3-D Mold Design System for Pseudo-Solid Product Models

Sang Hun Lee
Graduate School of Automotive Engineering,
Kookmin University, Seoul, Korea
+82-2-910-4835
shlee@kookmin.ac.kr

Sung Lak Lim
Graduate School of Automotive Engineering,
Kookmin University, Seoul, Korea
+82-2-910-5151

Abstract

This paper describes the parting and Boolean operations for a pseudo-solid model of a plastic part, and their application to injection mold design. Here, a pseudo-solid model means a sheet model that looks like a solid model, but its boundary is not closed. When a solid model created in a different CAD system is imported through a standard data exchange file format, in most cases, a pseudo-solid model may be created due to tolerance or some other problems. However, most existing mold design systems based on solid modeling kernels require a complete part solid model. Therefore, mold designers have to do time-consuming healing operations to convert a pseudo-solid to solid. To eliminate or reduce the healing pre-process for mold design, in this paper, we proposed the parting and Boolean Operations on pseudo-solid part models, and implemented them using UG/Open API of Unigraphics. The detailed algorithms are described in this paper.

Keywords: CAD; Boolean Operation; Non-manifold Modeling; Mold Design; Non-watertight Solid.

1. Introduction

As solid modeling systems have widely spread in the product design area recently, they are being introduced in mold design for products. To support solid mold design efficiently, dedicated CAD systems for mold design have been developed and introduced to mold manufacturers [1, 2, 3]. Since the CAD systems of mold manufacturers are different from those of customers, CAD data files are delivered in the form of a standard data exchange file like IGES [4]. However, in most cases, complete solid models are not constructed from the product CAD file due to tolerance or other problems. In this paper, such a non-watertight incomplete solid model is called a pseudo-solid model. The pseudo-solid model can be defined as a sheet body or a group of sheet bodies that cannot form a closed boundary of the solid body. Figure 1 illustrates a pseudo-solid model for a plastic injection molding part. It looks like a solid model. But, in fact, it is a sheet model because its surface is not closed due to the gaps shown in the circles of the figure.

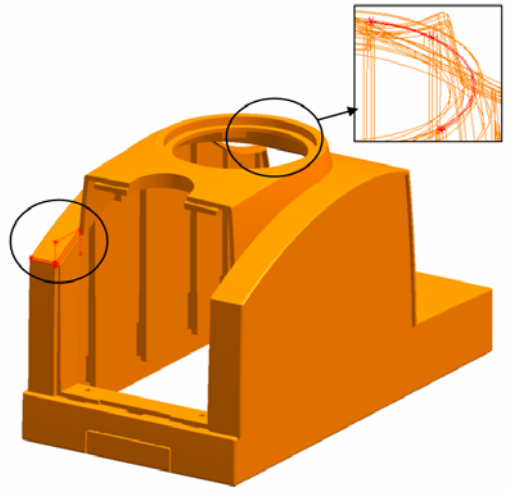


Figure 1. An example of pseudo-solid model.

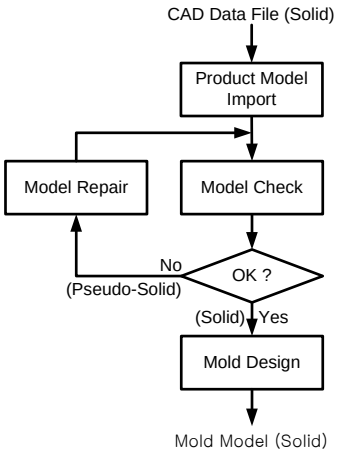


Figure 2. Current workflow of mold design.

Since the current dedicated systems for mold design have been developed based on solid modeling technology, it is mandatory to heal pseudo-solid models before starting the mold design process. Figure 2 shows the workflow for mold design. In this process, the healing work is a very tedious and cumbersome process that usually takes more than a few days.

