Statistical Approaches to Multi-scale Point Cloud Processing

Ranjith Unnikrishnan

CMU-RI-TR-08-15

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Robotics.

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

May 2008

Thesis Committee:
Martial Hebert (*chair*)
James A. Bagnell
Daniel Huber
Gabriel Taubin (*Brown University*)

Contents

List of Figures

List of Algorithms

Abstract

In recent years, 3D geometry has gained increasing popularity as the new form of digital media content. Due to advances in sensor technology, it is now feasible to acquire highly detailed 3D scans of complex scenes to obtain millions of data points at high sampling rates over large spatial extents. This ability to acquire high-resolution depth information brings with it the possibility of using 3D geometric data to construct detailed shape models and of perhaps combining 3D depth with visual appearance from images to address challenging problems in computer vision.

However, geometric information represented as a 3D point cloud presents challenges uniquely different from other data modalities such as images or audio. Due to a combination of reasons such as the spatial irregularity of the data and the implicit nature of 3D observations, an easy substitution of traditional signal processing operators from images for processing unorganized 3D points is not possible. Furthermore, traditional estimators from classical statistics are not suitable for processing data in this domain, and new algorithms as well as different criteria for evaluating these algorithms are necessary.

This dissertation contributes towards the development of two fundamental building blocks for processing point clouds. The first is of geometric model fitting, where we present a class of locally semi-parametric estimators that allows finite-sample analysis of accuracy and also explicitly addresses the problem of support-radius selection in local fitting. The second is of multi-scale filtering operators for point clouds that allow detection of interest regions whose locations as well as spatial extent are completely data-driven. The proposed approaches are distinguished from related work by operating directly in the input 3D space on unorganized points without assuming an available mesh or resorting to an intermediate global 2D parameterization.

Results are presented for several applications including surface reconstruction, accurate shape descriptor computation and repeatable interest region detection, on synthetic data, as well as outdoor aerial and ground-based data obtained with a laser scanner.

Funding Sources

This document was prepared through collaborative participation in the Robotics Consortium, and we gratefully acknowledge sponsorship by the U.S Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-209912.

Acknowledgments

I owe much gratitude to my advisor, Martial Hebert, for his guidance and for supporting my varied interests in computer vision over the past several years. His passion for teaching, and his technical rigor and insight in nearly every aspect of computer vision are unmatched and will remain a constant inspiration. I am grateful for having the chance to work closely with him and for learning as much I did through our interaction.

I thank my other thesis committee members: Gabriel Taubin, Drew Bagnell and Daniel Huber, for their guidance and for making their time and knowledge accessible to me. I owe a lot to Nicolas Vandapel, Caroline Pantofaru, and Jean-François Lalonde, for their friendship, collaboration and advice. I am grateful to the many members of the MISC reading group for the passionate and always entertaining discussions that have taught me a lot about computer vision.

I wish to also thank Alonzo Kelly, who was my advisor while I was in the M.S program, for his invaluable guidance during the start of my graduate studies and for helping to shape the way I conduct my research. I am very grateful to Suzanne Lyons for so efficiently taking care of the various administrative matters that come with being a Robotics graduate student. I am also indebted to innumerable friends at CMU and elsewhere for their constant motivation and for ensuring that I had a life outside graduate school.

Last but far from least, I am forever indebted to my family for their love and support. Without my wife Sushmita's endless encouragement and understanding through my years as a graduate student, this thesis would not have been possible.

Table of Notation

Below is a list of mathematical notation used throughout this document grouped by the chapters in which they first appear.

Chapter 2: Local Semi-parametric Analysis

$\mathcal{M}$ ..... Unknown manifold

$\mathbf{x}_i^o$ ..... Point lying on $\mathcal{M}$ $(i = 1 \dots n)$

$\mathbf{x}_i$ ..... Noisy observation of $\mathbf{x}_i^o$ with $\mathbf{x}_i = (x_i, y_i, z_i)$

η_i ..... Noise associated with observation $\mathbf{x}_i$ with $\eta_i = (\eta_{x,i}, \eta_{y,i}\eta_{z,i})$

Λ_i ..... Variance of noise η_i

θ ..... Unknown model parameters. $\theta = (\theta_1, \theta_2, \dots, \theta_m)$

$f(\mathbf{x}, \theta)$ ..... Function of position $\mathbf{x}$ whose zero-level set represents a surface with model parameters θ

$\mathbf{v}(\mathbf{x})$ ..... Problem-specific map from coordinates $\mathbf{x}$ to a higher dimensional vector, such as the monomials of $\mathbf{x}$

$\mathcal{N}(\mathbf{x}_i, r_i)$ ..... Neighborhood around point $\mathbf{x}_i$ defined by radius r_i. The r_i argument is sometimes dropped for brevity

$\mathbf{s}_i$ ..... Intrinsic coordinates of point $\mathbf{x}_i^o$ on $\mathcal{M}$

$\bar{\theta}_n$ ..... Estimate of θ from n observations

κ ..... Curvature of line or normal curve (at origin of interest)

τ ..... Torsion of line (at point of interest)

r ..... Radius of neighborhood (around point of interest) considered for geometric fitting

σ_0 ..... Std. deviation of noise in each coordinate

X ..... Random variable for the x-coordinate. Y and Z are defined similarly.

$\bar{X}_n$ ..... Estimate of $\mathbb{E}(X)$ from n samples of X.

μ_X ..... $= \mathbb{E}(X)$, mean of distribution of random variable X

$d_m(X)$ ..... $= \mathbb{E}(X - \mu_X)^m$, capturing centered statistical dispersion of random variable X

$c_m(X, Y)$ ..... $= \mathbb{E}[(X - \mu_X)(Y - \mu_Y)]^m$, capturing generalized covariance of two random variables X and Y.

$\hat{M}_n$ ..... Estimator of the scatter (covariance) matrix from the n points in a local neighborhood.

$\bar{M}$ ..... $= \mathbb{E}(\hat{M}_n)$, the expected value of the scatter matrix

Q ..... Perturbation matrix that deviates estimate of tangent (normal) away from its true value

δ ..... Spectral gap of $\hat{M}$

$B(r)$ ..... Error bound in estimate of tangent (normal)

$\|A\|_F$ ..... Frobenius norm of matrix A

$\mathbf{n}$ ..... Surface normal (at point of interest)

Π_φ ..... Normal plane at angle φ to some reference vector in the tangent plane, and containing the normal at that point.

Γ_φ ..... Normal curve formed by intersection of plane Π_φ with manifold $\mathcal{M}$

κ_1, κ_2 ..... Principal curvatures at point of interest. $\kappa_1 > \kappa_2$

Chapter 3: Constrained Local Regression

γ_i ..... Model parameters of one of the lines (or planes) at a sharp intersection in 2D (or 3D)

$\phi(\theta)$ ..... Vector of algebraic functions of parameters θ such that $\phi(\theta) = 0$ represents algebraic constraints to be satisfied for the corresponding manifold to be degenerate.

$\hat{\mathbf{x}}_i$ ..... Estimate of true position $\mathbf{x}_i^o$ of point on manifold $\mathcal{M}$

w_i ..... Weight on point $\mathbf{x}_i$ normally computed as a function of distance from the point of interest $\mathbf{x}$

Chapter 4: Multi-scale Signal Representation

$\alpha(s)$ ..... 2D curve parameterized by distance

$A(\mathbf{x}, t)$ ..... Kernel operator on position $\mathbf{x}$ with bandwidth t

$p_t(\mathbf{x})$ ..... Estimate of point density at location $\mathbf{x}$

$\mathcal{L}_\mathcal{M}$ ..... Laplace-Beltrami (LB) operator

H ..... Mean curvature (at point of interest)

$\tilde{\phi}(\mathbf{x}_i, \mathbf{x}_j, t)$ ..... Density normalized Gaussian kernel function

$\tilde{A}(\mathbf{x}, t)$ ..... Density normalized kernel operator on position $\mathbf{x}$ with bandwidth t

$F(\mathbf{x}, t)$ ..... Invariant computed at point $\mathbf{x}$ at scale t

CHAPTER 1

Introduction

This dissertation presents approaches to process and analyze 3D geometry represented as an unorganized collection of spatial points. In recent years, 3D geometry has gained increasing popularity as the new form of digital media content. One testament to this phenomenon is the marked increase in the number of digital models available on the web [1, 2, 3, 4, 41] for both commercial and non-commercial use. This emergence of 3D media has been partly fueled by new techniques to automatically create intricate 3D models from large sets of partial-view scans.

Due to advances in sensor technology, it is also now feasible to obtain highly detailed scans of complex scenes over large spatial extents, with millions of data points at high sampling rates [5]. Previously a specification of high resolution and high accuracy confined 3D model acquisition to small spatial extents. These restrictions on resolution and accuracy have limited the utility of 3D data for solving problems relevant to mobile robotics. Because of the relatively lower resolution of 3D laser range scanners in comparison with 2D cameras, the processing of 3D data has been largely confined to the computation of features that only reflect coarse geometric detail, sufficient to distinguish between a few classes of terrain type.

Now, the ability to acquire high-resolution depth information brings with it the possibility of using 3D geometric data to construct detailed shape models and of perhaps combining it with 2D appearance information toward solving the general scene understanding problem. This new possibility in turn brings with it the questions of how best to represent shape information obtained from 3D sensors, and how to process the 3D data managed with such a representation to accomplish useful visual tasks such as surface reconstruction

and interest region detection.

In this chapter, we motivate the work in this dissertation and present arguments for why raw point clouds are a suitable geometric primitive to represent shapes, discuss the kind of signal processing tools that are required to make use of the information they represent, and outline the challenges they present.

1.1 Representing 3D shape

A natural question raised by the problem of 3D data processing is that of how to represent or mathematically describe geometric information obtained from 3D sensors. Over the past several decades, this choice has largely been driven by a combination of computational hardware as well as the target application.

Early surface representations such as B-splines and Bezier curves were continuous approximations of shape, relying on interpolation schemes to lower memory consumption. Many of these continuous approximations are still in use today in commercial CAD systems. With the advent of computers with more memory and computational power, the continuous approximations gave way to discrete approximations, some of which we list below:

(i) **Range image:** Range images are formed when sensors such as optical triangulation scanners produce depth values on a regular 2D sampling lattice. Thus, they are conceptually equivalent to regular intensity images, with the exception of the intensity attribute replaced by the distance from the sensor to the physical location imaged by the corresponding pixel. Because of this equivalence, range images allow easy, though not always appropriate, substitution of operators from traditional 2D image signal processing to the domain of 3D data processing. Several researchers have opted to work exclusively in this representation to perform visual processing tasks [19, 47, 50].

However, the sensor geometry of laser range scanners need not confirm with a regular lattice structure of an image. Because of this, the construction of range-images from raw input point clouds can involve some loss of information as a result of multiple 3D points projecting to the same 2D range image pixel. Related to this, when there is a mismatch between range image resolution and scanner resolution, there may be cells for which no data points were observed. This situation of "empty" range cells is not one that traditional image processing operators, such as convolution filters, are equipped to handle.

In addition, for applications relevant to mobile robotics, it is reasonable to expect multiple observations to be made for the same scene from different vantage points.

In this scenario, it is not straightforward to combine information from multiple range images of the same scene without reverting to the original 3D input space.

(ii) **Polygon mesh:** Meshes are undirected graphs composed of a collection of vertices and edges, whose faces represent discrete piecewise approximations of surfaces. Unlike range images, meshes have the advantage of allowing irregular surface sampling. Triangulated meshes have particularly enjoyed much popularity due to a combination of a simple data format, and the fact the modern graphics hardware pipelines support extremely fast rendering of triangles. Much research has gone into the construction of meshes and toward extending signal processing concepts to them [30, 54, 59, 78, 99, 106, 107, 123].

However, meshes are normally not a direct output of a 3D sensor. The construction of a mesh from noisy data is not straightforward and requires denoising and additional preprocessing that can undesirably remove geometric detail. Like range image, meshes are also cumbersome to add information to, as, say the addition of a newly observed 3D point would require breaking and reconstructing the original mesh.

(iii) **Voxel grids:** Named by combining the words "volume" and "pixel", voxels represent spatial occupancy on a regular grid in 3D space. Conversion of input 3D data to a voxel grid can be done by simply dividing the space into a regular grid and storing sufficient statistics for the set of points falling into each grid cell. These statistics may be varied to suit the target application, from a simple indicator variable indicating occupancy, or more complex functions such as mean position or point scatter. Unlike range images, voxel grid representation are amenable to data addition, making them a popular choice in mobile robotics applications [111] where observations tend to have a high degree of spatial overlap.

However, like range images, they suffer from the same problems of information loss and empty cells which make use of traditional image processing operators unsuitable. In addition, the fidelity of voxel-based representations is limited by the grid resolution and although variable-resolution data structures like oct-trees may be used to adapt to the spatial distribution of the data, high-resolution grids still require a lot of memory for storage.

Why point clouds?

In contrast to the alternatives, working directly with **raw point clouds** in the input 3D space offers several advantages. Point clouds are a natural way to represent 3D sensor output and there is no assumption of available connectivity information or underlying topology. It is also better suited for dynamic applications requiring data addition and deformation.

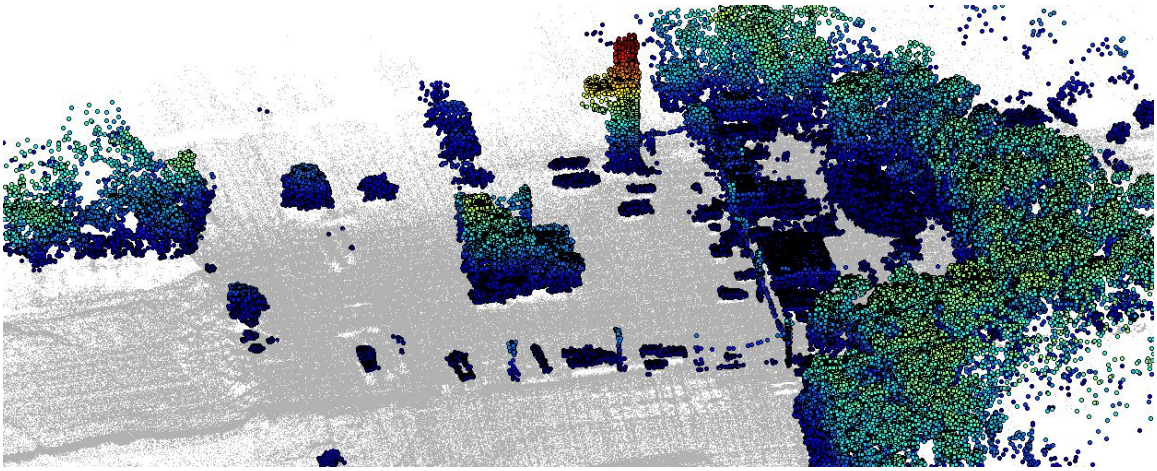


Figure 1.1: View of 3D points obtained from an aerial ladar scan containing 1.25 million points. Points labeled as associated with the ground are plotted smaller and colored gray for clarity. Remaining points are colored by elevation using the 'jet' colormap. Two relatively taller building structures may be observed near the center of the image, along with smaller man-made structures and unstructured vegetation in other areas. The irregular spatial arrangement of the observed points can be clearly seen.

Recent years have also seen a revival amongst the graphics community in the use of point clouds directly as a rendering primitive, as it circumvents the need for difficult mesh construction procedures. It is also a more data efficient alternative when the available point data density exceeds the viewing screen resolution [66], as there is no need to maintain, store and render the triangles associated with each edge of a mesh.

However, unlike other data modalities such as images, video or sound, geometric data processing presents a unique set of challenges that prevents the direct applicability of existing signal processing tools.

(i) **Irregular sampling:** Because of the irregular sampling pattern used by typical 3D sensors to acquire geometric information, point clouds typically lack any regular lattice-like structure. This prevents easy substitution of traditional signal filtering operators from images for point cloud processing.

(ii) **Implicit function:** Unlike images which possess a natural representation as an intensity function explicitly defined on spatial locations, the observed "signal" in point clouds is implicit and represented only by the spatial arrangement of the observed points. Because of this, traditional signal processing operators from images are not applicable.

(iii) **Range-dependent noise:** Many laser range sensors based on time-of-flight, multiple frequency phase shift, as well as other devices based on optical triangulation or structure-from-motion have the property that the noise level of the observation varies as a function of distance of the observed point to the sensor. In many applications, a model of this noise level variation is available and must be incorporated in the signal processing pipeline.

(iv) **Required invariance:** A natural requirement of low-level image processing techniques to the generation of output that is invariant to changes in the viewing conditions when the image was constructed. For instance, interest region detection algorithms are typically constructed to be invariant to image magnification or a scaling of image intensity, corresponding to a change in focal length and global scene illuminant intensity respectively. In the domain of 3D geometry processing, the underlying constant is surface shape, as opposed to appearance, and the change in sensing conditions that 3D point processing algorithms need to be invariant to is the sampling density of the acquired point cloud. This requirement constitutes another point of difference from image processing that is unaddressed in the literature.

There are other artifacts that are specific to existing laser sensor technology, such as those known as *mixed pixels*. These occur when, due to the non-point spot size of the beam, the recorded distance by the sensor is corrupted when the physical surface in contact with the laser beam has significant depth variation. We will not give separate attention to mixed pixels [109] in this document but treat their effect as one that is captured by our sensor noise model.

In the next section, we place the above challenges in the context of useful point cloud processing tasks. We then list our contributions towards meeting these challenges along with an outline of the document.

1.2 Motivating problem

To illustrate the types of point cloud processing problems relevant to practical application, consider the example task of constructing a patch-based shape model from point clouds. A patch-based model is one where the object or scene of interest is represented as a collection of shape descriptors, such that each descriptor encodes the shape of some small neighborhood in the original data. Such a strategy is appealing because, by focusing attention away from the entirety of the input data to small regions, we place ourselves in a better position to handle practical challenges such as missing data (occlusion) and other objects in the scene (clutter).

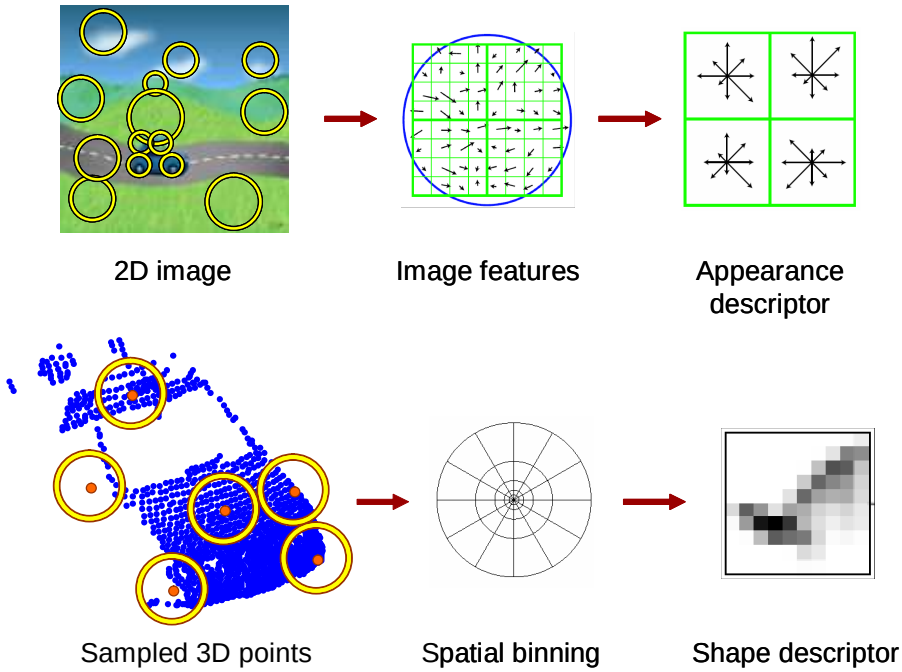


Figure 1.2: Procedural similarity in the construction of appearance models from images and shape models from 3D point samples.

We wish to emphasize that while the goal of this dissertation is *not* the construction of shape models, this task provides a suitable explanatory tool for two reasons. First, the construction of reliable shape models from points is of great practical value and requires good solutions to both the types of problems that we discuss below. Second, because of the procedural similarity of this task to the corresponding task of constructing patch-based appearance models from 2D images, there are several useful analogies that can be drawn to compare and contrast existing techniques in 3D point cloud processing to its 2D counterparts. (See Figure 1.2)

Several approaches to construct patch-based shape models, as well as other useful tasks such as scan registration follow a two-step process outlined below.

The first step is the selection of so-called *interest regions*, where each such region defines a point of interest and a neighborhood around that point that are in some way salient to the underlying shape. In the 2D image domain, this selection process is done using an interest region *detector*.

The second step in the processing pipeline is the construction of a shape descriptor for each interest region that encodes the geometry of that region in some way. Some popular shape descriptors include the spin image [53], the 3D shape context [15], and descriptors based on spherical harmonics [57]. In the 2D image domain, some authors choose to augment

descriptors with information reflecting their relative spatial locations [39]. We emphasize at this point that we will not be attempting to devise yet another kind of shape descriptor. Instead, we will focus on the properties that *any* shape descriptor ought to have in order to be reliable in the context of a patch-based shape model, and will work toward ensuring that those properties are satisfied.

For a patch-based model to be useful as a faithful representation of the object's shape or appearance, the algorithm used to execute the above two-step process is typically required to satisfy two criteria:

(i) **Coverage:** The number and spread of detected interest regions should be sufficient so as to adequately capture the variation in shape or appearance of the object of interest.

(ii) **Repeatability:** Both components of the model construction process – the detection of interest regions, and the computation of shape descriptors in those regions – should be as invariant as possible to changes in sensing conditions. For instance, a 3D region detected in one scan should also be detected on processing any other scan where it is visible. In the image domain, several detection algorithms exist to guarantee invariance to intensity scaling and affine deformation [79]. As mentioned earlier, the corresponding requirement in the 3D domain is that of invariance to changes in sampling and is not straightforward to achieve.

In what follows, we will show how the above two types of repeatability requirements naturally lead to the two instances of the point cloud processing operations we address in this document. In particular we show how the requirement of repeatability in computing shape descriptors is connected to the problem of geometric model fitting, and how the requirement of repeatability in region selection relates to the problem of interest region selection from unorganized point clouds.

Repeatability of shape descriptors

Although perhaps not immediately obvious, the problem of shape modeling, or *geometric fitting*, can be seen to arise out of the need to construct shape descriptors repeatably.

Several shape descriptors operate by constructing a histogram of point locations through a radial [15] or cylindrical [53] spatial arrangement of bins (Figures 1.2 and 1.3). While this procedure works well when the sampling density of points is high, the fidelity of the descriptor to the underlying shape degrades quickly as the number of sampled points decreases. This is partly due to a combination of the higher variability of the individual bins when there are fewer overall points and the fact that some bins in the descriptor may not have any observations at all.

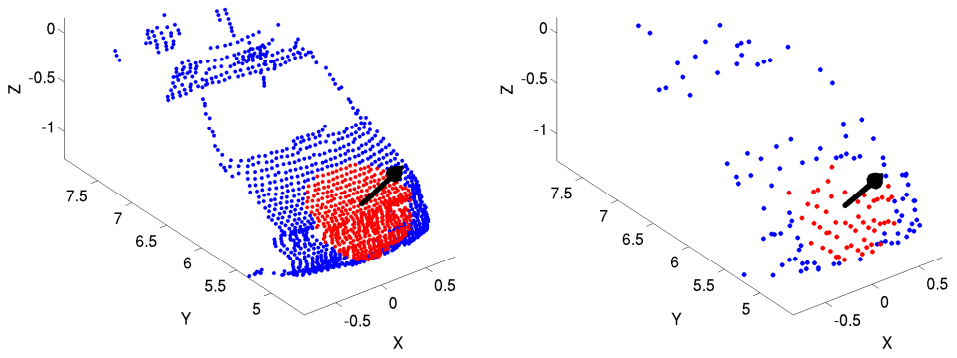


Figure 1.3: Shown are points from a full-resolution and a low-resolution scan of a car hood from the same position. The points in red correspond to the same physical region on the car. We would ideally like the shape descriptors constructed from both these neighborhoods to be identical in order to achieve invariance to changes in sampling density.

One way to circumvent this is to first fit a surface shape to the points that *have been* observed. Then new points can be synthetically resampled on the surface at sufficiently high and uniform density, and a descriptor then constructed from these synthesized points would be reliable again.[1]

For this modified descriptor construction procedure to work, the algorithm used to fit a surface to the few available points must have sufficiently high accuracy so as to not bias the reconstruction away from the true shape. However, as elaborated in Chapter 2, most existing methods are not appropriate for this geometric fitting task as they do not come with finite-sample guarantees of accuracy and require prior specification of certain *scale* parameters. In Chapter 2, we present an approach to geometric fitting that enjoys the benefits of both finite-sample error analysis as well as asymptotic convergence to the true solution, while simultaneously addressing the problem of scale selection. In Chapter 3, we extend this idea to show that non-smooth surface features, such as edges, can also be included in this modified regression framework.

Repeatability of region selection

The second type of repeatability requirement is that of invariance in the selection of interest regions to changes in point sampling density. Much of the work in geometry processing over the past two decades has focused on triangle mesh representations which, by assuming the availability of perfect connectivity information from the start, reduce the impact of

[1] A similar idea was employed in [23] where points were synthesized on faces of a mesh by interpolation to handle low-resolution datasets, although the implementation of this idea assumed that a triangulated mesh faithful to the underlying surface was already available.

potentially low number of points in the model. The use of unorganized point samples is more faithful to the format in which data is typically acquired by sensors, and avoids the burden of recovering this connectivity explicitly and of constructing triangulated surface representations.

However, existing work on interest region detection from point clouds suffers from several drawbacks. First, they make no formal guarantee on the repeatability of the detection with respect to sampling density. Second, although objects generally exhibit geometric variation over several spatial extents, existing interest region detectors operate on one or more fixed preset scales. Finally, many existing detectors rely on heuristic scoring functions that have to be individually crafted for specific geometric features [44], such as for edges or ridges or corners.

For these reasons, current practice [40, 76] for the selection of locations at which to compute shape descriptors has leaned towards either exhaustively or randomly selecting points and heuristically choosing preset neighborhood sizes over which to compute the descriptors. Such a strategy ignores the relative variation in the spatial extent of geometric structures and also risks introducing redundancy in the representation. Note that this procedure is in sharp contrast to its analog in 2D image processing, where several data-driven techniques [71, 79] exist to choose both locations and their associated neighborhood sizes for computing appearance features.

For these reasons, we attempt in Chapter 4 to develop multi-scale filtering operators for point clouds that are invariant to sampling density, and to use them to develop data-driven interest region detectors analogous to those commonly used to process 2D images.

Thus, in pursuing the solutions to both the above types of geometry processing tasks, this dissertation advocates the development of 3D point cloud operators that possess the same kinds of formal guarantees on repeatability as the signal processing techniques commonplace in the 2D image domain. We are optimistic that this technology will place computer vision practitioners in a better position to easily combine both 3D shape and 2D appearance information when they are available. For instance, new algorithms to build 2D appearance models could then be applied straightforwardly to construct 3D shape models, and vice versa.

1.3 Outline and Contributions

While a complete system that processes 3D data to perform scene understanding is beyond the scope of this document, this dissertation presents progress towards some fundamental building blocks that are crucial for achieving this goal.

(i) We show that the problem of local geometric fitting of unorganized points may be cast

as a statistical inference problem through the use of a local shape parameterization and a semi-parametric distribution model. Put in practice, this allows diverse problems such as denoising and surface reconstruction to be solved in the same mathematical framework. (Chapter 2)

(ii) We introduce a class of *locally semi-parametric* estimators for geometric fitting of smooth surfaces that (a) allows asymptotic as well as finite-sample analysis of accuracy and (b) explicitly addresses the problem of support-radius selection in local fitting. This enables the development of accurate reconstruction algorithms that can automatically adapt to variations in surface shape. We present results to demonstrate the improved stability of the algorithm in comparison with other algorithms that require a preset scale parameter. (Chapter 2)

(iii) We show that by modeling sharp geometric features like surface edges and intersections of curves as degenerate parameterizations of smooth surfaces, it is possible to reconstruct them from unorganized point clouds in the same semi-parametric regression framework as smooth surfaces. (Chapter 3)

(iv) We develop a multi-scale operator for point clouds that captures variation in shape at a point relative to its neighborhood, working analogously to the Laplacian filter in 2D images, but while enjoying the property of being invariant to changes in sampling density around that point. (Chapter 4)

(v) Using the developed multi-scale operator, we propose an algorithm for data-driven interest region detection. The approach distinguishes itself from related work by operating directly in the input 3D space without resorting to an intermediate global 2D parameterization or assuming an available connectivity mesh. (Chapter 4)

The chapters are organized into two parts, the first focusing on geometric modeling and the second on multi-scale processing. The document concludes with a discussion of future work in Chapter 5. The two appendices detail the derivation of the semi-parametric estimator specific to curve and surface reconstruction, and give an overview of matrix perturbation theory relevant to the presented work.

Part I

Geometric Modeling

CHAPTER 2

Local Semi-parametric Analysis

In recent years, there has been a resurgence in the use of raw point clouds [60] as the geometric primitive of choice for several modeling tasks such as rendering [7], editing [6, 85, 127], and compression [86].

Because of the increasing polygonal complexity of 3D models in current use and the associated overhead of processing and managing connectivity information in meshes, it is considerably more efficient to render points in high-resolution models rather than polygons. Interestingly, the argument for using point clouds as display primitives can be traced back to a visionary report by Whitted and Levoy [66] in 1985. The ongoing renaissance has resulted in several proposed methods for displaying models, such as surfel rendering [88, 127], splatting [21, 95, 128], and interpolation methods [8, 102, 115].

Nearly all of the above applications of point cloud processing require some additional knowledge of the underlying shape represented by the point samples. For example, rendering requires knowledge of surface normals at each point for visibility and lighting computation. Some shape compression algorithms utilize estimates of tangents to curves as predictors for shape outline encoding and iso-contour compression schemes in triangle-meshes [67].

A similar need for recovering information of surface shape exists in applications outside graphics. For the problem of path planning in mobile robot navigation, there is a frequent need to evaluate the traversability of terrain by reconstructing its shape from observed sparse laser data. Accuracy in the reconstruction is crucial in order to reliably determine *a priori* whether the vehicle will make all-wheel contact with the ground at each point of a candidate trajectory. Some shape descriptors, such as spin images [53], require knowledge of the surface normal at the the points where they are computed in order to guarantee

rotational invariance of the constructed features.

All the above applications require fitting a surface of some form to observed data. Due to the nature of sensing modalities, some immediate concerns arise from the above applications such as data sparseness, irregularity in sampling and range-dependent noise. The subject of this chapter is a mathematically sound approach to geometric fitting that addresses these challenges with finite-sample guarantees of accuracy.

2.1 Related Work

There are several approaches to curve and surface fitting, both non-parametric (tensor voting [77], radial basis functions, etc.) and parametric [67] (moving least-squares approximations [64], implicit parabolic fitting, b-splines, etc.). This section will outline some of the popular approaches in the literature.

There are many approaches to surface and curve reconstruction motivated by techniques from computational geometry. These include algorithms based on Delaunay triangulations [10, 25], such as the crust algorithm [9], the cocone algorithm [11] and its extension called tight cocone [31], and algorithms based on alpha shapes [33, 34]. As summarized in [43], the more successful approaches are based on the construction of Delaunay triangulations. Under the somewhat restrictive assumption of a closed bounded shape, the problem may be transformed into one of filtering the Delaunay tetrahedra whose union approximates the shape interior. The different approaches promise differing extents of theoretical guarantees on the reconstructed shape varying with assumptions on sampling density and smoothness. However, the fact that the reconstruction in these methods can only interpolate through the observed points affects the quality of their results with noisy and sparse data.

Most practical curve and surface reconstruction algorithms are based on local polynomial fitting and its variants. Recent work by Lewiner *et al.* [67] computed the coefficients of an arc-length parameterized third-order approximation to a curve by solving a weighted least-squares problem at each point using only the points in its local neighborhood. The implicit parameter in the algorithm was the considered neighborhood radius, which was preset by fixing the number of neighbors considered at each point. A similar neighborhood selection strategy was used by Cazals *et al.* [26] who fit the local representation of a manifold using coefficients of a truncated Taylor expansion, termed a jet. Hoppe *et al.* [48] and Zwicker *et al.* [127] compute normals to a surface at each point by fitting a plane to its k-nearest neighbors. The success of these algorithms depends crucially on the chosen value of k, and there is little guidance in the literature on how to make that choice.

In the computer vision community, much work has been done on geometric reconstruction using non-parametric tensor voting [77, 104]. A key step of this is a voting procedure

used to aggregate local information at each point or voxel of interest. The vote is in the form of a $d \times d$ tensor, where d is the data dimensionality, indicating preferred direction of normal (or tangent), and the eigen decomposition of the aggregate tensor at a point gives the desired result. Again, a crucial parameter is the choice of the size of the support region for vote collection, usually chosen heuristically. Work in [104] proposed a fine-to-coarse approach in which points likely to form curves are linked together at fine scale to form fragments, and then linked together incrementally as the scale is increased using a heuristic inspired by perceptual grouping. Our work focuses on sparser point sets than used in [104] and thus requires guarantees on the small sample behavior of the choice of estimator.

