# Support-Free Hollowing

Wang, Weiming; Liu, Yong-Jin; Wu, Jun; Tian, Shengjing; Wang, Charlie; Liu, Ligang ; Liu, Xiuping

**DOI**
10.1109/TVCG.2017.2764462

**Publication date**
2018

**Document Version**
Accepted author manuscript

**Published in**
IEEE Transactions on Visualization and Computer Graphics

# Support-Free Hollowing

Weiming Wang, Yong-Jin Liu, *Senior Member, IEEE,* Jun Wu, Shengjing Tian, Charlie C.L. Wang*, *Senior Member, IEEE,* Ligang Liu, and Xiuping Liu

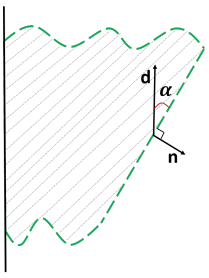
**Abstract**—Offsetting-based hollowing is a solid modeling operation widely used in 3D printing, which can change the model's physical properties and reduce the weight by generating voids inside a model. However, a hollowing operation can lead to additional supporting structures for fabrication in interior voids, which cannot be removed. As a consequence, the result of a hollowing operation is affected by these additional supporting structures when applying the operation to optimize physical properties of different models. This paper proposes a support-free hollowing framework to overcome the difficulty of fabricating voids inside a solid. The challenge of computing a support-free hollowing is decomposed into a sequence of shape optimization steps, which are repeatedly applied to interior mesh surfaces. The optimization of physical properties in different applications can be easily integrated into our framework. Comparing to prior approaches that can generate support-free inner structures, our hollowing operation can reduce more volume of material and thus provide a larger solution space for physical optimization. Experimental tests are taken on a number of 3D models to demonstrate the effectiveness of this framework.

**Index Terms**—shape optimization, support-free, hollowing, topology variation, 3D printing.

## 1 Introduction

The popularity of 3D printers has reduced the barrier to fabricate complex models. This fact motivates a lot of research in the computer graphics community. To achieve specific functions of fabricated models, a widely used strategy is to optimize the material distribution in the interior of the given 3D model, leading to inner voids in different forms. These carving operations extend the conventional offsetting-based hollowing operation in solid modeling. The hollowing method is found to be effective for a variety of physical objectives regarding mass properties (e.g., [1], [2], [3]) and the mechanical strength (e.g., [4], [5], [6]). However, a largely neglected aspect is that the interior voids give rise to a critical problem for fabrication – that is, additional support structures in such cavities are inevitably needed for some layer-based 3D printing processes such as SLA (*Stereolithography Apparatus*) and FDM (*Fused Deposition Modeling*).

Support structures are usually added below overhang regions to prevent the collapse of material during the fabrication (ref. [7], [8], [9]), which lead to additional material usage and give rise to problems such as difficulty of removal, surface damage and a prolonged printing time. Supports located in enclosed voids cannot be accessed and thus are difficult to take out. This is unlike exterior supports which can be removed manually or automatically if they are made of dissolvable material. Keeping interior supports in a fabricated model counteracts the objectives from the results of interior shape optimization. For instance, the additional supports shift the center of mass from the desired location and decrease the strength-to-weight ratio. To eliminate such supports, the optimized shape is typically printed in parts and glued together [10]. This treatment however leaves visual and mechanical defects.




To overcome the problem of adding interior support structures, our idea is to make all facets on the interior surface self-supported – i.e., the angle between its normal vector $\mathbf{n}$ and the printing direction $\mathbf{d}$, denoted by $\angle(\mathbf{n}, \mathbf{d})$, is less than $\alpha + 90°$, with $\alpha$ being called the *maximal self-supporting angle* (see the right inset). Here $\alpha$ is a physical coefficient related to the type of 3D printing processes and materials used in fabrication. For example, an angle of $45°$ is commonly used for FDM, while an angle of less than $30°$ is safe for SLA. When every facet on an interior void is support-free and no edge/vertex overhang is found (see [8] for a detail discussion for different types of overhangs), its surface is defined as self-supported.

In this work we propose a novel hollowing framework to generate support-free interior voids. Following the layer-upon-layer fabrication process, we decompose the 3D hollowing problem into a set of planar non-uniform offsetting problems. This boils down the challenge of support-free hollowing into computing non-uniform offsetting values in all layers, where support-free is formulated by geometric constraints on offsetting values in neighboring layers. The support-free hollowing operator can be used as an integral component for different applications. For instance, we demonstrate its effectiveness in interior shape optimization for reducing the material usage (see Fig. 1 for an example) and for optimizing the center of mass (e.g., Fig. 13), both resulting in interior voids that require no additional support

- *W. Wang, S.Tian and X. Liu are with School of Mathematical Sciences, Dalian University of Technology, Dalian, China.*
- *Y.-J. Liu is with the TNList, Department of Computer Science & Technology, Tsinghua University, Beijing, China.*
- *J. Wu and C.C.L. Wang are with Department of Design Engineering, Delft University of Technology, Delft, The Netherlands.*
- *L. Liu is with School of Mathematical Sciences, University of Science and Technology of China, Hefei, China.*
- *Corresponding Author: C.C.L. Wang (Email: c.c.wang@tudelft.nl).*

*Final manuscript submitted on October 16, 2017.*

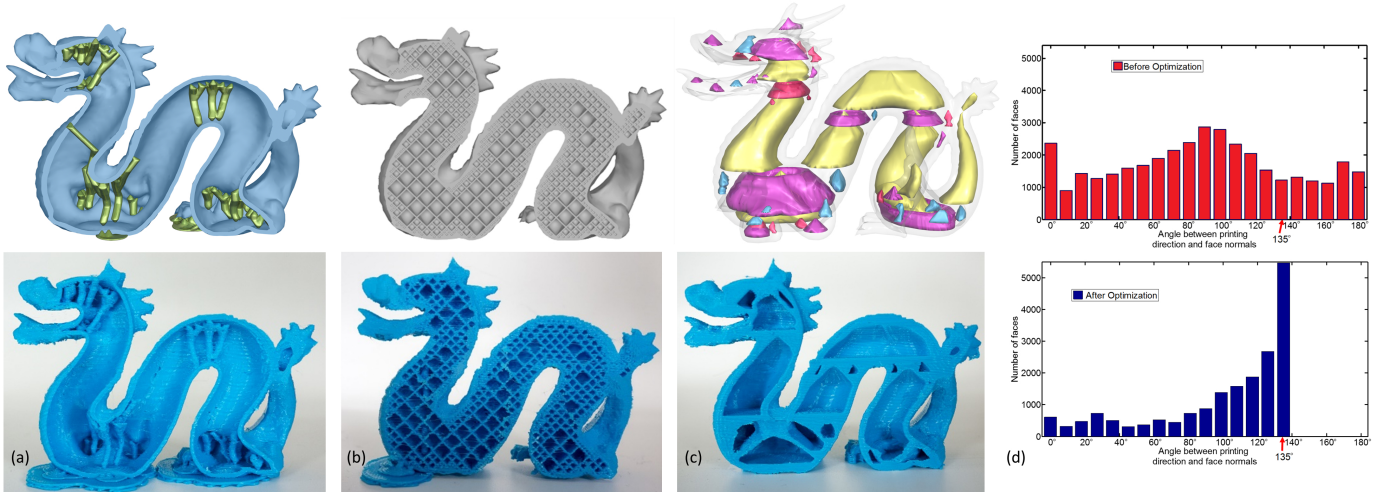



Fig. 1. Different from (a) conventional hollowing that needs a large number of additional supporting structures and (b) the rhombic infill structure [11] that can only reduce $38.0\%$ material usages, the hollowing operation developed in our framework can be repeatedly applied to a model for generating highly sparse interior structures and optimizing different physical properties – the result shown in (c) with $69.9\%$ less material used than the original solid model. Our hollowing framework generates interior voids with self-supported surfaces – no supporting structure is needed for 3D printing. (d) The histograms of the angles between surface normals of interior voids and the printing direction comparing (a) and (c).

structures. Our technical contributions include:

- A support-free hollowing operator with layer-based formulation which generates offset surfaces that are free of additional inner supports.
- A strategy of repeatedly applying the operation to generate inner voids with topology different from an input model, which enlarges the solution space of physical optimization.
- A demonstration of the hollowing framework for designing application-specific and support-free interior material layout.

The rest of this paper is structured as follows. After reviewing related work in Section 2, we illustrate the pipeline of the proposed support-free hollowing in Section 3. The optimization problem is presented in Section 4 with its constraints for support-free detailed in Section 5, solution space analyzed in Section 6, and the repetitive strategy for enlarging the solution space explained in Section 7. Extensive numerical and physical tests are presented in Section 8, and the paper is concluded in Section 9.

## 2 Related Work

Geometric modeling and optimization for 3D printing has received a lot of attention in recent years. The approaches closely related to our work are reviewed below in the categories of self-supporting structures design, offset surface generation, and shape optimization.

**Self-supporting structures** An intrinsic way for solving the problem of interior support is to produce self-supporting interior surface during the hollowing process. The term self-supporting has been used in different contexts. In architectural geometry it refers to a structure (i.e., an arrangement of blocks such as bricks, stones) which is in a static equilibrium configuration [12], [13], [14]. In this paper, self-supporting is a geometric property concerning overhang angles. Recently, Reiner and Lefebvre [15] proposed an interactive modeling tool to design self-supporting models. Wu et al. [11] introduced rhombic structures as a special self-supporting infill for 3D printing. The rhombic structure is adaptively refined according to an analysis of physical properties such as the center of mass and the stress distribution. The hollowing operator presented in this paper generates even sparser structures. For example, adaptive rhombic structures can only reduce 38.0% of the material volume for the dragon model, but our method can reduce up to 69.9% volume when using the same thickness of walls (see Fig. 1). This actually provides a larger solution space for optimizing different application-specific physical properties. We also note that Xie and Chen [16] recently proposed a method to generate support-free interior cavities. However, their method is voxel-based and then results in staircase-like inner surfaces.

