

# Multi-Piece Mold Design Based on Linear Mixed-Integer Program Toward Guaranteed Optimality

Stephen Stoyan, Yong Chen\*

Epstein Department of Industrial and Systems Engineering, University of Southern California, Los Angeles, CA 90089, USA

\*Corresponding author: [yongchen@usc.edu](mailto:yongchen@usc.edu), (213) 740-7829

**Abstract**—Multi-piece molds are a type of molding technology, which consist of more than two mold pieces and are assembled/disassembled like a space puzzle. Based on such molds, complex parts can be made for limited run production. Compared to traditional two-piece molds, parts with much more complex geometries can be made; however, this also brings challenge in designing such multi-piece molds. Previous works to address the problem are all based on heuristics. In this paper, we present a multi-piece mold design framework based on linear mixed-integer program. In our method, multi-piece mold design with guaranteed optimality on the number of mold pieces can be generated for any given CAD model of a molded part. The formulation of multi-piece mold design as a linear mixed-integer program is presented. The related multi-piece mold design framework is discussed. Some examples are provided which illustrate the effectiveness and efficiency of our approach.

**Keywords**—Computer-aided design; decision support; rapid tooling; linear mixed-integer programming

## 1. INTRODUCTION

Multi-piece molds such as space puzzle modeling developed by *Protoform GmbH* ([www.protoform.de](http://www.protoform.de)), can have more than two mold pieces. In the injection molding process, mold pieces are first hand-loaded into a mold base mounted on the injection molding machine. During part injection and cooling process, the mold pieces are accurately and securely clamped into the holding device. Finally each mold piece is hand-removed from the mold base to release the injection molded part. An example of multi-piece molds and related injection molded part provided by *Protoform GmbH* is shown in Figure 1. As shown in the figure, each mold piece can have its own *parting direction* (PD), along which, the mold piece can be separated from the part. (1) Compared to the traditional two-piece molds, multi-piece molding can produce limited run production parts with more complex geometries. (2) Compared to rapid prototyping (RP) processes such as Stereolithography Apparatus (SLA), multi-piece molding can efficiently produce a small quantity of parts (e.g. 100); more importantly, the fabricated parts can be made in desired injection molding materials that may not be available to RP processes. Hence multi-piece molding has become an important tooling technology in the era of mass customization, in which limited run production is increasingly becoming a common industrial practice.

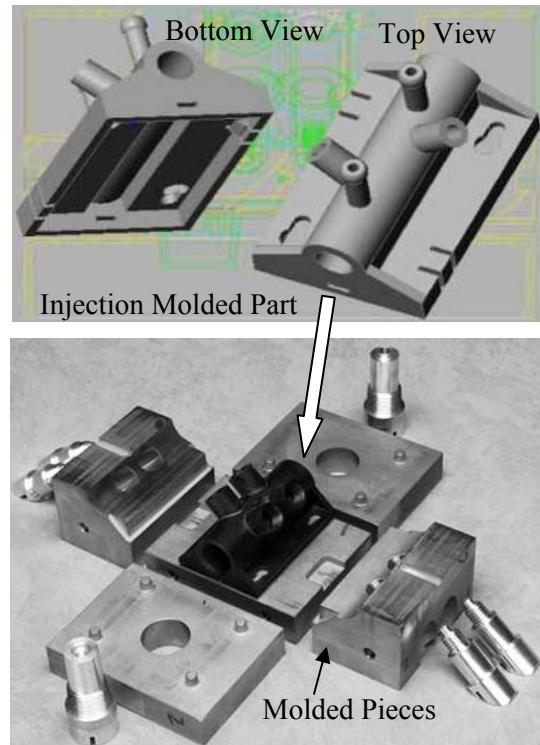


Fig. 1: An example of multi-piece molds given by *Protoform GmbH*.

Given the geometry of a part, depending on the selection of mold design variables, a different number of mold pieces may be required to form the part. It is desired to minimize the number of required mold pieces because fewer mold pieces reduces the tooling cost and simplifies the operation of the mold. Therefore, in this paper, we will consider the automation of multi-piece mold design problem defined as:

**Problem MPMD: Multi-Piece Mold Design.** *Given a solid part and a mold base, design the minimum number of mold pieces that can form the cavity of the part in the material injection process, and can be disassembled properly in the part ejection process.*

### 1.1 Related Work

The automation of mold design for injection molding has been extensively studied before. Some representative work can be found in [1~6]. Most of the work focuses on two-piece molds, including the determination of parting direction, parting line, parting surface, and undercut detecting. Among them, the selection of parting direction has received much

attention since it is an important step in the automatic mold design process. Recently, new programmable graphics hardware accelerated algorithms have also been presented to test the moldability of parts and help in redesigning them [7].

Chen and Rosen [8, 9] first presented a multi-piece mold design method that allows three-dimensional mold decomposition. More recently, Gupta's group presented a set of geometric algorithms for automated design of multi-piece permanent molds [10], sacrificial multi-piece molds [11], and multi-stage molding [12]. These works provide an excellent groundwork on multi-piece mold design to improve upon.

One key problem of the current approaches is that the multi-piece molds are designed based on heuristics. Hence no optimality can be guaranteed for a given arbitrary geometry. For example, in designing multi-piece permanent molds [10], a set of candidate parting directions are selected from (1) principal direction, (2) planar face normal, and (3) cylindrical/conical face axis. Based on them, a greedy scheme is then used in identifying the minimum number of mold pieces. In this paper, a multi-piece mold design method based on linear mixed-integer program is presented. In our method, a lower and upper bound on the number of mold pieces are first identified for a given part geometry. Based on them, heuristics can then be used in further improving the generated design. Our design framework, for the first time, provides a solid foundation in pursuing multi-piece mold design with guaranteed optimality.

## 1.2 Problem Formulation

Before discussing our approach, we first present a more accurate formulation of the aforementioned Problem MPMD as follows.

