Intersection-Free and Topologically Faithful Slicing of Implicit Solid

Pu Huang[1] Charlie C. L. Wang[1,2*] Yong Chen[2]

[1]Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Hong Kong

[2]Epstein Department of Industrial and Systems Engineering, University of Southern California

Abstract

We present a robust and efficient approach to directly slicing implicit solids. Different from prior slicing techniques that reconstruct contours on the slicing plane by tracing the topology of intersected line segments, which is actually not robust, we generate contours by a topology guaranteed contour extraction on binary images sampled from given solids and a subsequent contour simplification algorithm which has the topology preserved and the geometric error controlled. The resultant contours are free of self-intersection, topologically faithful to the given r-regular solids and with shape error bounded. Therefore, correct objects can be fabricated from them by rapid prototyping. Moreover, since we do not need to generate the tessellated B-rep of given solids, the memory cost our approach is low – only the binary image and the finest contours on one particular slicing plane need to be stored in-core. Our method is general and can be applied to any implicit representations of solids.

Keywords: Direct slicing, Solid, Implicit representation, Self-intersection free, Topologically faithful

1 Introduction

Slicing CAD models is a crucial operation for generating tool paths in rapid prototyping. Prior slicing algorithms focus on computing the intersection curves between a model represented by polygonal meshes and a sequence of parallel planes, which becomes an unstable step for the whole procedures of rapid prototyping if the triangular meshes are self-intersected (or overlapped). In addition, more and more modeling approaches represent objects with complex structures by implicit solids since such representations are mathematically compact and robust. When conventional slicing methods are used, the implicit solids must be first tessellated into triangular meshes and then be intersected by slicing planes. However, generating a self-intersection free and topologically faithful polygonal model from an implicit solid is not easy (see ref. [1, 2] for detailed discussions). Specifically, the triangular models produced can have problems like gaps, degenerated triangles, overlapped facets, non-manifold entities and self-intersections. Using conventional slicing techniques to generate contours for rapid prototyping from such problematic triangular meshes will result in an incorrect object. For example, as shown in Figs.1 and 2, unexpected gaps are produced on the Buddha model fabricated by *Fused Deposition Modeling* (FDM). This gap is caused by the self-intersected polygonal model, which brings in inverse in/out membership classifications to some planar contours (see the contours on the plane with height=1.79 inch in Fig.2).

The topology of a fabricated model is usually required to be homeomorphic to the given solid. This is very important to applications such like biomedical engineering – e.g., a fabricated model with an incorrect topology may merge two tubes that should be separated into one, which is very dangerous for medical treatments (see Fig.3). Meanwhile, shape approximation errors between the extracted contours and the exact ones defined by intersecting the given solid with the slicing plane must also be controlled. To fabricate an object with high accuracy in a conventional way of tessellation, a massive number of triangles may be generated and storing them in-core will use up the memory of a computer system. This motivates our research to develop a direct slicing algorithm to generate self-intersection free and topologically faithful contours from a general implicit solid.

Problem Definition: For a given implicit solid $H = \{\mathbf{p}|f(\mathbf{p}) \leq 0, \forall \mathbf{p} \in \Re^3\}$ and a slicing plane P, a contour $C = M \cap P$ is defined as a *topologically faithful* contour when M is a surface r-homeomorphic to the exact surface boundary, ∂H, of H. In this paper, we compute contours which are

1. topologically faithful when the given solid H is r-regular,
2. self-intersection free, and
3. with the shape approximation error minimized.

Our approach consists of three major steps. Firstly, a binary image B of an r-regular solid H is sampled on the slicing plane P, where an appropriately selected sampling distance r' (with its bound relating to the value r) ensures that the contour $\bar{C}^0$ generated from B is homeomorphic to C. Secondly, $\bar{C}^0$ is iteratively smoothed into $\bar{C}^m$ by a constrained Laplacian operator that prevents topological changes and self-intersections. Lastly, a constrained contour simplification is applied to simplify $\bar{C}^m$ into a contour $\tilde{C}$ which has fewer line segments and $\tilde{C}$ satisfies the three requirements previously given in the problem definition. Proofs for the correctness of $\tilde{C}$ are also given in this paper. A flowchart of our direct slicing approach is presented in Fig.4.

*Corresponding Author; Email: cwang@mae.cuhk.edu.hk

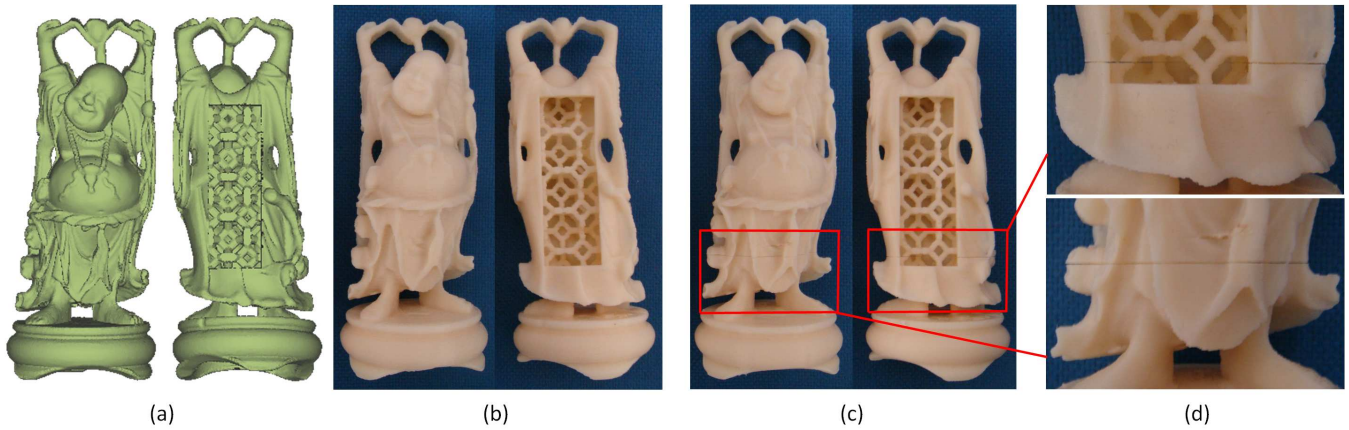

Figure 1: Incorrect contours generated by slicing a given model will produce a model with unwanted gaps (and/or membranes) in rapid prototyping: (a) the given Buddha model in implicit representation (actually Layered Depth-Normal Images (LDNI) [3]), (b) the correct model fabricated from contours generated by our approach, (c) the problematic object fabricated by slicing the locally self-intersected polygonal model extracted from (a) using a variation [4] of dual-contouring [5], and (d) a zoom-view of the incorrect layer. The models are fabricated by FDM.

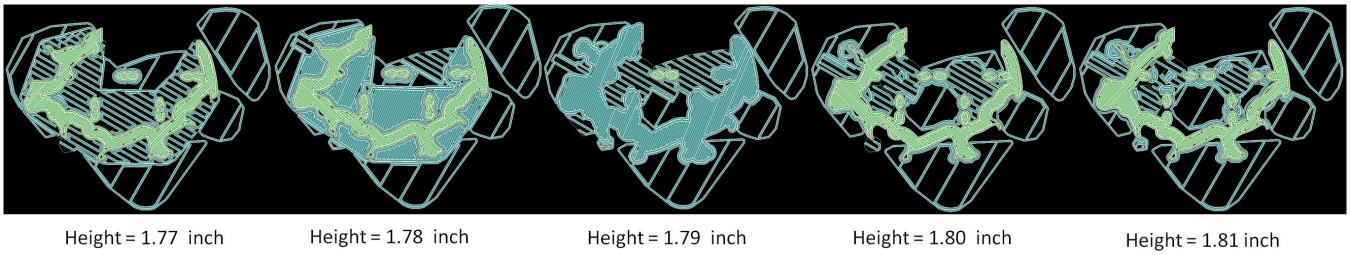

Figure 2: Tool path of five consecutive layers generated in Insight^{TM} version 7.0 by slicing a polygonal model extracted from the implicit Buddha solid given in Fig.1(a) by [4]. Each layer is in thickness of 0.01 inch, and the regions of the Buddha model and the supporting structures are displayed in green and cyan respectively. Pay attention that an incorrect in/out membership classification is given on the layer with height=1.79 inch, which is caused by a local self-intersection on the polygonal model. As a result, the inside of the Buddha model is filled with supporting structure by mistake.

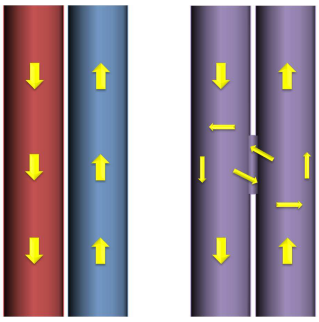

Figure 3: A example of tubes in biomedical engineering: (left) the expected tubes configuration, (right) the manufactured model with incorrect topology merging two tubes that should be separated – this is very dangerous for medical treatments.

The technical contribution of our approach is two-fold.

