Automated cross-sectional shape recovery of 3D branching structures from point cloud

Jacob Kresslein, Payam Haghighi, Jaejong Park, Satchit Ramnath, Alok Sutradhar, Jami J. Shah

PII: S2288-4300(17)30117-3
DOI: https://doi.org/10.1016/j.jcde.2017.11.010
Reference: JCDE 119

To appear in: *Journal of Computational Design and Engineering*

Received Date: 13 July 2017
Revised Date: 14 November 2017
Accepted Date: 15 November 2017

Please cite this article as: J. Kresslein, P. Haghighi, J. Park, S. Ramnath, A. Sutradhar, J.J. Shah, Automated cross-sectional shape recovery of 3D branching structures from point cloud, *Journal of Computational Design and Engineering* (2017), doi: https://doi.org/10.1016/j.jcde.2017.11.010

AUTOMATED CROSS-SECTIONAL SHAPE RECOVERY OF 3D BRANCHING STRUCTURES FROM POINT CLOUD

Jacob Kresslein, Payam Haghighi, Jaejong Park, Satchit Ramnath, Alok Sutradhar, Jami J. Shah

Department of Mechanical and Aerospace Engineering, The Ohio State University, Columbus, OH 43210, USA

Corresponding Author: Jacob Kresslein, Kresslein.1@osu.edu

ABSTRACT

Many applications rely on scanned data, which can come from a variety of sources: optical scanners, coordinate measuring machines, or medical imaging. We assume that the data input to these applications is an unorganized point cloud or mesh of vertices. The objective may be to find particular features (medical diagnostics or reverse engineering) or comparison to some reference geometry (e.g. dimensional metrology). This paper focuses on the feature fitting of a segmented point cloud, specifically for branched, organic structures or structural frames, and targets non-monolithic geometries. In this paper, a methodology is presented for the automated recovery of cross-sectional shapes using centrally located curves. We assume a triangulated surface mesh is generated from the scanned point cloud. This surface mesh is the starting point for our methodology. We then find the curve skeleton of the part which abstractly describes the global geometry and topology. Next after segmenting the curve skeleton into non-branching segments, orthogonal planes to the curve skeleton segments, at preset or adaptive intervals, make slices through the surface mesh edges. The intersection points are extracted creating a 2D point cloud of the cross section. A number of application specific post-processing operations can be performed after obtaining the 2D point cloud cross sections and the curve skeleton paths including: calculations such as area or area moments of inertia, feature fitting or recognition, and digital shape reconstruction. Case studies are presented to demonstrate capabilities and limitations, and to provide insight into appropriate uses and adaptations for the presented methodology.

Keywords: Digital Shape Reconstruction; Shape Recovery; Metrology; CAD; Feature Recognition; Medical Imaging

1. INTRODUCTION

3D scanning is used to construct digital models out of physical objects. Many types exist including contact and non-contact scanners. An example of a contact scanner is the coordinate measuring machine (CMM) which is primarily used in manufacturing and can provide very precise surface point cloud data. There is growing interest in 3D scanning devices as new developments in reverse engineering, rapid prototyping, quality control, and rapid product design emerge [1-3]. In manufacturing, CMM machines and laser scanners are used to quickly and accurately assess and capture dimensional data. This is accepted as one of the most effective ways to establish and maintain manufacturing process control. Once a manufactured part is scanned, various metrological procedures can measure its features and check the conformance to specified design tolerances. This is currently done by analyzing point cloud data that is obtained on the part. To that end, point cloud data is overlaid on the idealized geometry represented in a CAD model. Recent efforts have been made in the research of feature fitting and recognition for CMMs [4-8]. Measurements taken from these fitted geometries are often limited to pre-defined features. No generalized method for metrology of free-formed shapes exists. Current metrology of such parts is performed manually and relies on the experience of the CMM machine user or quality control engineers. Measurements are typically limited to feature lengths, angles, and diameters. In the case of branching parts or assemblies, extraction of cross-sectional shapes and sizes for the purposes of metrology and quality control is outside of current software capabilities.

Medical imaging has become a valuable tool to physicians for diagnostics, with imaging being used across a wide range of specializations. Beyond diagnostics, these images are widely used in the education and training of medical professionals [9]. 3D computed tomography (CT) scans make use of many x-rays recombined into a 3D image. A non-invasive imaging modality,

3D CT angiography, can provide vital information capturing the forms of blood vessels. This is particularly useful for the physicians to interactively characterize aneurysms [10]. Significant work has been done in the area of automated image segmentation of anatomical structures, with much of the work being aimed at recognizing these anatomical features, such as aneurysms, tumors, vessels, or lesion [11]. While many structures in anatomy take a branching, organic form, such as arteries, veins, and nerves, no method exists to our knowledge for the automated extraction of cross-sectional shapes of these anatomical features.

Reverse engineering often involves digital shape reconstruction. This is used to create digital models from physical objects represented by point cloud data. This technology provides an alternative route to obtaining computer aided design (CAD) models of a physical object that would otherwise require painstaking manual modeling efforts, particularly for complex geometries. This may be performed on manufactured parts or prototypes which did not utilize CAD technology [12, 13]. The introduction of 3D scanners has made it possible to collect many points from the surface of a physical object and has enabled digital shape reconstruction to come to fruition. The typical first step in the shape reconstruction workflow is triangular mesh generation which puts the unorganized scanned point data into a more useful form. Prismatic objects can be recreated with sketch-base operations and combined with Boolean operations. To accomplish this, many feature fitting algorithms are have been explored [5]. Free-form objects, however, are often recreated with fitted parametric surfaces [14]. While many advances in digital shape reconstruction have been made in recent years, few methodologies for reconstruction of branching organic objects or structural frames exist. Goyal et al. [15] addressed this problem for branching free-form reconstruction by extracting prominent cross sections using locally linear embedding and affinity propagation and organizing them into sweep components. This approach,

however, requires a very large number of cross sections to accurately represent the shape. In digital shape reconstruction, the need for reconstructing scanned branching, organic geometries or structural frames as they would have intuitively been constructed using CAD operations, i.e. as lofted cross sections along a path, remains unmet by current solutions.

This paper presents a generalized methodology to automate recovery of sweep paths and 2D cross sections from point cloud data for branching or structural (truss-like) shapes. Recommendations for how the methodology can be adapted and recommendations for post-processing operations of the extracted 2D cross-sectional shapes for application-specific use are discussed and explored. This procedure takes a watertight surface mesh as a starting point that can be generated from 3D point cloud data using a variety of point cloud noise filtering and surface mesh generation from point cloud algorithms [16, 17]. Four case studies with various shapes and different applications are discussed in Section 3. A case study for a simple truss illustrates the potential to use our methodology for cross-sectional area calculation as a post-processing step and considers alternative approaches to fitting paths. Bike frame and control arm case studies offer details for the performance of the proposed methodology for different feature types, such as features with small aspect ratios or features containing holes, and how future developments may address certain limitations and improve the quality of the extracted data. Finally, a case study for a CT scan of an Aorta and its major branches indicates how the cross-sectional shapes of a highly tortuous object can be successfully extracted.

2. METHODOLOGY

In this section, a methodology for cross-sectional shape extraction is formulated with the assumption that the object can be largely represented in CAD as one or more lofts along generalized sweep paths of the object and the orthogonal cross sections to those paths. A

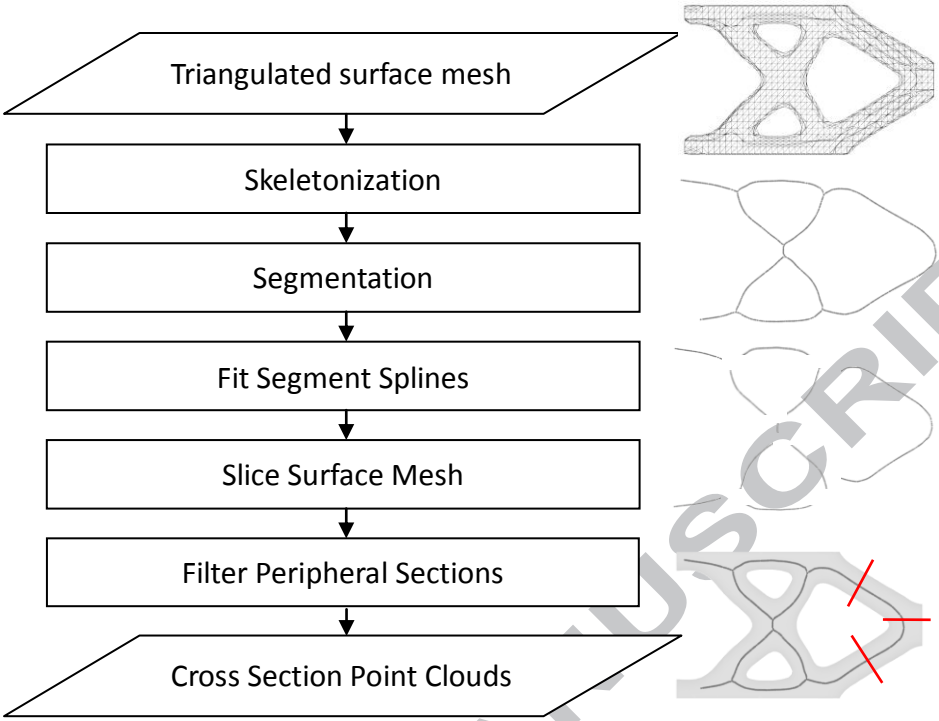


Fig. 1. Section Recovery Methodology Overview.

