Construction of Approximate Medial Shape Representations by Continuous Optimization

by

Daniel Rebain

B.Sc., University of Victoria, 2017

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

University of Victoria

Construction of Approximate Medial Shape Representations by Continuous Optimization

by

Daniel Rebain

B.Sc., University of Victoria, 2017

Supervisory Committee

Dr. Andrea Tagliasacchi, Co-Supervisor
(Department of Computer Science)

Dr. Kwang Moo Yi, Co-Supervisor
(Department of Computer Science)

Supervisory Committee

Dr. Andrea Tagliasacchi, Co-Supervisor
(Department of Computer Science)

Dr. Kwang Moo Yi, Co-Supervisor
(Department of Computer Science)

Abstract

The Medial Axis Transform (MAT) is a powerful tool for shape analysis and manipulation. Traditional methods for working with shapes usually define shapes as boundaries between some "inside" and some "outside" region. While this definition is simple and intuitive, it does not lend itself well to the construction of algorithms for a number of seemingly simple tasks such as classification, deformation, and collision detection. The MAT is an alternative representation of shape that defines the "inside" region by its center and thickness. We present a method of constructing the MAT which overcomes a significant limitation of its use with real-world data: *instability*. As classically defined, the MAT is unstable with respect to the shape boundary that it represents. For data sources afflicted by noise this is a serious problem. We propose an algorithm, LSMAT, which constructs a stable least squares approximation to the MAT.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Acknowledgements

I would like to thank my supervisors:

Andrea Tagliasacchi:

For starting my research career, and for all the time and effort he has spent supporting me through it.

and

Kwang Moo Yi:

For his invaluable guidance in broadening my interests and further developing my research skills.

I would also like to express my profound gratitude to the other students I have worked with in the Visual Computing lab at UVic. The discussions and collaborations I have had with you have helped me become a better researcher, and have made my time in grad school amazing.

Chapter 1

Introduction

1.1 Motivation

The processing and analysis of shapes is a central concept in a number of fields. Almost any application involving physical objects (real or virtual) will need to interact with shapes in some way. Examples of shape processing can be found all the way from the micro scale where images of cancer cells are detected, to the macro scale, where mountain ranges are analyzed to gain insight into the geological processes which created them.

Medial representations are a powerful tool for working with shapes: they encode the solid geometry of an object as the union of a collection of spheres of different radii and origins; see Figures 1.1 and 1.2. The process of converting an input model into a medial representation is generally referred to as the Medial Axis Transform (MAT), and the collection of origins of the spheres in a medial representation form a *medial skeleton*. While volumetric or surface mesh representations are more commonly used in computer graphics and computer vision, since its introduction by Blum et al. [Blum,

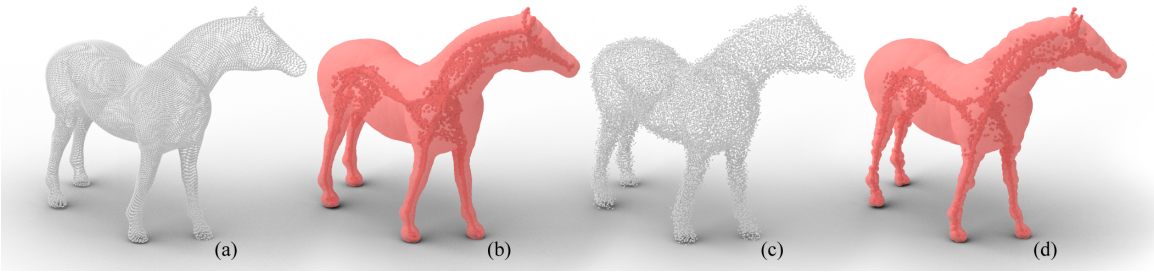

Figure 1.1: The LSMAT is a novel efficient least-squares formulation of the medial axis transform that operates on unorganized oriented point sets. (a, b) On noiseless input, the resulting medial representation is *stable*. (c, d) Its least-squares nature allows it to operate in the presence of heavy noise where most approaches would fail. We visualize the oriented point cloud with oriented splats, draw the union of medial spheres in light red, and their corresponding centers in dark red.

1967], medial representations have found applications in many 2D/3D geometric problems including animation [Baran and Popović, 2007, Yang et al., 2018, Lan et al., 2017, Angles et al., 2019], fabrication [Musialski et al., 2016, Zhang et al., 2015, Quadros, 2016, Ding et al., 2016], image processing [Tsogkas and Dickinson, 2017], shape analysis [Shapira et al., 2008, Peters et al., 2015, He et al., 2015], and real-time tracking [Tkach et al., 2016]. These applications have taken advantage of some useful properties of the MAT. For example, applications involving classification or analysis often benefit from the MAT's clearer representation of topology, which can be characterized by cycles and branches in the skeleton. Medial representations also describe a complete shape without holes and with easy to compute thickness, both of which are attractive in animation and physics simulation where the absence of these properties are a substantial inconvenience of other representations like surface meshes.

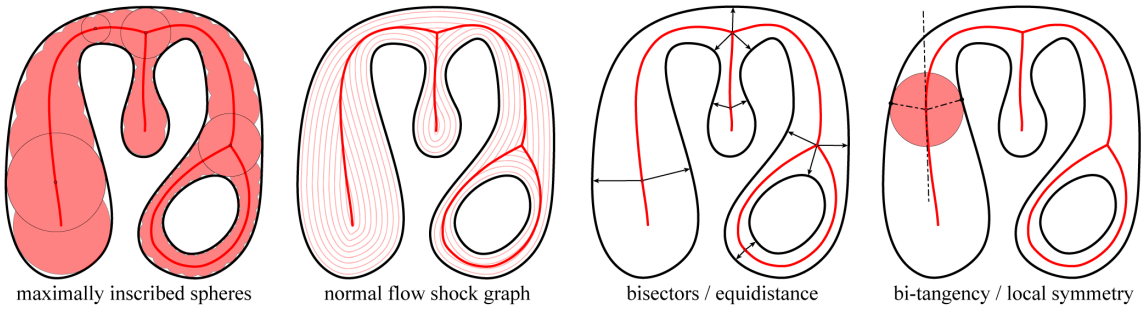

Figure 1.2: Visualization of four alternative definitions of medial axis; base image courtesy of [Tagliasacchi et al., 2016].

1.2 Definitions

As summarized in Figure 1.2, the medial axis transform may be defined in a few ways, with each definition producing equally valid and useful representations. In practice, each definition of the medial representation leads to a different way to compute the medial axis. For example, the *normal flow* variant leads to the commonly employed voxel thinning algorithms [Saha et al., 2016], while the *equidistance* definition leads to techniques leveraging Voronoi diagrams [Brandt and Algazi, 1992]. In this paper, we build over what is likely the most well known definition of the medial axis transform:

> **Definition 1.2.1.** *The Medial Axis Transform MAT($\mathcal{O}$) of $\mathcal{O}$ is the set of centers $\mathcal{M}$ and corresponding radii $\mathcal{R}$ of all maximally inscribed circles/spheres in $\mathcal{O}$.*

While previous work such as Ma et al. [Ma et al., 2012] has proposed methods to construct medial axis representations using this definition, it is interesting to note that they have treated the problem from a *combinatorial geometry* standpoint. Instead, we consider the medial axis transform from a *numerical geometry* perspective, where the

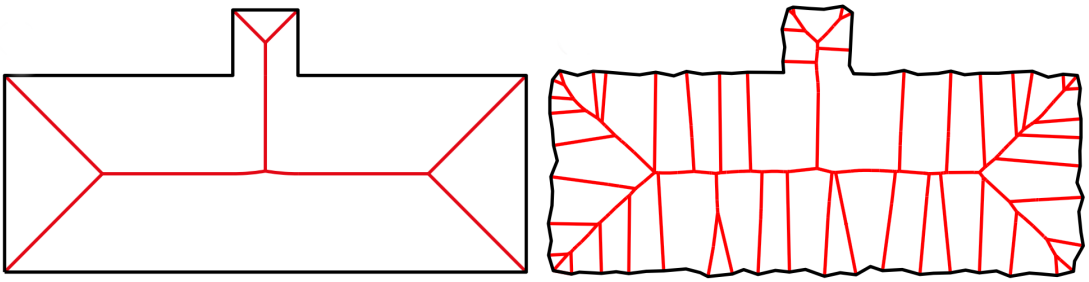

Figure 1.3: Small perturbations of a surface lead to large changes in the structure of the corresponding medial axis.

medial axis is given by the solution of an optimization problem. We achieve this by expressing the concepts of *maximality* and *inscription* in Def. 1.2.1 in a least squares form. The robustness of the approach to imperfections arises from the fact that least squares optimization attempts to find an approximate, as opposed to an exact, solution to the given problem. We are motivated in our approach by considering a least-squares problem as a maximum likelihood estimate of a function in the presence of noise obeying a Gaussian probability distribution.

1.3 Stability

A common pitfall of medial axis methods is their sensitivity to noise: in a representation which exactly satisfies the Medial Axis definition, new branches of the medial skeleton form at all negative local curvature extrema, as these points induce new maximal inscribed spheres near the surface. If our shape is represented via a piecewise linear boundary, such as a polygonal mesh in 3D, spurious branches form when the surface exhibits even minor levels of noise; see Figure 1.3. These issues are typically resolved by resorting to postprocessing, in which sets of spheres are filtered out ac-

cording to various engineered criteria. We present a new algorithm for computing the medial axis transform of an *oriented* point set (points with associated normal vectors) which is naturally capable of handling noise, outliers, and other artifacts that create characteristic problems for traditional methods.

1.4 Method outline

Our method takes an oriented point cloud as input, and produces a medial representation as output by minimizing a combination of a maximality energy, and an inscription energy. Our key technical challenge is in formulating these energies correctly; our maximality energy is designed to have a constant magnitude, ensuring that the optimization energy of each medial sphere is constant regardless of its radius, and our inscription energy is based around a locally supported approximation of the signed distance function of the point cloud. In order to prevent spheres from sliding towards local maxima of local shape thickness, we introduce a pinning constraint as a quadratic barrier energy. The resulting optimization problem is quadratic, with differentiable but non-linear energy terms, and we solve it with an iterative Gauss-Newton solver.

1.5 Contributions

Our main contributions in this work are as follows:

1. We provide a continuous, least squares formulation of the medial axis transform, which has been traditionally defined in the combinatorical context of computational geometry.

2. We provide an efficient, parallel algorithm for computing the MAT based on our

continuous definition.

3. We perform evaluation of our method and comparison against state-of-the-art methods to show that our approach provides greatly improved robustness to noise and outliers.

Part of this work was published in the paper:

- LSMAT Least Squares Medial Axis Transform
  Rebain, D., Angles, B., Valentin, J., Vining, N., Peethambaran, J., Izadi, S., & Tagliasacchi, A.
  Computer Graphics Forum (Vol. 38, No. 6, pp. 5-18), 2019.
  Presented at SGP 2019

1.6 Thesis Organization

This thesis is organized into the following sections:

1. Related Work: We start by reviewing relevant previous and contemporary approaches to medial axis computation.

2. The LSMAT Algorithm: Here we go into detail in describing the mathematical formulation of the optimization and objective function, and rigorously define its constituent energies. Next, we provide detail on the algorithmic structure of the optimization and provide details about its implementation.

3. Results and Evaluation: In this chapter we analyze the results that our method produces and reason about how well it achieves our goals. We also provide comparison against other state of the art methods to demonstrate our claim that we can produce better results than were previously possible.

4. Future Works: As our proposed algorithm opens a new area in medial axis research, we dedicate this chapter to providing a number of potential ways of building on it.

Chapter 2

Related Work

Work on computing the Medial Axis Transform has a long history, as it has been an active area of research since its introduction over 50 years ago. Given the vast amount of literature on the topic a full survey such as [Tagliasacchi et al., 2016] is needed to completely cover the topic. To guide our review, we focus on two properties of potential methods:

- Methods which are based on surface representations of shape, such as point clouds and triangular meshes. While there are a great many methods which consider images or voxel grid representations, we consider these to be beyond the scope of this work.

- Methods that actively attempt to resolve the stability or noise robustness issues with the MAT. These methods are of the greatest interest to us, and will form the basis of our later comparison section.

