CoreCavity: Interactive Shell Decomposition for Fabrication with Two-Piece Rigid Molds

KAZUTAKA NAKASHIMA, The University of Tokyo
THOMAS AUZINGER, IST Austria
EMMANUEL IARUSSI, CONICET, IST Austria
RAN ZHANG, IST Austria
TAKEO IGARASHI, The University of Tokyo
BERND BICKEL, IST Austria

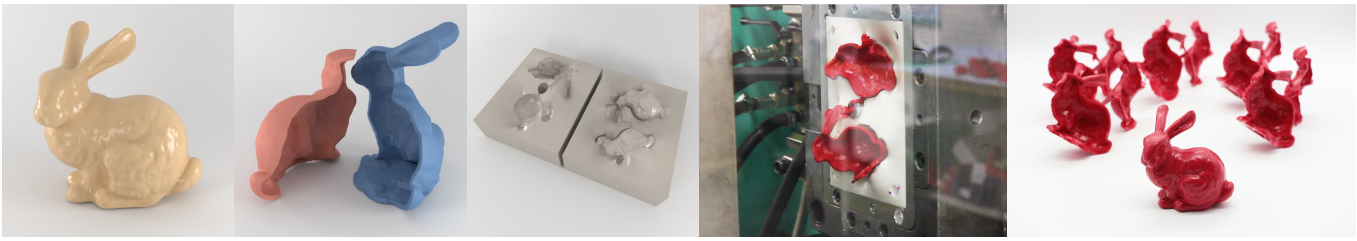


Fig. 1. From left to right: We propose a method to decompose an input shape (surface mesh) into molding-ready shell pieces and to generate the corresponding 3D printable molds. Using, for example, an injection molding machine, we can fabricate many physical replica at low cost and in a short amount of time.

Molding is a popular mass production method, in which the initial expenses for the mold are offset by the low per-unit production cost. However, the physical fabrication constraints of the molding technique commonly restrict the shape of moldable objects. For a complex shape, a decomposition of the object into moldable parts is a common strategy to address these constraints, with plastic model kits being a popular and illustrative example. However, conducting such a decomposition requires considerable expertise, and it depends on the technical aspects of the fabrication technique, as well as aesthetic considerations. We present an interactive technique to create such decompositions for two-piece molding, in which each part of the object is cast between two rigid mold pieces. Given the surface description of an object, we decompose its thin-shell equivalent into moldable parts by first performing a coarse decomposition and then utilizing an active contour model for the boundaries between individual parts. Formulated as an optimization problem, the movement of the contours is guided by an energy reflecting fabrication constraints to ensure the *moldability* of each part. Simultaneously, the user is provided with editing capabilities to enforce aesthetic guidelines. Our interactive interface provides control of the contour positions by allowing, for example, the alignment of part boundaries with object features. Our technique enables a novel workflow, as it empowers novice users to explore the design space, and it generates fabrication-ready two-piece molds that can be used either for casting or industrial injection molding of free-form objects.

Authors' addresses: Kazutaka Nakashima, The University of Tokyo, taka@ui.is.s.u-tokyo.ac.jp; Thomas Auzinger, IST Austria, thomas.auzinger@ist.ac.at; Emmanuel Iarussi, CONICET,, IST Austria, eiarussi@conicet.gov.ar; Ran Zhang, IST Austria, rzhang@ist.ac.at; Takeo Igarashi, The University of Tokyo, takeo@acm.org; Bernd Bickel, IST Austria, bernd.bickel@ist.ac.at.

CCS Concepts: • **Applied computing** → **Computer-aided manufacturing**; • **Computing methodologies** → *Shape analysis*; • **Mathematics of computing** → Mesh generation; • **Human-centered computing** → Graphical user interfaces;

Additional Key Words and Phrases: molding, fabrication, height field, decomposition

ACM Reference Format:
Kazutaka Nakashima, Thomas Auzinger, Emmanuel Iarussi, Ran Zhang, Takeo Igarashi, and Bernd Bickel. 2018. CoreCavity: Interactive Shell Decomposition for Fabrication with Two-Piece Rigid Molds . *ACM Trans. Graph.* 37, 4, Article 135 (August 2018), 13 pages. https://doi.org/10.1145/3197517.3201341

1 INTRODUCTION

In recent years, the rise of 3D printing technology has democratized the fabrication of complex shapes. Nevertheless, classical manufacturing approaches, such as molding, are often the preferred choice because of two main reasons: they scale extremely well with the amount of required copies, and they draw from a wide range of available base materials. Unfortunately, in contrast to 3D printing, which seemingly requires only a few clicks to replicate an object, obtaining a mold is much more challenging.

In this paper, we focus on designing re-usable rigid molds for *two-piece molding*, the simplest and most cost-efficient variant of molding. In two-piece molding, as illustrated in Fig. 2, two rigid mold pieces form a cavity filled with a liquid or pliable material, i.e., resin, plastic, metal, or ceramics, which then solidifies. Subsequently, the mold pieces are separated, and the cast object is released by a linear translation along a so-called *parting direction (PD)*. To make the removal possible, the top and bottom surfaces of the fabricated object must be representable as height fields for precluding overhangs and

overlaps. Previous work has extensively investigated finding the optimal parting directions [Khardekar et al. 2005], or methods to deform a given shape [Herholz et al. 2015], to obtain a valid height field configuration. Nevertheless, this constraint of molding makes faithfully reproducing many man-made and most organic objects as a single piece impossible. While introducing multi-piece molds and multiple parting directions helps overcome this limitation, our goal is to avoid adding complexity to the molding process and keeping our molds compatible with simple injection molding machines.

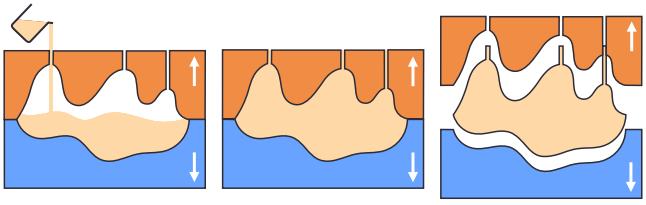


Fig. 2. Molding procedure with a two-piece mold. First, the mold pieces are assembled to form a cavity, into which the liquid object material is casted (left). Pockets of air are avoided by outlets in the top mold piece. After material solidification (center), the mold pieces are removed, and the object (right) remains.

A suitable strategy to overcome the aforementioned restrictions imposed by two-piece molds is the decomposition of the desired shape into a set of parts, in which each part itself satisfies the fabrication constraints (see Fig. 3). Such a decomposition needs to reconcile the technical guidelines for mold design with aesthetic considerations regarding the placement of boundaries between the individual parts. This task is highly complex, as the designer needs to anticipate the effect of decomposition on the manufacturability, aesthetics, and overall size of the mold, while trying to keep the number of individual parts small. Therefore, in practice, this task is usually reserved for experts.

We propose a computational approach that allows novice users to design a two-piece mold. Our interactive technique supports exploring the design space and finding a decomposition of a free-form shape into a set of solid parts. We assume that a target shape is provided as an orientable surface mesh. The inner surface is automatically generated as a thin-shell offset surface. In contrast to a solid fill, a thin shell reduces the required amount of material and the weight of the final object.

In this paper, we propose an energy formulation to quantify the quality of a part and solve a constrained optimization problem, in which the moldability is taken into account for each part. On this basis, for the purpose of decreasing the assembly cost, our method attempts to minimize the number of parts. We utilize a dedicated optimization method to compute the shape of the individual parts and consider both aesthetic and fabrication constraints. In addition, the integration of subjective user preferences by providing a set of intuitive interactive tools to edit the decomposition, steer the optimization process, and explore the solution space ensures that the designer stays in control over aesthetic considerations. We demonstrate the viability of our approach on a range of input shapes by creating molds that are used to fabricate a variety of models.

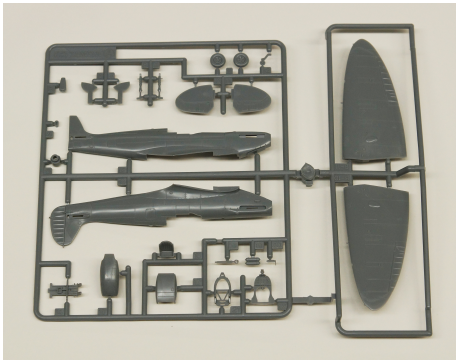


Fig. 3. Example of model airplane decomposed into a set of parts, fabricable with a two-piece mold.

Table 1. Comparison between our method and related work on mold design.

	[Herholz et al. 2015]	[Malomo et al. 2016]	Our method
Object parts	1	1	multiple
Mold pieces per part	multiple	1	2
Mold type	rigid	flexible	rigid
Mold assembly	complex	complex	simple
Representation	surface-only	surface-only	vol.-aware
Thin-shell supported	no	no	yes
Interactive editing	no	no	yes

2 RELATED WORK

Computer-aided design and molding. A comprehensive introduction and overview of recent techniques can be found in the work of Kazmer [2016]. Because of its industrial relevance, computer-aided mold design has received significant attention [Fuh et al. 2004]. While a mold design might be influenced by a multitude of aspects, such as cost, fabrication speed, required amount of casting material, material flow during molding, life-span of the mold, and aesthetics, a fundamental task in designing molds is to geometrically verify and model shapes that can be fabricated with this technique. In the remainder of this section, we will focus on this aspect.

