UNITED STATES PATENT AND TRADEMARK OFFICE

---

BEFORE THE PATENT TRIAL AND APPEAL BOARD

---

3D Systems Corporation et al.,

Petitioner

v.

Intrepid Automation, Inc.,
Patent Owner.

---

Inter Partes Review (IPR) Proceeding Number: 2025-01242
U.S. Patent No. 11,338,511

---

Exhibit Number: **Exhibit 1029 for Petitioner**

Exhibit Title: Yi *et al.*, "Delta DLP 3-D Printing of Large Models," *IEEE Transactions on Automation Science and Engineering*, Vol. 15, No. 3 (DOI: 10.1109/TASE.2017.2751664) ("Yi")

Date of Filing: July 18, 2025

Filed By: Petitioner 3D Systems Corporation et al.,

# Delta DLP 3-D Printing of Large Models

Ran Yi, Chenming Wu, Yong-Jin Liu, *Senior Member, IEEE*, Ying He, and Charlie C. L. Wang, *Senior Member, IEEE*

***Abstract*— This paper presents a 3-D printing system that uses a low-cost off-the-shelf consumer projector to fabricate large models. Compared with traditional digital light processing (DLP) 3-D printers using a single vertical carriage, the platform of our DLP 3-D printer using delta mechanism can also move horizontally in the plane. We show that this system can print 3-D models much larger than traditional DLP 3-D printers. The major challenge to realize 3-D printing of large models in our system comes from how to cover a planar polygonal domain by a minimum number of rectangles with fixed size, which is NP-hard. We propose a simple yet efficient approximation algorithm to solve this problem. The key idea is to segment a polygonal domain using its medial axis and afterward merge small parts in the segmentation. Given an arbitrary polygon $\Omega$ with $n$ generators (i.e., line segments and reflex vertices in $\Omega$), we show that the time complexity of our algorithm is $O(n^2 \log^2 n)$ and the number of output rectangles covering $\Omega$ is $O(Kn)$, where $K$ is an input-polygon-dependent constant. A physical prototype system is built and several large 3-D models with complex geometric structures have been printed as examples to demonstrate the effectiveness of our approach.**

***Note to Practitioners*—Low-cost 3-D printers and 3-D printing of large models are two important but often conflicting goals in manufacturing industry. Usually low-cost devices such as DLP 3-D printer using an off-the-shelf consumer projector cannot print large models, due to the small projection area of the projector. In this paper, we propose to horizontally move the platform such that a large area can be printed by the composition of multiple small projection areas. Based on this new mechanism, we propose a simple yet efficient algorithm to cover an arbitrary polygonal shape (possibly with holes or multiple disjoint polygons) by a small set of rectangles with fixed size. Our algorithm is theoretically sound and can be easily implemented. We built a physical prototype system of the proposed low-cost Delta DLP 3-D printer, which successfully prints several large models with satisfied mechanical properties.**

***Index Terms*—Manufacturing, mechanisms, primary topics, secondary topics.**

Manuscript received November 2, 2016; revised May 12, 2017; accepted September 10, 2017. Date of publication October 2, 2017; date of current version July 2, 2018. This paper was recommended for publication by Associate Editor S. Mishra and Editor D. Tilbury upon evaluation of the reviewers' comments. This work was supported in part by the Natural Science Foundation of China under Grant 61432003, Grant 61661130156, and Grant 61521002, in part by the National Key Research and Development Plan under Grant 2016YFB1001202, and in part by the Royal Society-Newton Advanced Fellowship under Grant NA150431, Grant MOE2013-T2-2-011, and Grant RG23/15. The work of C. C. L. Wang was supported in part by Hong Kong RGC General Research Fund under Grant CUHK/14207414 and in part by the Open Research Fund of Key Laboratory of High Performance Complex Manufacturing at Central South University, China. *(Ran Yi and Chenming Wu are joint first authors.) (Corresponding author: Yong-Jin Liu.)*

R. Yi, C. Wu, and Y.-J. Liu are with the TNList, Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China (e-mail: yr16@mails.tsinghua.edu.cn; wcm15@mails.tsinghua.edu.cn; liuyongjin@tsinghua.edu.cn).

Y. He is with the School of Computer Engineering, Nanyang Technological University, Singapore 639798 (e-mail: YHe@ntu.edu.sg).

C. C. L. Wang is with the Department of Design Engineering, TU Delft Robotics Institute, Delft University of Technology, 2628 Delft, The Netherlands (e-mail: c.c.wang@tudelft.nl).

This paper has supplementary downloadable multimedia material available at http://ieeexplore.ieee.org provided by the authors. The Supplementary Material contains material that is not included within the paper itself. This material is 15.1 MB in size.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASE.2017.2751664

## I. Introduction

DIGITAL light processing (DLP) is a 3-D printing technology that use ultraviolet (UV)-light to solidify liquid photopolymer [8]. The technique of DLP is widely employed in 3-D printing because of its fast printing speed and its simple mechanism in hardware. When preparing the information for fabrication, a 3-D CAD model is first sliced by a set of parallel planes and each slice is later converted into a 2-D mask image. By projecting the mask image onto a photocurable liquid surface, a layer of solid in the same shape can be formed. A 3-D object can be fabricated in this way layer by layer. Different from some other *stereolithography* (SLA) techniques which use point or line light sources, DLP uses an areal light source such that the whole mask image can be projected at the same time (e.g., the commonly used low-cost off-the-shelf consumer projector). As a result, the fabrication process of DLP 3-D printer is much faster than other point (or line)-based 3-D printing techniques.

Most photopolymers react to radiation in the UV wavelength ranges. To successfully solidify fluid photopolymers, sufficient light intensity must be projected onto the surface of liquid tank in DLP-based 3-D printing. Off-the-shelf projectors are good enough to take this job (e.g., a consumer-level 1080P Projector is used in the Phoenix Touch 1080P DLP 3-D Printer [2]). To ensure the light intensity and reduce the size of a machine, optimal lenses are placed between the projector and the working surface to shorten the distance of projection. An illustration of DLP-based 3-D printing can be found in Fig. 1. However, this setup results in a very small working area (e.g., only 34 mm $\times$ 34 mm) in our hardware setup when using a SONY VPL-EW246 projector), which prevents the fabrication of large models. Enlarging the distance between a projector and the working surface will need a more powerful light source to ensure the light intensity of projection for consolidation. More seriously, the resolution of projected masks is limited by the resolution of a projector, which is a technical barrier hard to overcome now. A project mask with lower resolution will lead to a fabrication result with less geometric details (see [27] and [28]).

To overcome aforementioned difficulty, our basic idea is to make multiple projections for each layer of the 3-D model to fabricate large models. Our method can work with different

1545-5955 © 2017 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

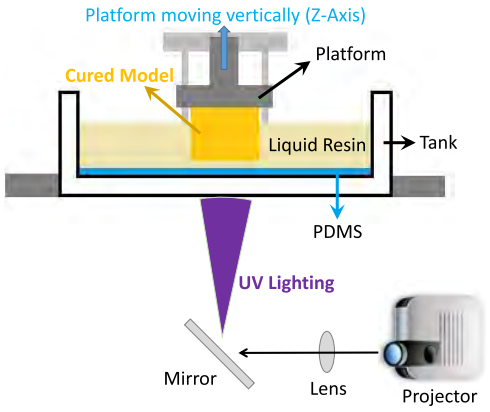



Fig. 1. Working principle of a conventional DLP 3-D printer, in which the platform can only move vertically.

projectors or other light sources, and then makes a good tradeoff between the powerfulness of a light source and the hardware cost. We develop a working system to realize this function of DLP-based 3-D printing with a large size. In addition to moving vertically (along $z$-axis), the working platform holding cured models can move and rotate horizontally (i.e., in the $x$-$y$ plane). This is realized by a delta mechanism—a parallel robot (details can be found in Section III). A practical challenge is how to decompose a large planar shape into an optimal set of smaller pieces, where each piece fits within the projected area of light source. The smaller number of pieces, the faster a planar polygonal region with large area can be fabricated. As discussed in Section II, this problem is generally NP-hard. In this paper, we improve upon our previous work [26] in three aspects and make the following new contributions.

1) We propose a simple yet effective algorithm based on a medial-axis-driven segmentation. The time complexity of our algorithm is $O(n^2 \log^2 n)$, where $n$ is the number of generators (i.e., line segments and reflex vertices) in the input polygon. As a comparison, the bounding-box-driven algorithm in [26] has $O(n^3 \log n)$ time complexity.
2) Our proposed algorithm covers an arbitrary multiple-connected polygon (i.e., a polygon having multiple holes) using a small number of rectangles with fixed sizes (i.e., the maximal area of projection), and in particular, the number of rectangles is bounded by $O(Kn)$, where $K$ is an input-polygon-dependent constant. This bound does not exist for the algorithm in [26].
3) A tension test has been taken on an INSTRON 5943 universal testing system, showing that a model printed by our proposed multiple projections for each layer has the same tension strength as the model printed by traditionally single projections.

## II. Related Work

Nowadays, 3-D printing techniques are widely used in manufacturing automation (see [17], [24], and [25]). In our work, we pay attention to a DLP-based 3-D printing called *MIP-based stereolithograhy* (MIP-SLA) [20], [21], which follows the SLA technique but replace the point or line light sources with an areal light source. As illustrated in Fig. 1, liquid photopolymer resin to be cured is contained in a transparent tank. In a bottom-up projection system, UV lights controlled by mask images are projected onto the bottom surface of resin and quickly cure it through the transparent tank. After a layer is cured, the platform is moved up (vertically in $z$-axis) and form a small gap between the built model and bottom of the tank. To prevent sticking the cured layer onto the tank, a coating material polydimethylsiloxane (PDMS) is used. This mask image projection (MIP) and resin curing process is iteratively applied to print the model layer by layer. A major drawback of this conventional DLP framework is that the platform can only move vertically and then the maximal cross sections of printed models are restricted by the area of projection.