However, in the view of CAM, a complete solid model is not necessarily needed. Traditionally, surface or sheet models have been used in this area. If the ultimate objective of mold design is to produce a mold, it is desirable to omit or reduce the healing process to a minimum, because most defects of pseudo-solid models are too small to affect the NC tool path generation.

To meet this requirement, in this paper, we proposed the parting and Boolean operations on pseudo-solid models, and developed a mold design system based on these operations, instead of traditional solid modeling operations. The pseudo-solid modeling operations were implemented using UG/Open API of Unigraphics [5]. The following sections are dedicated to a brief explanation of the mold design system and the algorithms of the parting and Boolean operations.

2. KMU-MOLD: Mold Design System

Figure 3 shows the design modules of KMU-MOLD, which is an injection mold design system developed by the CAD group of Kookmin University based on Unigraphics. The design modules of KMU-MOLD can be classified into three groups as illustrated in Figure. 3: core design, moldbase design, and miscellaneous support module groups. The core design group includes the modules for creating a pattern from a given product model, generating a core block and its pocket, modeling parting surfaces, splitting the core block by parting surfaces, generating insert cores, and modeling the cavity part of a slider. The moldbase design group includes the modules for generating a moldbase, generating the moving parts of a slider, modeling gates and runners, creating cooling channels, and generating a wide variety of standard parts. The miscellaneous support group includes the modules for providing product and mold information, generating a BOM, generating electrodes for mold cavity production, and so on.

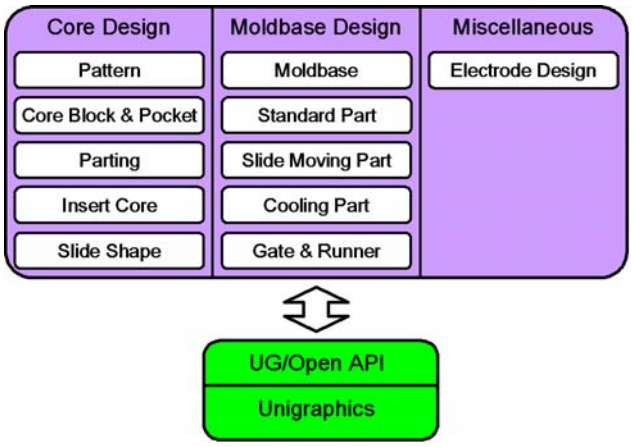


Figure 3. Design modules of KMU-MOLD.

To implement the mold design modules, it is essential to develop two kind of basic modeling operations on pseudo-solid models: parting and Boolean operations. The parting operation is necessary for design modules for core and cavity blocks. The Boolean operations are required for design modules for slide, pockets for core insert, holes for standard parts, and cooling channels. The following sections describe the algorithms of these operations in more detail.

3. Parting Operation

There are several technical terms related with this operation. The pattern means a solid or pseudo-solid model that is copied from the customer's product model and enlarged by the shrinkage rate of the specified plastic material to compensate for shrinkage after molding. The integrated core block represents a stock from which the pattern is subtracted for generating a cavity. This integrated block is split along with the parting lines to form the cavity block and the core block. The cavity block is implanted into the cavity plate, and the core block is implanted into the core plate. The shape of the core block is selected from a block, a cylinder, and an extruded solid with user-defined profile.

The procedure to split an integrated core block consists of six steps as illustrated in Figure 4.