skeletonization algorithm is utilized to obtain these paths represented by connected vertices. The curve skeleton is then segmented at its junctions to obtain non-branched curve skeleton segments. A geometric modeling kernel (i.e. ACIS Modeler [18]) was then leveraged to fit B-splines to each curve skeleton segment based on its vertices. Planes can then be positioned orthogonally to these fitted splines spaced at fixed intervals or adaptively along the length of the curve. The intersection of these planes and the edges of the surface mesh is used to generate a 2D point cloud. Problem-specific criteria can be used to determine the number of planes used in each segment and the planes' spacing along the curves. A clustering algorithm which leverages adjacency given within the surface mesh is used to filter out the noise points of the sampled cross section. Noise occurs due to the infinite plane intersecting with the peripheral geometry of the part not associated with the current curve skeleton segment. Optionally, post-processing

operations can be performed to extract additional information from the 2D point clouds or fit cross sections, depending on the application. For example, 2D point clouds can be made into faces to extract cross-sectional information such as area, moments of inertia, and principal inertial axes. Each of these steps is discussed in greater detail in the remainder of this section. The major steps in this methodology are summarized in Figure 1.

2.1 Skeletonization: Skeletonization is a method of shape abstraction which has useful applications in shape segmentation and reconstruction, as it describes both topology and geometry of a shape. The Blum medial axis [19] is the most known skeleton structure; however, its major limitation is its sensitivity to small surface perturbations. Due to this sensitivity, sweep paths were alternatively represented as curve skeletons in this paper. Curve skeletons can generally be defined as 1D structures locally centered in a shape [20]. While many definitions for curve skeletons and approaches to curve skeleton generation exist, each has its limitation, several of which are examined in detail in [21]. A surface contraction based method described and made available by Tagliasacchi *et al*. [22], which utilizes mean curvature flow, was implemented in this work for its computational scalability and invariance to surface perturbations. This approach begins with a watertight, triangulated surface mesh as the input and results in a curve consisting of vertices with linear connections.

A Voronoi diagram is initially computed from a watertight surface mesh. The mesh is then iteratively contracted via mean curvature flow. Local re-meshing ensures higher uniformity during contraction. Contraction iterations continue until the volume of the shape is zero. Then an edge collapse operation on the zero-volume manifold generates the final curve skeleton. The vertex connectivity will first be used as a basis for segmentation

of the branching curve skeleton into non-branching curve skeleton segments. The curve skeleton segments' vertices will then be used to create a smooth, B-spline curve for each

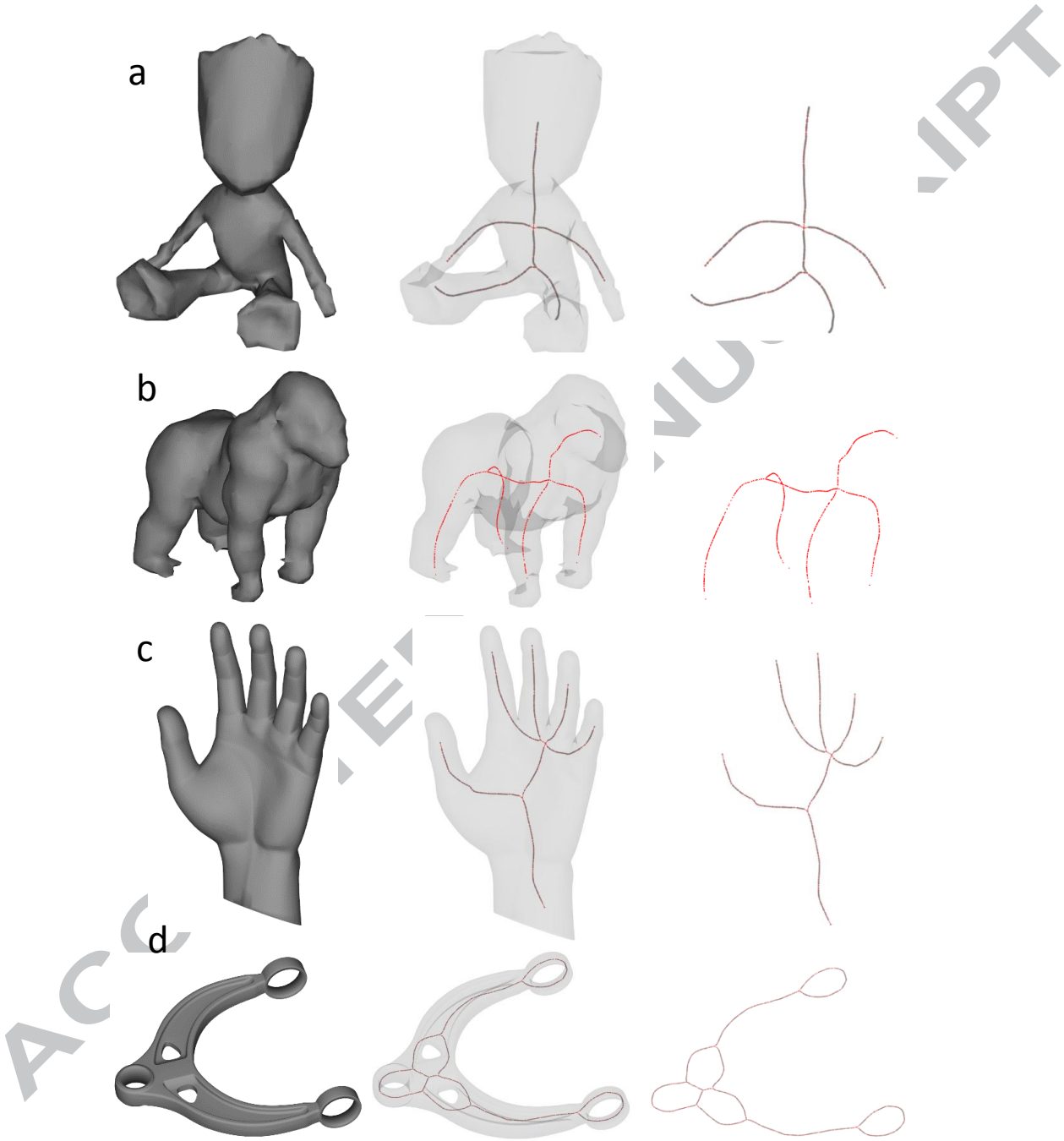


Fig. 2. Skeletonized Objects a) figurine b) gorilla c) hand d) control arm.