Closely related theoretical work by Mitra *et al.* [81] addresses the choice of optimal neighborhood size for normal estimation in surfaces using PCA. They derived a bound on the angular error between the estimated normal and true normal, and proposed the optimal radius as the value that minimized that bound. An iterative procedure was suggested that first estimated the local density and curvature, then computed the optimal radius for those values, and repeated the procedure until convergence. However, the obtained closed-form expression had two parameters that relied on knowledge of the observed data distribution and had to be fixed *a priori*. Furthermore, it was unclear whether the behavior predicted by the obtained closed-form expression agreed with real data for finite samples.

In what follows, we describe an approach to local geometric fitting that enjoys the benefits of both finite-sample error analysis as well as asymptotic efficiency. Section 2.2 formulates the problem and makes the case for the locally semi-parametric approach employed in the remainder of this chapter. We point out that Section 2.2 presents a generalization of our previous work in [110] whose analysis was restricted to 2D and 3D curves. Section 2.3 and Section 2.4 will then detail the application of the approach to the analysis of points lying on curves and its extension to surfaces respectively. We then present results in Section 2.5 to validate the behavior predicted by the model on finite real data and demonstrate its accuracy and stability with some applications. We then conclude in Section 2.7 with a summary of the claims made in this chapter along with some directions for future work.

2.2 Approach

In this section, we formulate the geometric fitting problem and develop our solution to it in a manner that satisfies the requirements of our domain. In the process of doing so, we will argue that traditional estimators from classical statistics are insufficient to deal with point sample data, and that both new estimators as well new methods for evaluating these estimators are necessary.

Overview

Before presenting our proposed solution to the geometric fitting problem, we briefly motivate our approach. Our goal in this chapter is two-fold. First, we wish to estimate the parameters of a geometric model that best fit our observed data. Second, we wish to simultaneously obtain some guarantee of the accuracy of our solution that is valid for the sparse datasets we may expect to work with.

To pursue the first objective, we make a design choice of modeling the scene as a combination of local compact regions, each represented by an implicit function having a small number of parameters. This design choice offers the flexibility of modeling scenes that may otherwise be too complex for a function to fit globally. However, this comes at the cost of requiring a principled method to automatically choose the neighborhoods.

One way to go about the second objective of deriving finite-sample guarantees is to first pursue the intermediate goal of deriving asymptotic guarantees that are valid when the number of data samples increases to infinity. Through analyzing this hypothetical scenario, we may hope to relate the estimation error with infinite data to the practical case of finite data. The framework of regression from classical statistics offers several tools for deriving these asymptotic guarantees. Hence, we approach the geometric fitting problem by first mapping it to the regression problem and deriving the asymptotic error for our chosen estimator.

To fold both the above objectives into one approach, we make an assumption on how points in a small neighborhood are spatially distributed. This step introduces the neighborhood size as another variable in the system along with the other unknown variables. By then measuring the deviation of the solution obtained with finite data from the ideal asymptotic data case, we obtain an *error bound* that involves both the unknown parameters encoding the underlying geometry as well as the neighborhood size. The best estimate of the model parameters may then be obtained by simply minimizing this error bound for both sets of unknown variables.

In the next subsection, we will begin by mapping the geometric model fitting problem to that of classical regression and then make the case for fitting local overlapping neighborhoods. Next, we will introduce a semi-parametric distribution model in order to perform asymptotic analysis, and finally obtain in closed form the finite-sample error bounds that we were pursuing. Sections 2.3 and 2.4 that follow will illustrate the application of these ideas for the task of fitting curves and surfaces, respectively, to unorganized points.

Formulation

Our starting point will be the set of available point samples $\{\mathbf{x}_i\} \in \mathbb{R}^d$. The points are assumed to be noisy observations of an unknown underlying curve or surface and follow the

noise model

$$\mathbf{x}_i = \mathbf{x}_i^o + \eta_i \qquad \text{where} \quad \eta_i \sim N(0, \Lambda_i), \tag{2.1}$$

with η_i denotes heteroscedastic (or point-dependent) sensor noise represented as zero-mean Gaussian with variance Λ_i. The points $\mathbf{x}_i^o$ represent the unknown true points lying on the manifold. Throughout this chapter, we will assume that the manifold (curve or surface) under study is smooth, and that the noise variance Λ_i is available through an error model of the sensor used to acquire the points.

One way to represent the underlying manifold mathematically is through a parameterized implicit equation

$$f(\mathbf{x}_i^o; \theta) = 0 \qquad \forall i, \tag{2.2}$$

where θ represent the unknown model parameters. In particular, we will be interested in the bilinear form

$$\mathbf{v}(\mathbf{x}_i^o)^{\mathrm{T}}\theta = 0, \tag{2.3}$$

where $\mathbf{v}$ denotes a problem-specific vector map.

For example, in the case of fitting a plane to 3D points $\mathbf{x} = [\,x\ y\ z\,]$, the vector map $\mathbf{v}(\mathbf{x})$ would simply be of the form

$$\mathbf{v}(\mathbf{x}) = \begin{bmatrix} x & y & z & 1 \end{bmatrix}^{\mathrm{T}},$$

and the model parameters $\theta = [\,n_x\ n_y\ n_z\ d\,]^{\mathrm{T}}$ would be a 4-element column vector, with $\mathbf{n} = [\,n_x\ n_y\ n_z\,]^{\mathrm{T}}$ indicating the normal to the plane and $d/\|\mathbf{n}\|$ equal to the distance from the plane to the origin.

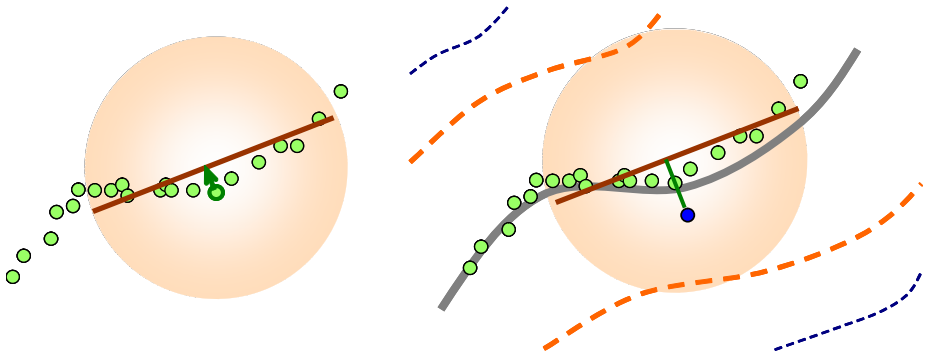
Local versus Global Fitting

At this stage, the problem specification is identical to several others in multi-view geometry. For instance, in the problem of estimating the 3×3 fundamental matrix F, the observed point pairs (x, y) and (x', y') are required to satisfy the epipolar constraint $\begin{bmatrix} x & y & 1 \end{bmatrix}^{\mathrm{T}} F \begin{bmatrix} x' & y' & 1 \end{bmatrix}$. By expanding the constraint in terms of the entries of F, its equivalent bilinear form may be obtained as

$$\mathbf{v}(x, y, x', y') = \begin{bmatrix} xx' & xy' & x & yx' & yy' & y & x' & y' & 1 \end{bmatrix}^{\mathrm{T}},$$

with model parameters θ representing the 9-element vector of F-matrix coefficients. Problems of this type have been studied by several researchers [55, 74, 108], often by the name of the non-linear errors-in-variables model.

However, there is one very important difference. In problems such as fundamental matrix estimation, all the observations satisfy a single *global* constraint equation as a natural consequence of epipolar geometry. However, in the domain of geometric fitting it is unrea-



(a) Denoising point clouds (b) Surface reconstruction

Figure 2.1: Point cloud processing tasks such as (a) denoising and (b) surface reconstruction can be treated as local geometric fitting problems, as illustrated here in a 2D example. For (a) noise removal, each point $\mathbf{x}_i$ is simply projected to a line (colored dark red) fit in its local neighborhood. For (b) surface reconstruction, a scalar field is constructed using the distance of each grid cell to its locally fit line. The zero-level curve (colored gray in (b)) correspond to the estimate of the curve that best fits the observed points.

sonable to expect that a single equation to capture the geometric complexity of an arbitrary scene. Conversely, such a model would potentially require far too many parameters, perhaps exceeding the number of available data points, thus making the fitting problem ill-posed. We comment on this difference further in Section 2.7.

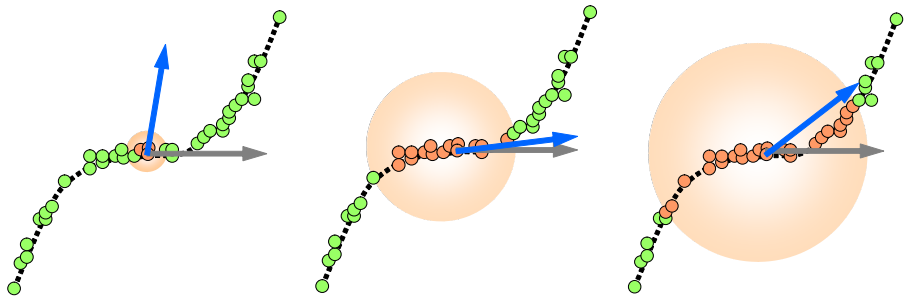
A more tractable approach is that of *local* geometric fitting, in which the chosen geometric model is assumed sufficient to explain a subset of observed points in a small neighborhood $\mathcal{N}(\mathbf{x}_i, r_i)$ of each point $\mathbf{x}_i$ where the r_i is the radius defining the neighborhood. Thus the implicit equation models the surface *locally* at *each* point $\mathbf{x}_i$ as

$$f(\mathbf{x}_j^o, \theta_i) = 0 \qquad \text{where} \quad \mathbf{x}_j \in \mathcal{N}(\mathbf{x}_i, r_i) \tag{2.4a}$$

$$\text{and} \qquad \mathbf{x}_j = \mathbf{x}_j^o + \eta_j \qquad \text{where} \quad \eta_j \sim N(0, \Lambda_j). \tag{2.4b}$$

Note that j indexes points in the neighborhood $\mathcal{N}(\mathbf{x}_i, r_i)$, and that there is now one model equation for *each* point $\mathbf{x}_i$ that is satisfied only by points in its neighborhood $\mathcal{N}(\mathbf{x}_i, r_i)$. The subscript i in θ_i emphasizes that the model parameters may be different at each point $\mathbf{x}_i$.

Several useful signal processing tasks may be handled in the above framework of local geometric fitting (see Figure 2.1). The task of **denoising** a point cloud may be accomplished by simply projecting each observed point $\mathbf{x}_i$ to the zero-level set of its locally fit surface $f(\mathbf{x}, \theta_i)$. The task of **surface reconstruction** can be done in a two-step process. First, divide the input space into a regular grid and perform the geometric fitting procedure in a neighborhood centered around each grid point. The fitted function evaluated at the grid



(a) $r \ll r_{opt}$ (b) $r = r_{opt}$ (c) $r \gg r_{opt}$

Figure 2.2: Effect of varying neighborhood radius (r) considered for computing tangent at a point on a sampled curve with respect to the optimal radius (r_{opt}). Estimated and true tangents are shown by the blue and grey arrows respectively

location gives the unsigned distance of the point to the surface. Performing this at each grid point gives an unsigned distance map, whose zero-level isosurface may be extracted using, say, a marching-cubes algorithm, to give the underlying surface.

Given the functional form of f, the remaining problems to be addressed are: (A) **scale selection**, or how to choose a neighborhood $\mathcal{N}(\mathbf{x}_i, r_i)$ at each point, and (B) **parameter estimation** how to compute the model parameters θ_i for that neighborhood. Traditionally these two problems have been addressed separately in the literature.

The first problem of scale selection is usually addressed by arbitrarily fixing the neighborhood size or the value of k in k-nearest neighborhood at each point[48, 67, 127], perhaps guided by some knowledge of the extent of the scene. This however, can have undesirable consequences as illustrated in Figure 2.2 for the case of tangent estimation (or equivalently, local line fitting) in a 2D curve. Using too small a radius can compromise the quality of the estimate due to the use of smaller number of noisy data points, while using too large a radius can permit a potentially dissimilar points in the neighborhood to adversely influence the estimate. Thus, it is crucial to make a choice of scale in model fitting that reflects the underlying geometry.

We will show later how the above two problems of model parameter estimation and scale selection may be solved *jointly*. For now we shall focus on getting good answers to the parameter estimation problem.

Reduction to statistical inference

In this section, we compare and draw connections between our problem of geometric fitting and the regression problem from classical statistics. This is done for two reasons - first, by mapping our problem into another that is well-studied, we hope to leverage solutions for the latter to be able to compute the model parameters θ_i at each point. Secondly, because our

revised problem formulation involves computing statistics in local neighborhoods having potentially a small number of points, we require a way of evaluating the accuracy of our solution within the established framework of statistical inference.

In what follows, we will drop the subscript i on the θ_i and r_i terms with the understanding that these two parameters depend on a neighborhood around $\mathbf{x}_i$ and that this point of interest is fixed.

A comparison of the geometric fitting equations (2.4) and the standard regression equation of the form

$$y_i = \beta x_i + \eta_i, \tag{2.5}$$

with model parameter β reveals some crucial differences. First, the model parameters θ in the geometric fitting problem satisfy an implicit equation through $f(\mathbf{x}, \theta) = 0$, whereas the relationship is explicit in the regression problem. Second, unlike in standard regression, there is no distinction into abscissa and ordinate variables in the fitting problem. Finally, the observation errors occur in all variables unlike in regression where the observation errors are assumed to exist only in the ordinate.

The first step in converting the geometric fitting problem into the classical regression framework is to convert the implicit equation (2.4a) into an *explicit* parametric equation. This can be done through the introduction of local coordinate system as

$$\mathbf{x}_j^o \equiv \mathbf{x}_j^o(\mathbf{s}_j, \theta), \tag{2.6}$$

where $\mathbf{s}_j$ denote unknown intrinsic coordinates on the surface $\mathcal{M}$. Note that the $\mathbf{s}_j$'s have the (lower) dimensionality of the manifold while the $\mathbf{x}_j$'s have the dimensionality of the input space.

The effect of introducing manifold coordinates $\mathbf{s}_j$ is to convert the system of equations (2.4) into the explicit system

$$\mathbf{x}_j = \mathbf{x}_j^o(\mathbf{s}_j, \theta) + \eta_j. \tag{2.7}$$

The equation (2.7) now has the same functional form as the standard regression problem (2.5) but with the addition of nuisance parameters $\mathbf{s}_j$. Thus, the unknowns in the system are $\{\theta, \mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n\}$ which exceed the number of equations.

Does it matter that the system of equations is under-determined? After all, techniques such as Expectation-Maximization (EM) are routinely used to solve systems through the introduction of nuisance parameters. As shall be shown, the problem arises not in solving the above equations, but instead in evaluating the *accuracy* of the obtained solution.

A well-accepted paradigm to evaluate the accuracy of an estimator $\bar{\theta}_n$ of θ from n observations is through its asymptotic efficiency. One way of evaluating this is through the

question: does $\bar{\theta}_n \to \theta$ as $n \to \infty$? This paradigm of asymptotic behavior is a concept central to classical statistical reasoning. Can this paradigm be applied to a solution to the geometric fitting problem?

Unfortunately, classical asymptotic analysis cannot be directly applied to the geometric fitting problem, at least in the form that we have presented so far, for two reasons. The first reason, as also pointed out by Kanatani [55], is that classical asymptotic analysis is applicable to systems having a fixed number of unknown parameters, whereas the number of unknowns in the geometric fitting problem increases with the number of observations. Another way to interpret this effect is that each additional observation $\mathbf{x}_{n+1}$ is not one more observation of a system with a fixed number of parameters, but an observation of new system with one more parameter $\mathbf{s}_{n+1}$. Thus the number of effective observations of the system is always one.

The second reason is that of insufficiency. A claim of asymptotic behavior as $n \to \infty$ does not necessarily translate to good results for finite n. In reality, we work with finite samples and expect n to be quite small for each neighborhood. Thus, in addition to asymptotic convergence, a bound on *how wrong* the estimate could be for finite n would be more useful in real application.

The next section will present a technique that enjoys the benefits of both, the ability to perform asymptotic analysis as well as to analyze finite-sample behavior, and in doing so will simultaneously address the problem of choosing the support region size.

Locally semi-parametric analysis

From the last section, we saw that a key hindrance to performing asymptotic analysis was the introduction of nuisance parameters $\{\mathbf{s}_i\}$. One way to circumvent this obstacle is to assume a parametric form for the distribution that generates the samples $\{\mathbf{s}_i\}$. i.e. Assume that the random variable T that generates samples $\{\mathbf{s}_i\}$ follows $T \sim \text{pdf}(\nu)$ where ν denotes the unknown hyperparameters of the chosen form of distribution. The net effect of this model is to replace the set of unknowns $\{\theta, \mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n\}$ by $\{\theta, \nu\}$. Because this fixes the number of unknowns, the standard asymptotic analysis may be performed without difficulty.

Now in general, the imposition of a distribution on samples $\{\mathbf{s}_i\}$ is a bad idea because the distribution over the unknown manifold $\mathcal{M}$ need not take a simple form. Indeed, for the same reasons cited earlier for local fitting, the form of the distribution may require too many parameters making the problem of determining the hyperparameters difficult.

However, in a *small* local neighborhood, the distribution will appear nearly uniform.

Therefore one appealing form of the distribution is

$$T \sim \text{Uniform}(r), \tag{2.8}$$

where r is the radius of the neighborhood under consideration. Note that the fixed set of parameters in the system are now θ and r, and that radius r has been introduced as a hyperparameter.

The impact of introducing radius r is that we are now able to map the problem into the standard statistical framework of regression while incorporating knowledge of the locality of the fitting problem. Thus, any error bounds we may obtain for a given estimator will involve the free variable r which may be optimized to improve the accuracy of the solution. Since the salient feature of this method is the combination of a local parametric model for the point sample distribution with a different model at each sample point, we refer to it as a **locally semi-parametric** approach.

To contrast this approach with related work by Kanatani [55], we wish to point out that his analysis of geometric fitting demonstrated the inadequacy of classical statistics using arguments similair to those in the previous section. However, the analysis in [55] deviates from ours in that its focus is on global fitting and hence the problem of choosing an appropriate neighborhood size does not arise. Its proposed analysis of estimator convergence rate as noise $\eta_i \to 0$ exhibits a dual nature to classical asymptotic analysis, but by itself is also asymptotic in nature. In contrast, the locally semi-parametric approach benefits from the simple forms of distributions locally to allow computation of 2^{nd} order statistics and associated finite-sample error bounds, as well as compute asymptotic values to check for any possible bias of the estimator.

The next two sections will demonstrate the application of the above approach to the problems of reconstructing 2D and 3D curves, and extend the results to reconstructing 2D surfaces.

2.3 Reconstruction of Curves

In this section, we illustrate the method of locally semi-parametric analysis through the task of reconstructing curves from sample points. Evaluating the accuracy of a reconstruction algorithm by comparing two curves is not straightforward. To make this evaluation more easily quantifiable, we define the objective to be that of estimating the tangent to the curve at each observed point, which is much easier to compare and quantify with other algorithms.

We exploit the property of local linearity in the curve through local principal component analysis (PCA) using an *adaptive* neighborhood size. Our estimate of the tangent at a point is the principal eigenvector of the scatter matrix computed in its local neighborhood. We

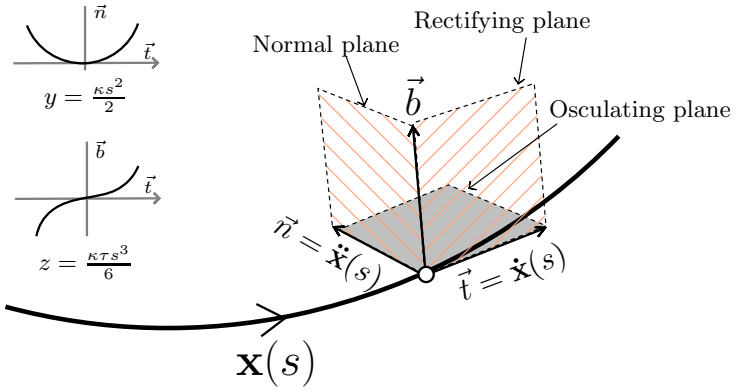


Figure 2.3: Model of local curve geometry

choose this estimator as it is simple in form and has been used by other researchers for related tasks [58, 77, 104].

We propose that, for spatial curves, the neighborhood size should be chosen such that the principal eigenvalue of the scatter matrix is most closely aligned with the true tangent to the curve. To make this choice, we derive an upper bound on the expected angular error induced by finite sampling and sample noise as a function of neighborhood radius. The optimal radius is then chosen as the value that minimizes this upper bound on angular error. The ability to bound the accuracy of the estimate for a given neighborhood radius is the contribution of the locally semi-parametric approach, which we detail below.

Curve model

Our available data is a set of n unordered points $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ in a local neighborhood. Each such point $\mathbf{x}_i = \{x_i, y_i, z_i\}$ may be thought of as a noisy observation of a true point lying on a smooth curve Γ at an (unknown) distance s_i along the curve.

Without loss of generality, we assume a Frenet reference frame (Figure 2.3) with origin located at the point of interest such that the tangent to the curve is aligned with the x-axis, the curvature vector in the plane of the osculating circle containing the point of interest is aligned with the y-axis and the normal to the osculating plane is aligned with the z-axis. The neighborhood considered around the origin is defined as all points lying within the distance r along the curve from the origin. We may then adopt the semi-parametric generative model with the samples s_i assumed to be generated from a uniform distribution

$S \sim \text{Uniform}(-r, r)$ with additive Gaussian noise $\eta \sim N(0, \sigma_0^2)$ as:

$$\begin{aligned} x_i &= s_i + \eta_{x,i} \\ y_i &= \frac{\kappa}{2} s_i^2 + \eta_{y,i} \\ z_i &= \frac{\kappa\tau}{6} s_i^3 + \eta_{z,i}, \end{aligned} \tag{2.9}$$

which is valid for slowly changing values of curvature (κ) and torsion (τ). i.e. in a local neighborhood around the point of interest, curvature κ and torsion τ are assumed bounded and near constant, i.e. $\dot{\kappa}(s), \dot{\tau}(s) \approx 0$. We also assume i.i.d. sensor noise that is zero-mean normally distributed with variance σ_0^2 affecting all three coordinates. In practice, this allows the value of σ_0 to differ across the scene to account for variation in noise level with distance from the laser sensor.

The covariance matrix for curves

One technique to estimate the direction of the local tangent at a given sample point on a curve is to look at the shape of a scatter matrix computed using points in its neighborhood [58, 77, 104]. If the curve is smooth, it is reasonable to expect that the scatter matrix will be elongated and that its major axis, or principal eigenvector, will approximate the direction of the local tangent for some appropriate (and unknown) range of neighborhood sizes. In this and the following subsection, we will derive and analyze the conditions under which this assumption will hold for both 2D and 3D curves.

The random variables X, Y and Z (denoted in capitals to distinguish them from the data) are functions of the random variable S, whose distribution is assumed to be locally uniform, and of the observation noise. Hence the distribution of X, Y and Z, as well as estimators of their 1^{st} and 2^{nd} order statistics will depend on the coefficients (κ, τ) and order of the functions (given in (2.9)) as well as properties of the uniform (for S) and Gaussian (for η) distributions.

We start by computing the mean and variance of the estimators used to construct the sample covariance matrix $\hat{M}_n$. We will denote the true means of random variables by μ (e.g. μ_X for X) and standard deviation by σ (e.g. σ_X^2 for variance of X). Then

$$\hat{M}_n = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{12} & M_{22} & M_{23} \\ M_{13} & M_{23} & M_{33} \end{bmatrix}, \tag{2.10}$$

where

$$M_{11}=\frac{\sum_i(x_i-\bar{X}_n)^2}{n-1} \qquad M_{12}=\frac{\sum_i(x_i-\bar{X}_n)(y_i-\bar{Y}_n)}{n-1} \tag{2.11}$$

$$M_{22}=\frac{\sum_i(y_i-\bar{Y}_n)^2}{n-1} \qquad M_{13}=\frac{\sum_i(x_i-\bar{X}_n)(z_i-\bar{Z}_n)}{n-1} \tag{2.12}$$

$$M_{33}=\frac{\sum_i(z_i-\bar{Z}_n)^2}{n-1} \qquad M_{23}=\frac{\sum_i(y_i-\bar{Y}_n)(z_i-\bar{Z}_n)}{n-1} \tag{2.13}$$

and $\bar{X}_n = \frac{1}{n}\sum_i x_i$ is the sample mean estimator for X, and similarly for $\bar{Y}_n$ and $\bar{Z}_n$.

Note that, as is well known from introductory statistics [118], the diagonal elements are unbiased estimators for variance (e.g. M_{11} is the estimator for variance σ_X^2 of X) and the off-diagonal elements are unbiased estimators of covariance (e.g. M_{13} is the estimator for covariance $\mathrm{cov}(X,Z)$ of X and Z).

From the distribution of $S \sim \mathrm{Uniform}(-r,r)$, we can then compute the expected values of each of the above quantities.

For example, using $X = S + \eta_X$

$$\begin{aligned}\mathbb{E}(M_{11}) &= \mathbb{V}(X_i) = \mathbb{V}(S+\eta) = \mathbb{V}(S_i) + \sigma_0^2 \\ &= \sigma_X^2 + \sigma_0^2 = \int_{-r}^{r} s^2 \frac{1}{2r} ds + \sigma_0^2 = \frac{r^2}{3} + \sigma_0^2.\end{aligned} \tag{2.14}$$

Using a similar procedure, we can derive the following identities under the model defined in (2.9).

$$\mathbb{E}(M_{12}) = \mathrm{cov}(X,Y) = \frac{\kappa}{2}\mathbb{E}(S^3) = 0 \tag{2.15}$$

$$\mathbb{E}(M_{13}) = \mathrm{cov}(X,Z) = \frac{\kappa\tau}{6}\mathbb{E}(S^4) = \frac{\kappa\tau}{30}r^4 \tag{2.16}$$

$$\mathbb{E}(M_{22}) = \mathbb{V}(Y) = \frac{\kappa^2}{4}\mathbb{V}(S^2) + \sigma_0^2 = \frac{\kappa^2}{45}r^4 + \sigma_0^2 \tag{2.17}$$

$$\mathbb{E}(M_{23}) = \mathrm{cov}(Y,Z) = \frac{\kappa^2\tau}{18}(\mathbb{E}(S^5) - \mathbb{E}(S^2)\mathbb{E}(S^3)) = 0 \tag{2.18}$$

$$\mathbb{E}(M_{33}) = \mathbb{V}(Z) = \left(\frac{\kappa\tau}{6}\right)^2 \frac{r^6}{7} + \sigma_0^2. \tag{2.19}$$

To proceed from here, we must use results on the variance of the sample variance and sample covariance estimators. We state them below, and their proofs may be found in Appendix A.1.

Identity 1 (Variance of the *sample variance* estimator)**.**

$$\mathbb{V}(\hat{\sigma}_X^2) = \frac{d_4(X)}{n} - \frac{(n-3)}{n(n-1)}\sigma_X^4 \tag{2.20}$$

for a random variable X, where

$$d_m(X) \triangleq \mathbb{E}(X - \mu_X)^m. \tag{2.21}$$

Identity 2 (Variance of the *sample covariance* estimator)**.**

$$\mathbb{V}(\hat{S}_{XY}) = \frac{c_2(X,Y)}{n} + \frac{\sigma_X^2\sigma_Y^2}{n(n-1)} - \frac{(n-2)}{n(n-1)}c_1^2(X,Y) \tag{2.22}$$

for random variables X and Y, where

$$c_m(X,Y) \triangleq \mathbb{E}\left[(X - \mu_X)(Y - \mu_Y)\right]^m. \tag{2.23}$$

Note that we use the hat symbol ($\hat{}$) to distinguish the estimator from the true quantity.

Under the curve model defined in (2.9), we can derive the expressions for $d_4(X)$, $d_4(Y)$ and $d_4(Z)$ in a manner similar to that used for (2.15)–(2.19), using the identity

$$d_4(X + \eta) = d_4(X) + 6\sigma_0^2 d_2(X) + 3\sigma_0^4 \tag{2.24}$$

for any random variable X affected by normally distributed independent noise $\eta \sim N(0, \sigma_0^2)$. Note that the simplification is because the odd moments of η vanish and $\mathbb{E}(\eta^4) = 3\sigma_0^4$. We may also similarly derive the expressions for c_1 and c_2 for all pairs of X,Y and Z.

Once we have the required values for c_i and d_i, we can then substitute them back in (2.20) and (2.22) to get the variance of the individual estimators, which we denote as $\mathbb{V}(M_{ij})$ with $i, j = \{1, 2\}$. The final expressions for $\mathbb{V}(M_{ij})$ are listed in Appendix A.4.

Observe that the estimator for sample covariance matrix may be expressed as the sum of the matrix of its expected value and a matrix of random variables as

$$\hat{M}_n = \bar{M} + Q. \tag{2.25}$$

Here $\bar{M} = \mathbb{E}(M)$ is a symmetric matrix with elements given by (2.15)–(2.19), and Q is a symmetric *perturbation matrix* of random variables each with mean 0 and variance given by the expressions listed in Appendix A.4.

Perturbation model

In the previous section, we were able to express the scatter matrix ($\hat{M}_n$) computed in a local neighborhood as a sum of an uncorrupted intrinsic quantity ($\bar{M}$) and a random matrix (Q) existing due to finite sampling and noise. In this section we compute the effect of the perturbation Q on the principal eigenvector of $\hat{M}_n$.

We denote the eigenvalues of $\bar{M} = \mathbb{E}(M)$ by $\lambda_1 \geq \lambda_2 \geq \lambda_3$. Let the eigenvector corresponding to λ_1 be $\mathbf{e}_1$. Let $\hat{e}_1$ be the eigenvector corresponding to the largest eigenvalue of the estimated $\hat{M}_n$. If Q is the symmetric perturbation to the positive semidefinite matrix $\bar{M}$, then the application of the matrix perturbation theorem V.3.4 from [100] can be shown to yield

$$\sin\left(\angle(\hat{\mathbf{e}}_1, \mathbf{e}_1)\right) \leq \frac{||Q||_F}{\delta}, \tag{2.26}$$

where $\angle(\hat{\mathbf{e}}_1, \mathbf{e}_1)$ denotes the angle between the estimated $\hat{\mathbf{e}}_1$ and the true $\mathbf{e}_1$. The quantity $\delta = \lambda_1 - \lambda_2$ is the *spectral gap* of the matrix $\mathbb{E}(M)$. $||Q||_F$ represents the Frobenius norm of matrix Q.[1] We refer the reader to Appendix B.1 for the proof of the above statement.

Since the matrix Q consists of random variables, we are confined to making probabilistic statements about $||Q||_F$. Using Chebyshev's inequality, the square of the value attained by each element Q_{ij} can be upper bounded by

$$Q_{ij}^2 \leq \frac{\mathbb{V}(M_{ij})}{n\epsilon}$$

with probability $1-\epsilon$, where $\mathbb{V}(M_{ij})$ is the variance of corresponding finite-sample estimator of covariance (or variance if $i = j$).

Thus, in order to minimize the error between the estimated eigenvector $\hat{\mathbf{e}}_1$ and the true eigenvector $\mathbf{e}_1$, it is necessary to minimizing the RHS of (2.26), which we denote as

$$B \triangleq ||Q||_F/\delta. \tag{2.27}$$

In the remainder is this section, we will analyze the function B for both 2D and 3D curves.

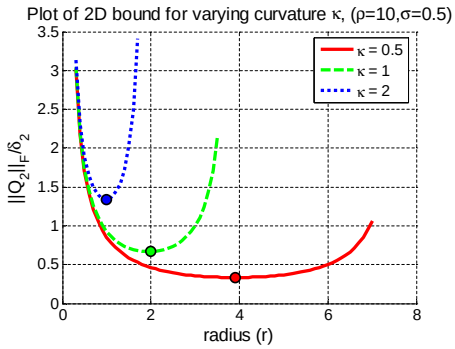
Angular bounds and their behavior

We first analyze the behavior of the perturbation bound to variation in sampling density, noise and curvature by looking at the slightly simpler case of 2D curves.