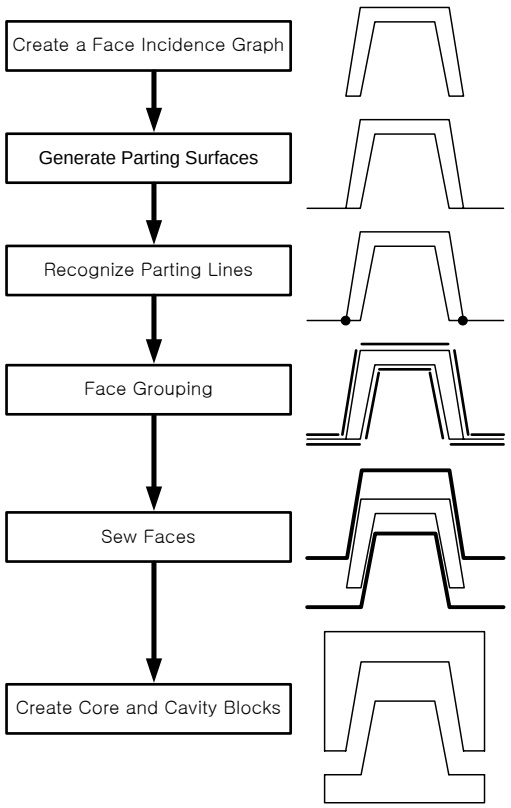
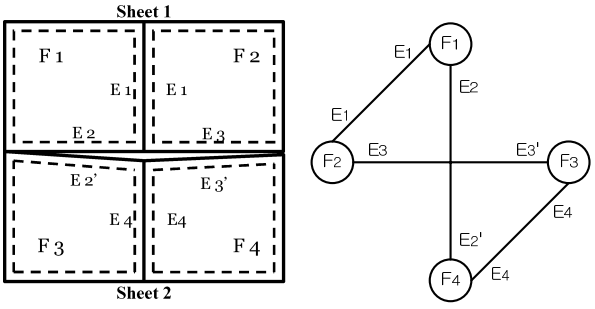


Figure 4. Parting process for a pseudo-solid model.

(Step 1) Creating a Face Adjacency Graph.

In a solid model, two adjacent faces always share an edge. However, in a pseudo-solid model, this rule is not satisfied all the time and place. For parting operation, it is necessary to find the pairs of these non-manifold edges and build a face adjacency graph. For instance, a pseudo-solid model is composed of two sheets, as shown in Figure 5(a). For each boundary edge of the sheets, the coincident edge is searched with varying the tolerance. Once all the pairs of non-manifold edges are searched, a face adjacency graph is established easily, as illustrated in Figure 5(b).



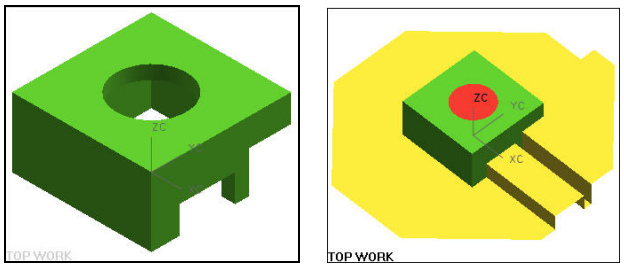
(a) **(b)**

Figure 5. Searching for adjacency information: (a) an example pseudo-solid model, (b) a face adjacency graph.

(Step 2) Creating Parting Surfaces.

Parting surfaces are created to split an integrated core block. They are classified into two groups: internal and external. An internal parting surface is created to split a passage, which is a hole through the part. It is generally bounded by topological entities constituting the passage. An external parting surface is created by sweeping the curve segments of the parting line to the outside of the integrated core block.

Parting surfaces can be created by some dedicated modeling capabilities of the system, or by the conventional surface modeling capabilities of Unigraphics. The dedicated parting surface modeling capabilities are Fill Holes on a Surface, Untrim Sheet by Edges Boundary, Extrude, Sweep, and Bounded Plane. The parting surface created by general modeling functions of Unigraphics should be registered as parting surface using the function called Register Parting Surface. An example part pattern model, which is used to illustrate the algorithm from now on, is presented in Figure 6 (a), and its internal and external parting surfaces are shown in Figure 6 (b).



(a) **(b)**

Figure 6. Generating parting surfaces: (a) a pseudo-solid pattern model, (b) internal and external parting surfaces.

(Step 3) Searching for Parting Lines.

In this step, the system searches for parting lines from the pattern and the parting surfaces. A loop of parting lines is defined as a set of edges where the pattern and the parting surfaces contact. Figure 7 illustrates two loops of parting lines obtained from the pattern and the parting surfaces in Figure 6.