**Problem MPMD: Multi-Piece Mold Design.** Given a polygonal model ( $P$ ) and a mold base ( $\Psi$ ), design a  $n$ -piece mold  $M = \{m_1, m_2, \dots, m_n\}$  to:

**Minimize:** number of mold pieces ( $n$ ).

**Subject to:**

- (1) Each  $m_i \in M$  is a connected solid;
- (2) Each  $m_i \in M$  has a parting direction  $d_i$  such that  $m_i$  can be disassembled along  $d_i$ ;
- (3)  $M = \bigcup_{i=1}^n m_i$  satisfies  $M = \Psi - P$ .

The multi-piece mold design problem considered in this paper is essentially the same as the ones defined in [8] and [10].

## 2. MULTI-PIECE MOLD DESIGN APPROACH

For a part to be moldable, every facet on the part needs to be accessible from at least one direction. Usually such a direction can be easily found for each individual facet; however, the core challenge in mold design is to find the minimum number of directions that are common to all the facets of the part.

### 2.1 Mold Design Based on Visibility of Faces

Chen *et al.* [13] formulated demoldability as a visibility problem and presented a set of important computational techniques such as visibility and *Gaussian* maps (*V-Maps* and *G-Maps*). For a planar surface  $F$ , its V-map is a hemisphere centered on the unit outward normal. By calculating the intersections of all V-maps of region faces, allowable draw ranges can be computed. Several approaches and algorithms based on spherical polygons have been presented for different applications [14-15]. However, the algorithms and related data structures are rather complicated. In addition, it takes considerable computational time to compute the exact V-map intersections based on spherical surfaces.

Instead, Chen and Rosen [8] presented an alternative approach based on linear program for evaluating the parting directions of a set of surfaces. Suppose a connected region  $R$  consists of face  $F_i$  (with unit face normal  $N_i$  and area  $A_i$ ,  $1 \leq i \leq n$ ). If a common direction  $d(d_x, d_y, d_z)$  exists such that every facet  $F_i$  can be accessible from such a direction,  $d$  is a candidate parting direction of the region. In addition, the ease of ejection can be used as the criterion to choose an optimal parting direction from a feasible range. For a face  $F_i$ , its ease of ejection can be determined by the draft angle and the area in shear contact with the related part face during the mold-opening operation (i.e.  $A_i(N_i \cdot d)$ ). Hence an optimization problem for determining a parting direction of region  $R$  can be formulated as follows.

$$\text{Maximize: } \sum_{i=1}^n A_i(N_i \cdot d)$$

$$\text{Subject to: } N_{xi}d_x + N_{yi}d_y + N_{zi}d_z \geq 0 \text{ for face } F_i$$

$$d_x^2 + d_y^2 + d_z^2 = 1 \text{ (sphere constraint)}$$

A unit sphere related to the sphere constraint can be approximated by a set of linear surfaces with acceptable errors. Suppose the equations of a planar surface  $M_\ell$  are  $M_{x\ell}d_x + M_{y\ell}d_y + M_{z\ell}d_z = \mu_\ell$  with face normal  $(M_{x\ell}, M_{y\ell}, M_{z\ell})$  toward the inside. A linear problem can be formulated for evaluating a parting direction of the region.

**Problem PDLP: Parting Direction Linear Problem.**

$$\text{Maximize: } \sum_{i=1}^n A_i(N_i \cdot d) \tag{2.1}$$

$$\text{Subject to: } N_{xi}d_x + N_{yi}d_y + N_{zi}d_z \geq 0 \text{ for face } F_i \tag{2.2}$$

$$M_{x\ell}d_x + M_{y\ell}d_y + M_{z\ell}d_z \geq \mu_\ell \text{ for face } M_\ell. \tag{2.3}$$

A linear program problem in 3 dimensions can be solved in  $O(n)$  time ( $n$  is the number of constraints) and on linear storage. Therefore, the running time to solve Problem PDLP is very fast, typically in milliseconds for thousands of faces based on a commercial solver (e.g. LINGO system - www.lindo.com). More importantly, the computation process of finding a parting direction for a set of faces becomes much easier and more robust.

Motivated by such a linear program approach, we further develop a linear mixed-integer program for determining a minimum number of parting directions for all the face of a given part. The optimization formulation is discussed in Section 3. A simple example (Test 1 in Section 6) is shown in Figure 2 to illustrate the objective in Problem PDLP, which is also used in the linear mixed-integer program. For a simple cube, there is an infinitely number of parting directions that can be used in its mold design. However, considering the ease of ejection, the two directions,  $d_1$  and  $d_2$  as shown in the figure, are the most desired. In such directions, the projection area

$$\sum_{i=1}^n A_i (|N_i \cdot d_1| + |N_i \cdot d_2|)$$
 is also the maximum.

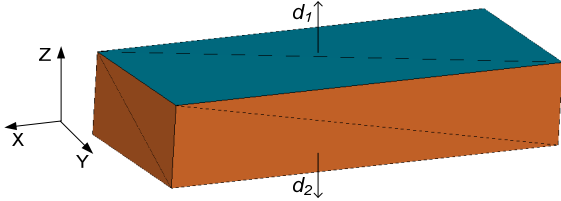


Fig. 2: A simple example to illustrate the optimization objective in PDLP.

## 2.2 Basic Elements for Multi-Piece Mold Design

In Problem PDLP, a single face  $F_i$  is used as the basic element in evaluating a feasible parting direction. However, such a face is not a good element in identifying a minimum number of parting directions in multi-piece mold design. This is due to the accessible direction to a face may be blocked by its neighboring faces (e.g. two neighboring faces between a concave edge); however, such blockage is hard to be incorporated if faces are directed used as the basic element.

As shown in Figure 2, any pair of directions can be used in mold design if the given part contains only convex edges. Based on such an idea, Chen *et al.* [13] computed a set of “pockets” based on the convex hull of an object to capture all the non-convex regions. Accordingly, the visibility map of each pocket can be computed and the parting directions to maximize the number of completely visible pockets can be determined.