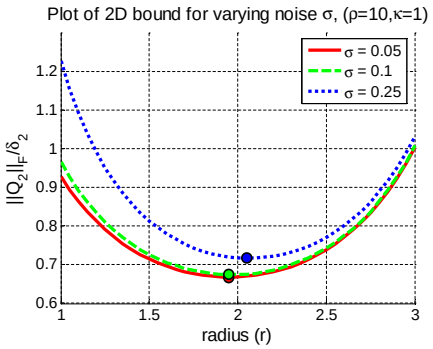
2D curves

We analyze the 2D case by working with the same assumptions as stated earlier except that we discard the z-coordinate (or equivalently nullify torsion). The scatter matrix in this case is obtained as the top left 2×2 sub-matrix of Q, which we will refer to as Q_2. From our perturbation model in Section 2.3, we know that the Frobenius norm of Q_2 is upper

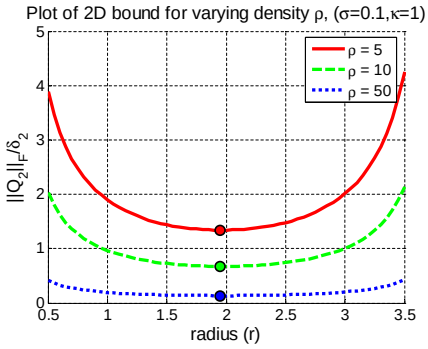
[1] A similar result was used in [82] where the authors invoked theorem V.2.8 from [100] to bound $||\hat{\mathbf{e}}_1 - \mathbf{e}_1||$ and analyzed the stability of document-link matrices constructed for ranking web pages.



(a) Varying curvature (κ)



(b) Varying sampling noise (σ)



(c) Varying sample density (ρ)

Figure 2.4: Plot of analytic 2D bound for varying sampling and geometry parameters

bounded with probability $1-\epsilon$ by

$$\begin{aligned} ||Q_2||_F^2 &\leq \frac{1}{n\epsilon}\sum_{i=1}^{2}\sum_{j=1}^{2}\mathbb{V}(M_{ij}) \\ &= \frac{1}{n\epsilon}\left[\mathbb{V}(M_{11})+\mathbb{V}(M_{22})+2\mathbb{V}(M_{12})\right]. \end{aligned} \tag{2.28}$$

The spectral gap δ_2 of the corresponding top-left 2×2 sub-matrix $\bar{M}_2$ of $\mathbb{E}(M)$ given by

$$\bar{M}_2 = \begin{bmatrix} \frac{r^2}{3}+\sigma_0^2 & 0 \\ 0 & \frac{\kappa^2}{45}r^4+\sigma_0^2 \end{bmatrix}. \tag{2.29}$$

is obtained easily by inspection as

$$\delta_2 = \frac{r^2}{3} - \frac{\kappa^2 r^4}{45}. \tag{2.30}$$

This implies that for the dominant eigenvector of $\bar{M}$ to be the vector $\begin{bmatrix}1 & 0\end{bmatrix}$, the value of

radius r must satisfy

$$0 < r < \sqrt{15}/\kappa. \tag{2.31}$$

The bound to be minimized then is

$$B_2(r) \triangleq \frac{||Q_2||_F}{\delta_2}. \tag{2.32}$$

To study the analytical behavior of this bound, we need to replace the discrete parameter n by a continuous function of radius r, and explicitly express their dependency. To do this, we use the assumption of minimum local point density ρ and substitute $n = 2\rho r$ to form the analytical plots that follow.

Note, however, that in the implementation of the proposed algorithm we directly set n in (2.32) to equal the number of points observed in the neighborhood of candidate radius r and do not ever need to estimate ρ. The assumption of an underlying ρ is used *only* for studying the expected behavior of the analytic bound in synthetic data and is *not* used at runtime.

Before proceeding, we point out that there are two expected limitations in the functional analysis of the derived expressions that will be relevant in their experimental validation. Firstly, although the bounds are discontinuous functions of high order polynomials in r, our analysis is restricted to the regime where the constraints (2.31) required for eigenvector dominance are satisfied. In this regime, the bound is convex with a unique minimum.

Secondly, and as also observed experimentally in [67, 81], the predicted error tends to 0 as $r \to 0$ for noise-free data. But for $\sigma_0 > 0$, the error tends to sharply increase for the same condition. This behavior is not reflected in our model as our continuous relaxation of n as $2\rho r$ is invalid for small r. Hence, we advocate the interpretation of the function only in terms of the behavior of its minima in the meaningful regimes of interest, rather than throughout the domain of the function.

Based on the analytical plots of $B_2(r)$ in Figures 2.4(a)–2.4(c), we make the following qualitative observations:

1. Complexity: The closed-form expression in (2.32) unfortunately does not have a simple form. However, it can be easily shown that the terms with coefficients $(n(n-1))^{-1}$ in the numerator of $B_2(r)$ are dominated by the others for integer values of $n \geq 2$, reducing the expression to the ratio of the root of a 6th degree polynomial and a 4th degree polynomial of r, both only containing even powers of r.

2. Variation with curvature κ: Figure 2.4(a) plots the function B_2 for multiple values of κ and fixed values of noise and sampling density. As one would expect, the optimal radius r tends to increase with decreasing curvature in order to compensate for noise

and sparsity, without exceeding the bounds in (2.31) when the eigenvector more closely aligned to the x-axis is no longer dominant.

3. Variation with sampling noise σ_0 : Figure 2.4(b) plots the function B_2 for multiple values of κ and fixed values of noise and sampling density. It can be seen that as the noise increases, the point of minima of B_2 increases but only approaching the required bounds for eigenvector dominance in (2.31).

4. Variation with sampling density ρ : Figure 2.4(c) plots the function B_2 for multiple values of sampling density and fixed values of noise and curvature. It is interesting to note that although the value of the bound decreases as expected with increased number of points, the location of the extremum hardly changes. This is in contrast with the observations in [81] for surfaces which varies r with $\rho^{-0.5}$. We validate this later in Section 2.5.

3D curves

The derivation and behavior of the angular bound for 3D curves is fairly similar to the 2D case. From Section 2.3, $||Q||_F$ is upper bounded with probability $1-\epsilon$ by

$$\begin{aligned} ||Q||_F^2 &\leq \frac{1}{n\epsilon}\sum_{i=1}^{3}\sum_{j=1}^{3}\mathbb{V}(M_{ij}) \\ &= \frac{1}{n\epsilon}\Big[\mathbb{V}(M_{11})+\mathbb{V}(M_{22})+\mathbb{V}(M_{33})+2\left(\mathbb{V}(M_{12})+\mathbb{V}(M_{13})+\mathbb{V}(M_{23})\right)\Big]. \end{aligned} \tag{2.33}$$

Substituting the results from Section 2.3 gives the required final expression (see Appendix A.4).

The matrix of expected values can be written as:

$$\bar{M} = \mathbb{E}(M) = \begin{bmatrix} \frac{r^2}{3}+\sigma_0^2 & 0 & \frac{\kappa\tau}{30}r^4 \\ 0 & \frac{\kappa^2}{45}r^4+\sigma_0^2 & 0 \\ \frac{\kappa\tau}{30}r^4 & 0 & \left(\frac{\kappa\tau}{6}\right)^2\frac{r^6}{7}+\sigma_0^2 \end{bmatrix}. \tag{2.34}$$

We denote the eigenvalues of $\bar{M}$ as $\lambda_1 \geq \lambda_2 \geq \lambda_3$. The spectral gap of $\bar{M}$ is not as straightforward due to its off-diagonal terms. However, we can lower bound the spectral gap using the following theorem by Gershgorin [28].

Theorem 1 (Gershgorin circle theorem)**.** *For an $n \times n$ matrix A with entries a_{ij}, define*

$$R_i = \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}|$$

Then each eigenvalue of A is in at least one of the discs

$$\{\lambda : |\lambda - a_{ii}| \leq R_i\}$$

A straightforward application of the Gershgorin circle theorem (GCT) [28] gives the system of inequalities

$$|\lambda_1 - \frac{r^2}{3} + \sigma_0^2| \leq \frac{\kappa\tau}{30} r^4 \tag{2.35}$$

$$\lambda_2 = \frac{\kappa^2}{45} r^4 + \sigma_0^2 \tag{2.36}$$

$$|\lambda_3 - \left(\frac{\kappa\tau}{6}\right)^2 \frac{r^6}{7} + \sigma_0^2| \leq \frac{\kappa\tau}{30} r^4. \tag{2.37}$$

Under the conditions of (2.31), this gives the additional constraint on radius as

$$r \leq \sqrt{28/5\tau}, \tag{2.38}$$

and a bound on spectral gap as:

$$\delta_3 \geq \frac{r^2}{3} - \frac{\kappa^2 r^4}{45} - \kappa\tau\frac{r^4}{15}, \tag{2.39}$$

Combining (2.33) and (2.39) with the continuous relaxation $n = 2\rho r$ in (2.27) gives the desired result.

The observations we make on the analytic behavior of $B(r)$ are analogous to those in the 2D case. The main effect of torsion is that due to its presence as an off-diagonal term in $\mathbb{E}(M)$, it always induces a finite angular offset of the dominant eigenvector in the rectifying plane (see Figure 2.3).

However as the radius is decreased, the off-diagonal term tends to 0 with r^4 while the leading eigenvector decays with r^2. Thus in moving from the 2D to 3D analysis, the overall effect of torsion is to decrease the optimal scale of analysis with increasing τ. This shift can be verified in Figure 2.5 which has the same parameters as the 2D curve of Figure 2.4(a) but with a non-zero torsion $\tau = 0.3$.

To summarize this section, we have shown how to relate the error in estimating the shape (specifically the tangent) of a curve locally from few observations to both its geometry related parameters, namely the curvature, torsion and sampling noise, as well as to the choice of support radius size. What this allows us to do in practice is to vary the free parameter of support radius size so that we get the best possible estimate of the model parameters. Furthermore, because an upper bound of the allowable search radius can be computed from the data, the search for the optimal radius that minimizes the error bound

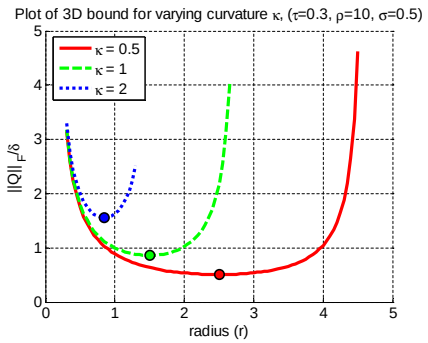


Figure 2.5: Plot of analytic 3D bound for varying curvature

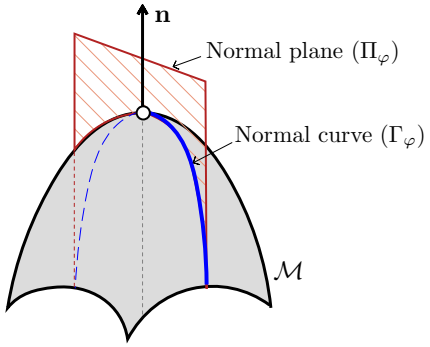


Figure 2.6: Model of local surface geometry

is easy to perform. From the plots showing the variation of the error bound to changes in geometry parameters, it can also be seen that the strategy of choosing a radius that minimizes the error bound has the behavior of automatically adapting the locality of the geometric fitting procedure to the unknown underlying shape.

In the next section, we will extend the above ideas from curves to the analysis of surfaces in 3D. We will then conclude this chapter with results validating the derived error bounds, showing the improved stability of the resulting estimators, and finish with an application for improving shape descriptors.

2.4 From Curves to Surfaces

We can now extend the analysis of the previous section from 2D curves to 3D surfaces by following elementary concepts from differential geometry.

Adhering to the notation from the previous section, our available data is a set of n unordered points $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, where each point $\mathbf{x}_i = \{x_i, y_i, z_i\}$ is now considered as a noisy observation of a point on a 3D surface $\mathcal{M}$. Without loss of generality, we assume a Darboux reference frame with origin located at the point of interest such that the surface normal $\mathbf{n}$ is aligned with the positive z-axis, and the principal curvatures κ_1 and κ_2 are aligned with the positive x-axis and y-axis respectively.

There exists a family Π of planes that contain the origin and its normal vector. Each such plane Π_φ, lying at an angle φ to some reference vector in the tangent plane, intersects the surface $\mathcal{M}$ at a curve Γ_φ termed the *normal curve*. (See Figure 2.6.)

The neighborhood considered around the origin on the surface is defined as all points lying within the distance r along *any* normal curve Γ_φ from the origin. For each curve Γ_φ, we may then adopt the semi-parametric generative model with the samples s_i assumed to

be generated from a uniform distribution $S \sim \text{Uniform}(-r, r)$ with additive Gaussian noise $\eta \sim N(0, \sigma_0^2)$ as

$$\begin{aligned} x_i &= s_i \cos(\varphi) + \eta_{x,i} \\ y_i &= s_i \sin(\varphi) + \eta_{y,i} \\ z_i &= \frac{\kappa}{2} s_i^2 + \eta_{z,i}, \end{aligned} \tag{2.40}$$

where the sectional curvature [32, 121] is given by

$$\kappa = \kappa_1 \cos^2(\varphi) + \kappa_2 \sin^2(\varphi). \tag{2.41}$$

As before, sensor noise is assumed i.i.d. and zero-mean normally distributed with variance σ_0^2 affecting all three coordinates.

Following this generative model, each sampled point is assumed to be generated by randomly picking an angle $\varphi \in [0, \pi]$ and then picking a point from its normal curve Γ_φ at distance $s \in [-r, r]$ following a uniform distribution in that interval.

Thus, the expected value of any function $g(X, Y, Z)$ of the associated random variables in a neighborhood $\mathcal{N}(0, r)$ under the above generative model may be obtained by integrating over each normal curve over all angles $\varphi \in [0, \pi]$ as

$$\mathbb{E}(g(X, Y, Z)) = \int_{-\infty}^{\infty} \int_{-r}^{r} \int_{0}^{2\pi} \frac{1}{2r} \frac{1}{\pi} g(X, Y, Z) p(\eta) d\varphi \; ds \; d\eta, \tag{2.42}$$

where $p(\eta)$ denotes the Gaussian distribution on the error variables.

The estimator for surface normal may be chosen in a similar manner as for tangents to curves, as the eigenvector corresponding to the minimum eigenvalue of the 3×3 scatter matrix $\hat{M}_n$ computed as per the expression (2.10) with terms given by (2.11) through (2.13).

The locally semi-parametric analysis of the defined surface normal estimator then proceeds similarly to the case of curves in the previous section. First, we decompose the scatter matrix estimator into two components – an expected matrix component $\bar{M} = \mathbb{E}(\hat{M}_n)$ and a perturbation matrix component Q that vanishes for infinite number of data points. The

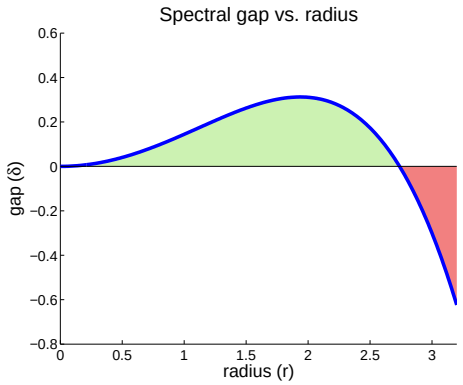


Figure 2.7: Plot of spectral gap of the scatter matrix associated with points sampled on a quadric surface with $\kappa_1 = \kappa_2 = 1$. The region of the curve lying above x-axis (shaded green) marks the radius interval where the minimal eigenvector is aligned with the true surface normal in the limit. When the spectral gap changes sign beyond a critical value of radius (about 2.74 in the above scenario), the minimal eigenvector is no longer a suitable estimator for surface normal.

expected matrix may be computed analytically using (2.40) and (2.42) as

$$\begin{aligned}\bar{M} = \mathbb{E}(\hat{M}_n) &= \begin{bmatrix} d_2(X) & c_2(X,Y) & c_2(X,Z) \\ c_2(X,Y) & d_2(Y) & c_2(Y,Z) \\ c_2(X,Z) & c_2(Y,Z) & d_2(Z) \end{bmatrix} \\ &= \begin{bmatrix} \frac{r^2}{6} + \sigma^2 & 0 & 0 \\ 0 & \frac{r^2}{6} + \sigma^2 & 0 \\ 0 & 0 & \frac{r^4\left(17\kappa 1^2 - 2\kappa 1\kappa 2 + 17\kappa 2^2\right)}{1440} + \sigma^2 \end{bmatrix}.\end{aligned} \tag{2.43}$$

Recall that for the case of curves, our chosen estimate (of the curve tangent) at each point was the dominant eigenvector of the scatter matrix. Since the estimate for surface normals is the *minimal* eigenvector as opposed to the maximal eigenvector, the corresponding spectral gap of interest is the difference between the smallest and second smallest eigenvalue.

From the previous equation, since $\bar{M}$ is a diagonal matrix, the spectral gap δ can be computed simply by inspection as

$$\delta = \frac{r^2}{6} - \frac{r^4\left(17\kappa_1^2 - 2\kappa_1\kappa_2 + 17\kappa_2^2\right)}{1440}. \tag{2.44}$$

Thus, for the minimal eigenvector of the computed scatter matrix $\hat{M}_n$ to be aligned

with the true surface normal, the condition $\delta > 0$ must be satisfied. This translates to

$$0 < r < \sqrt{\frac{240}{17\kappa_1^2 - 2\kappa_1\kappa_2 + 17\kappa_2^2}} \tag{2.45}$$

when $\kappa_1, \kappa_2 \geq 0$. Figure 2.7 plots the variation in spectral gap for $\kappa_1, \kappa_2 = 1$. It may be seen that beyond a certain critical radius, the spectral gap changes sign. Hence this critical radius may be used to bound the search for the radius r that minimizes the estimation error derived below.

From Section 2.3 $||Q||_F$ is upper bounded with probability $1 - \epsilon$ by

$$\begin{aligned} ||Q||_F^2 &\leq \frac{1}{n\epsilon} \sum_{i=1}^{3} \sum_{j=1}^{3} \mathbb{V}(M_{ij}) \\ &= \frac{1}{n\epsilon} \Big[\mathbb{V}(M_{11}) + \mathbb{V}(M_{22}) + \mathbb{V}(M_{33}) + 2\left(\mathbb{V}(M_{12}) + \mathbb{V}(M_{13}) + \mathbb{V}(M_{23})\right) \Big], \end{aligned} \tag{2.46}$$

where the individual variance terms may be computed using the Identities 1 and 2 from Section 2.3.

Thus the error in the estimate of the surface normal $B(r)$ from the true normal is bounded by

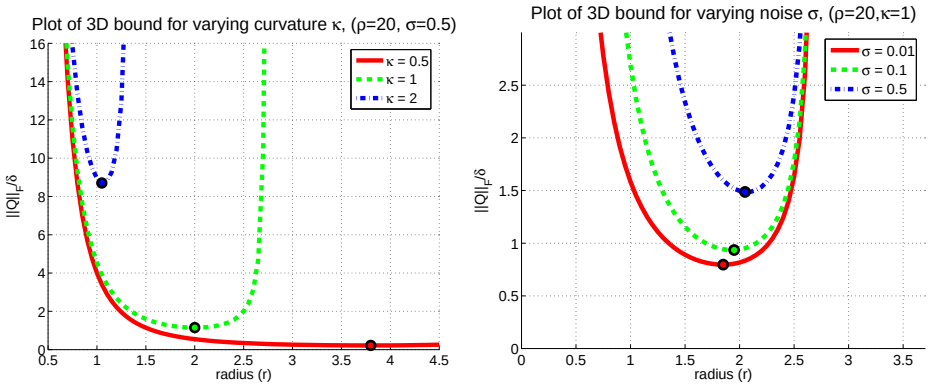
$$B(r) \triangleq \frac{||Q||_F}{\delta}. \tag{2.47}$$

From the plots of $B(r)$ in Figures 2.8(a)–2.8(c), we can make the following qualitative observations:

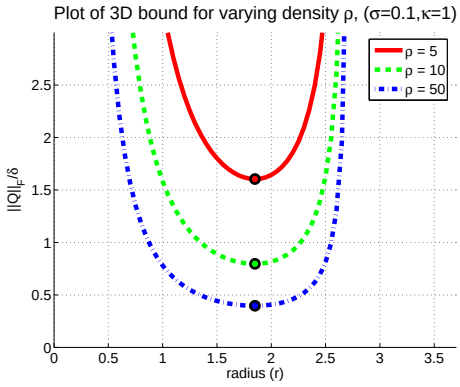
- Variation with curvature κ: Figure 2.8(a) plots the function $B(r)$ for multiple values of $\kappa_1 = \kappa_2 = \kappa$ and fixed values of noise and sampling density. As one would expect, the optimal radius r tends to increase with decreasing curvature in order to compensate for noise and sparsity upto the point where the eigenvector more closely aligned to the z-axis is no longer dominant.

- Variation with sampling noise σ_0 : Figure 2.8(b) plots the function $B(r)$ for multiple values of $\kappa_1 = \kappa_2 = \kappa$ and fixed values of noise and sampling density. As with the case of curves, with increasing noise, the point of minima of B increases but only approaching the required bounds for eigenvector dominance.

- Variation with sampling density ρ : Figure 2.8(c) plots the function $B(r)$ for multiple values of sampling density and fixed values of noise and curvature. As expected the value of the bound decreases with increased number of points, but the location of the extremum hardly changes.



(a) Varying curvature (κ) (b) Varying sampling noise (σ)



(c) Varying sample density (ρ)

Figure 2.8: Plots of analytic error bound for computing normals to surfaces of varying geometry parameters

2.5 Experiments

In this section we present experimental results to validate the theoretical behavior predicted by the models built in the previous section for curves and surfaces, and also to study the numerical accuracy and stability of the resulting algorithms. We start with an outline of the algorithm procedure and draw attention to a few implementation details below.

Algorithm and Implementation

Algorithm 1 details the procedure used to estimate tangents (normals) from points sampled from curves (surfaces). We describe the algorithm for the case of curves and draw attention to some implementation details below.

The algorithm is initialized with a starting value of radius $r^{(t)}$ for $t = 0$, that is chosen in our experiments as the distance to the k-th nearest neighbor for $k = 5$. For this starting neighborhood size $r^{(t)}$, we estimate the curvature ($\kappa^{(t)}$) and torsion ($\tau^{(t)}$) using [67] and

Algorithm 1 Estimation of tangents (normals) from unorganized points

Data: Points $X = \{\mathbf{x}_i\} \in \mathbb{R}^3$ with $i = 1 \dots n$.

1: Construct a graph $\mathbb{G}$ on the points from which approximate geodesic distances may be computed as graph path distances. Distance $d_{\mathbb{G}}(x_i, x_j)$ between any pair of points $\mathbf{x}_i, \mathbf{x}_j$ can be computed efficiently using Dijkstra's algorithm.
2: **for** $x \in \{\mathbf{x}_i\}$ **do**
3: Choose starting neighborhood radius $r_{t=0}$, say using distance to the k-th nearest neighbor for a small k.
4: **repeat**
5: Estimate curvature $\kappa_t(r)$ (and torsion $\tau_t(r)$) for points in geodesic radius $r = r_t$ from $\mathbf{x}_i$.
6: Perform a 1D line search to solve $r_{t+1} = \arg\min B(r_t, \kappa_t, \tau_t)$ subject to boundary conditions on r to enforce positive spectral gap $\delta > 0$.
7: **until** convergence of r_t, else $t = t + 1$
8: **end for**

use a sensor model to obtain the value of sample noise. Then we perform a 1D line-estimation on r to obtain the radius $r^{(t+1)}$ that minimizes the error bound of (2.32), subject to the constraint (2.31) using values computed for time t. We then re-estimate $\kappa^{(t+1)}$ and $\tau^{(t+1)}$ corresponding to the new value of radius r and iterate till convergence. To prevent large changes in estimates of r between iterations, we used a damping factor, although no significant difference in results was observed without it.

To estimate κ and τ for curves at each iteration, we use the procedure from [67] setting its scale parameter to the current estimate of r. Both the technique in [67] and our method for scale selection approximate distances between points along the underlying curve by a sum of edge distances in a graph constructed on the points. For the case of surfaces, we use the curvature estimation procedure suggested in [81].

One technical detail that we wish to point out is that, as a side-effect of choosing coordinate descent as our strategy for minimizing the error bound and of re-estimating the curve (or surface) parameters used to compute those bounds, the error curve being minimized is not guaranteed to be the same across successive iterations of the algorithm. In principle, this means that the above iterative procedure need not always converge. However, given that the error function is smooth across the radius parameter in the region of interest, it reasonable to expect convergence under the weak assumption of the error surface being smooth around the global minima across both the model parameters as well as the radius. We have not encountered a case where the above procedure does not converge in practice.

We chose to construct the graph as the sum of multiple edge-disjoint minimum spanning trees (DMST), suggested in [24] as an extension of minimal spanning trees (MST). This is done by first building the MST on the complete graph of the points, then removing

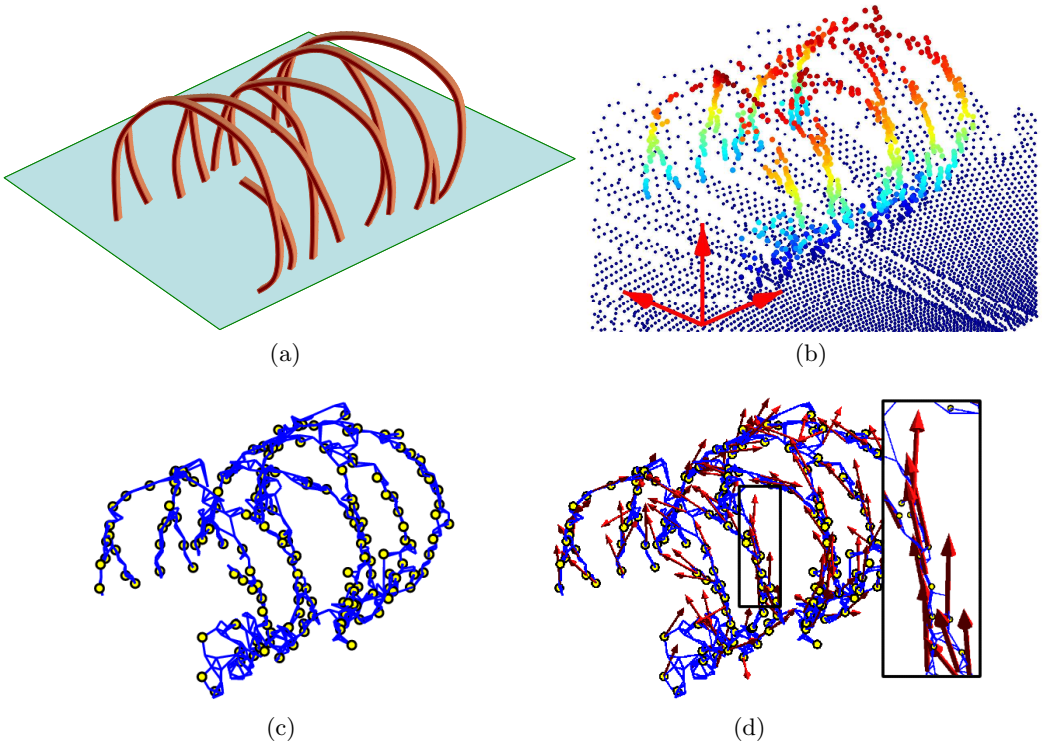
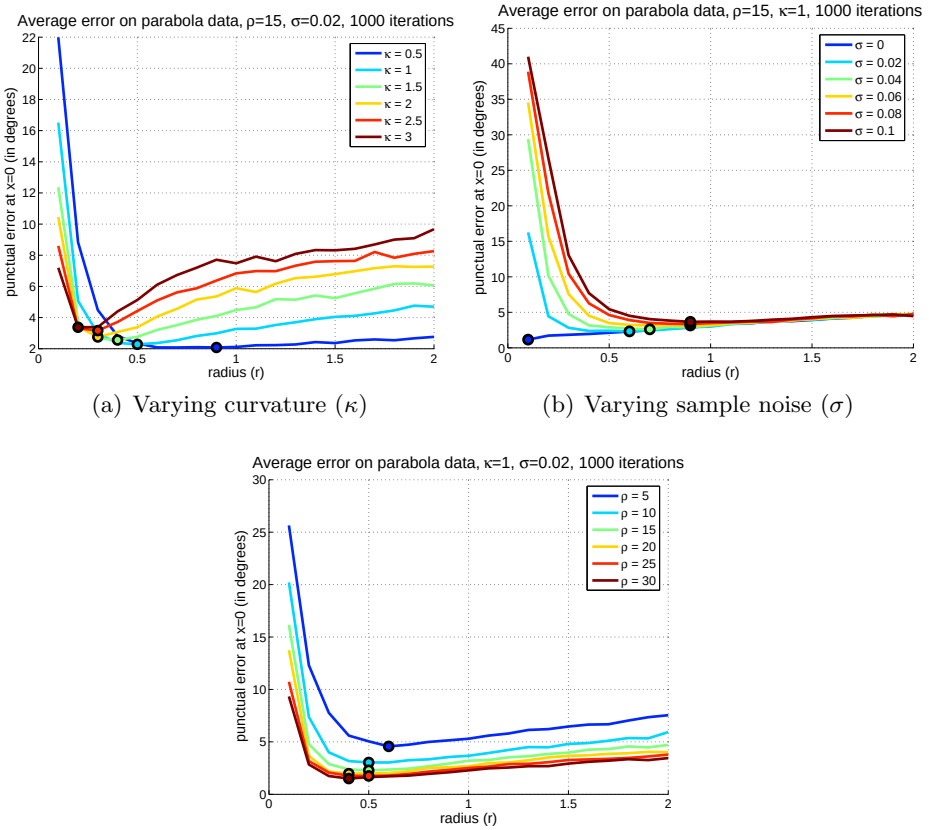


Figure 2.9: Laser scan of a concertina wire having the geometry of two oppositely wound helices of equal diameter: (a) Scene outline showing the concertina wire structure and the ground plane (b) Raw 3-D points color-coded by elevation [axis length = 0.5m], (c) DMST graph constructed on manually extracted non-ground points, (d) Estimated tangents using scale-adaptive PCA.

the edges that form the MST, and then iteratively constructing subsequent MST on the remaining edges. This graph construction procedure is followed by a post-processing step of rejecting edges with length greater than that determined by our assumed minimum global density (ρ_0). Figure 2.9 shows an example of a construction for points acquired from a concertina wire. The range sensor used is a SICK LMS-291 attached to a custom made scanning mount. The angular separation between laser beams is $\frac{1}{4}^\circ$ over a 100° field of view. The angular separation between laser sweeps is $\frac{2}{3}^\circ$ over a range of 115°.

The construction using DMSTs has some desirable properties over traditional k-nearest neighbor or ϵ-ball schemes. In practice, it produces connected graphs without undesirable gaps and does not induce edges to clump together in noisy regions having relatively higher point density. The only parameter to be chosen is the number of spanning trees (in our case, $= 2$ for curves and $= 4$ for surfaces) and we have observed it to be robust to changes in the dataset for our choice. In Section 5.1 we discuss the problem of geometric graph construction in more detail.



(a) Varying curvature (κ)

(b) Varying sample noise (σ)

(c) Varying sample density (ρ)

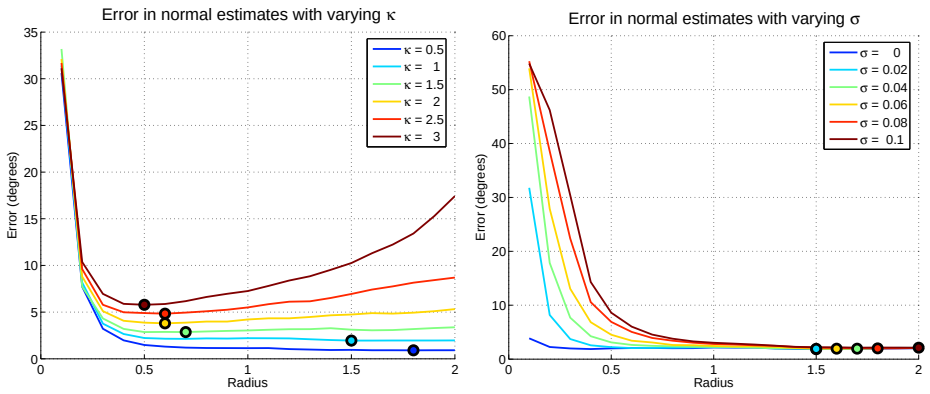
Figure 2.10: Plot of angular error observed when computing tangents from points sampled from 2D parabolas for varying sampling and geometric parameters. Colored markers indicate location of minima on the corresponding line. (Figure is best seen in color.)