In this paper, we present a new DLP system that allows the platform to move not only vertically (in $z$-axis) but also horizontally (in $x$-$y$ plane). Then, larger layers of 3-D models can be printed by multiple projections of a consumer projector. To decompose a planar shape into an optimal set of rectangles, two types of decompositions have been considered [10]: *partitions* and *coverings*. It is called a *partition* if a shape is decomposed into nonoverlapped subregions, the union of which is exactly equal to the target polygon. If the subregions are allowed to overlap, as long as their union is equal to the target polygon, they are called a *covering*. Both partition and covering problems have been well studied in computational geometry.

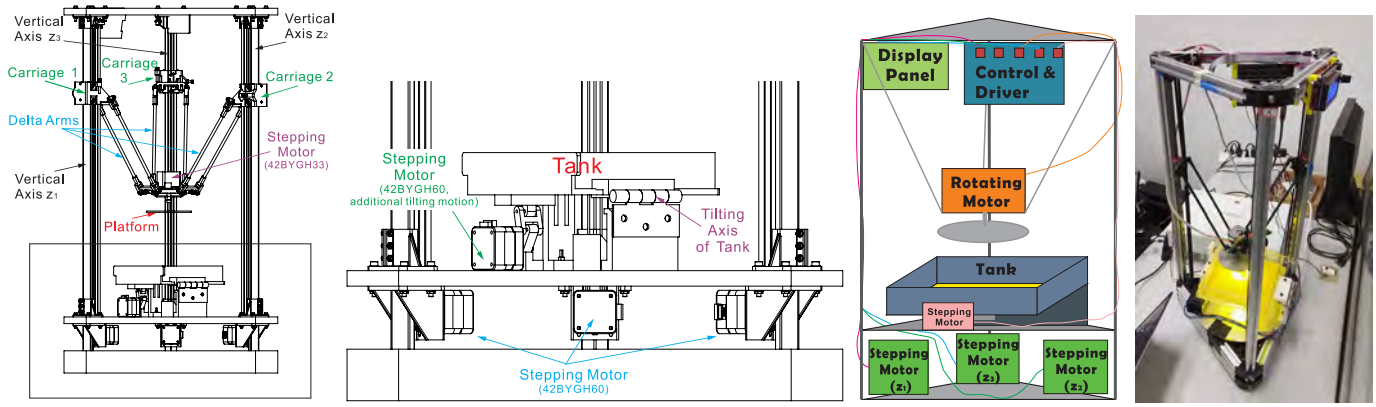
The problem of covering a polygon with a minimum number of convex components is NP-hard (see [6] and [19]). Even for the special problem of covering a rectilinear polygon with squares, finding a minimum of such covering is also NP-hard [3]. Many practical approximation algorithms have been proposed in the field of VLSI chip design (see [9] and [22]). These algorithms mainly focused on the study of a collection of rectangles with sides parallel to two orthogonal directions. However, the rectangles are allowed to be in any orientation in our application. A constant-factor approximation algorithm was proposed in [11]. But this method can only cover a polygon without any acute interior angles. In contrast, the planar shape of a layer in our system can have arbitrary polygons with holes.

The problem of partitioning a polygon with holes into a minimum number of convex subpolygons is NP-hard [12]. The special problem of partitioning a rectilinear polygon to a minimum number of squares is also NP-hard [1]. Most existing practical approximation algorithms to solve partition problems only deal with rectilinear/orthogonal polygons (see [10]). In our system, we are facing a more general and difficult problem to partition an arbitrary polygonal shape possibly with holes or multiple disconnected regions.

## III. Hardware

We implement a delta structure such that the platform can be moved both vertically and horizontally. A delta 3-D printer is in fact a parallel robot [4] and has been used in 3-D printing

3D Systems-Exhibit 1029-Page 2 of 12




(a) (b) (c) (d)

Fig. 2. Mechanical structure, control module, and our prototype of a delta DLP 3-D printer. Vector graphics in (a), (b), and (c) are provided for zoomed-in view examination. (a) Mechanical structure. (b) Zoomed-in view of the tilting device in the box of (a). (c) Control module. (d) Our system. See accompanying video for more details.

by *fused deposition modeling*. To the best of our knowledge, the delta structure has not been used in DLP 3-D printing yet.

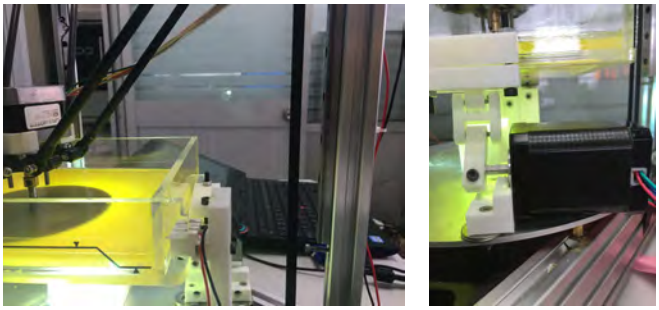
Refer to Fig. 2(a), a delta structure has three vertical axes labeled $z_1$, $z_2$, and $z_3$. Each axis has a carriage that can slide along the vertical sliding guide. A carriage and the platform are connected using a pair of parallel arms. The platform can be moved to any position in a cylindrical working envelope by positioning the three carriages along the vertical axes simultaneously using geometric algorithms. Details about these geometric algorithms can be found in [4].

The control module of a delta structure consists of an Arduino chip, an liquid-crystal display (LCD) display, and four stepping motors as illustrated in Fig. 2(c). Three stepping motors (42BYGH60) work cooperatively for driving the carriages to position the platform. Different from conventional delta structure, a lighter and smaller stepping motor (42BYGH33) is installed for rotating the platform around the $z$-axis. All these motors are driven by an A4988 DMOS Microstepping driver from Allegro Microsystem. The whole system is controlled and communicated by an Arduino chip. In addition, a common 1602 LCD display is connected with the Arduino board to show parameters including positions and angles of the platform for an easier debugging.

To use a delta structure in DLP 3-D printing, a device that can safely separate each solidified layer and the PDMS film is required. Accordingly, we design and install an adaptive tilting mechanical device with electrical control to manipulate the resin tank [Figs. 2(b) and 3]. In this device, one side of the resin tank is fixed by a detachable fixture with the aluminum hinge. The opposite side of the resin tank is fixed by another well-designed detachable fixture with a stepping motor (42BYGH60). We use a height-adjustable limit switch to form a closed loop control. Using this device, after solidification of each layer, the tank is gradually tilted to separate the layer and the PDMS film. We name the above hardware system with the tilting device as *delta DLP 3-D printer*.

## IV. Software

In DLP-based 3-D printing, the 3-D digital model of an object is usually sliced by a set of parallel planes and each thin slice is fabricated by projecting a 2-D mask image onto the surface of photocurable liquid. As illustrated in Fig. 4(a), the white region in each mask image means a full light intensity of projection and no light is projected into the black area. Each of the white regions can be modeled by a polygon and a slice could contain several disconnected polygons.



(a) (b)

Fig. 3. Tilting mechanical device that can safely separate each solidified layer and the PDMS film. One side of the resin tank is fixed by a detachable fixture with the aluminum hinge, and the opposite side is fixed by another well-designed detachable fixture with a stepping motor (42BYGH60). (a) Front view. (b) Side view.

To precisely define the problem (covering a polygon with rectangles) and describe our proposed solution (medial-axis-driven segmentation and covering), some preliminaries are first introduced below.

### A. Preliminaries

Denote the cardinality, closure, interior, and boundary of a set $X$ as $|X|$, $\overline{X}$, $\mathring{X}$ and $\partial X$, respectively. Given two distinct points $p_1$ and $p_2$ in $\mathbb{R}^2$, a *closed* line segment $\overline{p_1 p_2}$ connecting $p_1$ and $p_2$ is the union of two endpoints and the open line segment $p_1 p_2$.

*Definition 1:* A polygonal domain $\Omega$ is a bounded, connected open set in $\mathbb{R}^2$ and its boundary $\partial\Omega = \overline{\Omega} \setminus \Omega$ consists of a finite number of mutually disjoint simple closed curves. Each closed curve consists of a finite number of closed line segments.

3D Systems-Exhibit 1029-Page 3 of 12

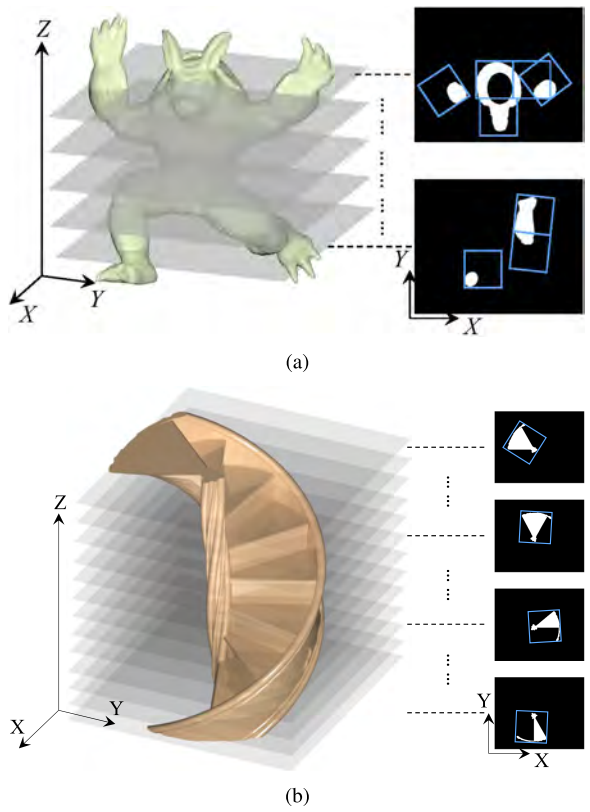



Fig. 4. To prepare mask images for DLP-based 3-D printing, 3-D models are sliced by a set of parallel planes (perpendicular to $z$-axis) and each slice is represented by a 2-D binary image for projection. The blue rectangles specify the maximal region can be projected (right): (a) Slices contain disjoint polygons and polygons can have holes. (b) Polygon in each slice can be covered one projection area only if the projected area can be translated and rotated.

In $\partial\Omega$, the simple closed curve that bounds the unbounded, connected component in $\mathbb{R}^2 \setminus \Omega$ is called the *outer* boundary of $\Omega$, and each of the remaining simple closed curves is called an *inner* boundary. The genus of $\Omega$ is the number $m$ of inner boundaries (we also say $\Omega$ has $m$ holes). A polygonal domain $\Omega$ is *simple* if it has no holes; otherwise, it is *multiply connected*.

