AUTOMATED ANALYSIS OF 3D POLYHEDRAL ASSEMBLIES : ASSEMBLY DIRECTIONS AND SEQUENCES

by

P.K. VENUVINOD

Professor and Head, Department of Manufacturing Engineering,
City Polytechnic of Hong Kong, Hong Kong

ABSTRACT

Some issues concerning the automated analysis of three dimensional (3D) assemblies consisting of polyhedral components (i.e. components with plane faces) are studied. The product is assumed to have been modelled in a Boundary Representation environment. An algorithm for mating surface identification is first developed. Noting that previous published work on the automated determination of assembly directions has largely been confined to the 2D domain, a new approach to the automated determination of the feasible range of assembly directions in the 3D domain is proposed. This consists of the concept of Principal Disassembly Directions which lie on the boundary of the region defining all feasible assembly directions. Finally, an algorithm for determining the translational freedom of components during assembly is presented. These algorithms enable a rapid construction of the disassembly tree which forms the basis for the automated generation of assembly sequences in the 3D domain.

INTRODUCTION

Current practices concerning the design-manufacture cycle in industry are iterative, wherein the designs move back and forth between designers and manufacturing engineers. This is because designers cannot always foresee the problems that might occur downstream in the manufacturing phase. This iterative process results in a significant loss of time. One approach to alleviating this problem is to develop computerised tools for speeding up the design-manufacture cycle. Designers these days typically use CAD systems which merely make the design process more convenient while leaving unaddressed the evaluation of the design in terms of manufacturing lead times and costs. There is a need for Design for Manufacture (DFM) which concentrates on the integration of manufacturing criteria into the product design process. With the availability of robust DFM tools, product and process design could take place as parallel activities, thus facilitating Concurrent Engineering.

Most products are unions of individually manufactured components and go through an assembly process. A major aspect of DFM therefore is concerned with Design for Assembly (DFA) whose primary aim is to ensure that the product is easy to assemble.

Many systems have been proposed in recent years to enable systematic product analysis from the point of view of DFA [Mile82] [Miya86] [Stur89] [Boot83]. Amongst these the Umass system of Boothroyd and Dewhurst [Boot83] seems to be the most widely used. However, current applications of the Umass system (and, indeed, of any DFA system) are either totally manual or computer assisted. It should be highly useful to a designer if the processes involved in DFA could be perfomed automatically. This would facilitate a faster assimilation of systematic DFA methods into industrial practice and form a step towards Concurrent Engineering. The present paper describes a part of a larger project in progress at the City Polytechnic of Hong Kong which aims to automate atleast the geometric aspects of the UMass system for DFA. The broad structure of this project is illustrated in Fig. 1. The system aims to determine the "design efficiency" of the given product. The product itself is modelled using Boundary Representation (BR) since one requires information on faces (eg mating faces) in DFA and such information is explicitly available when BR is adopted. The present paper focuses on the modules related to 'Assembly Pre-processing'.

Although the problem of assembly direction and sequencing has been studied extensively in recent years [DeFa87] [Ko87] [Woo87] [Dini92] [Lape92], only Woo has explicitly addressed the issues concerning their automated determination. Woo has developed the concept of visible region (VR) which defines the range of geometrically feasible assembly directions between a pair of mating components. Further he has developed an elegant procedure for generating the assembly tree which forms the basis for assembly sequence generation. However, Woo's work [Woo87] was confined to assemblies with polyhedral components in the 2D domain. This paper aims to extend these concepts to the 3D domain with particular reference to problems concerning mating surface identification, the visible region and translational freedom of components. As with [Woo87], the present work is confined to the analysis of assemblies made up of polyhedral components, i.e. components with plane faces only.