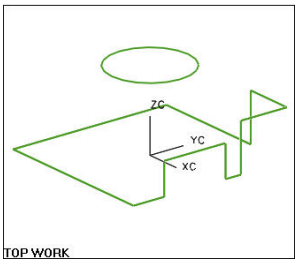
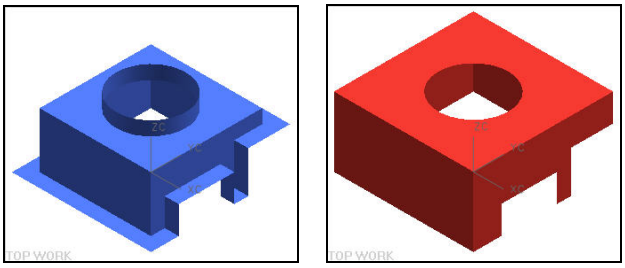


Figure 7. Searching for the parting lines.

(Step 4) Grouping Pattern Faces.

In this step, the faces of the pattern are grouped into an upper and a lower face set whose boundary is the parting lines. Figure 8 illustrates two face sets for the example pattern.

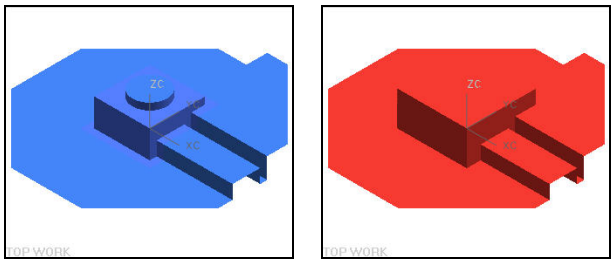


(a) **(b)**

Figure 8. Grouping the faces of pattern into two sets: (a) a lower face set, (b) an upper face set.

(Step 5) Sewing Faces to Create Upper and Lower Blades.

The parting surfaces are sewn to the upper face group to get an upper blade, as illustrated in Figure 9 (a). In the same way, the parting surfaces are sewn to the lower face group to get a lower blade, as illustrated in Figure 9 (b).



(a) **(b)**

Figure 9. Sewing the parting surfaces to lower and upper face groups: (a) lower blade, (b) upper blade.

(Step 6) Creating Core and Cavity Blocks.

Finally, core and cavity blocks are created using the upper and lower blades with referring to the dimensions of the integrated core block. The algorithm for creation of a core block is basically the same as that of a cavity block. Then, let us investigate the core

block generation algorithm. First, four lateral datum planes are created according to the dimensions of the integrated core block, as illustrated in Figure 10 (a). Next, the parting surfaces are trimmed by these datum planes, as illustrated in Figure 10 (b). Then, the edges are generated according to the shape of the integrated core block, as shown in Figure 10 (c). Finally, four lateral faces and one bottom face are created to complete a core block, as illustrated in Figure 10 (d).

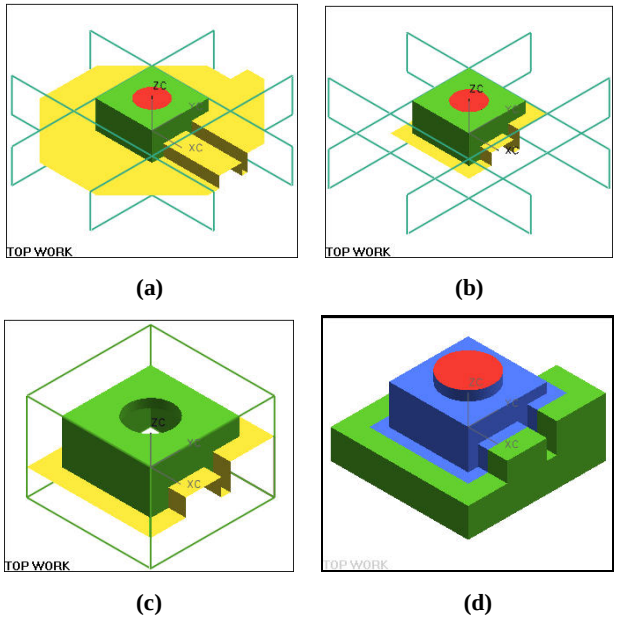


Figure 10. Generating a core block: (a) creating four datum planes, (b) trimming the upper blade using the datum planes, (c) generating lines, (d) creating the lateral and bottom faces to complete a core block.