**Offset surface generation** The conventional way of hollowing for reducing the material usage in 3D printing is based on the offsetting operator in solid modeling. Most of recent work focuses on how to efficiently compute the offsetting, e.g., using distance-field [17], [18], ray-rep [19] and parallel computing [20]. When fabricating the hollowed voids generated by uniform offsetting, supporting structures cannot be avoided. Non-uniform offsetting with varying thickness has been recently employed to design the physical properties of printed models in different aspects. For example, the static and dynamic stability is improved in [3]. Non-uniform offsetting is also used to control the elastic deformation in [21] and [22]. However, the problem of avoiding interior supporting structures has not been tackled in any of these prior approaches.

**Shape optimization** A lot of effort has been made to reduce the usage of materials in 3D printing by shape optimization. Wang et al. [4] proposed a skin-frame structure and designed an optimization method to minimize the frame volume subject to various constraints such as stiffness, stability and printability. Lu et al. [5] proposed a

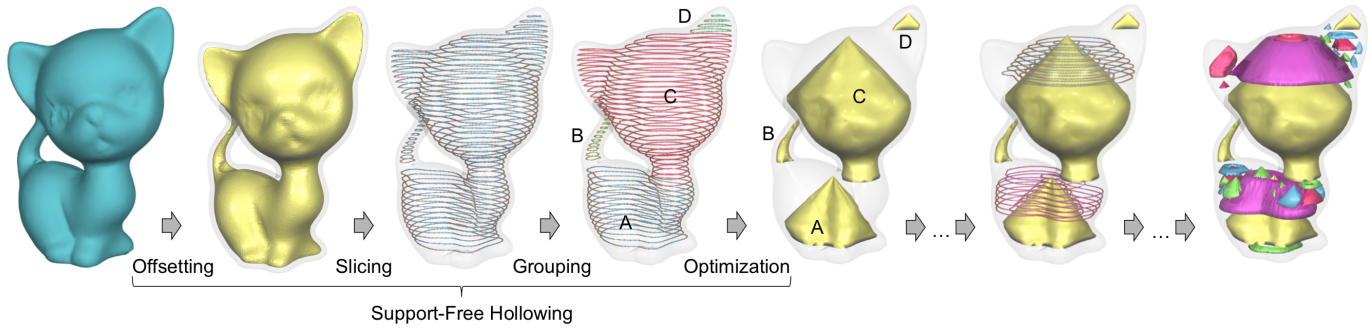



Fig. 2. Pipeline for computing the support-free hollowing on a solid model. From left to right, an input model, the surface after uniform offsetting, the sliced contours for hollowing, the grouping result by topology analysis for voids generation (slices in different groups, A-D, are displayed in different colors), and the optimized interior voids that are support-free. The support-free hollowing can be repeatedly applied until no more void can be formed – see the rightmost figure as the result with 39 voids.

honeycomb-cells structure and developed an optimal tessellation of this structure to maximize the porosity meanwhile sustaining the interior strength of a printed model. Wu et al. [6] presented a high-performance system for performing structural optimization using an efficient finite element analysis, building on the topology optimization method from mechanics [23]. Interior cavities have been introduced to improve both the static and the dynamic (i.e., spinning) stability for 3D printed models. This concept has been used in [1] to improve the static stability of a printed model, by using a voxel representation. Extensions along this line include alternative geometric discretizations for efficient computation (e.g., the adaptive octree [2], truss structures [24], ray-reps [25]), as well as various static/dynamic behaviors (e.g., spinning [2], floating [26], swinging [27]). In these approaches, support structures for a carved interior void cannot be avoided. Christiansen et al. added uniform infill patterns to support the interior void [28]. In this paper, we propose a new framework of support-free hollowing, which can be used as an operation to tackle different optimization applications of 3D printed models. On the other aspect, we also use the static stability problem as an example to demonstrate the flexibility of our framework.

## 3 Pipeline

The pipeline of the proposed support-free hollowing framework is illustrated in Fig. 2 by using the kitten model as an example. It consists of four essential steps:

1) Given a solid model represented by a 2-manifold mesh surface $\mathcal{M}$, we first uniformly offset it with a user specified thickness $t$ to get an inner surface $\tilde{\mathcal{M}}$ by uniform offsetting [29].
2) Slicing the newly generated inner surface $\tilde{\mathcal{M}}$ into a set of cross-sections, by using either a uniform thickness or a curvature-dependent adaptive thickness [30]. Note that, depending on the topology of the 3D model, it is possible that on the same layer there are multiple and disjoint contours.
3) According to the topology changes in successive layers, the cross-sections are clustered into groups, each of which represents a void, indicated separately by 'A'-'D' in Fig. 2. Neighboring cross-sections in the same group are connected by a strip triangulation algorithm [31].
4) The shape of each void obtained from the previous step is then optimized according to different applications, together with layer based support-free constraints. As a result, self-supported interior surfaces can be generated by using numerical optimization for all voids.

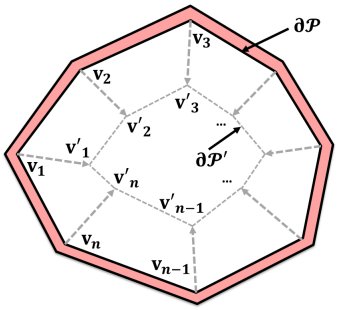
These four steps are repeatedly applied to a model to further optimize the application-specific objective function. The effect of repetition is shown as the right of Fig. 2. Different physical properties (e.g., total weight, static stability, etc.) can be incorporated into this framework as objectives of optimization.

## 4 Formulation

Our framework aims at minimizing different application-specific objective functions, while ensuring that the interior voids can be fabricated without additional inner support structures. The support-free constraint is formulated on a layer-based representation of the 3D model. The decomposition of a 3D model into a set of layers, which has the same process as layered fabrication, allows us to effectively evaluate the overhang angle, and thus to formulate the constraints in a compact form.

### 4.1 Notations and Variables

Following the common practice of 3D printing, we assume that the given model $\mathcal{M}$ with hollowed inner surfaces (denoted by $\mathcal{M}^H$) is fabricated layer-by-layer along a printing direction $\mathbf{d}$ with the thickness $h$ for each layer. Both $\mathcal{M}$ and $\mathcal{M}^H$ are sliced into a set of cross-sections perpendicular to $\mathbf{d}$. With the help of topology analysis taken on the sliced contours (details can be found in Section 7.1), contours are classified into different groups. As the outer surface $\mathcal{M}$ is unchanged during the optimization, we only consider the contours generated from the inner surfaces $\mathcal{M}^H$ in the rest of this paper. Contours in the same group will have simple topology as circles. The group of contours in different slices is denoted by $\Theta = \{\mathcal{P}_i\}$. Here $\mathcal{P}_i$ represents a planar domain at the $i$-th slice on $\mathcal{M}^H$. The optimization is performed on each group of contour. Different groups can thus be optimized in parallel. For simplicity, we restrict our discussion to one group in the following. Extending it to include all groups is straightforward.




For a planar contour $\mathcal{P} \in \Theta$, we assume its boundary $\partial\mathcal{P}$ consists of $n$ vertices $\{\mathbf{v}_j\}_{j=1}^n$, linked by ordered line segments (i.e., clockwise as it describes a hole). In our framework, the resulting optimized contour $\mathcal{P}'$ locates within the contour $\mathcal{P}$ since the contour shrinks inwards during the optimization process. The shrunk contour $\mathcal{P}'$ is obtained by shifting each vertex in $\partial\mathcal{P}$ along an inward direction $\mathbf{r}_j$, with a distance value $l_j$ to be determined by the optimization algorithm, that is

$$\mathbf{v}'_j = \mathbf{v}_j + l_j \mathbf{r}_j, \qquad j = 1, 2, \cdots, n. \tag{1}$$

The offsetting distance $l_j$ represents the design variable in our optimization. They are iteratively updated during the optimization processing. Performing offsetting on our layer-based formulation significantly simplifies the problem of avoiding intersection, which is notorious while offsetting a general triangle mesh (ref. [32], [33]). In Section 6 we will discuss the techniques to ensure that the resulting mesh is free of self-intersection and is manufacturable. Specifically, these desired properties are ensured by imposing lower/upper bounds for each $l_j$, and by properly determining shifting directions of vertices.

### 4.2 Optimization Problem

Generally, by using the notation and variables introduced above, objective functions can be defined according to different applications as

$$\min_{\{l_j\}} F(\{\mathcal{P}'_i\}, \{\mathcal{P}_i\}, h) \tag{2}$$

for all shifting distances $\{l_j\}$ on all contours $\{\mathcal{P}_i\}$ in the same group $\Theta$, where $h$ is the thickness of a layer.

The optimal offsetting values are subject to a set of constraints regarding manufacturability and application-specific functions. In summary, the constraints can be classified into two categories:

1) **Support-free:** The support-free constraints ensure the overhangs on the generated surfaces of interior voids are self-supported (see Section 5).
2) **Geometric correctness:** The geometric correctness means that the resulting mesh shall be free of self-intersection, and shall maintain a minimum thickness for manufacturability (see Section 6).