Commonly, moldability is tested by searching for parting directions that allow the opening of the mold without interlocking with the fabricated object or the mold itself [Chakraborty and Reddy 2009; Inui et al. 2014; Khardekar et al. 2005; Zhang et al. 2009]. Optimal parting directions are usually determined by trying to automatically recognize and avoid undercut features [Nee et al. 1997]. Additionally, criteria determining the quality of parting lines at which individual mold pieces join have been modeled to analyze given designs [Ravi and Srinivasan 1990], or to even compute these parting lines automatically, with a parting direction as input [Li et al. 2009].

To extend the range of moldable shapes, some authors have explored the use of multi-piece molds for dealing with concave regions that tend to generate overhangs [Herholz et al. 2015; Lin and Quang 2014; Priyadarshi and Gupta 2004]. These methods decompose an input surface into several height field patches for which corresponding mold pieces are computed. The approach by Lin et al. [2014] starts by identifying a small set of parting directions, and then follows a set of rules to assign surface facets to mold-piece regions. Herholz et al. [2015] used a graph cut-based approach to decompose and, as necessary, deform the surface of an object to approximate free-form geometry with height fields.

Common to all these approaches is that they fall into the category of *surface segmentation* methods. In general, for these methods, no obvious extension to the segmentation of solids with enclosed voids exist, because, for instance, for an outer surface patch, they have no knowledge of parting surfaces inside the volume and the orientation of inner surfaces, making the identification of the moldability of the resulting parts infeasible. Furthermore, a valid height field surface segmentation cannot be easily converted into moldable shells because of the potential intersections of extruded shells. This case is especially challenging in the presence of thin features, as shown in Fig. 5, because inner surfaces might touch and therefore require these regions to be treated as a single volume.

To prevent these issues, our proposed method performs a volume-based decomposition of the shape.

Relaxing the rigidity constraint, Malomo *et al.* [2016] used a computational approach to design flexible molds that can undergo considerable deformation during removal. However, the molds must be elaborately sealed before casting, thus compromising the time saving potential for which molding is used in the first place. Additionally, the extension to thin-shelled objects is non-trivial. Moreover, flexible molds are not suited for injection molding, as they cannot withstand the high pressure during injection. Tab. 1 summarizes the main differences and similarities between our method and the aforementioned approaches.

Decomposition for fabrication. Fabricating shapes as sets of parts that are subsequently assembled is a prominent strategy in manufacturing. In the field of computer graphics, previous work includes the decomposition into pyramidal pieces, which can be 3D printed without any support structure because of the lack of overhangs [Hu et al. 2014]. Wang *et al.* [2016] decomposed a polygonal model to minimize 3D printing artifacts by considering anisotropic printing resolution. Furthermore, decomposition allows users to print large shapes beyond the build size limitations of commodity 3D printers [Chen et al. 2015; Luo et al. 2012; Vanek et al. 2014; Yao et al. 2015]. Alemanno *et al.* 2014 [2014] proposed a technique to decompose a 3D digital shape into a set of interlocking pieces that can be represented as simple height fields. It relies on a manually provided very coarse polyhedral approximation of the input shape, which determines the decomposition. Currently, the problem of finding such a coarse approximation of a geometric shape is an open problem and requires further investigation to be solved in an automatic and robust manner on complex 3D models. Gao et al. [2015] introduced a multi-directional printing system with a revolving cuboidal build platform. While they also proposed a segmentation method into axis-aligned height field blocks to reduce the print and support material use, our problem setting differs significantly. Instead of being bound to an axis, we are trying to obtain a segmentation that ensures moldability, while exploring a large number of parting directions and free-form cuts.

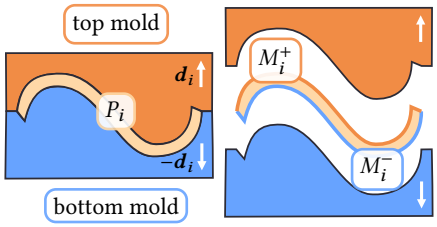
Commercial software for molding. Tools for computationally designing molds have also been commercially explored because of their significant industry relevance. Popular products, such as Moldflow, Moldex3D, or CADMOULD, simulate the molding process and are used for computer-aided engineering (CAE) verification and design improvement. However, these workflows are usually intended for experts, require proficient knowledge of the CAD packages they are interfacing with, such as Autodesk Fusion 360, SolidWorks, or CATIA, and start with an initial mold design created by an engineer. By contrast, our tool can fully and automatically decompose a given free-form shape into shell parts and provide a free-form parting line per part for efficient two-piece molding. Its interface is simple enough to be used without extensive training. Our approach is complementary and, in theory, could be combined with existing CAE tools.

3 PROBLEM FORMULATION

As input geometry, we assume a closed surface $\mathcal{S}_{\text{input}}$, which we want to fabricate using two-piece molding. We require the surface to be a discrete, connected, and suitably oriented manifold without self- or pairwise intersections and given as a triangle mesh. As output, we provide a set $\mathcal{P} = \{P_i : i = 1, \ldots, N_{\mathcal{P}}\}$ of $N_{\mathcal{P}}$ *parts*, which constitute a decomposition of a thin shell volume $\widetilde{V}$ of the potentially deformed input shape. The decomposition is both disjoint and covers all of the shell $\widetilde{V}$, i.e.,

$$P_i \subset \widetilde{V}, \quad P_i \neq \emptyset, \quad \bigcup_{i=1}^{N_{\mathcal{P}}} P_i = \widetilde{V}, \quad P_i \cap P_j = \emptyset \quad \text{for } i \neq j.$$

Each part P_i fulfills the fabrication constraints of two-piece molding and its surface is divided into two patches M_i^+ and M_i^-. The two patches correspond to the surface of the two mold pieces and they share the identical boundary (parting line), and again they are connected (but not necessarily simply connected), oriented, and free of intersections.



Additionally, each part is equipped with its parting direction $\boldsymbol{d}_i$ along which the mold pieces are assembled and disassembled. Note that we do not allow different parting directions for the two mold pieces to reduce fabrication complexity.

A fundamental constraint of molding is the required absence of overhangs and overlaps, which can inhibit the removal of the mold after the liquid material has been cast and solidified. Thus, the faces M_i^+ and M_i^- of the two mold pieces of a part P_i with parting direction $\boldsymbol{d}_i$ need to constitute height fields. As a necessary condition, we require for all triangles $t^+ \in M_i^+$ (resp. $t^- \in M_i^-$) that

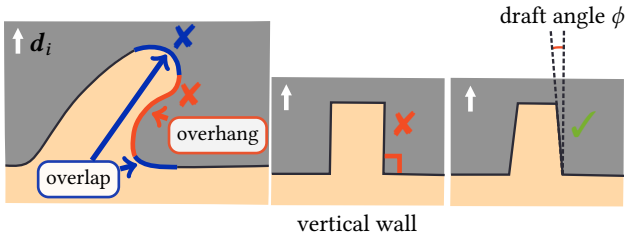


Fig. 4. Restrictions of molding. Overhang constraint (left, red): all normals of a mold face have to point into the same hemisphere as the parting direction $\boldsymbol{d}$. Global overlap constraint (left, blue): self-intersections when projecting the face along its parting direction $\boldsymbol{d}$ must not occur. Draft angle constraint (right): all triangles must be slightly tilted so that angles between parting direction $\boldsymbol{d}$ and their normal is smaller than $\pi/2$.

their outwards pointing normals $\boldsymbol{n}_{t^+}$ (resp. $\boldsymbol{n}_{t^-}$) satisfy

$$\boldsymbol{d}_i \cdot \boldsymbol{n}_{t^+} \geq 0 \quad (\text{resp. } \boldsymbol{d}_i \cdot \boldsymbol{n}_{t^-} \leq 0)$$

as illustrated in Fig. 4 (left, red). In practice, a completely vertical wall is inappropriate for molding (Fig. 4 (middle)), and all triangles must be tilted with a small angle ϕ, which is known as the *draft angle* (Fig. 4 (right)).

$$\arccos(\boldsymbol{d}_i \cdot \boldsymbol{n}_{t^+}) \leq \frac{\pi}{2} - \phi \quad (\text{resp. } \arccos(\boldsymbol{d}_i \cdot \boldsymbol{n}_{t^-}) \geq \frac{\pi}{2} + \phi) \quad (1)$$

Global overlaps can occur (see Fig. 4 (left, blue)), and they are characterized by an overlap of the triangles of either M_i^+ or M_i^- when projected via $p_{\boldsymbol{d}_i}(x) = (I - \boldsymbol{d}_i \boldsymbol{d}_i^{\mathrm{T}})\, x$ along the corresponding parting direction $\boldsymbol{d}_i$. To avoid this, we require that

$$p_{\boldsymbol{d}_i}(t_j) \cap p_{\boldsymbol{d}_i}(t_k) = \emptyset \quad (2)$$

for all pairs of disjoint triangles t_j and t_k of either M_i^+ or M_i^-. We call this kind of violation (i.e., overhang and overlap) *undercut*.

4 METHOD

The design of our method for solving the decomposition problem stated in the previous section is based on the requirement of providing an *interactive* user experience. Interactivity is essential for exploring the design space, understanding the trade-offs between aesthetic considerations and fabrication limitations, and for achieving a subjectively satisfying result while still assuring moldability. Because of the computational complexity of the problem, obtaining feasible solutions in short time frames is generally not possible. We use two key strategies to mitigate this problem: (i) we perform the decomposition on a lower resolution and then restore the fine details from the original input surface $\mathcal{S}_{\text{input}}$ as a post-process (see §4.2 and §6) and (ii) we follow a coarse-to-fine approach. First, we perform a coarse decomposition by using region growing based on triangles as atomic elements. Second, we optimize and smooth the boundary by using an active contour representation that can move continuously over the surface to achieve a visually preferable and easier moldable decomposition.

