International Journal of CAD/CAM Vol. 3, No. 2, pp. 85~95 (2003)

INTERNATIONAL JOURNAL OF CAD/CAM
www.ijcc.org

An Algorithm for Splitting a Box by a Loop and Its Applications in Manufacturing

Karuna P. Karunakaran*, Anoop Kheerwal, P. Vivekananda Shanmuganathan and Rohitashwa Shringi
Computer Graphics Laboratory, Department of Mechanical Engineering, Indian Institute of Technology, Powai, Bombay 400076, India

Abstract – During the design of dies and molds, the cavity of the object is obtained by subtracting it from a surrounding rectangular block. This box is subsequently split into two halves by the parting surface. Similar problems also occur in some RP processes such as LOM, SGC, SLS and 3DP where the machine produces a block inside which the prototype is buried. Determining the orientation of the object inside the box and the corresponding parting surface taking appropriate constraints into account have been addressed by several researchers. However, given the parting surface, the problem of splitting the box into a pair of halves has not received adequate attention; this is probably because it appears too simple. During the development of a software package called *OptiLOM* (now a module of an RP software Magics 8.0), the authors realized non-triviality of this problem since the loop can spread over as many as 5 faces of the box. In this paper, the authors have tried to bring out the importance of this problem and have presented their algorithm to solve it.

***Keywords*: Tool design, Rapid Prototyping (RP), Parting surface, Parting line, Parting direction**

1. Introduction

CNC machining or *Rapid Prototyping (RP)* can be employed when only a few pieces of an object are to be produced. When they are to be mass produced, processes such as injection molding, die casting and forging are used. These processes make use of a cavity whose geometry is the negative of the object with suitable shrinkage and machining allowances and drafts. When the raw material, which may be in liquid, semi-solid or powder form, is filled into this cavity and allowed to set, the object of the desired geometry and material is obtained. As one needs to take out the object from the cavity and reuse the cavity for making many copies of the object, the die/ mold box is made in two openable parts called *die/ mold halves*. Subsequently mounting and locating details are added to the two halves as shown in the Fig. 1. The surface at which both these halves split is called *parting surface*. The direction along which they separate is called *parting direction* (Fig. 1).

The orientation of the object inside the box is determined taking into account the factors such as the number undercuts, time for filling the mold etc. The criterions for selecting the suitable orientation for casting have been addressed by several researchers [1, 2]. The build orientation of RP parts is determined so as to minimize the error and the cost [3, 4]. When the part orientation is chosen, the pouring direction or parting direction also gets fixed. From the given parting direction, one can obtain the parting line and then parting surface. This problem also has been addressed by several researchers [5, 6, 7].

The parting surface is completely visible along the parting direction; otherwise, it cannot be called so. It consists of two loops. The inner loop lies on the object and the tool designers refer to it as *parting line*; it will be called *parting loop* in this paper. The outer loop lies on the box and it will be called *box loop* in this paper. Note that each of these loops need not be planar.

Traditionally, tool designers prefer to have a planar parting surface as far as possible for ease of machining on a conventional machine. However, as the objects become more and more complex, this is no longer

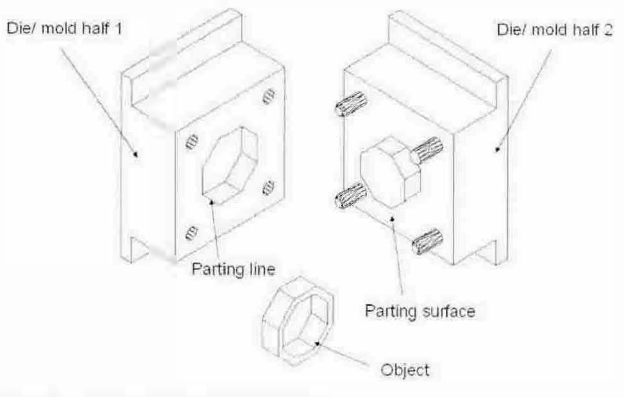


Fig. 1. A Pair of Die / Mold Halves.

*Corresponding author:
Tel: +91-22-2576-7530
Fax: +91-22-2572-6875
E-mail: karuna@me.iitb.ac.in

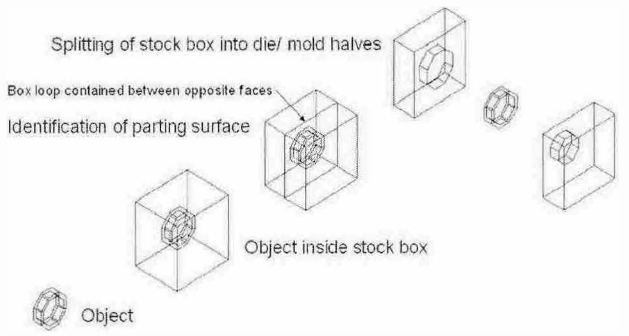

Splitting of stock box into die/ mold halves
Box loop contained between opposite faces
Identification of parting surface
Object inside stock box
Object

Fig. 2. Splitting of a mold box by a loop when the parting surface is simple.

possible. If so, they prefer to select the flattest *parting line* so as to keep the parting surface also as flat as possible [5]. In such an approach, the parting surface is very simple and it splits the box into almost two equal halves (Fig. 1). It is often trivial to split the die/ mold box using such a loop of the parting surface as illustrated in Fig. 2. As the parting surface and hence the box loop is contained within two opposite box faces without any wavy or spiky patterns, the box loop can be split into 4 face segments corresponding to each of the remaining four box faces. Each of these segments will split the corresponding box face into two parts, one belonging to die/ mold half 1 and the other belonging to die/ mold half 2. These halves will be called *stock halves* SH_1 and SH_2 in this paper.

It is no longer required to strive to keep the parting surfaces as flat as possible since CNC machines can handle all valid parting surfaces with equal ease. Therefore, the tool designer need not be constrained to select the flattest parting surface; he can instead concentrate on more important optimization criterions such as minimization of the area of the parting surface, perpendicularity of the parting surface with respect to the parting direction etc. In fact, there are situations where one would deliberately create a wavy parting surface. For instance, in the case of *RTV Silicon Rubber Molding*, the parting surface will have a wavy outer loop and a smoother inner loop [8]. This wavy pattern, rather than location pins, is used to locate the mold halves against each other (Fig. 3).

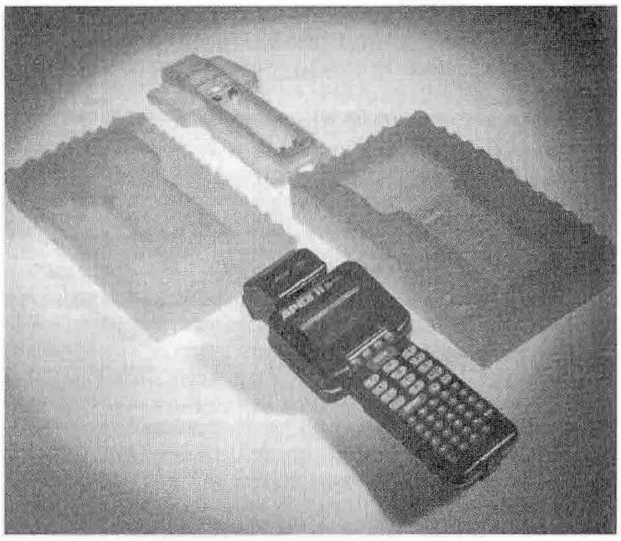


Fig. 3. Wavy parting surface of silicon rubber mold [8].