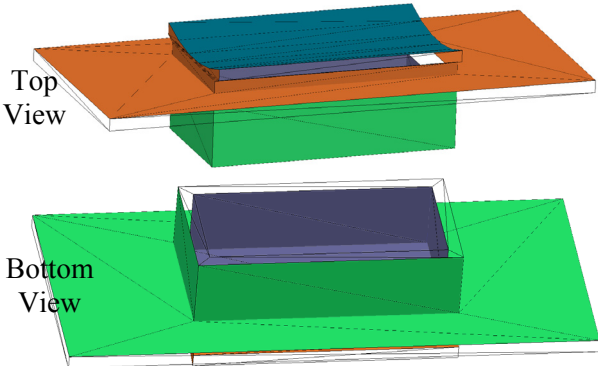


Fig. 3: An example of pockets for a test part.

An example of pockets for a test part (Test 3 in Section 6) is given in Figure 3. The four pockets of the part are shown in different colors. It can be seen that a pocket is a molding feature which contains multiple concave and convex edges.

As discussed in [8], pockets give us less design freedom in trying different combinations of elements; hence they are also not good elements for identifying a minimum number of parting directions in multi-piece mold design.

As discussed in [8], concave edges (i.e. the dihedral angle of two neighboring faces is bigger than  $180^\circ$ ) can capture the blockage of accessible directions between neighboring faces. Based on such edge classification, two types of elements can be defined as follows.

**Definition 2.1.** A face  $F$  is a **convex face** if all edges of  $F$  are convex edges.

**Definition 2.2.** A **Concave Region** is a set of connected faces such that: (1) all internal edges of the region are concave edges; and (2) all boundary edges of the region are convex or flat edges.

As an example, the concave regions for the test part in Figure 3 are shown in Figure 4. There are a total of 12 concave regions which are shown in different colors; in addition, there are 68 convex faces which are drawn in wireframe in the figure. Hence, compared to pockets, the design freedom based on them for identifying a minimum number of parting directions has been significantly increased.

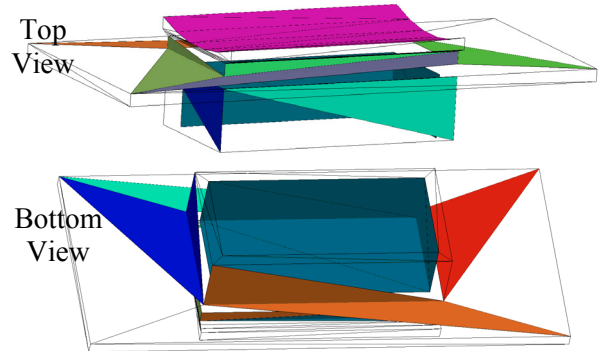


Fig. 4: An example of concave regions and convex faces for a test part.

Similar to [8], concave regions and convex faces are used as the basic elements in our multi-piece mold design method. A similar strategy has also been adopted in their generation. However, different from [8] that is based on a 3D geometric modeler (ACIS from *Spatial* – [www.spatial.com](http://www.spatial.com)), our current method is based on polygonal meshes (i.e. the input model is defined in a STL file). Hence the approach in computing concave regions and convex faces is accordingly modified as follows by adding a concept of *Convex\_Edge\_Vertex*.

- (1) Classify all edges as concave, convex and flat.
- (2) Assign two faces to the same region if they share a concave or flat edge.
- (3) Identify all the internal convex edges of each region and mark the vertices of such edges as *Convex\_Edge\_Vertex*. Notice we do not want a region to have internal convex edges (i.e. all the internal edges should be concave as shown in Definition 2.2).
- (4) Regenerate regions by assigning two faces to the same concave region if they share a concave edge, or a flat edge if such a flat edge has no vertex that is marked as *Convex\_Edge\_Vertex*.

### 2.3 Approach Overview

Based on the identified basic elements (a set of concave regions and convex faces), the essence of multi-piece mold design is to try a different combination of them, and accordingly identify a design with the best performance (i.e. the minimum number of mold pieces and the easiness of ejection if the same number of mold pieces is achieved).

There have been various approaches to solve such combination problem between elements (e.g. the well-known knapsack problem). General solution methods involve: (i) searching all possible combinations of variables, (ii) using heuristics such as greedy heuristic, or (iii) optimization methods such as cutting planes or the branch and bound method. Each solution method has advantages and disadvantages. Searching all combinations, for example, has costs with respect to solution time, especially with large-scale problems where validating a solution may take years. Heuristics typically generate fast solutions, but they do not guarantee optimality. Optimization methods can guarantee optimality and use techniques that converge faster than searching the whole solution space; however, they generally perform slower than heuristics. When models involve integer variables (such as the one in the next section), the problems with the various solution methods described above get even worse. Although optimization methods have their trade-offs, depending on the complexity of the problem, current state-of-the-art solvers such as *CPLEX* from ILOG ([www.ilog.com](http://www.ilog.com)) can solve large-scale problems in reasonable time.

As discussed in Section 1, the previous multi-piece mold design methods are based on heuristics. They are effective but do not provide optimality and also may fail for geometries that have not been considered when generating such heuristics. In this paper, we formulate the Problem MPMD into a linear mixed-integer program and solve it based on optimization methods. As shown in Section 6, by using *CPLEX*, such a problem can be solved within 10 seconds for optimal solutions of the four test problems.

An overview of our method based on an example (Test 2 in Section 6) is shown in Figure 5.

