

# Proximity-Based Point Cloud Reconstruction

DISSERTATION

zur Erlangung des akademischen Grades

**Doktorin der Technischen Wissenschaften**

eingereicht von

**Diana Marin, BSc., MEng.**

Matrikelnummer 12130539

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.Prof. Dipl.-Ing. Dipl.-Ing. Dr.techn. Michael Wimmer

Zweitbetreuung: Projektass. Mag.rer.soc.oec. Stefan Ohrhallinger, PhD

Diese Dissertation haben begutachtet:

---

Prof. Dr. Marc Alexa

---

Prof. Dr. Leif Kobbelt

Wien, 2. September 2024

---

Diana Marin





# Proximity-Based Point Cloud Reconstruction

DISSERTATION

submitted in partial fulfillment of the requirements for the degree of

**Doktorin der Technischen Wissenschaften**

by

**Diana Marin, BSc., MEng.**

Registration Number 12130539

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.Prof. Dipl.-Ing. Dipl.-Ing. Dr.techn. Michael Wimmer

Second advisor: Projektass. Mag.rer.soc.oec. Stefan Ohrhallinger, PhD

The dissertation has been reviewed by:

---

Prof. Dr. Marc Alexa

---

Prof. Dr. Leif Kobbelt

Vienna, September 2, 2024

---

Diana Marin



# Erklärung zur Verfassung der Arbeit

Diana Marin, BSc., MEng.

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Ich erkläre weiters, dass ich mich generativer KI-Tools lediglich als Hilfsmittel bedient habe und in der vorliegenden Arbeit mein gestalterischer Einfluss überwiegt. Im Anhang „Übersicht verwendeter Hilfsmittel“ habe ich alle generativen KI-Tools gelistet, die verwendet wurden, und angegeben, wo und wie sie verwendet wurden. Für Textpassagen, die ohne substantielle Änderungen übernommen wurden, haben ich jeweils die von mir formulierten Eingaben (Prompts) und die verwendete IT- Anwendung mit ihrem Produktnamen und Versionsnummer/Datum angegeben.

Wien, 2. September 2024

---

Diana Marin



# Acknowledgements

I would like to thank all the people who have helped in the process of this thesis and the doctoral degree, starting with my supervisor, Michael Wimmer, for the discussions and for offering me the big-picture look regarding my research. I am grateful for the opportunity of studying here and being able to take advantage of all the possibilities that arose during my degree. I wish to thank my second supervisor, Stefan Ohrhallinger, for the plenty of discussions that ensured our research was going forward. I am grateful for the encouragement, and the look-ahead view that motivated and helped me throughout these years. Thank you for your understanding and for indulging and supporting all of my ideas, requests, travels and conference attendances. I wish to thank my previous supervisor as well, Hamish Carr, for extending his support and advice beyond my undergraduate degrees, and helping me build my academic career.

I want to thank all of my collaborators with whom all projects have been a pleasure to work on, and in particular, I would like to thank my honorary supervisor Pooran Memari, for her infinite support and amazing collaboration, both while visiting École Polytechnique and afterward for our shared publication. I would also like to thank Filippo Maggioli, with whom the curves on surfaces collaboration was a shared and enjoyable effort, leading to a successful outcome.

I wish to extend another round of thanks to all my TU Wien colleagues, for the discussions and advice. In particular, I wish to thank Henry, Daniel and Lukas for always making the coffee breaks enjoyable and the discussions fruitful. I want to thank Annalena as well, my fellow volunteering partner and conference traveler, with whom I have shared responsibilities, organizer duties, presentations, and, of course, sweets.

I am grateful to my family for always supporting my academic endeavors and lovingly encouraging me from afar - Mami, Tati, Nenicu and Mamaia.

Finally, I want to express my deep gratitude to my boyfriend, Filip, for his unwavering support during every deadline, paper revision, and submission. You stood by me through all the stress and late nights, and celebrated both the small and big victories with me. I'm so thankful to have you by my side.



# Kurzfassung

Das Extrapolieren von Informationen aus unvollständigen Daten ist eine zentrale menschliche Fähigkeit, die es uns ermöglicht, Muster zu erkennen und Vorhersagen auf der Grundlage begrenzter Beobachtungen zu treffen. Ein anschauliches Beispiel ist unsere Fähigkeit, kohärente Formen aus scheinbar zufälligen Punktmengen wahrzunehmen. Wenn es keine vordefinierten Regeln gibt, wird die Rekonstruktion von Daten jedoch erschwert, da unklar ist, wie Punkte miteinander verbunden werden sollen oder welche Muster abgeleitet werden können. In der Computergrafik ist ein zentrales Ziel, diese menschliche Fähigkeit zu replizieren, indem Algorithmen entwickelt werden, die in der Lage sind, ursprüngliche Strukturen präzise zu rekonstruieren oder bedeutungsvolle Informationen aus Rohdaten zu extrahieren.

Diese Arbeit befasst sich mit der Rekonstruktion von Punktwolken unter Verwendung von Methoden, die auf Nähe basieren, wobei ein besonderer Schwerpunkt auf dem Einflussphären-Graphen (SIG) liegt. Wir erörtern die Kurvenrekonstruktion, bei der wir das Verbinden von Punkten zur Erstellung von Konturen automatisieren und dabei theoretische Garantien für unsere Methode bieten. Im Vergleich zu ähnlichen Methoden zur Rekonstruktion von Mannigfaltigkeitskurven erzielen wir die besten Ergebnisse. Wir erweitern unsere Kurvenrekonstruktion auf Mannigfaltigkeiten und überwinden die Herausforderungen, die mit dem Wechsel in unterschiedliche Domänen verbunden sind, wobei wir unsere theoretischen Garantien ausbauen. Wir sind in der Lage, Kurven aus spärlicheren Eingaben zu rekonstruieren als der aktuelle Stand der Technik und untersuchen verschiedene Szenarien, in denen diese Kurven existieren können. Wir analysieren die Eigenschaften des SIG als parameterfreie Nähecodierungsstruktur für dreidimensionale Punktwolken. Dabei führen wir neue räumliche Grenzen für die SIG-Nachbarn ein. Wir untersuchen, wie nah die Codierung an der tatsächlichen Oberfläche liegt, verglichen mit den häufig verwendeten  $k$ NN-Graphen, und bewerten unsere Leistung im Kontext der Normalenschätzung als Anwendung. Abschließend führen wir SING ein – einen Stabilität-eingeschlossenen Nachbarschaftsgraphen, der als nützliches Werkzeug für verschiedene Anwendungen, wie etwa das Clustering, dient und einen soliden theoretischen Hintergrund in der topologischen Datenanalyse aufweist.





# Abstract

Extrapolating information from incomplete data is a key human skill, enabling us to infer patterns and make predictions from limited observations. A prime example is our ability to perceive coherent shapes from seemingly random point sets, a key aspect of cognition. However, data reconstruction becomes challenging when no predefined rules exist, as it is unclear how to connect the data or infer patterns. In computer graphics, a major goal is to replicate this human ability by developing algorithms that can accurately reconstruct original structures or extract meaningful information from raw, disconnected data.

The contributions of this thesis deal with point cloud reconstruction, leveraging proximity-based methods, with a particular focus on a specific proximity-encoding data structure - the spheres-of-influence graph (SIG). We discuss curve reconstruction, where we automate the game of connecting the dots to create contours, providing theoretical guarantees for our method. We obtain the best results compared to similar methods for manifold curves. We extend our curve reconstruction to manifolds, overcoming the challenges of moving to different domains, and extending our theoretical guarantees. We are able to reconstruct curves from sparser inputs compared to the state-of-the-art, and we explore various settings in which these curves can live. We investigate the properties of the SIG as a parameter-free proximity encoding structure of three-dimensional point clouds. We introduce new spatial bounds for the SIG neighbors as a theoretical contribution. We analyze how close the encoding is to the ground truth surface compared to the commonly used  $k$ NN graphs, and we evaluate our performance in the context of normal estimation as an application. Lastly, we introduce SING – a stability-incorporated neighborhood graph, a useful tool with various applications, such as clustering, and with a strong theoretical background in topological data analysis.



# Contents

<b>Kurzfassung</b>	ix
<b>Abstract</b>	xi
<b>Contents</b>	xiii
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	2
1.2 Problem Statement . . . . .	2
1.3 Contributions . . . . .	3
1.4 Outline . . . . .	5
<b>2 Background and Related Work</b>	<b>7</b>
2.1 Point Clouds . . . . .	8
2.2 Choosing the Right Samples . . . . .	9
2.3 Connecting the Dots . . . . .	12
2.4 Plane and Simple . . . . .	19
2.5 Spatial Upgrades . . . . .	22
<b>3 SIGDT: Planar Curve Reconstruction</b>	<b>27</b>
3.1 Overview . . . . .	27
3.2 Introduction . . . . .	27
3.3 Method . . . . .	29
3.4 Results . . . . .	36
3.5 Conclusion and Future Work . . . . .	46
<b>4 SIGDV: Reconstructing Curves on Riemannian Manifolds</b>	<b>51</b>
4.1 Overview . . . . .	51
4.2 Introduction . . . . .	52
4.3 Background and notations . . . . .	53
4.4 Method . . . . .	55
4.5 Applications . . . . .	66
4.6 Conclusions . . . . .	72
	xiii

<b>5</b>	<b>SIG3D: Encoding Connectivity for Point Clouds</b>	<b>75</b>
5.1	Overview . . . . .	75
5.2	Introduction . . . . .	75
5.3	Method . . . . .	78
5.4	Results . . . . .	79
5.5	Conclusion . . . . .	90
<b>6</b>	<b>SING: Stability-Incorporated Neighborhood Graph</b>	<b>91</b>
6.1	Overview . . . . .	91
6.2	Introduction . . . . .	92
6.3	Method: proximity criterion & TDA . . . . .	93
6.4	SING Features and Advantages . . . . .	98
6.5	Results and Validation . . . . .	106
6.6	Discussion and Perspectives . . . . .	111
<b>7</b>	<b>Conclusion</b>	<b>113</b>
7.1	Summary . . . . .	113
7.2	Future directions . . . . .	114
	<b>Overview of Generative AI Tools Used</b>	<b>117</b>
	<b>List of Figures</b>	<b>119</b>
	<b>List of Tables</b>	<b>123</b>
	<b>List of Algorithms</b>	<b>125</b>
	<b>Bibliography</b>	<b>127</b>

## CHAPTER

# 1



# Introduction

As humans, we are able to extrapolate information from incomplete data, which allows us to infer patterns, fill in gaps, and make predictions based on limited observations. A simple yet powerful example is our gestalt perception of point sets; this ability to discern a coherent shape or pattern from seemingly random data points, seeing them as unified figures instead of disconnected points, is a key aspect of human cognition.

However, the problem of data reconstruction becomes significantly more difficult when there are no predefined rules or guidelines. When presented with raw data, it is not always clear how to connect the samples or what pattern to infer. As we observe in Figure 1.1, the lack of rules can easily create an undesirable or incorrect output. Therefore, one of the primary goals of reconstruction in computer science, of trying to extend human perceptual abilities to machines, is to create or infer the best possible rules or algorithms that can reliably reconstruct the original structure from the given data, or extract meaningful information from it. This involves developing methods that can automatically detect patterns, relationships, and underlying structures within the data, facilitating accurate and meaningful reconstructions.

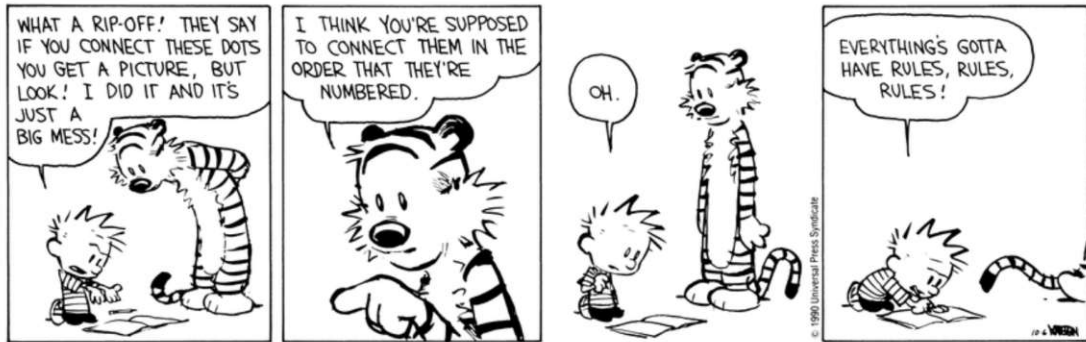


Figure 1.1: Inferring the rules from the input points alone is a tedious and complex task, which can easily result in a *big mess* [Wat90]. We aim to extract the set of rules that best fit our current analysis context, be it curve reconstruction, connectivity encoding or clustering.

## 1.1 Motivation

Data reconstruction plays a crucial role in computer graphics as it enables the conversion of raw point cloud data into meaningful and usable models. Point clouds consist of collections of discrete data points that represent the contour of a curve, the surface of an object, or define samples with similar properties, yet lack inherent structure or connectivity. The main goal of this thesis is to improve upon data reconstruction, which we define as recreating the original object or surface the point cloud represents or extracting critical information regarding the features of the data.

In point clouds, the primary way information is encoded is through the positions of the points in space, meaning that the absolute and relative positions of points carry critical information about the shape and features of the object. By analyzing these proximity relations, we can determine which connections are possible, allowing us to reconstruct curves, surfaces, or other geometric features. This approach is particularly powerful because it does not require additional information beyond the point positions themselves, making it a versatile and robust method for dealing with various types of point clouds. Hence, a second goal of this thesis is to focus on leveraging proximity in the reconstruction process by using the spatial relationships between data points to infer the underlying structure of the data being represented.

## 1.2 Problem Statement

The common denominator to all the goals this thesis tries to address is represented by extracting meaningful information from point cloud data. Our primary research question is whether proximity-based methods can be beneficial in this regard, and if so, which of

these methods yield the most effective results. We briefly define the type of input we use and the possible outputs we aim to obtain, from reconstruction to feature extraction.

**Point clouds** are collections of discrete points that represent a spatial distribution of data in two-dimensional (2D), three-dimensional (3D), or even higher-dimensional spaces, sometimes embedded on manifolds of lower dimensionality. These points are typically generated by sampling the contour or surface of objects, capturing the geometry and structure of the underlying space. In 2D, point clouds might represent contours or edges in images, while in 3D, they are often obtained from LiDAR scans or photogrammetry, defining the surface of physical objects and environments. On manifolds, point clouds can represent data distributed across complex, curved spaces, such as surfaces embedded in higher-dimensional spaces. The challenge of data reconstruction is to extract meaningful information from these raw data points—whether it is reconstructing the original shape, detecting patterns, or analyzing the spatial relationships between points.

In the case of **planar curve reconstruction**, the objective is to recover, from a set of scattered points in a two-dimensional plane, the curve they have been sampled from or a close approximation [OPP<sup>+</sup>21]. This task is similar to the *connecting the dots* game, where the goal is to find the most plausible curve that passes through or near the points.

Extending the concept to **higher dimensions**, curve reconstruction involves finding the most likely path or curve that a set of points would form in a multi-dimensional space, usually bound to a lower-dimensional manifold. This is significantly more complex than planar reconstruction due to the increased number of possible connections and patterns that can exist in higher dimensions. Moreover, not all theoretical guarantees transfer straightforwardly to higher-dimensional domains.

Spatial point clouds, which consist of large sets of points in three-dimensional space, are often used in applications like 3D modeling and geographic information systems (GIS). **Encoding connectivity** within these point clouds involves determining how points are related to one another, which is necessary for then creating coherent and accurate 3D models, or being able to estimate various properties, such as normals.

**Clustering** is the process of grouping a set of points based on their similarities or proximity to each other [EIO<sup>+</sup>22]. It is a critical step in data analysis, where the goal is to identify natural groupings or patterns within the data set. Clustering is widely used in machine learning, image processing, and many other fields, where the ability to identify and group similar data points is essential for tasks such as classification, pattern recognition, and anomaly detection.

## 1.3 Contributions

The main contributions of this thesis to the state of the art are:

### 1.3.1 Planar Curve Reconstruction

We improved the state of the art in planar curve reconstruction, obtaining better results for closed, manifold curves compared to the state-of-the-art. We also relaxed the theoretical guarantees for reconstruction, requiring fewer samples and less uniformity between consecutive samples. The initial work towards this contribution was first published and presented as a poster at Eurographics 2022 [MOW22a], and was later improved and published as a paper [MOW22b] in the Computer Graphics Forum and presented at Pacific Graphics 2022.

The idea for this paper came from multiple experiments with proximity graphs and regular discussions with Stefan Ohrhallinger, who was also the driving force behind the theoretical contribution. The implementation and most of the writing were done by the author, with help and supervision from both Stefan Ohrhallinger and Michael Wimmer.

### 1.3.2 Reconstructing Curves on Surfaces

We extended our work from the planar domain to Riemannian manifolds, overcoming the theoretical hurdles of such a change and being able to provide a theoretically sound curve reconstruction algorithm that requires fewer samples than the state-of-the-art [MMM<sup>+</sup>24]. This paper was presented at the Symposium on Geometry Processing 2024.

This paper's idea was ignited by a discussion with Filippo Maggioli, whose work is focused on extending classical concepts to new domains, after the presentation of the first paper. Development and writing were mainly a shared effort between the first two authors, helped and supervised by the rest of the paper's authors.

### 1.3.3 Connectivity Encoding of Point Clouds

A natural extension to our SIG-based ideas was to move to three-dimensional space, where we proved new distance bounds for the SIG neighbors and analyzed this graph as a proximity-encoding graph, comparing it to the classical  $k$ NN graphs. This work was initially published and presented as a poster at Eurographics 2023 [MOW23], being later transformed into a full paper [MOW24], presented at GRAPP 2024 and published in the conference proceedings. We have been invited to submit an extended version to the Communications in Computer and Information Science journal, which is currently in submission [MIOW24].

For this project, the implementation and theoretical background were done by the thesis author, while Stefan Ohrhallinger and Michael Wimmer supervised the development and offered feedback during the writing process.

### 1.3.4 Extending SIG and Clustering Applications

The idea of adding a parameter to the sphere-of-influence graph to allow for more flexibility was developed during regular discussions and brainstorming sessions with



Pooran Memari, during a three-month research stay at École Polytechnique Paris, under her supervision. Additional discussions with Amal Dev Parakkat unveiled applications of our idea to stipple art, while talks with Steve Oudot brought forth a connection to topological analysis. This collaboration resulted in a paper, published and presented as a conference paper at SIGGRAPH Asia 2024, that introduces a new proximity graph with a strong theoretical background in topological analysis and various applications in clustering [MPO<sup>+</sup>24].

The implementation was done by the thesis author, while the writing was a collaborative effort between all authors.

### 1.3.5 Other contributions

During the duration of the doctoral programme, the thesis author has also contributed to the following publications which have not been included in the thesis:

1. **Distributed Surface Reconstruction** [MKOW24] is a poster contribution, accepted and presented at Eurographics 2024, where we extend a newly introduced surface reconstruction algorithm [POEM24] to work in a distributed fashion on a local scientific cluster. The implementation was done mainly by Patrick Komon as part of his bachelor thesis, under the author's supervision, while writing was a team effort.
2. **SIGNificant Outlier Removal** [MIOW24] represents the extended version of "*Parameter-free connectivity for point clouds*" [MOW24], which is currently submitted to an issue of Springer Communications in Computer Science. We extend the spheres-of-influence graph applications in the field of outlier removal by analyzing the average incident edge length and removing points that are too different from the rest of the point cloud. We obtain results comparable to or better than other classical outlier removal methods. The implementation was done by the author, while the paper writing was a combined effort of the author and of Filip Ilic, supervised by Stefan Ohrhallinger and Michael Wimmer.
3. **Visualizing Group Structure in Compound Graphs: The Current State, Lessons Learned, and Outstanding Opportunities** [EMWR24]: we present a literature analysis of compound graphs' visualization, combine taxonomies of various studies in this field, and extract outstanding research opportunities that could further improve this sub-field. The author was part of the writing effort for this paper.

## 1.4 Outline

The thesis starts with an introduction to point sets, visiting various approaches to reconstruct the original data from which these points have been sampled in Chapter 2.

We focus on connectivity retrieval as it represents an important first step in reconstruction, and in particular, on the spheres-of-influence graph, which is one way of computing connectivity from disconnected samples. As we discover in the following chapters, this graph exhibits promising theoretical properties, which allow us to explore its usage in curve reconstruction in the planar case (Chapter 3), and in the manifold domain (Chapter 4). We analyze this graph as a parameter-free alternative to the classically used  $k$ NN graph in Chapter 5, offering new theoretical distance bounds between neighbors. We then exchange the parameter-free benefits while searching for improvements, and add a parameter to allow for more flexibility. We explore ways of choosing the *right* parameter and various applications of this new graph variant in Chapter 6. We summarize our contributions and provide a discussion of future directions in the final chapter, Chapter 7.

## CHAPTER 2

# Background and Related Work

In this section, we introduce the background information required to understand the subset of computer graphics that pertain to this thesis's contributions, starting from the input data, the problems we are trying to solve, and the various theoretical building blocks we use to reach each of the solutions.

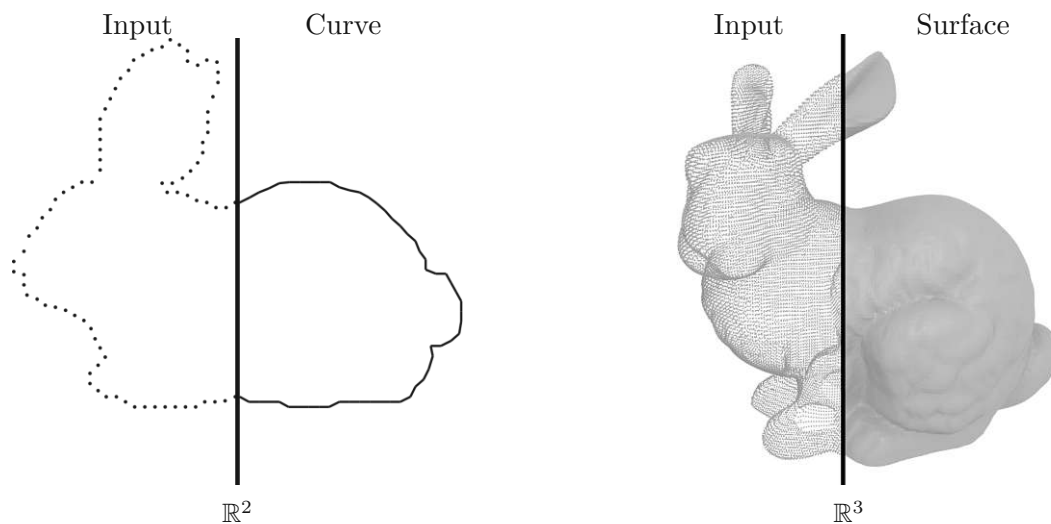


Figure 2.1: Examples of point clouds in the plane and in space, together with their corresponding reconstructions - curve and surface, respectively. Most of the research in the point cloud realm deals with these two variants, but as we observe in Section 2.5, point sets can exist in higher dimensions or under specific constraints.

## 2.1 Point Clouds

Point clouds are collections of unconnected, unstructured data usually described as coordinates in a given space. The most common variants of point sets live in two or three dimensions, and are usually used to encode information about the contour or the shape of some data - Figure 2.1. Point clouds can contain additional information besides the position, such as color (usually encoded as RGB values), confidence or intensity.

In the wild, planar point sets result from image processing, silhouette information, stipple data, or medical data, possibly obtained by extracting contour information. Three-dimensional point clouds are more common, especially due to the increase in laser scanning devices' availability, and are usually a representation of real-life objects. Each point represents a three-dimensional sample on the surface of the scanned object, subject to possible noise from the scanner. However, point samples are not only bound to these cases and can also exist in higher dimensions, possibly embedded in lower-dimensional subspaces, and represent various data, such as encoding measurements of real data at a specific position, or complex moving patterns in high dimensional spaces.

In any context where point clouds are generated, the presence of noise is inevitable. This noise may arise from line-of-sight errors, limitations in scanning devices, or inaccuracies in the generation algorithm. While various methods exist to handle noisy samples for both curve and surface reconstruction, the majority of theoretically grounded approaches, including the methods proposed in this thesis, assume noise-free input data. Achieving such clean input typically involves denoising the point cloud prior to reconstruction. Nevertheless, in practice, even methods not explicitly designed to handle noise often demonstrate resilience to small amounts of noise.

We now introduce the formal notation that we use throughout this chapter to refer to point clouds and some of their properties. We mostly refer to 2D point sets as input sets for curve reconstruction, and mention explicitly if this assumption does not hold in a particular context.

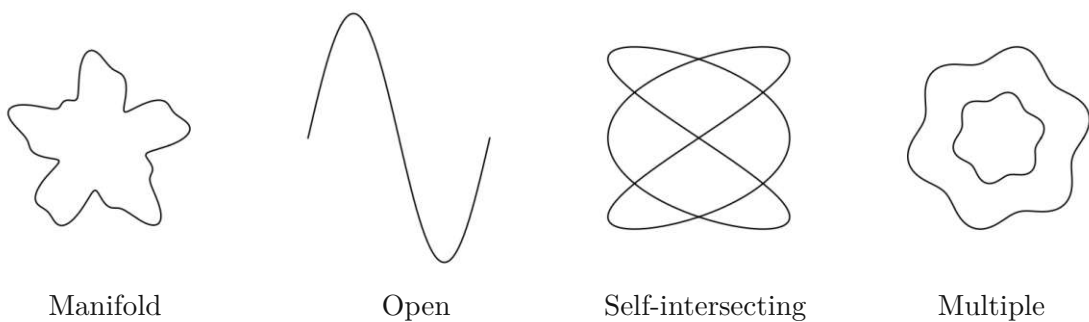


Figure 2.2: Various curve types - most methods in the literature focus on manifold single curves, as they are the most commonly encountered (silhouette data, object contours).

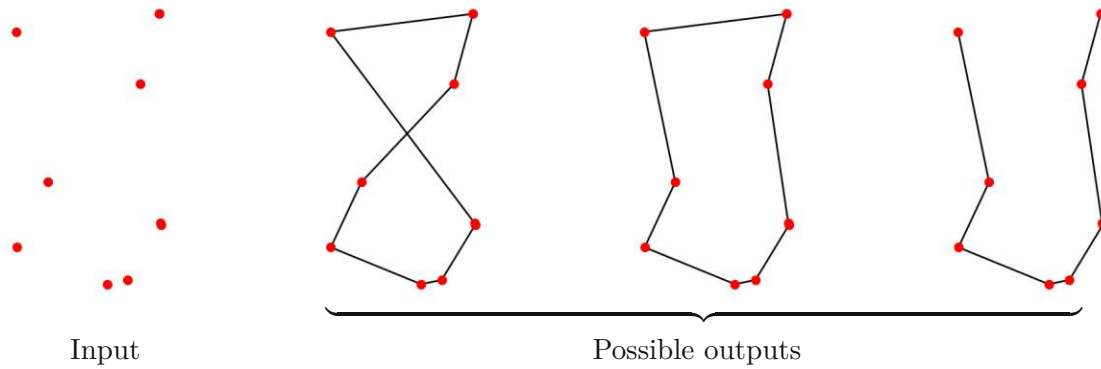


Figure 2.3: Given a sparse input that does not capture enough information about the curve’s features, any of the provided reconstructions could be correct. However, in most cases, priors are assumed about the expected solution - manifold or non-manifold, closed or open, resulting in specialized methods that best deal with each type of curve.

**Notation.** Let  $\mathcal{C}$  be a smooth planar curve: a closed, compact 1-manifold (loosely defined as a one-dimensional space that locally resembles a Euclidean space) embedded in 2D space [ABE98]. Most curve reconstruction methods deal with smooth, closed curves with a single connected component, but there are specialized methods as well, which are designed to reconstruct open curves, self-intersecting curves, or curves with multiple connected components - Figure 2.2. Let  $S$  be a subset of  $\mathcal{C}$ , with  $n$  elements. The problem of curve reconstruction aims to obtain  $\mathcal{C}'$  - a reconstruction of  $\mathcal{C}$  from  $S$ , an approximation of the ground truth through geometrical and topological lenses. This objective can be achieved through exact methods where  $\mathcal{C}'$  is a polygonal representation of  $\mathcal{C}$ , by extracting an ordering of the samples that matches the ground truth curve. However, in cases where the input does not offer enough information about the initial curve, multiple outputs are possible - Figure 2.3, and methods are usually biased for a specific type of boundary to be able to offer reconstruction guarantees. Another possibility besides explicit methods is to use approximation methods where the output -  $\mathcal{C}'$ , is also a smooth curve, that might include the set  $S$  or not, approximating  $\mathcal{C}$  based on some error metric.

Before we discuss the various methods employed to reconstruct curves from a set of samples, we need to first describe the properties of sampling techniques, how these properties influence the reconstruction, and the possible guarantees these algorithms might offer.

## 2.2 Choosing the Right Samples

Point clouds represent partial data about the ground truth, where not all information is available, since we only have access to disconnected samples. Hence, we need to be able to ensure that the samples that are part of the point cloud contain enough data for an accurate reconstruction. Furthermore, for a quick computation, we aim to have a

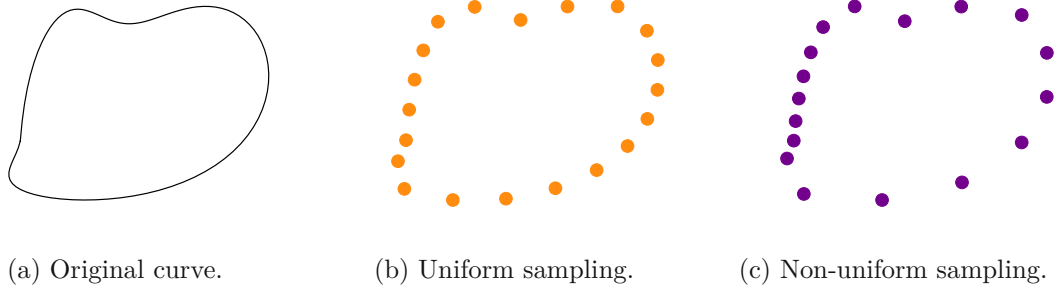


Figure 2.4: Distance-based sampling strategies, where the distance between consecutive samples is the only condition used to define this type of sampling.

small number of points but we do not want to lose any important features of the ground truth curves. Thus, the size of the point set and the amount and quality of encoded information have to be carefully balanced.

As a first approach to conditional sampling, the distance between consecutive samples can be used to create the subset  $S$  - Figure 2.4.

**Definition 1.** Uniform sampling is achieved if the distance between samples remains constant.

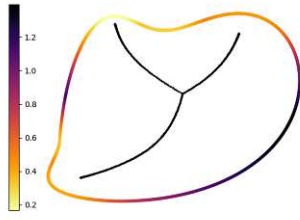
**Definition 2.** Non-uniform sampling allows for different distances between consecutive samples. However, bounding the ratio between consecutive samples by a constant -  $u$ , allows us to still ensure some theoretical guarantees for the reconstruction. The local non-uniformity ratio  $u$  is the ratio between the longer and the shorter distances of a sample to its neighbors on the curve:  $u = \frac{\text{long}}{\text{short}}$  [OM13].

However, this type of distance-based sampling does not take into account the size of the curve features (i.e., informally, how much the curve bends in various places, or formally - the curvature) and hence, features of the curves can be completely missed if they are smaller than the chosen distance between samples. To be able to sample according to the various curve features we need to introduce the concept of the local feature size [ABE98], which in turn, depends on the medial axis [Blu67].

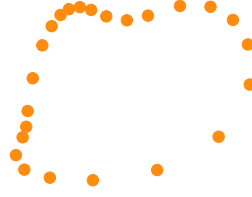
**Definition 3.** The medial axis is defined as the closure of points in  $\mathbb{R}^2$  that are closest to at least two points on the curve  $\mathcal{C}$  - depicted as the black curve in Figure 2.5a.

An alternative definition of the medial axis is the collection of centers of maximally inscribed circles in the shape. Informally, the medial axis can be imagined as the skeleton of the shape.

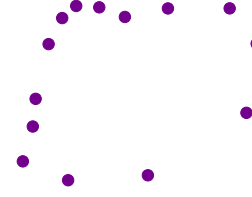
**Definition 4.** The local feature size of a point on the curve is defined as the shortest distance from the point to the medial axis - illustrated as color-coded for each point of the curve in Figure 2.5a.



(a) Color-coded distance to the medial axis of the points on the curve.



(b) Epsilon-sampling that depends on the local feature size of each point.



(c) Rho-sampling - local feature size-based, but sparser than the epsilon version.

Figure 2.5: Feature-based sampling strategies, where the distance to the medial axis (the local feature size) is taken into account. Samples are denser around high curvature points and sparse in the *flatter* parts of the curve.

By combining the local feature size with the distance between samples on the curve, we obtain a type of sampling that is able to better encode the curve's characteristics. Hence,  $\epsilon$ -sampling takes advantage of this union of information as follows:

**Definition 5.** A curve is  $\epsilon$ -sampled by the point set  $P$  if for each point  $c \in \mathcal{C}$ , the ratio between the distance to the closest sample in  $P$  and its local feature size is less than  $\epsilon$ .

Thus, this type of sampling is able to preserve the features of the curves, allowing for a number of samples that depends on the size of individual features - Figure 2.5.

We are still only considering the local feature size of individual points, and not using any information from neighboring samples, which usually encode similar data due to their proximity on a smooth curve. To relax this type of sampling, reach-based sampling has been introduced [OMW16], and it is based on interval-wide local feature size, and not the points' individual local feature size.

**Definition 6.** The reach [Fed59] of an interval  $I \subset \mathcal{C}$  is the minimum local feature size among points in the interval:

$$\text{reach}(I) = \inf_{p \in I} \text{lfs}(p). \quad (2.1)$$

**Definition 7.** A smooth curve  $\mathcal{C}$  is  $\rho$ -sampled [OMW16] by a sample set  $S \subset \mathcal{C}$  if every point  $p \in \mathcal{C}$  of the curve is closer to a sample than a  $\rho$ -fraction of the reach of the interval  $I = [s_0, s_1]$  of consecutive samples containing it:

$$\forall p \in I, \min_{s \in \{s_0, s_1\}} d_{\mathcal{M}}(p, s) < \rho \text{ reach}(I). \quad (2.2)$$

Even if these sampling strategies take into account the features of the curve, they still suffer from limitations - e.g. sharp angles, where the medial axis is extremely close to the curve and such local feature size-based sampling would require infinitely many closely spaced samples - Figure 2.6. Hence, there are methods designed to deal specifically with sharp features [DW00, DW02], and that can deal with curves that are not smooth.

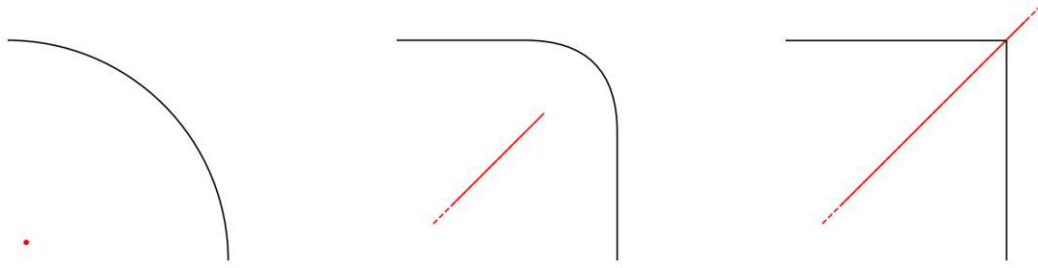


Figure 2.6: Sharp angles force the medial axis to extend extremely close to the formed corner, rapidly decreasing the local feature size. Hence, when sharp angles are present, feature size-based sampling usually fails, as infinitely many samples are required close to the sharp angles. The medial axis, in red, extends towards the corner as the angle becomes sharper. Note, that, practice, reconstruction methods achieve results better than their theoretical guarantee [OPP<sup>+</sup>21], and are sometimes able to deal with adversarial inputs.

## 2.3 Connecting the Dots

The first step in reconstructing the original data (be it curves or surfaces) from point clouds is to extract the connectivity among the input points - *i.e.* determining which samples would be neighboring each other in the original setting. Various proximity encoding graphs are used to approximate the connectivity. Examples include connecting points based on distance, imposing a constant number of neighbors for each point, computing a graph on the points and extracting edges with various properties, or using more complex distance functions between points.

### 2.3.1 Preset Parametric Proximity

Parametric proximity graphs are essential tools for defining the neighborhood relationships between points in point clouds, enabling the analysis of spatial structures and relationships. Two common types of proximity graphs that depend on specific parameters are the  $k$ -nearest neighbor ( $k$ NN) graph and the  $\epsilon$ -neighborhood graph. The  $k$ NN graph connects each point to its  $k$  closest neighbors, where  $k$  is a predefined parameter, ensuring that each point is connected to a fixed number of points - Figure 2.7. On the other hand, an  $\epsilon$ -neighborhood graph connects points that lie within a specified distance -  $\epsilon$ , from each other, creating edges only between points that are spatially close within that threshold.

However, this type of parameterized proximity is strongly influenced by the choice of the global parameter -  $k$  or  $\epsilon$ , which could result in too many connections if the covered area is too large, or isolated areas for too small values of the parameter. Hence, picking a value depends heavily on the particular type of data and how it is sampled. Being able to choose  $k$  or  $\epsilon$  usually requires more time to find a suitable value, but it allows the freedom of dealing with varied data types.



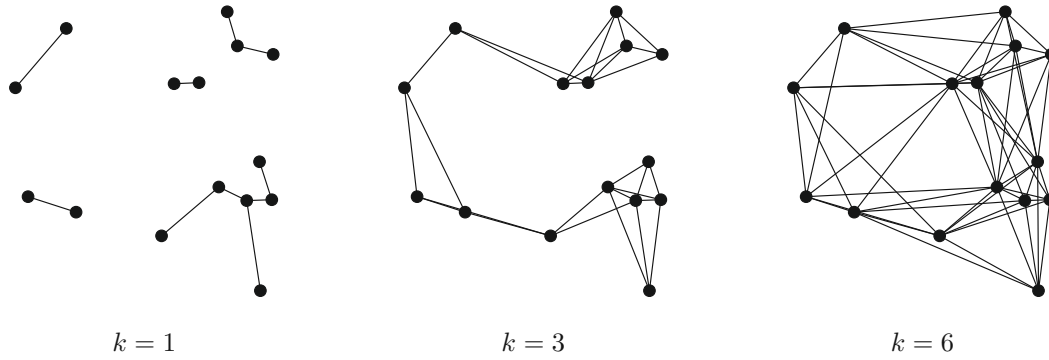


Figure 2.7:  $k$ NN proximity graphs created by setting a predefined number of neighbors for each node.

### 2.3.2 Delaunay Subsets

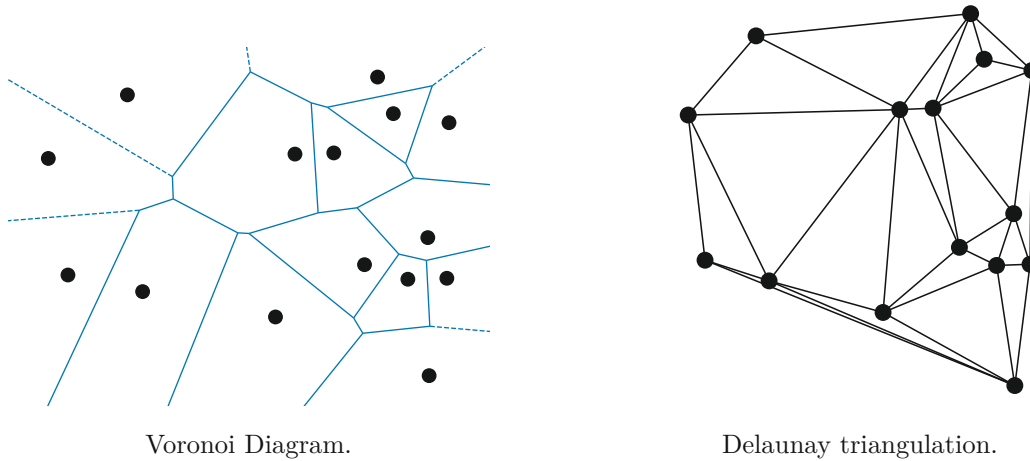
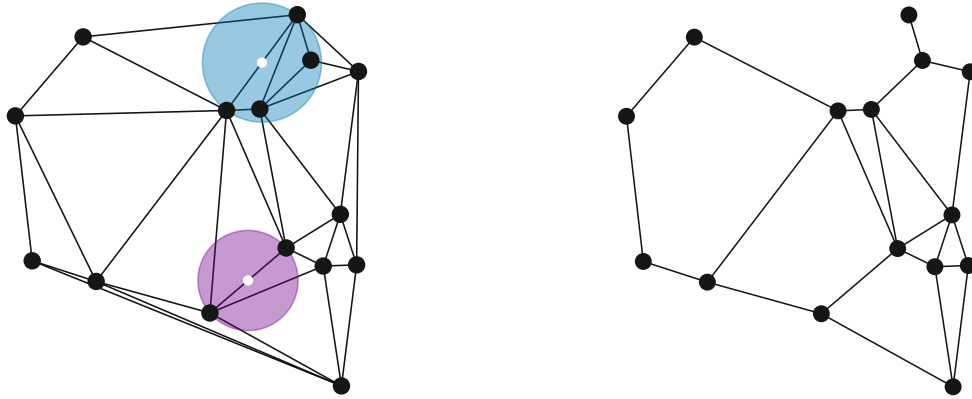


Figure 2.8: Duality of the Voronoi diagram and the Delaunay triangulation of a point set - any two neighboring Voronoi partitions create an edge in the Delaunay triangulation.

The Delaunay triangulation of a given set of points is a triangulation such that no point in the set is inside the circumcircle of any triangle in the triangulation. This holds in higher dimensions by considering simplices and their circumspheres, instead of triangles. The Voronoi diagram of a plane (or higher dimensional spaces) is a related geometric structure, where each region around an input sample contains all the locations closer to the current sample than to any other. The Delaunay triangulation and the Voronoi diagram are closely related, as the Delaunay Triangulation is the dual graph of the Voronoi partitioning. This means that the vertices of the Voronoi diagram correspond to the circumcenters of the triangles in the Delaunay triangulation - as illustrated in Figure 2.8. These two structures are fundamental in computational geometry [AKL13], with applications in multiple fields of computer science.



(a) Starting from the Delaunay triangulation, if the circle centered on each edge does not include any other samples, the edge is included in the Gabriel graph as well - the edge spanning the purple circle, as it is empty of samples, compared to the blue one.

(b) Resulting Gabriel graph once all edges that do not satisfy the empty circle property are removed from the starting Delaunay triangulation.

Figure 2.9: Gabriel graph creation.

Various subsets of the Delaunay triangulation can be used as connectivity graphs, such as the relative neighborhood graph [Tou80], the Gabriel graph [GS69], the  $\alpha$ -complex [Ede83], the  $\beta$ -skeleton [KR85] and the minimum spanning tree. For an edge  $pq$  to exist in the relative neighborhood graph, there cannot exist another point  $r$  that is closer to  $p$  or  $q$  than they are to each other. In the Gabriel graph, two points -  $p, q$ , are connected if the closed disc with diameter  $pq$ , passing through  $p$  and  $q$ , contains no other samples - Figure 2.9. The  $\alpha$ -complex contains all simplices of the Delaunay triangulation that can be enclosed with a circle of radius  $1/\alpha$ , empty of other samples. The  $\beta$ -skeleton has been introduced as a scale-invariant version of  $\alpha$ -shapes, where the edge  $pq$  is part of the graph if angles  $prq$  are bound by a threshold -  $\beta$ . The minimum spanning tree (MST) is a subset of edges that connects all points with the minimum possible total edge length, without forming any cycles. These graphs are subsets of the Delaunay triangulation [JT92], usually computed by pruning the triangulation - Figure 2.10.

### 2.3.3 Sphere-of-Influence Graph

Parameter proximity graphs and the Delaunay triangulation with its subgraphs are the most widely used starting steps for curve reconstruction, but throughout this thesis, we choose to focus on another proximity graph - the spheres-of-influence graph, which received little attention until now. It benefits from interesting theoretical properties while encoding connectivity well. This graph ended up being the *connecting* thread of the thesis, and of all the mentioned publications, due to its wide applicability and good results in multiple problems.

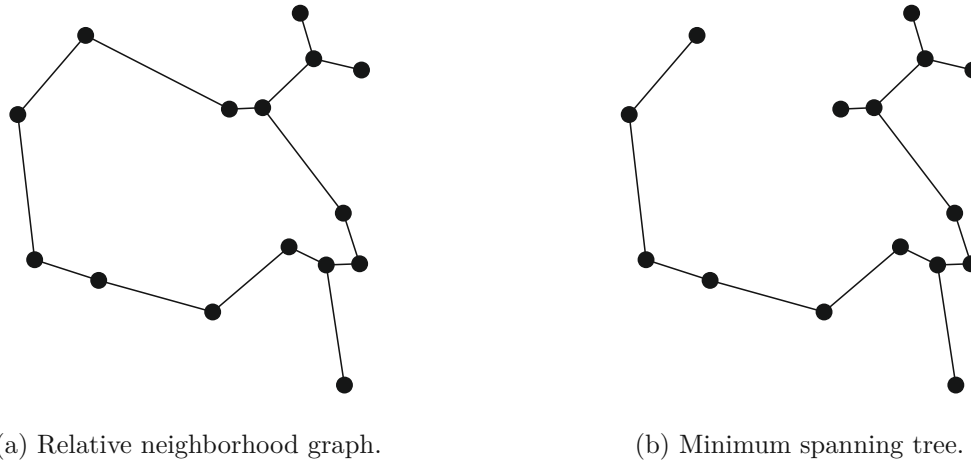
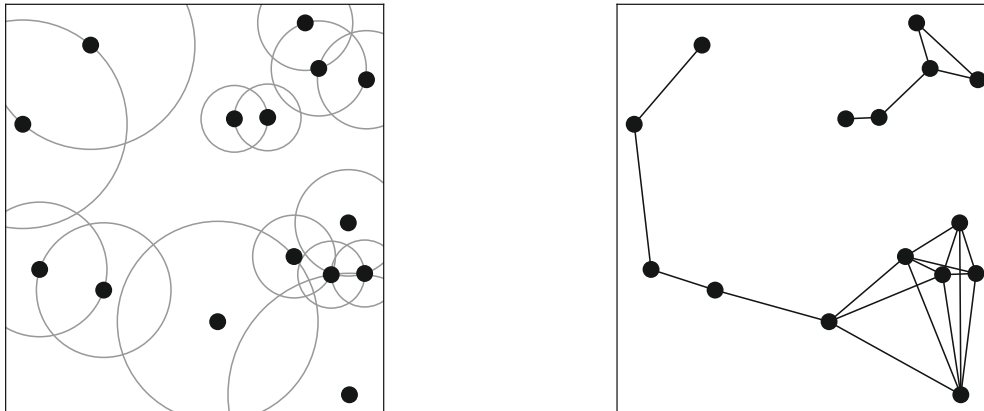


Figure 2.10: Other subsets of the Delaunay triangulation.

The spheres-of-influence graph - SIG [Tou88] has been introduced as a clustering method since it encodes proximity without the need for any parameters. The SIG has been used in implicit surface reconstruction methods [KZ04] to approximate local geodesic distances, as well as a density function over the input points to adapt the algorithm. This is an implicit method that uses the local kernels based on SIG to reconstruct the surface. The same approach, to use the SIG to locally define an implicit function, has been applied to collision detection between point clouds [KZ05]. Moreover, the SIG has recently gotten attention in reconstruction methods [dP22] where combined with the Delaunay graph it showed promising results for region reconstruction.