The minimal Euclidean distance between a point $x$ and $\partial\Omega$ is denoted by $dist(x, \partial\Omega)$. Let $\Lambda(x) = \{p \in \partial\Omega : \|x - p\|_2 = dist(x, \partial\Omega)\}$ be the set of closest boundary points of $x$ in $\partial\Omega$.

*Definition 2:* The medial axis $M(\Omega)$ of $\Omega$ is the set of points in $\Omega$ that has at least two closest boundary points

$$M(\Omega) = \{x \in \Omega : |\Lambda(x)| \geq 2\}.$$

Elements in $M(\Omega)$ are called medial points. If $|\Lambda(x)| \geq 3$, $x$ is called a branch point of degree $|\Lambda(x)|$ [see Fig. 5(b) (red points)].

A vertex in $\partial\Omega$ is called *reflex* if its interior angle is larger than $\pi$; otherwise, the vertex is called *convex*.

*Definition 3:* Generators in $\partial\Omega$ consists of open line segments and reflex vertices.

If there are finitely many generators in $\partial\Omega$, the closure of the medial axis $\overline{M}(\Omega)$ is a connected planar graph with finitely many nodes and edges [5], in which each node is a branch point and each edge is a single trimmed bisector of two generators [18].

1) If both generators are points, their bisector is a straight line.
2) If both generators are open line segments, their bisector is a straight line segment.
3) If the two generators are a point and an open line segment, their bisector is an open parabolic arc.

We call the edges in $M(\Omega)$ the *medial edges* and denote the two generators devoted to a medial edge $e$ as $g_1(e)$ and $g_2(e)$.

### B. Problem Identification

Let $\psi$ be the maximal area that can be projected by a light source, which is a rectangular region in our delta DLP 3-D printer. Each slice can be represented by a 2-tuple $(\Xi, z)$, where $z$ is the height of slice and $\Xi = (\Omega_1 \cup \Omega_2 \cup \cdots \cup \Omega_n)$ is a set of $n$ disconnected polygonal domains to be solidified. Note that any $\Omega_i$ can be multiply connected. In traditional DLP 3-D printers, the platform can only move vertically. In such case, even if a polygonal domain $\Omega_i$ in a slice can be covered by $\psi$, it is also possible that the whole model cannot be fabricated when $\psi \subseteq \cup_j \Omega_j$ for all layers [see Fig. 4(b)]. Our system overcomes this problem by allowing the platform to move and rotate in the $x$-$y$ plane, where the horizontal motion of platform is supervised by a geometric algorithm solving the following problem.

*Problem 1:* Given a set of disconnected polygonal domains $\Xi$ in a plane, $\Xi = (\Omega_1 \cup \Omega_2 \cup \cdots \cup \Omega_n)$, $\Omega_i \cap \Omega_j = \emptyset$, $\forall i \neq j$, find a minimum set of rectangles $\Psi = \{\psi_k, k = 1, 2, \cdots\}$, all with a fixed size $w \times h$, such that $\Xi \subseteq \cup_k \psi_k$, $\forall \psi_k \in \Psi$.

Here, each rectangle $\psi_k$ is called an *elementary rectangle* in the rest of this paper.

## V. Covering Algorithm

As aforementioned in Section II, Problem 1 is NP-hard. In this section, we propose a simple yet effective approximation algorithm to tackle this problem with a greedy heuristic: for each polygon $\Omega_i$ in $\Xi$, we find a minimal set of elementary rectangles to cover it.

The overall framework is illustrated in Fig. 5. First, we propose a polygon segmentation method based on the medial axis representation (see Section V-A). For each segmented part, we cover it using a minimal set of elementary rectangles with an optimal orientation (see Section V-C). The complexity of the proposed medial-axis-driven segmentation and covering algorithm (Algorithm 1) is analyzed in Section V-B. We further improve Algorithm 1 by introducing an efficient operation to merge small segmented parts (see Section V-E). The final algorithm (Algorithm 2) has $O(n^2 \log^2 n)$ time complexity and output $O(Kn)$ elementary rectangles, where $K$ is an input-polygon-dependent constant.

### A. Medial-Axis-Driven Segmentation and Covering

Consider a polygonal domain $\Omega$ (multiply connected in general) and its medial axis $M(\Omega)$. Each medial edge $e$ in

3D Systems-Exhibit 1029-Page 4 of 12

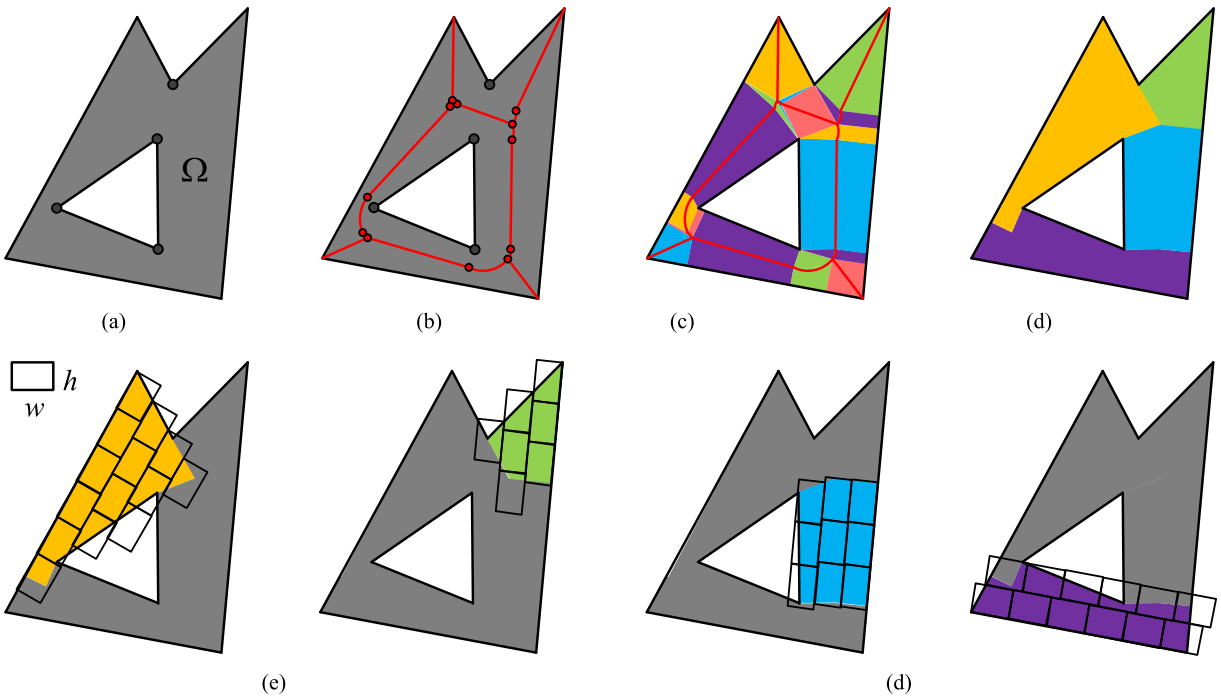



Fig. 5. Framework of our proposed covering method. (a) Input multiply connected polygon $\Omega$ with a hole. (b) Medial axis $M(\Omega)$ of $\Omega$. (c) Segmentation $\{\varpi(e_i) : e_i \in M(\Omega)\}$ of $\Omega$ induced by $M(\Omega)$. (see Section V-A). (d) Merging operation is further applied to merge small segmented parts in $\{\varpi(e_i)\}$. (see Section V-E). (e) For each merged part, a minimal set of elementary rectangles with optimal orientation is computed. (see Sections V-C and V-E).

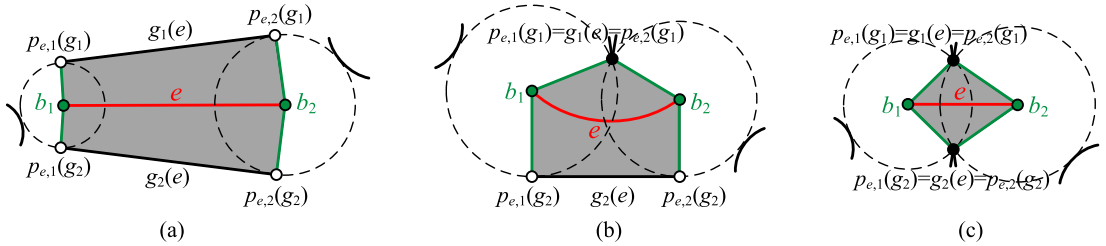



Fig. 6. Enclosed region associated with a medial edge $e$. Each medial edge $e$ in $M(\Omega)$ is delimited by two branch points $b_1$ and $b_2$. Each branch point $b_i$ (green dots) has $|b_i|$ closest boundary points in $\partial\Omega$ in which $p_{e,i}(g_1)$ is in $g_1(e)$ and $p_{e,i}(g_2)$ is in $g_2(e)$. Four line segments $\overline{b_1 p_{e,1}(g_1)}$, $\overline{b_1 p_{e,1}(g_2)}$, $\overline{b_2 p_{e,2}(g_1)}$, and $\overline{b_2 p_{e,2}(g_2)}$ (green lines), together with $g_1$ and $g_2$, enclose a region (shown as the gray shaded area) in $\Omega$. (a) If the generators are two line segments, the enclosed region is a hexagon. (b) If the generators are one line segment and one reflex vertex, the enclosed region is a pentagon. (c) If the generators are two reflex vertices, the enclosed region is a quadrangle.

$\overline{M}(\Omega)$ is delimited by two branch points[1] $b_1$ and $b_2$. Each branch point $b_i$ has $|\Lambda(b_i)|$ closest boundary points in $\partial\Omega$ in which one is in $g_1(e)$ (denoted as $p_{e,i}(g_1)$) and another is in $g_2(e)$ (denoted as $p_{e,i}(g_2)$). We connect these points into four line segments $\overline{b_1 p_{e,1}(g_1)}$, $\overline{b_1 p_{e,1}(g_2)}$, $\overline{b_2 p_{e,2}(g_1)}$, and $\overline{b_2 p_{e,2}(g_2)}$, which enclose a region in $\Omega$ together with $g_1$ and $g_2$. There are three types of enclosed regions.

