

Image-Based Slicing and Tool Path Planning for Hybrid Stereolithography Additive Manufacturing

Hang Ye

Department of Industrial and
Systems Engineering,
University at Buffalo,
The State University of New York,
Buffalo, NY 14260
e-mail: hye2@buffalo.edu

Chi Zhou¹

Department of Industrial and
Systems Engineering,
University at Buffalo,
The State University of New York,
Buffalo, NY 14260
e-mail: chizhou@buffalo.edu

Wenyao Xu

Department of Computer Science
and Engineering,
University at Buffalo,
The State University of New York,
Buffalo, NY 14260
e-mail: wenyaoxu@buffalo.edu

The hybrid stereolithography (SLA) process integrates a laser scanning-based system and a mask projection-based system. Multiple laser paths are used to scan the border of a 2D pattern, whereas a mask image is adopted to solidify the interior area. By integrating merits of two subsystems, the hybrid SLA process can achieve high surface quality without sacrificing productivity. For the hybrid system, closed polygonal contours are required to direct the laser scanning, and a binary image is also needed for the mask projection. We proposed a novel image-based slicing method. This approach can convert a 3D model into a series of binary images directly, and each image is corresponding to the cross section of the model at a specific height. Based on the resultant binary image, we use an image processing method to gradually shrink the pattern in the image. Boundaries of the shrunk image are traced and then restored as polygons to direct the laser spot movement. The final shrunk image serves as the input for the mask projection. Experimental results of test cases demonstrate that the proposed method is substantially more efficient than the traditional approaches. Its accuracy is also studied and discussed. [DOI: 10.1115/1.4035795]

Keywords: additive manufacturing, stereolithography, slicing, contour tracing, image shrinking

1 Introduction

Among various additive manufacturing (AM) processes, the stereolithography apparatus (SLA) was the first commercialized one [1], and it is one of the most popular AM technologies nowadays. There are two typical SLA configurations [1]. One is to use a laser to cure the liquid resin, and the laser scans the cross-sectional area point by point. An advantage of this configuration is that it can generate comparatively high accuracy, but it is time-consuming as the laser spot has to pass all points within the cross section. The other configuration is the application of projection devices, such as a digital light projection (DLP), to illuminate the liquid resin. It can dramatically reduce the production time because the whole layer is cured simultaneously by exposing a designed digital mask. But its accuracy is limited by the resolution of the projection device. Extensive research work has been done to improve the efficiency and/or accuracy of SLA from different perspectives [2–7,28].

However, most researchers in this field have been focusing on improving either one of the two configurations, and it would always result in making trade-off between efficiency and accuracy. This inherent trade-off has been identified as a major technology barrier existing across many AM processes [8]. Most recently, Zhou et al. proposed a hybrid SLA process to combine the laser scanning with the projection such that one subsystem can complement the other to improve the overall performance [9]. It adopts a laser as the energy source for the border of a 2D pattern, whereas a mask image is used to solidify the interior area. Because a single laser path results in ultrathin lines, multiple paths are necessary on the border area to offer sufficient mechanical strength and to securely bond the material on the boundary with the internal body [9]. The hybrid SLA system can take advantages

from both sides, i.e., achieving relatively high surface quality without sacrificing productivity.

Although traditional slicing and tool path planning approaches can be applied to the hybrid SLA system seamlessly, a configuration-specific method is desirable. In this paper, we will introduce a novel slicing and tool path planning method which can well suit the hybrid system. The remainder of this paper is organized as follows: Section 2 will review how the existing slicing and tool path planning approaches can be adopted by the hybrid system, and a new data flow paradigm will be proposed. Section 3 will introduce an image-based slicing method which can convert an STL format 3D model into binary images directly. The sampling accuracy will also be discussed in this section. Section 4 will describe an image contraction method to shrink the original binary image, and the resulting error will be investigated as well. The efficiency analysis and experimental results will be presented in Sec. 5. The conclusion will be discussed in Sec. 6.

2 Data Flow for Hybrid Stereolithography System

2.1 Flow Chart of Slicing and Tool Path Planning. The hybrid system synthesizes a laser scanning-based SLA system and a mask projection-based SLA system. Therefore, input files from both sides are required for this system. To be specific, at each layer two input files are necessary: (1) contours for laser scanning and (2) a binary image for mask projection. We will review how to use the traditional method to generate these two required input data in this section.

The AM process usually takes an stereolithography (STL) file as the raw data, which is the triangulated surface representation of a 3D object. To build an object in a layer by layer fashion, the shape of the cross section at each layer is required. The process to obtain the geometric property of the cross section area for each layer is usually termed as “slicing.” A common practice for slicing is to calculate the intersections between a given slicing plane and

¹Corresponding author.

Manuscript received September 21, 2016; final manuscript received December 21, 2016; published online March 8, 2017. Assoc. Editor: Zhijian J. Pei.

triangular meshes from the STL file, and then all intersections are re-organized to obtain a contour (simple polygon). For the laser scanning-based SLA process, a series of polygons are generated after slicing by offsetting, and the offset distance t is always negative, i.e., inward offsetting for outside contour and outward offsetting for inside contour. These offset polygons serve as paths for laser scanning in the hybrid system. For the mask projection-based SLA process, the input is a series of binary (or gray scale) images. The pattern in each image is defined by the area enclosed by a contour (or by nested outside and inside contours). The major steps to convert a contour into a binary image include: (1) select a sampling point for each pixel from the area the pixel covered; (2) test whether the selected point is in the area encompassed by the contour; (3) if the sampling point is in the area, set its corresponding pixel as a foreground pixel. Otherwise, its corresponding pixel will be set as background. The scanline rendering algorithm [10] which implements this idea is the common practice to convert contour information into a binary image. In the hybrid system, the area for the mask projection is defined by the innermost polygon generated from the outside contour and the outermost polygon generated from the inside contour. The liquid photopolymer at other areas on the cross section will be cured by the laser scanning.

In the upper part of Fig. 1, a bunny model is used to illustrate the traditional slicing and tool path planning method for the hybrid system. Figure 1(a) shows the original STL format 3D model. Figure 1(b) shows the sliced contour at a specific height. Figures 1(c)–1(e) show the offset contours, and these polygons will serve as laser paths. Figure 1(f) shows the binary image which will be used for the mask projection.

The existing slicing and tool path planning method reviewed above can fulfill the requirement of the hybrid SLA system, but it also suffers from several drawbacks. For example, during the slicing process, the connectivity among contour segments has to be restored. Compared with the intersection calculation, retrieving this topological information takes much more time. In addition, in order to conduct the offsetting operation accurately, the contour orientation has to be consistent, e.g., all outside contours are oriented counter-clockwise (CCW), whereas all inside contours are oriented clockwise (CW). Extra effort is required to maintain this consistency. Besides the aforementioned disadvantages, the most challenging part stems from the offsetting operation itself. Because offsetting is not a topology preserving operation, it is possible that the original object and its offset are not homeomorphic. This feature of topological inconsistency makes the polygon

offsetting problem difficult to tackle. Tremendous efforts have been made to maintain the robustness of offsetting operation [11–13], however, often times, such efforts increase the algorithm complexity dramatically.