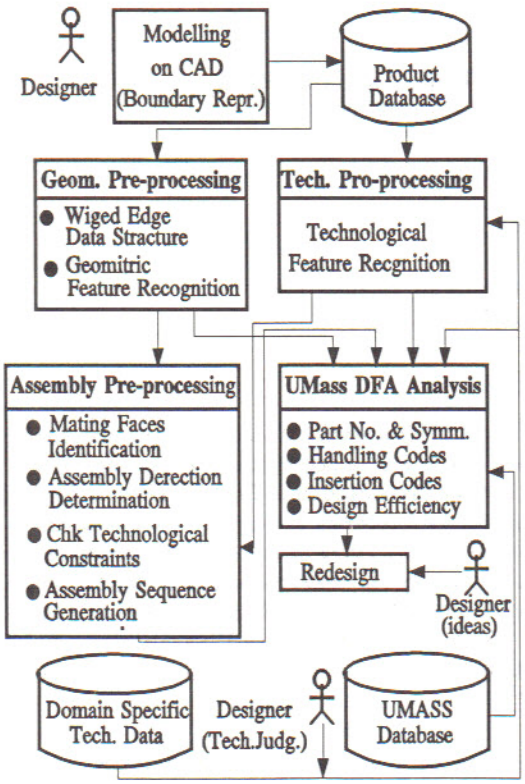


Figure 1 A Strategy for Automating Umass Design for Assembly

IDENTIFICATION OF MATING FACES AND LIAISONS

It is assumed that the product is modelled in the assembled form with respect to a global coordinate system (XYZ). The geometric information concerning each face of each component is stored as a normalised plane coefficient set (A,B,C,D) corresponding to the equation $Ax+By+Cz=D$ of the plane of the face. Since the present work has utilised a boundary representation during modelling, such data are easily derivable from the coordinates of any three vertices belonging to the face. If the modelling had used a CSG (constructive solid geometry) representation, it would have been necessary to derive the coefficient set using appropriate boundary evaluation algorithms. Next, the coefficients of the unit normal vector to each face are derived from the equation of the plane. Here, the positive direction of the unit normal vector is taken in the direction pointing *into* the component. Clearly, this direction is a candidate disassembly direction for the part.

De Fazio and Whitney [DeFa87] have characterised an assembly as a graph wherein the nodes represent components and the arcs represent certain user defined *liaisons*. For the present purpose, a liaison is said to exist between a pair of components if there is physical contact between the pair, i.e. the components mate. The mating could occur over one or more faces which may or may not be adjacent. Thus each face of each component needs to be checked for the possibility of mating with every face of every other component.

A candidate pair of faces may be identified as mating if they satisfy the following two criteria:

i The faces are coplanar and their unit normal vectors point in opposite directions - this is checked by matching the normalised plane coefficient sets and comparing the normal vectors.

ii. The faces overlap.

Three possible cases need to be considered in checking for criterion ii above. Firstly, the faces could

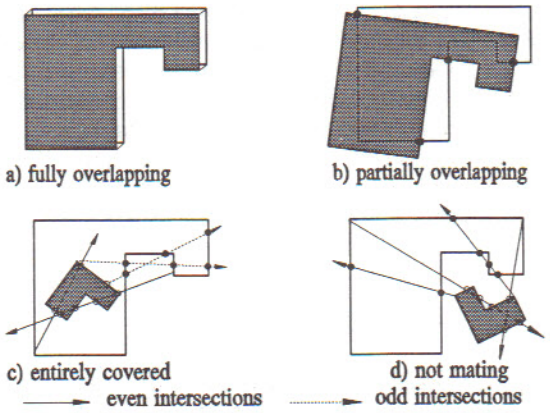


Figure 2 Three cases of mating faces

be fully overlapping - this is easily checked by matching the vertices of one face with those of the other (see Fig. 2a). Secondly, the faces could be partially overlapping. This is checked through the existence or otherwise of intersections between the edges of the two faces (see Fig. 2b). If the points of intersection lie within any edge of either face then the faces must be overlapping. The third case is that one of the candidate faces is entirely covered by the other face so that there would be no intersections within any edge of either face. It can be verified that the following procedure will simply and effectively detect such a case (see Fig. 2c):

i. Label one of the pair of candidate faces as the first face and the other as the second face.