In contrast to automatically trying to find a global optimum from scratch, such as, for example, using graph cut-based approaches [Herholz et al. 2015] in which small user input changes could lead to a drastically different solution, we advocate for an approach that has three key advantages that make it suitable for user interaction: (i) If desired, users can directly control the number of parts and their boundaries, which enables them to reflect their intent on the obtained result; (ii) user interaction affects the decomposition locally, and this makes it possible to concentrate on editing the region of interest; and (iii) the decomposition procedure and the effects of user interactions are visualized at interactive rates, thereby providing intuitive editing operations to non-expert users.

4.1 Overview

The workflow of our method is illustrated in Fig. 5 and Alg. 1. We begin with remeshing the input mesh $\mathcal{S}_{\text{input}}$, generating a low-resolution representation $\mathcal{S}_{\text{low}}$. Then, we extrude the surface $\mathcal{S}_{\text{low}}$ and generate a volumetric representation V to be decomposed (§4.2). During the shell generation, thin features are detected and marked to be fabricated as solids because they might be too fragile when split in separate shells. (Fig. 5 (middle left)). Over this shelled object, we perform a coarse decomposition (§4.3) by using a volume-aware region growing strategy, and generate a set of *parts* (Fig. 5 (middle)). For generating a decomposition with a small part count $N_{\mathcal{P}}$, we first try to decompose the object into two parts, and then increase the part count one by one until the moldability energy becomes smaller than the user-defined threshold γ. For each part, we compute the parting line for two-piece molding using a graph cut-based surface classification (§4.4). This results in a pair of connected surface *patches* that define the top and bottom surface of the part in the mold (orange and blue in Fig. 5 (middle right)). Using the obtained solution as an initial guess, we then represent the boundaries between parts as active contours that can continuously slide over the surface. We optimize these boundaries based on an energy that quantifies their smoothness and the moldability of the parts (§4.5). During this optimization, the parting lines of individual parts are updated as well.

Both before and after the coarse part decomposition and subsequent boundary optimization, users can add design constraints to reflect their intent (§5), while the solution is automatically updated. Finally, our system performs post-processing for restoring detail of $\mathcal{S}_{\text{input}}$ (§6.1) and moldability verification (§6.2), and then generates the mold geometry for fabrication. Runner structures (pipes for liquid material flow) are automatically added. After the molds are milled or 3D printed, a large number of copies can be efficiently fabricated using either resin casting or injection molding (§6.3).

4.2 Pre-computation

In the pre-computation stage, we prepare a volumetric shape representation, which is then used for decomposing the shape into moldable parts. To ensure the interactivity of our method, we start by computing a low-res representation $\mathcal{S}_{\text{low}} = (V_{\text{low}}, T_{\text{low}})$ of our input surface $\mathcal{S}_{\text{input}}$ with the use of instant field-aligned meshes [Jakob et al. 2015], where V_{low} and T_{low} denote vertices and triangles of $\mathcal{S}_{\text{low}}$, respectively. In our experiments, a triangle count of $N_\Delta \lesssim 10000$ provided a good trade-off between speed and accuracy. We also remesh $\mathcal{S}_{\text{input}}$ as necessary to obtain a high-quality, uniformly meshed high-resolution representation.

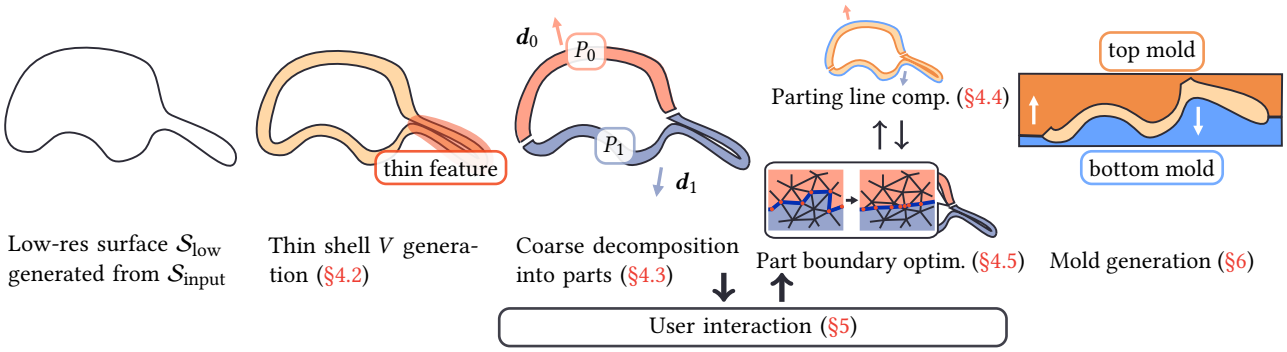
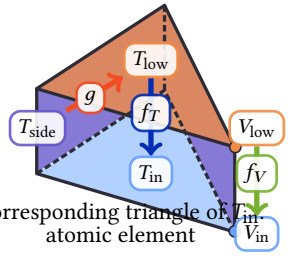


Fig. 5. We first compute an offset surface to obtain a thin-shell object (left). During the shell generation, we detect thin features (red) to be fabricated as solids (center left). Then, the shell object is decomposed into parts, and the parting line and part boudaries are optimized (center, center right). Users can reflect their design goal to the decomposition by user interaction. Finally, mold geometries are generated from the parting line (right).

Algorithm 1 Abstract workflow

Input: watertight and self-intersection free mesh $\mathcal{S}_{\text{input}}$
generate low-res representations $\mathcal{S}_{\text{low}}$ from $\mathcal{S}_{\text{input}}$ (§4.2)
generate columetric representation from $\mathcal{S}_{\text{low}}$ (§4.2)
$N_{\mathcal{P}} \leftarrow 2$
repeat
  user adds design constraints (§5)
  repeat decomposition into $N_{\mathcal{P}}$ parts
    volume-aware coarse decomposition (§4.3)
    boundary smoothing (§4.5) with parting line comp. (§4.4)
    if moldability energy $E_P >$ threshold γ **then**
      $N_{\mathcal{P}} \leftarrow N_{\mathcal{P}} + 1$
    end if
  until moldability energy becomes smaller than threshold γ
until user satisfied with the decomposition
generate mold geometry (§6)
fabricate target shape with two mold pieces (§6.3)

We aim to fabricate our shape as a thin-shelled object with an approximate user-specified wall thickness ω, which extends into the interior. To generate the internal surface of the thin shell, we push the vertices V_{low} inward along the inverse of the gradient direction of the signed distance field (< 0 inward and > 0 outward). In our implementation, we use the method of Jacobson et al. [2013] for signed distance calculation. While this approach might result in an internal surface with potentially degenerated triangles, flipped triangles, or self-intersection, our technique is robust against such deficiencies. Because of our internal surface generation, we have bijective mappings $f_V : V_{\text{low}} \rightarrow V_{\text{in}}$ and $f_T : T_{\text{low}} \rightarrow T_{\text{in}}$, where V_{in} and T_{in} denote the vertices and triangles of the internal surface, respectively. We define our *atomic elements* as triangle prisms whose top face is a triangle of T_{low}, and whose bottom face is the corresponding triangle of T_{in}.



Note that the quadrangle side faces of our atomic elements (i.e., triangle prisms) T_{side} are triangulated to ensure that all polygons are triangles, and we also have a surjective mapping $g : T_{\text{side}} \rightarrow T_{\text{low}}$.

Thin features (e.g., ears of bunny) require special attention during offsetting, as shells intersect. While an option would be to restrict the wall thickness to prevent intersections, in practice this might lead to thin and fragile parts. Instead, we fabricate such parts as solids (i.e., without an internal void) to increase strength and to improve the flow of the liquid material during the casting process. To detect thin features, we use a shape diameter function (SDF)-based segmentation [Shapira et al. 2008] over $\mathcal{S}_{\text{low}}$. For a point on the surface of the shape, its SDF value is twice the approximated distance to the medial axis. For robust detection, we identify segments whose minimal SDF value is smaller than twice the desired wall thickness ω and whose average SDF value is also smaller than three times ω. The label for thin features are propageted from triangles $\in T_{\text{low}}$ to atomic elements.

During the whole decomposition, we use uniformly sampled directions $\mathcal{D}$ as candidates of the parting directions. Thanks to this pre-defined set of candidate directions, we can pre-compute overhangs and overlaps to drastically speed up our workflow.

4.3 Coarse Shell Decomposition

Estimating the minimum required number of parts for a moldable decomposition is a challenging problem. Fekete and Mitchell [2001] showed that deciding if a 3D model of genus 0 can be decomposed into k polyhedra that resemble height fields is NP-hard. Therefore, we use a simple yet reasonable greedy approach to decompose the target shape into $N_{\mathcal{P}}$ moldable parts. We start with $N_{\mathcal{P}} = 2$ (recall that our target shape is a shelled object and therefore requires at least two parts), and iteratively increase $N_{\mathcal{P}}$ until a satisfying decomposition has been found.

For our scenario, we have two requirements for an atomic element-wise decomposition: (i) we want to explicitly control the number of segments and (ii) the resulting segments should be connected. We perform a decomposition similar to variational shape approximation (VSA) [Cohen-Steiner et al. 2004] over our atomic elements. In VSA, each region is represented by a pair of triangle and direction.