Laminated Object Manufacturing (LOM) is an RP process in which the objects are built by pasting paper laminates together. It follows a "paste-then-cut" approach wherein each rectangular laminate is pasted over the previous one and then the necessary part profiles are cut. The output of this process is a rectangular block with the required object trapped inside. Fig. 4 illustrates the LOM process developed by Helisys, USA. In order to enable extraction of the object, the remaining sheet

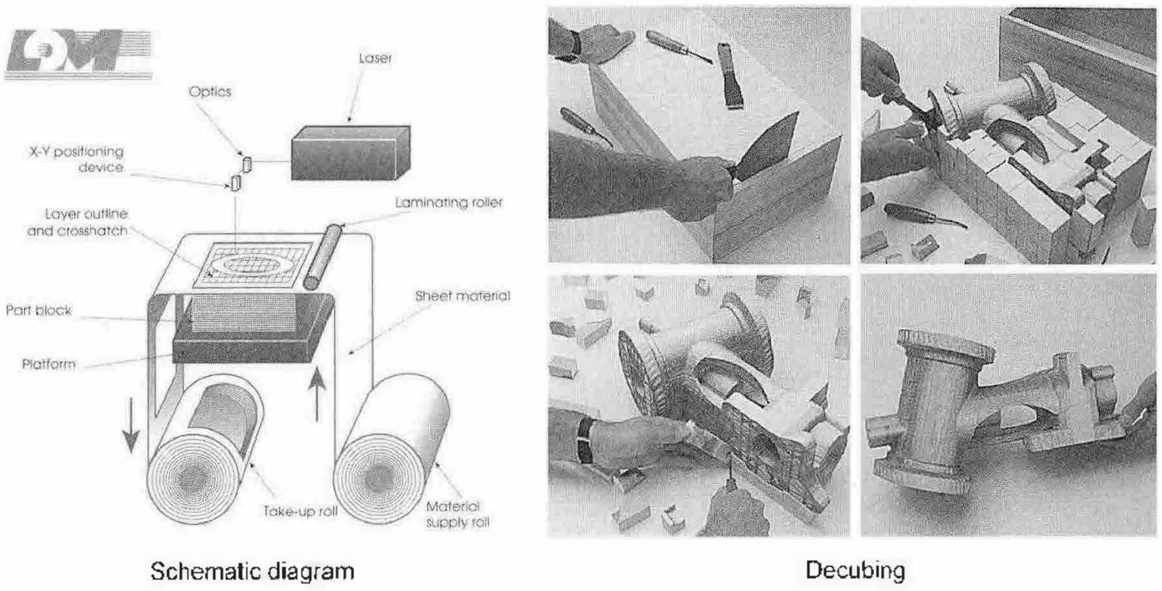
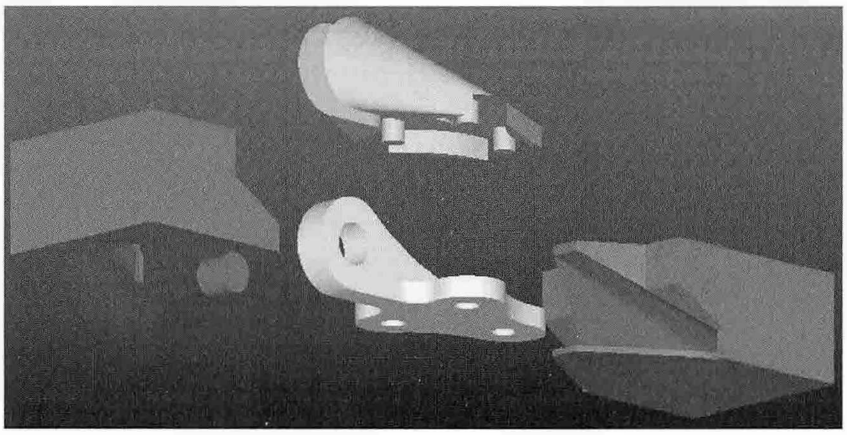


Fig. 4. Laminated object manufacturing [Courtesy Helisys Inc., USA].


(a) Output of a bracket processed by OptiLOM

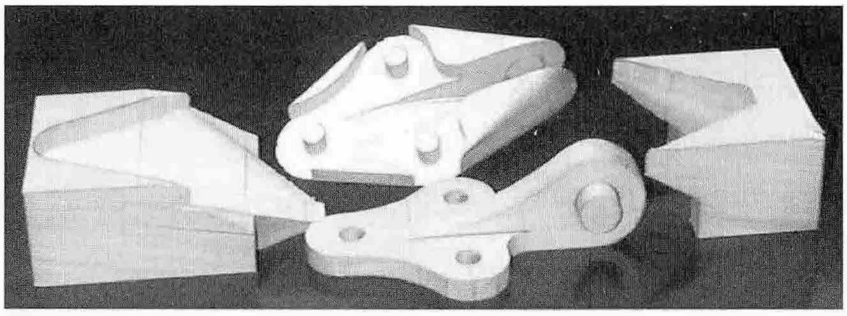

(b) Bracket made on a LOM machine after pre-processing using OptiLOM

Fig. 5. Building an object using OptiLOM.

in each layer is cut into square grids that grow into tiny tiles (see Fig. 4(a)). The operator 'decubes' or removes these tiles using sharp tools and extracts the object (see Fig. 4(b)). Making use of the remaining stock as support structure and cutting grids on it to enable extraction of the object are very innovative ideas in this 'paste-and-cut' approach. However, this method is very inefficient for two reasons: firstly, cutting efficiency is poor since laser spends most of its time in grid cutting; secondly, decubing takes several hours. The authors have developed an alternative stock cutting method which eliminates grid cutting and converts the elaborate decubing operation into a simple disassembly operation. They have implemented this method in a software module called *OptiLOM* that stands for "Optimization of LOM-RP" [9, 10]. OptiLOM is presently available as an optional module of a commercial RP package Magics [11].

