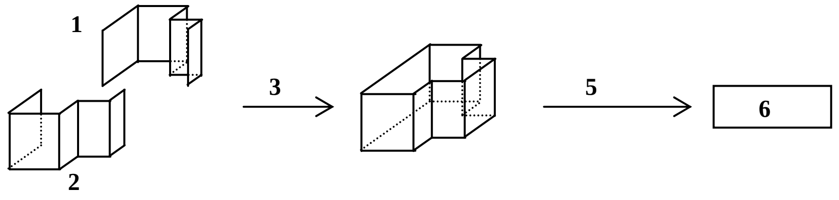
(2) For each pair of the decomposed features related to the X-splitting face, check whether there exists any common edge on the X-splitting face between the shell bodies of the two decomposed features, if so, these two decomposed features are the relevant decomposed features related to the X-splitting face, and their shell bodies are merged to a new one using Boolean union operation. Also these two decomposed features are merged to a new one that takes the new generated shell body as its shell body and replaces the two decomposed features that have been merged.

2) Determining the relevant decomposed features related to the Y-splitting face and getting each pair of the decomposed features that have the common edge on the Y-splitting face merged. The method is the same as that used in 1).

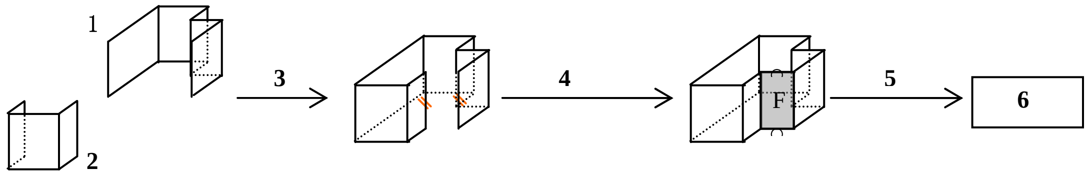
3) Determining the relevant decomposed features related to the Z-splitting face and getting each pair of the decomposed features that have the common edge on the Z-splitting face merged. The method is the same as that used in 1).

5.2 Reconstructing of all the lost faces of the merged decomposed features

In the process of decomposing a part, the face of the part that coincides with a splitting face is absorbed by the splitting face and does not appear in any sub-object of the part. Such faces are called lost faces. When a lost face is also a face of a machining feature, it becomes a lost face of the relevant decomposed features merged corresponding to the machining feature. For instance, the shaded face F in Fig. 7(b) is a lost face of the relevant decomposed features merged in the figure. Obviously, in order to correctly restore the machining feature of the part from the relevant decomposed features merged, all the lost faces must be reconstructed and added to the shell body of the relevant decomposed features merged during the merging of the relevant decomposed features. Fig.7 gives two processes of merging and re-recognizing the relevant decomposed features: one for the case that the relevant decomposed features merged has no lost face; the other for the case that the relevant decomposed features merged has some lost face.



(a) The case without any lost face



(b) The case with some lost face

1 Decomposed feature	**2 Decomposed feature2**
3 Merging	**4 Reconstructing of lost faces**
5 Re-recognizing	**6 A through pocket**

Fig. 7. Merging and re-recognizing of the relevant decomposed features

In view that the lost faces of the relevant decomposed features merged can be respectively reconstructed based on each splitting face, without loss of generality, we just describe the algorithm of reconstructing lost faces based on X-splitting face below. The algorithm consists of following three parts.

1) Determining of open edges of the shell body

Open edge refers to an edge of the shell body of the relevant decomposed features merged, which lies on the splitting face and is adjacent with only one face. For example, the edges marked with two inclined lines in Fig. 7(b) are two open edges. The significance of open edges is that they are the hints of lost faces. If there isn't any open edge on the splitting face, there is no lost face on that splitting face.

According to the above definition, determining open edges is quite simple. That is, for each edge of the shell body, the edge is checked whether it lies on the splitting face, if so, it is an open edge.

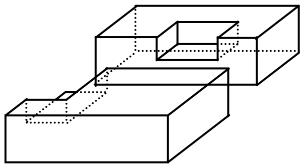
If a vertex of an open edge is not adjacent to any other open edge, it is called an open vertex. For instance, the vertices 1-4 in Fig. 9(a) are four open vertices, while the vertices 5-8 are not. In the figure, the edges pointed by arrowheads are open edges.

Finally, all the determined open edges and open vertices of the shell body, which lie on X-splitting face, are respectively recorded by two link lists, one is called open edge link list, and the other is called open vertex link list.