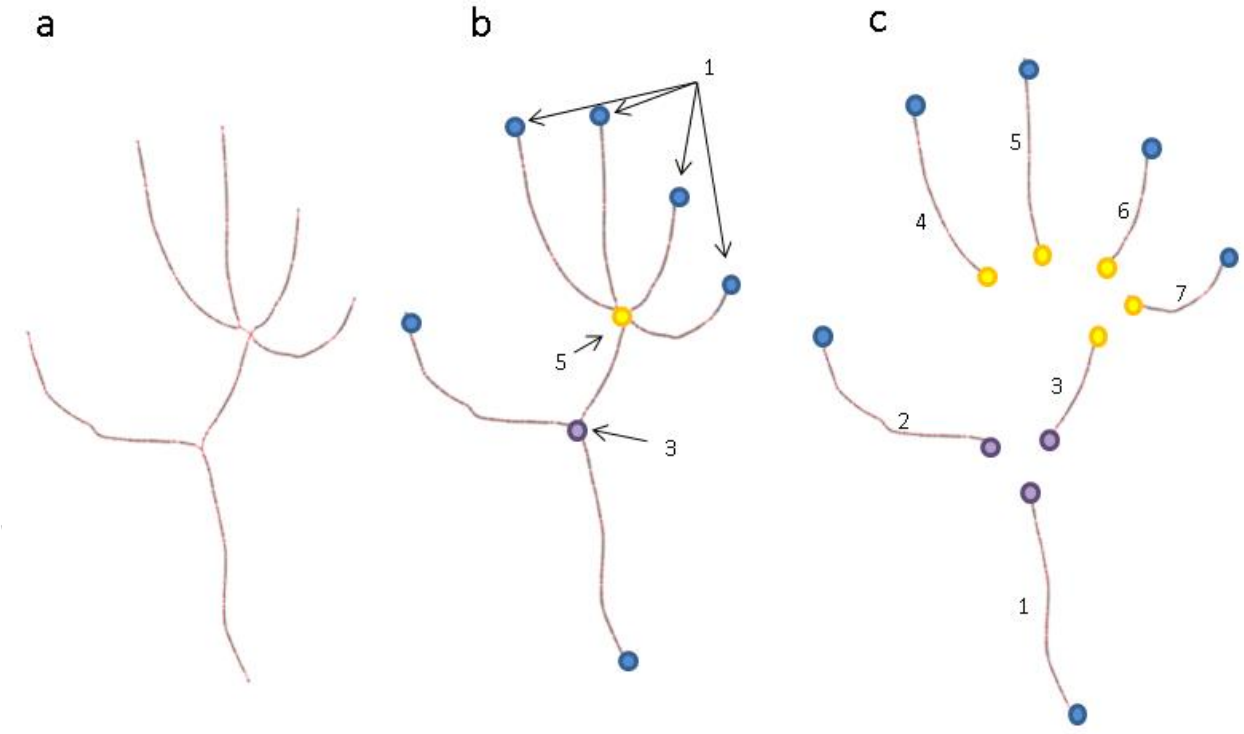


Fig. 3. Segmentation operation a) curve skeleton b) connectivity degrees of junctions

and ends c) segmented curve skeleton into seven curve skeleton segments

segment. Figure 2 gives examples of curve skeletons generated using mean curvature flow [23-26].

2.2 Segmentation: For each branch of the object to be sampled for cross sections individually, the curve skeleton is broken into curve skeleton segments. The segmentation algorithm described here considers the curve skeleton in the form of a graph, i.e. all vertices and the connection between them are available, but vertices are not necessarily ordered. We first define the *connectivity degree* for each vertex to be the number of vertices which are connected to that vertex by one edge. The curve skeleton is segmented at any vertex where the connectivity degree is greater than 2, which are the *junction* vertices. We define *end* vertices to be vertices with degree 1. A curve segment can, therefore, be bounded by *junction* or *end* vertices. Figure 3 illustrates the connectivity degrees of a curve skeleton and how they lead to segmentation. For the skeleton shown, there are two *junction* vertices, one of them is degree 3, and the other is degree 5 (Figure3(b)). The vertices of degree 1 are also shown. Figure 3(c) shows seven curve skeleton segments that are generated as a result of segmenting the overall curve skeleton at both of the junctions.

Algorithm 1. : Segmentation of the Curve Skeleton

```
vertex_list = get_skeleton_vert
edge_list = get_skeleton_edge
connectity = get_connectivity_degree(vertex_list,edege_list)
while all_edge_visited == FALSE
     new skeleton_segment
     vertex_temp == find_next_terminal_vertex(connectivity)
     if connectivity(vertex_temp) != visited_vertex_count(vertex_temp)
          vertex_temp = find_next_terminal_vertex(connectivity)
     else
          skeleton_segmement.add(vertex_temp)
          visited_vertex_count(vertex_temp) = ++1
     while connectivity(vertex_temp) == 2
          (vertex_temp,edge_temp) = find_unvistited_connection(vertex_temp)
          skeleton_segment.add(vertex_temp)
          visited_vertex_count(vertex_temp) = ++1
          visited_edge_count(edge_temp) = ++1
```

Starting with unordered lists of vertices and edges in the curve skeleton graph, the following pseudocode in Algorithm 1 outlines the segmentation algorithm. The algorithm relies on tracking the number of times vertices and edges are visited during new curve skeleton segment building. Vertices can be visited during this process a number of times equal to the connectivity degree. Edges can each be visited once. All curve skeleton segments are found when the edges have each been visited.

2.3 Spline-fitting: B-spline fitting of the curve skeleton segments not only provides the reference for the slicing plane spacing and orientations for each curve skeleton segment but can also serve as the sweep path for lofting CAD operations along a curve path in digital shape reconstruction. To avoid error due to the discrete nature of the curve skeleton segments' representations as connected vertices and connecting edges, a B-spline curve is fit to the points on the curve skeleton segments to generate a smooth curve. This parameterization allows the direction of curvature at any given point on the smooth curve to be easily computed. This is beneficial in the following step, where a

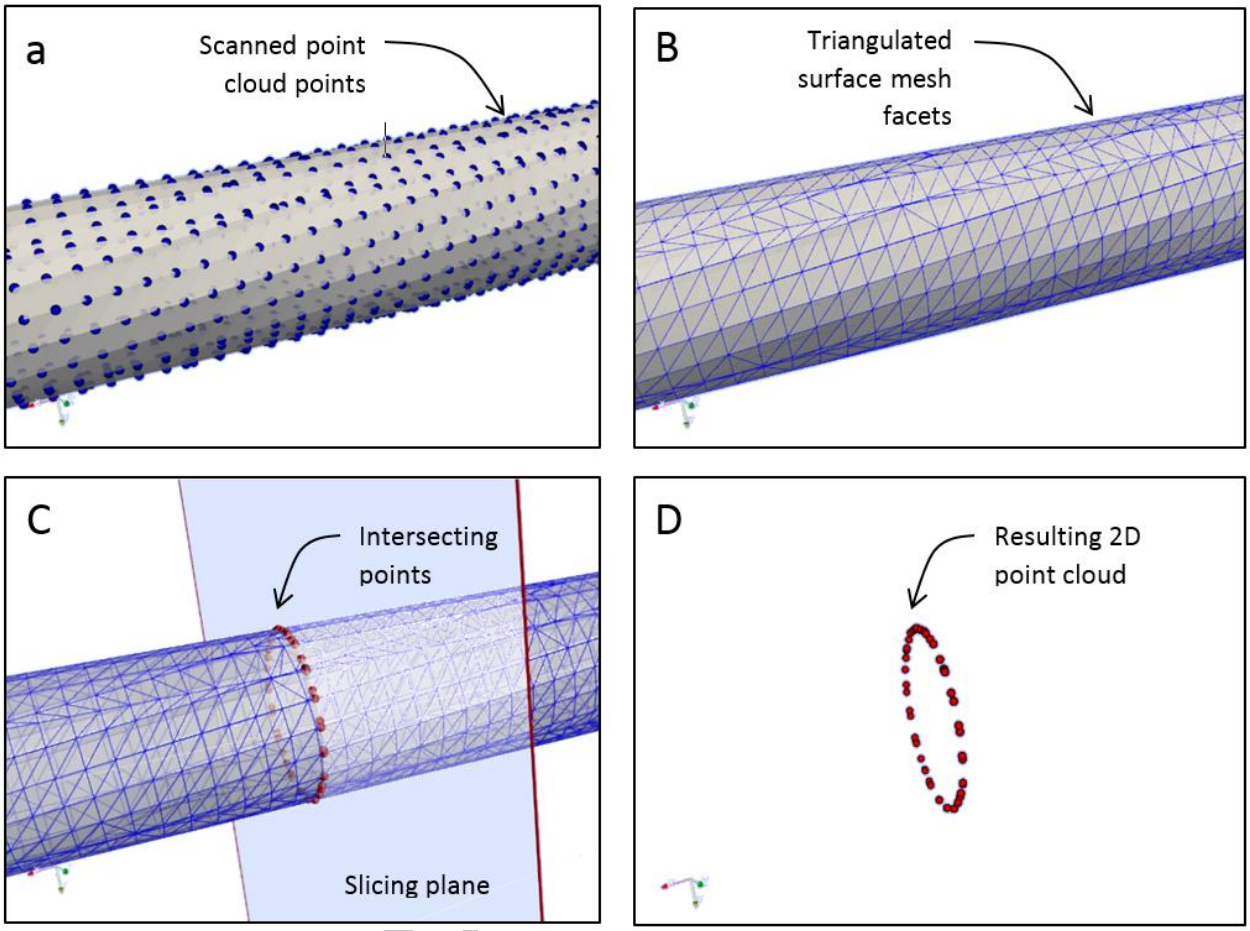
point on the curve skeleton and the direction vector at that point are used to define a slicing plane orthogonal to the curve skeleton.

In this paper, we use a B-Spline interpolation of all the points on the curve skeleton segment to form the smooth curve. However, the decision for the type of curve to fit can be application-specific. For objects with branches/members with standard paths, such as straight or rounded paths, least squares operations can be used to fit these path types to the curve skeleton segment vertices. For example, a truss with straight paths would benefit from fitting straight lines to the curve skeleton segments.