ii. Draw a semi-infinite line from any one of the vertices of the first face, parallel to the face, such that it intersects at least one of the edges of the second face. Count the number of intersections, n_1, of this line with the boundary of the second face.

iii. Likewise, draw a semi-infinite line from any one of the vertices of the second face, parallel to the face, such that it intersects at least one of the edges of the first face. Count the number of intersections, n_2, of this line with the boundary of the first face.

v. If either n_1 or n_2 is odd then one of the faces is completely covered by the other face. (See Fig. 2d which illustartes the observation that when the two faces are disjoint, both n_1 and n_2 will be even.)

In order to simplify calculations however the two faces may be projected on to a convenient principal plane (a plane containing any two of the global coordinate axes) and the test line is taken along one of the principal axes in the principal plane. The following algorithm implements this strategy.

Algorithm CheckMating (f_1,f_2)

Input: face plane coefficients, vertices and edges of faces f_1 and f_2

Output: mating/ not mating

1. If f_1 and f_2 are not coplanar (i.e. the face plane coefficient sets of f_1 and f_2 are not identical or the normals to f_1 and f_2 have positive dot product) then return "not mating" and stop.
2. Select a global coordinate axis (X,Y or Z) which is not parallel to either f_1 or f_2.
3. Project f_1 and f_2 on a plane perpendicular to the selected axis - let f_{1pr} and f_{2pr} be the projected faces.
4. If
   - i. any three non-collinear vertices of f_{1pr} have identical vertices in f_{2pr}, or
   - ii. any edge of f_{1pr} intersects any edge of f_{2pr}within the boundaries of the edges, or
   - iii. a semi-infinite line drawn from an arbitrary vertex of f_{1pr} intersects the edges of f_{2pr} odd number of times, or
   - iv. a semi-infinite line drawn from an arbitrary vertex of f_{2pr} inetrsects the edges of f_{1pr} odd number of times

   then return "mating" else return "not mating".

FEASIBLE ASSEMBLY DIRECTIONS BETWEEN A PAIR OF MATING COMPONENTS

Woo has addressed the problem of determining the geometrically feasible assembly directions between a pair of mating components [Woo87]. However, his study was confined to the two dimensional (2D) domain. We now extend his approach to the three dimensional (3D) domain with particular reference to assemblies made up of polyhedral components. Further, as with Woo, we will restrict our discussion to linear assembly, i.e. when two potentially mating components are brought into contact by a single linear motion of one of the parts.

In his 2D analysis, Woo [Woo87] included several propositions and definitions. We will now reproduce those which are equally valid in the 3D domain and modify the others appropriately. Next we will present a new method for determining the feasible assembly directions in the 3D domain.

Clearly the following two propositions and one definition from Woo [Woo87] are equally valid in the 3D domain :

Proposition 1 :

Disassembly provides more structure than assembly - it is easier to disassemble than assemble.

Proposition 2 :

If the components are rigid, then disassembly is reversible. Thus, it is more convenient to determine the disassembly directions and sequences first and then reverse them to obtain the assembly directions and sequences respectively.

Definition 1 :

A component c is said to be linearly disassemblable (or just disassemblable, for short) from a subassembly S in direction **d** if, for all points x in c, a RAY (**x**,**d**), which is a semi-infinite line from x in direction **d**, does not intersect S.

(The following discussion concerning the 3D domain contains appropriate modifications of one further definition and a Lemma from Woo. In the interest of brevity the proofs for the 3D domain, which generally follow those given by Woo for the 2D domain, are not described below.)