1) *Type 1 (Hexagon):* If the generators are two open line segments, the enclosed region is a hexagon [see Fig. 6(a)].
2) *Type 2 (Pentagon):* If the generators are one open line segment and one reflex vertex, the enclosed region is a pentagon [see Fig. 6(b)].
3) *Type 3 (Quadrangle):* If the generators are two reflex vertices, the enclosed region is a quadrangle [see Fig. 6(c)].

[1]Here, in addition to the branch points defined in Definition 2, degenerate branch points whose locations are at convex vertices in $\partial\Omega$ are also included. We omit the discussion of degenerate cases here (see [13]–[16]) for some details.

Denote by $\varpi(e)$ the enclosed region associated with a medial edge $e$. The set of enclosed regions $\{\varpi(e_i) : e_i \in M(\Omega)\}$ form a segmentation of $\Omega$.

Our basic idea is to cover each enclosed region $\varpi(e_i)$ by a minimal set of elementary rectangles. The covering methods for three types of enclosed regions are detailed below. In Section V-C, these covering methods are shown to be optimal. The pseudocode of the proposed medial-axis-driven segmentation and covering is presented in Algorithm 1.

*1) Type-1 Region Covering:* Refer to Fig. 7. The enclosed region of Type 1 is a hexagon, in which we consider to cover the trapezoid $\square(p_{e,2}(g_2), p_{e,1}(g_2), b_1, b_2)$ with four corner points $p_{e,2}(g_2)$, $p_{e,1}(g_2)$, $b_1$, and $b_2$ [gray shaded area in Fig. 7 (left)], using rectangles of fixed size $w \times h$. Due to

3D Systems-Exhibit 1029-Page 5 of 12

**Algorithm 1** $M(\Omega)$-Driven Polygon Segmentation and Covering

**Input:** A multiply connected polygonal domain $\Omega$ (with $m \geq 0$ holes) and a projection area of fixed size $w \times h$.

**Output:** A set of rectangles $\Psi = \{\psi_k, k = 1, 2, \cdots\}$, all with a fixed size $w \times h$, such that $\Omega \subseteq \cup_k \psi_k$, $\forall \psi_k \in \Psi$.

Compute the medial axis $M(\Omega)$ of $\Omega$ using the robust algorithm [16].
Segment $\Omega$ by computing the set of enclosed regions $\{\varpi(e_i) : e_i \in M(\Omega)\}$ (Section V-A).
$\Psi = \emptyset$
**for** each enclosed region $\varpi(e_i)$ **do**
  **if** both $g_1(e_i)$ and $g_2(e_i)$ are line segments **then**
    Cover $\varpi(e_i)$ with a set of rectangles $\Psi(e_i)$ using the method of *Type-1 Region Covering*.
    $\Psi = \Psi \cup \Psi(e_i)$.
  **else if** $g_1(e_i)$ and $g_2(e_i)$ are one open line segment and one reflex vertex **then**
    Cover $\varpi(e_i)$ with a set of rectangles $\Psi(e_i)$ using the method of *Type-2 Region Covering*.
    $\Psi = \Psi \cup \Psi(e_i)$.
  **else**
    Cover $\varpi(e_i)$ with a set of rectangles $\Psi(e_i)$ using the method of *Type-3 Region Covering*.
    $\Psi = \Psi \cup \Psi(e_i)$.
  **end if**
**end for**
Output $\Psi$.

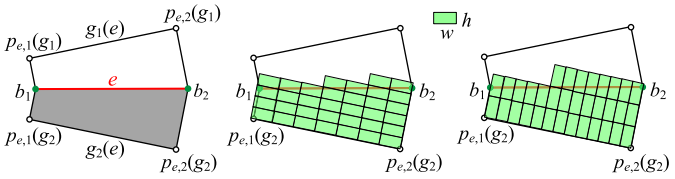



Fig. 7. Cover the enclosed region of Type 1. Both generators $g_1(e)$ and $g_2(e)$ of $e$ are open line segments. $b_1$ and $b_2$ are two branch points of the medial edge $e$. $p_{e,i}(g_j)$ is the closest boundary points of $b_i$ on $g_j(e)$, $i, j = 1, 2$. $w \times h$ is the fixed size of an elementary rectangle. To cover the trapezoid $\square(p_{e,2}(g_2), p_{e,1}(g_2), b_1, b_2)$ (gray shaded area in the left), started at the point $p_{e,2}(g_2)$, along the directions of $p_{e,2}(g_2) \rightarrow b_2$ and $p_{e,2}(g_2) \rightarrow p_{e,1}(g_2)$, elementary rectangles are tiled side by side without overlap, and with two orientations (middle and right). The tiling with a smaller number of rectangles is chosen to use.

axis symmetry, the trapezoid $\square(p_{e,2}(g_1), b_2, b_1, p_{e,1}(g_1))$ can be covered in a similar way.

Let $l(\overline{ab})$ be the length of line segment $\overline{ab}$. Without loss of generality, assume $l(\overline{b_2 p_{e,2}(g_2)}) \geq l(\overline{b_1 p_{e,1}(g_2)})$. Denote by $a \rightarrow b$ be the direction from point $a$ to point $b$. Starting at the point $p_{e,2}(g_2)$, along the directions of $p_{e,2}(g_2) \rightarrow b_2$ and $p_{e,2}(g_2) \rightarrow p_{e,1}(g_2)$, elementary rectangles are tiled side by side without overlap, until the trapezoid $\square(p_{e,2}(g_1), b_2, b_1, p_{e,1}(g_1))$ is fully covered. The tiling of elementary rectangles can have two orientations [see Fig. 7 (middle) and (right)] and we choose the one with a smaller number of elementary rectangles.

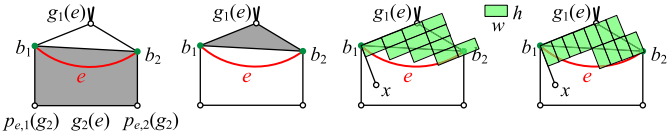



Fig. 8. Cover the enclosed region of Type 2. The generator $g_1(e)$ is a reflex vertex and the generator $g_2(e)$ is an open line segment. $b_1$ and $b_2$ are two branch points of the medial edge $e$. $p_{e,i}(g_2)$ is the closest boundary points of $b_i$ on $g_2(e)$, $i = 1, 2$. $w \times h$ is the fixed size of an elementary rectangle. The trapezoid $\square(p_{e,2}(g_2), p_{e,1}(g_2), b_1, b_2)$ (gray shaded area in the leftmost) is covered in the same way as in Fig. 7. $x$ is a point in the enclosed region such that the direction $b_1 \rightarrow x$ is perpendicular to the direction $b_1 \rightarrow g_1(e)$. To cover the triangle $\triangle(b_1, b_2, g_1(e))$ (gray shaded area in the middle left), started at the point $b_1$, along the directions of $b_1 \rightarrow g_1(e)$ and $b_1 \rightarrow x$, elementary rectangles are tiled side by side without overlap, and with two orientations (middle right and rightmost). Again, the tiling with a smaller number of rectangles is used.

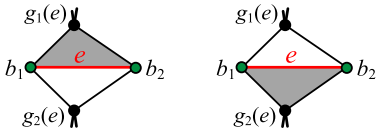



Fig. 9. Cover the enclosed region of Type 3. Both generators $g_1(e)$ and $g_2(e)$ of the medial edge $e$ are reflex vertices. $b_1$ and $b_2$ are two branch points of $e$. The enclosed region of Type-3 is a quadrangle and the medial edge $e$ separates it into two triangles $\triangle(b_1, b_2, g_1(e))$ and $\triangle(b_2, b_1, g_2(e))$. Each of the triangle area can be covered by the same way as in Fig. 8.

*2) Type-2 Region Covering:* Refer to Fig. 8. The enclosed region of Type 2 is a pentagon, in which the trapezoid $\square(p_{e,2}(g_2), p_{e,1}(g_2), b_1, b_2)$ [gray shaded area in Fig. 8 (leftmost)] can be covered in the same way as what is taken for Type-1 region. Now consider to cover the triangle $\triangle(b_1, b_2, g_1(e))$ with three vertices $b_1, b_2$, and $g_1(e)$ [gray shaded area in Fig. 8 (middle left)] using elementary rectangles.

Without loss of generality, assume $l(\overline{b_1 g_1(e)}) \geq l(\overline{b_2 g_1(e)})$. Let $x$ be a point in the enclosed region such that the direction $b_1 \rightarrow x$ is perpendicular to the direction $b_1 \rightarrow g_1(e)$. To cover the triangle $\triangle(b_1, b_2, g_1(e))$ (gray shaded area in the middle left), started at the point $b_1$, along the directions of $b_1 \rightarrow g_1(e)$ and $b_1 \rightarrow x$, elementary rectangles are tiled side by side without overlap, until the triangle $\triangle(b_1, b_2, g_1(e))$ is fully covered. The tiling of elementary rectangles can have two orientations [see Fig. 8 (middle right) and (rightmost)] and akin to the case of Type-1 region covering, we choose the one with a smaller number of elementary rectangles.

*3) Type-3 Region Covering:* Refer to Fig. 9. The enclosed region of Type-3 is a quadrangle and the medial edge $e$ separates it into two triangles $\triangle(b_1, b_2, g_1(e))$ and $\triangle(b_2, b_1, g_2(e))$. Each of the triangle area can be covered in the same way as what is taken above for Type-2 region.

### B. Complexity of Algorithm 1

*Property 1:* Given a multiply connected polygonal domain $\Omega$ with $n$ generators and $m$ holes, Algorithm 1 has $O(n(\log n + m))$ time complexity, and it outputs $O(Kn)$ elementary rectangles, where $K$ is an input-polygon-dependent constant.

3D Systems-Exhibit 1029-Page 6 of 12

*Proof:* Computing the medial axis of a multiply connected domain $\Omega$ with $m$ holes takes $O(n(\log n + m))$ time [23]. Segmentation by computing enclosed regions associated with medial edges takes $O(n)$ time. For each enclosed region, computing the covering of elementary rectangles takes $O(1)$ time. Putting it all together, Algorithm 1 has $O(n(\log n + m))$ time complexity.