Validation

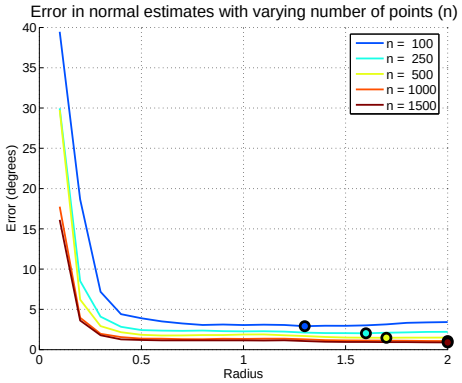
Curves in 2D and 3D

As a first step, we test our model by attempting to validate the behavior predicted by the analytical bounds of Section 2.3 for the 2D case. This is done by analytically computing the estimation error on points sampled from synthetic 2D curves The test curve is a 2D parabola and the error in tangent direction is evaluated at the apex for various values of curvature and point density. The estimation is done using PCA for various values of neighborhood radius. The reader is encouraged to compare Figures 2.10(a)-2.10(c) with the analytic curves of Figures 2.4(a)-2.4(c).

Figure 2.10(a) shows the observed angular error with varying curvature κ of the parabola. It can be seen to show the predicted systematic decrease in scale for increased curvature. The variation of estimation error with sample noise σ_0^2 in Figure 2.10(b) shows the increase



(a) Varying curvature (κ) (b) Varying sample noise (σ)



(c) Varying no. of points (n)

Figure 2.11: Plots of angular error observed when computing normals from points sampled from paraboloid surfaces for varying geometry and sampling parameters. Colored markers indicate location of minima on the corresponding line. (Figure is best seen in color.)

in optimal scale for increased noise. Figure 2.10(c) shows the relatively small change in choice of optimal scale except at a low density. It also shows the expected decrease in error with increasing sample density.

Surfaces

We perform a validation experiment similar to that done for curves but with a paraboloid surface. The estimate of surface normal at a fixed point is evaluated for varying values of geometric and sampling parameters. Figures 2.11(a), 2.11(b) and 2.11(c) show variation in estimation error the radius of curvature, sampling noise level and number of points. The conclusion obtained by comparing these figures with the corresponding figures in Section 2.4 are similar to those for curves, and are omitted here for brevity.

Stability and Accuracy

Curves

We choose to compare the proposed method with the polynomial fitting algorithm of [67], as the latter performed nearly uniformly better experimentally on a variety of synthetic curves against a large family of other fitting approaches based on Gaussian smoothing, Fourier transforms and others.

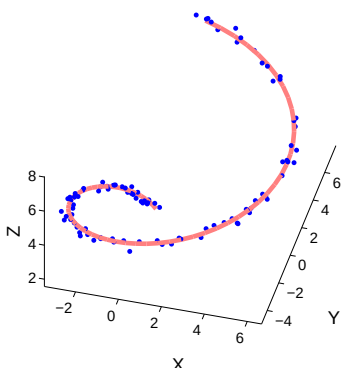
Figure 2.12 presents results on 100 samples from two synthetic curves, a 2D hypocycloid and a 3D conical helix (as also used in [67]). The hypocycloid has the parametric form $\big(4\cos(t) - \cos(2t),\ 4\sin(t) + \sin(2t)\big)$ with $t \in [0, 2\pi]$ and the helix has the form $(t\cos(t), t\sin(t), t)$ with $t \in [\pi/2, 5\pi/2]$. These two are presented as their constantly varying curvature violates the assumptions made in both algorithms, and PCA is intuitively not expected to perform well on them under its simplistic assumption of local linearity. The algorithms were run for 30 datasets each for varied sample noise (σ). A range of values for radius r_0 were used to fix the scale for polynomial fitting and correspondingly serve as the starting point of the proposed PCA algorithm.

As seen in Figure 2.12, the scale-adaptive PCA performs surprisingly well in terms of error rate, and is much more stable to varying values of r_0. Similar results were observed on comparison with other 2D and 3D curves from [67].

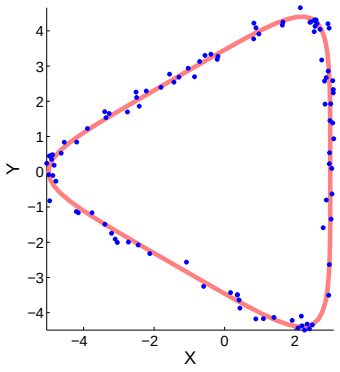
Surfaces

The accuracy of the adaptive PCA algorithm for surface normal estimation was tested using data from an open space natural environment containing a 1.5 m high pile of gravel surrounded by short cut and non cut grass. We collected high resolution, high density data with the Zoller+Fröhlich (Z+F) laser and also collected low-resolution aerial data for the same scene with the CMU autonomous helicopter [63, 80]. The two data sets were co-registered. The Z+F data was used to produce the ground truth used to estimate the normal reconstruction error in the aerial data.

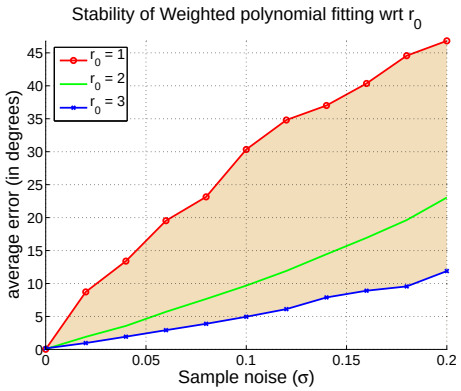
Figure 2.13 shows the results obtained. Figure 2.13-(a) shows the computed normals and the support regions for selected points in the aerial data. Figure 2.13-(b) shows the normal and support regions for the same points but overlaid on top of the high-resolution ground data. Points in Figure 2.13-(a) are color-coded by the difference between the error in estimated normals and the lowest possible error obtainable for any choice of support region in the aerial data. The algorithm was run on 6604 points and the median error was only 3.74° with an interquartile range of 5.42°.



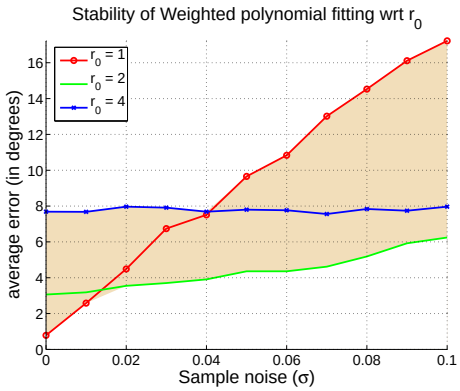
(a) 3D conical helix



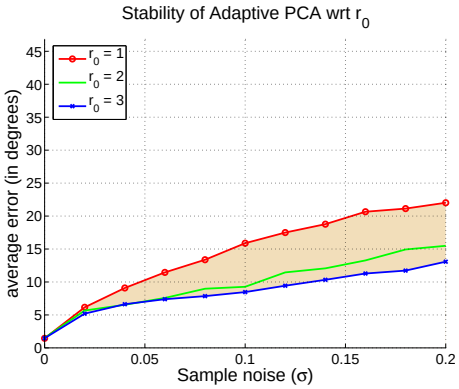
(b) 2D hypocycloid



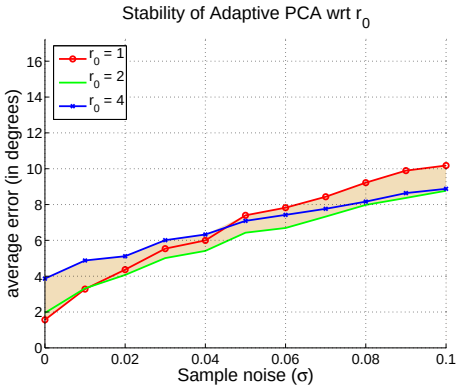
(c) Polynomial fitting - Helix



(d) Polynomial fitting - Hypocycloid

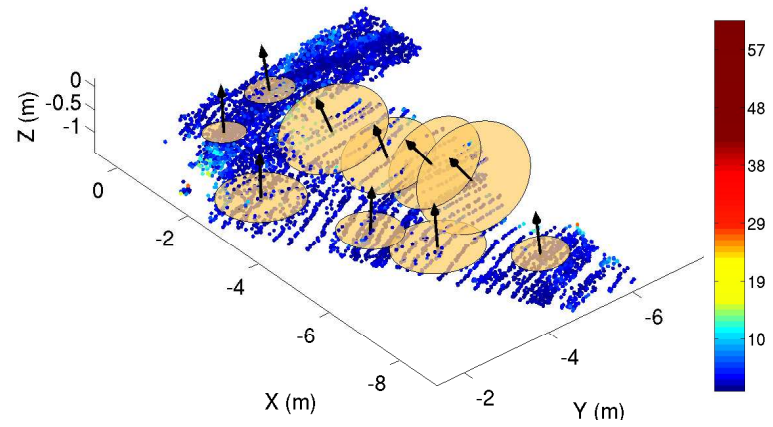


(e) Adaptive PCA - Helix

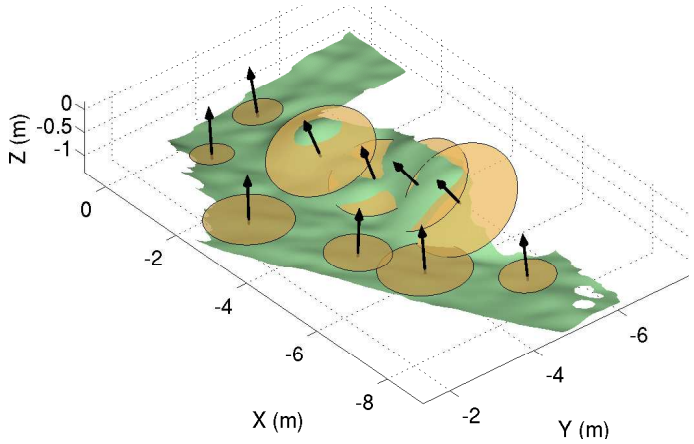


(f) Adaptive PCA - Hypocycloid

Figure 2.12: Plot of observed error on (a) 3D conical helix and (b) 2D hypocycloid dataset. The top row plots the true curves along with a typical example of points sampled from them. The middle row (c-d) plots error obtained with the method of [67] and the bottom row (e-f) plots error with the proposed scale-adaptive PCA. Error plots in the same column have the same axis limits. The lower variation in the more stable PCA method as indicated by its thinner shaded region can be clearly seen.



(a) Low-resolution raw data



(b) High-resolution data used as ground-truth

Figure 2.13: Normal estimation from the Zoller+Fröhlich (Z+F) laser dataset. (a) Estimated surface normals and corresponding support region for selected points are overlaid on top of the low-resolution data. Points are colored by the angular error (in degrees) of the estimated normal. Error is computed using ground-truth obtained from a high-resolution mesh (b) that also have the normals and support regions overlaid. (See text for details.)

2.6 Application: Improving descriptor repeatability

As mentioned in Section 1.2, the construction of patch-based shape models requires the selection of interest regions and the construction of reliable shape descriptors from points observed in those regions. In Chapter 4 we will look at the first task of multi-scale interest region detection. This section focuses on the challenges posed by the latter task of shape descriptor construction.

The construction of reliable shape descriptors is challenging when given sparse data. This is because the binning process used to construct shape descriptors, such as spin images [53], shows high variability with few data points and could possibly result in empty

bins where no observations are made.

One way to circumvent this is to synthetically resample new points from the underlying shape of the object. This can be done in our presented framework by simply randomly generating points in the vicinity of observed points in the neighborhood of interest and "denoising" them starting from their random initial positions, so that they lie on the surface estimated at that point. Each randomly initialized 3D point is projected onto the plane fitted to the local neighborhood of the point (Figure 2.1). The neighborhood size used for estimating the plane parameters is allowed to adapt to the local surface in accordance with the method proposed in Section 2.4. For brevity, we will refer to the spin image generated using a combination of observed and synthetically sampled points as the synthesized spin image.

We constructed an experiment to demonstrate the utility of this technique as follows. We start with a high-resolution point cloud of car (which we will refer to as the model cloud) containing about 16000 points, as well as 15 individual scans (referred to as test clouds) of the same car from directions spanning 360°. The test and model clouds were acquired using a SICK laser scanner. The scans were clutter free and the level of noise was relatively low, the car having been scanned from a distance of only about 6m away from the sensor. The resolution of the model cloud, measured in terms of the number of points, varied between 10-15 times that of the test clouds.

Using this data, we perform the following experiment. We choose several points at random from each test cloud and construct a spin image at each location for a pre-specified support radius using the traditional binning approach as well as with the resynthesized points. Spin images are also constructed for the corresponding points on the model cloud for the same radius. The distances between corresponding pairs of spin images are computed for both traditional and synthesized spin images using the Bhattacharya distance metric

$$d_{\text{Bhatt}}(\mathbf{p}, \mathbf{q}) = 1 - \sum_i \sum_j \sqrt{p_{ij} q_{ij}}, \tag{2.48}$$

where p_{ij} and q_{ij} represent the values of the (i, j)-th bin for the two spin image histograms $\mathbf{p}$ and $\mathbf{q}$.

The *relative difference* in the distances between the model and traditional spin image and the model and synthesized spin image is taken to be the measure of improvement in the spin image quality.

The experiment is repeated for a range of subsampled versions of the test cloud and the parameter of subsampling rate was selected among 1 (no subsampling) 3, 5 and 10 (lowest resolution). Other parameters that were varied are the spin image radius (0.5m or 1m) and the number of spin image bins in the radial axis direction (5, 10 or 15). The number of bins in the longitudinal axis (perpendicular to the surface) is set to be twice the number

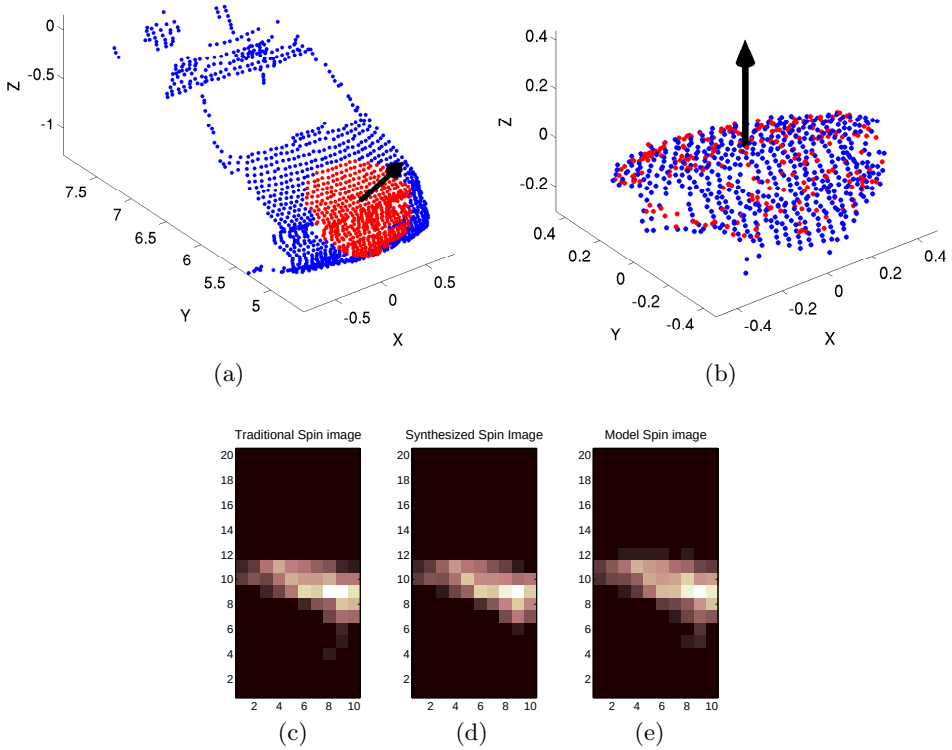


Figure 2.14: (a) View of an example full-resolution (1523 points) test cloud acquired by scanning the front of a car. The red points denote the neighborhood considered for the spin image computed at the point of interest which is located at the base of the black arrow. The arrow indicates the locally computed surface normal. (b) shows the same local neighborhood around the selected point along with estimated surface normal. The blue points are the ones observed in the scan and the red points are synthesized. (c) shows the traditional spin image generated at that point of interest using only the observed (blue) points, and (d) shows the synthesized spin image generated using both observed and resampled points. (e) shows what the spin image should look like based on a high-resolution model cloud of the same car having about 10 times as many points as the test cloud. It may be observed that since the test cloud has sufficiently high resolution with respect to the spin image bin size, the traditional spin image in (c) is visually quite similar to the model spin image (e), and so is the synthesized image (d).

of bins in the radial axis direction. Figures 2.14 and 2.15 show examples of the spin image augmentation procedure on a test cloud and a severely subsampled version of the same scan having only 1/150th the points of the model cloud.

Each experiment performed with a fixed set of parameters involved a comparison of 300 spin images. The number of points synthesized per spin image is fixed to 250. To obtain the score for each setting of the parameters, 100 trials were performed and their results averaged.

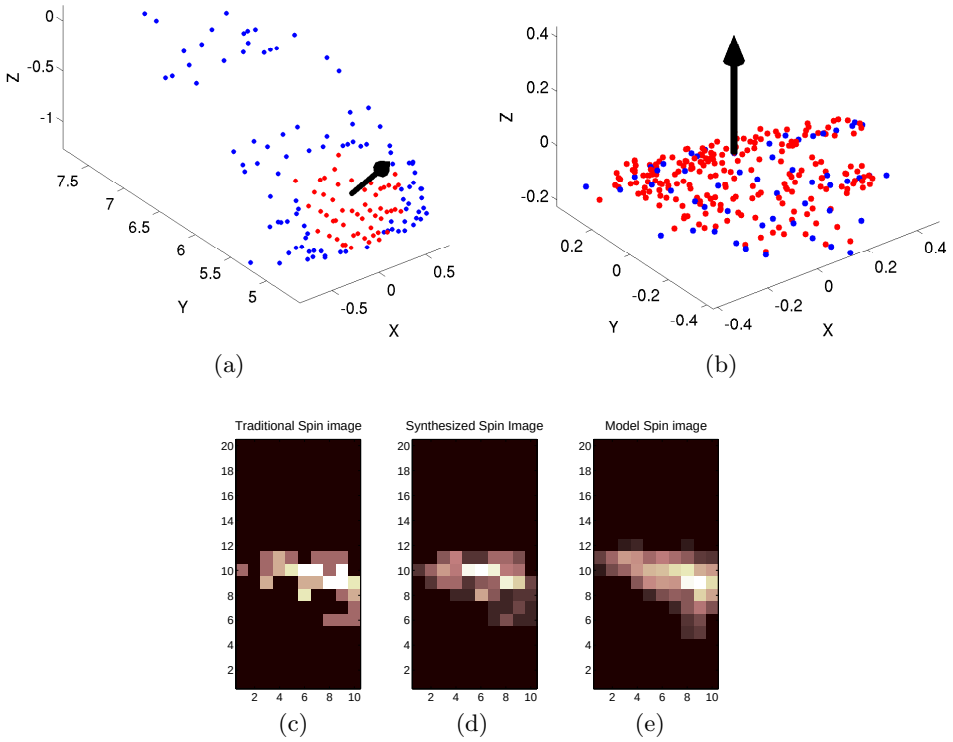
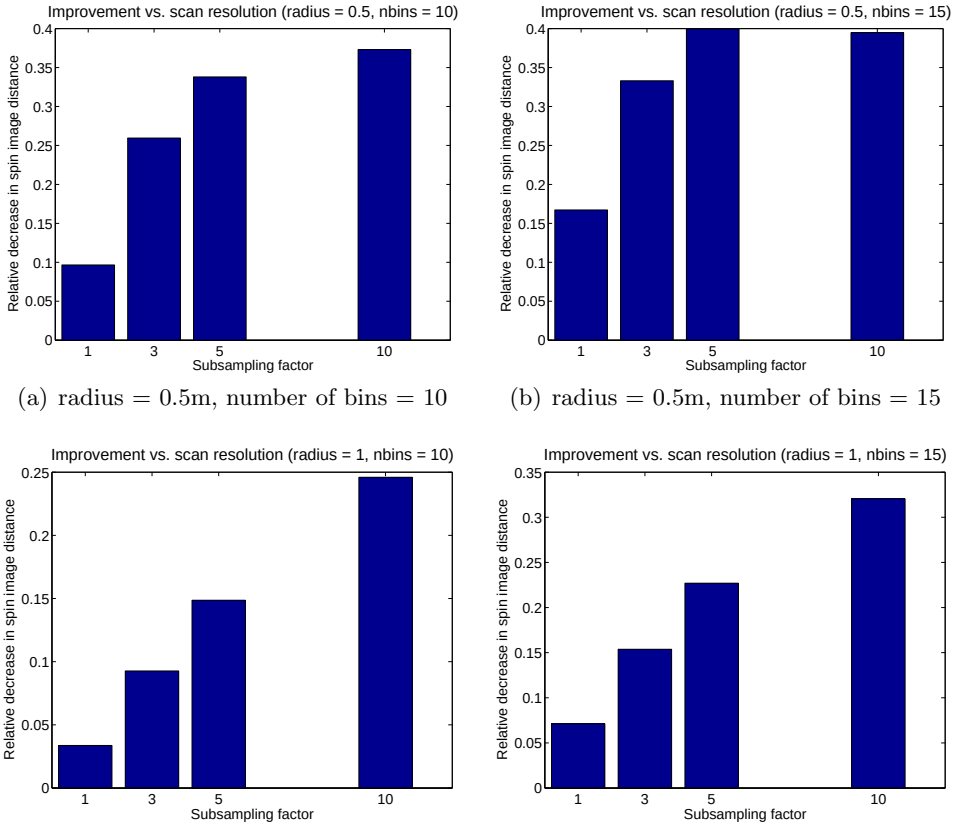


Figure 2.15: (a) View of an example low-resolution (153 points) test cloud obtained by subsampling the original scan from Figure 2.14(a). The red points denote the neighborhood considered for the spin image computed at the point of interest which is located at the base of the black arrow. The arrow indicates the locally computed surface normal. (b) shows the same local neighborhood around the selected point along with estimated surface normal. The blue points are the ones observed in the scan and the red points are synthesized. (c) shows the traditional spin image generated at that point of interest using only the observed (blue) points, and (d) shows the synthesized spin image generated using both observed and resampled points. (e) shows what the spin image should look like based on a high-resolution model cloud of the same car having about 10 times as many points as the test cloud. It may be observed that since the test cloud has relatively low resolution with respect to the spin image bin size, the traditional spin image in (c) is visually quite different from the model spin image and contains several empty cells (e). In contrast, the synthesized image (d) is visually more similar and smooth.

Figure 2.16 plots the relative difference in spin image distance from the model spin image for varying subsampling rates and bin sizes. It may be seen that the synthesized spin images show consistent improvement in fidelity of between 10% and 40% for moderate and high number of histogram bins (10-15 bins per histogram axis), for a range of subsampled versions of test data (subsampling ratio of $1-10$). This conclusion remains unchanged with increase in the radius of the neighborhood of points considered while constructing each spin



(a) radius = 0.5m, number of bins = 10

(b) radius = 0.5m, number of bins = 15

(c) radius = 1m, number of bins = 10

(d) radius = 1m, number of bins = 15

Figure 2.16: Improvement in spin image for varying subsampling rates and number of spin image bins for a fixed radius. The x-axis in each plot shows the subsampling rates tested and the y-axis plots the relative decrease in spin image distance from the corresponding spin image on the high-resolution model cloud. The first column shows the result with number of bins = 10 and the second column shows the result with number of bins = 15. The top and bottom rows show results for a spin image support radius of 0.5m and 1m respectively.

image.

Thus, this experiment demonstrates the utility of using a point synthesis procedure in conjunction with the proposed regression method for improving the repeatability of computed spin image shape descriptors.

2.7 Discussion

As elaborated in Section 2.2, the problem of fitting lines or surfaces to unorganized point clouds is one that does not fit easily into the realm of classical statistical methods. Furthermore, in trying to make formal guarantees in the accuracy of traditional methods, one is forced to make do with asymptotic guarantees which do not necessarily translate to real

world performance. What this work shows is that it is possible to relate surface (or curve) geometry to finite-sample accuracy of a statistical estimator, and exploit the available free parameter of support radius to minimize this estimation error. Thus, with principled scale selection, the error in tangent estimation using a simple estimator such as naïve local PCA can be made comparable, somewhat counter-intuitively, to the best fixed-scale alternative based on local polynomial fitting.

Generalization: It is important to realize that the method of analysis proposed in this work is not restricted to the PCA-based estimators presented in Section 2.3 and Section 2.4. These particular choices were largely motivated by the need to study existing approaches that are popularly used [48, 77, 81, 104, 127] for the same kinds of tasks. There is potential for modifying other algorithms that may be recast as local model fitting methods so that the chosen support radius is automatically adapted to the fitted surface. Indeed, when the local model equation $f(\mathbf{x}_i^o;\theta)$ is the bilinear form $\mathbf{v}(\mathbf{x}_i^o)^{\mathrm{T}}\theta$ of (2.3), the equivalent model parameter estimation problem will take the form of an eigenvector problem, and the matrix perturbation results of this chapter will be applicable.

It is also not unreasonable to expect that some of these alternatives may have better numerical accuracy and stability than the algorithms presented in this chapter. For instance, by choosing a vector map $\mathbf{v}(\mathbf{x})$ with 2^{nd} order moments, we can fit a local quadric instead of a local plane for the surface fitting problem in Section 2.4. By using quadrics, we can eliminate the need for a separate procedure to estimate the mean curvature κ and explicitly reparameterize $\mathbf{x}$ as a function of only the coefficients of the fitted surface.

The same theoretical analysis can also be performed for the more robust variant of weighted PCA for some fixed family of weighting functions (e.g. Gaussian). This would potentially make the proposed algorithm more robust overall to outliers as well as to poor graph construction.

Dynamic updates: In applications involving mobile sensors, it is not uncommon to visit areas more than once and make multiple observations of a scene. Thus, an interesting question to ask is whether the proposed algorithm is amenable to dynamic point addition.

There are two components of the proposed approach that are affected by the addition of points. The first component is that of computing (and minimizing) the error bound to select the neighborhood radius, and the corresponding estimation of the model parameters. The complexity of computation of the bound is unchanged, as only the parameter n changes in the expression with each point insertion. Also, for the fitting problems considered in this thesis, the bilinear form of the locally fit implicit function ensures that the model fitting problem is cast as an eigenvector problem. In the case of fitting planes and curves, direct computation of the modified eigenvectors is inexpensive since the matrix size is small (3×3). When the model has a higher number of parameters (say θ is an m-vector), this computation may be expensive to perform for each added point as the size of $\hat{M}_n$ will grow to $m \times m$.

Fortunately, methods for compute principal and minimal eigenvectors in this scenario are well studied, more recently by the name of incremental PCA [22, 42, 69]. However, this increased speed will necessarily come at the cost of increase in memory consumption as one would need to store the scatter matrix associated with each point at which the model parameters are to be re-estimated.

The second component in the proposed algorithm to consider is that of the graph used to compute geodesic distances (approximated as path distances). The ability to modify the graph to accommodate additional points naturally depends on the manner of construction. The question of how quickly this can be done when the graph construction is based on DMSTs is currently an open problem. However, one may expect that its answer will depend on how quickly each minimum spanning tree formed in the original can be adjusted to accommodate new points. While efficient algorithms to handle point and edge insertions in MSTs exist for the restricted case of points in a plane, efficient algorithms for this in higher dimensions are a subject of study in computation geometry[36, 37]. A practical trade-off may be to use approximate k-nearest neighbor graphs in which point insertions are handled by connecting each newly added point its k-nearest neighbors. However, there is a need to investigate the possibility of using dynamic MST algorithms for graph construction, as well as of using dynamic Dijkstra-like algorithms for efficient neighborhood search without having to recompute distance matrices for the entire set of points.

Alternatives using global fitting: One of the premises of this chapter was that the fitting of implicit functions to model the entire dataset may not be feasible, either due to the complexity of the scene or the potentially large number of model parameters that would be required. We briefly discuss some recent efforts at overcoming these challenges.

In the machine learning literature, there has been renewed interest in the use of max-margin methods as well as Gaussian Processes (GP) for solving non-linear non-parametric regression problems [73, 90, 115]. Work in [114] presented an approach to algebraic curve fitting using a support vector formulation, that fit a global implicit function whose zero-level set contained a specific set of data points. The problem is formulated in a manner nearly identical to classical hard-margin linear SVM classification, with the penalty function proportional to the value of the implicit function evaluated at the specified set of input points. To avoid a degenerate solution, a set of off-manifold points are also specified and associated with labels $+1$ and -1 to depending on whether they are interior or exterior to the surface of interest. The authors of [114] also proposed a modified rational polynomial kernel function as being more suitable for the geometric fitting problem. More recently, work in [92] demonstrated how algorithms based on GPs can be implemented efficiently and scaled to large datasets, and showed an application to surface reconstruction.

The chief advantage of this class of approaches is that they are unaffected by variation in the distribution of points. Because there is no need to construct elaborate graphs to

compute distances between points, these methods are also elegant in their execution.

However, these methods come with certain disadvantages as well. It is currently unclear how to incorporate the knowledge of noise level in the reconstruction. In SVM-based methods, there is the common question of how to select the value of the so-called c-parameter that controls the tradeoff between the quality of the fit to the data and the regularization. Incorrectly set values can lead sometimes to topologically unstable solutions and to the reconstruction qualitatively appearing over-smoothed [114]. In the case of GP-based methods [92], the constraints on the off-manifold points are currently given by specifying not just the locations of the points but also the function values at those points. This is currently done by computing the normal at the points and projecting along it, which is error prone. More generally, it is unclear how the specification of these locations and values at off-manifold points affects the accuracy of the reconstruction. While we are currently unaware of an instance of this class of approaches that addresses all these concerns, it is certainly an area that deserves further study.

CHAPTER 3

Constrained Local Regression

The previous chapter showed how the problem of fitting smooth manifolds to point clouds can be analyzed in the framework of traditional regression to yield useful guarantees on accuracy. The goal of this chapter is to overcome the limitation posed by one specific assumption of the previous method – that of the underlying curves or surfaces being smooth.

We show that the fitting of sharp geometric features such as intersections of curves or surfaces from irregular point samples can similarly be cast as a non-parametric regression problem. As before, the proposed method does not assume prior availability of connectivity information, and avoids computing surface normals or meshes at intermediate steps.

3.1 Motivation

One assumption made by the method proposed in Chapter 2, or in fact any traditional regression algorithm, is that the curves (or surfaces) being estimated vary *smoothly*. This assumption can pose challenges in several domains, particularly those involving man-made objects, where the underlying geometry consists of surfaces that are only piece-wise smooth. Such objects possess sharp features such as corners and edges which are created when these smooth surfaces intersect. The reconstruction of these sharp features is particularly challenging as noise and sharp features are inherently ambiguous, and physical limitations in scanner resolution prevent proper sampling of such high-frequency features.

Because of this limitation, current approaches to model fitting have evolved toward treating sharp geometric features in a completely different manner from smooth surfaces. Consider an example of two planar faces meeting at an edge, and that a set of points are

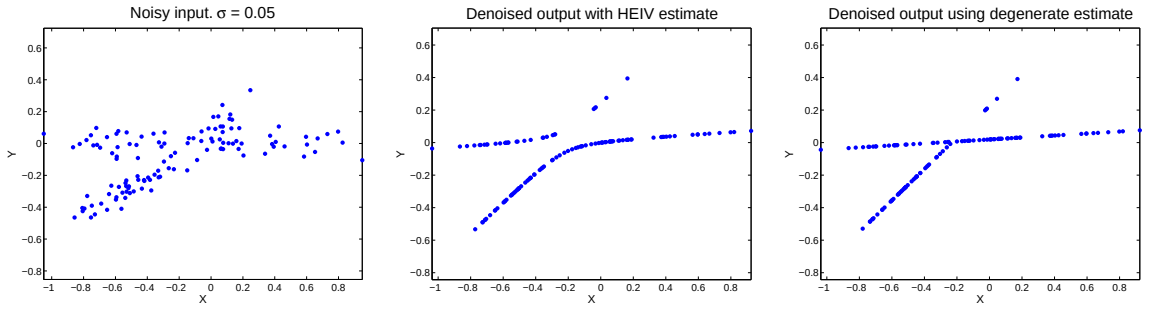
sampled from this edge. As elaborated in Section 3.2, current approaches usually involve explicitly recovering the parameters of each planar face as well as the association of each point to one of the two faces. Techniques based on Expectation Maximization (EM) or robust M-estimators tend to perform poorly when applied to a small number of points simply because they require relatively accurate initialization.

The claim made in this chapter is that model fitting of sharp geometric features involving the intersection of two planar faces (or lines in 2D) can be performed in the same traditional regression framework as used for smooth surfaces in the previous chapter. The key idea is that sharp geometric features can thought of as cases of "degenerate" smooth surfaces having an associated subset of model parameters. The way we implement this idea is by modeling a surface intersection as a degenerate higher-order polynomial, A degenerate polynomial is defined as one that can be factored into the product of lower dimensional polynomials, where each factor represents a low-dimensional linear subspace in the context of this document.

