3D Reconstruction from Sparse Unorganized Cross Sections

Student Name: Nidhi Agarwal

IIIT-D-MTech-CS
July 20, 2015

Indraprastha Institute of Information Technology
New Delhi

Thesis Committee
Dr. Ojaswa Sharma (Advisor)
Dr. Saket Anand (External Reviewer)
Dr. Rajiv Raman (Internal Reviewer)

Submitted in partial fulfillment of the requirements
for the Degree of M.Tech. in Computer Science

Keywords: 3D reconstruction, sparse cross-sections, Hermite Mean-value interpolation

Certificate

This is to certify that the thesis titled **3D Reconstruction from Sparse Unorganized Cross Sections** submitted by **Nidhi Agarwal** for the partial fulfillment of the requirements for the degree of *Master of Technology* in *Computer Science & Engineering* is a record of the bonafide work carried out by her under my guidance and supervision at Indraprastha Institute of Information Technology, Delhi. This work has not been submitted anywhere else for the reward of any other degree.

Dr. Ojaswa Sharma
Indraprastha Institute of Information Technology, New Delhi

Abstract

In this thesis work, we propose an algorithm for smooth 3D object reconstruction from unorganized planar cross sections. We address the problem in its full generality, and show its effectiveness on sparse set of cutting planes. Our algorithm is based on construction of a globally consistent signed distance function over the cutting planes. It uses a divide-and-conquer approach utilizing Hermite mean-value interpolation for triangular meshes. This work improvises on recent approaches by providing a simplified construction that avoids need for post-processing to smoothen the reconstructed object boundary. We provide results of reconstruction and its comparison with other algorithms.

Acknowledgments

This work would not have been possible without support from a number of people. Foremost, I would like to extend my deepest gratitude to Dr. Ojaswa Sharma for his expert guidance and for the extremely productive brainstorming sessions I had with him. I am grateful to my friends, seniors and juniors who offered fresh perspectives on my research. I am also thankful to IIIT-Delhi for providing excellent infrastructure and support.

Last but never the least, I am immensely grateful to my parents, family members and close friends, for their invaluable support and unconditional love.

Nidhi Agarwal

New Delhi

Contents

List of Figures

List of Algorithms

List of Tables

Chapter 1

Introduction

The term *reconstruction* can be stated as to make an approximate surface representation of an object, given a finite set of its cross sections. Intersecting a three dimensional model with a cutting plane results into cross sections, as shown in Figure 1.1. Multiple arrangements for a given set of planes are possible, as illustrated in Figure 1.2.

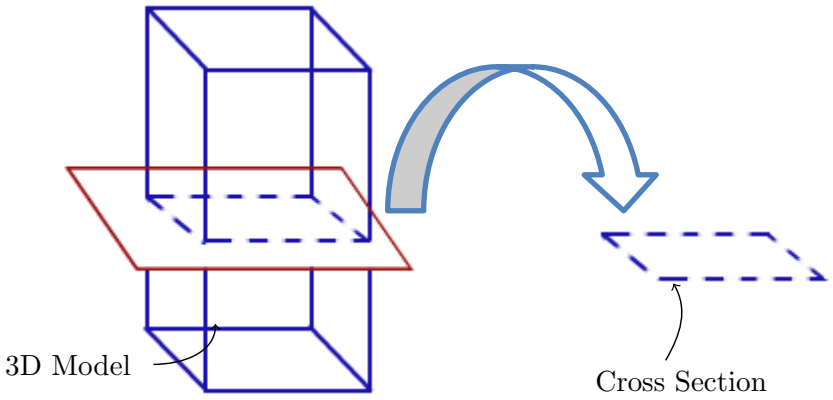
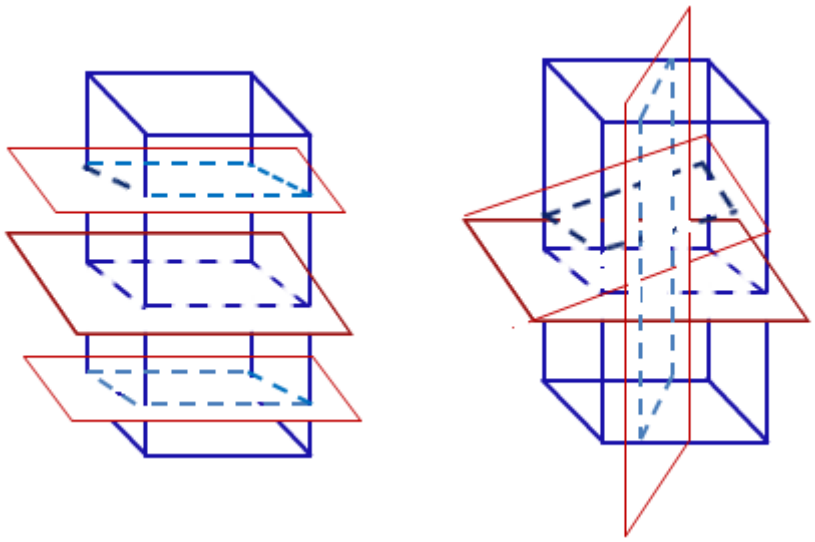


Figure 1.1: 3D model intersection with a plane (in Red color), resulting a planar cross section



(a) Parallel cutting planes (b) Unorganized cutting planes

Figure 1.2: Various configurations of Cross sections

1.1 Overview

Various technical and scientific problems involve dealing with solids and surfaces. In architecture and design, medical diagnosis, biomedical research, interaction with 3D models is very common and extensive. 3D surface need to be reconstructed after scanning the original object, resulting into a series of input cross sections for required reconstruction. In geometry processing also, reconstruction of a complete surface from given incomplete data is also an intensively studied problem.

Reconstructing the boundary of a solid 3D model from a finite set of planar cross sections has attracted much attention in the literature from past two decades. Various techniques are introduced in the past that solve the reconstruction problem differently. 3D object reconstruction from point cloud is a widely explored domain in computer graphics. Such problems appear, for example, in 3D freehand ultrasound imaging. A cross-section is formed by a set of closed contours

defining the boundary of the material of interest to be reconstructed.

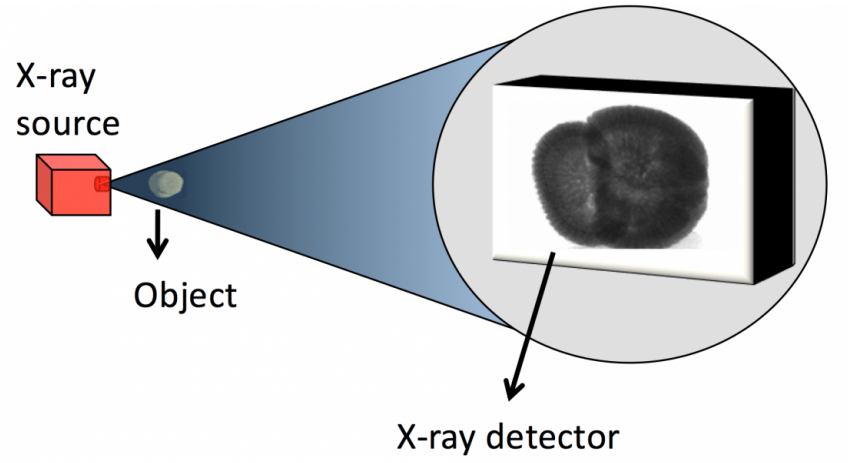
Most work on the subject deals with interpolation between parallel cross sections, that comprise of contours defining the geometry and topology of the intersections of the object with a series of (usually equally-spaced) parallel planes as shown in Figure 1.2(a). There exists a vast and significant literature on this problem as well. However, our work focuses on the more general and difficult case of reconstructing three dimensional surface of objects using arbitrarily arranged planar contours representing cross sections through the objects. This is a relatively less explored research problem.

1.2 Research Motivation

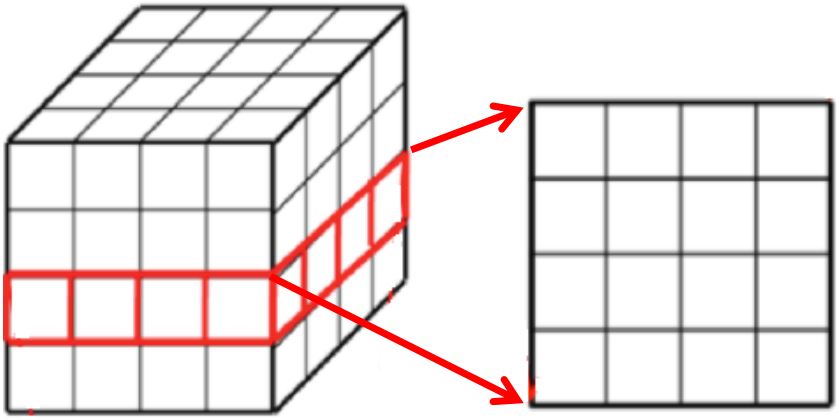
The need for surface reconstruction of an object, given unorganized set of cross sections is the result of advancement in various fields such as medical imaging technology, digitization of objects and geographical information systems. In medical field, data generated by different imaging techniques such as ultrasound, computed axial tomography (CAT), magnetic resonance imaging (MRI), computed tomography (CT) and nuclear magnetic resonance (NMR) provide a set of contours through the 3D object of interest, as shown in Figure 1.3.

These resulting cross sections that are not necessarily parallel, are the basis for interpolation of required boundary surface. The reconstructed organ can be viewed using various digital techniques which are very helpful for analyzing and representing three dimensional internal structures of human anatomy such as bones, tissues and tumors. It is considered to be an important diagnostic aid in the medical world.

Similarly in CAD (Computer Aided Design), geometry of an object is specified by a series of contours which can be used for interpolation of original object further. Another important application is geology, where terrain surfaces are usually created by interpolation of various



(a)



(b)

Figure 1.3: CT Scan

planar contours of terrain at different heights.

1.3 Aim

This work improvises on recent approaches by providing a simplified reconstruction algorithm. Following are the objectives of this work :

- to present a robust technique for continuous and smooth 3D object reconstruction from a given sparse set of unorganized cross sections. Here, the problem is addressed in its full generality, and is shown to be effective on sparse set of cutting planes.
- to propose a robust and simplified technique for reconstruction that does not require any post processing to smooth the reconstructed object boundary.

Chapter 2

Related Work

2.1 Overview of Previous Approaches

Sidlesky et al. [24] analyzed topological properties of solution to this reconstruction problem. The authors observe that a line not intersecting the object does not contribute to the reconstruction. Their algorithm enumerates all possible reconstructions that satisfy the interpolation and topological equivalence with the given input. Due to a large number of possible reconstructions, complexity of their algorithm is exponential in nature. There may be cases for which several reconstructions are topologically valid for a unique set of given cross sections.

Similar approaches based on the Voronoi diagram are suggested by Liu et al. [14], and Memari and Boissonnat [17]. Liu et al. construct medial axis of each partitioned cell (a convex polyhedron) and approximate the reconstruction by lifting the cross sections on the medial axis. Memari and Boissonnat, on the other hand, use the Voronoi diagram of the cross sections within each partitioned cell. The authors also provide rigorous proof of their reconstruction, where they propose a topological reconstruction method based on the Delaunay triangulation of the set of segments of intersecting lines. The authors claim an improvement over the method

by Liu et al. by producing reconstructions that are not topologically affected by lines that do not intersect with the object under consideration. Their reconstruction boundary, however, is a piecewise linear approximation of the boundary of the original object and lacks smoothness.

Some of the algorithms in literature attempted to solve the problem of reconstruction in much more generality. Boissonnat [6], presented the first reconstruction algorithm without any constraints. For surface reconstruction, Boissonnat generated delaunay triangulation for each segment. The author then projected the triangulations onto one another resulting in a set of tetrahedrons for maximizing the sum of their volumes.

The more general problem is to remove the constraint on arrangement of cutting planes, all the input cross sections are parallel to each other. Bogush et al. [5] presented his algorithm with non-parallel cross sections. After that Boissonnat and Memari [16], and Liu et al. [14] uses completely atbitrarily oriented cross sections. In their work, first the arrangement of cutting planes is calculated and then using medial axis, each cell on the arrangement is subdivided. Contours which are lying on the cell boundary are projected on this medial axis and the image portions of this projection are matched. Then the surface is reconstructed by connecting the matching portions of the original contour.