The optimization problem is solved by the classic interior-point algorithm [34] which shows a fast convergence rate. More discussions can be found in Section 8.4.

## 5 Constraints for Support-Free

The support-free constraints are formulated as the relationship between the contours on neighboring layers in the same group, $\mathcal{P}_i$ and $\mathcal{P}_{i+1}$, representing the holes on cross-sections. The analysis for adding supports is taken by the projection $\Pi(\cdot)$ of $\mathcal{P}_{i+1}$ onto $\mathcal{P}_i$. Based on the working principles used in the industry of additive manufacturing [35], supports are added in the regions where $\mathcal{P}_i$ is outside $\Pi(\mathcal{P}_{i+1})$. The region enclosed by $\mathcal{P}_i$ and outside $\Pi(\mathcal{P}_{i+1})$ is denoted by $\Omega$ (see also the illustration shown in Fig. 3). When considering the self-supporting property of materials, this condition is converted to only add supports at the places with large feature size in $\Omega$ when $\Omega \neq \emptyset$ (ref. [9]). This support-free condition can be converted into constraints imposed on the slope of surface connecting $\mathcal{P}_{i+1}$ and $\mathcal{P}_i$, which can be further broken down into the following constraints for faces, edges and ceilings.

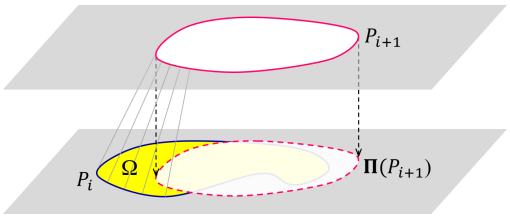



Fig. 3. An illustration for the projection based condition for supporting structures – when the region of $\Omega$ (the yellow region) is not empty, supporting structure needs to be added on the surface connecting $\mathcal{P}_i$ and $\mathcal{P}_{i+1}$. Note that, the projection is taken on the contours of an inner void.

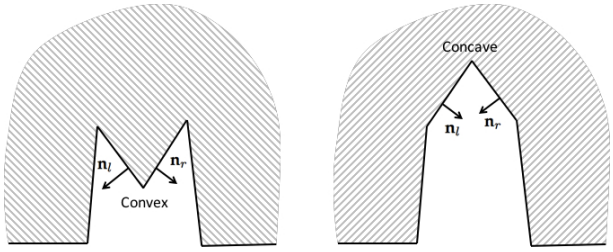



Fig. 4. Self-supported property is not automatically preserved on a convex edge of the inner mesh surface (see left) even when its left face and its right face are self-supported, but it is preserved on a concave edge (see right).

### 5.1 Face Constraint

The inner surface is obtained by a strip triangulation connecting neighboring polygonal contours. For all triangle faces, according to the definition of overhang angle, the following constraint applies,

$$(\mathbf{n}_{f_j} \cdot \mathbf{d}) \geq \cos(\frac{\pi}{2} + \alpha), \tag{3}$$

where $\mathbf{n}_{f_j}$ is the face normal of the $j$-th face, $f_j$, of the inner void mesh, $\mathbf{d}$ is the printing direction, and $\alpha$ is the maximal self-supporting angle.

### 5.2 Edge Constraint

Controlling the overhang angles of faces is not sufficient for the support-free of the whole inner surface since it cannot prevent the generation of edge overhangs. See Fig. 4 (left) for a 2D illustration. While the normal vectors of an edge's left and right faces, $\mathbf{n}_l$ and $\mathbf{n}_r$, satisfy the angle constraint in Eq.(3), the edge itself is not supported from beneath. In fact, overhang is prevented on a concave edge on the mesh surface of an inner void automatically by the face constraint,

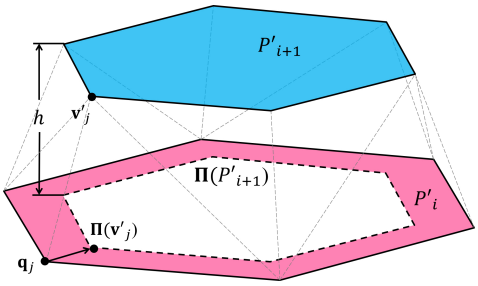



Fig. 5. An illustration for support-free constraint between two neighboring layers, where the dash lines illustrate the mesh generated by strip triangulation [31], [36].

but can still exist on a convex edge. Distinguishing these two cases in the numerical optimization framework leads to a formulation with a variant number of constraints as a convex edge can be deformed into a concave edge during the shape optimization, and vice versa. Therefore, to make a simpler formulation, the support-free constraint is applied to all edges connecting to different slices. We exclude edges whose endpoints are located in the same slice, since in this case the edge will always have one neighbor-face above and the other neighbor-face below the slice, which means that it does not form a convex shape as shown in the left of Fig. 4. Note that, when such an convex edge is shown on the shape resulted from uniform offsetting before optimization, it will be eliminated by the topology analysis of contour grouping. Specifically, the left and the right cavities will be clustered into different groups – e.g., see the groups A, B and C in Fig. 2.

Mathematically, to prevent forming edge overhang during the optimization, we formulate the support-free constraint on an edge across two neighboring slices, $\mathcal{P}'_i$ and $\mathcal{P}'_{i+1}$ as the relative displacement of vertices. The relevant notations are shown in Fig. 5. Specifically, the vertices of $\mathcal{P}'_{i+1}$ and $\mathcal{P}'_i$ should satisfy

$$\|(\Pi(\mathbf{v}'_j) - \mathbf{q}_j)\| < h \cdot \tan(\alpha), \tag{4}$$

where $\mathbf{v}'_j$ is a vertex of $\mathcal{P}'_{i+1}$, $\mathbf{q}_j$ on $\partial\mathcal{P}'_i$ is the endpoint of an edge that connects with $\mathbf{v}'_j$, and $h$ is the layer thickness. However, adding this constraint to the facing-up edges will lift up them and the adjacent faces to form an upward surface, which reduces the volume of an inner void and therefore also the solution space. To solve this problem, a sign function is added in the formulation below to avoid over-constraining those facing-up edges. And the constraint is re-formulated into a quadratic form to achieve a faster convergence in optimization.

$$-sgn(\mathbf{n}_e \cdot \mathbf{d})\|(\Pi(\mathbf{v}'_j) - \mathbf{q}_j)\|^2 < h^2 \tan^2(\alpha) \tag{5}$$

Here $\mathbf{n}_e$ is the normal vector of the edge $\mathbf{v}'_j\mathbf{q}_j$, which is evaluated by the average of the edge's left and right face-normals. This classifier of sign function follows the method of Deuss et al. [14] to detect edge overhangs. As a result, the edge constraint is only applied to those edges with facing-down normals.

### 5.3 Ceiling Constraint

While the face constraint Eq.(3) modifies inclined faces to a satisfactory slope, the uppermost and horizontal faces at the ceiling have their normals constantly pointing downwards. To satisfy self-supporting constraint for such faces, our idea is to shrink these faces such that each of them degenerate into either a point or an edge, supported by the faces below it. This is realized by restricting the area of the uppermost slice to be infinitesimal as

$$A(P_{uppermost}) < \varepsilon, \tag{6}$$

where $\varepsilon$ is a parameter to control the area of the uppermost slice in a group (i.e., $\varepsilon = 10^{-5}$ is used in our implementation and all examples shown in this paper).

## 6 Geometric Constraints and Bounds

In our layer-based formulation, the optimization problem is solved in an interlaced manner. The shifting directions of vertices in each layer are resolved inside the layer, and the values of displacements in shifting are determined globally by solving the minimization problem in Eq.(2). For every given contour $\mathcal{P}$, the shifting direction $\mathbf{r}_i$ for each vertex $\mathbf{v}_i$ is expected to:

1) let $\mathcal{P}'$ be homeomorphic to $\mathcal{P}$ and also have similar shapes,
2) avoid self-intersection on $\mathcal{P}'$, and
3) provide relative large space for shifting.

These expectations are in fact coupled with each other. In general a larger displacement $l_i$ for every $\mathbf{v}_i$ is more likely to cause self-intersections. Furthermore, small displacements on all vertices of $\mathcal{P}'$ lead to a shape similar to $\mathcal{P}$.

The range of variables (i.e., the displacements of shifting on all vertices) must be controlled for the following reasons:

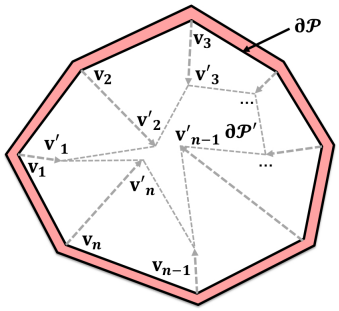
- **intersection-free** – if the bounds of shifting are not specified, the newly generated polygon $\mathcal{P}'$ on each layer can easily intersect with other polygons or be self-intersected;
- **shape control** – when the neighboring vertices on $\mathcal{P}$ have significantly different displacements, sharp corners which result in mechanical weakness can be formed on $\mathcal{P}'$;
- **manufacturability** – printing materials will fail to accumulate at the regions with thin-walls.

The right inset illustrates the case that leads to sharp corners when the factor of shape control is not considered. To fulfill these expectations, we set lower and upper bounds of displacements for points on $\mathcal{P}'$,

$$l_j^L \le l_j \le l_j^U. \tag{7}$$

and use Voronoi Diagram to generate shifting directions.