(a) Spheres-of-influence of each node, where the radius is equal to the distance to the node's nearest neighbor.

(b) Nodes are connected when their spheres-of-influence overlap to create the SIG.

Figure 2.11: Sphere-of-influence graph (SIG) creation.

**Definition 8.** In SIG, two vertices are connected by an edge if the distance between them is less than the sum of distances to their respective nearest neighbors. More formally, we connect  $a$  and  $b$  if

$$d(a, b) \leq nn(a) + nn(b), \quad (2.3)$$

where  $d(a, b)$  is the distance between two points  $a$  and  $b$  and  $nn(a)$  is the distance  $d$  between  $a$  and its nearest neighbor.

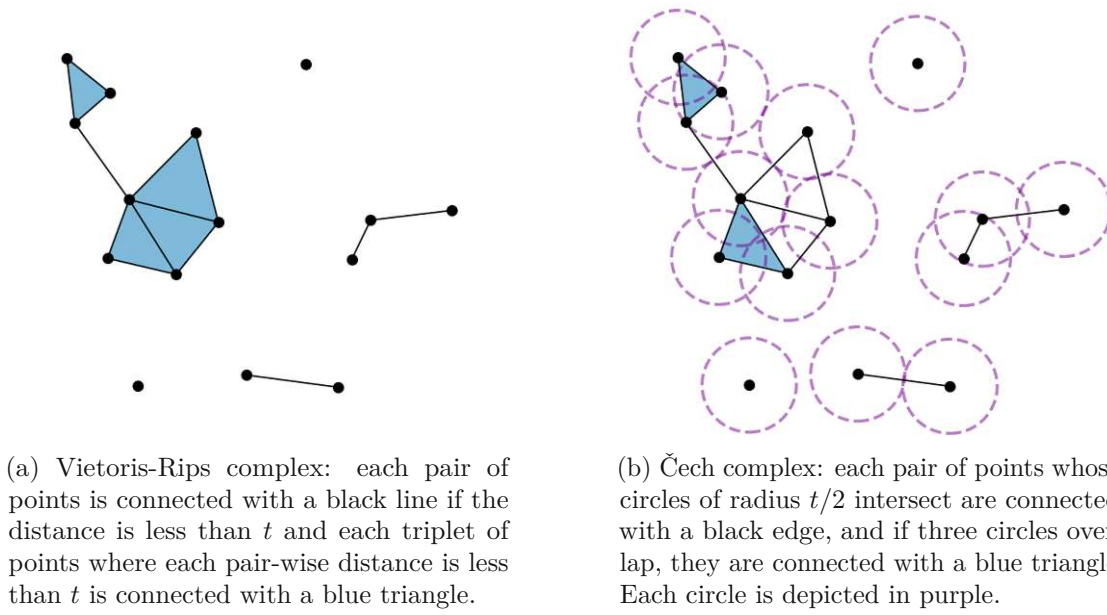
Visually, the SIG can be interpreted as centering a circle at each vertex whose radius is equal to the distance to its nearest neighbor, *i.e.*, just touching this closest vertex, as can be observed in Figure 2.11. We then connect with an edge all vertices whose circles intersect. This relation encodes spatial proximity without requiring a fixed number of neighbors that the user has to specify, such as the  $k$ -neighborhood. Unlike many other proximity graphs, the SIG is not a subset of the Delaunay triangulation and does not include a triangulation of the input.

### 2.3.4 Clustering

Another tangent direction to the current section, *Connecting the dots*, is represented by clustering - assigning the same label to samples that exhibit similar features. Even though clustering is not directly related to reconstruction from point sets, one of our works is related to clustering. We are using an extended version of the spheres-of-influence graph to define a clustering method with a strong background in topological data analysis in Chapter 6.

We recognize the impossibility of providing a comprehensive survey of existing work on clustering, since this only represents a subset of the thesis, and we choose to focus on revisiting the concepts most closely related to our context. Data-clustering algorithms have a rich history since the early works such as [M<sup>+</sup>67], followed by significant research advancement, including the introduction of spectral clustering [SM00, VL07], center-based methods [BMD<sup>+</sup>05, Teb07], mixture models [FR02], mean shift techniques [CM02], density-based spatial clustering (DBSCAN) [EKSX96], single-linkage technique [GR69], affinity propagation algorithms [FD07], various adaptations of K-means clustering [ASI20], and innovations in feature selection [WT10], among others. Recent literature on learning-based clustering, such as [CLX<sup>+</sup>19, AC20], have focused on 3D point clouds, along with surveys [RPY<sup>+</sup>22, RXL<sup>+</sup>23].

As previously mentioned, a popular proximity criterion that considers local density is the  $k$ NN graph. However, it may not always be robust to noise in the data. On the other hand, the  $\epsilon$ -neighborhood graph offers stability results and is known for its resilience to noisy datasets but does not take local density into account. The proximity formulation that we introduce in Chapter 6 combines the advantages of both approaches by integrating the local density considerations of  $k$ NN with the robustness to noise inherent in  $\epsilon$ -neighborhood graph, also leading to a more comprehensive clustering solution.



(a) Vietoris-Rips complex: each pair of points is connected with a black line if the distance is less than  $t$  and each triplet of points where each pair-wise distance is less than  $t$  is connected with a blue triangle.

(b) Čech complex: each pair of points whose circles of radius  $t/2$  intersect are connected with a black edge, and if three circles overlap, they are connected with a blue triangle. Each circle is depicted in purple.

Figure 2.12: Distance-based complexes, which represent one snapshot of a filtration for a given value of the parameter.

**Topological data analysis.** In this section, we briefly review some of the material in Topological Data Analysis (TDA) relevant to our contributions. For a more detailed introduction to the subject, we refer the reader to standard textbooks such as [EH10, Oud15].

**Definition 9.** A filtration  $\mathcal{F}$  of a topological space  $K$  over a totally ordered set  $T$  is a family  $(\mathcal{F}_t)_{t \in T}$  of subspaces of  $K$  that are nested in terms of inclusion, that is:

$$\forall t \leq t' \in T, \mathcal{F}_t \subseteq \mathcal{F}_{t'}.$$

The *Vietoris-Rips filtration*  $\text{VR}$  is a popular choice of simplicial filtration in TDA applications. Given a point cloud  $P$  equipped with a dissimilarity  $d$ , it is a filtration of the full simplex  $K = 2^P$  (i.e., the power set of  $P$  viewed as a simplicial complex) indexed over  $T = \mathbb{R}^+$ . For any  $t \in \mathbb{R}^+$ , the subcomplex  $\text{VR}_t(P, d)$  of  $2^P$  formed by those simplices that appear before or at  $t$  is called the *(Vietoris-)Rips complex* of  $P$  of parameter  $t$ .

Similarly, the Čech complex can be defined, where simplices correspond to the intersections of balls of radius  $t/2$  centered at the points of  $P$ .  $\text{VR}(P, d)_t$  generalizes the  $t$ -ball graph of  $P$  in the following way: the vertices represent the points of  $P$ , and a simplex (not just an edge) exists if and only if its diameter is smaller than  $t$ . Varying the value of  $t$  from 0 to  $+\infty$  gives the Rips filtration of  $(P, d)$ . An example of Rips and Čech complexes for a given value of  $t$  using the classical Euclidean distance is illustrated in Figure 2.12.

Such filtrations are powerful tools in TDA, used to identify significant topological features, such as connected components, loops, and voids, within data and across multiple scales.

By varying the parameter  $t$ , the Rips filtration captures how these features persist or disappear, allowing for the identification of persistent structures that reflect the intrinsic geometry and topology of the data. These persistent topological features can then be used for clustering by grouping data points based on shared topological characteristics, effectively separating data into clusters that correspond to distinct topological patterns. However, the parameter  $t$  is still an absolute distance and does not incorporate any data about the point set. We show in Chapter 6 how by replacing the usual distance metric with a relative neighborhood metric, we are able to capture more information about various point sets.

### 2.3.5 Towards Reconstruction

Once the connectivity of a point set has been determined, various properties of the point set can be estimated before the actual reconstruction happens. Usually, such additional attributes, such as normals, represent an aid or a requirement of some reconstruction methods.

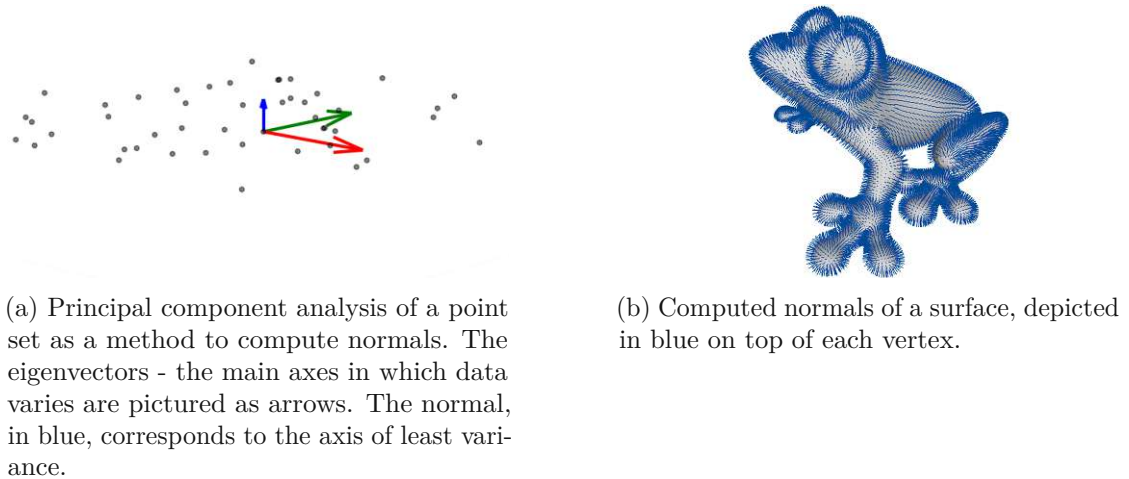


Figure 2.13: Surface normals and their computation.

Normal estimation for point clouds has been a heavily researched area of computer graphics, as it usually represents a first step or requirement in surface reconstruction, *e.g.*, for Poisson reconstruction [KBH06] or data-driven approaches such as Point2Surf [EGO<sup>+</sup>20]. One of the basic methods for computing normals for unstructured point clouds is using Principal Component Analysis (PCA). This method considers a local patch of vertices and finds the axis of least variance since the points should vary the least in the normal direction - Figure 2.13. This is equivalent to computing the covariance matrix of the local neighborhood of each point, and choosing the eigenvector corresponding to the smallest eigenvalue as the point's normal. It is to be noted that this method does not gener-

ate a consistent orientation of normals, and that usually happens in a post-processing step [HDD<sup>+</sup>92].

The relation between the choice of neighborhood and normal estimation has been extensively studied [MN03]. Other methods improve on the tangent plane estimation by using a weighted approach when considering the local neighborhood [PKKG03] or by fitting algebraic spheres [GG07].

Another avenue for estimating normals for point clouds has been developed with the concept of *poles* [AB99]. This method is based on computing the Voronoi diagram on the input points and extracting the normals as the line connecting each sample point and the farthest Voronoi vertex to their Voronoi cell (the pole). This method is sensitive to noise, but it has been improved to handle noisy samples [DG06] where it requires additional parameters.

Recently, multiple data-driven approaches have been developed, which usually take advantage of large data repositories to learn the geometric relations between the point clouds' structure and their expected normals. Some methods use a Hough transform to estimate normals [BM16]. PCPNet [GKOM18] builds on the PointNet architecture [CSKG17] and uses local patches to estimate the properties of point clouds with various noise levels and sampling densities. However, these types of methods require a long processing time and possible re-training depending on the type of data.

An alternative to dealing with noise in the existing point clouds is to first denoise them and then use a connectivity/normal computation approach that works well on clean data. In this direction, some data-driven approaches deal with noise [RLBG<sup>+</sup>20] by building on PCPNet [GKOM18] to classify outliers and then reproject noisy samples on the original surface. Others use a noisy point distribution model to estimate scores and the direction of the surface [LH21]. Classical, analytical approaches for denoising include filtering in various forms, such as the bilateral filter that takes normals into account [DdF17] or voting schemes for feature detection that later help in normal repositioning [LXZ<sup>+</sup>20].

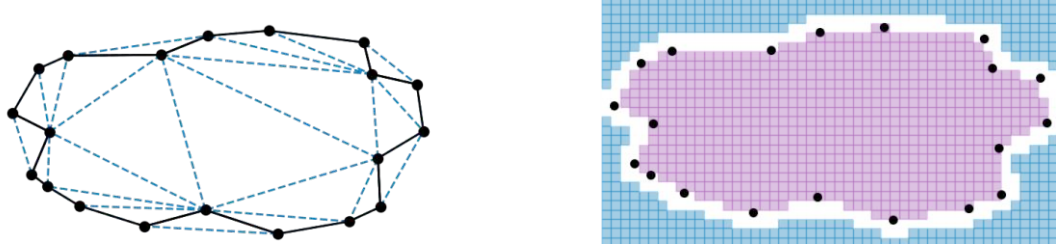
Moreover, connectivity retrieval has applications in other data-driven approaches. Methods that use graph convolutional neural networks on point clouds for various purposes, such as reconstruction or segmentation, create a graph on the input points (and sometimes on the data from the deeper layers of the network as well) and use it to learn about the data and infer the surface or various labels on the input points [LLC<sup>+</sup>24, SR20, WHH<sup>+</sup>19]. Usually, they use  $k$ NN or a fixed radius neighbor search as their connectivity encoding.

## 2.4 Plane and Simple

This section provides an overview of existing curve reconstruction methods, which are generally classified into two main categories - Figure 2.14. The first category encompasses explicit (interpolatory) methods, where the output is represented by an ordering of the original input points, resulting in a polygonal reconstruction. These methods usually use a starting graph, filtering some of the edges based on various criteria. The second



category includes implicit (approximating) methods, where the curve is defined indirectly, usually as the level set of an indicator function that separates the inside and the outside of the curve.



(a) Explicit reconstruction. A graph is computed using the input points, and a subset of the edges is chosen, subject to various conditions.

(b) Implicit reconstruction. The curve is approximated as the zero set of an indicator function between inside - purple, and outside - blue.

Figure 2.14: Planar reconstruction.

### 2.4.1 Explicit Methods

Explicit methods output an ordering of the input points as reconstruction. The Delaunay triangulation represents a building block for multiple methods in this category due to its theoretical guarantees that are subject to sampling criteria, as previously mentioned. One of the first methods to use the Delaunay triangulation was CRUST [ABE98], where the  $\epsilon$ -sampling, discussed in Section 2.2, was also introduced. By choosing a subset of Delaunay edges, the reconstruction is guaranteed to be correct for  $\epsilon < 0.252$ .

Another method to reconstruct a curve was introduced by Dey and Kumar [DK99] – NN-CRUST, and it is a proximity-based method. It uses the edges between nearest neighbors as a starting crust, and for every leaf vertex, adds the shortest edge situated in the half-plane defined by the normal on the edge placed at the leaf vertex. Their results show that proximity-based methods capture the boundary shape, but the sampling limitations of this method (reconstruction is guaranteed for  $\epsilon < 1/3$ ) suggest the possibility of further improvements.

The NN-CRUST was further improved in HNN-CRUST [OMW16] to guarantee reconstruction up to  $\epsilon < 0.47$ . This method uses a similar idea to the half-plane but places the half-plane's normal as the bisector of the chosen edge.

In CONNECT2D [OM13], they used an initial graph  $BC_0$ , representing an approximation of the boundary, which they augmented by inflating and sculpting to reconstruct the boundary. The initial boundary graph is a subset of the Delaunay triangulation computed such that the minimum boundary length is approximated, all of the vertices are interpolated with a degree of at least 2, and are part of a single connected component. This graph was then processed by inflating, in order to make the curve manifold and contain all vertices on the boundary or inside of it. Candidate triangles were considered



for non-manifold vertices and sorted by the increase in total boundary length. Candidate triangles were then added to the graph to make vertices manifold (i.e. degree 0 or 2) until all of the vertices had become manifold. The resulting graph was processed by sculpting next. Sculpting tries to make the boundary interpolate any vertex that is currently isolated and inside of the boundary. This was done by sorting candidate triangles that are incident to isolated vertices and adding their edges to the graph to include the vertices in the boundary. They guarantee reconstruction for  $\epsilon < 0.5$  but require a non-uniformity ratio between distances of consecutive samples of  $u < 1.609$ .

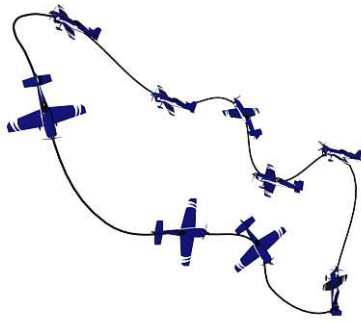
GATHANG [DW02] introduced a method specialized in reconstructing data sets with sharp corners. They used the angle and the ratio between edge lengths to filter Delaunay edges for the boundary and their performance is best for this sub-category of curves.

### 2.4.2 Implicit Methods

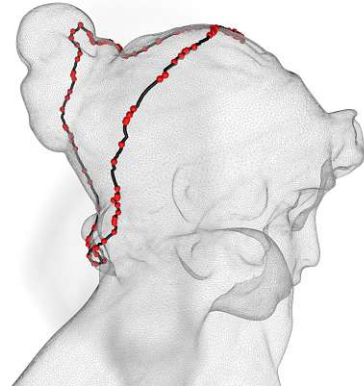
Implicit methods use a function computed over the entire domain of the input and the curve is usually approximated as the zero-set of that function. Important mentions in this type of reconstruction are Signed Distance Functions (SDFs) [HDD<sup>+</sup>92], Poisson-based reconstructions [KBH06], and radial basis functions [CBC<sup>+</sup>01]. These methods are usually applied already for surface reconstruction. The signed distance functions compute the signed distance between the points in the plane and the curve, and reconstruct the curve as the zero-set of the function. Poisson methods formulate the curve reconstruction as computing the curve whose gradient field of an indicator function is the most similar to the normals of the curve. Radial basis functions try to fit a radial basis function to a signed distance field computed over the input and compute the isoline of this smooth function. Our works rather fit in the explicit category, as the input points are interpolated and no function is computed over the domain.

An outlier for the explicit/implicit taxonomy is represented by transport methods. Optimal transport [GCSAD11] reconstructs a curve starting from the Delaunay triangulation and removing vertices and their incident edges by minimizing a global cost. This method is efficient for the reconstruction of noisy curves, a group of input that explicit methods cannot usually correctly reconstruct. Another recent method [CSLV20] tries to solve the curve reconstruction problem as an optimal homologous chain problem. They construct a lexicographic ordering on the simplices of the input to find a minimal chain that bounds the input.

A comprehensive collection of results on multiple types of data sets and different curve reconstructions can be found in the 2D Points Curve Reconstruction Survey and Benchmark [OPP<sup>+</sup>21]. Their results show that most algorithms tend to perform better than their theoretical  $\epsilon$ -sampling guarantee.



(a) Motion in three-dimensional space, *i.e.* a combination of position and rotation, can be imagined as existing in a higher dimensional space. Then, a single trajectory of an object is a curve bound to a manifold embedded in this higher dimensional space.



(b) Curves can exist on surfaces, as decorative elements on three-dimensional models or planar elements applied to surfaces. Samples are marked as red spheres, and the curve is computed as a black polyline connecting the samples.

Figure 2.15: Curves on Riemannian manifolds.

## 2.5 Spatial Upgrades

One might consider curve reconstruction to naturally only evolve into surface reconstruction while moving to higher dimensional spaces. However, there are multiple directions this increase in dimensionality can take: reconstructing curves that exist in 3D space, curves that are bound to manifolds embedded in arbitrary dimensions, or surface reconstruction. We introduce all these reconstruction variants, with a focus on curves bound to manifolds, since Chapter 4 introduces a reconstruction method for such inputs. As surface reconstruction does not constitute a contribution of this thesis, we will limit our discussion to a brief overview of the topic.

### 2.5.1 Non-planar Curves

Some curve reconstruction methods can be extended to **curves in higher dimensions** with no algorithmic changes [DK99], whereas other methods have been devised to account for this change in dimensionality. One such approach is based on thinning a point cloud using moving least square: for each input point, its neighborhood is approximated using a curve or a surface. Hence, this method is based on a good estimation of the local neighborhood as well as a good local fitting scheme. The authors choose to approximate the local neighborhood with an Euclidean minimum spanning tree, and apply a weighted quadratic function, obtaining a regression plane for each point, on which they reproject the points to thin the point cloud [Lee00].

The minimum spanning tree is a popular choice for approximating neighborhoods in unknown domains, being used for **curve reconstruction on Riemannian manifolds** as

well [SC13]. These curves are bound to exist on a manifold in any dimension. Intuitively, we can imagine a manifold as composed of multiple planar-like pieces glued together to form a continuous space. By using the minimum spanning tree with distance computations restricted to the manifold, they obtain a guaranteed reconstruction if the sampling follows some specific, dense rules.

Curve reconstruction on Riemannian manifolds has been approached as an extension of planar metrics and sampling conditions to Riemannian manifolds [SC13]. However, they show that the classical definition of the medial axis does not hold on surfaces if the medial axis is constrained to live on the surface as well. The authors also introduce a new sampling criterion, based on the minimum value between the distance to the medial axis and the injectivity radius. We introduce the formal definitions of these concepts and we relax their dense sampling requirements in Chapter 4 to allow for reconstruction with fewer samples. To the best of our knowledge, this is currently the only work that extends the problem of curve reconstruction to Riemannian manifolds - Figure 2.15.

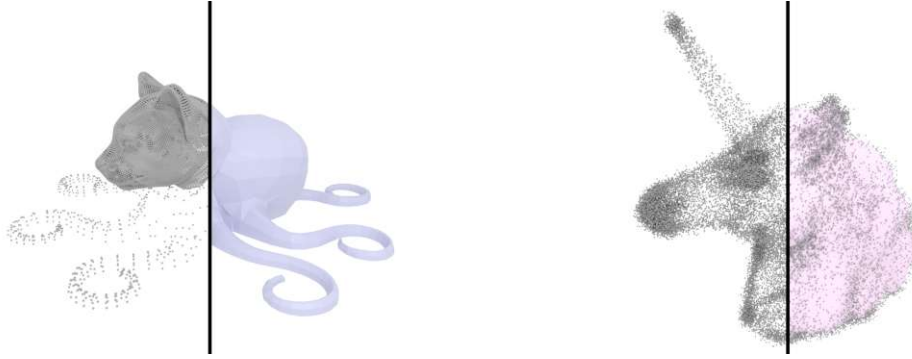
In the same realm of elements that exist on top of surfaces, vector graphics can be utilized to enhance surface design by acting as decorations, enabling intricate and precise elements that are not limited to the underlying geometry. Vector graphics on planar surfaces have been thoroughly researched and are being used in multiple tools [Ink, Ado]. Recently, editing and importing curves on surfaces have received interest in the graphics field, partly due to the improvement in computing geodesic paths efficiently [SC20]. Users can interact with designs directly on the mesh, by either editing splines on a 2D local projection of the surface [PSOA18], which is usually prone to artifacts due to the projection procedure, as explored in [YLT19], or editing splines directly on the surfaces, bypassing the projection artifacts, by using geodesic metrics on the surface [NPP21, MNPP21]. However, these editing methods require user input or a predefined ordering of the samples to construct the curves on the surface, which is the missing link we are providing, as a building block for further editing splines on surfaces, in Chapter 4.

## 2.5.2 Surface Reconstruction

Surface reconstruction is a fundamental problem in computer graphics, representing the classical evolution of curve reconstruction to three dimensions. Besides the ill-posedness of the problem, where not enough data is available for a single and complete result, additional issues affect this procedure. Common challenges include noise, which can diminish the accuracy of the reconstruction, and non-uniform sampling, where points may be irregularly distributed across the surface. Effective reconstruction methods must address these challenges to accurately represent the underlying surface - Figure 2.16.

Since the field of surface reconstruction deals with the three-dimensional version of curve reconstruction, we organize the existing methods into a similar taxonomy: explicit and implicit. Most of the explicit methods rely on the Delaunay triangulation in its three-dimensional version to extract triangles that define the surface. The CRUST for curve reconstruction has been extended to surface reconstruction [ABK98], where the triangle

crust of the mesh is computed as a filter over three-dimensional Delaunay triangulation, and improved to handle noisy inputs, but the output models suffer from many additional triangles [ACK01]. However, most of the recent works in the surface reconstruction direction fall in the implicit category, as they allow for more flexibility and a convenient integration with data-driven methods.



(a) Due to equipment issues or line-of-sight errors, point clouds can be highly non-uniform in sampling, and the reconstruction method has to account for this.

(b) Another common challenge in surface reconstruction is the existence of noise, which has to be removed for a correct reconstruction. The noisy samples (grey) are depicted on top of the correct surface (pink).

Figure 2.16: Various challenges encountered in surface reconstruction.

Implicit surface reconstruction is pioneered by Hoppe's et al. [HDD<sup>+</sup>92] method that uses tangent planes as a local approximation of the surface. The Poisson [KBH06] and Screened Poisson [KH13] methods are widely used in practice, requiring normals to compute a signed distance function whose zero set approximates the surface. These methods have been recently extended to work in a distributed fashion [KH23]. Usually, the implicit methods generate a signed distance function that is turned into a triangulated mesh using Marching Cubes [LC87] or similar approaches. An extensive characterization of the implicit methods in the field of surface reconstruction is presented in [BTS<sup>+</sup>17], where methods are organized with respect to input requirements, artefacts and resulting outputs. More recent methods improve upon the Poisson reconstruction by incorporating an isovalue constraint into the equation, which aids in achieving consistent normal orientation and, as a result, enhances the reconstruction quality [XSL<sup>+</sup>23]. Other methods use an iterative approach to remove the need for normals from the classical method [HWW<sup>+</sup>22].

Recently, learning-based methods are gaining popularity, such as Point2Mesh [HMGC020], where they use the input as a self-prior in order to obtain insight in the form of repeating geometry. Points2Surf [EGO<sup>+</sup>20] trains on local and global patches at the same time, using modified variants of PointNet [CSKG17], obtaining a high-quality mesh that preserves fine details as well. Neural-IMLS [WWW<sup>+</sup>24] learns a noise-resilient signed distance function instead of computing in a self-supervised manner. POCO [BM22]

uses an encoder-decoder network that directly processes points, thereby avoiding the discretization artifacts associated with voxel-based representations.

For a recent overview of surface reconstruction methods, with a focus on data-driven approaches, together with a benchmark and thorough evaluation, we refer the reader to a comprehensive survey [HWW<sup>+</sup>24].

*Having presented all the necessary theoretical tools and explored a range of proximity-based methods for point cloud reconstruction, we will now present the individual contributions of this thesis. Each contribution builds upon the foundational concepts discussed in this chapter, providing a detailed examination of the topics and offering new insights and advancements in the field.*



# CHAPTER 3

## SIGDT: Planar Curve Reconstruction

We introduce our first application of the spheres-of-influence graph - curve reconstruction in the planar case. The contents of this chapter have been adapted from the paper “*SIGDT2D: Curve Reconstruction*”, published in the Computer Graphics Forum and presented at Pacific Graphics 2022 [MOW22b].

### 3.1 Overview

Determining connectivity between points and reconstructing their shape boundaries are long-standing problems in computer graphics. One possible approach to solve these problems is to use a proximity graph. We propose a new proximity graph computed by intersecting the to-date rarely used proximity-based graph called spheres-of-influence graph (SIG) with the Delaunay triangulation ( $DT$ ). We prove that the resulting graph, which we name SIGDT, contains the piece-wise linear reconstruction for a set of unstructured points in the plane for a sampling condition superseding current bounds and capturing well practical point sets’ properties. As an application, we apply a dual of boundary adjustment steps from the CONNECT2D algorithm to remove the redundant edges. We show that the resulting algorithm SIG-CONNECT2D yields the best reconstruction accuracy compared to state-of-the-art algorithms from a recent comprehensive benchmark, and the method offers the potential for further improvements, e.g., for surface reconstruction.

### 3.2 Introduction

Reconstructing a curve based on given samples with no additional information other than their position is a difficult task, considering that no connectivity information is

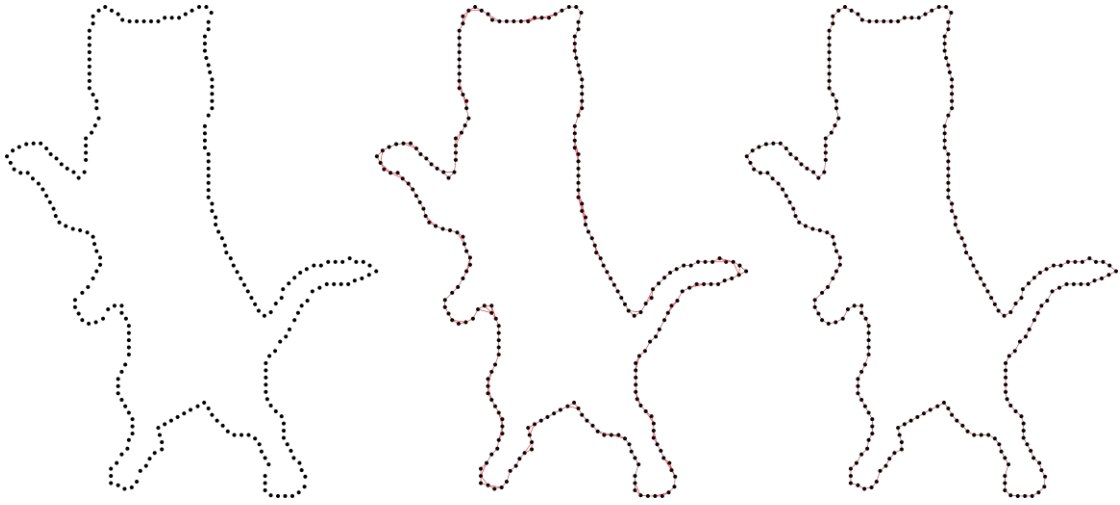


Figure 3.1: Starting from unstructured points (left), our proximity graph SIGDT (center) already contains the reconstructed boundary (right).

present. As a fundamental problem, with extension to surface reconstruction, it has received a lot of attention in the field during the last decades. The reconstruction usually implies generating a graph on the input points and filtering/adding edges to recover the connectivity. The resulting shape should interpolate all of the input points, and approximate best the boundary of the shape that the points were sampled from. Ideally, the reconstruction should be agnostic of the distance between samples and preferably not depend on parameters. However, in practice, this proves to be difficult, especially when multiple types of shapes are considered, such as open curves or multiply connected curves, since a larger distance between samples could mean either a hole in the boundary or unevenly spaced sampling. Hence, we restrict our main method to manifolds and curves with sharp corners.

We introduce a new proximity graph, based on the intersection of the *Spheres-of-influence graph* (SIG) and the Delaunay triangulation (*DT*), which we name SIGDT and present below. We show its good connectivity property and prove that it contains the piece-wise linear reconstruction of the samples for an enhanced bound of a sampling condition. In order to filter the reconstruction edges from the SIGDT, we apply the *inflating* and *sculpting* operations from the CONNECT2D algorithm [OM13], yielding a manifold boundary for the input point set.

We present the following three contributions:

- We introduce the graph SIGDT by intersecting the SIG with the *DT*. SIGDT represents connectivity well and is parameter-free.
- We show its good connectivity by proving that it contains the reconstruction edges for an enhanced sampling condition bound that conforms very nicely to point sets



in practice.

- As an application, we show manifold curve reconstruction by filtering edges from SIGDT, surpassing the state-of-the-art.

The full source-code is publicly available at <https://gitlab.cg.tuwien.ac.at/dmarin/sigconnect2d>.

### 3.3 Method

Here we first present the SIGDT, show its superior connectivity as a proximity graph, and prove its property of containing the reconstruction. Then, we propose a curve reconstruction algorithm as an application that filters edges of the SIGDT with operations from the CONNECT2D [OM13] algorithm, for which pseudo-code is listed as Algorithm 3.1 and a step-by-step illustration is presented in Figure 3.3.

We will next provide the definition of the terms used in the algorithm. The boundary is the connected set of edges that includes all edges of the graph  $G$  in its interior, e.g., the non-dashed edges in Figure 3.3f. A non-manifold vertex has more than two incident edges on that boundary, i.e., there the boundary is pinched together, or the spaces defined by its incident edges are exterior, see Figure 3.3e. An isolated vertex is not included in the boundary, see Figure 3.3f.

#### 3.3.1 Boundary-containing Proximity Graph SIGDT

Since the SIG is not contained in the  $DT$  but the latter has nice properties such as representing a decomposition of the plane into triangles (which is useful for applications such as curve reconstruction), we design a new proximity graph as the intersection of SIG and  $DT$ , which we name SIGDT. This graph combines the advantage of the local proximity offered by the SIG with the maximization of minimum angle triangles provided by the  $DT$ . Figure 3.2 shows a visual comparison with the  $BC_0$  proximity graph, which is a superset of the  $EMST$  constraining vertex degree to  $\geq 2$  instead of  $\geq 1$  and is used for CONNECT2D curve reconstruction. While SIGDT contains all the edges of the reconstruction,  $BC_0$  misses many of them. We show further quantitative comparisons of SIGDT with other proximity graphs in the results in Section 3.4.

We compute SIGDT as follows: In order to determine whether a Delaunay edge is in the SIG, we need to check whether its length is smaller than the sum of the nearest neighbor distance of both its vertices. For that, we first determine the shortest incident edge per vertex in the  $DT$  and store its length as its *nearest neighbor distance*. Then, we process all Delaunay edges in the  $DT$  and add conforming edges to the SIG definition in order to create the SIGDT. This results in  $\mathcal{O}(n \log n)$  time complexity.

The SIGDT graph guarantees containing the reconstruction under some sampling conditions.

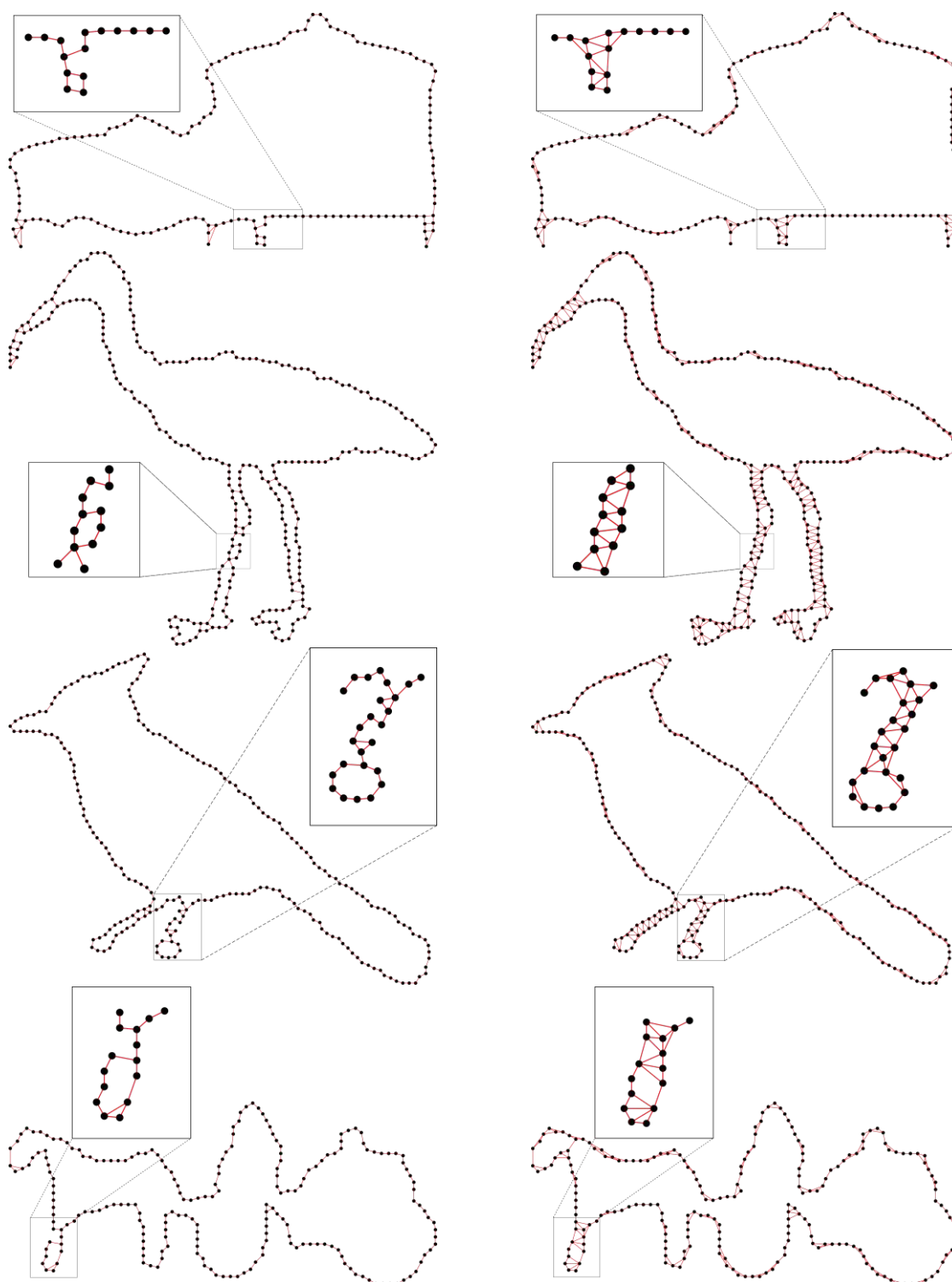


Figure 3.2: Comparison of  $BC_0$  (left) and SIGDT (right) in terms of encoding proximity. SIGDT manages to include all of the required edges of the reconstruction while  $BC_0$  misses many edges.

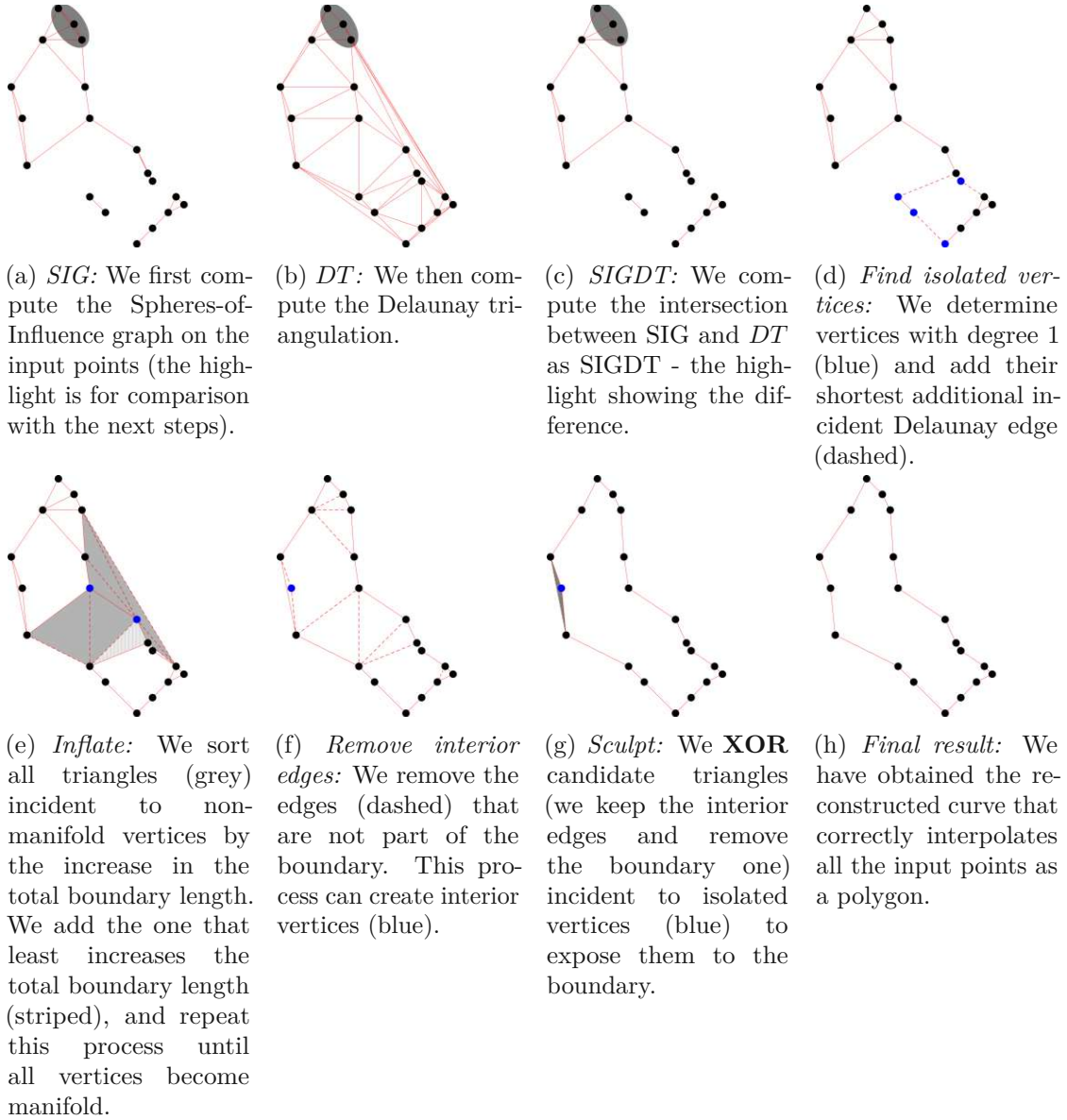


Figure 3.3: Overview of SIG-CONNECT2D algorithm.

---

**Algorithm 3.1: SIG-Connect2D**


---

**Data:** Input point set  $P$   
**Result:** Output edge set  $R$

```

1  $G = \{\}$ 
2 Compute Delaunay triangulation  $DT(P)$ 
3 for  $p \in P$  do
4   | Compute  $NN$  as shortest edge incident to  $p$ 
5 end
6 for  $p \in P$  do
7   | for  $q \in 1 - ring - neighbourhood(p)$  do
8     | Add edge  $pq$  to  $G$  if  $pq \leq NN(p) + NN(q)$ 
9   | end
10 end
11 while  $\exists$  non-manifold vertices in  $boundary(G)$  do
12   |  $T = \text{triangles} \in DT$  exterior to and incident to  $boundary(G)$ 
13   | Add  $t \in T$  that least increases the length of  $boundary(G)$ 
14 end
15  $G = boundary(G)$ 
16 while  $\exists$  isolated vertices  $\in DT$  interior to  $boundary(G)$  do
17   |  $T = \text{triangles} \in DT$  interior and incident to  $boundary(G)$ 
18   | Add  $t \in T$  that least increases the length of  $boundary(G)$ 
19 end
```

---

**Proof for  $\rho < 1, u < 2$ :** We will now show that SIGDT guarantees to contain the reconstruction of  $C$  if it is sampled with  $\rho < 1/7$  and a local non-uniformity ratio of  $u < 2/2$  between distances to samples adjacent on  $C$ . A  $\rho < 1$ -sampling is equivalent to an  $\epsilon < 0.5$ -sampling [OMW16]. This improves on CONNECT2D [OM13], which proves reconstruction for  $\epsilon < 0.5$  as well but requires  $u < 1.609$ , and handles a more relaxed  $\epsilon$ -sampling than the  $\epsilon < 0.47$  for HNN-CRUST [OMW16], although the latter does not require any uniformity. We construct the proof by showing that each edge between consecutive samples is contained in the SIG as well as in the  $DT$ .

We need to repeat two statements [ABE98] for our proof:

**Corollary 1.** *A disk centered at a point  $p \in C$  with radius at most  $lfs(p)$  intersects  $C$  in a topological disk (Corollary 4).*

**Lemma 1.** *Any Euclidean disk containing at least two points of a smooth curve in the plane either intersects the curve in a topological disk or contains a point of the medial axis (or both) (Lemma 1).*

Now we can prove the following theorem for the SIGDT graph:

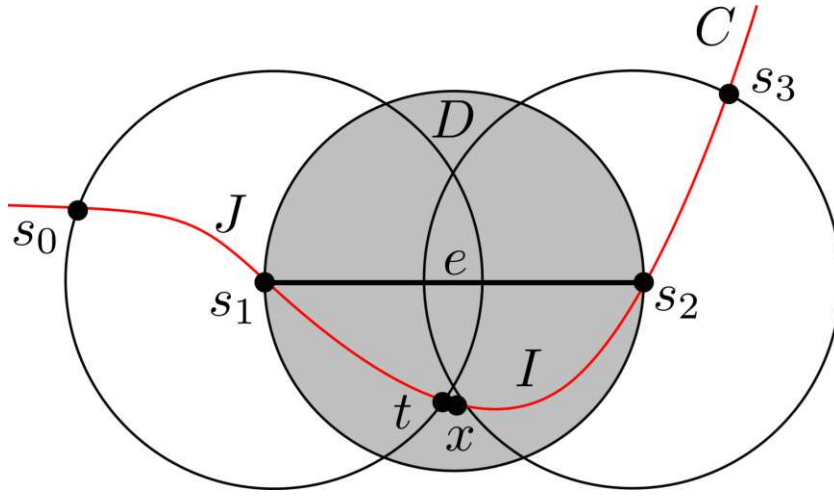


Figure 3.4: The red curve  $C$  passes through consecutive samples  $s_0, s_1, s_2, s_3$  and contains the intervals  $J[s_0, t]$  for  $t \in I[s_1, s_2]$  and  $I[s_1, s_2]$ .  $x \in I$  is the point farthest from the endpoints of  $I$ . The edge  $e$  is in  $SIG$  because the white disks centered at its endpoints  $s_1, s_2$  with radii as nearest neighbor distances overlap. The shaded disk  $D$  centered at edge  $e$  must be empty of samples other than  $s_1, s_2$  to be in  $DT$  (by being in the Gabriel graph).

**Theorem 1.** *SIGDT contains the reconstruction  $R$  of a smooth planar curve  $C$  from a set of points  $P$  that is sampled with  $\rho < 1, u < 2$ .*

*Proof.* See Figure 3.4 for illustration. Let  $s_0, s_1, s_2, s_3$  be consecutive samples on  $C$  and we want to show that the edge  $e(s_1, s_2) \in R$ . For this, we need to prove both  $e \in SIG$  and  $e \in DT$ .

1. Prove that  $e \in SIG$ : The non-uniformity ratio  $u < 2$  requires that  $\|s_0, s_1\|, \|s_2, s_3\| > \frac{\|s_1, s_2\|}{2}$ , yielding  $\|s_0, s_1\| + \|s_2, s_3\| > \|s_1, s_2\|$ . Thus, the disks centered at  $s_1, s_2$ , with radii  $\|s_0, s_1\|, \|s_2, s_3\|$  respectively, overlap. Hence,  $e$  is part of  $SIG$ , provided that  $s_0, s_3$  are the nearest neighbor samples on  $C$  to  $s_1, s_2$ , respectively. This is the case since the disk centered at  $s_1$  with radius  $\|s_0, s_1\|$  intersects  $C$  in the interval  $J[s_0, t]$ , with  $t$  in the interval  $I[s_1, s_2] \in C$ . Thus that disk does not contain any other samples according to Corollary 1. This is proven similarly for  $s_2, s_3$ .
2. Prove that  $e \in DT$ : For the point  $x \in I$  farthest from  $s_1, s_2$ ,  $\|x, s_{[1|2]}\| \geq \frac{\|e\|}{2}$ . Since the curve is  $\rho$ -sampled with  $\rho < 1$ ,  $\text{reach}(I) > \frac{\|e\|}{2}$  and thus  $\text{lfs}(p) > \frac{\|e\|}{2}$  for any  $p \in I$ . If the smallest disk  $D$  including  $e$  is empty of other samples,  $e$  is a Gabriel edge. Since the Gabriel graph  $\subseteq DT$ , this would mean that  $e \in DT$ . We prove  $D$  to contain only  $I$  by contradiction: Assume a point  $q \in C \setminus I$  to exist in  $D$ . Then,  $C \cap D$  is not a topological disk, and therefore  $D$  contains a medial point of

$C$  (Lemma 1). Since  $D$  has radius  $\frac{\|s_1, s_2\|}{2}$  and contains a medial point, there exists a point  $q \in C \cap D$  with  $\text{lfs} < \frac{\|e\|}{2}$  which contradicts above  $\text{lfs} > \frac{\|e\|}{2}$ .

