A New Adaptive Slicing Algorithm Based on Slice Contour Reconstruction in Layered Manufacturing Process

Suyun Liu[1] Ajay Joneja[2] and Kai Tang[3]

[1]Dept of IEDA, Hong Kong University of Science Technology, sul217@lehigh.edu
[2]Div of ISD, Hong Kong University of Science Technology, joneja@ust.hk
[3]Dept of MAE, Hong Kong University of Science Technology, mektang@ust.hk

Corresponding author: Ajay Joneja, joneja@ust.hk

Abstract. Layered manufacturing processes operate by building the part in layers, or slices. The layer thickness is controlled to trading off between part accuracy and fabrication time. Several adaptive slicing approaches have been proposed to reduce the printing time while improving the accuracy. In this paper, a novel adaptive slicing algorithm based on slice contour reconstruction is presented. This approach can be adapted for standard additive manufacturing machines because it does not rely upon using multiple distinct values of layer thickness along the build direction. A simple geometric engine allows our approach to be adapted to existing machines. Our contour reconstruction algorithm, based on Medial axis computation, leads to 30%-50% reduction on volumetric deviation and distance deviation between the original part surface and the deposited part surface. Alternatively, for a given user-specified tolerance, our approach allows increases of layer thickness by over 20%, resulting is commensurate savings in build-time.

Keywords: 3D printing, dimensional tolerance, medial axis, voronoi diagram
DOI: https://doi.org/10.14733/cadaps.2021.1425-1447

1 INTRODUCTION

This paper addresses the problem of reducing build times in Layered Manufacturing (LM) technologies in which 3D parts are constructed by depositing a series of planar layers. Current techniques can reduce the manufacturing lead time in such applications by 30-50 percent [9], but there is much demand for speeding up LM. An attractive property of LM is that it is relatively independent of model geometry complexity; this discriminates it from most traditional manufacturing processes.

Most commercial LM systems use a multi-stage processing pipeline [9]: 3D modeling, Data conversion, Pre-processing, Model slicing, Part deposition and Post-processing. The 3D model of the mechanical product is created by a CAD system and converted into a tessellated format (STL). The tessellated model is often

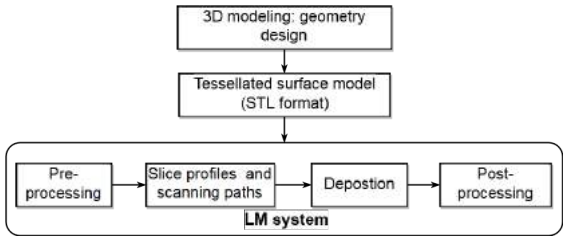


Figure 1: Data process pipeline in Layered Manufacturing

preprocessed via a set of operations, i.e. translation, rotation, and support and infill structure generation. The processed model is then sliced by intersecting with a series of horizontal planes to get a set of layers bounded by ***contours***, which are closed polygons. The spacing between successive planes denotes the ***slice thickness*** and the solid between them is called a ***slice***. Finally, the printing head fabricates each layer by traversing along the computed scanning path. The extra materials, such as the support structure, are removed from the part in the Post-processing stage.

The quality of the prototype is determined by several key process variables including the part orientation, the layer thickness, the deposition speed and environment variables such as temperature and humidity variations. The part (and process) quality may be measured in terms of the part strength, dimensional accuracy, surface finish, build time or the material utilization. To achieve higher part quality, the four most common issues addressed by researchers include [13]: (i) optimal part orientation, (ii) slicing strategy, (iii) process parameter optimization, (iv) post-treatment. The determination of the slice thickness is a key parameter involving the trade-off between part accuracy and deposition time. In particular, due to stair-step (or staircase) errors, smaller slice thickness leads to lower surface roughness but requires at higher build time. Therefore, some researchers have explored methods for ***Adaptive Slicing,*** in which they allow for using layers of different slice thickness at different heights along the build direction. However, a big issue with this approach is that practically no existing RP machine provides this option. In this paper, we explore an approach for improving the layer accuracy even when layer thickness remains constant over the build.

The rest of the paper is organized as follows. Section two reviews different categories of adaptive slicing algorithms. Section 3 and Section 4 present present and initial and an improved improved algorithm using adaptive slice profiles. Case studies illustrating the effectiveness of the improved algorithm over the traditional one are presented. Finally, some remarks and future research challenges are discussed in Section 5.

2 LITERATURE REVIEW

Most commercial 3D printers implement the slicing process based on planar intersections of the faceted CAD model in STL format, which is computationally robust and efficient to implement. The slice at each layer is approximated by extruding the cross-section by a distance equal to the layer thickness along the build direction. Several categories of adaptive slicing algorithms have been proposed for the purpose of achieving higher accuracy and/or lowering deposition time [24]. Model slicing can be carried out on tessellated models, analytical part shapes, and even on point cloud data. Before we look into the details of these algorithms, several key approaches to measure part accuracy are presented.

2.1 Error Measurements

To measure the deviation or alternatively error between the deposited part surface and the original part surface at each slice height, there are several commonly used measures: volumetric deviation[19], *cusp height*[11],

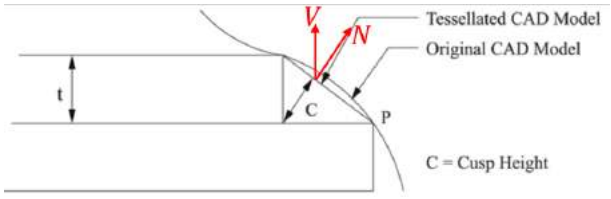


Figure 2: Staircase effect and Cusp height

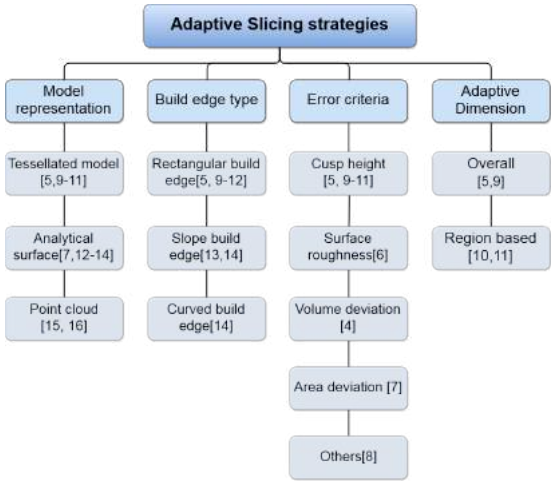


Figure 3: Adaptive slicing categories based on different criteria

surface roughness[25], area deviation[31], circumference difference and gravity deviation[21]. Among these, the first three approaches are most popular in adaptive slicing.

Intuitively, the *volumetric error* is defined as the volume difference between the built-up part and the original CAD model. This error can be computed for any layer, and summed over the entire part. Eqn. (1), which accounts for the overbuilt as well as the under-built regions, give an approach to calculate volume difference. Here V_O and V_P denote volume of the original CAD part and the fabricated part respectively. Kumar and Choudhury [19] explored an algorithm to calculate the volume deviation between a CAD model and built-up part in five axis laminated object manufacturing. They concluded that reduction on volume deviation largely improves feature recognition.

$$\Delta'_V = (V_O \cup V_P) - (V_O \cap V_P) \tag{1}$$

In engineering practice, *roughness* is a traditional measure for surface quality. Mechanical (using touch probes) or optical profilometers (using projected light) are commonly used to measure it. Roughness was first used for evaluating LM part by Reeves and Cobb [25] to improve the surface accuracy in Stereo-Lithography Apparatus (SLA). In the context of planning, a geometric definition is useful. *Cusp height* is the error associated to the staircase effect, originally described by Dolenc and Makela [11] . In Fig. 2, suppose that the build direction is Vparallel to the Z axis. Given a point on the curved surface or tessellated surface, the cusp height error is given by $C = t * cos\theta$, where $\theta =< V, N >$, N is the normal vector to the part surface and t is the layer thickness. However, when the normal N is perpendicular to the slice level plane, this expression becomes invalid; in this case, the horizontal distance on the slicing plane is used as a substitute.

2.2 Adaptive Slicing

The total deposition time for a part increases approximately linearly with the number of slices for a given part orientation. Therefore, using different layer thicknesses at different heights along the build direction has been studied by several researchers to achieve a good compromise between part accuracy and deposition efficiency. These algorithms are classified into different categories from different points of view (see Fig. 3). We provide a brief discussion of the relevant works by categorizing them based on the type of input model format: slicing of analytical models, tessellated models and point cloud data.

2.2.1 Slicing of tessellated models

Dolenc and Makela [11] first introduced the notion of an idealized *cusp height* for finding the best slice thickness for a given tessellated model. They calculated the layer thickness required to satisfy surface tolerance (δ) defined by maximum cusp height (C_{max}) at each layer. Specifically, denote $\vec{n_z}$ as the vertical component of normal vector at a given point p_i within the layer. The upper bound of layer thickness is obtained by Eqn.(2):

$$t_{p_i} = min\{t_{max}, C_{max}/\vec{n_z}\} \tag{2}$$

where t_{max} is the maximum available layer thickness of the machine. The optimal layer thickness for a slice is determined by a set of discrete points $P = \{p_i, i = 0..n\}$ along the contour.

$$t_{optimal} = max\{t_{min}, min\{t_{p_i}\}\} \tag{3}$$

where t_{min} is the minimum available layer thickness in the RP machine. The determination of thickness for each layer is implemented iteratively from bottom to top. Sabourin *et al.* [27] refined this algorithm and developed their slicing algorithm by first cutting the model into slabs of the maximum layer thickness. Each slab is then divided (if necessary) into an integer number of thinner slabs whose thickness is determined by measuring *cusp height* using the same formulation as in [11]. The model's finer features were preserved by using thinner slices in regions of high curvature.