The lower bound $l_j^L$ prevents generating thin shells. The manufacturability regarding a minimum thickness can naturally be satisfied by using the shape of uniform offsetting as the lower bound of shifting. Specifically, as the initial position of all



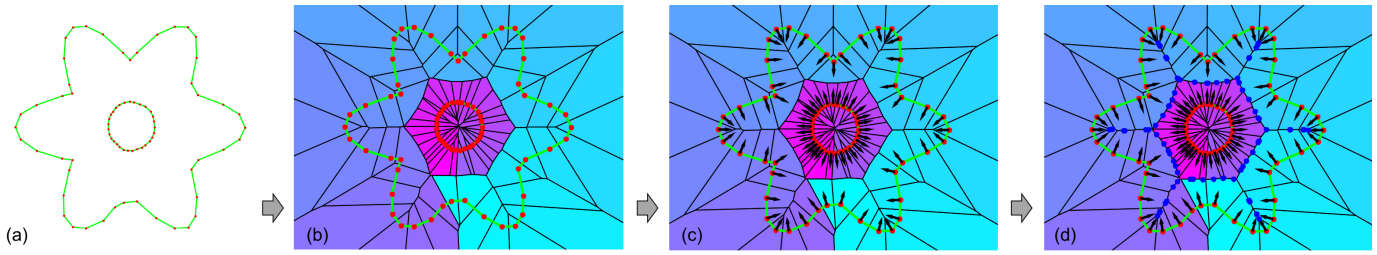



Fig. 6. Computing the shifting directions and the upper bounds of shifting points on a complex contour: (a) initial contour $\mathcal{P}'$ can be uniformly sampled into points, (b) Voronoi diagram is constructed for the sample points, (c) shifting directions (the black arrows) are determined with the help of Voronoi diagram's poles, and (d) intersections points (the blue points) at the boundaries of Voronoi cells give the upper bounds of optimization.

vertices on $\mathcal{P}$ are computed from the surfaces of inner voids generated by uniform offsetting, we assign the lower bounds of all points on $\mathcal{P}'$ as *zero*, that is

$$l_j^L = 0.$$

The upper bound $l_j^U$ is introduced to avoid self-intersection. For a cross-section with inner holes, we first uniformly sample it into a set of points (see Fig. 6(a)). A Voronoi diagram can be generated for these points (see Fig. 6(b)). Poles of a Voronoi diagram's cells were used in [37] to determine the normal vectors of a surface represented by dense 3D sample points. Here the pole of each Voronoi cell $\mathcal{C}(\mathbf{v}_j)$ ($\exists \mathbf{v}_j \in \mathcal{P}'$), $\mathbf{p}(\mathcal{C}(\mathbf{v}_j))$, as the normal of a planar curve at $\mathbf{v}_j$ is computed to determine the shifting direction of $\mathbf{v}_j$ as

$$\mathbf{r}_j = \pm \frac{\mathbf{v}_j - \mathbf{p}(\mathcal{C}(\mathbf{v}_j))}{\|\mathbf{v}_j - \mathbf{p}(\mathcal{C}(\mathbf{v}_j))\|} \tag{8}$$

with the sign determined by letting $\mathbf{r}_j$ be consistent with the orientations of an initially determined inner void. Suppose the ray along $\mathbf{r}_j$ with origin at $\mathbf{v}_j$ intersects the boundary of $\mathcal{C}(\mathbf{v}_j)$ at a point $\mathbf{q}(\mathcal{C}(\mathbf{v}_j))$ (see the blue points in Fig. 6(d)), the upper bound of optimization applied to $\mathbf{v}_j$ is assigned as

$$l_j^U = \|\mathbf{v}_j - \mathbf{q}(\mathcal{C}(\mathbf{v}_j))\| \tag{9}$$

to ensure an intersection-free hollowing. To have a better shape control, a truncated Laplacian smoothing as

$$l_j^U = \min\{l_j^U, \frac{1}{2}(l_{j-1}^U + l_{j+1}^U)\} \tag{10}$$

can be applied to the upper bounds on all vertices, where $-1$ and $+1$ denotes the predecessor and the successor of the $j$-th vertex on the same loop of a contour.

It is worth noting that, at the place of complex contours with many geometric details, the shifting directions and the upper bounds calculated by the above strategy may lead to a small solution space. Although it rarely happens, the support-free constraints (Eqs.(3) and (5)) for these contours are difficult to satisfy. To tackle these cases, we can slightly smooth their 2D contour from $\mathcal{P}$ to $\tilde{\mathcal{P}}$ and then compute an optimized contour from $\tilde{\mathcal{P}}$ instead of $\mathcal{P}$. Although this may make the shape of $\tilde{\mathcal{P}}$ different from $\mathcal{P}$, it helps the convergence of the numerical optimization. Moreover, generating voids with shapes significantly different from an input model does not affect the visual quality of the hollowed object.

## 7 Topology Analysis

In this section we present grouping of the contours based on an analysis of the topology of slices, and propose a repeated strategy to enlarge the solution space based on an analysis of the topology of solution space from offsetting.

### 7.1 Grouping of Contours

Models with complex topology (e.g., with branches or loop handles) lead to slices with their contours having varying topology. The contours are clustered into separate groups, each of which form an interior void with simple topology – i.e., *genus zero*). The following rules are applied for grouping:

- Different contours in the same slice cannot belong to the same group;
- When being projected along the printing direction, two contours in neighboring slices cannot belong to the same group if the overlapping area is less than a certain percentage of any of these two contours' area – we choose $50\%$ in all our examples;
- When a contour overlaps with multiple projected contours in a neighboring slice, it can only be assigned to the group of the contour with maximal overlap.

By using these rules, contours generated on the uniformly offsetted inner surface are clustered into different groups. This is implemented by iteratively using a flooding algorithm to group contours on neighboring slices (see Fig. 7 for an example). As a result, interior surfaces with simple topology can be formed. Note that an overhang with large area is very likely to happen when the area of a contour is much larger than the contour above it. The second rule above is applied to prevent this case. Without this rule requiring small area-variation, the result of optimization tends to generate a void with volume much smaller than the total volume of voids formed by separating the contours into different groups – see the contour between C and D in Fig. 8 for an example.

### 7.2 Topology of Solution Space

The interior void from offsetting generally has the same topology as the initial model. Fig. 9 shows the optimized voids for four basic shapes of genus zero. While the shape of the interior void varies from and depends on the exterior surface, it shares the same topology as the initial input model. Due to the overhang constraint, non-uniform offsetting creates large solid parts on the top of the models,

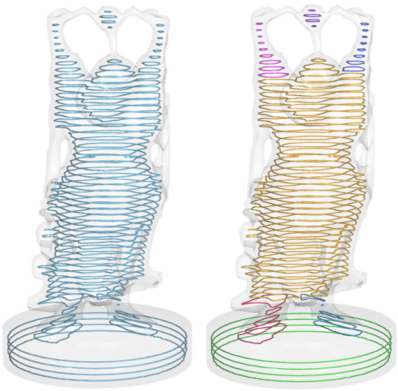


Fig. 7. Contours on the Buddha model (left) are clustered into seven different groups, which are displayed in different colors (right). Contours in each group are connected by strip triangulation to generate an interior surface with simple topology (i.e., genus zero).

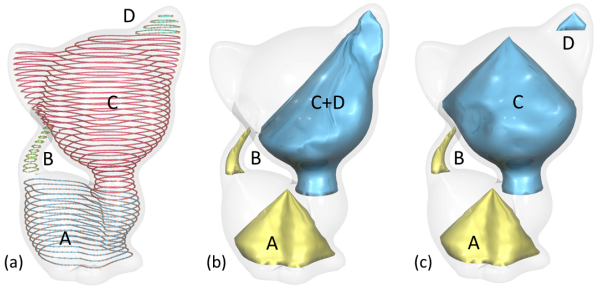



Fig. 8. Without applying the rule of area-variation between neighboring contours, the group of contours in C and D (see (a)) will be classified into the same group. This will result in a void (see (b)) with volume $5.14 \times 10^3\ mm^3$. When they are separated into different groups, two voids with the total volume of $6.27 \times 10^3\ mm^3$ can be generated as shown in (c).

i.e., the space between the transparent exterior surface and golden interior surface.

These large solid parts can be reduced by repeatedly applying the hollowing operation, taking the output model from the last optimization as the input for the next optimization. By doing so, more voids (with a smaller volume) can be generated inside the solid. Porous structures with complex topology and small total volume can be formed. Fig. 10 shows a sequence of results from this repeated hollowing. The final model on the rightmost has a much more complex topology than the initial model. In practice, we repeatedly apply the hollowing operation until 1) the volumes of all newly formed voids are less than $8mm^3$ or 2) the number of slices is less than two in all newly formed voids.

## 8 Results and Discussion

After presenting implementation details at the beginning of this section, we demonstrate support-free hollowing in two design applications – lightweight optimization and static stability optimization. The effectiveness of our approach is verified on a number of 3D model with varying complexity.

### 8.1 Implementation Details

**Configuration** Our algorithm has been implemented in C++, while using a nonlinear optimization library of Matlab

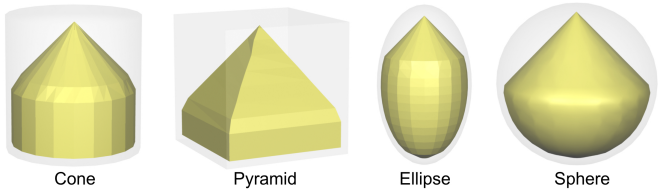



Fig. 9. The possible basic shapes of a support-free void with genus zero topology.