□

Having shown that an edge  $e$  between consecutive samples along  $C$  under  $\rho < 1$  and  $u < 2$  is part of both SIG and  $DT$ , we have proven that the SIGDT contains the reconstruction under the given sampling conditions.

**Corollary 2.** *Since Theorem 1 [OMW16] proves that any  $\epsilon < r$ -sampling is also a  $\rho < r/(1-r)$ -sampling, an  $\epsilon < 0.5$ -sampling is also a  $\rho < 1$ -sampling and contains the reconstruction of  $C$  if  $u < 2$ .*

The above theorem only guarantees the SIGDT to contain the reconstruction edges but may contain additional edges in both its interior and exterior.

#### 3.3.2 Using SIGDT with Connect2D's Dual Operations

We apply two steps from the CONNECT2D algorithm [OM13] to the SIGDT: *inflating* and *sculpting*, in order to improve our reconstruction. These additional steps are summarized below, and they greedily minimize the total edge length of the boundary.

##### Inflating SIGDT to a Manifold

The boundary subset of SIGDT, named  $B$ , contains all vertices either on  $B$  or its interior.  $B$  may thus not be a manifold as it can be pinched at vertices that have  $> 2$  incident boundary edges. Inflating transforms such non-conforming vertices into manifold ones by selecting incident triangles *exterior* to the boundary and adding their edges to  $B$  so that the triangle becomes interior to the boundary. Candidate triangles for all non-conforming vertices are sorted in a priority queue in ascending order by the increase in total boundary length, which is computed by adding the length of new edges and subtracting the length of edges to be removed. We add candidate triangles to non-conforming vertices until they become manifold. However, by adding the edges of new triangles to the graph, some of the edges can become interior to the boundary. Hence, we remove any edge that is not incident to a triangle marked as *outside* (i.e. we remove all edges that are not on the boundary). This procedure is performed in  $\mathcal{O}(n \log n)$  time complexity and guarantees a manifold boundary  $B'$  as its result. More details on the exact implementation and proofs of the theoretical guarantee can be found here [OM13]. The inflating procedure is illustrated in Figure 3.5.

##### Sculpting the Manifold to Interpolate Interior Vertices

The manifold boundary  $B'$  resulting from inflating contains all vertices either on  $B'$  or interior to it. These isolated interior vertices have to be connected to the boundary

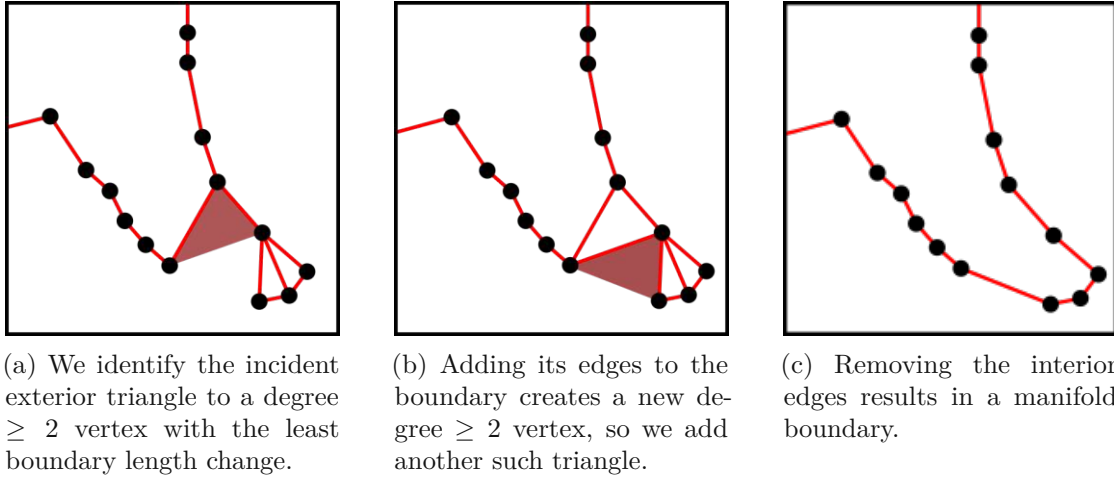


Figure 3.5: Step-by-step *inflating* procedure on a close-up to make the curve manifold.

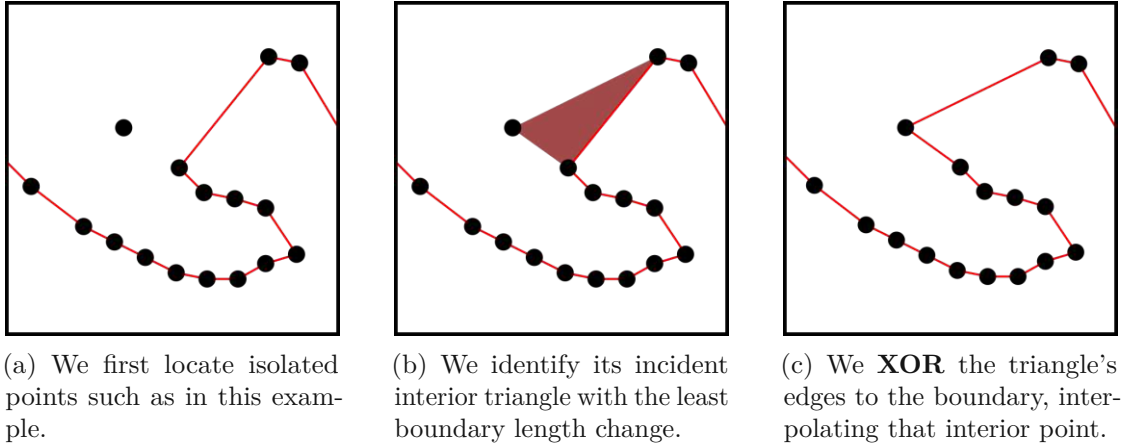


Figure 3.6: Step-by-step *sculpting* procedure on a close-up of a manifold curve to interpolate interior points.

so that the reconstruction can interpolate all the points. Triangles incident to vertices *interior* to  $B'$  are sorted by the same boundary length increase criterion as for inflating. When a candidate triangle is added to  $B'$ , we **XOR** that triangle's edges with  $B'$  (i.e. remove the triangle edge that is already part of the graph and add the other edges that are not yet part of the graph). This step does not increase the overall complexity of the algorithm, being performed in  $\mathcal{O}(n \log n)$  time. This step exposes the interior vertices to the boundary. This fails if the  $DT$  does not contain a Hamiltonian cycle. The details on the procedure and guarantees can be found here [OM13]. The sculpting process is illustrated in Figure 3.6.



#### Eliminate Leaf Vertices (optional)

In the cases where  $C$  is not sampled as densely as required, artefacts such as leaf vertices may appear. In our experiments, we found that this does not affect results in general but eliminating these further improves reconstruction quality for point sets with sharp corners. We apply this optional step to the SIGDT before *inflating*:

We increase the degree of the leaf nodes to two by adding their shortest incident Delaunay edge to SIGDT, forming *SIGDT2*, which has vertex degree  $\geq 2$  everywhere. This step loops over all the vertices and finds the shortest incident edge to each leaf vertex that is not part of the graph yet. This takes  $\mathcal{O}(kn)$  operations, assuming a constant degree  $k$  of vertices, for the average *DT*, ignoring contrived cases which do not arise often in practice.

As an overview, we have described SIG-CONNECT2D as a SIG-based curve reconstruction method that uses an unstructured point set as input and reconstructs the boundary of the shape that the points have been sampled from. SIG-CONNECT2D starts with the SIGDT graph, and then applies *inflating* and *sculpting* on it as in the CONNECT2D algorithm - an overview of this algorithm is presented in Figure 3.3. The next section presents our results compared to state-of-the-art curve reconstruction algorithms.

## 3.4 Results

We have tested our proposed method, SIG-CONNECT2D, against 15 state-of-the-art curve reconstruction algorithms (see names in Figure 3.7) using the 2D Points Curve Reconstruction Survey and Benchmark [OPP<sup>+</sup>21], where these are referenced together with source code and the data sets used for our evaluations below. Note that OPTIMALTRANSPORT has been eliminated from their evaluation since the input it aims to reconstruct is dense, with a high percentage of outliers and noise, and cannot exactly reconstruct clean, sparse inputs (an example of how it fails can be seen in Table 3.1), being also highly dependent on the number of iterations. We have then analyzed results for the reconstruction of manifold curves, curves with sharp corners, and a subset of well-sampled manifold curves in terms of exact reconstruction, and examples can be seen in Figure 3.19. Furthermore, we analyzed our method in terms of the root mean square error (RMSE) to the ground-truth curve for noise-free data sets, noisy data sets, and data sets with outliers to form a thorough evaluation. Also, we compute the overlap of sets of edges of the proximity graphs SIG, *DT*, SIGDT and reconstruction for a large data set.

**Manifold Curves:** We have compared our results against the above-mentioned reconstruction algorithms on 1257 noise-free point sets. These data sets represent a subset of the original benchmark data set since we have chosen to only use ground-truth data sets that interpolate all input points. Our algorithm SIG-CONNECT2D shows the best accuracy (91.5% compared to second-best CONNECT2D with 90.3%). Figure 3.7 shows the improved reconstruction as a visual comparison on manifold data sets, numbers are



given in Table 3.2. An example of a point set fed to all algorithms is shown in Table 3.1 where only our SIG-CONNECT2D reconstructs it correctly.

**Proximity Graphs Overlap:** Using the same data sets, we have computed the average percentage of SIG edges that are also in the  $DT$  and vice-versa, as well as the average percentage of SIGDT edges that are part of the reconstruction and vice versa. The results show that SIG is usually mostly contained in the  $DT$  - 90.9%, especially considering that the nearest neighbor graph is contained in both. However, as expected,  $DT$  contains more edges that are not part of SIG - only 47.2% of  $DT$  edges are in SIG. Furthermore, in practice, even without imposing the sampling and non-uniformity criteria, the reconstruction is contained in SIGDT in almost all cases (99.9%), indicating that our sampling condition covers practical point sets extremely well. This represents the best overlap among tested proximity graphs -  $BC_0$  achieves to contain, on average, 99.6% of the ground-truth edges, while  $kNN$  graphs, for  $k \in \{2, 3, 4\}$ , achieve at most 98.6% ( $2NN$  - 98.4%,  $3NN$  - 98.5% and  $4NN$  - 98.6%). However, SIGDT usually has more edges than the reconstruction - only 76.8% of SIGDT edges are part of the correct reconstruction. These results show that the SIGDT is a good indicator of the proximity of the graph, as it misses almost no edges.

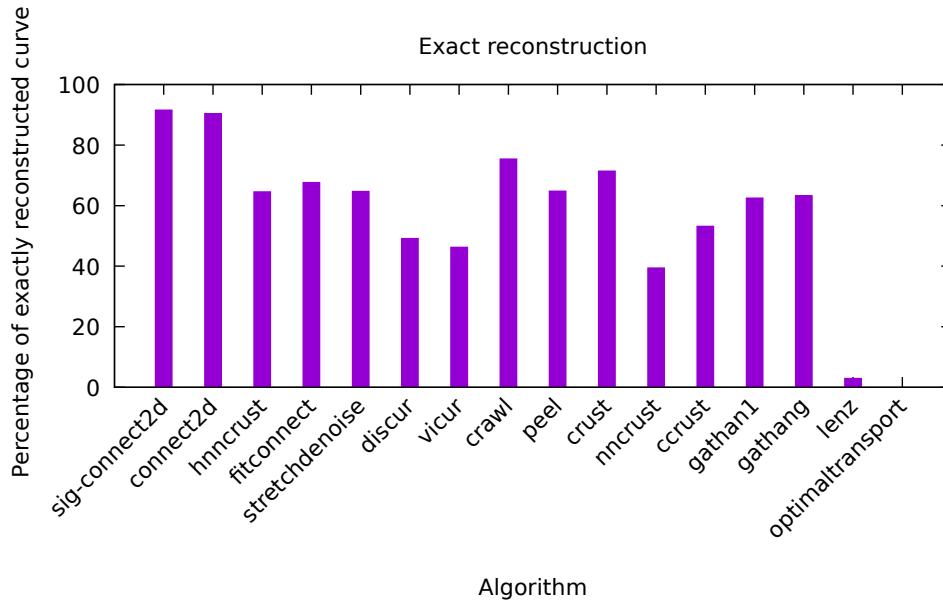


Figure 3.7: Reconstruction of manifold curves.

**Sharp Corner Curves:** We have tested our method on a data set consisting of 47 input sets, comparing it against the same other curve reconstruction methods as above. The best results are achieved by GATHANG [DW02], at 80.9% accuracy of the exact reconstruction, followed by our method at 70.2% - the complete results are presented in Figure 3.8 and Table 3.2. However, these results are expected since GATHANG is

### 3. SIGDT: PLANAR CURVE RECONSTRUCTION

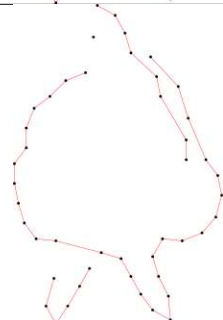
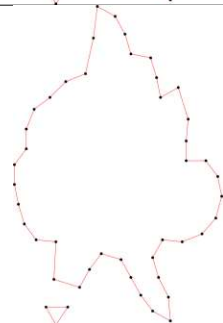
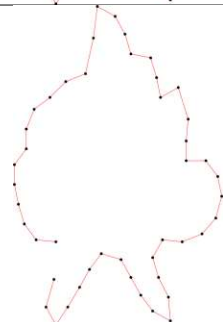
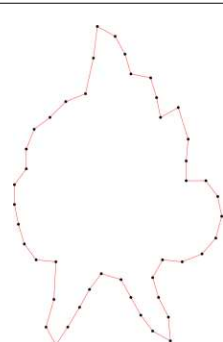
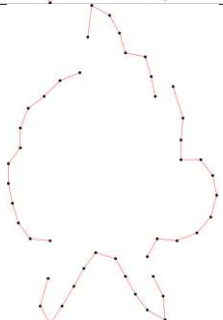
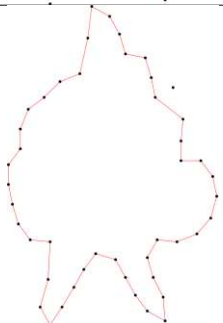
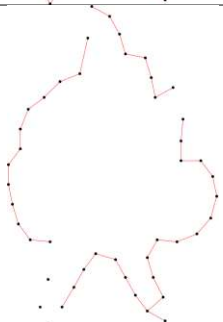
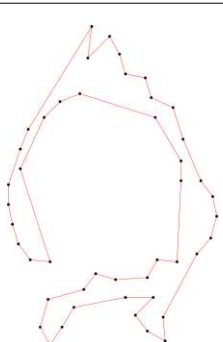
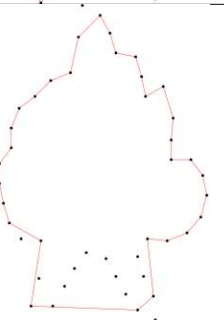
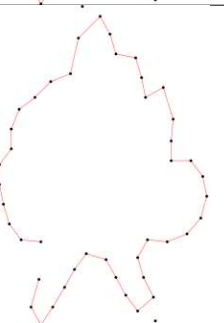
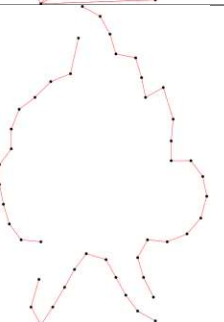
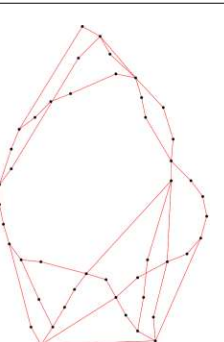
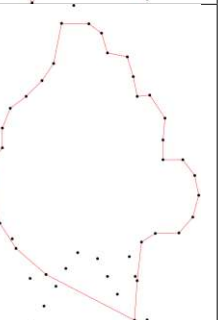
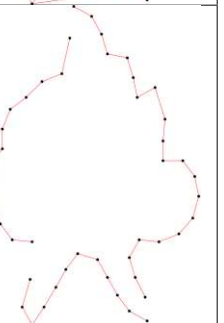
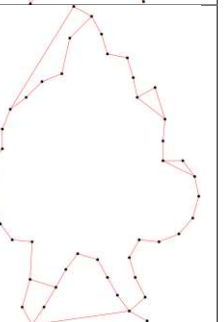
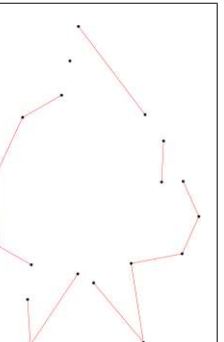
GATHANG	DISCUR	CRUST	SIG-CONNECT2D
			
GATHAN1	VICUR	CCRUST	CONNECT2D
			
FITCONNECT	CRAWL	HNNCRUST	LENZ
			
STRETCHDENOISE	PEEL	NNCRUST	OPTIMALTRANSPORT
			

Table 3.1: The resulting reconstructions for our algorithm compared to the other 15 state-of-the-art algorithms [OPP<sup>+</sup>21] on manifold curve input. Our SIG-CONNECT2D is the only one to correctly reconstruct this point set.

specialized for sharp-corner reconstruction, but performs worse in the general case of manifold curves as seen above.

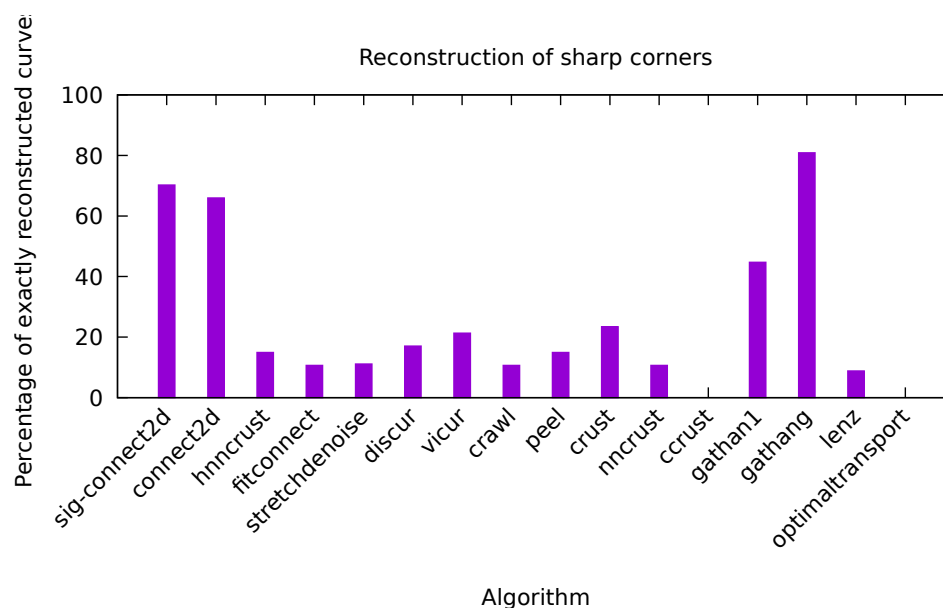


Figure 3.8: Reconstruction of curves with sharp corners.

**Open and Multiple Curves:** Our method is guaranteed to output a manifold reconstruction of the output through the usage of inflating and sculpting[OM13]. For this reason, SIG-CONNECT2D is not suitable for such types of input. However, we provide the symmetric difference in area (computed using BOOST’s `boost_sym_difference`) between the output of different algorithms and the correct output. Our method interpolates the input points and does not achieve exact open curves or multiple curves as expected, but the results are still similar to the expected output, as visible in the symmetric difference results in Figure 3.9 and Figure 3.10, and examples are presented in Figure 3.20.

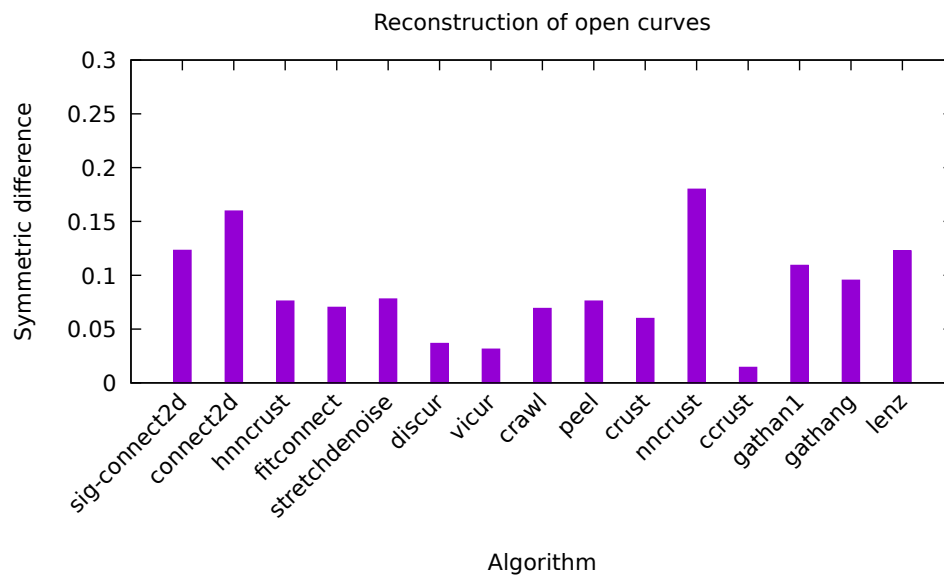


Figure 3.9: Symmetric area difference for open curves.

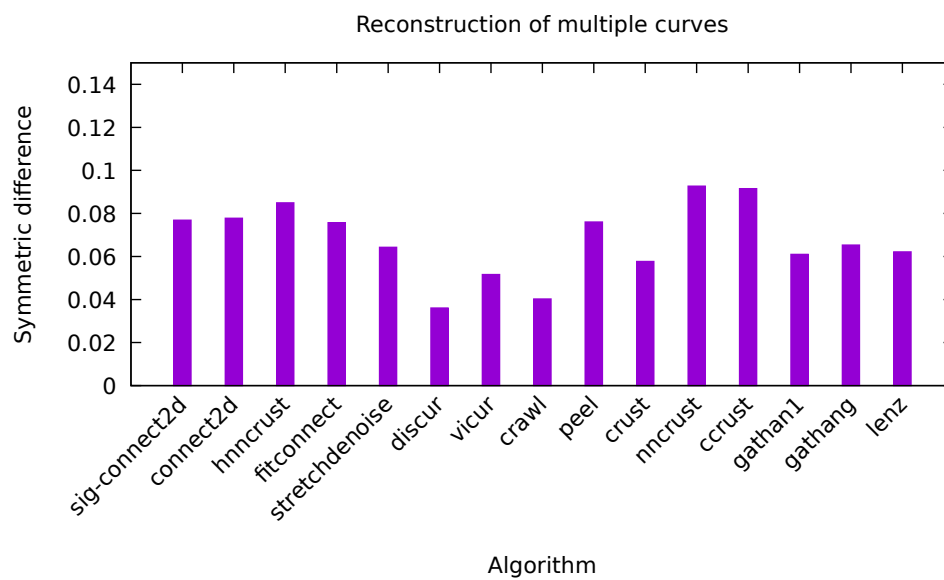


Figure 3.10: Symmetric difference of area for multiple curves.

Algorithm	Manifold	Sharp	Open	Multiple	$\epsilon < 1$	Manifold runtime (ms)	Large data set runtime (ms)
SIG-CONNECT2D	<b>91.5%</b>	70.2%	0.0%	3.7%	<b>95.8%</b>	0.9	75
CONNECT2D	90.3%	65.9%	0.0%	0.0%	95.0%	0.5	99
HNNCRUST	64.5%	14.8%	43.4%	53.7%	67.2%	0.6	49
FITCONNECT	67.7%	10.6%	8.6%	22.2%	71.0%	237.1	-
STRETCHDENOISE	64.7%	11.1%	9.5%	24.0%	68.0%	207.4	-
DISCUR	49.0%	17.0%	39.1%	46.2%	50.7%	279.7	-
VICUR	46.2%	21.2%	<b>52.1%</b>	46.2%	47.6%	357.5	-
CRAWL	75.3%	10.6%	21.7%	40.7%	78.2%	0.4	182
PEEL	64.7%	14.8%	43.4%	<b>57.4%</b>	67.3%	2.8	2455
CRUST	71.3%	23.4%	43.4%	38.8%	74.3%	0.6	39
NNCRUST	39.3%	10.6%	8.6%	12.9%	41.3%	0.1	11
CCRUST	53.1%	0.0%	30.4%	22.2%	55.7%	0.8	796
GATHAN1	62.4%	44.6%	13.0%	24.0%	65.3%	0.3	18
GATHANG	63.2%	<b>80.8%</b>	21.7%	35.1%	65.7%	1.9	293
LENZ	2.8%	8.8%	0.0%	0.0%	3.0%	8.2	12672
OPTIMALTRANSPORT	0.0%	0.0%	0.0%	0.0%	0.0%	71.2	542

Table 3.2: Results for precision of exact reconstruction of curves with different characteristics and average runtime in milliseconds, as well as the runtime for a large point set with 9991 points, for which not all algorithms managed to produce an output.

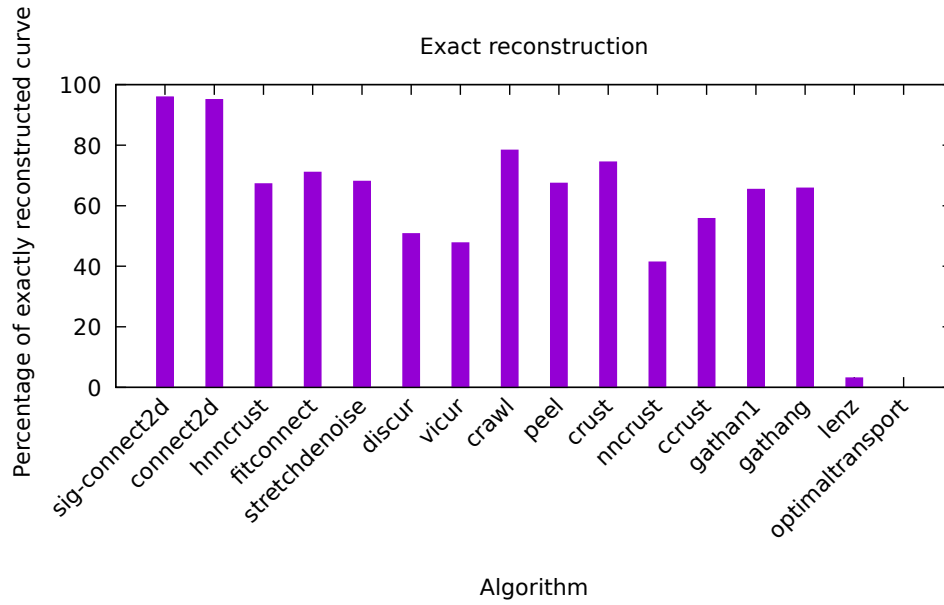


Figure 3.11: Reconstruction accuracy for graph-based  $\epsilon < 1$ .

**Well-Sampled Manifold Curves:** Since some of the ground-truth curves in the data set are sparsely sampled, and therefore cannot be reconstructed well by any algorithm, we have selected a subset of the 1257 data sets from the manifold curve test set. We filter 1183 data sets  $\epsilon$ -sampled with  $\epsilon < 1$  based on local feature size computed on graphs and repeat the above comparison on manifold curves. Our algorithm performs best at 95.8%, followed by the original CONNECT2D at 95.0%, showing that the SIG captures the connectivity better for well-sampled curves, and comes quite close to reconstructing all curves sampled with graph-based  $\epsilon < 1$  in practice. Results are presented in Figure 3.11 and in Table 3.2.

**Runtime:** We have specified the time complexity of all steps of our method in the respective descriptions, and their upper bound is  $\mathcal{O}(n \log n)$  in terms of the  $n$  input points - typical for the Delaunay-based curve reconstruction algorithms. The average runtime of each algorithm run on the complete data set of 1257 point sets on an Intel Core i7-7700HQ processor is presented in Figure 3.12 and in Table 3.2. The empirical results ( $\approx 1$  ms per point set, with an average of 260 points) confirm the theoretical bounds on the time complexity and are in line with the other methods except fastest NN-CRUST. We have also tested on a large point set with 9991 points, which takes 75ms (Table 3.2), indicating runtime is almost linear.

**RMS error:** We have tested how closely our reconstruction approximates a cubic Bézier curve by sampling it with different  $\epsilon$  values and computing the RMS error between the reconstructed curve and the original (see Figure 3.13). Our algorithm performs similarly to the majority of evaluated algorithms. Furthermore, for the same setup of  $\epsilon$ -sampling a cubic Bézier curve with  $\epsilon = 0.1, 0.2, 0.3, 0.4, 0.5$ , we have tested our algorithm on 20

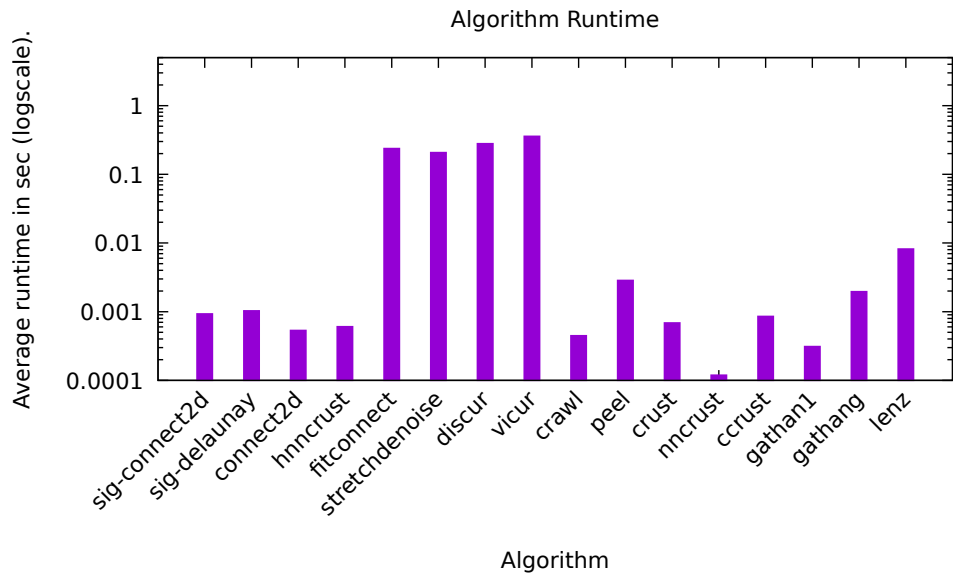


Figure 3.12: Average runtime of manifold curve reconstruction.

differently generated point sets for each  $\epsilon$  value by varying the starting sample on the curve. This shows that the theoretical guarantee of the SIGDT includes the accurate reconstruction in all cases, even without constraining  $u$ .

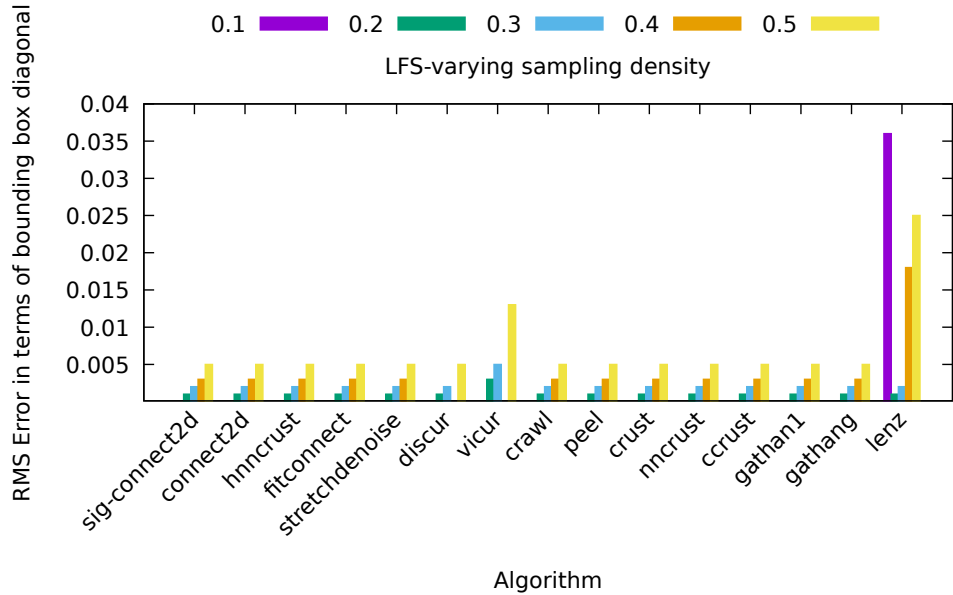


Figure 3.13: RMSE of reconstructions for varying  $\epsilon$ -sampling.

**Noise:** Even if our method is designed for non-noisy input, we have tested its reliability

### 3. SIGDT: PLANAR CURVE RECONSTRUCTION

against noise by computing the RMS error against the ground truth. We have added uniform noise to some of the input curves and run the reconstruction algorithms. These results are visible in Figure 3.15 and indicate that our method performs similarly to CONNECT2D, as expected. Another way of testing the resilience to noise was by adding lfs noise on samples along a cubic Bézier curve. The results are competitive, as shown in Figure 3.16. We also present the output of running the algorithm on noisy curves in Figure 3.14, which achieves, as expected, an interpolation of the input points.

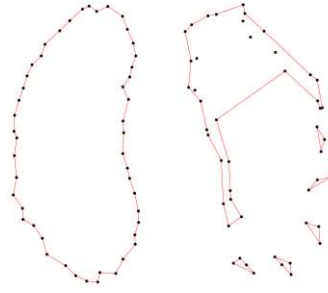


Figure 3.14: Reconstruction of data sets perturbed with uniform noise as a percentage of the bounding box diagonal. The left data set uses 0.01% uniform noise, and the original shape is correctly reconstructed, while the right one uses 0.03%, thus failing to recreate the ground truth as the sampling becomes too sparse.

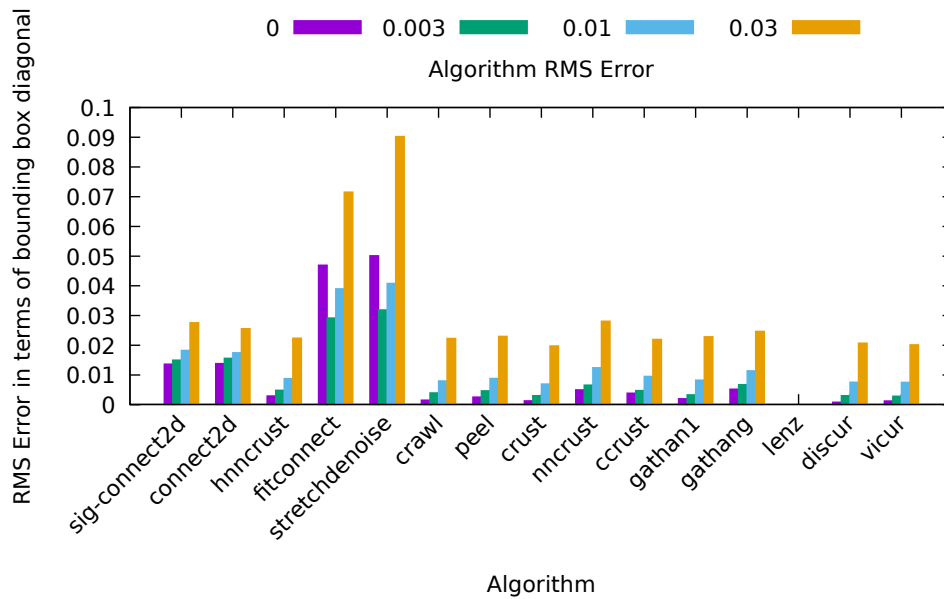


Figure 3.15: RMSE of reconstructed curves from inputs contaminated with uniform noise.

**Outliers:** We have tested our algorithm's reliability when outliers are present by adding a percentage of outliers to some of our input curves. The results are in line with the



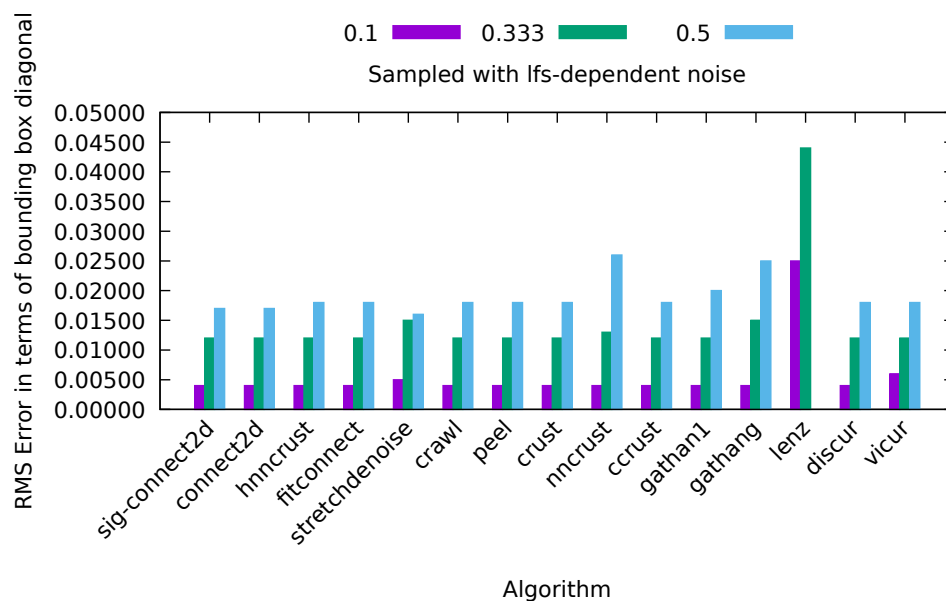


Figure 3.16: RMSE of curves from inputs with lfs-based noise.

majority of algorithms and are displayed in Figure 3.17.

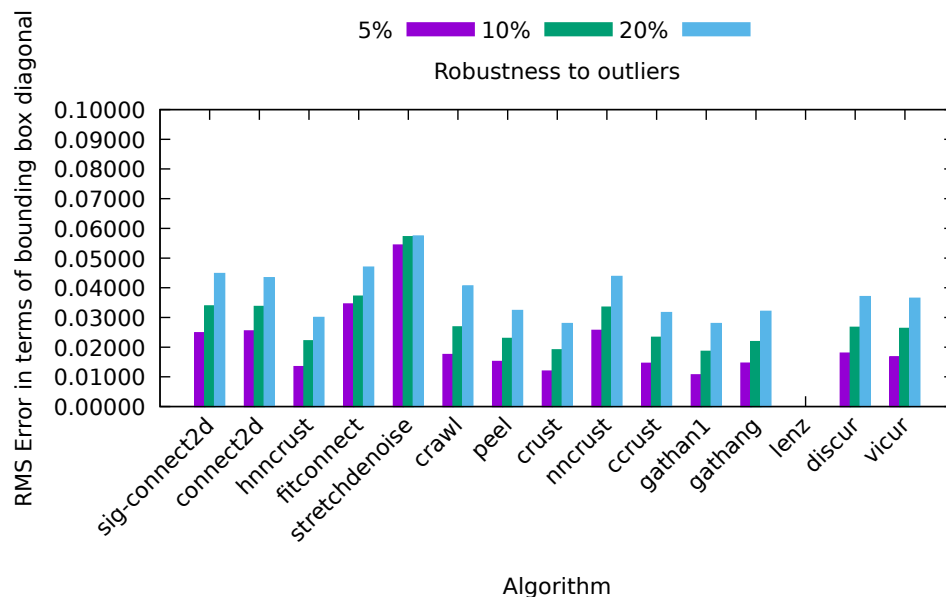


Figure 3.17: Reconstruction of curves with varying outlier share.

**Limitations:** We present in Figure 3.21 some of the cases where our algorithm fails to produce the exact reconstruction of the input data. However, most of the points of failure are represented by multiple components and non-uniform sampling that go beyond the

theoretical limits of our method. These cause the algorithm to try to create a boundary interpolating all components together or to fall into local minima when geodesically far samples become geometrically close. However, for non-manifold curves, we provide an analysis of the symmetric difference of area between the results of various algorithms and the ground truth in Figure 3.18. Our method fails to produce exact reconstruction of such curves, but the results are close to the original curve, as illustrated in Figure 3.20.

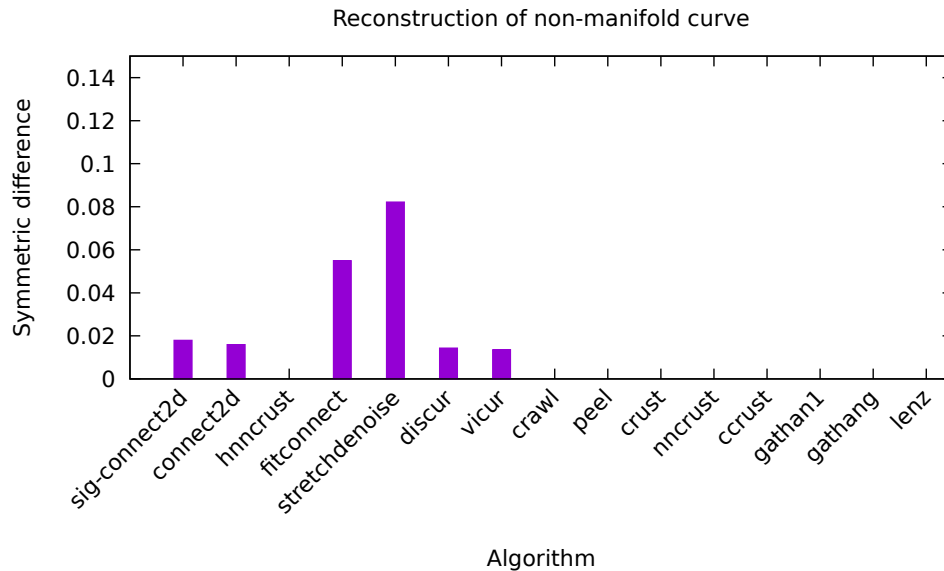
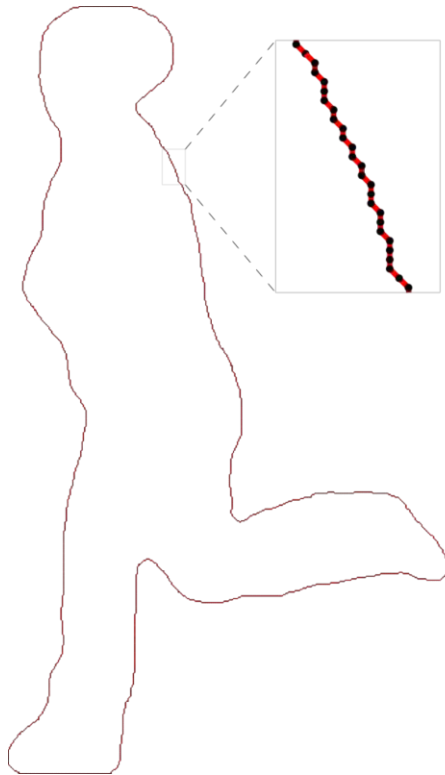


Figure 3.18: Symmetric area difference for non-manifold curves.

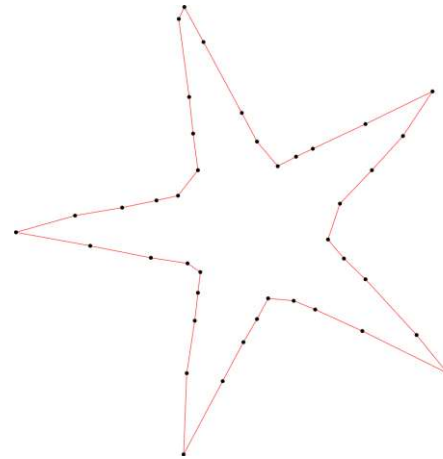
### 3.5 Conclusion and Future Work

We propose a new proximity graph in 2D,  $SIGDT = SIG \cap DT$ , and show that it better captures connectivity between points than other proximity graphs, and does so without requiring a specific number of neighbors as a parameter, as  $kNN$  would. We prove that SIGDT contains the reconstruction for planar curves for some enhanced sampling conditions. Together with filtering steps for redundant edges from an existing method, our method SIG-CONNECT2D correctly reconstructs the manifold boundary of the input set in more cases than the state-of-the-art, and reconstructs almost all well-sampled point sets. As current results are promising, they encourage further improvements in this approach. Hence, our future work includes:

- Extending SIG-CONNECT2D to robustly reconstruct multiple curves, open curves, and non-manifold inputs;
- Extending SIGDT to 3D and SIG-CONNECT2D to surface reconstruction.



(a) Reconstruction of dense curves - 1712 input points.

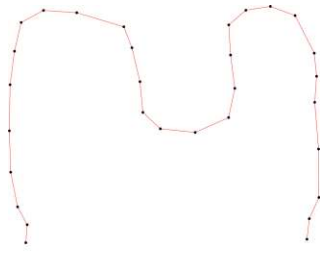


(b) Reconstruction of sharp corner curves.

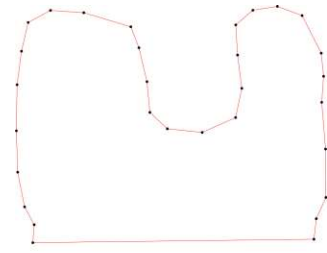


(c) Reconstruction of curves from silhouette images extracted from image databases.

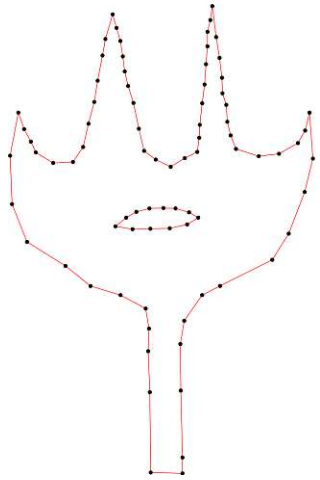
Figure 3.19: Reconstruction of different types of curves.



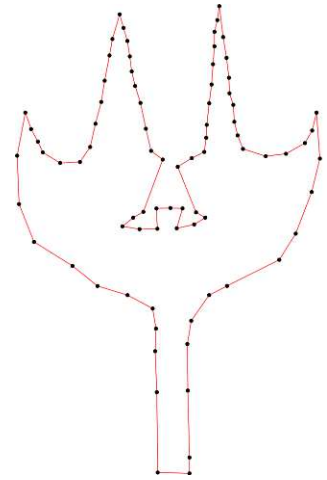
(a) Desired open curve.



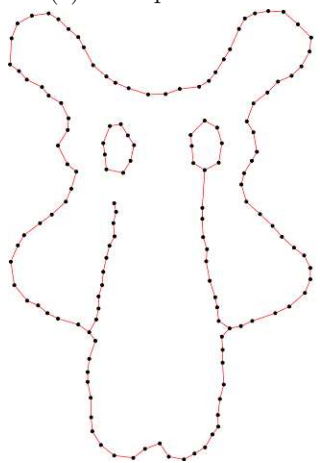
(b) Our reconstruction.



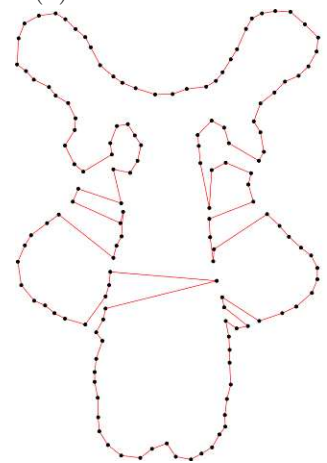
(c) Multiple curves.