The methods above output the thickness for each layer based on worst geometry error to meet the tolerance criteria. Later, Sabourin *et al.* [26] proposed a region-based adaptive slicing algorithm that applies two levels of thickness inside a slice. They developed a particular RP system to support filling the interior regions quickly with thicker layers while building the exterior regions slowly with thinner layer. An average 50 percent reduction in build time was achieved without reducing the part surface quality.

Justin and Jan [30] also looked into the local adaptive slicing algorithm where a set of different layer thickness values may be used in each layer. Their algorithm first identified peak features, based on which the tessellated model is sliced into uniform thick slabs of the maximum available thickness. Then, according to the unbuilt geometry in each slab, a number of thinner layers are computed. Similar to [26], they implemented the strategy on a modified commercial machine (a Stratasys^{TM} FDM 1600) using a custom software interface. In their case study, a precise comparison among uniform slicing, conventional adaptive slicing and the proposed local adaptive slicing algorithm showed up to 33 percent time-saving at a given level of surface tolerance.

2.2.2 Slicing on analytical models

Direct slicing, or Slicing of the analytical Solid/Surface CAD model was proposed by several researchers. The claimed advantages of this approach include preservation of the geometric and topological correctness, higher model accuracy, reduction in the the preprocessing time, and smaller file sizes. Accurate CAD models can be represented by Constructive Solid Geometry representation (CSG) or Boundary representation (BRep), and the model surfaces may be simple implicit surfaces or complex curved ones, with the latter being described via B-Splines or Non-Uniform Rational B-spline Surfaces (NURBS).

Zhao and Luc [31] adopted direct slicing on the solid model (CSG), determining the layer thickness based on measuring the Area deviation ratio defined by two consecutive layers; see Eqn. (4). The model is divided into blocks and the layer thickness is calculated iteratively in each block by considering the maximum area deviation.

$$|\frac{(A_i - A_i')}{A_i}| < \delta \tag{4}$$

where A_i and A_i' are areas of two adjacent cross-sections, and δ is the tolerance defined as maximum area deviation.

Suh and Wozny [29] proposed an adaptive slicing approach similar to [31], but based on BRep. For each block, each iteration considers a dense-enough set of sample points from current height contour and calculates the layer thickness by Eqn. (5) for upward facets and by Eqn. (6) for downward facets.

$$d = -\rho sin\theta + \sqrt{\rho^2 cos^2\theta - 2\rho\delta - \delta^2} \tag{5}$$

$$d = \begin{cases} \rho cos\theta - \sqrt{\rho^2 cos^2\theta - 2\delta\rho - \rho^2} & \rho^2 cos^2\theta - 2\delta\rho - \rho^2 > 0 \\ \rho cos\theta & otherwise \end{cases} \tag{6}$$

where ρ denotes the radius of sphere that approximates the curvature at surface point, θ is the angle between horizontal plane and the facet containing the point, and δ is user-defined surface tolerance.

In actual built parts, the side walls of deposited layers are not absolutely vertical. This has some implications on methods to improve the surface accuracy. Hope *et al.* [15] adopted sloping build edge to better match the shape of the part surface; they ensure C_0 continuity at the surface, thereby reducing the staircase effect. They developed the TruSurf system (a kind of SLA system) for depositing sloping side edges. The input model is represented by NURBS surfaces. In their implementation, the *cusp height* error is measured on a set of points sampled from the top slicing plane, the bottom slicing plane and the middle plane. A more recent work by Huang *et al.*[16] proposed adaptive slicing algorithm using curved built edges. Curved and sloping build edges require higher degree of freedom in deposition head control, and are not commonly seen in commercial machines.

2.2.3 Slicing on the point cloud

There are some methods where the point cloud is the only input used to fabricate the 3D prototype. Since these approaches are directly related to our work, we merely reference the works of Szu-Shen [7] and Javidrad *et al.* [17] here for completeness.

2.2.4 Our contribution

In most commercial machines, the nozzle diameter determines the slice thickness, e.g. in the FDM-1650 system from Stratasys. As far as we know, commercial deposition based RP systems currently do not support deposition scheme with adaptive layer thickness. Meanwhile, to implement sloping or curved layer surface deposition, more axes of control, namely four or more degrees of freedom, are required. This capability is also missing in most state-of-the-art machines. Computationally, almost all the slicing algorithms require calculations of error based on dense sampling points, which makes the layer thickness calculation inefficient.

This paper presents a new adaptive slicing algorithm for tessellated models. Our method does not rely upon varying the slice thickness (although it can be used in that context also), and is therefore compatible with most existing RP machines. For generality, the proposed adaptive slicing algorithm measures error in terms of real volumetric deviation and maximum horizontal distance error. We use well-known, robust, and efficient computational geometric techniques. In addition, our algorithm operates on accurate tessellated CAD model and therefore every slice takes the full mesh information into consideration.

3 THE PROPOSED APPROACH

Commercial RP machines use the slice contour at the nominal height of the model to construct the next layer. Local geometry variations and machine specifications dictate the magnitude of layer thickness. In our proposed scheme, for each slice, we proceed to search an alternative contour based on the geometry of the zone between the top and bottom planes. The contour is computed to yield an error no worse than the traditional strategy. The efficacy of this approach is first demonstrated by selecting an actual contour at an appropriately chosen height between the heights of the bottom and top of the slice. Next, a more general strategy is introduced, which computes a contour by considering a silhouette of the slice. This approach is presented in subsection 3.3. The part orientation has distinct impact on the part accuracy and deposition time – however, our approach is effective independent of the build direction. Therefore we shall assume in the following that the ideal part orientation has already been determined based on appropriate criteria [22] , e.g. minimum support volume, minimum area of supported surface, or avoiding support on important features.

3.1 Validation Model

To verify the feasibility, we measure the volumetric error, from the tessellated model relative to a square-edge layered geometry, to search for the optimal height. In objective function (Eqn. 7), V_O and V_E are the geometry volume of the original slice and the extruded slice respectively, where the volume of extruded slice is the function of height $h \in [h_{lower}, h_{upper}]$, so is the volume deviation error ΔV.

$$\begin{aligned} min \;\; & \Delta V(h) = V_O \cup V_E(h) - V_O \cap V_E(h) \\ & s.t. h_{lower} \leq h \leq h_{upper} \end{aligned} \tag{7}$$

The pre-experiment is done in the commercial CAD software, CATIA^{TM}, which supports robust boolean operation and accurate mess measurement. Take a Pikachu model for instance, Fig.4(b) presents the original slice geometry and the extruded slice geometry taken from particular height of model in Fig.4(a). By Boolean operation on these two slices, the corresponding volume deviation is calculated via mass measurement tool. Furthermore, consider there is no guarantee on the part shape, the objective function is non-trivially non-convex and discontinuous. Therefore, the Simulated Annealing searching algorithm embedded in the CATIA optimizer is selected. Two examples, respectively from height range $[5mm, 6mm]$ and $[8mm, 9mm]$, illustrate the volume deviation along the height. The result seems the volume deviation is convex relative to height in both common cases, which by no means gives some implications: by simply replacing the lower cross-section curve by a contour at height between the layer, the volumetric error will be reduced by up to 50%. Consequently, under the same tolerance requirement, thicker layer thickness can be adopted and deposition time is significantly reduced. However, the optimizer is run with high complexity to converge, more than 1.5 min for one common slice. Another technical issue is when the slice is composed of multiple loops, the extrusion operation and Boolean operation require special handling, which further obstruct its practicality.

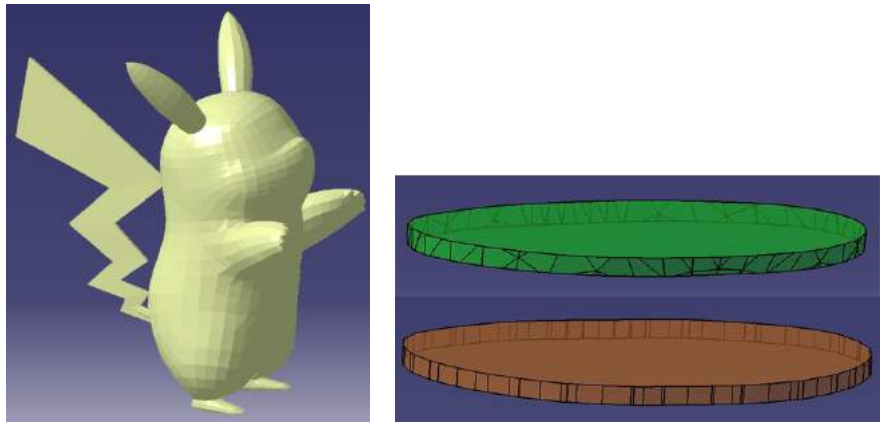
3.2 Definitions

Before we further explore alternative scheme where the entire contour is reconstructed from the slice geometry, some important definitions are given as the technological support.