The methods of Liu et al. [14] are among the few that support multi-colored cross-sections. Bermano et al. [4] describe an online algorithm for reconstructing simultaneously multi-colored object from arbitrarily-oriented, possibly partial, cross-sections. An additional desired feature is partially defined cross-sections, that is, sections in which portions have the "unknown" label, resulting from areas scanned improperly or not scanned at all. Such a situation cannot be handled by all the reconstruction algorithms described so far since in this case the cells of the arrangement of the (portions of) planes are not convex any more. An algorithm by Barequet and Vaxman [3] combines all the above-mentioned features plus the ability to handle partially-defined sections.

Boissonnat and Memari [16] used the Delaunay triangulation for reconstruction. Input to the algorithm is a set of intersecting planes along with their intersections with the object. The authors consider a partitioning of space by all cutting planes in the space, and extract a closed triangular mesh within each partitioned cell. The mesh serves as an approximation of the object from its intersections with the boundary of cell. To complete the reconstruction, all the reconstructed segments within each cell are glued together.

The work of Sharma and Anton [19] suggests a different approach to reconstruction via continuous deformations. Generalizing on homotopy based reconstruction from organised cross sections, the authors perform reconstruction in an implicit setting by formulating homotopies in each partitioned cell. The authors define smooth functions (piecewise quadratic) along every cutting line. We note that this reconstruction algorithm suffers from two main problems:

- the piecewise quadratic functions associated with each cutting lines are a good choice locally, but are inconsistent globally. This choice results in multiple values of the function at points of intersection of two or more cutting lines.
- even though the resulting curve is very smooth (see proof of smoothness in [19]), geometrically the curve is not simple and consists of very high curvature near the cutting lines.

The algorithm, though topologically motivated, does not result in a geometrically fair boundary.

The solutions described above have assumed that all input slices are complete, there is no missing data within the cross sections and the slices are segmented correctly. In practical cases, there may be some uncertain regions with incomplete information that are not reliably segmented. The algorithms suggested in [3, 4], attempt to solve the partial-slice situation where data may be missing in portions of the sections. Barequet and Amir [3] reconstruct the original object by interpolating simultaneously all the cross sections. Their algorithm attempts to minimize the

surface area of the reconstruction by using an offset distance function in order to locally decide which contour features are to bind. Smoothing is performed as a post-processing operation to clean-up the resulting surface.

An efficient algorithm is presented by Fuchs et al. [9], that is based on an euler tour of a toroidal graph in order to obtain an optimal solution. Time complexity of this algorithm is $O(n^2 logn)$, where n is the total number of vertices on the contours bounding the triangles.

2.2 Contributions and Novelty

This work makes following novel contributions:

- This work solves the problem of 3D reconstruction with a sparse set of unorganized cross sections. The algorithm described here solves the problem in its most general setting. A divide and-conquer approach is followed to solve this problem.
- Main contribution is to formulate a signed distance based globally consistent function on the cutting planes.
- Another novelty is in mean-value Hermite interpolation in polygonal partitions by augmenting their boundary such that the function on the boundary is linear.
- The resulting reconstruction is continuous and smooth that results from a simple and robust algorithm. No post processing is required as in the case of most algorithms in literature.

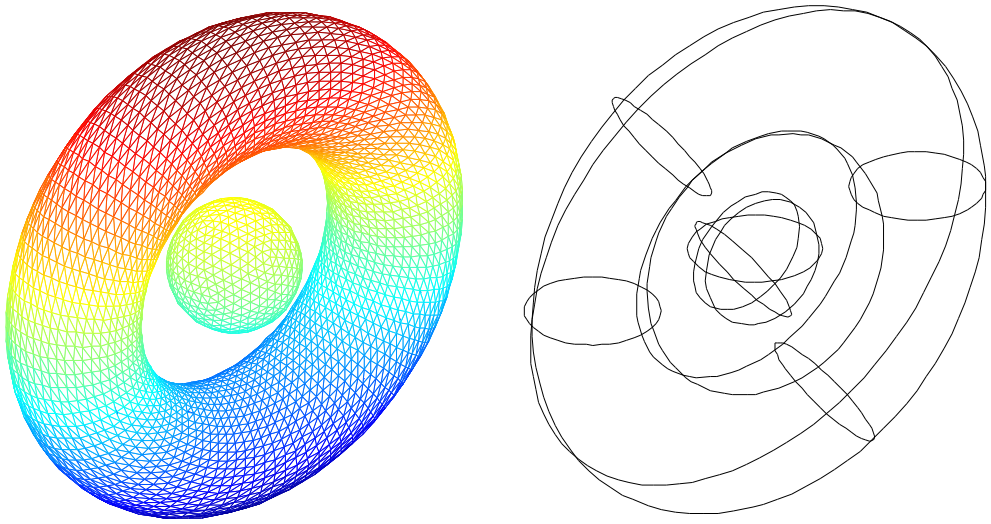
Chapter 3

Problem Description

3.1 Mathematical Formulation of Problem

The problem of object reconstruction from unorganized planar cross sections can be formally described as follows. Given a set of cutting planes $\Pi = \{\pi_i, i \in [0, n-1]\}$ in $\mathbb{R}^3$, and intersections $\mathcal{S} = \{\mathcal{O} \cap \pi_i, i \in [0, n-1]\}$ with a compact 3-manifold $\mathcal{O}$, compute a continuous and smooth reconstruction $\mathcal{R}$ similar to $\mathcal{O}$. The similarity of $\mathcal{R}$ to $\mathcal{O}$ implies that $\mathcal{R} \cap \Pi = \mathcal{S}$, and $\partial\mathcal{R}$ be a smooth surface.

In order to reconstruct the object, the only information available about the 3D model is, intersection of its surface with the cutting planes. Each of these cross sections obtained are assumed to be simple and closed curves as shown in Figure 3.1. A closed curve, obtained from one of the cutting planes is called a *contour*. Sequence of these contours is used for constructing a piecewise planar approximation to the surface of original object.



(a) (b)

Figure 3.1: (a) Torus (b) Cross sections obtained after intersecting Torus with 4 Cutting planes.

Intersections $\mathcal{S}$ on any plane π_i may have multiple disjoint components and the same on two different cutting planes can intersect with each other. Figure 3.2 shows cross sections superimposed on the object.

As mentioned in [17], the reconstruction $\mathcal{R}$ is a manifold with boundary similar to $\mathcal{O}$ satisfying the constraint that $\mathcal{S} = \mathcal{R} \cap \mathcal{L}$. It is also desirable that $\mathcal{R}$ be topologically similar to $\mathcal{O}$. In other words, the surface reconstructed from the input contours assures that its intersections with the given set of cutting planes should be identical to the contours lying on them.

The correctness of the reconstructions obtained is quantified using various volume and area based ratios such as, ratio of area of reconstructed object and the area of original object and the volume of reconstructed object and volume of original object. Another measure used is - Hausdroff distance between the two models which gives a good measure of the distance between them.

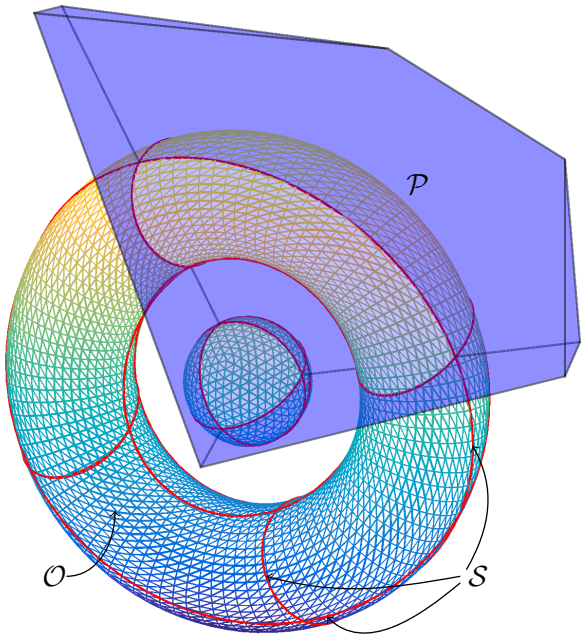


Figure 3.2: Planar cross sections of an object and polyhedron cell.

Chapter 4

Reconstruction Algorithm: Overview

Starting with a given set of cross sections $\mathcal{S}$, we restrict reconstruction to domain $\Omega \subseteq \mathbb{R}^3$ that encloses all cross sections. We address the shortcomings of [19] (discussed in section 2.1) by constructing a globally consistent function using the signed distance function on the cutting planes. The use of signed distance also makes the algorithm robust while keeping the resulting reconstruction simple.

We follow a divide and conquer approach to reconstruction. From the arrangement of cutting planes, we perform a partitioning of Ω into a set of convex polyhedrons $\mathcal{H}$. A distance function can be defined at any point in Ω considering the cross section boundaries as generators. This distance function serves as a basis to construct a globally consistent signed distance function (SDF) for any point on the cutting planes. This is possible since every cross section contains information about inside and outside regions within the respective cutting plane. Constrained Delaunay Triangulation (CDT) is computed on every face of polyhedrons in $\mathcal{H}$. The SDF is evaluated at every vertex of the triangulations in $\mathcal{H}$; by construction, the SDF evaluates to zero at vertices of the cross-section edges. The reconstruction algorithm then interpolates the signed distance value inside each polyhedron in a Hermite fashion to ensure C^1 continuity across

partitions. A reconstruction surface is then recovered as the zero level set of the computed field. Our reconstruction algorithm is summarised in Algorithm 1.

Input: Intersections $\mathcal{S}$ on Π
Output: $\mathcal{R}$
Compute polyhedral partitioning $\mathcal{H}$ of Ω from Π
foreach polyhedron $\mathcal{P} \in \mathcal{H}$ **do**
  foreach polygonal face p of $\mathcal{P}$ **do**
    $\mathcal{S}' = \text{Clip}(\mathcal{S}, p)$
    Compute CDT (add constraint edges from $\mathcal{S}'$)
    Evaluate SDF at each vertex of CDT
  end
  foreach point $\mathbf{x} \in \mathcal{P}$ **do**
    Compute mean-value coordinates λ of $\mathbf{x}$
    Compute $\mathcal{F}(\mathbf{x})$ using Hermite interpolation
  end
end
$\mathcal{R} \leftarrow \ker \mathcal{F}$

Algorithm 1: The reconstruction algorithm.

4.1 Domain partitioning and boundary triangulation

We consider a bounding box around the domain enclosing the set of cross sections. The arrangement of cutting planes naturally partition the bounding box into a set $\mathcal{H}$ of convex polyhedrons. For surface reconstruction within each polyhedron, we perform a Constrained Delaunay Triangulation of each face. The constraint edges belong to the boundary curves of cross sections of the respective cutting plane clipped by the face polygon. By definition, these constraint edges sample the actual object surface.

4.2 Globally consistent function over polyhedron faces

The reconstruction algorithms suggested in [4, 19] work in an implicit setting. An implicit field is constructed by propagating function values from boundaries of polyhedra that partition the

space. The surface of interest in such a setting is usually defined as the zero level set of the implicit function. Since the cross sections provide inside and outside information, Bermano et al. in [4] used the characteristic function to define an implicit function on any cutting plane π_i with cross-section $s_i \in \mathcal{S}$ as

$$f_\chi(\mathbf{x}) = \begin{cases} 1, & \mathbf{x} \in s_i / \ \partial s_i \\ 0, & \mathbf{x} \in \partial s_i \\ -1, & \mathbf{x} \notin s_i. \end{cases} \tag{4.1}$$

According to this definition, $f_\chi(\mathbf{x})$ is discontinuous at the cross section boundaries; which will result in discontinuous implicit field at polyhedron faces. The resulting reconstruction thus suffers from ripples on the surface. To alleviate this problem, the authors suggest performing bi-Laplacian smoothing as post-processing. Sharma and Anton [19] define smooth C^1 functions along linear cross sections (in a 2D setting) using piecewise quadratic polynomials with zeros chosen to be the points in ∂s_i (here π_i is a cutting line). Even though these functions are continuous and smooth along their respective cutting lines, problems arise when two or more cutting lines intersect and the point of intersection has distinct function values from different quadratic functions defined on these cutting lines. This can lead to a discontinuous implicit field that is undefined at the points of intersections of all cross sections. Such a problem would also arise in a 3D setting when a suitable quadratic function is defined locally for a cutting plane.