- For an r-regular solid represented by the implicit indicator function (i.e., '$-$' for inside and '$+$' for outside), the contours generated by our approach are topologically faithful, self-intersection free and with the shape approximation error controlled. Therefore, the topology and shape of the final model fabricated by using these contours are preserved. Rigorous proofs of these good properties are given.
- As a direct slicing approach, it is efficient in computation and memory-usage since only the information on a particular slicing plane is involved, which is different from those techniques which first polygonize a given solid into B-reps and then generate contours from the B-reps (ref. [6]).

Solids represented by several implicit representations, including Layered Depth-Normal Images (LDNI) [3, 4], Binary Space Partition (BSP) [34] and Adaptively Supported Radial Basis Functions (RBF) [35], are tested in this paper to demonstrate the functionality of our approach.

The remainder of the paper is organized as follows. After reviewing the related work in Section 2, Section 3 presents the method to generate the first topologically faithful contours. The smoothing technique presented in Section 4 is applied to the contours to further improve their fairness while preventing self-intersections. A variational shape approximation algorithm is introduced in Section 5 to generate

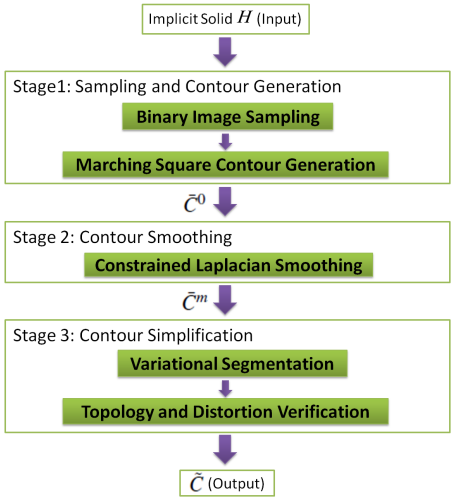

Figure 4: A flowchart of our direct slicing approach.

topology-preserved and error-bounded contours which have fewer line segments from the smoothed ones. The experimental results are discussed in Section 6 and our paper ends with the conclusion section.

2 Related Work

The basic problem for directly slicing an implicit solid is to compute the intersection between the solid and a plane. According to a review in [8], the methods can be classified into two categories: analytical and numerical. Analytical methods find precise intersection points by solving polynomial equations derived from implicitization [9]. However, these methods can only be applied to algebraic surfaces and the computing speed is generally slow. Numerical strategies like subdivision [10] and marching [11,12] do not require precise analytical representation of surface boundary. Subdivision methods intersect a tessellated piecewise linear approximation of the given implicit surface with a plane, which have the problem that some small intersection loops will be missed if the subdivision stops at an improper level. Marching based methods (e.g. [13]) always start from an initial point, and then proceed to march along a curve, but they suffer from robustness problem at critical points of contours (i.e., the point where two loops join into one). Tracing based contour generation algorithms like [13] may also miss some small loops. In a follow-up work in [14], the authors divide the MLS surfaces into several slabs with each slab having the same topology, then use a tracing strategy to generate contours for each slab separately. However, unlike our approach, their method is specialized for *Moving Least-square Surfaces* (MLS) and they do not provide rigorous proof of the self-intersection free property and the topological faithfulness as our approach. Recently, their work is further extended to use *Point Set Surface* (PSS) to compute spline NC paths for high-speed machining (ref. [15]).

In the computer graphics community, divide-and-conquer algorithms like Marching Cubes (MC) [16, 17] and Dual Contouring (DC) [5] have been developed to polygonize implicitly defined scalar fields. To generate a smooth surface, real values are defined on grid nodes. In [18], authors introduced a compact method for extracting smooth isosurfaces on grids with only binary values stored on nodes, which inspires the constrained smoothing step in our approach (Section 4). The theoretical work presented in [19] provides the criteria of topological equivalence between a 3D object surface and the model reconstructed from discrete binary samples stored on grid nodes. We apply these criteria to our approach to govern the steps of binary image sampling and contour extraction (see Section 3).

The last step of our algorithm needs to simplify a contour into a new one with fewer line segments. This relates to mesh simplification work in literature, where some approaches collapse mesh elements greedily (e.g., [20–23]) and others cluster faces into several proxies like planes or quadratic surface patches (e.g., [24–32]). Among these methods, the *Variational Shape Approximation* (VSA) method in [32] has the global distortion error minimized. Our contour simplification algorithm follows the strategy of VSA and is modified to ensure that the simplified contour is intersection-free and homeomorphic to the exact contour.

3 Sampling and Contour Generation

We start to analyze the appropriate sampling rate to guarantee the extraction of topologically faithful contours by briefly reviewing the relevant definitions and theorems given in [19].

Definition 1. A solid $H \subset \Re^3$ is called r-regular if, for each point $\mathbf{p} \in \partial H$, there exist two osculating open balls of radius r to ∂H at $\mathbf{p}$ such that one lies entirely in H and the other lies entirely out.

Theorem 1. For an r-regular solid H, the boundary surface, M, generated by a *topology preserving method* on cubic grids is r-homeomorphic to the exact surface boundary, ∂H, if the cube width r' of grids satisfies $\sqrt{3}r' < r$.

The above definition and theorem are derived from definition 1 and theorem 16 of [19], which is the foundation of our binary image sampling and topologically faithful contouring. Note that r' here is different from r' used in [19]. Details about the *topology preserving methods* can also be found in [19].

3.1 Sampling

To generate topologically faithful contours so that a model homeomorphic to the exact boundary of H can be fabricated from them, we sample the 2D solid, $H \cap P$, into a binary image I and then generate a contour $\bar{C}^0$ from I. The following theorem is derived to guarantee that $\bar{C}^0$ is topologically faithful.

Proposition 1. For an r-regular solid H, the contour generated by a *topology preserving method* on square grids is topologically faithful if the width r' of the squares satisfies $\sqrt{3}r' < r$.

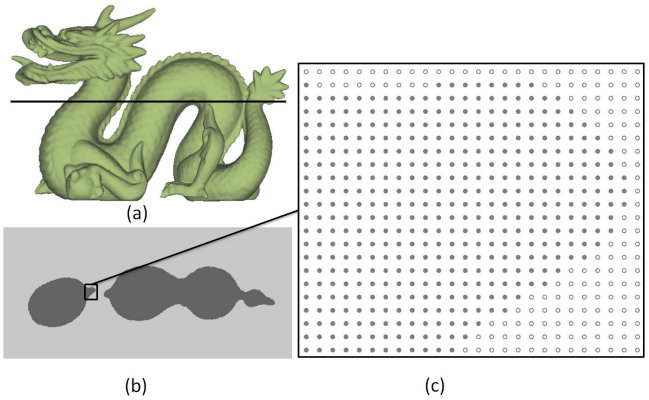

Figure 5: A binary image sampled from a Dragon solid: (a) the solid H is intersected by a slicing plane P, (b) the region of $\bar{H} = H \cap P$ on the binary image, and (c) a zoom-view of the binary image, where the nodes inside H are displayed in solid dots while outside nodes are shown in hollow.

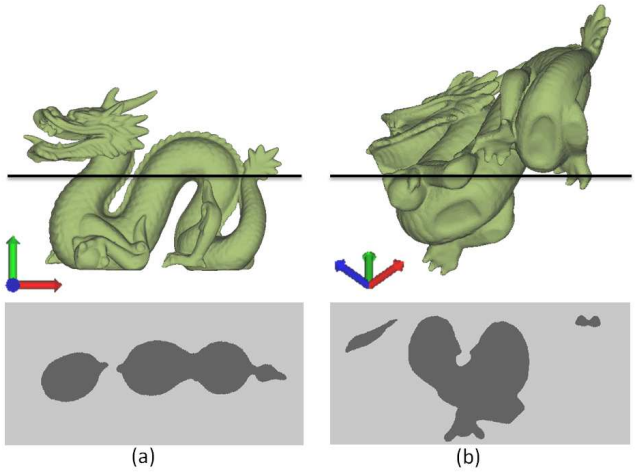

Figure 6: Binary image generation of slicing in different orientations: (left) with slicing direction $(0, 1, 0)$ and (right) slicing along $(1, 1, 1)$. The bottom row shows the binary images generated on example slicing planes.

Proof. Without loss of generality, we can assume that the slicing plane P overlaps the boundary of a layer of 3D grids used in Theorem 1. Moreover, the square grids are the boundary of these 3D grids. Therefore, when $\sqrt{3}r' < r$ is given on the planar grids, the criterion given in Theorem 1 is also satisfied.

□

Following this proposition, we sample a binary image I from $\bar{H}$ on the slicing plane P with the pixel distance r'. We first rotate the given solid H into the coordinate system with x-o-y plane parallel to the slicing plane P. Then, the dimension of the binary image can be determined by the intersection between P and the bounding box of H. The resolution is defined according to the value of r'. Fig.5 shows an example of the binary image sampled from a solid of Dragon model. Note that the sampling of binary image is not limited to the planes perpendicular to the major axes. As long as the in/out membership tests can be efficiently conducted on the given solid H, we can generate the binary image on a slicing plane in any orientation. Fig.6 demonstrates the binary images generated by the slicing planes in different orientations.