3.2.1 Silhouette edge

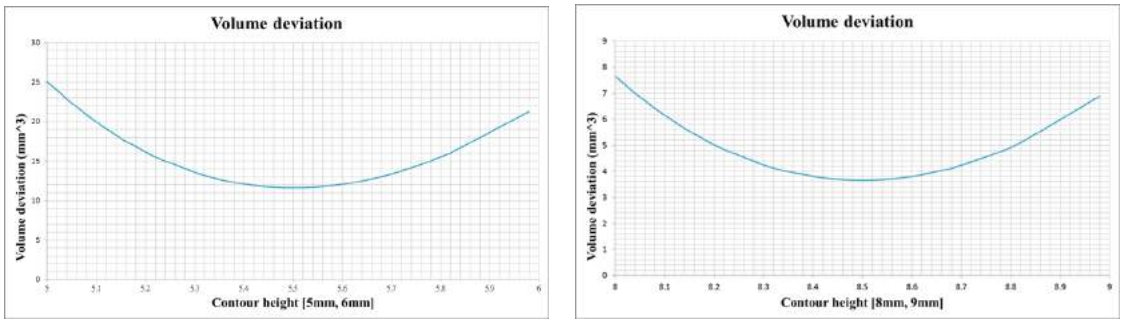
In computational graphics, the edge incident to two facets that are front-facing and back-facing with respect to the viewing direction is defined as *Silhouette*. If Eqn.8 holds for two neighboring facets with normal $\vec{n_1}$ and $\vec{n_2}$, their common edge is silhouette edge with respect to viewers.

$$dot(e\vec{y}e\vec{V}ec, \vec{n_1}) * dot(e\vec{y}e\vec{V}ec, \vec{n_2}) < 0 \tag{8}$$



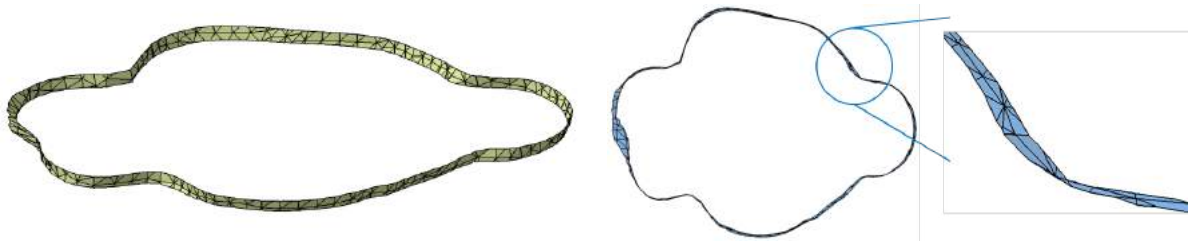
(a) Pikachu model (b) The original and extruded slice

Figure 4: The pre-experiment model and slice



(a) $h \in (5mm, 6mm)$ (b) $h \in (8mm, 9mm)$

Figure 5: Volume deviation $\Delta V(h)$



(a) slice mesh (b) projection view

Figure 6: A 3D slice mesh

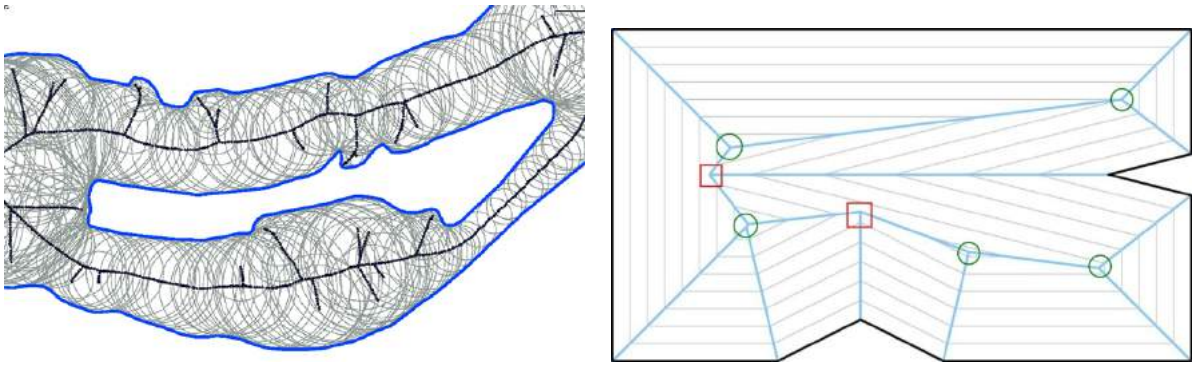
If we look at the slice mesh (Fig.6(a)) from the direction perpendicular to the cutting plane, namely the build direction, the outermost and innermost boundary are composed of full or partial silhouette edges, as well as full or partial edges on the two contour curves. As can been seen in Fig.6(b), the upper and lower contour curves may intersect with each other at some places. Furthermore, silhouette edges within the layer may also intersect two contour curves.

3.2.2 Centerline

For the purpose of computing distance error, We focus on the two-dimensional planar polygonal domain, in the shape of ***strip***, formed by the union of the projection of all or partial faces that compose the slice mesh surface. To minimize the horizontal distance between the vertical-walled surface and the original surface, we wish to find the "centerline" of the polygonal strip. Therefore, Medial axis and straight skeletons are two candidate schemes.

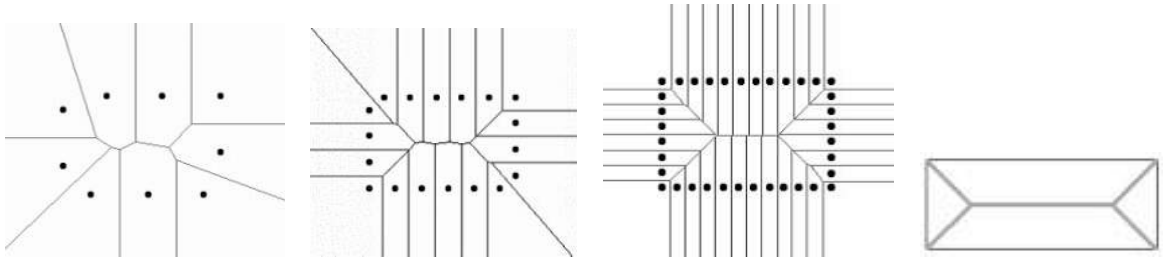
The Medial axis (MA) was introduced first by Bium[5] in 1967. Refer to Fig.7(a), consider a subset S of points in the n-dimensional Euclidean space which is bounded by planar curve C. An open ball inside C not contained in any other open ball inside C is denoted as the maximum ball. The MA of S is defined as the locus of centers of its maximum balls. For a polygon, the medial axis edges are either straight segments or parabolic curves. As for accurate computation, Lee [20] developed an $O(nlogn)$ algorithm to generate MA by computing the Voronoi diagram of a set of line segments making up a simple polygon. Aichholzer *et al.* [1] extended the domain to planar shapes bounded by polynomial spline curves. No stable and efficient algorithm is known for arbitrarily non-simple polygons. The straight skeleton (SS) is another kind of representation for the topological skeleton which is composed exclusively of straight segments. For instance, Fig.7(b) shows the straight skeleton of non-convex polygon. It was first defined for simple polygons by Aichholzer *et al.* [2] : consider a continuous shrinking process where the edges of the polygon are moved inwards parallel to themselves at a constant rate. The locus of the vertices of the propagating wave-front composes the straight skeleton. In the straight skeleton calculation, Aichholzer *et al.* [2] also provided an $O(n^2logn)$ algorithm where n is the number of vertices on an arbitrary simple polygon. Aichholzer and Aurenhammer [3] extended the previous work to polygons with holes. Their algorithm was based on triangulation and ran in $O(n^3logn)$ for the worst case. Cheng and Vigneron proposed an algorithm with $O(n\sqrt{n}log^2n)$ expected time complexity for a non-degenerate polygon with holes, which unfortunately is too complicated to implement.

In the straight skeleton scheme, regardless of its high computation complexity and instability, the bisectors as the components are not equidistant to its defining edges, but to the supporting lines. In other words, the straight skeleton might not be located at the appropriate center of the polygon. Therefore, the medial axis scheme is more appropriate in our application. However, the precise computation of medial axis is numerical unstable and furthermore, contains parabolic segments. Consequently, we adopt an approximate approach of



(a) Medial axis of Curve C (b) Straight skeleton

Figure 7: Two candidate schemes of centerline



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

Figure 8: Voronoi vertices converge to Medial axis while sampling density increases

computing the Medial axis via the Voronoi diagram of a sampling of points from boundaries.

3.2.3 Medial axis approximation by Voronoi diagram

Voronoi diagram is a planar partition scheme with respect to a set of sites. Consider a set of n point sites $P = \{p_1, p_2, ..., p_n\}$, each point p_i is associated with a unique region called Voronoi cell $Vor(p_i)$. Voronoi vertex denotes the point equidistant to at least three sites in P and Voronoi edge is the set of points equidistant to two sites. All the Voronoi vertices, Voronoi edges as well as Voronoi cells compose the Voronoi diagram or graph of P, denoted as $Vor(P)$. There exist three paradigms of Voronoi diagram construction algorithms, including divide-and-conquer algorithm by Shamos and Hoey [28], randomized incremental algorithm by Guibas and Knuth [14] and line sweep algorithms by Fortune [12]. These algorithms are quite efficient to run in $O(nlogn)$ time with $O(n)$ space.