- (1) A given part to be injection molded is shown in Figure 5.a. The part has 64 triangles. They can be classified into 3 concave regions (drawn in different colors) and 40 convex faces (drawn in wireframe) as shown in Figure 5.b.
- (2) Based on such elements, a linear mixed-integer program is formulated and programmed in *CPLEX*. Accordingly two parting directions,  $d_1 = (0, 0, 1)$  and  $d_2 = (0, -1, 0)$ , are identified by the optimization solver.
- (3) The optimization solution provided us a lower bound on the solution (i.e.  $n=2$ ). Based on them, all the concave regions and convex faces can be combined into two mold piece regions (shown in two different colors in Figure 5.c). Since multiple solutions may exist, a best one may be identified based on molding design knowledge.
- (4) Based on the generated mold piece regions and related parting directions, parting lines and parting surfaces can

be identified. Accordingly two mold pieces,  $M_1$  and  $M_2$ , can be constructed (refer to Figure 5.d). Notice there is a one to one correspondence between the mold pieces and the mold piece regions. Also the two mold pieces can be assembled and disassembled properly in the related parting directions. Additional locking features can be added in the mold pieces.

In this paper, we will mainly focus on the process of generating mold piece regions from concave regions and convex faces. Accordingly, the remainder of the paper is organized as follows. In Section 3, we present the optimization formulation for Problem MPMD. In Section 4, we discuss the face connectivity of mold piece regions based on the identified part directions. The further improvements of the mold design are discussed in Section 5. The test results are discussed in Section 6. Finally, we give conclusions in Section 7.

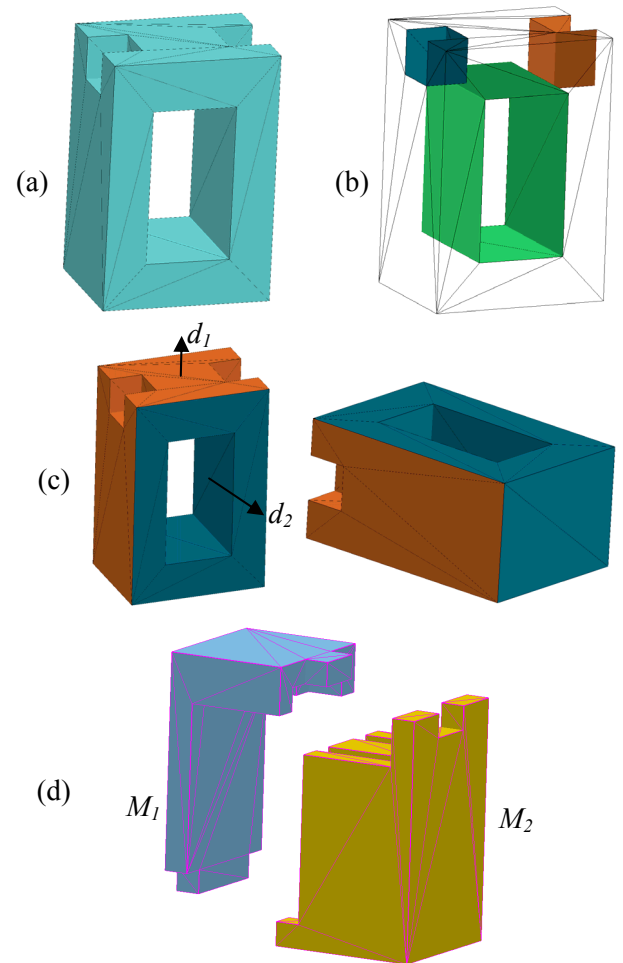


Fig. 5: An overview of our method.

### 3. OPTIMIZATION FORMULATION FOR COMPUTING PARTING DIRECTIONS

Before getting into the intricate details of the problem, we first outline the decision variables and parameters associated with the model. To begin we define:

- $n$ : the total number of parting directions of mold pieces;

$p$ : the total number of concave regions;  
 $m$ : the total number of approximated spherical surfaces;  
and their respective sets as:

$\phi := \{i : i \in [1, n]\}$ : the set of parting directions;  
 $\psi := \{k : k \in [1, p]\}$ : the set of concave regions;  
 $\theta := \{\ell : \ell \in [1, m]\}$ : the set of spherical surfaces.

In addition to the number of concave regions, there also exists a number of surfaces associated with each region, which do not necessarily have the same length. Thus, we define the set of surfaces associated with each concave region and introduce the set of convex surfaces as:

$\beta_k$ : the set of surfaces in each concave region  $k=1, \dots, p$ ;  
 $\beta_c$ : the set of convex surfaces.

The decision variables involved in the optimization problem are as follows:

$d_i = [d_x^i, d_y^i, d_z^i]$ : the vector of parting directions for mold pieces  $i=1, \dots, n$ ;  
 $g_i(k)$ : the binary variable used to enforce constraints related to parting directions  $i=1, \dots, n$  for  $k=1, \dots, p$  concave regions;  
 $g_i^c(\beta_c)$ : the binary variable used to enforce constraints related to convex surfaces for parting direction  $i=1, \dots, n$ ;  
 $z_i$ : a continuous variable used to satisfy the absolute value function in the objective.

Finally, the parameters involved with the decision variables in the design are defined as:

$N(k, \beta_k) = [N_x(k, \beta_k), N_y(k, \beta_k), N_z(k, \beta_k)]$ : the matrix with  $\beta_k$  surfaces related to concave regions  $k=1, \dots, p$ ;  
 $N^c(\beta_c) = [N_x^c(\beta_c), N_y^c(\beta_c), N_z^c(\beta_c)]$ : the matrix of convex faces with  $\beta_c$  surfaces;  
 $M_\ell = [M_x^\ell, M_y^\ell, M_z^\ell]$ : the vector defining approximated spherical surfaces  $\ell=1, \dots, m$ ;  
 $A(k, \beta_k)$ : the area of concave regions with  $\beta_k$  surfaces for  $k=1, \dots, p$ ;  
 $A^c(\beta_c)$ : the area of convex regions with  $\beta_c$  surfaces;  
 $u_\ell$ : a scalar value related to the approximated spherical surfaces  $\ell=1, \dots, m$ ;  
 $L$ : the lower bound of parting direction for mold pieces;  
 $U$ : the upper bound of parting direction for mold pieces.