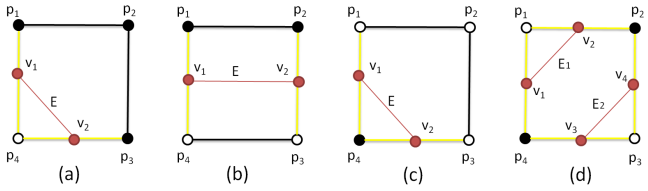

Figure 7: Lookup table for the marching square method with topology preserved. Sticks are in yellow. Sampling nodes inside the solid H are shown in black while the outside nodes are displayed in white. The contour edges linking sticks are labelled as E.

3.2 Topologically faithful contouring

After obtaining a binary image, the marching square method introduced in [16] is used to generate an approximate contour $\hat{C}^0$ that is topologically faithful. Defining the edge on a square grid with different in/out status on its two endpoints as a *stick*, the contour $\hat{C}^0$ can be formed by the line segments linking the middle points of sticks in all grids. Note that, in the rest of this paper, endpoints are not included when we refer to a *stick* (i.e., it is defined on an open interval). Fig. 7 shows the lookup table we used in marching square method to construct contour edges.

Proposition 2. The contour generated by using the lookup table shown in Fig.7 is topologically faithful.

Proof. The proof is straightforward and quite similar to that of Proposition 1. As the binary image I is sampled at a rate according to Proposition 1 and the planar square grids are considered as the boundary of 3D grids, the only topologically ambiguous configuration shown in the lookup table (i.e., Fig.7(d)) is derived by the topology preserving contouring method (e.g., Ball Union in [19]). Note that, ambiguity of the configuration shown in Fig.7(d) comes from that the contours in this square could be either 1) two edges $E_1 = (v_1, v_2)$ and $E_2 = (v_3, v_4)$ or 2) another two edge $E_1 = (v_1, v_3)$ and $E_2 = (v_2, v_4)$.

Specifically, the contour reconstructed in this way on a slicing plane P can also be considered as using the plane P to intersect a discrete surface of ∂H generated by the Ball Union method presented in [19]. Its topology in each square is the same as what we list in Fig.7.

□

3.3 r-Regularity and Accuracy in Rapid Prototyping

The contour generated by the above method is ensured to be on a surface homeomorphic to the boundary of an r-regular solid H. Although not all implicit solids are r-regular, this assumption is reasonable to the models to be fabricated by *rapid prototyping* (RP). The value of r actually relates

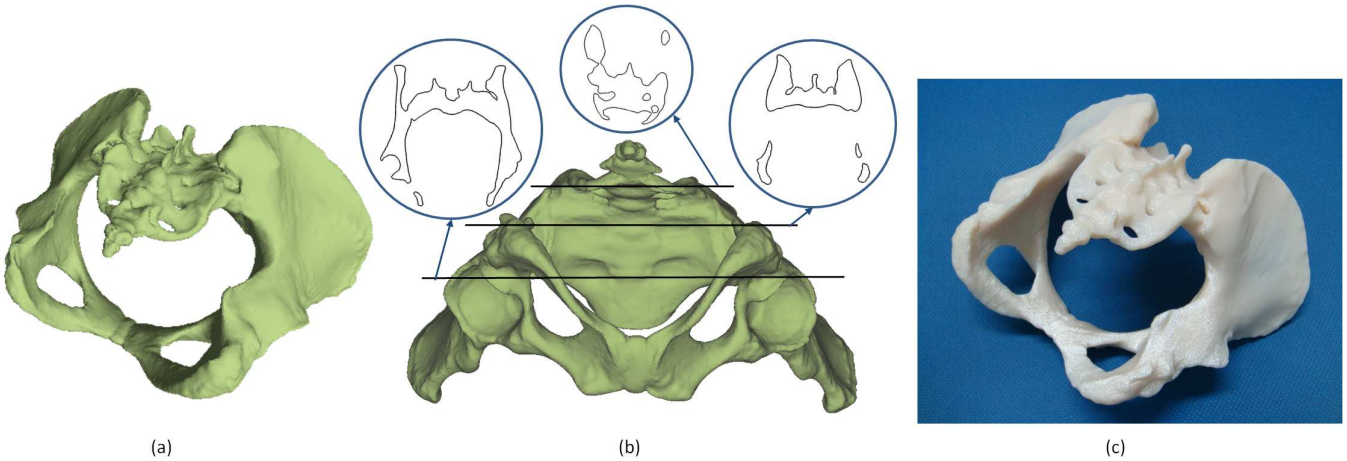

Figure 8: An example model fabricated from the contours generated by our method with $r' = 3.93 \times 10^{-3}$: (a) the given Donna model in the LDNI representation, (b) the sliced contours of respective layers at 2.30, 3.00 and 3.66 inch heights, and (c) the resultant model fabricated by FDM.

to the smallest component that can be fabricated by an RP machine (e.g., the diameter of plastic filaments on an FDM machine as well as the finest position that can be provided by motion controller). Most commercial *x-y* positioning systems used in rapid prototyping can achieve the precision of 10^{-3} inch. The diameter of plastic filaments is usually greater than 10^{-2}. Moreover, for an implicit solid represented by LDNI [3, 4], the accuracy of the solid is also limited by the resolution of LDNI. It is meaningless to make the grid width of binary image smaller than that of LDNI. Therefore, take the Donna model shown in Fig. 8 as an example, we choose $r' = 4.00 \times 10^{-3}$ to generate the binary images, slightly larger than that of LDNI (3.93×10^{-3}). Usually, selecting r' with a value no larger than 5×10^{-3} is good enough for models fabricated by FDM. For models with very small values of r, our algorithm needs a grid with very high resolution which could be a problem of computer memory. In this case, solid models need to be processed to r-regular with a larger r by using the techniques like morphological operators.

4 Constrained Smoothing

Since line segments on the contour $\bar{C}^0$ are generated by linking the middle point of sticks, there is no self-intersection on $\bar{C}^0$. However, the shape of $\bar{C}^0$ is not smooth (e.g., the contour shown in Fig. 10(a)), so, we apply a Laplacian operator based smoothing technique to improve it. The advantage of Laplacian smoothing is its efficiency and stability, but the major drawback is that the unwanted shrinkage always occurs when it is iteratively applied to a closed shape (in 2D or 3D). A constrained Laplacian smoothing is developed here, which intrinsically solves the shrinkage problem since we guarantee to generate topologically faithful contours. In other words, the smoothed contours cannot violate the in/out status of any sampling node on the binary image I. To ensure that, a good strategy is to constrain each vertex $\mathbf{v}_i$ on the contour so that it must stay on the stick on which it initially lies. In addition, by applying this '*sliding-on-stick*' strategy, we can guarantee that the resultant contours are self-intersection free.

Proposition 3. When moving the vertices on a contour, no intersection occurs if the vertices are only moved on the sticks holding them.

Proof. For a vertex $\mathbf{v}_i$ that is generated from the stick t_i, sliding it on t_i can only bring itself to a new position belonging to the point set $\{\mathbf{p} | \mathbf{p} = \alpha \mathbf{p}_i^s + (1-\alpha)\mathbf{p}_i^e), \alpha \in (0,1)\}$, where $\mathbf{p}_i^s$ and $\mathbf{p}_i^e$ are the two endpoints of t_i respectively. For any contour edge E, if we allow its two vertices $\mathbf{v}_i$ and $\mathbf{v}_j$ to slide only on their respective sticks t_i and t_j, E can sweep out a range Φ which contains any of its possible occurrence position.

For the contour edges with $\mathbf{v}_i$ and $\mathbf{v}_j$ located on two adjacent sticks t_i and t_j, the region Φ is a triangular region formed by the endpoints of t_i and t_j excluding the boundaries not overlapped with t_i or t_j (e.g., $\mathbf{p}_1\mathbf{p}_3$ and $\mathbf{p}_4$ in Fig.9(a)). For the contour edges with $\mathbf{v}_i$ and $\mathbf{v}_j$ on two opposite sticks (e.g., Fig.9(b)), Φ is a square excluding the boundaries not overlapped with t_i or t_j (e.g., $\mathbf{p}_1\mathbf{p}_2$ and $\mathbf{p}_3\mathbf{p}_4$ in Fig.9(b)). Since the sweeping envelopes of the edges on the contour do not have any overlap with each other, the proposition is thus proved.

□

Proposition 4. Deforming a topologically faithful contour by moving its vertices only on the sticks holding them will generate a contour which is still topologically faithful.

Proof. The proof of this proposition is straightforward. First, deforming a contour in this way will not change the 'in'-'out' status of any samples on the binary image I since the contour is not moved across any of the samples. Second, self-intersections will not be generated during such a kind of deformation (see Proposition 3).