Before proceeding further, we wish to briefly comment on some superficial similarities between this key idea and some related work. The use of a high-order polynomial product to represent a combination of (low-order polynomial) subspaces is not new. Work by Taubin [105] fit complex 3D curves to data, and used a high-order polynomial to represent the intersection of surfaces that formed the curve. It used an approximation to the distance function that reduced the fitting problem to an easily solvable generalized eigenvector problem, but implicitly made the assumption of *uniform* noise covariance on the points. More recently, Vidal *et al.* [113] proposed the Generalized Principal Component Analysis (GPCA) algorithm to model combinations of linear subspaces. In brief, the key differences between the approach in chapter and GPCA are that our formulation allows noise models to be easily incorporated, and that unlike the approach in [113], there is no need to resort to a separate estimation procedure to compute the parameters of the individual subspaces. We will elaborate on these differences further at the end of Section 3.4 after presenting the solution to the proposed formulation.

Thus we propose to overcome the limitation of previous methods through the use of a modified local regression technique that models sharp intersections as a degenerate higher-order polynomial representing the implicit product of lower dimensional subspaces. The geometric fitting of these sharp features may then be done by iteratively first solving a generalized eigenvector problem to fit a smooth surface and then subjecting the solution to some non-linear constraints required for the model parameters to represent a degenerate surface. By using a local parameterization, the component of the model fitting problem involving the generalized eigenvector solution may then be analyzed in the manner presented in Chapter 2.

Thus a high-level description of how to perform local semi-parametric analysis of points



(a) (b) (c)

Figure 3.1: Example of denoising a toy dataset by global fitting of an implicit degenerate polynomial (a) Input data consisting of points from two intersecting line segments corrupted with uniform Gaussian noise of standard deviation $\sigma = 0.5$ (b) Denoised data using an implicit quadratic fit with the HEIV estimator [75]. Note that the sharp feature formed by the intersection is not preserved. (c) Denoised output after imposing degeneracy constraints on fit coefficients fixes this problem.

sampled from a sharp geometric feature would be as follows:

(i) Starting with a candidate neighborhood, find model parameters that fit a smooth surface to the data points in that neighborhood.

(ii) Using a local parameterization of the smooth surface, find the neighborhood radius that best minimizes the error bound in the estimate of those parameters.

(iii) Project the model parameters to satisfy the non-linear constraints characteristic of degenerate surfaces.

(iv) Redefine the neighborhood size using the new estimate of radius, and repeat from Step (i) till the model parameters converge.

This chapter focuses on steps (i) and (iii) of this process by modeling and solving the regression problem associated with fitting sharp geometric features. In Appendix B we briefly outline the theory of eigenvector perturbation relevant to the generalized eigenvector problem that is solved in Step (i), and omit the derivation of the associated error bound.

3.2 Related Work

There have been several proposed approaches to recover geometry from noisy point samples, many of which were outlined previously in Section 2.1. In this section we highlight some of the approaches while paying special attention to the way they handle intersecting surfaces, as well as inspect some commonly made implicit assumptions about the sampled points.

In general, past approaches have often made simplifying assumptions about the data due to the ill-posed nature of the problem. Methods based on classical regression typically assume that the geometry can be treated locally as a smooth surface, which is clearly a problem at surface intersections. Most approaches assume the noise in the data to be isotropic and homogeneous, perhaps because they often lead to convenient closed-form analytical expressions. However, noise is almost always highly directional and dependent on the distance of the point to the sensor. This is, for example, the case with laser range scanners. Ignoring the anisotropy in the noise model typically results in a systematic bias in the surface reconstruction [75]. Some methods assume the reliable availability of additional information about the geometry, such as connectivity information (meshes) and surfaces normals, and try to produce estimates of geometry that agree with this information. However, the estimation of meshes is error-prone, and differential quantities like surface normals and tangents are, of course, not even well-defined at intersections.

Surface estimation from noisy point samples may be posed naturally as an instance of the local *regression* problem from classical statistics. A popular non-parametric technique in this category is locally weighted regression, also known in its more general form as Savitzky-Golay filtering [89]. As explained in [45], it adapts well to non-uniformly sampled data and exhibits less bias at boundaries.

The moving least squares (MLS) technique [65] builds on this by first computing a locally approximating hyperplane and then applying a locally weighted regression procedure to the data projected to the hyperplane. The technique works well with noise but is unable to reproduce sharp features due to its implicit assumption of a single locally smooth surface.

Fleishman *et al.* [97] fit quadratic polynomials locally to data and used standard techniques from robust statistics in the fitting process. Using residual plots, points across an intersection were identified and treated as outliers in this framework, and subsequent stages of local classification and reprojection were required to obtain the final result. The technique relied on an initially finding low-noise local regions to obtain a reliable estimate of the quadratic fit, which may not always be feasible.

Hoppe *et al.* [49] proposed a method based on associating tangent planes with each point, and defining the distance of a point to the implicit surface as the orthogonal distance to the tangent plane associated with the point closest to it. However, the unreliable initial estimates of tangent planes due to noise and high-frequency features resulted in the final models being oversmoothed.

Wang *et al.* [117] proposed a more complicated procedure involving a sequence of voxelization and gap-filling, topological thinning and mesh-generation. Based on local connectivity, each voxel is classified as being at a junction, boundary and surface interior. The procedure has several points of failure, particularly at regions that are not densely sampled with respect to the voxel size.

Hubeli *et al.* [51] proposed a multi-resolution processing scheme that preserved local connectivity at junctions and boundaries, but required a mesh to make the initial classification.

The method presented in this chapter combines the strengths of some of the previous approaches. We modify a locally weighted smoother to implicitly represent potentially multiple linear subspaces through a degenerate high-order polynomial. This allows us to explicitly model edge intersections instead of trying to fit a highly non-smooth surface. The use of a local smoother preserves the adaptability to varying sample density. By posing the regression as a constrained energy minimization problem, we can easily incorporate anisotropic error models in the data. We derive the algorithm in Section 3.3, explain its implementation in Section 3.5, and examine its behavior through several experiments in Section 3.6.

3.3 Constrained Local Regression

In this section, we describe a modified regression algorithm that will enable us to recover noise-free surfaces from noisy point cloud data, while preserving high-frequency geometric features. We will first consider the case of 2D data to simplify the explanation of the main idea.

Problem definition and approach

We assume that we are given a set of points $\{\mathbf{x}_i\} \in \mathbb{R}^d$ that are assumed to be noisy observations of the positions of true points $\mathbf{x}_i^o \in \mathbb{R}^d$ that lie on a locally continuous, but not necessarily smooth surface. The associated noise covariances $\Lambda_i \in \mathbf{S}_+^d$ at each point are assumed to be known, for instance, through a noise model of the sensor used to acquire the points. The points are assumed to be *irregular*, in the sense that they do not follow a known regular sampling distribution, and *unstructured* in the sense that the local connectivity of the points, such as in the form of a mesh, is not available.

Our goal will be to compute an estimate $\hat{\mathbf{x}}_i$ of the true point position $\mathbf{x}_i^o$ corresponding to each observed point $\mathbf{x}_i$. The operating assumption will be that points in a local neighborhood, $\mathcal{N}(\mathbf{x}_i)$ of $\mathbf{x}_i$ may be modeled as belonging to one or more linear subspaces. This naturally suggests a maximum likelihood (or equivalently defined minimum energy) formulation of the problem, subject to the constraint that the noise-free points lie on one or more subspaces. Since the parameters of the models, number of models, as well as the association of the points to each subspaces are unknown, a popular strategy is to attempt a procedure of iterative model fitting and data association, such as Expectation-Maximization (EM).

However, such iterative procedures tend to be error prone when performed with few and noisy data points, as may be expected for our problem.

Instead, we propose to model the problem as one of maximum likelihood estimation subject to *two* types of constraints. The first type of constraint ensures that each noise-free point in the neighborhood of interest lies on a high-order polynomial whose degree is an upper bound on the number of subspaces in that neighborhood. The second type of constraint is a function of the coefficients of the polynomial, which restricts the family of allowable polynomials to degenerate forms that can represent combinations of linear subspaces. In practice, we will sometimes relax the constraint of degeneracy to make the optimization problem more tractable at the expense of admitting a single non-linear manifold but restrict them to locally developable surfaces.

Constraints in the 2D case

In the case of 2D data, each local neighborhood can be modeled as as consisting of a pair of linear subspaces. Thus, locally the shape may be described implicitly as a zero-level set of the equation $(\gamma_1^{\mathrm{T}}\mathbf{x} + d_1)(\gamma_2^{\mathrm{T}}\mathbf{x} + d_2) = 0$, where $\gamma_i \in \mathbb{R}^2$, $d_i \in \mathbb{R}$ are the parameters for each of the two linear subspaces (lines in the case of 2D data). Note that this subsumes the case where the subspaces coincide. Expanding out the terms yields an inhomogeneous 2^{nd} degree polynomial in 2 variables, which we will refer to as x and y corresponding to each spatial dimension.

Let us denote the coefficients of each monomial in the polynomial as given by the expression

$$\theta_1 x^2 + \theta_2 y^2 + \theta_3 xy + \theta_4 x + \theta_5 y + \theta_6 = 0. \tag{3.1}$$

This may be rewritten in matrix form as

$$\begin{bmatrix}\mathbf{x} & 1\end{bmatrix} \begin{bmatrix} 2\theta_1 & \theta_3 & \theta_4 \\ \theta_3 & 2\theta_2 & \theta_5 \\ \theta_4 & \theta_5 & 2\theta_6 \end{bmatrix} \begin{bmatrix}\mathbf{x} \\ 1\end{bmatrix} = \begin{bmatrix}\mathbf{x} & 1\end{bmatrix} A \begin{bmatrix}\mathbf{x} \\ 1\end{bmatrix} = 0. \tag{3.2}$$

It is a known result in algebraic geometry that a quadratic in two variables reduces to a product of two linear factors only if A is singular [13]. In fact, the case where A has only rank one corresponds to the case where the subspaces (lines) coincide.

The determinant in this case may be written explicitly to yield the equality

$$4\theta_1\theta_2\theta_6 + \theta_3\theta_4\theta_5 - (\theta_2\theta_4^2 + \theta_1\theta_5^2 + \theta_6\theta_3^2) = 0, \tag{3.3}$$

which can be used to constrain the solution for the θ_i's. We will denote such constraints on the coefficients of the polynomial as $\phi(\theta) = 0$.

Constrained optimization

Together with the constraint on coefficients we can pose the task as a constrained optimization problem defined at each point of interest $\mathbf{x} \in \{\mathbf{x}_i\}$ given by

$$\arg\min_{\{\hat{\mathbf{x}}_i\},\theta} \sum_i w_i(\mathbf{x}_i - \hat{\mathbf{x}}_i)^{\mathrm{T}}\Lambda_i^{-1}(\mathbf{x}_i - \hat{\mathbf{x}}_i), \tag{3.4}$$

subject to two sets of constraints:

(i) The first set of constraints is $\theta^{\mathrm{T}}\mathbf{v}(\hat{\mathbf{x}}_i) = 0 \ \ \forall i$, where $\theta \in \mathbb{R}^m$ is the vector of monomial coefficients and $\mathbf{v}(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}^m$ is the mapping from the d-dimensional point to the monomials formed by its coordinates. For the 2D case ($d = 2$), the number of monomial terms $m = 6$.

(ii) The second set of constraints $\phi(\theta) = 0$ is that on the monomial coefficients, which is (3.3) in the case of 2D data.

The weighting term w_i is used to give more importance to points closer to the point of interest $\mathbf{x}$. We can define w_i using a kernel loss function, such as a truncated Gaussian function centered at $\mathbf{x}$, so as to suitably delineate the neighborhood of interest $\mathcal{N}(\mathbf{x})$. Our implementation uses the Epanechnikov kernel $w_i = 1 - \|\mathbf{x} - \mathbf{x}_i\|^2/\sigma^2$ for $\|\mathbf{x} - \mathbf{x}_i\| < \sigma$ and 0 elsewhere, chosen because of its finite support and asymptotically optimal properties in related tasks such as kernel regression [116]. Here σ determines the length scale, which may be chosen differently for each $\mathbf{x}$. We comment on its selection later in Section 3.5. In what follows, we will sometimes drop the dependence on $\mathbf{x}$ in the notation for clarity, with the understanding that the optimization problem is being solved for points in a local neighborhood of *each* $\mathbf{x} \in \{\mathbf{x}_i\}$.

The standard approach to solving such a constrained optimization problem is by finding the stationary points of the associated Lagrangian

$$\sum_i \tfrac{1}{2} w_i(\mathbf{x}_i - \hat{\mathbf{x}}_i)^{\mathrm{T}}\Lambda_i^{-1}(\mathbf{x}_i - \hat{\mathbf{x}}_i) + \sum_i \lambda_i \theta^{\mathrm{T}}\mathbf{v}(\hat{\mathbf{x}}_i) + \alpha^{\mathrm{T}}\phi(\theta), \tag{3.5}$$

where $\{\lambda_i\}$ and α are the Lagrange multipliers.

To proceed further, we linearize the equations around the current estimate of $\mathbf{x}_i$'s and θ. Let $\Delta\mathbf{x}_i = \hat{\mathbf{x}}_i - \mathbf{x}_i$ and $\Delta\theta = \theta_0 - \theta$, where θ_0 is the current estimate of the true θ. To reduce notational clutter, we denote $\nabla\mathbf{v}(\mathbf{x}_i)$ by $\nabla\mathbf{v}_i$ and $\nabla\phi(\theta_0)$ by $\nabla\phi_0$. This yields the equation

$$\begin{aligned}\tfrac{1}{2}\sum_i w_i \Delta\mathbf{x}_i^{\mathrm{T}}\Lambda^{-1}\Delta\mathbf{x}_i \; + \sum_i \lambda_i \left(\theta_0^{\mathrm{T}}\mathbf{v}(\mathbf{x}_i) + \mathbf{v}(\mathbf{x}_i)^{\mathrm{T}}\Delta\theta \; + \theta_0^{\mathrm{T}}\nabla\mathbf{v}_i\Delta\mathbf{x}_i\right) \\ +\alpha^{\mathrm{T}}\left(\phi(\theta_0) + \nabla\phi_0\Delta\theta\right) = 0.\end{aligned} \tag{3.6}$$

Taking derivatives with respect to $\Delta\theta$, $\Delta\mathbf{x}_i$ and the Lagrange multipliers yields the system of equations:

$$w_i{\Lambda_i}^{-1}\Delta\mathbf{x}_i + \lambda_i\theta_0^{\mathrm{T}}\nabla\mathbf{v}_i = 0 \qquad \sum_i \lambda_i\mathbf{v}^{\mathrm{T}}(\mathbf{x}_i) + \alpha^{\mathrm{T}}\nabla\phi_0 = 0 \tag{3.7}$$

$$\theta_0^{\mathrm{T}}\mathbf{v}(\mathbf{x}_i) + \mathbf{v}(\mathbf{x}_i)^{\mathrm{T}}\Delta\theta + \theta_0^{\mathrm{T}}\nabla\mathbf{v}_i\Delta\mathbf{x}_i = 0 \qquad \phi(\theta_0) + \nabla\phi_0\Delta\theta = 0. \tag{3.8}$$

The solution to the above set of equations can be written as

$$\Delta\theta = -\phi(\theta_0)(\nabla\phi_0^{\mathrm{T}}\nabla\phi_0)^{-1}\nabla\phi_0 \tag{3.9}$$

$$\lambda_i = w_i\big(\theta_0^{\mathrm{T}}\nabla\mathbf{v}_i^{\mathrm{T}}\Lambda_i\nabla\mathbf{v}_i\theta_0\big)^{-1}\mathbf{v}(\mathbf{x}_i)^{\mathrm{T}}(\theta_0 + \Delta\theta) \tag{3.10}$$

$$\Delta\mathbf{x}_i = -\tfrac{1}{w_i}\Lambda_i\nabla\mathbf{v}_i\theta_0\lambda_i = -\Lambda_i\nabla\mathbf{v}_i\theta_0\big(\theta_0^{\mathrm{T}}\nabla\mathbf{v}_i^{\mathrm{T}}\Lambda_i\nabla\mathbf{v}_i\theta_0\big)^{-1}\mathbf{v}(\mathbf{x}_i)^{\mathrm{T}}(\theta_0 + \Delta\theta). \tag{3.11}$$

The above solutions to the linearized constrained optimization problem suggests an iterative gradient-descent algorithm in which a candidate initial value of θ_0 is computed and the values of θ and the $\hat{\mathbf{x}}_i$'s are progressively modified until the constraints are satisfied.

The initial value of θ_0 may be chosen as the result of an unconstrained optimization using the Fundamental Numerical Scheme (FNS) algorithm [27] or the related Heteroscedastic Errors in Variables (HEIV) method [75] based on solving a generalized eigenvalue problem. We detail the derivation of this initialization procedure in Section 3.4.

Constraints in the 3D case

In the case of 3D data, we consider the choice of model corresponding to an upper bound of 2 linear subspaces (planes) in each local neighborhood under consideration. This may be described formally as a zero-level set of the equation $(\gamma_1^{\mathrm{T}}\mathbf{x} + d_1)(\gamma_2^{\mathrm{T}}\mathbf{x} + d_2) = 0$, where $\mathbf{x} \in \mathbb{R}^3$ and $\gamma_i \in \mathbb{R}^3$, $d_i \in \mathbb{R}$ are the parameters for each of the two planes. Note that this again subsumes the case where the subspaces coincide. Expanding out the terms yields an inhomogeneous 2^{nd} degree polynomial in 3 variables (denoted x, y and z).

Let us denote the coefficients of each monomial in the polynomial as given by the expression

$$\theta_1x^2 + \theta_2y^2 + \theta_3z^2 + \theta_4xy + \theta_5yz + \theta_6xz + \theta_7x + \theta_8y + \theta_9z + \theta_{10} = 0. \tag{3.12}$$

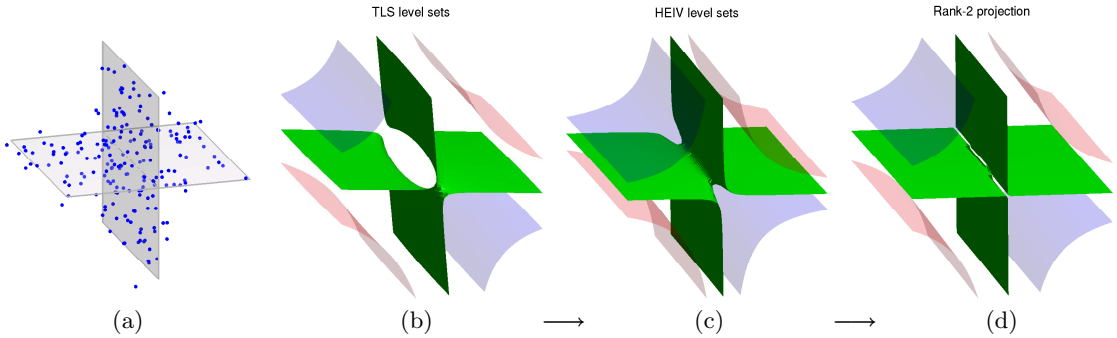


Figure 3.2: Illustration of sequence of optimization steps in an example of global fitting of (a) noisy observations of points lying on two planes. Level-set surfaces are shown at values 0 (green), 0.15 (red) and −0.15 (blue), and are drawn for parameters estimated with (b) TLS, which are used to initialize solution to the (c) HEIV estimate, which when subject to degeneracy constraints yields the best fit at the intersection of the planes as shown in (d).

This may be rewritten in matrix form as

$$\begin{bmatrix}\mathbf{x} & 1\end{bmatrix}\begin{bmatrix}2\theta_1 & \theta_4 & \theta_6 & \theta_7\\ \theta_4 & 2\theta_2 & \theta_5 & \theta_8\\ \theta_6 & \theta_5 & 2\theta_3 & \theta_9\\ \theta_7 & \theta_8 & \theta_9 & 2\theta_{10}\end{bmatrix}\begin{bmatrix}\mathbf{x}\\ 1\end{bmatrix} = \begin{bmatrix}\mathbf{x} & 1\end{bmatrix} A \begin{bmatrix}\mathbf{x}\\ 1\end{bmatrix} = 0. \tag{3.13}$$

Following the argument in Section 3.3, it is easy to see that matrix A must be of rank 2 for the associated quadric surface to represent a pair of planes. This is equivalent to the constraints that the determinant of A as well as each of its 3×3 minors are zero. We have observed it sufficient to relax the constraints on the minors and retain the constraints only on the principal minor formed by the degree 2 coefficients, as

$$\det(B) = \det\left(\begin{bmatrix}2\theta_1 & \theta_4 & \theta_6\\ \theta_4 & 2\theta_2 & \theta_5\\ \theta_6 & \theta_5 & 2\theta_3\end{bmatrix}\right) = 0. \tag{3.14}$$

Geometrically, the use of this particular subset of constraints restricts the family of surfaces represented by the polynomial coefficients to the family of parallel or intersecting planes, and cylinders. Using the parameters estimated with this subset of constraints, we may then construct the matrix A, find its rank-2 approximation using its SVD decomposition, and recover the parameters of the degenerate polynomial from the rank-2 matrix.

Figure 3.2 illustrates the sequence of steps involved in estimating the polynomial coefficients for a synthetic dataset consists of noisy points lying on two planes intersecting at right angles. Level-set surfaces are displayed for the polynomial coefficients estimated at

each step of fitting all the points. It can be seen that the TLS solution misfits the geometry, the HEIV solution tends to oversmooth the intersection (as in Figure 3.1 for 2D data) and enforcing the degeneracy constraints recovers the true geometry in this example.

3.4 Unconstrained initialization

In this section, we outline the derivation of the parameter estimation procedure when the parameters do not obey the degeneracy constraint equations $\phi(\theta) = 0$. We will restrict our attention to the Heteroscedastic Errors in Variables (HEIV) estimator [75], following a method of analysis similar to [27]. The parameters θ_{HEIV} estimated as a result of this procedure is used to initialize the algorithm derived in Section 3.3.

As before, we are given a set of points $\{\mathbf{x}_i\} \in \mathbb{R}^d$ that are assumed to be noisy observations of the positions of true points $\mathbf{x}_i^o \in \mathbb{R}^d$ that lie on a smooth surface.

The points are assumed to line on a hyperplane defined by $\theta^{\mathrm{T}}\mathbf{v}(\hat{\mathbf{x}}_i) = 0 \quad \forall i$, where $\theta \in \mathbb{R}^m$ is the vector of monomial coefficients and $\mathbf{v}(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}^m$ is the mapping from the d-dimensional point to the monomials formed by its coordinates. To condition the problem, we will also enforce the unit norm constraint $\|\theta\| = 1$ on the parameters. Our goal is to obtain the maximum likelihood solution subject to the above two types of constraints.

Such problems are typically solved by finding the stationary point of the Lagrangian

$$\arg \min_{\{\hat{\mathbf{x}}_i\},\theta,\{\lambda_i\},\alpha} \ \frac{1}{2}\sum_i w_i(\mathbf{x}_i - \hat{\mathbf{x}}_i)^{\mathrm{T}}\Lambda_i^{-1}(\mathbf{x}_i - \hat{\mathbf{x}}_i) + \sum_i \lambda_i \theta^{\mathrm{T}}\mathbf{v}(\hat{\mathbf{x}}_i) + \frac{1}{2}\alpha(\theta^{\mathrm{T}}\theta - 1). \quad (3.15)$$

Taking the derivative with respect to $\hat{\mathbf{x}}_i$ gives

$$w_i\Lambda_i^{-1}(\mathbf{x}_i - \hat{\mathbf{x}}_i) + \lambda_i \nabla\mathbf{v}(\hat{\mathbf{x}}_i)^{\mathrm{T}}\theta = 0. \quad (3.16)$$

which may be manipulated to give

$$\begin{aligned} \Rightarrow \quad & w_i\Lambda_i^{-\frac{1}{2}}(\mathbf{x}_i - \hat{\mathbf{x}}_i) = -\lambda_i\Lambda_i^{\frac{1}{2}}\nabla\mathbf{v}_i^{\mathrm{T}}\theta \\ \Rightarrow \quad & w_i^2(\mathbf{x}_i - \hat{\mathbf{x}}_i)^{\mathrm{T}}(\mathbf{x}_i - \hat{\mathbf{x}}_i) = \lambda_i^2\theta^{\mathrm{T}}\nabla\mathbf{v}_i\Lambda_i\nabla\mathbf{v}_i^{\mathrm{T}}\theta \\ \Rightarrow \quad & \lambda_i^2 = \frac{w_i^2(\mathbf{x}_i - \hat{\mathbf{x}}_i)^{\mathrm{T}}\Lambda_i^{-1}(\mathbf{x}_i - \hat{\mathbf{x}}_i)}{\theta^{\mathrm{T}}\nabla\mathbf{v}_i\Lambda_i\nabla\mathbf{v}_i^{\mathrm{T}}\theta}. \end{aligned} \quad (3.17)$$

Also,

$$\begin{aligned} & w_i(\mathbf{x}_i - \hat{\mathbf{x}}_i) = -\lambda_i\Lambda_i\nabla\mathbf{v}_i^{\mathrm{T}}\theta \\ \Rightarrow \quad & w_i\theta^{\mathrm{T}}\nabla\mathbf{v}_i(\mathbf{x}_i - \mathbf{x}_i) = \lambda_i\theta^{\mathrm{T}}\nabla\mathbf{v}_i\Lambda_i\nabla\mathbf{v}_i^{\mathrm{T}}\theta \\ \Rightarrow \quad & \lambda_i^2 = w_i^2\left(\frac{\theta^{\mathrm{T}}\nabla\mathbf{v}_i(\mathbf{x}_i - \mathbf{x}_i)}{\theta^{\mathrm{T}}\nabla\mathbf{v}_i\Lambda_i\nabla\mathbf{v}_i^{\mathrm{T}}\theta}\right)^2. \end{aligned} \quad (3.18)$$

From (3.17) and (3.18)

$$(\mathbf{x}_i - \hat{\mathbf{x}}_i)^{\mathrm{T}} \Lambda_i^{-1} (\mathbf{x}_i - \hat{\mathbf{x}}_i) = \frac{[\theta^{\mathrm{T}} \nabla \mathbf{v}_i (\mathbf{x}_i - \mathbf{x}_i)]^2}{\theta^{\mathrm{T}} \nabla \mathbf{v}_i \Lambda_i \nabla \mathbf{v}_i^{\mathrm{T}} \theta}. \tag{3.19}$$

Using the approximation that

$$\begin{aligned} \mathbf{v}(\mathbf{x}_i) &\approx \mathbf{v}(\hat{\mathbf{x}}_i) + \nabla \mathbf{v}_i (\mathbf{x}_i - \hat{\mathbf{x}}_i) \\ &= \nabla \mathbf{v}_i (\mathbf{x}_i - \hat{\mathbf{x}}_i), \end{aligned} \tag{3.20}$$

we get

$$(\mathbf{x}_i - \hat{\mathbf{x}}_i)^{\mathrm{T}} \Lambda_i^{-1} (\mathbf{x}_i - \hat{\mathbf{x}}_i) = \frac{\theta^{\mathrm{T}} \mathbf{v}(\mathbf{x}_i) \mathbf{v}(\mathbf{x}_i)^{\mathrm{T}} \theta}{\theta^{\mathrm{T}} \nabla \mathbf{v}_i \Lambda_i \nabla \mathbf{v}_i^{\mathrm{T}} \theta}. \tag{3.21}$$

Let us denote the terms A_i and B_i as

$$A_i \triangleq w_i \mathbf{v}(\mathbf{x}_i) \mathbf{v}(\mathbf{x}_i)^{\mathrm{T}} \tag{3.22}$$

$$\text{and} \quad B_i \triangleq \nabla \mathbf{v}_i^{\mathrm{T}} \Lambda_i \nabla \mathbf{v}_i. \tag{3.23}$$

Then the maximum likelihood solution for parameters θ may be obtained by substituting the likelihood function through (3.21) through solving the system

$$\arg\min_{\theta} \sum_i \frac{\theta^{\mathrm{T}} A_i \theta}{\theta^{\mathrm{T}} B_i \theta}. \tag{3.24}$$

Taking the derivative with respect to θ gives the condition

$$\frac{\mathrm{d}}{\mathrm{d}\theta} \left(\sum_i \frac{\theta^{\mathrm{T}} A_i \theta}{\theta^{\mathrm{T}} B_i \theta} \right) = \sum_i \frac{2}{(\theta^{\mathrm{T}} B_i \theta)^2} \left[(\theta^{\mathrm{T}} B_i \theta) \theta^{\mathrm{T}} A_i - (\theta^{\mathrm{T}} A_i \theta) \theta^{\mathrm{T}} B_i \right] \tag{3.25}$$

$$\Rightarrow \qquad \sum_i \frac{\theta^{\mathrm{T}} A_i}{\theta^{\mathrm{T}} B_i \theta} - \sum_i \frac{(\theta^{\mathrm{T}} A_i \theta)}{(\theta^{\mathrm{T}} B_i \theta)^2} \theta^{\mathrm{T}} B_i = 0. \tag{3.26}$$

Thus it can be seen that the parameters θ obey the generalized eigenvector equation given by

$$\left(\sum_i \frac{A_i}{\theta^{\mathrm{T}} B_i \theta} \right) \theta = \left(\sum_i \frac{(\theta^{\mathrm{T}} A_i \theta)}{(\theta^{\mathrm{T}} B_i \theta)^2} B_i \right) \theta. \tag{3.27}$$

This motivates the fixed-point iteration procedure given by:

$$S(\theta(k)) \theta(k+1) = \lambda_k C(\theta(k)) \theta(k+1), \tag{3.28}$$

Algorithm 2 Denoise points by constrained local fitting

Data: Points $X = \{\mathbf{x}_i\} \in \mathbb{R}^d$ with noise covariance $\{\Lambda_i\}$

1: **for** $\mathbf{x} \in X$ **do**
2: Compute **weights** $w_i = k(\mathbf{x} - \mathbf{x}_i)$ where k is a loss function such as a Gaussian.
3: Find the **total least squares solution** θ_{TLS} to the unconstrained fitting problem. The least square solution is simply equal to the minimal eigenvector of the weighted covariance matrix formed by the $\mathbf{v}(\mathbf{x}_i)$'s, i.e. the minimal eigenvector of $\sum_i w_i \mathbf{v}(\mathbf{x}_i)\mathbf{v}(\mathbf{x}_i)^{\text{T}}$.
4: Use θ_{TLS} to initialize the iterative solution to an **unconstrained optimization** procedure [75]. The solution to the unconstrained problem θ_{HEIV} can be obtained through an fixed-point iteration procedure given by (3.28).
5: Iteratively enforce the **degeneracy constraint** (3.3) using equations (3.9) (or (3.14) in the case of 3D) along with the unit norm constraint $\|\theta\| = 1$ and initializing with θ_{HEIV}.
6: Compute the new estimate of denoised point locations $\{\hat{\mathbf{x}}_i\}$ using (3.11).
7: **end for**

where λ_k is the smallest generalized eigenvalue, and S and C are given by:

$$S(\theta) = \sum_i \frac{A_i}{\theta^{\text{T}} B_i \theta} \qquad C(\theta) = \sum_i B_i \frac{\theta^{\text{T}} A_i \theta}{(\theta^{\text{T}} B_i \theta)^2}, \tag{3.29}$$

with $A_i = w_i \mathbf{v}(\mathbf{x}_i)\mathbf{v}(\mathbf{x}_i)^{\text{T}}$ and $B_i = \nabla \mathbf{v}_i^{\text{T}} \Lambda_i \nabla \mathbf{v}_i$

A few observations are worth noting. First, our formulation is based on maximum likelihood and so easily allows the incorporation of knowledge of heteroscedastic noise models of the points. In contrast, GPCA does not consider noise in the points and are hence susceptible to instability in coefficients estimated the polynomial fitting. Second, unlike [113], there is no need to estimate the individual subspaces or the association of each point to the subspaces. Hence, in that respect, this treatment of denoising a sharp intersection is no different procedurally than from a smooth parameterized surface. Lastly, our focus is on *local* rather than global fitting of the data, since the data in our application cannot necessary be described globally by linear subspaces.

3.5 Algorithm and Implementation

From the solution of the constrained optimization problem in the previous section, we may construct our denoising procedure as given in Algorithm 2. We draw attention to some details that influence the performance of the proposed method.

Support radius: The choice of support radius used to compute the weights w_i in the kernel function influences the algorithm in two ways:

(i) First, the proposed method assumes an upper bound of 2 subspaces in the volume of interest, which need not be the case for every choice of support size. The chosen support radius must be one for which the modeling assumption is valid.