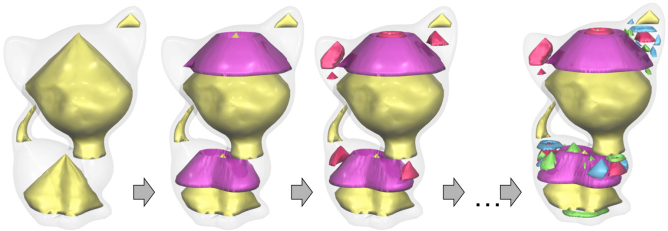


Fig. 10. The results obtained by repeatedly applying the support-free hollowing operation developed in this paper – from left to right, $55.3\%$, $68.6\%$, $72.1\%$ and $75.2\%$ of the Kitten model's volume have been converted into voids.

(i.e., the Interior Point Algorithm [34]). All the examples were computed on a standard desktop PC equipped with an Intel(R) Core(TM) i7-3770 CPU running at 3.40GHz with 32GB memory. The physical models were fabricated by a MakerBot 3D printer with a working envelope of $285\,mm \times 153\,mm \times 155\,mm$. The printing thickness is $0.2mm$, and PLA plastic is the used as the material of printing.

**Parameters** To ensure the manufacturability of a hollowed model and also provide a relative strong mechanical stiffness, the wall thickness of the initial offsetting is set as $1.0\,mm$ and the thickness in the subsequent optimization is set to $0.8\,mm$. The height of all models are scaled to $50\,mm \sim 60\,mm$. A maximal self-supporting angle of $45°$ is used for most models, while other values are also studied for analyzing the effect of the overhang angle.

**Strip triangulation** A strip triangulation algorithm [31] is adopted to construct the inner surface of voids, by connecting contours in neighboring slices. Note that, although the vertices on the inner offset surface locate inside the input model $\mathcal{M}$, it is possible that edges or faces linking these vertices intersect with $\mathcal{M}$. We perform an efficient intersection detection using the OBB-tree based proximity query [38]. For those triangles intersecting with $\mathcal{M}$, we assign a weighting factor of infinity in the optimal strip triangulation which is generated by dynamic programming [31], [36]. This effectively ensures a collision-free triangulation (see Fig. 11 for an illustration). If no intersection-free strip triangulation can be found (which rarely occurs in our tests), we separate the upper and the lower contours into different groups.

**Printing direction** Our approach assumes a prescribed printing direction. Integrating the direction as a design variable into the optimization is currently out of reach. A practical solution is to test multiple directions, and select

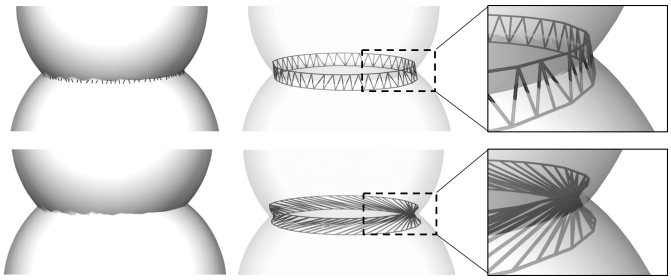


Fig. 11. A naïve strip triangulation may lead to intersection between inner and outer surfaces at the highly curved regions (see the top row), which can be solved by an optimal triangulation (as the bottom row).

the best outcome.

### 8.2 Optimization Objective: Lightweight

The objective of minimizing the material volume for making a lightweight design is equivalent to minimize the area enclosed by $\mathcal{P}$ and $\mathcal{P}'$. Let $A(\cdot)$ be the area of a closed contour. The objective is to find a set of non-uniform offsetting distances $l_j$ to minimize the solid volume,

$$\min_{\{l_j\}} \sum_{\mathcal{P}\in\Theta} h(A(\mathcal{P}) - A(\mathcal{P}')). \tag{11}$$

where $h$ is the thickness of a layer that is assumed to be a constant value in our current implementation. Applying the hollowing operation to all groups of contours can significantly reduce the volume of an input model by producing inner voids. We can further reduce the weight of a model by repeatedly applying the hollowing operation multiple times. Specifically, the resultant solid of a hollowing operation is utilized as the input model of next round of hollowing (see the examples shown in Figs.10 and 12).

### 8.3 Optimization Objective: Static Stability

The objective function in our framework (i.e., Eq.(2)) can be formulated to tackle a variety of applications with demands on physical-properties, such as static stability [1], spinning [2], floating [3], [26], [39], and swinging [27]. We use the static stability as an example to demonstrate how to optimize the center of mass by our support-free hollowing, which can be easily extended to other objective functions.

Assume $\mathcal{M}$ will be fabricated by $m$ layers of slices with thickness $h$, the mass center of a hollowed model $\mathcal{M}'$ can be approximated by

$$\mathbf{c}(\mathcal{M}') = \frac{V(\mathcal{M})\mathbf{c}(\mathcal{M}) - \sum_{i=1}^{m} hA(\mathcal{P}_i')\mathbf{c}(\mathcal{P}_i')}{V(\mathcal{M}) - \sum_{i=1}^{m} hA(\mathcal{P}_i')} \tag{12}$$

where $V(\cdot)$ denotes the volume of an object and $A(\cdot)$ returns the area enclosed by a planar contour. Here, both $V(\mathcal{M})$ and $\mathbf{c}(\mathcal{M})$ can be pre-computed by the layer-based discretization (i.e., using $\{\mathcal{P}_i\}$).

For a given model $\mathcal{M}$, its static stability can be ensured if the projection of its center of mass, $\mathbf{c}(\mathcal{M})$, along the direction of gravity falls inside the convex hull of its contacting points on the ground [1]. Defining $\mathbf{c}(\mathcal{M}')^{\perp}$ as the projection of $\mathbf{c}(\mathcal{M}')$ onto the ground, the criterion of static stability is $\mathbf{c}(\mathcal{M}')^{\perp} \in \mathcal{H}(\mathcal{M})$ with $\mathcal{H}(\mathcal{M})$ denoting the convex hull of $\mathcal{M}$'s contact points on the ground, which is constant as we do not change the outer surface of $\mathcal{M}'$. In practice, we use an inward-offset of $\mathcal{H}(\mathcal{M})$, denoted by $\mathcal{H}_r^{\downarrow}(\mathcal{M})$, with the offsetting distance $r$ as half radius of $\mathcal{H}(\mathcal{M})$'s inscribed circle so that a marginal stand can be avoided.

To make a form easier to be solved in the optimization framework, the closest point $\mathbf{q}$ of $\mathbf{c}(\mathcal{M}')^{\perp}$ on $\mathcal{H}_r^{\downarrow}(\mathcal{M})$'s boundary, denoted by $\partial\mathcal{H}_r^{\downarrow}(\mathcal{M})$, is employed to define the objective function as

$$\min_{\{l_i\}} \sum_{\mathcal{P}\in\Theta} \|\mathbf{c}(\mathcal{M}') - \mathbf{q}\|^2 \tag{13}$$

with

$$\mathbf{q} = \arg\min_{\mathbf{p}\in\partial\mathcal{H}_r^{\downarrow}(\mathcal{M})} \|\mathbf{p} - \mathbf{c}(\mathcal{M}')^{\perp}\|. \tag{14}$$

Figure 13 shows the optimized result of static stability obtained by our algorithm. As shown in Fig. 13(a), the initial solid Letter-P model cannot reach static stability. In this figure, the blue point represents the mass center $\mathbf{c}(\mathcal{M}')$ of a model $\mathcal{M}'$, the red point is its projection on the ground $\mathbf{c}(\mathcal{M}')^{\perp}$, and the green point is the center of $\mathcal{H}(\mathcal{M})$'s inscribed circle (i.e., $\mathbf{c}(\mathcal{H}(\mathcal{M}))$). By applying our method, the projected center of mass is successfully moved inside the convex hull of contacting points on the ground (see Fig. 13(b)).

### 8.4 Experiments and Comparison

Tests have been conducted on a number of 3D models to verify the effectiveness of support-free hollowing. In the following tests, volume reduction is selected as the objective function.

**Self-supporting angle** When different 3D printing methods or different materials are used, a 3D printed model can have different maximal self-supported angles for overhangs. As the support-free constraints imposed in our framework are adaptable to this (i.e., the value of $\alpha$), hollowing results according to different values of $\alpha$ can be easily obtained. Figure 14 shows the support-free hollowing with $\alpha = 45^{\circ}$ (top) and $30^{\circ}$ (bottom). In general a larger self-support angle leads a larger hollowed volume, i.e., a lighter model. Figure 15 shows the results after the first round of hollowing for reducing a cube model with different self-supporting angles.

**Convergence** As aforementioned, we solve the numerical optimization problem by using the *interior point algorithm* – a solver provided by Matlab. In practice, the optimization converges fast. As shown in Fig. 16, our system reduces the area of non-support-free faces to *zero* in a small number of iterations. The numerical optimization is computed on a hollowed model with contours group by group. Figure 16 shows the optimization on the group with the largest volume.