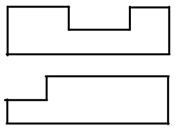
2) Reconstructing of lost edges of the shell body

The edge of the part that is no longer in any of its sub-objects after the part decomposition is called a lost edge. When a lost edge is also an edge of a machining feature, it becomes a lost edge of the relevant decomposed features merged corresponding to the machining feature. For example, the edges with circular marks in the shaded face F in Fig. 7(b) are two lost edges of the relevant decomposed features merged in the figure. It is obvious that all the edges of a lost face except open edges are lost edges. Thus, in order to reconstruct all the lost faces of the shell body, next step is to reconstruct all its lost edges.

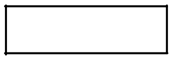
By analyzing the reasons why lost edges are lost, we divide all lost edges into two kinds. One is the lost edge absorbed by the separating boundary formed by the intersection between a splitting face and the part. Specifically, the separating boundary is defined as the boundaries' union of the two faces of two sub-objects, which are generated during the part decomposition and coincide with the splitting face, as illustrated in Fig. 8. For example, the lost edges in Fig 9(b) are such kind of lost edges. Another kind of lost edge refers to the lost edge absorbed by the intersection between two splitting faces. The black line in Fig. 11(b) belongs to this kind.



(a) Two sub-objects

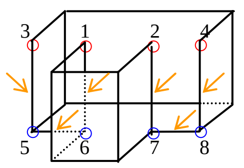


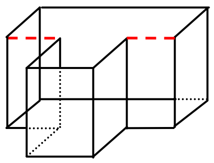
(b) Boundaries of two sub-objects on the splitting face



(c) The separating boundary

Fig. 8. Illustration of the separating boundary of a splitting face





(a)Open point queue (red color) **(b)Lost edges**

Fig. 9. Reconstructing of lost edges

According to the definition and classification of lost edges, we reconstruct all the lost edges of the shell body as follows. Here also just X-splitting face is taken into account.

Generating the separating boundary corresponding to X-splitting face

Find out all the faces that lie on X-splitting face from all the sub-objects first; then perform Boolean union operations on them. The boundary of the resultant face of the Boolean union operations is the separating boundary.

(1) Determining and ordering the open vertices on the same line of the separating boundary

For each line of the separating boundary and each intersection line between two splitting faces, find out all the open vertices on it from the open vertex link list and make them ordered according to their parameter values in the line. In this way, a number of open vertex queues are obtained.

(2) Reconstructing lost vertices and adding them to open vertex queues

The lost vertex refers to the common vertex of two lost edges. For example, vertices 5 and 7 in Fig.11(c) are two lost vertices.

According to the types of lost edges, all the possible lost vertices can be classified into three kinds. The first one is that the lost vertex is exactly a vertex of the separating boundary, i.e. the two lost edges adjacent to the lost vertex are both on the separating boundary. The second one is that the lost vertex is the intersection point of the separating boundary and the intersection line between two splitting faces. The third one is that the lost vertex is the intersection point of the two intersection lines between splitting faces.

Taking Fig. 10 as an example, where the cube is uniformly decomposed into 8 sub-objects, the vertices with circular marks in the figure are all kinds of possible lost vertices.

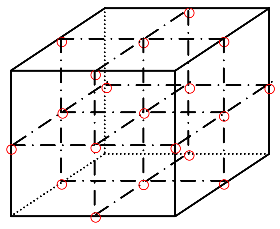
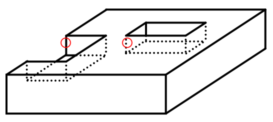


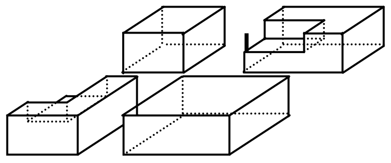
Fig. 10. All possible lost points

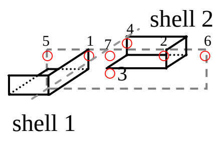
With the above definition and classification, we reconstruct all lost vertices and adding them to open vertex queues as follows:

First, generate all the possible lost vertices, which is quite easy according to the above classification; Then for each open vertex queue, find out the lost vertices that lie on the line that all the vertices in the open vertex queue lie on from all the possible lost vertices, if there is such lost vertex, further check whether the lost vertex is on the enclosing box of the shell body, if so, it is a real lost vertex and properly inserted into the open vertex queue according to its parameter value in the line.

Taking Fig.11 as an example, vertices 5 and 7 in Fig.11(c) are the two lost vertices and they are reconstructed and added into the open vertex queue (1,2). The lost vertex 7 is also added into the open vertex queue (4) and the open vertex queue (3).