This triangle serves as a seed for growing the region *as flat as possible* with respect to the provided direction. We modify VSA to take moldability criteria into consideration, and each region represents a part in our decomposition. For a decomposition with $N_{\mathcal{P}}$, we first select $N_{\mathcal{P}}$ elements randomly, and for each region r, we initialize the parting direction $\boldsymbol{d}_r$ with $\boldsymbol{n}_t$.

We then grow regions iteratively by assigning elements with a minimum cost:

$$C(t,r) = \begin{cases} -|\boldsymbol{n}_t \cdot \boldsymbol{d}_r| + \delta & t \text{ belongs to a thin feature} \\ -\boldsymbol{n}_t \cdot \boldsymbol{d}_r + \delta & \text{otherwise,} \end{cases}$$

$$\delta = \begin{cases} 0 & t \text{ does not have overlap with } r \\ w_O & \text{otherwise.} \end{cases}$$

$C(t,r)$ represents the cost for adding an element t to a region r, where $\boldsymbol{d}_r$ is the parting direction for region r, and $\boldsymbol{n}_t$ denotes its outer surface triangle normal.

Intuitively, this energy measures how flat a triangle is while taking overlaps into consideration. For outer triangles belonging to thin features, we allow flips because almost half of the outer triangles of thin features touch the top mold and the others touch the bottom mold (see the thin feature and mold pieces in Fig. 5).

Once all elements are assigned to a region, we find new seeds for the next iteration. We first calculate the best direction for each region by computing $\arg\min_{\boldsymbol{d}\in\mathcal{D}} \sum_{t\in r} C(t,r)$. Then we compute best seed triangle for each region by computing $\arg\min_{t\in T_r} C(t,r)$, where T_r is a set of triangles assigned to a region r. If the seeds are unchanged or we reach an upper limit of iterations (we limit the number of iterations up to 20 times as Cohen-Steiner *et al.* [2004] suggested), we finish the coarse decomposition.

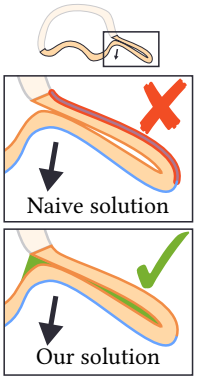
During coarse decomposition, users can add no-cut design constraints (§5) with a brush-like interface.

A no-cut constraint is given as a set of edge-connected triangles u_i over the outer surface, and all atomic elements whose top triangles belong to u_i will be assigned to the same region. To ensure this, we treat the set of atomic elements corresponding to u_i as a single element for region growing. The cost for assigning an atomic element whose outer triangle $t \in u_i$ to a region r is calculated as $\sum_{t_i\in u_i} C(t_i,r)$, and when assigning to a region, all other atomic elements corresponding to u_i are assigned simultaneously.

As output, we obtain $N_{\mathcal{P}}$ parts. Then, for each part we compute an optimal parting line.

4.4 Parting Line Computation

Because our targeted fabrication technique is two-piece molding, we need to classify the surface of a part P_i into two connected patches, M_i^+ and M_i^- (blue and orange in the inset figure). While a tempting naive solution would be to simply assign all outer surfaces of our elements to M_i^+, this approach would fail, for example, in thin areas. With the naive solution, the blue patch has a large overhang (red) and this would cause large deformations. With our solution, the undercut caused by T_{in} is removed by adding some material (green) inside.



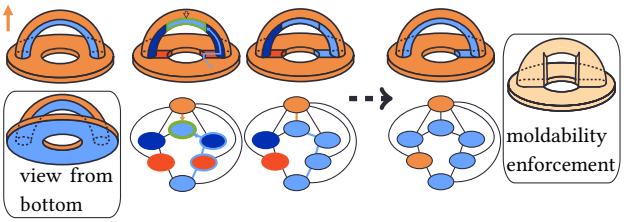


Fig. 6. Surface classification and moldability enforcement. The part consists of a half solid torus glued to a flat ring. M^- (light blue) is not single connected. We iteratively change the labeling and finally obtain a single connected component for M_i^+ and M_i^-.

Therefore, we propose to perform a graph cut-based classification over the surface of each part. The energy terms for the graph cut are

$$E_{unary,\ upper}(t) = \begin{cases} 0 & t \in T_{\text{in}} \\ 1.0 + \boldsymbol{n}_t \cdot \boldsymbol{d} & \text{otherwise} \end{cases}$$

$$E_{unary,\ lower}(t) = \begin{cases} 0 & t \in T_{\text{in}} \\ 1.0 + \boldsymbol{n}_t \cdot (-\boldsymbol{d}) & \text{otherwise} \end{cases}$$

$$E_{binary}(t_0,t_1) = \max(0, \boldsymbol{n}_{t_0} \cdot \boldsymbol{n}_{t_1}),$$

where $\boldsymbol{n}_t$ is the normal vector of triangle t and $\boldsymbol{d}$ is the corresponding parting direction for P_i. For unary terms $E_{unary,\ upper}$, $E_{unary,\ lower}$, we do not care if $t \in T_{\text{in}}$ causes some undercuts because the inside is completely invisible after assembly, and we can add some material to remove such undercuts. Adding some material and making the shell thicker also helps liquid material flow during fabrication. For the binary term E_{binary}, we encourage the boundary between M_i^+ and M_i^- to align to the edges with sharp dihedral angles (e.g., part boundary) for suppressing the visual artifacts caused by parting lines.

Recall that we need a single connected component for M_i^+ and M_i^- because of the nature of two-piece molding. However, this graph cut does not guarantee each segment is a single connected component, which occurs when there are remaining undercuts because of the relaxed moldability constraints. Therefore, we tweak the classification after the graph cut and ensure that both of M_i^+ and M_i^- are single connected components (Fig. 6). After the graph cut, each triangle has its label for the classification (top first column). We first update their labels according to whether triangles have overlap. Specifically, we have four types of labels (i) M_i^+ without overlap (orange), (ii) M_i^+ with overlap (red), (iii) M_i^- without overlap (light blue), and (iv) M_i^- with overlap (blue) (top second column).

Second, over connected components of these labels, we construct a graph with asymmetric consts for graph edges (bottom 2nd column).

$$w(c_0,c_1) = \begin{cases} 0 & \text{if both } c_0 \text{ and } c_1 \text{are assigned} \\ & \text{to the same segment } (M_i^+ or M_i^-) \\ |p_{\boldsymbol{d}}(c_1)| & \text{otherwise ,} \end{cases}$$

where $p_{\boldsymbol{d}}(c)$ is the projected area of connected component c. We iteratively change the label of these nodes and finally obtain a single

connected components for M_i^+ and M_i^-. In this graph, we first find the largest M_i^+ component and M_i^- component in terms of projected area (top and bottom nodes), and fix the labeling of these nodes. Then, in projected area's decreasing order, we compute the shortest path from the largest M_i^+ (orange) and M_i^- (light blue). If the path from the largest M_i^+ is shorter, we change the labels of all nodes along the path to M_i^+. Otherwise, we change to M_i^-. For example, in Fig. 6, path from the largest M_i^- is shorter and all labels of nodes along the path are changed to M_i^-. In addition, the labels along the path are fixed and every time the labeling are updated, the costs for the graph edges are also updated. This operation is repeated until the labels of all nodes are fixed.

4.5 Part Boundary Smoothing

The boundaries of the parts after region growing follow a set of edges of $\mathcal{S}_{\text{low}}$. Because of the discrete representation, they are usually jaggy and therefore reflect undesired aesthetic and suboptimal moldability conditions (see §4.3). We improve the solution by representing the boundary between parts as an active contour model, also called snakes, and perform moldability-aware smoothing and local optimization of the snakes. This approach, as demonstrated by our results, provides aesthetic and molding-friendly smooth boundaries.

We use the same definition as that utilized by Bischoff *et al.* [2005] to define the boundaries. Snakes are constituted by a set of snake vertices that lie on the edges or vertices of $\mathcal{S}_{\text{low}}$ and by a set of snake edges connecting snake vertices. The position is represented as weighted sum $\alpha \boldsymbol{v}_0 + (1-\alpha)\boldsymbol{v}_1$, where $\boldsymbol{v}_0$ and $\boldsymbol{v}_1$ are the endpoints of a mesh edge. To decompose the shelled object, we add the corresponding points on T_{in} as $\alpha f_V(\boldsymbol{v}_0) + (1-\alpha) f_V(\boldsymbol{v}_1)$ and subdivide the triangles in T_{low} and T_{in} accordingly. Additionally, we also triangulate the sides.

4.5.1 Objectives. The two moldability constraints given by Eq. 1 and Eq. 2 are hard constraints. Similar to Herholz *et al.* [2015], we relax these constraints and quantify their violation with a smooth *moldability energy* function.

Formally, we cast the boundary optimization as a minimization problem,

$$\underset{\text{snakepos}}{\arg\min}\, E = \sum_{i=1}^{N_{\mathcal{P}}} E_P(P_i) + w_l E_{\text{smooth}}\,, \tag{3}$$

where $E_P(P_i)$ measures the moldability for P_i,E_{smooth} quantifies the smoothness of the boundary, and w_l is used to reflect the users' preferred smoothness of the part boundary.