The Medial axis can be approximated by extracting Voronoi edges from Voronoi diagram. Brandt [6] investigated the convergence of Voronoi diagram to Medial axis skeleton and proved that as the sampling density increases, the subset of Voronoi vertices converges to the Medial axis. See the Fig.8 for demonstration. Therefore, to compute the Medial axis skeleton, as the centerline for a planar strip polygon in our case, the Voronoi edges and Voronoi vertices inside the shape are supposed to be extracted to form a new contour.

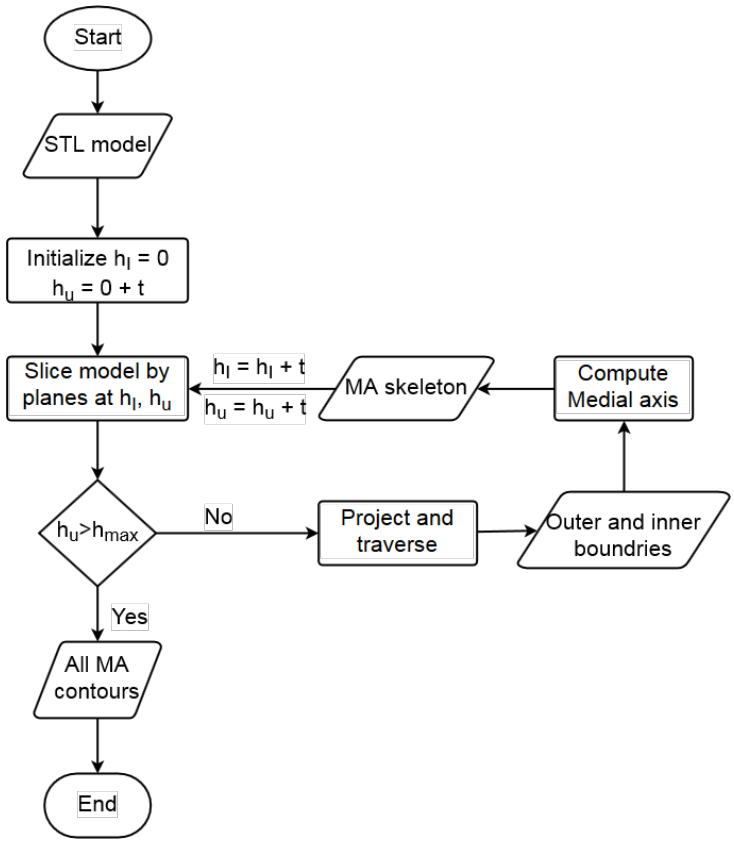


Figure 9: Algorithm Flowchart

3.3 Contour Reconstruction Algorithm

In our algorithm, we use the STL format as the input and measure both volume deviation and maximum horizontal deviation to evaluate its accuracy improvement. Without loss of generality, we shall use a scaled-up layer thickness to illustrate the algorithm. Before we process the model, the data in the STL file will be converted to produce a compact half-edge data structure[23]. The classical data structure provides us necessary topological information and simplifies geometry operation complexity. Our proposed adaptive slicing algorithm is outlined in Fig.9. Briefly, given an input CAD model and a pre-selected layer thickness t, the model is then sliced from bottom to top iteratively. For each slice, the 2D outermost and innermost boundaries, termed as *boundary pair*, are computed after projecting 3D mesh surface onto the horizontal plane. The Medial axis skeleton is finally extracted from the Voronoi graph of sampling points from the computed boundary pair.

A benchmark part, the Stanford bunny model in Fig.10, is used to illustrate the key ideas. The uniform slicing stage is the trivial intersection operation and by Euler's formula, computing the intersection curves and updating the slice mesh information can be done in $O(n)$ where n is the number of vertices. The slice meshes at several height are shown in Fig.11.

1) boundary computation

Each slice layer may produce one or more boundary pairs, referring to the ear position in Fig.11. Assume a non-degenerate case where each layer contains one or more separate sub-regions, namely, each sub-slice do

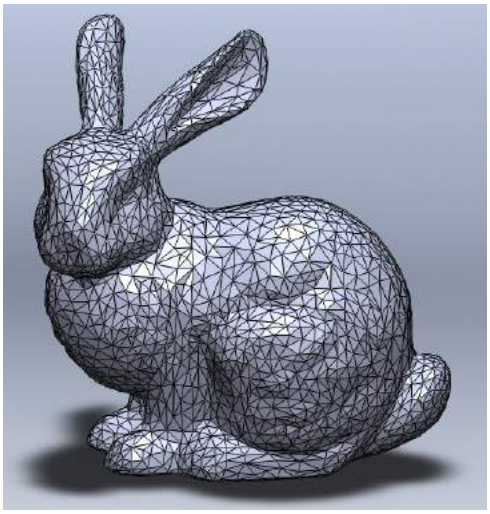


Figure 10: The Stanford bunny model

not intersect other sub-slices at the same height. Projection of 3D slices onto the horizontal plane might produce intersection points and thus destroy the original connectivity with respect to geometry complexity. All such intersections can be discovered by standard range searching techniques [10] in $O(nlogn + Ilogn)$, where I is the output size. After updating the data structure for 2D slice mesh in constant time, the boundary pairs can be computed by traversing and identifying all the silhouette edges. In our implementation, we skip constructing and updating half-edge data structure for entire 3D and 2D slice mesh.

Instead, we carry out two construction stages: (i) traversing on the upper and lower profiles. Specifically, for a mesh instance in Fig.12(a), the upper and lower profiles (Fig.13(a)) are the intersection curves between the two (bottom and top) slicing planes and the CAD model. The output from this stage is denoted as the initial boundary pair (Fig.13(b)). (ii) traversing and update the boundary pair by identifying silhouette edges within the layer. All the silhouette edges falling inside two slicing planes are identified as the candidates edges (see the green edges in the Fig.14(a)) to update the boundaries. We start from an extreme vertex among these on the initial boundary pair and silhouette edges, i.e. leftmost vertex , and traverse in counterclockwise order along the loop by moving to its end vertex that is incident to the "most right hand side" edge incident to current vertex (see the Fig.14(b)). This process end at backing to the starting vertex and output the final outermost boundary. Once we got the outermost boundary, the innermost boundary could be identified as well. The well-computed boundary pair corresponding to the slice mesh instance is given in Fig.12(b).

2) Medial axis contour extraction

Each boundary pair forms a polygonal domain with a hole. As analyzed in the section 3.2.3, to get a good approximation of the Medial axis skeleton, dense sampling point are uniformly taken from the boundary pair by setting the spacing as the minimum length among all the edges. In our implementation, the Voronoi diagram is constructed via a robust $O(nlogn)$ scheme provided by CGAL[18], where the points are inserted one by one and the Voronoi graph is updated incrementally. The corresponding Voronoi diagram of the previous instance (Fig.12(b)) is presented in Fig.15.

Denote P as the polygonal domain and P_s is the sampling points set from the boundaries where s is the sampling spacing. Make MA_s denotes the Medial axis skeleton extracted from $Vor(P_s)$, Voronoi graph constructed based on the sampling points. $DT(P_s)$ is the corresponding Delaunay graph of $Vor(P_s)$.

Observation 3.3.1. All the Voronoi edges dual to Delaunay edges that are incident to two distinct vertices

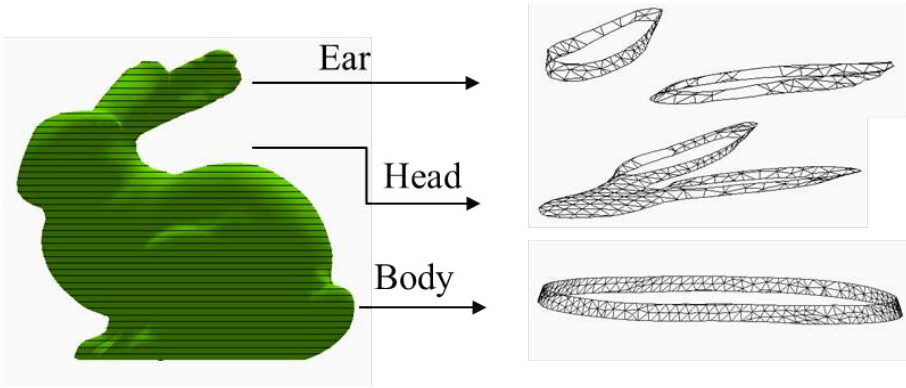
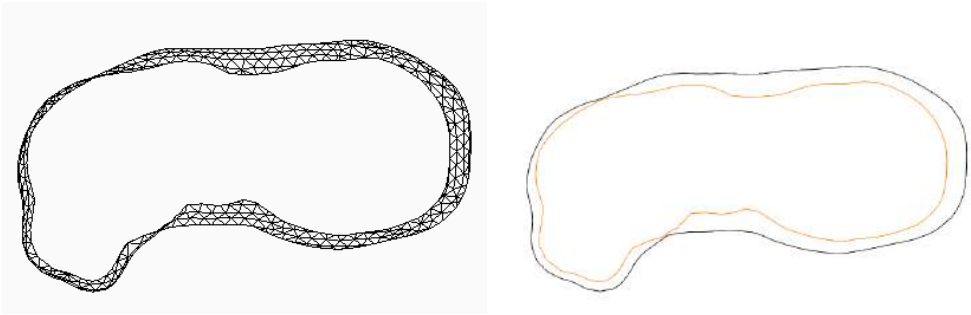