(a) A part **(b) Four sub-objects** **(c) Lost vertices are added into open vertex queues**

Fig. 11. Reconstructing and using of lost vertices

(3) Generating lost edges

For each open vertex queue, starting from its first open vertex, every pair of open vertices are taken out in order and used to generate a new edge with these two open vertices as two vertices of the new edge. Such generated edges are all the lost edges. Taking Fig.9 as an example, the lost edge whose two vertices are open vertices 3 and 1 is generated first, and then the lost edge whose two vertices are open vertices 2 and 4 is generated.

3) Reconstructing of lost faces of the shell body

According to the fact that the boundaries of all the lost faces on a splitting face are completely composed of the open edges and lost edges on the splitting face, we reconstruct all the lost faces on X-splitting face as follows: First generate all the loops composed of the open edges and lost edges on X-splitting face by tracing the connected edges; Then construct all the lost faces with the generated loops.

5.3 Re-recognizing of The Relevant Decomposed Features Merged

For the relevant decomposed features merged, after all its lost faces on all splitting faces are reconstructed and added into the shell body of the relevant decomposed features merged, MFR is performed on the shell body to recognize and construct the machining feature to which the relevant decomposed features merged correspond. In this way, all the machining features of the part that are decomposed during the part decomposition are restored. For example, after re-recognizing of the two relevant decomposed features merged as shown in Fig. 7(b), a through pocket is recognized and constructed, which is exactly a machining feature of the part.

6. GENERATING AND TRANSFERRING OF THE MACHINING FEATURE MODEL

After all the machining features are recognized from all the sub-objects as well as the relevant decomposed features merged, the complete machining feature model of the whole part is generated and sent back to the client sending the sub-objects as follows.

1) Generating the complete B-rep of the part

The complete B-rep of the part is generated in the server by performing Boolean union operations among all the sub-objects.

2) Generating the complete machining feature model of the part

The complete machining feature model of the part consists of all the recognized machining features. After all the machining features of the part are recognized and constructed, since the faces referred by the recognized machining features are from different sub-objects and the shell bodies of the relevant decomposed features merged, it is needed to make them all come from the complete B-rep of the part. Specifically, for each face referred by a machining feature, its corresponding face in the complete B-rep of the part is found out and used to replace it in the representation of the machining feature.

3) Sending back the complete machining feature model of the part

After the complete machining feature model of the part is generated, it is converted into XML format and transferred to the client sending the sub-objects. Due to the limitation of space, the specific definitions of the XML schemes for machining feature models are omitted here.

7. IMPLEMETATION

The proposed algorithm has been implemented in the prototype named NetMFR on the Microsoft Visual Studio C++6.0 and ACIS6.0 geometric modeling kernel. In the implementation, Client/Server architecture is adopted and Socket are used to achieve the communication between client and server over TCP. The client is responsible for decomposing a complex part as well as its stock into a number of sub-objects and sending them to server one by one, while the server is in charge of performing MFR on each received sub-object, generating the complete machining feature model of the part and sending it back to the client.

In order to improve the efficiency of MFR over network, multi-thread mechanism is adopted to realize parallelization between the transmission of sub-objects and the MFR on them. When a sub-object as well as its stock gets to the server, the system creates a thread and performs MFR on the received sub-object with this thread at once, and the next sub-object is being transferred synchronously.

In Fig. 12-18, we give a sample of performing MFR on a complex part using our prototype system NetMFR. In the sample, the test part containing 527 faces and 112 machining features is decomposed into 8 sub-objects (see Fig.12-13). During the MFR on the sub-objects, 85 normal machining features (see Fig. 14) and 48 decomposed features (see Fig. 15) are recognized. The machining features restored by merging and re-recognizing of the decomposed features are shown in Fig 16, and 13 reconstructed lost faces are shown in Fig. 17, marked by thick lines. Finally, several key steps of the MFR performed on the test part using NetMFR are shown in Fig. 18.