The range of directions can be determined from the "mating faces", the sequence of faces shared by c in S. If a viewer can see (in the opposite direction of RAY (**x**,**d**)) all points x on the mating faces without having to go through S then c is removable. We may use the notion of a region in which the viewer should reside :

Definition 2 :

A *visible region* VR ($\{f_i\}$) into a sequence of faces $\{f_i\}$ is the intersection of half spaces $HS(f_i)$ induced by face f_i in the direction away from component c.

Lemma 1 :

A component c is disassemblable from a subassembly S if the visible region VR ($\{f_i\}$,c) is unbounded.

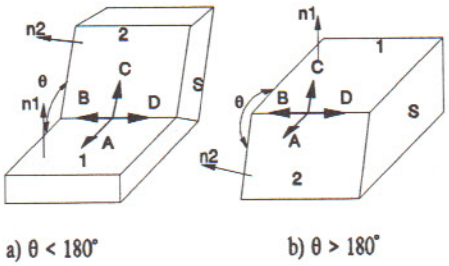
We now propose a method for determining the boundaries of VR for a given sequence of mating faces in the case of 3D polyhedral components.

It is clear that when mating (between c and S) occurs along a single face, the VR is the half space (generated by the mating face) that includes the normal vector, **n**, into c. Clearly, any disassembly direction **d** is feasible if its component along **n** does not oppose **n**, i.e. $\mathbf{d}*\mathbf{n} \geq 0$ (* : scalar product).

Consider now the situation when there are multiple mating faces whose planes intersect. In order to detrmine the VR, we need to intersect a multiple number of half spaces. As an alternative, we propose the notion of *Principal Disassembly Direction Set (PDDS)* which is characterised by the following :

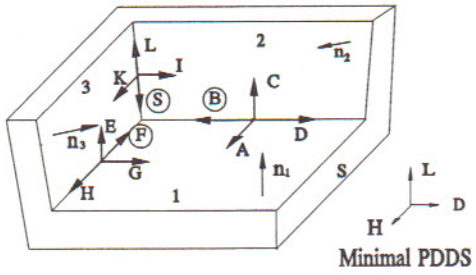
- i. A Principal Disaasembly Direction (PDD) is a unit direction vector representing a feasible dissembly direction lying on the boundary of VR.
- ii. A linear (positive) addition of one or more PDDs is a feasible disassembly direction.
- iii. A Principal Disassembly Direction Set (PDDS) is a set of PDDs that completely defines the VR; the "wire frame" formed by the PDDS is the VR.

In the interest of conceptual clarity we will present below an intuitive explanation of the proposed approach followed by a more rigorous statement of the resulting algorithms.



a) θ < 180° b) θ > 180°

Figure 3 The Case of Two Mating Faces



Minimal PDDS

Figure 4 The Case of Three Faces (All θ_s < 180 deg.)

Consider Fig. 3a which illustrates a case with two intersecting mating planes (1 and 2) between component c and subassembly S. $\mathbf{n}_1$ and $\mathbf{n}_2$ are the unit normal vectors (pointing towards the interior of c) of planes 1 and 2 respectively. Clearly any linear additive (positive) combination of $\mathbf{n}_1$ and $\mathbf{n}_2$ is a *feasible* disassembly direction. The feasibility of a given direction **d** can be tested by checking if $\mathbf{d}^*\mathbf{n}_1 \geq 0$ and $\mathbf{d}^*\mathbf{n}_2 \geq 0$. The union of all possible feasible assembly directions is the VR. The problem is that there are infinite number such directions. We need a simpler way of characterising the VR.

