2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Daejeon Convention Center
October 9-14, 2016, Daejeon, Korea

Delta DLP 3D Printing with Large Size

Chenming Wu[1*], Ran Yi[1*], Yong-Jin Liu[1†], Ying He[2] and Charlie C.L. Wang[3]

***Abstract*— We present a delta DLP 3D printer with large size in this paper. Compared with traditional DLP 3D printers that use a low-cost off-the-shelf consumer projector and a single vertical carriage, the platform of our delta DLP 3D printer can also move horizontally in the plane. We show that this structure allows the printer to have a larger printing area than the projection area of a projector. Our system can print 3D models much larger than traditional DLP 3D printers. The major challenge to realize delta 3D printing with large size comes from how to partition an arbitrary planar polygonal shape (possibly with holes or multiple disjoint polygons) into 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 time complexity of our algorithm is $O(n^3 \log n)$, where n is the number of edges in the polygonal shape. A physical prototype system is built and several large 3D models with complex geometric structures have been printed as examples to demonstrate the effectiveness of our approach.**

I. INTRODUCTION

Digital Light Processing (DLP) is a technology in 3D printing that use UV-light to solidify liquid photopolymer [1]. The technique of DLP is widely employed in 3D printing because of its fast printing speed and its simple mechanism in hardware. When preparing the information for fabrication, a 3D CAD model is first sliced by a set of parallel planes and each slice is later converted into a 2D mask image. By projecting the mask image onto a photocurable liquid surface, a layer of solid in the same shape can be formed. A 3D 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 3D printer is much faster than other point (or line) based 3D printing techniques.

Most photopolymers react to radiation in the *ultraviolet* (UV) wavelength ranges. To successfully solidify fluid photopolymers, sufficient light intensity must be projected onto the surface of liquid tank in DLP-based 3D 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 3D Printer [2]). To ensure the light intensity and reduce the size of a machine, optimal lenses are placed between the project and the working surface to shorten the distance of projection. An illustration of DLP-based 3D printing can be found in Fig.1. However, this results in a very small working area (e.g., only $34mm \times 34mm$ in our hardware setup when using a SONY VPL-EW246 projector).

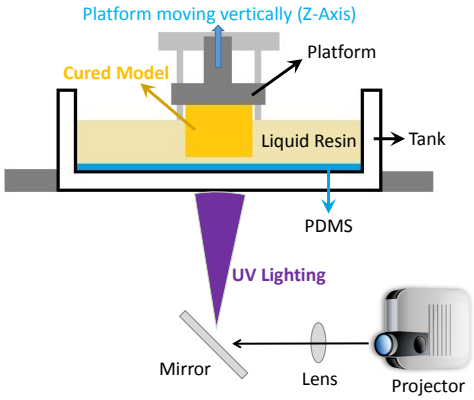


Fig. 1. Working principle of a conventional DLP 3D printer, in which the platform can only move vertically.

The basic idea of our work is to make multiple projections for each layer of the 3D model to fabricate models with larger sizes. We develop a working system to realize this function of DLP-based 3D printing with 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 (see details 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. As discussed in Section II, this problem is generally NP-hard. We present a simple yet effective algorithm in Section V to cover a polygonal shape (possibly with holes and disjoint components) using limited number of rectangles with fixed sizes (i.e., the maximal area of projection). This is the major contribution of our work.

II. RELATED WORK

A DLP-based 3D printing approach called *mask-image-projection-based stereolithograhy* (MIP-SL) was proposed in [3], [4], which follows the SLA technique but replace

*This work was supported by the Natural Science Foundation of China (61322206, 61432003, 61521002, 61661130156) and Royal Society-Newton Advanced Fellowship.

[1]C. Wu, R. Yi and Y-J Liu are with the TNList, Department of Computer Science and Technology, Tsinghua University, Beijing, China liuyongjin@tsinghua.edu.cn

[2]Y. He is with School of Computer Engineering, Nanyang Technological University, Singapore. YHe@ntu.edu.sg

[3]C.C.L. Wang is with the Department of Design Engineering and TU Delft Robotics Institute, Delft University of Technology, Netherlands. c.c.wang@tudelft.nl

*C. Wu and R. Yi contributed equally to this paper

†Corresponding author

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 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.