2.2 Image-Based Method. The method to solve an engineering problem can be categorized as either an analytical method or a numerical method. The approach reviewed in Sec. 2.1 adopts the analytical method to generate the exact contour of the cross section area. In the later polygon offsetting operation, the offset polygonal contours are also generated by the analytical method. An analytical method holds a significant advantage that it provides an exact solution to the original problem, whereas a numerical method can only provide an approximate solution. However, in the real world every engineering setup has its finite resolution, and any feature beyond that cannot be handled by the system. Under this circumstance, obtaining an exact solution for an engineering problem might not be necessary. Furthermore, in order to preserve the accuracy, the time complexity of the analytical method is usually higher than the numerical method. In other words, solving the same problem numerically can achieve higher efficiency with the trade-off of accuracy. Motivated by this rationale, in this paper, we will derive an image-based approach which is essentially numerical. This approach can provide higher efficiency than the existing analytical method while maintaining sufficient accuracy at the same time.

The proposed approach generates binary images directly from an STL file. As each binary image represents the cross section area of a 3D object at a specific height, it can be used in the mask projection based SLA process directly. But for the hybrid system, this image needs to be further processed to create: (1) contour paths to direct the laser scanning and (2) a shrunk image for the mask projection. We first extract the boundary pixels from the image by the image contour tracing algorithm, and then carry out image shrinking by the image processing technique. The extracted image contours will be converted into laser scanning paths, and the final resultant binary image will be used for the mask projection. In the lower part of Fig. 1, the same bunny model is used to demonstrate the proposed approach. Figure 1(g) shows the original image at a specific height. Figures 1(h)–1(j) are intermediate shrunk images. Figure 1(k) shows the final shrunk image. Figures 1(l)–1(n) show restored contours to direct the laser scanning. In Secs. 3–4, we will elucidate that the numerical method (image-based slicing and tool path planning) can achieve equivalent input

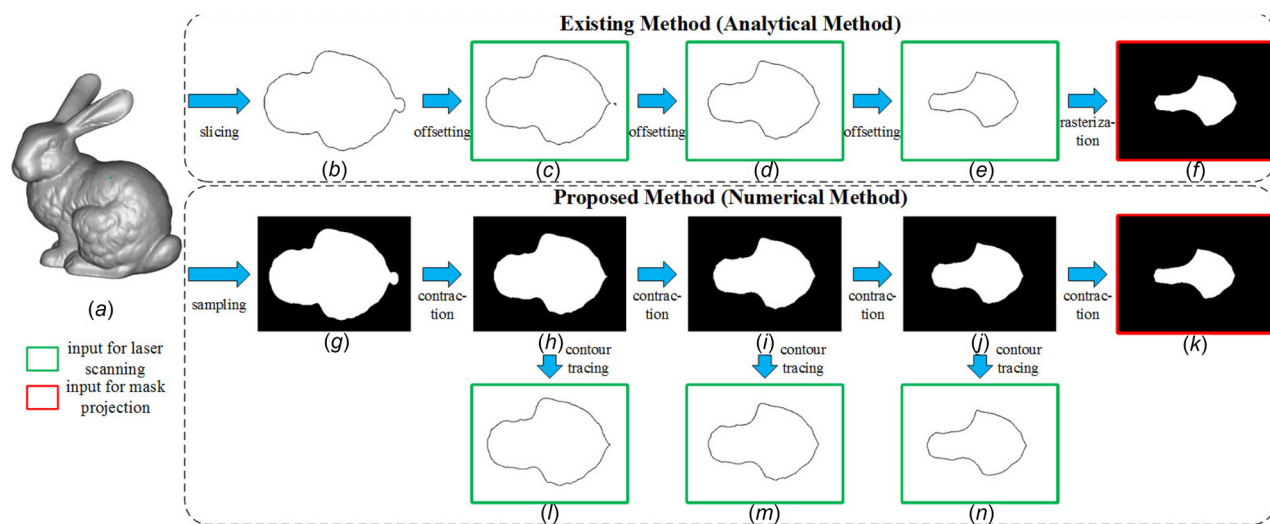


Fig. 1 Bunny model: (a) original 3D model, (b) original contour, (c) first offset contour, (d) second offset contour, (e) third offset contour, (f) mask image, (g) original image, (h) first shrunk image, (i) second shrunk image, (j) third shrunk image, (k) final mask image, (l) first reconstructed contour, (m) second reconstructed contour, and (n) third reconstructed contour

data for the printing process with substantially higher efficiency than the analytical method.

3 Image-Based Slicing Algorithm

3.1 Existing Slicing Approach. Most existing slicing methods fall into one of the two categories: nontopology-based and topology-based. For both methods, it is necessary to compute the intersection between a given line segment (defined by two vertices $V_1(X_1, Y_1, Z_1)$ and $V_2(X_2, Y_2, Z_2)$) and a horizontal plane ($Z = Z_k$), and it can be easily calculated by linear interpolation [9].

For the nontopology-based method, each triangular facet from an STL model needs to be traversed to check whether it intersects with the current slicing plane or not. If n triangles intersect with current slicing plane, $2n$ end points from n line segments will be computed, and their connectivity can be determined by the “closest point” method [1,29]. The topology-based method is also known as “marching” algorithm. The key of this method is the half-edge data structure (also referred as “doubly-connected edge list” in Ref. [14]). All edges from an STL model need to be pre-processed to make mesh connectivity explicit. With this information, we can march from one triangle to the next until getting back to the starting triangle which closes the polygonal contour [15].

The nontopology-based method is straightforward, but ordering line segments is time-consuming. Although the time complexity of the topology-based method is lower than the nontopology-based method, a sophisticated data structure has to be created, and the construction of this data structure can also be time-consuming. In addition, this method is not as robust as the nontopology-based method. The idea of “information reuse” introduced in Refs. [16] and [17] aims to improve the slicing efficiency from another perspective. It takes advantage of known information obtained from a template model to accelerate the slicing of its derivatives. Although this method is very efficient for derived models, the template model still has to be sliced by either the topology-based or the nontopology-based method.

3.2 Image-Based Slicing Method. In this section, an image-based method is presented. It generates a binary image with a finite resolution, denoted as γA , to represent a 2D planar area A directly from an STL file. Major steps of the image-based slicing algorithm include: (1) convert the STL file into the sampling point cloud and (2) generate a series of images according to sampling points from bottom to top in an accumulative manner.

3.2.1 Sampling Point Conversion. For a solid S in \mathbb{R}^3 , its STL file can be seen as a triangulated surface representation of S . In the context of geometric modeling, a surface is defined as “an orientable continuous 2D manifold embedded in \mathbb{R}^3 [18], denoted as ∂S . Here, ∂S is an infinite set whose elements are orientable points. Sampling points are some points we intentionally select from ∂S in order to approximately represent the surface. In other words, the set of sampling points I is a proper subset of ∂S .