Denote by $L$ the largest edge length in all generators of $\partial\Omega$, and by $R$ the radius of the largest inscribed circle in $\Omega$. Below we show that Algorithm 1 outputs $O((R(L+R)/w \times h)n)$ elementary rectangles.

First, consider covering the enclosed region of Type 1. Refer to Fig. 7. We have $l(\overline{b_2 p_{e,2}(g_2)}) \leq R$ and $l(\overline{p_{e,1}(g_2)p_{e,2}(g_2)}) \leq L$. Then, to cover the trapezoid $\square(p_{e,2}(g_2), p_{e,1}(g_2), b_1, b_2)$, the number of elementary rectangles, in two different orientations of tiling, is less than $\lceil (L/w) \rceil \times \lceil (R/h) \rceil$ and $\lceil (R/w) \rceil \times \lceil (L/h) \rceil$, respectively. In both orientations, the number is bounded by $O(RL/wh)$.

Second, consider covering the enclosed region of Type 2. Refer to Fig. 8. Let $l_1$ be the line passing through two points $b_1$ and $g_1(e)$ and $d_1$ be the perpendicular distance from $b_2$ to $l_1$. Since $l(\overline{b_2 g_1(e)}) \leq R$, we have $d_1 \leq R$. Similarly, let $l_2$ be the line passing through two points $b_1$ and $x$ and $d_2$ be the perpendicular distance from $b_2$ to $l_2$. We have $d_2 \leq 2R$. Then, to cover the triangle $\triangle(b_1, b_2, g_1(e))$, the number of elementary rectangles, in two different orientations of tiling, is less than $2\lceil (R/w) \rceil \times \lceil (R/h) \rceil = O(R^2/wh)$.

Third, the case of covering the enclosed region of Type 3 can be analyzed in the same way as in the case of Type 2. Therefore, the same bound $O(R^2/wh)$ is held for the case of Type 3. □

### C. Optimal Orientations of Elementary Rectangles

In this section, we show that the orientation of elementary rectangles is optimal for covering enclosed regions of three types in Algorithm 1.

*Property 2:* If all elementary rectangles in covering the enclosed region $\varpi(e)$ associated with a medial edge $e$ have the same orientation, the orientation computed in Algorithm 1 is optimal.

### D. Sketch of Proof

We only prove the case that the enclosed region is of Type 1. The case with regions of Type 2 and Type 3 can be proved similarly. We redraw the configuration shown in Fig. 7 as Fig. 10, in which we define a planar coordinate system $\{o, x, y\}$ with $o$ being located at the branch point $b_1$ and $x$ being along with the direction $b_1 \rightarrow b_2$. We define the orientation of an elementary rectangle by a tilting angle $\theta_0$ with respect to the $x$-axis. Denote by $\theta$ the angle between the medial edge $e$ and one generator $\overline{p_{e,1}(g_2)p_{e,2}(g_2)}$. Here, we only prove the orientation angle $\theta_0$ in the range $[-\theta, \theta]$ [yellow shaded area in Fig. 10 (left)]. The case $\theta_0 \in [-(\pi/2), -\theta] \cup [\theta, (\pi/2)]$ can be proved in a similar way.

Using the orientation angle $\theta_0$, we can determine a bounding rectangle passing through the points $p_{e,2}(g_2)$, $p_{e,1}(g_2)$, $b_1$, and $b_2$ [gray shaded area in Fig. 10 (right)]. One side of the bounding rectangle has the same orientation angle $\theta_0$ and we denote by $\theta_1$ the angle between this side and the line segment $\overline{p_{e,1}(g_2)p_{e,2}(g_2)}$. Since $0 \leq \theta < (\pi/2)$ and $\theta_1 \in [0, 2\theta]$, we have $0 \leq \theta_1 < \pi$. Let $l_0 = l(\overline{p_{e,1}(g_2)p_{e,2}(g_2)})$. Then, the lengths of two sides of the bounding rectangle are $l_1 = r_2 \cos\theta_1 + l_0 \sin\theta_1$ and $l_2 = r_1 \sin\theta_1 + l_0 \cos\theta_1$, respectively. In the worst case, $\lceil (l_1/w) \rceil \lceil (l_2/h) \rceil$ or $\lceil (l_2/w) \rceil \lceil (l_1/w) \rceil$ elementary rectangles are required to cover the trapezoid $\square(p_{e,2}(g_2), p_{e,1}(g_2), b_1, b_2)$. The optimal orientation angle is then to minimize the function

$$f(\theta_1) = (r_2 \cos\theta_1 + l_0 \sin\theta_1)(r_1 \sin\theta_1 + l_0 \cos\theta_1).$$

We have

$$f(\theta_1) = l_0 r_2 + l_0(r_1 - r_2)\sin^2\theta_1.$$

Since $r_1 \geq r_2$ and $0 \leq \theta_1 < \pi$, $f(\theta_1)$ reaches the minimum when $\theta_1 = 0$, which is the orientation angle specified in Algorithm 1.

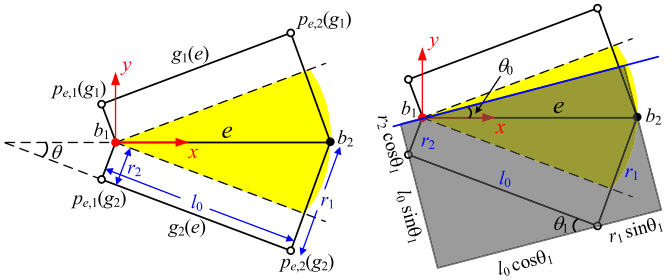



Fig. 10. Geometry configuration of the enclosed region of Type 1 (see Fig. 7). Planar coordinate system $\{o, x, y\}$ is setup (left). $\theta$ is the angle between the medial edge $e$ and one generator $\overline{p_{e,1}(g_2)p_{e,2}(g_2)}$. The length $l(\overline{p_{e,1}(g_2)p_{e,2}(g_2)})$ is $l_0$, and $l(\overline{b_2 p_{e,2}(g_2)}) = r_1$ and $l(\overline{b_1 p_{e,1}(g_2)}) = r_2$. The orientation of an elementary rectangle is defined by a tilting angle $\theta_0$ with respect to the $x$-axis. Yellow shaded area shows the region in which the orientation angle $\theta_0$ is in the range $[-\theta, \theta]$. Given an arbitrary orientation angle $\theta_0 \in [-\theta, \theta]$ (the blue line), a bounding rectangle passing through the points $p_{e,2}(g_2)$, $p_{e,1}(g_2)$, $b_1$, and $b_2$ can be determined (gray shaded area) (right). This bounding rectangle has two orthogonal sides, in which one side has the same orientation angle $\theta_0$; $\theta_1$ is the angle between this side and the line segment $\overline{p_{e,1}(g_2)p_{e,2}(g_2)}$.

### E. Covering Algorithm with Merging

In practice, some enclosed regions in our medial-axis-driven segmentation $\{\varpi(e_i) : e_i \in M(\Omega)\}$ can be very small [see Fig. 5(c)]. In this section, we propose an enhanced algorithm that recursively merges two adjacent enclosed regions until the number of elementary rectangles does not decreased anymore. Its pseudocode is presented in Algorithm 2.

Note that the medial axis $M(\Omega)$ is a connected planar graph with $O(n)$ branch points and $O(n)$ medial edges [5], where $n$ is the number of generators in $\partial\Omega$. For each medial edge $e_i$ in $M(\Omega)$, we try to merge it with one of its adjacent medial edges $e_j$ using the method presented below. The merging operation is iteratively performed until none of edges can be merged any more. Since the merging process frequently inquires adjacent edges information, $M(\Omega)$ is represented by a variant[2] of doubly connected edge list [7] in Algorithm 2.

[2]The original doubly connected edge list is used to represent a planar subdivision. In our application, we only maintain records for medial edges and branch points, without the face information.

3D Systems-Exhibit 1029-Page 7 of 12

**Algorithm 2** $M(\Omega)$-Driven Polygon Covering With Merging Operation

**Input:** A multiply connected polygonal domain $\Omega$ (with $m \geq 0$ holes) and a projection area of fixed size $w \times h$.

**Output:** A minimal set of rectangles $\Psi = \{\psi_k, k = 1, 2, \cdots\}$, all with a fixed size $w \times h$, such that $\Omega \subseteq \cup_k \psi_k$, $\forall \psi_k \in \Psi$.

1: Compute the medial axis $M(\Omega)$ and store it in a doubly-connected edge list $\mathcal{E}$.
2: Assign a number $n(e) = 0$ to each medial edge $e$ in $M(\Omega)$.
3: Segment $\Omega$ by computing the set of enclosed regions $\{\varpi(e_i) : e_i \in M(\Omega)\}$ (Section V-A).
4: $\Psi = \emptyset$
5: **for** each enclosed region $\varpi(e_i)$ **do**
6: Cover $\varpi(e_i)$ with a set of rectangles $\Psi(e_i)$ using the code in lines 5-14 in Algorithm 1.
7: $\Psi = \Psi \cup \Psi(e_i)$.
8: $n(e_i) = |\Psi(e_i)|$.
9: **end for**
10: Set a Boolean variable $flag = TRUE$.
11: **while** $flag == TRUE$ **do**
12: $flag = FALSE$.
13: **for** each edge $e_i$ in $\mathcal{E}$ **do**
14: **for** each adjacent edge $e_j$ of $e_i$ **do**
15: Compute the bounding box $Box(e_i, e_j)$ of $\varpi(e_i) \cup \varpi(e_j)$ with minimal area.
16: Apply the covering method in Section V-E to the region $\varpi(e_i) \cup \varpi(e_j)$ with $Box(e_i, e_j)$, for computing the set of elementary rectangles $\Psi_{temp}$ that covers $\varpi(e_i) \cup \varpi(e_j)$.
17: $n' = |\Psi_{temp}|$.
18: **if** $n' \leq n(e_i) + n(e_j)$ **then**
19: Generate an artificial edge $e'$.
20: $\varpi(e') = \varpi(e_i) \cup \varpi(e_j)$ and $n(e') = n'$.
21: $\Psi = (\Psi \cup \Psi_{temp}) \setminus (\Psi(e_i) \cup \Psi(e_j))$
22: Update $\mathcal{E}$ by replacing $e_i$ and $e_j$ with $e'$.
23: $flag = TRUE$.
24: **end if**
25: **end for**
26: **end for**
27: **end while**
28: Output $\Psi$.