4.5.2 Moldability Energy. To quantify the violation of the moldability constraints (i.e., estimated amount of deformation), we define the corresponding energy term E_P as

$$E_P(P_i, \boldsymbol{d}_i) = w_H E_H(P_i, \boldsymbol{d}_i) + w_L E_L(P_i, \boldsymbol{d}_i) \tag{4}$$

with the w_H-weighted overhang penalization E_H given as

$$E_H(P_i, \boldsymbol{d}_i) = \sum_{t^+ \in M^+} U(t^+, \boldsymbol{d}_i) + \sum_{t^- \in M^-} U(t^-, -\boldsymbol{d}_i)$$

$$U(t, \boldsymbol{d}) = \begin{cases} -\cos\Big(\min\big(\theta + \phi, \pi\big)\Big)|t| & \theta > \frac{\pi}{2} - \phi \;\wedge\; t \notin T_{\text{in}} \\ 0 & \text{otherwise}\,, \end{cases}$$

where θ represents an angle between the normal of the triangle $\boldsymbol{n}_t$ and parting direction d, namely, $\theta = \arccos(\boldsymbol{n}_t \cdot \boldsymbol{d})$. Essentially, we sum the areas $|t| = |p_{\boldsymbol{d}}(t)|$ of triangles t projected along $\boldsymbol{d}$ for all triangles that violate Eq. 1. To ensure that the triangles are tilted with more than draft angle ϕ, we shift the angle between $\boldsymbol{n}_t$ and $\boldsymbol{d}$ by ϕ. Additionally, we ignore the inner triangles T_{in} because we can resolve the violations they caused without any visual artifacts, as previously explained.

We use the same strategy to penalize overlaps that violate Eq. 2 with the w_L-weighted energy term

$$E_L(P_i, \boldsymbol{d}_i) = \sum_{t_1^+, t_2^+ \in M^+} O(t_1^+, t_2^+, \boldsymbol{d}_i) + \sum_{t_1^-, t_2^- \in M^-} O(t_1^-, t_2^-, -\boldsymbol{d}_i)\,,$$

where the projected area of the overlap of two triangles is penalized. We define the overlaps as

$$O(t_1, t_2, \boldsymbol{d}) = \begin{cases} \frac{1}{2}|p_{\boldsymbol{d}}(t_1) \cap p_{\boldsymbol{d}}(t_2)| & \begin{array}{c} t_1 \neq t_2 \\ \wedge\, (\boldsymbol{n}_{t_1} \cdot \boldsymbol{d})(\boldsymbol{n}_{t_2} \cdot \boldsymbol{d}) \geq 0 \\ \wedge\, t_1, t_2 \notin T_{\text{in}} \end{array} \\ 0 & \text{otherwise}\,, \end{cases}$$

avoiding a second penalization of overhang triangles. In the same manner with E_H, we ignore the triangles in T_{in}. Large w_L and w_H enforce moldability constraints more strictly. Note that for a given set $\mathcal{D}$ of parting directions, the projected areas of both $|p_{\boldsymbol{d}}(t)|$ and $|p_{\boldsymbol{d}}(t_1) \cap p_{\boldsymbol{d}}(t_2)|$ can be precomputed, thus removing these costly computations from the innermost loop of our method.

4.5.3 Regularization. To obtain a aesthetically pleasing boundary, smooth decompositions are often desired. Furthermore, in practice, smooth boundaries are preferred as a general design guideline because jagged and thin parts might break off. To address these design considerations, we use a smoothing energy term

$$E_{\text{smooth}} = \sum_i \frac{(x_{i-1} - x_i) \cdot (x_{i+1} - x_i)}{\|(x_{i-1} - x_i)\|\,\|(x_{i+1} - x_i)\|} \Big/ \#\text{snakes vertex}\,,$$

measuring the average of angles between consecutive line segments at snake vertex positions x_i .

4.5.4 Optimization. As Eq. 3 depends on the global geometry configuration, computing an analytic gradient is non-trivial. Instead, we first evaluate a desired displacement direction for each vertex, which we choose as the gradient of the smoothness energy. In this subspace, we compute the gradient of Eq. 3 by projecting new snake positions onto the closest surface and using finite differences. We then apply simple gradient descent and recompute the parting lines of the newly obtained parts after each iteration (§4.4).

5 USER INTERACTION

Before and after the decomposition, users can edit the decomposition by using the following four functions: no-cut, cut, merge, and selective smoothing (Fig. 7). Before the decomposition, users can specify sets of edge-connected triangles, and these sets will be fabricated without division. This is achieved by assuming that the sets are single elements in region growing, as explained. After the decomposition, users can edit the boundary with the cut, merge, and selective smoothing functions. To cut the parts, we simply introduce new snake vertices and snake edges according to the user input. After cutting the parts, we update the corresponding parting directions for each divided part by computing $\arg\min_{\boldsymbol{d}\in\mathcal{D}} E_P(P_i, \boldsymbol{d})$ with fixed part P_i. To merge parts p_0 and p_1 that are neighbors, we compute the new part $p_2 = p_0 \cup p_1$ and the best parting direction by $\arg\min_{\boldsymbol{d}\in\mathcal{D}} E_P(P_2, \boldsymbol{d})$. Users can apply the smoothing to user-selected vertices to increase smoothness locally. During user interaction, the moldability energy is tracked. When it exceeds the threshold γ, the system provides a warning. Furthermore, the estimated amount of deformation is visualized to allow users to see where the shape might be deformed.

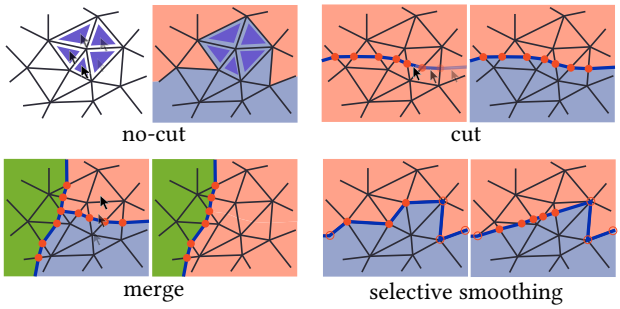


Fig. 7. During interactive editing, users can edit the decomposition using four functions: no-cut, cut, merge, and selective smoothing.

6 MOLD GENERATION

After the satisfactory decomposition of the thin shelled low-res representation V into a set of parts $\mathcal{P}$ was achieved, the corresponding molds can be automatically generated. Before actual fabrication (see §6.3), we still need to correct the following three issues that stem from the approximations we used to guarantee interactivity: (i) useless inner triangles of thin features still remain inside of the solid hull of thin features, (ii) surface details were lost during the remeshing (see §4.2), and (iii) because of the relaxation of the moldability constraints (see Eq. 4), small overhangs or global overlaps can be present along the parting directions. We solve these issue in the same order and start by obtaining surface meshes that fully enclose thin parts. After performing detail restoration (see §6.1), we eliminate all remaining violations of the moldability constraints by using swept volumes (see §6.2).

6.1 Detail Restoration

To restore the fine details of the input surface $\mathcal{S}_{\text{input}}$, which were lost during the initial remeshing for generating low-res representation, we augment the outer surface (i.e., $\mathcal{S}_{\text{low}}$) with the original surface geometry. As illustrated in Fig. 8, we use a two-step process. First, the outer surface $\mathcal{S}_{\text{low}}$ of each part P_i is offset outwards such that the corresponding patch of $\mathcal{S}_{\text{input}}$ is fully contained in the thereby thickened part $\widehat{P}_i$. By computing the intersection of $\mathcal{S}_{\text{input}}$ and $\widehat{P}_i$, we obtain a high-detail replacement for the outer surface $\mathcal{S}_{\text{low}}$. Note that the interior surface is kept as is because it is generally not visible in the assembled model. In the remainder of this section, we use P_i to refer to this *replacement*.

6.2 Moldability Enforcement

As we relaxed the moldability constraints and as the outer mold face was replaced with a different geometry, we cannot guarantee that the individual parts P_i of the decomposition are strictly moldable; minor overhangs or global overlaps could still persist. To eliminate all such artifacts, we follow a similar strategy as that used by Herholz *et al.* . [2015]. Because our fabrication is two-sided, we apply the moldability enforcement to M_i^+ and M_i^- independently. We first try to resolve most violations (i.e., overhangs and overlaps) by an as-rigid-as-possible (ARAP) [Sorkine and Alexa 2007] based deformation. The remaining violations are addressed using swept volumes. We compute swept volumes along the parting directions $\boldsymbol{d}_i$. As illustrated in Fig. 9, each of the two mold faces, M_i^+ and M_i^-, can be swept along its corresponding parting directions $\boldsymbol{d}_i$ and $-\boldsymbol{d}_i$

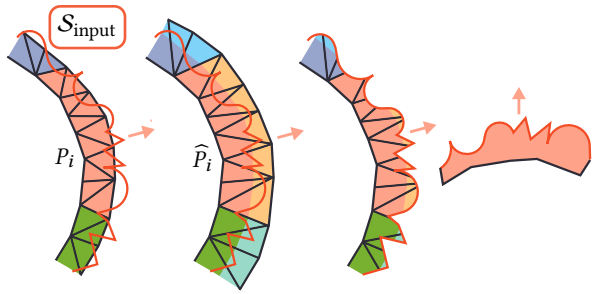


Fig. 8. Detail restoration. A part P_i resulting from the decomposition of the low-resolution representation $\mathcal{S}_{\text{low}}$ (left) is offset to contain the original high-resolution surface $\mathcal{S}_{\text{input}}$ (center). A high-detail replacement of the outer surface P_i is obtained by intersecting both.