In this work, we propose to use a globally consistent and continuous function defined over the arrangement of cutting planes. We build such a function using the distance field of cross section boundaries $\partial\mathcal{S}$. This function can then be evaluated at points on polyhedron faces.

4.3 Hermite interpolation using Mean-value Coordinates

In order to reconstruct $\mathcal{O}$ inside a polyhedron, we interpolate distance values inside polyhedron partitions. Mean-value Coordinates (MVCs) [7] provide a simple and robust transfinite interpolation of a function defined on the boundary of a planar domain. At any point $\mathbf{x}$ inside an open, bounded and convex region Ψ, the mean-value boundary integral has a solution

$$g(\mathbf{x}) = \int_0^{2\pi} \frac{f(\mathbf{p}(\mathbf{x},\theta))}{\rho(\mathbf{x},\theta)} \mathrm{d}\theta \Big/ \int_0^{2\pi} \frac{1}{\rho(\mathbf{x},\theta)} \mathrm{d}\theta \ ,$$

where f is a function defined on $\partial\Psi$, $\mathbf{p}(\mathbf{x},\theta)$ is the point of intersection of $\partial\Psi$ with a semi-finite line starting $\mathbf{x}$ and at an angle θ with the x-axis, and $\rho(\mathbf{x},\theta)$ is the distance $||\mathbf{p}(\mathbf{x},\theta) - \mathbf{x}||$. In a similar fashion, MVCs can be applied to non-convex domains. For the special case when Ψ is a triangular mesh, Ju et al. [11] generalize MVCs that are continuous everywhere and smooth inside the mesh. In our case, polyhedron faces are triangulated to obtain a mesh. In essence, we discretise the signed distance function $f_{\mathcal{S}}$ at polyhedron boundary for mean-value interpolation.

We observe that the function $f_{\mathcal{S}}$ interpolated with MVCs results in a smooth surface everywhere in Ω except at the cutting planes (i.e., at the boundary of any two polyhedrons, as can be seen in Figure 4.1). This is attributed to the fact that the gradient of the MVC's at points on a common face between any two polyhedrons is not same (since the polyhedrons may have arbitrary convex shapes).

Given normals of $\partial\mathcal{O}$ at points in $\partial\mathcal{S}$, a Hermite interpolation will ensure that the gradient of $\mathcal{R}$ is consistent at polyhedron boundaries. Dyken and Floater in [7] suggest Hermite mean-value interpolation for a region with parametric boundary. We use the higher order barycentric coordinates of Langer and Seidel [12], where any barycentric coordinate can be lifted to a higher order and used for Hermite interpolation. In particular, using the first two terms of the Taylor

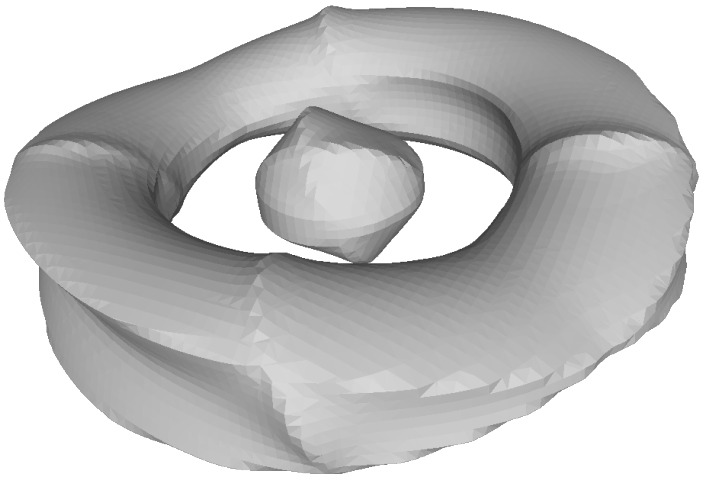


Figure 4.1: Reconstruction with MVCs without Hermite interpolation.

series, the interpolation function becomes

$$\hat{f}(\mathbf{x}) = \sum_{i=0}^{k-1} \hat{\lambda}_i (f_i + \nabla f_i \cdot (\mathbf{x} - \mathbf{v}_i)), \tag{4.2}$$

where f_i, and ∇f_i are function values, and function gradients at the vertices respectively, and $\hat{\lambda}_i = g \circ \lambda_i$ are the higher order barycentric coordinates with $g : \mathbb{R} \mapsto \mathbb{R}$ being a smooth piecewise polynomial function as suggested in [12]. In many situations, normals at the cross sections are not available. In absence of any prior information about the normals, we suggest computing an in-plane approximation of the normals using finite differencing at points of cross section curves in a particular cutting plane. Figure 4.2 shows a reconstruction with Hermite interpolation.

While, the reconstructed object has a consistent overall topology and a smooth surface, the approximated normals may not be accurate and could result in undesired artefacts (see Figure 4.2). A better estimate of the normals can be used in such situations to alleviate this problem.

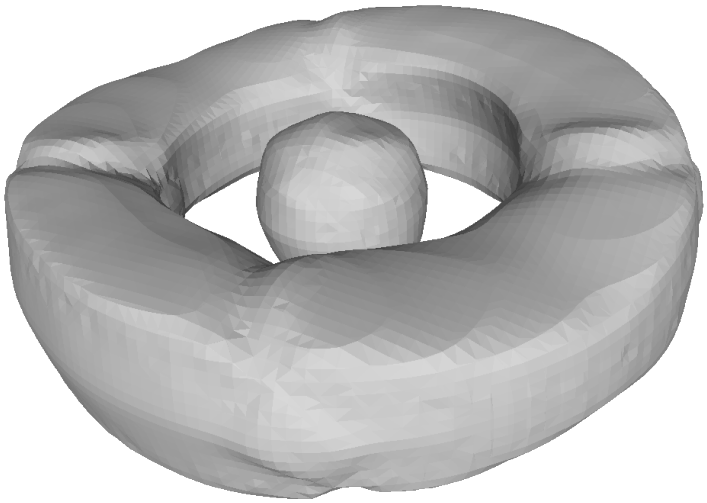


Figure 4.2: Reconstruction with higher order MVCs with Hermite interpolation.

Chapter 5

Implementation Details of Algorithm

Basic overview of the algorithm that is followed for reconstruction is given in previous chapter (Chapter 4). In this chapter, the complete algorithm is described along with its implementation details. Steps of the full algorithm are shown in Figure 5.1.

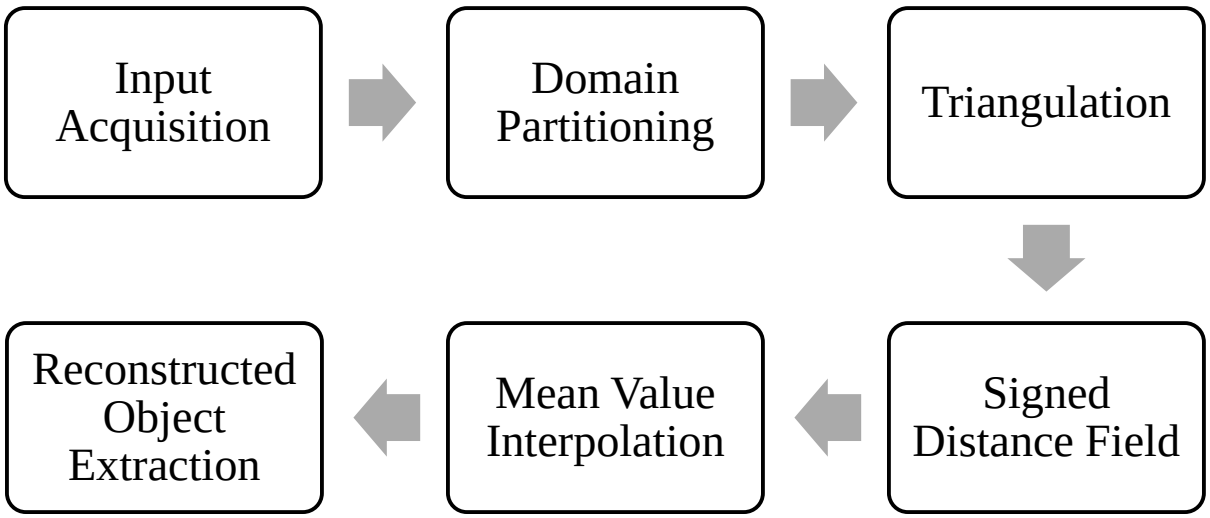


Figure 5.1: Steps of the Full Algorithm

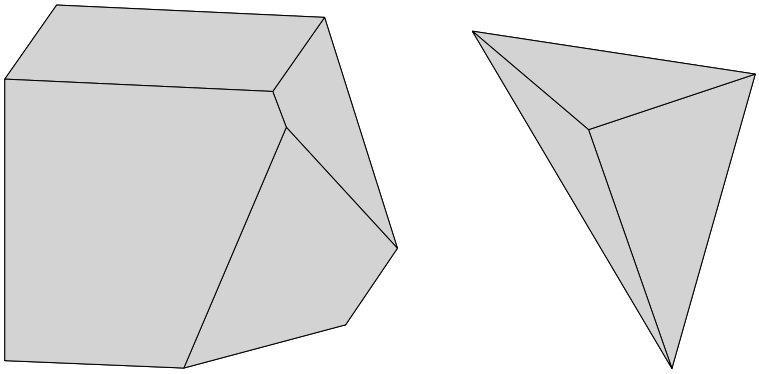


Figure 5.2: Polytopes obtained after partitioning.

5.1 Domain Partitioning

Set of cross sections are given as input to the algorithm. For the domain partitioning, a bounding box (Ω) is considered such that all the given cross sections are inside the bounding box. The cutting planes (Π) intersect the bounding box (Ω), resulting in partitioning it into a set of convex polytopes, $\mathcal{H}$ (shown in Figure 5.2). The algorithm followed for partitioning is summarized in Algorithm 2.

Input: Ω and Π
Output: Set of polytopes, $\mathcal{H}$
foreach cutting plane Π **do**
  foreach Bounding box partition Ω_i of Ω **do**
    Ω_1 = Clip(Ω_i, p)
    if Ω_1 is null then discard Ω_1
    else Remove duplicate nodes and faces from Ω_1

    Ω_2 = Clip(Ω_i, p') (p' is p with normals reversed)
    if Ω_2 is null then discard Ω_2
    else Remove duplicate nodes and faces from Ω_2
  end
end

Algorithm 2: Domain Partitioning algorithm.

5.2 Adding Cross Sections

The next step for reconstruction is to add the cross sections of the original object $\mathcal{O}$ on each face of the set of polytopes $\mathcal{H}$ obtained in previous step. The main motivation behind adding these cross sections is to gather the topological and geometrical details of the original object. These cross sections boundaries are the generators for the reconstruction and contain information about inside and outside regions within the respective cutting planes. The algorithm for cross sections addition is described in Algorithm 3.

Input: Set of polytopes $\mathcal{H}$ and set of Cross sections $\mathcal{S}$ on Π
Output: Cross sections on each face of polytope in $\mathcal{H}$
foreach polyhedron $\mathcal{P} \in \mathcal{H}$ **do**
  foreach polygonal face p of $\mathcal{P}$ **do**
    if p is generated by any cutting plane Π_i
    s = clip($\mathcal{S}$, p) using Sutherland Hodgman algorithm
    add s to polytope face
  end
end

Algorithm 3: Algorithm for adding cross sections on polytope face.

For adding cross sections, first we need to identify if the face is generated by any cutting plane or is part of bounding box Ω. After that cross sections are clipped by face of polytope. Then that clipped segment of cross section is added to that face.

5.3 CDT

Constrained Delaunay Triangulation or CDT is actually a generalization of Delaunay triangulation with certain pre-defined edges. Delaunay Triangulation is triangulation with a special condition of having empty circumcircle such that no point is inside the circumcircle of any triangle. An example is shown in Figure 5.3.