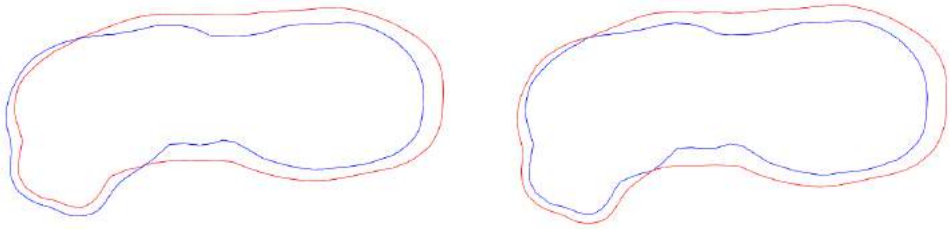
Figure 11: Slice meshes at several height



(a) Top view on 3D slice mesh

(b) The boundary pair

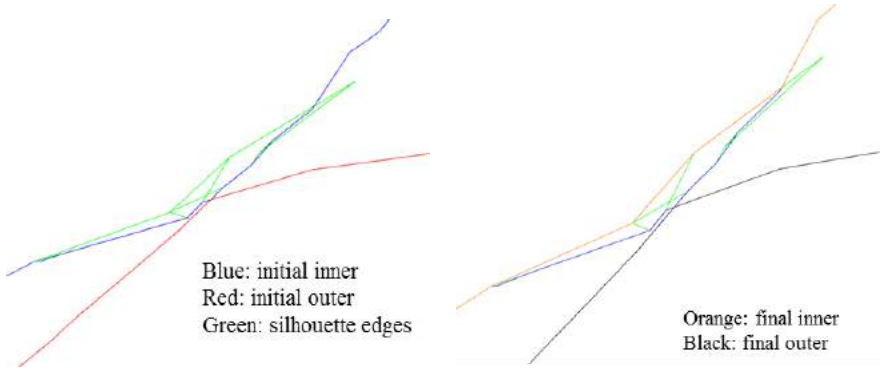
Figure 12: The slice instance and its boundary pair



(a) Before the 1st traverse

(b) After the 1st traverse

Figure 13: Before and after the first traversing stage



(a) Before the 2nd traverse (b) After the 2nd traverse

Figure 14: Before and after the second traversing stage

are identified as the components of Medial axis skeleton. Therefore if $dual(e_i) \in DT(P_s)$ has two endpoints located at different boundary loop, we conclude $e_i \in MA_s$ where $e_i \in Vor(P_s)$.

By observation 3.3.1, Fig.16 is the Medial axis skeleton extracted from the above Voronoi graph. When the bounded edges are quite close, the aliasing effect impairs quality of the approximate skeleton locally. Therefore, A simple fairing process is carried out by adding more sampling points locally, where the lower the local smoothness is, the denser sampling strategy is taken. An example in Fig.17 illustrates a partial skeleton before and after the simple fairing process.

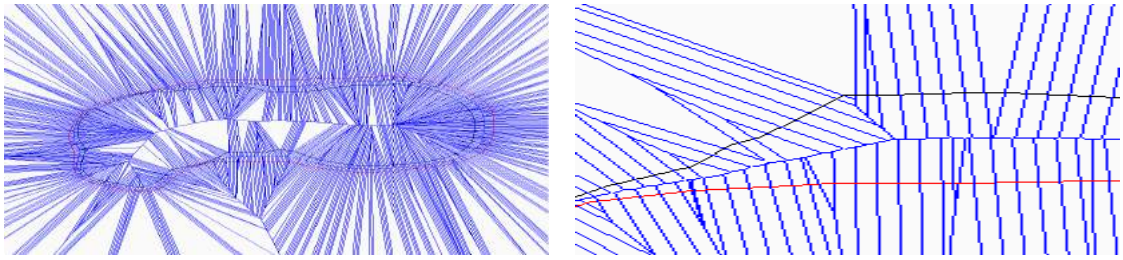
3.4 Error Computation

Both volumetric error and horizontal distance error can be measured theoretically. The volume deviation (Eqn.(1)) is calculated by the CAD software. And the distance error is measured from all the points on the 3D slice mesh to the extruded surface. All the vertices on the original 3D slice mesh are projected onto the horizontal plane and bounded by the boundary pair. Note that the maximum distance deviation must occur among the points located on the boundaries. And the distance from a point to the contour is essentially the distance from a point to a set of connected segments. Furthermore, since the Voronoi diagram provides a planar partition scheme where each sampling point on the boundaries is associated to a unique Voronoi cell. This promotes the efficiency on computing the distance error for the Medial axis contour. Generally, There are two cases, contributing points and non-contributing points, to be discussed.

1) Contributing points

The contributing points refer to sampling points associated to a unique (bounded or unbounded) Voronoi cell that contributes to the MA contour. The contributing points achieve its minimum distance error at its Voronoi edges adjacent to the new contour. In Fig.18, points P_1, P_2, P_3, P_4 and P_5 are sample points on the boundaries. The Voronoi edges on the new contour are drawn by red lines and others are dashed. The dashed blue lines are corresponding Delaunay edges .

Denote $dist(P_i, MA_s)$ as the distance from a sampling point P_i to the MA contour. Since the Voronoi cell $Vor(P_1)$ is associated to P_1 and has Voronoi edges AB and BC on the MA contour. When Delaunay egdes P_1P_2 and P_1P_3 intersect Voronoi edges AB and BC, we have $dist(P_1, MA_s) = min\{\frac{1}{2}P_1P_2, \frac{1}{2}P_1P_3\}$. And when one or both of P_1P_2 and P_1P_3 do not intersect their dual Voronoi edges, either $dist(P_1, MA_s) = min\{P_1A, \frac{1}{2}P_1P_3\}$ (equivalently $dist(P_1, MA_s) = min\{P_1C, \frac{1}{2}P_1P_2\}$ or $dist(P_1, MA_s) = min\{P_1A, P_1B, P_1C\}$ holds. Since any edge on the MA contour is adjacent to two Voronoi cell, for each contributing point, we



(a) Partial Voronoi graph (b) Zoomed Voronoi graph

Figure 15: Voronoi diagram

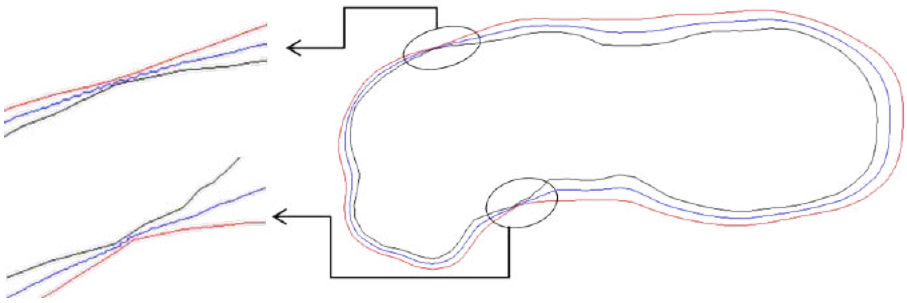


Figure 16: The Medial axis skeleton

can find at least one sample point equidistant or less distant to the MA contour as it. Therefore, minimizing maximum distance deviation is significantly guaranteed by the proposed Medial axis approximation strategy.

2) Non-contributing points

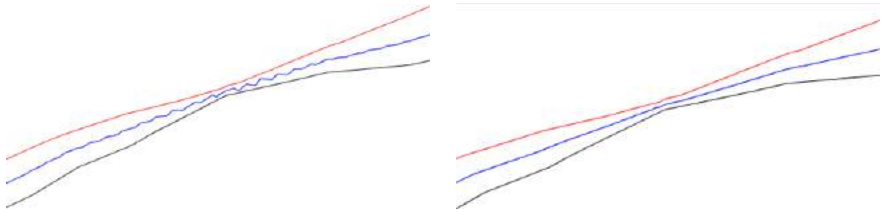
Some sampling points on the boundaries are not associated with a Voronoi cell that incident to any edge on the MA contour. To compute the corresponding distance error, a traversal to its neighboring Voronoi cells is necessary.

For instance illustrated in Fig.19, points $P_1, .., P_5$ are sample points on the boundaries. the Voronoi cell $Vor(P_1)$ doesn't contribute to the MA contour. Therefore, the edges accounting for $P_1's$ distance error are Voronoi edges adjacent to the contour and on the Voronoi cells $Vor(P_3)$ and $Vor(P_5)$. We have $dist(P_1, MA_s) = min\{P_1A, P_1B, P_1C, P_1D\}$ for this example. The non-contributing might produce a higher error than all the contributing points and hence dominate the surface tolerance. This requires further refinement on the MA contour.