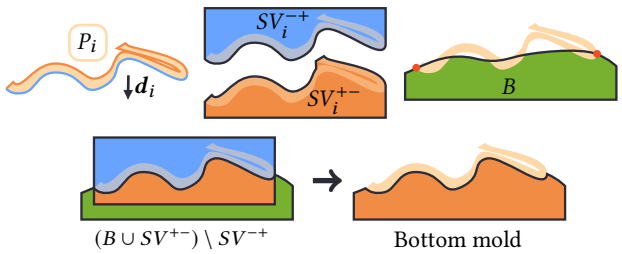


Fig. 9. Sweeping along $\boldsymbol{d}_i$ and $-\boldsymbol{d}_i$ removes the remaining undercut. In addition, we also compute the smooth touching surface (B) that passes through the parting line (red circle). The final mold geometries are generated by mesh boolean with these volumes.

or along the opposite directions. In case swept volumes are used, the potential intersections of the added volume with other object parts need to be checked and removed. The two resulting swept volumes SV_i^{+-} and SV_i^{-+} can be combined using boolean operations to yield fabricable mold pieces.

6.3 Sprues, Runners, and Design Finishing

With the aforementioned pieces, we assembled the mold geometry by considering the properties of the casting process. As shown in Fig. 2, the assembled mold requires in- and outlets to allow for resin insertion and avoid the creation of air pockets. In addition, mold geometries have two requirements: (i) all parts need to be connected by runners (channel structures for material flow) and gates (connection between parts and runners), and (ii) the touching surface between the two mold pieces should exactly pass through the parting lines of all parts. To achieve such a mold geometry, we begin with arranging all parts so that their parting directions become the +Z direction. The smooth surfaces between the parting lines and the outer borders of the molds are generated by interpolating the height field with the use of radial basis functions with a Gaussian kernel and vertices on the parting lines as constraints [Carr et al. 2001]. For runners, we use simple pre-defined structures. For the gates, we simply place them at the lowest and locally highest points in the Y-coordinates on the parting line for each part and connect them with runners. Finally, we compute mesh boolean $(B \cup (\bigcup SV^{+-})) \setminus (\bigcup SV^{-+})$ for the bottom and top molds. In practice, mesh boolean for mold geometry generation and moldability enforcement can be performed concurrently.

In our system, users can make a common mold for all the parts (see the bunny and airplane model in Fig. 13), or separate molds for each part (see the others in Fig. 13). A large common mold reduces the effort for fabrication (i.e., manual casting). On the other hand, separate molds allow the fabrication of each part with a different material and the creation of a larger object by maximally scaling up the mold pieces to the printing volume of a 3D printer.

The final mold geometry can be realized using either 3D printing or milling. As the individual mold pieces are height fields, supporting structures are not required for 3D printing. Moreover, conventional three-axis milling – if the diameter of the milling bit and head allow – can also be used. We utilize photopolymerization based 3D printing to fabricate all mold pieces shown in the paper.

7 RESULTS

We used our system to decompose a variety of shapes and compute the corresponding molds; a detailed overview of our results is given in Fig. 11 and Fig. 13. For all examples, all models are scaled to fit into a unit cube, and we set the weight for penalizing overhangs and overlaps $w_H = 1.0, w_L = 1.0$, the weight for penalizing overlaps during the coarse decomposition $w_O = 10^6$, $w_l = 1.0$, the thickness of the shell $\omega = 0.03$, the draft angle $\phi = \frac{\pi}{180}$, and the user-defined threshold for the amount of deformation $\gamma = 0.05$. The set $\mathcal{D}$ of possible parting directions is given by the vertex coordinates of an icosphere with three subdivisions (162 vertices). Because of the symmetry of the icosphere, storing only one hemisphere of it and obtaining all remaining directions of $\mathcal{D}$ by mirroring is sufficient.

Statistics on the results, including mesh complexity and timings can be found in Tab. 2. For the precomputation of triangle–triangle overlaps for coarse decomposition, we use early rejection through axis-aligned bounding boxes. Thus, the actual number of projected triangle overlaps is the dominant factor with regard to runtime, which is in the order of seconds for all our low-res meshes $\mathcal{S}_{\text{low}}$. Additionally, column four to seven report statistics on the user interactions, including the number of no-cut, cut, and merge operations, as well as the timing on performing these individual operations.

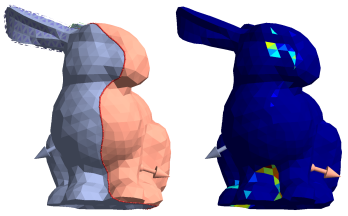
We started testing our system with simple shapes, such as sphere and torus (Fig. 11). These examples enable us to intuitively validate the quality of the decomposition. For a sphere, our system found a $N_{\mathcal{P}} = 2$ parts decomposition with two nearly perfect half-spheres, as expected. We obtained a similar result for a torus.

The boundary optimization method provides appealing and smooth boundaries in just a few steps. (Fig. 10) shows the fast decay of the smoothness energy for the bunny model, starting from an automatically obtained coarse initialization, while the moldability energy E_P stays approximately constant. In this example, the moldability energy is lower than the user-defined threshold $\gamma = 0.05$ (green), and this decomposition is acceptable.

A key feature of our method is its ability to be applicable to shapes with non-zero genus (see the torus, kitten, fertility, and sculpture examples). Furthermore, thin parts are successfully detected and suitable molds for their fabrication as solid pieces generated (see the wings of airplane, arms of the fertility model, and ears of the bunny). Since our method provides the user with the ability to specify the location of part boundaries, it is possible to create semantically meaningful moldable parts. As example, the Beethoven model was successfully decomposed into parts that reflect the coloring of the original model. We compared the smooth RBF surface with a naive solution, where only a flat surface was used (Beethoven). This results in more challenging borders, which are more fragile and difficult to handle in practice.

Decompositions of the sphere, torus, kitten, airplane, and bunny examples were computed fully automatic. For the fertility and sculpture example, the automatically generated decomposition was similar with the result shown in Fig. 13, but minor user interactions with our system were required to obtain the visually pleasing results. For the Beethoven example, our system generates an initial decomposition with fewer parts, as shown in the accompanied video. However, our interface allows users to intuitively split the initial solution into additional parts for fabricating them with different color. In current implementation, detected thin features are automatically labeled as no-cut regions.

For example, the arms of the Fertility model are automatically labeled as no-cut regions, and there were no additional user defined no-cut regions required. For a demonstration of the interactive design system we refer to the accompanied video.



In addition, our system visualizes the deformation estimate in real time, allowing users to identify potentially problematic regions

Table 2. Result statistics. The timing was measured on an Intel Core i7 (2.2GHz) with 16GB RAM with single thread. Several interactions were not used for generating the examples (e.g., sphere did not need no-cut, cut, and merge interactions), but we measured the calculation times as a reference. For smoothing, we measured the time for appling smoothing to all the snakes vertices.

	# triangles ($\mathcal{S}_{\text{input}}$ / $\mathcal{S}_{\text{low}}$)	pre-comp. for coarse decomp. per direction	coarse decomp. per iteration	# no-cut	cut (# / timitng)	merge (# / timitng)	smoothing (timing)	# parts
Sphere	20480 / 320	3.84 ms	11.0 ms	0	0 / 5.60 ms	0 / 26.2 ms	90.8 ms	2
Torus	20000 / 1152	34.9 ms	82.0 ms	0	0 / 15.0 ms	0 / 71.4 ms	665 ms	2
Kitten	39206 / 2310	136 ms	396 ms	0	0 / 47.4 ms	0 / 209 ms	3964 ms	2
Airplane	119866 / 19136	13910.28 ms	31711 ms	0	0 / 692 ms	0 / 1872 ms	176574 ms	2
Fertility	51534 / 8788	2011.67 ms	4886.17 ms	0	1 / 208 ms	1 / 891 ms	51580 ms	2
Bunny	49700 / 2740	210 ms	2110 ms	0	0 / 53.2 ms	0 / 232 ms	3430 ms	2
Sculpture	72006 / 5888	875.25 ms	2550 ms	0	8 / 415 ms	8 / 820 ms	40633 ms	6
Beethoven	419424 / 2000	118.62 ms	1446 ms	1	5 / 210 ms	0 / 648 ms	18973 ms	7

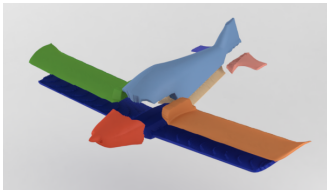
(colored red) and eventually adapting their design. We also fabricated most of our models, using either resin casting or injection molding.

Casting. The casting process begins by covering each part of the mold with a release agent to facilitate demolding of the cured pieces. Because of the generated runners and gates and because the mold is not perfectly airtight at parting lines, we did not encounter problems with trapped air in practice. For the result shown in Fig. 13, we use ultra-low viscose urethane casting resins with a high shore and a pot life of seven minutes for the Beethoven example and five minutes for the others. The manually placed pins serve the sole purpose to easily open the molds. The resulting piece may have some imperfections along the parting line, such as a thin *molding flash*, a typical artifact caused by the leakage of the material between the two touching surfaces of the mold. This excess material is removed using a rasp. Finally, the casted pieces are glued together along the seams to complete the target model.