(d) Our reconstruction.

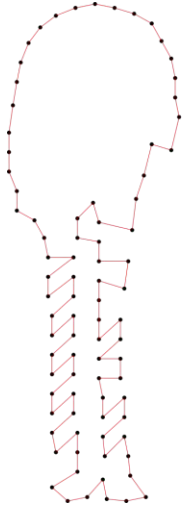


(e) Non-manifold curves.

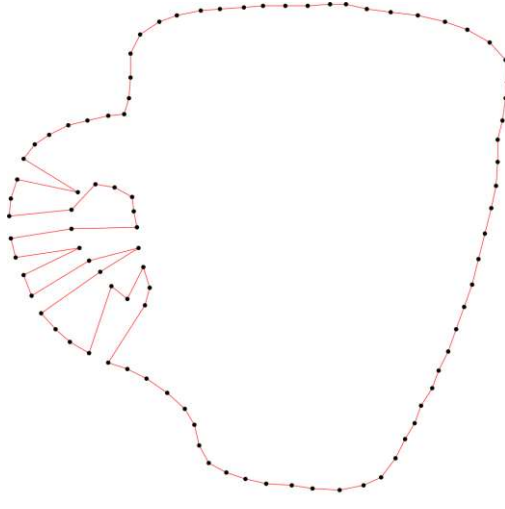


(f) Our reconstruction.

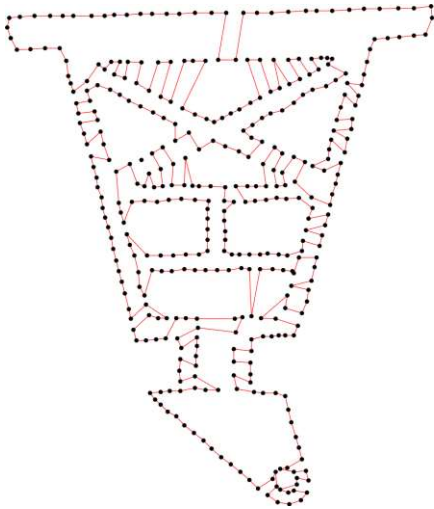
Figure 3.20: Reconstruction of open, multiple and non-manifold curves. Our method is not designed for these categories since we expect the output to be a manifold curve that interpolates all the input points. However, the results are similar to the expected output.



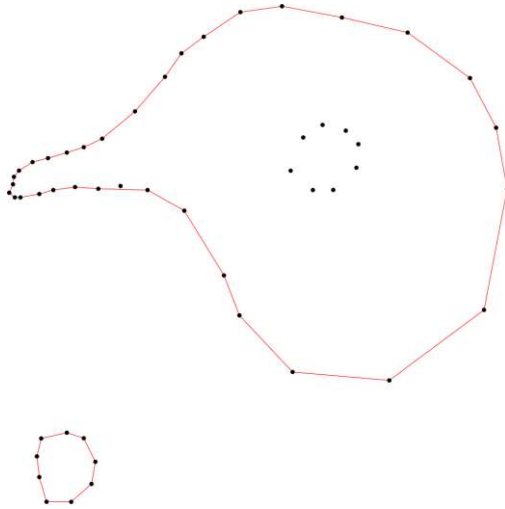
(a) Too sparse sampling for close curves causes the algorithm to fall into local minima.



(b) Nested components fail since *Inflate* and *Sculpt* assume a single boundary interpolating all points if they are spaced closely enough.



(c) The non-uniform sampling prevents SIGDT from containing all the required edges and the multiple disconnected elements fail.



(d) Some of the input points are not interpolated at all, since *Sculpt* cannot handle nested components.

Figure 3.21: Failure cases for SIG-CONNECT2D.



# SIGDV: Reconstructing Curves on Riemannian Manifolds

We leave the familiar planar setting for our curve reconstruction quest and we move to more complex domains - Riemannian manifolds. We extend and adapt our work presented in Chapter 3 to manifolds, extending both the theory and the algorithm to new applications. The contents of this chapter have been adapted from the paper “*Reconstructing Curves from Sparse Samples on Riemannian Manifolds*”, published in the Computer Graphics Forum and presented at the Symposium on Geometry Processing 2024 [MMM<sup>+</sup>24].

## 4.1 Overview

Reconstructing 2D curves from sample points has long been a critical challenge in computer graphics, finding essential applications in vector graphics. The design and editing of curves on surfaces has only recently begun to receive attention, primarily relying on human assistance, and where not, limited by very strict sampling conditions. In this work, we formally improve on the state-of-the-art requirements and introduce an innovative algorithm capable of reconstructing closed curves directly on surfaces from a given sparse set of sample points. We extend and adapt the planar curve reconstruction method introduced in Chapter 3 to the realm of surfaces while dealing with the challenges arising from working on non-Euclidean domains. We demonstrate the robustness of our method by reconstructing multiple curves on various surface meshes. We explore novel potential applications of our approach, allowing for automated reconstruction of curves on Riemannian manifolds.

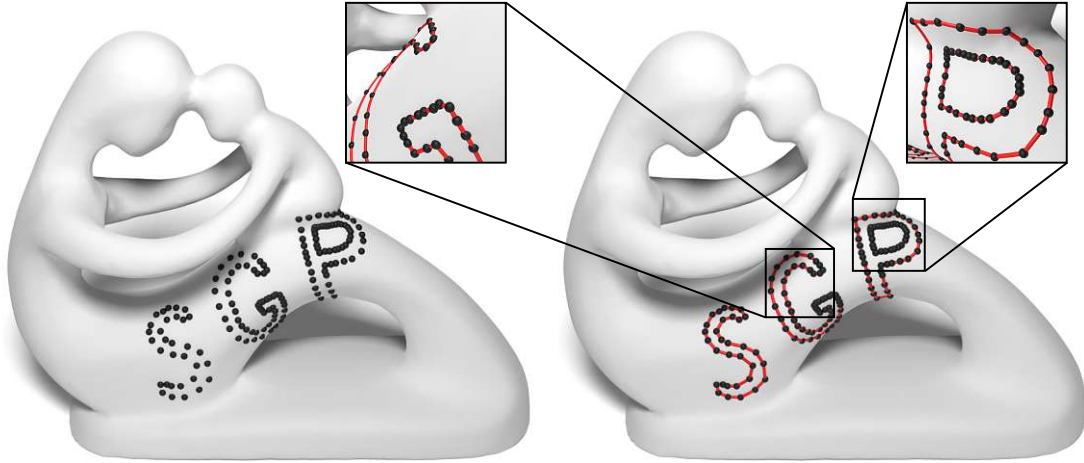


Figure 4.1: Reconstruction of multiple curves on the fertility mesh. The samples (left) are denser where the local feature size is small - around the serif of the  $G$  for example, matching our sampling condition. Our method is able to reconstruct multiple closed curves at the same time (right), with sharp features and close sheets.

## 4.2 Introduction

Vector graphics represents an important research area in computer graphics, and it is widely applied in many fields, spanning from design and art to engineering. One of the important reasons for its success is the ability to generate infinite resolution smooth complex visualizations with relative ease while requiring only little input geometry.

Recently, an increasing interest has been devoted to moving 2D vector graphics onto surfaces, trying to address certain issues stemming from texturing methods. Texturing is a well-established approach for defining patterns and decorations on surfaces, but it generally relies on finite-resolution images and parameterization. The latter is not always available or could be difficult or expensive to define. Procedural textures try to overcome these problems, defining patterns via mathematical functions and algorithms. However, they generally rely on multi-dimensional noise functions that are then sampled on the surface [EMP<sup>+</sup>03, Har01, MBMR22]. These algorithms are usually agnostic of the underlying geometric properties and can incur high computation times.

Besides works generalizing sample-based texture synthesis to triangular meshes [WL01, Tur01], only in recent years some solutions have been proposed which try to leverage properties of non-Euclidean metric spaces and define patterns directly on surfaces, either via recursive structures [NPP21] or simulated behaviors [MMMR22]. Another avenue of development for texture synthesis is represented by neural networks that generate a textured mesh in the style of an input image [MG23].

Despite the existence of various solutions for decorating surfaces, the problem of constructing lines and curves on discrete manifolds has not been addressed satisfactorily yet. Defin-



ing curves and shapes directly on surfaces is innovative for design applications [PSOA18], and has a relevant impact in the processing of archaeological data [KST08, GTSK13] by extracting specific decorations from the models. However, little research has been devoted to improving the definition and the reconstruction of curves on discrete surfaces, besides efforts to generalize Bézier curves [MNPP21]. Furthermore, the existing techniques are generally centered around human interaction, as they are designed to be tools for artists and end users, and even the most recent solutions require an ordered sequence of samples [MP23]. To the best of our knowledge, Shah *et al.* [SC13] provided the first and only solution for dealing with curve reconstruction on Riemannian manifolds. Still, their proposed method can only deal with dense uniform samplings and only guarantees to reconstruct curves in limited settings. By generalizing state-of-the-art theoretical results and algorithmic solutions for planar curve reconstruction to arbitrary manifold domains, we introduce a more robust method that, given an unordered collection of points over a Riemannian manifold that follows our relaxed sampling conditions, always produces an ordered sequence identifying a closed curve on the surface (see Figure 4.1).

Our contributions are summarized in the following:

- we propose a solution that extends existing state-of-the-art theory and techniques for 2D curve reconstruction to manifold domains, overcoming the challenges arising from translating the problem into non-Euclidean spaces;
- we improve the state-of-the-art sampling conditions for curve reconstruction on Riemannian manifolds [SC13], allowing for sparser sampling for curve reconstruction on manifold domains;
- we perform a qualitative study of our method on real-world data coming from established applications, where the previous solution fails due to the sparsity of the samples.

### 4.3 Background and notations

Informally speaking, a  $d$ -dimensional manifold is a collection of pieces of the  $d$ -dimensional hyperspace that are deformed and glued together to form a continuous smooth domain. Such domain can be embedded in a higher dimensional space (like in the case of 2-dimensional surfaces embedded in 3D space), or exist on their own (like the 3D rotation group  $SO(3)$  [HH13]). For a formal and complete definition of Riemannian manifolds, we refer to the books by Morita and Do Carmo [Mor01, DC16].

To refer to any such  $d$ -dimensional manifold (including continuous curves) we use calligraphic letters (*i.e.*,  $\mathcal{M}$ ,  $\mathcal{C}$ ), and we denote the geodesic distance function over a manifold  $\mathcal{M}$  with  $d_{\mathcal{M}}$ . Furthermore, we refer to any discrete representation of a manifold  $\mathcal{M}$  with the symbol  $\hat{\mathcal{M}}$ .

Specifically, we represent discrete 2-dimensional surfaces as triplets  $\hat{\mathcal{M}} = (V, E, T)$ , where  $V$  is a set of vertices,  $T$  is a set of oriented triangles among vertices, and  $E$  is a set of unordered edges induced by the triangles in  $T$ . We also represent discrete curves (*i.e.*,

1-dimensional manifolds) as tuples  $\hat{\mathcal{C}} = (V, E)$ , where  $V$  is a set of vertices and  $E$  is a set of edges between vertices. Throughout this work, we assume the manifoldness of discrete surfaces and curves [CDGDS13], and we assume to work with curves made by a single component and forming a closed non-self-intersecting loop.

### 4.3.1 Curve sampling

In this section, we mention the definitions related to sampling planar curves that relate to our contribution, and conclude with our own definitions that extend to manifolds.

Given a smooth curve  $\mathcal{C}$  and a set of samples  $S \subset \mathcal{C}$ , reconstructing the curve is the process of constructing a discrete curve  $\hat{\mathcal{C}} = (S, E)$  such that an edge  $e = (s_i, s_j)$  is in  $E$  if and only if  $s_i$  and  $s_j$  are consecutive samples along the curve  $\mathcal{C}$ .

Sampling techniques include distance-based sampling, such as uniform - Definition 1, and non-uniform - Definition 2, and local-feature size-based sampling, such as  $\epsilon$ -sampling (Definition 5) and  $\rho$ -sampling (Definition 7).

### 4.3.2 Differential geometry

We now lift the definitions required for the local feature size from the plane to manifolds, to be able to introduce sampling conditions on manifolds in the subsequent sections.

Euclidean disks are widely used in relation to planar curve reconstruction, and they have a straightforward generalization to arbitrary metric spaces.

**Definition 1.** Let  $(\mathcal{M}, d_{\mathcal{M}})$  be a metric space,  $p \in \mathcal{M}$  a point on  $\mathcal{M}$ , and  $r \in \mathbb{R}$ . The  $r$ -ball centered at  $p$  is the region  $\mathcal{B}_{p,r}$  of points at a distance less than  $r$  from  $p$ :

$$\mathcal{B}_{p,r} = \{x \in \mathcal{M} : d_{\mathcal{M}}(p, x) < r\} . \quad (4.1)$$

The closure  $\text{cl}\mathcal{B}_{p,r}$  of the  $r$ -ball also includes its boundary:

$$\partial\mathcal{B}_{p,r} = \{x \in \mathcal{M} : d_{\mathcal{M}}(p, x) = r\} . \quad (4.2)$$

The work of Shah *et al.* [SC13] identifies the limitations of the local feature size in non-Euclidean metric spaces, and exploits the notions of *cut locus* and *injectivity radius* to overcome these issues. Informally speaking, the cut locus of a point  $p$  on the manifold  $\mathcal{M}$  can be seen as the set of all points  $q$  such that there are at least two distinct minimizing geodesics from  $p$  to  $q$  – see Figure 4.2.

**Definition 2.** Let  $\mathcal{M}$  be a  $d$ -dimensional Riemannian manifold, possibly with boundary  $\partial\mathcal{M}$ , and equipped with a connection defining an exponential map  $\exp_p : T_p(\mathcal{M}) \rightarrow \mathcal{M}$  at every point  $p \in \mathcal{M}$ . The cut locus of  $p$  in the tangent space  $T_p(\mathcal{M})$  is defined as the set  $C_{T(\mathcal{M})}(p)$  of all vectors  $v \in T_p(\mathcal{M})$  such that the parametric curve  $\exp_p(tv)$  is a minimizing geodesic for  $t \in [0, 1]$  and not minimizing for  $t > 1$ .

The cut locus of  $p$  on the manifold  $\mathcal{M}$  is the set  $C_{\mathcal{M}}(p)$  of all points  $q \in \mathcal{M}$  that are the image of a vector  $v \in C_{T(\mathcal{M})}$  under the exponential map.

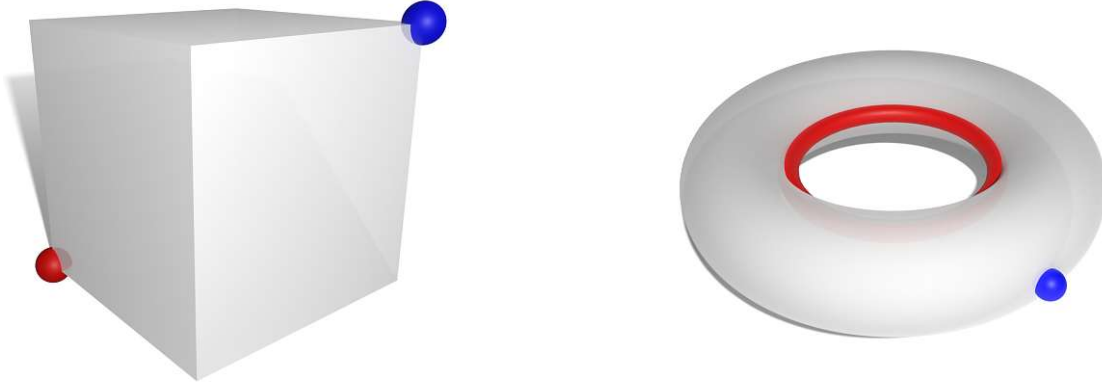


Figure 4.2: Examples of cut locus (in red) given a point (in blue) on different surfaces. The cut locus can be a single point (left) or an entire curve (right).

Intuitively, the injectivity radius is the maximum size of an  $r$ -ball around a point that preserves injectivity when mapped the Euclidean space – refer to Theorem 1 and Figure 4.5.

**Definition 3** ([DC16]). *Let  $p \in \mathcal{M}$  be a point on the manifold  $\mathcal{M}$  and  $C_{\mathcal{M}}(p)$  be its cut locus. The injectivity radius of  $p$  is*

$$i_{\mathcal{M}}(p) = \inf_{q \in C_{\mathcal{M}}(p)} d_{\mathcal{M}}(p, q). \quad (4.3)$$

The injectivity radius of the manifold  $\mathcal{M}$  is thus defined as

$$\mathfrak{I}_{\mathcal{M}} = \inf_{p \in \mathcal{M}} i_{\mathcal{M}}(p). \quad (4.4)$$

By using these tools, Shah *et al.* [SC13] show that curves can be reconstructed using a minimum spanning tree if they are uniformly  $\vartheta$ -sampled, with  $\vartheta < \min(\inf_{p \in \mathcal{C}} \text{lfs}(p), \mathfrak{I}_{\mathcal{M}})$ .

## 4.4 Method

In this chapter, we advance the task of reconstructing curves on manifolds, addressing curves that are sampled more sparsely and non-uniformly compared to the state-of-the-art.

The standard notion of local feature size is not suitable for non-Euclidean metric spaces. In Figure 4.3 we show two examples of curves where the local feature size presents undesired behaviors. For instance, we can have  $r$ -balls that contain the entire curve without containing the medial axis, or even curves that do not have a medial axis, making it impossible to define the local feature size.

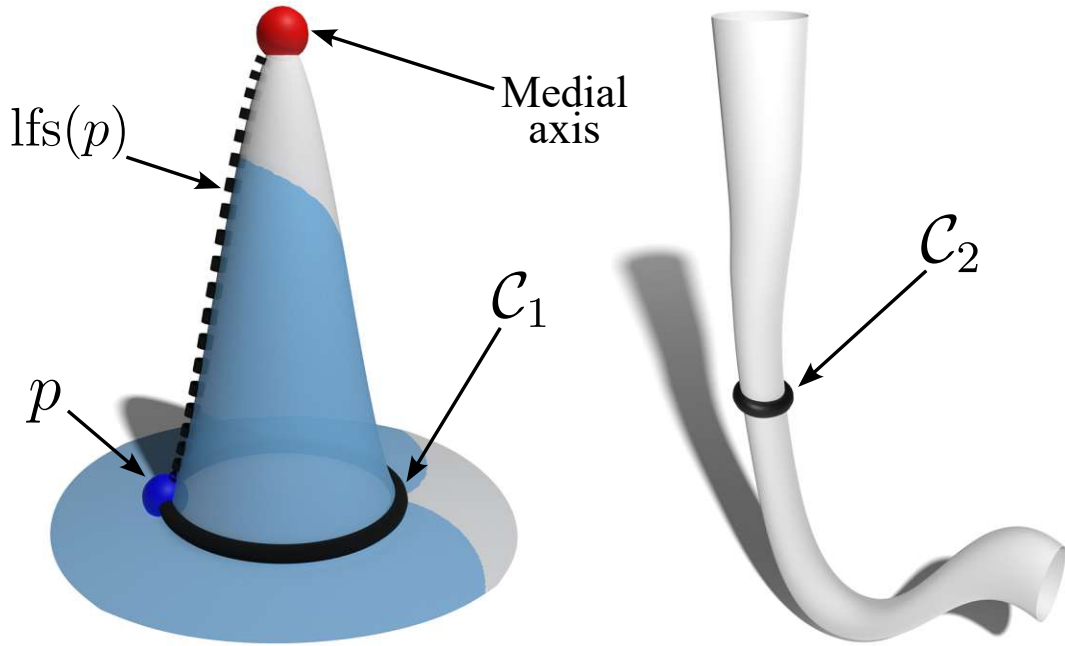


Figure 4.3: Two examples of curves on surfaces (solid black) where the local feature size exhibits undesired behaviors. Left: an  $r$ -ball (shaded in light blue) around  $p$  that contains the entire curve  $C_1$ , but no point of the medial axis. Right: a curve  $C_2$  on the surface with an empty medial axis (*i.e.*, the local feature size is undefined).

By taking inspiration from Shah *et al.* [SC13], we exploit the injectivity radius to strengthen the definition of  $\rho$ -sampling, making it suitable for manifold domains and showing that we can preserve its fundamental properties (Section 4.4.1). Then, we generalize a proximity graph successfully used for 2D curve reconstruction to arbitrary metric spaces, proving that analogous sampling conditions apply (Section 4.4.2). And finally, we present an algorithm that relies on the properties of our graph to reconstruct curves on Riemannian manifolds (Section 4.4.3).

#### 4.4.1 Non-uniform sparse sampling on surfaces

The solution proposed by Shah *et al.* [SC13] consists of bounding the local feature size to the injectivity radius of the entire surface. While this approach effectively solves the previous issues, it also makes the sampling very dense when the manifold contains very narrow sections or sharp features: the smallest injectivity radius of the surface affects the overall sampling conditions, even if the curve might not pass close to these regions. Instead, we use the local feature size and the reach by considering the injectivity radius of individual points. In this way, we can define properties inside topologically flat regions and ensure that they behave similarly to a Euclidean space, avoiding cases where the topology of the  $r$ -ball introduces pathological issues – see the left frame of Figure 4.3, where the local feature size is larger than the injectivity radius.

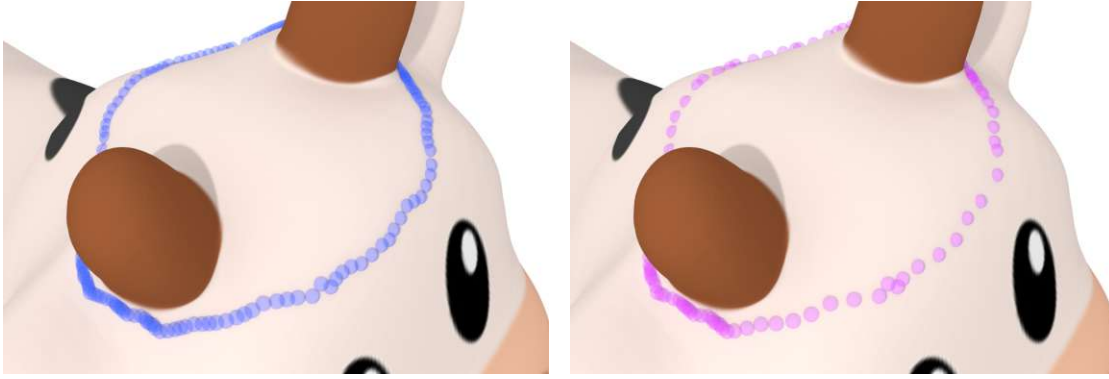


Figure 4.4: A comparison between a dense uniform sampling, with 281 samples, satisfying the conditions described by Shah *et al.* [SC13] (on the left) and a non-uniform  $\rho$ -sampling scheme with only 124 samples (on the right).

**Definition 4.** Let  $\mathcal{C} \subset \mathcal{M}$  be a closed curve on the manifold  $\mathcal{M}$ . For every point  $p \in \mathcal{C}$ , we define the injective local feature size of  $p$  as

$$\text{ilfs}(p) = \min(\text{lfs}(p), i_{\mathcal{M}}(p)) . \quad (4.5)$$

For every interval  $I \subset \mathcal{C}$  of the curve, we extend the injective reach of  $I$  as

$$\text{ireach}(I) = \min_{p \in I} \text{ilfs}(p) . \quad (4.6)$$

From Definition 4, the extension of  $\rho$ -sampling to its injective counterpart follows naturally. The example from Figure 4.4 shows the difference between the uniform dense sampling required for the method from Shah *et al.* [SC13] against a more relaxed non-uniform  $\rho$ -sampling scheme.

The sample sets in Figure 4.4 were computed from a discrete, very dense initial curve by approximating the medial axis, and implicitly, the local feature size, using the Voronoi diagram [DZ04]. We then employed a backtracking approach to extract a subset of samples that satisfy our sampling conditions. Generating an exact and minimal sampling remains an open problem.

**Curve properties.** We start from a classical result in differential geometry about the injectivity radius.

**Theorem 1** ([Kli95]). Let  $\mathcal{M}$  be a  $d$ -dimensional Riemannian manifold. For every point  $p \in \mathcal{M}$ , the restriction of the exponential map  $\exp_p$  to  $U \subset T_p(\mathcal{M})$ , such that  $\exp_p(U) = \mathcal{B}_{p,r}$ ,  $\mathcal{B}_{p,r}$  is injective for all  $r \leq i_{\mathcal{M}}(p)$  and it is a diffeomorphism onto its image.

From this result, we can deduce that if  $r \leq i_{\mathcal{M}}(p)$ , then there exists a diffeomorphism between the  $r$ -ball  $\mathcal{B}_{p,r}$  centered at  $p$  and the Euclidean  $d$ -dimensional ball  $\mathbb{B}^d$ , as we can see in the example depicted in Figure 4.5. We can thus generalize the following important result that holds in 2D:

**Lemma 1** ([ABE98]). *Let  $\mathcal{C} \subset \mathbb{R}^2$  be a closed curve in the plane. For every point  $p \in \mathbb{R}^2$  and every  $r \in \mathbb{R}^+$ , if the Euclidean disk  $D$  of radius  $r$  and centered at  $p$  contains at least two points of  $\mathcal{C}$ , then either  $D \cap \mathcal{C}$  is a topological 1-disk, or  $D$  contains a point of  $\Gamma(\mathcal{C})$ , or both.*

For Riemannian manifolds of arbitrary dimensions we get:

**Lemma 2.** *Let  $\mathcal{C} \subset \mathcal{M}$  be a closed curve on the manifold  $\mathcal{M}$ . For every point  $p \in \mathcal{M}$  and every positive real value  $r \leq i_{\mathcal{M}}(p)$ , if the  $r$ -ball  $\mathcal{B}_{p,r}$  centered at  $p$  contains at least two points of  $\mathcal{C}$ , then either  $\mathcal{B}_{p,r} \cap \mathcal{C}$  is a topological 1-disk, or  $\mathcal{B}_{p,r}$  contains a point of  $\Gamma(\mathcal{C})$ , or both.*

*Proof.* We denote the intersection between the  $r$ -ball and the curve with  $\mathcal{F} = \mathcal{B}_{p,r} \cap \mathcal{C}$ .

If  $\mathcal{F}$  is a topological 1-disk, there is nothing to prove.

If  $\mathcal{F} \neq \mathcal{C}$  and  $\mathcal{F}$  is not a topological 1-disk, then  $\mathcal{F}$  must contain at least two connected components. Let  $q_1 \in \mathcal{F}$  be the closest point in  $\mathcal{F}$  to  $p$ , and let  $f_1$  be the connected component of  $\mathcal{F}$  containing  $q_1$ . If  $q_1$  is not unique, then  $p \in \Gamma(\mathcal{C})$  and the proof is complete. Otherwise, let  $f_2 \neq f_1$  be a second closest connected component of  $\mathcal{F}$  to  $p$ , and let  $q_2 \in f_2$  be the point in  $f_2$  closest to  $p$ . Let  $\delta_i(x) = \inf_{y \in f_i} d_{\mathcal{M}}(x, y)$  be the distance from each point  $x$  to the connected component  $f_i$ . Let us consider the geodesic shortest path  $\gamma_p^{q_2}$ , which is contained in  $\mathcal{B}_{p,r}$  by definition of  $r$ -ball. We know that at point  $p$  it holds that  $\delta_1(p) < \delta_2(p)$  and at  $q_2$  it holds that  $\delta_1(q_2) > \delta_2(q_2)$ . Since the distance functions change continuously, there must be some point  $x$  along the path  $\gamma_p^{q_2}$  such that  $\delta_1(x) = \delta_2(x)$ , where the closest connected component of  $\mathcal{F}$  to  $x$  changes. If  $x \notin \Gamma(\mathcal{C})$ , then there must be another point  $q_3 \in \mathcal{C}$  such that  $q_3 \notin f_1, f_2$  and  $d_{\mathcal{M}}(x, q_3) < \delta_2(x) \leq d_{\mathcal{M}}(x, q_2)$ . Using the triangle inequality, we get that  $d_{\mathcal{M}}(p, q_3) < d_{\mathcal{M}}(p, x) + d_{\mathcal{M}}(x, q_3)$ . Hence, we get that  $d_{\mathcal{M}}(p, q_3) < d_{\mathcal{M}}(p, x) + d_{\mathcal{M}}(x, q_3) < d_{\mathcal{M}}(p, x) + d_{\mathcal{M}}(x, q_2)$ . Using the fact that  $x$  lies on the geodesic shortest path between  $p$  and  $q_2$ , we have  $d_{\mathcal{M}}(p, x) + d_{\mathcal{M}}(x, q_2) = d_{\mathcal{M}}(p, q_2)$ , which implies that  $d_{\mathcal{M}}(p, q_3) < d_{\mathcal{M}}(p, q_2)$ , which contradicts our assumption that  $f_2$ , which contains  $q_2$ , is the second closest connected component (since  $q_3 \notin f_1, f_2$ ). Thus,  $x \in \mathcal{B}_{p,r}$  must belong to the medial axis  $\Gamma(\mathcal{C})$ .

If  $\mathcal{F} = \mathcal{C}$ , then the curve  $\mathcal{C}$  is entirely contained in  $\mathcal{B}_{p,r}$ . Let  $\delta_{\mathcal{C}}(x) = \min_{y \in \mathcal{C}} d_{\mathcal{M}}(x, y)$  be the minimal distance from each point to the curve. We consider the  $\ell$ -neighborhood  $T_{\ell}(\mathcal{C}) = \{x \in \mathcal{M} : \delta_{\mathcal{C}}(x) < \ell\}$  of the curve  $\mathcal{C}$  and the restriction of the neighborhood  $N_{\ell}(\mathcal{C}) = \{x \in \mathcal{B}_{p,r} : \delta_{\mathcal{C}}(x) < \ell\}$  to the  $r$ -ball  $\mathcal{B}_{p,r}$ . Since  $\mathcal{B}_{p,r}$  is an open set, for small values of  $\ell$ , the  $\ell$ -neighborhood  $T_{\ell}(\mathcal{C})$  is topologically equivalent to a  $d$ -dimensional solid torus and is entirely contained within  $\mathcal{B}_{p,r}$ . Thus  $N_{\ell}(\mathcal{C}) = T_{\ell}(\mathcal{C})$  has genus 1. On the



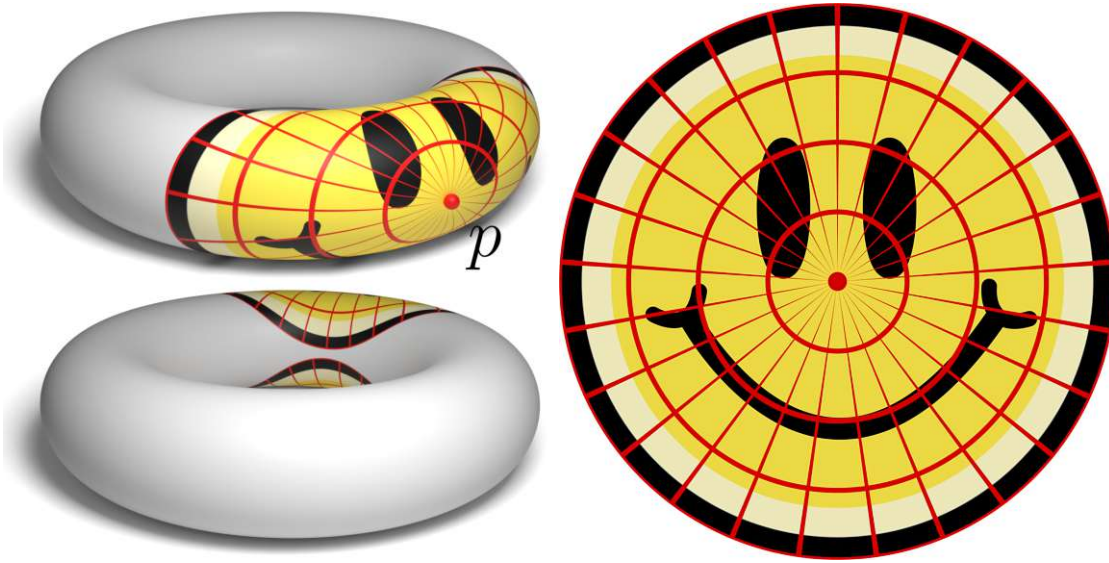


Figure 4.5: Mapping of an  $r$ -ball around a point  $p$  on a surface (left) to the Euclidean disk  $\mathbb{B}^2$  (right). The mapping is possible because  $r \leq i_{\mathcal{M}}(p)$ .

other hand,  $\mathcal{B}_{p,r}$  has finite size, and thus for values of  $\ell$  large enough  $N_{\ell}(\mathcal{C})$  covers the entire  $r$ -ball and hence,  $N_{\ell}(\mathcal{C}) = \mathcal{B}_{p,r}$ . Since  $r \leq i_{\mathcal{M}}(p)$ , the  $r$ -ball  $\mathcal{B}_{p,r}$  is topologically equivalent to a  $d$ -dimensional Euclidean ball (whose genus is 0), and so must be  $N_{\ell}(\mathcal{C})$ . Since the curve  $\mathcal{C}$  is immutable during the growing process of  $N_{\ell}(\mathcal{C})$ , and no elements are removed while increasing  $\ell$ , the genus decrease of  $N_{\ell}(\mathcal{C})$  cannot happen via cutting the solid torus. Thus, for some  $\ell$  there must be a non-contractible curve along the boundary of  $N_{\ell}(\mathcal{C})$  that collapses in a single point  $q \in \mathcal{B}_{p,r}$  to achieve a genus decrease. This means that  $q$  must be at a distance  $\ell$  from at least two different points on the curve, and thus, must be a point on the medial axis.  $\square$

**Corollary 1.** *For every point  $p \in \mathcal{C}$ , and for  $r \leq \text{ilfs}(p)$ , the ball  $\mathcal{B}_{p,r}$  intersects  $\mathcal{C}$  in a topological 1-disk.*

*Proof.* Let  $\mathcal{B}_{p,r}$  be a  $r$ -ball centered at  $p$ , for some  $r \in \mathbb{R}^+$ , that does not intersect  $\mathcal{C}$  in a topological 1-disk. If that is the case, then either  $r > i_{\mathcal{M}} > \text{ilfs}(p)$  or  $\mathcal{B}_{p,r}$  contains at least one point  $m$  of the medial axis (by Lemma 2). This would lead to  $r > d_{\mathcal{M}}(p, m) > \text{lfs}(p) > \text{ilfs}(p)$ .  $\square$

In Figure 4.6 we show an example of a planar curve intersected by Euclidean disks. At point  $p_1$ , the disk bounded by  $\text{ilfs}(p_1)$  intersects the curve in a topological 1-disk, as imposed by Corollary 1. The other points depict the three possible cases imposed by Lemma 1 and its generalization - Lemma 2:  $p_2$  intersects the curve in a topological 1-disk;  $p_3$  does the same, but it also contains part of the medial axis;  $p_4$  intersects the curve in two disconnected components, and thus it must contain a part of the medial axis.

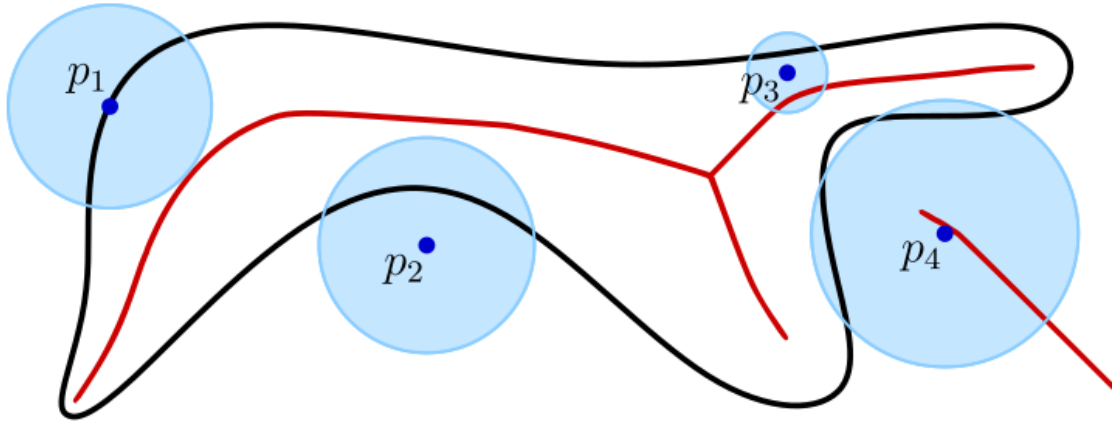


Figure 4.6: A planar curve (in black) and its medial axis (in red). Given some points (in dark blue), the disks centered at them (in light blue) either intersect the curve in a topological 1-disk, intersect the medial axis, or both.

**Properties of the sampling.** From Lemma 2 and Corollary 1, we can infer information on how the samples are distributed.

**Proposition 1.** *Let  $S \subset \mathcal{C}$  be a sampling of the curve. For every point  $p \in \mathcal{C}$ , let  $s_0, s_1 \in S$  be the samples such that the interval  $I = (s_0, s_1)$  is the smallest open interval between samples that contains  $p$ . If there exists a point  $q \in \mathcal{C}$  not belonging to the closure of  $I$  that is closer to  $p$  than both  $s_0$  and  $s_1$  are to  $p$ , then  $d_{\mathcal{M}}(p, q) \geq \text{ilfs}(p)$ .*

*Proof.* Let us denote  $\delta = d_{\mathcal{M}}(p, q)$ , and consider the  $r$ -ball  $\mathcal{B}_{p,\delta}$  of size  $\delta$  and centered at  $p$ .

Since  $s_0$  and  $s_1$  lie outside  $\mathcal{B}_{p,\delta}$ ,  $p$  lies inside  $\mathcal{B}_{p,\delta}$ , and  $\mathcal{B}_{p,\delta}$  touches  $q$ , then either the intersection  $\mathcal{F} = \mathcal{C} \cap \mathcal{B}_{p,\delta}$  has two connected components, or  $\mathcal{C}$  is tangent to  $\mathcal{B}_{p,\delta}$  at  $q$ .

If  $\mathcal{F}$  has two connected components, then by Corollary 1  $\delta > \text{ilfs}(p)$ .

If  $\mathcal{C}$  is tangent to  $\mathcal{B}_{p,\delta}$  at  $q$ , then for every  $\varepsilon > 0$ , the ball  $\mathcal{B}_{p,\delta+\varepsilon}$  would intersect  $\mathcal{C}$  in two connected components, meaning  $\delta + \varepsilon > \text{ilfs}(p)$ . Hence,  $\delta \geq \text{ilfs}(p)$ .  $\square$

Note that the statement of Proposition 1 does not require that  $p \notin S$ . Indeed, if  $p$  is a sample, the two samples that define the smallest interval containing  $p$  are its adjacent samples. This is the reason we use the open interval containing  $p$ , and not the closed version. Given the proper constraints of  $\rho$ , we can provide additional guarantees.

**Corollary 2.** *Let  $S \subset \mathcal{C}$  be a sampling of the curve, and  $s_0, s_1 \in S$  be two adjacent samples defining an interval  $I = [s_0, s_1] \subset \mathcal{C}$ . If  $S$  is a  $\rho$ -sampling with  $\rho < 1$ , then for every point  $p \in I$ , the closest sample to  $p$  is either  $s_0$  or  $s_1$ .*



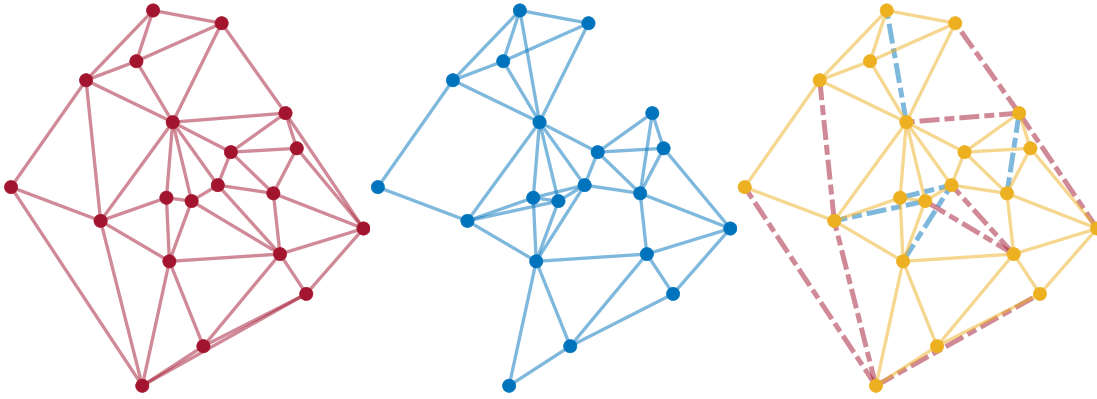


Figure 4.7: For a set of points in 2D, the intersection between Delaunay triangulation (red) and SIG (blue) creates the SIGDT (yellow). The edges removed from Delaunay triangulation and SIG are shown with dashed lines of the graphs' corresponding colors.

*Proof.* Assume the closest sample to  $p$  is some  $s_i \neq s_0, s_1$ , and let us denote  $\delta = \min(d_{\mathcal{M}}(p, s_0), d_{\mathcal{M}}(p, s_1))$ . Then  $\delta > d_{\mathcal{M}}(p, s_i) \geq \text{ilfs}(p) \geq \text{ireach}(I)$ . But by definition of  $\rho$ -sampling,  $\delta < \rho \text{ireach}(I)$ , which contradicts  $\delta > \text{ireach}(I)$  for  $\rho < 1$ .  $\square$

**Proposition 2.** *Let  $S \subset \mathcal{C}$  be a  $\rho$ -sampling, with  $\rho < 1$ . For any two consecutive samples  $s_0, s_1$ , let  $I = [s_0, s_1]$  be the interval between them. Then we have  $d_{\mathcal{M}}(s_0, s_1) < 2\text{ireach}(I)$ .*

*Proof.* Consider a point  $p$  in the interval  $I = [s_0, s_1]$  such that  $d_{\mathcal{M}}(s_0, p) = d_{\mathcal{M}}(s_1, p)$ . That point must be at  $d \geq \frac{1}{2}d_{\mathcal{M}}(s_0, s_1)$  from  $s_0$ , and by definition of  $\rho$ -sampling with  $\rho < 1$ ,  $d < \text{ireach}(I) \leq \text{ilfs}(p)$ . By Corollary 2, the closest samples to  $p$  must be  $s_0$  and  $s_1$ , and we have  $d_{\mathcal{M}}(s_0, s_1) \leq 2d < 2\text{ireach}(I)$ .  $\square$

#### 4.4.2 SIGDV graph on manifolds

In Chapter 3, we introduced the SIGDT proximity graph, obtained by intersecting the Delaunay triangulation with the Spheres-of-Influence graph, defined in 8.

The SIGDT is proved, in the planar setting, to contain the correct reconstruction of the curve under  $\rho$ -sampling conditions with  $\rho < 1$  and non-uniformity ratio  $u < 2$ , and being a subgraph of the Delaunay triangulation, it is also a sparse graph. These conditions make it a good starting point for finding the reconstruction of the curve. We provide an example of the construction of the SIGDT in Figure 4.7.

While the Spheres-of-Influence graph is only defined in terms of distances, lifting the Delaunay triangulation to manifold domains is less intuitive. The lack of a coordinate system makes it difficult to generalize classical construction algorithms like the Bowyer-Watson method [Bow81, Wat81] to non-Euclidean metric spaces. However, we use the dual graph of the Delaunay triangulation – the Voronoi partitioning of the vertex set, which is defined only in terms of distances [AKL13]. This duality has already been

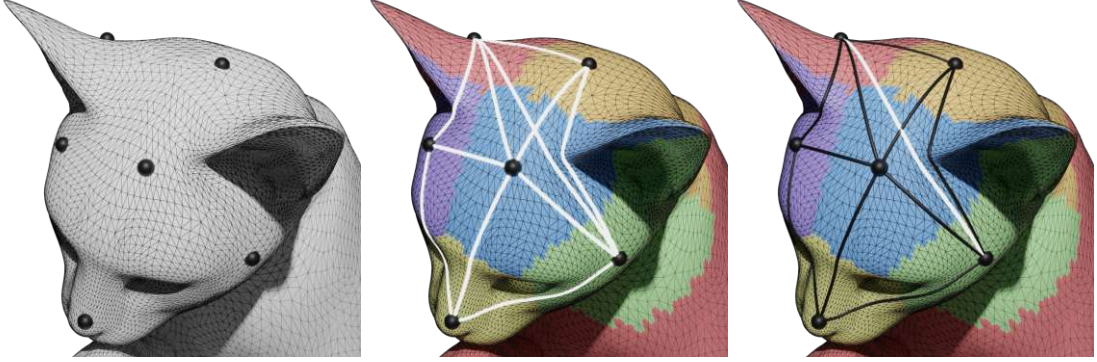


Figure 4.8: Left: a triangular mesh with some surface samples. Center: the geodesic Voronoi partitioning induced by the samples on the mesh, and its dual graph (white) obtained by locating the shortest geodesic paths between samples in adjacent partitions. Right: the SIGDV (black) obtained by removing the edges not in the SIG.

exploited for lifting the Delaunay triangulation to surfaces and has proved to maintain many properties of the 2D triangulation, such as providing angle stability and containing the nearest neighbor graph [Wan15, LFXH17]. Furthermore, it can be shown that computing the Voronoi partitioning of discrete manifolds is an efficient operation that can be achieved in time  $\mathcal{O}(|V| \log |V| \log k)$ , being  $|V|$  the number of vertices and  $k$  the number of samples [PC06, MBRM24].

Differently from the Euclidean space, the dual of a Voronoi decomposition does not always guarantee a valid triangulation of the point set, as there are some additional constraints that the samples must satisfy [ACDL00, DZM07]. If these are not satisfied, the dual still forms a sparse graph that encodes the proximity of samples well. In order to obtain the dual of the Voronoi, we connect samples that generate adjacent partitions, as shown in Figure 4.8. By removing all the edges from the Voronoi dual that do not also satisfy Equation 2.3, we obtain the intersection between the Spheres-of-Influence graph and the dual Voronoi graph, which we denote as *SIGDV*, and which provides a generalization for the SIGDT graph.

We prove the following statements about our graph.

**Lemma 3.** *Let  $s_0, s_1 \in \mathcal{C}$  be consecutive samples from a  $\rho$ -sampling  $S \in \mathcal{C}$ , with  $\rho < 1$ . The edge  $e(s_0, s_1)$  belongs to the dual Voronoi graph of  $S$ .*

*Proof.* Let  $I = [s_0, s_1] \subset \mathcal{C}$  be the interval of  $\mathcal{C}$  connecting samples  $s_0$  and  $s_1$ , and let  $p \in I$  be a point such that  $d_{\mathcal{M}}(s_0, p) = d_{\mathcal{M}}(s_1, p)$ . We consider the  $r$ -ball  $\mathcal{B}_{p,r}$  centered at  $p$ , where  $r = d_{\mathcal{M}}(p, s_i)$  is the distance from  $p$  to its closest sample  $s_i$ .

By Corollary 2,  $s_i$  must be either  $s_0$  or  $s_1$ , yielding  $r = d_{\mathcal{M}}(s_0, p) = d_{\mathcal{M}}(s_1, p)$ . Then the boundary between the Voronoi cells of  $s_0$  and  $s_1$  is not empty (as it contains at least  $p$ ), and the edge  $e(s_0, s_1)$  is the dual of their shared Voronoi boundary.  $\square$

**Lemma 4.** *Let  $s_0, s_1, s_2, s_3 \in \mathcal{C}$  be consecutive samples from a  $\rho, u$ -sampling  $S \in \mathcal{C}$ , with  $\rho < 1$  and  $u < 2$ . The edge  $e(s_1, s_2)$  belongs to the Spheres-of-Influence graph of  $S$ .*

*Proof.* We split the proof into four cases.