2.1 Medial Axis Computation of Sampled Surfaces

Assuming the surface is *sampled* by a sufficiently dense set of points, the Voronoi diagram can be used to compute the medial axis transform with relative ease. For a closed boundary in 2D, any interior Voronoi vertex, as well as Voronoi edges connecting interior vertices, approximate the medial axis with a convergence guarantee [Brandt and Algazi, 1992]. Unfortunately, Amenta et al. [Amenta and Bern, 1999] has shown how this property does not hold in 3D due to small, flat tetrahedra that form as part of 3D Voronoi cells, resulting in vertices near the surface that are not part of the medial axis. Thankfully, as we increase the sampling rate, the Voronoi poles (simply described, the farthest Voronoi vertices from the surface in either direction) do converge to the medial axis, allowing the use of 3D Voronoi diagrams for the task at hand. These methods suffer two fundamental shortcoming: ① they are *global* and optimize the entire set of centers at the same time, and ② the extracted axis is heavily susceptible to even minor levels of noise; see Figure 2.1.

Zhong and Chen [Zhong and Chen, 2018] propose a global method which computes the MAT from a distance function which can be made robust to noise. This is effective but requires a discretization of the domain that limits its ability to handle multi-scale or high dimensional data.

2.1.1 Sphere-at-a-point Methods

Leveraging an *oriented* sampling, Ma et al. [Ma et al., 2012] proposed an alternative way to compute the transform by marrying the maximally inscribed definition of Figure 1.2(a) to the bi-tangency of Figure 1.2(d). Unlike Voronoi methods, which consider the entire point set at once, their method can compute a maximal sphere at a point in *isolation*, leading to extremely efficient GPU implementations [Jalba

et al., 2013]. These methods represent the *most efficient* way of computing the medial axis, but, similarly to Voronoi methods, they suffer stability issues; see Figure 3.1. Another method for computing a local approximation to the axis was proposed by Shapira et al. [Shapira et al., 2008] via casting rays in a cone oriented along the anti-normal. The distance between the point and the intersection with the surface is aggregated by a robust function (e.g. median) to estimate the *shape diameter function*. While this approximation has found widespread use as a shape descriptor, the radius estimate suffer of *bias*, and the algorithm does not generalize to point clouds.

2.1.2 Shape Approximation Methods

Recently, a new class of techniques has been proposed which attempt to *approximate* watertight surfaces via *Sphere Meshes* – linearly swept spherical primitives [Thiery et al., 2013, Thiery et al., 2016]. This is achieved via local mesh decimation relying on iterative edge collapses, where *spherical quadrics* are employed in place of the traditional quadric metrics [Garland and Heckbert, 1997]. In many cases the produced model *resembles* a medial axis. When executed on 3D data the result can contain tetrahedra, however, the medial axis of a 3D shape is known to consist only of points, curves, and surfaces. Addressing these concerns, the *Medial Meshes* work by Sun et al. [Sun et al., 2015] extended sphere meshes to decimate a medial axis mesh, and discard unstable branches. Marrying sphere meshes to medial meshes, Li et al. [Li et al., 2015] followed up this work and proposed QMAT, a more computationally efficient version based on spherical quadrics. Furthermore, [Pan et al., 2019] presents improvements to QMAT that increase its controlability and sensitivity to shape features. While these techniques, in juxtaposition to our local method, are global, they can only cope with minor levels of noise: *"with very noisy input, however, the*

simplified medial mesh is not a stable representation"; see [Sun et al., 2015, Fig.9]. The follow-up work by Li et al. [Li et al., 2015] performs slightly better as it can optimize for sphere centers, but our algorithm can still cope with noise that is *one order of magnitude* larger; see [Li et al., 2015, Fig.16].

2.2 Instability and Filtering Techniques

Any techniques that attempt to compute exact solutions to the MAT definition will unavoidably suffer from instability. Filtering techniques attempt to *remove* portions of the medial axis that do not contribute significantly to the geometry *reconstructed* as the union of medial spheres. As shown in [Miklos et al., 2010, Fig.2], this works fairly well for inputs with little or no noise. Conversely, with large noise (and/or outliers), these methods become mostly inappropriate; see Figure 2.1.

2.2.1 Angle Filtering: θ-Medial Axis

One way to identify the significance of a point is the largest angle formed by the center of the corresponding maximal sphere and two of its tangent points on the shape boundary. The θ-medial axis of [Foskey et al., 2003], filters out balls associated with low aperture angle. While this filtering can disconnect portions of the medial axis, see Figure 2.1(c), the issue can be avoided by homotopy-preserving pruning [Attali and Montanvert, 1997].

2.2.2 Distance Filtering: λ-Medial Axis

Another metric for filtering discards a sphere whenever the distance between its tangent points on the surface lie below a certain threshold [Chazal and Lieutier, 2005, Amenta

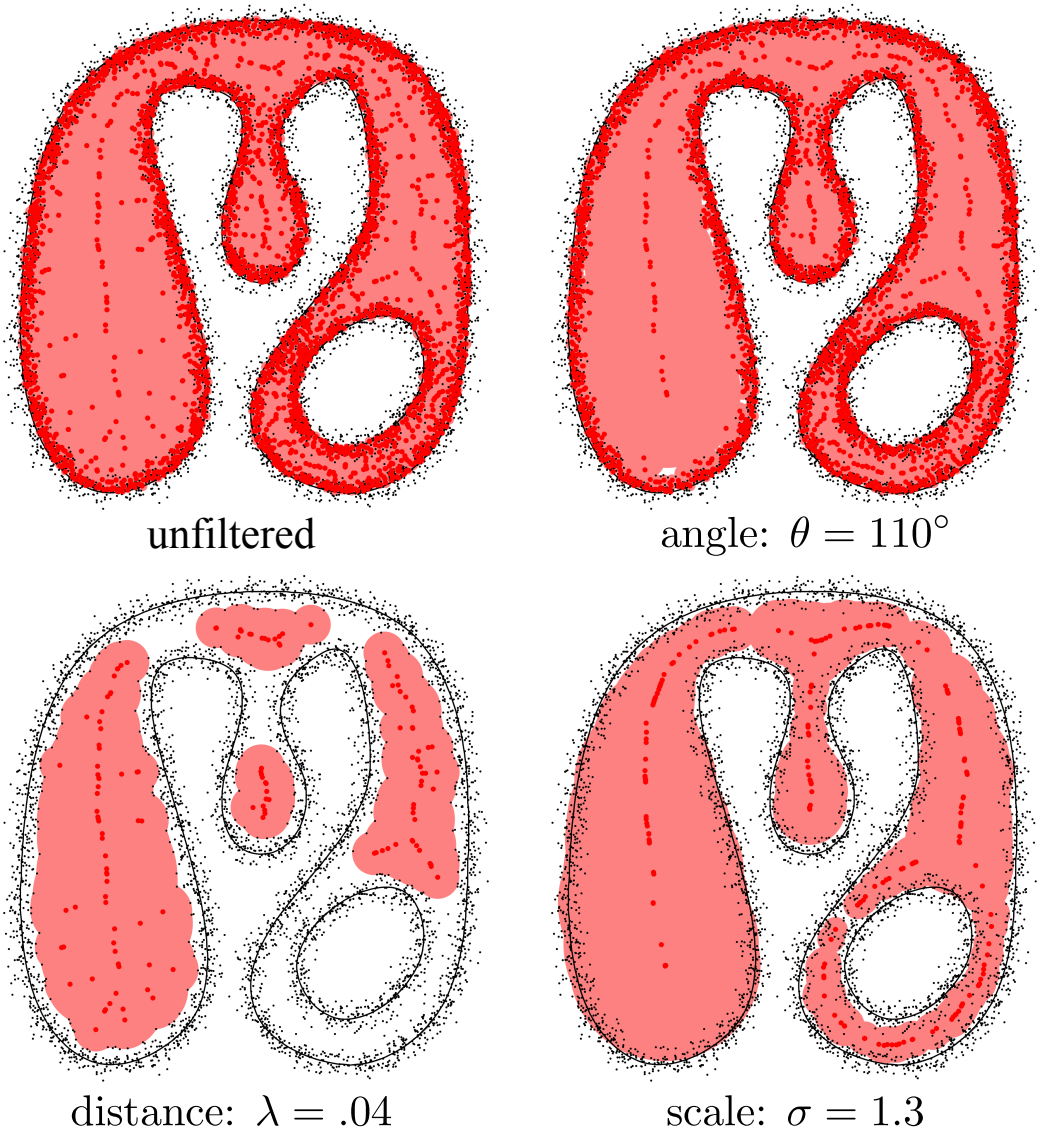

Figure 2.1: Effectiveness of standard filtering Voronoi methods with high level of noise. As expected, neither angle nor distance methods were capable of filtering out all noise for any selected threshold value. Leveraging global information, the scale axis is able to deal with noise, but with a significant computational burden. Further, note that we provide these technique the ground truth inside/outside labeling, which is *not available* in our context.

et al., 2001]. As Figure 2.1(d) illustrates, this results in a loss of features even before all noise has been removed. This shortcoming makes these solutions inappropriate whenever the input shape contains structures at different scales.

2.2.3 Scale Filtering: σ-Medial Axis

The scale axis transform introduced by Giesen et al. [Giesen et al., 2009] and extended to 3D by Miklos et al. [Miklos et al., 2010] are methods built over the *maximality* property of the medial axis. First medial spheres are scaled by a given $\sigma > 1$ value, and spheres that are no longer maximal (i.e. now contained entirely inside another scaled sphere) get discarded. The medial axis of the scaled union of balls is then computed, and its spheres unscaled by a factor $1/\sigma$. This method is *global* as it processes the whole point set at once, and computes multiple Voronoi diagrams in the process, resulting in reduced computational efficiency – e.g. ≈ 2 *min.* for a mesh with $\approx 100k$ vertices, see [Miklos et al., 2010, Tab.4.1].

2.3 Machine Learning Techniques

In recent years, neural networks and deep learning have begun to see applications to problems all across computer science, and the MAT is no exception [Ke et al., 2017, Liu et al., 2017, Liu et al., 2018, Panichev and Voloshyna, 2019]. These works have so far largely focused on finding skeletons in the image domain. In this context they can leverage the powerful convolutional neural network architecture, which has proven to be wildly effective in a very wide range of image-based tasks.

While these methods are no doubt capable of state of the art results on images, we are unaware of any neural networks which have been shown to match geometry-based

approaches on higher-dimensional or unstructured data where surface-based methods are required and their advantages are less significant.

Chapter 3

The LSMAT Algorithm

This chapter lays out the technical details of our method.

- We establish in Section 3.1 the notation which we use to describe our mathematical formulation.
- Before describing our algorithm, in Section 3.2 we review highly relevant method of Ma et al. [Ma et al., 2012], which computes the medial axis by searching for maximal spheres via *local* analysis.
- We then formulate our numerical optimization to compute medial spheres in Section 3.3.
- Finally in Section 3.4, we give the algorithmic and implementation details needed to reproduce the LSMAT algorithm.

3.1 Notation

We are given in input an *oriented* point cloud $\{(\mathbf{p}_n, \mathbf{n}_n)\}_{n=1...N}$ drawn from the solid object $\mathcal{O}$ with watertight surface/boundary $\partial\mathcal{O}$. The cloud is affected by noise with standard deviation $\sigma_{\mathbf{p}}$. We give all numerical values of $\sigma_{\mathbf{p}}$ and other parameters which represent distance values as *percentages* relative to the diagonal of the bounding box of the input shape. With $\mathcal{M}$ we refer to the *internal* medial axis of $\mathcal{O}$; the set of medial points whose centers are inside the shape. With $\mathbf{s} = (\mathbf{c}, r)$ we indicate a sphere centered at $\mathbf{c}$ of radius r, and with t the index of the solver iteration.