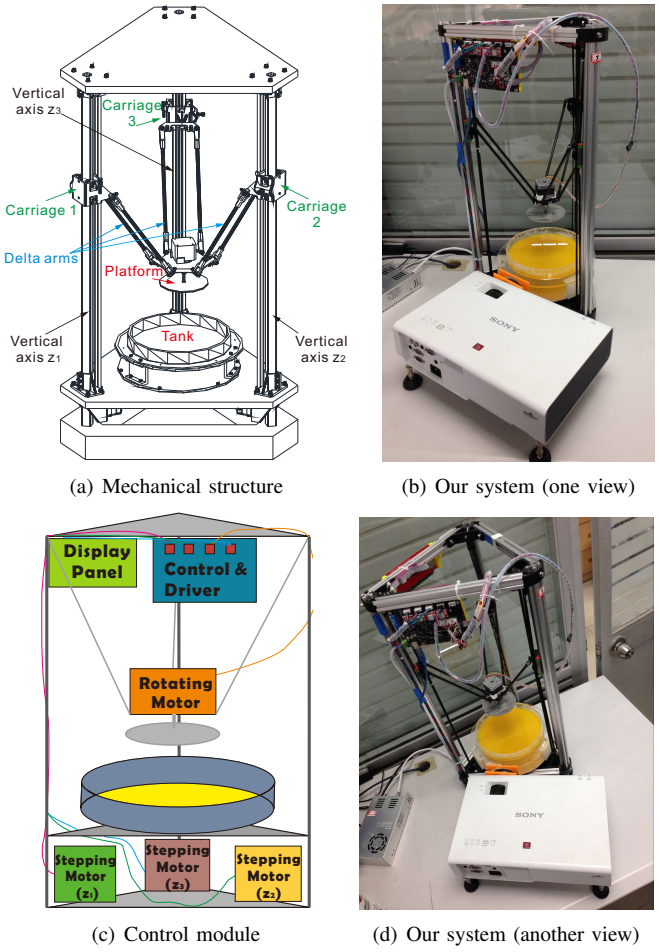
In this paper, we present a new DLP system that allows the platform to move not only vertically (in z-axis) but also horizontally (in x-y plane). Then, larger layers of 3D models can be printed by multiple projections of a consumer projector. To decompose a planar shape into an optimal set of rectangles, two types of decompositions have been considered [5]: *partitions* and *coverings*. It is called a *partition* if a shape is decomposed into non-overlapped sub-regions the union of which is exactly equal to the target polygon. If the sub-regions 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 (ref. [6], [7]). Even for the special problem of covering a rectilinear polygon with squares, finding a minimum of such covering is also NP-hard [8]. Many practical approximation algorithms have been proposed in the field of VLSI chip design (e.g., [9], [10]). 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 sub-polygons is NP-hard [12]. The special problem of partitioning a rectilinear polygon to a minimum number of squares is also NP-hard [13]. Most existing practical approximation algorithms to solve partition problems only deal with rectilinear/orthogonal polygons (e.g., [5]). 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 3D printer is in fact a parallel robot [14] and has been used in 3D printing by *fused deposition modeling* (FDM). To the best of our knowledge, the delta structure has not been used in DLP 3D printing yet.



(a) Mechanical structure (b) Our system (one view)

(c) Control module (d) Our system (another view)

Fig. 2. The mechanical structure, control module and our prototype of a delta DLP 3D printer. Vector graphics in (a) and (c) are provided for zoom-in examination.

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 [14].

The control module of a delta structure consists of an Arduino chip, a 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 drived 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

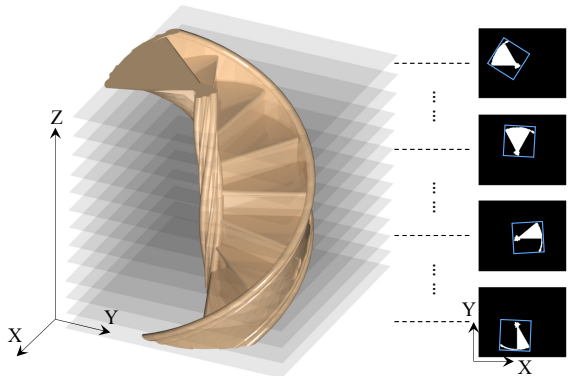


Fig. 3. To prepare mask images for DLP-based 3D printing, a digital model is usually sliced by a set of parallel planes (perpendicular to z-axis) and each slice is represented by a 2D binary image for projection. The blue rectangles shown in the right images specify the maximal region can be projected.

platform for an easier debugging. We name this hardware system as *delta DLP 3D printer*.