□

The constrained smoothing technique introduced in [18] ensures every vertex sliding on its stick by projecting the dis-

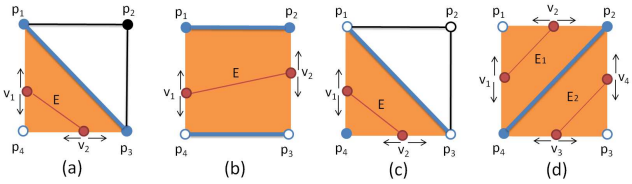

Figure 9: The sweeping envelopes for contour edges in grids with different configurations. The region Φ is displayed in orange and the excluded vertices and edges are displayed in blue.

placed vertex back to the nearest point of its stick. However, some vertex may eventually have orthogonal displacement to its stick and the contour is stuck in an sub-optimal shape (see Fig. 10(e)). We observed that for each contour vertex $\mathbf{v_i}$, its two adjacent vertices $\mathbf{v_{i-1}}$ and $\mathbf{v_{i+1}}$ have only 5 configurations for their position on different sticks with considering rotational symmetry (see Fig. 11). For all these configurations, the line segment connecting $\mathbf{v_{i-1}}$ and $\mathbf{v_{i+1}}$ intersect with either the stick holding $\mathbf{v_i}$ (configuration (b), (c), (d) and (e)) or an ending point of this stick (configuration (a)). Based on this observation, we investigate a sliding-based constrained smoothing without using projection. On each vertex $\mathbf{v_i}$, it is performed in two steps.

1. Calculating the intersection point $\mathbf{v_{int}}$ between the line segment connecting $\mathbf{v_{i-1}}$ and $\mathbf{v_{i+1}}$ and the stick griping $\mathbf{v_i}$ with its two ending points counted.

2. Moving $\mathbf{v_i}$ in the ratio of τ towards $\mathbf{v_{int}}$

$$\mathbf{v_i}^{new} = \mathbf{v_i} + \tau(\mathbf{v_{int}} - \mathbf{v_i})$$

   where $\tau = 0.4$ is selected to balance the efficiency and the stability of computation.

The contour $\bar{C}^0$ can be smoothed into $\bar{C}^m$ by repeatedly applying these two steps to all vertices until the average movement of vertices in an iteration is less than 10^{-3} of r'. Figs.10(b), (c) and (d) give a comparison of the smoothing results between sliding-based, projection-based constrained smoothing and ordinary Laplacian smoothing. Besides, it is obvious that $\bar{C}^m$ is homeomorphic to $\bar{C}^0$ since no intersection occurs during this contour evolution.

5 Contour Simplification

The smoothed contour, $\bar{C}^m$, usually provides a very good shape approximation of the exact contour generated by $\partial H \cap P$. However, to ensure the topological faithfulness, a relative small value of r' may be selected for a model with large dimensions. This leads to contours with a lot of very short line segments, which significantly increase the memory cost. The situation becomes more serious if the software controlling the RP machine does not run in an out-of-core manner (i.e., loading the contours for all layers from the contour file at the same time). Moreover, using too many small line segments

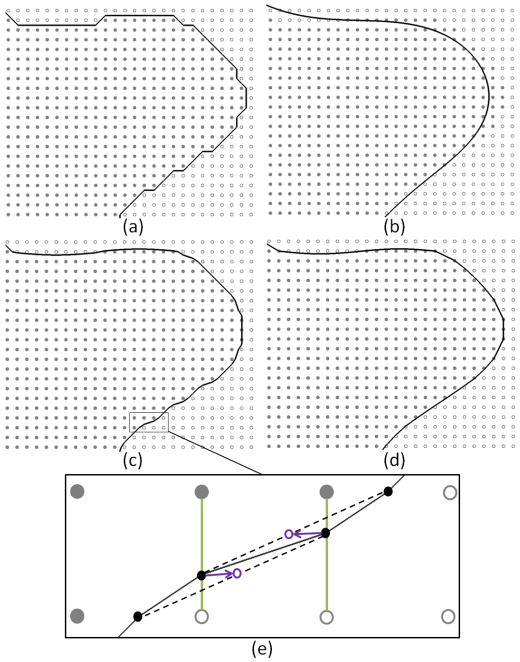

Figure 10: A comparison among different smoothing strategies on the contour generated for the binary image region shown in Fig. 5(c): (a) the contour reconstructed by the topology preserving marching square method, (b) the shrinking contour after ordinary Laplacian smoothing, (c) the resultant contour after projection-based constrained smoothing, (d) the resultant contour after sliding-based constrained smoothing, and (e) the zoom-in view of contour vertices stuck in sub-optimal shape.

to represent the contours will dramatically decrease the efficiency of subsequent processing steps in RP like generation of supporting structure and tool-path planning. Based on our observation, in the smoothed contours, there are always several successive edges lying almost in the same straight line, which implies these edges can be simplified into one single edge with little distortion error introduced. Therefore, a contour simplification algorithm preserving topology and shape approximation error is investigated in this section to further improve the topologically faithful contours for slicing implicit solids.

5.1 Variational segmentation

The variational shape approximation approach [32] imitates the well known Lloyd's algorithm [33] to cluster mesh entities into several regions, and for each region, it uses a planar proxy to approximate the whole region. Lloyd's algorithm based relaxation procedure is employed to minimize the global shape distortion error. We adopt their basic idea and develop our constrained simplification algorithm for 2D contours.

For any contour, at the very beginning, we randomly select n edges $\{E_i\}$ as seeds which will be used to start growing a cluster. The number of clusters, n, can be selected to be proportional to the total number of edges on this contour ($\frac{1}{\alpha}$ with

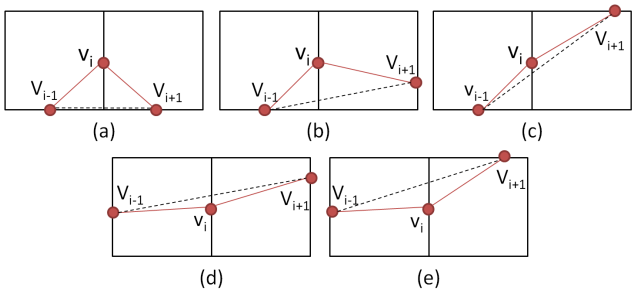

Figure 11: All the five configurations for the position of $\mathbf{v_{i-1}}$ and $\mathbf{v_{i+1}}$ on different sticks with considering rotational symmetry.

a positive integer α as clustering ratio). Here, each proxy is a line defined by a point $\mathbf{x}_i$ on the line and a normal vector $\mathbf{n}_i$ perpendicular to the direction of the line. The proxies, Q_i, are initialized by the seed edges.

We need to grow the proxies on the contour simultaneously and build n clusters minimizing the shape approximation error. For each seed edge E_i, we insert its two adjacent edges, E_j and E_k, into a minimal queue, Υ. The queue is keyed by the distortion error presented on the edges according to a particular proxy (e.g., the edges E_j and E_k inserted according to the proxy Q_i have the weights, $D(E_j, Q_i)$ and $D(E_k, Q_i)$, in Υ). Note that it is possible to have an edge inserted in the queue more than once (i.e., by different proxies adjacent to the edge). Here, the distortion error is measured by L^2 norm. Generally, the L^2 error metric for any region R and its proxy Q is defined as

$$L^2(R, Q) = \int \int_{\mathbf{x}\in R} \|\mathbf{x} - \Pi_Q(\mathbf{x})\|^2 d\mathbf{x} \tag{1}$$

where $\Pi_Q(\mathbf{x})$ means the orthogonal projection of the argument on the proxy Q. The L^2 distortion error $D(E, Q)$ between an edge E and the linear proxy Q can be evaluated by

$$D(E, Q) = \frac{1}{3}(d_0^2 + d_1^2 + d_0 d_1)\|E\| \tag{2}$$

where d_0 and d_1 are the orthogonal distance from two endpoints of E to the line defined on Q and $\|E\|$ represents the length of edge E.

The growing of clusters is performed by repeatedly removing the edge from the top of Υ (i.e., the edge with the smallest distortion error). For each edge E_t removed from Υ, we check if it has been assigned to a proxy. If not, we assign it to the proxy Q_p which is used to evaluate its distortion error, $D(E_t, Q_p)$, in Υ. Otherwise, no operation is given according to E_t. After the edge E_t is assigned to a proxy Q_p, the two edges E_l and E_r adjacent to E_t are inserted into Υ according to the weights, $D(E_l, Q_p)$ and $D(E_r, Q_p)$, if they have not been assigned to any proxy. The removing and inserting operations will not stop until Υ becomes empty, i.e., when every edge has been assigned to a proxy. This relaxation based clustering process always provides connected and non-overlapped segmentations on a contour.