Suppose the binary image to be generated is positioned at a rectangular area defined by its four vertices $(0, 0, 0)$, $(W, 0, 0)$, $(W, H, 0)$, and $(0, H, 0)$, and a 3D model is also placed in this planar domain. If the resolution of the image is $m \times n$, the area each pixel covered is a grid with the width $d = W/m$ (or H/n). Among all elements in ∂S , the points with the following X - and Y -coordinates are selected as sampling points. The Z -coordinate for a sampling point can be obtained by calculating the intersection between the line represented by Eq. (1) and the triangular facet defined by its three vertices

$$\begin{cases} X_{ij} = (i - 0.5)d, & \text{for } i = 1, 2, \dots, m \\ Y_{ij} = (j - 0.5)d, & \text{for } j = 1, 2, \dots, n \end{cases} \quad (1)$$

The pseudocode for converting an STL file into sampling points is shown as below. Algorithm `SCANLINERENDERING` ($T[i], d$) is the

scanline triangle rendering [10], and it is used to determine incident pixels for a triangular facet. Algorithm calculate $Z(X, Y)$ is to compute the Z -coordinate for a sampling point. Two examples of sampling point conversion are shown in Fig. 2. There are two types of sampling points: entering point and exiting point. In the context of geometric modeling, the entering point of a solid S can be defined as the first point of a local feature we meet along a given direction, and the exiting point is the last point we meet along the same direction.

Algorithm 1 Convert To Sampling Points

Input: triangular facet T , pixel width d
Output: sampling point cloud I

01. $I \leftarrow \phi$
02. **for** each triangular facet $T[i]$ in T **do**
03. $incidentPixel \leftarrow \text{ScanlineRendering}(T[i], d)$
04. **for** each pixel $incidentPixel[j]$ in $incidentPixel$ **do**
05. $S.col \leftarrow incidentPixel[j].column$
06. $S.row \leftarrow incidentPixel[j].row$
07. $X \leftarrow (S.column - 0.5) \times d$
08. $Y \leftarrow (S.row - 0.5) \times d$
09. $S.Z \leftarrow \text{Calculate } Z(X, Y)$
10. $I \leftarrow I \cup S$
11. **end for**
12. **end for**
13. **return** I

3.2.2 Binary Image Generation. In Sec. 3.2.1, the STL file has been converted into sampling points. However, the sampling points are loosely collected, and they cannot be used as the input for a layer-based manufacturing process. In this section, those sampling points will be converted into a series of images in an accumulative manner, and each image is corresponding to the cross section area for a given slicing plane ($Z = i \times d_{LT}$, for $i > 0$, and d_{LT} is the layer thickness). The basic idea of this accumulative method is that the image for the $(i + 1)$ th layer, P_{i+1} , is generated by manipulating the pixel values of the image for the i th layer, P_i , according to the sampling points between these two consecutive layers.

For the $(i + 1)$ th layer, the binary image P_{i+1} with a finite resolution $m \times n$ can be defined by a set of pixel values $g_{i+1,j}$ ($1 \leq j \leq m \times n$), where j is corresponding to the pixel position. Initially, $g_{i+1,j}$ is set as the same as $g_{i,j}$. At position j , for each sampling point Q_k between the i th layer and the $(i + 1)$ th layer, $g_{i+1,j}$ has to be updated once. These updates represent local feature

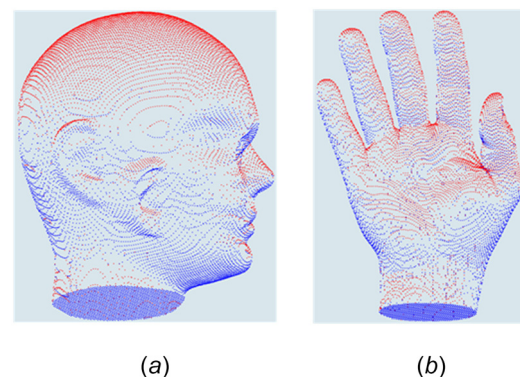


Fig. 2 Sampling point conversion: (a) head model and (b) hand model

changes along the building direction. For a closed 2D manifold, the entering point and the exiting point must occur one after the other, and at each pixel position the total number of sampling points must be even. Therefore, the relation between $g_{i,j}$ and $g_{i+1,j}$ can be summarized as in the following equation:

$$g_{i+1,j} = \begin{cases} g_{i,j} & \text{if } \# \text{ of } Q_k \text{ is even} \\ (g_{i,j} + 1) \bmod 2 & \text{if } \# \text{ of } Q_k \text{ is odd} \end{cases} \quad (2)$$

The pseudocode for the binary image generation is shown as below. The image is generated from the bottom to the top, and each sampling point only takes effect once. Therefore, sorting all sampling points according to their Z -coordinates can make this algorithm more efficient, and only the sampling points that fall between two consecutive layers need to be visited to generate an image corresponding to a specific height. In Fig. 3, an example object is used to demonstrate how binary images are generated from sampling points. Initially, all pixels in the image are set as background. At the zeroth layer, since all sampling points are entering points, all pixels in the circular area are set as foreground. At the first layer, all sampling points between the zeroth and the first layers are exiting points. Therefore, based on P_0 , all pixels in the annulus are set as background. Following the same rule, we can derive P_2, P_3 , and P_4 .

3.3 Sampling Accuracy. Because the proposed image-based slicing algorithm selects a finite number of points from ∂S as sampling points, some features of ∂S might be missing. However, it is necessary to guarantee the topological equivalence between an original 3D object S and the reconstruction of its digital images. In other words, S and its reconstruction have to be homeomorphic. The importance of homeomorphism has been briefly discussed in Ref. [19,27]. Intuitively, the denser a sampling point cloud is, the more possible a 3D object S and its reconstruction is homeomorphic. But if the sampling point cloud is too dense, its generation and processing will be extremely time-consuming. In addition, as each projector has its own finite resolution, it will be pointless to use higher resolution images. In this section, we will briefly discuss the sufficient condition to ensure the homeomorphism between S and its reconstruction.

The following definitions and theorem are adopted from Ref. [20], and Definition 1 and Theorem 1 are Definition 1 and Theorem 16 in Ref. [20], respectively.

DEFINITION 1. A set $S \subset \mathbb{R}^3$ is called r -regular if, for each point $x \in \partial S$, there exist two osculating open balls of radius r to ∂S at x such that one lies entirely in S and the other lies entirely in the complementary set of S .

DEFINITION 2. Any set G which is a translated and rotated version of the set $(2r'/\sqrt{3})\mathbb{Z}^3$ is called a cubic r' -grid, and its elements are called grid points.

THEOREM 1. Let S be an r -regular object and G be a cubic r' -grid with $2r' < r$. Then, the result of a topology preserving reconstruction method is r -homeomorphic to S .

From Definition 2, the edge length of a cubic r' -grid $d' = 2r'/\sqrt{3}$. According to Theorem 1, we can derive the following conclusion: For an r -regular object S and a cubic r' -grid G , if the edge length of G , d' , satisfies $d' < (\sqrt{3}/3)r$, the result of a topology preserving reconstruction method is r -homeomorphic to S .

In the context of additive manufacturing, in the X - Y plane, d' is represented by the pixel width d . And in building direction (Z -axis), d' is represented by the layer thickness d_{LT} . Therefore, we can conclude that for an r -regular object S , if both the pixel width d and the layer thickness d_{LT} are less than $(\sqrt{3}/3)r$, the homeomorphism between S and its reconstruction can be guaranteed.