IV. SOFTWARE

In DLP-based 3D printing, the 3D digital model of an object is usually sliced by a set of parallel planes and each thin slice is fabricated by projecting a 2D mask image onto the surface of photocurable liquid. As illustrated in Fig. 3, 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.

Let $\mathcal{R}$ be the maximal area that can be projected by a light source, which is a rectangular region in our delta DLP 3D printer. Each slice can be represented by $\mathcal{S} = (\mathcal{P}, z)$, where z is the height of slice and $\mathcal{P} = (P_1 \cup P_2 \cup \cdots \cup P_n)$ is a set of n disconnected polygons, $\{P_i\}$, to be solidified. In traditional DLP 3D printers, the platform can only move vertically. In such case, even if a polygon P_i in a slice can be covered by $\mathcal{R}$, it is also possible that the whole model cannot be fabricated when $\cup_i P_i \supseteq \mathcal{R}$ for all the z values (see Fig. 3 for an example). Our system overcome 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 I: Given a polygon soup $\mathcal{P}$ in a plane, which contains several disconnected polygons $\{P_1, P_2, \cdots, P_n\}$ ($P_i \cap P_j = \emptyset$, $\forall i \neq j$), find a minimum set Ω of rectangles $\{R_k, k = 1, 2, \cdots\}$ all with a fixed size $w \times h$ such that $\mathcal{P} \subseteq \cup_k R_k$, $\forall R_k \in \Omega$.

Here, each rectangle R_k is called a *covering rectangle* in the rest of this paper.

V. DECOMPOSITION 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. A greedy heuristic is applied: for each polygon P in the soup $\mathcal{P}$, we find a

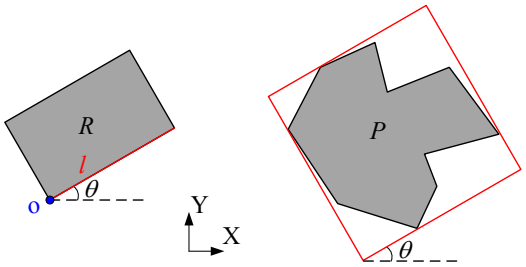
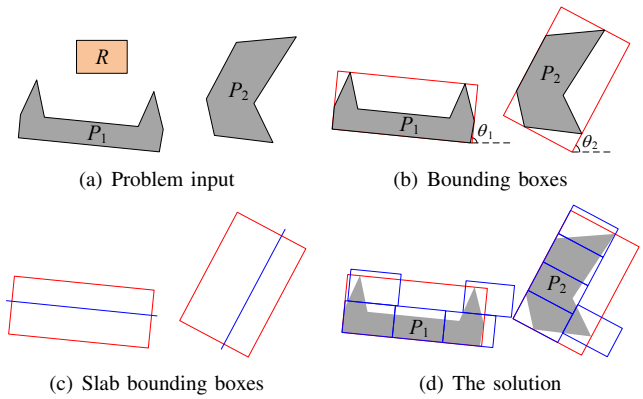


Fig. 4. The geometric location of a covering rectangle R in x-y plane is uniquely determined by a position $o(x, y)$ and a tilting angle θ, which is the angle between the bottom edge l of R and the x-axis. The tilted bounding box of a polygon P is also uniquely determined by the tilting angle θ.



(a) Problem input (b) Bounding boxes

(c) Slab bounding boxes (d) The solution

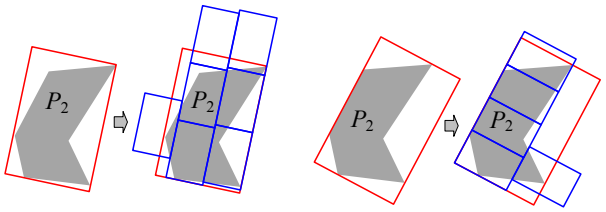
Fig. 5. An illustrative example shows the flow chart of our algorithm. (a) The problem is to find a small set of non-overlapped rectangles R to cover the polygonal shape $P_1 \cup P_2$. (b) Two bounding boxes with different orientations θ_1 and θ_2 are placed. (c) The bounding boxes are cut into slabs according to the width w (or the height h) of R. (d) Place covering rectangles in each slabs to get the solution.

bounding box of P and cover it by a set of non-overlapped rectangles with size $w \times h$.