2.4 Facet Edge Slicing: A 2D point cloud representing a cross section can be obtained by slicing the object normal to the fitted curve skeleton segment. The slicing plane can be defined by a point on the fitted segment and the curvature direction vector at that point. The appropriate locations of the orthogonal planes along the curve skeleton and the

Fig 4. Plane Slicing. (a) Scanned point cloud (b) Surface Mesh (c) Sampling Plane intersecting surface mesh edges (d) 2D point cloud of intersection between edges

number of planes in each curve skeleton segment are problem-specific parameters. This can be hard-coded for a set numbers of planes and intervals of spacing, as implemented in this paper, or adaptive algorithms can be developed for requirements such as accuracy. The points on the curve skeleton segments splines used to define the plane locations will be called *center points* since they are centrally located within their corresponding cross sections which are to be sliced.



One approach for acquiring these 2D cross-sectional shapes, which we call the plane-projection approach, is to find vertices of the mesh within a close distance from the plane and project them onto the sampling plane. Alternatively, in what we call the plane-slicing approach, we can find all the edges in the mesh that intersect a plane then find the location of these intersections on the plane. The plane-slicing approach will give a precise representation of the geometry since a point cloud will be directly generated from the cross section of the surface mesh at the slicing location. In contrast, the plane-projection approach is not a true cross-section sample of the surface mesh and is more likely to give a fuzzy representation of the object's cross section. Because the edge slicing approach gives a precise representation of the cross section, this method was chosen over the plane projection approach.

Figure 4(a) and 4(b) first show an initial scanned part represented as a 3D point cloud and the triangulated mesh from the point cloud, respectively. Figure 4(c) illustrates a slicing plane through the triangulated mesh and the intersection points of the intersection of the plane and the edges of the surface mesh. These intersection points give rise to the 2D point cloud of the cross section (Figure 4(d)).

Plane slicing inherently faces issues due to sampling occurring on an infinite plane. This method can potentially sample points belonging to a curve skeleton segment that is not currently being considered, as seen in Figure 5(a) and (b). In the figure, there are 4 segments. The current segment being considered is segment 2; however, segments 3 and 4 are intersected by the slicing plane. Additionally, if there is large curvature in the curve skeleton segment, the slicing plane could intersect multiple locations along the same curve skeleton segment. Figure 5(c) shows a slicing plane which intersects segment 4 in two locations.

For a given plane, all slices other than the one that envelops the current center point are composed of "noise points", as shown in Figure 6. These noise points should be filtered out and excluded from the 2D point cloud representation of the section shape. The most obvious approach would be to use a proximity criterion measuring distance from each of the points to the curve skeleton point and exclude points outside a threshold distance. This distance would be difficult to generalize and is not likely to account for all cases. Another approach could be to implement a surface segmentation algorithm [27-29] which will associate the surface mesh vertices with a particular curve skeleton segment. Since surface segmentation can be complex and are not the focus of this paper, this approach was not implemented. A simple, yet robust solution is to form clusters of all the

intersected edges for a slicing plane based on the edge connectivity. The clusters' average distances to the center point can then be calculated. Only edges which belong to the closest cluster are used to extract the 2D point cloud for that slicing plane, and the noise points are excluded. The clustering filter implemented is discussed in detail in the following subsection.

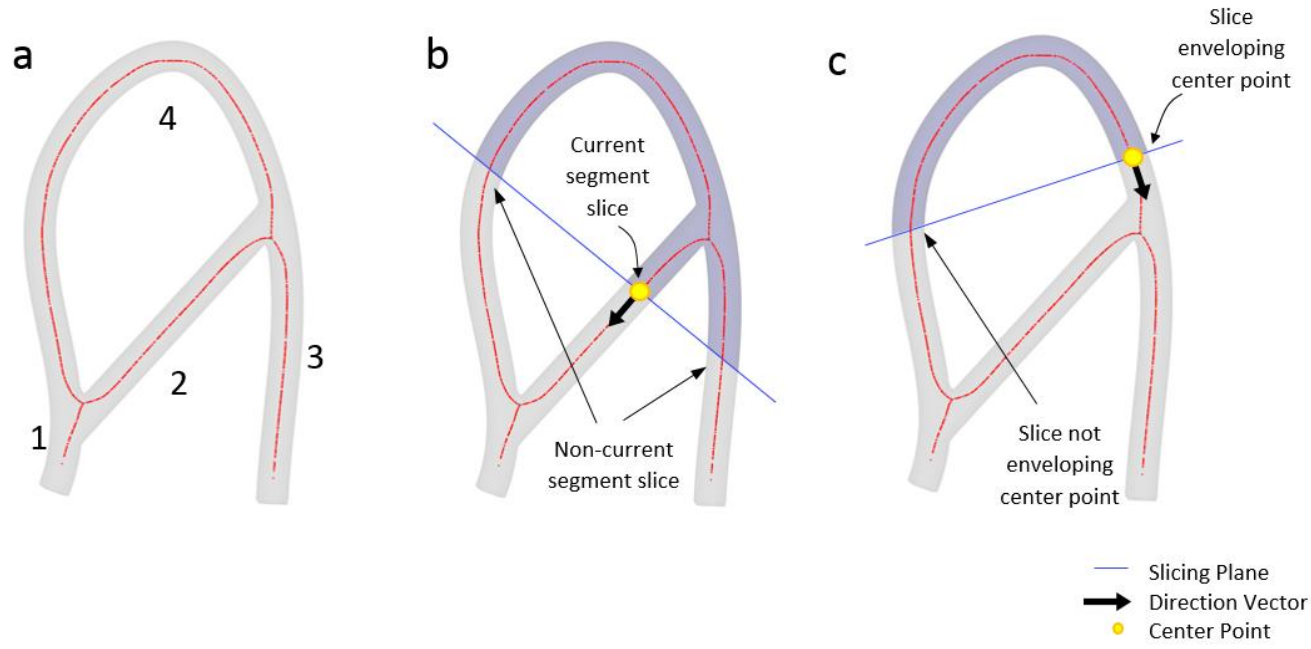


Fig. 5. Potential Infinite Plane Sampling Problems (a) Four segments (b) Non-current segment slicing (c) Multiple self-slicing due to large curvature.

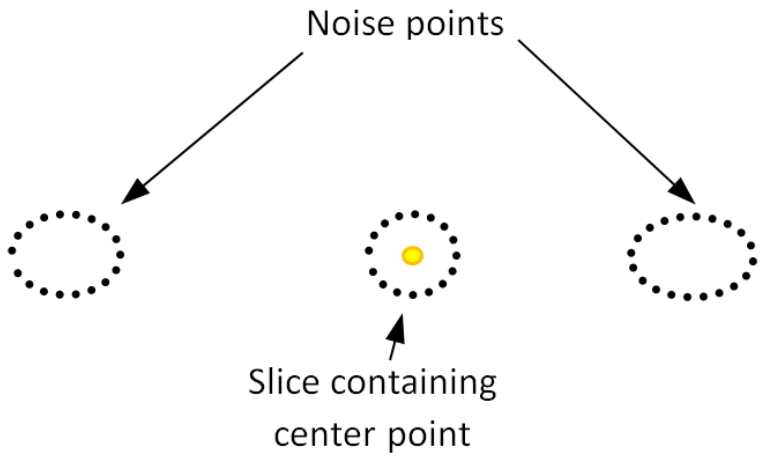


Fig. 6. 2D point cloud for the slicing plane in Figure 6(b)

2.5 Peripheral Edge Clustering Filter: To have a single slice associated with each slicing plane, noise points should be excluded (as shown in Figure 6). A clustering filter algorithm given in Algorithm 2 pseudocode below was developed to achieve this. The algorithm utilizes a flag vector, a queue, and clusters of edges. After all of the edges that are intersected by the slicing plane are found, a flag vector with indices for each of these edges is created. The flag vector serves to track which edges still need to be added to the queue. When the queue is not empty, all of the edges that share a vertex with the edge at the head of the queue are found. If these edges are not flagged, i.e. they have not been added to the queue, they are pushed to the queue and flagged. The queue head is then popped and added to the current cluster. This is repeated while the queue is not empty. If the queue is empty, a new cluster is created and the first unflagged edge in the flag vector is pushed to the queue. This process continues until every edge is flagged and the queue is empty, meaning that every edge has been added to the cluster to which it belongs. Because the edges sharing

Algorithm 2. : Clustering Filter

```
edge = get_edges
center_point
direction = get_direction_vector(center_point)
get_sliced_edges(edge,center_point,direction)
while all_edges_flagged == TRUE
     if queue_empty == TRUE
          new cluster
          edge_temp = get_first_unflagged_edge
          flag(edge_temp)
          push.queue(edge_temp)
     else
          edge_temp = head.queue
          edges_w_shared_vertex = get_edges_sharing_vertex(edge_temp)
          if flag(edges_w_shared_vertex) == FALSE
              flag(edges_w_shared_vertex)
              push.queue(edges_w_shared_vertex)
          pop.queue
          add_to_current_cluster(edge_temp)
my_cluster_edges = get_closest_cluster(center_pt)
point_cloud_2D = get_sliced_edges(my_cluster,center_point,direction)
```

a vertex are added to the queue, and a new cluster cannot be created until the queue is empty, each cluster is formed in its entirety before moving onto the next cluster. Once all the clusters have been formed, the cluster that contains the center point can be identified. Point clouds are found at the intersections of the surface mesh and the slicing plane for each cluster. The cluster with the point cloud with the closest average distance to the center point is the cluster that contains the center point. This 2D point cloud data is then stored, and the 2D point cloud extraction process is repeated for every slicing plane in the object.