3.2 The Sphere-shrinking Algorithm

The algorithm proposed by Ma et al. [Ma et al., 2012] is an iterative method that shrinks an initially large sphere until it satisfies the medial axis properties. Assuming $\mathbf{p}$ is a point on the boundary, this process leverages two properties: ① that the sphere passing through $\mathbf{p}$ should be empty; and ② that $\mathbf{p} - \mathbf{c}$ for a smooth input curve is parallel to the contact point normal $\mathbf{n}$. The sphere-shrinking algorithm is initialized with a large sphere, passing through $\mathbf{p}$, containing the entire point set, and whose center $\mathbf{c}$ lies along the ray $(\mathbf{p}, -\mathbf{n})$. Whenever the closest point $\mathbf{f} \in \partial\mathcal{O}$ from $\mathbf{c}$ is closer than r (i.e. the sphere is not empty), the center $\mathbf{c}$ is updated by finding the sphere that is tangent to $(\mathbf{p}, \mathbf{n})$ and passing through $\mathbf{f}$; see Figure 3.1. Finally, the loop terminates when the sphere radii change across iterations is below numerical precision. This algorithm is highly efficient as each sample can be processed completely independently from the others, but as it relies on combinatorial search it does not cope well with noisy inputs. Our formulation borrows from this formulation, but generalizes it by ① reformulating it into a *continuous* optimization problem, and ② making it more

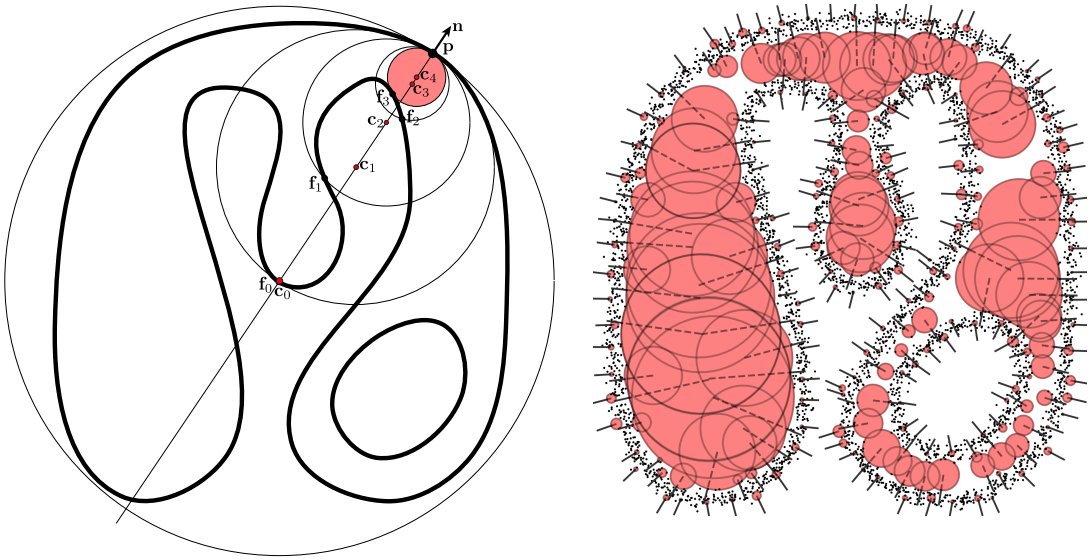

Figure 3.1: (left) The sphere-shrinking of Ma et al. [Ma et al., 2012], and (right) its resulting maximal spheres on a noisy point set. The solid lines indicate point normals, which define the space along which updated positions of the associated sphere centers may lie (dashed lines).

robust to noise and outliers.

3.3 Maximally Inscribed Spheres in Least Squares

We define a least-squares optimization capable of generating *maximally inscribed spheres* given an *oriented* point cloud $\mathcal{P}$. Our optimization energy for a sphere $\mathbf{s}$ consists of the combination of two terms:

$$E_{\text{medial}} = \omega_1 \, E_{\text{maximal}} \; + \; \omega_2 \, E_{\text{inscribed}} \tag{3.1}$$

We will now proceed to describe these terms in detail, and then provide comparison against alternative formulations in Section 4.2.

3.3.1 Maximality

The maximality energy creates a constant positive pressure term that tends to increase the sphere size at each iteration. By design, we define this energy to have a *constant magnitude*, that is, whose contribution to the optimization energy is constant regardless of the radii of the given medial sphere. This is achieved by considering the radius at the previous optimization iteration, which is constant for the purposes of optimization:

$$E_{\text{maximal}} = ||r - (r^{t-1} + \epsilon)||_2^2 \tag{3.2}$$

During optimization, the magnitude of this energy's residual will always evaluate to ϵ.

3.3.2 Inscription

If the expression of the (SDF) signed distance function $\Phi(\mathbf{x})$ were available to us, an inscription constraint could be easily expressed by the inequality $\Phi(\mathbf{c}) < r$, which we

can express in our least squares formalism as:

$$E_{\text{inscribed}} = \mathcal{R}(r + \Phi(\mathbf{c}))^2 \tag{3.3}$$

$$\mathcal{R}(x) = max(x, 0) \tag{3.4}$$

where the difference $r+\Phi(\mathbf{c})$ indicates the separation between the sphere boundary and the surface. The inclusion of the ramp function $\mathcal{R}(x)$ means we only penalize a sphere when it violates the inscription condition by crossing the surface. Unfortunately, an SDF function for our oriented point set $\mathcal{P}$ is not readily available without computing a full surface reconstruction of the point cloud [Berger et al., 2016]. However, the family of moving least square (MLS) methods are capable of building locally supported approximations in a neighborhood of $\partial\mathcal{O}$ – that is, as $\Phi \to 0$. For example, the MLS formulation by Kolluri [Kolluri, 2008] approximates Φ as a weighted sum of locally supported point-to-plane functions:

$$\Phi(\mathbf{x}) = \frac{\sum_n \varphi_n(\mathbf{x}, h)\, \mathbf{n}_n \cdot (\mathbf{x} - \mathbf{p}_n)}{\sum_n \varphi_n(\mathbf{x}, h)} \tag{3.5}$$

where in our case $\varphi(x, h)$ is a smoothly decaying radial basis function [Guennebaud and Gross, 2007] with compact support $[0, h]$:

$$\varphi(x, h) = b\left(\frac{x}{h}\right); \quad b(x) = \begin{cases} (1 - x^2)^4 & x < 1 \\ 0 & x \geq 1 \end{cases} \tag{3.6}$$

$$\varphi_n(\mathbf{x}, h) = \varphi(\|\mathbf{x} - \mathbf{p}_n\|_2, h) \tag{3.7}$$

Note that as Equation 3.3 evaluates the function Φ at the medial center, which could be potentially far from $\partial\mathcal{O}$, this representation is not immediately appropriate. However, in the context of registration, several works have shown how a quadratic

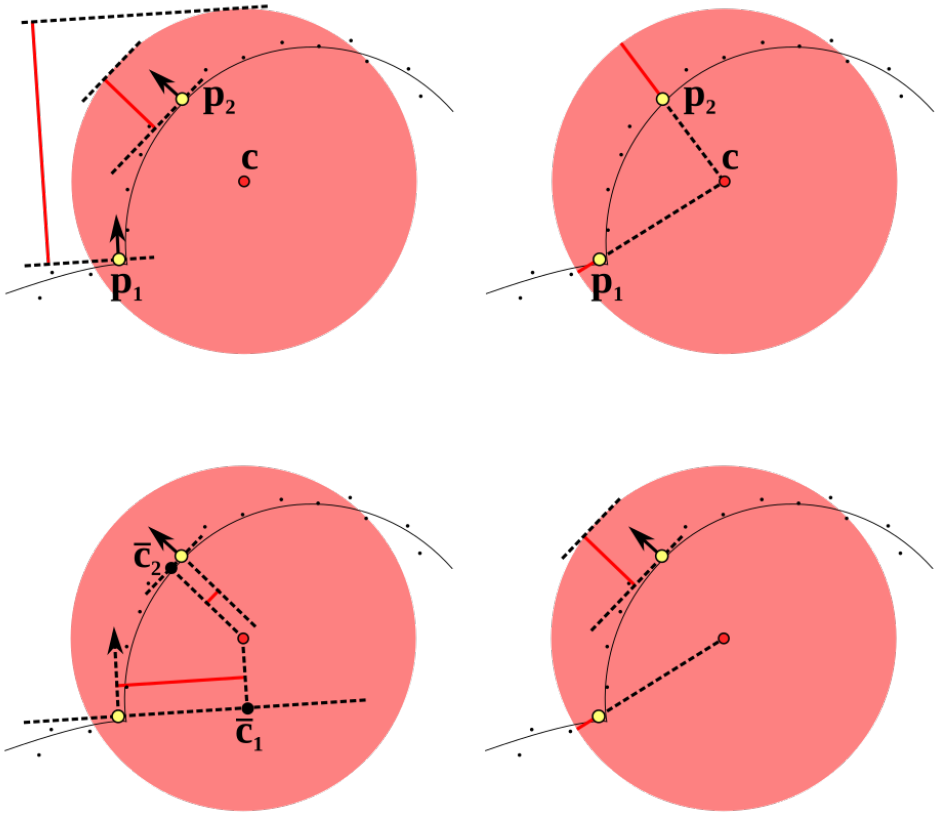

Figure 3.2: **top left:** The function $\Phi_{\text{halfplane}}$ uses point normals to give a signed distance value that penalizes spheres which move outside the shape, but may give nonsensical values for points belonging to parts of the surface that are not well approximated by the sphere. **top right:** The function Φ_{point} does not suffer from this issue but does not enforce that the sphere should respect the orientation of the surface. **bottom left:** We mix between the two distances independently for each point using the distance between the point and the projection of the sphere center onto the plane defined by the point. **bottom right:** This allows us to use the sided distance only for points where it makes sense, and use the unsigned point-to-plane function otherwise.

approximation of Φ^2 can be built by appropriately blending *point-to-point* with *point-to-plane* distance functions [Pottmann and Hofer, 2003, Mitra et al., 2004]. In more details, point-to-point distances are a good approximation of Φ^2 in the *far-field* (i.e. for $\Phi^2 \gg 0$), while point-to-plane distances are more suitable in the *near-field* (i.e. for $\Phi^2 \to 0$). Let us first define $\bar{\mathbf{c}}_n = \mathbf{c}^{t-1} - \mathbf{n}_n(\mathbf{c}^{t-1} - \mathbf{p}_n) \cdot \mathbf{n}_n$, the projection of the center on the hyperplane of point $\mathbf{p}_n$. We can use this to interpolate between the two metrics (see also Figure 3.2):

$$\Phi_{\text{blend}}(\mathbf{s})^2 = \text{mix}(\Phi_{\text{plane}}(\mathbf{s})^2, \Phi_{\text{point}}(\mathbf{s})^2, \varphi_n(\bar{\mathbf{c}}_n, h_{\text{blend}})) \tag{3.8}$$

$$\Phi_{\text{plane}}(\mathbf{s}) = \mathcal{R}(r - (\mathbf{p}_n - \mathbf{c}) \cdot \mathbf{n}_n) \tag{3.9}$$

$$\Phi_{\text{point}}(\mathbf{s}) = \mathcal{R}(r - \|\mathbf{p}_n - \mathbf{c}\|_2) \tag{3.10}$$

$$\text{mix}(a, b, x) = xa + (1 - x)b \tag{3.11}$$

Based on the geometry of Φ^2_{blend}, we can then re-formulate our inscription with respect to each oriented point $\mathbf{p}_n$. Analogously to Equation 3.5, this energy is accumulated over all points in $\mathcal{P}$:

$$E_{\text{inscribed}} \approx \sum_n \underbrace{\varphi(\mathcal{R}(||\mathbf{c}^{t-1} - \mathbf{p}_n|| - r^{t-1}), h_{\text{support}})}_{\neq 0 \text{ for a subset of the N points}} \Phi_{\text{blend}}(\mathbf{s})^2 \tag{3.12}$$

The parameter h_{blend} defines the scale used for blending between distance types, and h_{support} limits how far outside of a sphere a point may be before its contribution falls to zero. As is typical in robust optimization (e.g. IRLS) the weights are computed with respect to the parameters $\mathbf{s}^{t-1}$ at the previous iteration – inscription is evaluated only for points within, or in proximity of the sphere in its previous geometric configuration $\mathbf{s}^{t-1}$.