It is not necessary that all 3D models are r -regular. For instance, a polyhedron does not fall in this category. However, for the model to be built by an AM process, the magnitude of the

parameter r can be interpreted as the size of the minimal spatial feature of the model. This is because any AM setup has a finite resolution, and objects fabricated by this specific machine can only present features whose sizes are larger than this defined resolution. This resolution is comprised of two independent components, the lateral resolution (X - and Y -axis direction) and the vertical resolution (Z -axis direction). For an SLA process, the lateral resolution is defined by the diameter of a laser spot (for the laser scanning-based SLA process) or the physical size of a pixel (for the mask projection-based SLA process). The vertical resolution is defined by the minimal step length of the elevator and the penetration depth of the resin. If the minimal feature from an object is beyond these thresholds, it cannot be built as desired. Therefore, in order to properly build an object with minimal spatial feature size r , both the pixel size and the layer thickness have to be less than $(\sqrt{3}/3)r$.

Algorithm 2 Binary Image Generation

Input: sampling point cloud I , layer thickness d_{LT}

Output: binary image set P

```

01.  $P \leftarrow \phi$ 
02.  $I' \leftarrow$  Sort  $I$  according to  $Z$ -coordinate
03. for  $j \leftarrow 1$  to  $m \times n$  do
04.    $g[j] \leftarrow 0$ 
05.    $num[j] \leftarrow 0$ 
06. end for
07.  $i \leftarrow 1$ 
08.  $k \leftarrow 1$ 
09.  $size \leftarrow$  size of  $I'$ 
10. while  $k \leq size$  do
11.   if  $I'[k].Z \leq I'[1].Z + i \times d_{LT}$  then
12.      $j \leftarrow (I'[k].row - 1) \times n + I'[k].column$ 
13.      $num[j] \leftarrow num[j] + 1$ 
14.      $k \leftarrow k + 1$ 
15.   else
16.     for  $j \leftarrow 1$  to  $m \times n$  do
17.       if  $num[j] \bmod 2 = 1$  then
18.          $g[j] \leftarrow (g[j] + 1) \bmod 2$ 
19.       end if
20.        $P[i] \leftarrow P[i] \cup g[j]$ 
21.        $num[j] \leftarrow 0$ 
22.     end for
23.      $P \leftarrow P \cup P[i]$ 
24.      $i \leftarrow i + 1$ 
25.   end if
26. end while
27. return  $P$ 

```

4 Image-Based Tool Path Planning

4.1 Existing Tool Path Planning Method. As shown in the upper part of Fig. 1, the existing tool path planning for the hybrid SLA process takes the polygonal contour (Fig. 1(b)) as input, and this contour information is obtained from slicing. Each polygonal contour L_0 encloses an area A_0 , and successively negative offsetting (shrinkage) will be conducted on this area with the offset distance t_i . Consequently, a series of shrunk areas A_i for $i = 1, 2, 3, \dots, n$ will be generated, and their corresponding polygonal contours L_i for $i = 1, 2, 3, \dots, n - 1$ will serve as laser paths. The area enclosed by the last polygonal contour, A_n , will be converted into a binary image to serve as the input for the mask projection.

The offsetting operation here should be the solid offsetting [21,22], which can be defined as follows:

DEFINITION 3. The regularized solid offset of a regular and non-empty set A by a negative distance t (shrinking) is defined as $A \downarrow^* t = \{p : d(p, c^*A) \geq t\}$, where $d(p, c^*A) = \inf_{q \in A} \|p - q\|$.

Here, since the planar area A_i is an infinite set, it is impossible to implement the solid offsetting operation according to its definition. The common practice for offsetting a polygonal area is to move each segment of the polygon along its normal by the offset distance t , and this operation is named as normal offsetting [21]. However, the resultant polygon generated by the normal offsetting might not be simple any more, and furthermore, the identification and removal for the self-intersections are difficult.

4.2 Image-Based Tool Path Planning Approach. According to Definition 3, if $p \in A$, then, $d(p, c^*A) = d(p, \partial A)$, and if $p \notin A$, then $d(p, c^*A) = 0$. Therefore, the negative offset can be viewed as shifting the center of a circular disk with radius t along ∂A , removing all the area swept by the disk, and the remainder is $A \downarrow^* t$. The image-based tool path planning method, which will be introduced later in this section, is inspired by this idea. In the proposed method, binary images are used to represent A , ∂A , and the disk with radius t . This kind of representation may lead to losing a certain level of accuracy due to the finite resolution from an image, but it enables the computerized realization of aforementioned “sweep” and “removal” processes. More importantly, as a field which has been extensively studied, the image processing provides us a wealth of readily available tools that can guide us to realize desired goals.

The proposed tool path planning method extracts boundary pixels from the image generated by the image-based slicing method. Then based on these pixels, an image processing technique is applied to generate the shrunk image. The centers of boundary pixels from intermediate shrunk images will be used to define laser paths, and the final shrunk image will serve as the input for the mask projection.

4.2.1 Image Contour Tracing. According to the proposed image-based slicing algorithm, a binary image will be generated from the STL file for a specific layer. This binary image may contain multiple isolated objects, and each object may have multiple contours, e.g., object with holes. Similar to representing a 2D area by its contour, each object on the binary image can also be represented by its boundary pixels. In this section, we will use the eight-connected inner boundary as the image contour. The pixel connectivity (four-connected and eight-connected) and two types of boundaries (inner and outer) were defined in the literature [23].

Two classic image contour tracing algorithms were offered by literature [23] and [24], respectively. They both start from a known boundary pixel, and then search its neighborhood to identify the next boundary pixel. As a contour of a given object is a closed loop, the contour propagation process should stop if the next boundary pixel found would be the same as the starting pixel. To tackle the multicontour case, we first scan the whole image from the northwest corner to check the left neighbor for each foreground pixel. If a foreground pixel has a background left neighbor, it will be selected as a seed pixel, and all seed pixels will be stored in a set. By doing this, we can ensure that each contour has at least one pixel being selected as a seed. Then, the first seed pixel will serve as the starting pixel to trace a contour. During the contour propagation, the set of seed pixels will be updated, and all seed pixels on this traced contour will be removed. When a contour is finished, if the set of seeds becomes empty, all contours on the image have been traced. Otherwise, we will trace a new contour from another seed pixel in the set.

4.2.2 Image-Based Contour Offset. In the traditional method, when the polygonal contour is derived, we will conduct a negative offset by distance t to obtain a contracted area. In this section, we

will generate the contracted binary image by the image processing technique.

Assuming P_0 is the original image generated by the image-based slicing algorithm, its image contour C_0 can be traced by contour tracing algorithms. The image-based contour offsetting is implemented through shrinking the original image P_0 by applying a structuring element E_i along the boundary C_0 . Specifically, a dilation operation is conducted on the contour image C_0 by the structuring element E_i , and the shrunk image P_i can be calculated as the Boolean difference between P_0 and dilated C_0 . This relationship can be represented as Eq. (3), where “ \oplus ” denotes the dilation, and “ $-$ ” denotes the Boolean difference

$$P_i = P_0 - (C_0 \oplus E_i) \quad (3)$$

Based on this rule, the original image is shrunk n times. The corresponding contours C_1, C_2, \dots, C_{n-1} , are used to generate paths to