Figs 3a and 3b show a simple way of characterising the VR in two such cases in terms of a set of four unit vectors (A,B,C,D). B and D are directed parallel to the line of intersection between mating planes 1 and 2 but in mutually opposite directions. Clearly, disassembly is possible along B and D and they lie on the boundary of the VR. The other two direction vectors, namely A and C, are chosen parallel to planes 1 and 2 respectively while (i) both are perpendicular to the line of intersection of the mating planes and (ii) a motion along either of them does not have a component opposing $\mathbf{n}_1$ or $\mathbf{n}_2$. Condition (ii) is necessary to distinguish between the scenarios depicted in Figs 3a and 3b. In Fig. 3a the angle between planes 1 and 2 when measured inside c is less than 180° whereas in Fig. 3b it is greater than 180°. The directions of A and C should be taken such that $\mathbf{A}^*\mathbf{n}_2$ and $\mathbf{C}^*\mathbf{n}_1$ are both positive. It is intuitively evident that the "wire frame" formed by the PDD set {A,B,C,D} can completely represent the VR in this case. Further, each of these PDDs as well as any linear (positive) addition of one or more of these PDDs is a feasible disassembly direction. Hence, at this stage, the PDDS has only four PDDs (A,B,C,D).

Consider now the case when a third mating plane (plane 3) is included in the analysis as shown in Fig. 4. Now we have two additional pairs (1/3 and 2/3) of intersecting mating planes to consider. The addition of pair (1/3) demands the consideration of four new directions (E,F,G,H) as potential PDDs. Likewise the addition of pair (2/3) demands the consideration of four further directions (I,J,K,L). It is necessary now to ensure that the potential PDDs collected so far remain to be feasible disassembly directions despite the addition of plane 3. This may be effected by checking that each of the scalar products of each of the PDDs (A,B,,,L) with $\mathbf{n}_1$, $\mathbf{n}_2$ and $\mathbf{n}_3$ is ≥ 0. However, it may be noted that it is only necessary to check (A,B,C,D) against $\mathbf{n}_3$, (E,F,G,H) against $\mathbf{n}_2$ and (I,J,K,L) against $\mathbf{n}_1$ as a consequence of the way the PDDs have been defined. On this basis we note that B, F and J violate the feasibility condition. We thus reject B, F and J so that the updated PDDS now is {A,C,D,E,G,H,I,K,L}. The PDDS may be compressed by deleting identically directed elements. Further, since the final PDDS is intended to provide a "wire frame" representation of the VR, we may delete the PDDs which are coplanar with and take up an intermediate position between two other PDDs which are not collinear. Checking for coplanarity merely involves checking the scalar product of each PDD with each mating-plane-normal-vector. Likewise, checking for intermediate position only involves comparing the dot products between pairs of PDDs. The final PDDS thus obtained represents the Minimal PDDS (MPDDS) since it contains the minimum number of PDDs required to completely specify the VR. Further any linear positive addition of the PDDs is a feasible disassembly direction. At this stage it is also useful to define a Central Disassembly Direction (CDD) of the VR. The CDD may be determined by vectorially adding all the PDDs in the MPDDS and normalising the resulting vector.

The algorithm for generating the MPDDS for a pair of mating polyhedral components with N mating faces (when N > 1) is described as follows :

Algorithm GenerateMPDDS (c,S)

input: number of mating faces,N, between component c and subassembly S; unit normal vector, $\mathbf{n}_i$, of each mating face i : 1 to N.

output: MPDDS with the direction cosines of each PDD specified

1. form the normal vector set consisting of the N unit normal vectors.
2. compress normal vector set:
   delete greater than one instance of identical normal vector, **n**.

no. of intersecting mating planes N'= N-number of normal vectors deleted.

3. For each pair (i/j) of intersecting mating planes identify the four associated PDDs (d_1,d_2,d_3,d_4 - all unit direction vectors) and add them to the initial PDDS:

   Let d_1, d_2, d_3 and d_4 be the four initial PDDs resulting from (i,j).