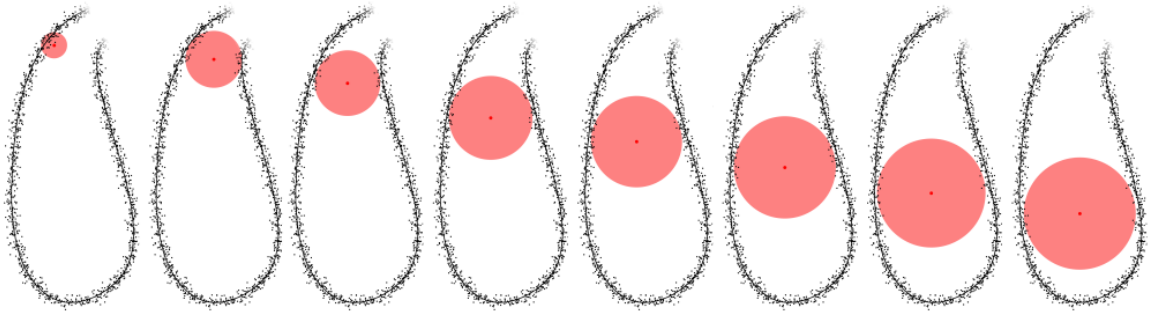

Figure 3.3: While the algorithm is initialized in a neighborhood, optimizing for larger spheres will cause the solver to have the sphere converge to areas of locally maximal radii.

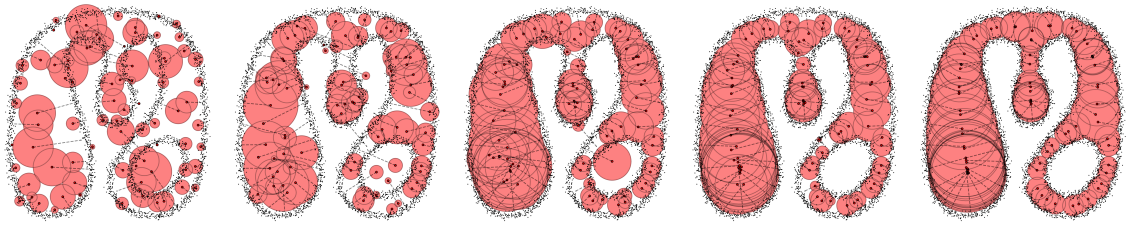

Figure 3.4: Iterations of LSMAT optimization on an input noisy point set. We randomly initialize sphere centers and radii, and demonstrate the excellent convergence properties of our optimization. Here the center-pin correspondence is marked by the dotted line. Notice how although some spheres are initially outside, the optimization pushes them into the interior of the point set.

3.3.3 Pinned spheres

The medial axis provides an estimate of the *local thickness* of the shape through its sphere local radius. However, as our variational formulation attempts to create *larger* spheres, nothing prevents a sphere from traveling along medial branches wherever we have a non-vanishing medial radii gradient on $\mathcal{M}$; e.g. a sphere would slide from the tip of a cone to its base; see Figure 3.3. We can avoid this issue by "pinning" medial spheres. Generalizing the exact constraints of the sphere-shrinking algorithm by [Ma et al., 2012], we subject the sphere associated with a given point $\mathbf{p}$ to the *hard* constraint :

$$\|\mathbf{c} - \mathbf{p}\| - r \leq d_{\text{pin}} \tag{3.13}$$

which keeps the sphere in contact with a sphere of radius d_{pin} centered at $\mathbf{p}$. We include this constraint in our optimization via the penalty method [Nocedal and Wright, 2006] which approximates the hard constraint within a least squares framework, yielding a quadratic barrier energy:

$$E_{\text{pinning}} = \mathcal{R}(\|\mathbf{c} - \mathbf{p}\| - (r + d_{\text{pin}}))^2 \tag{3.14}$$

3.4 Optimization

Similarly to [Lipman et al., 2007], our optimization induces the definition of medial sphere as the *fixed point solution* of an update equation:

$$\mathbf{s} = \mathcal{F}(\mathbf{s}) = \arg\min_{\mathbf{s}} \; E_{\text{medial}} + E_{\text{pinning}} \tag{3.15}$$

Our optimization problem is quadratic, but while its energy terms are differentiable, they are not linear. Hence, we iteratively compute a solution via Gauss-Newton

Algorithm 1 Gauss-Newton

The basic algorithm for computing the D parameters of a sphere at the next step of the optimization given the sphere at the previous step $\mathbf{s}$ and a set of constraint residual functions $\mathcal{C}$.

1: **procedure** STEP($\mathbf{s}$, $\mathcal{C}$)
2: $J^T J \leftarrow \text{zeros}(D, D)$
3: $J^T R \leftarrow \text{zeros}(D, 1)$
4: **for** $\mathcal{C}_i \in \mathcal{C}$ **do**
5: $R_i = \mathcal{C}_i(\mathbf{s})$
6: $J_i = \nabla_{\mathbf{s}} \mathcal{C}_i(\mathbf{s})$
7: $J^T J \leftarrow J^T J + J_i^T J_i$
8: $J^T R \leftarrow J^T R + J_i^T R_i$
9: $\Delta\mathbf{s} = (J^T J)^{-1} J^T R$
10: $\mathbf{s} = \mathbf{s} - \Delta\mathbf{s}$

(see Algorithm 1). This requires the linearization of the arguments of the quadratic functions with respect to $\mathbf{c}$ and r, which is straightforward in our setting.

3.4.1 Implementation

Due to the independent nature of the per-sphere optimization, our implementation is straightforward. At each step we compute Jacobian matrices and residual vectors separately for each sphere and thus are only required to solve a $D \times D$ linear system for each, where D is the number of degrees of freedom of a single sphere (3 or 4 in our experiments). This property eliminates the need for any advanced solvers or factorizations and enables the implementation to be completely parallel over the spheres, as required for effective GPU acceleration. Additional performance optimization can be achieved by collecting the points for each sphere whose contribution is non-zero using accelerating data structures such as kd-trees [Friedman et al., 1977].

3.4.2 Parallelism

When writing our 3D implementation, we took advantage of GPU acceleration to gain a significant decrease in run-times. While we were not able to match the speed of our sphere-shrinking reference implementation (also GPU accelerated, see Table 4.1), we did achieve nearly comparable per-iteration performance while producing substantially higher-quality results (see Figure 4.6).
Pseudocode of our GPU implementation can be found in Algorithm 2.

3.4.3 Gradients of Energy Components

Implementing the optimization described above requires evaluating the energy terms as residuals (the expressions inside the squares). In this section we provide the

Algorithm 2 Parallel LSMAT

The procedure for computing a single step of the parallel optimization, given the set of n spheres at the current iteration S and constraint residual functions $\mathcal{C}$.

procedure ParallelStep(S, $\mathcal{C}$)
    $J^T J \leftarrow \text{zeros}(n, D, D)$
    $J^T R \leftarrow \text{zeros}(n, D, 1)$
    for $\mathcal{C}_i \in \mathcal{C}$ **do**
        for $\mathbf{s}_j \in S$ **do in parallel**
            $R_{i,j} = \mathcal{C}_i(\mathbf{s}_j)$
            $J_{i,j} = \nabla_{\mathbf{s}} \mathcal{C}_i(\mathbf{s}_j)$
            $(J^T J)_j \leftarrow (J^T J)_j + J_{i,j}^T J_{i,j}$
            $(J^T R)_j \leftarrow (J^T R)_j + J_{i,j}^T R_{i,j}$

    for $\mathbf{s}_j \in S$ **do in parallel**
        $\Delta \mathbf{s}_j = (J^T J)_j^{-1} (J^T R)_j$
        $\mathbf{s}_j = \mathbf{s}_j - \Delta \mathbf{s}_j$

analytic formulas for each residual as well as that of its gradient with respect to its corresponding sphere. Gradients are written in the form $\nabla_{\mathbf{s}} f(\mathbf{s}) = \left[\nabla_{\mathbf{c}} f(\mathbf{s}), \frac{\partial f(\mathbf{s})}{\partial r}\right]$.

Inscription Term. Given oriented surface point $(\mathbf{p}_n, \mathbf{n}_n)$:

$$\nabla_{\mathbf{s}} \Phi_{\text{plane}}(\mathbf{s}) = \mathcal{H}(\Phi_{\text{plane}}(\mathbf{s}))[-\mathbf{n}_n, 1] \tag{3.16}$$

$$\nabla_{\mathbf{s}} \Phi_{\text{point}}(\mathbf{s}) = \mathcal{H}(\Phi_{\text{point}}(\mathbf{s})) \left[\frac{\mathbf{p}_n - \mathbf{c}}{\|\mathbf{p}_n - \mathbf{c}\|_2}, 1\right] \tag{3.17}$$

$$\nabla_{\mathbf{s}} \Phi_{\text{blend}}(\mathbf{s}) = \text{mix}(\nabla_{\mathbf{s}} \Phi_{\text{plane}}(\mathbf{s}), \nabla_{\mathbf{s}} \Phi_{\text{point}}(\mathbf{s}), \varphi_n(\bar{\mathbf{c}}_n, h_{\text{blend}})) \tag{3.18}$$

Note that this intentionally does not include the weight term in Equation 3.12 based on proximity to the sphere. The reason for this is that this weight is taken to be from the previous iteration, much like weights in the IRLS algorithm. Similarly to that algorithm's weights, these make the optimization robust and local without interfering with the optimum implied by the inner energy function.

Maximality Term.

$$\mathbf{R}_{\text{maximal}} = r - (r^{t-1} + \epsilon) \tag{3.19}$$

$$\nabla_{\mathbf{s}} \mathbf{R}_{\text{maximal}} = [0, 1] \tag{3.20}$$

Again, this energy is written in terms of a past value of our optimization variable. For the purposes of the gradient in our optimization, we are directly penalizing the deviation of the radius from a constant.

Pinning Term. Given pin point $\mathbf{p}$:

$$\mathbf{R}_{\text{pinning}} = \mathcal{R}(\|\mathbf{c} - \mathbf{p}\| - (r + d_{\text{pin}})) \tag{3.21}$$

$$\nabla_{\mathbf{s}} \mathbf{R}_{\text{pinning}} = \mathcal{H}(\mathbf{R}_{\text{pinning}}) \left[\frac{\mathbf{c} - \mathbf{p}}{\|\mathbf{c} - \mathbf{p}\|_2}, -1\right] \tag{3.22}$$

Chapter 4

Results and Evaluation

We show medial representations generated by our method, in both 2D and 3D, throughout the paper and in Figure 4.1. As shown, our method produces valid medial representations for a wide variety of shapes and models, and in the presence of noise and outliers. Throughout the paper, we process all input with the same parameters whose values were set according to the analysis in Section 4.6. We evaluate our method in several ways. First, we consider variations of the inscription and maximality energy, and show how alternative formulations fail. Second, we evaluate the performance of our algorithm against ground truth on synthetic benchmarks contaminated by noise and outliers. Third, we compare our generated medial representations against the state-of-the-art in both 2D and 3D, again in the presence of noise. Finally, we provide an analysis of our algorithm parameters.

Figure 4.1: A gallery of LSMAT results with varying levels of noise on shapes with complex topology and varying feature size. In the callout for the octopus, notice how the MLS kernel overlaps nearby surfaces, yet the algorithm can cope by producing erroneously located medial spheres with a *zero* radii.

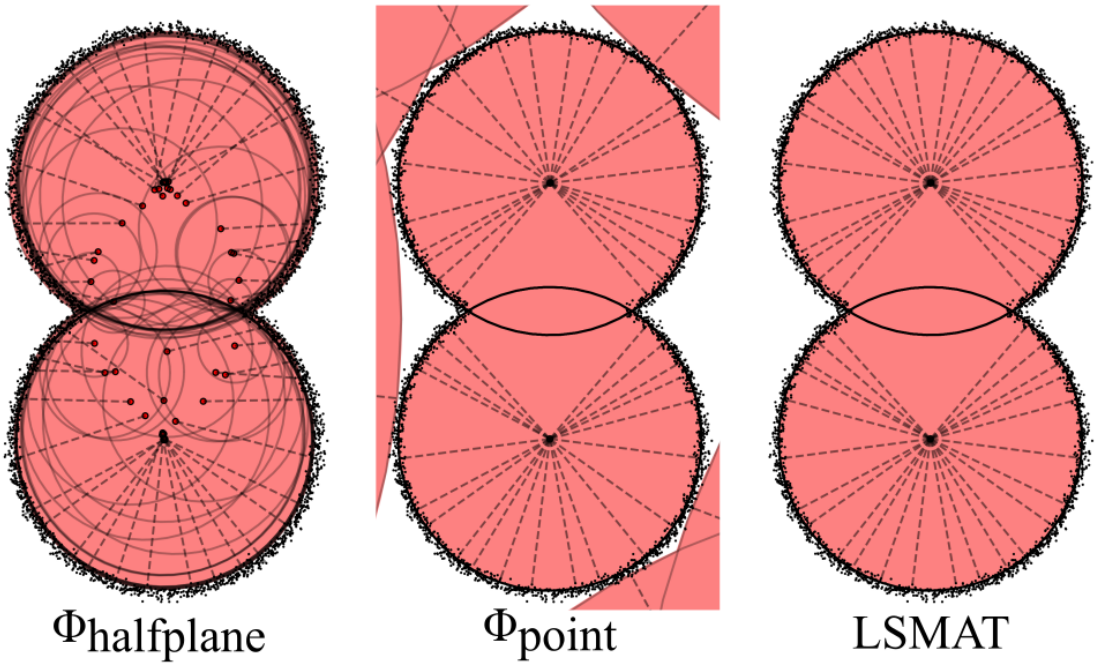

Figure 4.2: Qualitative evaluation of inscription energy variants.