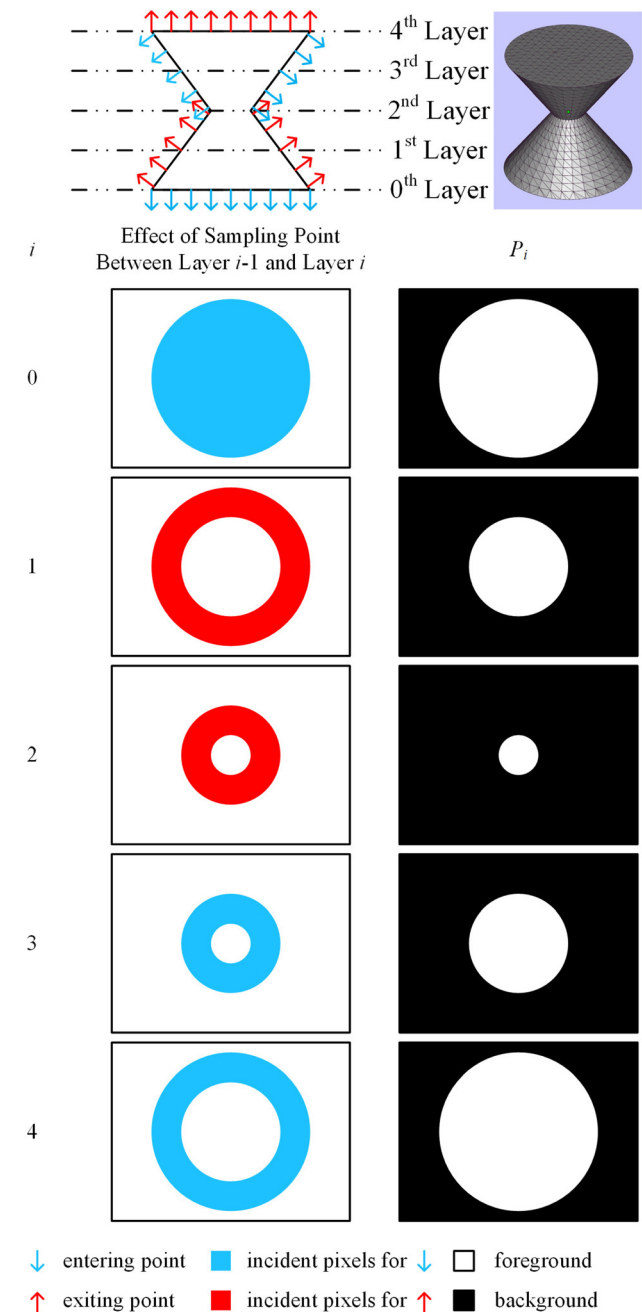


Fig. 3 Three-dimensional model used for image generation demonstration

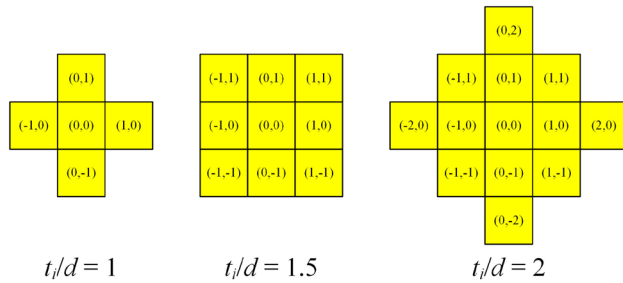


Fig. 4 Examples of structuring element

direct the laser scanning, and the final shrunk image P_n serves as the input for the mask projection.

The structuring element E_i is constructed according to the desired shrinkage magnitude t_i . If we use two integers u and v to represent the horizontal (column) and vertical (row) positions of a pixel, and assume that the origin of the structuring element is located at the center of pixel $(0, 0)$, then, all pixels that satisfy $\sqrt{u^2 + v^2} \leq (t_i/d)$ will be selected as members of structuring element, where d is the pixel width. Figure 4 shows several examples of structuring elements. Obviously, the shape represented by a structuring element is an approximation of a circular area with radius t_i .

In this application, all structuring elements generated are symmetric about the origin, so E_i and its reflection \bar{E}_i are identical. Therefore, based on the definition of dilation [25], the operation represented by Eq. (3) is equivalent to placing a circular disk with radius t_i on each inner boundary pixel (center to center), and setting all pixels whose centers are covered by these disks as background. The resultant image is the shrunk image P_i .

4.3 Error Analysis for Image Shrinkage Process

4.3.1 Error Source Identification. In order to analyze the error in the image shrinkage process, we define the following notations: ∂ is the boundary of a given area, \downarrow^* is the regularized solid offset by a negative distance, c^* is the complementary set of a given set, γ is the binary image for an area under a given resolution, π is the area enclosed by the inner image contour center

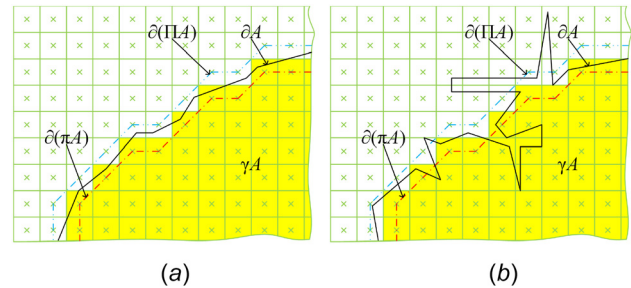


Fig. 6 Polygonal contour and image contour center chain: (a) an example case where assumption is valid and (b) an example case where assumption is invalid

chain of the image converted from a given area, Π is the area enclosed by the outer image contour center chain of the image converted from a given area, \surd is the image shrinkage by the proposed method, and its output is an area enclosed by the inner image contour center chain of the resultant image.

As discussed in the previous session, the regularized solid offset of a regular and nonempty set A by a negative distance t can be viewed as shifting a planar circular disk with radius t along ∂A , and removing the area swept by this disk from A . An example of solid offsetting an “L” shape is shown in the upper part of Fig. 5.

Compared with the regularized solid offset, the image shrinkage realized by the image processing approach (1) uses $\partial(\pi A)$ instead of ∂A as the trail to place the planar disk with radius t , and (2) applies circular disks with radius t only on the centers of boundary pixels rather than all points on the trail. The lower part of Fig. 5 demonstrates this process using the same L shape. We now analyze the errors resulting from these two sources, respectively.

4.3.2 Analysis for Error From Source I. Generally, the error from source I, i.e., the difference between ∂A and $\partial(\pi A)$, is unbounded. This is because any image has its finite resolution, and the smallest feature which the image is able to represent cannot go beyond a certain level. Similar to the spatial sampling accuracy discussed in Sec. 3.3, the sufficient condition of preserving planar features in an image is that the physical pixel size d has