Extraction of the object from the stock box has analogy with the extraction of casting from its mold - the conventional LOM and OptiLOM respectively being analogous to sand casting and permanent mold casting processes. In OptiLOM, rather than fragmenting the remaining stock into tiny tiles, it is segmented into two stock halves that open about a parting surface of minimum area. This optimal parting surface is obtained for the *convex hull* of the object, rather than for the object itself, due to its complete visibility along any pair opposite directions. The convex hull is further segmented into the object and plugs. The plugs are so shaped that they do not get entangled inside the *concave* or 'undercut' regions of the object. The plugs whose drawing directions coincide with the opening direction of the stock halves are merged with the corresponding stock halves. The object, all the plugs and both the stock halves form the stock block. All these pieces are made together in the LOM machine. For disassembly, first the convex hull will be extracted by opening the stock halves. Subsequently the plugs that fill the concave portions of the object will be extracted from the convex hull. Thus, OptiLOM eliminates grid cutting and decubing resulting in drastic reduction in prototyping time and improved quality of the prototype. Making a bracket using OptiOM is illustrated in Fig. 5. OptiLOM essentially is a preprocessor to the LOM machine; it accepts the STL file of the object as input and outputs the STL file of the object as well as those of the stock halves and plugs. More detailed description of OptiLOM can be found in [10].

There is, however, a major difference between the two applications of the present problem, viz., die/ mold design and OptiLOM. In die/ mold, the pair of opposite parting directions always coincides with the normals of one of the pairs of opposite faces of the box. On the contrary, the parting direction could be quite arbitrary in OptiLOM. Therefore, the authors encountered various combinations of the parting surfaces in OptiLOM which are not very common with dies/ molds. Therefore, they had to develop the most general method of splitting the box using the outer loop of the parting surface to handle all possible cases.

More recently they found another application for the second part of the methodology, viz., splitting a box face loop into segment loops, in Boolean operators for polyhedral objects. In these Boolean operations, it is required to break large triangles into several loops. In this application, the loop referred as "box face loop" in this paper is not rectangular initially but is a triangle of one operand. The face segments come from the intersection of this triangle with the triangles of the other operand. The authors feel very strongly that the methodology proposed in this paper can be used with suitable modifications in many other applications.

Having presented the important applications, the problem will be defined in detail. This will be followed by a detailed description of the algorithms and illustrations.

2. Problem Definition

A box loop *BL* is lying on a stock box *SB*. It is required to split the stock box *SB* into two topologically valid parts about this loop *BL* so that they can be opened along the parting directions ($+\hat{d}, -\hat{d}$) as shown in Fig. 6. The box as well as the conventions to denote its vertices, edges and faces is shown in Fig. 7. The stock box has its opposite faces parallel to XY, YZ and ZX planes. The boundary loops of the six sides of the box, called *Box Face Loops* $BFL_i (i \in [0-5])$, are denoted in terms of the box vertices as given in Table 1 such that they satisfy the right hand thumb rule w.r.t. the respective outward normal.

The box loop *BL* has the following characteristics:

- The loop lies on the box.
  It will not have knots. This is because it is the outer boundary of the parting surface and a parting surface cannot have folds.
- The number of the box faces the box loop can lie on can vary from 1 to 5. Some of the possible cases are shown in Fig. 8(a-e). The result of splitting the box into a pair of stock halves can be seen for each case in the right hand side of Fig. 8.
- The box loop can pass through a corner of the box B or cut its edge.
- The box loop may have some part of it coincident with an edge of the box.
- The box loop is assumed to be a 3D polygon.
- Box loop can have spikes. Spikes or saw tooth shapes on the box loop can occur when the triangles

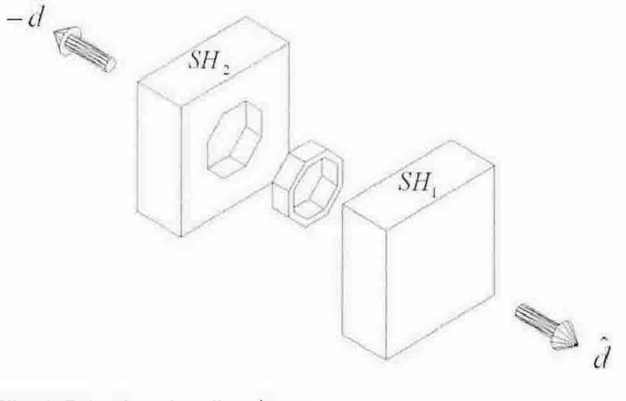


Fig. 6. Pair of parting directions.

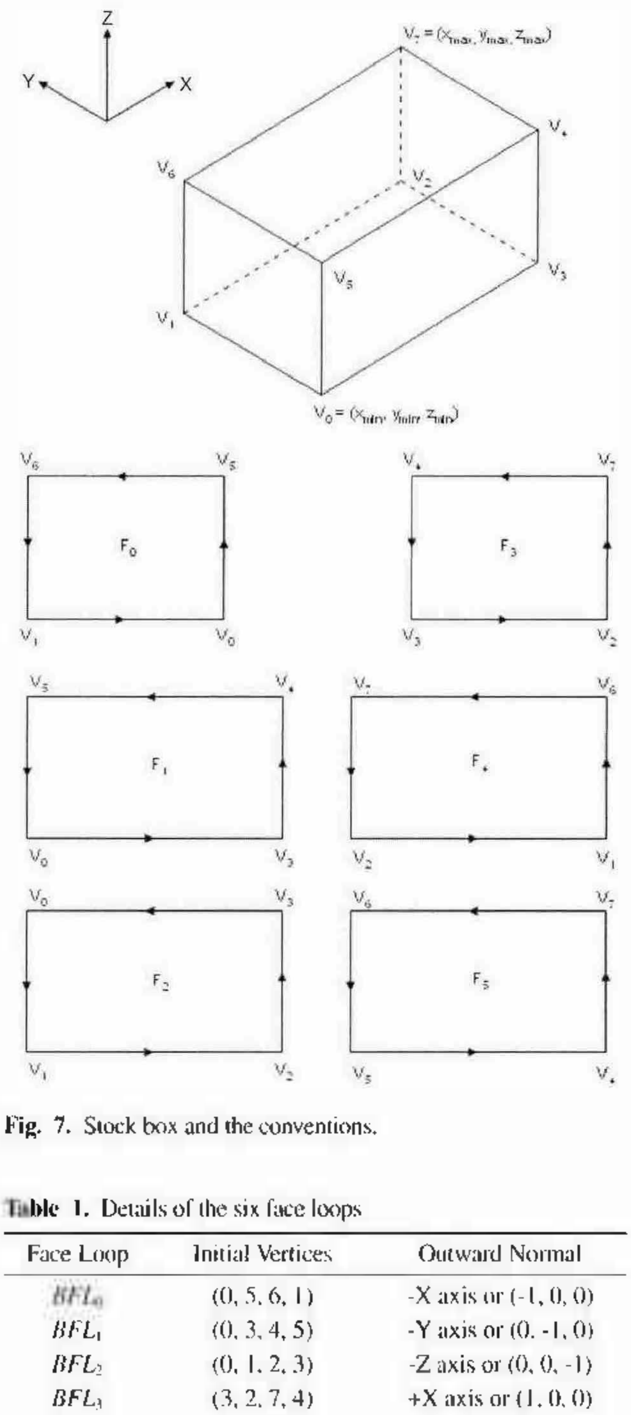
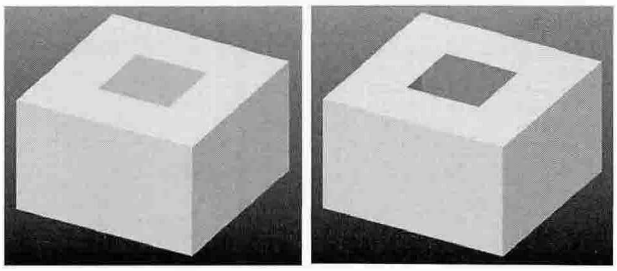
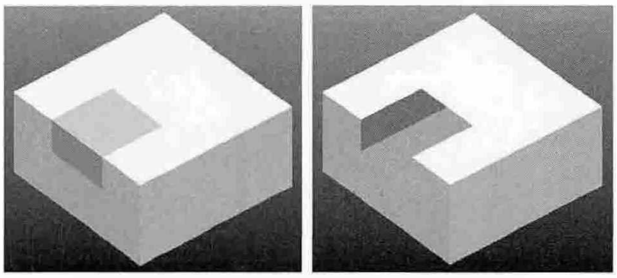