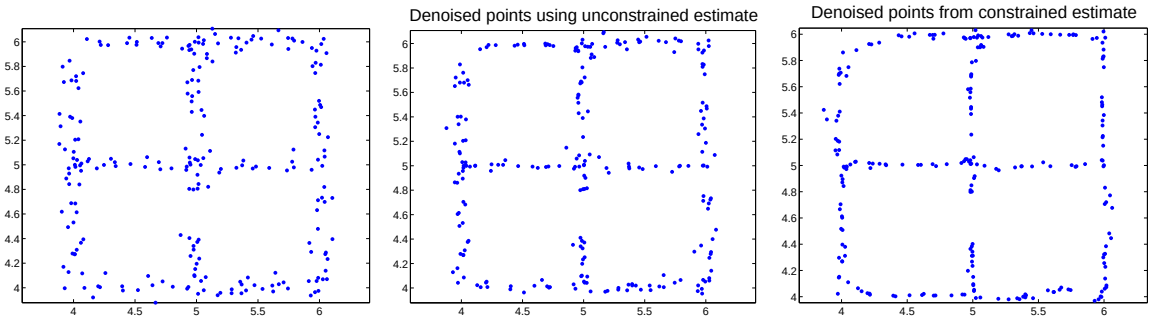
We propose to treat the first problem as one of model selection – specifically of choosing between fitting an edge or a smooth surface to the points. This is done easily within the procedure of Algorithm 3.5 as follows. At the end of STEP 4, we compute the denoised locations $\hat{\mathbf{x}'}_i$ using (3.11) and $\theta = \theta_{\text{HEIV}}$, and we store them. Then, at the end of STEP 6, we compare the likelihood computed using (3.4) with both the point locations $\hat{\mathbf{x}'}_i$ from the HEIV estimate, as well as with those with the degenerate model parameters. If the likelihood of the latter is less than the former, we do not update the HEIV estimate before proceeding to the next iteration.

(ii) Secondly, even when the assumption of number of subspaces is valid, there is a tradeoff between choosing too small a radius, risking poor estimates due to the fewer number of points, or too large a radius, risking the unfavorable influence of points that do not belong to the local model.

This problem may be solved using the technique of local semi-parametric analysis from the previous chapter. The local parameterization is given by the geometry model of (2.40) which was used in Section 2.4 for surface fitting. The only remaining point of difference between the previous chapter and this one, is in the equation for matrix perturbation bounds. Because this chapter addresses heteroscedastic noise, the resulting formulation requires the solution of a *generalized* eigenvector problem unlike the ordinary eigenvector problem of the last chapter. Appendix B briefly outlines the theory of eigenvector perturbation relevant to the generalized eigenvector problem.

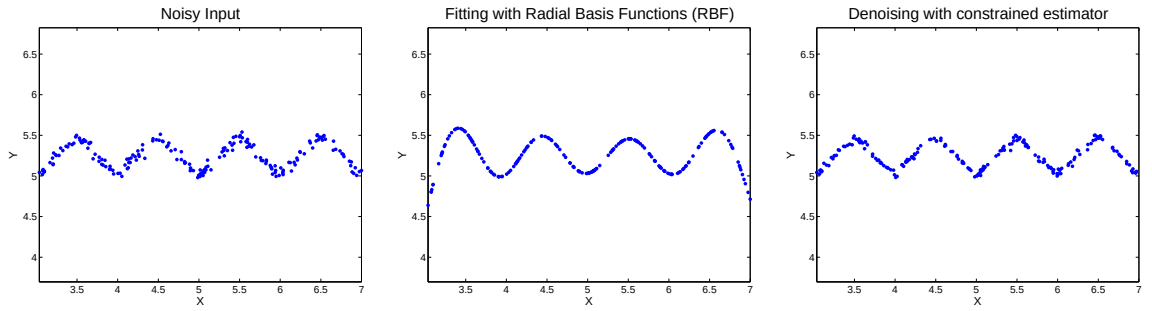
In the results presented in this chapter, we intentionally choose the heuristic strategy of choosing the support radius that gives the best fit, in a maximum likelihood sense, to the corresponding neighborhood of the interest point, excluding the point itself to prevent the trivial solution of zero radius. This is done to allow a fairer comparison between the proposed method and other methods such as radial basis functions, such as shown in Figure 3.4. In practice, we have observed that when the number of manifolds is under- or over-estimated, this strategy desirably tends to reduce the support radius and choose a one-manifold solution when enforcing the degeneracy constraint.

Robustness: The use of weights w_i also suggests the use of robust statistics to identify outliers to the model [97]. One strategy to identify points that have a large influence on the estimated model parameters, such as using eigenvector perturbation bounds [110] for the generalized eigenvalue problem (3.28) or using influence functions. In our experiments,



(a) (b) (c)

Figure 3.3: Example of denoising a toy dataset by *local* fitting of an implicit degenerate polynomial (a) Input data consisting of points from six line segments corrupted with spherical Gaussian noise of standard deviation $\sigma = 0.5$ (b) (b) Denoised data using an implicit quadratic fit with the HEIV estimator [75]. (c) Denoised output after imposing degeneracy constraints on coefficients.



(a) (b) (c)

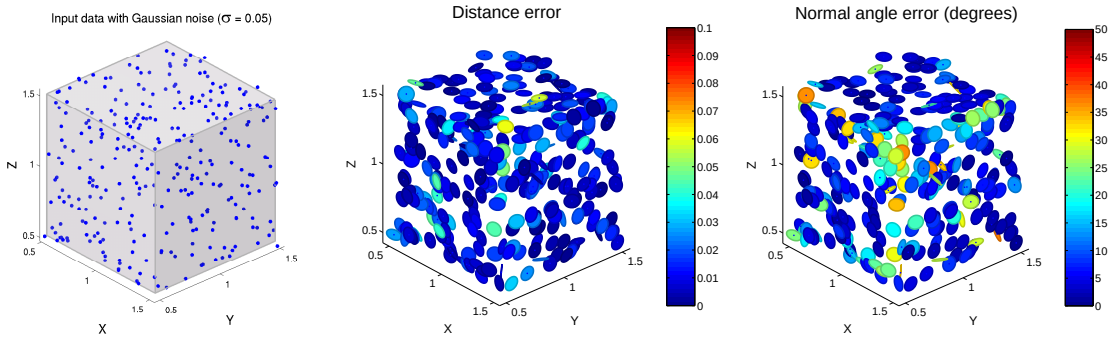
Figure 3.4: Example of denoising samples from a triangular wave function (a) Input data corrupted with spherical Gaussian noise of standard deviation $\sigma = 0.5$ (b) Denoised data using radial basis function based smoother with Gaussian kernel. (c) Denoised output after *local* degenerate polynomial fitting.

we use a simple greedy strategy of evaluating leave-one-out fitting score and ignoring the point as an outlier if it is not a good fit with its neighbors.

3.6 Experiments

Evaluation

We performed a series of controlled experiments of synthetic data in known configurations to evaluate the behavior of the denoising algorithm. Figure 3.3 shows an example where an Epanechnikov loss function with bandwidth 0.3 was used to denoise a 2x2 square grid pattern of points. The use of a constraint enforcing degeneracy in the polynomial can be



(a) (b) (c)

Figure 3.5: Example of denoising samples from 3 faces of a regular cube (a) Input data corrupted with uniform noise of standard deviation $\sigma = 0.02$. Denoised points are shown with patches color coded by (b) distance error and (c) surface normal angle error.

seen to preserve the intersections better than using an HEIV smoother.

Figure 3.4 compares the proposed fitting procedure with a standard interpolation algorithm based on radial basis functions (RBF). The RBF algorithm has two parameters [120]. The first controls the width of the Gaussian kernel which influences the locality of the smoothing. The other controls the tolerance to fitting error, i.e. a value of zero would lead to interpolation between the points, while higher values allow greater fitting error. The parameters were tuned so that the results best matched the ground-truth in the sense of least-square error. It can be seen that the proposed algorithm does a better job of preserving sharp changes in the function and is more stable at the boundary, while the RBF function tends smooths over the high curvature regions.

In Figure 3.5, we test the proposed algorithm on 300 noisy 3D samples (spherical Gaussian with standard deviation 0.05) from 3 faces of a unit cube, and compared it against using the HEIV estimator from [75]. The use of the proposed estimator reduced the minimum error in normal angle over the dataset from 0.67° to 0.46° and the median distance of the points to their corresponding planes from 0.012 to 0.009 units.

3.7 Concluding remarks

To conclude this part of the document, we summarize the claims implied by the results in the preceding chapters as follows:

(i) Many useful tasks in point cloud processing, such as surface reconstruction and noise removal, involve reasoning about unknown underlying shape and can be posed as problems of local geometric fitting. The problem of local geometric fitting of smooth surfaces in turn may be converted into the form of a statistical inference problem

through use of a local shape parameterization combined with a semi-parametric point cloud distribution model.

(ii) Sharp geometric features, such as surface edges or intersections of curves, may be modeled as degenerate parameterizations of smooth surfaces. One significant advantage of this strategy is the ability to model non-manifold regions where differential analysis is not normally possible, and do this without actually having to estimate the parameters of the component manifolds.

(iii) The introduction of a semi-parametric distribution model allows one to perform asymptotic analysis of the estimator and so determine whether the computed model parameters converge to their true quantities for increasing number of data samples. In addition, through perturbation analysis around the asymptotic estimate, one can also obtain error bounds on the value of the estimate that are valid for finite samples.

(iv) By expressing the error bounds in terms of the local analytical scale, it is possible effectively adapt the support radius in local fitting to the unknown underlying geometry, noise level and sample density. This thus provides a solution to the scale-selection problem that is ubiquitous in regression

The next part of the document will look at a problem that is complementary to geometric fitting - that of analyzing shape from unorganized point samples to detect interest regions at multiple scales. This will allow us to build region-based shape models in a manner that is robust to changes in the way the points are sampled from the shape.

Part II

Multi-scale Analysis

CHAPTER 4

Multi-scale Signal Representation

Many approaches to solving vision problems such as registration and object detection, both in the 2D and 3D domains, work with compact intermediate representations of the input data. In the 2D image domain, it is common practice [72] to start by automatically finding a set of distinguished points and associated neighborhoods in the image, following which descriptors are computed in each region. This strategy of using local patches has proven to be well-suited to take on the practical challenges of occlusion, clutter and intra-class variation. The success of these local representations in image processing has been driven in large part by developments in building multi-scale representations of images [70] popularized by their more recent application to finding scale-invariant interest regions [72, 79].

When the input is geometric data in the form of an unorganized 3D point cloud, the analogous interest region detection methods currently in use seem almost primitive, if not relatively unprincipled in comparison. For problems such as object recognition and scan registration, current practice [40, 76] is to compute a descriptor either *exhaustively* at each point or at *randomly* or uniformly *distributed* locations in the point cloud. Furthermore, the descriptor is computed over a support radius that is usually preset to one or more values based largely on intuition, instead of being guided by the available data. This heuristic selection process is followed irrespective of the choice of descriptor, whether it be the spin image [53], 3D shape context [15], harmonic shape contexts [40, 57] or something else. To compensate for the increase in redundancy that this process introduces, some practitioners opt to perform a post-processing step of clustering the descriptors [76] and reducing the selection to only the cluster centers.

Why is there such a huge difference between approaches for processing 2D images and

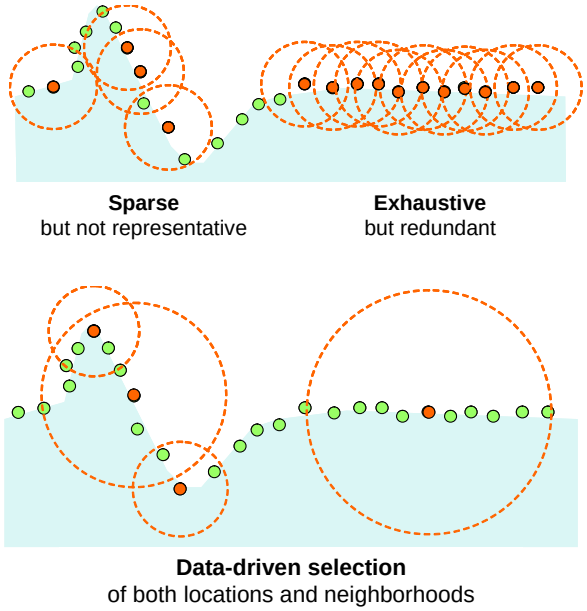


Figure 4.1: Random selection of locations for computing shape descriptors can miss geometric structures whose spatial extent are not comparable to the preset choice of support radius. Exhaustive selection strategies introduce redundancy in areas with little shape variation. A data-driven approach can judiciously choose both locations and associated support radius if guided by local shape.

3D point clouds?

The main reason for this inconsistency is that while a rigorous theory of scale selection exists for images [52, 70], a direct equivalent does not really exist for unorganized point clouds. Scale theory for images has focused largely on analyzing functions observed on a regular 2D (or even n-D) lattice. However point clouds lack any organized lattice structure. Unlike image intensity, the observed geometry "signal" in point clouds is implicit and is represented only through the spatial arrangement of the observed locations. Thus the applicability of traditional scale theory for images to discrete point cloud data is unclear.

This chapter addresses the above inconsistency through two contributions. First, we derive multi-scale filtering operator for point clouds that captures variation in shape at a point relative to its neighborhood, working analogously to the Laplacian filter in 2D images, but while enjoying the property of being invariant to changes in sampling density around that point.

Second, we show how this filter may be used in an algorithm for interest region detection working *directly* in the input point cloud domain and *without* relying on polygonal meshes or accurate surface normals. We wish to emphasize that our goal is not to develop a new 3D shape descriptor, but to provide a principled way of *repeatably* selecting regions over which

descriptors should be computed. By this, we enable a change in current practice from using random locations and preset neighborhood sizes to using a completely data-driven strategy that finds locations and their associated support radii automatically.

4.1 Related work

To distinguish our proposed approach from other related methods, it is necessary to understand the exact nature in which other methods draw connections to scale-space axioms from the 2D image domain. It is also useful to identify cases where these interpretations are insufficient or even incorrect for point clouds. To do this, we begin by summarizing the development of scale theory for images and mention its salient points.

Scale theory in 2D images. The beginning of scale theory in images is usually attributed to Witkin [122], who proposed a scale space representation as a transformation of an input signal $f(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}$ to a one-parameter family of signals $f(\mathbf{x}, t) : \mathbb{R}^d \times \mathbb{R}^+ \to \mathbb{R}$, where the non-negative t denotes the *scale parameter*. Witkin obtained a *Gaussian scale space* representation by convolving $f(\mathbf{x})$ with Gaussians of increasing width as $f(\mathbf{x}, t) = f(\mathbf{x}) * G(\mathbf{x}, t)$ where $G(\mathbf{x}, t) = (2\pi t^2)^{-1/2} \exp\left(-\|\mathbf{x}\|^2/2t^2\right)$ and the asterisk denotes convolution.

Koenderink [61] pointed out the connection between Gaussian scale space and the *diffusion equation*

$$\frac{\partial f(\mathbf{x}, t)}{\partial t} = \Delta f(\mathbf{x}, t) \equiv \sum_i^d \frac{\partial f(\mathbf{x}, t)}{\partial x_i x_i}, \tag{4.1}$$

where Δ denotes the *Laplacian* operator. When subject to the initial condition $f(\mathbf{x}, 0) = f(\mathbf{x})$, the solution of (4.1) can be shown to be Gaussian convolution with G as the unique Green's function for the above system.

Lindeberg [70] later showed that the amplitude of scale-normalized derivative operators assumes a unique maximum for a value of t proportional to the "wavelength" of the signal at that point. This property provides a way to choose feature support regions in proportion to this characteristic length of the signal.

Lowe [72] proposed to linearize the diffusion equation with a simple forward explicit Euler scheme so that successive convolutions of the signal may be obtained as

$$f(\mathbf{x}, t + \delta t) = f(\mathbf{x}, t) + \delta t \Delta f(\mathbf{x}, t). \tag{4.2}$$

Although the semi-group property of Gaussian convolution makes the above linearization unnecessary, this forward scheme also implies that the *scale-normalized Laplacian* $t^2 \Delta f(\mathbf{x}, t)$ of the signal can be numerically approximated by taking the difference of two Gaussian filtered signals with the Gaussians differing in scale by a constant factor.

3D Mesh processing. Several methods have been proposed to extend the above connection of 2D scale space with the diffusion equation to surfaces represented by a *connectivity mesh*. Seminal work by Taubin [106] replaced the continuous Laplacian operator Δ from the diffusion equation by its discrete counterpart, the *graph Laplacian* L_g. For a set of sample points $\{\mathbf{x}_i\}$ forming vertices of a graph, the unnormalized graph Laplacian is defined as the operator over a function f on the points as

$$L_g f(\mathbf{x}_i) = \sum_{j,\{i,j\}\in\mathbb{E}} (f(\mathbf{x}_i) - f(\mathbf{x}_j))w_{ij}, \tag{4.3}$$

where the summation is over graph edges (i, j) in edge set $\mathbb{E}$, and w_{ij} are positive edge weights.

By using the graph Laplacian L_g and replacing the scalar image intensity function $f(\mathbf{x})$ in (4.2) by the Euclidean coordinate vector $\mathbf{x}$, one can progressively "smooth" the 3D points by successive application of (4.2). Closely related work by Kobbelt [59] considered similar discrete approximations of the Laplacian for the task of interpolatory mesh subdivision.

Two limitations of the above approaches are that (1) they are relatively slow due to the restriction placed by the choice of discretization, and more importantly that (2) the discretization that was employed assumes the presence of uniformly sampled data over the surface, hence generating geometric smoothing artifacts when that assumption was violated. Work by Desbrun *et al.* [30] attempted to correct these deficiencies by using a backward Euler discretization of the corresponding diffusion equation, and a modified curvature estimator to compensate for unequal mesh triangulation. The smoothing process was then carried out using curvature-based flow, and gave good results even on non-uniformly sampled datasets. However, the methods worked on the assumption that a connectivity mesh was available to compute the required normalization factors, making them unsuitable for unorganized data points.

Our main criticism of this line of work in mesh processing is that, by modifying the 3D points directly, we believe it is trying to solve a different problem. Altering the extrinsic geometry as part of a multi-scale representation is akin to changing not only the pixel values but also the pixel coordinates in images, and incorrectly changes the observations. This different problem of mesh simplification, while still useful for tasks like progressive rendering and compression, is not directly applicable to multi-scale interest region detection.

On that note, many researchers have observed [30, 96, 106] that the resulting Gaussian flow or mean curvature flow from (4.2) alters the original geometry of the data in undesirable ways through global volume change and fragmentation. Thus, using these "simplified" meshes or point clouds as part of a larger system is not straightforward.

Other relevant work. Pauly *et al.* [87] measure a quantity termed *surface variation*,

given by $\sigma_n(x) = \lambda_0/(\lambda_0 + \lambda_1 + \lambda_2)$, where the λ_i's are eigenvalues of the sample covariance matrix computed in a local n-point neighborhood of sample point x. They propose the natural scale at a point to be the neighborhood size for which the corresponding σ_n achieves a local extremum. However the variation in σ_n is extremely noisy and heuristic pre-smoothing procedures have to be applied in order to recover any trends in its variation.

Recent work [83, 84] presented a method to detect multi-scale corner and edge features by constructing a global 2D parameterization of the data and filtering surface normals in the 2D space. The approach relies strongly on having a connectivity mesh to faithfully construct the 2D parameterization, and also on the prior availability of good surface normals, as does [68]. Also none of the above methods come with any guarantee that variability in the distribution of the point samples will not affect the result of the proposed algorithm.

4.2 Multi-scale operators for point clouds

From the overview of related work in the previous section, we may conclude that discretization of the 2D diffusion equation is not the appropriate starting point for developing scale-space analogies in unorganized point clouds. In this section we will pursue a different line of reasoning to develop multi-scale operators while preserving the advantages of working in the original input space and not relying on good initial estimates of surface normals or meshes. We will build on these operators in Section 4.3 to devise an algorithm for estimating at each point a natural scale that is representative of its local shape. Results using this algorithm are presented in Section 4.4 followed by discussion of the approach in Section 4.5.

As was first reported by Weickert [119], Gaussian scale space theory was axiomatically derived by Iijima [52] as far back as 1959. Iijima's starting point [119] was to define the mathematical form of an *integral operator* that maps input functions to their one-parameter multi-scale representations. We will follow this route by first defining the form of such an operator for continuous surfaces, and then progressively modify the operator to satisfy the various requirements of our problem domain. In particular, our focus will be on determining not point locations but *values* defined on the points that will exhibit interesting properties independent of the sampling distribution generating those points.

Case of 2D curves

We start by considering the integral operator $A : \mathbb{R}^d \times \mathbb{R}^+ \rightarrow \mathbb{R}^d$, with $d = 2$ for the 2D case. A is defined to have the form

$$A(\alpha(s), t) = \int_{\Gamma} \phi(s, u, t)\alpha(u)du \tag{4.4}$$

that operates on positions on a *smooth* 2D curve Γ parameterized by distance as the function $\alpha(s) : \mathbb{R} \to \mathbb{R}^2$. Note that (i) the form of the transformation is linear, and (ii) the integration is performed *along* the curve Γ and not in the Euclidean $\mathbb{R}^2$ space. The implication of the latter is that, by defining the operator in terms of intrinsic geometry, there is no reference to an extrinsic coordinate system. Thus the operator is invariant to rigid transformations in the 2D space.

For the operator to be translation invariant in the 1D intrinsic system, the kernel ϕ must be representable in the form $\phi(s-u,t)$. We fix the kernel to be a Gaussian

$$\phi(s,u,t) = (2\pi t^2)^{-\frac{1}{2}} \exp\left(-\frac{(s-u)^2}{2t^2}\right), \tag{4.5}$$

so that $\int_{-\infty}^{\infty} \phi(s,u,t)du = 1$.

How does the above operator relate to the geometry of the curve? To answer this, let us fix a local coordinate system at a point $\mathbf{x} = \alpha(0)$, for convenience, and examine the effect of the operator at the point. By Taylor expansion of $\alpha(u)$ around $u = 0$,

$$\alpha(u) = \mathbf{x} + u\dot{\alpha}(0) + \tfrac{u^2}{2}\ddot{\alpha}(0) + \ldots,$$

we can derive

$$\begin{aligned} A(\mathbf{x},t) &= \int_{-\infty}^{\infty} (2\pi t^2)^{-\frac{1}{2}} e^{-\frac{u^2}{2t^2}} \alpha(u) du \\ &\approx \mathbf{x} + (2\pi t^2)^{-\frac{1}{2}} \int_{-\infty}^{\infty} \tfrac{u^2}{2} e^{-\frac{u^2}{2t^2}} \ddot{\alpha}(0) du \\ &= \mathbf{x} + \ddot{\alpha}(0)\tfrac{t^2}{2} = \alpha(0) + \kappa_{\mathbf{x}}\mathbf{n}_{\mathbf{x}}\tfrac{t^2}{2}. \end{aligned} \tag{4.6}$$

Hence the effect of the operator is to displace the point $\mathbf{x}$ in direction of the normal $\mathbf{n}_{\mathbf{x}}$ to the curve, and in proportion to its curvature $\kappa_{\mathbf{x}}$.

Extension to 3D surfaces

Using the previous result, it is straightforward to extend the operator from (4.4) to the case of a surface in 3D. Consider now the neighborhood around the origin $\mathbf{x}$ on a surface $\mathcal{M}$ having normal direction $\mathbf{n}_{\mathbf{x}}$. There then exists a family of planes Π_φ that all contain $\mathbf{n}_{\mathbf{x}}$ and whose normals lie in the tangent space of $\mathbf{x}$ at angle φ to some reference tangent. The intersection of each plane Π_φ with the surface $\mathcal{M}$ is a *normal curve* Γ_φ parameterized by the function $\alpha_\varphi(0)$.

Using the fact that the normal to each curve $\alpha_\varphi(0)$ at $\mathbf{x}$ is $\mathbf{n}_{\mathbf{x}}$, we can deduce for each normal curve that

$$A(\mathbf{x},t) = A(\alpha_\varphi(0),t) \approx \mathbf{x} + \kappa_{\mathbf{x},\varphi}\mathbf{n}_{\mathbf{x}}\tfrac{t^2}{2} \ ,$$

where $\kappa_{\mathbf{x},\varphi}$ is the *normal curvature* associated with tangent direction φ. we can average over all tangent vector directions φ to get

$$A(\mathbf{x},t) \approx \mathbf{x} + \tfrac{1}{2\pi}\int \kappa_{\mathbf{x},\varphi}\mathbf{n}_{\mathbf{x}}t^2 d\varphi = \mathbf{x} + H\mathbf{n}_{\mathbf{x}}t^2 \ , \tag{4.7}$$

where H is the *mean curvature* at the origin.[1]

Any two orthogonal tangent directions at angles φ and $\varphi + \pi/2$ may be used to form a local coordinate system at $\mathbf{x}$. Using the property of mean curvature on surfaces $H = (\kappa_\varphi + \kappa_{\varphi+\pi/2})/2$ gives

$$A(\mathbf{x},t) \approx \mathbf{x} + \left[\ddot{\alpha}_\varphi(0) + \ddot{\alpha}_{\varphi+\frac{\pi}{2}}(0)\right]\frac{t^2}{2} = \mathbf{x} + \tfrac{t^2}{2}L_{\mathcal{M}}\mathbf{x} \ ,$$

where $L_{\mathcal{M}}$ is the **Laplace-Beltrami (LB) operator** [28, 93]. The LB operator is the natural analogue of the Laplacian Δ from Euclidean space, but operates in an *intrinsic* coordinate system defined on a manifold. The modified interpretation of the A operator is that of displacing a point along its normal direction $\mathbf{n}$ in proportion to its *mean curvature* H.

Non-uniform sampling

So far, our continuous domain analysis was done under the assumption that the points on the curve or surface were uniformly distributed. In reality, the nature of sensor geometry induces a variation in sampling density that needs to be accounted for. Our method will follow the same structure as Lafon's [62] analysis of the discrete Laplacian operator, with the important differences that our chosen operator $A(\alpha(s),t)$ integrates over the sub-manifold instead of in Euclidean $\mathbb{R}^3$ space, and is analyzed as specifically applied to the extrinsic surface coordinate function $\alpha(s)$.

We consider again from Section 4.2 the case where points are sampled from the 2D curve $\Gamma = \alpha(s)$ but now follow an unknown but smooth probability distribution $p(s)$. The expected value of the operator A may then be obtained by integrating over the modified measure $\mu(s) = p(s)ds^2$, so that

$$A(\alpha(0),t) = \frac{1}{d(0,t)}\int_{-\infty}^{\infty}(2\pi t^2)^{-\frac{1}{2}}e^{-\frac{u^2}{2t^2}}\alpha(u)p(u)du \ , \tag{4.8}$$

where the normalization factor

$$d(0,t) = \int_{-\infty}^{\infty}(2\pi t^2)^{-\frac{1}{2}}e^{-\frac{u^2}{2t^2}}p(u)du \tag{4.9}$$

[1] It should be clear that mean curvature H varies with location $\mathbf{x}$, and an explicit subscript is omitted for clarity.

[2] Another approach is to integrate over the modified volume element $\sqrt{\det g}ds$ as done in Hein [46].

ensures the effective kernel weights sum to 1.

Linearizing the now combined function $\alpha(s)p(s)$ as before yields

$$\begin{aligned} A(\alpha(0),t) &\approx \left[p(0)+\ddot{p}(0)\tfrac{t^2}{2}\right]^{-1}\Big[p(0)\alpha(0) \\ &\quad + \tfrac{t^2}{2}\left[p(0)\ddot{\alpha}(0)+2\dot{\alpha}(0)\dot{p}(0)+\alpha(0)\ddot{p}(0)\right]\Big] \\ &\approx \alpha(0)+\tfrac{t^2}{2}\ddot{\alpha}(0)+\tfrac{t^2}{2p(0)}\Big[2\dot{\alpha}(0)\dot{p}(0)+\alpha(0)\ddot{p}(0)\Big]. \end{aligned} \tag{4.10}$$

Comparing with (4.6), it can be seen that effect of the variation in sampling density $p(s)$ is felt through the additional last term in (4.10) that corrupts the estimate of the curvature vector. A similar expression may be obtained for surfaces.

Invariance to sampling distribution. Lafon [62], followed by Hein [46] and others proposed to remove this additional term in integral operators through modification of the kernel function ϕ through an estimate of the density $p(s)$. In particular, consider

$$\tilde{\phi}(s,u,t) = \frac{\phi(s,u,t)}{p_t(s)p_t(u)}\ , \tag{4.11}$$

where $p_t(s)$ is the kernel density estimate at s with kernel bandwidth t as

$$p_t(s) = \int \phi(s,u,t)p(u)du. \tag{4.12}$$

Note that the above continuous domain expression for $p_t(s)$ must be approximated in practice with finite samples as $\sum_i \phi(s,u_i)$ where $\mathbf{x}_i = \alpha(u_i)$. We now show that the above modification of (4.11) eliminates the dependence on the sampling distribution.

We know from linearizing $p(u)$ around s that

$$p_t(s) = \int (2\pi t^2)^{-\frac{1}{2}} e^{-\frac{(s-u)^2}{2t^2}} p(u)du \ \approx\ p(s)+\ddot{p}(s)\tfrac{t^2}{2}.$$

Therefore

$$\begin{aligned} &\int_{-\infty}^{\infty} \tilde{\phi}(s,u,t)\alpha(u)p(u)du \\ &\quad\approx \tfrac{(2\pi t^2)^{-\frac{1}{2}}}{p_t(s)} \int_{-\infty}^{\infty} e^{-\frac{(s-u)^2}{2t^2}} \alpha(u)\left[1-\tfrac{\ddot{p}(u)}{p(u)}\tfrac{t^2}{2}\right]du \\ &\quad\approx \tfrac{1}{p_t(s)}\left[\alpha(s)+\tfrac{t^2}{2}\ddot{\alpha}(s)-\tfrac{\ddot{p}(s)}{p(s)}\tfrac{t^2}{2}\right], \end{aligned} \tag{4.13}$$

and the normalizing factor transforms to

$$\tilde{d}(s,t) = \int_{-\infty}^{\infty} \tilde{\phi}(s,u,t)p(u)du \approx \tfrac{1}{p_t(s)}\left[1-\tfrac{\ddot{p}(s)}{p(s)}\tfrac{t^2}{2}\right]. \tag{4.14}$$

Dividing (4.13) by (4.14) gives

$$\begin{aligned}\tilde{A}(\alpha(s),t) &= \tfrac{1}{\tilde{d}(s,t)}\int_{\Gamma}\tilde{\phi}(s,u,t)\alpha(u)du \\ &\approx \alpha(s)+\tfrac{t^2}{2}\ddot{\alpha}(s).\end{aligned} \tag{4.15}$$

Thus the use of the density normalized kernel $\tilde{\phi}$ removes the dependence of the result on variations in sampling density $p(s)$. A similar result can be obtained for surfaces, with the $\ddot{\alpha}(s)$ term replaced by the mean curvature normal $H\mathbf{n_p}$.

4.3 A scale-selection algorithm

In order to try and define what a *characteristic scale* at a point may be, it is useful to recall its analogy in 2D intensity images. A pixel location is considered salient or "interesting" by virtue of the distinctiveness of its intensity relative to that of its neighboring pixels. A change in the definition of this neighborhood can make the pixel seem less or more salient. Thus a point in a periodic intensity pattern of monotone frequency is most distinctive relative to a neighborhood of radius equal to its wavelength.

In our problem since the available information is of object shape, the distinctiveness at a point may be captured through the *variation in shape* at that point relative to its neighborhood. For example, on a perfect plane or sphere no point is salient with respect to its neighbors as the local shape is identical for any choice of neighborhood. Consider the addition of a small "bump" caused by perturbing the surface of a sphere. A point on the bump is now salient due its increased shape variation with respect to the sphere. However, relative to a neighborhood much larger than the spatial extent of the bump, the perturbation is not significant.

Scale-space extrema

We propose to capture this *locality* in shape variation by inspecting the variation in $\tilde{A}(\mathbf{x},t)$ as a function of the size of the neighborhood t used to estimate it. The relationship between the $\tilde{A}(\mathbf{x},t)$ operator and the mean curvature suggests a way to do this, and also provides a simple interpretation for the case of constant curvature surfaces.

We know from (4.7) and the derivation in previous section with the density normalized kernel that

$$\|\mathbf{x}-\tilde{A}(\mathbf{x},t)\| = H\tfrac{t^2}{2}\sqrt{\mathbf{n}_{\mathbf{x}}^{\mathrm{T}}\mathbf{n}_{\mathbf{x}}} = H\tfrac{t^2}{2} \tag{4.16}$$

or that the norm of the shift induced by the $\tilde{A}$ operator is proportional to the mean curvature.

Consider the scalar function F formed by exponential damping of the above expression

$$F(\mathbf{x}, t) = \frac{2\|\mathbf{x} - \tilde{A}(\mathbf{x}, t)\|}{t} e^{-\frac{2\|\mathbf{x}-\tilde{A}(\mathbf{x},t)\|}{t}}. \tag{4.17}$$

Differentiating (4.17) with respect to the scale parameter t and using (4.16) gives

$$\frac{\partial F(\mathbf{x}, t)}{\partial t} = \frac{\partial}{\partial t}(Hte^{-Ht}) = He^{-Ht}(1 - Ht),$$

which achieves a maximum at $t_{\max} = \frac{1}{H}$ for $H \neq 0$.