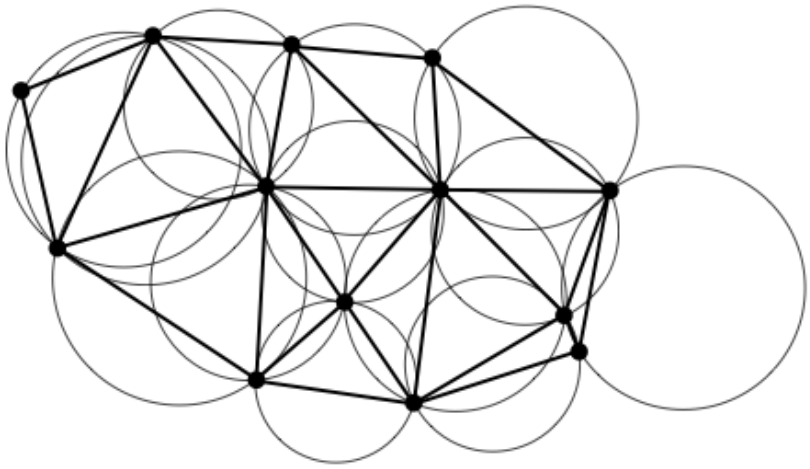


Figure 5.3: Delaunay Triangulation : All triangles have empty circumcircle.
Source: [23]

Given a set of n vertices in a plane along with some straight line edges, a *Constrained Delaunay Triangulation (CDT)* is triangulation of the given vertices that includes the specified edges as part of triangulation. That is, the given edges are preserved and not split into smaller edges further by avoiding the insertion of steiner points [**?**]. An example of a CDT is shown in Figure 5.4. The pre-defined boundary points are not guaranteed to satisfy the criterion for being Delaunay triangulation. So, a CDT may not always be a Delaunay triangulation, illustrated in Figure 5.5.

As mentioned in the section 4.1, a Constrained Delaunay Triangulation is performed on each face of polytope. For triangulation we used a two-dimensional quality mesh generator and Delaunay triangulator, Triangle [20]. For triangulation, we provide face edges of a polytope along with the cross sections (that are obtained during sampling of the original 3D model) on that face, if present as constrained edges. Triangle library then returns the dense triangulation of the face.

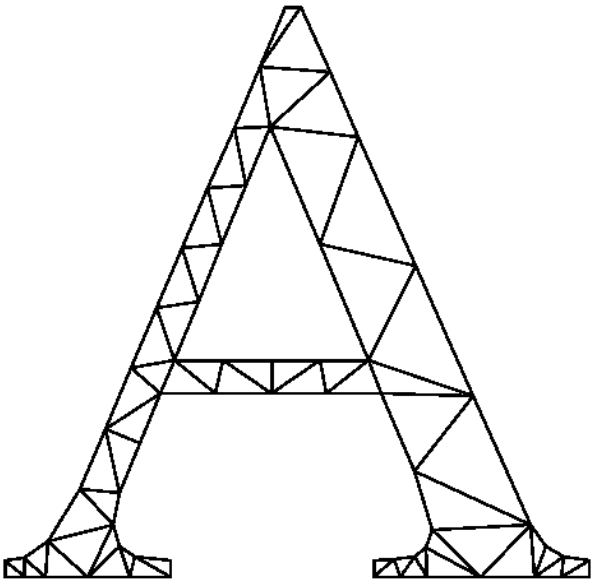
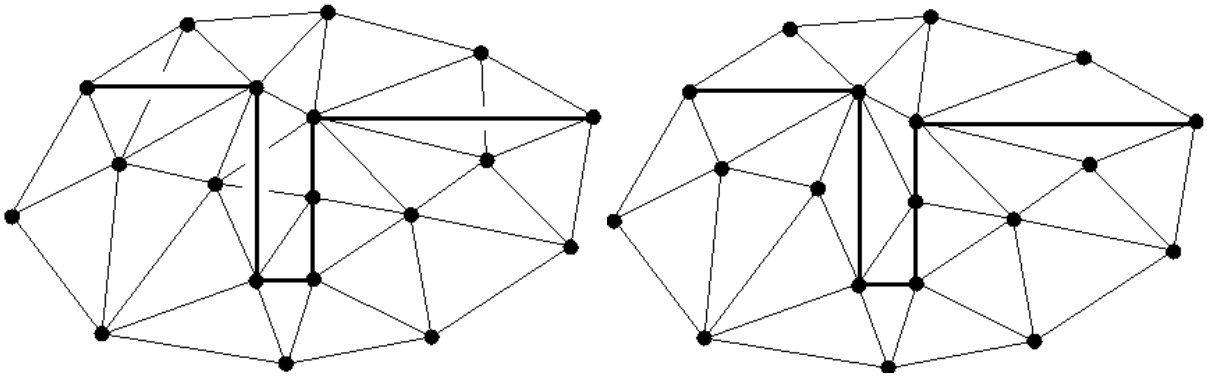


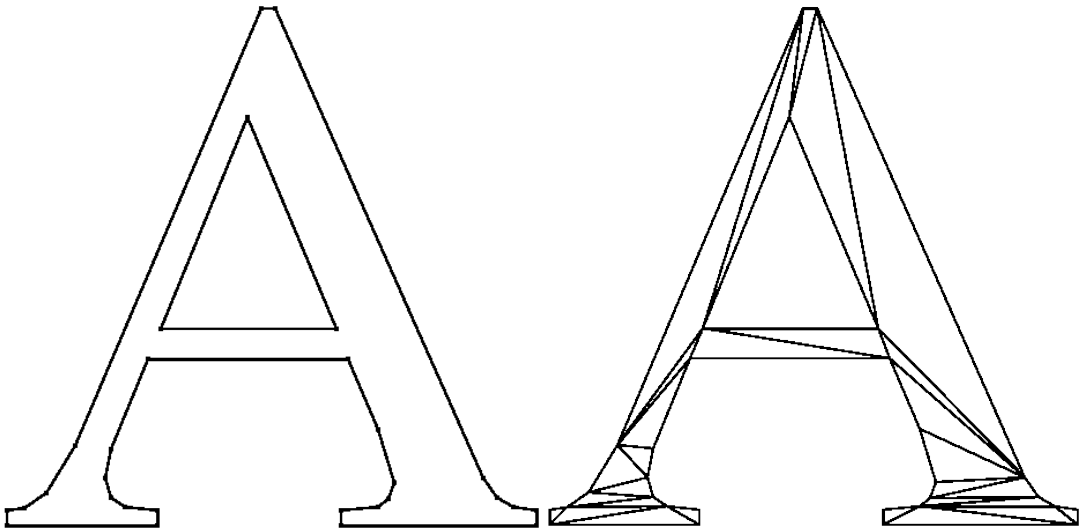
Figure 5.4: Constrained Delaunay Triangulation(CDT).
Source: [21]



(a) (b)

Figure 5.5: (a) Delaunay Triagulation (b) Constrained Delaunay triangulation conforming with given boundaries.

Source: [8]



(a) (b)

Figure 5.6: (a) A Mesh (b) Triangulation with steiner points added on boundary of the given mesh. Source: [21]

5.4 Signed Distance Field (SDT)

Globally consistent signed distance function, for each point on the cross section boundaries is calculated. These SDT values are then used for interpolation inside polyhedrons. The algorithm followed for SDT calculation is explained in Algorithm 4.

5.4.1 Computing signed distance function

The set of points in $\partial\mathcal{S}$ refer to boundaries of the cross sections, i.e., points where the boundary $\partial\mathcal{O}$ of the original object intersects with the set of cutting planes Π. The distance function calculates the distance from any point $\mathbf{x}$ to the closest point on the set $\partial\mathcal{S}$.

Input: $\partial\mathcal{S}$, boundaries of the cross sections
Output: $\mathcal{R}$
Compute polyhedral partitioning $\mathcal{H}$ of Ω from Π
foreach polyhedron $\mathcal{P} \in \mathcal{H}$ **do**
  foreach polygonal face X of $\mathcal{P}$ **do**
    foreach triangulated node x of X **do**
      dist(x) = $\inf_{\mathbf{p}\in\partial\mathcal{S}}(||\mathbf{p}-\mathbf{x}||)$
      if x is inside the cross section on X
        sdt = dist
      else
        sdt = -dist
    end
  end
end

Algorithm 4: The SDT calculation algorithm.

$$\text{dist}_{\partial\mathcal{S}}(\mathbf{x}) = \inf_{\mathbf{p}\in\partial\mathcal{S}}(||\mathbf{p}-\mathbf{x}||).$$

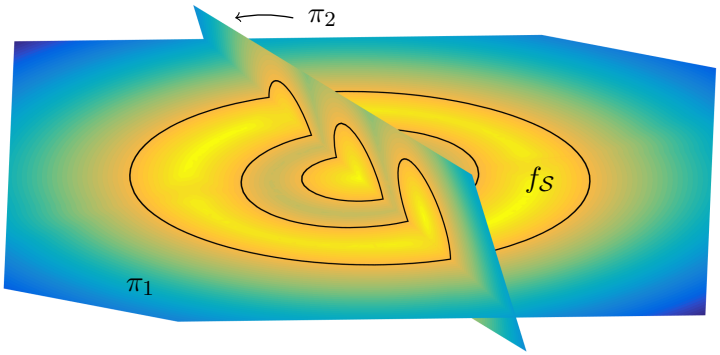


Figure 5.7: Globally consistent signed distance field over cutting planes.

A global distance field $\text{dist}_{\partial\mathcal{S}}$ can be computed at all the planes in Π. At any plane π_i, $\text{dist}_{\partial\mathcal{S}}$ can be converted into a signed distance field $f_{\mathcal{S}}$ using the inside and outside information from s_i.

$$f_{\mathcal{S}}(\mathbf{x}) = \text{dist}_{\partial\mathcal{S}}(\mathbf{x}) f_{\chi}(\mathbf{x}). \tag{5.1}$$

This definition of $f_{\mathcal{S}}$ is consistent over the domain and is continuous everywhere since the signed distance function will agree at the intersection points of $\mathcal{S}$ (see Figure 5.7). We evaluate $f_{\mathcal{S}}$ at mesh vertices of polyhedron faces for interpolation.

5.5 Mean Value Interpolation

Up to this point, constructed polytopes with each face triangulated and each triangulated node with an SDT value. In order to reconstruct object, we need to interpolate these distance values inside the polytopes. We used Mean value coordinates for interpolating functions defined over the boundary to inside of polyhedron as explained in [11]. As for closed convex polygons, mean value coordinates is an excellent method for constructing such an interpolant. A generalized form of mean value coordinates from polytopes to closed triangular meshes is used here. MVCs are continuous everywhere inside a given triangular mesh.

5.6 Reconstructed Object Extraction

For the extraction of zero level set, Marching Cubes algorithm [15] is used. This algorithm is used for computing a triangulated mesh of an iso-surface within the 3D matrix of scalar values $\mathcal{C}$ (called voxels) at iso-surface value. Main applications of this algorithm is in medical field, in medical visualizations such as MRI and CT scan images.

The Marching cubes algorithm constructs the surface in following steps:

- *Create Cells*:

  The algorithm considers a cube for locating the surface. The cube is created from 8 pixel values, four each from two adjacent slices.

- *Classify each vertex*:

  In order to determine intersection of surface and the cube, algorithm assigns 1 to vertex of the cube if it lies inside the surface and 0 if lies outside the surface. In this manner, topology of the surface within a cube can be determined by finding the location of intersection. The possible combinations in which a surface can intersect the cube are 256 (8 vertices in each cube, $2^8 = 256$). Out of these 256 possible cases only 14 are unique which reduces the complexity of triangulation, shown in Figure 5.8 [1]. A look-up table is created for surface-edge intersections corresponding to vertex labels.