*1) Merging Operation:* Let $n(e_k)$ be the number of elementary rectangles that cover $\varpi(e_k)$ by the methods in Section V-A.

The merging operation has three steps.

- *Step 1:* The bounding box $Box(e_i, e_j)$ of $\varpi(e_i) \cup \varpi(e_j)$ with minimal area is computed by the rotating calipers algorithm [22].
- *Step 2:* The covering method for the region $\varpi(e_i) \cup \varpi(e_j)$ in $Box(e_i, e_j)$ is applied – details can be found below.
- *Step 3:* If the number $n'$ of elementary rectangles output from the covering method in Step 2 is not larger than $n(e_i) + n(e_j)$, this merging is performed and the adjacency-list representation is updated by replacing both $e_i$ and $e_j$ with an abstract edge $e'$, with $\varpi(e') = \varpi(e_i) \cup \varpi(e_j)$ and $n(e') = n'$.

Since the rotating calipers algorithm takes $O(n \log n)$ time and the covering in bounding-box method below takes $O(n \log n)$ time, The time complexity of the merging operation is $O(n \log n)$.

*2) Cover a Region in Bounding Box:* At Step 2 in the merging operation, an arbitrary region $P = \varpi(e_i) \cup \varpi(e_j)$ with a bounding box of minimal area is needed to be covered by a minimal set of elementary rectangles. This task is fulfilled by the following method.

Denote by $o_i$, $i = 1, 2, 3, 4$, the four corner points of the bounding box, and denote by $e_1$ and $e_4$ the two adjacent sides of $o_1$ [see Fig. 11(a)]. For each side of the bounding box, we use line segments with length $w$ (or $h$) to cover it as illustrated in Fig. 11(b). In our implementation, both $w$ and $h$ are tried, and the one results in a smaller number of elementary rectangles is selected to use.

Without loss of generality, when $e_4$ is selected to be covered by line segments with length $h$ [see Fig. 11(b)], slabs are formed by adding lines perpendicular to the line segments (denoted by $l_l$ and $l_u$ for lower and upper ones). The line segment and the slab are denoted by $s$ and $S$, respectively [see Fig. 11(c)]. The following steps are then used to cover $S \cap P$ by a minimum number of rectangles $\{\psi_i\}$.

- *Step 1:* Denote the set of all polygonal vertices falling into $S$ as $\mathcal{V}(S)$.
- *Step 2:* Compute the intersection points between $l_l$, $l_u$ and $P$ and denote them as $\mathcal{I}_l$ and $\mathcal{I}_u$.
- *Step 3:* Sort all points in $\mathcal{I} = \mathcal{I}_l \cup \mathcal{I}_u \cup \mathcal{V}(S)$ in the ascending order of coordinate values along the direction of line $l_l$.
- *Step 4:* If $\mathcal{I}$ is empty, stop the algorithm; otherwise, pop up a point $pt$ from $\mathcal{I}$ (with the minimum coordinate) and remove it from $I = I \setminus pt$.
- *Step 5:* Place a rectangle $\psi$ in $S$ by aligning its lower-left corner as $o = pt$ and its edges as $\overline{e}_1 \subset l_l$, $\overline{e}_3 \subset l_u$, where $o = \overline{e}_1 \cap \overline{e}_4$ and $\overline{e}_i$ $(i = 1, 2, 3, 4)$ are four edges of $\psi$.
  - Step *5.1:* If $\overline{e}_2$ intersects $P$, locate a rectangle $\psi$ at the point $\overline{e}_1 \cap \overline{e}_2$ by the same alignment method in Step 5.
  - *Step 5.2:* Remove all the points in $\mathcal{I}$ falling into this $\psi$.
  - *Step 5.3:* Repeat the above steps until $\overline{e}_2$ does not intersect $P$ anymore.
- Step 6: Go back to Step 4.

An example is shown in Fig. 11(d) and (e) to illustrate the covering results in slabs $S$ and $S'$.

The intersection points between every boundary line of slabs and the polygon can be efficiently computed in $O(n)$ time by finding the intersection of a set of parallel line segments and the polygon. Therefore, the set of elementary rectangles covering $P$ can be determined in $O(n \log n)$ time.

*3) Complexity of Algorithm 2:* There are at most $O(n \log n)$ merging operations and each merging operation takes $O(n \log n)$ time. Then, all merging operations in

3D Systems-Exhibit 1029-Page 8 of 12

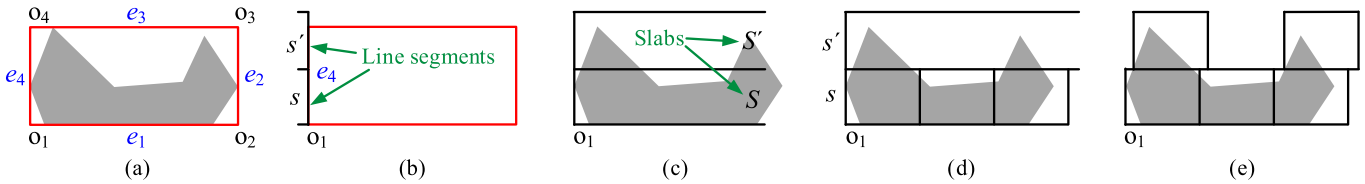



Fig. 11. Steps for covering an arbitrary region $P$ (gray shaded area) in a bounding box. (a) Geometry configuration of the bounding box. (b) Cover the box edge $e_4$ by two line segments $s$ and $s'$ in an end-to-end manner. (c) Each line segment $s$ (or $s'$) defines a slab $S$ (or $S'$). (d) Cover the slab $S \cap P$ by a minimum set of rectangles $\{\psi_i\}$. (f) Cover the $S' \cap P$.

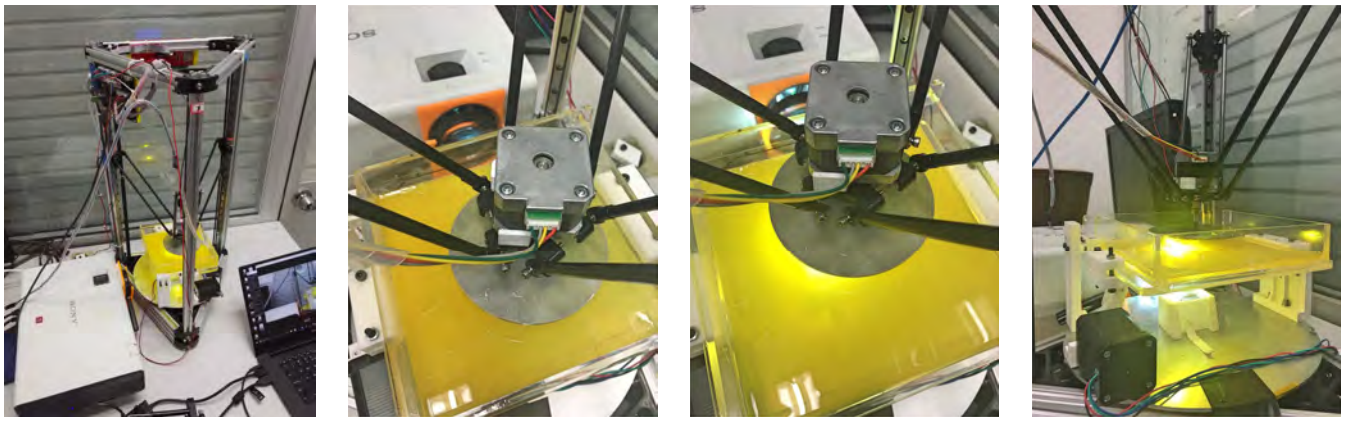


Fig. 12. Photographs to illustrate different stages of fabrication using our delta DLP 3-D printer with the tilting device. More details can be found in accompanying demo video.

TABLE I
STATISTIC OF FABRICATION

| Model | Kitten | | Bunny | | Crocodile | | Cup | | Stairway | | Boat | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| #Triangles | 2,470 | | 10,996 | | 6,906 | | 10,598 | | 9,842 | | 6,966 | |
| Size | ×1.0 | ×2.0 | ×1.0 | ×2.0 | ×1.0 | ×2.0 | ×1.0 | ×2.0 | ×1.0 | ×2.0 | ×1.0 | ×2.0 |
| #Layers | 458 | 916 | 299 | 499 | 291 | 581 | 166 | 331 | 364 | 728 | 88 | 175 |
| Height ($mm$) | 46.1 | 91.9 | 30.2 | 50.2 | 29.4 | 58.4 | 16.9 | 33.4 | 36.7 | 73.1 | 9.1 | 17.8 |
| Total Time | 115 min. | 574 min. | 76 min. | 283 min. | 74 min. | 384 min. | 42 min. | 334 min. | 92 min. | 182 min. | 23 min. | 76 min. |

Algorithm 2 take $O(n^2 \log^2 n)$. Meanwhile, each updating of $\mathcal{E}$ by replacing two edges $e_i$ and $e_j$ with a new edge $e'$ takes $O(n)$ time. There are at most $O(n \log n)$ updating that take $O(n^2 \log n)$ time. On the other hand, since the merging operation is performed only if the new number $n'$ of elementary rectangles is not larger than the current number $n(e_i) + n(e_j)$ (line 18 in Algorithm 2), the number of elementary rectangles output from Algorithm 2 is still bounded by $O(Kn)$. Therefore, we have the following result.

*Property 3:* Given a multiply connected polygonal domain $\Omega$ with $n$ generators and $m$ holes, Algorithm 2 has $O(n^2 \log^2 n)$ time complexity, and it outputs $O(Kn)$ elementary rectangles, where $K$ is an input-polygon-dependent constant.