**More results** Figure 17 shows results on six different models. The dancing children model (f) has a much complex topology, and shows a large number of internal voids resulting from the repeated optimization. The computational statistics are summarized in Table 1. Our approach takes about 17.2 to 59.7 minutes to hollow a model in average, which is short comparing to the time of 3D printing fabrication. The volume of models can be reduced in the range

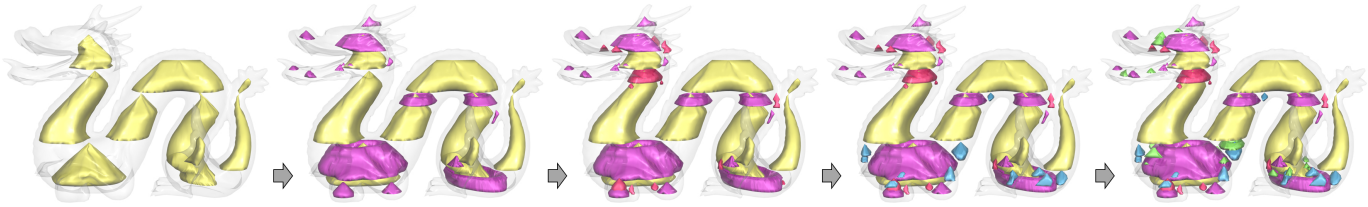


Fig. 12. The volume of a Dragon model can be significantly reduced after repeatedly applying the hollowing operation developed in our framework – inner voids generated in different rounds of hollowing are displayed in different colors. From left to right, the operations result in a volume reduction of $42.9\%$, $57.6\%$, $61.1\%$, $64.9\%$ and $66.6\%$ respectively.

TABLE 1
Computational statistics of our support-free hollowing approach

| Model | Figure | Height ($cm$) | Input Mesh | | | Output Mesh | | | Reduced Volume (%) | Running Time (min.) |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | # Vertices | # Faces | Vol. ($cm^3$) | # Vertices | # Faces | Vol. ($cm^3$) | | |
| Dragon | 12 | 5.55 | 15, 045 | 30, 091 | 27.35 | 25, 599 | 51, 028 | 9.14 | 66.56 | 43.4 |
| Kitten | 2 | 5.00 | 5, 780 | 11, 519 | 15.60 | 21, 873 | 43, 569 | 3.85 | 75.33 | 31.3 |
| Buddha | 14(a) | 5.61 | 5, 601 | 11, 224 | 7.99 | 20, 016 | 39, 966 | 3.05 | 61.83 | 39.4 |
| Buddha | 14(b) | 5.61 | 5, 601 | 11, 224 | 7.99 | 23, 942 | 47, 770 | 3.47 | 56.58 | 47.5 |
| Rabbit | 17(a) | 5.00 | 10, 075 | 20, 146 | 9.46 | 19, 860 | 39, 596 | 2.51 | 73.47 | 18.2 |
| Bear | 17(b) | 5.95 | 13, 826 | 27, 648 | 20.39 | 20, 150 | 40, 043 | 4.79 | 76.51 | 24.6 |
| Horse | 17(c) | 5.60 | 5, 000 | 9, 996 | 33.99 | 16, 750 | 35, 222 | 9.44 | 72.23 | 21.2 |
| Duck | 17(d) | 5.00 | 5, 344 | 10, 688 | 86.11 | 13, 125 | 26, 218 | 20.26 | 76.35 | 17.2 |
| Armodino | 17(e) | 5.83 | 8, 000 | 15, 984 | 10.16 | 22, 967 | 45, 441 | 3.77 | 62.90 | 38.4 |
| Children | 17(f) | 5.33 | 14, 958 | 29, 978 | 27.45 | 31, 915 | 63, 445 | 12.80 | 53.37 | 59.7 |

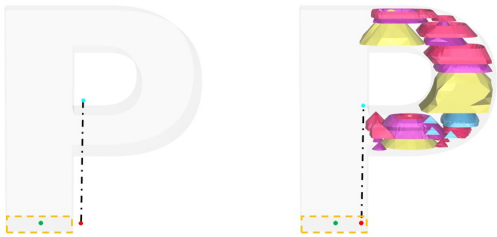


Fig. 13. The initial projected mass center of the Letter-P model is located outside the convex hull of its contacting points on the ground (a). With the optimization of our framework, the model can achieve static stability by introducing self-supported inner voids (b).

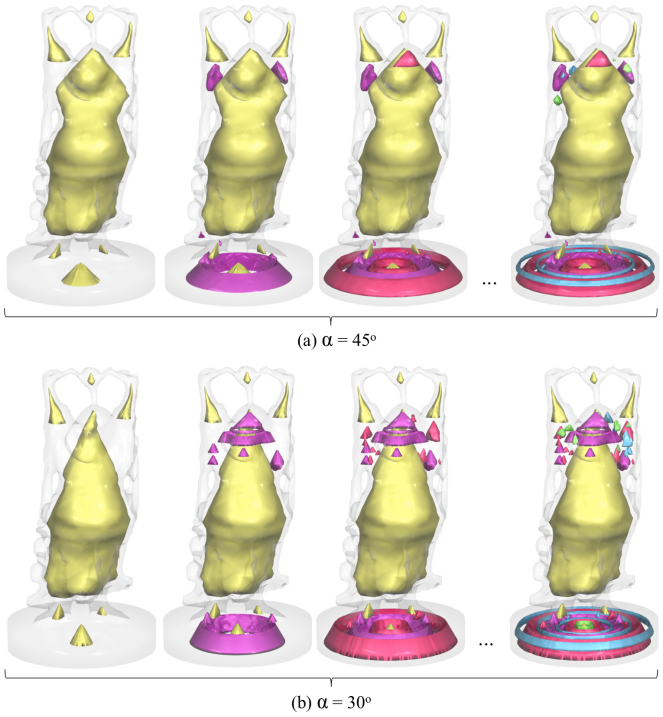



Fig. 14. Progressive results of repeatedly applied support-free hollowing with different maximal self-supporting angles, $\alpha$. The resultant hollowing voids have a total volume of $4.931 \times 10^3\, mm^3$ for $\alpha = 45°$ (a), and $4.514 \times 10^3\, mm^3$ for $\alpha = 30°$ (b).

from $53.4\%$ up to $76.5\%$ – voids generated in all hollowed models are support-free.

**Comparison** Hollowing by uniform offsetting is a method widely used in solid modeling to reduce material costs. However, it cannot guarantee to produce support-free voids. As can be found in Figs. 1(a) and 18(a), both uniformly hollowed Dragon and Kitten models need a large number of additional interior structures to support the overhanging surfaces. This additional supports will very likely change the value of an objective function that has been optimized – for example, adding more weights or moving the center of mass. To avoid this, an optimized result that is support-free becomes very important. Recently, Wu et al. [11] proposed an algorithm to generate optimized infill structures with self-supporting rhombus. Although the models generated by their method can be support-freely manufactured, the specific rhombic structures used in their method limit the sparsity of inner voids and therefore also the solution space. The new method presented in this paper is able to generate support-free inner voids in many other kinds of shapes, as shown in Figs. 1 and 18. The new method can achieve a larger volume reduction than [11]. Consequently, our approach has a larger solution space for satisfying design demands. For example the hand model shown in Fig. 19, a stable stand cannot be found by using the rhombic infill since the self-supporting rhombic infill is rather dense.

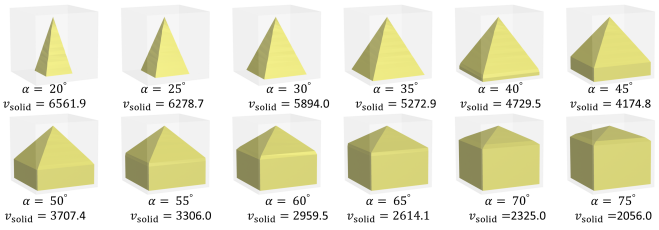



Fig. 15. As the maximal self-supporting angle increases from $20°$ (top leftmost) to $75°$ (bottom rightmost), the remaining solid volume decreases from $6.562 \times 10^3$ to $2.056 \times 10^3$ (unit: $mm^3$).

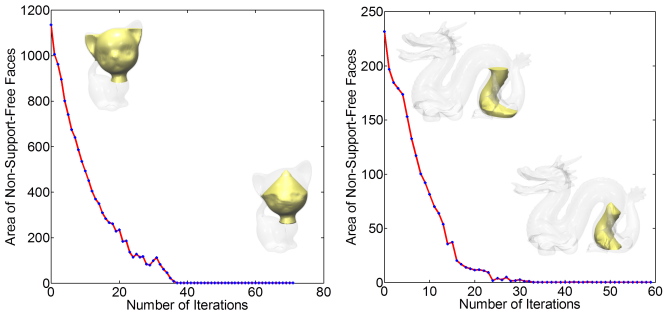



Fig. 16. The convergence curves of computation by our framework on the Dragon and the Kitten model. To evaluate the performance in support-free optimization, the change of total areas that need to add supporting structures is studied.

However, the support-free hollowing successfully computes a stable configuration with much sparser structures (right).

### 8.5 Discussion

Our support-free hollowing is based on the commonly used offsetting operator. Offsetting implicitly restricts the topology of the resulting shape, and thus restricts the solution space. To enlarge the solution space, a repetitive hollowing strategy is proposed, leading to a further reduction of the objective value.

Our current implementation is not optimized for speed. Since each void can be optimized independently, a CPU parallelization for these tasks is straightforward. Multi-resolution hollowing can potentially further improve the speed. A relatively slow numerical computation in our approach is mainly caused by the large number of variables to be optimized. The strategy of using a reduced representation of shape by linear combination of basis functions (e.g., [39]) can be employed in our future research to speed up the computation. We leave this as future work.