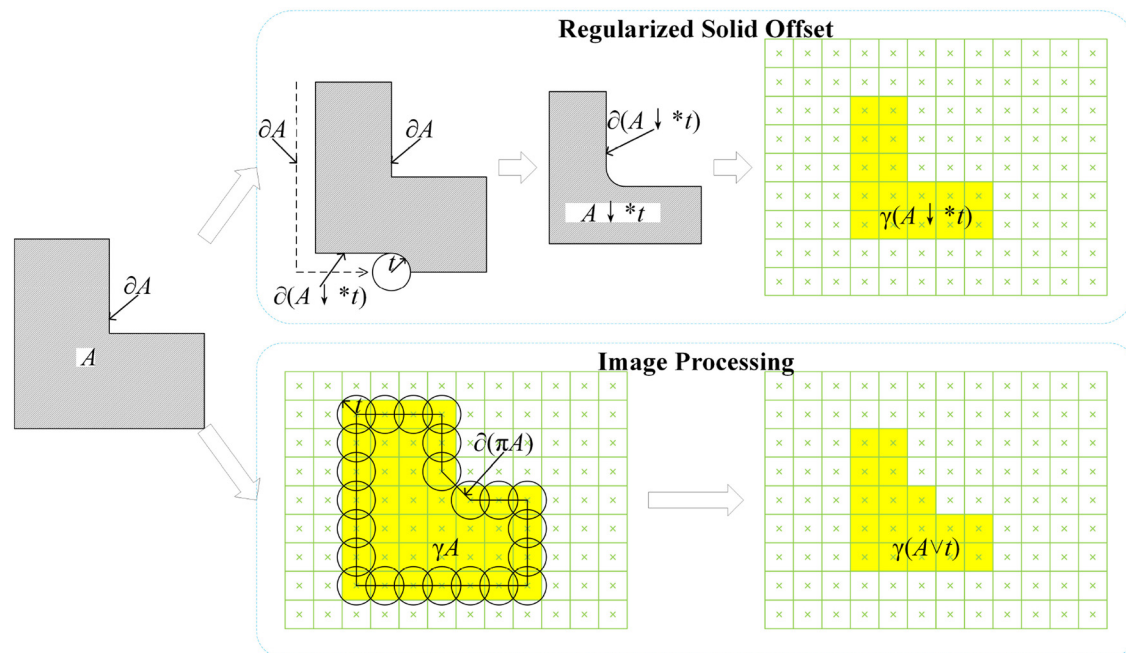


Fig. 5 Comparison of solid offset and image processing methods

to be smaller than $(\sqrt{2}/2)r_{\text{planar}}$, where r_{planar} is the smallest planar feature size to be preserved.

Following this, if we assume that all necessary planar features are able to be represented by the binary image with a given resolution, ∂A lies between inner image contour center chain $\partial(\pi A)$ and outer image contour center chain $\partial(\Pi A)$, as shown in Fig. 6(a). The following error analysis is based on this assumption. In Fig. 6(b), ∂A goes beyond $\partial(\pi A)$ and $\partial(\Pi A)$, and the sizes of corresponding features are smaller than $\sqrt{2}d$. Hence, it is not guaranteed that these features can be represented by the image.

According to the aforementioned assumption, we have: $\Pi A \supseteq A \supseteq \pi A$. For eight-connected image contours, the distance between any point p on $\partial(\pi A)$ and the complement of ΠA is: $d(p, c^*(\Pi A)) = d(p, \partial(\Pi A)) \leq d$. Hence, we have: $\Pi A \supseteq (\Pi A) \downarrow^* d$. According to the inclusion [21], we can derive $(\Pi A) \downarrow^* d \supseteq A \downarrow^* d$. To summarize, we have

$$A \supseteq \pi A \supseteq A \downarrow^* d \quad (4)$$

According to Eq. (4) and inclusion, we have: $A \downarrow^* t \supseteq (\pi A) \downarrow^* t \supseteq (A \downarrow^* d) \downarrow^* t$. Because $(A \downarrow^* d) \downarrow^* t = A \downarrow^* (d+t) = (A \downarrow^* t) \downarrow^* d$, we can derive

$$A \downarrow^* t \supseteq (\pi A) \downarrow^* t \supseteq (A \downarrow^* t) \downarrow^* d \quad (5)$$

Because $\gamma((A \downarrow^* t) \downarrow^* d)$ is at most “thinner” than $\gamma(A \downarrow^* t)$ by 1 pixel, $\gamma((\pi A) \downarrow^* t)$ is at most thinner than $\gamma(A \downarrow^* t)$ by 1 pixel as well. This concludes the error stemmed from source I.

4.3.3 Analysis for Error From Source II. The error from source II stems from only applying the circular disk with radius t on a finite number of points on $\partial(\pi A)$ rather than all points, and the set of pixel centers are a subset of $\partial(\pi A)$. Therefore, $\gamma((\pi A) \downarrow^* t)$ is thinner than $\gamma(A \vee t)$.

All eight neighbors of a given pixel can be categorized as either edge-connected neighbors or corner-connected neighbors. To simplify the discussion, we set the center of a given pixel p_i as origin $(0, 0)$, and $r = t/d$. We then evaluate the difference between two pixel sets that are set as background by two methods.

For the edge-connected case, let p_{i+1} locate at $(1, 0)$. By the solid offset method, the pixel (u, v) satisfies $(u-s)^2 + v^2 \leq r^2$ will be set as background, where $u, v \in \mathbb{Z}, s \in \mathbb{R}$, and $0 \leq s \leq 1$. Based on the proposed method, only the pixel (u, v) that satisfies $u^2 + v^2 \leq r^2$ or $(u-1)^2 + v^2 \leq r^2$ will be set as background. Compared with conducting solid offsetting on πA with a negative distance t , the pixel (u, v) which satisfies Eq. (6) may be missed being set as background

$$\begin{cases} u^2 + v^2 > r^2 \\ (u-1)^2 + v^2 > r^2 \\ (u-s)^2 + v^2 \leq r^2 \end{cases} \quad (6)$$

As no integer solution exists for Eq. (6), we can conclude that the proposed approach would not miss setting any pixel as background for the edge-connected case.

The corner-connected case is more complicated than the edge-connected case. We set p_{i+1} at $(1, 1)$, and follow the same fashion. The missed pixel (u, v) satisfies

$$\begin{cases} u^2 + v^2 > r^2 \\ (u-1)^2 + (v-1)^2 > r^2 \\ (u-s)^2 + (v-s)^2 \leq r^2 \end{cases} \quad (7)$$

From Eq. (7), we can derive $u+v=1$. As shown in Figs. 7(a) and 7(b), the pixel (u, v) whose center falls in the shaded area and on the line $u+v=1$ will be missed by the proposed method. It is obvious that when $r < (\sqrt{2}/2)$, the shaded area would not cover any pixel center, so the proposed method would not miss any pixel. An example is shown in Fig. 7(a). When $r < (\sqrt{2}/2)$, it

starts to miss pixels. The length of segment ab can be calculated as $|ab| = |ac| - |bc| = r - \sqrt{r^2 - (1/2)}$, and its maximum value is $|ab|_{\text{max}} = \sqrt{2}/2$ when $r < (\sqrt{2}/2)$. An observation is that the distance between any two neighboring pixel centers on the line $u+v=1$ is $\sqrt{2}$, so there exists at most one missed pixel center on segment ab , and the existence of missed pixel depends on the value of r . Therefore, we can conclude that the error stemmed from source II is 1 pixel. In other words, $\gamma((\pi A) \downarrow^* t)$ is at most thinner than $\gamma(A \vee t)$ by 1 pixel.

4.3.4 Conclusion for Overall Error. Because both $\gamma(A \downarrow^* t)$ and $\gamma(A \vee t)$ are at most “thicker” than $\gamma((\pi A) \downarrow^* t)$ by 1 pixel, we can also conclude that there is at most 1 pixel difference (can be either thinner or thicker) between images $\gamma(A \downarrow^* t)$ and $\gamma(A \vee t)$.

5 Efficiency Analysis and Implementation

The efficiency of the proposed algorithms for the hybrid SLA system has been analyzed from the time and the space complexity perspectives, respectively. The proposed algorithm was implemented using the c++ programming language with the Microsoft® Visual c++ compiler, and four 3D models were tested. The test results were compared with those generated by the traditional approach.