The orange stippled line is the perpendicular bisector of P_1P_2, and it intersects Voronoi edge BF at points I. Note that P_2, P_3 and P_5 form a Delaunay triangle whose circumscribed circle does not contain P_1. Since the reflex vertex P_1 is located between $Vor(P_3)$ and $Vor(p_5)$, $dist(P_1, MA_s) = P_1B$ holds in this case. A basic refinement strategy is to move vertex B to the location of intersection point I. Notice that only three Voronoi cells, namely $Vor(P_2)$, $Vor(P_3)$ and $Vor(P_5)$, are influenced. Consider the distance of the sampling points on the boundaries to the modified contour. We get two observations as following:

Observation 3.4.1. $dist(P_3, MA'_s) < dist(P_3, MA_s)$ and $dist(P_5, MA'_s) < dist(P_5, MA_s)$ hold, where MA'_s is the refined contour.

Proof. When P_2P_3 intersects with Voronoi edge AB (or $\angle P_3AB < \frac{\pi}{2}$), As B moves to I, $dist(P_3, MA'_s)$ is equal to the distance from P_3 to AI. Since $\angle P_3AI < \angle P_3AB < \frac{\pi}{2}$, $dist(P_3, MA'_s) < dist(P_3, MA_s)$ hold.



(a) Before fairing (b) After fairing

Figure 17: Partial Medial axis skeleton

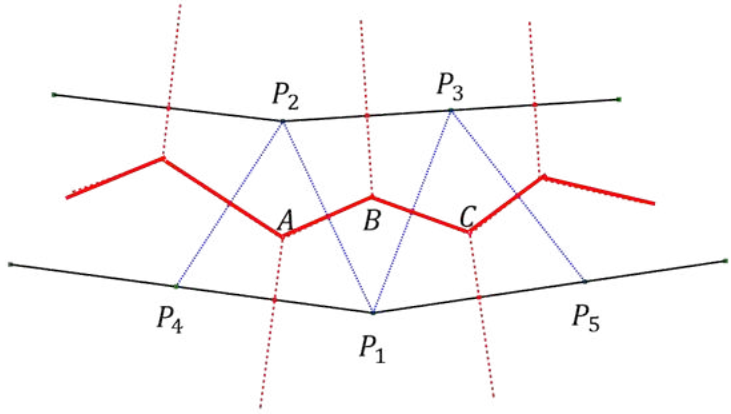


Figure 18: Contributing points instance

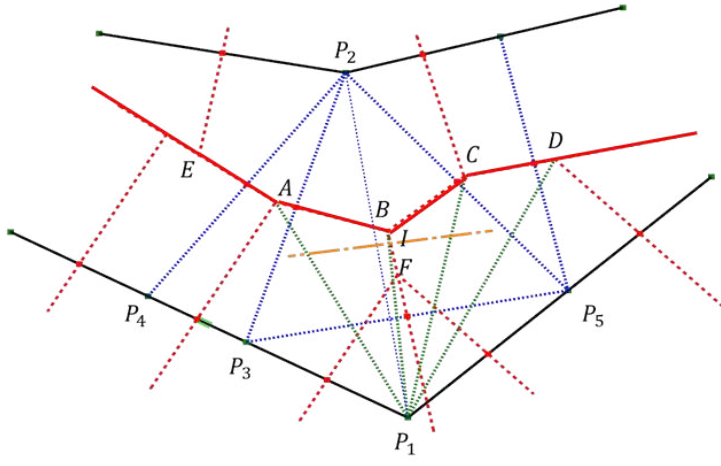


Figure 19: Non-contributing points instance

On the contrary, when P_2P_3 does not intersect with Voronoi edge AB (or $\angle P_3AB > \frac{\pi}{2}$), $dist(P_3, MA'_s) = min\{AP_3, IP_3\} < dist(P_3, MA_s)$. The same logic applies for point P_5.

Observation 3.4.2. The maximum distance deviation decreases since $dist(P_2, MA_s) < dist(P_2, MA'_s) < dist(P_1, MA'_s) < dist(P_1, MA_s)$ holds.

Proof. For point P_2, we have $dist(P_2, MA_s) = min\{\frac{1}{2}P_2P_4, \frac{1}{2}P_2P_3, \frac{1}{2}P_2P_5\}$ when the Delaunay edges EA, AB and BC intersect with corresponding Voronoi edges. If $dist(P_2, MA_s)$ is achieved at edge AB, after movement, P_2P_3 and P_2P_5 might no longer intersect with AI and CI respectively. Thus, $dist(P_2, MA'_s) = min\{\frac{1}{2}P_2P_4, P_2A, P_2I, P_2C\} > dist(P_2, MA_S)$. Otherwise, $dist(P_2, MA'_S) = dist(P_2, MA_S)$. For reflex point P_1, since I is closer to Voronoi vertex F in $Vor(P_1)$, we get $dist(P_1, MA'_s) = P_1I < P_1B$. In light of the fact $P_1I = P_2I$, we arrive at $dist(P_2, MA) < dist(P_2, MA'_s) < dist(P_1, MA'_s) < dist(P_1, MA_s)$.

Notice that as long as $dist(P_2, MA_s)$ is achieved at Voronoi edge EA, vertex B is allowed to move even closer to P_1 along the edge BF. And the maximum distance error can be reduced more. This gives us a basis for our refinement strategy as in Fig.20. Denote Θ as the maximum distance error among all the contributing points. The red points are non-contributing points, among which P_1, P_2 and P_3 are the endpoints on the edges. To define the modification range on the original MA contour, the two nearest contributing points P_4 and P_5 on both sides are located first. Since the contour is a closed profile, two Voronoi cells $Vor(P_4)$ and $Vor(P_5)$ must be incident to the common Voronoi vertex S. On the exact Medial axis tree, P_1 and P_3 are two leave nodes connecting to the truck by vertex S.

In Fig.20, L_1 and L_2 are two perpendicular lines passing through Voronoi vertex A. The space between L_1 and L_2 accounts for the distance error of P_3. R_1 and R_2 accounts for the distance error of Q_3 and Q_4. The remaining non-contributing points are accounted by the Voronoi vertex S. Therefore, it suffices to limit the refinement range on the Voronoi edges of $Vor(P_4)$ and $Vor(P_5)$. Voronoi vertices A and B define the refinement range. By drawing circles of radius Θ centered at P_1 and P_3, we obtain intersection points C and D between SP_3/SP_1 and corresponding circles. As a feasible scheme, the refinement edges AB, CD, DB replace the original Voronoi edges bounded by A and B.

The above strategy also needs to take the influence of sample points P_6 and P_7 on the opposite side into consideration, which means we need to ensure their distance error are controlled by the un-modified edges on the contour. Otherwise, the strategy cannot guarantee error reduction. Unfortunately, depending on the complexity of the neighboring geometry, and in particular when the polygonal domain has higher genus, this approach does not minimize the error in same cases. Therefore, in our implementation this variation was not implemented, and the exact error is measured based on the basic MA contour.

3.5 Complexity Analysis

Assume n is the number of vertices on the CAD model. As noted above, the time complexity of the uniform slicing stage is $O(nlogn)$. And for a slice, the complexity of projecting and updating the mesh topology is $O(nlogn + Ilogn)$, where I is the size of intersection points. Construction the MA skeleton is dominated by Voronoi diagram computation. Denote the minimum sample spacing as δ and the longest edge on the boundary pair as l_{max}. The size of sampling points is bounded by $\frac{l_{max}}{\delta}n$. Thus, Voronoi diagram could be constructed in $O(\frac{l_{max}}{\delta}nlogn)$ and composed of $O(\frac{l_{max}}{\delta}n)$ Voronoi cells. By Eular's formula, to traverse Voronoi graph to extract Medial axis contour consumes linear time to number of Voronoi edges, namely $O(\frac{l_{max}}{\delta}n)$.

In conclusion, the overall algorithm time complexity is $O(\frac{h}{t}\frac{l_{max}}{\delta}nlogn)$ for a model of height h and layer thickness t.

4 CASE STUDY

To evaluate the improvement in terms of part accuracy and deposition efficiency, a Stanford bunny model is taken as the benchmark. Several key parameters are given: Height ($5cm$), Length ($5.21cm$), Width ($3.97cm$),