The design described in the earlier sections requires the solution to two optimization problems. Given a set of constraints that define the mold we want to create, the first problem (I) entails finding the minimum number of vector parting directions  $d_i$  necessary to design the mold. After we know the minimum number of parting directions, the second problem (II) involves maximizing the surface area covered in forming the mold pieces. The first optimization problem (I) is the following:

$$\min \sum_{i=1}^n |d_i| \quad (3.1)$$

$$\text{s.t. } g_i(k) (N(k, \beta_k) d_i) \geq 0 \quad \forall i \in \phi, k \in \psi \quad (3.2)$$

$$\sum_{i=1}^n g_i(k) \geq 1 \quad \forall k \in \psi \quad (3.3)$$

$$g_i^c(\beta_c) (N^c(\beta_c) d_i) \geq 0 \quad \forall i \in \phi \quad (3.4)$$

$$\sum_{i=1}^n g_i^c(\beta_c) \geq 1 \quad (3.5)$$

$$M_\ell d_i \geq u_\ell \quad \forall i \in \phi, \ell \in \theta \quad (3.6)$$

$$|d_i| \geq L \quad \forall i \in \phi \quad (3.7)$$

$$|d_i| \leq U \quad \forall i \in \phi \quad (3.8)$$

$$d_i \in \mathbb{R} \quad \forall i \in \phi \quad (3.9)$$

$$g_i(k) \in \mathbb{B} \quad \forall i \in \phi, k \in \psi \quad (3.10)$$

$$g_i^c(\beta_c) \in \mathbb{B} \quad \forall i \in \phi. \quad (3.11)$$

After solving the problem above, we let  $n$  be the number of  $d_i \neq 0$  in the solution to (3.1)-(3.11). Then, the second optimization problem (II) is:

$$\max \sum_{i=1}^n \sum_{k=1}^p A(k, \beta_k) |N(k, \beta_k) d_i| + \sum_{i=1}^n A^c(\beta_c) |N^c(\beta_c) d_i| \quad (3.12)$$

$$\text{s.t. } (3.2) - (3.11). \quad (3.13)$$

The constraints are the same for both problems since they define the boundaries of the mold to be created. Constraints (3.2)-(3.3) and (3.4)-(3.5) ensure that at least one parting direction vector  $d_i$  satisfies the desired inequality and the rest can be turned "off" via the binary variables  $g_i(k)$  and  $g_i^c(\beta_c)$ ; respectively. The problem with both (3.1)-(3.11) and (3.12)-(3.13) is the objective functions and constraints are nonlinear and also involve binary variables  $g_i(k)$  and  $g_i^c(\beta_c)$ . However, the NonLinear Mixed-Integer Program (NLMIP) of (3.1)-(3.11) and (3.12)-(3.13) can be converted to an equivalent linear program with the introduction of a decomposition strategy and a few additional variables and constraints. The linear transformation allows the problem to be much more tractable and easier to solve.

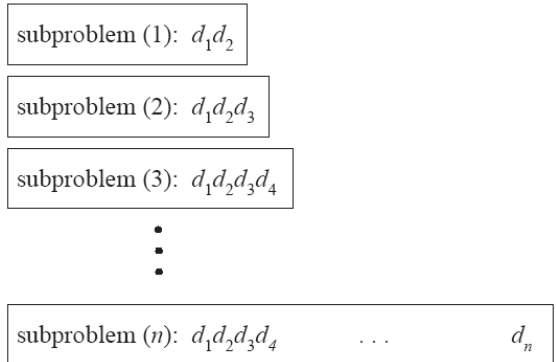


Fig. 6: Problem (I) subproblem decompositions for variables  $d_i$ .

We now describe the linear transformations and decomposition strategy used to make the problem more tractable. In the objective of problem (I), the NLMIP minimizes the number of variables  $d_i$  uses in the solution. Since we are looking for the lowest number  $i$  that satisfies (3.2)-(3.11), we decompose problem (I) into subproblems that increase in size by one variable  $d_i$ . As shown in Figure 6, problem (3.1)-(3.11) is solved for the number of variables corresponding to each subproblem until a solution is generated, in which case we stop.

Given the nature of the problem, we know that at least two variables will be needed to generate a solution, thus subproblem (1) begins with  $d_1$  and  $d_2$ . Then we stop after the first instance when subproblem ( $i \leq n$ ) obtains an optimal solution. The objective of problem (II) aims to maximize the surface area associated with the absolute value of the direction vector and the corresponding surface region. This has a linearly equivalent set of equations by introducing the following:

$$\max \sum_{k=1}^p \frac{1}{A(k, \beta_k)} \left( \sum_{i=1}^n z_i(k) \right) \quad (3.14)$$

$$\text{s.t. } z_i(k) \leq N(k, \beta_k) d_i \quad \forall i \in \phi, k \in \psi \quad (3.15)$$

$$z_i(k) \leq -N(k, \beta_k) d_i \quad \forall i \in \phi, k \in \psi. \quad (3.16)$$

This is also done for  $|N^c(\beta_c) d_i|$  where  $z_i^c$  is used, which is shown in equations (3.22), (3.25) and (3.26) below.

The nonlinear constraints in the problem can also be addressed in a similar fashion, which make the problem more tractable and less complex. The nonlinear constraints in (3.2) and (3.4) can be converted into linear constraints by using the following equation:

$$\Lambda(1 - g_i(k)) + N(k, \beta_k) d_i \geq 0 \quad \forall i \in \phi, k \in \psi, \quad (3.17)$$

where  $\Lambda$  is a large value. Here, when  $g_i(k)$  is equal to zero then this constraint is essentially turned "off" since the large  $\Lambda$  value will satisfy the inequality for any  $d_i$ . When  $g_i(k)$  is equal to one then  $N(k, \beta_k) d_i \geq 0$  must be satisfied, which is the desired function of constraint (3.2). This is again repeated for  $g_i^c(\beta_c) N^c(\beta_c) d_i$ , as is shown below in equation (3.29).