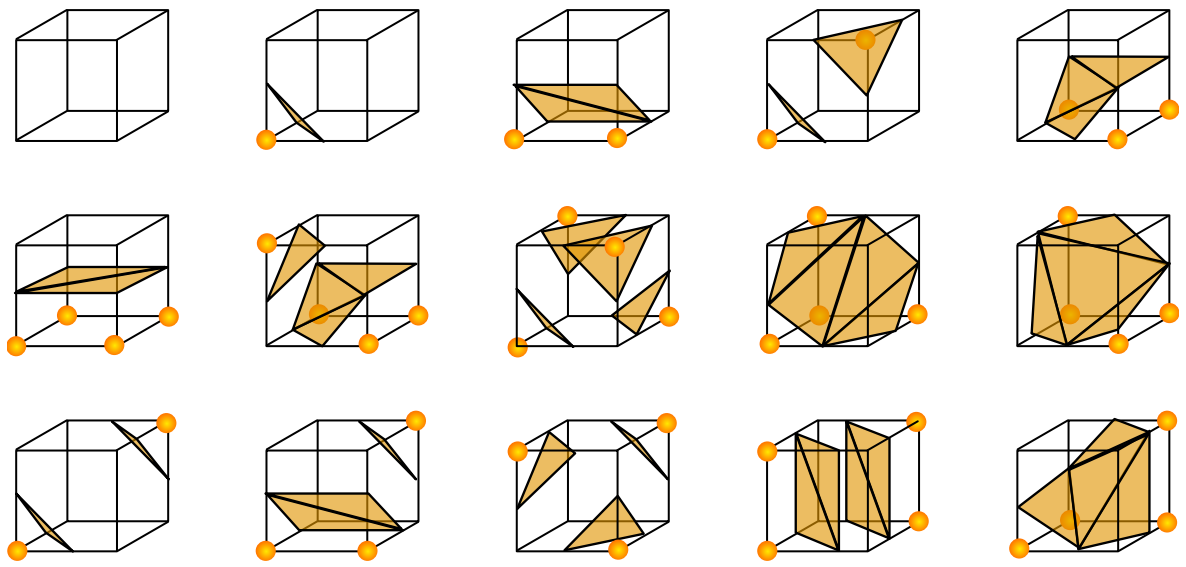


Figure 5.8: Unique Cube Configurations
Source: [1]

- *Build an index*:

  From the binary labeling of each vertex, an 8 bit index is created for each case, which

contains one bit for each vertex of a cube.

- *Get edge list*:

  An edge list for for a given cube configuration can be obtained using the 8 bit index

- *Interpolate triangle vertices*:

  The normals are calculated for each cube vertex using gradient vector. For estimating gradient vectors at the surface of interest, these are first estimated at the cube vertices and then interpolated at the intersections. The vectors are calculated using central differences along the three coordinate axes. Then these normal values are interpolated at triangle vertices. The gradients are divided by their lengths for generating unit normals which are required for rendering. Then the algorithm linearly interplates these normals at the point of intersection.

Chapter 6

Results and comparison

We show that our algorithm is capable of producing smooth and convincing results with very few cross sections. This is in contrast to other algorithms that use hundreds of cross sections to achieve a similar level of accuracy. With an increase in number of cutting planes that sample an object at regions of high curvature, reconstruction accuracy is expected to increase. Furthermore, none of our meshes presented here are post-processed in any manner or smoothed. While such a post-processing does improve the overall mesh appearance, it becomes difficult to preserve the intersection constraint during the smoothing operation. We believe that such a post-processing must be exercised with caution.

6.1 2D Results

Our focus is on reconstructing 3D meshes, but we applied our algorithm on 2D objects as well. There are many algorithms cited in literature for 2D object reconstruction.

We produce results of our reconstruction algorithm on 2D shapes intersected by a set of unorganized cutting lines, and compare these with the reconstructions of Sharma and Anton [19] and

Bermano et al. [4]. In order to produce results for the method in [4], we use the characteristic function defined on the boundary with the mean-value interpolation. In all our examples, we use a sparse set of cutting lines for reconstruction. With an increase in the number of cutting lines that sample an object at regions of high curvature, reconstruction accuracy is expected to increase.

In order to compare accuracy of our algorithm with other methods, we use area and distance based measures discussed in [19]. Here, the basis of comparison of accuracy is the original 2D model. The absolute difference in areas of the model $\mathcal{A}_m$ and that of the reconstruction $\mathcal{A}_r$ signifies change in areas between the two shapes. The ratio $\Delta\hat{\mathcal{A}}$ of the difference in areas with respect to the area of original model can be written as

$$\Delta\hat{\mathcal{A}} = \frac{(\mathcal{A}_m \cup \mathcal{A}_r) - (\mathcal{A}_m \cap \mathcal{A}_r)}{\mathcal{A}_m}.$$

A lower value of $\Delta\hat{\mathcal{A}}$ indicates a closely matching reconstructed shape. A similar length based measure can be formulated, but we find that this measure could be misleading for reconstructions with sparse cross sections (see Figure 6.1 for an intuition on length of reconstructions compared with that of the original shape). We instead use the Hausdorff measure [2,18] to see how closely the points on the boundary of the two shapes match. The Housdorff distance between two curves L and L' is

$$d_H(L, L') = \sup_{x \in L} \inf_{x' \in L'} d(x, x'),$$

where $d(\cdot, \cdot)$ is a distance metric. In our comparison, we use the ratio $\hat{d}_H$ of the Hausdorff distance (with Euclidean distance metric) and the length of the bounding box diagonal. A lower value of $\hat{d}_H$ indicates better reconstructed curve.

The first example in our results is that of a triangle shape (see Figure 6.1). Reconstructions from Sharma and Anton, and from Bermano et al. show that the curve has a tendency of

being orthogonal to the cutting lines, which degrades the overall quality of reconstruction. This observation is also mentioned in [4], where the authors perform additional smoothing operation to get rid of the staircase effect. In contrast, our reconstruction is smooth and follows the shape closely. The two measures also point to a more accurate reconstruction by our algorithm (see Table 6.1).

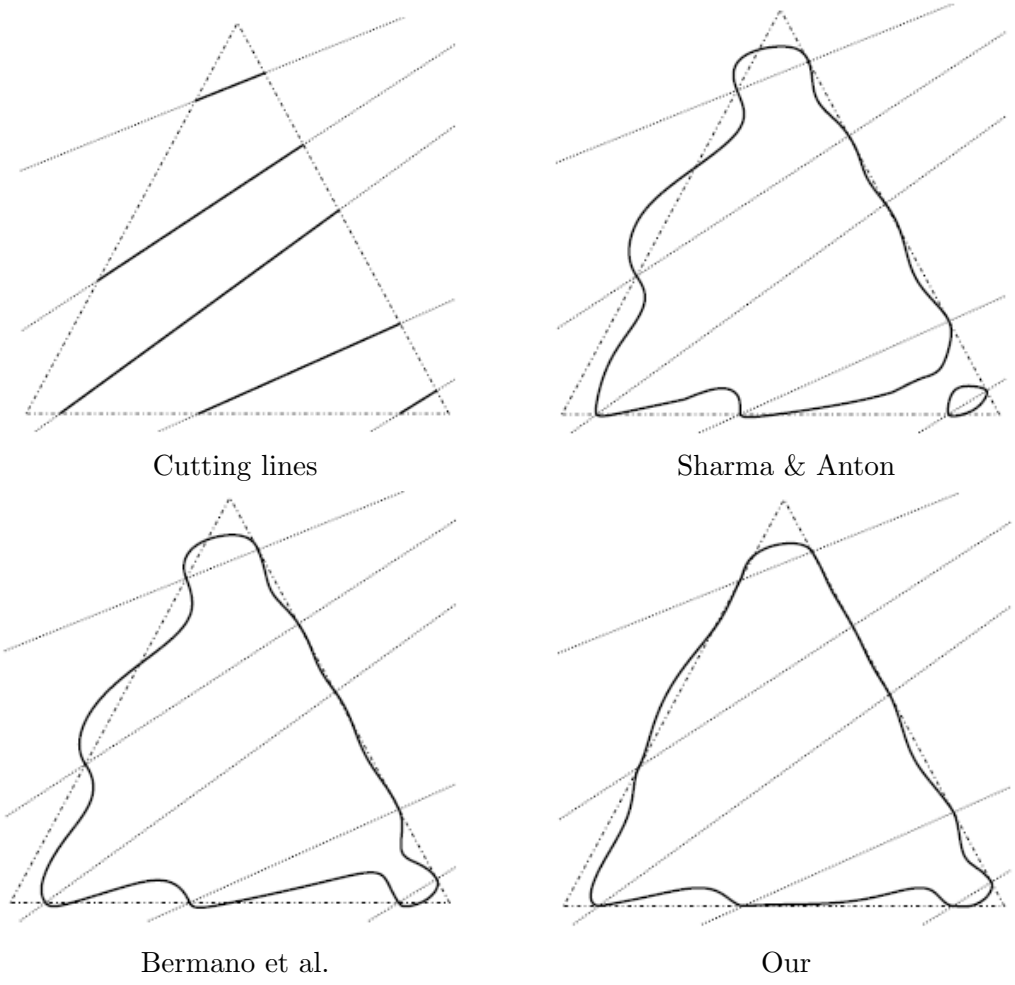
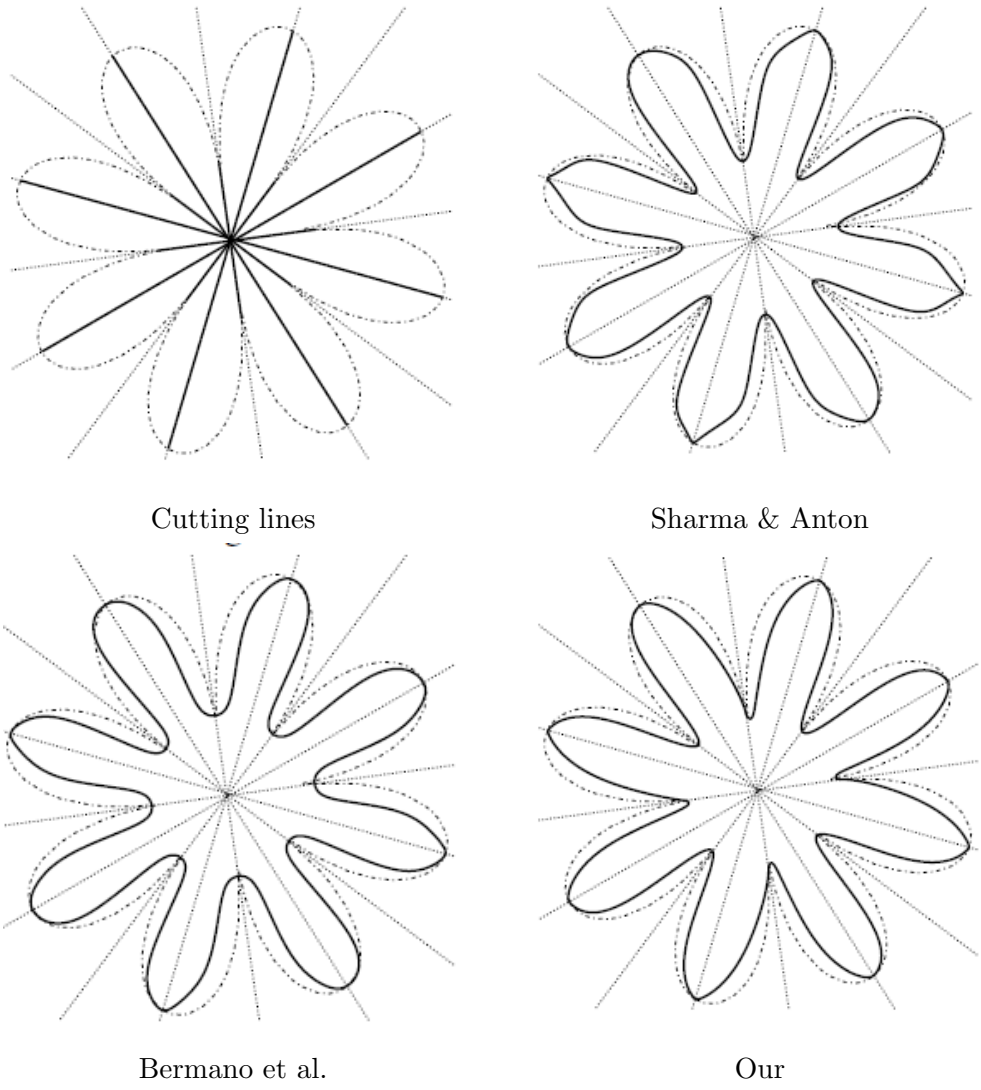


Figure 6.1: Reconstruction results with triangle shape

Method	$\Delta\hat{\mathcal{A}}$ (%)	$\hat{d}_H$ (%)
Sharma & Anton	16.8636	21.0045
Bermano et al.	14.5426	20.8096
Our	9.3230	20.7580

Table 6.1: Accuracy comparison with triangle shape



Cutting lines Sharma & Anton

Bermano et al. Our

Figure 6.2: Reconstruction results with flower shape

In our next example, we consider a symmetric flower object with eight equally spaced radial cutting lines. Figure 6.2 shows various reconstructions. It can be noted that the object is in fact orthogonal to the cutting lines near the petals. This results in similar reconstructions using the characteristic function and our distance function $f_{\mathcal{S}}$. However, at the singular points in the shape the former still enforces the reconstructed curve to be orthogonal to the cutting lines, while our reconstruction tries to match the original shape. Reconstruction by Sharma and Anton, on the other hand, shows sharp corners at the flat regions near the petals, despite showing better accuracy measures in Table 6.2.