First of all, a tilting angle θ is employed to define the rotation on a covering rectangle R and the rotated bounding box of a polygon P — see Fig. 4 for an illustration. To reduce the space of problem solving, we impose a constraint that the bounding box of a polygon P and its covering rectangles, R_i's, all having the same tilting angle. In practice, this constraint does not only simplify the problem but also speed up the speed of fabrication. Specifically, the platform of fabrication rotates only once (by the 42BYGH33 motor) and then moves in parallel (by the delta structure) to fabricate the solid according to a polygon. Therefore, in our system, motion planning is processed as solving the following problem.

Problem II: Considering the polygon soup $\mathcal{P}$ defined in Problem I, for each polygon $P \in \mathcal{P}$, find a tilting angle θ and a minimal set $\Omega(\theta)$ of non-overlapped rectangles $\{R_k, k = 1, 2, \cdots\}$ all with a fixed size $w \times h$ and tilted with angle θ, such that $P \subseteq \cup_k R_k$, $\forall R_k \in \Omega$.

When solving Problem II, another heuristic is employed in our algorithm: the optimal covering with parallel rectangles always has no overlap between the rectangles. Figure 5 provides an illustration of our algorithm's flow chart. After



(a) By the minimal bounding box (b) With an optimal tilting angle

Fig. 6. Problem cannot be optimally solved by using a minimal bounding-box: (a) seven rectangles are needed when covering P_2 (Fig. 5) in the orientation of a minimal bounding-box, and (b) only five covering rectangles are needed when having an optimal tilting angle.

computing the bounding boxes of each polygon with different orientations, each bounding box can be first cut into slabs with width equaling to w (or h). The covering rectangle R is then packed in each slab in a non-overlapped way. The orientation with minimal number of covering rectangles is used as the titling angle in the final fabrication to generate mask images.

Based on the above framework, our algorithm consists of two parts:

- **Inter step:** given a tilting angle θ, cover a polygon by a minimal number of covering rectangles in this tilting angle (Section V-A).
- **Outer loop:** find an optimal tilting angle θ_{opt} for each polygon (Section V-B).

It is worthy to note that computing a bounding box with minimal area (e.g., by the rotating calipers algorithm [10], [15]) does not always result in a minimal set of non-overlapped rectangles. An example is shown in Fig. 6. Although determining an optimal tilting angle takes more computing time, a covering with smaller number of rectangles can be found.

A. Optimal covering with a fixed orientation

We first tackle the problem of inner step in our algorithm. Given a predefined tilting angle θ, the polygon P is rotated by this angle first. The bounding box after rotation can also be easily determined by the minimal/maximal coordinates of P's vertices. Having four corner points of the bounding box defined as o_i $(i = 1, 2, 3, 4)$, the two adjacent sides of a corner point o_i is denoted as e_{i1} and e_{i2} (see Fig. 7(a) and (b)). For each side of the bounding box, we use line segments with length w (or h) to cover them as illustrated in Fig. 7(c). In our implementation, both w and h are tried, and the one leads to smaller number of covering rectangles at the end of computation is selected as the final result.

Without loss of generality, when e_4 is selected to be covered by line segments with length h (see Fig. 7(c)), 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. 7(d)). The following steps are then used to cover $S \cap P$ by a minimum number of rectangles, $\{R_i\}$.

- **Step 1:** Denote the set of all polygonal vertices falling into S as $\mathcal{V}(S)$.

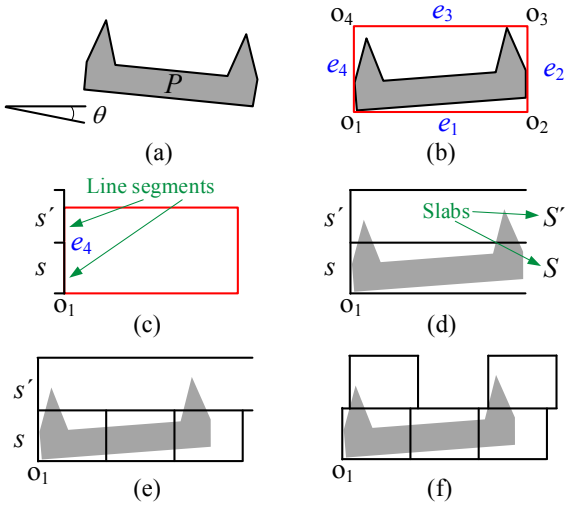


Fig. 7. Steps for covering with a fixed orientation: (a) given a predefined tilting angle θ for an orientation, (b) rotate the polygon P and find its axis-aligned bounding box, (c) cover the box edge e_4 by two line segments s and s' in an end-to-end manner, (d) each line segment s (or s') defines a slab S (or S'), (e) cover the slab $S \cap P$ by a minimum set of rectangles R, and (f) cover the $S' \cap P$.