Finally, the upper and lower bound constraints of (3.7) and (3.8) can be linearly expressed by defining  $d_i = d_i^+ - d_i^-$ ,  $\forall d_i^+ \geq 0, d_i^- \geq 0$ . Hence,  $d_i^+$  and  $d_i^-$  are simply the positive and negative components of  $d_i$ , respectively. Then, satisfying the absolute value of the constraints is provided by the following equations:

$$\lambda(\delta_i) + d_i^+ \geq L \quad \forall i \in \phi \quad (3.18)$$

$$\lambda(1 - \delta_i) + d_i^- \geq L \quad \forall i \in \phi \quad (3.19)$$

$$-\lambda(\delta_i) + d_i^+ \leq U \quad \forall i \in \phi \quad (3.20)$$

$$-\lambda(1 - \delta_i) + d_i^- \leq U \quad \forall i \in \phi, \quad (3.21)$$

where  $\delta_i \in \mathbb{B} \quad \forall i \in \phi$ .

Thus, the original problem of solving problem (I) and (II) separately can now be done in one equivalent linear problem.

This equates to solving the following linear Mixed-Integer Program (MIP):

$$\max \sum_{k=1}^p \frac{1}{A(k, \beta_k)} \left( \sum_{i=1}^n z_i(k) \right) + \frac{1}{A^c(\beta_c)} \left( \sum_{i=1}^n z_i^c(\beta_c) \right) \quad (3.22)$$

$$\text{s.t. } z_i(k) \leq N(k, \beta_k) d_i \quad \forall i \in \phi, k \in \psi \quad (3.23)$$

$$z_i(k) \leq -N(k, \beta_k) d_i \quad \forall i \in \phi, k \in \psi \quad (3.24)$$

$$z_i^c(\beta_c) \leq N^c(\beta_c) d_i \quad \forall i \in \phi \quad (3.25)$$

$$z_i^c(\beta_c) \leq -N^c(\beta_c) d_i \quad \forall i \in \phi \quad (3.26)$$

$$\Lambda(1 - g_i(k)) + N(k, \beta_k) d_i \geq 0 \quad \forall i \in \phi, k \in \psi, \quad (3.27)$$

$$\sum_{i=1}^n g_i(k) \geq 1 \quad \forall k \in \psi \quad (3.28)$$

$$\Lambda(1 - g_i^c(\beta_c)) + N^c(\beta_c) d_i \geq 0 \quad \forall i \in \phi \quad (3.29)$$

$$\sum_{i=1}^n g_i^c(\beta_c) \geq 1 \quad (3.30)$$

$$M_\ell d_i \geq u_\ell \quad \forall i \in \phi, \ell \in \theta \quad (3.31)$$

$$d_i = d_i^+ - d_i^- \quad \forall i \in \phi \quad (3.32)$$

$$\lambda(\delta_i) + d_i^+ \geq L \quad \forall i \in \phi \quad (3.33)$$

$$\lambda(1 - \delta_i) + d_i^- \geq L \quad \forall i \in \phi \quad (3.34)$$

$$-\lambda(\delta_i) + d_i^+ \leq U \quad \forall i \in \phi \quad (3.35)$$

$$-\lambda(1 - \delta_i) + d_i^- \leq U \quad \forall i \in \phi \quad (3.36)$$

$$d_i^+ \geq 0 \quad \forall i \in \phi \quad (3.37)$$

$$d_i^- \geq 0 \quad \forall i \in \phi \quad (3.38)$$

$$g_i(k) \in \mathbb{B} \quad \forall i \in \phi, k \in \psi \quad (3.39)$$

$$g_i^c(\beta_c) \in \mathbb{B} \quad \forall i \in \phi \quad (3.40)$$

$$z_i(k) \in \mathbb{R} \quad \forall i \in \phi, k \in \psi \quad (3.41)$$

$$z_i^c(\beta_c) \in \mathbb{R} \quad \forall i \in \phi \quad (3.42)$$

$$\delta_i \in \mathbb{B} \quad \forall i \in \phi, \quad (3.43)$$

where (3.22)-(3.43) is solved using an increasing number of parting directions  $d_i$  until the first instance that generates a solution, as described above and shown in Figure 6.

#### 4. CONNECTIVITY OF MULTI-PIECE MOLD DESIGN

As discussed in Section 2.3, the formulated linear Mixed-Integer Program can be programmed in a *CPLEX* optimization solver. Advanced MIP algorithms based on methods such as branch and bound can be used in finding an optimized solution.

One benefit we gain based on our approach is that we can easily incorporate different draft angle requirements in our problem. That is, for a part to be injection molded, its surfaces that are parallel to the parting direction must be drafted at least an angle  $\gamma$  in order to ease the ejection of the part and reduce the damaging possibility of the part and molds. The minimal draft angle mainly depends on the molding process and material. For some molding processes such as urethane rubber molding, a zero or even slightly negatively draft angle are acceptable.

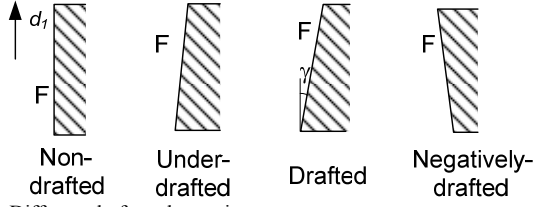


Fig. 7: Different draft angle requirements.

For a minimal draft angle  $\gamma$ , we can compute  $\tau = \sin(\gamma)$  and use it to replace 0 in Equations (2.2) and (3.27). Hence our optimization formulation can be changed as:

$$N_{xi}d_x + N_{yi}d_y + N_{zi}d_z \geq \tau \text{ for face } F_i, \text{ and}$$

$$\Lambda(1 - g_i(k)) + N(k, \beta_k)d_i \geq \tau, \forall i \in \phi, k \in \varphi.$$

Accordingly, the computed optimization solution, if any, will satisfy the given draft angle requirements. In addition, any non-drafted or under-drafted surfaces can be detected if no solution is found for them.