After obtaining an n-clustering result from a set of seed edges, we need to update each proxy Q_i in order to minimize the distortion error between Q_i and its corresponding region R_i (the cluster). We update the linear proxy Q_i by recomputing its normal direction $\mathbf{n}_i$ and the site point $\mathbf{x}_i$ where the line passes through. $\mathbf{x}_i$ is simply the barycenter of its corresponding region R_i, which is given by

$$\mathbf{x}_i = \frac{\sum_{E\in R_i} \|E\|(\mathbf{v}_s + \mathbf{v}_e)}{2\sum_{E\in R_i} \|E\|} \tag{3}$$

where $\mathbf{v}_s$ and $\mathbf{v}_e$ are the two endpoints of an edge E. $\mathbf{n}_i$ could be determined by computing the eigenvector corresponding to the smallest eigenvalue of the covariance matrix of R_i. The covariance matrix M_i of R_i can be calculated by

$$M_i = \sum_{E\in R_i} \|E\|(\mathbf{ACA}^T + \mathbf{v}_s\mathbf{v}_s{}^T + \mathbf{v}_s\mathbf{v}_b{}^T + \mathbf{v}_b\mathbf{v}_s{}^T) - \mathbf{x}_i\mathbf{x}_i^T \sum_{E\in R_i} \|E\| \tag{4}$$

where

$$\mathbf{A} = \begin{bmatrix} \mathbf{v}_e - \mathbf{v}_s & \mathbf{0} \end{bmatrix}, \mathbf{C} = \begin{bmatrix} \frac{1}{3} & 0 \\ 0 & 0 \end{bmatrix} \text{ and } \mathbf{v}_b = \tfrac{1}{2}(\mathbf{v}_e - \mathbf{v}_s).$$

After updating all n proxies, we search a new seed edge E_i in each region R_i which has the smallest distortion error according to the new proxy. Then, a new segmentation is computed starting from these new seed edges.

After applying this growing-updating process for several iterations (e.g., 20 iterations always can provide satisfactory results), the whole contour has been successfully segmented into n regions and the overall distortion error has been minimized (see Fig.12(a) for an example).

5.2 Topology and distortion verification

After performing the segmentation of a smoothed contour $\bar{C}^m$, the simplest way to generate a simplified contour is to replace the edges on $\bar{C}^m$ by n edges where each links the starting and the ending points of a region R_i. However, simplifying contours in this way will make some sample points on the binary image I which are originally 'inside' the region, $H \cap P$, become 'outside', or vice versa – i.e., topological faithfulness is not preserved. Moreover, as shown in Fig.13, intersections and degenerate contours can be generated on the contours which are intersection-free before the simplification. Another important issue we concern about is the bound of distortion error introduced by simplification. As shown below, we investigate a novel verification procedure to solve both the topology faithfulness and distortion error bound problems together. We notice that every vertex is guaranteed to be still on its corresponding stick after smoothing. In other words, the variational clustering actually starts from a topologically faithful and intersection-free contour. The verification procedure is based on this assumption of the input contour.

Proposition 5. For a region on the contour defined by a sequence of connected vertices $\{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_n\}$, it can be simplified into a single edge $\bar{E}$ connecting $\mathbf{v}_1$ and $\mathbf{v}_n$ by sliding its $(n-2)$ internal vertices $\{\mathbf{v}_2, \mathbf{v}_3, \cdots, \mathbf{v}_{n-1}\}$ on their respective sticks $\{t_2, t_3, \cdots, t_{n-1}\}$ if and only if the resultant edge $\bar{E}$ intersects all of the sticks: $\{t_2, t_3, \cdots, t_{n-1}\}$.

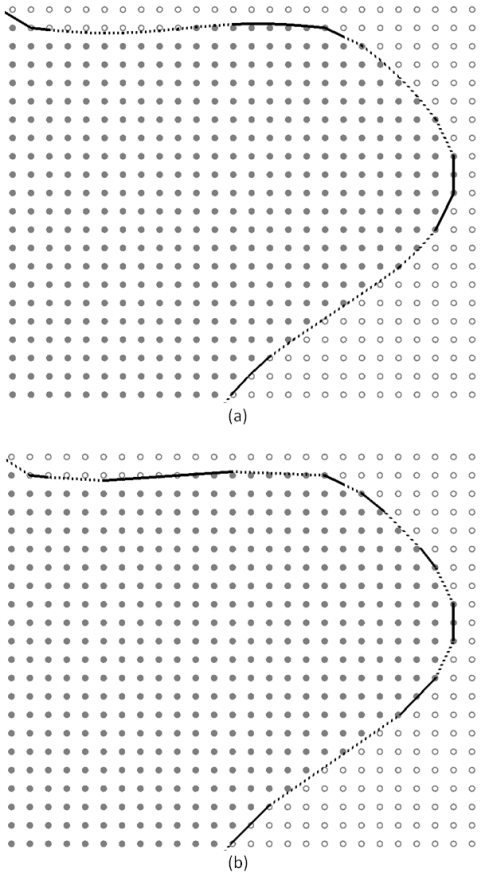

Figure 12: An illustration of contour simplification for the smoothed contour shown in Fig.10(d): (a) the variational clustering result on the contour with different line type representing different regions, and (b) the final simplified contour after topology and distortion verification.

Proof. See Fig. 14 for an illustration.

(1) *Sufficiency*: Suppose the resultant edge $\bar{E}$ intersects all of the sticks $\{t_2, t_3, \cdots, t_{n-1}\}$, the intersection point $\mathbf{v}'_k$ between $\bar{E}$ and t_k is on both $\bar{E}$ and t_k. Since both $\mathbf{v}_k$ and $\mathbf{v}'_k$ is on t_k, it is obvious that $\mathbf{v}_k$ can move to $\mathbf{v}'_k$ just by sliding on t_k. Hence, the whole contour region can be simplified into $\bar{E}$ by sliding the $n-2$ internal vertices $\{\mathbf{v}_2, \mathbf{v}_3, \cdots, \mathbf{v}_{n-1}\}$ on their respective sticks $\{t_2, t_3, \cdots, t_{n-1}\}$.

(2) *Necessity*: Suppose that the contour region can be simplified into a single edge $\bar{E}$ by sliding the $n-2$ internal vertices $\{\mathbf{v}_2, \mathbf{v}_3, \cdots, \mathbf{v}_{n-1}\}$ on $\{t_2, t_3, \cdots, t_{n-1}\}$, each $\mathbf{v}_k$ on t_k ($k \in [2, n-1]$) can have a corresponding point $\mathbf{v}'_k$ projected on the simplified segment $\bar{E}$ by sliding $\mathbf{v}_k$ on t_k. $\mathbf{v}'_k$ is on both t_k and $\bar{E}$. Thus, t_k and $\bar{E}$ intersect each other.

This proposition is thus proved. □

Deforming an intersection-free and topologically faithful contour by sliding the vertices on the sticks holding them will NOT change the properties of topological faithfulness and intersection-free (see Propositions 3 and 4). Because of this, we develop the verification procedure below.

For each segmented region R_i, we first estimate its simplified edge $\bar{E}_i$ by connecting its two ending vertices. We verify whether $\bar{E}_i$ can be obtained from sliding the internal vertices on their corresponding sticks by testing if $\bar{E}_i$ intersects all these sticks(see Proposition 5 for the correctness of such a test). If $\bar{E}_i$ intersects all the tested sticks, we go to the distortion error test. The distortion error between R_i and $\bar{E}_i$ can be evaluated by

$$D'(R_i, \bar{E}_i) = \sum_{E \in R_i} D(E, L(\bar{E}_i)) \tag{5}$$

where $L(\bar{E}_i)$ gives the line equation of $\bar{E}_i$. The simplification on R_i can pass this test only if $D'(R_i, \bar{E}_i) <= \epsilon$, where ϵ is a user specified distortion tolerance. If either of these two tests fails, the region R_i is further separated into two regions which are determined by a local variational clustering conducted only in R_i. This trial-and-error procedure is recursively performed until all the segmented regions on the contour satisfy both of the two verifications. Then, each segmented region is converted into a simplified edge of the contour. Note that this trial-and-error procedure is guaranteed to stop since in the worst case, each region is a single edge on the smoothed contour $\bar{C}^m$. Still, the topologically faithful condition is satisfied and the distortion error for every such region is simply *zero*. The effectiveness of our verification technique is demonstrated in Fig. 12(b).

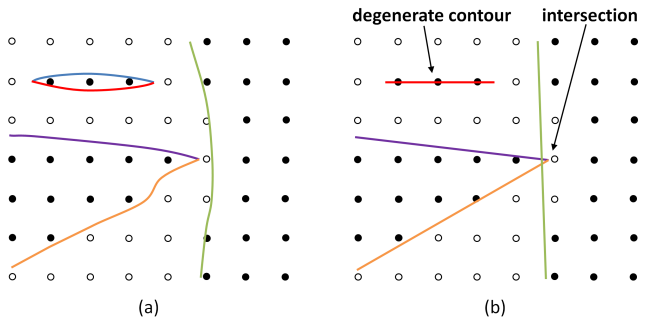

Figure 13: A contour that is originally intersection-free could become intersected or degenerate by replacing the curved region with line segments.

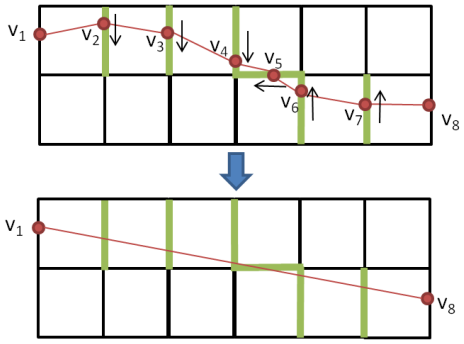

Figure 14: Sliding the six vertices on their respective sticks in order to form a single edge connecting the starting and ending vertices of this region.