3. CASE STUDIES

A number of case studies demonstrate the capabilities of the proposed methodology. First, a simple truss structure is examined to demonstrate how our methodology extracts cross sections from objects with branching paths that are straight segments and have standard cross-sectional

shapes, in this case, rectangular cross sections. Additionally, using the extracted cross section shapes, the area is calculated and compared to known cross-sectional areas demonstrating a potential calculation that can be performed on the extracted shapes. Next, cross sections are extracted from a bike frame, which consists of mostly cylinders with large aspect ratios, but also contains certain members with small aspect ratios. The influence of these geometrical traits on cross section extraction is discussed. A control arm example provides a scenario where elongated, high aspect ratio features, as well as holes and ring-like features, are present. Lastly, a 3D CT scan offers an example of an object with highly tortuous branches and variable, irregular cross sections, an object with a geometry that would previously be difficult to automate cross section shape extraction. Together these case studies are used to represent capabilities and limitations, and to give indications for potential uses of the methodology. These case studies are implemented in C++ making use of the ACIS modeling kernel. Curve skeletons are generated using the StarLab application from [18]. All test cases were run on a PC with 2.8 GHz Intel Xeon E5-1603V4 CPU and 32.0 GB of RAM.

3.1 Simple Truss:

Starting with the uniform surface mesh of a simple truss structure, using the StarLab application a curve skeleton is generated, shown in Figure 7(b). This curve skeleton, represented as vertices and edges, is then segmented into non-branching curve skeleton segments. Eleven segments are found for this object. B-splines are then fit to each of the curve skeleton segments. Then orthogonal plane slicing can proceed. This example uses 5 slicing planes for each segment with the first and last plane being offset from the ends of the segment 15% of the segment length, ensuring junctions are not sliced. While many of the slicing planes intersect the peripheral geometry, they are successfully filtered out by

the clustering filter. The final output is a list of 2D point cloud coordinates. These coordinates can be mapped back to the original 3D space using the geometrical representation of the slicing planes and their working coordinate axes as was done for visualization purposes in Figure 7(c) and (d) and other point cloud output figures. While the scope of our methodology only gives the point cloud points, post-processing operations, such as cross-sectional area calculations, can be used in different applications as we discuss here briefly.

This first case study is an example of geometry with straight loft paths between constant cross sections within each of its members. The truss in Figure 7 has different cross-sectional areas between its different members and a constant thickness. After our code was used to extract cross sections from the truss, a convex hull was then taken from the 2D point cloud to construct an edge that bounds the face of the cross section using a gift-wrapping algorithm. This was an acceptable approach for this application since cross-sectional shapes were known to be convex. However, when convexity is not known or cross sections in the object are known to have concave regions, the edge created should not be approximated as the convex hull. An active contour would be a better approach for obtaining this edge [30].

Once the edge of the cross section is known, the area of this shape was then calculated by summing the areas of the triangles formed by each of the convex hull lines and the center point. The known cross-sectional areas are compared to the areas derived from the 2D point clouds using our methodology. Additional properties that could be calculated from this 2D shape are moments of inertia, the center of mass, or the principal inertial axes.

Due to the imperfect centeredness, and therefore small amounts of waviness, inherently present in any curve skeletonization technique, the sampling planes do not have exact orthogonal orientation leading to small deviations from the true areas. As discussed previously, when the curve paths are known to follow standard paths, such as straight lines, other curve fitting procedures can be performed. In this case, fitting straight lines may be advantageous over using B-Splines. Table 1 compares the calculated area, known area and corresponding percent errors. The sections compared are shown in Figure 7(b). Percent errors of the calculated areas vs. the known areas range from -2.5890% to 3.1028%. This object contained 7,171 vertices on its surface and had a CPU time of 5.842 seconds.

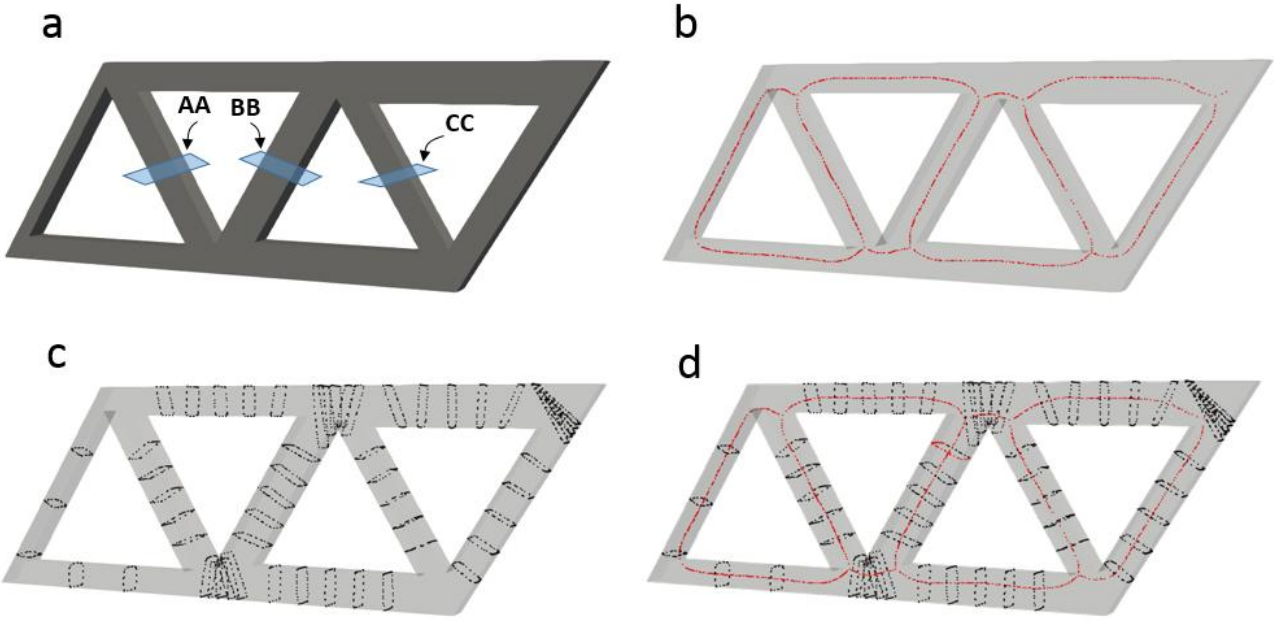


Fig. 7. Variable cross-section area extraction of truss a) slicing planes b) skeletonized truss c) extracted cross sections d) curve skeleton and extracted cross sections.

Table 1: Section Area Calculation Error

Section	Area (mm^2)	Area Calculated (mm^2)	Percent Error
AA	30	29.2233	-2.5890%

BB	35	34.0835	-2.6186%
CC	25	25.7757	3.1028%

3.2 Bike Frame: A bike frame is examined using the presented methodology shown in Figure 8. Again, five slices per segment were used with equal spacing while excluding 15% of the segment length from either end. This object can be considered as an assembly of connected beams. As seen in Figure 8(c), the circular cross-sections are extracted for most of the members. The sampling failed to give cross sections orthogonal to the cylinders' axes in areas that have small aspect ratios and in curve skeleton segments have a free *end* vertex that does not form a junction. For segments with free *ends*, the curve skeletons do not extend all the way to the surface, leading to loss of cross section information beyond these *ends*. This is an inherent trait of curve skeletons that could be addressed by extrapolating the curve to the surface in future developments. While these limitations exist, the approach still gives well-positioned cross-sections for much of the frame.