4.1 Variants of Inscription Energy

We consider two modified formulations of Equation 3.12 in which we either penalize the squared distance to points, or the squared distance to half-planes as opposed to our blended formulation. We investigate this behavior by randomly initializing the algorithm, and snapping $\varphi_n(\bar{\mathbf{c}}_n)$ to either zero or one for all points. When $\varphi_n(\bar{\mathbf{c}}_n) = 0$, point-to-point energy is used and the algorithm attempts to make *spheres empty* in a least square sense. However, nothing prevents the optimization from generating maximal spheres *outside* the shape in the ambient space. When $\varphi_n(\bar{\mathbf{c}}_n) = 1$, point-to-plane energy is used, and the algorithm attempts to make *half-spaces empty*. As illustrated in Figure 4.2, this results in difficulties for the algorithm in dealing with sharp concavities. In the example above, we expect all centers to cluster to one of the two centers, but spheres in the neighborhood of the sharp concavity might read the normal of a point on the opposite side, with a halfplane requesting the radius of a sphere intersecting with it to be significantly smaller. This problem is caused by the fact that point-to-point and point-to-plane energies approximate the squared SDF of a point set respectively in the far and near field. Our LSMAT formulation respects this geometric property, and deals with both issues at once.

4.2 Variants of Maximality Energy

We consider two alternative formulations of the maximality condition in our optimization: penalizing the squared inverse of the sphere radius, expressed as $\|1/r\|^2$, or directly specifying $R_{\max}$, the maximum size of a medial sphere, and optimizing $\|r - R_{\max}\|^2$. While intuitively these are potentially feasible solutions, they incur a significant limitation: the amount of "pressure" a medial sphere will apply will be

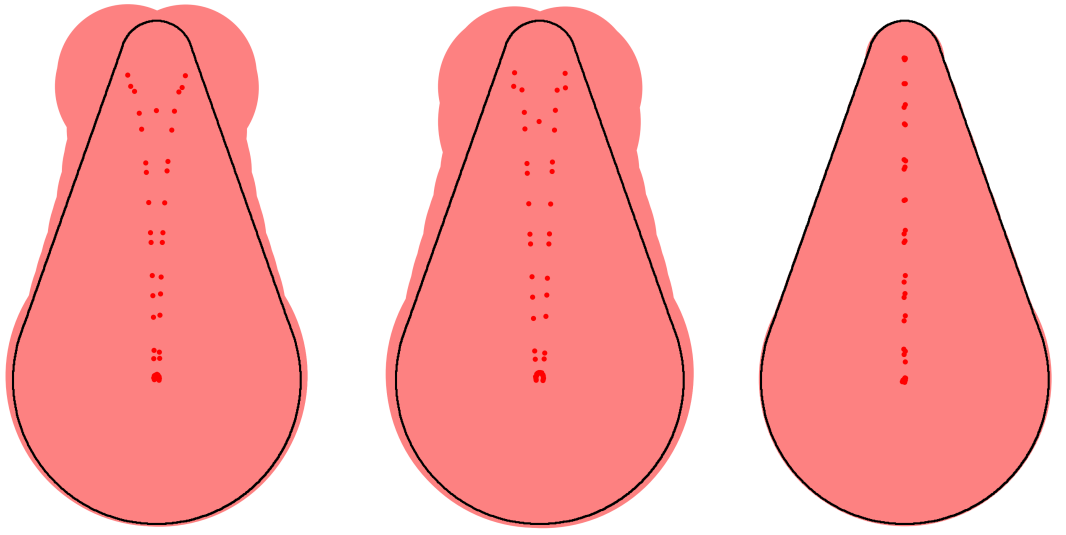

Figure 4.3: Qualitative evaluation of maximality energy variants.

dependent on its size. That is, small spheres will be associated with a higher energy level. For example, using the first formulation, as $r \to 0$ its gradient will tend to $-\infty$; as illustrated in Figure 4.3, this can cause spheres in the neighborhood of small features to bulge out. Our solution, detailed in Equation 3.2, does not encounter this problem, as our energy is *constant* regardless of the sphere size.

4.3 Quantitative Evaluation Metric

We consider ground truth geometry polluted by noise, and evaluate the quality of an extracted axis by computing the distance of each medial center to the closest point on

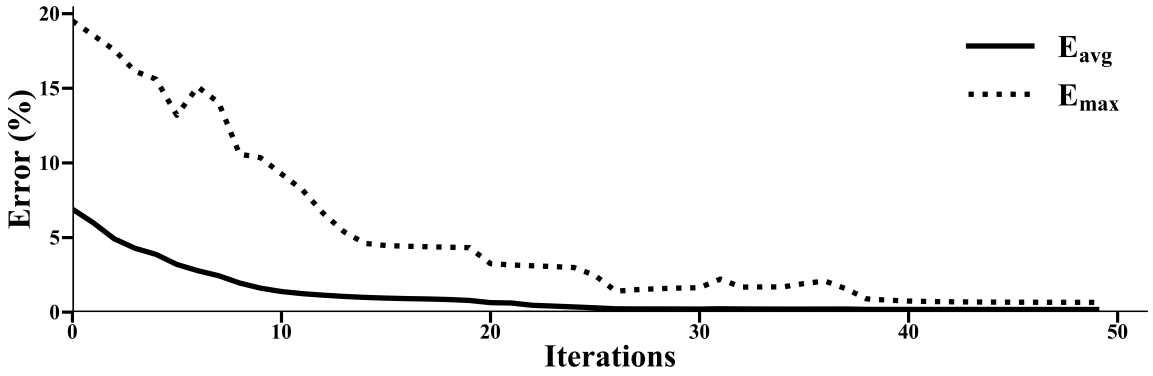

Figure 4.4: As the optimization of Equation 3.15 is executed, the average/max ground truth errors converge to the maximum precision. This plot illustrates the error for the iterations visualized in Figure 3.4.

the ground truth axis $\tilde{M}$:

$$E_{\text{avg}} = \frac{1}{N} \sum_{n} \underset{\tilde{\mathbf{c}}_n \in \tilde{\mathcal{M}}}{\arg\min} \|\mathbf{c}_n - \tilde{\mathbf{c}}_n\|_2 \tag{4.1}$$

$$E_{\max} = \max_{n} \underset{\tilde{\mathbf{c}}_n \in \tilde{\mathcal{M}}}{\arg\min} \|\mathbf{c}_n - \tilde{\mathbf{c}}_n\|_2 \tag{4.2}$$

We compute $\tilde{M}$ via the medial axis module of the python skimage package, derived from the ridges of the distance transform from the ground truth boundary $\partial\mathcal{O}$. As we discretize images with a bounding box at a 1024^2 resolution, the "numerical precision" of the metrics above is of one pixel. To obtain scale invariance, we report these errors in relation to the diagonal of the bounding box. In Figure 4.4, we visualize the convergence of our iterative optimization scheme.

4.4 State-of-the-art Comparisons in 2D

Through the metric from Section 4.3, we quantitatively evaluate the performance of LSMAT against local filtering methods, as well as the global scale axis transform.

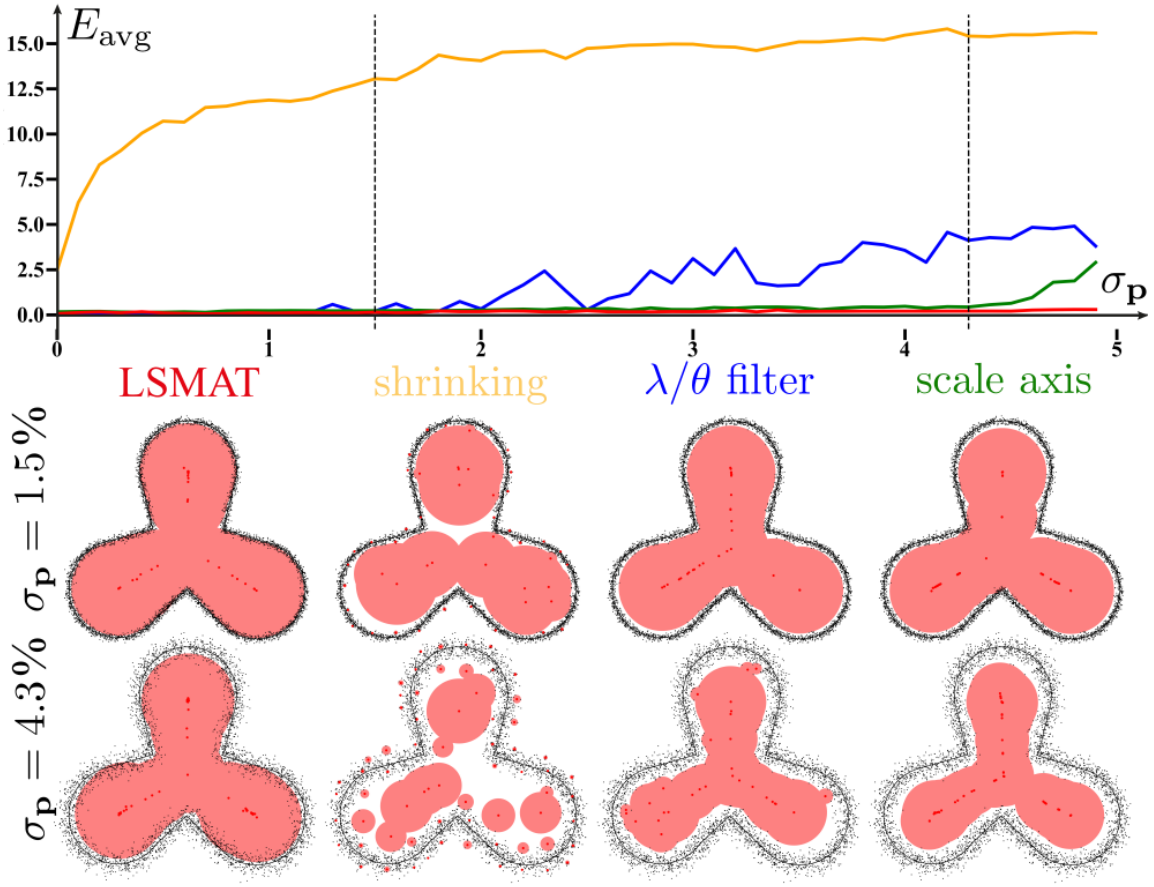

Figure 4.5: (top) Quantitative evaluation on state-of-the-art methods showing increasing error as we increase the noise level. (bottom) Qualitative results corresponding to the two dashed lines in the plot.

Figure 2.1 shows how neither distance nor angle filtering is very effective; hence we employ a compound variant where we first filter by distance with $\lambda > \sigma_p$, and then by angle $\theta > 110°$. For the scale axis, we set $\sigma = 1.3$, and for our method, we use our default parameters. We gradually increment the level of noise, and plot the corresponding error metric in Figure 4.5, as well as a few example frames from the plot. Note that LSMAT correctly captures the shape of the ground truth boundary, whereas all other local methods fail.