- **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 R 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 R.
  - *Step 5.1:* If $\overline{e}_2$ intersects P, locate a rectangle R 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 R.
  - *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 has been shown in Fig. 7(e) and (f) 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. Here, n is the number of edges in the polygon. In the worst case, there are $O(n)$ points in $\mathcal{I}$ in Step 3. As a result, our algorithm runs in $O(n^2 \log n)$ time.

B. Finding an optimal orientation

The problem of finding an optimal orientation is formulated as finding a minimal value of a function $f(\theta)$, which returns the minimal number of rectangles to cover a polygon

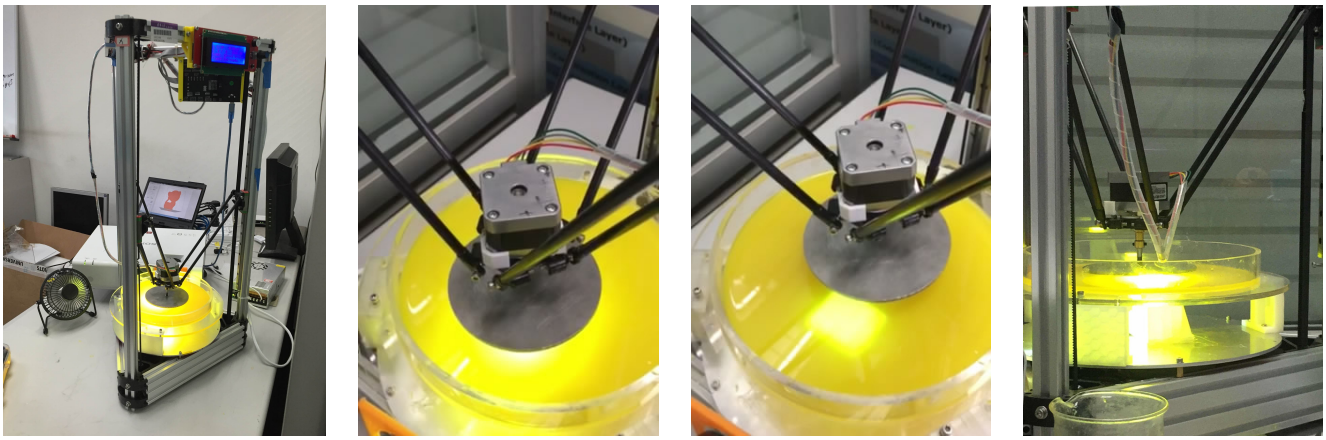


Fig. 8. Photographs to illustrate different stages of fabrication using our delta DLP 3D printer. More details can be found in supplemental demo video.

P. As the function value is integer and the function $f(\theta)$ can only be implicitly evaluated, we approximate its function value by a real function $\widehat{f}(\theta)$ and then find the 'optimal' value of θ by applying numerical optimization on $\widehat{f}(\theta)$.

First of all, we evaluate the value of $f(\theta)$ at a few sampled angles, which consists of two parts:

1) a uniform sampling $\{\theta_i = \frac{i-1}{6}\pi\}$ $(i = 1, \cdots, 6)$, and
2) a set of angles by aligning each edge of P's convex hull along the vertical axis.