   $\mathbf{d}_1 = \mathbf{n}_i \mathrm{x} \mathbf{n}_j$ (normalised) ,
   $\mathbf{d}_2 = -\mathbf{n}_i \mathrm{x} \mathbf{n}_j$ (normalised),
   $\mathbf{d}_3 = \mathbf{n}_i \mathrm{x} (\mathbf{d}_1$ or $\mathbf{d}_2)$ (normalised) such that $\mathbf{d}_3 * \mathbf{n}_j \geq 0$, and
   $\mathbf{d}_4 = \mathbf{n}_j \mathrm{x} (\mathbf{d}_1$ or $\mathbf{d}_2)$ (normalised) such that $\mathbf{d}_4 * \mathbf{n}_i \geq 0$.

   (x : vector product, * : scalar product)

4. Compress the initial PDDS to yield MPDDS:
   i. delete more than one instance of identical PDD,
   ii. delete any PDD which is coplanar with two other non-collinear PDDs and takes up an intermediate position between them:
   If $\mathbf{d}_i = -\mathbf{d}_j$ then $\mathbf{d}_i$ and $\mathbf{d}_j$ are collinear;
   If $\mathbf{d}_i \mathrm{x} \mathbf{d}_j$ (normalised) = $\mathbf{d}_i \mathrm{x} \mathbf{d}_k$ (normalised) then $\mathbf{d}_i$, $\mathbf{d}_j$ and $\mathbf{d}_k$ are coplanar;
   If $\arccos(\mathbf{d}_i * \mathbf{d}_j) < \arccos(\mathbf{d}_i * \mathbf{d}_k)$ then $\mathbf{d}_j$ is intermediate between $\mathbf{d}_i$ and $\mathbf{d}_k$.

5. The MPDDS may be graphically dispalyed adjacent to the product assembly display as an aid to the assembly planner.

It may be noted that an increase in the number of mating faces results in a proportional increase in the size of the initial PDDS. However, we can expect in general that this will also be accompanied by an increase in the orientational diversity of the mating faces so that many of the initial PDDs will have to be rejected because they violate the feasibility condition. This will, almost always, lead to an MPDDS of limited size.

ASSEMBLY SEQUENCE, ASSEMBLY TREE AND CRITERIA FOR DISASSEMBLABILITY

De Fazio and Whitney [DeFa87] have examined the problem of assembly sequence and like Woo [Woo87] have noted that, in general, there are many possible assembly sequences and that an assembly sequence can be taken as the reverse of the corresponding disassembly sequence. Further, the array of possible assembly sequences is determined by precedence relationships which require repeated answers to two questions : "What liaisons must be done prior to doing the ith liaison?" and "What liaisons must be left to be done after doing liaison i?". In [DeFa87] the answers to both these questions were expected to be provided by the user in an interactive mode. However, it turns out that in most situations the answers to these questions are mainly dependent upon geometric constraints. Woo [Wo87] has developed an elegant method for developing the disassembly tree (which, in a way, encodes the geometric precedence relationships) for problems in the 2D domain. This method is found to be equally applicable to the 3D domain. We therefore briefly describe Woo's method as follows (with minor modifications to suit the 3D domain).

Woo's Method - slightly modified :
Classify the faces on each component as *boundary faces or mating faces* : boundary faces only mate with the background *B*, i.e. they do not mate with other components. Create a *mating graph* (MG) of the assembly with the nodes representing components and the arcs representing mating; include the background *B* as a node. Begin constructing the assembly tree by taking the assembly as the root node. Next determine which components with boundary faces can be disassembled. Include these as the leaf nodes in the second level of the tree. Continue expanding the tree by retrieving the mates of each leaf node and testing the mates for disassemblability. From the mating graph, retrieve only the non-background mates for each of the new leaves. If a mate occurs elsewhere in the tree, discard it from further consideration. (Readers are directed to [Woo87] for a detailed explanation of this algorithm and evidence of its effectiveness.)