Z
Y
X
$V_7 = (x_{max}, y_{max}, z_{max})$
V_4
V_6
V_2
V_3
V_5
V_1
$V_0 = (x_{min}, y_{min}, z_{min})$
V_6 V_5 F_0 V_1 V_0
V_4 V_7 F_3 V_3 V_2
V_5 V_4 F_1 V_0 V_3
V_7 V_6 F_4 V_2 V_1
V_0 V_3 F_2 V_1 V_2
V_6 V_7 F_5 V_5 V_4

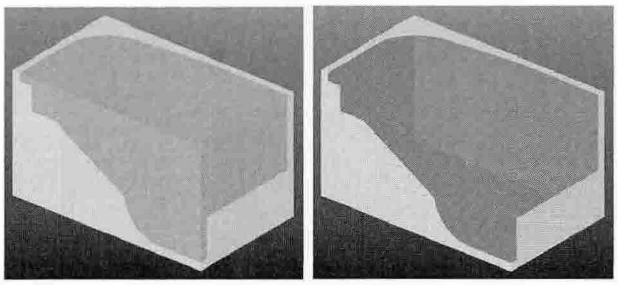
Fig. 7. Stock box and the conventions.

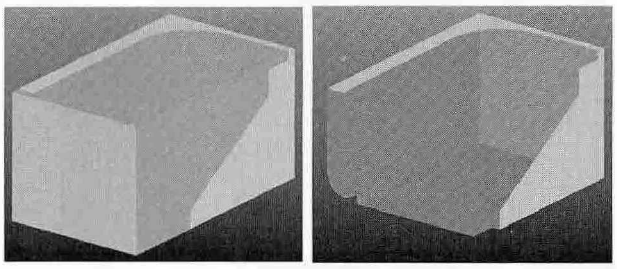
Table 1. Details of the six face loops

Face Loop	Initial Vertices	Outward Normal
BFL_0	(0, 5, 6, 1)	-X axis or (-1, 0, 0)
BFL_1	(0, 3, 4, 5)	-Y axis or (0, -1, 0)
BFL_2	(0, 1, 2, 3)	-Z axis or (0, 0, -1)
BFL_3	(3, 2, 7, 4)	+X axis or (1, 0, 0)
BFL_4	(1, 6, 7, 2)	+Y axis or (0, 1, 0)
BFL_5	(4, 7, 6, 5)	+Z axis or (0, 0, 1)


(a) Box loop lying on one face


(b) Box loop lying on two faces


(c) Box loop lying on three faces


(d) Box loop lying on four faces

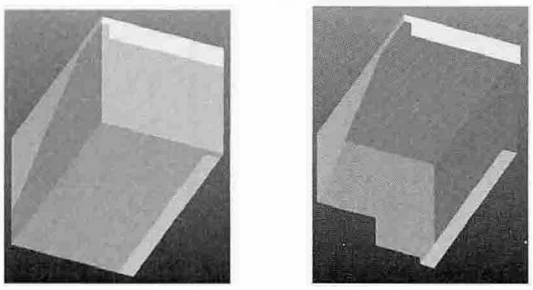
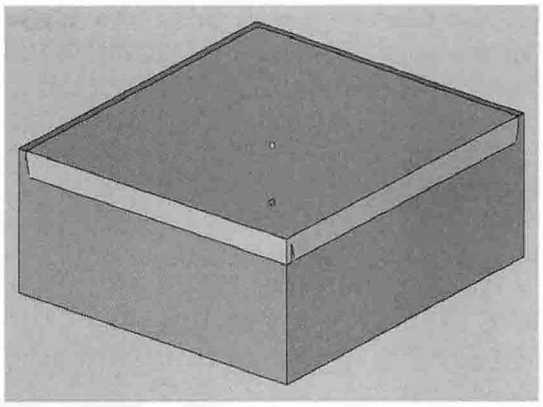

(e) Box loop lying on five faces

Fig. 8. Various cases of the box loop.

defining the object have high aspect ratios. Fig. 9(a) shows SH_1 and SH_2 with a short spike. Under extreme conditions, there could be more spikes which could run across the stock face as shown in Fig. 9(b). This leads to a spiked parting surface that is difficult disassemble. Therefore, this condition has been bypassed.


(a) A small spike

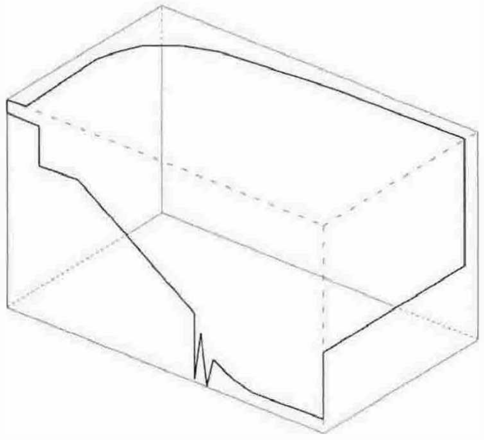

(b) Spikes running across the stock face

Fig. 9. Spikes in the box loop.

3. Methodology

The methodology consists of two distinct parts. In the first part, the box loop is divided into face segments. Thus, each box face F_i will be associated with a set of face segments $\{FS_{ij}\}$ and the box face loop BFL_i. BFL_i will be initialized with the four vertices of F_i as given in Table 1. In the second part, each of the face segments FS_{ij} is combined with an appropriate portion of the box face loop BFL_i to form the segment loops $\{SL_k\}$. Every FS_{ij} gives rise to one and only one segment loop. The following are the steps for splitting the box into the pair of stock halves:

- Adding faces not cut by the *box loop* to stock half 1 or 2 appropriately
- Splitting the *box loop* into *face segments*
- Constructing *box face loops* by inserting the end points of all the face segments of the face
- Identifying *segment loops* of each box face
- Identifying the stock half for each *segment loop* and adding it after triangulation.