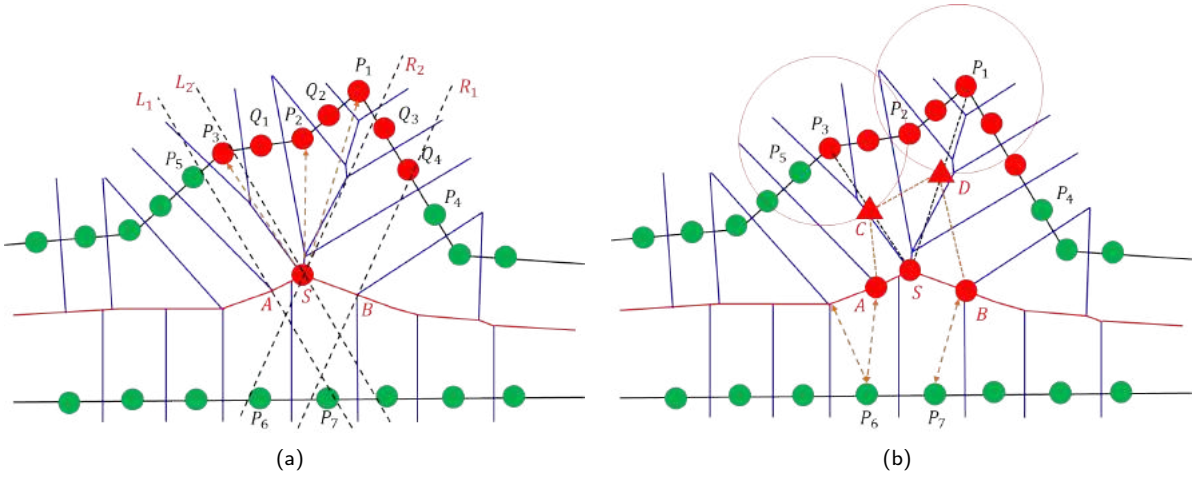


Figure 20: A feasible refinement scheme for more general cases

model components (5036 vertices, 30204 edges, and 10068 faces), and total volume ($28.148cm^3$), shown in Fig.21 . An available FDM machine (UP Plus 2) provides layer thickness range from $0.15mm$ to $0.4mm$ with step $0.05mm$. We firstly adopt the minimum thickness $0.15mm$ to evaluate part accuracy improvement. The distance errors of the lower contour and the MA contour at each layer are visualized in the scatter chart in Fig.22. The traditional strategy presents an average $0.0307cm$ and maximum $0.160cm$ distance error while the MA contour gives an average $0.0178cm$ and maximum $0.107cm$. Hence, average 47.3% distance error reduction is achieved. Note that there are about 2.1% defect cases (at height around $4.75cm$ on the model) where the distance error of the MA contour is not bounded by that of the corresponding lower contour due to the non-contributing vertices. A bunch of slice samples are taken to verify the volume deviation, shown in the histogram of Fig.23. Among the slice samples, the volume deviation reduction ranges from 14.3% to 49.1%, 42.4% on average. Intuitively, the slice mesh that contains more vertical faces (opposite to slope faces) along the build orientation receives less improvement by the contour reconstruction. On the contrary, the MA contour strategy makes difference in feature recognition improvement, for example, the slices at $2.94cm$ and $3.15cm$.

With the average error reduction more than 40%, we are allowed to use a larger layer thickness under the MA contour strategy while keeping the accuracy within the tolerance. Therefore, we make the distance error of $0.15mm$ under the traditional strategy as the benchmark. In the Fig.24 and Fig.25, the distance errors for the slice geometry deposited by $0.25mm$ and $0.2mm$, where the slice contours are reconstructed by the MA contour strategy, are calculated for the entire part. As a result, the MA contour of $0.2mm$ reduces the distance error by 24% on average and $0.25mm$ achieves 8.5% error reduction. With the layer thickness $0.2mm$, all layers excluding the 4.4% defect cases, where the further refinement is required, are bounded by the benchmark. Therefore, we can simply pick the $0.2mm$ as a feasible thickness. This suffices to reduce the number of layers from 334 to 250 and hence leads to up to 25% deposition time reduction.

Finally, we simply construct an extruded CAD model (see Fig.26) by extruding each contour by layer thickness. For the purpose of model re-design, a more elegant is to build the new model by the mean of contour interpolation[4].