Case 1: If  $s_1$  is the nearest neighbor of  $s_2$ , or vice versa, then the edge  $e$  trivially belongs to SIG.

Case 2: If  $s_0, s_3$  are the nearest neighbors of  $s_1$  and  $s_2$  respectively, the non-uniformity ratio  $u < 2$  imposes  $d_{\mathcal{M}}(s_0, s_1), d_{\mathcal{M}}(s_2, s_3) > \frac{1}{2}d_{\mathcal{M}}(s_1, s_2)$ , meaning that  $d_{\mathcal{M}}(s_0, s_1) + d_{\mathcal{M}}(s_2, s_3) > d_{\mathcal{M}}(s_1, s_2)$ . Then the edge  $e$  belongs to SIG by definition.

Case 3: Now, suppose neither  $s_0$  nor  $s_2$  is a nearest neighbor for  $s_1$ , but  $s_3$  is a nearest neighbor for  $s_2$ . Then, the distance from  $s_2$  to its nearest neighbor is  $d_2 = d_{\mathcal{M}}(s_2, s_3)$ , and let  $d_1 = d_{\mathcal{M}}(s_1, s_i)$  be the distance from  $s_1$  to its nearest neighbor  $s_i$ . If  $d_1 + d_2 \geq d_{\mathcal{M}}(s_1, s_2)$ , the edge belongs to SIG by definition.

Otherwise, we must have  $d_1 < d_{\mathcal{M}}(s_1, s_2) - d_2$ . Because of the non-uniformity ratio  $u < 2$ , we know that  $\frac{1}{2}d_{\mathcal{M}}(s_1, s_2) < d_2 < 2d_{\mathcal{M}}(s_1, s_2)$ , and hence  $d_1 < \frac{1}{2}d_{\mathcal{M}}(s_1, s_2)$ . By Proposition 2,  $d_1 < \text{ireach}([s_1, s_2]) \leq \text{ilfs}(s_1)$ . However, since  $s_i$  is not adjacent to  $s_1$ , by Proposition 1 we have  $d_1 \geq \text{ilfs}(s_1)$ , which contradicts the above  $d_1 < \text{ilfs}(s_1)$ .

Case 4: Finally, suppose neither  $s_0$  nor  $s_2$  is a nearest neighbor for  $s_1$ , and neither  $s_1$  nor  $s_3$  is a nearest neighbor for  $s_2$ . If that is the case, a sample  $s_i$  must lie at distance  $d_1 = d_{\mathcal{M}}(s_i, s_1)$ , and Proposition 1 requires  $d_1 \geq \text{ilfs}(s_1)$ . We also must have a sample  $s_j$  lie at distance  $d_2 = d_{\mathcal{M}}(s_2, s_j)$  from  $s_2$ , which is the nearest neighbor of  $s_2$ , and Proposition 1 requires  $d_2 \geq \text{ilfs}(s_2)$ . If  $d_1 + d_2 \geq d_{\mathcal{M}}(s_1, s_2)$ , then the edge  $e$  belongs to SIG by definition. Suppose then  $d_1 + d_2 < d_{\mathcal{M}}(s_1, s_2)$ , and let us denote  $r_{1,2} = \min(\text{ilfs}(s_1), \text{ilfs}(s_2))$ . Proposition 2 requires  $d_1 + d_2 < d_{\mathcal{M}}(s_1, s_2) < 2r_{1,2}$ . Since  $d_1 \geq \text{ilfs}(s_1) \geq r_{1,2}$ , we have  $d_2 < r_{1,2}$ , which contradicts  $d_2 \geq \text{ilfs}(s_2) \geq r_{1,2}$ .  $\square$

Finally, we prove that SIGDV contains the correct reconstruction under analogous sampling conditions as the 2D case:

**Theorem 2.** *Let  $\mathcal{M}$  be a  $d$ -dimensional Riemannian manifold, possibly with boundary  $\partial\mathcal{M}$ , and equipped with a geodesic distance  $d_{\mathcal{M}} : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}$ . Let  $\mathcal{C} \subset \mathcal{M}$  be a curve on the manifold  $\mathcal{M}$ , and  $S = \{s_1, \dots, s_k\} \subset \mathcal{C}$  be a  $\rho, u$ -sampling of  $\mathcal{C}$ , with  $\rho < 1$  and  $u < 2$ . The edge connecting any pair of consecutive samples is part of the SIGDV of  $S$ .*

*Proof.* By Lemma 3 the edge  $e$  belongs to the dual Voronoi connectivity, and by Lemma 4 the edge  $e$  belongs to SIG. Then the edge also belongs to their intersection SIGDV.  $\square$

In Figure 4.9 we illustrate an example application of Theorem 2 to a curve on a 2-dimensional surface. The smooth curve wrapping around the head of the dog shape is sampled with a set of points, which are then connected with SIGDV. If two samples are adjacent on the curve, they share an edge inside SIGDV.

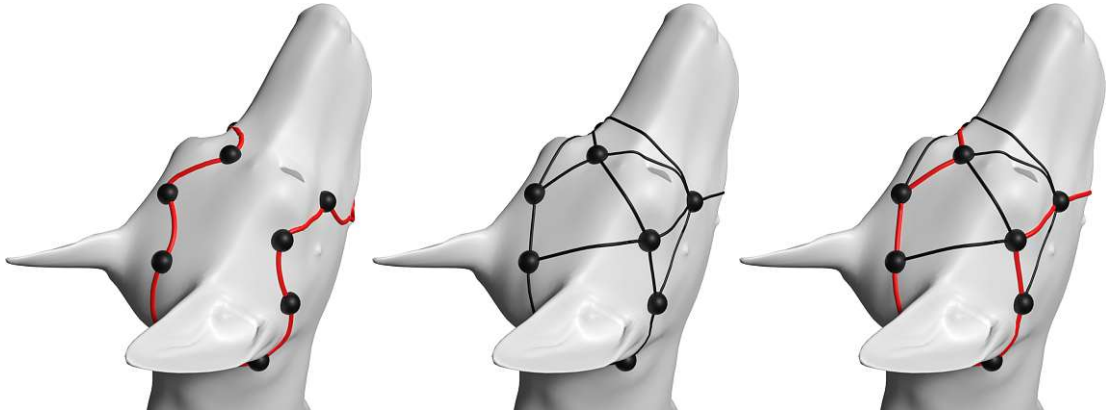


Figure 4.9: Left: a smooth closed curve (red) sampled at some vertices (black). Middle: the vertices are connected using SIGDV (black), including the edges between consecutive samples. Right: the shortest Hamiltonian path (red) inside SIGDV.

#### 4.4.3 Curve reconstruction as a Traveling Salesman Problem

While Theorem 2 guarantees that SIGDV contains the correct reconstruction, it may contain additional edges. This problem was also addressed in Chapter 3, for the SIGDT planar curve reconstruction algorithm. In the SIGDT pipeline, we propose to apply a series of inflating and sculpting operations until they obtain a manifold curve. In this case, this is not applicable, as in non-Euclidean spaces, such as manifolds, it is not always possible to define the notions of “inside” and “outside” of a curve.

To overcome this issue, we note that the edges of the correct reconstruction should create a cycle such that each sample is visited only once. Various ways of extracting these edges exist in the literature, such as the identification of a Hamiltonian cycle [IN07, Bjo14] or the solution to a Traveling Salesman Problem (TSP). While we are not guaranteed that the TSP solution encodes the ground truth ordering in general, it has been shown that outputting a shortest path adheres to the Gestalt principles and provides a good heuristic in the planar case [AM00, OM13, OPP<sup>+</sup>21].

In our experiments, this heuristics proved to generalize well on manifold domains, as shown in the example from Figure 4.9.

The Traveling Salesman Problem is a fundamental problem in computer science, and it is at the very core of many optimization problems. Given a set of points, TSP requires finding the shortest path that connects all the points, thus solving the problem of curve reconstruction as well. The problem can be formulated in different spaces and with different connectivity constraints, but all of them have been shown to be NP-hard [Kar75]. However, the importance of the TSP led many researchers to deploy approximate solutions, especially relying on a nearest-neighbor approach [JM97, RBP07] – even though these solutions tend to have a large approximation factor. Other techniques [DMC96, SM15] proved to be much more effective, while paying a small price on time efficiency.

Most of the research related to the TSP assumes complete graphs (*i.e.*, each node is connected to every other node). While we can accept solutions containing edges not belonging to SIGDV, we know that this graph contains the correct reconstruction under some sampling conditions, and hence we want to bias the algorithm to use SIGDV edges, while at the same time guaranteeing an efficient solution.

We start from a solution that exploits the Minimum Spanning Tree of a graph to produce a 2-optimal approximation to the problem (*i.e.*, a solution that is at most two times as expensive as the optimal one) [CLRS22]. Our method is summarized in Algorithm 4.1, where the input graph  $G$  is the SIGDV graph and the matrix  $\mathbf{D}$  contains in the entry  $\mathbf{D}_{ij}$  the geodesic distance  $d_{\mathcal{M}}(i, j)$  between the  $i$ -th and the  $j$ -th samples.

---

**Algorithm 4.1:** Algorithm for solving the TSP biased towards SIGDV edges.

---

**Data:**  $G = (V, E)$ ,  $\mathbf{D} \in \mathbb{R}^{|V| \times |V|}$   
**Result:** Ordered set of samples  $P$

- 1  $T \leftarrow$  minimum spanning tree of  $G$  w.r.t.  $\mathbf{D}$
- 2  $P \leftarrow$  ordering of  $V$  following a pre-order DFS visit of  $T$
- 3 **while**  $P$  is unchanged **do**
- 4     Find  $(i, j), (\ell, h) \in P$  s.t.  $\mathbf{D}_{ij} + \mathbf{D}_{\ell h} > \mathbf{D}_{i\ell} + \mathbf{D}_{jh}$
- 5      $P \leftarrow P \setminus \{(i, j), (\ell, h)\}$
- 6      $P \leftarrow P \cup \{(i, \ell), (j, h)\}$
- 7 **end**
- 8 **return**  $P$

---

We first compute the minimum spanning tree  $T$  of the graph  $G$  and via a pre-ordered Depth-First Search (DFS) visit of  $T$  we find an ordering of the nodes in  $V$  that we use to build an initial cycle  $P$  (Lines 1-2). Since the metric  $d_{\mathcal{M}}$  respect the triangular inequality, the cycle  $P$  is guaranteed to be a 2-optimal approximation of the TSP [CLRS22]. However,  $P$  does not offer any guarantee for the cycle to be a local minimum of the problem. Thus, we post-process the ordering by searching for any pair of edges  $(i, j), (\ell, h)$  such that if we swap them into  $(i, \ell), (j, h)$  we obtain a shorter cycle, and we iterate this process until no new pairs of edges are found (Lines 4-8). This edge-swap procedure was originally proposed by Croes [Cro58] for solving the TSP with an approximation factor of  $\mathcal{O}(\log n / \log \log n)$  on Euclidean instances (being  $n$  the number of vertices) [BHZ23].

**Relaxed sampling conditions.** The sampling conditions imposed by Theorem 2 allow for a sparser sampling compared to the state-of-the-art. However, when these conditions are not met, we do not have the guarantee that SIGDV contains the correct reconstruction or any other Hamiltonian cycle. Intersecting the dual Voronoi graph with the SIG could even produce a graph with multiple disconnected components. In some applications (see Figure 4.1) the input could be known to include samples from distinct closed loops and the clustering can be used to identify multiple curves. If the user desires to reconstruct only a single curve, we alter the graph to produce a single connected



component. First, we build the complete graph  $\mathfrak{G} = (\mathfrak{V}, \mathfrak{E})$  of the connected components, where each node  $N_i \in \mathfrak{V}$  represents a single component of SIGDV, and the length of the edge  $(N_i, N_j) \in \mathfrak{E}$  is given by  $\min_{u \in N_i, v \in N_j} d_{\mathcal{M}}(u, v)$ . Then, we compute the minimum spanning tree  $\mathfrak{T}$  of this graph. For every edge  $(N_i, N_j) \in \mathfrak{T}$ , we add to SIGDV the edge  $(u^*, v^*) = \arg \min_{u \in N_i, v \in N_j} d_{\mathcal{M}}(u, v)$ . This pre-processing guarantees that we obtain a single connected component, while, at the same time, adding non-SIGDV edges with minimum total edge length. The choice between single and multiple curve reconstruction is left to the user for full flexibility.

## 4.5 Applications

We investigate several diverse applications of our method. We first compare it to the approach proposed by Shah *et al.* [SC13] (MST) for motion tracking applications, where we show that we are able to reconstruct complex paths with far fewer samples. We show how our method can be applied in processing archaeological data, and how it can improve contour matching results, and we discuss the applicability of our solution to isoline extraction from sparse samples for scientific visualization.

Our method has been implemented in C++, using Eigen [GJ<sup>+</sup>10] and Geometry Central [SC<sup>+</sup>19]. The implementation is available at <https://github.com/filthynobleman/curvesurf>. For computing Voronoi diagrams in high-dimensional spaces we use Qhull [BDH96], and for computing the geodesic paths used in our visualizations we use the Flip Geodesics algorithm [SC20].

### 4.5.1 Motion tracking

In their work about curve reconstruction on Riemannian manifolds, Shah *et al.* [SC13] propose, as an application, the reconstruction of curves in the space of rigid motions  $SE(3)$  to track the motion of an object from position and rotation samples. To apply our method to this task, we first need to define how to compute a Voronoi diagram in  $SE(3)$ .

As discussed by González-López *et al.* [GLR95], the group  $SE(3)$  can be expressed as the product  $SO(3) \times \mathbb{R}^3$ ,  $SO(3)$  being the group of rotations in 3D space. The authors propose a method for computing the Voronoi decomposition in  $SO(3)$  relying on the following result.

**Proposition 3** ([GLR95]). *Given a set of rotations  $R = \{r_i\}_{i=1}^k$  in  $SO(3)$  and a set of rotation matrices  $R' = \{\mathbf{R}_i\}_{i=1}^k$  in  $\mathbb{R}^{3 \times 3}$  such that  $\mathbf{R}_i$  represents the rotation of  $r_i$ , the Voronoi decomposition  $\text{Vor}(R, d_{SO(3)})$  of  $SO(3)$  w.r.t. the rotations  $R$  can be obtained as*

$$\text{Vor}(R, d_{SO(3)}) = \text{Vor}(R', d_{\mathbb{R}^9}) \cap SO(3). \quad (4.7)$$

We improve this result for an easier computation of the Voronoi diagram in the group  $SO(3)$ .

**Proposition 4.** *Given a set of rotations  $R = \{r_i\}_{i=1}^k$  in  $SO(3)$  and a set of quaternions  $Q = \{q_i\}_{i=1}^k$  in  $\mathbb{H}$  such that  $q_i$  represents the rotation  $r_i$ , then the Voronoi decomposition  $\text{Vor}(R, d_{SO(3)})$  of  $SO(3)$  w.r.t. the rotations  $R$  can be obtained as*

$$\text{Vor}(R, d_{SO(3)}) = \text{Vor}(Q, d_{\mathbb{R}^4}) \cap SO(3). \quad (4.8)$$

*Proof.* Let  $r_p, r_q \in SO(3)$  be two rotations, and let  $p = (p_w, p_x, p_y, p_z), q = (q_w, q_x, q_y, q_z) \in \mathbb{H}$  be the two quaternions representing them. Let  $q^*$  be the conjugate of  $q$ . We know that the angular distance between  $r_p$  and  $r_q$  can be expressed as the angle of rotation of the quaternion  $t = (t_w, t_x, t_y, t_z) = pq^*$ , which in turn is given by  $2 \arccos(t_w) = 2 \arccos(\langle p, q \rangle)$ .

We also know that the squared Euclidean distance between  $p$  and  $q$  is given by  $\|p - q\|^2 = \|p\|^2 + \|q\|^2 - 2\langle p, q \rangle$ . Since the quaternions express rotations, they have unitary norm, meaning  $\|p - q\|^2 = 2 - 2\langle p, q \rangle$ . Thus, the angular distance between  $r_p$  and  $r_q$  can be expressed as

$$d_{SO(3)}(r_p, r_q) = 2 \arccos(\langle p, q \rangle) = 2 \arccos\left(1 - \frac{\|p - q\|^2}{2}\right).$$

Since  $\langle p, q \rangle$  is bounded by  $-1$  and  $1$  ( $\langle p, q \rangle = \|p\|\|q\|\cos(\theta) = \cos(\theta)$ , as  $p$  and  $q$  are unitary), and since  $\arccos(\cdot)$  is a decreasing function on  $[-1, 1]$ , then for any rotation  $r_w \in SO(3)$  and its corresponding quaternion  $w \in \mathbb{H}$  we get that  $d_{SO(3)}(r_p, r_w) < d_{SO(3)}(r_q, r_w) \iff \langle p, w \rangle > \langle q, w \rangle$ , but we also know that  $\langle p, w \rangle > \langle q, w \rangle \iff \|p - w\|^2 < \|q - w\|^2$ . Thus, it must be true that

$$d_{SO(3)}(r_p, r_w) < d_{SO(3)}(r_q, r_w) \iff \|p - w\| < \|q - w\|.$$

This proves that the distance functions  $d_{SO(3)}$  and  $d_{\mathbb{R}^4}$  preserve the distance relationships between rotations in  $SO(3)$ , meaning that  $\text{Vor}(R, d_{SO(3)}) = \text{Vor}(Q, d_{\mathbb{R}^4}) \cap SO(3)$ . Although our result is partially connected to the theory of the Riemannian center of mass [Kar77], we aimed to provide a precise and tailored proof for the specific case of rotations in  $SO(3)$  and their Voronoi decomposition.  $\square$

As proven by González-López *et al.* [GLR95], this result does not generalize to  $SE(3)$ , and we must only rely on the Voronoi decomposition of the embedding space. However, by reducing the dimensionality of the embedding space of  $SO(3)$  from  $\mathbb{R}^9$  to  $\mathbb{R}^4$ , we also reduce the dimensionality of the embedding space of  $SE(3)$  from  $\mathbb{R}^{12}$  to  $\mathbb{R}^7$ . Thus, we can use the Voronoi decomposition in  $\mathbb{R}^7$  as a reasonable approximation of the Voronoi decomposition in  $SE(3)$ .

In Figure 4.10 we present an example of motion reconstructed using MST and our solution, and we show the minimum number of samples required by each method to correctly reconstruct the motion sequence – 21 for MST and 9 for our method. While our method

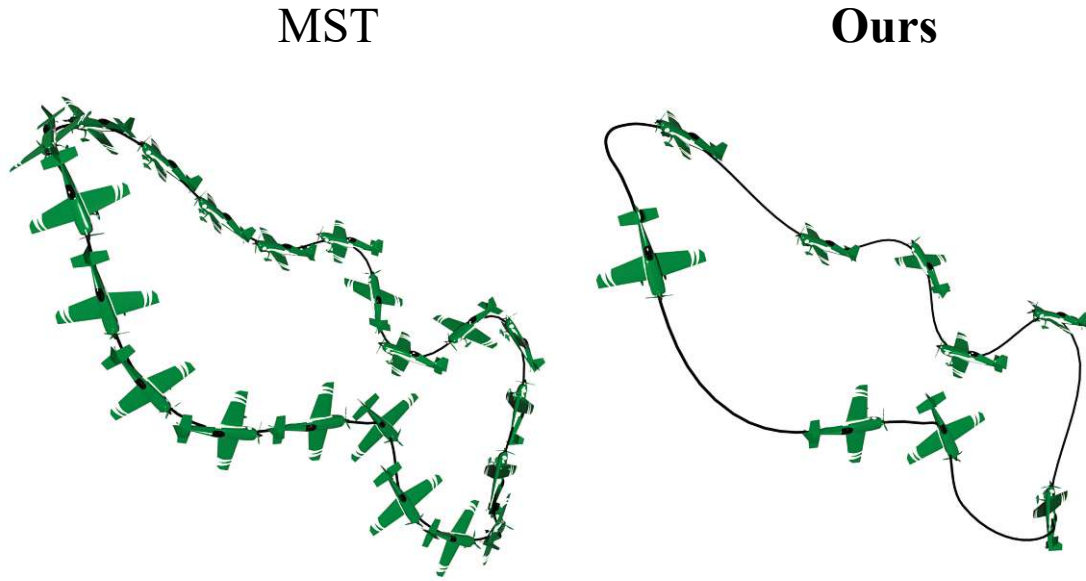


Figure 4.10: Minimal number of samples for reconstructing the motion of an airplane along a path using MST - left (21), and our approach - right (9). The curve shows the ground truth path of the airplane.

The airplane model has been created by Kemal Çolak and distributed by Sketchfab under the license CC BY 4.0.

can deal with sparse non-uniform sampling and still correctly recover the ordering of the samples, it is evident here that MST requires a more dense and uniform sampling scheme. Indeed, the path presents a challenging shape, as the two close turns in the bottom half are very close in space, and the adjacent samples on the curve are very different in terms of rotations. Such curves can be adversarial for MST, as the distance in  $SE(3)$  between non-adjacent samples is small, preventing their method from finding a closed curve.

#### 4.5.2 Virtual cultural heritage

Scientific visualization and 3D shape analysis are already prevalent in archaeological applications as instruments for handling the always-increasing amount of data available to cultural heritage research [Tal14].

Successful applications of computational geometry to the analysis of archaeological data include the identification of demarcating curves on low reliefs, statues, and other various kinds of cultural heritage artifacts [GTSK13, TBF16], curve matching tasks in 2D and 3D for solving the problem of fragment reconstruction [MK03] and pattern extraction of culturally significant patterns [YP20].

The problem of reconstructing curves on surfaces is closely related to these applications, as there can be cases where missing information from lost pieces could be inferred by knowing the structure of the underlying artifact (*e.g.* partial designs on vases that could



MST

Ours



Figure 4.11: Reconstruction of complex curves (white) on the surface of a clay vase from a set of manually chosen samples (blue). The MST algorithm (left) fails to reconstruct meaningful contours, while ours (right) correctly identifies the two shapes. The vase model has been created by Laura Shea and distributed by Sketchfab under the license CC BY-NC-SA 4.0 DEED.

be reconstructed from still existing pieces). In Figure 4.11 we show that our method can faithfully reconstruct complex shapes on the surface of a vase, precisely contouring the black drawings from a sparse sampling. We also compare our result against the MST algorithm, which is unable to find a closed shape due to the presence of multiple connected components and the large distance among some of the samples.

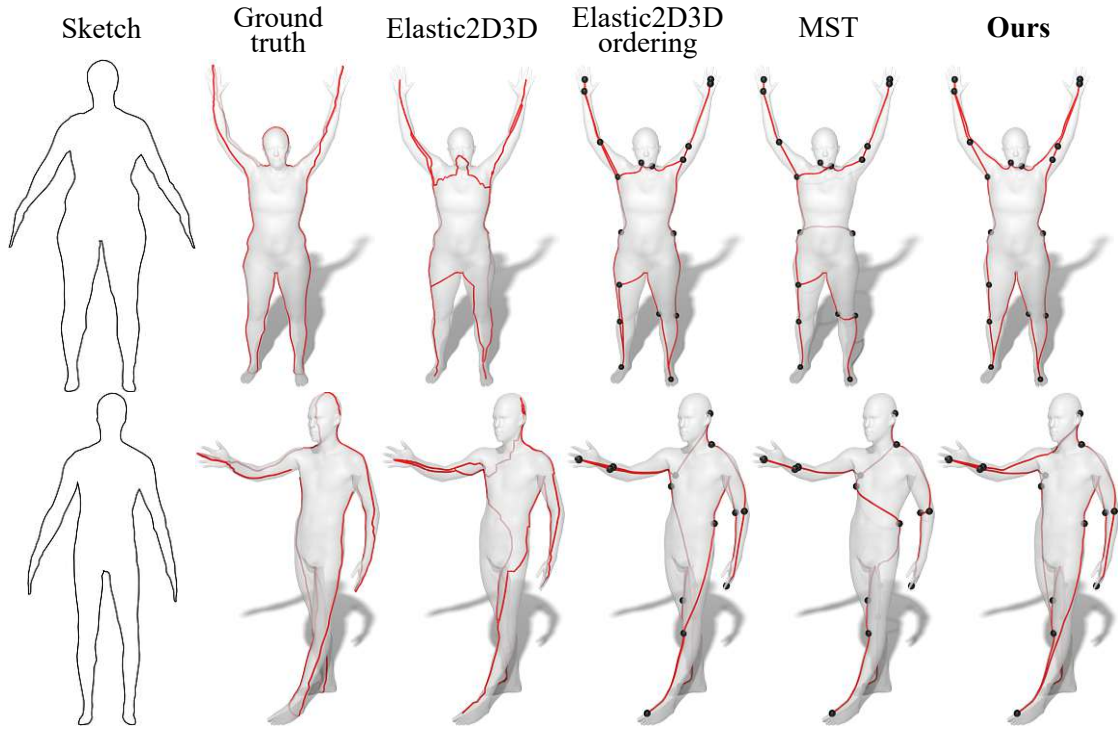


Figure 4.12: Examples of how our curve reconstruction method is applied to refine a contour. Given a contour obtained with ELASTIC2D3D [LRS<sup>+</sup>16], we sample it sparsely and reconnect the samples using the ordering of the contour, MST, and our algorithm.

### 4.5.3 Contour matching

Contour matching is a novel task in computer graphics and vision applications that has recently achieved high attention [LRS<sup>+</sup>16, RLB23]. The problem this field is concerned with is how to meaningfully relate a surface to the curve defining its contour, even under non-rigid deformation of the surface [WSSC11].

While the existing approaches reliably compute non-rigid correspondences between shapes and their contour, the task is still new and even state-of-the-art methods do not always provide high-quality guarantees, producing self-intersecting and locally degenerate curves. We show that our curve reconstruction technique can be exploited as a refinement step to improve the contour quality.

For our evaluation, we use the method proposed by Löhner *et al.* [LRS<sup>+</sup>16] (ELASTIC2D3D), selecting some instances from the dataset that the authors derive from the FAUST shape collection [BRLB14] where the method produces many degeneracies. From the curve obtained with ELASTIC2D3D, we randomly extract 3% of the vertices, and we run our algorithm for reconstructing the contour over the target shape.

Even if ELASTIC2D3D may produce self-intersections and locally degenerate curves, it captures the overall shape of the contour. Thus, a sparse sampling of the curve should

give a good sparse contour. As shown in the examples from Figure 4.12, with our algorithm we can smooth the results of ELASTIC2D3D, resolving the self-intersections and producing non-degenerate curves. For reference, we also connect the samples with their original ordering, proving that our method's reordering of the samples improves the overall result.

MST cannot produce a valid closed curve in any of the tested cases, as the conditions imposed by the algorithm require the sampling to be denser and more uniform for the minimum spanning tree to result in a chain. This is primarily due to the presence of degeneracies in the curve produced by ELASTIC2D3D in the form of jagged edges. Hence, two points sampled from a degenerate section of the curve would very likely result in a branch on the minimum spanning tree. We validate this argument by running the MST algorithm on different sampling densities spanning from 1% to 100% of the curve vertices produced by ELASTIC2D3D, and the minimum spanning tree never results in a chain.

#### 4.5.4 Sparse data visualization

Having shown that our method can improve contours already computed by reordering a sparse set of their samples, we now move onto another definition of contours, namely their existence as isolines (*i.e.*, lines connecting data points with the same value). We show how our method can be used to visualize large data by only using a sparse subset of it.

Analyzing large datasets and being able to extract meaningful visual information is a challenge for various scientific visualization fields. We choose to focus on meteorological data in this application, and considering the increase in computational power and available tools, the information we can obtain has increased exponentially. Methods to quickly process massive amounts of data are required to allow scientists to interpret and extract the most relevant information. One method to deal with very dense data is to choose a subset that acts as an overview, and then use this global view of data to find and focus on a specific area of interest at a higher resolution [KW19, MHS<sup>+</sup>20]. In the case of prohibitively large datasets, such as hourly records of multiple weather parameters, processing solely a subset of the data might be the only way to manipulate information at such a scale. This facilitates a quicker understanding of the data at different levels while using less data and hence, less computational power.

In the field of visualizing meteorological data, an important problem arises from the numerical issues of projecting information that exists on the Earth's surface onto a plane, where at least one of the following properties has to be sacrificed: distances, angles, or areas. We bypass this challenge by working with samples directly on surfaces and allowing for a three-dimensional interaction and visualization of data. A common way of visualizing various properties of meteorological data is presented by contour lines (or isolines) where data points with equal or similar (up to a threshold) values are connected to represent all the possible points that have similar data [RBS<sup>+</sup>17].

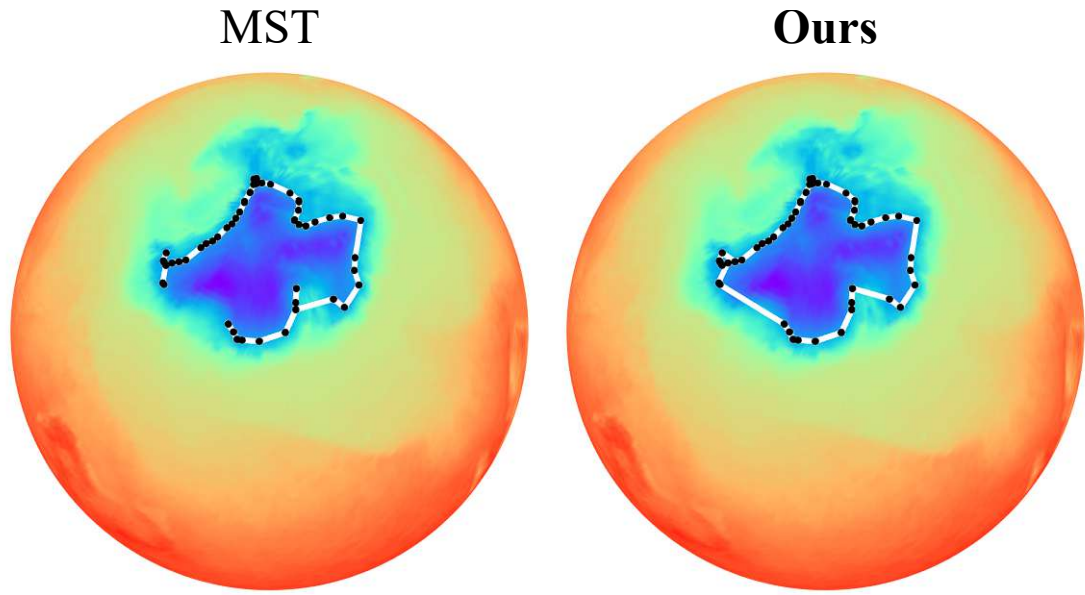


Figure 4.13: Samples (black) with approximately the same temperatures connected using the MST algorithm (left) and our method (right) to form a contour line (white) for a region with a temperature of  $230 \pm 0.3\text{K}$  on the globe, for a single timestep.

Meteorological datasets, such as the ones provided by the European Centre for Medium-Range Weather Forecasts [fMRWFE], contain massive amounts of data for every recorded timestep, at a resolution of 0.25 degrees in both latitude and longitude, with up to 90 parameters such as temperature, humidity, pressure. Analyzing the entirety of the data at a global level is a tedious and complex task, and we propose to combine the deformation-free surface representation with a sparse sampling of the dataset to reconstruct contours for an easier understanding of global information.

In Figure 4.13, we use the global temperature values recorded on 31<sup>st</sup> of March 2024, at midnight, and use a subset of the values (a quarter of the initial array, by skipping every second and fourth row) to minimize the amount of data. The texture of the mesh represents the ground truth temperatures. We then extract the points with a temperature of  $230 \pm 0.3\text{K}$  and connect them to obtain the corresponding contour line, only requiring 63 samples to reconstruct the isoline. In contrast, the MST algorithm is unable to reconstruct the curve due to the non-uniformity of the samples. The data we used is based on data and products of the European Centre for Medium-Range Weather Forecasts (ECMWF), under CC BY 4.0. license.

## 4.6 Conclusions

We have generalized the sampling requirements of closed curves from the Euclidean plane  $\mathbb{R}^2$  to Riemannian manifold domains of arbitrary dimension. By taking inspiration from

the literature on planar curve reconstruction, we designed a new method for recovering the original connectivity from a sparse sampling by biasing an instance of the traveling salesman problem with a coarse graph, which exhibits strong theoretical guarantees. We have proved the effectiveness of our method by testing it in different scenarios and various applications involving real-world data, showing that it outperforms the existing previous solution and that it provides a correct result, while in many cases the state-of-the-art method fails.

As for future directions, we intend to explore possible relaxations of the starting graph to offer stronger theoretical guarantees even under adversarial samplings. Furthermore, we notice that when the sampling conditions satisfy our constraints, the original reconstruction forms a Hamiltonian cycle inside the SIGDV graph. Since this path is not guaranteed to be the shortest tour, an algorithm for the TSP could be a sub-optimal strategy and more sophisticated techniques for identifying such a cycle could be worth future investigation. Another possible avenue of future development could involve a more compelling approach for handling multiple closed loops and the exploration of techniques for dealing with more general classes of curves, as these could provide an important tool in pattern extraction, where not all elements are closed smooth curves. Finally, as the reconstruction of curves in non-Euclidean domains is a barely explored topic, we identify the absence of a unified validation benchmark for quantitative analyses and we intend to fill this gap in the future.



# SIG3D: Encoding Connectivity for Point Clouds

We analyze the properties of the SIG in proximity encoding of point clouds. We prove the existence of a distance bound on SIG neighbors, using it to improve our performance. The contents of this chapter have been adapted from the paper “*Parameter-Free Connectivity for Point Clouds*” [MOW24], presented at GRAPP 2024 - the 19th International Conference on Computer Graphics Theory and Applications, published in the Conference Proceedings, and invited for an extended submission to the Communications in Computer and Information Science Springer journal, currently in submission [MIOW24].

## 5.1 Overview

Determining connectivity in unstructured point clouds is a long-standing problem that has still not been addressed satisfactorily. In this chapter, we analyze an alternative to the often-used  $k$ -nearest neighborhood ( $k$ NN) graph - the Spheres of Influence Graph (SIG). We show that the edges that are neighboring each vertex are spatially bounded, which allows for fast computation of SIG. Our approach shows a better encoding of the ground truth connectivity compared to the  $k$ NN for a wide range of  $k$ , and additionally, it is parameter-free. Our result for this fundamental task offers potential for many applications relying on  $k$ NN, e.g., parameter-free normal estimation, and consequently, surface reconstruction, motion planning, simulations, and many more.

## 5.2 Introduction

In recent times, point clouds have gained popularity as a data representation, owing to advancements in scanning technology. However, the initial state of disorganized points



necessitates further processing to reconstruct the original surface’s connectivity from which they were sampled. This connectivity allows for the estimation of surface properties such as neighborhoods or normals. Various methods are available for estimating the intrinsic properties of point clouds, but they often rely on selecting the appropriate parameters tailored to the specific data type. For instance, this includes determining neighborhood connectivity using the user-specified ‘ $k$ ’ in  $k$ -nearest neighbor ( $k$ NN) calculations. Additionally, point cloud sampling can be non-uniform due to the acquisition method or tainted by artifacts like noise and outliers. These conditions make the process of parameter selection for scanned data a challenging and time-consuming endeavor.

Extracting connectivity from point clouds is a significant research challenge, not only due to its essential role as a first step in surface reconstruction but also because it forms the foundational input for graph-based learning tasks involving point clouds. Regarding surface reconstruction, connectivity graphs serve as the fundamental structure on which triangles are constructed, and as a computation base for normals, which are sometimes required as part of the reconstruction process. For learning-based tasks, establishing a method for connecting the input points that closely aligns with the original surface is imperative, since the creation of an actual surface is not necessary.

We propose a fast computation of the *spheres-of-influence* graph (SIG) and we analyze its properties as a proximity graph. This graph recovers the original connectivity of the surface better than the widely used  $k$ NN graph, as it can be seen in Figure 5.1, while dropping the need for users to search for a suitable parameter that typically varies depending on the input and its local properties. Our method achieves the best results on various models with different features. Hence, our method offers a good scaffolding for further processing of point clouds, such as normal estimation, surface reconstruction, or graph convolutional neural networks. We show that our method not only encodes the original connectivity better than  $k$ NN but, as an application, provides a good base for normal estimation, while remaining parameter-free.

We present the following contributions:

- We introduce an effective method for constructing the spheres-of-influence graph, proving novel spatial constraints for this parameter-free graph.
- Our method is evaluated against ground truth meshes, demonstrating superior connectivity representation with a reduced space requirement when compared to traditional  $k$ NN graphs.
- As an application, we offer an analysis of normal computation on point clouds, highlighting our method’s ability to deliver competitive results.



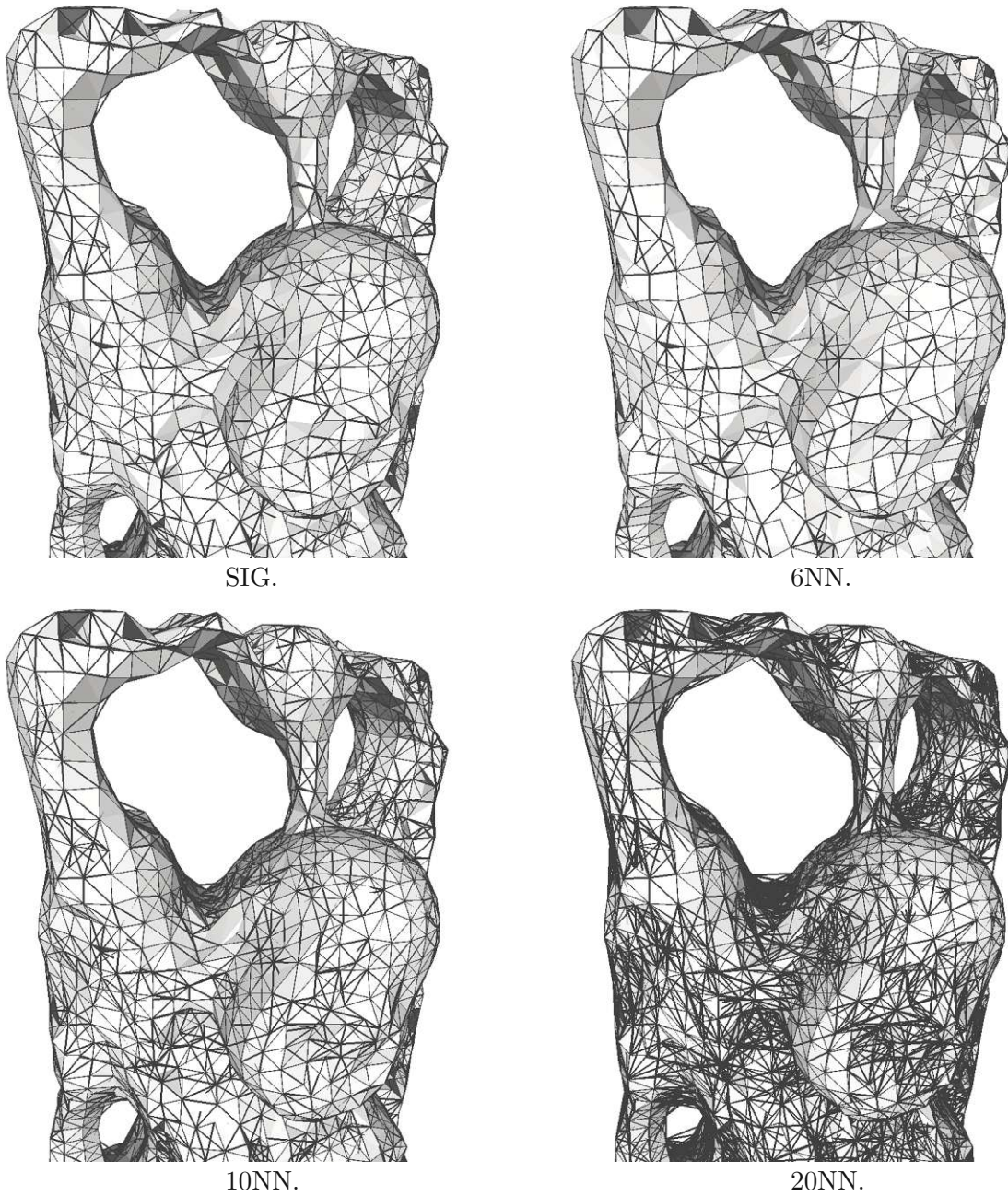


Figure 5.1: Results of our SIG and the  $k$ NN graphs for  $k = \{6, 10, 20\}$  for the connectivity of the Buddha statue from the Stanford repository. Our graph captures the connectivity well, without many redundant edges, and without the need for a parameter compared to the  $k$ NN graphs. The original surface is shown in light gray, with the various graphs overlaid in black.

### 5.3 Method

We define a mesh as a collection of vertices -  $V = \{p \in \mathbb{R}^3\}$  and triangulated faces -  $T = \{(a, b, c) | a, b, c \in V, a \neq b \neq c \neq a\}$ . Using only the set of unstructured points  $V = \{p \in \mathbb{R}^3\}$ , we aim to recover a set of edges that connect the given samples as similar as possible to the original connectivity of the mesh which is encoded in  $T$ . We define similarity here as edge connectivity, instead of the exact triangulation of the original mesh. Our method uses the spheres-of-influence graph, and we provide an improved algorithm to efficiently compute this proximity graph. We are using the vertices of triangulated meshes as our input in order to have access to a ground truth connectivity to which we can compare our results.

**Spheres-of-influence graph.** The SIG, defined in 8, encodes the spatial proximity of vertices well, without the need for a parameter. It contains the Nearest Neighbor graph [Tou88], as the edge between points and their respective nearest neighbor will always satisfy the condition.

Various properties of the sphere-of-influence graph have been investigated, such as the bound on the size of the graph [Dwy95] or its behavior under different metric spaces [MQ99]. However, even if the number of edges has been proven to be linear, algorithms to efficiently compute the graph have not yet been researched, especially considering that SIG is not included in the Delaunay triangulation and hence, the latter cannot be used as a starting point for pruning [JT92]. We show that the neighbors of each vertex in SIG are bounded inside a fixed radius - Figure 5.2, and we use this spatial constraint to improve the speed of the computation.

**Theorem 2.** *For all vertices  $a \in V$ , all SIG edges  $(a, b), b \in V, a \neq b$ , are contained in a radius of at most  $2nn(a)$  from  $a$  or in a radius of at most  $2nn(b)$  from  $b$ , where  $nn(v)$  is the distance between  $v$  and its nearest neighbor, for  $v \in V$ .*

*Proof.* We will prove the statement by contradiction. Let us compute the distance to the nearest neighbor for each vertex and store the result as  $nn(v)$  for all vertices  $v \in V$ . For each vertex  $v$ , retrieve all the vertices within  $2nn(v)$  distance of  $v$  and only store the edges that satisfy the SIG criterion. Let us call the obtained graph FastSIG (FSIG).

Since we are trying to prove Theorem 2 by contradiction, we assume that there exists at least an edge that is outside the range boundaries we have defined (within  $2nn(a)$  for each vertex). This means that there has to exist an edge  $(a, b)$  which is in SIG but not in FSIG. Hence,

$$\|a, b\| \leq nn(a) + nn(b), \quad (5.1)$$

since  $(a, b) \in \text{SIG}$ , and:

$$\|a, b\| > 2nn(a) \text{ and } \|b, a\| > 2nn(b), \quad (5.2)$$

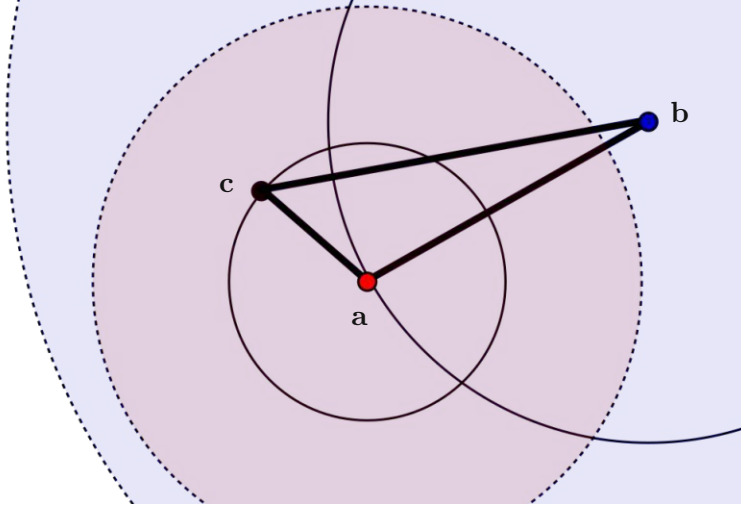


Figure 5.2: We illustrate the spheres of influence of vertices **a** and **b** in 2D as outlined circles, with a radius equal to the distance to their respective nearest neighbor. The bounding radius of the incident SIG edges are colored disks with dashed borders of radius equal to twice the SIG radius. We can observe that even if **b** is not in the incidence radius of **a**, **a** is contained in **b**'s radius, showing that SIG edges are contained in at least one of the endpoints' incidence radii.

from  $(a, b) \notin \text{FSIG}$ . Processing these inequalities further, we get:

$$\|a, b\| > nn(a) + nn(b), \quad (5.3)$$

which contradicts our assumption of  $(a, b) \in \text{SIG}$ . Since we arrived at a contradiction, we have proved that all SIG edges are also in FSIG.  $\square$

Having proved that SIG edges live in a fixed boundary from every node, we use this information to retrieve all the neighboring nodes within the given radius using a *kdtree*, as presented in Algorithm 5.1.

## 5.4 Results

We are aiming to provide an alternative for  $k\text{NN}$  that is parameter-free, fast to compute, efficient to store, and achieves better or similar surface properties to  $k\text{NN}$  for a wide variety of data. In this section, we will show how SIG satisfies all these requirements, representing a good, parameter-free alternative to  $k\text{NN}$  for estimating unstructured point clouds' connectivity.

We have tested our method on a varied dataset of points clouds exhibiting various features, such as various types of non-uniform sampling and sharp edges. We compared our results to the  $k\text{NN}$  neighborhood for connectivity recovery, for usual values of  $k = \{6, 10, 20\}$ .

**Algorithm 5.1:** Fast SIG computation.

---

**Data:**  $V = \{v \in \mathbb{R}^3\}$   
**Result:**  $\text{SIG} = \{(a, b) : a, b \in V, a \neq b, \|a, b\| \leq nn(a) + nn(b)\}$

```

1 SIG = {};
2 create kd-tree  $KT$  of  $V$ ;
3 for  $v \in V$  do
4   | find  $nn(v)$  using  $KT$ ;
5 end
6 for  $v \in V$  do
7   |  $N = KT.\text{findNbrInRange}(2nn(v))$ ;
8   | for  $u \in N$  do
9     | if  $\|u, v\| \leq nn(u) + nn(v)$  then
10    |   | SIG +=  $(u, v)$ ;
11    |   end
12  | end
13 end

```

---

We did not use  $k$ -values lower than 6 since this is the average degree of vertices in triangulated meshes, as can be shown using Euler's formula. As an application, we have also computed normals using PCA from our graph and from the  $k$ NN graphs and compared these with the face-based normals of the original meshes.

**Dataset.** We have used a subset of the Thingy10K [ZJ16] dataset of meshes that are manifold and have less than 1k vertices for our quantitative results, consisting of around 3k different models. Most of these meshes exhibit CAD-like features in the form of having samples mainly along edges, with thin and long faces. This type of sparse sampling affects the results of our connectivity measures, as, on average, none of the investigated connectivity graphs can perfectly capture the original connectivity, but we have chosen to include this type of data as well to test the resilience of our method in the presence of sparse sampling. For qualitative results, we have included some of the meshes from the Stanford repository (<https://graphics.stanford.edu/data/3Dscanrep/>) - the resulting graphs using the Stanford bunny are presented in Figure 5.3.