Notice, however, in our MIP formulation, only the demoldability requirement has been incorporated. We had difficulties in converting the face connectivity of a mold piece region into computable formulation; hence such connectivity has not been incorporated. Hence, the optimal solution given by solving the MIP is actually a lower bound on the multi-piece mold design. That is, if without considering the connectivity of mold piece regions, the minimum number of mold pieces is found to be  $n$  (e.g.  $n=5$ ), it is impossible to find a better solution (i.e.  $n < 5$ ) after the connectivity of  $n$  mold piece regions has been considered.

An example of such connectivity problem is shown in Figure 8. For a test part as shown in Figure 4, a minimum of three parting directions has been identified after solving the MIP (i.e.  $d_1$ ,  $d_2$ , and  $d_3$  as shown in Figure 8). Accordingly three mold piece regions ( $m_1$ ,  $m_2$ , and  $m_3$ ) are required for them respectively. However, for a concave region ( $CVR_1$ ), its solution is  $d_1$  since it has the biggest projection areas in such a direction; nonetheless,  $CVR_1$  is not connected to other regions that use the same parting direction.

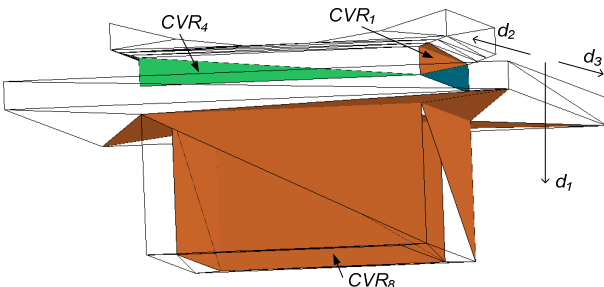


Fig. 8: An example of connectivity of mold piece regions.

Based on a set of given parting directions, it is trivial to check the face connectivity between concave regions and identify all the isolated regions (suppose  $m$  of them are identified). Accordingly we have an upper bound on the multi-piece mold design. That is, a solution must be able to be found by using  $n+m$  mold pieces after considering both demoldability and face connectivity requirements.

## 5. REFINING MULTI-PIECE MOLD DESIGN

Based on the computed parting directions, we can keep on refining the generated mold piece regions such that a mold design can be achieved that is closer to the lower bound ( $n$ ) instead of the upper bound ( $n+m$ ). This can be done based on various heuristics. For example, we can compute the projection areas of each concave region for the given parting directions (i.e.  $\sum_{i=1}^{\beta_k} A_i(N_i \cdot d)$ ). An example of such results is shown in Table 1 for the test case in Figure 8. In the table, a mark ‘ $\times$ ’ is assigned to a region if a related parting direction cannot satisfy the demoldability of the region.

TABLE 1  
THE PROJECTION AREA OF NINE CONCAVE REGIONS FOR TEST 3

Parting Dirs.	Concave Region # (9 out of 12)								
	0	1	2	3	4	5	6	7	8
$d_1$ (0,0,-1)	$\times$	0.3	$\times$	$\times$	$\times$	4	5	5	<b>14</b>
$d_2$ (1,0,0)	0	0	0	$\times$	<b>0.5</b>	0	2.5	$\times$	$\times$
$d_3$ (-1,0,0)	0	0	0	<b>0.5</b>	$\times$	0	$\times$	2.5	$\times$

Hence, for  $CVR_1$  that is identified as “isolated region” in direction  $d_1$ , we can check the other directions  $d_2$  and  $d_3$ . Since they are both demoldable and connected, we can reassign  $CVR_1$  to another mold piece region instead of adding a new one. Notice, as shown in Table 1, there are several concave regions that have a unique assignment to a related parting direction. For example,  $CVR_3$ ,  $CVR_4$ , and  $CVR_8$  can only be assigned to  $d_3$ ,  $d_2$ , and  $d_1$  respectively. We call such concave regions as the *core concave regions* of the related mold piece regions. Their assignments will not be changed during the refining process while other concave regions and convex faces may be. In addition, we can also compute a connectivity table of all the changeable concave regions and convex faces based on the core concave regions.

In addition to demoldability and face connectivity, a wealth of mold design and manufacturing knowledge has been characterized into a set of heuristics [1~12]. These heuristics can also be considered in the process of refining mold piece regions. For example, it is desired to have a smooth parting line. Hence for a mold design result generated by our system for test 3, which is shown in Figure 9, we can further refine it by changing the assignment of some faces for achieving a smoother parting line (refer to red lines as shown in the figure).

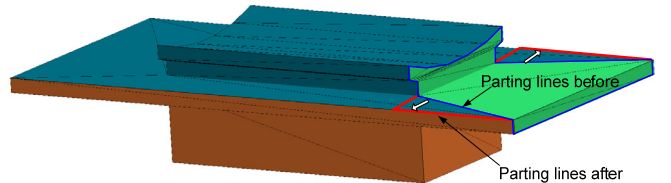


Fig. 9: An example of incorporating mold design heuristics.

## 6. EXAMPLES

Four test examples are given to illustrate our method. Test 1, 2, and 3 are shown in Figure 2, 5 and 3 respectively. Test 4

is shown in Figure 10. A summary of the computed results is given in Table 2. The results were generated by solving the model described in Section 3. The running time is based on a 3GHz Intel Xeon CPU using CPLEX 9.0.