As seen in Figure 8, aspect ratios can have a significant impact on the quality of cross sections extracted. Figure 8(b) shows some curve skeleton segments belonging to the member with small and large aspect ratios. It can be observed in this figure that the small aspect ratio curve skeleton segments are more prone to deviations from the cylinder axis within the surface mess, while the large aspect ratio members give very straight and centered segments. Consequently, quality of the cross sections extracted is highly related to aspect ratios, with small aspect ratios resulting in elliptic sections while the large aspect ratios gave circular sections for the circular members indicated in Figure 8(b). This object contained 22,729 vertices on its surface and had a CPU time of 62.149 seconds.

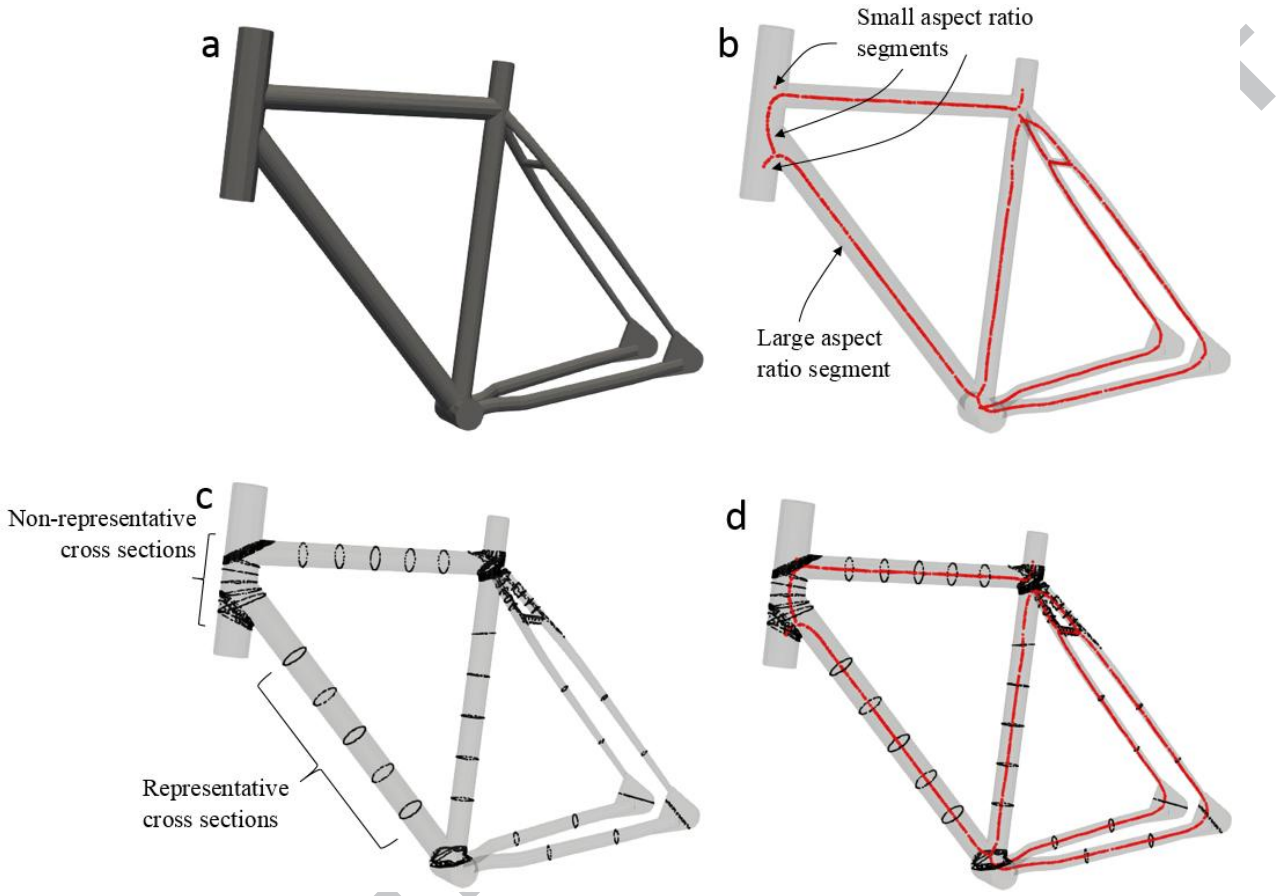


Fig. 8. Bike frame cross section extraction a) model b) skeleton c) extracted cross sections d) skeleton and sections.

3.3 Control Arm: A control arm, shown in Figure 9(a), offers a common scenario in which the object contains both elongated, beam-like features, small holes relative to the object, and ring-like features. While the object can be skeletonized properly (Figure 9(b)), we examine the quality of the sections in the beam-like, hole-like, and ring-like regions. In the two arms, these features result in representative cross sections using our code since these features were large aspect ratios and can intuitively be considered as sweeps along the curve skeleton segment curves. Sections taken from curve skeleton segments on the

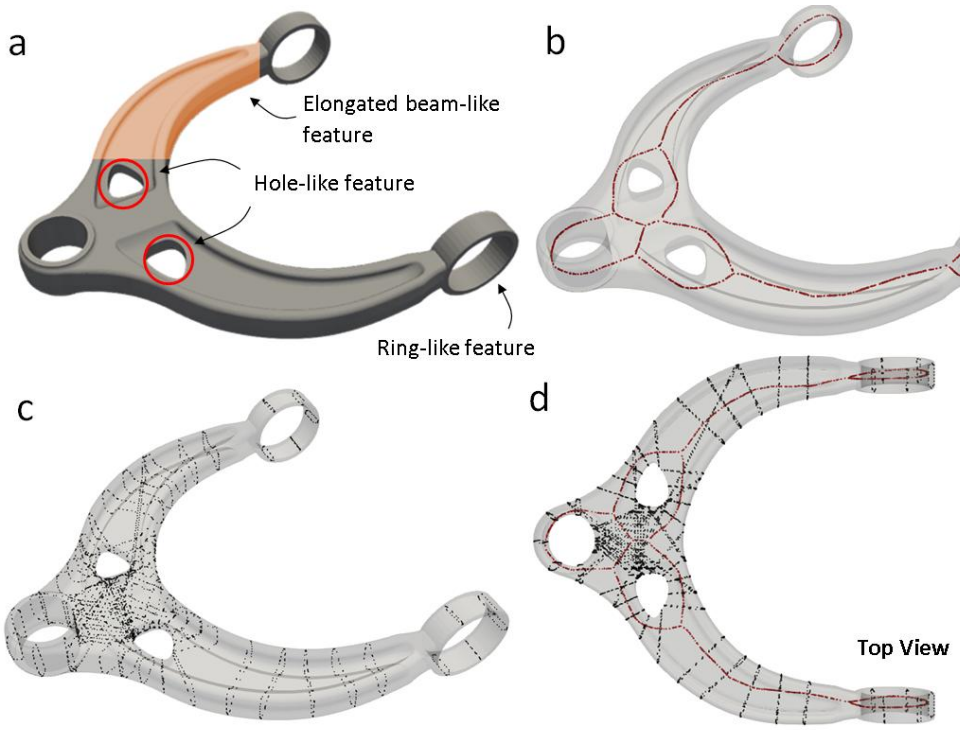


Fig. 9. Control arm cross section extraction a) model showing elongated, beam-like feature and hole b) skeleton c) extracted cross sections d) skeleton and sections (top view).

ring-like features gave quality cross sections as well for the same reason, i.e. the cross sections were small relative to the length of the curve skeleton segment length.

Curve skeleton segments near the hole-like features offer a different scenario. Here the holes are small relative to the part. As a result, there is a lot of material surrounding the holes. When cross sections are extracted from these segments, the cross sections often extend into neighboring segments while still forming a single cluster as can be seen in Figure 9(c) and (d). This is clarified further in Figure 10 with the cross section extraction of an object with holes. As shown in the figure, slices near the junctions are particularly prone to extending into other members while extracting sections. This can result in members that cannot be reconstructed as lofts along a generalized path and members with