**Connectivity.** The connectivity of the ground truth is what our graph aims to reconstruct. However, the original edge set is not a unique representation of the intrinsic connectivity of points, as it is highly dependent on the chosen triangulation, e.g., edges may flip, as can be visible in Figure 5.4. Hence, we do not aim to reproduce the exact edges of the ground truth meshes (i.e., comparing the 1-ring of each vertex), but to create a good approximation of the connectivity of the entire surface.

In order to quantify how close our graph is to the original connectivity, we used the DeltaCon metric [KSV<sup>+</sup>16], where a value of 0 means the graphs are completely different



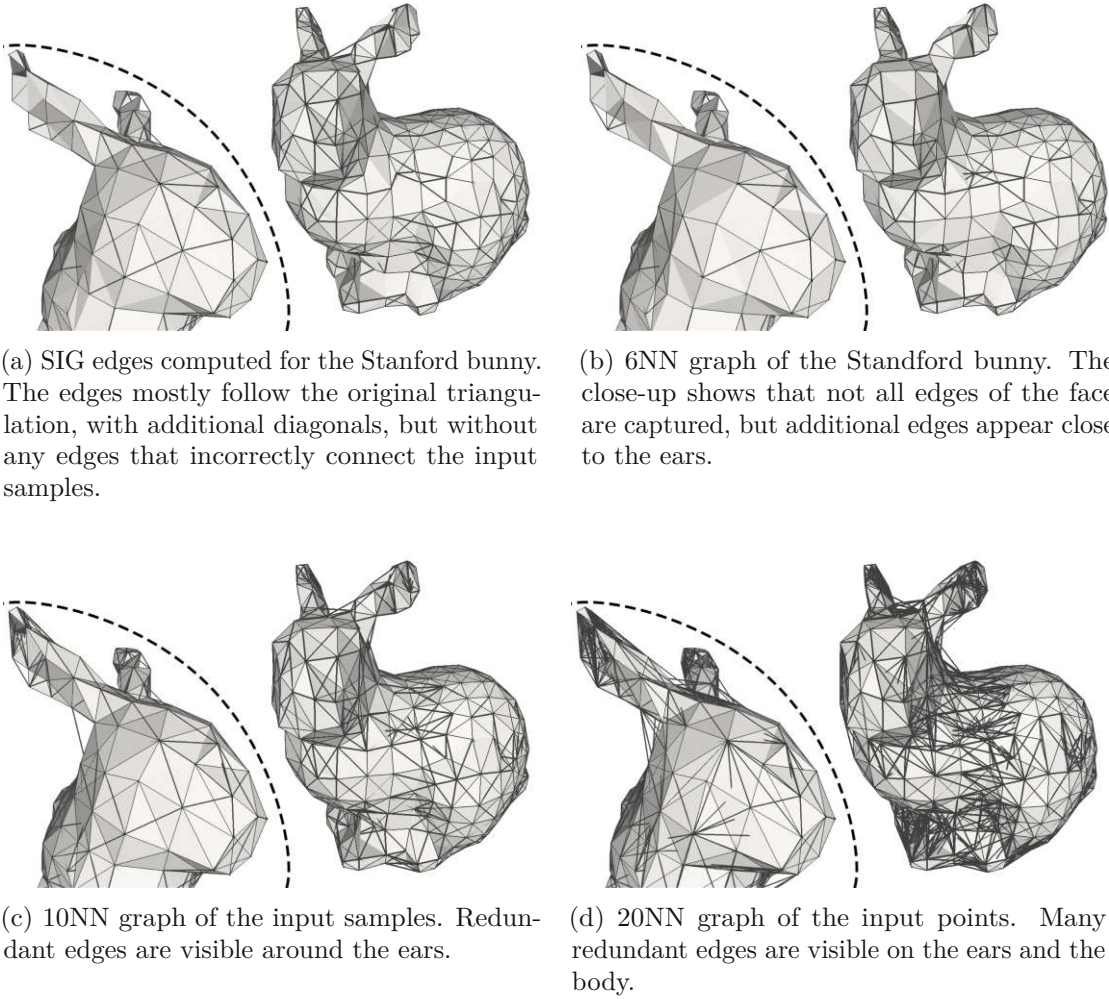


Figure 5.3: Results of our graph and  $k$ NN for  $k = \{6, 10, 20\}$  on the Stanford bunny, where the original surface is presented in gray, with various connectivity graph edges overlaid in black.

and a score of 1 implies identical graphs. DeltaCon computes a matrix of node-to-node influence for each graph and the final score is a difference between these two  $n^2$  matrices. This is equivalent to using  $m$ -ring neighborhoods ( $m \in [1, n]$ ) for each vertex with decreasing weights as we move farther from the current node. Moreover, this metric benefits from edge awareness - disconnecting changes are penalized more than removing an edge from a complete graph, and this is a property that highly influences the type of connectivity we want to measure for our method. For a detailed definition of the metric and its implementation, we direct to the original works [KSV<sup>+</sup>16].

We have computed DeltaCon for the initial 3431 meshes from Thingy10K, with various

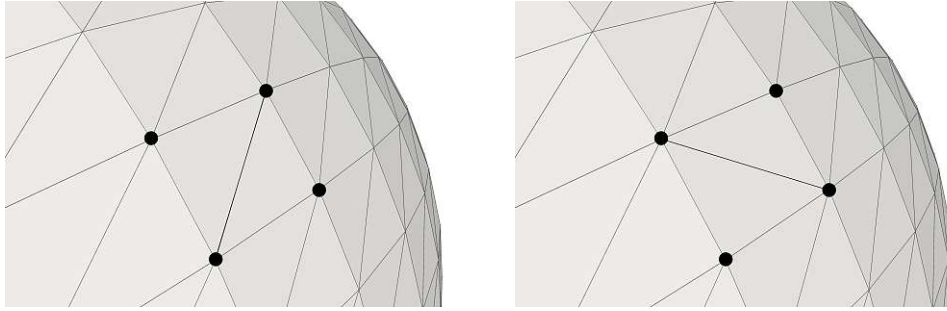


Figure 5.4: Local changes in the triangulation choice - such as edge flips, do not affect the overall connectivity of the mesh. The four highlighted vertices create the same connected surface in both figures.

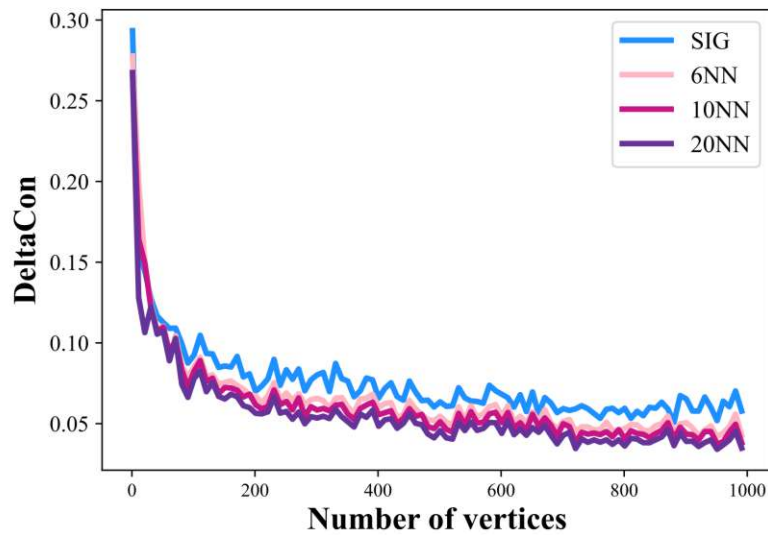


Figure 5.5: DeltaCon graph similarity metric - a higher value corresponds to a closer similarity to the original graph.

numbers of vertices and different sampling densities. The results are presented in Figure 5.5, where SIG consistently obtains higher scores than the  $k$ NN graphs. The results are clustered in equally sized buckets depending on the number of vertices. For each bucket, we present the averaged DeltaCon measurement over all inputs with the total number of vertices in the specified range. Even if the metric does not achieve 1, as the graphs are not identical (as we do not aim for this), our graph manages to encode the original connectivity better than the  $k$ NN graphs. The maximum DeltaCon value achieved by all methods is also lower than 1 due to the sparse sampling of the dataset, as mentioned previously.

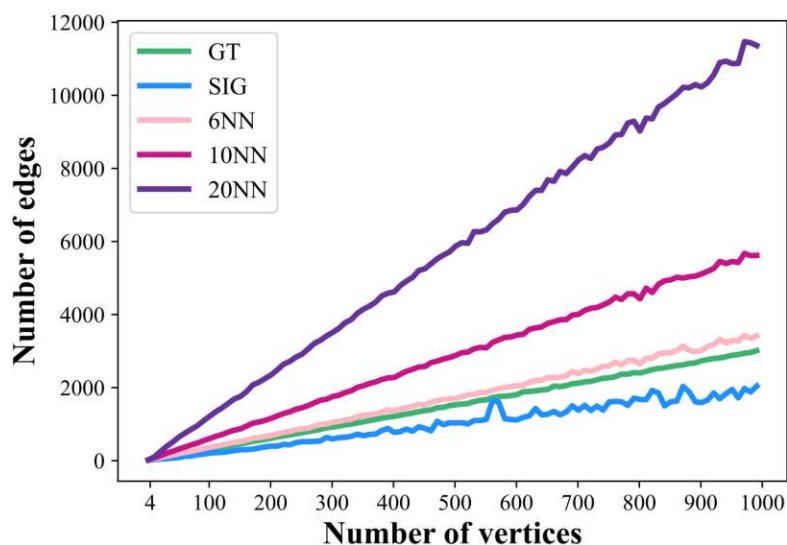


Figure 5.6: Number of edges in graphs plotted against the number of vertices. We consistently obtain the smallest number of edges by a large margin, while still correctly preserving the original connectivity.

**Storage.** Low storage represents another requirement for our method, since we try to use the minimum amount of edges that preserve the connectivity by making use of the spatial proximity properties of SIG. The total number of edges for each graph can be seen in Figure 5.6, where the results are presented with respect to the number of vertices. Our method achieves the lowest number of edges and hence, has the lowest storage requirement. Our number of edges is lower than the ground truth since the ground truth number of edges is extracted from triangulated meshes, which contain some redundant edges with respect to connectivity.

**Surface approximation - geodesic.** We are also comparing the distance between pairs of nodes in our graph to the geodesic distance over the ground truth surface, computed using the Heat Method [CWW17]. This way, we measure how close the graph edges follow the surface. For all pairs of nodes in the original mesh, we compute the ratio between the geodesic distance over the original mesh and the shortest distance between the same nodes in SIG and between the same nodes in the  $k$ NN graphs. The results are shown in Figure 5.7, where we present the ratio of distances in relation to the number of vertices in the input. The differences between buckets are due to the varying number of input datasets in each bucket, as well as the type of meshes - a single mesh with close sheets and sparse sampling, such as the one presented in Figure 5.8 highly increases the error for the connectivity graphs. We aim to obtain a resulting ratio of 1, as values lower than 1 indicate longer paths in the proximity graphs, while values higher than 1

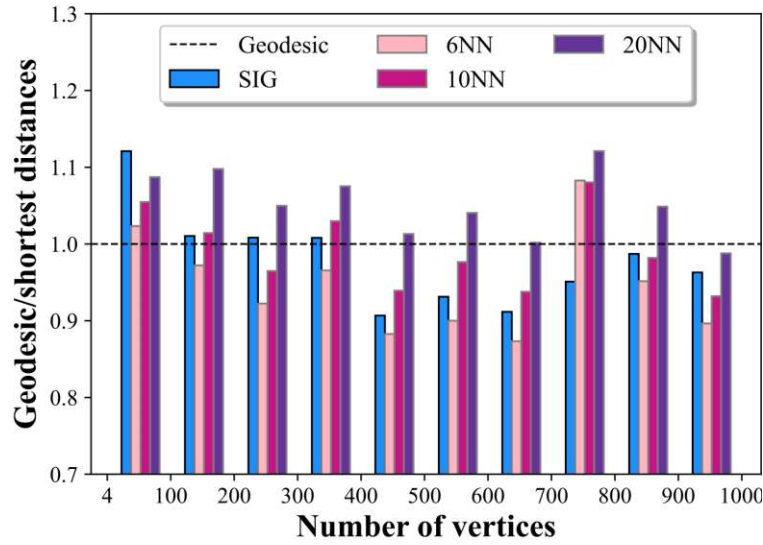


Figure 5.7: Ratio of the geodesic distance traced on the surface to the shortest path in the computed graph, ideally 1.

would imply the existence of shortcuts (too many edges) in the connectivity graphs. Our method is consistently close to the desired ratio of 1 across the tested meshes, without the need to tune any parameters. Even if for some of the tested input ranges some of the  $k$ NN graphs have better results, these are not universal and the user would need to adjust  $k$  depending on the usage, which is an issue solved by our method.

**Application: normal estimation.** We propose SIG as a connectivity graph that encodes surface properties well, and can act as an alternative for the commonly used  $k$ NN graphs. One usual application is normal computation for unstructured point clouds using PCA. Even though more advanced normal computation methods have been developed, constructing them using PCA on a connectivity graph still represents a widely used method and good results in this direction indicate an overall good representation of the underlying surface. Moreover, we are not aiming to improve the normal computation in general, but to show that our graph can be used in similar applications as  $k$ NN.

For each vertex, we computed the covariance matrix using its neighbors and extracted the normal as the normalized eigenvector corresponding to the smallest eigenvalue. We do not consistently orient the normals, as this can be done in a post-processing step and we are only interested in the angle difference when compared to the ground truth, which can be computed without the consistent orientation. For the original meshes, we used the triangulated faces to compute the normals. We do not use the ground truth edge graph since that would bias the normal computation in the direction of a specific triangulation. Instead, face-based normal computation takes into account more information about the surface, and not only a specific 1-ring. Then, we computed the



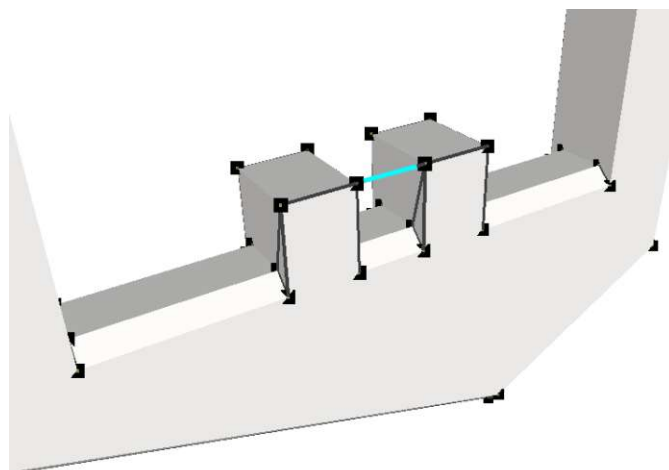


Figure 5.8: Sparse sampling of close sheets (the parallel tower-like structures in the center) generates edges across the surface. These skew our measurement of the geodesic ratio to the shortest distance, as the surface originally follows the  $U$ -structure, while our graph shortcuts it through the edge connecting the two towers, highlighted in blue. However, similar behavior is exhibited by the  $k$ NN graphs, since all of them are distance-based.

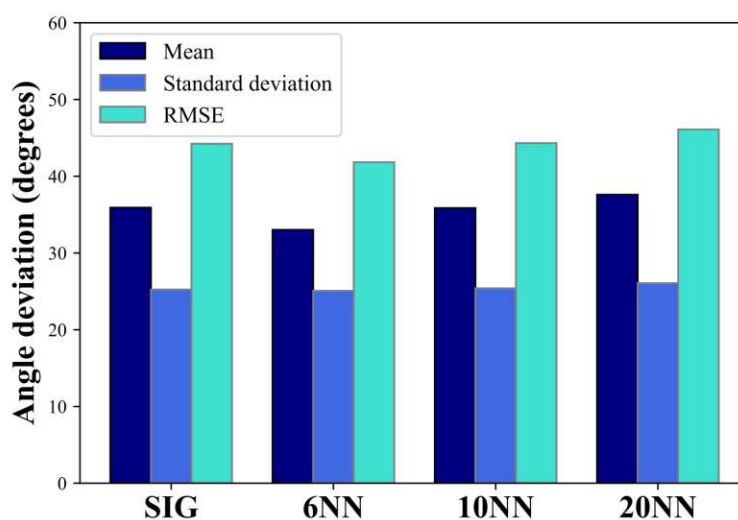
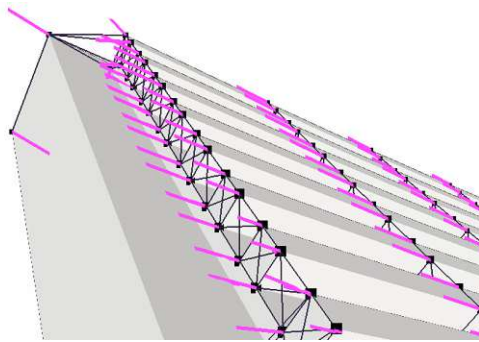
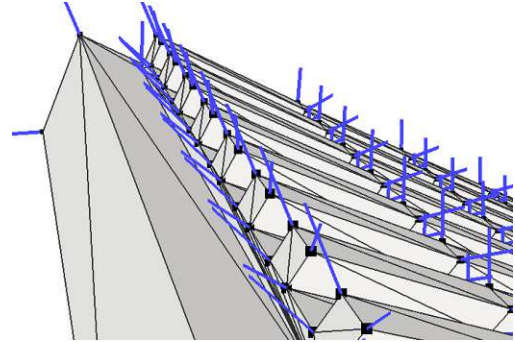


Figure 5.9: Angle variation between normals computed using PCA over connectivity graphs and original, face-based normals. We compute the mean and the standard deviation of the angle difference, and the root mean square error. All of the methods achieve similar deviations. The overall error is high due to the sharp angles in the input dataset.

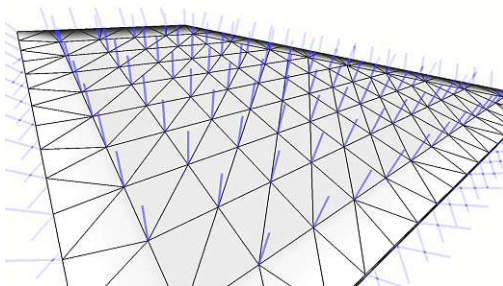


(a) Ground-truth normals computed using the incident faces of each vertex.

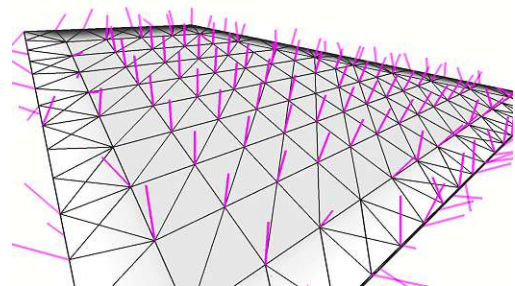


(b) Normals computed with PCA using the SIG connectivity.

Figure 5.10: Example of how CAD-like models with sparse sampling affect the computed connectivity, and hence, the normal computation. The sparse sampling creates parallel layers and normals are oriented accordingly since the top vertices do not get connected across layers. However, this is an issue encountered by distance-based methods in general.

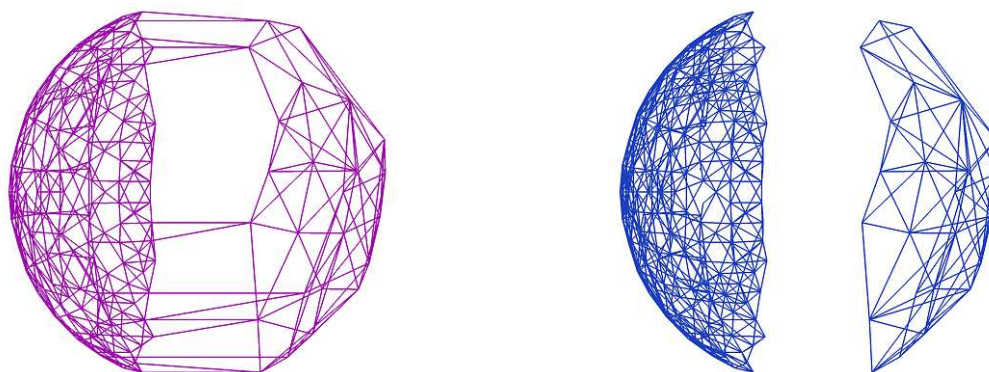


(a) Ground-truth normals computed using the incident faces of each vertex.



(b) PCA normals computed using SIG are very close to the ground truth.

Figure 5.11: Improved normal computation of our method for more uniformly sampled meshes. Normals are computed for every vertex, but since we do not orient them consistently, some of them are facing the other way and are not visible. Note that for our evaluation, orientation does not matter, and consistency is usually achieved with a post-processing step.



(a) 6NN creates bridges between the two hemispheres, since the graph does not consider local densities. Increasing the  $k$  value only aggravates the issue, as more bridge edges will be constructed.

(b) SIG only creates edges on each hemisphere, without crossing the gap between the two surface sheets.

Figure 5.12: Different sampling densities on two surface sheets that are close together - the two hemispheres are not connected as ground truth.

average angle deviation for SIG and the  $k$ NN graphs when compared to the ground truth normals. Results are presented in Figure 5.9, where the difference among the various tested graphs is less than 1 degree for all the metrics. Since the chosen dataset contains surfaces that are sparsely sampled, the normal computation achieved high errors for all graphs. Thus, for this metric, we resampled the chosen dataset (adding new vertices along edges longer than a specified threshold and retriangulating), obtaining models with up to 7k vertices. An example of how sparse sampling, which is also an issue in LIDAR scans, can cause problems, is presented in Figure 5.10, where there is not enough information for the vertices placed on edges to have their normal computed correctly. However, for well-sampled models, the normals are close to the ground truth - Figure 5.11. The overall angle deviation is still high for all methods, since some of the meshes exhibit sharp angles, for which the normal computation is also erroneous for all graphs.

**Timings** Due to the proved bounded radius in which SIG neighbors are found, our method's timings are comparable to  $k$ NN, as can be observed in Figure 5.13. We observe a linear increase in the computational time with the number of vertices, which is expected due to the linear nature of our algorithm. Our method is only slightly slower than 6NN, but achieves better results overall and manages to do so with many fewer edges.

All experiments have been performed using an AMD Ryzen 7 5800 processor. Both  $k$ NN and SIG graphs have been implemented using the `scipy.spatial.KDTree` in python, which uses Cython as the backbone, mimicking C++ performance. We believe the performance can be further increased for our code, but our current aim was to fairly compare the two methods in the same environment, showing similar performance.

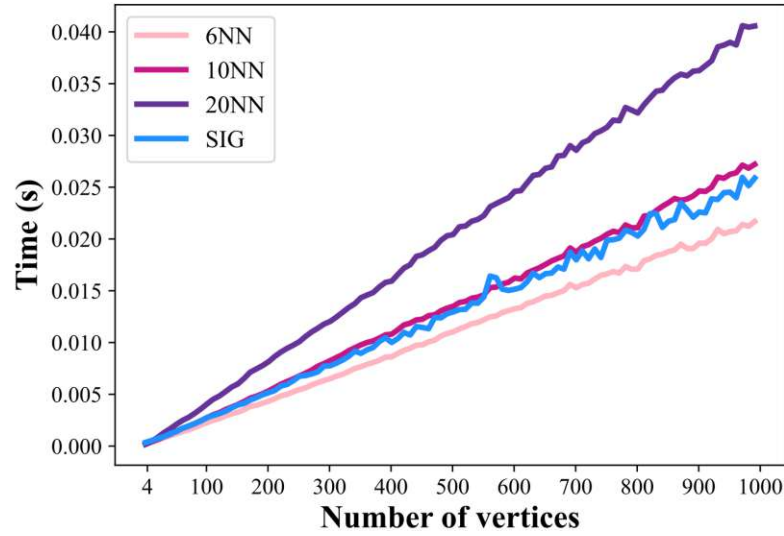


Figure 5.13: Timings of our method compared to  $k$ NN graphs for inputs with various numbers of vertices.

**Limitations.** Since our method depends on distances, non-uniform sampling may negatively affect the result, as is commonly the case with connectivity reconstruction. An example can be seen in Figure 5.14, where the spider’s legs and top of the body are uniformly sampled along parallel layers. For real-world point clouds, LIDAR scans can produce such artifacts. These configurations are difficult to handle for all methods, as they may fail to connect subsets of the point cloud. Such issues could be mitigated by employing an incremental SIG - using the next nearest neighbors until a specific vertex degree or average neighbor angle has been reached for the current graph.

However, in the case of different sampling densities on distinct parts of the same object, our method has an advantage over  $k$ NN. Our method is less likely to create edges between surface sheets that are geometrically close, but geodesically distant (Figure 5.12).

Our method is not robust to noise by design, as it computes a distance-based neighborhood, exhibiting similar drawbacks as  $k$ NN graphs. Noisy point clouds could still be used with our method, provided they have been cleaned in a pre-processing step using, for example, PointCleanNet [RLBG<sup>+</sup>20], the Bilateral Filter [DdF17] or Score-Based Denoising [LH21], as demonstrated in Figure 5.15.

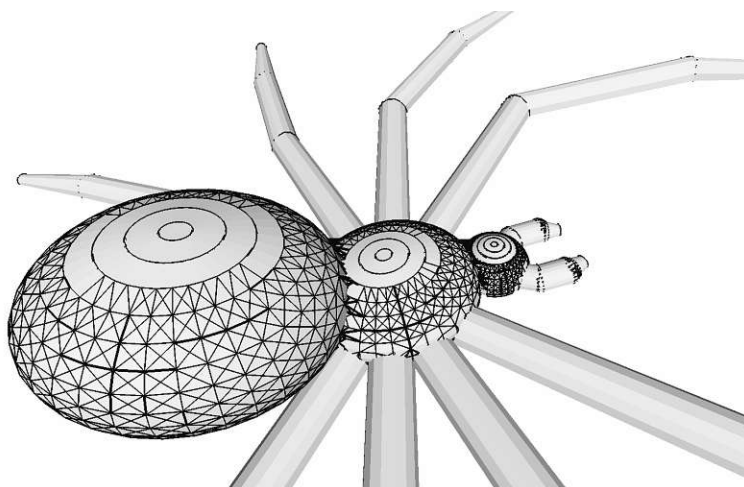


Figure 5.14: Parallel layers of sampling result in disconnected ring-like structures as can be observed on the spider’s body and legs. However,  $k$ NN graphs exhibit similar issues if the sampling is too sparse and nodes are clustered in layers.

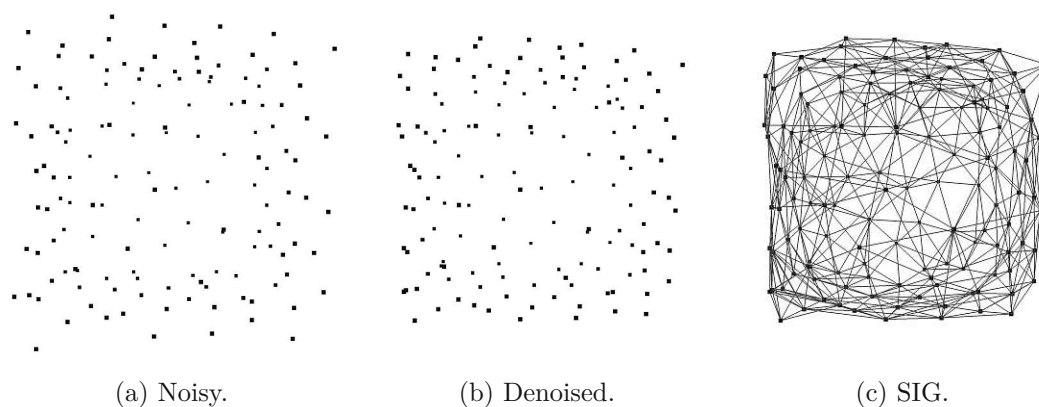


Figure 5.15: Denoising pipeline that would allow our method to give good results on noisy point clouds. The noisy point cloud of a cube is denoised in 5.15b using the Bilateral Filter [DdF17], and then used as input for our SIG computation, resulting in a graph that closely approximates the original cube.

## 5.5 Conclusion

We present an alternative to the commonly used  $k$ NN graphs for establishing the connectivity of point clouds: the SIG, a parameter-free proximity graph. In our work, we introduce novel spatial constraints for the extent of SIG edges, leveraging this new property to enhance its computational efficiency. We demonstrate the SIG's improved connectivity representation that is parameter-free. Moreover, as a sparse graph, it has a lower edge count, thus minimizing storage requirements. Consequently, it offers three key advantages over  $k$ NN: no need for parameter tuning, sparsity, and improved connectivity encoding. As an incidental application, we have shown that computed normals are of competitive quality to  $k$ NN.

**Future work.** We aim to utilize the advantage of parameter-free improved connectivity for surface reconstruction. Moreover, we are planning to use the SIG neighborhood (with possible extensions to  $n^{\text{th}}$  nearest neighbor) in graph-convolutional networks for learning from 3D point cloud data. We also plan to investigate how our method can create an advantage in other fields, such as motion planning and simulations.



# CHAPTER 6

## SING: Stability-Incorporated Neighborhood Graph

The final SIG exploration in this thesis deals with adding a parameter to the neighboring condition to allow for more flexibility. We observe that our approach can be considered a symmetrization of the neighborhood graph, and we analyze this new variant using topological data analysis tools. The contents of this chapter have been adapted from the paper “*SING: Stability-Incorporated Neighborhood Graph*” [MPO<sup>+</sup>24], accepted as a conference paper, and presented at SIGGRAPH Asia 2024.

### 6.1 Overview

We introduce the Stability-Incorporated Neighborhood Graph (SING), a novel density-aware structure designed to capture the intrinsic geometric properties of a point set. We improve upon the Spheres-of-Influence graph by incorporating additional features to offer more flexibility and control in encoding proximity information and capturing local density variations. Through persistence analysis on our proximity graph, we propose a new clustering technique and explore additional variants incorporating extra features for the proximity criterion. Alongside the detailed analysis and comparison to evaluate its performance on various datasets, our experiments demonstrate that the proposed method can effectively extract meaningful clusters from diverse datasets with variations in density and correlation. Our application scenarios underscore the advantages of the proposed graph over classical neighborhood graphs, particularly in terms of parameter tuning.

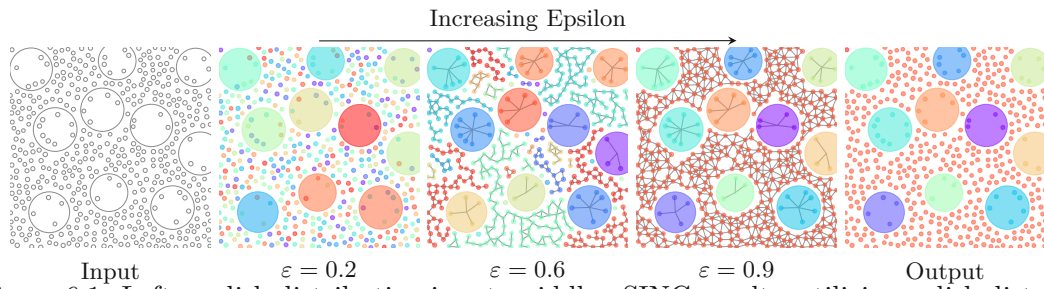


Figure 6.1: Left - a disk distribution input; middle - SING results, utilizing a disk distance from [ENMGC19], without any prior class information nor post-processing. Increasing  $\varepsilon$  (left to right) yields nested proximity graphs and diverse clustering effects, represented in random colors; right - output clusters.

## 6.2 Introduction

**Context and motivation** Clustering serves as a fundamental algorithmic procedure in data analysis, extensively employed in extremely diverse fields such as biology, astronomy, art, medicine, as well as computer graphics and vision. Despite these important applications, existing clustering algorithms suffer from a variety of drawbacks, and no universal solution has emerged. In this work, we propose an intuitive proximity criterion, leading to a stable and efficient clustering algorithm which consistently achieves accurate grouping across diverse data types. Experimental results demonstrate that our method, characterized by its simplicity and elegance, matches or exceeds the performance of state-of-the-art clustering algorithms across multiple application scenarios. It effectively handles density variations and multi-class data while robustly extending to noisy datasets using stable persistence-based parameter tuning.

**Key Idea** The proximity criterion introduced in this chapter represents a generalization of the classical neighborhood graph concept. Instead of simply connecting each point to its nearest neighbor, we utilize the distance to the nearest neighbor as a feature to determine its proximity to other points. This approach elegantly incorporates local density information into the proximity measure in a formal manner, offering an intuitive improvement to traditional methods.

**Validation via Clustering** Leveraging our proximity measure between element pairs allows for the application of various clustering techniques that depend on element similarities in different manners.

For instance, in center-based algorithms like  $k$ -means, a small set of potential cluster centers is initialized from the data and iteratively refined. In affinity propagation, data points interact via a graph structure to select a subset of points as representatives. Our proximity criterion based on local density offers computational efficiency, flexibility in terms of similarity constraints, and stability advantages derived from the persistence analysis approach.



**Stability via TDA tools** To better interpret our proximity criterion and to incorporate stability and robustness of the induced clustering with respect to noise on the overall data, we employ a common tool in topological data analysis (TDA): Persistent Homology. Specifically, we focus on order-one homology and the analysis of connected components, which simplifies the process. Persistent diagrams allow us to capture topological features of our proximity graph across different scales, leading to optimal parameter values and offering proven stability in the resulting clusters.

**Highlights of Contributions** The proposed proximity graph is characterized by its intuitive simplicity, ease of use, and adaptability to high-dimensional spaces and non-Euclidean metrics. Unlike many existing clustering methods, our SING clustering algorithm does not require prior knowledge of the number of clusters, as it optimizes the clustering outcome based on the stability of the created clusters in terms of parameters and persistence lifespans. Its stability is demonstrated based on the existing stability results of persistent homology. It has remarkable flexibility in terms of the type of distance that can be fed into it, opening up interesting avenues of research for proximity adaptation. This includes consideration for anisotropic metrics, surface curvature, or user-defined local constraint encoding via distance prescription for interactive analysis. We showcase the flexibility of our method via various application scenarios employing different metrics or analytical approaches. While we mainly focus on applications related to stipple clustering, our method finds broad applicability across different domains, such as data segmentation, multi-class disk distribution analysis, shape reconstruction, and network topology analysis. We briefly investigate these tangent directions, and we leave further exploration and research into metric choices as a potential source of inspiration for future work in the CG community. The source code is available online - <https://github.com/dianam76/SING>.

## 6.3 Method: proximity criterion & TDA

### 6.3.1 Context: Extending the SIG criterion

The foundation of our method is represented by the Spheres-of-Influence Graph (SIG) - Definition 8. We mention that its proximity criterion exists in any dimension and can be modified to account for different types of data, such as disks, since the definition of the graph is purely based on distances.

This definition can also be viewed as a symmetrization of the nearest neighbor graph since the existence of an edge relies on the properties of both endpoints. While this definition manages to encode proximity well, as shown in Chapter 5, it cannot adapt to varying properties of the data, such as different densities or correlations. In this work, we consider an extension of this definition from two aspects: by considering generalized distance functions  $d$ , as well as by adding a parameter  $\varepsilon$  to Equation 2.3's inequality:

$$d(a, b) \leq \varepsilon(nn(a) + nn(b)). \quad (6.1)$$

This allows us to change the connectivity of the proximity graph, capturing features with varying levels of significance. This parameter offers a novel degree of flexibility on top of the distance function  $d$  in defining the graph structure, allowing users to explore various cluster configurations. In Section 6.4.3, we also explore the possibility of employing local parameters. However, enabling users to select a parameter (even at the global level) for each dataset type poses challenges. Therefore, we offer an automatic parameter-tuning procedure based on tools from topological data analysis. As the creation of edges and their regrouping in our proximity graph relies on  $\varepsilon$ , we employ persistent homology to identify meaningful values for this parameter. We observe the formation of clusters (connected components in the SING graph) and analyze their duration. In the next sections, we first revisit key principles of Topological Data Analysis before presenting our contributions.

### 6.3.2 Background in TDA

In this section, we briefly repeat and review some of the material in TDA that will be used in this chapter. For a more detailed introduction to the subject, we refer the reader to standard textbooks such as [EH10, Oud15].

A *filtration*  $\mathcal{F}$  of a topological space  $K$  over a totally ordered set  $T$  is a family  $(\mathcal{F}_t)_{t \in T}$  of subspaces of  $K$  that are nested in terms of inclusion, that is:

$$\forall t \leq t' \in T, \mathcal{F}_t \subseteq \mathcal{F}_{t'}.$$

$\mathcal{F}$  is *simplicial* if  $K$  is a simplicial complex and if every  $\mathcal{F}_t$  is a subcomplex of  $K$ .

The *(Vietoris-)Rips filtration*  $\text{VR}$  is a popular choice of simplicial filtration in TDA applications. Given a point cloud  $P$  equipped with a dissimilarity  $d$ , it is a filtration of the full simplex  $K = 2^P$  (i.e., the power set of  $P$  viewed as a simplicial complex) indexed over  $T = \mathbb{R}^+$ , in which each simplex  $\sigma = \{p_0, \dots, p_m\} \subseteq P$  appears at index  $t = \max_{0 \leq i \leq j \leq m} d(p_i, p_j)$ . For any  $t \in \mathbb{R}^+$ , the subcomplex  $\text{VR}_t(P, d)$  of  $2^P$  formed by those simplices that appear before or at  $t$  is called the *(Vietoris-)Rips complex* of  $P$  of parameter  $t$ .

$\text{VR}(P, d)_t$  generalizes the  $t$ -ball graph of  $P$  in the following way: the vertices represent the points of  $P$ , and a simplex (not just an edge) exists if and only if its diameter is smaller than  $t$ . Varying the value of  $t$  from 0 to  $+\infty$  gives the Rips filtration of  $(P, d)$ .

When  $K$  is a finite simplicial complex (as is the case, e.g., for the Rips filtration and throughout this chapter), applying homology in degree  $r$  with coefficients in some fixed field  $\mathbf{k}$  to the filtration  $\mathcal{F}$  yields a family of finite-dimensional  $\mathbf{k}$ -vector spaces connected by  $\mathbf{k}$ -linear maps. This family is called a *persistence module*. It is known to admit in this setting a complete algebraic invariant called the *persistence barcode* of  $\mathcal{F}$  in degree  $r$ , denoted  $B_r(\mathcal{F})$ , which takes the form of a finite multiset of intervals  $[a_i, b_i)$ , each of which encodes the lifespan of some topological feature of degree  $r$  appearing in the filtration. Note that multiple features can have identical lifespans, hence the multiset structure of

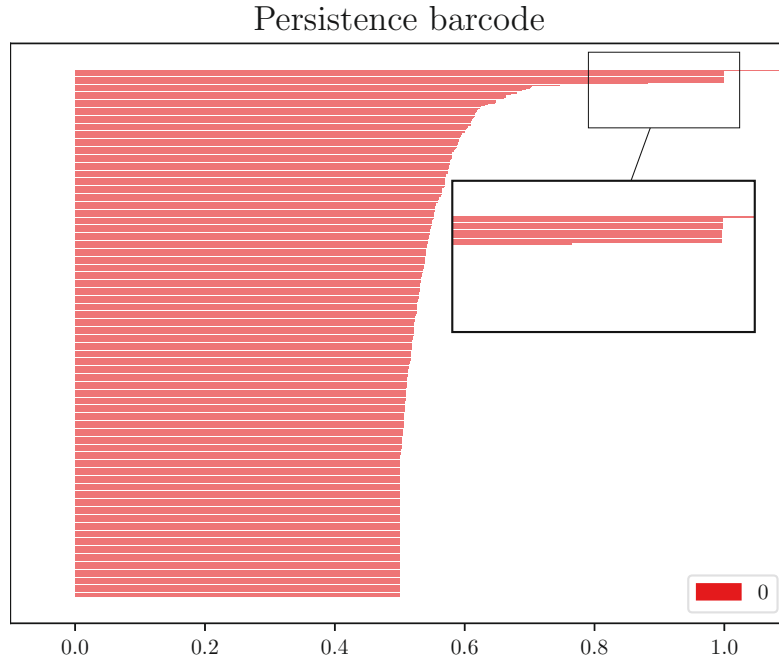


Figure 6.2: Persistence Barcode. The zoomed-in region shows the stable region we are interested in. In a stable region, the number of connected components does not change in the specific interval, meaning that, most probably, those clusters carry some geometric meaning since  $\varepsilon$  needs to change a lot before collapsing connected components.

the barcode. Topological features can be, for instance, connected components ( $r = 0$ ), handles/tunnels ( $r = 1$ ), enclosed voids ( $r = 2$ ), or many other things. See Figure 6.2.

The *persistence diagram* of  $\mathcal{F}$  in degree  $r$ , noted  $\text{PD}_r(\mathcal{F})$ , is an alternative graphical representation of the barcode as a multiset of points above the diagonal  $y = x$  in the plane. More precisely, every copy of the interval  $[a_i, b_i)$  in  $B_r(\mathcal{F})$  becomes a copy of the point  $(a_i, b_i)$  in  $\text{PD}_r(\mathcal{F})$ , and vice-versa. We will let  $r = 0$ , focusing on connected components in the filtration, and we will henceforth omit the parameter in our notations.

As multisets of points in the plane, persistence diagrams can be viewed as discrete measures in which each diagram point has unit mass. Such measures may have different total masses, therefore using classic distances between probability measures requires some adaptation. Typically, one enriches each diagram with infinitely many copies of the diagonal  $y = x$  to even out the total masses, making no distinction between different infinite values. In this context, the *bottleneck distance* is the Wasserstein distance  $W^\infty$  between the enriched diagrams:

$$d_b(\text{PD}(\mathcal{F}), \text{PD}(\mathcal{G})) := W^\infty(\text{PD}^+(\mathcal{F}), \text{PD}^+(\mathcal{G})),$$

where  $\text{PD}^+(\cdot)$  denotes the persistence diagram enriched with infinitely many copies of the diagonal  $y = x$ .

The persistence barcode or diagram of the Rips filtration exhibits the consistency (or lack thereof) of topological features hidden in the dataset across scales, thus it helps identify relevant scales at which to analyze or process the data. This methodology is backed up by a sound stability theory, in particular by the fact that the map sending a point cloud  $P$  to the persistence diagram of its Rips filtration is provably Lipschitz continuous. In our proofs, we will use a more generic version of this stability theory, phrased as follows:

**Theorem 3** (stability [CSEH07]). *Let  $K$  be a finite simplicial complex, and let  $f, g: K \rightarrow \mathbb{R}$  assign a real value to each simplex in  $K$ . Then, the two families of sublevel-sets  $f^{-1}((-\infty, t])$  and  $g^{-1}((-\infty, t])$  for  $t$  ranging over  $\mathbb{R}$  define two simplicial filtrations  $\mathcal{F}, \mathcal{G}$  of  $K$  such that:*

$$d_b(\text{PD}(\mathcal{F}), \text{PD}(\mathcal{G})) \leq \max_K |f - g|.$$

### 6.3.3 Key notion: density-sensitive semimetric

The proximity criterion introduced in Equation (6.1) can be interpreted as measuring the dissimilarity  $\hat{d}_P(a, b)$  between the points  $a, b$  against a threshold  $\varepsilon$  as follows:

$$\hat{d}_P(a, b) := \frac{d(a, b)}{nn(a) + nn(b)} \leq \varepsilon. \quad (6.2)$$

Observe that  $\hat{d}_P$  is a density-weighted version of the original metric  $d$ : consider indeed the 1-dimensional nearest-neighbor density estimator [Sil18, §5.2], which is defined inversely proportional to the distance to the nearest data point. Dividing the distance  $d(a, b)$  by  $nn(a) + nn(b)$  as in Equation (6.2) is equivalent, up to a constant factor, to multiplying  $d(a, b)$  by the harmonic mean of the 1-dimensional nearest-neighbor density estimates at  $a$  and  $b$ .

We also note that  $\hat{d}_P$  is defined only when  $P$  has at least 2 points, otherwise  $nn(\cdot)$  itself is undefined. Note also that  $\hat{d}_P$  is a semimetric, not a metric, as it may not satisfy the triangle inequality. For instance, taking  $P = \{-\eta, 0, 1/2, 1, 1 + \eta\}$  on the real line gives  $\hat{d}_P(0, 1) = 1/2\eta$  and  $\hat{d}_P(0, 1/2) + \hat{d}_P(1/2, 1) = 2/(1 + 2\eta)$ , which infringes the triangle inequality as soon as  $\eta < 1/2$  (and in fact makes the infringement as bad as it can be since  $\hat{d}_P(0, 1) \rightarrow +\infty$  while  $\hat{d}_P(0, 1/2) + \hat{d}_P(1/2, 1) \rightarrow 2$  as  $\eta \rightarrow 0$ ). However, the triangle inequality will not be needed in the following derivations.

### 6.3.4 SING and its connection to TDA

The Stability-Incorporated Neighborhood Graph (or SING for short) of  $(P, d)$  of parameter  $\varepsilon$  is defined as the  $\varepsilon$ -neighborhood graph of  $P$  in the semimetric  $\hat{d}_P$ . By design, its connectivity adapts to the local density of the data. Moreover, it coincides with the 1-skeleton graph of  $\text{VR}_\varepsilon(P, \hat{d}_P)$ , so one can use the Rips filtration of  $(P, \hat{d}_P)$  and its persistence diagram to determine a suitable value for parameter  $\varepsilon$ .

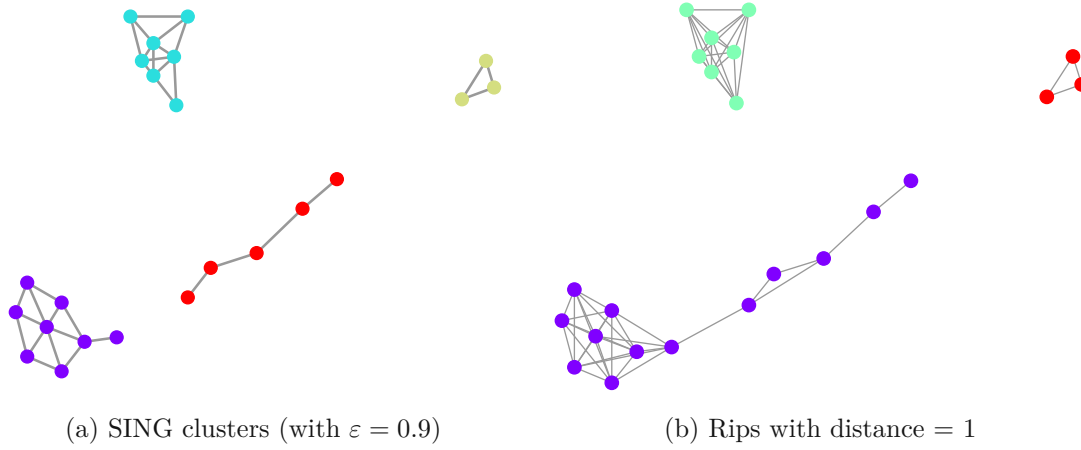


Figure 6.3: Connected components of SING compared to the Rips complex.