Injection Molding. We also tested one of our molds on a professional industrial injection molding machine. Using our system, we decomposed the Stanford bunny into two shell pieces and generated a single two-piece mold. No adjustments to our workflow were required, except for defining the position of the sprue bush to prevent the shell pieces from colliding with it. Manual post-processing was limited to drilling holes for fixing the mold to the machine and for inserting the cylindrical metal sprue bush. We waived the addition of a mechanism for automatic ejection because of the low production volume. However, adding ejector pins would be straight forward and only require the placement of a few cylindrical holes. The injection molded samples are made of polypropylene with a material consumption of 26g for the actual model and 5g for the sprue. The material cost per sample was approximately 10 cents.

Comparison to [Herholz et al. 2015]. Herholz et al. [2015] presented a method for decomposing and deforming a surface mesh into height-field patches. The targeted fabrication technique, casting with multi-mold pieces, significantly differs from our two-piece molding. However, their method could be re-purposed for generating thin shell objects for two-piece molding. For highlighting the difference to [Herholz et al. 2015], we computed decompositions of several models with both methods. For computing the decompositions of thin shell objects based on [Herholz et al. 2015], we first segment the surface with [Herholz et al. 2015], and then extrude the patches inward to obtain thin shell parts. Finally, we compute a parting line for each part, and apply moldability enforcement to ensure moldability.

For the comparison shown in Fig. 12, we used the same value for the parameter γ that corresponds to the maximally allowed amount of deformation. Because [Herholz et al. 2015] does not allow overlaps even if they could be solved by the moldability enforcement (i.e., deformation) and furthermore does not allow two-sided parts (e.g., the arms of the fertility), their method results in a larger number of parts. Specifically, [Herholz et al. 2015] decomposes the kitten model into five parts, fertility into eight parts, and the bust sculpture into six parts. Our method can decompose all the models into two parts. Furthermore, we evaluate the difference from the original shape by computing the Hausdorff distance from original shape to the deformed shell parts. Fig. 12 also shows the ratio of the distance to the diagonal of the bounding box of the shape. In all comparisons, results generated with our method have less deformation than [Herholz et al. 2015].



Additionally, the decompositions of the airplane highlight the effect of our volume-awareness on the decomposition result. As shown in the inset figure, the method of Herholz et al. [2015] generates very thin and fragmented parts (e.g., the wings are splitted into thin flat parts) because their method is not aware of thin features. On the other hand, our

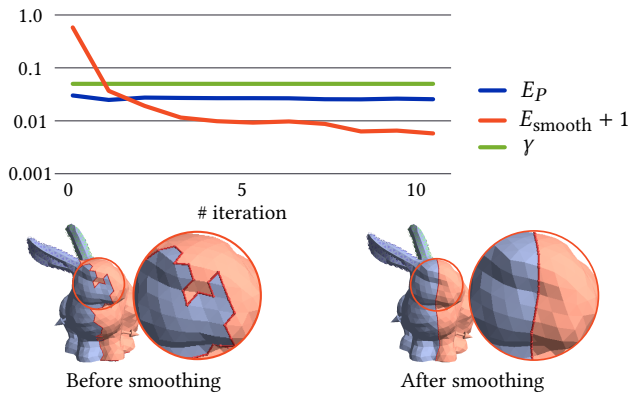


Fig. 10. Energy during boundary smoothing. The smoothness energy (red) sharply decreases in the first few iterations and then converges around an optimal solution. The moldability energy E_P is shown in blue.

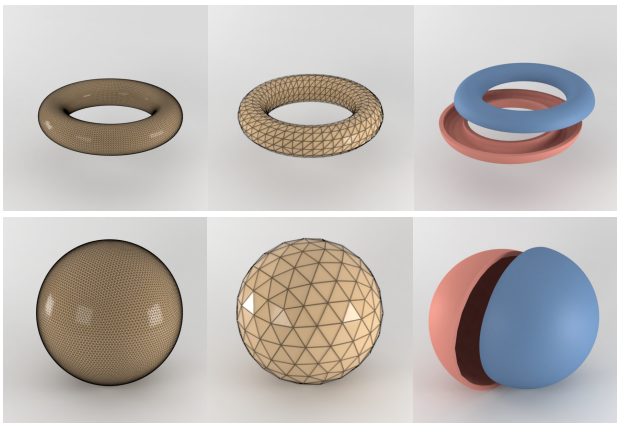


Fig. 11. Test with simple sphere and torus. (From left to right) High-res detailed input $\mathcal{S}_{\text{input}}$, low-res representation $\mathcal{S}_{\text{low}}$, final decomposition.

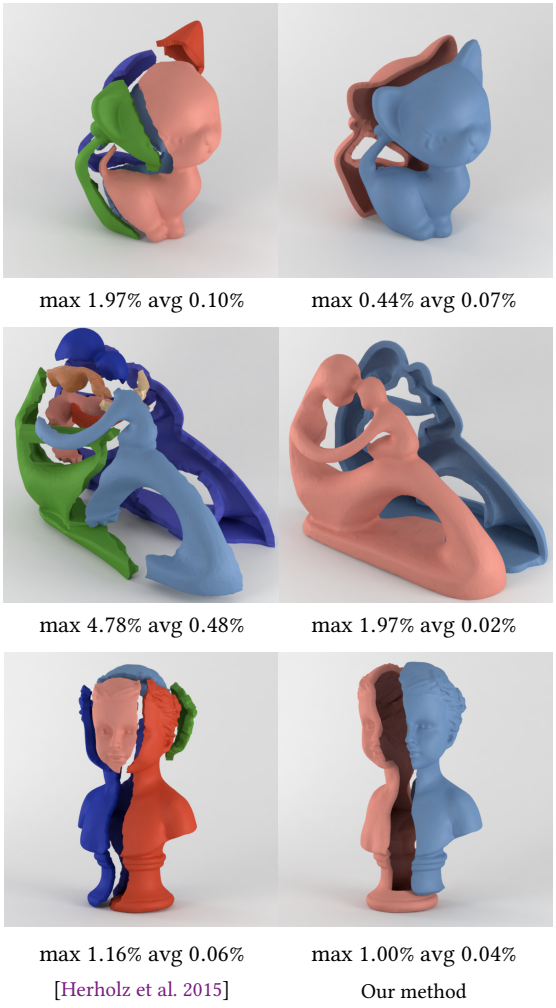


Fig. 12. Side-by-side comparison of [Herholz et al. 2015] and our method. From top to bottom: Kitten, fertility, and bust sculpture. The average and maximal Hausdorff distances to the input mesh are given as ratio with respect to the model's bounding box diagonal.

method keeps the wings intact, which improves both aesthetics and stability (Fig. 13).

8 LIMITATIONS AND FUTURE WORK

Although our method can deal with a wide range of shapes, it inevitably has some limitations. With regard to the position of cutting seams, our energy formulation for active contours only considers moldability and smoothness, and further aesthetic considerations are left to the users. Therefore, the contours might sometime move in a subjectively undesired direction. This issue could be limited by using perceptual models that quantify the visual quality of a decomposition [Lee et al. 2005; Secord et al. 2011; Zhang et al. 2015].

The current placement of each part on the mold is a simple rotation so that its parting direction points upward along +Z. An obvious extension would be an optimal packing, which we left for future work. Another interesting avenue for future work would be to extend our system to take the flow and the temperature of the material into account. Especially for geometrically challenging cavities and for optimizing economic factors, flow and heat dissipation are important performance indicators.

Currently, after fabrication, users still need to assemble the individual parts with glue. The addition of connectors to aid part alignment and model assembly could obsolete this time-consuming process. For future work, we plan to investigate and integrate connectors that are sufficiently general to guarantee undercut-free connector geometry even for connecting challenging shell pieces.

9 CONCLUSION

In this paper, we propose an interactive decomposition method for two-piece molding. The proposed method solves a computationally complex problem (i.e., moldability check) in interactive speed by reducing the complexity of the target shape (coarse-to-fine approach)

Fig. 13. (From left to right) High-res detailed input $\mathcal{S}_{\text{input}}$, low-res representation $\mathcal{S}_{\text{low}}$, final decomposition, corresponding automatically generated mold pieces, and fabrication results. (From top to bottom) We denote our examples as kitten, airplane, fertility, bunny, sculpture, and Beethoven. Each pair of mold pieces corresponding to the part with the same color. Bunny example is designed for injection molding, and the others are designed for manual resin casting. For the Beethoven model, we omit the computation of the smooth surface during the mold generation.

and relaxing the constraints for moldability. The method successfully decomposes a wide variety of shapes as shown in Fig. 11 and Fig. 13. To demonstrate the applicability of our method for hobbyist makers and in an industrial setting, we fabricated several physical copies of popular models in computer graphics with resin casting and injection molding.

ACKNOWLEDGMENTS

From Thingiverse, we obtained the models kitten (thing:12694), airplane (thing:689974), and Beethoven (thing:4108), while bunny was obtained from the Stanford 3D Scanning Repository and fertility from VISIONAIR. Some models were modified (e.g., filling holes). The sculpture model was generated with the *Sculpture Generator 1* by Carlo H. Séquin, UC Berkeley. Kazutaka Nakashima is supported by the Japan Science and Technology Agency ACT-I Grant No.: JPMJPR16UK (http://www.jst.go.jp/kisoken/act-i/en/project/111C001/111C001_20.html), and Graduate Program for Social ICT Global Creative Leaders (GCL) of The University of Tokyo by Japan society for the promotion of science. Thomas Auzinger and Bernd Bickel are supported by the European Research Council Starting Grants under Grant No.: 715767 (https://cordis.europa.eu/project/rcn/206323_en.html). Ran Zhang is supported by the Marie Skłodowska-Curie Innovative Training Networks under Grant No.: 642841 (https://cordis.europa.eu/project/rcn/193953_en.html). Takeo Igarashi is supported by the Japan Society for the Promotion of Science KAKENHI under Grant No.: 17H00752 (https://kaken.nii.ac.jp/en/grant/KAKENHI-PROJECT-17H00752/).