4.5 State-of-the-art Comparisons in 3D

We qualitatively evaluate the performance of LSMAT in three dimensions by comparing our results to those generated by the SAT [Miklos et al., 2010], QMAT [Li et al., 2015], and sphere-shrinking [Ma et al., 2012] methods. To compare with methods that expect a mesh in input, we finely re-triangulate the surface, and apply normal displacement perturbation with $\sigma_{\mathbf{p}} \in [0\%, 2\%]$ relative to the diagonal bounding box. To ensure that the same noise characteristics apply to all methods, the surface normals sampled from these perturbed meshes are used for the 3D point clouds given to LSMAT. On the *fertility* model, LSMAT produces a convincing medial axis representation even when the input oriented point cloud is affected by extreme noise – i.e. with a magnitude close to the one of the *local feature size*. For the *vase* model, LSMAT still produces a smoother and more faithful medial axis representation than the existing state-of-the-art. Even in the presence of minor amounts of noise (second row), LSMAT faithfully produces a medial axis representation with a smooth surface, and whose skeleton resembles that of the uncorrupted model. The results are particularly encouraging on the horse dataset, where we attempted to use a very small number of target primitives for QMAT (≈ 500) and a large value of scale for SAT (≈ 1.3). Our formulation is the

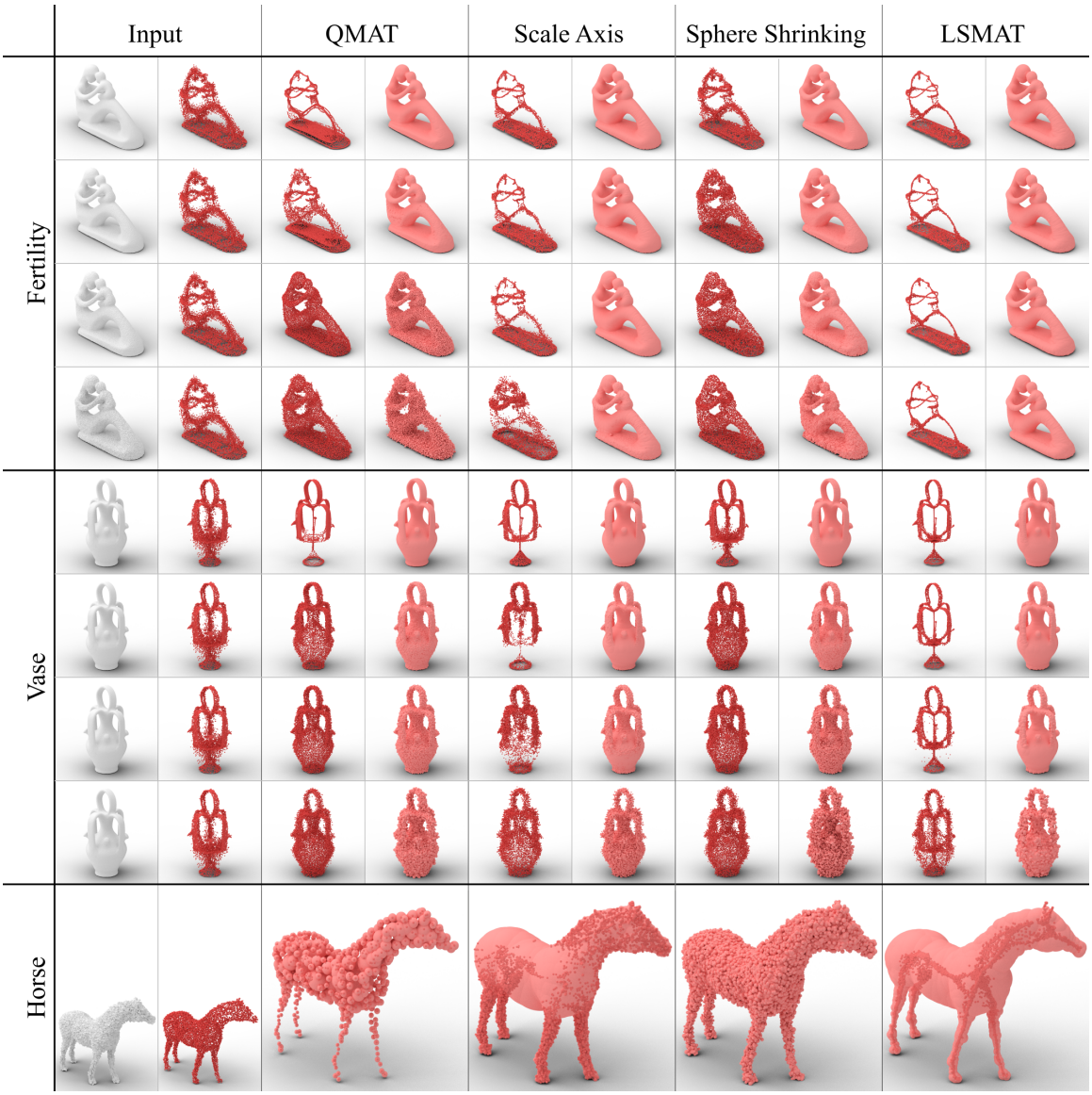

Figure 4.6: Qualitative evaluation on state-of-the-art methods as we increase the noise level. The QMAT [Li et al., 2015] and SAT [Miklos et al., 2010] in the first two columns are global methods that assume a *watertight* surface, while the sphere-shrinking [Ma et al., 2012] and the LSMAT proposed here are *local* methods. The medial spheres in the input column are unfiltered MAT results shown for comparison.

Model ($\sigma_\mathbf{p}$)	**LSMAT**		**Sphere Shrinking**	
10k Spheres	Time	Iterations	Time	Iterations
Fertility (0%)	11.8s	40	0.30s	9
Fertility (1%)	11.6s	40	0.35s	11
Fertility (2%)	12.0s	40	0.33s	10
Fertility (5%)	16.2s	40	0.34s	10
Vase (0%)	11.2s	70	0.16s	9
Vase (1%)	12.1s	70	0.17s	10
Vase (2%)	11.8s	70	0.16s	9
Vase (5%)	6.1s	70	0.16s	9
Horse (2%)	6.8s	60	0.16s	9

Table 4.1: Run-times for LSMAT and Sphere Shrinking results presented in Figure 4.6. Both algorithms are GPU accelerated and run on the same hardware. This is intended as a differential comparison, and does not represent a real-time use case.

only one capable of computing a relatively noiseless arrangement of medial spheres. The timings for our LSMAT and sphere shrinking results can be found in Table 4.1. All experiments were run on a machine with an Intel Xeon E5-1650 CPU and an Nvidia GTX 1080 GPU.

4.6 Algorithm Parameter Analysis

Our algorithm depends on five parameters: the kernel sizes h_{blend} and h_{support}, the relative weight ω_1/ω_2, pinning distance d_{pin}, and radius expansion constant ϵ. The effect of varying the first four given different levels of noise is shown in Figure 4.9.

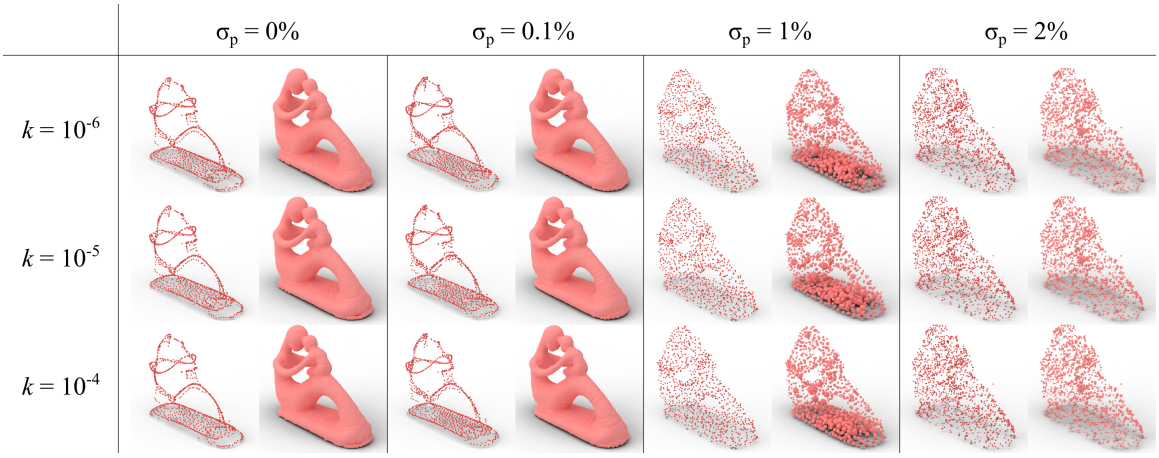

Figure 4.7: Parameter sweep for QMAT. The X axis represents noise, while the Y axis shows results for different values of the parameter k. We found that k had little effect on the noise tolerance of the algorithm. For the left two columns, the noise values are the minimum and maximum noise levels shown in the original publication, which yield good results. However, for the higher noise values that we test against QMAT fails to produce a useful medial representation. As noted in Section 2.1.2, this is a known and acknowledged limitation for this family of methods.

Parameter	Value
ω_1/ω_2	$0.007\sigma_{\mathbf{p}} + 0.02$
h_{blend}	$0.74\sigma_{\mathbf{p}} + 0.49$
h_{support}	$0.74\sigma_{\mathbf{p}} + 0.49$
d_{pin}	$0.75\sigma_{\mathbf{p}}$

Table 4.2: Empirically discovered values for our algorithm parameters. $\sigma_{\mathbf{p}}$ is the standard deviation of the expected noise distribution. This value may be known as a characteristic of the data source, or may need to be found by analyzing the data.

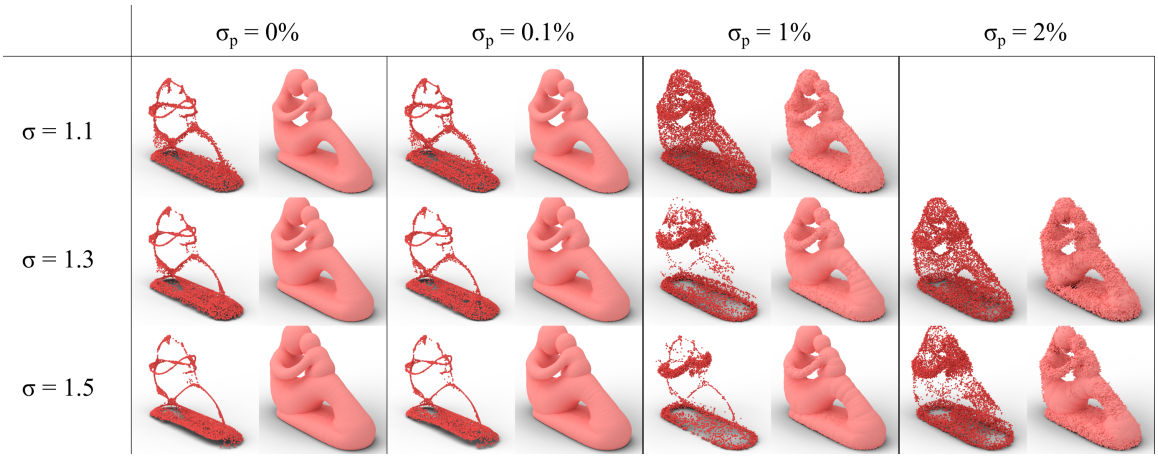

Figure 4.8: Parameter sweep for Scale Axis. The X axis shows the same noise levels used in Figure 4.7, while the Y axis shows results for different values of the scale parameter σ. As expected, this method performs well in areas where the local feature size is much larger than the noise. The Scale Axis Transform is in some cases able to recover a useful medial representation even when the unfiltered MA is highly corrupted. The missing images are due to the implementation failing to complete within the allowed time for that combination of parameters. The expected output is a thin distribution of points in the center. Many points near the surface indicate a failure to overcome input noise.