A cavity block can be generated in the same way except that a top face is created instead of a bottom face. Figure 11 illustrates a cavity block for the example pseudo-solid pattern.

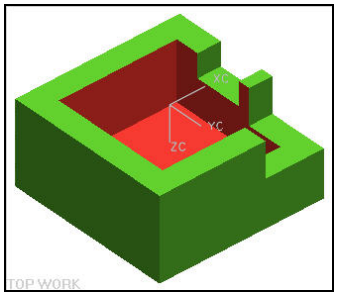


Figure 11. Generating a cavity block.

4. Boolean Operations on Pseudo-Solid Models

The algorithm for Boolean operations on pseudo-solid models is comprised of five main steps, as described in the flowchart of Figure 12. This algorithm is devised taking into consideration of the UG/Open API routines. The details of this algorithm are described in the following steps for an operation on two bodies illustrated in Figure 13.

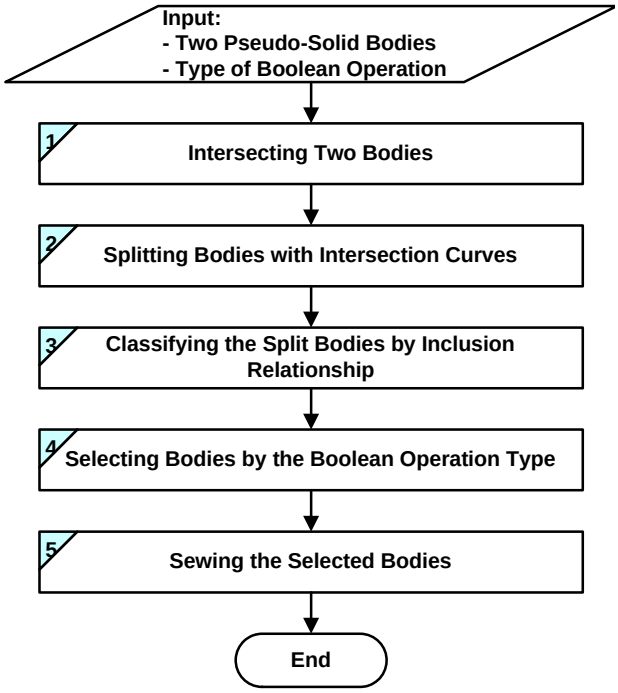


Figure 12. Overall procedure of the algorithm for Boolean operations on pseudo-solid models.

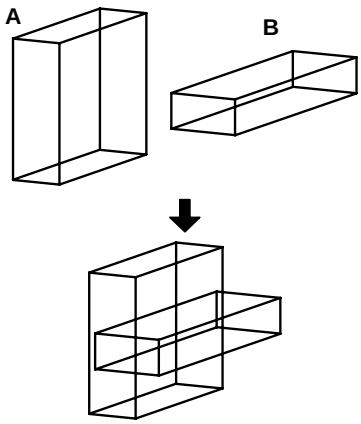


Figure 13. Target and tool bodies for Boolean operations.

(Step 1) Intersecting Two Bodies.

The intersection curves between all the faces of a target body A and all the faces of a tool body B are calculated, as illustrated in Figure 14.