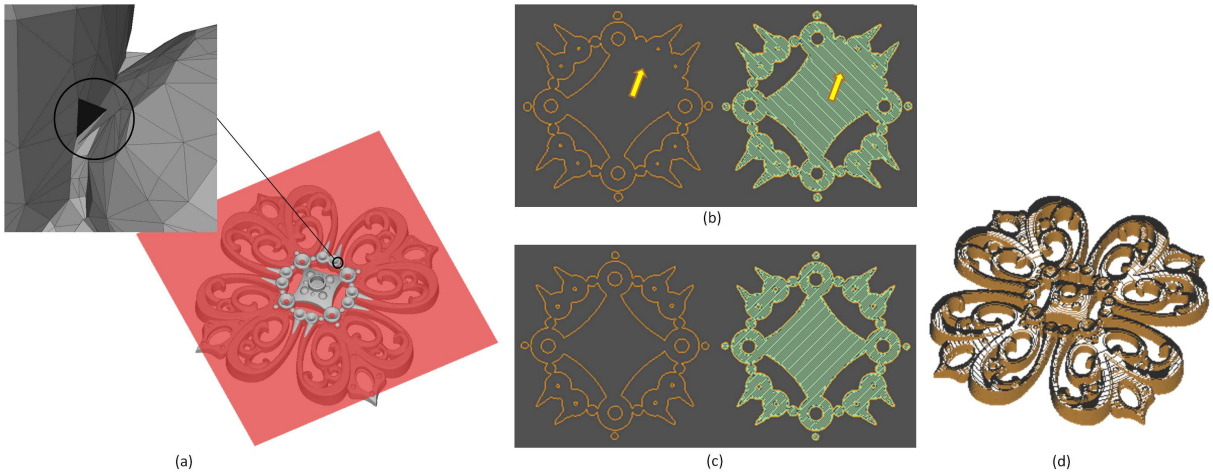

Figure 15: An example of slicing the Filigree model: (a) a mesh tessellated from implicit Filigree model, (b) the contours and their corresponding tool path (in green) generated by Insight^{TM} version 7.0 on the layer with 0.77 inch height, (c) the contours generated by our approach for the same layer and their corresponding tool path, (d) the rendered slicing contours generated by our approach.

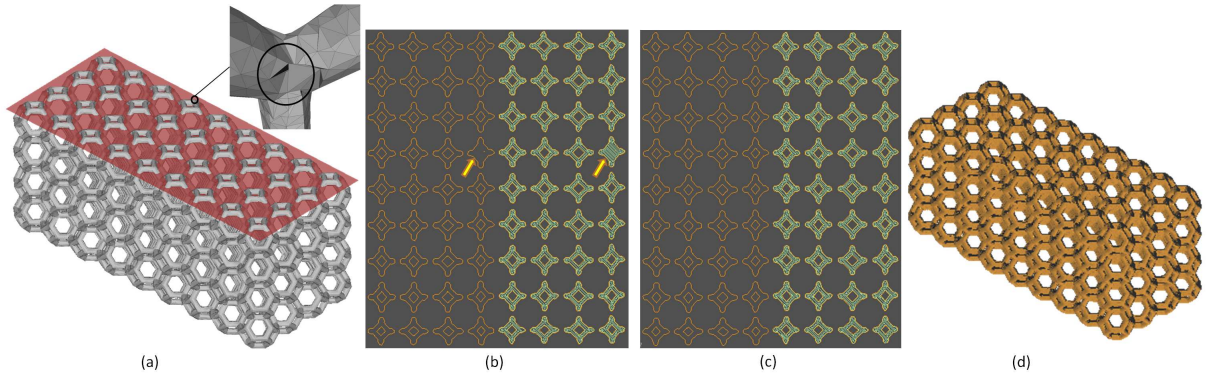

Figure 16: An example of slicing the Truss model: (a) a mesh tessellated from implicit Truss model, (b) the contours and their corresponding tool path (in green) generated by Insight^{TM} version 7.0 on the layer with 2.44 inch height, (c) the contours generated by our approach for the same layer and their corresponding tool path, (d) the rendered slicing contours generated by our approach.

6 Results and Discussion

We have implemented the proposed approach in a C++ program. The examples shown in this paper are all tested on a PC with Intel Core 2 Quad CPU Q6600 2.4GHz.

The two engineering models shown in Figs.15 and 16 give a comparison between prior slicing algorithm (in commercial software) and our approach. Due to the self-intersection in the tessellated triangular meshes from implicit solids, prior slicing algorithm may produce incorrect contours, and consequently the tool path of part material will also have defects (see Figs.15(b) and 16(b)). Unwanted films will be produced for both of the two examples as a result. In addition, the filigree and truss models demonstrate that our approach can easily handle the solids with complex topology.

In order to verify the effectiveness of our topology verification technique, we analyzed the resultant contours for Buddha and Truss models which are shown in Figs.17 and 18. For the layer at 2.90 inch height of Buddha, self-intersection will be removed if we use the topology verification (see Fig.17(c) and (d)). Meanwhile, the topology verification technique can successfully prevent degenerate contours (see Fig.18(c) and (d)) caused by narrow intersection regions.

Three biomedical models, Donna, Hand-complex and Spine are shown in Figs.8, 21 and 22 respectively. From the rendered slicing contours shown in Figs.21(b) and 22(b) and the fabricated model by FDM in Fig.8(c), we can see that the resultant contours provide very good shape approximation of the original solids. For Hand-complex and Spine, we conduct a study on the effect of clustering ratio α (see Sec.5) to the performance of our approach. As can be seen from Figs.21(c) and 22(c), the maximum regional distortion error are generally increasing as α increases. However, the rate of error increasing becomes slow as α becomes larger because the approach has to perform more topology and distortion-error verification operations under this situation. We also compare the number of resultant contour edges and the time consumption for different values of α. The resultant con-

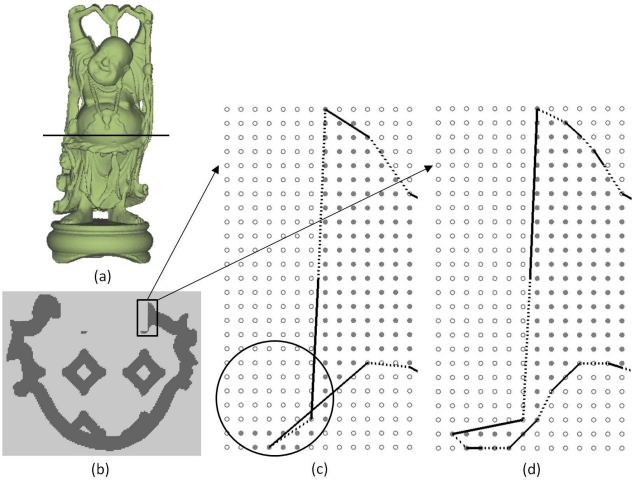

Figure 17: A comparison between resultant contours with and without topology verification: (a) the given Buddha model in the LDNI representation, (b) a binary image sampled from the layer with 2.90 inch height, (c) the resultant contour without topology verification, (d) the resultant contour with topology verification.

tour edge number first decreases as expected when $\alpha < 10$, while it increases slowly after $\alpha \geq 10$. This is because the contours require at least certain number of edges to guarantee topologically faithful and distortion-error bounded properties. Once the variational segmentation does not provide enough regions to satisfy these two properties, more regions will be generated through subdividing regions locally in verification stage. Since the subdivided regions cannot freely move to optimally fit the contour shape as what they can do in the variational segmentation stage, the more subdivision is applied, the more contour edges tend to be generated on the results. Time consumption increases quickly as α becomes too large because we need to perform more verification operations, which repeatedly check whether two line segments intersect each other to detect violation of topologically faithful property. Based on our experimental tests, selecting α between 10 and 15 shows a good trade-off. All testing results presented in this paper are generated by using $\alpha = 10$.

Our direct slicing approach is general for any implicit representations. Fig.19 and 20 give demonstration of our approach on BSP and RBF solids respectively. In our prototype implementation of slicing RBF solids, we use the adaptively supported RBF generator which is available on Ohtake's software website[1]. Statistics of experimental tests are shown in Table 1-3 for LDNI, BSP and RBF representations respectively. For all the tests on LDNI, the grid width r' of binary images is set to be no less than that of the input LDNI solid. Hence, the resolution of LDNI should be large enough to make sure the value of r' is in the order of 10^{-3}. The column, $\max(D'(R_i, \bar{E}_i))$, reports the maximum regional distortion error defined by Eq.(5) on simplified contours. In our tests, the tolerance of regional distortion error is actually set as square of r'. Therefore, it is easy to find that

[1] http://www.den.rcast.u-tokyo.ac.jp/~yu-ohtake

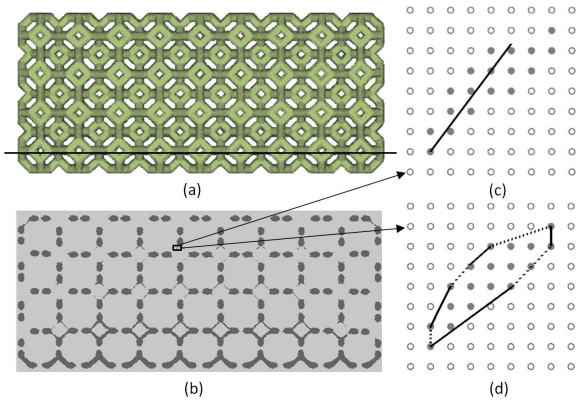

Figure 18: A comparison between resultant contours with and without topology verification: (a) the given Truss model in the LDNI representation, (b) a binary image sampled from the layer with 0.30 inch height, (c) the resultant contour without topology verification, (d) the resultant contour with topology verification.

our constrained shape simplification can successfully bound the distortion error with respect to the tolerance. We use the thickness of 0.01 inch for all the slicing tests, and the number of slicing layers are listed in the last column of the tables.