## VI. EXPERIMENTS AND DISCUSSION

We built a physical prototype system as proposed in Section III (see Fig. 12) and implement the algorithm proposed in Section V on this hardware system. In our practice, a printed 3-D model usually has hundreds to thousands of layers and the polygon soup in each layer has a few disconnected polygons with tens to hundreds of vertices. The running time of our algorithm is from tens of seconds to a few minutes for generating motion paths of the printer's platform on a PC with an Intel I7-860 CPU (2.80 GHz) and 8-GB RAM. Furthermore, models in different sizes are fabricated to verify the performance of our delta DLP 3-D printer (see Fig. 13). When the models are small, each of their slices can be completed covered by the region of a single projection $\psi$. There is no need to apply the segmentation and covering algorithm. Such models are fabricated by our system without applying horizontal movement. We then scale these models by 2.0 in all axes. As a result, the models become larger and can only be fabricated by applying decomposition and horizontal movement.

Statistic of fabrication is listed in Table I. Note that the thickness of each layer in fabrication is 0.1 mm, and the solidification time of each projection is 10 s. The speed of rotation by 42BYGH33 motor is 0.3 rad/s, and the maximal speed for translation is 80 mm/s. It is easy to find that our method has a very good scalability—when increasing the volume of solid to be fabricated by 8×, the total fabrication

3D Systems-Exhibit 1029-Page 9 of 12

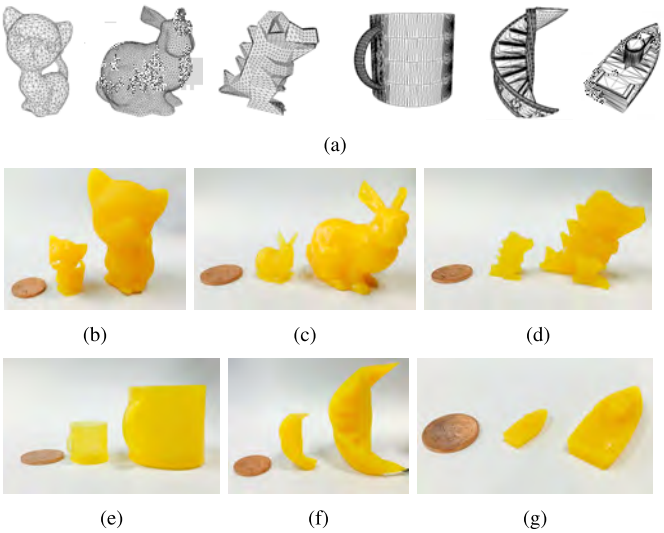



Fig. 13. Results of fabrication—all are fabricated by our system, where the smaller ones are fabricated by single projection for each layer as their cross sections can be fully covered by the region of one projection $\psi$. The bigger models are made by multiprojections for each layer. (a) 3-D digital models. (b) Kitten. (c) Bunny. (d) Crocodile. (e) Cup. (f) Stairway. (g) Boat.

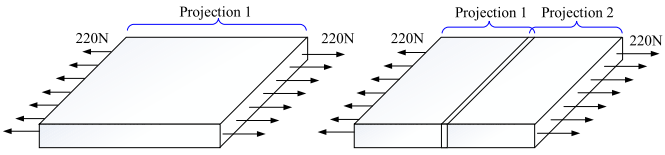



Fig. 14. Two methods for fabricating samples to verify the mechanical strength of models. a) one layer one projection and b) one layer by two projections—half for each. These two different samples are undergoing the same tensile stretch test with the load force up to 220 N (see Fig. 15).

time only increases 4.99×, 3.72×, 5.19×, 7.95×, 1.98× and 3.30×, respectively. Fig. 12 shows the photographs of our system taken during the process of fabrication.

### *A. Mechanical Strength by Multiple Projections*

To verify the mechanical strength of a model fabricated by decomposing a large layer into smaller pieces in our delta DLP 3-D printing, we fabricate models with the same area of cross sections (i.e., 32 mm × 10 mm) whose thickness is 0.5 *mm*. The models are made by two different methods: 1) each layer by one complete projection [see Fig. 14 (left)] and 2) each layer is decomposed into two projections—half for each [see Fig. 14 (right)].

The fabricated specimens have their tensile stiffness tested on an INSTRON 5943 universal testing system (see Fig. 15). To overcome the sliding friction, two small folded pieces of 240 grit sandpaper are placed near the short sides of samples. The testing results show that the samples made by two different methods (i.e., by multi and single projections) are intact when the load force exceeds 220 N (the range of force sensor is 0-250 N). These samples had almost the same yield strength and offset yield strength (see Fig. 16). In these tests, similar mechanical stiffness is observed on the models fabricated by the above mentioned methods. In other

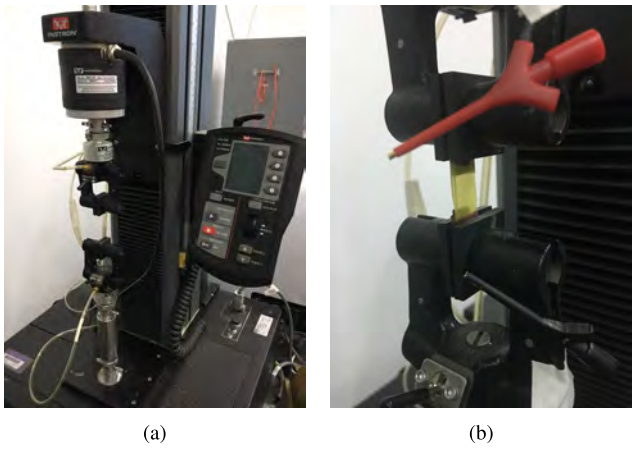



Fig. 15. INSTRON 5943 universal testing system used in our tests for tensile stiffness. (a) INSTRON 5943 system. (b) Platform with tested sample.

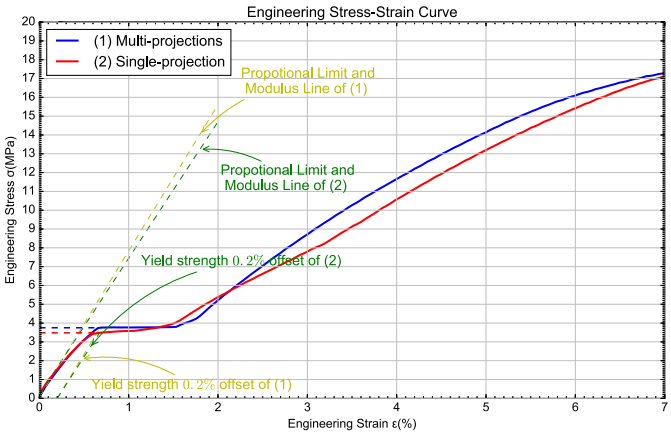



Fig. 16. Tensile stretch testing results using INSTRON 5943 universal testing system. The results show that the specimen made by two different methods (i.e., by multi and single projections) have almost the same yield strength and offset yield strength. Yield strength is strength at which metal or alloy show significant amount of plastic deformation. Offset yield strength is that strength at which 0.2% plastic deformation takes place for those materials that do not exhibit a yield point, such as plastic material.

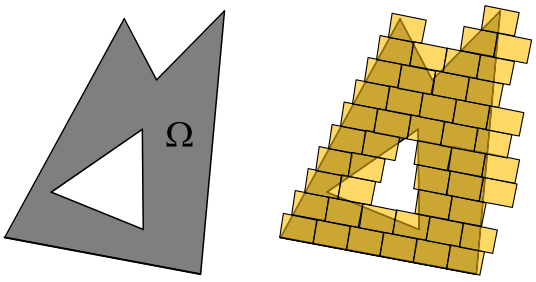



Fig. 17. 48 elementary rectangles (right) are found by the method [26] to cover the multiply connected polygon Ω (left). As a comparison, only 42 elementary rectangles are used by our proposed method [see Fig. 5(e)].

words, no significant weakness is found on the models made by decomposing a large cross section into smaller ones to be solidified.

The models fabricated by our delta DLP printer can satisfy the requirement of general usage. We further note that when

3D Systems-Exhibit 1029-Page 10 of 12

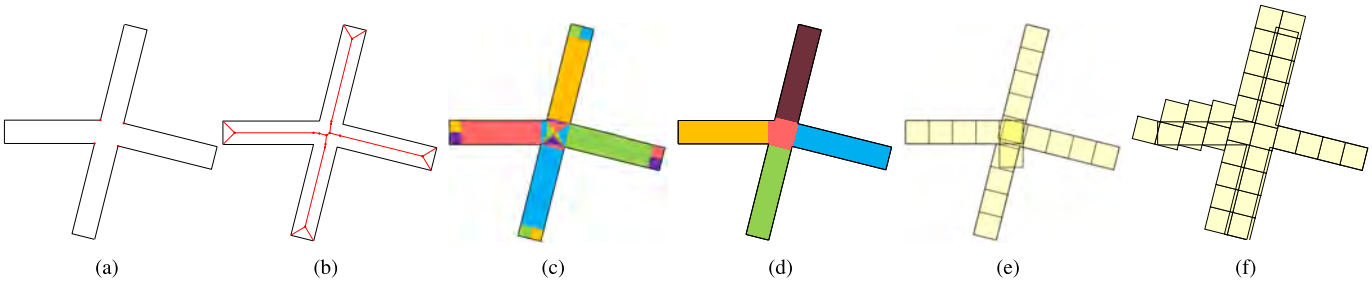



Fig. 18. One example region to be covered using our method and the previous method [26] . (a) Input polygon $\Omega$. (b) Medial axis $M(\Omega)$. (c) Medial-axis-driven segmentation of $\Omega$. (d) Segmentation after applying the merging operation. (e) Covering result of our method with 20 elementary rectangles. (f) Covering result of the previous method [26] needs 31 elementary rectangles in total.

being applied to a practical model (see Fig. 13), the boundaries in different slabs (in $x$-$y$ planes) and in different slices (along the $z$-axis) are stagger from each other. This can further enhance the mechanical stiffness of fabricated models.