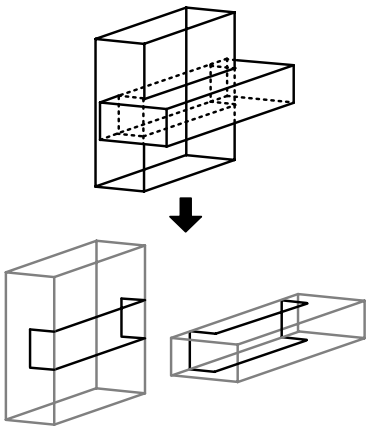


Figure 14. Calculation of intersection curves for two pseudo-solid models.

(Step 2) Splitting Bodies.

The faces are classified into intersecting faces and non-intersecting faces. Each group of the connected non-intersecting faces are extracted and constitute a sheet body. The faces in a group are obtained by traversing the neighboring faces, starting from any seed face in the group without crossing the intersecting faces. Then, each intersecting face is extracted and split by the intersection curves. If the intersection curves on a face are not continuous, they are connected by moving vertices or adding edges. Figure 15 illustrates sheet bodies extracted from the original pseudo-solids in Figure 13.

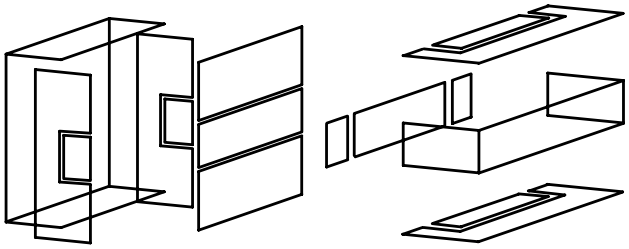


Figure 15. Splitting bodies with intersection curves.

(Step 3) Classifying the Relative Location of Each Sheet Body.

In this step, the relative location of each of the sheet bodies obtained in Step 2 with respect to its counterpart pseudo-solid is identified. The relative location is represented by IN, OUT, or ON. In case of ON, the location is further classified into ON_SAME and ON_OPPOSITE. ON_SAME means that the normal directions of a sheet and its counterpart pseudo-solid are the same. ON_OPPOSITE means that the normal directions of a sheet and its counterpart pseudo-solid are opposite.

To determine the relative location, we use a ray test method. In this method, a ray emanated from a point on a sheet in an arbitrary direction is first generated, and then the number of intersections between the ray and the counterpart pseudo-solid are counted. If the number is even, the sheet is classified OUT; if odd, IN. If the ray intersects any vertex or edge, the ray direction is modified until no intersection occurs. Figure 16 shows the result of the classification for the example models.

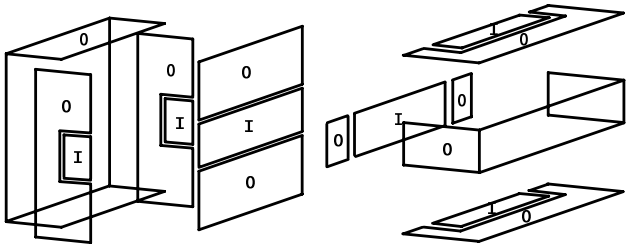


Figure 16. Classifying the relative location of each sheet body.

(Step 4) Collecting Sheet Bodies.

The proper sheet bodies are collected based upon the type of Boolean operation being applied. Table 1 gives a rule to be followed to collect the proper sheets. For the union operation, the sheets marked by O in the column of UNION are collected. For the intersection and difference operations, the sheets marked by O in the columns of INTERSECTION and DIFFERENCE are collected, respectively.

Table 1. Sheet selection rules for each type of Boolean operations.

		Union	Intersection	Difference
Solid A	OUT	O	X	O
	IN	X	O	X
	ON_SAME	O	O	X
	ON_OPPOSITE	X	X	O
Solid B	OUT	O	X	X
	IN	X	O	O
	ON_SAME	X	X	X
	ON_OPPOSITE	X	X	X

(Step 5) Sewing the Collected Sheets.

The sheet bodies collected in Step 4 are sewn together at their common boundaries. The result is a pseudo-solid model, which is composed of a single or multiple sheet bodies. Figure 17 shows the result of union operation.