Thus the characteristic scale may be defined in the case of a perfect sphere or plane simply as its radius of curvature, in direct analogy to the wavelength for monotone intensity patterns in images. In the case where the shape is not of constant mean curvature, $t_{\max}$ loses its simple interpretation, as is true of characteristic scale in images for non-monotone frequencies [70]. In the previous example of the perturbation on a sphere, the local shape at a point is a composition of two constant curvature surfaces, and there will exist two characteristic scales reflecting the two components.

Implementation

Algorithm 3 outlines the procedure based on the extrema property of (4.17). The set of scales $\{t_k\}$ were fixed as $t_k = t_0(1.6)^k$, where t_0 is a base scale. We draw attention to a few implementation details below.

Graph construction. (Step 1) Since the integral operator A requires knowledge of geodesic distances between each point pair, we require a way to compute them from the few observed points. A common strategy is to construct a sparse Euclidean graph on the points and approximate the geodesic distances by distance along the shortest path in the graph. We choose to use disjoint minimum spanning trees (DMST) [24], although other valid constructions include ϵ-graphs and mutual k-nearest neighbor graphs. The construction using DMSTs has some desirable properties over traditional k-nearest neighbor or ϵ-ball schemes. In practice, it produces connected graphs without undesirable gaps and does not induce edges to clump together in noisy regions having relatively higher point density.

Region extraction (Step 10) for an extremum detected at scale t is currently done by grouping together points that are within a graph distance proportional to t from the extremum point. This is analogous to the use of a multiple of the scale σ as the patch radius in 2D images [79].

Algorithm 3 Scale selection algorithm

Data: Points $X = \{\mathbf{x}_i\} \in \mathbb{R}^3$ with $i = 1 \ldots n$, and a set of scales $T = \{t_k\}$ to be considered.

1: Construct a graph $\mathbb{G}$ on the points from which approximate geodesic distances may be computed as graph path distances. Distance $d_{\mathbb{G}}(x_i, x_j)$ between any pair of points $\mathbf{x}_i, \mathbf{x}_j$ can be computed efficiently using Dijkstra's algorithm.

2: **for** $t \in \{t_k\}$ **do**

3: **for** $x \in \{\mathbf{x}_i\}$ **do**

4: Compute estimate of density

$$p_t(\mathbf{x}_i) = \sum_j \phi(\mathbf{x}_i, \mathbf{x}_j, t),$$

$$\text{where} \quad \phi(\mathbf{x}_i, \mathbf{x}_j, t) = (2\pi t^2)^{-\frac{1}{2}} \exp\left(-\tfrac{d^2_{\mathbb{G}}(\mathbf{x}_i, \mathbf{x}_j)}{2t^2}\right).$$

5: Compute weights for each pair (i, j) as

$$\tilde{\phi}(\mathbf{x}_i, \mathbf{x}_j, t) = \phi(\mathbf{x}_i, \mathbf{x}_j, t) / \left[p_t(\mathbf{x}_i) p_t(\mathbf{x}_j)\right].$$

6: Evaluate the operator $\tilde{A}$

$$\tilde{A}(\mathbf{x}_i, t) = \tfrac{\sum_j \tilde{\phi}(\mathbf{x}_i, \mathbf{x}_j, t) \mathbf{x}_j}{\sum_j \tilde{\phi}(\mathbf{x}_i, \mathbf{x}_j, t)}.$$

7: Compute the invariant F

$$F(\mathbf{x}_i, t) = \tfrac{2\|\tilde{A}(\mathbf{x}_i, t) - \mathbf{x}_i\|}{t} \exp\left(-\tfrac{2\|\tilde{A}(\mathbf{x}_i, t) - \mathbf{x}_i\|}{t}\right).$$

8: **end for**

9: Declare interest points to be those having extremum values of $F(\mathbf{x}_i, t_k)$ both in a geodesic neighborhood of radius t_k as well as over a range of scales (t_{k-1}, t_{k+1}).

10: Designate points in the geodesic t_k-neighborhood of each interest point as forming its interest region at that scale.

11: **end for**

4.4 Experiments

In this section we present supporting experimental results to demonstrate the utility of the proposed method for detecting interest points at multiple scales. Some of the datasets presented in this section were obtained from 3D models that were available in the form of a triangulated connectivity mesh. We wish to emphasize here that the algorithm does *not* have any knowledge of the mesh during runtime, and the role of the mesh is *only* to aid visualization of the results.

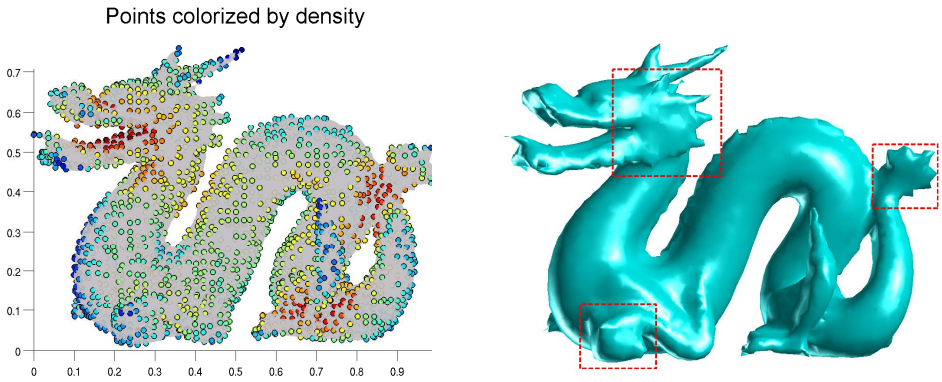


Figure 4.2: (left) Colorized plot showing variation of point density on the dragon model colored in gray. (right) Rendering of the underlying mesh. Results in boxed regions are analyzed in Figure 4.6.

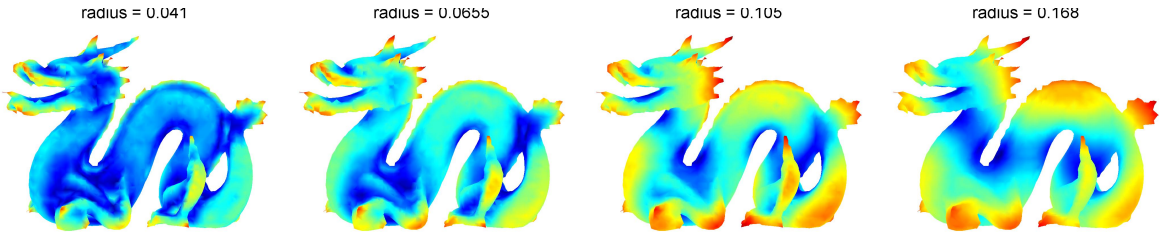
Qualitative behavior

To illustrate the behavior of the multi-scale operator, we focus on its application to a low-resolution version of the 'dragon' model, originally from the Stanford Scanning Repository. One important characteristic of this model is its complex shape with both finer spike-like features on its limbs, tail and back, as well as coarser features such as the stubby feet and tail. Figure 4.2 shows the 5205 points on the model colorized by point density, and illustrates the non-uniformity of the samples.

Figure 4.3 shows the norm of the density normalized Laplacian $\|\mathbf{x}_i - \tilde{A}(\mathbf{x}_i, t)\|$ for a few choices of scale t. The corresponding values of F are shown in Figure 4.4, and the regions associated with their scale-space extrema are shown in Figure 4.5. For each scale-space extremum in Figure 4.5, the mesh faces formed by points belonging to its interest region are given the same random color. Note that for the purpose of comparison with Figures 4.3 and 4.4 all the associated interest regions in Figure 4.5 are shown without the use of a threshold. A suitable choice of threshold may be used in practice to remove the least salient detections.

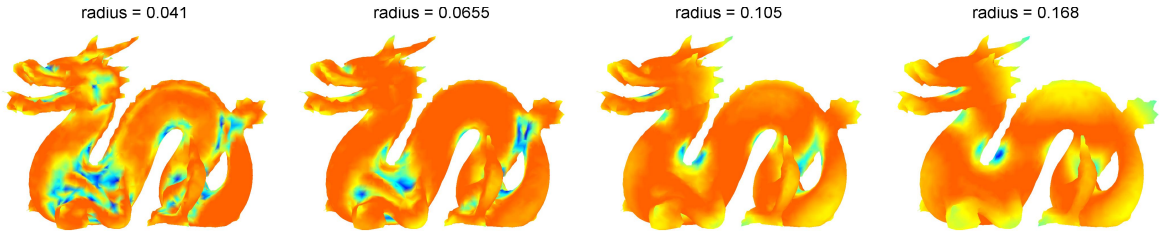
For the smallest values of t considered, the extrema correspond largely to noise due to discretization. When the value of t is increased, it can be seen (Figure 4.5) that features reflecting the geometry of the model become visible, such as the spikes on the feet and tail-end, and the features on the claws and mouth are progressively detected as hoped. Figure 4.6 shows zoomed views of some interesting parts of the model where complex shapes exist in close proximity and are correctly detected at different scales. For example, the first row shows the tail having fine short spikes at its edges that are detected for small values of t, while the coarser overall club-like shape is detected at a larger scale.

Figure 4.7 shows the results from processing a sparse outdoor scan of vehicles in a



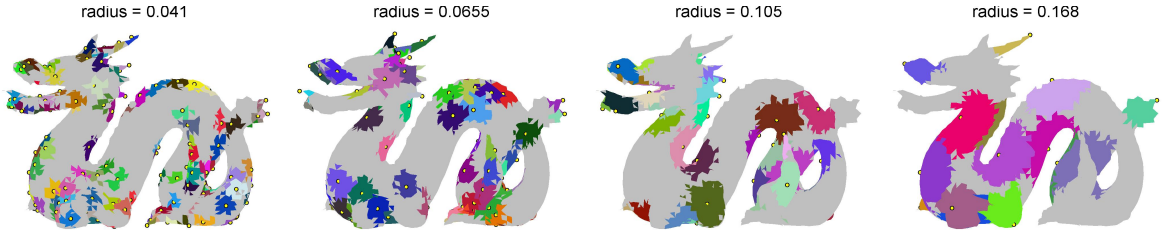
(a) (b) (c) (d)

Figure 4.3: Plot of norm of density-normalized Laplacian operator on the 'dragon' model for increasing kernel widths. Plots use the 'jet' colormap. Note that knowledge of the underlying mesh was not used in the computation.



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

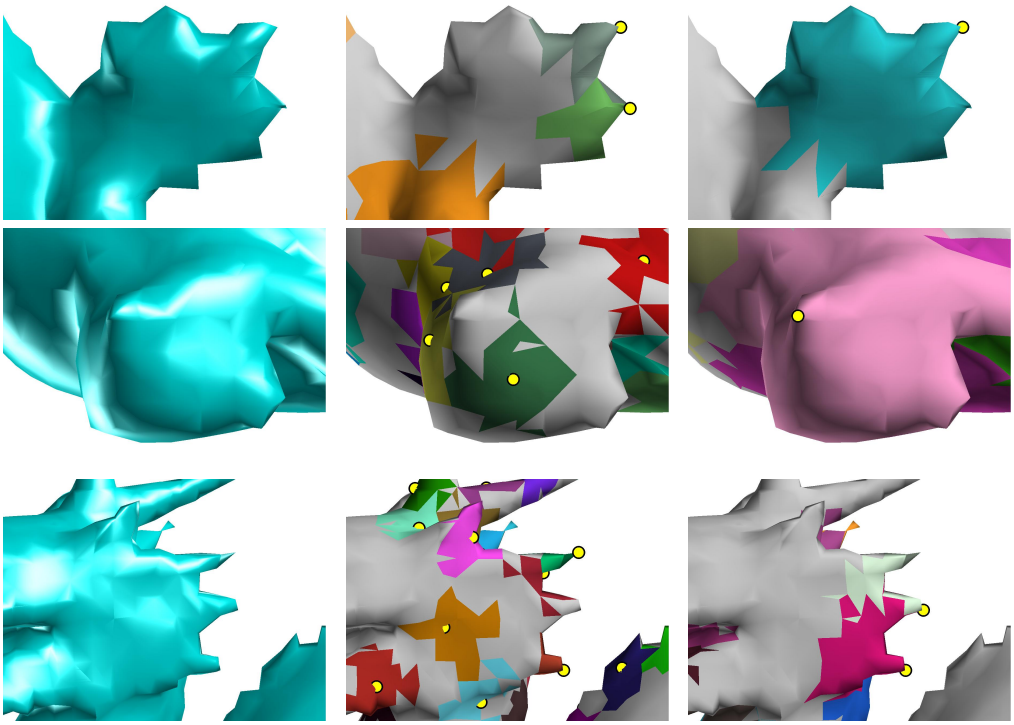
Figure 4.4: Plot of invariant F for increasing kernel widths, corresponding to values in Figure 4.3, and colorized with the 'jet' colormap.



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

Figure 4.5: Plot showing extrema and corresponding sizes for increasing kernel widths, corresponding to F values in Figure 4.4. Each colored patch on the model corresponds to the region associated with a extremum point detected at that scale. Yellow dots mark extrema locations.

parking lot. While the geometric interpretation of the results is not as vivid as the previous example, the detected intermediate regions correctly segment points on the vehicle into front and side components based on the difference in local curvature of those regions. Figures 4.9, 4.10 and 4.11 show results from processing aerial ladar scans of outdoor urban environments with similar behavior.



(a) (b) (c)

Figure 4.6: Interest regions detected in parts of the 'dragon' model from Figure 4.2 rendered in the column (a). Columns (b) and (c) show colored patches for each detected region corresponding to finer and coarser shapes, respectively. Yellow dots mark extrema locations. Figures are best seen in color.

Repeatability

To quantitatively measure the robustness of the algorithm to noisy data, we added Gaussian random noise with multiple values of standard deviation σ to points sampled with replacement from the 'bunny' and 'dragon' models from the Stanford database and computed the repeatability of the detections.

We define the overlap score between two interest regions as the ratio of the intersection to the union of the two smallest 3D spheres containing the points corresponding to those regions. Note that this metric for computing overlap is the direct 3D analog of the metric that is well accepted as the benchmark for 2D interest region detection in images [79]. For each detected interest region in the noise-free reference model, we assign its corresponding region in the noisy, resampled test model as the region with which it has the maximum overlap score. We then analyze the repeatability of the detection through computing the average of the overlap score between corresponding regions.

Figure 4.8 plots the average overlap of matched interest points for different noise levels

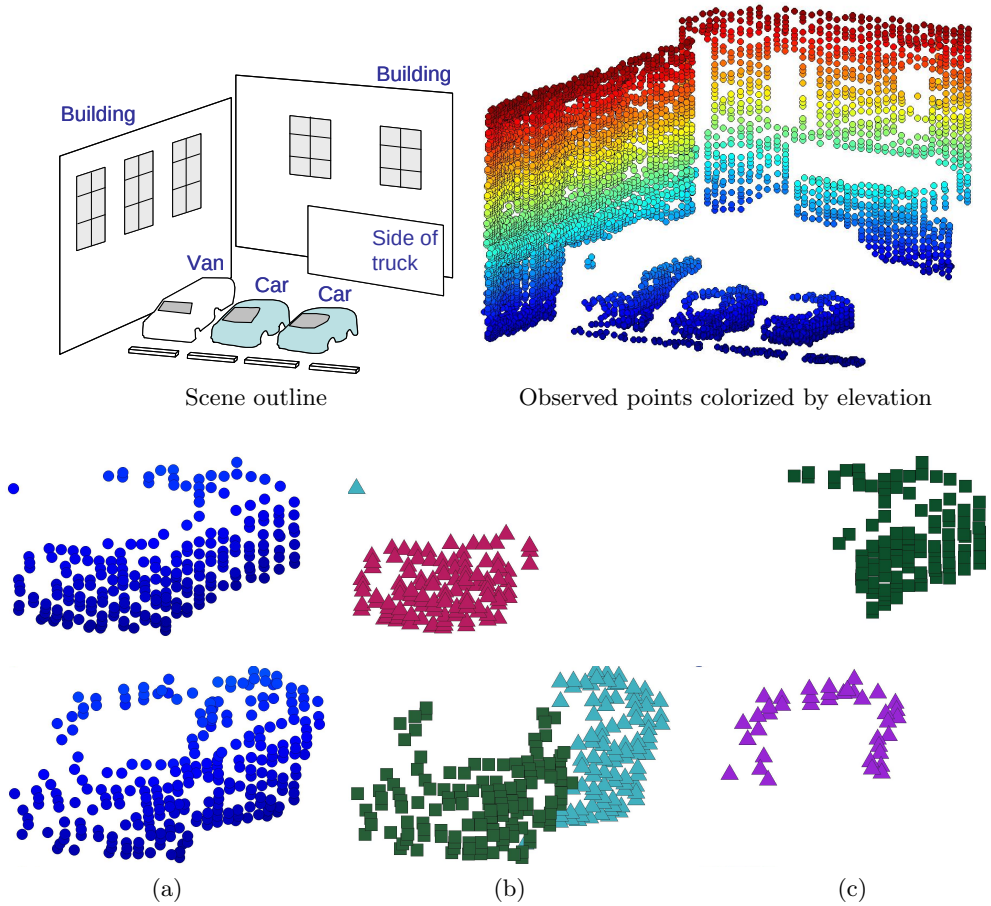
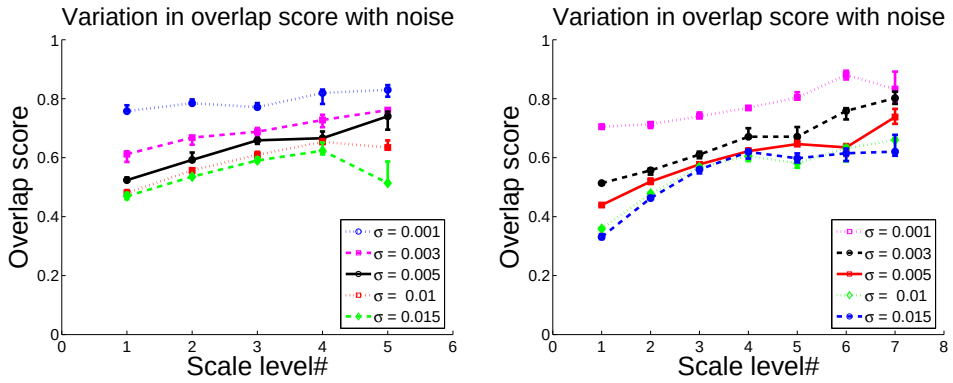


Figure 4.7: Top figure plots points from a sparse 3D scan of vehicles in a parking lot. Column (a) shows magnified view of the points on the two cars highlighted in the scene outline. Columns (b) and (c) plot points colored by interest regions detected at different scales for the input points in the corresponding row.

$\{\sigma\}$. The x-axis is the scale level number k indexing $\{t_k\}$, the set of scales considered.

Two observations are noteworthy. First, as can be expected, the overall repeatability score decreases with increasing noise level σ. Second, the repeatability of finer scale features (low scale level numbers) is consistently lower than that of coarser large-scale features. This should also not be surprising, as noise in point positions is indistinguishable from high-frequency shape details. Thus addition of Gaussian noise should adversely affect the repeatability of a geometric feature whose spatial extent is comparable to the standard deviation of the noise distribution.

When interpreting the repeatability score values, it should be noted that the overlap metric is far more stringent when used with 3D points than with 2D images for two reasons. First, due to the difference in dimensionality, an overlap score of 60%, for example, corresponds to a difference in radius of only 15% for two 3D spheres centered at the exact



(a) 'Bunny' dataset (b) 'Dragon' dataset

Figure 4.8: Variation in average overlap score of matched interest regions for various scales across increasing noise levels. Errorbars show upper and lower quartiles from 10 trials for each standard deviation (σ).

same location while it corresponds to as much as a 23% difference in circular patch radius in 2D. It has been verified in the 2D image domain [79] that an overlap error of 50% can be handled with a sufficiently robust descriptor. This is equivalent to an error of 65% in 3D, or an overlap score of just 35%. In addition, the ability to find extrema at precisely the same locations in 3D point samples of the same shape is severely limited by several factors such as point density and noise. Thus, for the 3D point cloud domain, a low repeatability score is not as strong a negative indicator, when compared to 2D images, of the ability to match descriptors computed for those regions.

Redundancy: One drawback of using fixed-scale quantities such as surface variation [87] to find interest regions is that, although the measure may correlate with change in surface geometry, it is unclear how to relate the computed values across different neighborhood sizes. Detectors based on these measures tend to select regions centered at the same points for multiple scales. Thus, the naive approach of simply choosing every interest region detected for a range of preset scales usually results in regions that have high degree of overlap, and thus a high degree of redundancy in the representation.

We quantify the redundancy in a given set of detected multi-scale interest regions by computing the average overlap between any pair of regions in the set. Overlap was computed in the same manner as in the repeatability experiment, and the comparison was made between a detector based on the surface variation score [87] (defined in Section 4.1) and our proposed invariant $F(\mathbf{x}, t)$. A lower overlap score is an indicator of higher variation in the detected interest regions and thus lower redundancy.

Using our proposed method, the redundancy score for the 'dragon' model reduced from 0.166 to 0.114, and reduced from 0.165 to 0.152 for the 'bunny' model, corresponding to difference of 32% and 8% respectively. Although this difference is dataset-dependent, we

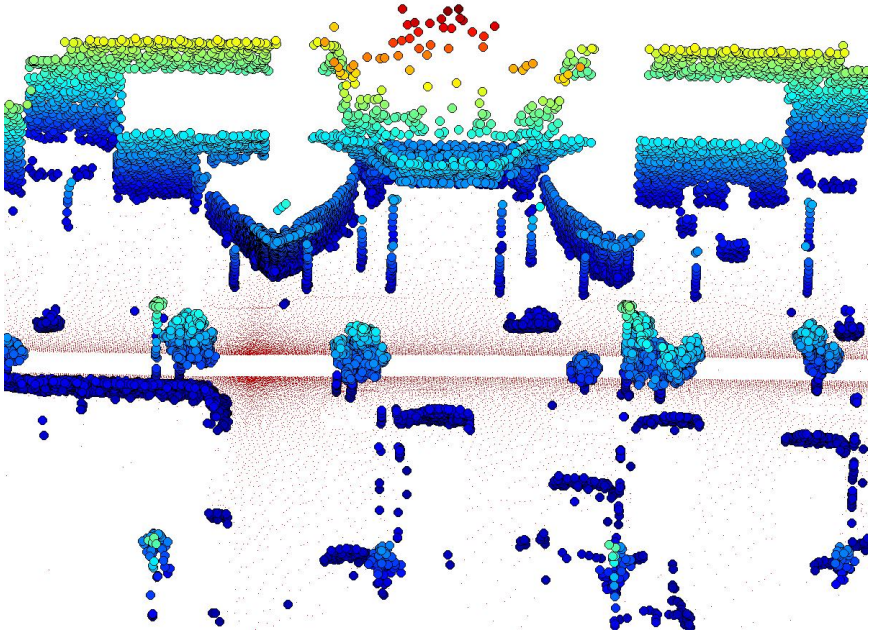
have observed this lowering of redundancy score to be consistent across datasets and noise levels. At the same time, our proposed method also gives a larger number of interest regions.

4.5 Discussion

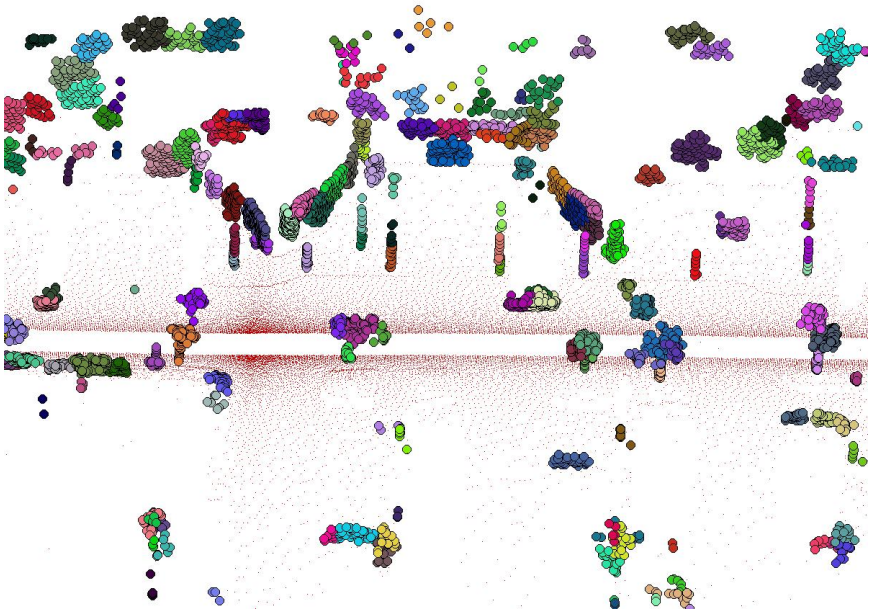
This work presented a filtering operator that works directly in the input point cloud domain to generate multi-scale representations that reflect the underlying geometry of the data. Our approach has deviated from traditional multi-scale analysis in that we construct an operator $\tilde{A}$ that does *not* have the property of being a semi-group by adopting the density normalization of Section 4.2. The impact of this is largely reflected as increased computational cost. However, it benefit is felt as the removal of dependence on the point sample distribution. This invariance to point sample distribution is a fundamental requirement for analyzing shape from 3D point clouds, unlike the case of images, where the pixels at which scene intensity is sampled always follow a regular lattice arrangement by design.

One drawback of not using a 2D parameterization in our approach is that it is unclear how to detect oriented features such as corners and ridges. Thus it may be worthwhile to introduce local 2D parameterization in conjunction with the scale-invariant operators introduced here.

This approach also leaves open the possibility of richer classes of operators that could be obtained by modifying the base function ϕ or using its higher order derivatives. A more exhaustive characterization of the possible choices could lead to a useful larger family of interest region detectors and deserves further study.

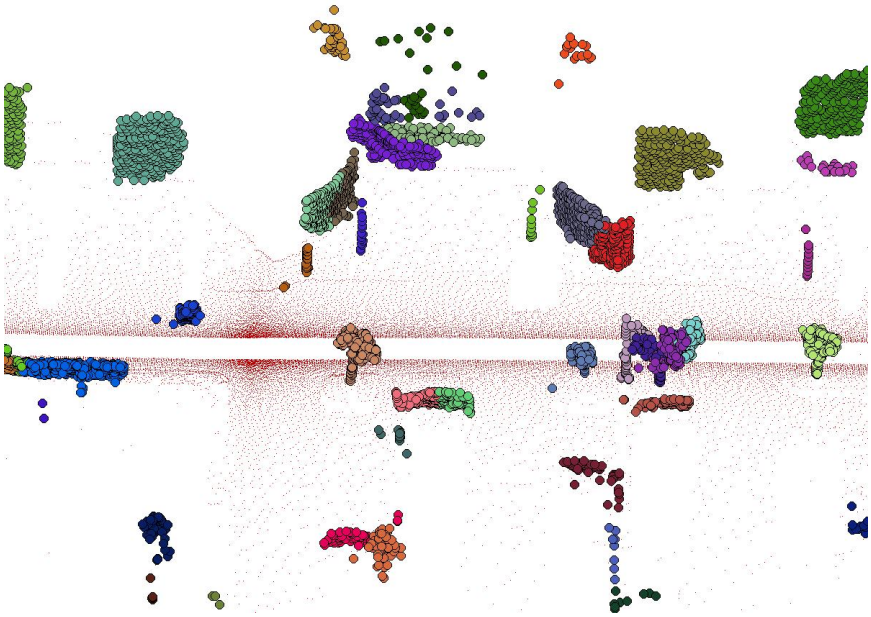


(a) Aerial view of area of interest. Red points were classified as associated with the ground plane and were not considered for interest region detection. Remaining points are colored by elevation using the 'jet' colormap.

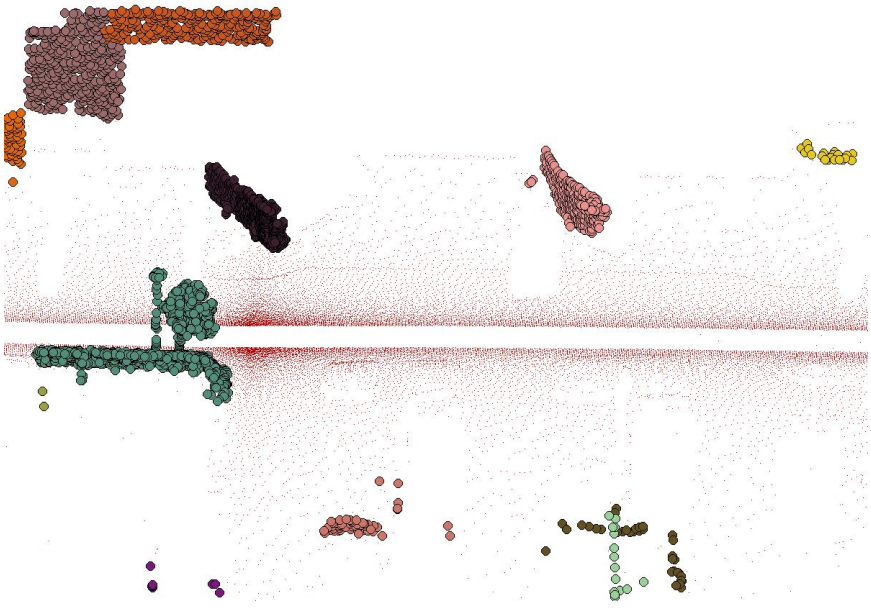


(b) Detected interest regions for support radius $(1.6)^2 = 2.56$m

Figure 4.9: Interest region detection on the *waterfront_1* dataset $(136,400$ points) collected from an aerial scan

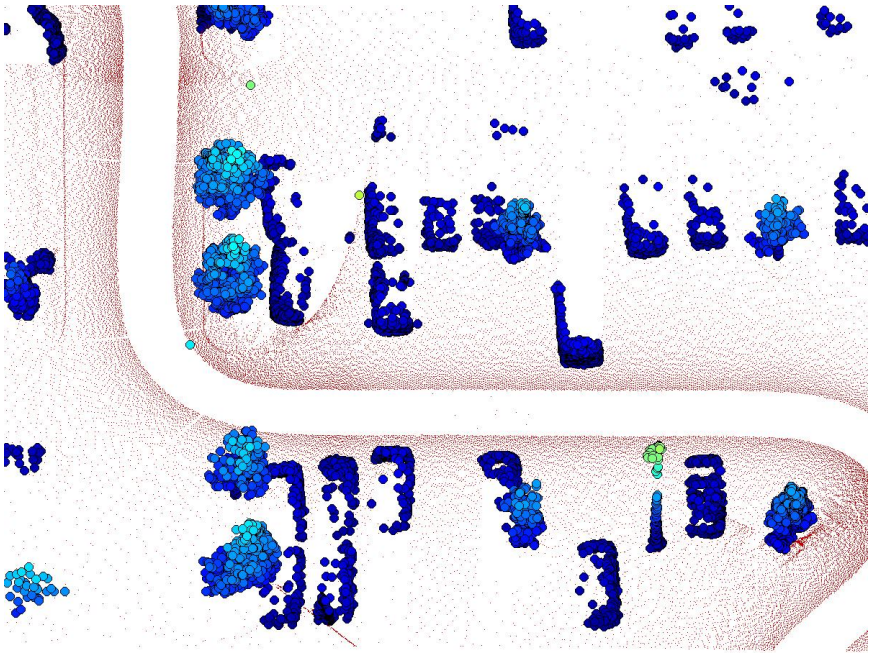


(a) Detected interest regions for support radii of $(1.6)^4 = 6.55$m

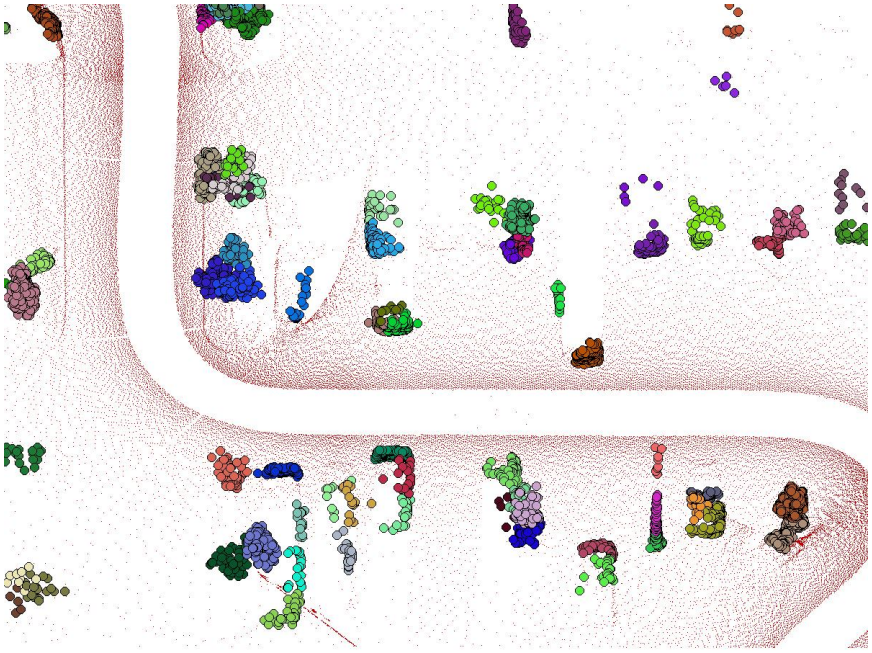


(b) Detected interest regions for support radii of $(1.6)^6 = 16.67$m

Figure 4.9: (contd.) Interest region detection on the *waterfront_1* dataset ($136,400$ points) collected from an aerial scan.