7 Conclusion

In this paper, we present a direct slicing approach for implicit solids. We investigate two main techniques, constrained Laplacian smoothing and contour simplification, which can produce self-intersection free and topologically faithful contours. In addition, we provide the proofs for the correctness of our approach. The approach presented in this paper also allows good distortion error control on the generated contours and has been shown to be very efficient.

Even though uniform binary image works well, the whole approach could be more efficient if we can make it in an adaptive resolution. We consider this as our near future work. The challenge is to retain the self-intersection free and topologically faithful properties on the contours after moving the computation into an adaptive sampling strategy. Meanwhile, the distortion error introduced between layers will also be considered and modeled to further improve the quality of models fabricated from the contours generated by our approach.

Acknowledgement

The research conducted in this paper is supported by Hong Kong RGC/GRF grants (CUHK/417109 and CUHK/417508). The third author is supported by the National Science Foundation grant CMMI-0927397. The authors would like to thank Yuen-Shan Leung and Allan Mok for generating the LDNI solid and the FDM object shown at the beginning of this paper, and Ms. Siu Ping Mok for proof reading the manuscript.

Table 1: Statistics of Experimental Tests on LDNI

Examples	Model Size (L×W×H)	LDNI Resolution	LDNI Grid Width	r'	Time* (in min)	$\max(D'(R_i, \bar{E}_i))$	Contour Edges No. Before vs. After Simplification	Layer No.
Dragon	0.90×2.00×1.41	1500	1.44e-3	1.50e-3	2.10 (0.48)	3.46e-9	458k / 79k	414
Spine	10.30×1.72×2.18	3000	3.77e-3	4.00e-3	7.68 (2.61)	1.49e-7	1305k / 267k	218
Hand	4.65×6.63×2.27	3500	2.08e-3	2.50e-3	8.53 (3.01)	1.66e-8	1338k / 255k	227
Donna	7.14×6.10×4.41	2000	3.93e-3	4.00e-3	12.55 (3.10)	6.02e-8	2468k / 368k	441
Truss	2.40×5.82×2.54	1300	4.93e-3	5.00e-3	20.18 (0.88)	2.32e-7	3973k / 1027k	254
Filigree	8.00×8.00×1.21	4000	2.20e-3	2.50e-3	43.13 (2.78)	3.96e-8	3110k / 545k	121

*The reported time includes binary image sampling (parallelized using multi-thread processing library – OpenMP), contour reconstruction, constrained Laplacian smoothing and contour simplification. The value in brackets represents time for binary image sampling.

Table 2: Statistics of Experimental Tests on BSP

Examples	Model Size (L×W×H)	BSP Tree Size	BSP Tree Complexity#	r'	Time* (in min)	$\max(D'(R_i, \bar{E}_i))$	Contour Edges No. Before vs. After Simplification	Layer No.
Oct-flower	3.50×3.50×2.49	151309	21	2.00e-3	4.89 (2.77)	3.13e-9	834k / 90k	249
Rocker-arm	5.00×1.52×2.57	156813	22	2.00e-3	5.25 (1.95)	3.13e-9	1359k / 166k	257
B-Torus	5.05×5.05×3.86	225319	23	2.50e-3	18.41 (5.93)	1.40e-8	3823k / 428k	386
Gear	6.35×6.35×3.90	50099	23	3.00e-3	20.8 (6.26)	2.49e-8	3897k / 420k	390

*The reported time includes binary image sampling (parallelized using multi-thread processing library – OpenMP), contour reconstruction, constrained Laplacian smoothing and contour simplification. The value in brackets represents time for binary image sampling.
#The complexity of BSP tree is defined as the average depth of its leaf nodes.

Table 3: Statistics of Experimental Tests on Adaptively Supported RBF

Examples	Model Size (L×W×H)	Points No. for Surface Fitting	Basic Functions Number	r'	Time* (in min)	$\max(D'(R_i, \bar{E}_i))$	Contour Edges No. Before vs. After Simplification	Layer No.
Armadillo	3.17×2.88×3.78	80000	72787	3.00e-3	9.16 (4.73)	3.01e-8	1548k / 348k	387
S-chair	3.50×6.32×3.35	16113	14010	2.00e-3	14.98 (11.45)	1.01e-8	1366k / 197k	335
Horse	5.02×2.30×4.17	19851	9797	2.00e-3	26.33 (20.61)	1.27e-8	2094k / 294k	417
Bunny	6.23×4.83×6.17	34834	21068	3.00e-3	29.85 (18.68)	3.59e-8	3528k / 588k	617

*The reported time includes binary image sampling (parallelized using multi-thread processing library – OpenMP), contour reconstruction, constrained Laplacian smoothing and contour simplification. The value in brackets represents time for binary image sampling.

References

[1] Ju T. and Udeshi T. Intersection-free Contouring on An Octree Grid, Proceedings of Pacific Graphics, 2006.

[2] Varadhan R., Krishnan S., Zhang L., and Manocha D., Reliable Implicit Surface Polygonization using Visibility Mapping, Proceedings of Symposium on Geometry Processing, 2006.

[3] Chen Y. and Wang C. C. L., Layered Depth-Normal Images for Complex Geometries - Part One: Accurate Sampling and Adaptive Modeling. ASME IDETC/CIE 2008 Conference, 28th Computers and Information in Engineering Conference, New York City, New York, 2008, DETC2008-49432.

[4] Wang C. C. L. and Chen Y., Layered Depth-Normal Images for Complex Geometries C Part Two: Manifold-Preserved Adaptive Contouring. ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conferences, New York City, New York, August 3- 6, 2008, DETC2008-49576.

[5] Ju T., Losasso F., Schaefer S. and Warren J., Dual Contouring of Hermite Data. ACM Transactions on Graphics, 21(3), 339-346, 2002.

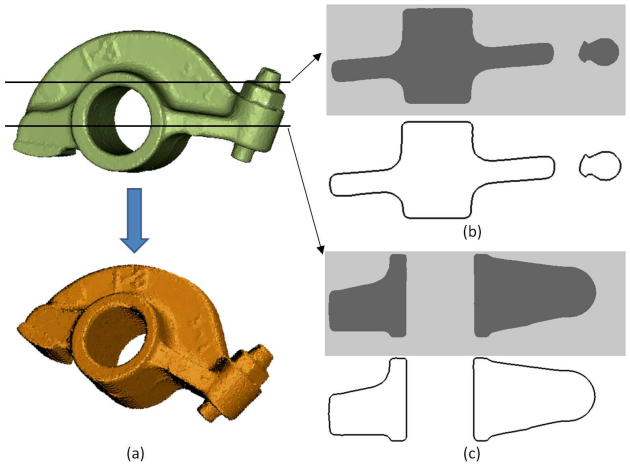

Figure 19: A demonstration of our approach on BSP solid: (a) the given Rocker-arm model in BSP representation and the resultant rendered contours, (b) the binary image and corresponding contours for the layer with 1.66 inch height, (c) the binary image and corresponding contours for the layer with 0.85 inch height.

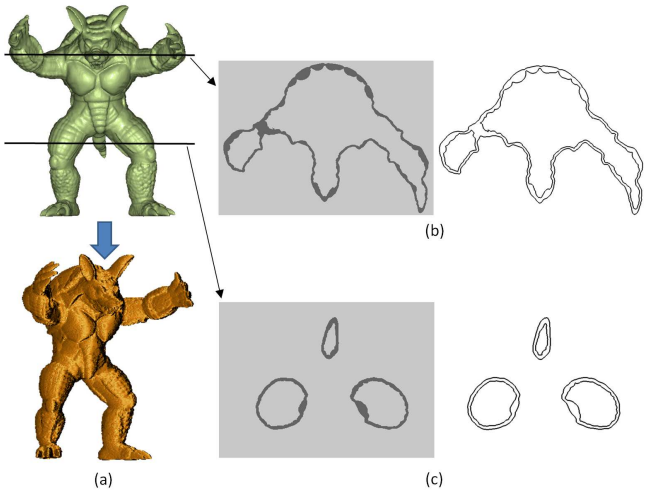

Figure 20: A demonstration of our approach on RBF solid: (a) the given Armadillo model in adaptively supported RBF representation and the resultant rendered contours, (b) the binary image and corresponding contours for the layer with 2.90 inch height, (c) the binary image and corresponding contours for the layer with 1.30 inch height.