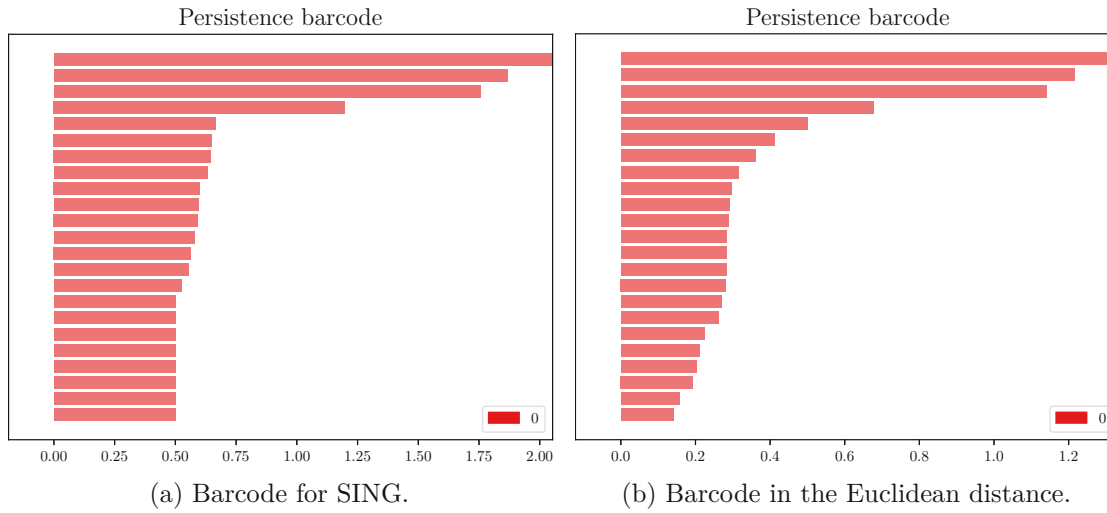


Figure 6.4: Barcodes of the Rips filtrations for SING and for the  $\varepsilon$ -neighborhood graph in the ambient Euclidean distance, respectively, on the data of Figure 6.3. The barcode for SING indicates the possibility of having 3 or 4 clusters, while the other barcode only indicates the possibility of having 3 clusters.

Thus, the SING enjoys the same ease of use and flexibility as  $\varepsilon$ -neighborhood graphs in general while addressing their lack of sensitivity to the local density. Besides, as shown in Section 6.4.5 (Proposition 1), the Rips filtration associated with SING comes with theoretical stability guarantees, as does the Rips filtration in the ambient metric  $d$ , although in a weaker form.

Figure 6.4 illustrates the benefit of replacing the metric  $d$  by the weighted semimetric  $\hat{d}_P$  in the  $\varepsilon$ -neighborhood graph construction, in terms of the expressivity of the persistence barcode of its associated Rips filtration, missing the possibility of the four clusters.

We note that for some contexts, we might be interested in defining the SING complex accordingly as the  $\text{VR}_\varepsilon(P, \hat{d}_P)$ , whose 1-skeleton, as mentioned before, corresponds to our introduced SING graph. This is nicely illustrated in our example of Figure 6.3. Interestingly, the left-hand side figure shows both SING and the  $\varepsilon$ -neighborhood graph in the Euclidean distance, which coincide but for different parameter values. However, the range of values that produce this "good" graph with SING is larger than its analog with the  $\varepsilon$ -neighborhood graph in the Euclidean distance—hence the interest in the SING.

As a last remark, let us once again highlight the flexibility of SING in terms of the input metric. In a configuration similar to the one in Figure 6.3, if the desired clusters are indeed the ones presented on the right, incorporating an anisotropic metric favoring the diagonal direction would also enable us to achieve this clustering while maintaining stability.

## 6.4 SING Features and Advantages

We will now further analyze the graph and its properties, especially in comparison with other proximity graphs or other clustering techniques, highlighting the benefits of using such a stability-incorporated neighborhood graph.

### 6.4.1 Intrinsic Geometric Features

The graph naturally encodes proximity, and even with a global  $\varepsilon$  parameter for the entire dataset, the graph definition still inherently captures local density. Moreover, this graph is unlikely to suffer from long edges, spanning the entire input or extremely high-degree nodes (except for adversarial cases – samples on a circle with the center of the circle). Unlike  $k$ NN graphs, we do not impose strict bounds on vertex degrees or distances between points. We compare our graph to popular proximity graphs in Figure 6.5.

SING can effectively capture both complex and regular geometric patterns, which are typically challenging to analyze and generate. Figure 6.6 illustrates such a configuration, which finds motivation in various contexts such as simulation. For example, in a spring system aiming to preserve a specific shape, our proximity graph seems more relevant to be employed for construction. Also, as expected, in the presence of noise (as long as the extent is not too large), our graph still captures the original shape of the data, Figure 6.9.

### 6.4.2 Other distances

Since the definition of our graph only depends on distance computations, it can be used in any metric space and with various distance metrics. The persistence analysis benefits from the same advantages of only requiring distances between vertices for the computation.

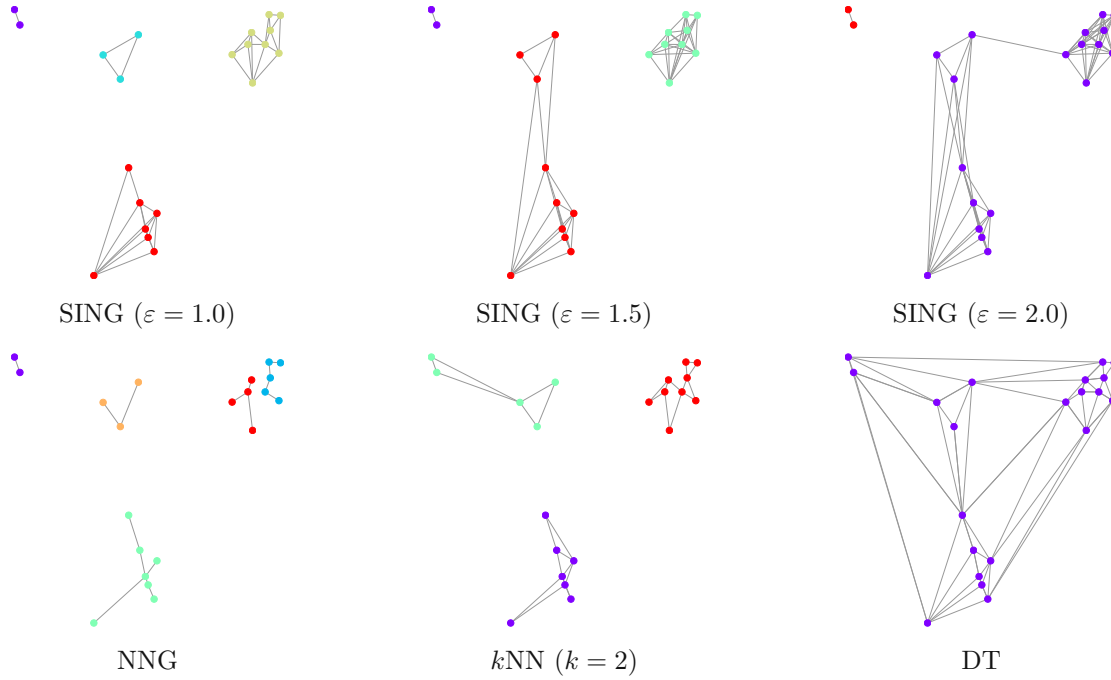
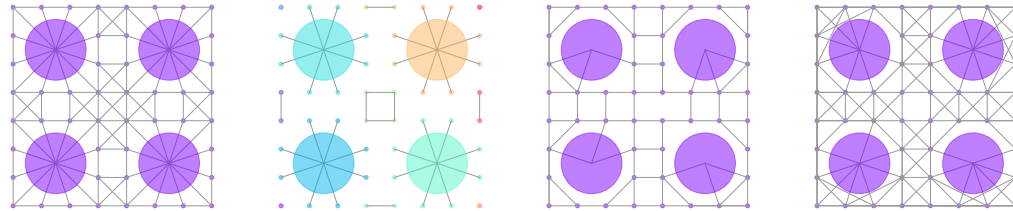


Figure 6.5: As a proximity graph, SING encodes the evolution of various clusters without being limited by a predefined number of neighbors (as kNN) and offers more flexibility compared to Delaunay Triangulation (DT).



(a) SING with a single connected component. Using radius information, we further extract the types of disks in the distribution.

(b) SING for  $\varepsilon = 0.55$ , where the main clusters are already forming - the main disks, along with grid elements.

(c) Nearest Neighbor Graph does not manage to capture enough information to fully encode the geometric features of the data.

(d) KNN for  $k = 6$  does not manage to capture all the large disk connections, even if the grid pattern is almost fully represented.

Figure 6.6: Structured pattern data connected using various proximity graphs.



This flexibility allows us to compute the SING for other metrics, such as the disk-based distance of [ENMGC19], applied to disk distribution clustering. For two disks with radii  $r_1$  and  $r_2$ , let  $d$  be the distance between the disks' centers, assuming, without loss of generality, that  $r_1 \geq r_2$ . Their disk distance is:

$$d_{\text{disks}} = \begin{cases} f/(4r_1 - 4r_2) & d \leq r_1 - r_2 \\ (f - 4r_1 + 7r_2)/(3r_2) & r_1 - r_2 < d \leq r_1 + r_2 \\ f - 4r_1 + 2r_2 + 3 & \text{otherwise,} \end{cases} \quad (6.3)$$

$$f = \max(d + r_1 + r_2, 2r_1) \quad (\text{extent}) \quad (6.4)$$

$$- \text{clip}(r_1 + r_2 - d, 0, 2r_2) \quad (\text{overlap}) \quad (6.5)$$

$$+ d + r_1 - r_2. \quad (6.6)$$

We feed this distance to our proximity criterion and conduct experiments for disk distribution clustering. The original data includes information about the class of each disk. However, in the absence of such information, the SING is perfectly able to extract similarities between different disks and output the relevant classification.

This is done by analyzing the persistence barcode of the data and using the stable intervals as guidance for meaningful  $\varepsilon$  choices, leading to relevant clustering of data, Figure 6.1. As can be seen in the final result, the large disks are only connected to the smaller ones, which they overlap, while the outer smaller disks are all connected in a single component, which exactly matches the input behavior. By using further details from the distance metric (i.e., classifying edges by the type of overlap they encode), we can easily extract the three classes present in the input. However, this dataset also showcases the limitations of a method that is only based on proximity – the single disk in the top-left corner, which is too far away from disks of the similar class to be easily clustered with them.

### 6.4.3 Local density adaptation: flexible variant

**Density constraints.** In some point patterns, differentiating between different areas is not encoded by geometrically separating the points from different sections but by changing the density of the pattern (e.g., sparse points to represent the background of a stipple art image). To allow for different density encodings in our graph, we propose introducing a new parameter that enables edge creation only if the density is similar enough. We implement this by multiplying by the ratio between the nearest neighbors - Algorithm 6.1. We raise this ratio to a user-defined power, enforcing dissimilar densities to increase the measurement between points - Figure 6.7. Setting the density parameter to 0 brings us back to the original formulation.

### 6.4.4 Complexity analysis

**Nearest neighbor search complexity** Note that achieving sub-linear query time for nearest neighbor search in arbitrary dimension is a notoriously hard problem, especially in



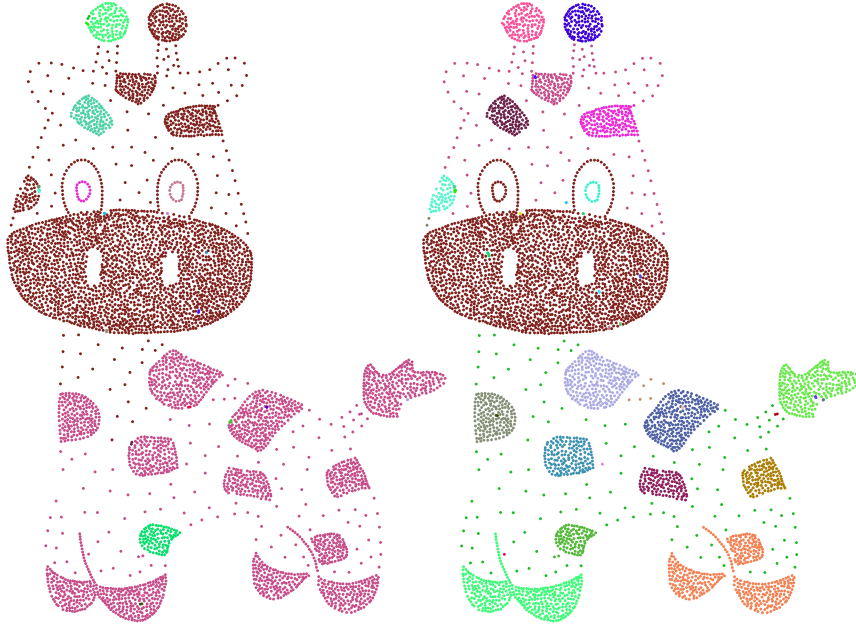


Figure 6.7: On the left, clustering results without the incorporation of the density parameter, where samples are linked based on our proximity criterion. On the right, results incorporating the density parameter, effectively clustering the spots on the giraffe. Following a connected-component-based splitting, we can further refine the clustering.

---

**Algorithm 6.1:** Density SING computation.

---

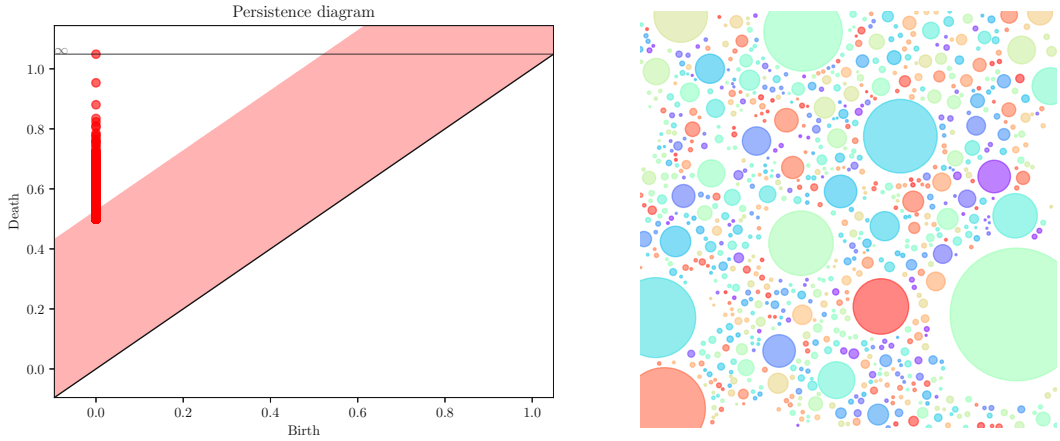
**Data:**  $P = \{p_1, \dots, p_n\} \subset \mathbb{R}^2$ , density  $\rho$   
**Result:**  $\text{SING} = \{(a, b) : a, b \in P, a \neq b\}$

```

1 SING := {};
2 Find  $nn(p) \forall p \in P$ ;
3 for each pair  $(a, b) \mid a, b \in P, a \neq b$  do
4    $\hat{d}_P(a, b) := \|a - b\|_2 / (nn(a) + nn(b)) \times (\max(nn(a), nn(b)) / \min(nn(a), nn(b)))^\rho$ 
5 end
6  $PD := \text{ComputePersistenceDiagram}(\hat{d}_P)$ ;
7  $\varepsilon := \text{ExtractOptimalValue}(PD)$ ;
8 for each pair  $(a, b) \mid a, b \in P, a \neq b$  do
9   if  $\hat{d}_P(a, b) \leq \varepsilon$  then
10    SING := SING  $\cup \{(a, b)\}$ 
11  end
12 end

```

---



(a) Confidence interval for given dataset.

(b) Example of noisy results for  $\varepsilon = 0.5$ , within the noise interval.

Figure 6.8: Stability: extracting a confidence interval for our method using TDA. On the left, the red diagonal band spans the noisy values of our parameter, while on the right, we showcase clustering results for a noisy value. Small variations around this value result in multiple changes in terms of the number of clusters and no clear cluster structure is visible.

high dimensions where the concentration of measure phenomena occur. In low dimensions there are classic data structures that allow for nearest neighbor search in sub-linear time. For instance, in 2D, one can build a hierarchical Delaunay triangulation in  $O(n \log n)$  time and then use it for  $O(\log n)$  time nearest neighbor queries. In 3D, this approach already has quadratic time complexity for the construction step in the worst case. Lastly, in arbitrary dimensions, locality-sensitive hashing allows achieving quasi-linear construction and then sub-linear query time.

**SING Complexity** For  $\varepsilon \leq 1$ , the number of edges in SING is linear in the input size, resulting in an efficient algorithm in terms of space complexity. Regarding time complexity, the current  $O(n^2)$  implementation of the SING algorithm, outlined in Algorithm 1, may be reduced to sub-quadratic construction time and even  $O(n \log n)$  time in small dimensions (typically 2D), using classic data structures.

#### 6.4.5 Stability and Robustness

Let  $(X, d)$  be a metric space. Given a point cloud  $P$  in  $X$ , i.e. a finite subset  $P \subseteq X$ , we write  $|P|$  for its cardinality and  $\delta_P$  for its minimum pairwise distance:  $\delta_P := \min_{p \neq q \in P} d(p, q) > 0$ . We equip the set of point clouds in  $X$  with the Wasserstein distance  $W^\infty$ , which is valued in  $\mathbb{R}^+ \cup \{+\infty\}$  and is finite whenever the two point clouds

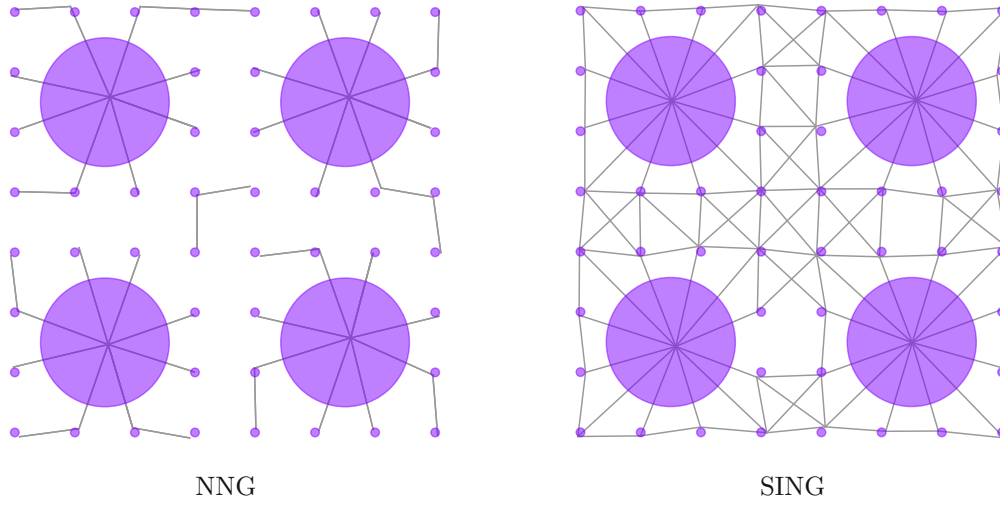


Figure 6.9: Proximity graphs on pattern data perturbed with noise – SING still captures the original connectivity, showcasing the stability of the method.

under consideration have the same cardinality:

$$\forall P, Q \subseteq X, W^\infty(P, Q) := \begin{cases} \min_{\substack{\gamma: P \rightarrow Q \\ \text{bijection}}} \max_{p \in P} d(p, \gamma(p)) & \text{if } |P| = |Q| \\ +\infty & \text{otherwise.} \end{cases} \quad (6.7)$$

This turns the set of point clouds into an extended metric space. Meanwhile, we equip the space of persistence diagrams with the bottleneck distance  $d_b$ . Our stability guarantees are stated as follows:

**Proposition 1.** *The map  $P \mapsto \text{PD}(\text{VR}(P, \hat{d}_P))$  is continuous on the subspace of point clouds of cardinality at least 2 in  $X$ .*

The condition that  $|P| \geq 2$  in the statement is not an artifact: it comes from the fact that  $\hat{d}_P$  is not defined when  $|P| < 2$ —see Equation (6.2). The proof of Proposition 1 relies on the stability theorem for persistence diagrams (Theorem 3) and is provided in the following subsection. Note that it requires the triangle inequality for the ambient metric  $d$  in  $X$ . However, in practice, as in the disk distance experiments, the proximity criterion is perfectly definable for semimetrics on  $X$ , leading to relevant graphs and desirable clusters - Figure 6.9.

**Remark** The stability guarantee offered for SING by Proposition 1 is weaker than the one known for  $\varepsilon$ -neighborhood graphs in the ambient metric  $d$  [CDSO14]. Primarily

because it is only a continuity result, not a Lipschitz continuity result: our proof, although is not tight, exhibits enough of the structure of  $\hat{d}_P$  to suggest that  $\hat{d}_P$  itself is not globally Lipschitz continuous but only locally Lipschitz continuous, with a local Lipschitz constant that grows with  $1/\delta_P$  – see Equation 6.12. Secondly, our stability guarantee is expressed in terms of the Wasserstein distance on point clouds in  $X$ , not in terms of the usual Hausdorff distance: in fact, there is no analog of Proposition 1 when the space of point clouds in  $X$  is equipped with the Hausdorff distance. Here is a counterexample. Given any positive  $\varepsilon \leq 1/3$ , consider two point clouds  $P = \{0, 1\}$  and  $Q = \{0, 1, 1 + \varepsilon\}$  on the real line  $\mathbb{R}$ . Their Hausdorff distance is  $\varepsilon$ . Meanwhile, we have  $\hat{d}_P(0, 1) = 1/2$  so  $\text{PD}(\text{VR}(P, \hat{d}_P)) = \{(0, +\infty); (0, 1/2)\}$ , whereas  $\hat{d}_Q(0, 1) = 1/(1 + \varepsilon)$ ,  $\hat{d}_Q(1, 1 + \varepsilon) = 1/2$  and  $\hat{d}_Q(0, 1 + \varepsilon) = 1$  so  $\text{PD}(\text{VR}(Q, \hat{d}_Q)) = \{(0, +\infty); (0, 1/2); (0, 1/(1 + \varepsilon))\}$ , hence  $d_b(\text{PD}(\text{VR}(P, \hat{d}_P)), \text{PD}(\text{VR}(Q, \hat{d}_Q))) = \frac{1-\varepsilon}{2(1+\varepsilon)}$ . This quantity goes to  $1/2$  while the Hausdorff distance between  $P$  and  $Q$  goes to zero as  $\varepsilon \rightarrow 0^+$ . As a consequence, the map  $P \mapsto \text{PD}(\text{VR}(P, \hat{d}_P))$  is not continuous in the Hausdorff distance.

### Stability Proof

In this section, we provide proof for Proposition 1. Let  $(X, d)$  be a metric space. For clarity, since the proof involves different point clouds in  $X$  equipped with different (semi-)metrics, we always write  $nn(x)$  explicitly as the distance of  $x$  to its nearest neighbor in its point cloud. For this, we use the notation  $\text{NN}_P(x)$ , which refers to the nearest neighbor of  $x$  in  $P$  in the ambient metric  $d$ .

*Proof.* Let  $P \subseteq X$  be of size  $n \geq 2$ . Given any point cloud  $Q \subseteq X$  such that  $W^\infty(P, Q) < \delta_P/4$  (this implies in particular that  $Q$  has the same cardinality as  $P$  since  $\delta_P/4 < +\infty$ ), let us write  $\eta$  for the distance  $W^\infty(P, Q)$  and  $\gamma: P \rightarrow Q$  for a bijection that realizes this distance as per (6.7). By the triangle inequality, we have for all  $p \in P$ :

$$\begin{aligned} d(\gamma(p), \text{NN}_Q(\gamma(p))) &\leq d(\gamma(p), \gamma(\text{NN}_P(p))) \\ &\leq d(\gamma(p), p) + d(p, \text{NN}_P(p)) \\ &\quad + d(\text{NN}_P(p), \gamma(\text{NN}_P(p))) \\ &\leq d(p, \text{NN}_P(p)) + 2\eta. \end{aligned}$$

By symmetry ( $\gamma$  being a bijection), we obtain:

$$|d(\gamma(p), \text{NN}_Q(\gamma(p))) - d(p, \text{NN}_P(p))| \leq 2\eta. \quad (6.8)$$

In particular, we have:

$$\frac{1}{2} \delta_P < \delta_P - 2\eta \leq \delta_Q \leq \delta_P + 2\eta < \frac{3}{2} \delta_P. \quad (6.9)$$

With this being established, we can now compute for all  $p, q \in P$ :

$$\begin{aligned}
\hat{d}_Q(\gamma(p), \gamma(q)) &\stackrel{(6.2)}{=} \frac{d(\gamma(p), \gamma(q))}{d(\gamma(p), \text{NN}_Q(\gamma(p))) + d(\gamma(q), \text{NN}_Q(\gamma(q)))} \\
&\stackrel{(6.8)}{\leq} \frac{d(p, q) + 2\eta}{d(p, \text{NN}_P(p)) + d(q, \text{NN}_Q(q)) - 4\eta} \\
&= \hat{d}_P(p, q) \frac{1}{1 - \frac{4\eta}{d(p, \text{NN}_P(p)) + d(q, \text{NN}_Q(q))}} \\
&\quad + \frac{2\eta}{d(p, \text{NN}_P(p)) + d(q, \text{NN}_Q(q)) - 4\eta} \\
&\stackrel{(6.9)}{\leq} \hat{d}_P(p, q) \left( 1 + \frac{8\eta}{d(p, \text{NN}_P(p)) + d(q, \text{NN}_Q(q))} \right) \\
&\quad + \frac{2\eta}{d(p, \text{NN}_P(p)) + d(q, \text{NN}_Q(q)) - 4\eta} \\
&\stackrel{(6.9)}{\leq} \hat{d}_P(p, q) \left( 1 + \frac{16\eta}{3\delta_P} \right) + \frac{4\eta}{3\delta_P - 8\eta}.
\end{aligned}$$

Thus, for all  $p, q \in P$  we have:

$$\hat{d}_Q(\gamma(p), \gamma(q)) - \hat{d}_P(p, q) \leq \frac{16\eta}{3\delta_P} \max_{P \times P} \hat{d}_P + \frac{4\eta}{3\delta_P - 8\eta}. \quad (6.10)$$

The same calculation yields for all  $\gamma(p), \gamma(q) \in Q$ :

$$\hat{d}_P(p, q) - \hat{d}_Q(\gamma(p), \gamma(q)) \leq \frac{16\eta}{3\delta_P} \max_{Q \times Q} \hat{d}_Q + \frac{4\eta}{3\delta_P - 8\eta}. \quad (6.11)$$

By combining (6.10) and (6.11), we get:

$$\begin{aligned}
\max_{P \times P} |\hat{d}_P - \hat{d}_Q(\gamma(\cdot), \gamma(\cdot))| &\leq \varepsilon \\
\text{where } \varepsilon &= \frac{16\eta}{3\delta_P} \left( 1 + \frac{16\eta}{3\delta_P} \right) \max_{P \times P} \hat{d}_P + \left( 1 + \frac{16\eta}{3\delta_P} \right) \frac{4\eta}{3\delta_P - 8\eta}.
\end{aligned} \quad (6.12)$$

Notice that the right-hand side of the inequality depends only on  $P$ , which is fixed, and that it goes to zero as  $\eta \rightarrow 0$ , i.e., as  $Q$  converges to  $P$  in the metric  $W^\infty$ . We can then use this inequality in conjunction with the stability theorem for persistence diagrams (Theorem 3), to show that the bottleneck distance between the persistence diagrams of the Vietoris-Rips filtrations of  $(P, \hat{d}_P)$  and  $(Q, \hat{d}_P)$  goes to zero as  $Q$  converges to  $P$ . We now formalize this argument.

The bijection  $\gamma: P \rightarrow Q$  induces an isomorphism of simplicial complexes between the full simplex over  $P$  (noted  $2^P$ ) and the full simplex over  $Q$  (noted  $2^Q$ ), that is, a bijection  $\Gamma: 2^P \rightarrow 2^Q$  that preserves the dimensions of the simplices and their incidence relations. By pulling back the dissimilarity  $\hat{d}_Q$  onto  $P$  via  $\gamma$ , we make  $\Gamma$  preserve also the appearance

time of each simplex in the Rips filtration and thus become an isomorphism of simplicial filtrations between  $\text{VR}(P, \hat{d}_Q(\gamma(\cdot), \gamma(\cdot)))$  and  $\text{VR}(Q, \hat{d}_Q)$ . Hence,

$$\text{PD}(\text{VR}(P, \hat{d}_Q(\gamma(\cdot), \gamma(\cdot)))) = \text{PD}(\text{VR}(Q, \hat{d}_Q)). \quad (6.13)$$

Now, by (6.12) the two filtrations of  $P \times P$  given by the families of sublevel-sets of the maps  $\hat{d}_P$  and  $\hat{d}_Q(\gamma(\cdot), \gamma(\cdot))$  are  $\varepsilon$ -interleaved. It follows that the extensions of the two maps to the full simplex  $2^P$ , defined on each face  $\sigma = \{p_0, \dots, p_r\} \subseteq P$  by  $\max_{\sigma \times \sigma} \hat{d}_P$  and  $\max_{\sigma \times \sigma} \hat{d}_Q(\gamma(\cdot), \gamma(\cdot))$  respectively, also have  $\varepsilon$ -interleaved sublevel-sets filtrations. By construction, these filtrations are precisely  $\text{VR}(P, \hat{d}_P)$  and  $\text{VR}(P, \hat{d}_Q(\gamma(\cdot), \gamma(\cdot)))$ , respectively. Therefore, by Theorem 3:

$$d_b \left( \text{PD}(\text{VR}(P, \hat{d}_P)), \text{PD}(\text{VR}(P, \hat{d}_Q(\gamma(\cdot), \gamma(\cdot)))) \right) \leq \varepsilon. \quad (6.14)$$

Combining (6.13) and (6.14), we get:

$$\begin{aligned} & d_b \left( \text{PD}(\text{VR}(P, \hat{d}_P)), \text{PD}(\text{VR}(Q, \hat{d}_Q)) \right) \\ & \leq \varepsilon = \frac{16\eta}{3\delta_P} \left( 1 + \frac{16\eta}{3\delta_P} \right) \max_{P \times P} \hat{d}_P + \left( 1 + \frac{16\eta}{3\delta_P} \right) \frac{4\eta}{3\delta_P - 8\eta}. \end{aligned}$$

Since  $P$  is fixed here, the right-hand side of the inequality goes to zero as  $\eta \rightarrow 0$ , i.e., as  $Q$  converges to  $P$  in the metric  $W^\infty$ . This shows that the map  $P \mapsto \text{PD}(\text{VR}(P, \hat{d}_P))$  is continuous at this particular  $P$ . Since this is true for any  $P \subseteq X$  of size  $n \geq 2$ , the proof is complete.  $\square$

## 6.5 Results and Validation

**Parameters** In order to take advantage of the connection to TDA, we employ further analysis to extract stable intervals for our parameter. In persistence analysis, components that disappear shortly after creation are considered noise. Visually, the persistence diagram encodes this as points very close to the diagonal. We aim to extract a stable interval for our parameter, where we are certain, up to a confidence ratio, that we will not encounter noisy components. We do this by subsampling our input set multiple times, with repetition, and computing the persistence diagram of each subset. We compute the distance from each of these subsampled diagrams to the original full input diagram and extract a band of possible noise around the diagonal (Figure 6.8). We then consider viable  $\varepsilon$  values the ones outside the computed noise band around the diagonal.

**Performance & Implementation Details** All experiments were performed using an AMD Ryzen 7 5800 CPU. We implemented our method in `python`, using the Gudhi library for TDA and various packages from `scikit` and `sclearn` for data structures and the methods we compared to. The runtime of our current, non-optimized implementation spans from 1.5s for an input size of 500 points to 130s for 50k points. The source code is available online - <https://github.com/dianam76/SING>.

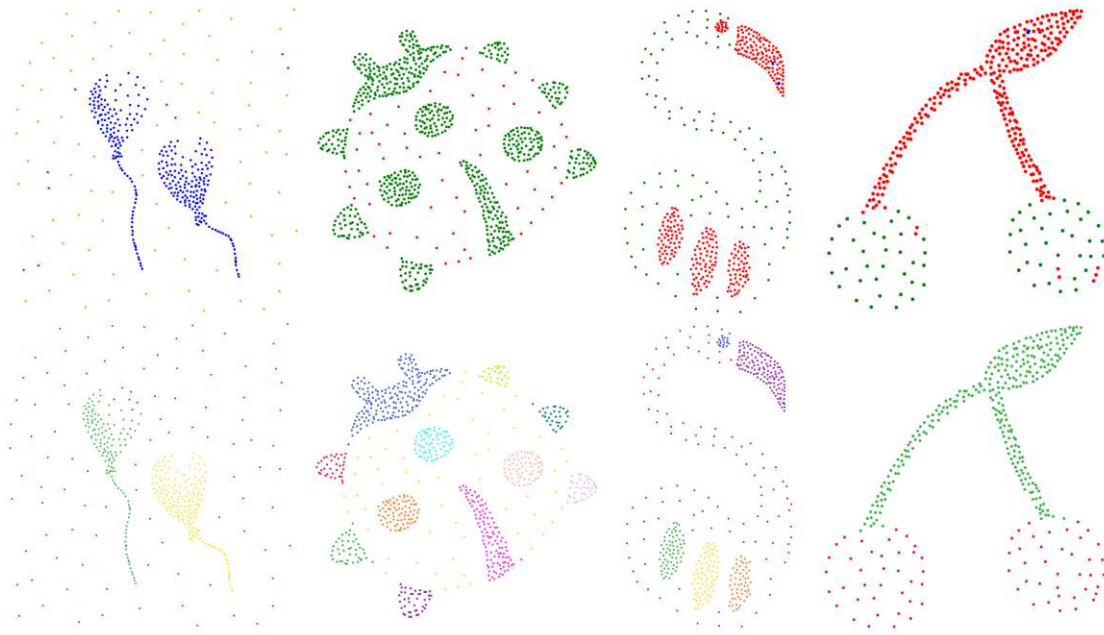


Figure 6.10: Our clustering results (top) compared to DBSCAN (bottom). Our method connects similar densities in a single connected component due to the density-based proximity definition. The independent structures within the same density clusters can further be decomposed into distance-only connected components using a post-processing step.

**Applications** In addition to introducing our proximity criterion, we provide a category-based representation of its applications. Our analysis and validation of this method span various applications, including clustering and data segmentation, reconstruction (with a specific focus on 2D), stipple art coloring or editing, and network topology analysis, while briefly discussing potential advantages for anisotropic clustering, which falls outside this chapter's scope.

### 6.5.1 Clustering and Data Segmentation

Clustering based on the SING-connected components integrates local density considerations into the  $\epsilon$ -neighborhood graph and, consequently, into the Rips complexes, providing a significant generalization, as discussed for the example of Fig. 6.3. Moreover, throughout our experimentation, we explored the most relevant clustering methods in terms of local density consideration, including  $k$ -means, density-based spatial clustering of applications with noise (DBSCAN), as well as the clustering induced by  $\epsilon$ -neighborhood graphs. Among these, DBSCAN yielded the most optimal grouping outcomes in general, and we compare our results to their clusters in Figure 6.10. We analyze the evolution of our clusters compared to Single Linkage [GR69] in Figure 6.11, showing how our proximity encoding captures more information about the input.



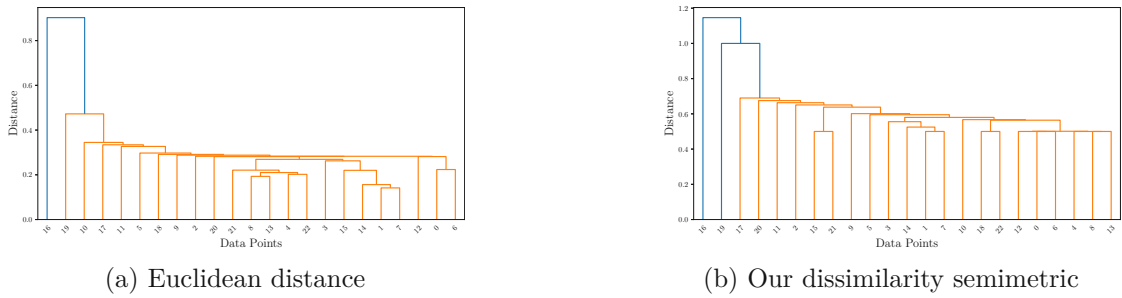


Figure 6.11: Dendrograms representing single linkage clustering of the dataset presented in Figure 6.3. The single linkage connects components with minimal distance, and the dendrogram illustrates the merging pattern. We observe that by using our semimetric, an additional large cluster (in blue) is formed. Thus, our semimetric captures cluster formations that would be missed by the Euclidean one.

### 6.5.2 Multi-Class Disk Distribution Analysis

Distributions can also be represented through disk distribution, where the radius of the data can encode additional information. For example, the ecosystem examples presented in [ENMGC19] encode the size of the natural elements such as vegetation types. Such types of distributions are commonly used in artificial ecosystem generation in tools such as Ecobrush [GLCC17]. For such disk distributions, the class of each disk has to be known in advance to be able to extract intra- and inter-class relationships. In Figure 6.1, we are able to extract the classes (as individual clusters) given only the input coordinates and radii. We compute the persistence barcode of our data by varying the  $\varepsilon$  parameter (as in Figure 6.2). We then use the stable intervals in the diagram to guide the parameter choice in the direction of the most meaningful clusters (Figure 6.1, right). We could further group all large disks in the same class using the topology of our SING graph (in this case, by observing they all have the same type of connections), considering filters on the disk distance if needed (in this case, the distance values reflect the fact that all of their neighbors are placed inside the disk).

### 6.5.3 Stipple Art Manipulation

Stipples are patterns of points where the visual information is encoded through density and correlation. Clustering stippling patterns into visually meaningful regions is challenging and does not necessarily align with our visual perception. Despite this lack of ground truth, SING clustering provides promising results, as showcased by Fig. 6.12. For challenging density-varying stipples, we use the SING variation that accounts for the density parameter. Some examples are shown in Figure 6.10, where our results for layer extraction are similar to DBSCAN [EKSX96]. However, selecting parameters for DBSCAN relies only on data properties, lacking an easily inferred optimal parameter value, in contrast to our method. Note that connecting samples by local density may form small, packed clusters, as seen in the cherry example. This can be adjusted via



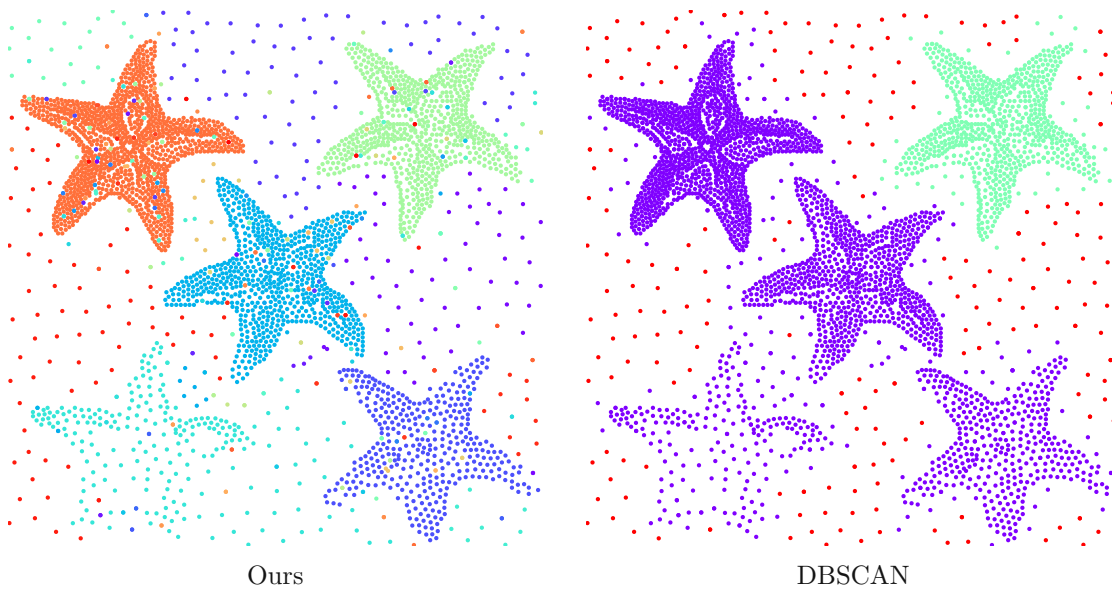


Figure 6.12: A point pattern synthesized from an image, using the method in [HRS<sup>+</sup>23], is segmented using the SING and DBSCAN clustering.

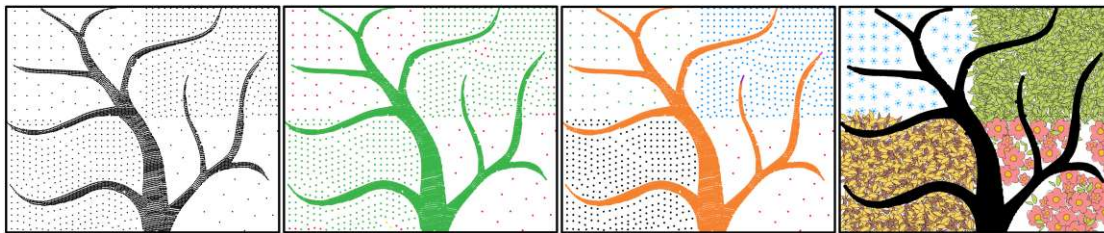


Figure 6.13: Left to Right: Input stipples, Result of DBSCAN [EKSX96], Our clustering result, Result after replacing our clusters with stamps.

the  $\epsilon$  parameter or interpreted as a feature in post-processing, given the absence of a ground-truth segmentation in stipple art. Moreover, in Figure 6.13, the slight variation in density is not captured by DBSCAN. Having such a layered representation allows for easy and meaningful manipulation of the art, like editing the distribution [HRS<sup>+</sup>23] and representation (see Figure 6.14). Automatically generated stipple patterns that exhibit varying density require more parameter tuning, as the difference between distinct portions of the input image is not sharp, and changing density across an area is used to illustrate various visual effects – Figure 6.15.

#### 6.5.4 2D Reconstruction

Shape reconstruction is the process of identifying the shape induced by a set of points [TPM20, TPM21, MPM15]. This is a well-known ill-posed problem in Computational Geometry, with applications in GIS and WSN. Thanks to SING, as shown in Figure 6.16, we can

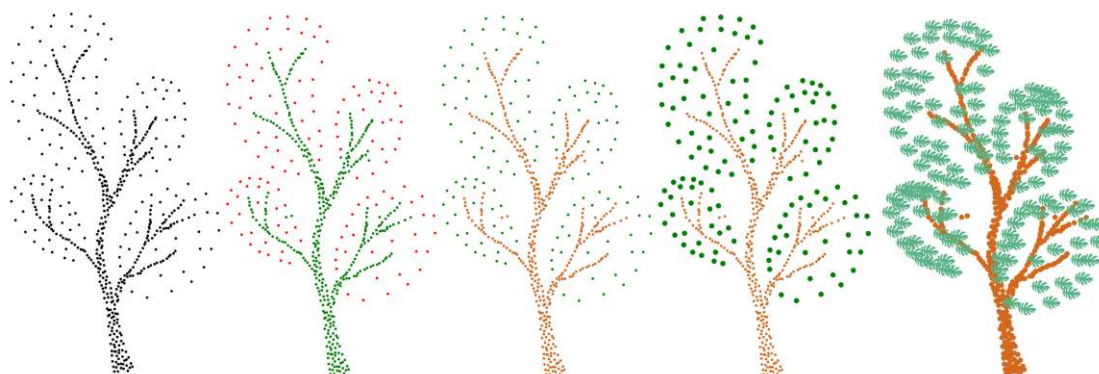
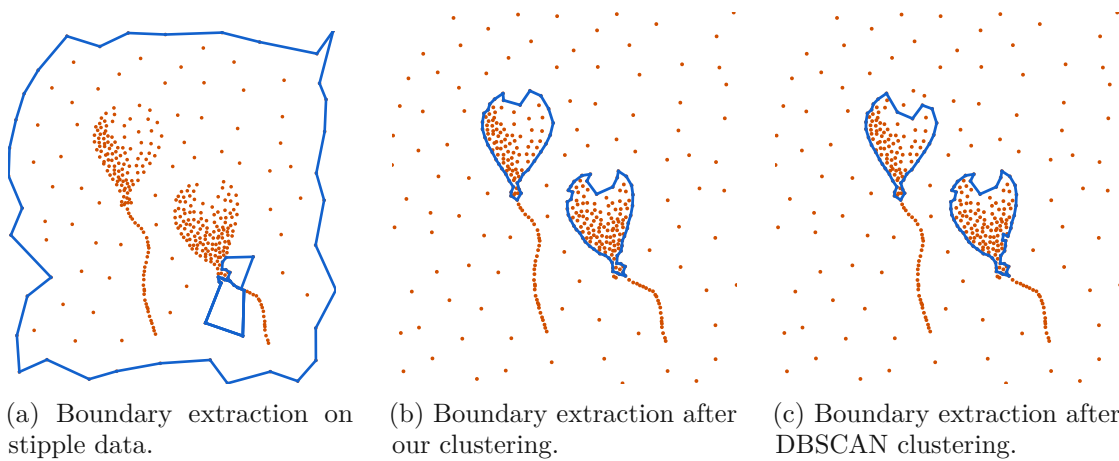


Figure 6.14: Left to Right: Input stipples, clustering result, the result after varying color, the result after editing stipple size, replacing stipple with pattern.



Figure 6.15: Despite variations in density, our method (right) successfully extracts the bird's body as a single component, unlike DBSCAN (left), which separates it based on density fluctuations. Additionally, our method effectively distinguishes different parts of the feet into meaningful clusters. However, certain dense regions pose challenges for our method to cluster in a meaningful semantic way. Input point pattern from [DGBOD12].



(a) Boundary extraction on stipple data. (b) Boundary extraction after our clustering. (c) Boundary extraction after DBSCAN clustering.

Figure 6.16: Boundary extraction of stipple art using Discern [TPM21]. All boundary methods we have tested fail on inputs with multiple densities. However, running them on clustered input results in meaningful boundaries.

easily group points in a meaningful manner and then extract their boundary to determine the shape. Additionally, with our SING variant, we can further extend the traditional shape reconstruction problem by considering density variations and approaching a level of efficiency closer to human perception in this context. Extracting the boundary directly from our graph is promising as well and worth future investigation.

### 6.5.5 Network Topology Analysis

Network graph analysis and classification involves understanding the overall structure of the graph and making predictions based on that structure [KGB19]. In our geometric context, we experiment on spatial networks, which have a clear positional embedding. Leveraging our persistence-based proximity criterion, which effectively captures topological characteristics across different scales, enables us to gain a comprehensive understanding of the graph’s shape and connectivity. As a result, our approach encodes, under the same edge budget, a more meaningful simplification of original data compared to the Rips complex, capturing the original shape of the road network better - Figure 6.17.

## 6.6 Discussion and Perspectives

**Limitations** Figure 6.15 shows various clustering imperfections produced by our method. This is due to variations in the density information and the fact that the current version of our method does not incorporate explicit part labels or “semantic” knowledge. It also motivates a semantic extension of our work in order to disambiguate such cases. Additionally, while the selection of a suitable value for  $\varepsilon$  is largely guided by TDA, a comparable approach for determining the density parameter is currently

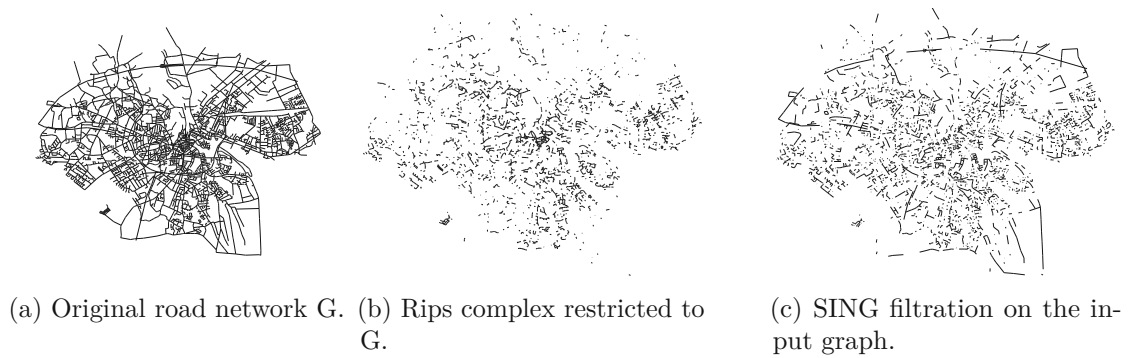


Figure 6.17: Network simplification under a fixed edge budget, utilizing the Oldenburg road network [MXA04]. Our results enable improved overall connectivity in the generated network graph (decreasing the connected components count – 1751 compared to 1839).

lacking in our current implementation. For further efficiency improvement, optimizing our bottleneck distance computation could also lead to better runtime.