The set of all these tilting angles is denoted by Θ. The approximation function is then constructed to interpolate the function values at these samples, that is

$$\widehat{f}(\theta_i) = f(\theta_i) \quad (\forall \theta_i \in \Theta). \tag{1}$$

Due to the good property of interpolation, the *radial basis functions* (RBFs) [16] is employed here. That is,

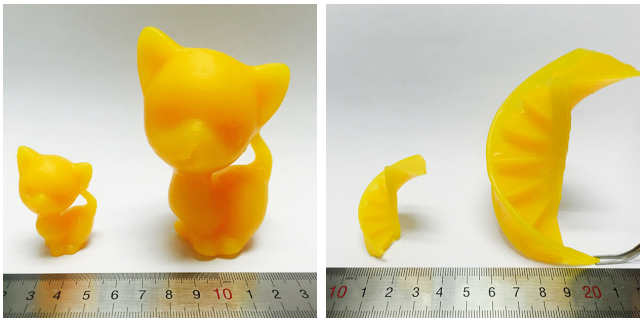
$$\widehat{f}(\theta) = \sum_{\theta_k \in \Theta} \lambda_k \Phi(\theta - \theta_k) \tag{2}$$

where λ_k is a coefficient for each θ_k, and we choose the Gaussian RBF $\Phi(r) = e^{-r^2}$ due to its property of positive definite. Substituting Eq.(1) into Eq.(2), all the coefficients λ_k can be determined by solving a linear system. Given the analytical form of $\widehat{f}(\theta)$, we find its minimal value θ_{opt} using a variant of Brent's method [17]. After obtaining an 'optimal' titling angle θ_{opt}, the optimal covering by R can be determined by the method introduced in Section V-A.

Complexity Analysis: In the worst case, there are $O(n)$ elements in Θ and the algorithm to evaluate the function value of $f(\theta)$ is in the complexity of $O(n^2 \log n)$. As a result, our overall optimization algorithm has a time complexity of $O(n^3 \log n)$.

VI. EXPERIMENTS AND DISCUSSION

We have implemented this proposed algorithm in our hardware system. In our practice, a printed 3D model usually have 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 (2.80GHz) and 8GB RAM. Furthermore, models in different sizes are fabricated to verify the performance of our delta DLP 3D printer (see Fig. 9). When the models are small, each of their slices can be completed covered by the region of a single projection, R. There is no need to apply the decomposition 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 are too large and they can only be fabricated by applying decomposition and horizontal movement.



(a) Kitten (b) Stair

Fig. 9. Results of fabrication — all are fabricated by our system, where the smaller ones are fabricated without applying horizontal movement on the platform as their cross-sections can be fully covered by the region of a projection, R.

TABLE I
STATISTIC OF FABRICATION

Model	Kitten		Stairway	
#Triangles	2,470		9,842	
Size	×1.0	×2.0	×1.0	×2.0
#Layers	460	922	367	736
Height (mm)	47.2	93.4	37.9	74.8
Total Time	92 min.	608 min.	78 min.	222 min.

Statistic of fabrication is listed in Table I. Note that, the thickness of each layer in fabrication is $0.1mm$, and the solidification time of each projection is 10 sec. The speed

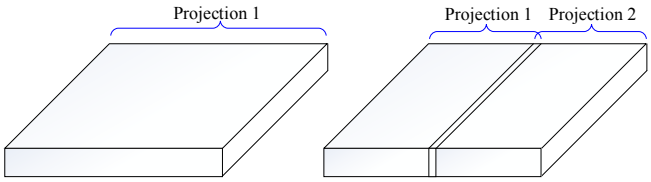


Fig. 10. Two methods for fabricating samples to verify the mechanical strength of models: I) one layer one project and II) one layer by two projections – half for each.

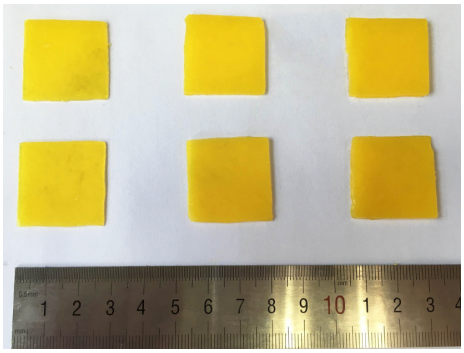


Fig. 11. Printed samples for verifying the mechanical strength of models fabricated by our method: from left to right, $0.5mm$, $1.0mm$ and $1.5mm$ respectively in thicknesses – all have the size of $26mm \times 26mm$. Top row is fabricated by method I and the bottom row by method II.

of rotation by 42BYGH33 motor is 0.3 rad./sec., and the maximal speed for translation is 80 mm/sec. It is easy to find that our method has a very good scalability — when increasing the volume of solid to be fabricated by $8\times$, the total fabrication time only increases $6.60\times$ and $2.85\times$ respectively. Figure 8 shows the photographs of system in the process of fabrication.