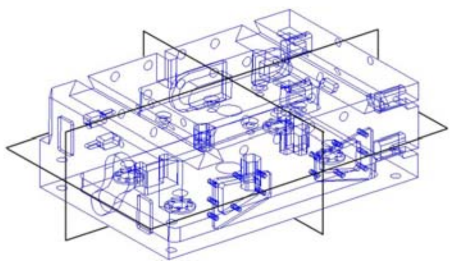


Fig. 12. Test part and three splitting faces

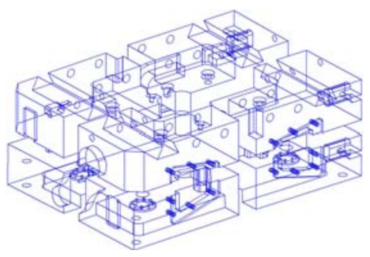


Fig. 13. 8 sub-objects of the part

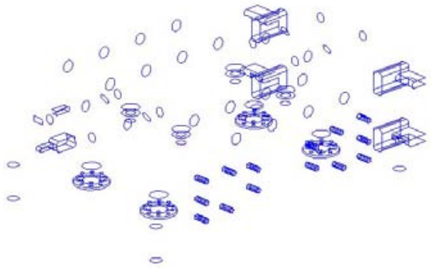


Fig.14. Recognized normal machining features

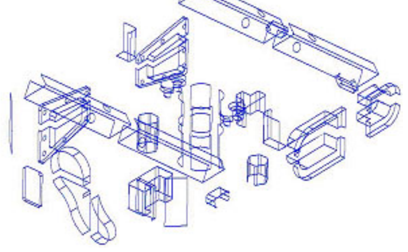


Fig.15. Recognized decomposed features

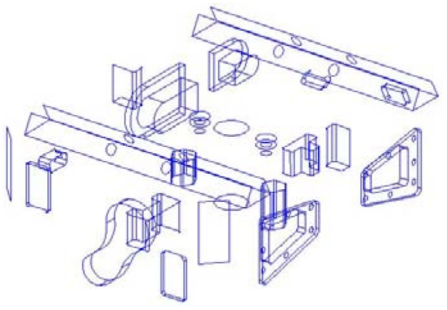


Fig.16. Restored machining features

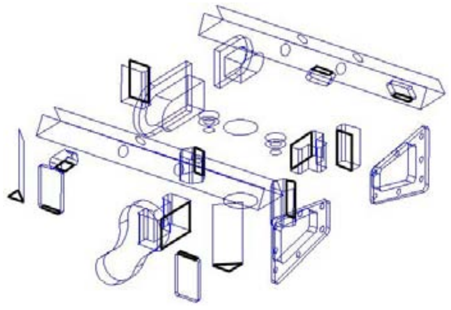
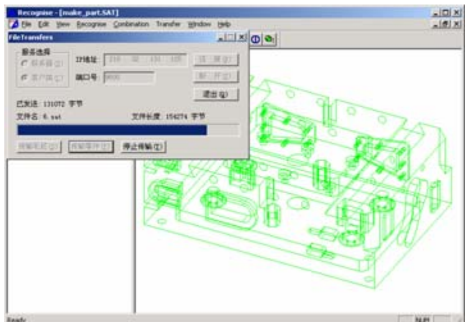
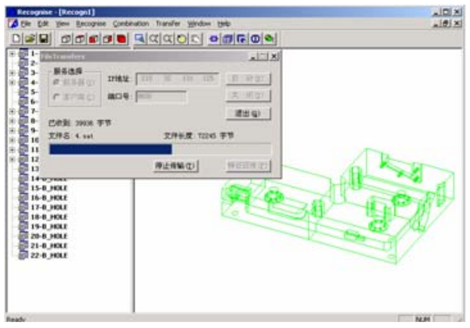
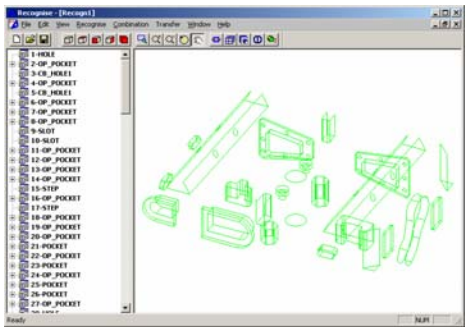


Fig.17. Reconstructed lost faces


(a) The client is starting to transfer sub-objects of the part to the server


(b) The server is receiving a sub-object while performing MFR on others.