3.1. Adding faces not cut by the box loop to stock half 1 or 2 appropriately

As the box loop can spread to a maximum of 5 box faces, there will be one or more box faces which will not be touched by the box loop. Such box faces can be converted into a pair of triangular faces and added to SH_1 or SH_2 according to the visibility criteria. If the box face is visible along the parting direction $+\vec{d}$, it is added to the SH_1; otherwise to SH_2. If the dot product $\vec{d} \cdot \hat{n} > 0$ (where $\hat{n}$ is the outward normal of the box face), the box face is visible. The pseudo-code for the same is given in Algorithm 1.

Algorithm 1 Adding box faces untouched by the box loop to stock half 1 or 2 appropriately

```
For each box face $F_i$
  If number of face segments is zero
  {
    Triangulate box face;
    If $\vec{d}.\hat{n} > 0$
        add both these triangles to SH$_1$;
    Else add them to SH$_2$;
  }
```

3.2. Splitting the box loop into face segments

Without loss of generality, let the stock half shown in Fig. 10 be SH_2. A typical box loop along with its direction is as shown in the figure. The first vertex of the box loop could be any of its vertices. Accordingly, the first vertex of the loop could lie either inside a box face (Fig. 11(a)), on an edge of the stock box (Fig.

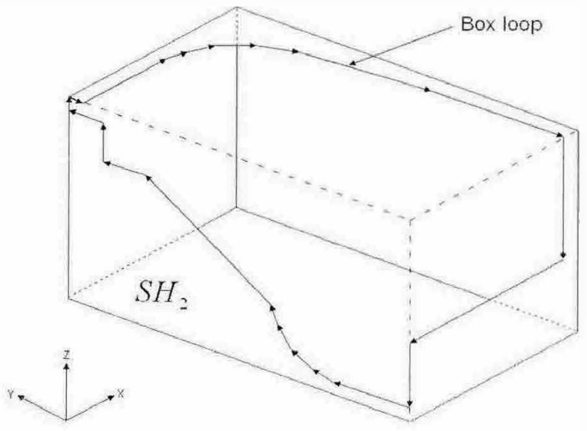
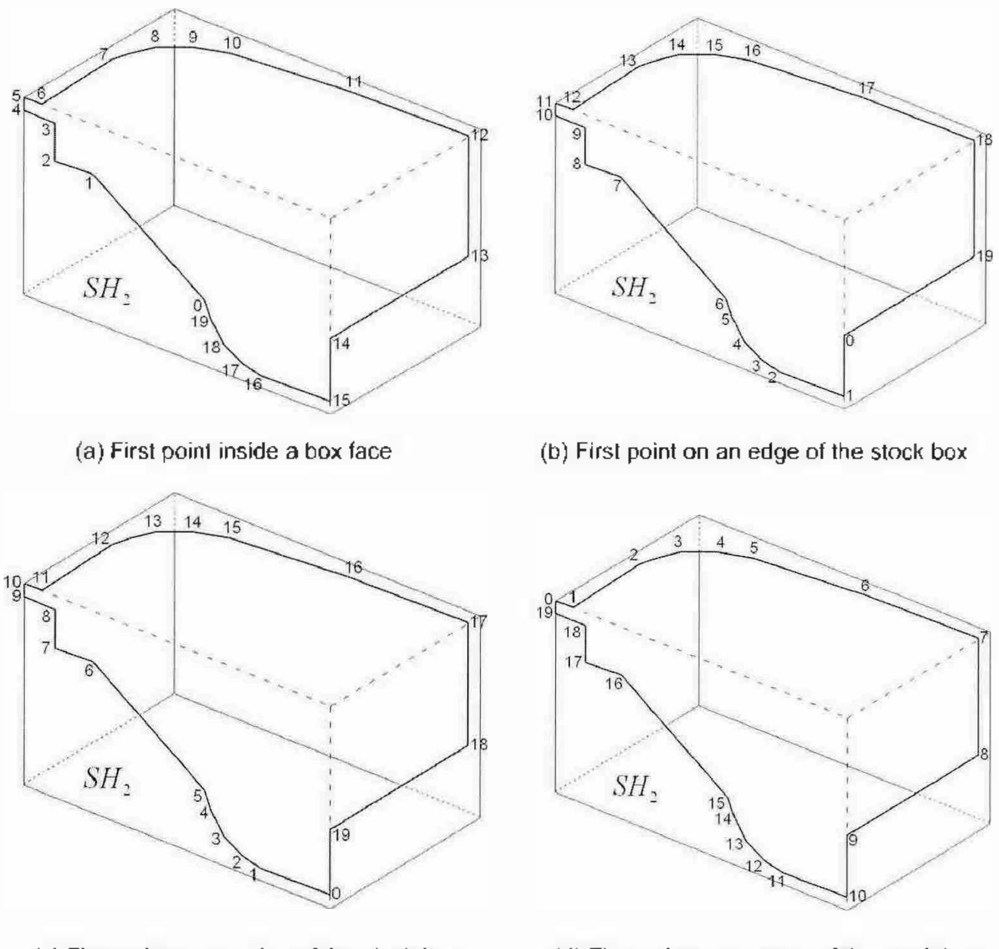


Fig. 10. A box loop.


SH_2
(a) First point inside a box face
(b) First point on an edge of the stock box
(c) First point on an edge of the stock box
(d) First point on a vertex of the stock box
Fig. 11. Various possibilities of the first point in box loop.

11(b) & (c)) or on a vertex of the stock box (Fig. 11(d)).

It is required to identify the segments of the box loop belonging to each box face. These segments will be referred as *face segments*. Each face segment will be uniquely identified with a box face. It will have its start and end vertices on an edge or vertex of the stock box and all its other vertices will lie inside a box face. In order to collect these face segments one after the other, it is preferable to have the first vertex of the box loop satisfy the following conditions:

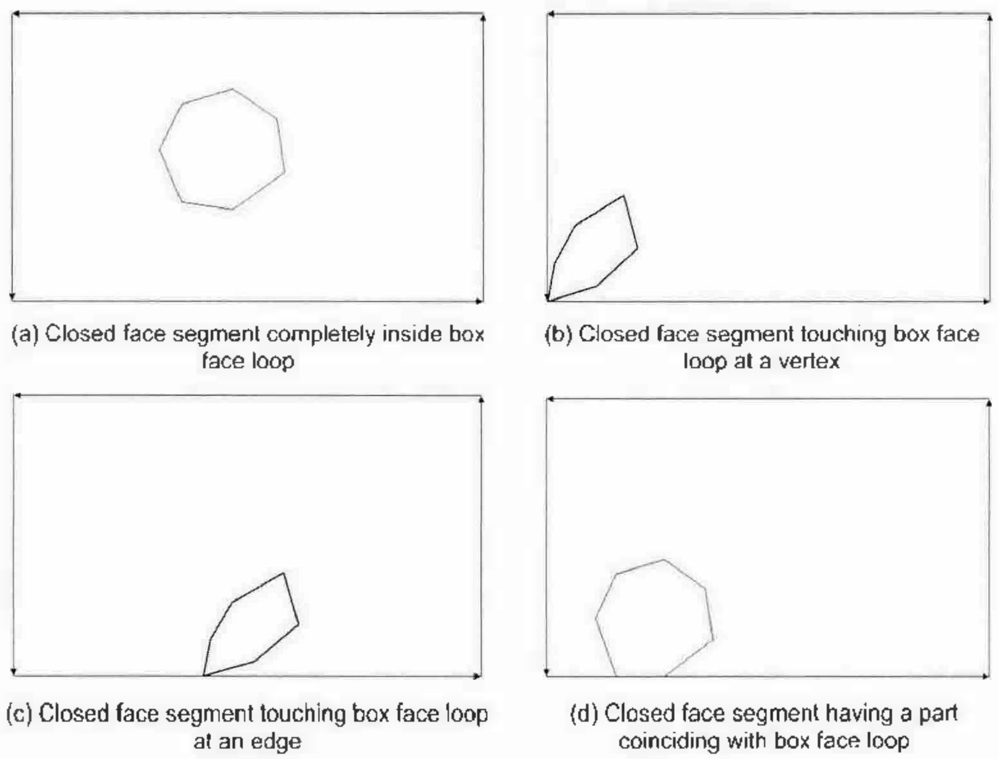
- The first vertex of the box loop shall lie on an edge or vertex of the stock box
- Its second vertex shall lie inside a face of the stock box or at any other edge.