REFERENCES

Giuseppe Alemanno, Paolo Cignoni, Nico Pietroni, Federico Ponchio, and Roberto Scopigno. 2014. Interlocking Pieces for Printing Tangible Cultural Heritage Replicas. (2014). https://doi.org/10.2312/gch.20141312

Stephan Bischoff, Tobias Weyand, and Leif Kobbelt. 2005. Snakes on Triangle Meshes. In *Bildverarbeitung fÃijr die Medizin 2005*. Springer-Verlag, 208–212. https://doi.org/10.1007/3-540-26431-0_43

J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. 2001. Reconstruction and representation of 3D objects with radial basis functions. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques - SIGGRAPH '01*. ACM Press. https://doi.org/10.1145/383259.383266

Pritam Chakraborty and N. Venkata Reddy. 2009. Automatic determination of parting directions, parting lines and surfaces for two-piece permanent molds. *Journal of Materials Processing Technology* 209, 5 (mar 2009), 2464–2476. https://doi.org/10.1016/j.jmatprotec.2008.05.051

Xuelin Chen, Hao Zhang, Jinjie Lin, Ruizhen Hu, Lin Lu, Qixing Huang, Bedrich Benes, Daniel Cohen-Or, and Baoquan Chen. 2015. Dapper: decompose-and-pack for 3D printing. *ACM Transactions on Graphics* 34, 6 (oct 2015), 1–12. https://doi.org/10.1145/2816795.2818087

David Cohen-Steiner, Pierre Alliez, and Mathieu Desbrun. 2004. Variational shape approximation. *ACM Transactions on Graphics* 23, 3 (aug 2004), 905. https://doi.org/10.1145/1015706.1015817

Sándor P. Fekete and Joseph S. B. Mitchell. 2001. Terrain Decomposition and Layered Manufacturing. *International Journal of Computational Geometry & Applications* 11, 06 (dec 2001), 647–668. https://doi.org/10.1142/s0218195901000687

J.Y.H. Fuh, M. W. Fu, and A.Y.C. Nee. 2004. *Computer-Aided Injection Mold Design and Manufacture*. CRC Press. https://www.crcpress.com/Computer-Aided-Injection-Mold-Design-and-Manufacture/Fuh-Fu-Nee-Fu/p/book/9780824753146

Wei Gao, Yunbo Zhang, Diogo C. Nazzetta, Karthik Ramani, and Raymond J. Cipra. 2015. RevoMaker: Enabling multi-directional and functionally-embedded 3D printing using a rotational cuboidal platform. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology - UIST '15*. ACM Press. https://doi.org/10.1145/2807442.2807476

Philipp Herholz, Wojciech Matusik, and Marc Alexa. 2015. Approximating Free-form Geometry with Height Fields for Manufacturing. *Computer Graphics Forum* 34, 2 (may 2015), 239–251. https://doi.org/10.1111/cgf.12556

Ruizhen Hu, Honghua Li, Hao Zhang, and Daniel Cohen-Or. 2014. Approximate pyramidal shape decomposition. *ACM Transactions on Graphics* 33, 6 (nov 2014), 1–12. https://doi.org/10.1145/2661229.2661244

Masatomo Inui, Hidekazu Kamei, and Nobuyuki Umezu. 2014. Automatic detection of the optimal ejecting direction based on a discrete Gauss map. *Journal of Computational Design and Engineering* 1, 1 (jan 2014), 48–54. https://doi.org/10.7315/jcde.2014.005

Alec Jacobson, Ladislav Kavan, and Olga Sorkine-Hornung. 2013. Robust inside-outside segmentation using generalized winding numbers. *ACM Transactions on Graphics* 32, 4 (jul 2013), 1. https://doi.org/10.1145/2461912.2461916

Wenzel Jakob, Marco Tarini, Daniele Panozzo, and Olga Sorkine-Hornung. 2015. Instant field-aligned meshes. *ACM Transactions on Graphics* 34, 6 (oct 2015), 1–15. https://doi.org/10.1145/2816795.2818078

David O. Kazmer. 2016. *Injection Mold Design Engineering*. Carl Hanser Verlag. http://www.hanserpublications.com/Products/385-injection-mold-design-engineering-2e-ebook.aspx

Rahul Khardekar, Greg Burton, and Sara McMains. 2005. Finding feasible mold parting directions using graphics hardware. In *Proceedings of the 2005 ACM symposium on Solid and physical modeling - SPM '05*. ACM Press. https://doi.org/10.1145/1060244.1060271

Chang Ha Lee, Amitabh Varshney, and David W. Jacobs. 2005. Mesh saliency. *ACM Transactions on Graphics* 24, 3 (jul 2005), 659. https://doi.org/10.1145/1073204.1073244

Weishi Li, R.R. Martin, and F.C. Langbein. 2009. Molds for Meshes: Computing Smooth Parting Lines and Undercut Removal. *IEEE Transactions on Automation Science and Engineering* 6, 3 (jul 2009), 423–432. https://doi.org/10.1109/tase.2009.2021324

Alan C. Lin and Nguyen Huu Quang. 2014. Automatic generation of mold-piece regions and parting curves for complex CAD models in multi-piece mold design. *Computer-Aided Design* 57 (dec 2014), 15–28. https://doi.org/10.1016/j.cad.2014.06.014

Linjie Luo, Ilya Baran, Szymon Rusinkiewicz, and Wojciech Matusik. 2012. Chopper: partitioning models into 3D-printable parts. *ACM Transactions on Graphics* 31, 6 (nov 2012), 1. https://doi.org/10.1145/2366145.2366148

Luigi Malomo, Nico Pietroni, Bernd Bickel, and Paolo Cignoni. 2016. FlexMolds: automatic design of flexible shells for molding. *ACM Transactions on Graphics* 35, 6 (nov 2016), 1–12. https://doi.org/10.1145/2980179.2982397

A.Y.C. Nee, M.W. Fu, J.Y.H. Fuh, K.S. Lee, and Y.F. Zhang. 1997. Determination of Optimal Parting Directions in Plastic Injection Mold Design. *CIRP Annals* 46, 1 (1997), 429–432. https://doi.org/10.1016/s0007-8506(07)60858-0

Alok K. Priyadarshi and Satyandra K. Gupta. 2004. Geometric algorithms for automated design of multi-piece permanent molds. *Computer-Aided Design* 36, 3 (mar 2004), 241–260. https://doi.org/10.1016/s0010-4485(03)00107-6

B. Ravi and M.N. Srinivasan. 1990. Decision criteria for computer-aided parting surface design. *Computer-Aided Design* 22, 1 (jan 1990), 11–18. https://doi.org/10.1016/0010-4485(90)90024-7

Adrian Secord, Jingwan Lu, Adam Finkelstein, Manish Singh, and Andrew Nealen. 2011. Perceptual models of viewpoint preference. *ACM Transactions on Graphics* 30, 5 (oct 2011), 1–12. https://doi.org/10.1145/2019627.2019628

Lior Shapira, Ariel Shamir, and Daniel Cohen-Or. 2008. Consistent mesh partitioning and skeletonisation using the shape diameter function. *The Visual Computer* 24, 4 (jan 2008), 249–259. https://doi.org/10.1007/s00371-007-0197-5

Olga Sorkine and Marc Alexa. 2007. As-rigid-as-possible Surface Modeling. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing (SGP '07)*. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 109–116. http://dl.acm.org/citation.cfm?id=1281991.1282006

J. Vanek, J. A. Garcia Galicia, B. Benes, R. Měch, N. Carr, O. Stava, and G. S. Miller. 2014. PackMerger: A 3D Print Volume Optimizer. *Computer Graphics Forum* 33, 6 (may 2014), 322–332. https://doi.org/10.1111/cgf.12353

W. M. Wang, C. Zanni, and L. Kobbelt. 2016. Improved Surface Quality in 3D Printing by Optimizing the Printing Direction. *Computer Graphics Forum* 35, 2 (may 2016), 59–70. https://doi.org/10.1111/cgf.12811

Miaojun Yao, Zhili Chen, Linjie Luo, Rui Wang, and Huamin Wang. 2015. Level-set-based partitioning and packing optimization of a printable model. *ACM Transactions on Graphics* 34, 6 (oct 2015), 1–11. https://doi.org/10.1145/2816795.2818064

Chunjie Zhang, Xionghui Zhou, and Congxin Li. 2009. Feature extraction from freeform molded parts for moldability analysis. *The International Journal of Advanced Manufacturing Technology* 48, 1-4 (sep 2009), 273–282. https://doi.org/10.1007/s00170-009-2273-7

Xiaoting Zhang, Xinyi Le, Athina Panotopoulou, Emily Whiting, and Charlie C. L. Wang. 2015. Perceptual models of preference in 3D printing direction. *ACM Transactions on Graphics* 34, 6 (oct 2015), 1–12. https://doi.org/10.1145/2816795.2818121

Received January 2018; revised April 2018; final version May 2018; accepted May 2018