It is interesting to compare our result with the infill structure generated by Christiansen et al. [28], where the uniform infill structures are automatically added to support a 3D printed model. Although sparse infill can be generated by their method, the property of self-supporting cannot be guaranteed. As illustrated in Fig. 20, the overhang region formed at the ceiling of inner surface will be damaged during 3D printing as no supporting infill is added below it. Our result (see Fig. 20, bottom row) with the same total volume (i.e., also the same weight) does not have such a problem, since our method ensures the geometric property of self-supporting.

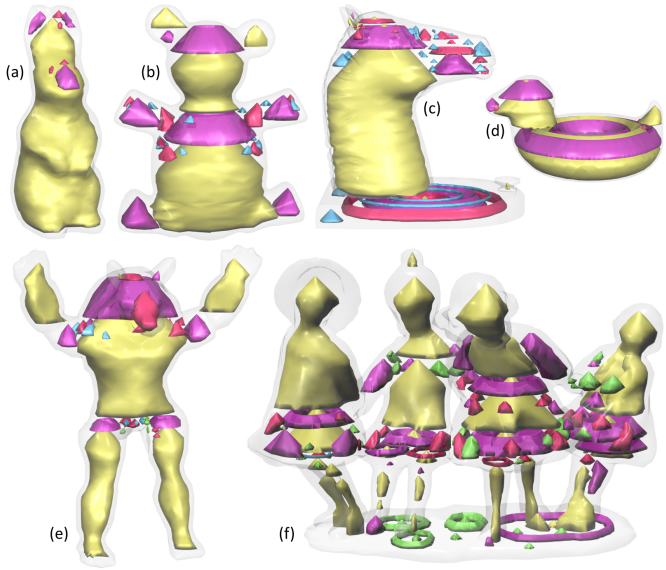



Fig. 17. More results generated by our support-free hollowing framework. Models with complex topology such as (f) can be well handled by our method. The different colors indicate the progressive results of repeated hollowing.

## 9 Conclusion

In this paper, we have presented a computational framework that is able to generate optimized hollowing models according to different objective functions. The inner surfaces of a hollowed model are guaranteed to be support-free. This eliminates the need of additional supporting structure for 3D printing. The functionality of our approach has been demonstrated by the applications of lightweight design and static stability. The effectiveness of our method has been verified on a number of models.

## Acknowledgments

This work is partially supported by the Natural Science Foundation of China (61725204, 61702079, 61432003, 61628211, 61661130156, 61370143), China Postdoctoral Science Foundation (2016M601308), and Fundamental Research Fund (DUT16RC(3)061). C.C.L. Wang would also thank the support of Hong Kong RGC GRF Grant (14207414) and the Open Research Fund of Key Laboratory of High Performance Complex Manufacturing at Central South University, China.

## References

[1] R. Prévost, E. Whiting, S. Lefebvre, and O. Sorkine-Hornung, "Make it stand: balancing shapes for 3d fabrication," *ACM Trans. Graph.*, vol. 32, no. 4, p. 81, 2013.

[2] M. Bächer, E. Whiting, B. Bickel, and O. Sorkine-Hornung, "Spin-it: Optimizing moment of inertia for spinnable objects," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 96:1–96:10, Jul. 2014.

[3] P. Musialski, T. Auzinger, M. Birsak, M. Wimmer, and L. Kobbelt, "Reduced-order shape optimization using offset surfaces," *ACM Trans. Graph.*, vol. 34, no. 4, pp. 102:1–102:9, Jul. 2015.

[4] W. Wang, T. Y. Wang, Z. Yang, L. Liu, X. Tong, W. Tong, J. Deng, F. Chen, and X. Liu, "Cost-effective printing of 3d objects with skin-frame structures," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 6, p. 177, 2013.

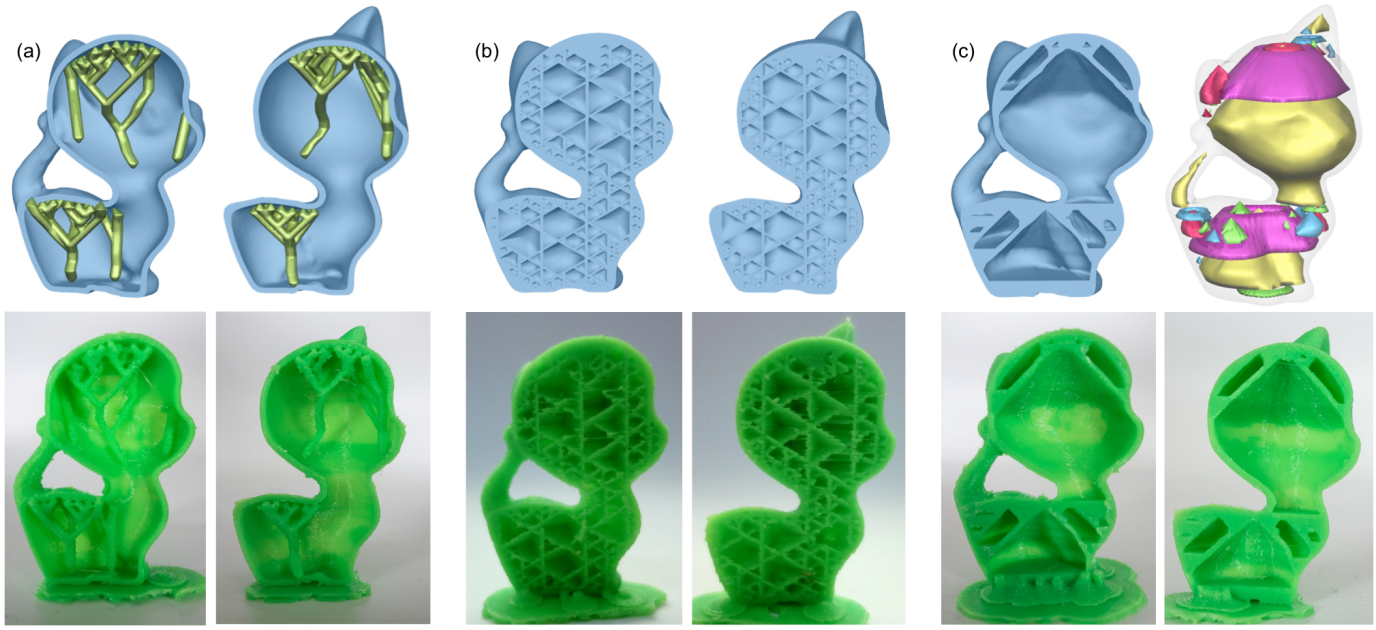



Fig. 18. Comparison of the results generated by different methods on the Kitten model – both the computational results and the 3D printed models are shown here: (a) the result of uniform hollowing needs interior supporting structures for fabrication, (b) the result with rhombic infill structures [11] (with $45.5\%$ volume reduced), and (c) the result of support-free hollowing presented in this paper (with $75.3\%$ volume reduced).

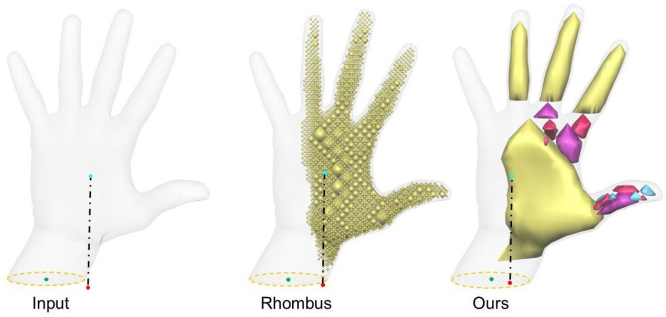



Fig. 19. An input hand model (left) can hardly be made static-stable by using the self-supporting rhombic infill [11] (middle), while our method is able to compute sparse support-free voids to achieve this goal (right). The vertical lines indicate the projection of the center of mass.

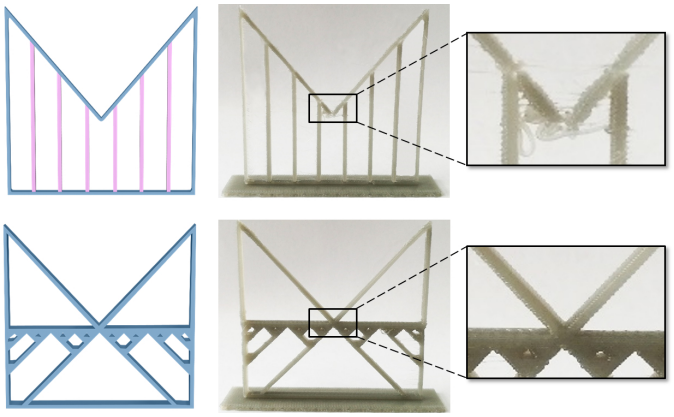


Fig. 20. Top row: A model with infill structure generated by the method of Christiansen et al. [28] – the total volume is $3,894\,mm^3$. Bottom row: A model generated by our method with a volume of $3,872\,mm^3$. The results of fabrication produced by a FDM printer are given on the right.


[5] L. Lu, A. Sharf, H. Zhao, Y. Wei, Q. Fan, X. Chen, Y. Savoye, C. Tu, D. Cohen-Or, and B. Chen, "Build-to-last: Strength to weight 3d printed objects," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 4, p. 97, 2014.

[6] J. Wu, C. Dick, and R. Westermann, "A system for high-resolution topology optimization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 3, pp. 1195–1208, March 2016.

[7] J. Dumas, J. Hergel, and S. Lefebvre, "Bridging the gap: Automated steady scaffoldings for 3d printing," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 98:1–98:10, Jul. 2014.