To verify the mechanical strength of a model fabricated by decomposing a large layer into smaller pieces in delta DLP 3D printing, we fabricate models with the same area of cross-sections (i.e., $26mm \times 26mm$) but different thickness ($0.5mm$, $1.0mm$ and $1.5mm$ respectively). The models are made by two different methods: I) each layer by one complete projection (left of Fig. 10) and II) each layer is decomposed into two projections — half for each (see the right of Fig. 10). Some fabricated samples can be found in Fig. 11.

These samples are tested in both the tensile stretch and the bending tests. In both tests, the similar mechanical stiffness is observed on the models fabricated by methods I and II. In other words, no significant weakness is found on the models made by decomposing a large cross-section into smaller ones to be solidified. The detailed test data by universal material testing machine, as well as tests on multiple z-layers, will be reported in a future work. The models fabricated by our delta DLP printer can satisfy the requirement of normal usage. Moreover, when being applied to a practical model (e.g., the ones shown in Fig. 9), 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.

VII. CONCLUSIONS

In this paper, we present a system for DLP-based 3D printing with large size. The hardware of our system is based on an extension of parallel delta robot and a conventional DLP printer. 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 done by a single projection. A simple yet effective algorithm is developed to decompose the large area of a layer into 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. The results showed that similar stiffness can be found. The functionality of our system has been verified on freeform models in different sizes. One future work is to design an adaptive tilting mechanical structure for overcoming the restriction on parallelism between the platform and the bottom of tank. When the system is not carefully calibrated for this parallelism in our current setup, high stress is incurred and the printing could be failed. Another future work is to improve the partitioning algorithm with better approximate ratio by considering the skeleton of the polygonal shape [18].

REFERENCES

[1] I. Gibson, D.W. Rosen, B. Stucker, Additive Manufacturing Technologies. Springer, 2010.
[2] http://www.fslaser.com/
[3] Y. Pan, Y. Chen, C. Zhou, Fabrication of smooth surfaces based on mask projection stereolithography, in Proc. Solid Freeform Fabrication Symposium, pp. 263-278, 2011.
[4] Y. Pan, Y. Chen, C. Zhou, Fast Recoating Methods for the Projection-based Stereolithography Process in Micro- and Macro-scales, in Proc. Solid Freeform Fabrication Symposium, pp. 846-862, 2012.
[5] J.M. Keil, Polygon Decomposition, in Handbook of Computational Geometry, Chapter 11, pp. 491-518, 2000.
[6] J. O'Rourke, K.J. Supowit, Some NP-hard polygon decomposition problems. IEEE Transactions on Information Theory, vol. 29, no. 2, pp. 181-190, 1983.
[7] J.C. Culberson, R.A. Reckhow, Covering Polygons Is Hard. Journal of Algorithms, vol. 17, no. 1, pp. 2-44, 1994.
[8] L.J. Aupperle, H.E. Conn, J.M. Keil, J. O'Rourke, Covering orthogonal polygons with squares, in Proc. 26th Allerton Conf. Commun. Control Comput., pp. 97-106, 1988.
[9] A. Hegedüs., Algorithms for covering polygons by rectangles. Computer-Aided Design, vol. 14, no. 5, pp. 257-260, 1982.
[10] F.P. Preparata, M.I. Shamos, Compuational Geometry: An Introduction. Springer-Verlag, 1985.
[11] C. Levcopoulos, 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 Volume 1269), pp. 27-41, 2005.
[12] A. Lingas, The power of non-rectilinear holes, in Proc. 9th Internat. Colloq. Automata Lang. Program. (Lecture Notes in Computer Science Volume 140), pp. 369-383, 1982.
[13] http://cs.stackexchange.com/questions/16661/tiling-an-orthogonal-polygon-with-squares/16801#16801
[14] C. Bell, 3D Printing with Delta Printers. Apress, 2015.
[15] G. Toussaint, Solving geometric problems with the rotating calipers. In Proc. IEEE MELECON'83, Athens, Greece, 1983.
[16] M.D. Buhmann. Radial Basis Functions: Theory and Implementations. Cambridge University Press, 2003.
[17] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, Numerical Recipes in C. Second Edition, Cambridge University Press, 1992.
[18] Y.J. Liu, Semi-continuity of skeletons in 2-manifold and discrete Voronoi approximation. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 34, no. 9, pp. 1938-1944, 2015.