A requirement of the above algorithm is that we need to have robust criteria for deciding when a component is disassemblable. The first criterion in this regard is that the VR of the component with respect to the subassembly (at the particular instant in constructing the disassembly tree) is unbounded. This is easily established in the case of 3D polyhedral assemblies using the concept of MPDDS described in the previous section. If the MPDDS is empty then the VR is bounded and the component is not disassemblable in any direction. However, the existence of an unbounded VR is a necessary but not a sufficient condition for disassemblability. As an illustration consider the assembly (in 2D, for simplicity) in Fig. 5. We note that the VRs of components q and r are bounded. Hence they are not disassemblable in any direction at this stage. Components (p,s,t) have unbounded VRs but this does not mean that all of them are necessarily disassemblable. For instance, t mates with the background B and has a non-empty MPDDS consisting of direction **x** but t is not disassemblable along **x** from the rest of the assembly because a motion along **x** will be obstructed by p. Hence it is necessary to test the translational freedom of each component with unbounded VR before we can establish its disassemblabilty.

Another problem to be addressed is that the test for translational freedom needs to be carried out with respect to every possible direction within the VR. This is clearly not practical. Fortunately, Dini and Santochi [Dini92] have recently observed that, for problems in the 3D domain, it is enough to check for translational freedom in three mutually orthogonal directions. If there is translational freedom in any one of the chosen three directions then the component is disassemblable. This observation vastly simplifies the problem by limiting the number of checks on translational freedom to three test directions. However, this is not clearly true if the three test directions are arbitrarily chosen. For instance,

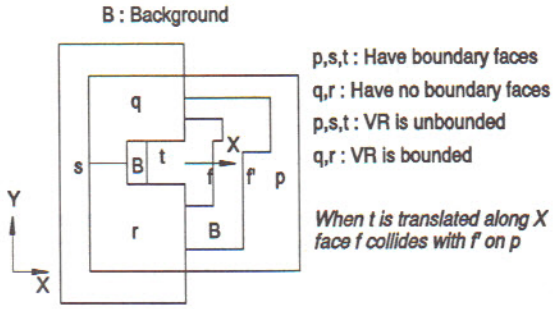


Figure 5 The Need for Checking Translational Freedom

component s in Fig. 5 has no translational freedom in direction other than -**x**. Hence the set of test directions must include -**x**. It is therefore proposed that the test direction set include the Central Disassembly Direction (CDD) of the VR and two other mutually orthogonal directions in a plane perpendicular to the CDD. Of course, if the MPDDS consists of only one PDD, it is adequate to use this PDD as the only test direction.

Having chosen a test direction, **d**, it is now necessary to examine the criteria for translational freedom in that direction. Dini and Santochi [Dini92] use CAD simulations for this purpose. Ko and Lee [Ko87] have used a similar approach using swept volumes. It is not clear however whether this was done automatically or through manual intervention. In the following, we describe an algorithm for automatically testing the disassemblability with regard to translational freedom in a given test direction **d** when the components are defined using Boundary Representation. The algorithm tries to identify which, if any, faces of component c and subassembly S are likely to collide when the former is translated along direction **d**. The approach is designed to minimise the number of faces to be tested for collision.

Algorithm CheckDisassemblability(c,S,**d**)
{c:component, S:subassembly, **d**: unit vector along test direction}
Input: face plane coefficient sets, vertices and edges of c and S; direction cosines of **d**.
Output: disassemblable / not disassemblable

1. i. Locate an infinite line L parallel to **d**.
   ii. Locate the point of intersection P_i of L with each face plane i of c.
   iii. Locate the point of intersection P_j of L with each face j of subassembly S.
   iv. For each pair of points (P_i,P_j), compute the displacement D_{ij} from P_i to P_j in the direction of **d**.
   v. Flag the faces from S which have negative D_{ij} with any face i of c and discard them from further consideration.
   vi. If $\min(D_{ij})=0$ then return "not disassemblable" and stop.
2. Identify faces $f_{i'}$ and $f_{j'}$ with minimum D_{ij}.
3. Project $f_{i'}$ and $f_{j'}$ on a plane perpendicular to **d** : let $f_{i'pr}$ and $f_{j'pr}$ be the projected faces.
4. Call algorithm CheckMating($f_{i'pr}$,$f_{j'pr}$). If result is "mating" then return "not disassemblable" and stop.
5. Flag faces $f_{i'}$ and $f_{j'}$ and prevent them from furrther consideration.
6. If no more face pairs (f_i,f_j) are left for consideration return "disassemblable" and stop.
7. Go to step 3.