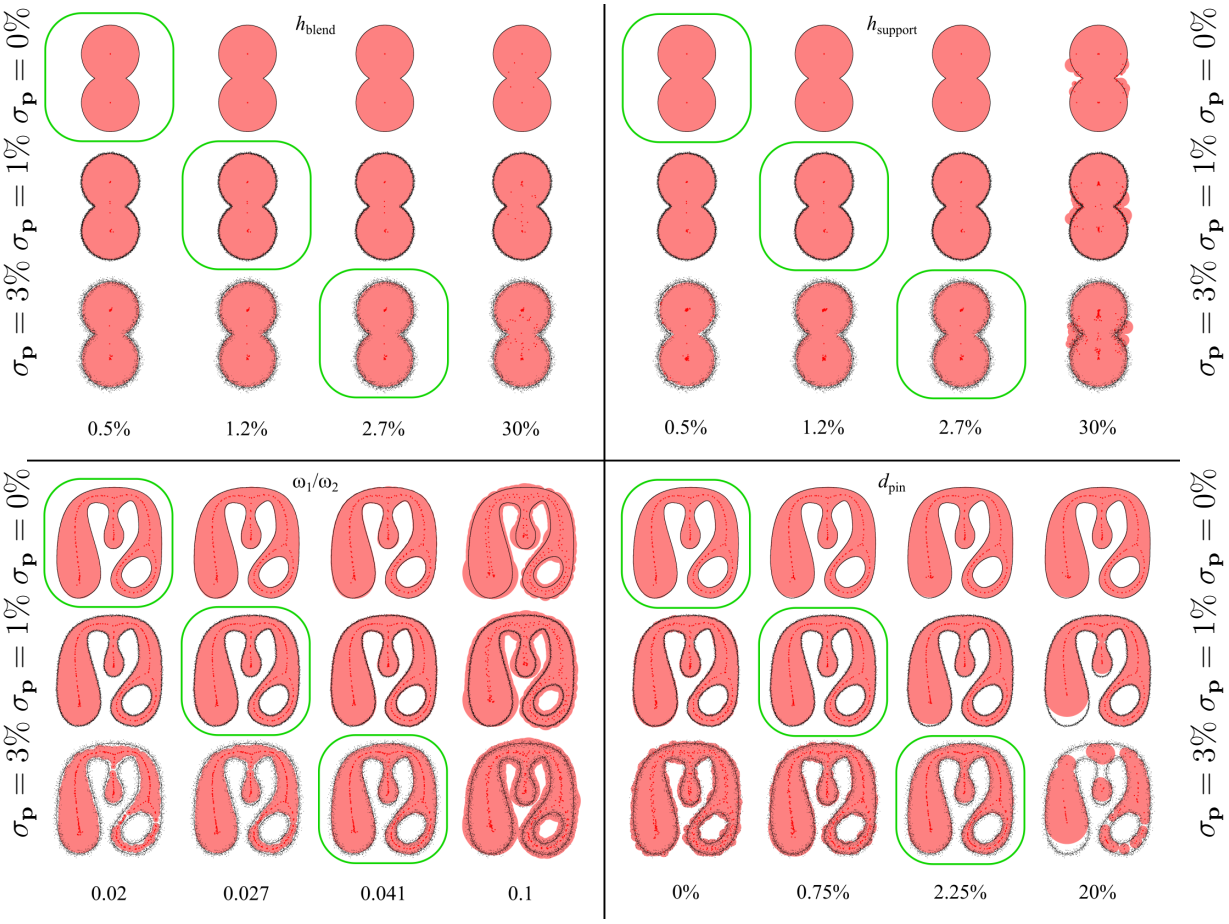

Figure 4.9: A qualitative analysis of parameters in our algorithm. Each quadrant shows the result of sweeping a parameter (horizontal) for different noise values (vertical). The highlighted images represent the "default" parameter choice as defined in Section 4.6. The right column in each quadrant shows an unreasonably high choice to demonstrate that there is an upper bound for each parameter.

Note we only consider the ratio between the ω_*, as the two energies balance each other. As h_{blend} increases, spheres near sharp concave corners begin to shrink as they eventually use the point-to-plane distance for all points in their neighborhood. When h_{support} grows significantly beyond the local feature size, the MLS formulation is no longer able to recover a meaningful surface. As ω_1/ω_2 increases and the relative importance of maximality versus inscription increases, spheres expand until they no longer form a faithful medial representation and ultimately escape the point cloud. Finally, as d_{pin} grows, the spheres are allowed to slide further from their starting positions towards areas of locally maximal radius. We assume that estimates of the input noise characteristics are available, and choose our default value for each of these four parameters based on empirically derived linear functions of $\sigma_{\mathbf{p}}$; see Table 4.2 for the values. Through experiments we observed that it is sufficient to choose a constant value for the fifth parameter ϵ, as different values of it merely change the optimal relation between ω_1/ω_2 and $\sigma_{\mathbf{p}}$. For all our experiments we use $\epsilon = 100\%$. We believe improved formulations of LSMAT could eventually coalesce some of these parameters hence improving the ease of use of our algorithm.

Chapter 5

Future works

Expressing the Medial Axis Transform as a non-linear least squares problem opens up several interesting avenues for future research.

5.1 Robustness to Outliers

Least squares problems can be interpreted as a maximum-likelihood (ML) estimation given a *Gaussian* probability distribution of the noise variables. If the input is corrupted by other forms of noise, one could replace *Gaussian* with other error

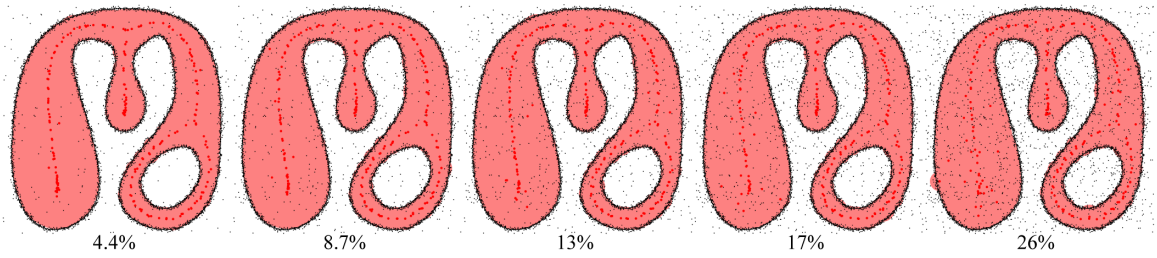

Figure 5.1: Iteratively Reweighted LSMAT and its ability to cope with an increasing number of outliers (as % of input point set).

distributions, and derive the corresponding ML optimization scheme. For example, if we assume the probability distribution of the noise to be Laplacian, the least-squares problems would simply be transformed into an ℓ^1 (i.e. least norm) optimization. However, these type of problems can still be computed with Gauss-Newton type methods by using iteratively re-weighted least squares (IRLS) techniques [Daubechies et al., 2010]. As illustrated in Figure 5.1 this results in an IR-LSMAT algorithm that can cope with significant amounts of outliers. While these results are promising, the convergence speed of the optimization is severely reduced, as a much smaller value for ε was needed to produce these results. The generalizability of LSMAT to ℓ^p robust norms [Bouaziz et al., 2013] is particularly interesting. More specifically, consider the optimization problem:

$$\arg\min_{\mathbf{c},r} \sum_n |\|\mathbf{c} - \mathbf{p}_n\| - r|^p. \tag{5.1}$$

For $p = 2$ the problem is convex, hence generating a single solution regardless of initialization. However, as $p \to 0$ the problem is non-convex and the local minima reached by optimization depends on the initialization. Our preliminary investigation revealed how these local minima correspond to spheres belonging to the *symmetry set*, a superset of the medial axis; see Figure 1.2(d) and [Tagliasacchi et al., 2016, Sec. 2.1.4]. How to exploit (5.1) to efficiently compute the MAT of a point set is an interesting venue for future works.

5.2 Smoothness Priors

Our pinning formulation is highly efficient, but its local nature can be also considered a intrinsic limitation. For example, in the noisy *maple leaf* example in Figure 4.1, at times the estimated medial spheres could be over and/or under-estimated in size,

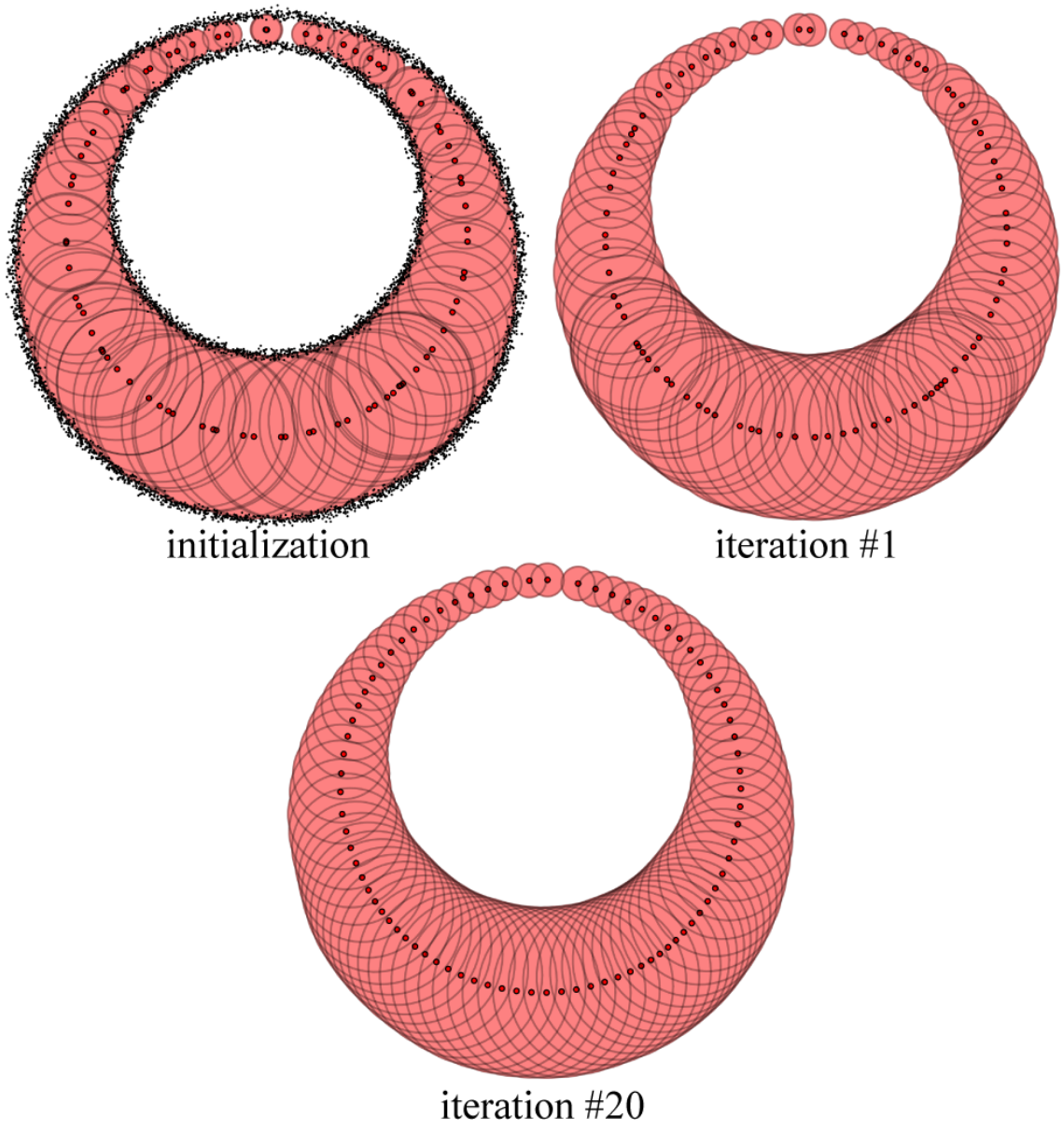

Figure 5.2: Optimizing LSMAT centers placement. To highlight the smoothness of the resulting shape, we only display the input point set overlaid to the initialization.

resulting in medial centers that do not necessarily sample the underlying piecewise-smooth manifold of the MAT. However, if our input is a sampling of a smooth surface $\partial\mathcal{O}$, then we know that [Siddiqi and Pizer, 2008]: ① the medial centers $\mathbf{c}_*$ should be lying on a piecewise smooth manifold, and ② the sphere radius function on this manifold should vary smoothly. Another desirable characteristic might be to have a uniform sampling of this manifold. We can convert these priors into least-squares energies, resulting in a *maximum a-posteriori* optimization. In Figure 5.2 we illustrate a few iterations of this optimization on the *moon* shape, where the "pinning" constraints from Section 3.3 have been disabled. While the optimization behaves as expected, this suffers similar shortcomings to those illustrated in Figure 3.3, and would result in a single sphere if executed for $t \to \infty$. Modifying the variational LSMAT formulation to obtain a regularly sampled distribution of medial centers is an interesting venue for future works.