5.1 Efficiency Analysis

5.1.1 Time Complexity. The time complexity of the image-based slicing algorithm is analyzed as follows. Assume an STL format 3D model has τ triangular facets, reading vertex coordinates, and normal vectors for all facets takes $O(\tau)$ time. For each triangular facet, suppose that it has κ incident pixels on average, then calculating and saving the point cloud for the model take $O(\tau \times \kappa)$ time. Therefore, the time complexity for converting an STL file into the point cloud takes $O(\tau \times \kappa)$ time. The total number of sampling points, denoted as σ , equals to $\tau \times \kappa$, so sorting all sampling points according to their Z-coordinates takes $O(\sigma \log \sigma)$ time. To generate the binary image set for the model from the point cloud, each sampling point needs to be traversed, so the time complexity is $O(\sigma)$. Here, we assume that the number of sampling points σ is much larger than the required resolution and the total number of layers. Therefore, the overall time complexity for the proposed slicing algorithm is $O(\sigma \log \sigma)$.

The path planning process for a specific layer starts at identifying the seed pixel from a given image. The seed pixel identification needs to check each pixel, so the time complexity is $O(m \times n)$. For a given boundary pixel p_i , it needs to check all eight neighbors at the most to find the next boundary pixel p_{i+1} . If we assume that for a given image, there exists λ boundary pixels, then locating all boundary pixels takes $O(\lambda)$ time. Obviously, the total amount of boundary pixels λ is less than the image resolution $m \times n$, so the entire contour tracing process takes $O(m \times n)$ time. Since the number of pixels in a structuring element is much

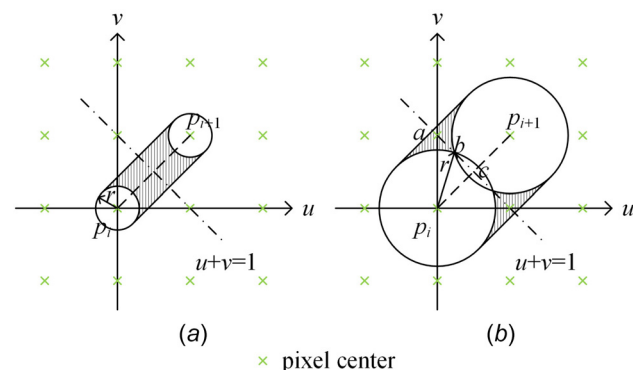


Fig. 7 Corner-connected case

Table 1 Processing time for tested models

Model	Bell	Hand	Ring knots	Turbine rotor
# <i>T</i>	210,500	109,834	33,730	124,336
Size (<i>L</i> × <i>W</i> × <i>H</i> , mm)	26.42 × 26.11 × 40.76	26.02 × 14.36 × 37.97	22.58 × 14.33 × 23.15	55.88 × 55.88 × 25.40
# <i>L</i>	407	379	231	253
<i>t_t</i> (s)	122.225	83.561	42.536	71.207
<i>t_p</i> (s)	9.644	7.449	4.540	7.270

#*T* is the number of triangles, and #*L* is the number of layers. *t_t* and *t_p* are the times for traditional method and proposed approach, respectively

smaller than the image resolution, constructing the structuring element takes $O(1)$ time. Therefore, generating one laser path and obtaining the resultant image take $O(\lambda)$ time. As λ is substantially less than $m \times n$, the time complexity of one round image shrinking is $O(m \times n)$.

5.1.2 Space Complexity. For the image-based slicing algorithm, as all σ sampling points need to be stored, the space complexity for the proposed slicing algorithm is $O(\sigma)$. The path planning process has to save the pixel value for each pixel, and thus its space complexity is $O(m \times n)$.

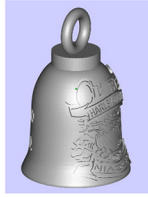
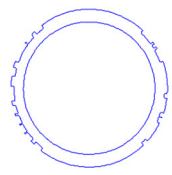
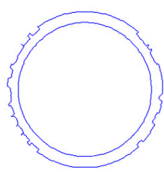
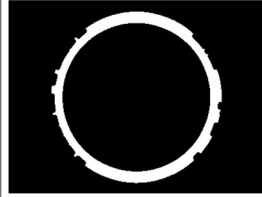






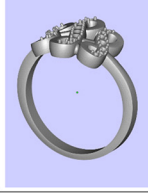





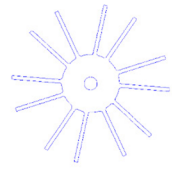
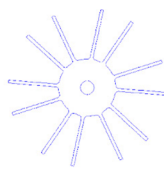
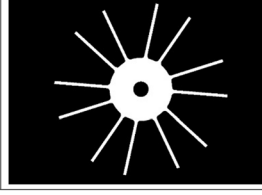
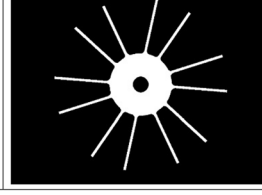
5.2 Implementation and Experimental Result. Both the proposed method and the traditional approach are implemented,

and Clipper [26], an open source library for clipping and offsetting lines and polygons, is adopted to implement the traditional offsetting method. The same set of 3D models (STL files) has been tested, and for each layer the outputs are a series of contours which serve as laser paths and an image for the mask projection. The test environment is as follows:

Sixty four bit Windows 10 Pro system laptop with Intel(R) Core(TM) i7-4600U, CPU @ 2.10 GHz 2.69 GHz and 8 GB RAM.

The layer thickness is set as 0.1 mm, the image resolution is 1024 × 768, and the physical size of the image is 80 × 60 mm. We conduct four rounds of offsetting (or image shrinking), and the offset distance is set as -0.078125 mm, which is equal to a single pixel width. Table 1 shows model information and processing

Table 2 The result comparison for tested models

Model	2 nd Laser Path by Traditional Approach	2 nd Laser Path by Proposed Method	Final Image for Mask Projection by Traditional Approach	Final Image for Mask Projection by Proposed Method
Bell 	150 th Layer 	150 th Layer 	150 th Layer 	150 th Layer 
Hand 	250 th Layer 	250 th Layer 	250 th Layer 	250 th Layer 
Ring Knot 	210 th Layer 	210 th Layer 	210 th Layer 	210 th Layer 
Turbine Rotor 	100 th Layer 	100 th Layer 	100 th Layer 	100 th Layer 

times for all four models. The output comparison for a specific layer is shown in Table 2. As can be seen from the tables, the proposed image-based slicing and path planning algorithm is ten times faster than the traditional analytical method without losing accuracy with respect to the printer resolution. However, it should be noted that the image-based algorithm may not find obvious benefit if the machine resolution is improved by more than ten times with the advance of machine and process development in the future.