If the above conditions are satisfied, then, the first vertex of the box loop itself will be the first vertex of a face segment. Thus, all vertices of the box loop from the first vertex till the vertex that lies on an edge or vertex of the stock box will be the face segment. Therefore, it is desirable to have the first vertex of box loop as 6, 1, 0 and 1 for the cases shown in Figs. 11(a), (b), (c) and (d) respectively. This is achieved by cyclically shifting the vertices of the box loop till the first vertex satisfies the above two conditions. Then, a face segment is created by copying the points from the box loop till an edge or vertex point is reached. Algorithm 2 describes this procedure. The first part of this algorithm does the cyclic shifting of the vertices and the second part extracts the face segment. This algorithm will be called repeatedly till the entire box loop is divided into face segments.

This algorithm will be called repeatedly till all the face segments are separated from the box loop, i.e., till no more vertices are left in the box loop. Note that each face segment belongs to a unique box face and each box face may have more than one face segment. For instance, box face F_0 will have 3 face segments for the case shown in Fig. 9(b). Therefore, any face segment

Algorithm 2 Separating a face segment from the box loop

```
// Cyclically shifting the vertices of box loop :
//―――――――――――――――――――――――――
If all vertices of the box face have been ticked,
   Exit ;
While (( first vertex of the box loop has been ticked ) or
        ( first vertex of the box loop is neither on an edge nor on a vertex
         of stock box ) or
        ( first and second vertices lie on the same edge ))
   Cyclically shift all vertices of box face by one place ;

// Moving vertices from box loop to face segment :
//―――――――――――――――――――――――――
i = 0 ;
while (true)
{
   Add the i-th vertex of the box loop to the face segment ;
   If (the i-th vertex is either on an edge or a vertex of the stock box)
      break the while loop;
   If (i  != 0) Tick that vertex for having used ;
   i = i + 1 ;
}
```


(a) Closed face segment completely inside box face loop

(b) Closed face segment touching box face loop at a vertex

(c) Closed face segment touching box face loop at an edge

(d) Closed face segment having a part coinciding with box face loop

Fig. 12. Three possible cases of closed face segments.

will be represented with two subscripts as FS_{ij} where i refers to the face it belongs to and j refers to its count in that face.

It may be further noted that the face segments are open curves most of the time. The end points of any open face segment could be on a vertex or edge of the box face. Accordingly there could be several possible cases for the open face segments.

When the box loop is contained within a face, it degenerates into a single closed face segment. Four possible cases of closed face segment occur as shown in Fig. 12. The closed face segment shown in Fig. 12(a) is completely inside the face. In Figs. 12(b) & (c), the closed face segment touches the face loop at one vertex or edge. The closed face segment of Fig. 12(d) has a part of it coinciding with the face loop.

If the *face segment* is a closed one, it is added to SH_1 or SH_2 according to the visibility criteria. The remaining part of its face is added to the other stock half.

3.3. Constructing box face loops

All the 6 box faces are initialized by the four vertices each as shown in Table 1. Each face F_i has an array of face segments FS_{ij}. For instance, box face F_0 of Fig. 9(b) has three face segments, viz., AB, BC and CD as shown in Fig. 13. The pair of end points of each face segment FS_{ij} is inserted into the box face loop BFL_i at the corresponding edge so as to maintain the cyclic nature. The same is illustrated in Fig. 14 for box face F_0 of Fig. 9(b).

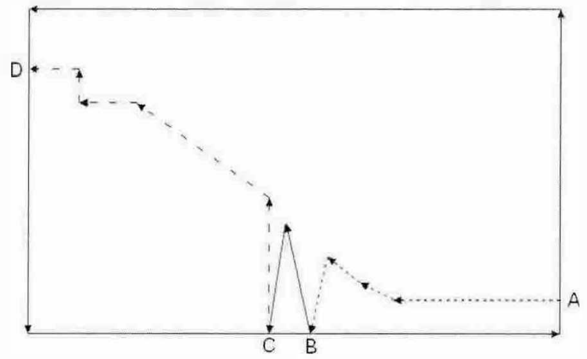


Fig. 13. Box face F_0 with 3 face segments AB ≡ FS_{00}, BC ≡ FS_{01} and CD ≡ FS_{02}.

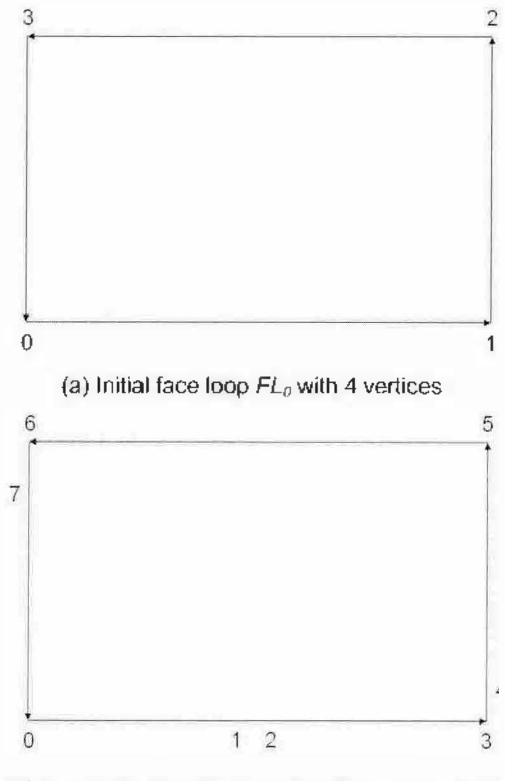

3
2
0
1
(a) Initial face loop FL_0 with 4 vertices
6
5
7
0
1 2
3
(b) Face loop FL_0 after inserting the end vertices of the face segments

Fig. 14. Construction of a face loop.