5.3 Optimization Acceleration

In our experiments, we initialize the optimization with random sphere positions and radii. Nonetheless, given how a smooth object is composed of smooth piecewise manifolds $\mathcal{M}$, and a smooth radius function $\mathcal{R}$ defined thereon, one could easily envision a locally bootstrapped version of the algorithm, where unsolved medial balls are initialized with the $\mathbf{c}_*, r_*$ of their neighbors – which could also be re-interpreted as a *multi-scale* solver. Notice that techniques such as the sphere-shrinking algorithm from [Ma et al., 2012] do not permit this type of acceleration.

5.4 Shape Approximation vs. Reconstruction

A number of methods leverage the medial axis for *interpolatory* reconstruction [Dey, 2006], and our work is a stepping stone towards the creation of *approximating* reconstruction [Berger et al., 2016] algorithms based on the medial axis. Methods such as Medial Meshes [Sun et al., 2015] and QMAT [Li et al., 2015] assume a reconstruction of the watertight surface is *already* available, and attempt to extract its *approximation* via swept-spheres. Conversely, our work could be extended to directly compute a *reconstruction* of the input point cloud by minimizing data fitting metrics based on Hausdorff distances [Sun et al., 2015] or spherical quadrics [Li et al., 2015]. A "topology surgery" step could then be interweaved with our optimization to stitch medial spheres together and create a medial mesh.

5.5 Orienting a Point Set

Finally, while our formulation is based on an oriented point set and an approximation of its SDF in the near/far field, an interesting variant of our algorithm could consider an unoriented point set, where the quantities to be optimized for would be the radii $[r_1, r_2]$, and contact plane $[\mathbf{k}, \mathbf{n}]$ of a pair of *twin spheres*. The contact point $\mathbf{t}$ should then be optimized on the manifold $\partial\mathcal{O}$, while the complementary outside/inside label of each sphere be optimized to result in a *smooth signing* of the environment space, analogous to recent efforts in variational reconstruction of non-oriented point clouds [Mullen et al., 2010].

Chapter 6

Conclusions

We have introduced the Least Squares Medial Axis Transform, or LSMAT, a method of approximating the computation of the medial axis for a set of oriented points. Our method addresses some serious shortcomings of other available methods:

1. Lack of robustness to noise
2. Instability of output
3. Reliance on global shape information

We address points 1 and 2 by formulating our method as a least squares optimization. By treating the contribution of each surface point as a single residual in our objective function, we can perform an optimization for the medial axis which will converge to a result where the surface is at the mean of the surface point distribution. This mean will naturally remain stable under perturbation of individual points.
We also address point 3 by defining our optimization energy only in terms of the local neighbourhood of a sphere. By doing so we gain efficiency and the ability to operate on data without necessarily requiring the data for an entire shape, which might be

important in some applications like geospatial data analysis where datasets may be very large.

Our results and comparisons show that we have achieved our goal of being able to process noise-corrupted point cloud data and recover a stable, meaningful medial representation.

Bibliography

[Amenta and Bern, 1999] Amenta, N. and Bern, M. (1999). Surface reconstruction by voronoi filtering. *Discrete & Computational Geometry*.

[Amenta et al., 2001] Amenta, N., Choi, S., and Kolluri, R. K. (2001). The power crust. In *Proceedings of the ACM Symposium on Solid Modeling and Applications*.

[Angles et al., 2019] Angles, B., Rebain, D., Macklin, M., Wyvill, B., Barthe, L., Lewis, J., von der Pahlen, J., Izadi, S., Valentin, J., Bouaziz, S., et al. (2019). Viper: Volume invariant position-based elastic rods. *arXiv preprint arXiv:1906.05260*.

[Attali and Montanvert, 1997] Attali, D. and Montanvert, A. (1997). Computing and simplifying 2d and 3d continuous skeletons. *Computer vision and image understanding*.

[Baran and Popović, 2007] Baran, I. and Popović, J. (2007). Automatic rigging and animation of 3d characters. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*.

[Berger et al., 2016] Berger, M., Tagliasacchi, A., Seversky, L., Alliez, P., Guennebaud, G., Levine, J., Sharf, A., and Silva, C. (2016). A survey of surface reconstruction from point clouds. *Proc. Eurographics (State of the Art Reports)*.

[Blum, 1967] Blum, H. (1967). *A transformation for extracting new descriptors of shape*.

[Bouaziz et al., 2013] Bouaziz, S., Tagliasacchi, A., and Pauly, M. (2013). Sparse iterative closest point. *Computer Graphics Forum (Proc. of SGP)*.

[Brandt and Algazi, 1992] Brandt, J. W. and Algazi, V. (1992). Continuous skeleton computation by voronoi diagram. *CVGIP: Image Understanding*.

[Chazal and Lieutier, 2005] Chazal, F. and Lieutier, A. (2005). The λ-medial axis. *Graphical Models*.

[Daubechies et al., 2010] Daubechies, I., DeVore, R., Fornasier, M., and Güntürk, C. S. (2010). Iteratively reweighted least squares minimization for sparse recovery. *Communications on pure and applied mathematics*.

[Dey, 2006] Dey, T. K. (2006). *Curve and surface reconstruction: algorithms with mathematical analysis*.

[Ding et al., 2016] Ding, D., Pan, Z., Cuiuri, D., Li, H., and Larkin, N. (2016). Adaptive path planning for wire-feed additive manufacturing using medial axis transformation. *Journal of Cleaner Production*, 133:942–952.

[Foskey et al., 2003] Foskey, M., Lin, M. C., and Manocha, D. (2003). Efficient computation of a simplified medial axis. In *Proceedings of the ACM Symposium on Solid Modeling and Applications*.

[Friedman et al., 1977] Friedman, J. H., Bentley, J. L., and Finkel, R. A. (1977). An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software (TOMS)*, 3(3):209–226.

[Garland and Heckbert, 1997] Garland, M. and Heckbert, P. S. (1997). Surface simplification using quadric error metrics. In *Proc. of ACM SIGGRAPH*.

[Giesen et al., 2009] Giesen, J., Miklos, B., Pauly, M., and Wormser, C. (2009). The scale axis transform. In *Proceedings of the Twenty-fifth Annual Symposium on Computational Geometry*.

[Guennebaud and Gross, 2007] Guennebaud, G. and Gross, M. (2007). Algebraic point set surfaces. In *ACM Transactions on Graphics (TOG)*.

[He et al., 2015] He, S., Choi, Y.-K., Guo, Y., Guo, X., and Wang, W. (2015). A 3d shape descriptor based on spectral analysis of medial axis. *Computer Aided Geometric Design*, 39:50–66.

[Jalba et al., 2013] Jalba, A. C., Kustra, J., and Telea, A. C. (2013). Surface and curve skeletonization of large 3d models on the gpu. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

[Ke et al., 2017] Ke, W., Chen, J., Jiao, J., Zhao, G., and Ye, Q. (2017). Srn: side-output residual network for object symmetry detection in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1068–1076.

[Kolluri, 2008] Kolluri, R. (2008). Provably good moving least squares. *ACM Transactions on Algorithms (TALG)*.

[Lan et al., 2017] Lan, L., Yao, J., Huang, P., and Guo, X. (2017). Medial-axis-driven shape deformation with volume preservation. *The Visual Computer*, 33(6-8):789–800.

[Li et al., 2015] Li, P., Wang, B., Sun, F., Guo, X., Zhang, C., and Wang, W. (2015). Q-mat : computing medial axis transform by quadratic error minimization. *ACM Trans. Graph.*

[Lipman et al., 2007] Lipman, Y., Cohen-Or, D., Levin, D., and Tal-Ezer, H. (2007). Parameterization-free projection for geometry reconstruction. *ACM Trans. on Graphics (Proc. of SIGGRAPH).*

[Liu et al., 2017] Liu, C., Ke, W., Jiao, J., and Ye, Q. (2017). Rsrn: Rich side-output residual network for medial axis detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1739–1743.

[Liu et al., 2018] Liu, C., Ke, W., Qin, F., and Ye, Q. (2018). Linear span network for object skeleton detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 133–148.

[Ma et al., 2012] Ma, J., Bae, S. W., and Choi, S. (2012). 3d medial axis point approximation using nearest neighbors and the normal field. *The Visual Computer.*

[Miklos et al., 2010] Miklos, B., Giesen, J., and Pauly, M. (2010). Discrete scale axis representations for 3d geometry. *ACM Trans. Graph.*

[Mitra et al., 2004] Mitra, N. J., Gelfand, N., Pottmann, H., and Guibas, L. (2004). Registration of point cloud data from a geometric optimization perspective. In *Proc. of SGP.*

[Mullen et al., 2010] Mullen, P., De Goes, F., Desbrun, M., Cohen-Steiner, D., and Alliez, P. (2010). Signing the unsigned: Robust surface reconstruction from raw pointsets. *Computer Graphics Forum (Proc. of SGP).*

[Musialski et al., 2016] Musialski, P., Hafner, C., Rist, F., Birsak, M., Wimmer, M., and Kobbelt, L. (2016). Non-linear shape optimization using local subspace projections. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*.

[Nocedal and Wright, 2006] Nocedal, J. and Wright, S. J. (2006). *Sequential quadratic programming*. Springer.

[Pan et al., 2019] Pan, Y., Wang, B., Guo, X., Zeng, H., Ma, Y., and Wang, W. (2019). Q-mat+: An error-controllable and feature-sensitive simplification algorithm for medial axis transform. *Computer Aided Geometric Design*, 71:16–29.

[Panichev and Voloshyna, 2019] Panichev, O. and Voloshyna, A. (2019). U-net based convolutional neural network for skeleton extraction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0.

[Peters et al., 2015] Peters, R., Ledoux, H., and Biljecki, F. (2015). Visibility analysis in a point cloud based on the medial axis transform. In *UDMV*, pages 7–12.

[Pottmann and Hofer, 2003] Pottmann, H. and Hofer, M. (2003). Geometry of the squared distance function to curves and surfaces. In *Visualization and mathematics III*, pages 221–242. Springer.

[Quadros, 2016] Quadros, W. R. (2016). Laytracks3d: A new approach for meshing general solids using medial axis transform. *Computer-Aided Design*, 72:102–117.

[Saha et al., 2016] Saha, P. K., Borgefors, G., and di Baja, G. S. (2016). A survey on skeletonization algorithms and their applications. *Pattern Recognition Letters*.

[Shapira et al., 2008] Shapira, L., Shamir, A., and Cohen-Or, D. (2008). Consistent mesh partitioning and skeletonisation using the shape diameter function. *The Visual Computer*.

[Siddiqi and Pizer, 2008] Siddiqi, K. and Pizer, S. (2008). *Medial representations: mathematics, algorithms and applications*.

[Sun et al., 2015] Sun, F., Choi, Y.-K., Yu, Y., and Wang, W. (2015). Medial meshes–a compact and accurate representation of medial axis transform. *IEEE transactions on visualization and computer graphics*, 22(3):1278–1290.

[Tagliasacchi et al., 2016] Tagliasacchi, A., Delame, T., Spagnuolo, M., Amenta, N., and Telea, A. (2016). 3d skeletons: a state-of-the-art report. *Proc. Eurographics (State of the Art Reports)*.

[Thiery et al., 2013] Thiery, J.-M., Guy, E., and Boubekeur, T. (2013). Sphere-meshes: Shape approximation using spherical quadric error metrics. *ACM Trans. Graph.*

[Thiery et al., 2016] Thiery, J.-M., Guy, E., Boubekeur, T., and Eisemann, E. (2016). Animated mesh approximation with sphere-meshes. *ACM Trans. Graph.*

[Tkach et al., 2016] Tkach, A., Pauly, M., and Tagliasacchi, A. (2016). Sphere-meshes for real-time hand modeling and tracking. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*.

[Tsogkas and Dickinson, 2017] Tsogkas, S. and Dickinson, S. (2017). Amat: Medial axis transform for natural images. In *2017 IEEE International Conference on Computer Vision (ICCV)*.

[Yang et al., 2018] Yang, B., Yao, J., and Guo, X. (2018). Dmat: Deformable medial axis transform for animated mesh approximation. In *Computer Graphics Forum*, volume 37, pages 301–311. Wiley Online Library.

[Zhang et al., 2015] Zhang, X., Xia, Y., Wang, J., Yang, Z., Tu, C., and Wang, W. (2015). Medial axis treean internal supporting structure for 3d printing. *Computer Aided Geometric Design*, 35:149–162.

[Zhong and Chen, 2018] Zhong, Y. and Chen, F. (2018). Computing medial axis transformations of 2d point clouds. *Graphical Models*, 97:50–63.