[6] Chua C. K., Leong K. F. and Lim C. S., Rapid Prototyping: Principles and Applications, World Scientific, Singapore, 2003.

[7] Fan R., Wang C. C. L. and Jin X., General Transformation of LDNI Solid. Technical Report of CUHK. 2010.

[8] Luo R. C. and Ma Y., A Slicing Algorithm For Rapid Prototyping and Manufacturing. Proceedings of IEEE International Conference on Robotics and Automation, 1995, 3, 2841-2846, 1995.

[9] Farouki R. T., The Characterization of Parametric Surface Sections. Computer Vision, Graph and Image Processing, 33, 209-236, 1986.

[10] Lee R. B. and Fredericks D. A., Intersection of Parametric Surfaces and a Plane. IEEE Computer Graphics and Application. 4(8), 112-117, 1981.

[11] Barnhill R. E. and Kersey S. N., A Marching Method for Parametric Surfacdsurface Intersection. Computer Aided Geometric Design. 7, 257-280, 1990.

[12] Barnhill R. E., Farin G. E., Jordan M. and Piper B.R., Surface/surface Intersection. Computer Aided Geometric Design. vol. 4, 3-16, 1987.

[13] Yang P. and Qian X., Adaptive Slicing of Moving Least Squares Surfaces: Toward Direct Manufacturing of Point Set Surfaces. ASME Transactions Journal of Computing and Information Science in Engineering, 8(3), 2008.

[14] Qiu Y., Zhou X. and Qian X., Direct Slicing of Cloud Data with Guaranteed Topology for Rapid Prototyping. International Journal of Advanced Manufacturing Technology. accepted, 2010.

[15] Liu Y., Xia S. and Qian X., Direct Numerical Control (NC) Path Generation: From Discrete Points to Continuous Spline Paths. ASME Transactions Journal of Computing and Information Science in Engineering, 12(3), 2012.

[16] Lorensen W. E. and Cline H. E., Marching Cubes: A high Resolution 3D Surface Construction Algorithm. Computer Graphics, 21(4), 163-169, 1987.

[17] Kobbelt L. P., Botsch M., Schwanecke U. and Seidel H., Feature Sensitive Surface Extraction from Volume Data. Proceedings of ACM SIGGRAPH 2001, 57-66, 2001.

[18] Chica A., Willliams J., Andujar C., Brunet P., Navazo I., Rossignac J. and Vinacua A., Pressing: Smooth Isosurfaces with Flats from Binary Grids. Computer Graphics Forum. 27(1), 36-46, 2008.

[19] Stelldinger P., Latecki L. J. and Siqueira M., Topological Equivalence between a 3D Object and the Reconstruction of Its Digital Image. IEEE Transactions on Pattern Analysis and Machine Intelligence. 29(1), 126-140, 2007.

[20] Hoppe H., Progressive meshes. Proceedings of ACM SIGGRAPH 1996, 99-108, 1996

[21] Klein R., Liebich G. and Strasser W., Mesh Reduction with Error Control. IEEE Visualization Proceedings, 311-318, 1996.

[22] Michael Garland and Paul S. Heckbert. Simplifying Surfaces with Color and Texture using Quadric Error Metrics. Proceedings of IEEE Visualization, 263-269, 1998.

[23] Lindstrom P. and Turk G., Fast and Memory Efficient Polygonal Simplification. Proceedings of IEEE Visualization, 279-286, 1998.

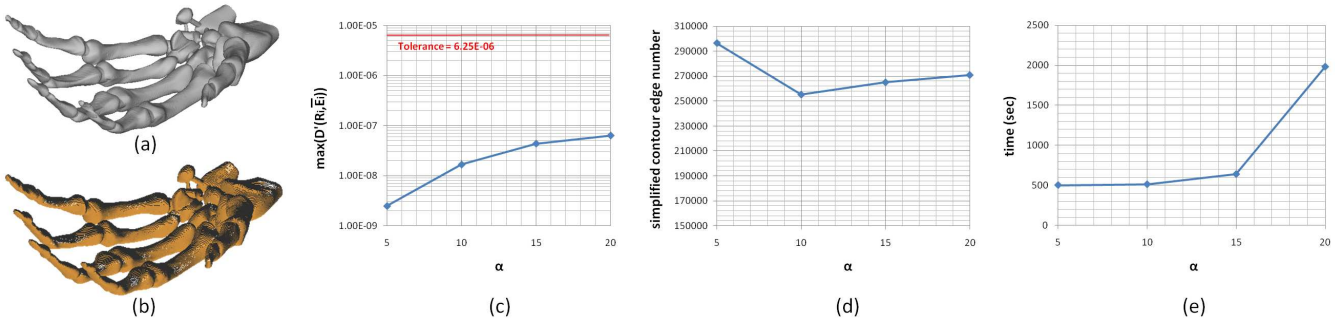

Figure 21: An example of slicing the Hand-complex model: (a) the original Hand-complex model, (b) the rendered slicing contours, (c) the chart of maximum regional distortion error versus the clustering ratio α, (d) the chart of simplified contour edge number versus α, (e) the chart of time consumption versus α.

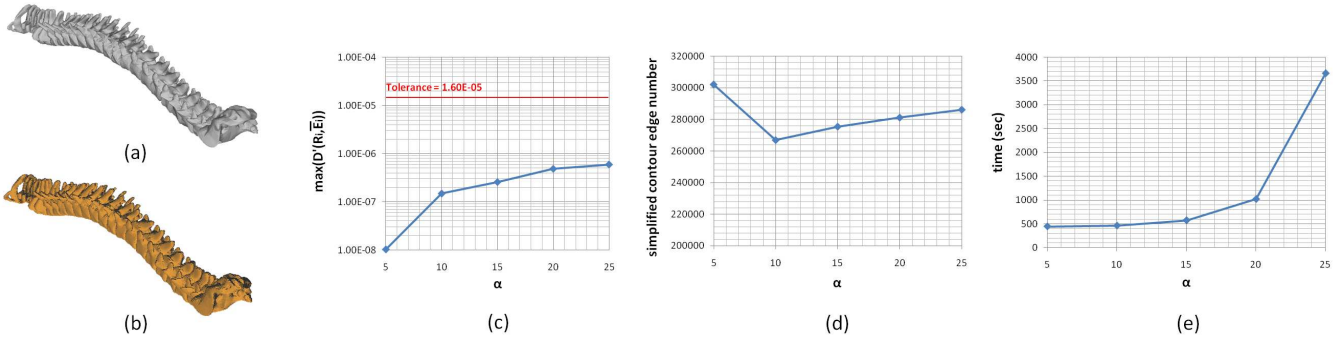

Figure 22: An example of slicing the Spine model: (a) the original Spine model, (b) the rendered slicing contours, (c) the chart of maximum regional distortion error versus the clustering ratio α, (d) the chart of simplified contour edge number versus α, (e) the chart of time consumption versus α.

[24] Maillot J., Yahia H. and Verroust A., Interactive Texture Mapping. Proceedings of ACM SIGGRAPH 1993, 27-34, 1993.

[25] Kalvin A. D. and Taylor R. H., Superfaces: Polygonal Mesh Simplification with Bounded Error. IEEE Computer Graphics and Applications, 16(3), 64-77, 1996.

[26] Inoue K., Itoh T., Yamada A., Furuhata T. and Shimada K., Clustering A Large Number Of Faces For 2-Dimensional Mesh Generation. Eightth International Meshing Roundtable, 281-292, 1999.

[27] Sheffer A., Model Simplification for Meshing Using Face Clustering. Computer Aided Design. 33. 925-934, 2000.

[28] Sander P., Snyder J., Gortler S. and Hoppe H., Texture Mapping Progressive Meshes. Proceedings of ACM SIGGRAPH 2001, 409-416, 2001.

[29] Garland M., Willmott A. and Heckbert P. S., Hierarchical Face Clustering on Polygonal Surfaces. Proceedings of the 2001 symposium on Interactive 3D graphics. 2001.

[30] Grinspun E. and Schröder P., Normal Bounds for Subdivision-Surface Interference Detection. Proceedings of IEEE Scientific Visualization, 333-340, 2001.

[31] Lévy B., Petitjean S., Ray N. and Maillot J., Least Squares Conformal Maps for Automatic Texture Atlas Generation. Proceedings of ACM SIGGRAPH 2002, 362-371, 2002.

[32] Cohen-Steiner D., Alliez P. and Desbrun M., Variational Shape Approximation. Proceedings of ACM SIGGRAPH 2004, 905-914, 2004.

[33] Lloyd A. P., Least Square Quantization in PCM. IEEE Transactions on Information Theory. 28(2), 129-137, 1982.

[34] Fuchs H., Kedem Z.M. and Nalor B.F., On Visible Surface Generation by a Priori Tree Structures. Proceedings of ACM SIGGRAPH 1980, 124-133, 1980.

[35] Ohtake Y, Belyaev A and Seidel H.P., 3D Scattered Data Approximation with Adaptive Compactly Supported Radial Basis Functions. Proceedings of the Shape Modeling International 2004, 31-39, 2004.