CONCLUSION

Several algorithms have been proposed in the above to address the problem of automatically determining the assembly sequences in assemblies composed of 3D polyhedral components which are modelled using Boundary Representation. These include identifying the mating faces and examining the visible region and translational freedom. A new concept called Principal Disassembly Directions is proposed to facilitate the definition of VR in the 3D domain. Much of the computational work required by the algorithms is related to simple geometrical calculations and vector algebra. The algorithms are so structured as to disable consideration of entities (such as PDDs and faces) as soon as it is deemed that they have no bearing on further computations. These features facilitate rapid processing.

The broad strategy adopted is similar to that used in many previous works where disassembly directions and sequences are first computed and these are reversed to determine the assembly directions and sequences. A consequence of this is that the approach is applicable only when all components are rigid and only linear assembly is considered. Further, while the algorithms include geometric considerations, the technological constraints impinging on the problem have been left unaddressed. However, it is heartening to note that several recent works [Dini92] [Lape92] have attempted to address the technological constraints. It is hoped that the algorithms developed in the present work would be found useful in such 'larger' systems in the phases devoted to the consideration of geometric constraints. Lastly, most engineering assemblies involve components with atleast polyhedral and cylindrical features. However, extending the present work to include cylindrical features is believed to be not a major problem in view of the inherent symmetry of cylindrical features.

ACKNOWLEGEMENTS

The author wishes to acknowledge the technical support provided by his students Man C.H., Lo K., and Ming, Y.W. to the work reported in this paper. In particular, Ming had contributed significantly to the development of the concept of PDDS. Special thanks are due to the City Polytechnic of Hong Kong for providing the financial support for the larger project on DFA.

REFERENCES

[Boot83] Boothroyd, G., and Dewhurst, P., Design for Assembly, Dept. of Mechanical Engineering, University of Massachusetts, 1983.

[DeFa87] De Fazio, T.L., and Whitney, D.E.,"Simplified Generation of All Mechanical Assembly Sequences", IEEE Journal of Robotics and Automation, Vol. RA-3, No. 6, pp 640-658, 1987.

[Dini92] Dini, G., and Santochi, M., " Automated Sequencing and Subassembly Detection in Assembly Planning", Annals CIRP, Vol. 41, No. 1, pp 1-4, 1992.

[Ko 87] Ko, H., and Lee, K., "Automatic Assembly Procedure Generation from Mating Conditions", Computer-aided Design, Vol.19, No.1, pp 3-10, 1987.

[Lape92] Laperriere, L., and ElMaraghy, H.A., "Planning of Products Assembly and Disassembly, Annals of the CIRP, Vol. 41, No. 1, pp 5-9, 1992.

[Mile82] Miles, L.D., Techniques of Value Analysis and Engineering, 2nd Ed., Mcgraw-Hill, 1982.

[Miya86] Miyakawa, S., and Toshirjo, O., The Hitachi Assemblability Evaluation Method (AEM)", Int. Conf. Product Design for Assembly, Newport, RI, USA, 15-17 April,1987.

[Stur89] Sturges, R.H., "A Quantification of Manual Dexterity : The Design for assembly Calculator", J. Robotics and Computer Integrated Manufacture, Vol. 6, No. 3, pp 237-252, 1989.

[Woo87] Woo, T.C., "Automatic Disaseembly" Proc. SAE/EAD Int. Computer Graphics Conference, Michigan, pp 163-168, April 7-9, 1987.