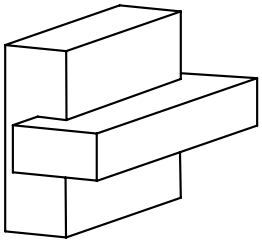


Figure 17. Result of union operation.

5. Examples

Boolean operations on pseudo-solid models are applied to the design module for insert cores. The modeling procedure for an insert core consists of two steps. First, a sketch on a datum plane is created and swept to create a workpiece for an insert core, as illustrated in Figure 18. Next, the swept workpiece is intersected with a core block to create an insert core, as illustrated in Figure 19. Finally, the swept workpiece is subtracted from the core block to create a pocket or hole for assembly of the insert core, as illustrated in Figure 20.

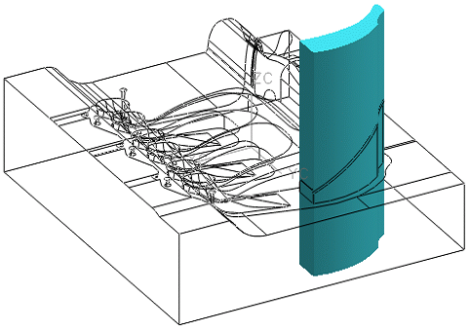


Figure 18. Sweeping a sketch profile.

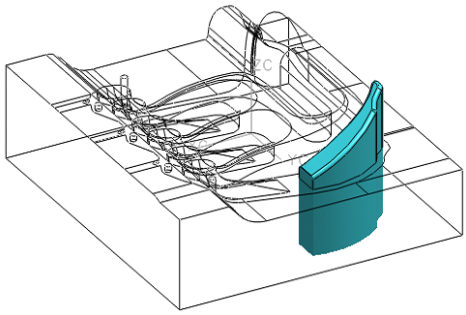


Figure 19. Intersecting a swept volume and a core block.

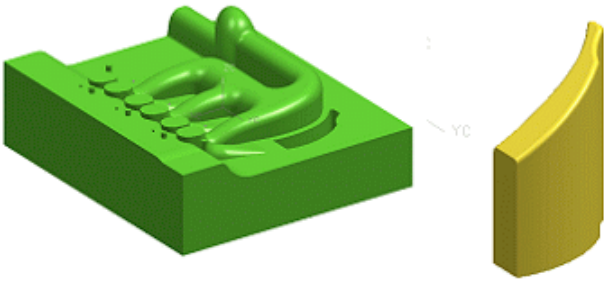


Figure 20. Generating an insert core.

6. Conclusion

In this paper, we proposed the parting and Boolean operations on pseudo-solid models, and their application to injection mold design in Unigraphics. By introducing these operations to mold design, the productivity of the mold design can be highly raised by reducing healing work to a minimum, and the CAM process can be launched early by receiving the surface data of the mold cavity before the healing stage. However, the current system needs further development for enabling 'wave geometry' capabilities provided by Unigraphics, which automatically modifies the related parts when a precedent feature is modified. In addition, if there is a severe topological defect in the product model, these operations may fail. In this case, the user needs to repair the product model to get an acceptable pseudo-solid model.

Acknowledgements

This Work was supported by KITECH and JY Solutec.

References

[1] Lee, S.H. and Lee, K. An Integrated CAD System for Mold Design in Injection Molding Process. In *Production Engineering Division, The Winter Annual Meeting of the ASME*, Chicago, PED-Vol.32, 1998, 257-271.

[2] Fujitsu, L. *MOLDWARE CAD user manual.* 1997.

[3] IMOLD. *http://www.eng.nus.edu.sg/imold/.* 2002.

[4] Lee, K. *Principles of CAD/CAM/CAE Systems.* Addison-Wesley, Cambridge, U.K., 1999.

[5] Electronic Data Systems Corporation. *UG/OPEN API Reference Version 18.0.* Unigraphics Division, 2002.