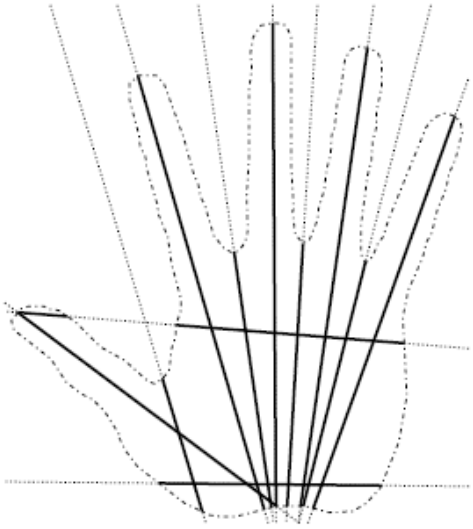
Method	$\Delta\hat{\mathcal{A}}$ (%)	$\hat{d}_H$ (%)
Sharma & Anton	21.3419	1.7205
Bermano et al.	29.6662	2.1745
Our	25.3255	1.9519

Table 6.2: Accuracy comparison with flower shape

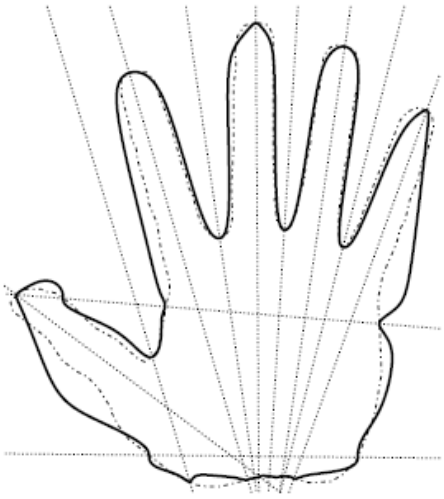
Our next set of examples are reconstructions of complex shapes of a hand and a rabbit. We sample main features in these objects with 10 and 11 cutting lines for the hand and the rabbit shapes respectively. For both shapes (and particularly more pronounced in the rabbit example), our results in Figures 6.3 and 6.4 clearly show a superior reconstruction with very little error (see Tables 6.8 and 6.4).

Method	$\Delta\hat{\mathcal{A}}$ (%)	$\hat{d}_H$ (%)
Sharma & Anton	15.2869	1.8608
Bermano et al.	12.3499	1.7004
Our	12.5666	1.7202

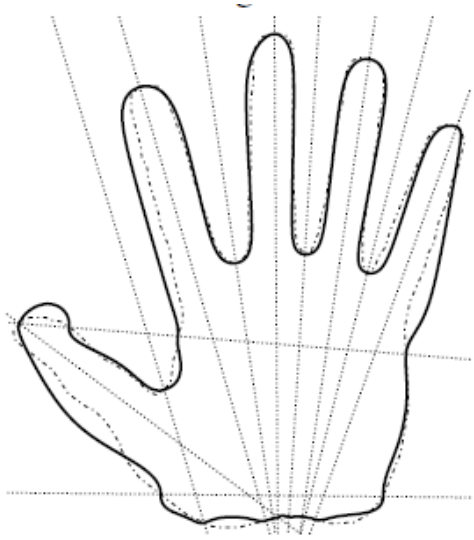
Table 6.3: Accuracy comparison with hand shape



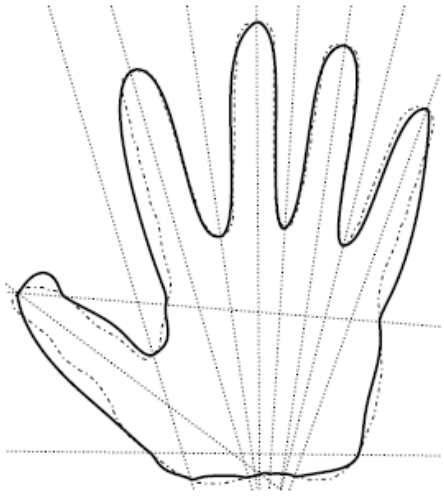
Cutting lines



Sharma & Anton



Bermano et al.



Our

Figure 6.3: Reconstruction results with hand shape

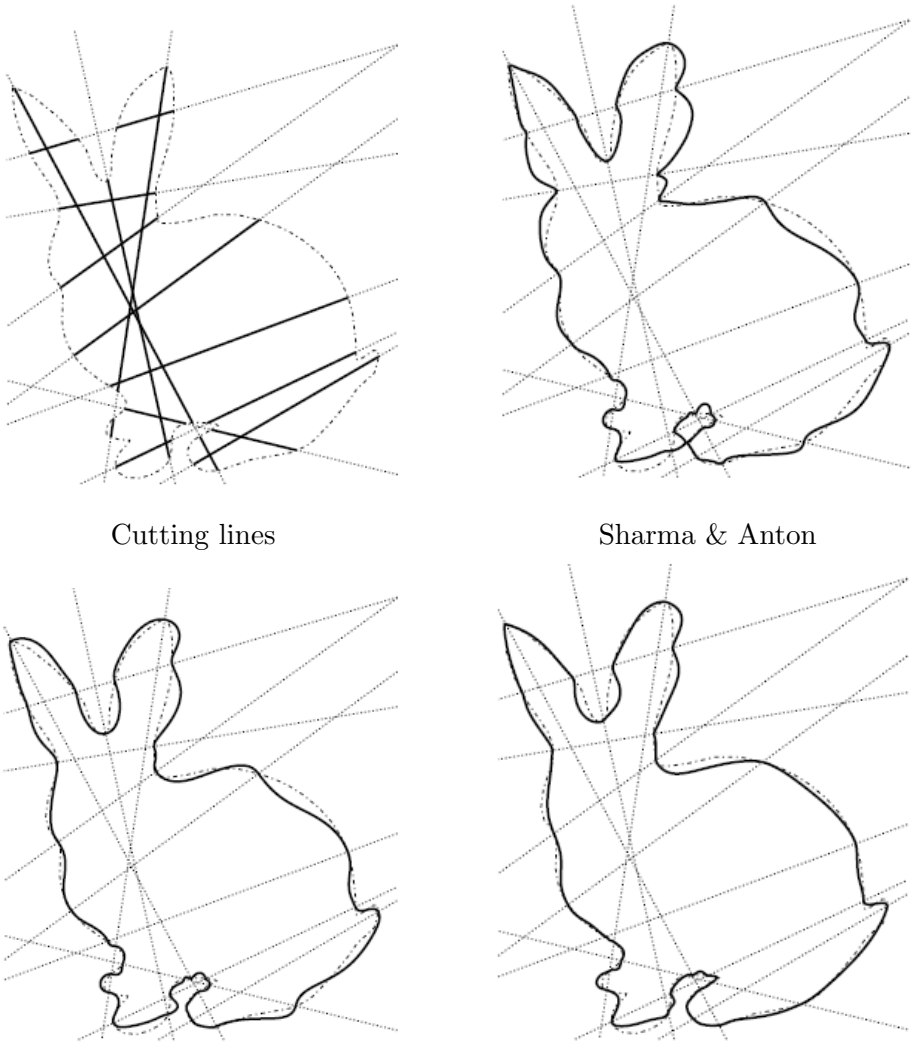


Figure 6.4: Reconstruction results with rabbit shape

Method	$\Delta\hat{\mathcal{A}}$ (%)	$\hat{d}_H$ (%)
Sharma & Anton	12.7368	1.6001
Bermano et al.	8.3649	1.2573
Our	5.8062	1.1229

Table 6.4: Accuracy comparison with rabbit shape

6.2 3D Results

We compare results from our method with those from algorithms similar to [19] and [4]. For comparisons, we extended the homotopy based 2D algorithm described in [19] to shapes in 3D. The original 2D algorithm defines a piecewise quadratic function along any cutting line that is derived from the zeros along the cutting line. We realise that a 3D analog of the same is difficult to formulate for an arbitrary planar cross section consisting of a set of closed contours. Therefore, we defined a continuous function on any cutting plane to be the signed distance function generated from the respective cross section curves. Note that this function is different from our globally consistent function $f_{\mathcal{S}}$ and exhibits a local behaviour similar to that indicated in section 4.2. Reconstruction algorithm discussed in [4] is based on the characteristic function f_{χ} defined over the cutting planes. We evaluate the use of characteristic function for reconstruction and compare the results with our method.

The following results are generated on synthetic cross sections obtained by slicing a 3D mesh with a set of cutting planes. We use the Geom3D package [13] to partition the volume into a set of polyhedrons. The faces of the polyhedrons are triangulated using the Triangle library [22]. The 3D meshes used here are obtained from INRIA GAMMA 3D mesh research database [10], and Large geometric models archive [25].

In order to compare performance of our algorithm with that of other methods, we compute three accuracy measures. Along with simple ratios of volume and surface areas, we use Housdorff distance based measure as discussed in [19]. The original 3D meshes are used as ground truth for evaluation. The ratio of volumes is computed as $\hat{\mathcal{V}} = \mathcal{V}_{\mathcal{R}}/\mathcal{V}_{\mathcal{O}}$ where $\mathcal{V}_{\mathcal{R}}$ and $\mathcal{V}_{\mathcal{O}}$ are the volumes of $\mathcal{R}$ and $\mathcal{O}$ respectively. In a similar fashion, ratio of surface areas is computed as $\hat{\mathcal{A}} = \mathcal{A}_{\mathcal{R}}/\mathcal{A}_{\mathcal{O}}$, where $\mathcal{A}_{\mathcal{R}}$ and $\mathcal{A}_{\mathcal{O}}$ are the surface areas of $\mathcal{R}$ and $\mathcal{O}$ respectively. A value of one for these ratios indicates a better reconstruction (although not necessarily always).

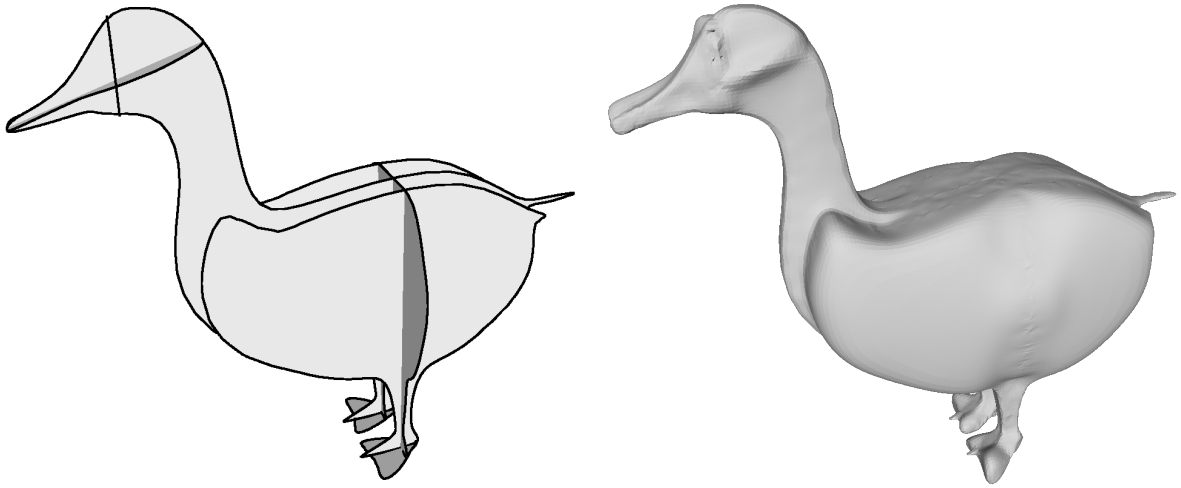
In reconstructions shown in Figures 6.5, 6.7, 6.6, 6.8 and 6.10, it can be observed that the

reconstructed surface is smooth everywhere and combines the cross sections correctly. The comparisons in Figure 6.11 and accuracy measures show that our reconstruction generates a fair surface matching closely with the original object. The homotopy based algorithm results in smooth surface patches with creases at intersection points due to inconsistencies in the function defined on the cutting planes. On the other hand, a reconstruction based on the characteristic function is smooth everywhere except at the cross sections where the discontinuity in the surface is very much evident. The reconstruction quality also depends on the resolution of the volumetric grid. In these example reconstructions, the computations are performed in grids with maximum dimension set to 250.