**Future work** Subsampling and point set simplification both seem to be natural contexts in which our proximity criterion can be formalized and employed. Surface curvature and anisotropic metrics incorporation were already mentioned as promising inspirations for future work. Another theoretical future direction would be to investigate whether the SING is a spanner and to characterize the corresponding stretch factor. This is based on the fact that for  $\varepsilon$  tending to infinity, SING approaches the complete graph – a 1-spanner from a threshold  $\varepsilon$  value. Since our similarity metric extends straightforwardly to higher dimensions with low computational cost, it would also allow for the analysis of group behavior in animal swarms based on object detection input, for example. Furthermore, the improved connectivity can assist in creating richer features in images or point cloud data, such as for photogrammetry or for training networks on point data. This extension to higher dimensions and related application scenarios are left for future work due to our current unoptimized implementation and its being beyond this chapter’s scope.

# Conclusion

We have presented various methods to reconstruct or extract important information from point clouds. We used the spheres-of-influence graph and its extensions to explore various goals related to point cloud reconstruction through the lens of proximity.

## 7.1 Summary

In Chapter 3, we proposed a new proximity graph computed by intersecting the spheres-of-influence graph (SIG) with the Delaunay triangulation (DT). We showed that the resulting graph, termed SIGDT, encompasses the piecewise linear reconstruction for a set of unstructured points in the plane, under a sampling condition that surpasses existing bounds and effectively represents the characteristics of point sets. As an application, we implement post-processing steps, adapted from the Connect2D algorithm, to remove redundant edges. Our findings indicate that the proposed algorithm, SIG-Connect2D, delivers superior reconstruction accuracy when compared to state-of-the-art algorithms from a curve reconstruction benchmark.

We expand the curve reconstruction process to Riemannian manifolds in Chapter 4, where we relax the existing state-of-the-art sampling requirements and introduce a novel algorithm capable of reconstructing closed curves directly on surfaces from a sparse set of sample points. We adapt and extend the SIG-Connect2D planar reconstruction method to operate within a new domain, addressing the challenges of non-Euclidean spaces. The robustness of our approach is demonstrated through successful reconstructions of multiple curves on various surfaces. Additionally, we explore new potential applications of our method, enabling automated curve reconstruction on Riemannian manifolds.

We tackle the problem of connectivity encoding for point clouds in Chapter 5, by using SIG as a parameter-free alternative to the commonly used  $k$ NN graph, serving as an initial step towards surface reconstruction and a strong base for normal estimation. We



demonstrate that the neighboring edges in the SIG are spatially bounded, facilitating fast computation. Our approach provides a more accurate representation of the ground truth connectivity compared to the  $k$ NN graphs across a wide range of  $k$  values, while also being parameter-free.

Chapter 6 explores the extension of the SIG into SING (Stability-Incorporated Neighborhood Graph), which leverages thorough topological data analysis to capture the intrinsic geometric properties of a point set. By building on the spheres-of-influence graph and incorporating additional features, we offer greater flexibility and control in encoding proximity information and accounting for local density variations. Through persistence analysis of our proximity graph, we introduce a new clustering technique and explore additional variants with enhanced proximity criteria. Our detailed analysis and comparisons across various datasets show that the proposed method effectively identifies meaningful clusters, even in the presence of density and correlation variations. The evaluated application scenarios highlight the advantages of our graph over traditional neighborhood graphs, particularly in terms of minimizing the need for parameter tuning.

## 7.2 Future directions

Throughout our exploration of proximity-based methods for point cloud reconstruction, we discovered multiple directions that could be further explored, related to our contributions, or to concepts encountered during the research.

**Surface reconstruction** A natural direction that needs to be explored in depth is surface reconstruction. Firstly, our work on distributed surface reconstruction [MKOW24] would benefit from a more thorough evaluation for a stronger contribution. Considering the recent work extending Poisson Reconstruction to work in a distributed fashion [KH23], we can better place our result in the literature. Secondly, another avenue of development in the surface reconstruction direction is using SIG directly for an explicit surface reconstruction method, similar to extending SIG-Connect2D, our curve reconstruction algorithm, to surfaces. However, recent works in the wide domain of surface reconstruction mainly target implicit methods, especially since they can easily be adapted to work as part of learning approaches, and that is an important factor that should be considered as well.

**Machine learning-based methods** We aim to investigate the usage of the SIG or other proximity graphs (different from the commonly used  $k$ NN graphs) in graph neural networks with the aim of feature detection, noise removal, or surface reconstruction. Exploring how to modify existent architectures to work with the degree-varying SIG could lead to promising results, since we have shown that SIG encodes proximity better than  $k$ NN. However, it might encode too little information for meaningful learning results, which could be mitigated by using  $n$ SIG - using the distance to the  $n^{th}$  nearest neighbor instead of the closest one as the sphere of influence of each point.

**Outlier and noise detection** We started investigating outlier detection in our under review work [MIOU24], where we statistically analyze the length of SIG edges to detect outliers, which are usually isolated compared to inliers. Using similar tools to detect and remove noisy samples could represent an interesting avenue for future development. However, since the graph is still purely distance-based, a noise removal method would have to be more complex and include more information about the data to be able to deal with varying amounts and distributions of noise.

**Curves on surfaces dataset** While working on our Riemannian manifold curve reconstruction method, we observed the lack of an available dataset in this direction. Even though curves can naturally exist on surfaces, as decorations or boundaries of various sections of the mesh, there is no such dataset that could help in creating a general and available benchmark for curve reconstruction or related tasks. We aim to investigate this direction, at least as a qualitative manner to evaluate the reconstruction methods in the field.

**Theoretical developments** Developing the SING, which we have described in Chapter 6, as a stability-incorporated neighborhood graph, and analyzing its properties as TDA tool, we observed this graph becomes a spanner of the point set when  $\varepsilon$  approaches infinity (*i.e.* there is a path between every pair of points, of length equal to a multiplier of the Euclidean distance between the ends; the stretch factor is 1 as  $\varepsilon$  approaches infinity, as all points are connected). We would be interested in the threshold where SING becomes a spanner, or when each independent cluster becomes a spanner and what is the stretch factor of these structures.

**General outlook** After investigating the role and effects of proximity in point cloud reconstruction, I am able to observe the attraction of simplicity - basic tools, such as new distance formulations or little-explored graphs can contain a varied collection of useful properties. They are often able to encode complex information and can be used as strong building blocks in more complex reconstruction methods. I hope this trend will only continue and we will observe more simple, seemingly obvious, but powerful ideas.





# Overview of Generative AI Tools Used

Generative AI tools were only used as an aid for grammar mistakes, possible rephrasing, and the translation of the Abstract, as a non-German speaker.



# List of Figures

1.1	The problem of curve reconstruction. . . . .	2
2.1	Point cloud examples. . . . .	7
2.2	Various curve types. . . . .	8
2.3	Difficulty of reconstruction given a sparse input. . . . .	9
2.4	Distance-based sampling. . . . .	10
2.5	Local feature size dependent sampling. . . . .	11
2.6	Medial axis of sharp angles. . . . .	12
2.7	$k$ NN examples. . . . .	13
2.8	Voronoi-Delaunay duality. . . . .	13
2.9	Gabriel graph creation. . . . .	14
2.10	Other subsets of the Delaunay triangulation. . . . .	15
2.11	Sphere-of-influence graph (SIG) creation. . . . .	15
2.12	Distance-based complexes. . . . .	17
2.13	Surface normals and their computation. . . . .	18
2.14	Planar reconstruction. . . . .	20
2.15	Curves on Riemannian manifolds. . . . .	22
2.16	Various challenges encountered in surface reconstruction. . . . .	24
3.1	Boundary inclusion in SIGDT. . . . .	28
3.2	Difference between original boundary approximation and ours. . . . .	30
3.3	Overview of SIG-CONNECT2D algorithm. . . . .	31
3.4	Proof of boundary inclusion in SIGDT. . . . .	33
3.5	Inflating the curve. . . . .	35
3.6	Sculpting the curve. . . . .	35
3.7	Reconstruction of manifold curves. . . . .	37
3.8	Reconstruction of curves with sharp corners. . . . .	39
3.9	Symmetric area difference for open curves. . . . .	40
3.10	Symmetric difference of area for multiple curves. . . . .	40
3.11	Reconstruction accuracy for graph-based $\epsilon < 1$ . . . . .	42
3.12	Average runtime of manifold curve reconstruction. . . . .	43
3.13	RMSE of reconstructions for varying $\epsilon$ -sampling. . . . .	43
3.14	Uniform noise reconstruction. . . . .	44
3.15	Uniform noise reconstruction RMSE. . . . .	44

3.16	RMSE of curves from inputs with lfs-based noise. . . . .	45
3.17	Reconstruction of curves with varying outlier share. . . . .	45
3.18	Symmetric area difference for non-manifold curves. . . . .	46
3.19	Reconstruction of different types of curves. . . . .	47
3.20	Results for out-of-scope types of curves. . . . .	48
3.21	Failure cases for SIG-CONNECT2D. . . . .	49
4.1	Reconstruction of multiple curves on the fertility mesh. . . . .	52
4.2	Cut locus examples. . . . .	55
4.3	Non-planar medial axes. . . . .	56
4.4	Sampling condition comparison. . . . .	57
4.5	Injectivity radius. . . . .	59
4.6	Medial axis behavior in 2D. . . . .	60
4.7	SIGDT graph creation. . . . .	61
4.8	SIGDV graph creation. . . . .	62
4.9	Hamiltonian path on SIGDV as reconstruction. . . . .	64
4.10	Motion tracking reconstruction. . . . .	68
4.11	Curve reconstruction on surfaces. . . . .	69
4.12	Contour matching improvements. . . . .	70
4.13	Sparse data visualization. . . . .	72
5.1	SIG versus $k$ NN graphs - Buddha statue. . . . .	77
5.2	Spheres of incidence - planar example. . . . .	79
5.3	SIG versus $k$ NN graphs - Stanford bunny. . . . .	81
5.4	Triangulation-independent connectivity. . . . .	82
5.5	DeltaCon similarity metric. . . . .	82
5.6	Number of edges comparison. . . . .	83
5.7	Geodesic ratio difference. . . . .	84
5.8	Adverse sparse sampling of close sheets. . . . .	85
5.9	Angle deviation of normals. . . . .	85
5.10	Sparse sampling limitations for normal computation. . . . .	86
5.11	Normal computation results. . . . .	86
5.12	Results for different sampling densities. . . . .	87
5.13	Timings. . . . .	88
5.14	Parallel layers in sampling. . . . .	89
5.15	Allowing for noisy input. . . . .	89
6.1	SING results on disk data. . . . .	92
6.2	Persistence Barcode. . . . .	95
6.3	Connected components of SING compared to the Rips complex. . . . .	97
6.4	Barcodes of Rips and SING filtrations. . . . .	97
6.5	Proximity graphs compared to SING. . . . .	99
6.6	Structured pattern data connected using various proximity graphs. . . . .	99
6.7	The importance of the density parameter. . . . .	101

6.8	Confidence intervals and examples of noisy results. . . . .	102
6.9	Noisy pattern data proximity graph results. . . . .	103
6.10	Clustering results compared to DBSCAN. . . . .	107
6.11	Single linkage dendrogram results. . . . .	108
6.12	Density-varying results on real-life data. . . . .	109
6.13	Comparison with DBSCAN on density-varying stipples. . . . .	109
6.14	Editing clusters using our method. . . . .	110
6.15	Results on density-varying stipples. . . . .	110
6.16	Boundary extraction of stipple art. . . . .	111
6.17	Network simplification under a fixed edge budget. . . . .	112



# List of Tables

3.1	Correctness comparison to state-of-the-art. . . . .	38
3.2	Exact reconstruction results. . . . .	41





# List of Algorithms

3.1	SIG-Connect2D . . . . .	32
4.1	SIGDV-biased TSP solver. . . . .	65
5.1	Fast SIG computation. . . . .	80
6.1	Density SING computation. . . . .	101



# Bibliography

- [AB99] Nina Amenta and Marshall Bern. Surface Reconstruction by Voronoi Filtering. *Discr. & Comp. Geom.*, 22:481–504, 12 1999. doi:10.1007/PL00009475.
- [ABE98] Nina Amenta, Marshall Bern, and David Eppstein. The Crust and the beta-Skeleton: Combinatorial Curve Reconstruction. *Graphical Models and Image Processing*, 60:125–135, 01 1998.
- [ABK98] Nina Amenta, Marshall Bern, and Manolis Kamvysselis. A New Voronoi-Based Surface Reconstruction Algorithm. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, page 415–421, New York, NY, USA, 1998. Association for Computing Machinery. doi:10.1145/280814.280947.
- [AC20] Syeda Mariam Ahmed and Chee Meng Chew. Density-based clustering for 3d object detection in point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10608–10617, 2020.
- [ACDL00] Nina Amenta, Sunghee Choi, Tamal K Dey, and Naveen Leekha. A simple algorithm for homeomorphic surface reconstruction. In *Proceedings of the sixteenth annual symposium on Computational geometry*, pages 213–222, 2000.
- [ACK01] Nina Amenta, Sunghee Choi, and Ravi Krishna Kolluri. The Power Crust. In *Proceedings of the Sixth ACM Symposium on Solid Modeling and Applications*, SMA '01, page 249–266, New York, NY, USA, 2001. Association for Computing Machinery. doi:10.1145/376957.376986.
- [Ado] Adobe Inc. Adobe illustrator. Accessed: 2024-08-31. URL: <https://adobe.com/products/illustrator>.
- [AKL13] Franz Aurenhammer, Rolf Klein, and Der-Tsai Lee. *Voronoi Diagrams and Delaunay Triangulations*. WORLD SCIENTIFIC, 2013. doi:10.1142/8685.

- [AM00] Ernst Althaus and Kurt Mehlhorn. TSP-based curve reconstruction in polynomial time. In *Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, pages 686–695, 2000.
- [ASI20] Mohiuddin Ahmed, Raihan Seraj, and Syed Mohammed Shamsul Islam. The k-means algorithm: A comprehensive survey and performance evaluation. *Electronics*, 9(8):1295, 2020.
- [BDH96] C Bradford Barber, David P Dobkin, and Hannu Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)*, 22(4):469–483, 1996.
- [BHZ23] Ulrich A Brodowsky, Stefan Hougardy, and Xianghui Zhong. The approximation ratio of the k-opt heuristic for the euclidean traveling salesman problem. *SIAM Journal on Computing*, 52(4):841–864, 2023.
- [Bjo14] Andreas Bjorklund. Determinant sums for undirected hamiltonicity. *SIAM Journal on Computing*, 43(1):280–299, 2014.
- [Blu67] Harry Blum. A transformation for extracting new descriptions of shape. *Models for the perception of speech and visual form*, pages 362–380, 1967.
- [BM16] Alexandre Boulch and Renaud Marlet. Deep Learning for Robust Normal Estimation in Unstructured Point Clouds. *Computer Graphics Forum*, 35(5):281–290, 2016. doi:10.1111/cgf.12983.
- [BM22] Alexandre Boulch and Renaud Marlet. POCO: Point Convolution for Surface Reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6302–6314, June 2022.
- [BMD<sup>+</sup>05] Arindam Banerjee, Srujana Merugu, Inderjit S Dhillon, Joydeep Ghosh, and John Lafferty. Clustering with Bregman divergences. *Journal of machine learning research*, 6(10), 2005.
- [Bow81] A. Bowyer. Computing Dirichlet tessellations\*. *The Computer Journal*, 24(2):162–166, 01 1981. doi:10.1093/comjnl/24.2.162.
- [BRLB14] Federica Bogo, Javier Romero, Matthew Loper, and Michael J Black. FAUST: Dataset and evaluation for 3D mesh registration. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3794–3801, 2014.
- [BTS<sup>+</sup>17] Matthew Berger, Andrea Tagliasacchi, Lee M. Seversky, Pierre Alliez, Gaël Guennebaud, Joshua A. Levine, Andrei Sharf, and Claudio T. Silva. A Survey of Surface Reconstruction from Point Clouds. *Computer Graphics Forum*, 36(1):301–329, 2017. doi:10.1111/cgf.12802.

- [CBC<sup>+</sup>01] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction and Representation of 3D Objects with Radial Basis Functions. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, page 67–76, New York, NY, USA, 2001. Association for Computing Machinery. doi:10.1145/383259.383266.
- [CDGDS13] Keenan Crane, Fernando De Goes, Mathieu Desbrun, and Peter Schröder. Digital geometry processing with discrete exterior calculus. In *ACM SIGGRAPH 2013 Courses*, pages 1–126, 2013.
- [CDSO14] Frédéric Chazal, Vin De Silva, and Steve Oudot. Persistence stability for geometric complexes. *Geometriae Dedicata*, 173(1):193–214, 2014.
- [CLRS22] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, 2022.
- [CLX<sup>+</sup>19] Chao Chen, Guanbin Li, Ruijia Xu, Tianshui Chen, Meng Wang, and Liang Lin. Clusternet: Deep hierarchical cluster network with rigorously rotation-invariant representation for point cloud analysis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4994–5002, 2019.
- [CM02] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002.
- [Cro58] Georges A Croes. A method for solving traveling-salesman problems. *Operations research*, 6(6):791–812, 1958.
- [CSEH07] David Cohen-Steiner, Herbert Edelsbrunner, and John Harer. Stability of Persistence Diagrams. *Discrete Comput. Geom.*, 37(1):103–120, January 2007. doi:10.1007/s00454-006-1276-5.
- [CSKG17] R. Qi Charles, Hao Su, Mo Kaichun, and Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *2017 IEEE CVPR*, pages 77–85, 2017. doi:10.1109/CVPR.2017.16.
- [CSLV20] David Cohen-Steiner, André Lieutier, and J. Vuillamy. Lexicographic Optimal Homologous Chains and Applications to Point Cloud Triangulations. In *SoCG*, 2020.
- [CWW17] Keenan Crane, Clarisse Weischedel, and Max Wardetzky. The Heat Method for Distance Computation. *Commun. ACM*, 60(11):90–99, October 2017. doi:10.1145/3131280.
- [DC16] Manfredo P Do Carmo. *Differential geometry of curves and surfaces: revised and updated second edition*. Courier Dover Publications, 2016.

- [DdF17] Julie Digne and Carlo de Franchis. The Bilateral Filter for Point Clouds. *Image Processing On Line*, 7:278–287, 10 2017. doi:10.5201/ipol.2017.179.
- [DG06] Tamal K. Dey and Samrat Goswami. Provable surface reconstruction from noisy samples. *Computational Geometry*, 35(1):124–141, 2006. Special Issue on the 20th ACM Symposium on Computational Geometry. doi:10.1016/j.comgeo.2005.10.006.
- [DGBOD12] Fernando De Goes, Katherine Breeden, Victor Ostromoukhov, and Mathieu Desbrun. Blue noise through optimal transport. *ACM Transactions on Graphics (TOG)*, 31(6):1–11, 2012.
- [DK99] Tamal K. Dey and Piyush Kumar. A Simple Provable Algorithm for Curve Reconstruction. In *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '99, page 893–894, USA, 1999. Society for Industrial and Applied Mathematics.
- [DMC96] Marco Dorigo, Vittorio Maniezzo, and Alberto Coloni. Ant system: optimization by a colony of cooperating agents. *IEEE transactions on systems, man, and cybernetics, part b (cybernetics)*, 26(1):29–41, 1996.
- [dP22] Luiz Henrique de Figueiredo and Afonso Paiva. Region reconstruction with the sphere-of-influence diagram. *Computers & Graphics*, 107:252–263, 2022. doi:10.1016/j.cag.2022.08.002.
- [DW00] Tamal K. Dey and Rephael Wenger. Reconstruction Curves with Sharp Corners. In *Proceedings of the Sixteenth Annual Symposium on Computational Geometry*, SCG '00, page 233–241, New York, NY, USA, 2000. Association for Computing Machinery.
- [DW02] Tamal Dey and Rephael Wenger. Fast Reconstruction of Curves with Sharp Corners. *Int. J. Comput. Geometry Appl.*, 12:353–400, 10 2002. doi:10.1142/S0218195902000931.
- [Dwy95] Rex A. Dwyer. The expected size of the sphere-of-influence graph. *Comp. Geometry*, 5(3):155–164, 1995. doi:10.1016/0925-7721(94)00025-Q.
- [DZ04] Tamal K Dey and Wulue Zhao. Approximating the medial axis from the voronoi diagram with a convergence guarantee. *Algorithmica*, 38:179–200, 2004.
- [DZM07] Ramsay Dyer, Hao Zhang, and Torsten Möller. Voronoi-Delaunay duality and Delaunay meshes. In *Proceedings of the 2007 ACM symposium on Solid and physical modeling*, pages 415–420, 2007.

- [Ede83] Edelsbrunner, H. and Kirkpatrick, D. and Seidel, R. On the shape of a set of points in the plane. *IEEE Transactions on Information Theory*, 29(4):551–559, 1983. doi:10.1109/TIT.1983.1056714.
- [EGO<sup>+</sup>20] Philipp Erler, Paul Guerrero, Stefan Ohrhallinger, Niloy J Mitra, and Michael Wimmer. Points2surf learning implicit surfaces from point clouds. In *Computer Vision–ECCV 2020, Proceedings, Part V*, pages 108–124. Springer, 2020.
- [EH10] Herbert Edelsbrunner and John L Harer. *Computational topology: an introduction*. American Mathematical Society, 2010.
- [EIO<sup>+</sup>22] Absalom E. Ezugwu, Abiodun M. Ikotun, Olaide O. Oyelade, Laith Abualigah, Jeffery O. Agushaka, Christopher I. Eke, and Andronicus A. Akinyelu. A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects. *Eng. Appl. Artif. Intell.*, 110(C), apr 2022. doi:10.1016/j.engappai.2022.104743.
- [EKSX96] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. Density-based spatial clustering of applications with noise. In *Int. Conf. knowledge discovery and data mining*, volume 240, 1996.
- [EMP<sup>+</sup>03] David S Ebert, F Kenton Musgrave, Darwyn Peachey, Ken Perlin, and Steven Worley. *Texturing & modeling: a procedural approach*. Morgan Kaufmann, 2003.
- [EMWR24] Henry Ehlers, Diana Marin, Hsiang-Yun Wu, and Renata G Raidou. Visualizing Group Structure in Compound Graphs: The Current State, Lessons Learned, and Outstanding Opportunities. In *VISIGRAPP (1): GRAPP, HUCAPP, IVAPP*, pages 697–708, 2024.
- [ENMGC19] Pierre Ecomier-Nocca, Pooran Memari, James Gain, and Marie-Paule Cani. Accurate Synthesis of Multi-Class Disk Distributions. *Computer Graphics Forum*, 38(2):157–168, 2019. doi:10.1111/cgf.13627.
- [FD07] Brendan J Frey and Delbert Dueck. Clustering by passing messages between data points. *science*, 315(5814):972–976, 2007.
- [Fed59] Herbert Federer. Curvature measures. *Transactions of the American Mathematical Society*, 93(3):418–491, 1959.
- [fMRWFE] European Centre for Medium-Range Weather Forecasts (ECMWF). European Centre for Medium-Range Weather Forecasts (ECMWF). Accessed: 2024-08-31. URL: <http://www.ecmwf.int/>.



- [FR02] Chris Fraley and Adrian E Raftery. Model-based clustering, discriminant analysis, and density estimation. *Journal of the American statistical Association*, 97(458):611–631, 2002.
- [GCSAD11] Fernando Goes, David Cohen-Steiner, Pierre Alliez, and Mathieu Desbrun. An Optimal Transport Approach to Robust Reconstruction and Simplification of 2D Shapes. *Comp. Graph. Forum*, 30:1593–1602, 08 2011. doi:10.1111/j.1467-8659.2011.02033.x.
- [GG07] Gaël Guennebaud and Markus Gross. Algebraic Point Set Surfaces. *ACM Trans. Graph.*, 26(3):23–es, jul 2007. doi:10.1145/1276377.1276406.
- [GJ<sup>+</sup>10] Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. 2010.
- [GKOM18] Paul Guerrero, Yanir Kleiman, Maks Ovsjanikov, and Niloy J. Mitra. PCP-Net Learning Local Shape Properties from Raw Point Clouds. *Computer Graphics Forum*, 37(2):75–85, 2018. doi:10.1111/cgf.13343.
- [GLCC17] J. Gain, H. Long, G. Cordonnier, and M.-P. Cani. EcoBrush: Interactive Control of Visually Consistent Large-Scale Ecosystems. *Computer Graphics Forum*, 36(2):63–73, 2017. doi:10.1111/cgf.13107.
- [GLR95] M.J. González-López and Tomas Recio. *Voronoi computability in SE(3)*, pages 137–148. 12 1995. doi:10.1515/9783110881271.137.
- [GR69] John C Gower and Gavin JS Ross. Minimum spanning trees and single linkage cluster analysis. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 18(1):54–64, 1969.
- [GS69] K. Ruben Gabriel and Robert R. Sokal. A New Statistical Approach to Geographic Variation Analysis. *Systematic Biology*, 18(3):259–278, 09 1969. doi:10.2307/2412323.
- [GTSK13] Ayelet Gilboa, Ayellet Tal, Ilan Shimshoni, and Michael Kolomenkin. Computer-based, automatic recording and illustration of complex archaeological artifacts. *Journal of Archaeological Science*, 40(2):1329–1339, 2013.
- [Har01] John C Hart. Perlin noise pixel shaders. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS workshop on Graphics hardware*, pages 87–94, 2001.
- [HDD<sup>+</sup>92] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. In *Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '92, page 71–78, New York, NY, USA, 1992. Association for Computing Machinery. doi:10.1145/133994.134011.

- [HH13] Brian C Hall and Brian C Hall. *Lie groups, Lie algebras, and representations*. Springer, 2013.
- [HMGCO20] Rana Hanocka, Gal Metzer, Raja Giryes, and Daniel Cohen-Or. Point2Mesh: A Self-Prior for Deformable Meshes. *ACM Trans. Graph.*, 39(4), aug 2020. doi:10.1145/3386569.3392415.
- [HRS<sup>+</sup>23] Xingchang Huang, Tobias Ritschel, Hans-Peter Seidel, Pooran Memari, and Gurprit Singh. Patternshop: Editing Point Patterns by Image Manipulation. *ACM Transactions on Graphics (TOG)*, 42(4):1–14, 2023.
- [HWW<sup>+</sup>22] Fei Hou, Chiyu Wang, Wencheng Wang, Hong Qin, Chen Qian, and Ying He. Iterative poisson surface reconstruction (iPSR) for unoriented points. *ACM Trans. Graph.*, 41(4), jul 2022. doi:10.1145/3528223.3530096.
- [HWW<sup>+</sup>24] ZhangJin Huang, Yuxin Wen, ZiHao Wang, Jinjuan Ren, and Kui Jia. Surface Reconstruction from Point Clouds: A Survey and a Benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–20, 2024. doi:10.1109/TPAMI.2024.3429209.
- [IN07] Kazuo Iwama and Takuya Nakashima. An improved exact algorithm for cubic graph TSP. In *Computing and Combinatorics: 13th Annual International Conference, COCOON 2007, Banff, Canada, July 16-19, 2007. Proceedings 13*, pages 108–117. Springer, 2007.
- [Ink] Inkscape Project. Inkscape. Accessed: 2024-08-31. URL: <https://inkscape.org>.
- [JM97] David S Johnson and Lyle A McGeoch. The traveling salesman problem: A case study in local optimization. *Local search in combinatorial optimization*, 1(1):215–310, 1997.
- [JT92] Jerzy W. Jaromczyk and Godfried T. Toussaint. Relative neighborhood graphs and their relatives. *Proc. IEEE*, 80:1502–1517, 1992.
- [Kar75] Richard M Karp. On the computational complexity of combinatorial problems. *Networks*, 5(1):45–68, 1975.
- [Kar77] Hermann Karcher. Riemannian center of mass and mollifier smoothing. *Communications on pure and applied mathematics*, 30(5):509–541, 1977.
- [KBH06] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson Surface Reconstruction. In Alla Sheffer and Konrad Polthier, editors, *Symposium on Geometry Processing*. The Eurographics Association, 2006. doi:10.2312/SGP/SGP06/061-070.

- [KGB19] Alexander P. Kartun-Giles and Ginestra Bianconi. Beyond the clustering coefficient: A topological analysis of node neighbourhoods in complex networks. *Chaos, Solitons & Fractals: X*, 1:100004, 2019. doi:10.1016/j.csfx.2019.100004.
- [KH13] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Trans. Graph.*, 32(3), jul 2013. doi:10.1145/2487228.2487237.
- [KH23] M. Kazhdan and H. Hoppe. Distributed Poisson Surface Reconstruction. *Computer Graphics Forum*, 42(6):e14925, 2023. doi:10.1111/cgf.14925.
- [Kli95] Wilhelm Klingenberg. *Riemannian geometry*, volume 1. Walter de Gruyter, 1995.
- [KR85] David G. Kirkpatrick and John D. Radke. A Framework for Computational Morphology. *Machine Intelligence and Pattern Recognition*, 2:217–248, 1985.
- [KST08] Michael Kolomenkin, Ilan Shimshoni, and Ayellet Tal. Demarcating curves for shape illustration. In *ACM SIGGRAPH Asia 2008 papers*, pages 1–9. 2008.
- [KSV<sup>+</sup>16] Danai Koutra, Neil Shah, Joshua T. Vogelstein, Brian Gallagher, and Christos Faloutsos. DeltaCon: Principled Massive-Graph Similarity Function with Attribution. *ACM Trans. Knowl. Discov. Data*, 10(3), feb 2016. doi:10.1145/2824443.
- [KW19] Jae Kwang Kim and Zhonglei Wang. Sampling techniques for big data analysis. *International Statistical Review*, 87:S177–S191, 2019.
- [KZ04] Jan Klein and Gabriel Zachmann. Point Cloud Surfaces Using Geometric Proximity Graphs. *Comput. Graph.*, 28(6):839–850, dec 2004.
- [KZ05] Jan Klein and Gabriel Zachmann. Interpolation Search for Point Cloud Intersection. In *Proc. of WSCG 2005*, pages 163–170, University of West Bohemia, Plzen, Czech Republic, January 31 – February 7 2005.
- [LC87] William E. Lorensen and Harvey E. Cline. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH ’87, page 163–169, New York, NY, USA, 1987. Association for Computing Machinery. doi:10.1145/37401.37422.
- [Lee00] In-Kwon Lee. Curve reconstruction from unorganized points. *Computer Aided Geometric Design*, 17(2):161–177, 2000. doi:10.1016/S0167-8396(99)00044-8.

- [LFXH17] Yong-Jin Liu, Dian Fan, Chun-Xu Xu, and Ying He. Constructing Intrinsic Delaunay Triangulations from the Dual of Geodesic Voronoi Diagrams. *ACM Trans. Graph.*, 36(2), apr 2017. doi:10.1145/2999532.
- [LH21] S. Luo and W. Hu. Score-Based Point Cloud Denoising. In *2021 ICCV*, pages 4563–4572, oct 2021. doi:10.1109/ICCV48922.2021.00454.
- [LLC<sup>+</sup>24] Dilong Li, Chenghui Lu, Ziyi Chen, Jianlong Guan, Jing Zhao, and Jixiang Du. Graph Neural Networks in Point Clouds: A Survey. *Remote Sensing*, 16(14), 2024. doi:10.3390/rs16142518.
- [LRS<sup>+</sup>16] Zorah Lahner, Emanuele Rodola, Frank R Schmidt, Michael M Bronstein, and Daniel Cremers. Efficient globally optimal 2d-to-3d deformable shape matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2185–2193, 2016.
- [LXZ<sup>+</sup>20] Zheng Liu, Xiaowen Xiao, Saishang Zhong, Weina Wang, Yanlei Li, Ling Zhang, and Zhong Xie. A feature-preserving framework for point cloud denoising. *Computer-Aided Design*, 127:102857, 2020. doi:10.1016/j.cad.2020.102857.
- [M<sup>+</sup>67] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.
- [MBMR22] Filippo Maggioli, Daniele Baieri, Simone Melzi, and Emanuele Rodolà. Newton’s Fractals on Surfaces via Bicomplex Algebra. In *ACM SIGGRAPH 2022 Posters*, pages 1–2. 2022.
- [MBRM24] Filippo Maggioli, Daniele Baieri, Emanuele Rodolà, and Simone Melzi. ReMatching: Low-Resolution Representations for Scalable Shape Correspondence, 2024. arXiv:2305.09274.
- [MG23] Shailesh Mishra and Jonathan Granskog. CLIP-based Neural Neighbor Style Transfer for 3D Assets. In Vahid Babaei and Melina Skouras, editors, *Eurographics 2023 - Short Papers*. The Eurographics Association, 2023. doi:10.2312/egs.20231006.
- [MHS<sup>+</sup>20] Mohammad Sultan Mahmud, Joshua Zhexue Huang, Salman Salloum, Tamer Z Emara, and Kuanishbay Sadatdiynov. A survey of data partitioning and sampling methods to support big data analysis. *Big Data Mining and Analytics*, 3(2):85–101, 2020.
- [MIOW24] Diana Marin, Filip Ilic, Stefan Ohrhallinger, and Michael Wimmer. SIGnificant Outlier Removal. *Under Review*, 2024.

- [MK03] Jonah C McBride and Benjamin B Kimia. Archaeological fragment reconstruction using curve-matching. In *2003 Conference on Computer Vision and Pattern Recognition Workshop*, volume 1, pages 3–3. IEEE, 2003.
- [MKOW24] Diana Marin, Patrick Komon, Stefan Ohrhallinger, and Michael Wimmer. Distributed Surface Reconstruction. In Lingjie Liu and Melinos Averkiou, editors, *Eurographics 2024 - Posters*. The Eurographics Association, 2024. doi:10.2312/egp.20241037.
- [MMM<sup>+</sup>24] D. Marin, F. Maggioni, S. Melzi, S. Ohrhallinger, and M. Wimmer. Reconstructing Curves from Sparse Samples on Riemannian Manifolds. *Computer Graphics Forum*, 43(5):e15136, 2024. doi:10.1111/cgf.15136.
- [MMMR22] Filippo Maggioni, Riccardo Marin, Simone Melzi, and Emanuele Rodolà. MoMaS: Mold Manifold Simulation for Real-time Procedural Texturing. *Computer Graphics Forum*, 2022. doi:10.1111/cgf.14697.
- [MN03] Niloy J. Mitra and An Nguyen. Estimating Surface Normals in Noisy Point Cloud Data. In *Proceedings of the Nineteenth Annual Symposium on Computational Geometry*, SCG '03, page 322–328, New York, NY, USA, 2003. Association for Computing Machinery. doi:10.1145/777792.777840.
- [MNPP21] C. Mancinelli, G. Nazzaro, F. Pellacini, and E. Puppo. b/Surf: Interactive Bezier Splines on Surface Meshes. *IEEE Transactions on Visualization & Computer Graphics*, (01):1–1, may 2021. doi:10.1109/TVCG.2022.3171179.
- [Mor01] Shigeyuki Morita. *Geometry of differential forms*. Number 201. American Mathematical Soc., 2001.
- [MOW22a] Diana Marin, Stefan Ohrhallinger, and Michael Wimmer. SIG-based Curve Reconstruction. In Basile Sauvage and Jasminka Hasic-Telalovic, editors, *Eurographics 2022 - Posters*. The Eurographics Association, 2022. doi:10.2312/egp.20221013.
- [MOW22b] Diana Marin, Stefan Ohrhallinger, and Michael Wimmer. SIGDT: 2D Curve Reconstruction. *Computer Graphics Forum*, 41(7):25–36, October 2022. doi:10.1111/cgf.14654.
- [MOW23] Diana Marin, Stefan Ohrhallinger, and Michael Wimmer. Parameter-Free and Improved Connectivity for Point Clouds. In Gurprit Singh and Mengyu (Rachel) Chu, editors, *Eurographics 2023 - Posters*. The Eurographics Association, 2023. doi:10.2312/egp.20231023.
- [MOW24] Diana Marin, Stefan Ohrhallinger, and Michael Wimmer. Parameter-Free Connectivity for Point Clouds. In *Proceedings of the 19th International*

*Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - GRAPP*, pages 92–102. INSTICC, SciTePress, 2024. doi:10.5220/0012394900003660.

- [MP23] Claudio Mancinelli and Enrico Puppo. Computing the Riemannian center of mass on meshes. *Computer Aided Geometric Design*, 103:102203, 2023.
- [MPM15] Subhasree Methirumangalath, Amal Dev Parakkat, and Ramanathan Muthuganapathy. A unified approach towards reconstruction of a planar point set. *Computers & Graphics*, 51:90–97, 2015. International Conference Shape Modeling International. doi:10.1016/j.cag.2015.05.025.
- [MPO<sup>+</sup>24] Diana Marin, Amal Dev Parakkat, Stefan Ohrhallinger, Michael Wimmer, Steve Oudot, and Pooran Memari. SING: Stability-Incorporated Neighborhood Graph. *Under Review*, 2024.
- [MQ99] T.S Michael and T Quint. Sphere of influence graphs in general metric spaces. *Mathematical and Computer Modelling*, 29(7):45–53, 1999. doi:10.1016/S0895-7177(99)00061-8.
- [MXA04] Mohamed Mokbel, Xiaopeng Xiong, and Walid Aref. SINA: Scalable Incremental Processing of Continuous Queries in Spatio-temporal Databases. *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 04 2004. doi:10.1145/1007568.1007638.
- [NPP21] Giacomo Nazzaro, Enrico Puppo, and Fabio Pellacini. GeoTangle: Interactive Design of Geodesic Tangle Patterns on Surfaces. *ACM Trans. Graph.*, 41(2), nov 2021. doi:10.1145/3487909.
- [OM13] S. Ohrhallinger and S. Mudur. An Efficient Algorithm for Determining an Aesthetic Shape Connecting Unorganized 2D Points. *Computer Graphics Forum*, 2013. doi:10.1111/cgf.12162.
- [OMW16] Stefan Ohrhallinger, S.A. Mitchell, and Michael Wimmer. Curve Reconstruction with Many Fewer Samples. *Computer Graphics Forum*, 35:167–176, 08 2016. doi:10.1111/cgf.12973.
- [OPP<sup>+</sup>21] Stefan Ohrhallinger, Jiju Peethambaran, Amal D Parakkat, Tamal K Dey, and Ramanathan Muthuganapathy. 2D Points Curve Reconstruction Survey and Benchmark. In *Computer Graphics Forum*, volume 40, pages 611–632. Wiley Online Library, 2021.
- [Oud15] Steve Y Oudot. Persistence theory: from quiver representations to data analysis. *Mathematical Surveys and Monographs*, 209:218, 2015.
- [PC06] Gabriel Peyré and Laurent D Cohen. Geodesic remeshing using front propagation. *International Journal of Computer Vision*, 69:145–156, 2006.



- [PKKG03] Mark Pauly, Richard Keiser, Leif Kobbelt, and Markus Gross. Shape modeling with point-sampled geometry. *ACM SIGGRAPH 2003 Papers, SIGGRAPH '03*, 01 2003. doi:10.1145/1201775.882319.
- [POEM24] Amal Dev Parakkat, Stefan Ohrhallinger, Elmar Eisemann, and Pooran Memari. BallMerge: High-quality Fast Surface Reconstruction via Voronoi Balls. *Computer Graphics Forum*, 2024. doi:10.1111/cgf.15019.
- [PSOA18] Mario Poerner, Jochen Suessmuth, Davoud Ohadi, and Vincenz Amann. Adidas TAPE: 3-d Footwear Concept Design. In *ACM SIGGRAPH 2018 Talks, SIGGRAPH '18*, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3214745.3214761.
- [RBP07] Shubhra Sankar Ray, Sanghamitra Bandyopadhyay, and Sankar K Pal. Genetic operators for combinatorial optimization in tsp and microarray gene ordering. *Applied intelligence*, 26:183–195, 2007.
- [RBS<sup>+</sup>17] Marc Rautenhaus, Michael Böttinger, Stephan Siemen, Robert Hoffman, Robert M Kirby, Mahsa Mirzargar, Niklas Röber, and Rüdiger Westermann. Visualization in meteorology—a survey of techniques and tools for data analysis tasks. *IEEE Transactions on Visualization and Computer Graphics*, 24(12):3268–3296, 2017.
- [RLB23] Paul Roetzer, Zorah Lähner, and Florian Bernard. Conjugate Product Graphs for Globally Optimal 2D-3D Shape Matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21866–21875, 2023.
- [RLBG<sup>+</sup>20] Marie-Julie Rakotosaona, Vittorio La Barbera, Paul Guerrero, Niloy J. Mitra, and Maks Ovsjanikov. PointCleanNet: Learning to Denoise and Remove Outliers from Dense Point Clouds. *Comp. Graph. Forum*, 39(1):185–203, 2020. doi:10.1111/cgf.13753.
- [RPY<sup>+</sup>22] Yazhou Ren, Jingyu Pu, Zhimeng Yang, Jie Xu, Guofeng Li, Xiaorong Pu, Philip S. Yu, and Lifang He. Deep Clustering: A Comprehensive Survey. *IEEE transactions on neural networks and learning systems*, PP, 2022. URL: <https://api.semanticscholar.org/CorpusID:252780393>.
- [RXL<sup>+</sup>23] Xingcheng Ran, Yue Xi, Yonggang Lu, Xiangwen Wang, and Zhenyu Lu. Comprehensive survey on hierarchical clustering algorithms and the recent developments. *Artificial Intelligence Review*, 56(8):8219–8264, 2023.
- [SC13] Pratik Shah and Samaresh Chatterji. On the Curve Reconstruction in Riemannian Manifolds: Ordering Motion Frames. *Journal of mathematical imaging and vision*, 45:55–68, 2013.

- [SC<sup>+</sup>19] Nicholas Sharp, Keenan Crane, et al. GeometryCentral: A modern C++ library of data structures and algorithms for geometry processing. 2019.
- [SC20] Nicholas Sharp and Keenan Crane. You Can Find Geodesic Paths in Triangle Meshes by Just Flipping Edges. 39(6), nov 2020. doi:10.1145/3414685.3417839.
- [Sil18] Bernard W Silverman. *Density estimation for statistics and data analysis*. Chapman & Hall, 2018.
- [SM00] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.
- [SM15] N Sathya and A Muthukumaravel. A review of the optimization algorithms on traveling salesman problem. *Indian Journal of Science and Technology*, 8(29):1–4, 2015.
- [SR20] Weijing Shi and Raj Rajkumar. Point-gnn: Graph neural network for 3d object detection in a point cloud. In *Proc. of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1711–1719, 2020.
- [Tal14] Ayellet Tal. 3D shape analysis for archaeology. In *3D Research Challenges in Cultural Heritage: A Roadmap in Digital Heritage Preservation*, pages 50–63. Springer, 2014.
- [TBF16] Maria-Laura Torrente, Silvia Biasotti, and Bianca Falcidieno. Feature Identification in Archaeological Fragments Using Families of Algebraic Curves. In *GCH*, pages 93–96, 2016.
- [Teb07] Marc Teboulle. A Unified Continuous Optimization Framework for Center-Based Clustering Methods. *Journal of Machine Learning Research*, 8(1), 2007.
- [Tou80] Godfried T. Toussaint. The relative neighbourhood graph of a finite planar set. *Pattern Recognition*, 12(4):261–268, 1980. doi:10.1016/0031-3203(80)90066-7.
- [Tou88] Godfried T Toussaint. A graph-theoretical primal sketch. In *Machine Intelligence and Pattern Recognition*, volume 6, pages 229–260. Elsevier, 1988.
- [TPM20] Safeer Babu Thayyil, Amal Dev Parakkat, and Ramanathan Muthuganapathy. An input-independent single pass algorithm for reconstruction from dot patterns and boundary samples. *Computer Aided Geometric Design*, 80:101879, 2020. doi:10.1016/j.cagd.2020.101879.



- [TPM21] Safeer Babu Thayyil, Jiju Peethambaran, and Ramanathan Muthuganapathy. A sampling type discernment approach towards reconstruction of a point set in  $\mathbb{R}^2$ . *Computer Aided Geometric Design*, 84:101953, 2021. doi:10.1016/j.cagd.2020.101953.
- [Tur01] Greg Turk. Texture synthesis on surfaces. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 347–354, 2001.
- [VL07] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17:395–416, 2007.
- [Wan15] Xiaoning Wang. Intrinsic computation of voronoi diagrams on surfaces and its application, 2015. doi:10.32657/10356/65864.
- [Wat81] D. F. Watson. Computing the n-dimensional Delaunay tessellation with application to Voronoi polytopes\*. *The Computer Journal*, 24(2):167–172, 01 1981. doi:10.1093/comjnl/24.2.167.
- [Wat90] Bill Watterson. Calvin and Hobbes, 1990. Accessed: 2024-08-02. URL: <https://www.gocomics.com/calvinandhobbes/1990/10/06>.
- [WHH<sup>+</sup>19] Lei Wang, Yuchun Huang, Yaolin Hou, Shenman Zhang, and Jie Shan. Graph Attention Convolution for Point Cloud Semantic Segmentation. In *2019 CVPR*, pages 10288–10297, 2019. doi:10.1109/CVPR.2019.01054.
- [WL01] Li-Yi Wei and Marc Levoy. Texture synthesis over arbitrary manifold surfaces. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 355–360, 2001.
- [WSSC11] Thomas Windheuser, Ulrich Schlickewei, Frank R Schmidt, and Daniel Cremers. Geometrically consistent elastic matching of 3d shapes: A linear programming solution. In *2011 International Conference on Computer Vision*, pages 2134–2141. IEEE, 2011.
- [WT10] Daniela M Witten and Robert Tibshirani. A framework for feature selection in clustering. *Journal of the American Statistical Association*, 105(490):713–726, 2010.
- [WWW<sup>+</sup>24] Zixiong Wang, Pengfei Wang, Peng-Shuai Wang, Qiujie Dong, Junjie Gao, Shuangmin Chen, Shiqing Xin, Changhe Tu, and Wenping Wang. Neural-IMLS: Self-Supervised Implicit Moving Least-Squares Network for Surface Reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 30(8):5018–5033, 2024. doi:10.1109/TVCG.2023.3284233.

- [XSL<sup>+</sup>23] Dong Xiao, Zuoqiang Shi, Siyu Li, Bailin Deng, and Bin Wang. Point normal orientation and surface reconstruction by incorporating isovalue constraints to poisson equation. *Computer Aided Geometric Design*, 103:102195, 2023. doi:10.1016/j.cagd.2023.102195.
- [YLT19] Cem Yuksel, Sylvain Lefebvre, and Marco Tarini. Rethinking Texture Mapping. *Computer Graphics Forum*, 38(2):535–551, 2019. doi:10.1111/cgf.13656.
- [YP20] Hua Qi Yao and Zhang Gao Peng. Feature Extraction and Redesign of Bronze Geometry Patterns in Shang and Zhou Dynasties of China. *Int. J. Eng. Res. Technol*, 9(2):267–271, 2020.
- [ZJ16] Qingnan Zhou and Alec Jacobson. Thingi10K: A Dataset of 10, 000 3D-Printing Models. *ArXiv*, abs/1605.04797, 2016. URL: <https://api.semanticscholar.org/CorpusID:39867743>.