3.4. Identifying segment loops of a face

At this juncture, the face segments for each of the six box faces have been identified and each of the six box face loops includes the end points of the corresponding face segments. With this information, it is possible to divide the box face loop into a set of segment loops. Let $FS_{ij}(j \in [0-(n_i-1)])$ be the face segments of i^th box face F_i and n_i is its number of face segments. Let BFL_i be the corresponding box face loop. All these face segments will be open. The procedure to deal with the closed face segment has already been discussed in Section 3.2. The procedure to divide the box face loop BFL_i into a set of segment loops SL_{ij} using the open face segments will be explained here.

Table 2. Comparison of splitting the box loop into face segments and forming face loop from a face segment

Splitting the Box Loop into Face Segments	Forming Face Loop from Face Segment
Input loop is box loop.	Inputs are face segment and box face loop.
The outputs are the face segments.	The output is the face loop.
Vertices of the box loop are cyclically shifted so as to get the desirable characteristics of the first two vertices.	Vertices of the box face loop are cyclically shifted so as to get the desirable characteristics of the first vertex.
The desirable characteristics are that • the first vertex shall be on an edge or vertex of the stock box and • the first and second vertices shall not lie on the same edge of the box face.	The desirable characteristic is that the end points of the same face segment shall be encountered first when traversed from the first vertex of the box face loop in both the directions.

Each open face segment will make a closed loop along with a part of the corresponding box face loop. Subsequently, this loop will be added to SH_1 or SH_2 according to the criteria discussed later in Section 3.5. When such loops are added to SH_1 or SH_2, the corresponding face segment is deleted from the box face and the box face loop will be updated by subtracting the region corresponding to the segment loop.

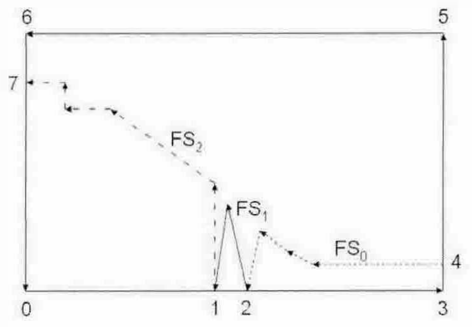
Splitting the box loop into face segments and forming segment loops from face segments can be thought of inverse to each other or duals. A loop is split into open segments in the former case and a loop is formed from an open segment in the latter case. So the same approach of moving cyclically the vertices of the loop will be employed here too; the loop is box loop in the former case and face loop in the latter case. However, the criterion to be satisfied by the first point is different in both cases. In the latter case, when marched from the first vertex on both the directions, the first encountered segment shall be the same. A comparison between these duals is presented in Table 2.

When the first vertex is adjusted to meet the desired condition, the corresponding face segment will break the box face loop into two parts. The part of the box face loop that contains the first vertex and the face segment will form a loop which will be referred as *segment loop*. Each segment loop will have a flag called 'Is_Reversed' which is 'false' by default. The direction of the segment loop will be maintained to be same as that of the box face loop. If it is required to reverse the face segment in order to maintain this direction, then the 'Is_Reversed' flag will be set to 'true'. After forming the segment loop, the vertices of the box face loop between the face segment and the first vertex (inclusive of the first vertex) are deleted and the vertices of the face segment are added to the face loop while maintaining the original direction of the face loop. Finally the face segment is deleted from the face. This procedure is given in Algorithm 3.

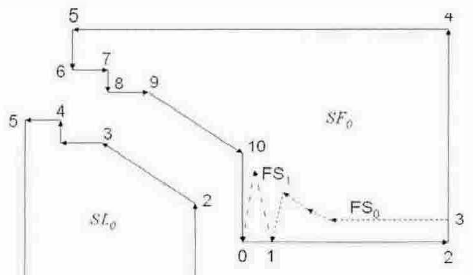
The sequence of forming the segment loops and updating the face loop are shown in Fig. 15. The initial box face loop shown in Fig. 15(a) is such that the face segment $FS_{3,0}$ is encountered when traversed from the first vertex. Therefore, the vertices of this box face loop do not require any cyclic shift. The first segment loop $SL_{0,0}$ is formed and the box face loop is updated as shown in Fig. 15(b). Figs. 15(c) and (d) show the formation of second and third segment loops and the corresponding updated box face loops. The vertices of the box face loop do not require cyclic shift in these two cases as well. As all face segments have been exhausted, the remaining box face loop is added as the

Algorithm 3 Forming a segment loop from the face segment and box face loop

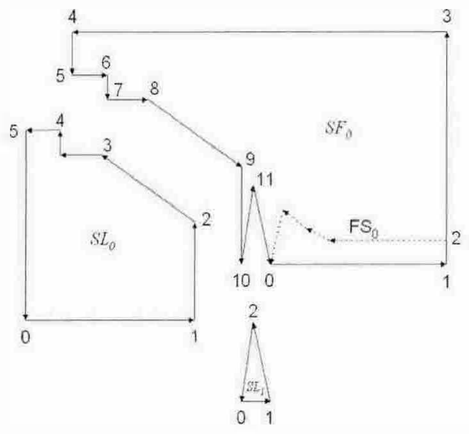
```
// Ensuring that first vertex of box face loop is such that the first
// encountered face segment is the same when marched from its first vertex
// in both directions :
// -----------------------------------------------------------------------
• While (number of face segments of the face > 0)
  {
      • Traverse the box face loop in the forward direction till an end point of any face segment is encountered ; Let this segment be $FS_{ij}$
        and the corresponding vertex of the box face loop be m ;
      • Traverse the box face loop in the reverse direction till an end point of any face segment is encountered; Let this segment be $FS_{ik}$
        and the corresponding vertex of the box face loop be n ;
      • If ( j==k )
          break the while loop ;
        Else
          cyclically shift all vertices of box face by one place;
  }

// Forming the segment loop :
// -------------------------
• Add the vertices of the box face loop from m-1 to 0 to the segment loop ;
• If (the i-th vertex is the last vertex of the face segment $FS_{ik}$)
  {
      Reverse the face segment ;
      Is_Reversed = true ;
  }
• Add all the vertices of the face segment to the segment loop ;
• Add the vertices of the box face loop from 0 to n-1 to the segment loop ;

// Updating the box face loop:
// ---------------------------
• Delete the vertices of the box face loop from m till the end ;
• Delete the vertices of the box face loop from 0 to n ;
• Reverse the face segment ;
• Add the vertices of the face segment to box face loop ;
```



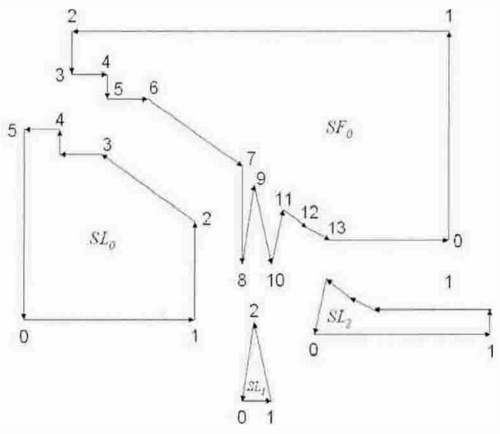
(a) Initial status



(b) Separation of the first segment loop and updation of the face loop



(a) Separation of the second segment loop and updation of the face loop



(b) Separation of the third segment loop and updation of the face loop

Fig. 15. Formation of a segment loops.

fourth segment loop. This way, the face loop is broken into 4 segment loops.