Reconstructions of Duck and Rook (Figures 6.5, and 6.6) clearly show that the choice of normals at cross section points leads to the reconstructed surface becoming orthogonal to the cutting planes. These reconstructions, however, are free from noise and are topologically correct. In comparison with other reconstructions, the Duck mesh shows better accuracy measures with our algorithm, as illustrated in Table 6.5. While being good accuracy measures, the volume and area ratios do not capture subtle variations in the surface. We notice that the reconstruction of Rook is low on these two ratios, while the Housdorff measure is able to accurately capture the variations in surface (see Table 6.6). The next example of Femur in Figure 6.7 and in Table 6.7 shows good results with our method with as few as seven cross sections.

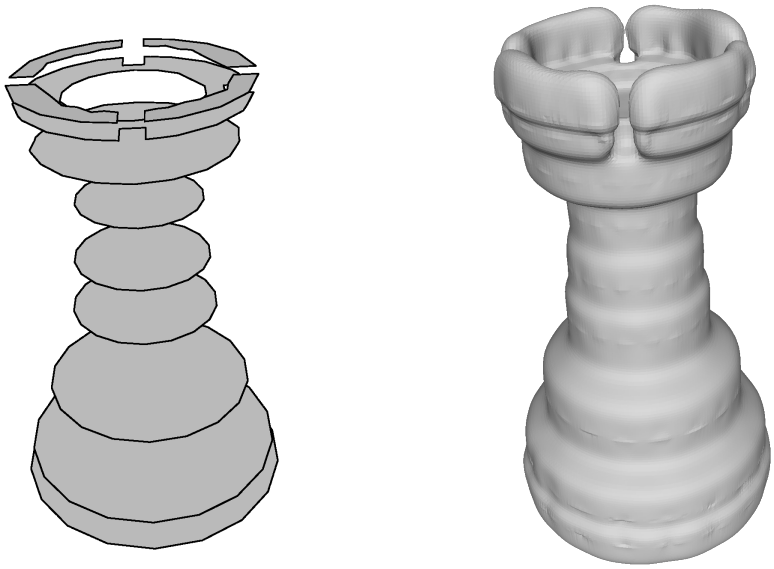
Algorithm	$\hat{\mathcal{V}}$	$\hat{\mathcal{A}}$	$\hat{d}_H$
$\mathcal{R}_{Hom}$	0.8464	0.9331	0.0096
$\mathcal{R}_{\chi}$	0.9899	1.0305	0.0063
$\mathcal{R}_{\mathcal{S}}$	1.1051	1.0871	0.0057

Table 6.5: Accuracy comparison with Duck mesh



(a) (b)

Figure 6.5: Reconstruction results with Duck. (a) A set of seven cross sections, (b) reconstructed surface computed on a grid of size $227 \times 91 \times 250$.

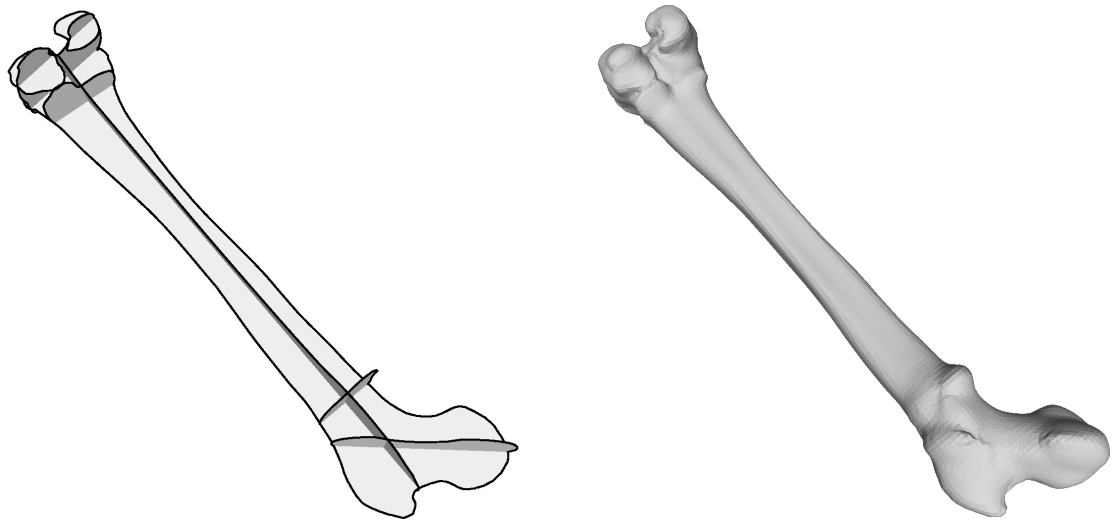


(a) (b)

Figure 6.6: Reconstruction results with Rook. (a) A set of nine cross sections, (b) reconstructed surface computed on a grid of size $250 \times 158 \times 156$.

Algorithm	$\hat{\mathcal{V}}$	$\hat{\mathcal{A}}$	$\hat{d}_H$
$\mathcal{R}_{Hom}$	0.6189	1.0038	0.0192
$\mathcal{R}_{\chi}$	1.0686	1.0668	0.0095
$\mathcal{R}_{\mathcal{S}}$	1.2243	1.1594	0.0064

Table 6.6: Accuracy comparison with Rook mesh



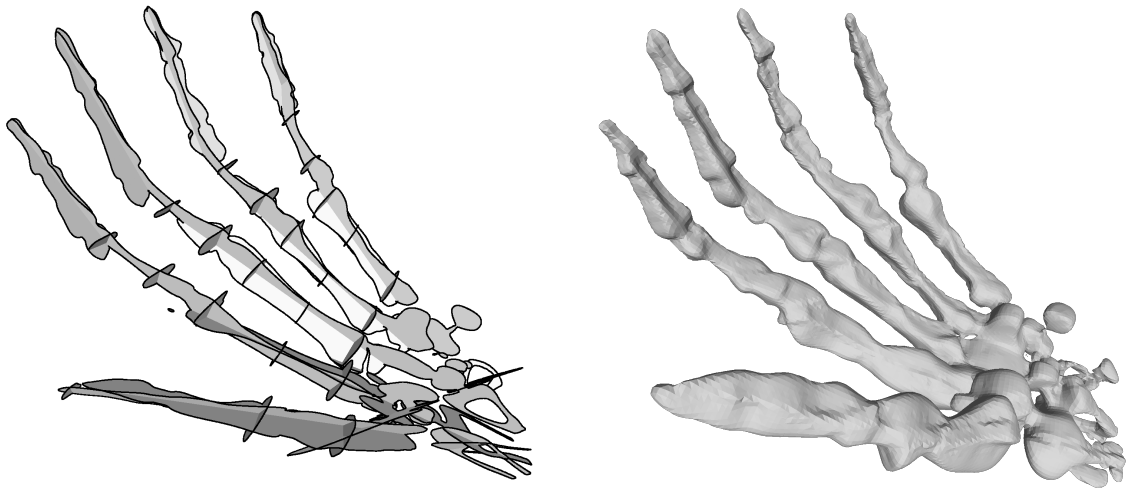
(a) (b)

Figure 6.7: Reconstruction results with Femur. (a) A set of seven cross sections, (b) reconstructed surface computed on a grid of size $38 \times 57 \times 250$.

Algorithm	$\hat{\mathcal{V}}$	$\hat{\mathcal{A}}$	$\hat{d}_H$
$\mathcal{R}_{Hom}$	0.3447	0.7501	0.0151
$\mathcal{R}_{\chi}$	0.9541	0.9784	0.0036
$\mathcal{R}_{\mathcal{S}}$	0.9966	1.0116	0.0034

Table 6.7: Accuracy comparison with Femur mesh

The next three examples of Hand, Horse and Dragon (Figures 6.8, 6.9 and 6.10) show reconstructions of complex objects with our algorithm. A geometric feature can only be reconstructed if it is sampled. Our algorithm also takes advantage from the absence of a signal (that indicates that there is no part of object present at that point). The homotopy based reconstructions show defective surfaces for these meshes (also indicated by low scores in accuracy figures). The characteristic function based approach matches closely with ours in terms of the low frequency details of the surface, but produces a lot of high frequency noise in general.

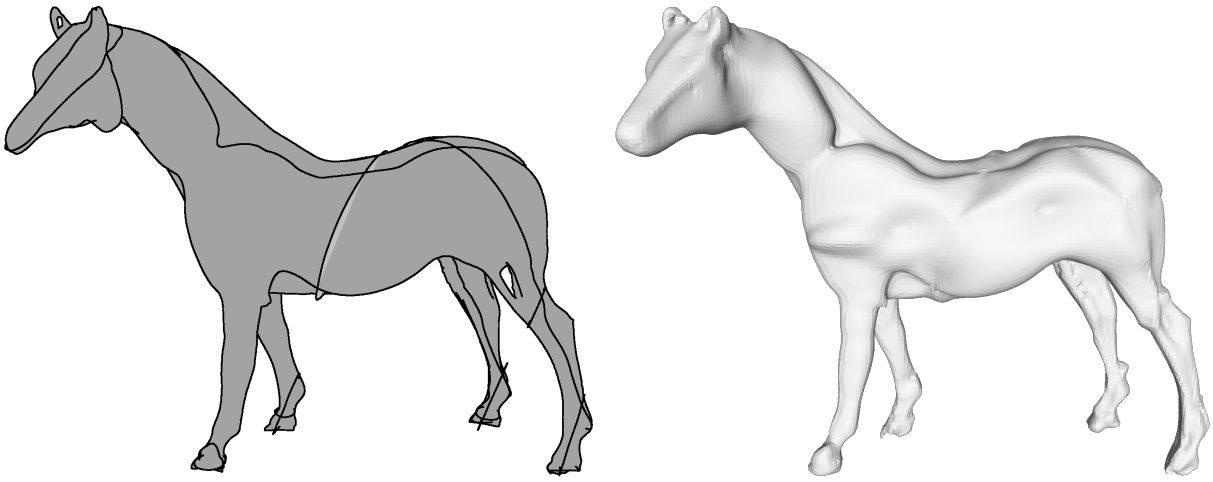


(a) (b)

Figure 6.8: Reconstruction results with Hand. (a) A set of 16 cross sections, (b) reconstructed surface computed on a grid of size $174 \times 250 \times 85$.