### *B. Comparison with Previous Work*

A bounding-box-based covering method was proposed in a previous work [26]. This method covers a multiply connected polygon $\Omega$ using a set of nonoverlapped elementary rectangles with the same orientation. To find an optimal orientation, this method computes the numbers of elementary rectangles at some discrete orientations and build an interpolating function $f(\theta)$ using radial basis functions. Then, the optimal orientation is set to be angle $\widehat{\theta}$ that minimizes the function $f(\theta)$. This optimization strategy is only heuristic and does not have a theoretic foundation.

In this paper, we propose a medial-axis-driven segmentation and covering method. For each segmented part, the orientation of covering elementary rectangles is optimized using a local geometric analysis (see Section V-C). Furthermore, our method has a smaller time complexity (i.e., $O(n^2 \log^2 n)$) and guarantees to output $O(Kn)$ the number of elementary rectangles, where $K$ is an input-polygon-dependent constant. Differently, the previous method has $O(n^3 \log n)$ time complexity and does not have any control on the output number of elementary rectangles.

In addition to better complexity, our method can find a smaller number of covering elementary rectangles than [26] also in practice. For the multiply connected polygon shown in Fig. 5(a), our method finds 42 covering elementary rectangles [see Fig. 5(e)]. As a comparison, the previous method [26] needs 48 rectangles to cover the same region (see Fig. 17). One more example is shown in Fig. 18. In this example, the previous method [26] uses 31 elementary rectangles to cover an input region while our new method needs only 20 rectangles.

## VII. Conclusion

In this paper, we present a system for DLP-based 3-D printing of large models. The hardware of our system is based on an extension of parallel delta robot and a conventional DLP printer together with a tilting device. The major technical contribution of our work is an approach to move working platform horizontally when the area of a layer to be solidified is larger than the maximal region that can be produced by a single projection. A simple yet effective medial-axis-driven segmentation and covering algorithm is developed to decompose the large area of a layer into the small number of projected regions. The mechanical strength of models fabricated by decomposition has been studied and compared with models resulted from conventional DLP process, where similar stiffness can be found. The functionality of our system has been verified by fabricating freeform models in different sizes and a good scalability of our approach is observed.

## References

[1] *Tiling an Orthogonal Polygon With Squares is NP Hard*. Accessed: 2017. [Online]. Available: http://cs.stackexchange.com/questions/16661/tiling-an-orthogonal-polygon-with-squares/16801#16801

[2] *Full Spectrum Laser*. Accessed: 2017. [Online]. Available: http://www.fslaser.com/.

[3] L. J. Aupperle, H. E. Conn, J. M. Keil, and J. O'Rourke, "Covering orthogonal polygons with squares," in *Proc. 26th Allerton Conf. Commun. Control Comput.*, 1988, pp. 97–106.

[4] C. Bell, *3D Printing With Delta Printers*. New York, NY, USA: Apress, 2015.

[5] H. I. Choi, S. W. Choi, and H. P. Moon, "Mathematical theory of medial axis transform," *Pacific J. Math.*, vol. 181, no. 1, pp. 57–88, 1997.

[6] J. C. Culberson and R. A. Reckhow, "Covering polygons is hard," *J. Algorithms*, vol. 17, no. 1, pp. 2–44, 1994.

[7] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars, *Computational Geometry: Algorithms and Applications*. 3rd ed. Berlin, Germany: Springer-Verlag, 2008.

[8] I. Gibson, D. W. Rosen, and B. Stucker, *Additive Manufacturing Technologies*. New York, NY, USA: Springer, 2010.

[9] A. Hegedüs, "Algorithms for covering polygons by rectangles," *Comput.-Aided Des.*, vol. 14, no. 5, pp. 257–260, 1982.

[10] J. M. Keil, "Polygon decomposition," in *Handbook of Computational Geometry*. Amsterdam, The Netherlands: Elsevier, 2000, pp. 491–518.

[11] C. Levcopoulos and J. Gudmundsson, "Approximation algorithms for covering polygons with squares and similar problems," in *Randomization and Approximation Techniques in Computer Science* (Lecture Notes in Computer Science), vol. 1269. Berlin, Germany: Springer-Verlag, 1997, pp. 27–41.

[12] A. Lingas, "The power of non-rectilinear holes," in *Proc. 9th Int. Colloq. Auto. Lang. Program.*, vol. 140. 1982, pp. 369–383.

[13] Y.-J. Liu, "Semi-continuity of skeletons in two-manifold and discrete Voronoi approximation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1938–1944, Sep. 2015.

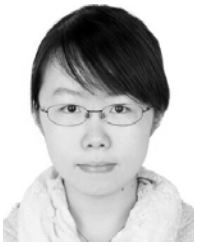
[14] Y.-J. Liu, Z. Chen, and K. Tang, "Construction of iso-contours, bisectors, and Voronoi diagrams on triangulated surfaces," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1502–1517, Aug. 2011.

[15] Y.-J. Liu, D. Fan, C.-X. Xu, and Y. He, "Constructing intrinsic Delaunay triangulations from the dual of geodesic Voronoi diagrams," *ACM Trans. Graph.*, vol. 36, no. 2, pp. 15:1–15:15, 2017.

[16] Y.-J. Liu, C.-C. Yu, M.-J. Yu, K. Tang, and D.-S. Kim, "A robust divide and conquer algorithm for progressive medial axes of planar shapes," *IEEE Trans. Vis. Comput. Graphics*, vol. 22, no. 12, pp. 2522–2536, Dec. 2016.

**3D Systems-Exhibit 1029-Page 11 of 12**

[17] H. Luan and Q. Huang, "Prescriptive modeling and compensation of in-plane shape deformation for 3-D printed freeform products," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 1, pp. 73–82, Jan. 2017.
[18] A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu, *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*, 2nd ed. Hoboken, NJ, USA: Wiley, 2000.
[19] J. O'Rourke and K. J. Supowit, "Some NP-hard polygon decomposition problems," *IEEE Trans. Inf. Theory*, vol. IT-29, no. 2, pp. 181–190, Mar. 1983.
[20] Y. Pan, Y. Chen, and C. Zhou, "Fabrication of smooth surfaces based on mask projection stereolithography," in *Proc. Solid Freeform Fabrication Symp.*, 2011, pp. 263–278.
[21] Y. Pan, Y. Chen, and C. Zhou, "Fast recoating methods for the projection-based stereolithography process in micro- and macro-scales," in *Proc. Solid Freeform Fabrication Symp.*, 2012, pp. 846–862.
[22] F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction*. New York, NY, USA: Springer-Verlag, 1985.
[23] V. Srinivasan and L. R. Nackman, "Voronoi diagram for multiply-connected polygonal domains I: Algorithm," *IBM J. Res. Dev.*, vol. 31, no. 3, pp. 361–372, May 1987.
[24] Y. Tang, G. Dong, Q. Zhou, and Y. F. Zhao, "Lattice structure design and optimization with additive manufacturing constraints," *IEEE Trans. Autom. Sci. Eng.*, to be published. [Online]. Available: http://ieeexplore.ieee.org/document/7913670/, doi: 10.1109/TASE.2017.2685643.
[25] A. Wang, S. Song, Q. Huang, and F. Tsung, "In-plane shape-deviation modeling and compensation for fused deposition modeling processes," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 2, pp. 968–976, Apr. 2017.
[26] C. Wu, R. Yi, Y.-J. Liu, Y. He, and C. C. L. Wang, "Delta DLP 3D printing with large size," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 2155–2160.
[27] C. Zhou and Y. Chen, "Additive manufacturing based on optimized mask video projection for improved accuracy and resolution," *J. Manuf. Process.*, vol. 14, no. 2, pp. 107–118, 2012.
[28] C. Zhou, Y. Chen, and R. A. Waltz, "Optimized mask image projection for solid freeform fabrication," *J. Manuf. Sci. Eng.*, vol. 131, no. 6, pp. 061004-1–061004-12, 2009.



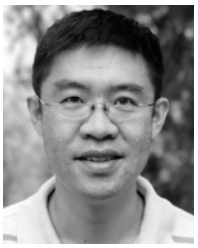
**Ran Yi** received the B.Eng. degree from Tsinghua University, Beijing, China, in 2016, where she is currently pursuing the Ph.D. degree with the Tsinghua National Laboratory for Information Science and Technology, Department of Computer Science and Technology.

Her current research interests include 3-D printing, computational geometry, and computer graphics.



**Chenming Wu** received the B.Eng. degree from the Beijing University of Technology, Beijing, China, in 2015. He is currently pursuing the master's degree with the Tsinghua National Laboratory for Information Science and Technology, Department of Computer Science and Technology, Tsinghua University, Beijing.

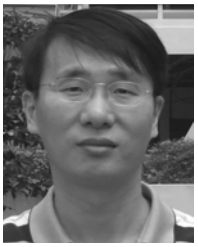
His current research interests include 3-D printing, computational geometry, and computer graphics.



**Yong-Jin Liu** (M'13–SM'16) received the B.Eng. degree from Tianjin University, Tianjin, China, in 1998, and the Ph.D. degree from the Hong Kong University of Science and Technology, Hong Kong, in 2004.

He is currently an Associate Professor with the Tsinghua National Laboratory for Information Science and Technology, Department of Computer Science and Technology, Tsinghua University, Beijing, China. His current research interests include computational geometry, computer graphics, and computer-aided design.

Dr. Liu is a member of the Association for Computing Machinery (ACM).



**Ying He** received the bachelor's and master's degrees in electrical engineering from Tsinghua University, Beijing, China, in 1997 and 2000, respectively, and the Ph.D. degree in computer science from Stony Brook University, Stony Brook, NY, USA, in 2006.

He is an Associate Professor with the School of Computer Engineering, Nanyang Technological University, Singapore. He is interested in the problems that require geometric computing and analysis.



**Charlie C. L. Wang** (SM'–) received the B.Eng. degree in mechatronics engineering from the Huazhong University of Science and Technology, Wuhan, China, and the M.Phil. and Ph.D. degrees in mechanical engineering from the Hong Kong University of Science and Technology, Hong Kong.

He is currently a Professor and Chair of Advanced Manufacturing, Department of Design Engineering, Delft University of Technology, Delft, The Netherlands.

Dr. Wang is a Fellow of the American Society of Mechanical Engineers.

3D Systems-Exhibit 1029-Page 12 of 12