(c)The server is restoring the machining features from decomposed features

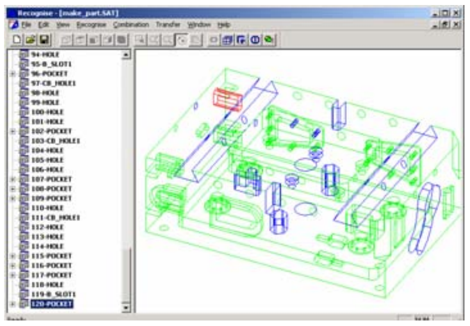

(d) The client is displaying the machining feature model of the part

Fig. 18. Several key steps of the MFR performed on the test part using NetMFR

8. CONCLUSION

To meet the demand of network centric design and manufacturing, a divide-and-conquer algorithm for MFR over network is presented in this paper. The main contributions of the work include:

a) Making the degree of complexity of the objects that need to be transferred and recognized during the MFR over network greatly decreased by using divide-and-conquer algorithm, which makes it possible to remarkably improve the efficiency of the MFR over network;

b) Developing an effective approach for correctly restoring a machining feature from its decomposed features and splitting faces based on the reconstruction of the lost vertices, lost edges and lost faces.

c) Enabling the traditional MFR algorithms to deal with much more complicated parts by incorporating divide-and-conquer strategy into them.

The future work is to further optimize and parallelize the algorithm.

9. ACKNOWLEDGEMENT

The authors are very grateful to the financial support from NSF of China (No.60273057), the Trans-Century Training Programme Foundation for Talents by the Education Ministry of China and Key Project of Science and Technology of Zhejiang Province (2005C21007,2005C21096).

REFERENCES

[1] Shah J.J, Anderson D., Kim Y.S., and Joshi S., A Discourse on Geometric Feature Recognition From CAD Models. Journal of Computing and Information Science in Engineering, 1(1), 2001, pp. 41-51.

[2] Shah J.J.,and Mantyla M., Parametric and Feature-Based CAD/CAM John-Wiley and Sons Inc. 1995.
[3] Joshi, S., and Chang, T., Graph-Based Heuristics for Recognition of Machined Featured Features from a 3D Solid Model, Comput. -Aided Des., 20, No.2, 1988, pp.58-66.
[4] Corney, J., and Clark, D., Method for Finding Holes and Pockets that connect Multiple Faces in 2 & 1/2 D Objects, Comput. –Aided Des., 26, No.3, 1991, pp.215-224.
[5] Vandenbrande, J., and Requicha, A., Spatial Reasoning for the Automatic Recognition of Machinable Features in Solid Models, IEEE Trans. Pattern Anal. Mach. Intell., 15, No. 12, 1993, pp.1269-1285.
[6] Regli, W. Gupta, S., and Nau, D., Extracting Alternative Machining Features: An Algorithmic Approach, Res. Eng. Des., 7, No. 3, 1995, pp.173-192.
[7] Woo, T., Feature Extraction by Volume Decomposition, Proc. Conf. On CAD/CAM Technology in Mechanical Engineering, MIT, Cambridge, MA, 1982, pp.76-94.
[8] Waco, D., and Kim, Y.S., Geometric Reasoning for Machining Features Using Convex Decomposition, Comput.-Aided Des., 26, No. 6, 1994, pp.477-489.
[9] Sakurai, H., and Chin, C., Definition and Recognition of Volume Features for Process Planning, Advances in Feature Based Manufacturing, J. J. Shah, M. Mantyla, and D. Nau, eds., Elsevier Science Publishers, New York, 1994,pp.65-80.
[10] Shen, Y., and Shah, J., Feature Recognition by Volume Decomposition Using Half-Space Partitioning. Proc. 20th ASME Design Automation Conference. 1994, pp.575-583.
[11] Sakurai, H., Volume Decomposition and Feature Recognition: Part I-Polyhedral Objects, Comput.-Aided Des., 27, No.11, 1995, pp.833-843.
[12] Sakurai, H., and Dave, P., Volume Decomposition and Feature Recognition: Part 2-Curved Objects, Comput.-Aided Des., 28, No. 6, 1996, pp.517-537.
[13] Fuh J.Y.H.,and Li W.D., Advances in Collaborative: The-State-of-the-Art Proc.CAD'04 Conf.,Pattaya,Thailand, 2004, pp.387-396
[14] William C Regli, Satyandra K Gupta and Dana S Nau, Towards multiprocessor feature recognition. Computer-Aided Design, 1997, Vol 29(1) 37-51
[15] Agesh Belludi and Derek Yip-Hoi, A Hybrid Approach to Feature Recognition Using Approximate and Partial Exact CAD Models. DETC2002/DAC-34101
[16] Yoonhwan Woo and Hiroshi Sakurai, Recognition of maximal features by volume decomposition. Computer-Aided Design, 2002, Vol 34(3) 195-207
[17] Gao S. and Shah J. J., Automatic recognition of interacting machining features based on minimal condition subgraph. Computer Aided Design, 1998, Vol. 30(9).727-739.