3.5. Identifying the stock half for each segment loop

The next step is to identify the stock half to which each segment loop belongs. As a flag 'Is_Reversed' is associated with each segment loop, this task is very straight forward. If Is_Reversed = false, the segment loop is added to stock half SH_1; otherwise, it is added to stock half SH_2. Finally the face segment should be triangulated and added to the appropriate stock half.

3. Conclusions

In tool design and some RP applications, it is required split a block surrounding an object into two parts to enable the extraction of the object. While identification of optimal orientation and calculation of parting surface have been researched fairly well, separating the boundary of the box into topologically valid halves has not received adequate attention. While this task was not very complicated in tool design, the RP applications encountered by the authors could create the parting surface spreading anywhere from one to five faces of the box, thus making the splitting reasonably complex. The methodology developed and successfully implemented by them has been presented in this paper. More recently they found another application for the second part of the methodology in Boolean operators for polyhedral objects. The authors feel very strongly that the methodology proposed in this paper can be used with suitable modifications in many other applications.

Nomenclature

Stock box (B)	- The box surrounding the object.
Box face (F_i)	- Rectangular face of the box. The stock box will have 6 such box faces.
Stock halves (SH_1 and SH_2)	- These are obtained by splitting the stock box by the parting surface.
Parting direction ($(+\hat{d}, -\hat{d})$)	- This is a pair of opposite directions along which the stock halves SH_1 and SH_2 open.
Parting surface (PS)	- The surface that splits the stock box B into stock halves SH_1 and SH_2. This will have two loops, viz., parting loop and box loop.
Parting loop (PL)	- This is the inner loop of the parting surface. Tool designers refer to it as "parting line". This need not be planar.
Box loop (BL)	- This is the outer loop of the parting surface. It lies on the stock box. This

	too need not be planar.
Box face loop (BFL_i)	- Loop defining the boundary of a stock face. BFL_i will be initialized with the four vertices of the corresponding stock face. However, subsequently the ends of its face segments also will be inserted. This also keeps getting updated and hence may not remain rectangular till the end.
Face segment (FS_{ij})	- This is the part of a box loop belonging to a box face. FS_{ij} is the j-th face segment of the i-th face.
Segment loop (SL_{ij})	- One segment loop will be prepared from each face segment FS_{ij}. Each of these loops will be triangulated and added to either SH_1 or SH_2.
Is_Reversed	- This is a flag which is set to true of the direction of the face segment is reversed while forming the segment loop. If this flag is false, the segment loop will be added to SH_1; otherwise, it will be added to SH_2.

References

[1] Chen, L. L., Chou, S. Y. and Woo, T. C. (1993), "Parting directions for mould and die design", *Computer-Aided Des.*, **25**(12), 762-768.

[2] Chen, L. L., Chou, S. Y. and Woo, T. C. (1995), "Partial visibility for selecting a parting direction in mold and die design", *Journal of Manufacturing Systems*, **14**(5), 319-320.

[3] Alexander, P., Seth, A. and Dutta, D. (1998), "Part orientation and build cost determination in layered manufacturing", *Computer-Aided Design*, **30**(5), 343-356.

[4] Rattanawong, W., Masood, S. H. and Iovenitti, P. (2001), "A volumetric approach to part-build orientations in rapid prototyping," *Journal of Materials Processing Technology*, **119**(1-3), 348-353.

[5] Majni, J., Gupta, P. and Janardan, R. (1996), "Computing a flattest undercut-free parting line for a convex polyhedron with application to mold design", *FCRC 96 Workshop* Springer-Verlag, 121-132.

[6] Hui, K. C. (1997), "Geometric aspects of the mouldability of parts", *Computer-Aided Design*, **29**(3), 197-208.

[7] Fu, M. W., Nee, A. Y. C. and Fuh, J. Y. H. (2002), "The application of surface visibility and moldability to parting line generation", *Computer-Aided Design*, **34**(6), 469-480.

[8] Hilton, P. D. and Jacobs, P. F. (Ed.), (2000), Rapid Tooling: "Technologies and Industrial Applications", Marcel Dekker, 51-94.

[9] Karunakaran, K. P., Shivmurthy Dibbi, P. Vivekananda Shanmuganathan, D. Satyanarayana Raju and Srinivasarao Kakaraparti (2000), "Optimal Stock Removal in LOM-RP", *I MECH E Journal of Engineering Manufacture*, **214**(Part B), 947-951.

[10] Karunakaran, K. P., Shivmurthy Dibbi, P. Vivekananda Shanmuganathan, Srinivasarao Kakaraparti and D. Sathyanarayana Raju (April 2002), "Efficient Stock Cutting in Laminated Manufacturing", *Computer-Aided Design*, **34**(4), 281-298.

[11] Materialise, N. V., Leuven (Belgium), www.materialise.be

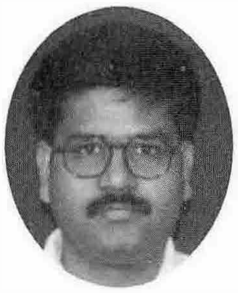
Karuna P. Karunakaran is an Associate Professor in Mechanical Engineering at IIT Bombay. He received his B.E. Hons. in Mechanical Engineering from Anna University in 1984, M.Tech. in Aircraft Production Engineering from IIT Madras in 1988 and Ph.D. from IIT Kanpur in 1994. He is a Humboldt Fellow. His research interests are Rapid Prototyping & Tooling, CNC, Automation and Computer Graphics.

Anoop Kheerwal will complete his B.Tech. in Mechanical Engineering at IIT Bombay in April 2004. He is doing his bachelor thesis entitled "Applications of Visibility-Based Algorithms for Rapid Prototyping". His research interests are in the areas of CAD/CAM, particularly Rapid Prototyping and Computer Graphics.

P. Vivekananda Shanmuganathan is an Assistant Professor in Mechanical Engineering at VIT, Vellore. He received his B.E. in Mechanical Engineering from Madurai Kamaraj University in 1992, M.E. in Computer-Integrated Manufacturing from Bharathiar University in 1996 and Ph.D. from IIT Bombay in 2003. His research interests include Walking Robots, Machine Vision, RP&T and Computer Graphics.

Rohitashwa Shringi is a Reader in Kota Government Engineering College and is pursuing Ph.D. in Mechanical Engineering at IIT Bombay. He received B.E. in Production Engineering from Punjab University, Chandigarh in 1984 and PGDIE in Industrial Engineering from NITIE, Mumbai in 1991. His research interests include process control, simulation, solid modeling and CAD/CAM.

Karuna P. Karunakaran


P. Vivekananda Shanmuganathan

Anoop Kheerwal

Rohitashwa Shringi