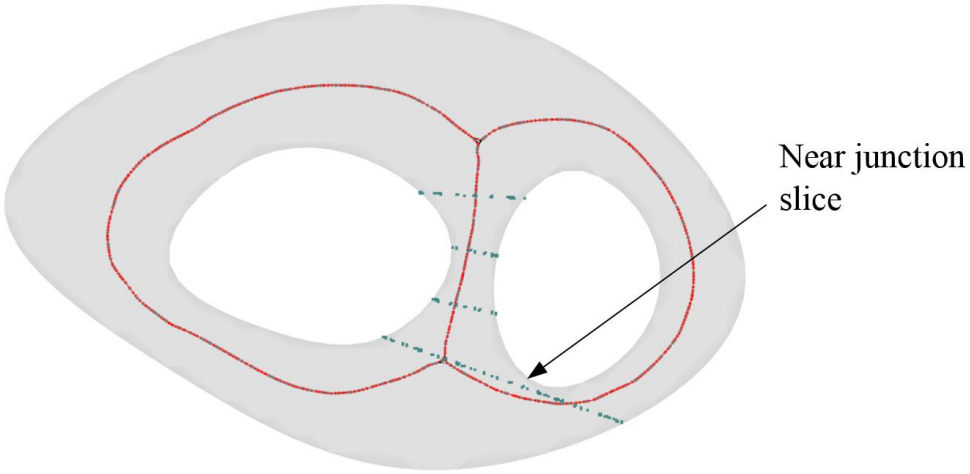
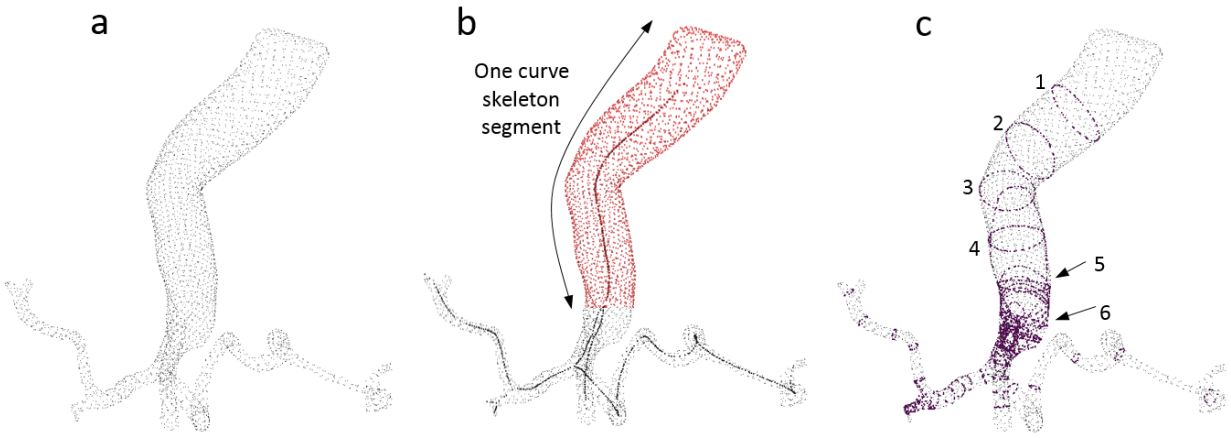


Fig. 10. Near junction sampling. Clustering filter does not exclude section slicing from neighbor skeleton segment because the slice is still one cluster.

the overlapping material. Sections taken from the segments describing these holes should not be examined using our methodology. This object contained 4,564 vertices on its surface and had a CPU time of 4.885 seconds.

3.4 Aorta: This case study offers an example of how our methodology may be used when point cloud data is obtained from a scan, such as 3D CT scan, of an arterial system. Figure 11 shows the resulting point cloud from a scan of the celiac artery and its branches as well as the slices produced using our code. As shown in the figure, this approach can handle tortuous, complex organic shapes observed in nature. For this object, a curve



skeleton is obtained which closely matches the 3D geometry of the network. The curve skeleton is broken at the junctions, creating 12 segments. It can be observed that due to the highly curved and branched nature of the object slicing planes intersect the object at several locations, but only the slice containing the center point was kept in the final output. Seventy-two 2D profiles were obtained, six for each segment. These slices were equally spaced while excluding 15% of the segment length on both ends. Using the sweep paths and 2D profiles that define the object's shape, a number post-processing procedures can be

implemented for various purposes. For instance, a cardiologist interested in patency of an artery at certain locations may use a 3D scan to assist in this assessment. Using the presented method, the shapes and areas can be determined at these separate locations giving crucial information about the internal sizes/shapes of the vessel. Furthermore, patient-specific blood flow modeling could be enabled by this technology. For patients with restricted blood flow due to plaque buildup, a physician can modify a personalized model used in blood flow simulation to get insight into how the blood vessel may be expanded to achieve adequate flow. This can also enable more intelligent and

personalized prosthetic implant design [31]. This object contained 13,602 vertices on its surface and had a CPU time of 44,835 seconds.

4. DISCUSSION

The proposed methodology can be presented as a proof-of-concept for cross section extraction and a non-application-specific tool. Depending on the application, steps in the process can be modified. For example, in the angiography case, many cross sections may be needed to accurately represent the scanned shape. For cases with many segments, it may be desirable to establish criteria for a minimum number of planes required to represent the scanned shape. If too many cross sections are used, digital shape reconstruction may be overly complex. Another modification, as discussed for the truss case study, could be to fit straight lines to the curve skeleton segments if the paths are known to be straight. The number of planes of planes may be changed depending on the application. While this was hard-coded in our implementation, an adaptive approach to determining the number of spacing of planes could be developed for specific applications. Still, for parts with known cross sections, such as the circular cross sections in the bike frame case study, circles could be fit to the extracted point cloud cross sections. As with any slicing algorithm, the accuracy of the 2D slice is only as accurate as the generated surface mesh since the 2D cross sections are exact slices of the surface represented by the mesh.

The presented test cases demonstrated capabilities of the section recovery methodology. While tortuous shapes can be reconstructed from extracted section shapes and curve skeletons, and cross-sectional properties such as area can be automatically calculated, certain limitations exist in the proposed methodology. A limitation arises due to inexact centeredness of curve skeletons. As seen in many of the figures, the curve skeleton is not perfectly centered in the part. This can lead to planes slicing the part which is not exactly orthogonal. This could be

problematic in applications such as metrology where the orientation that the measurement is taken is critically important. Future developments should allow the user to make minor modifications to the skeleton, such as filtering out small branches, smoothing or straightening segments, or manipulating the centeredness of the skeleton as scene in Figure 12. Another limitation discussed with our methodology relates to the section sampling near the joints or near holes. As shown in Figure 10, as planes are placed near the joints, cross sections extend into neighboring members. In future developments, these regions should be automatically identified and excluded from the cross section extraction, and the user should be alerted.

In summary, we present a methodology to automatically extract cross-sectional shapes from a surface mesh created from scanned point cloud data. Our approach leverages a contraction based skeletonization method which uses mean curvature flow. Curve skeletons are then segmented, and B-splines are fitted to each segment. Orthogonal planes lying on the fitted curve skeletons segments serve as cross section sampling planes. Planar point clouds are created on these planes by the intersection points of the edges of the surface mesh. A clustering filter algorithm is presented to exclude "noise" points. The methodology is assessed in the context of four case studies, examining various scenarios that may be encountered.

While the methodology is limited in its ability to represent shapes at their junctions as well as shapes that have members with small aspect ratios, we presented a methodology with potential for obtaining cross section shapes in large length to cross-sectional area aspect ratio, beam-like regions. Various post-processing steps can be implemented. In the future work, we plan to expand our investigation to junction handling for shape reconstruction.

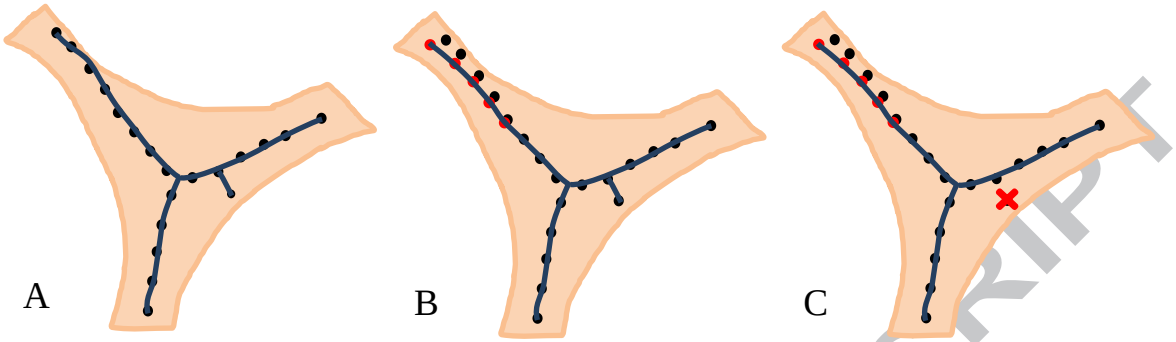


Fig. 12. Mesh editing application concept a) Before editing b) Re-centered c) Branch clean-up.