TABLE 2  
EXPERIMENTAL RESULTS OF FOUR TESTS

Test #	Total Tri #	Concave Region #	Convex Face #	Resulted Parting Dir. #	Running Time (sec)
1	12	0	12	$d_1, d_2$ (0, 0, 1), (0, 0, -1)	0.01
2	64	3	40	$d_1, d_2$ (0, 0, 1), (0, -1, 0)	0.05
3	108	12	68	$d_1, d_2, d_3$ (0, 0, -1), (1, 0, 0), (-1, 0, 0)	0.13
4	528	86	250	$d_1, d_2, d_3, d_4$ (0, 1, 0), (0, 0, -1), (-1, 0, 0), (1, 0, 0)	8.15

The largest problem we solved is test 4 which has 86 concave regions and 250 convex faces. CPLEX solved the linear MIP to optimality with a CPU time of 8.15 seconds including the computation of both  $(d_1, d_2)$ ,  $(d_1, d_2, d_3)$  and  $(d_1, d_2, d_3, d_4)$ .

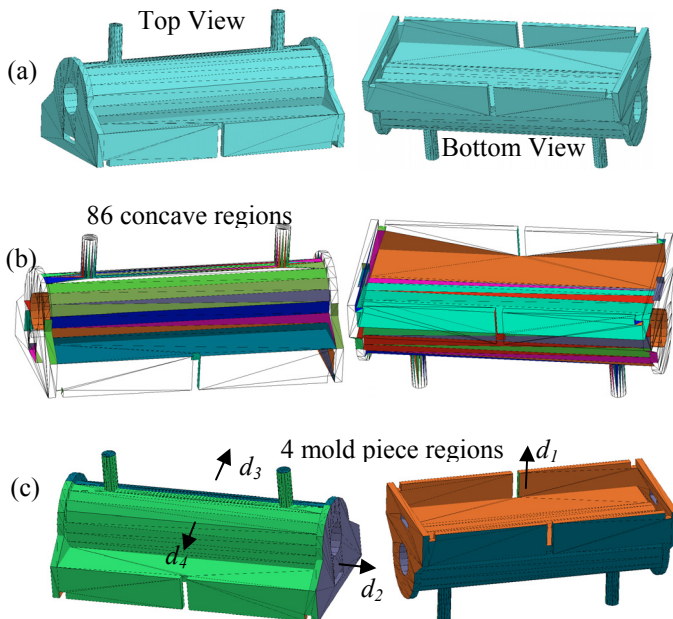


Fig. 10: Screen captures of computed results of test 4.

## 7. CONCLUSION

For the multi-piece mold design of an arbitrary part model, we presented a novel approach by formulating a linear mixed-integer program based on a set of basic elements, concave regions and convex faces. By using a state-of-the-art optimization solver, such a problem can be solved for optimal

solutions in reasonable time. Hence the optimality of multi-piece mold design can be provided by identifying a lower and upper bound on the number of mold pieces. The multiple-piece mold design can be further improved based on heuristics. Four examples were given and the test results have demonstrated the effectiveness and efficiency of our method.

## ACKNOWLEDGMENT

We acknowledge Prof. *Satyandra K. Gupta* at University of Maryland for providing us the part models of test 3 and 4.

## REFERENCES

- [1] K. Hui, Geometric Aspects of the Mouldability of Parts, *Computer-aided Design*, 29(3), pp. 197–208, 1996.
- [2] T. Wong, S. T. Tan, and W. S. Sze. Parting line formation by slicing a 3D CAD model. *Engineering with Computers*, 14(4), pp. 330-343, 1998.
- [3] M. W. Fu, J. Y. H. Fuh, A. Y. C. Nee, Undercut Feature Recognition in an Injection Mould Design System, *Computer-aided Design*, 31, pp. 777–790, 1999.
- [4] Z. Yin, H. Ding, Y. Xiong. Virtual prototyping of mold design: Geometric mouldability analysis for near-net-shape manufactured parts by feature recognition and geometric reasoning. *Computer Aided Design*, 33(2), pp. 137-154, 2001.
- [5] X. G. Ye, J. Y. H. Fuh, K. S. Lee. Automatic Undercut Feature Recognition for Side Core Design of Injection Molds. *Journal of Mechanical Design*, 126, pp. 519-526, 2004.
- [6] S. McMains, X. Chen. Finding undercut-free parting directions for polygons with curved edges. *Journal of Computing and Information Science in Engineering*, 6(1), pp. 60-68, 2006.
- [7] R. Kharderkar, G. Burton, and S. McMains. Finding feasible mold parting directions using graphics hardware. *Computer Aided Design*, 38(4), pp. 327-341, 2006.
- [8] Y. Chen, D. W. Rosen. A region based method to automated design of multi-piece molds with application to rapid tooling, *Journal of Computing and Information Science in Engineering*, 2(2), pp. 86-97, 2002.
- [9] Y. Chen, D. W. Rosen. A reverse glue approach to automated construction of multi-piece molds, *Journal of Computing and Information Science in Engineering*, 3(3), pp. 219-230, 2003.
- [10] A. Priyadarshi, S. K. Gupta. Geometric algorithms for automated design of multi-piece permanent molds. *Computer-aided Design*, 36(3), pp. 241--260, 2004.
- [11] J. Huang, S. K. Gupta, K. Stoppel. Generating sacrificial multi-piece molds using accessibility driven spatial partitioning. *Computer-Aided Design*, 35(13), pp. 1147--1160, 2003.
- [12] A. K. Priyadarshi, S. K. Gupta. Algorithms for generating multi-stage molding plans for articulated assemblies. *Robotics and Computer Integrated Manufacturing*, 32(3/4), pp.350-365, 2009.
- [13] L. L. Chen, S. Y. Chou, T. C. Woo. Parting directions for mould and die design” *Computer-Aided Design*, 25, pp. 762—768, 1993.
- [14] T. C. Woo, Visibility maps and spherical algorithms, *Computer-Aided Design*, 26(1), pp. 6–16, 1994.
- [15] S. Dhaliwal, S. K. Gupta, J. Huang, A. Priyadarshi. Algorithms for computing global accessibility cones. *Journal of Computing and Information Science in Engineering*, 3(3), pp.200-209, 2003.