Algorithm	$\hat{\mathcal{V}}$	$\hat{\mathcal{A}}$	$\hat{d}_H$
$\mathcal{R}_{Hom}$	0.6694	0.6980	0.0061
$\mathcal{R}_{\chi}$	0.9039	0.8732	0.0037
$\mathcal{R}_{\mathcal{S}}$	1.0411	0.9802	0.0031

Table 6.8: Accuracy comparison with Hand mesh



(a) (b)

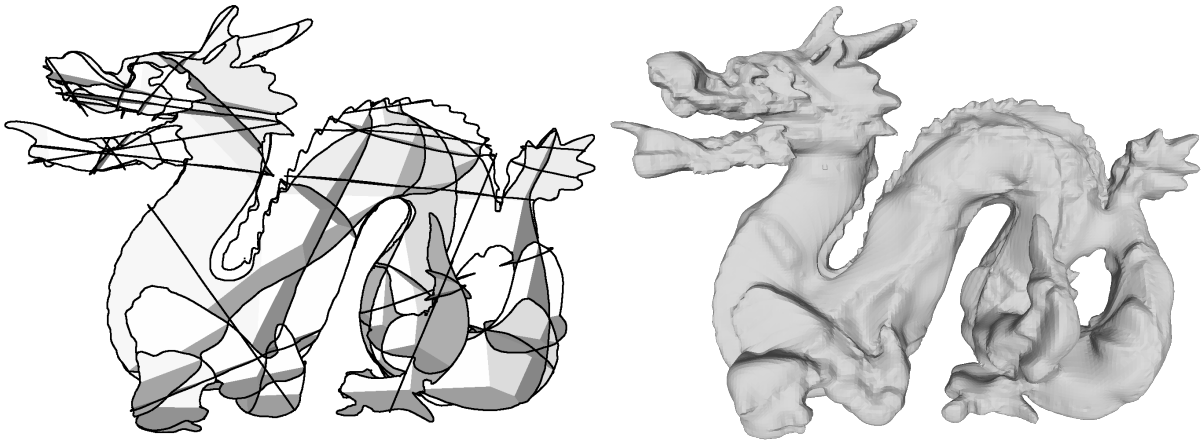
Figure 6.9: Reconstruction results with Horse. (a) A set of 7 cross sections, (b) reconstructed surface computed on a grid of size $250 \times 113 \times 208$.

Algorithm	$\hat{\mathcal{V}}$	$\hat{\mathcal{A}}$	$\hat{d}_H$
$\mathcal{R}_{Hom}$	0.7034	0.8793	0.0094
$\mathcal{R}_{\chi}$	1.0114	1.1412	0.0041
$\mathcal{R}_{\mathcal{S}}$	1.0532	1.0567	0.0036

Table 6.9: Accuracy comparison with Horse mesh

Algorithm	$\hat{\mathcal{V}}$	$\hat{\mathcal{A}}$	$\hat{d}_H$
$\mathcal{R}_{Hom}$	0.9159	0.8966	0.0049
$\mathcal{R}_{\chi}$	0.9903	1.0041	0.0034
$\mathcal{R}_{\mathcal{S}}$	1.0177	1.0334	0.0029

Table 6.10: Accuracy comparison with Dragom mesh



(a) (b)

Figure 6.10: Reconstruction results with Dragon. (a) A set of 23 cross sections, (b) reconstructed surface computed on a grid of size $111 \times 250 \times 176$.

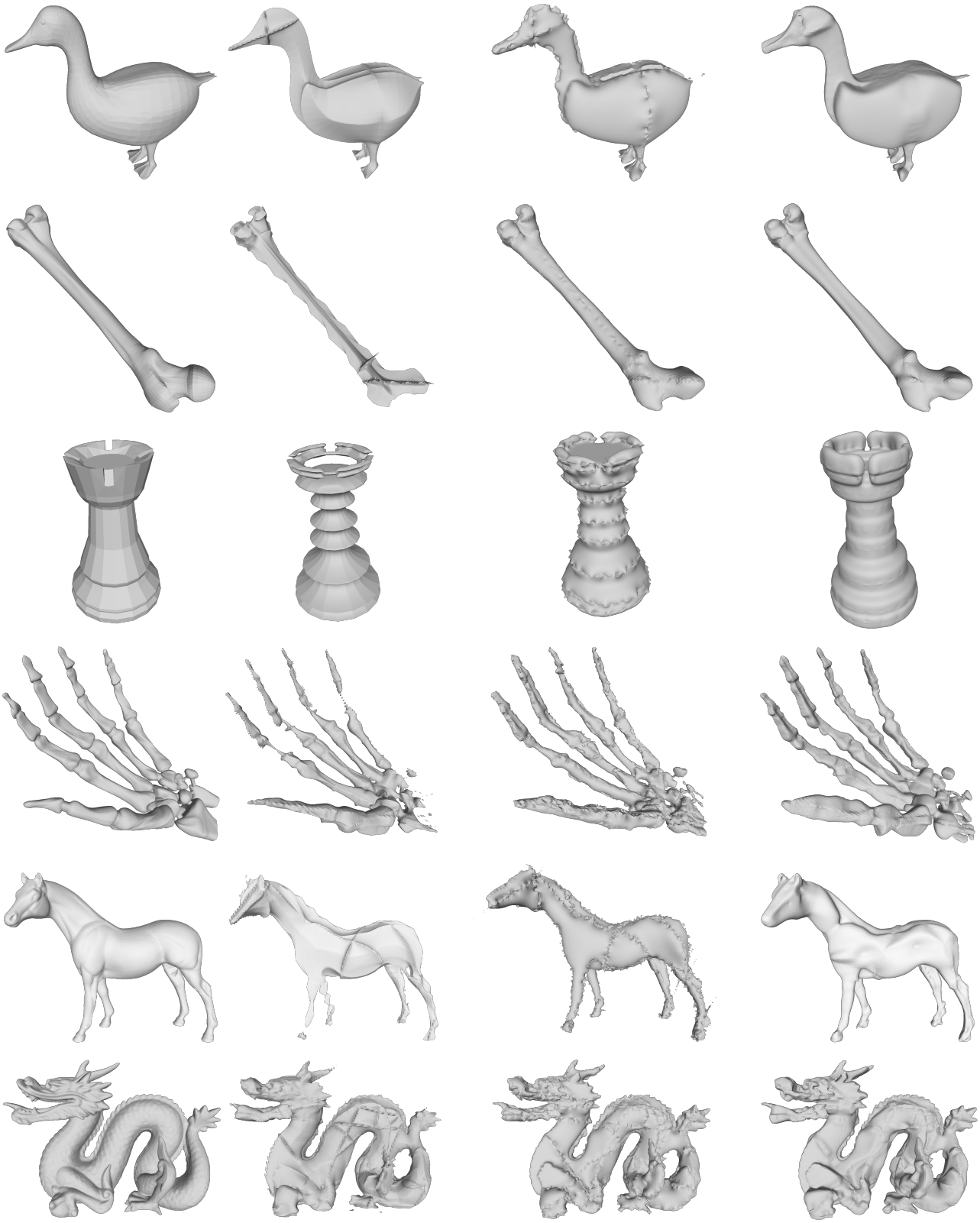


Figure 6.11: Comparison of results of reconstructions from different methods. (a) Original mesh, (b) homotopy based reconstruction, (c) characteristic function based reconstruction, and (d) Signed Distance Field based reconstruction.

Chapter 7

Conclusion and Future Work

7.1 Summary and Discussion

In this work, we addressed the problem of surface reconstruction using given set of closed contours. And for this a simple and robust algorithm from sparse set of unorganized linear cross sections is proposed, which is superior to existing solutions as hermite mean value interpolation is introduced to avoid any requirement for post processing. We illustrated a specialized construction of the signed distance function over the cutting planes that enables a consistent and smooth reconstruction. The proposed algorithm is very intuitive and easy to implement. This solves the problem of reconstructing 3D surfaces from an unorganized set of contours at a new level of generality.

The main drawback of this algorithm is its running time complexity. Another issue can be the handling of large number of slices if number of cutting planes used is quite large. At this point, algorithm becomes cumbersome and difficult to handle arrangement of slices.

7.2 Future Work

In its current form, the algorithm can be further improved by a better choice of normals at points of cross sections. Such normals may be computed by an optimization process. Different barycentrc coordinated in Hermite mean value interpolation should be tried in future. It could generate even better results.

Another interesting problem in 3D is of reconstruction from linear cross sections (instead of planar), but the challenge with this is to consistently partition the space into a set of polyhedra. We plan to address these challenges as part of our future work.

Bibliography

[1] Marching cubes. https://en.wikipedia.org/wiki/Marching_cubes.

[2] Aspert, N., Santa-Cruz, D., and Ebrahimi, T. MESH: Measuring errors between surfaces using the Hausdorff distance. In *IEEE International Conference on Multimedia and Expo* (2002), vol. 1, pp. 705–708.

[3] Barequet, G., and Vaxman, A. Reconstruction of multi-label domains from partial planar cross-sections. In *Computer Graphics Forum* (2009), vol. 28, Wiley Online Library, pp. 1327–1337.

[4] Bermano, A., Vaxman, A., and Gotsman, C. Online reconstruction of 3D objects from arbitrary cross-sections. *ACM Transactions on Graphics (TOG) 30*, 5 (2011), 113.

[5] Bogush, A., and Tuzikov, A. 3d object reconstruction from non-parallel cross-sections. In *17th International Conference on Pattern Recognition (ICPR04)*, vol. 3. IEEE, 2011, pp. 542–545.

[6] Boissonnat, J. D. Shape reconstruction from planar cross sections. *Computer Vision, Graphics, and Image Processing 44*, 1 (1988), 1–29.

[7] Dyken, C., and Floater, M. S. Transfinite mean value interpolation. *Computer Aided Geometric Design 26*, 1 (2009), 117–134.

[8] Fleischmann, P. Boundary integrity. http://www.iue.tuwien.ac.at/phd/fleischmann/node51.html.

[9] Fleischmann, P. Steiner points and steiner triangulation. http://www.iue.tuwien.ac.at/phd/fleischmann/node54.html.

[10] Fuchs, H., Kedem, Z. M., and Uselton, S. P. Optimal surface reconstruction from planar contours. *Communications of the ACM 20*, 10 (1977), 693–702.

[11] George, P.-L. INRIA GAMMA 3D mesh research database. https://www.rocq.inria.fr/gamma/download/, 2013.

[12] Ju, T., Schaefer, S., and Warren, J. Mean value coordinates for closed triangular meshes. In *ACM Transactions on Graphics (TOG)* (2005), vol. 24, ACM, pp. 561–566.

[13] Langer, T., and Seidel, H. P. Higher order barycentric coordinates. In *Computer Graphics Forum* (2008), vol. 27, Wiley Online Library, pp. 459–466.

[14] Legland, D. Geom3D. http://www.pfl-cepia.inra.fr/index.php?page=geom3d, 2012.

[15] Liu, L., Bajaj, C., Deasy, J. O., Low, D. A., and Ju, T. Surface reconstruction from non-parallel curve networks. In *Computer Graphics Forum* (2004), vol. 27, Wiley Online Library, pp. 155–163.

[16] Lorensen, W. E., and Cline, H. E. Marching cubes: A high resolution 3d surface construction algorithm. In *ACM SIGGRAPH Computer Graphics* (1987), vol. 21, ACM, pp. 163–169.

[17] Memari, P., and Boissonnat, J. D. Shape reconstruction from unorganized cross sections. In *5th Eurographics symposium on Geometry processing (SGP '07)* (2007), ACM, pp. 89–98.

[18] Memari, P., and Boissonnat, J. D. Provably good 2D shape reconstruction from unorganized cross-sections. In *Computer Graphics Forum* (2008), vol. 27, Wiley Online Library, pp. 1403–1410.

[19] Munkres, J. *Topology*. Prentice Hall, 1999.

[20] Sharma, O., and Anton, F. Homotopic object reconstruction using natural neighbor barycentric coordinates. In *Transactions on Computational Science XIV*. Springer, 2011, pp. 188–210.

[21] Shewchuk, J. R. Triangle : A two-dimensional quality mesh generator and delaunay triangulator. https://www.cs.cmu.edu/~quake/triangle.html.

[22] Shewchuk, J. R. Triangle : A two-dimensional quality mesh generator and delaunay triangulator, definitions (of several geometric terms). https://www.cs.cmu.edu/~quake/triangle.defs.html.

[23] Shewchuk, J. R. Triangle: Engineering a 2d quality mesh generator and delaunay triangulator. In *Applied computational geometry towards geometric engineering*. Springer, 1996, pp. 203–222.

[24] Shewchuk, J. R. Lecture notes on delaunay mesh generation. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.102.3695&rep=rep1&type=pdf, 1999.

[25] Sidlesky, A., Barequet, G., and Gotsman, C. Polygon reconstruction from line cross-sections. In *18th Canadian Conference on Computational Geometry (CCCG)* (Kingston, Ontario, 2006).

[26] Turk, G., and Mullins, B. Large geometric models archive. http://www.cc.gatech.edu/projects/large_models/, 2003.