ACKNOWLEDGMENTS

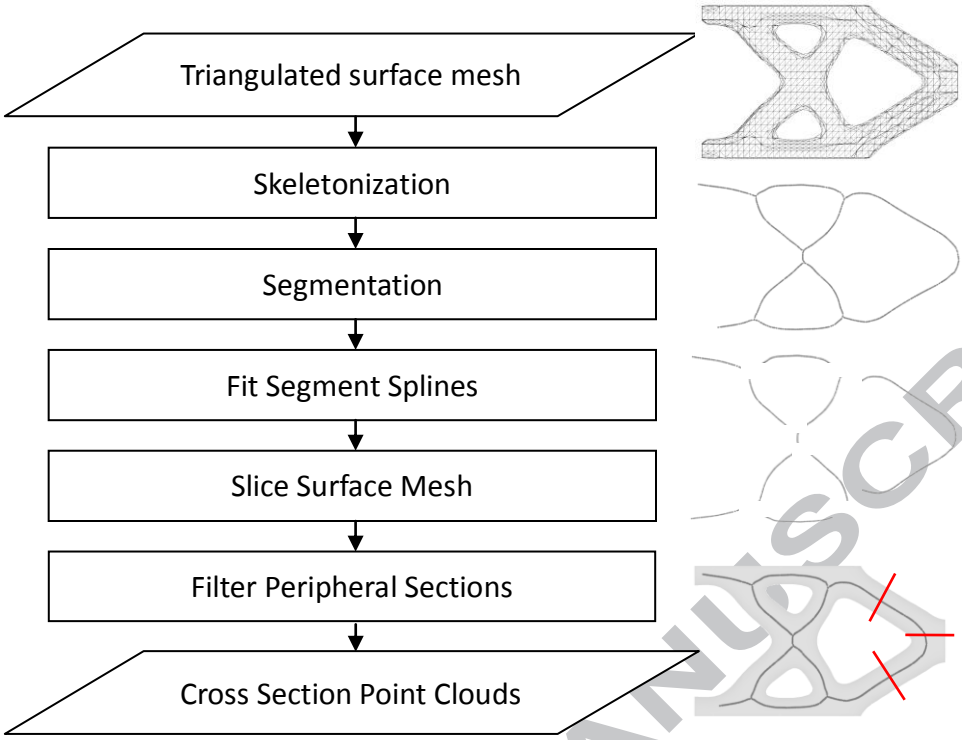
The authors would like to acknowledge the financial support from the Honda Research Institute North America and National Science Foundation through Award 1521801 and Veterans Affairs grant No. 5I01BX000418-06.

REFERENCES

[1] Jarng S, Yang H, Lee J. CAE Solid Element Mesh Generation from 3D Laser Scanned Surface Point Coordinates. Korean Journal of Computational Design and Engineering. 2005;10(3):162-7.
[2] Lee C-H, Park S-C. Scan Tool-Path Generation for Laser Pattern Machining. Korean Journal of Computational Design and Engineering. 2011;16(4):300-4.
[3] Park H-T, Chang M-H, Park S-C. Slicing a Point Cloud. Korean Journal of Computational Design and Engineering. 2007;12(2):146-52.
[4] Mohan P, Shah J, Davidson JK, editors. A library of feature fitting algorithms for GD&T verification of planar and cylindrical features. ASME 2013 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference; 2013: American Society of Mechanical Engineers.
[5] Mohan P, Haghighi P, Shah JJ, Davidson JK. Development of a library of feature fitting algorithms for CMMs. International Journal of Precision Engineering and Manufacturing. 2015;16(10):2101-13.
[6] Mohan P, Shah J, Davidson J, editors. Simulated and experimental verification of CMM feature fitting algorithms. ASME 2015 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference; 2015: American Society of Mechanical Engineers.
[7] Vemulapalli P, Mohan P, Shah JJ, Davidson JK, editors. User Defined Assembly Features and Pattern Recognition From STEP AP203. ASME 2014 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference; 2014: American Society of Mechanical Engineers.
[8] Mohan M. MATSUB:
Fuzzy logic based material substitution advisor for legacy parts 2011.
[9] Silén C, Wirell S, Kvist J, Nylander E, Smedby Ö. Advanced 3D visualization in student-centred medical education. Medical teacher. 2008;30(5):e115-e24.
[10] Kato Y, Sano H, Katada K, Ogura Y, Hayakawa M, Kanaoka N, et al. Application of three-dimensional CT angiography (3D-CTA) to cerebral aneurysms. Surgical neurology. 1999;52(2):113-22.
[11] Soler L, Delingette H, Malandain G, Montagnat J, Ayache N, Koehl C, et al. Fully automatic anatomical, pathological, and functional segmentation from CT scans for hepatic surgery. Computer Aided Surgery. 2001;6(3):131-42.
[12] Murshed M, Dixon A, Shah J. Neutral definition and recognition of assembly features for legacy systems reverse engineering. ASME Paper No DETC2009-86739. 2009.
[13] Mohan M. MATSUB: Fuzzy logic based material substitution advisor for legacy parts 2011.
[14] Varady T. Automatic procedures to create CAD models from measured data. Computer-Aided Design and Applications. 2008;5(5):577-88.
[15] Goyal M, Murugappan S, Piya C, Benjamin W, Fang Y, Liu M, et al. Towards locally and globally shape-aware reverse 3D modeling. Computer-Aided Design. 2012;44(6):537-53.
[16] HOPPE H, DEROSE T, DUCHAMP T, MCDONALD J, STUETZLE W, Cunningham S. SURFACE RECONSTRUCTION FROM UNORGANIZED POINTS. Siggraph 92 : Conference Proceedings. 1992;26:71-8. PubMed PMID: WOS:A1992BZ28L00009.
[17] Wolff K, Kim C, Zimmer H, Schroers C, Botsch M, Sorkine-Hornung O, et al. Point Cloud Noise and Outlier Removal for Image-Based 3D Reconstruction. Proceedings of 2016 Fourth International Conference on 3d Vision (3dv). 2016:118-27. doi: 10.1109/3dv.2016.20. PubMed PMID: WOS:000391542200014.
[18] Ataiya, Ialhashim. starlab-mcfskel GitHub.com2015 [cited 2017]. Available from: https://github.com/ataiya/starlab-mcfskel.
[19] Blum H. ªA Transformation for Extracting New Descriptors of Shape, º Models for the Perception of Speech and Visual Form, W. Wathen-Dunn, ed. Cambridge, Mass.: MIT Press; 1967.
[20] Cornea ND, Silver D, Min P. Curve-skeleton properties, applications, and algorithms. IEEE Transactions on visualization and computer graphics. 2007;13(3):0530-548.
[21] Tagliasacchi A, Delame T, Spagnuolo M, Amenta N, Telea A, editors. 3D Skeletons: A State-of-the-Art Report. Computer Graphics Forum; 2016: Wiley Online Library.
[22] Tagliasacchi A, Alhashim I, Olson M, Zhang H, editors. Mean curvature skeletons. Computer Graphics Forum; 2012: Wiley Online Library.
[23] Low Poly Baby Groot thingiverse.com2017 [cited 2017]. Available from: https://www.thingiverse.com/thing:2155138.
[24] Gorilla thingiverse.com2016 [cited 2017]. Available from: https://www.thingiverse.com/thing:1957584/#files.
[25] Hand grabCAD.com2016 [cited 2017]. Available from: https://grabcad.com/library/hand-surfaces-to-solid-1.

[26] Lower Control Arm grabCAD.com2012 [cited 2017]. Available from: https://grabcad.com/library/lower-control-arm-dot-dot-dot.
[27] Attene M, Falcidieno B, Spagnuolo M. Hierarchical mesh segmentation based on fitting primitives. The Visual Computer. 2006;22(3):181-93.
[28] Lien J-M, Keyser J, Amato NM, editors. Simultaneous shape decomposition and skeletonization. Proceedings of the 2006 ACM symposium on Solid and physical modeling; 2006: ACM.
[29] Liu R, Zhang H, editors. Segmentation of 3D meshes through spectral clustering. Computer Graphics and Applications, 2004 PG 2004 Proceedings 12th Pacific Conference on; 2004: IEEE.
[30] Lee Y, Lee S, editors. Geometric snakes for triangular meshes. Computer Graphics Forum; 2002: Wiley Online Library.
[31] Carvalho DD, dos Santos TR, von Wangenheim A, editors. Measuring arterial diameters for surgery assistance, patient customized endovascular prosthesis design and post-surgery evaluation. Computer-Based Medical Systems, 2006 CBMS 2006 19th IEEE International Symposium on; 2006: IEEE.

Conflict of Interest: We do not have any conflicts of interest.



Section Recovery Methodology Overview.

Manuscript Highlights

- A methodology for automated extraction of cross-sectional shapes for branching structures is presented
- Methodology utilized skeletonization of a surface mesh as a reference for sampling planes
- Surface mesh is sliced to extract a 2D point cloud
- Filter algorithm for exclusion of peripheral slicing is presented
- Several case studies are examined to demonstrate the results of the implementation for the methodology giving the resulting cross sections