[8] J. Vanek, J. A. G. Galicia, and B. Benes, "Clever support: Efficient support structure generation for digital fabrication," *Comput. Graph. Forum*, vol. 33, no. 5, pp. 117–125, 2014.

[9] P. Huang, C. C. L. Wang, and Y. Chen, "Algorithms for layered manufacturing in image space," in *ASME Advances in Computers and Information in Engineering Research*, 2014, pp. 377–410.

[10] X.-R. Wei, Y.-H. Zhang, and G.-H. Geng, "No-infill 3d printing," *3D Research*, vol. 7, no. 3, p. 24, 2016.

[11] J. Wu, C. C. Wang, X. Zhang, and R. Westermann, "Self-supporting rhombic infill structures for additive manufacturing," *Computer-Aided Design*, vol. 80, pp. 32–42, 2016.

[12] E. Vouga, M. Höbinger, J. Wallner, and H. Pottmann, "Design of self-supporting surfaces," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 87:1–87:11, Jul. 2012.

[13] Y. Liu, H. Pan, J. Snyder, W. Wang, and B. Guo, "Computing self-supporting surfaces by regular triangulation," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 92:1–92:10, Jul. 2013.

[14] M. Deuss, D. Panozzo, E. Whiting, Y. Liu, P. Block, O. Sorkine-Hornung, and M. Pauly, "Assembling self-supporting structures," *ACM Trans. Graph.*, vol. 33, no. 6, pp. 214:1–214:10, Nov. 2014.

[15] T. Reiner and S. Lefebvre, "Interactive modeling of support-free shapes for fabrication," in *Proceedings of Eurographics 2016 - Short Papers*, 2016.

[16] Y. Xie and X. Chen, "Support-free interior carving for 3d printing," *Visual Informatics*, vol. 1, no. 1, pp. 9–15, 2017.

[17] D. Pavic and L. Kobbelt, "High-Resolution Volumetric Computation of Offset Surfaces with Feature Preservation," *Computer Graphics Forum*, 2008.

[18] S. Liu and C. C. Wang, "Fast intersection-free offset surface generation from freeform models with triangular meshes," *IEEE Transactions on Automation Science and Engineering*, vol. 8, no. 2, pp. 347–360, 2011.

[19] Y. Chen and C. C. L. Wang, "Uniform offsetting of polygonal model based on layered depth-normal images," *Computer-Aided Design*, vol. 43, no. 1, pp. 31–46, 2011.

[20] C. C. L. Wang and D. Manocha, "GPU-based offset surface computation using point samples," *Computer-Aided Design*, vol. 45, no. 2, pp. 321–330, 2013.

[21] B. Bickel, P. Kaufmann, M. Skouras, B. Thomaszewski, D. Bradley, T. Beeler, P. Jackson, S. Marschner, W. Matusik, and M. Gross, "Physical face cloning," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 118:1–118:10, Jul. 2012.

[22] X. Zhang, X. Le, Z. Wu, E. Whiting, and C. C. Wang, "Data-driven bending elasticity design by shell thickness," *Computer Graphics Forum*, vol. 35, no. 5, pp. 157–166, 2016.

[23] O. Sigmund, "A 99 line topology optimization code written in matlab," *Struct. Multidiscip. Optim.*, vol. 21, no. 2, pp. 120–127, Apr. 2001.

[24] D. Yamanaka, H. Suzuki, and Y. Ohtake, "Density aware shape modeling to control mass properties of 3d printed objects," in *SIGGRAPH Asia 2014 Technical Briefs*. ACM, 2014, p. 7.

[25] J. Wu, L. Kramer, and R. Westermann, "Shape interior modeling and mass property optimization using ray-reps," *Computers & Graphics*, vol. 58, pp. 66 – 72, 2016.

[26] L. Wang and E. Whiting, "Buoyancy optimization for computational fabrication," *Computer Graphics Forum (Eurographics 2016)*, 2016.

[27] H. Zhao, C. Hong, J. Lin, X. Jin, and W. Xu, "Make it swing," *Comput. Aided Geom. Des.*, vol. 43, pp. 226–236, 2016.

[28] A. N. Christiansen, R. Schmidt, and J. A. Brentzen, "Automatic balancing of 3d models," *Computer-Aided Design*, vol. 58, no. 0, pp. 236–241, 2015.

[29] C. C. Wang and Y. Chen, "Thickening freeform surfaces for solid fabrication," *Rapid Prototyping Journal*, vol. 19, no. 6, pp. 395–406, 2013.

[30] W. Wang, H. Chao, J. Tong, Z. Yang, X. Tong, H. Li, X. Liu, and L. Liu, "Saliency-preserving slicing optimization for effective 3d printing," *Computer Graphics Forum*, vol. 34, no. 6, pp. 148–160, 2015.

[31] D. Meyers, S. Skinner, and K. Sloan, "Surfaces from contours," *ACM Trans. Graph.*, vol. 11, no. 3, pp. 228–258, 1992.

[32] M. Campen and L. Kobbelt, "Polygonal boundary evaluation of minkowski sums and swept volumes," *Computer Graphics Forum*, vol. 29, no. 5, pp. 1613–1622, 2010.

[33] Y. Chen and C. C. L. Wang, "Uniform offsetting of polygonal model based on layered depth-normal images," *Computer-Aided Design*, vol. 43, no. 1, pp. 31–46, 2011.

[34] J. Nocedal and S. Wright, *Numerical Optimization*, 2nd ed. Springer, 2006.

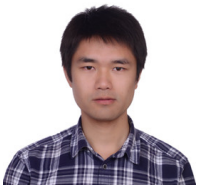
[35] L. Liu, A. Shamir, C. Wang, and E. Whitening, "3d printing oriented design: Geometry and optimization," in *SIGGRAPH Asia 2014 Courses*, 2014.

[36] C. C. L. Wang and K. Tang, "Optimal boundary triangulations of an interpolating ruled surface," *Journal of Computing and Information Science in Engineering*, vol. 5, no. 4, pp. 291–301, 2005.

[37] N. Amenta, S. Choi, and R. K. Kolluri, "The power crust," in *Proceedings of the Sixth ACM Symposium on Solid Modeling and Applications*. ACM, 2001, pp. 249–266.

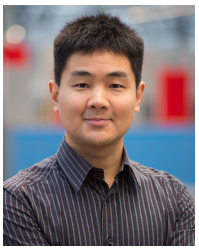
[38] S. Gottschalk, M. C. Lin, and D. Manocha, "OBBTree: A hierarchical structure for rapid interference detection," in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '96, 1996, pp. 171–180.

[39] P. Musialski, C. Hafner, F. Rist, M. Birsak, M. Wimmer, and L. Kobbelt, "Non-linear shape optimization using local subspace projections," *ACM Transactions on Graphics*, vol. 35, no. 4, 2016.

**Weiming Wang** is currently a lecturer in Dalian University of Technology, Department of Mathematical Sciences. He received his B.S and PhD degrees in 2010 and 2016 from Dalian University of Technology. His research interests are Computer Graphics and 3D Printing.

**Yong-Jin Liu** is currently an associate professor with the TNList, Department of Computer Science and Technology, Tsinghua University, China. He received his B.Eng degree from Tianjin University, China, in 1998, and the PhD degree from the Hong Kong University of Science and Technology, Hong Kong, China, in 2004. His research interests include computational geometry, computer graphics and computer-aided design. He is a senior member of the IEEE and a member of ACM.



**Jun Wu** is an Assistant Professor at the Department of Design Engineering, Delft University of Technology, The Netherlands. Prior to his current position, he was a H.C. Ørsted postdoc fellow at the Technical University of Denmark. He received a PhD in Computer Science in 2015 from the Technical University of Munich, Germany, and a PhD in Mechanical Engineering in 2012 from Beihang University, Beijing, China. His research is focused on geometric and physical modeling, with applications in surgical simulation and computational fabrication.

**Shengjing Tian** is currently a graduate student in Dalian University of Technology. His research interests are scene reconstruction.

**Charlie C.L. Wang** is currently a Professor and Chair of Advanced Manufacturing in the Department of Design Engineering at Delft University of Technology, The Netherlands. Prior to this position, he was a Professor of Mechanical and Automation Engineering at the Chinese University of Hong Kong (CUHK), where he started his academic career in 2003. Prof. Wang received a few awards from professional societies including the ASME CIE Excellence in Research Award (2016), the Best Paper Awards of ASME CIE Conferences (twice in 2008 and 2001 respectively), and the NAMRI/SME Outstanding Paper Award (2013). He serves on the editorial board of a few journals, including Computer-Aided Design and IEEE Transactions on Automation Science and Engineering. His research interests are geometric computing, computer-aided design, advanced manufacturing, and computational physics.

**Ligang Liu** is a Professor at the School of Mathematical Sciences, University of Science and Technology of China. His research interests include digital geometric processing, computer graphics, and image processing. He serves as the associated editors for journals of IEEE Transactions on Visualization and Computer Graphics, IEEE Computer Graphics and Applications, Computer Graphics Forum, Computer Aided Geometric Design, and The Visual Computer. He served as the conference co-chair of GMP 2017 and the program co-chairs of GMP 2018, CAD/Graphics 2017, CVM 2016, SGP 2015, and SPM 2014. His research works could be found at his research website: http://staff.ustc.edu.cn/~lgliu

**Xiuping Liu** is currently a Professor in Dalian University of Technology, Department of Mathematical Sciences. She received her B.S. and M.S. degrees from Jilin University in 1987 and 1990, and PhD degree from Dalian University of Technology in 1999. Her research interests are computational geometry and computer graphics