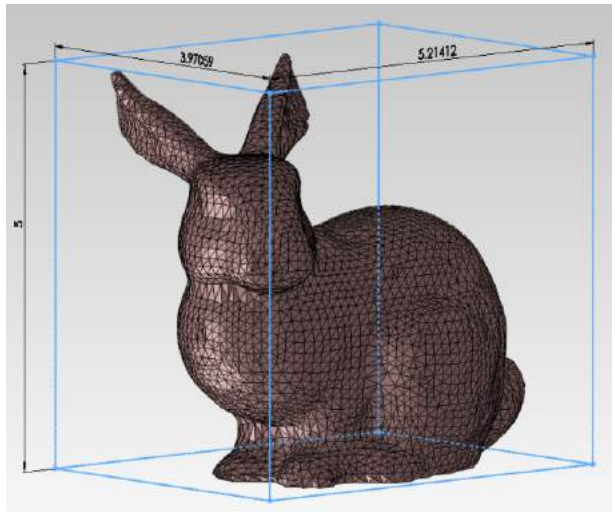


Figure 21: The Stanford bunny model

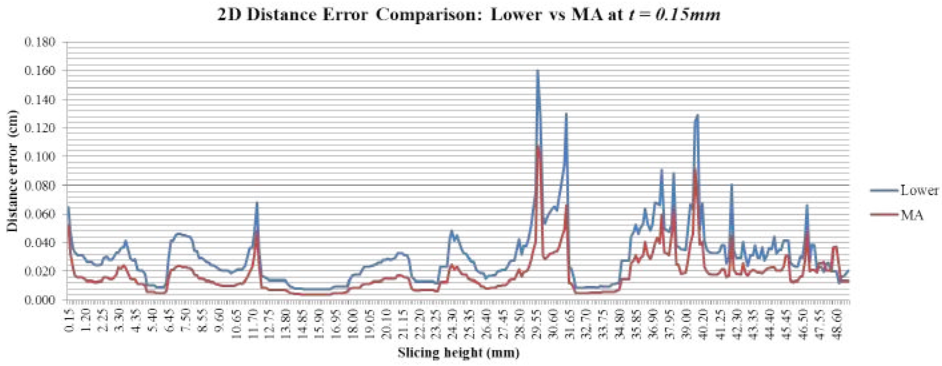


Figure 22: 2D distance error: Lower vs MA at $0.15mm$

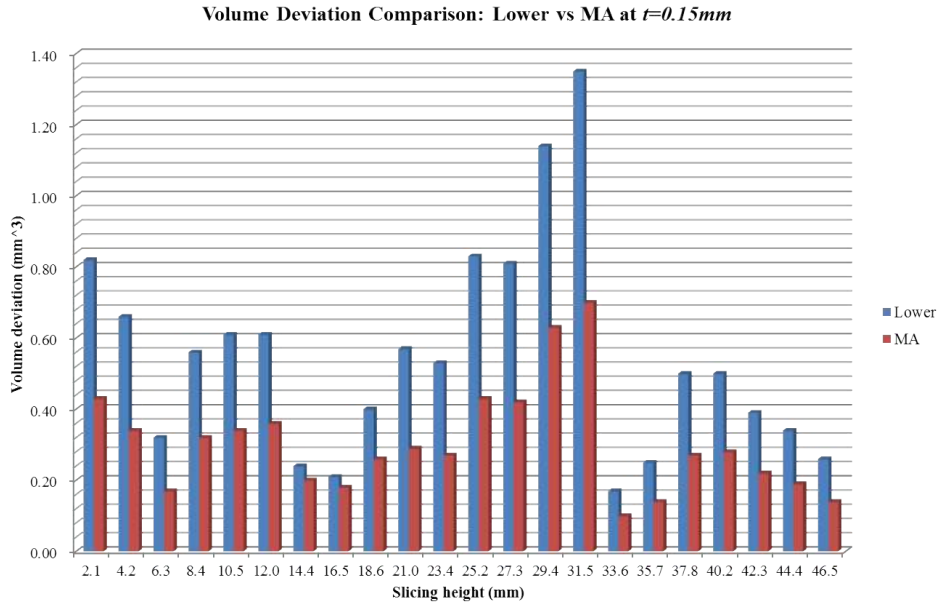


Figure 23: Volume deviation: Lower vs MA at $0.15mm$

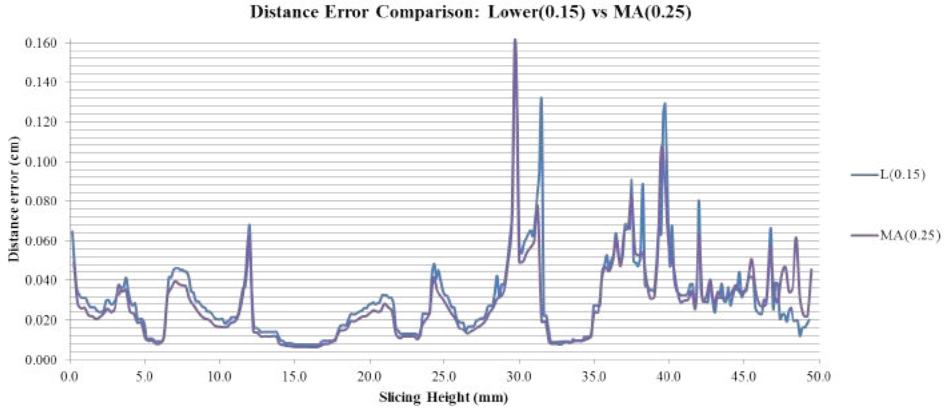


Figure 24: Distance error: Lower($0.15mm$) vs MA $0.25mm$

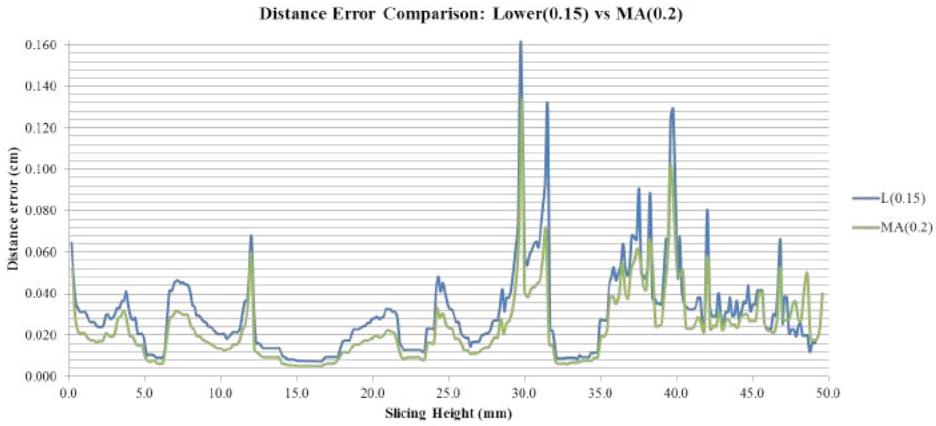


Figure 25: Distance error: Lower($0.15mm$) vs MA $0.20mm$

Figure 26: The extruded bunny model

5 CONCLUSIONS

In this paper, a novel adaptive slicing strategy based on contour reconstruction is implemented based on traditional uniform slicing system. The approach adopts simple, efficient and robust geometric techniques to construct new contour for each slice on the model. The results of case studies show the MA contour yields a more than 40% error reduction while keeping the printing efficiency. Furthermore, it leads to 20%-30% time reduction by implementing the new contour with the slices of larger layer thickness. The new contour data, or a new constructed model based on the new contours, can be directly feeding into the LM system for fabrication. There are several directions for future works. First, In our implementation, we adopt the Voronoi diagram of sampling points to approximate the Medial axis. A robust and accessible computation scheme for exact Medial axis computation might improve the accuracy and efficiency further. Secondly, A general and rigorous refinement scheme in terms of non-contributing vertices is a potential research topic. Finally, by the shortest path search algorithm in anisotropic regions[8], an optimal contour might be directly searched on the 3D slice mesh surface.

ACKNOWLEDGEMENTS

This research was supported in part by RGC GRF grant 16213519 and the Departments of IEDA, and MAE in HKUST.

ORCID

Ajay joneja, http://orcid.org/0000-0002-6797-1253
Kai Tang, http://orcid.org/0000-0002-5184-2086

REFERENCES

[1] Aichholzer, O.; Aigner, W.; Aurenhammer, F.; Hackl, T.; Jttler, B.; Rabl, M.: Medial axis computation for planar freeform shapes. Computer-Aided Design, 41(5), 339–349, 2009. http://doi.org/10.1016/j.cad.2008.08.008.

[2] Aichholzer, O.; Alberts, D.; Aurenhammer, F.; Grtner, B.: Straight skeletons of simple polygons. In Proc. 4th Internat. Symp. of LIESMARS, 114–124, 1995.

[3] Aichholzer, O.; Aurenhammer, F.: Straight skeletons for general polygonal figures in the plane. In International Computing and Combinatorics Conference, 117–126. Springer, 1996. http://doi.org/10.1007/3-540-61332-3_144.

[4] Barequet, G.; Goodrich, M.T.; Levi-Steiner, A.; Steiner, D.: Contour interpolation by straight skeletons. Graphical Models, 66(4), 245–260, 2004. http://doi.org/10.1016/j.gmod.2004.05.001.

[5] Bium, H.: A transformation for extracting new descriptions of shape. In Symposium on Modeis for the Perception of Speech and Visua1 Form, 1967.

[6] Brandt, J.W.: Convergence and continuity criteria for discrete approximations of the continuous planar skeleton. CVGIP: Image Understanding, 59(1), 116–124, 1994. http://doi.org/10.1006/ciun.1994.1007.

[7] Chen, J.S.S.; Feng, H.Y.: Contour generation for layered manufacturing with reduced part distortion. The International Journal of Advanced Manufacturing Technology, 53(9), 1103–1113, 2011. http://doi.org/10.1007/s00170-010-2886-x.

[8] Cheng, S.W.; Na, H.S.; Vigneron, A.; Wang, Y.: Approximate shortest paths in anisotropic regions. SIAM Journal on Computing, 38(3), 802–824, 2008. http://doi.org/10.1137/06067777X.

[9] Chua, C.K.; Leong, K.F.; Lim, C.S.: Rapid prototyping: principles and applications. World Scientific, Reading, MA, 2010. http://doi.org/10.1142/6665.

[10] de Berg, M.; Cheong, O.; van Kreveld, M.; Overmars, M.: Computational Geometry: Algorithms and Applications. Springer Science & Business Media, 2008. http://doi.org/10.1007/978-3-540-77974-2.

[11] Dolenc, A.; Makela, I.: Slicing procedures for layered manufacturing techniques. Computer-Aided Design, 26(2), 119–126, 1994. http://doi.org/10.1016/0010-4485(94)90032-9.

[12] Fortune, S.: A sweepline algorithm for voronoi diagrams. In Proceedings of the second annual symposium on Computational geometry, 313–322. ACM, 1986. http://doi.org/10.1145/10515.10549.

[13] Galantucci, L.M.; Lavecchia, F.; Percoco, G.: Experimental study aiming to enhance the surface finish of fused deposition modeled parts. CIRP Annals-Manufacturing Technology, 58(1), 189–192, 2009. http://doi.org/10.1016/j.cirp.2009.03.071.

[14] Guibas, L.J.; Knuth, D.E.; Sharir, M.: Randomized incremental construction of delaunay and voronoi diagrams. Algorithmica, 7(1), 381–413, 1992. http://doi.org/10.1007/BF01758770.

[15] Hope, R.J.P., RL; Roth: Adaptive slicing with sloping layer surfaces. Rapid Prototyping Journal, 3(3), 89–98, 1997. http://doi.org/10.1108/13552549710185662.

[16] Huang, B.; Singamneni, S.B.: Curved layer adaptive slicing (clas) for fused deposition modelling. Rapid Prototyping Journal, 21(4), 354–367, 2015. http://doi.org/10.4028/www.scientific.net/AMM.446-447.263.

[17] Javidrad, F.; Pourmoayed, A.: Contour curve reconstruction from cloud data for rapid prototyping. Robotics and Computer-Integrated Manufacturing, 27(2), 397–404, 2011. http://doi.org/10.1016/j.rcim.2010.08.008.

[18] Karavelas, M.I.: Voronoi diagrams in cgal. In 22nd European Workshop on Computational Geometry (EWCG 2006), 229–232, 2006.

[19] Kumar, C.; Roy Choudhury, A.: Volume deviation in direct slicing. Rapid Prototyping Journal, 11(3), 174–184, 2005. http://doi.org/10.1108/13552540510601309.

[20] Lee, D.T.: Medial axis transformation of a planar shape. IEEE Transactions on Pattern Analysis and Machine Intelligence, (4), 363–369, 1982. http://doi.org/10.1109/TPAMI.1982.4767267.

[21] Luo, R.C.; Tzou, J.H.: Implementation of a new adaptive slicing algorithm for the rapid prototyping manufacturing system. IEEE/ASME Transactions on Mechatronics, 9(3), 593–600, 2004. http://doi.org/10.1109/TMECH.2004.835332.

[22] Majhi, J.; Janardan, R.; Smid, M.; Gupta, P.: On some geometric optimization problems in layered manufacturing. Computational Geometry, 12, 219–239, 1999. http://doi.org/10.1016/S0925-7721(99)00002-4.

[23] McGuire, M.: The half-edge data structure, 2000. See http://www.flipcode.com/articles/article halfedgepf.shtml.

[24] Mohan Pandey, P.; Venkata Reddy, N.; Dhande, S.G.: Slicing procedures in layered manufacturing: a review. Rapid prototyping journal, 9(5), 274–288, 2003. http://doi.org/10.1108/13552540310502185.

[25] Reeves, P.E.; Cobb, R.C.: Reducing the surface deviation of stereolithography using in-process techniques. Rapid Prototyping Journal, 3(1), 20–31, 1997. http://doi.org/10.1108/13552549710169255.

[26] Sabourin, E.; Houser, S.A.; Helge Bhn, J.: Accurate exterior, fast interior layered manufacturing. Rapid Prototyping Journal, 3(2), 44–52, 1997. http://doi.org/10.1108/13552549710176662.

[27] Sabourin, S.A., Emmanueland Houser; Helge Bhn, J.: Adaptive slicing using stepwise uniform refinement. Rapid Prototyping Journal, 2(4), 20–26, 1996. http://doi.org/10.1108/13552549610153370.

[28] Shamos, M.I.; Hoey, D.: Closest-point problems. In International Computing and Combinatorics Conference, 151–162. IEEE, 1975. http://doi.org/10.1109/SFCS.1975.8.

[29] Suh, Y.S.; Wozny, M.J.: Adaptive slicing of solid freeform fabrication processes. In Solid Freeform Fabrication Symposium, 404–411. DTIC Document, 1994.

[30] Tyberg, J.; Helge Bhn, J.: Local adaptive slicing. Rapid Prototyping Journal, 4(3), 118–127, 1998. http://doi.org/10.1016/S0261-3069(99)00012-6.

[31] Zhao, Z.; Luc, L.: Adaptive direct slicing of the solid model for rapid prototyping. International Journal of Production Research, 38(1), 69–83, 2000. http://doi.org/10.1080/002075400189581.