6 Conclusion

The hybrid stereolithography process has the potential to substantially increase the system throughput without losing the fabrication quality. Geometry processing operations such as the contour slicing and the tool path planning are among the key procedures of the hybrid process. The traditional analytical approach suffers from multiple technical difficulties, including but not limited to, high time complexity and low reliability due to the challenge of maintaining geometry integrity. In this paper, we proposed a novel image-based numerical approach for slicing and tool path planning for the hybrid Stereolithography system. Such a numerical approach converts the traditional continuous domain (e.g., contour) into a discrete domain (e.g., image) by accounting for the limited hardware resolution. Therefore, the proposed numerical method is more configuration-specific than the traditional analytical approach. Meanwhile, because of the nature of parallel computation, the proposed image-based algorithm has the potential to be implemented on the graphics processing unit (GPU), and thus, it is able to achieve higher computational efficiency. Both the theoretical analysis and the experimental validation verified that the proposed approach can achieve higher computational efficiency than the traditional technique without losing the desired accuracy.

Acknowledgment

The authors gratefully appreciate the financial support from National Science Foundation (NSF) through CNS-1547167.

References

- [1] Gibson, I., Rosen, D. W., and Stucker, B., 2010, *Additive Manufacturing Technologies: Rapid Prototyping to Direct Digital Manufacturing*, Springer, New York.
- [2] Nakamoto, T., and Yamaguchi, K., 1996, "Consideration on the Producing of High Aspect Ratio Micro Parts Using UV Sensitive Photopolymer," 7th International Symposium on Micro Machine and Human Science (IEEE), Nagoya, Japan, Oct. 2–4, pp. 53–58.
- [3] Monneret, S., Loubere, V., and Corbel, S., 1999, "Microstereolithography Using a Dynamic Mask Generator and a Non-Coherent Visible Light Source," *Proc. SPIE*, **3680**, pp. 553–561.
- [4] Stampfl, J., Fouad, H., Seidler, S., Liska, R., Schwager, F., Woesz, A., and Fratzl, P., 2004, "Fabrication and Moulding of Cellular Materials by Rapid Prototyping," *Int. J. Mater. Prod. Technol.*, **21**(4), pp. 285–296.
- [5] Xu, G., Zhao, W., Tang, Y., and Lu, B., 2011, "Novel Stereolithography System for Small Size Objects," *Rapid Prototyping J.*, **12**(1), pp. 12–17.

- [6] Jiang, C.-P., 2010, "Accelerating Fabrication Speed in Two-Laser Beam Stereolithography System Using Adaptive Crosshatch Technique," *Int. J. Adv. Manuf. Technol.*, **50**(9–12), pp. 1003–1011.
- [7] Kang, H.-W., Park, J. H., and Cho, D.-W., 2012, "A Pixel Based Solidification Model for Projection Based Stereolithography Technology," *Sens. Actuators A*, **178**, pp. 223–229.
- [8] Bourell, D. L., Leu, M. C., and Rosen, D. W., 2009, "Roadmap for Additive Manufacturing: Identifying the Future of Freeform Processing," The Roadmap for Additive Manufacturing Workshop (RAM), Alexandria, VA, Mar. 30–31, pp. 1–102.
- [9] Zhou, C., Ye, H., and Zhang, F., 2015, "A Novel Low-Cost Stereolithography Process Based on Vector Scanning and Mask Projection for High-Accuracy, High-Speed, High-Throughput, and Large-Area Fabrication," *ASME J. Comput. Inf. Sci. Eng.*, **15**(1), p. 011003.
- [10] Hughes, J. F., van Dam, A., McGuire, M., Sklar, D. F., Foley, J. D., Feiner, S. K., and Akeley, K., 2013, *Computer Graphics: Principles and Practices*, Addison-Wesley, Upper Saddle River, NJ.
- [11] Choi, B. K., and Park, S. C., 1999, "A Pair-Wise Offset Algorithm for 2d Point-Sequence Curve," *Comput.-Aided Des.*, **31**(12), pp. 735–745.
- [12] Park, S. C., and Shin, H., 2002, "Polygonal Chain Intersection," *Comput. Graphics*, **26**(2), pp. 341–350.
- [13] Chen, X., and McMains, S., 2005, "Polygon Offsetting by Computing Winding Numbers," *ASME Paper No. DETC2005-85513*.
- [14] de Berg, M., Cheong, O., van Kreveld, M., and Overmars, M., 2008, *Computational Geometry: Algorithms and Application*, Springer, Berlin.
- [15] Rock, S. J., and Wozny, M. J., 1991, "Utilizing Topological Information to Increase Scan Vector Generation Efficiency," International Solid Freeform Fabrication Symposium (SFF), Austin, TX, Aug. 12–14, pp. 1–9.
- [16] Kwok, T.-H., Ye, H., Chen, Y., Zhou, C., and Xu, W., 2016, "Mass Customization: Reuse of Digital Slicing for Additive Manufacturing," *ASME Paper No. DETC2016-60140*.
- [17] Ye, H., Zhou, C., and Xu, W., 2016, "Mass Customization: Reuse of Topology Information to Accelerate Slicing Process for Additive Manufacturing," International Solid Freeform Fabrication Symposium (SFF), Austin, TX, Aug. 7–10, pp. 53–66.
- [18] Botsch, M., Kobbelt, L., Pauly, M., Alliez, P., and Lévy, B., 2010, *Polygon Mesh Processing*, A K Peters, Natick, MA.
- [19] Huang, P., Wang, C. C. L., and Chen, Y., 2013, "Intersection-Free and Topologically Faithful Slicing of Implicit Solid," *ASME J. Comput. Inf. Sci. Eng.*, **13**(2), p. 021009.
- [20] Stelldinger, P., Latecki, L. J., and Siqueira, M., 2007, "Topological Equivalence Between a 3d Object and the Reconstruction of Its Digital Image," *IEEE Trans. Pattern Anal. Mach. Intell.*, **29**(1), pp. 126–140.
- [21] Rossignac, J. R., and Requicha, A. A. G., 1986, "Offsetting Operations in Solid Modelling," *Comput. Aided Geom. Des.*, **3**(2), pp. 129–148.
- [22] Nadler, S. B., 1978, *Hyperspaces of Sets: A Text With Research Questions*, Marcel Dekker, New York.
- [23] Sonka, M., Hlavac, V., and Boyle, R., 2008, *Image Processing, Analysis, and Machine Vision*, Thomson Learning, Toronto, ON, Canada.
- [24] Engedy, I., and Horváth, G., 2009, "A Global, Camera-Based Mobile Robot Localization," 10th International Symposium of Hungarian Researchers on Computational Intelligence and Informatics (CINTI), Budapest, Nov. 12–14, pp. 217–228.
- [25] Gonzalez, R. C., and Woods, R. E., 2007, *Digital Image Processing*, Prentice Hall, Upper Saddle River, NJ.
- [26] Johnson, A., 2014, "Clipper," accessed Dec. 20, 2016, <http://www.angusj.com/delphi/clipper.php>
- [27] Huang, P., Wang, C. C. L., and Chen, Y., 2014, "Algorithms for layered manufacturing in image space," *Advances in Computers and Information in Engineering Research*, Vol. 1, ASME, New York, Chap. 15.
- [28] Sun, C., Fang, N., Wu, D. M., and Zhang, X., 2005, "Projection Micro-Stereolithography Using Digital Micro-Mirror Dynamic Mask," *Sens. Actuators A*, **121**(1), pp. 113–120.
- [29] Vatani, M., Rahimi, A. R., Brazandeh, F., and Nezhad, A. S., 2009, "An Enhanced Slicing Algorithm Using Nearest Distance Analysis for Layer Manufacturing," *Int. J. of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering*, **3**(1), pp. 74–79.