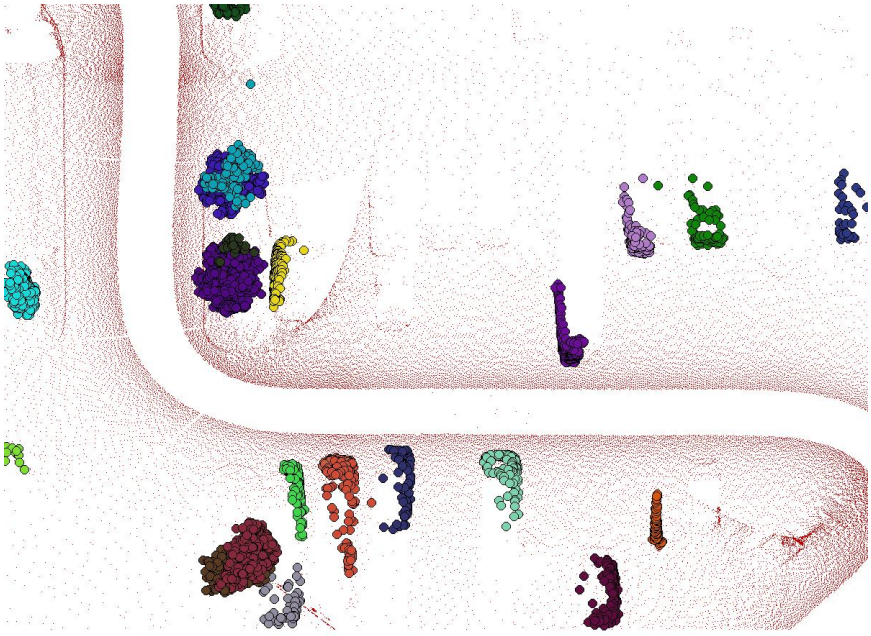


(a) Aerial view of area of interest. Red points were classified as associated with the ground plane and were not considered for interest region detection. Remaining points are colored by elevation using the 'jet' colormap.



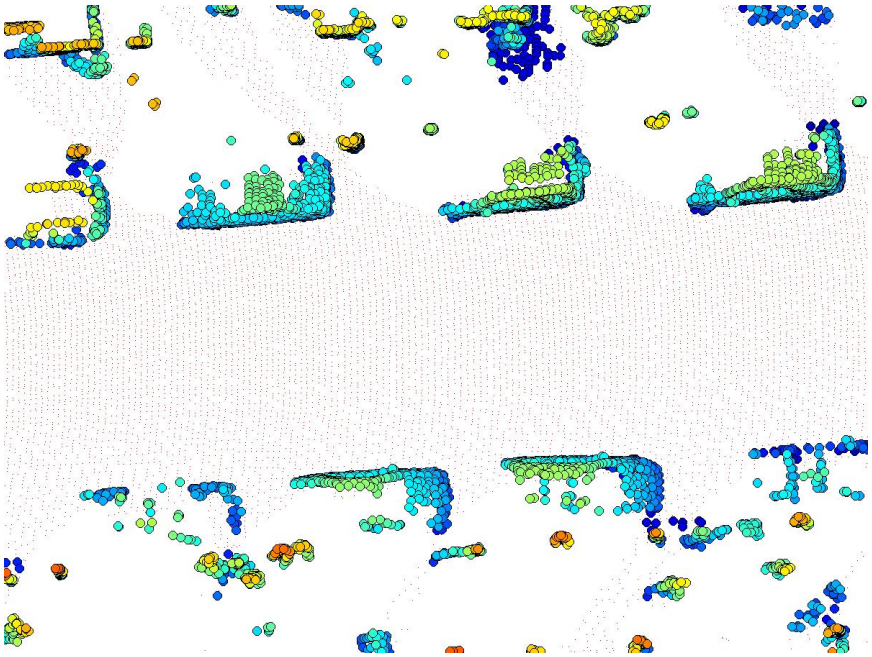
(b) Detected interest regions for support radius $(1.6)^2 = 2.56$m

Figure 4.10: Interest region detection on the *waterfront_2* dataset (214, 473 points) collected from an aerial scan.

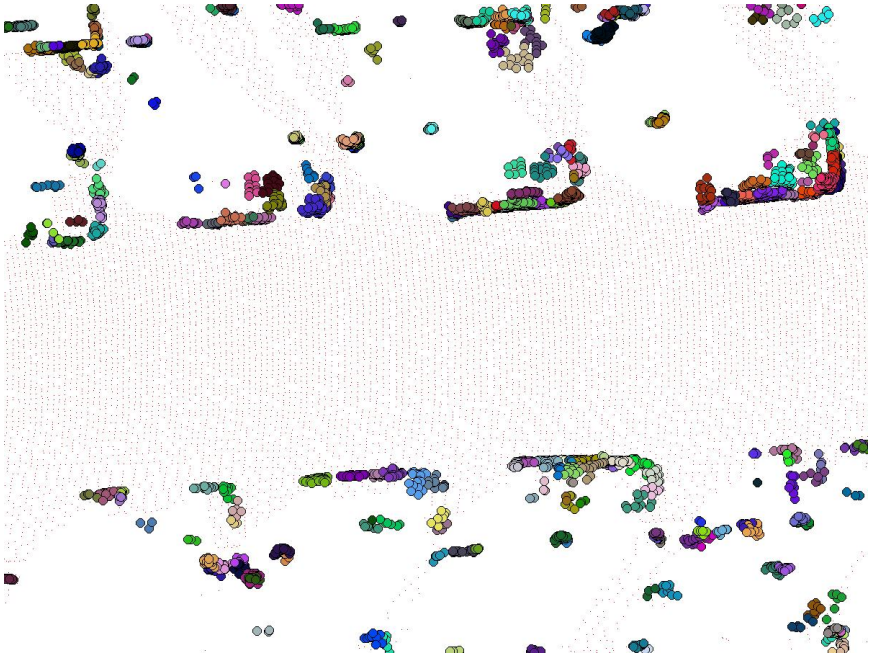


(a) Detected interest regions for support radii of $(1.6)^4 = 6.55$m

Figure 4.10: (contd.) Interest region detection on the *waterfront_2* dataset (214, 473 points) collected from an aerial scan

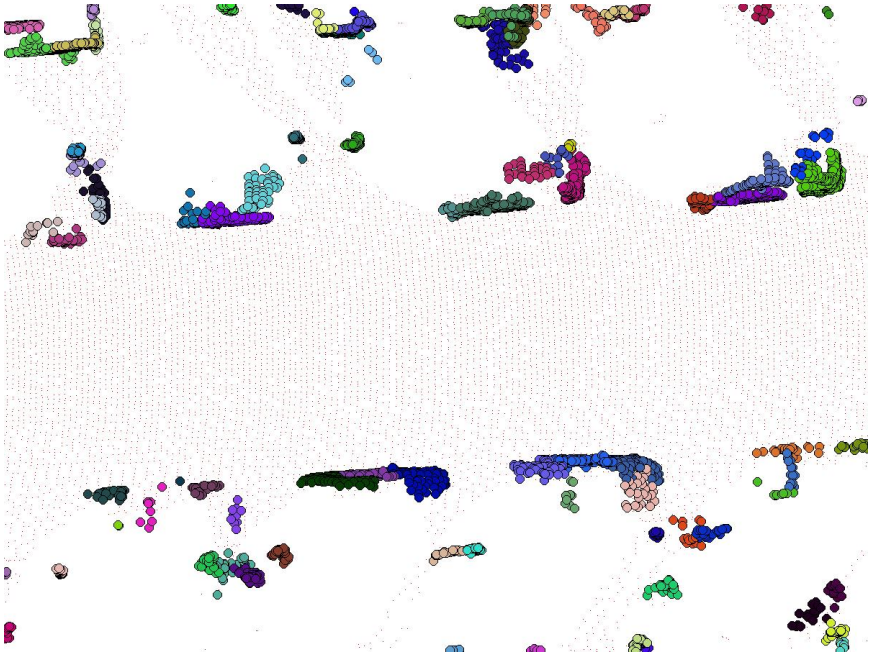


(a) Aerial view of area of interest. Red points were classified as associated with the ground plane and were not considered for interest region detection. Remaining points are colored by elevation using the 'jet' colormap.



(b) Detected interest regions for support radius $(1.6)^2 = 2.56$m

Figure 4.11: Interest region detection on the *forbes* dataset (154, 333 points) collected from an aerial scan.



(a) Detected interest regions for support radii of $(1.6)^4 = 6.55$m

Figure 4.11: (contd.) Interest region detection on the *forbes* dataset ($154,333$ points) collected from an aerial scan

CHAPTER 5

Moving Forward

In this dissertation, we have presented some fundamental building blocks for processing shape information represented as point clouds, and demonstrated their utility on shape reconstruction and multi-scale analysis. We conclude with a discussion of some extensions and improvements to the algorithms proposed in this thesis to serve as a potential starting point for future work in this area.

5.1 Geometric graph construction

Many of the algorithms we have proposed can benefit from knowledge of geodesic distances between points. We supply this information by constructing a neighborhood graph with the vertices representing the observed points and weights on edges between points equal to the Euclidean distances between points. The path distances on the graph between two points is then taken to approximate geodesic distances.

A couple of observations are worth noting. First, we wish to emphasize that such a graph construction is not unique to our work and it is shared by several other approaches to surface reconstruction and multi-scale analysis [44, 49, 87, 112]. Graph construction is also relevant for problems outside the realm of 3D geometry such as manifold learning [14, 94, 103, 126] and clustering [24, 38] which are fundamental problems in machine learning. It is also relevant to image segmentation algorithms that rely on the construction of an affinity matrix [98], as the non-zero entries of the matrix specify the topology of the graph being considered.

Second, there is a strong distinction between constructing a geometric graph and a

polygonal mesh, which is a representation we wished to avoid from the start. Polygonal meshes require that each face formed by edge of the graph it represents form a piece-wise approximation of the surface itself. Often, mesh construction algorithms are also required to satisfy conditions such as upper bounds on the angle subtended by any vertex of a polygon, small difference in edge lengths of any face, maximum edge length in a face, the degree of any vertex etc. In contrast, the requirement of graph path distances approximating geodesic distances is a far less stringent condition for graph construction.

However, at present, the construction of a neighborhood graph in a manner such that graph distances best approximate geodesic distances in the presence of noisy data is an open problem. In fact, despite its wide applicability, this problem appears to be relatively unstudied.

In what follows, we review some approaches to graph construction commonly used in the literature and survey the theoretical results known so far about them with respect to graph distances.

ϵ-ball graphs are formed by connecting any pair of points with an edge if the distance between the points is less than *epsilon*. **k-nearest neighbor (KNN) graphs** are formed by connecting every point $\mathbf{x}_i$ with another point $\mathbf{x}_j$ if $\mathbf{x}_j$ is amongst the k nearest neighbors of $\mathbf{x}_i$. A symmetrized version of KNN graphs is obtained by changing the above condition to connect every $\mathbf{x}_i$ with $\mathbf{x}_j$ if $\mathbf{x}_j$ is among the k nearest neighbors of $\mathbf{x}_i$ *or* if $\mathbf{x}_i$ is among the k nearest neighbors of $\mathbf{x}_j$.

ϵ-ball graphs and KNN graphs constitute the most commonly used types of graphs in the literature [14, 38, 44, 103], and have received the most study. In [18], Bernstein *et al.* proved the convergence of graph path distances in ϵ and KNN graphs to geodesic distances in the limit as the point density increased to infinity, on condition as the data points obeyed some sampling conditions. They also have the advantage that they are relatively fast to implement, thanks to the many data structures (KD-trees, cover trees, ball trees) available for range interval queries, and Approximate Nearest Neighbor (ANN) [12] search methods that give approximate nearest neighbor results in near-linear time.

However, both of these types of graph constructions suffer from the problem that they strongly depend on data-dependent parameters. i.e. the automatic selection of ϵ and k is not straightforward. Furthermore, neither of them guarantees that the resulting graph will be connected, and in practice they tend to form too many edges between points located in regions of higher point density, as they place little restriction on the maximum number of edges that at each point may participate in.

A different type of graph construction is based on **minimum spanning trees (MST)**. Work in [87] uses a greedy algorithm to construct a MST from points using a combination of feature weights and Euclidean distance to define the overall edge weights. Geometric features such as edges or ridges then determined as subsets of this graph. MST graphs have

the desirable property that they are guaranteed to be connected by construction. However, they are sensitive to noise.

Work in [24] proposed two alternate graph constructions to try and combine multiple MSTs in order to reduce their sensitivity to noise. The first construction termed **perturbed minimum spanning trees (PMST)** are constructed by combining several, say t, MSTs each of which is fit to a perturbed version of the data. The perturbation is done by adding a small amount of zero-mean noise to each data point independently. In other words, two points are connected by an edge if they were connected by an edge in the MSTs constructed for all the perturbed versions of the point set. This strategy discourages the addition of edges that connect accidentally proximal points. The downside is in choosing an appropriate noise model, and the number of MSTs to be combined.

The second construction termed **disjoint minimal spanning tree (DMST)** is constructed as an extension of Kruskal's minimum spanning tree algorithm. A MST is first constructed starting from a complete graph. The edges that form the MST are then removed from the complete graph and the MST is constructed again from the reduced edge set. This process is repeated t times, where $t = 1$ yields the original MST. Here the only parameter to be chosen is t unlike PMSTs which require an additional specification of the noise model. In our experiments, we set to $t = 2$ for curves in 2D and $t = 4$ for surfaces in 3D. We have observed these parameter settings to suitable for all the datasets we tested on, both synthetic data and well as data from outdoor settings.

The biggest drawback of DMSTs is computational complexity, particularly that of computing all pair-wise distances to construct each MST. In a plane, the Euclidean MST has a nice property of being a strict subtree of the Delaunay triangulation of the points. This reduces the computational complexity from $O(n^2)$ to $O(n \log n)$ for n points. However, this property does not hold in the 3D domain.

One possibility to reduce the computational complexity when combining MSTs may be to impose a graph "prior" by starting with a reduced graph instead of the complete graph. This could be done by building an ϵ-graph or KNN graph for large ϵ or k, although at the risk of making poor irreversible commitments early on. Other possibilities include greedily constructing a k-connected graph [125], or using a KNN in combination with a MST to guarantee connectivity property, and other such variations. With several potential applications of a scalable solution to the geometric graph construction problem, further work in this area is certainly warranted.

5.2 Fast filtering operators

The multi-scale filtering operator proposed in Chapter 4 has two kinds of computational bottleneck. The first is that on normalizing the operator $\tilde{A}$ by the point density it loses

the property of being a semi-group. Thus unlike the Laplacian of Gaussian (LoG) operator which may be approximated by a Difference of Gaussian (DoG) operator [72], we are not aware of a convenient approximation for multi-scale operators in 3D.

The second bottleneck is the computation of the operator itself, which involves computing quantities of the form

$$\tilde{A}(\mathbf{y}, t) = \frac{\sum_j \tilde{\phi}(\mathbf{y}, \mathbf{x}_j, t)\mathbf{x}_j}{\sum_j \tilde{\phi}(\mathbf{y}, \mathbf{x}_j, t)}, \tag{5.1}$$

which requires $O(n^2)$ computation time in a naive implementation.

This cost is reduced considerably in practice by bounding the region of influence of the unnormalized kernel $\phi(\mathbf{y}, \mathbf{x}_j, t)$ at the expense of a small loss in accuracy. Recall from Algorithm 3 that the kernel $\phi(\mathbf{y}, \mathbf{x}_j, t)$ is computed as a function of the geodesic distance between $\mathbf{y}$ and $\mathbf{x}_j$. Since, Euclidean distance is always less than or equal to the geodesic distance, this truncation can be done by choosing only points $\mathbf{x}_j$ that are at Euclidean distance $3t$ or less from the point of interest $\mathbf{y}$ using an efficient data structure, such as a KD-tree. However, the benefits of truncation reduce quickly for increasing value of bandwidth parameter t.

Another approach for scaling the computation of $\tilde{A}(\mathbf{x}, t)$ efficiently to very large datasets may be through a category of techniques known as multipole methods, more commonly known in the computer vision and machine learning literature by its variants the Fast Gauss Transform [35] and the Improved Fast Gauss Transform [35, 91, 124]. These methods have been successfully shown to reduce the computation of weighted sums of Gaussians at n points from $O(n^2)$ to $O(n)$ with bounded error.

The key idea is to factor the kernel $\phi(\mathbf{y}, \mathbf{x}_j, t)$ into a product of terms that depend only on $\mathbf{y}$ and $\mathbf{x}_j$ as

$$\phi(\mathbf{y}, \mathbf{x}_j, t) = \sum_{k=1}^{p} \psi_k(\mathbf{y}, t)\psi_k(\mathbf{x}_j, t). \tag{5.2}$$

In the case where an exact factorization is not possible, an approximate expansion may be carried out in a local neighborhood. Then quantities such as the numerator and denominator of $\tilde{A}$ may be computed separately as

$$\begin{aligned}
\sum_j \tilde{\phi}(\mathbf{y}, \mathbf{x}_j, t) &= \sum_j w_j \sum_{k=1}^{p} \psi_k(\mathbf{y}, t)\psi_k(\mathbf{x}_j, t) \\
&= \sum_{k=1}^{p} \psi_k(\mathbf{y}, t) \left(\sum_j w_j \psi_k(\mathbf{x}_j, t) \right) \\
&= \sum_{k=1}^{p} A_k(t)\psi_k(\mathbf{y}, t).
\end{aligned} \tag{5.3}$$

If the A_k terms can be pre-computed in $O(n)$, then the evaluation of the above expression for all query points $y = \{\mathbf{x}_i\}$ can also be done in $O(n)$ time.

Previous work [35, 91] has shown how the kernel expansions of the form described above may be performed exactly for polynomial kernels and approximately for Gaussian kernels in Euclidean space. It will be interesting to study whether such a useful expansion could be obtained for the Gaussian kernel in non-Euclidean (manifold) spaces as would be appropriate to multi-scale operators for point clouds.

Yet another strategy that would complement the above approach would be to develop a quadrature rule [16, 17] for computing the integral operator so that a far fewer number of points would need to be evaluated at the expense of a small loss in numerical accuracy.

5.3 Summary

In conclusion, we wish to emphasize that techniques to process geometric information in the form of 3D point clouds are to undoubtedly become increasingly important as the ability to acquire large amounts of detailed depth scans becomes more accessible. The work presented in this thesis contributes some fundamental tools in geometry signal processing for surface fitting, denoising and multi-scale representation. We hope that it will motivate other researchers to develop a larger suite of mathematically sound tools to process 3D data.

During the process of algorithm development, we benefitted greatly by starting from the simplest instances of the geometry processing problems we cared about that exhibited the same challenges of the harder more general problem. Although the problems seemed misleadingly oversimplified at times, they were in fact valuable sources of intuition. This method of analysis is also evident in the manner of explanation of the ideas presented in this dissertation, where we frequently started with 2D problems and generalized their solution to 3D. We thus encourage other practitioners to follow a similar approach to problem analysis.

An underlying theme of this work has been the derivation of 3D point cloud operators that possess the same kinds of formal guarantees on repeatability as the signal processing techniques that are so common in the 2D image domain. Through the development of sound 3D point processing tools that have the same procedural functionality as current tools for image processing, we are optimistic that an eventual alignment of image and geometry processing algorithms for solving computer vision problems is near.

Appendices

APPENDIX A

Derivation of Error Bounds

In this appendix we detail the derivation of the error bounds for curve fitting used in Chapter 2. The purpose of this elaboration is two-fold. First, by breaking down the derivation into problem-independent identities and their application using the curve geometry model to obtain the more problem-specific end results, we wish to emphasize that the method of analysis proposed in Chapter 2 may be generalized to other problems involving estimation of geometry model parameters. Second, in the interest of keeping the document as self-contained as possible, we wish to make the analytic expressions of the error bounds readily accessible to readers not interested using Mathematica™ or similar analytic solvers to rederive these results.

The organization of this appendix is as follows. In Appendix A.1 we first prove the identities listed in Section 2.3 of the mean and variance of the sample variance and sample covariance estimator of any random variable. We then list some useful statistics of the uniform distribution in Appendix A.2. We then apply these statistics with the identities from Appendix A.1 to the curve noise model (2.9) and derive useful expressions involving higher-order moments and covariances in Appendix A.3. Finally in Appendix A.4, we derive the bounds on the variance of the sample estimators and thus the error bounds of the estimator.

A.1 Estimators and their properties

In this section, we derive some useful general identities expressing the mean and variance of the sample mean, sample variance and sample covariance estimators.

To avoid introducing further notation, we use the symbols X and $\{x_i\}$ in this section to denote the random variable of interest and its samples, but with the understanding that the expressions in this section are true for *any* random variable. Later in Appendix A.4, we will use these identities to derive bounds specific to the curve noise model of Section 2.3.

Sample mean

Consider a random variable X and the estimator of its mean μ_X from n samples $\{x_i\}$:

$$\bar{X}_n = \frac{1}{n}\sum_i x_i. \tag{A.1}$$

It follows that the sample mean is a random variable with the following properties:

$$\mathbb{E}(\bar{X}_n) = \frac{1}{n} n\mathbb{E}(X_i) = \mu_X \tag{A.2}$$

$$\mathbb{V}(\bar{X}_n) = \frac{1}{n^2} n\mathbb{V}(X_i) = \frac{\mathbb{V}(X_i)}{n}. \tag{A.3}$$

Sample Variance

Consider the estimator of variance of X from its n samples $\{x_i\}$:

$$\hat{\sigma}_X^2 = \tfrac{1}{n-1}\sum_i (x_i - \bar{x}_n)^2 \tag{A.4}$$

The expected value of this estimator (also a random variable) is given by

$$\begin{aligned}
\mathbb{E}(\hat{\sigma}_n^2) &= \tfrac{1}{n-1}\mathbb{E}\left[\sum_i (X_i - \mu_X) - (\bar{X}_n - \mu_X)^2\right] \\
&= \tfrac{1}{n-1}\sum_i \left[\mathbb{E}(X - \mu_X)^2 + \mathbb{E}(\bar{X}_n - \mu_X)^2 - 2\,\mathbb{E}(X_i - \mu_X)(\bar{X}_n - \mu_X)\right] \\
&= \tfrac{1}{n-1}\left[n\mathbb{V}(X) + n\frac{\mathbb{V}(X)}{n} - 2\,\mathbb{E}(\bar{X}_n - \mu_X)(\bar{X}_n - \mu_X)\right] \\
&= \tfrac{1}{n-1}(n + 1 - 2)\mathbb{V}(X) = \mathbb{V}(X).
\end{aligned}$$

Hence $\hat{\sigma}_X^2$ is an unbiased estimator.

The variance of the estimator $\hat{\sigma}_n^2$ is a little less straightforward. We first prove some useful identities.

Identity 3.

$$\hat{\sigma}_n^2 = \tfrac{1}{2n(n-1)}\sum_{i,j}(x_i - x_j)^2$$

Proof.

$$\begin{aligned}
\text{RHS} &= \tfrac{1}{2n(n-1)} \sum_{i,j} \left[(x_i - \bar{x}_n) - (x_j - \bar{x}_n)\right]^2 \\
&= \tfrac{1}{2n(n-1)} \left[n \sum_i (x_i - \bar{x}_n)^2 + n \sum_j (x_j - \bar{x}_n)^2 - 2 \sum_{i,j} (x_i - \bar{x}_n)(x_j - \bar{x}_n) \right] \\
&= \tfrac{1}{2n(n-1)} \left[2n \sum_i (x_i - \bar{x}_n)^2 - 2 \left[\sum_i (x_i - \bar{x}_n) \right]^2 \right] \\
&= \tfrac{1}{n-1} \sum_i (x_i - \bar{x}_n)^2
\end{aligned}$$

□

To conveniently represent the higher order centered moments of a random variable X having mean μ_X and variance σ_X^2, we use the notation $d_m(X)$ previously introduced in (2.21) as

$$d_m(X) \triangleq \mathbb{E}(X - \mu_X)^m. \tag{A.5}$$

Note that $d_0(X) = 1$, $d_1(X) = 0$ and $\sigma_X^2 = d_2(X)$ under the above definition.

Identity 4.

$$\mathbb{E}\left[(X_i - X_j)^m\right] = \sum_{k=0}^{m} \binom{m}{k} d_k d_{m-k} \tag{A.6}$$

Proof.

$$\begin{aligned}
(X_i - X_j)^m &= ((X_i - \mu_X) - (X_j - \mu_X))^m \\
&= \sum_{k=0}^{m} \binom{m}{k} (X_i - \mu_X)^k (X_j - \mu_X)^{m-k}
\end{aligned}$$

Using independence of $X_i - \mu_X$ and $X_j - \mu_X$, and taking expectations on both sides gives the desired result. □

With the above two identities, we may prove the Identity 1 from Section 2.3 which we restate here for reference.

Identity 1. (Variance of the *sample variance* estimator)

$$\boxed{\mathbb{V}(\hat{\sigma}_X^2) = \frac{d_4(X)}{n} - \frac{(n-3)}{n(n-1)} \sigma_X^4} \tag{A.7}$$

Proof. By the property of covariance of sums of variables

$$\mathbb{V}(\hat{\sigma}_X^2) = \tfrac{1}{(4n(n-1))^2} \sum_{i,j,k,l} \operatorname{cov}\left((x_i - x_j)^2, (x_k - x_l)^2\right) \tag{A.8}$$

There are 3 cases for the right hand side:

1. Distinct i, j, k, l :In this case $\operatorname{cov}\left((x_i - x_j)^2, (x_k - x_l)^2\right) = 0$ by independence.

2. Distinct i, j in $\operatorname{cov}\left((x_i - x_j)^2, (x_i - x_j)^2\right)$: There are $2n(n-1)$ terms fulfilling this case.

$$\begin{aligned}
&\operatorname{cov}\left((x_i - x_j)^2, (x_i - x_j)^2\right) \\
&\quad = \mathbb{E}(x_i - x_j)^4 - [\mathbb{E}(x_i - x_j)^2]^2 \\
&\quad = d_4(X) + 8d_1(X)d_3(X) + 6d_2^2(X) + d_4(X) - (2d_2(X) + 2d_1^2(X))^2 \\
&\quad = 2d_4(X) + 2d_2(X)^2
\end{aligned}$$

3. Distinct i, j, k in $\operatorname{cov}\left((x_i - x_j)^2, (x_k - x_j)^2\right)$: There are $4n(n-1)(n-2)$ terms fulfilling this case.

$$\begin{aligned}
\operatorname{cov}\left((x_i - x_j)^2, (x_k - x_j)^2\right) &= \mathbb{E}[(x_i - x_j)^2(x_j - x_k)^2] \\
&= (d_4(X) + 3d_2^2(X)) - 4d_2^2(X) \\
&= d_4(X) - d_2^2(X)
\end{aligned}$$

Summing up the weighted contributions of each case gives the desired result. □

Sample Covariance

Consider the sample covariance estimator between random variables X and Y from n samples $\{x_i\}$ and $\{y_i\}$:

$$\hat{S}_{XY} = \frac{1}{n-1} \sum_i (x_i - \bar{X}_n)(y_i - \bar{Y}_n) \tag{A.9}$$

Let the true value of the covariance be denoted by σ_{XY}. The expected value of the estimator is given by:

$$\begin{aligned}
\mathbb{E}(\hat{S}_{XY}) &= \frac{1}{n-1} \sum_i \mathbb{E}(x_i - \bar{X}_n)(y_i - \bar{Y}_n) \\
&= \frac{1}{n-1} \sum_i \Big[\mathbb{E}(x_i - \mu_X)(y_i - \mu_Y) - \mathbb{E}(x_i - \mu_X)(\bar{Y}_n - \mu_Y) \\
&\qquad - \mathbb{E}(\bar{X}_n - \mu_X)(y_i - \mu_Y) + \mathbb{E}(\bar{X}_n - \mu_X)(\bar{Y}_n - \mu_Y)\Big] \\
&= \frac{n}{n-1} \left[\sigma_{XY} - \frac{\sigma_{XY}}{n} - \frac{\sigma_{XY}}{n} + n\frac{\sigma_{XY}}{n^2}\right] \\
&= \sigma_{XY}
\end{aligned}$$

Hence the estimator is unbiased.

We proceed to derive variance of $\hat{S}_{XY}$ in a manner similar to that of $\hat{\sigma}_X^2$ by first proving some identities.

Identity 5.

$$\hat{S}_{XY} = \frac{1}{2n(n-1)} \sum_{i,j} (x_i - x_j)(y_i - y_j) \tag{A.10}$$

Proof.

$$\begin{aligned}
\text{RHS} &= \frac{1}{2n(n-1)} \sum_{i,j} \left[(x_i - \bar{X}_n) - (x_j - \bar{X}_n)\right] \left[(y_i - \bar{Y}_n) - (y_j - \bar{Y}_n)\right] \\
&= \frac{1}{2n(n-1)} \sum_{i,j} \Big[(x_i - \bar{X}_n)(y_i - \bar{Y}_n) - (x_i - \bar{X}_n)(y_j - \bar{Y}_n) \\
&\qquad\qquad - (x_j - \bar{X}_n)(y_i - \bar{Y}_n) + (x_j - \bar{X}_n)(y_j - \bar{Y}_n)\Big] \\
&= \frac{1}{2n(n-1)} \left[n \sum_i (x_i - \bar{X}_n)(y_i - \bar{Y}_n) + n \sum_j (x_j - \bar{X}_n)(y_j - \bar{Y}_n) \right] \\
&= \frac{1}{(n-1)} \sum_i (x_i - \bar{X}_n)(y_i - \bar{Y}_n)
\end{aligned}$$

□

For convenience, we use the notation $c_m(X, Y)$ previously introduced in (2.23) as

$$c_m(X, Y) \triangleq \mathbb{E}\left[(X - \mu_X)(Y - \mu_Y)\right]^m . \tag{A.11}$$

Note that $c_1(X, Y) = \sigma_{XY}$ and $c_0(X, Y) = 1$. To avoid cluttering the equations, we will drop the explicit arguments (X, Y) when it is clear from the context.

Identity 6.

$$\mathbb{E}[(X_i - X_j)(Y_i - Y_j)] = 2c_1 \tag{A.12}$$

Proof.

$$\begin{aligned}
\text{LHS} &= \mathbb{E}(X_i - \mu_X)(Y_i - \mu_Y) + \mathbb{E}(X_j - \mu_X)(Y_j - \mu_Y) \\
&\quad + \mathbb{E}(X_i - \mu_X)(Y_j - \mu_Y) + \mathbb{E}(X_j - \mu_X)(Y_i - \mu_Y) \\
&= 2c_1
\end{aligned}$$

□

Identity 7.

$$\mathbb{E}[(X_i - X_j)^2 (Y_i - Y_j)^2] = 2(c_2 + \sigma_X^2 \sigma_Y^2 + 2c_1^2) \tag{A.13}$$

Proof.

$$\begin{aligned}\text{LHS} &= \mathbb{E}\Big(\big[(X_i-\mu_X)^2 + (X_j-\mu_X)^2 + 2(X_i-\mu_X)(X_j-\mu_X)\big] \\ &\quad \big[(Y_i-\mu_Y)^2 + (Y_j-\mu_Y)^2 + 2(Y_i-\mu_Y)(Y_j-\mu_Y)\big]\Big) \\ &= \mathbb{E}\left[2(X_i-\mu_X)^2(Y_i-\mu_Y)^2 + 2(X_i-\mu_X)^2(Y_j-\mu_Y)^2 + 4c_1.c_1\right] \\ &= 2c_2 + 2\sigma_X^2\sigma_Y^2 + 4c_1^2\end{aligned}$$

□

With the above two identities, we may prove the Identity 2 from Section 2.3 which we restate here for reference.

Identity 2. (Variance of the *sample covariance* estimator)

$$\mathbb{V}(\hat{S}_{XY}) = \frac{c_2(X,Y)}{n} + \frac{\sigma_X^2\sigma_Y^2}{n(n-1)} - \frac{(n-2)}{n(n-1)}c_1^2(X,Y) \tag{A.14}$$

Proof. By the property of variance of a sum of variables:

$$\mathbb{V}(\hat{\sigma}_{XY}) = \tfrac{1}{(2n(n-1))^2}\sum_{i,j,k,l}\operatorname{cov}\left((x_i-x_j)(y_i-y_j),(x_k-x_l)(y_k-y_l)\right) \tag{A.15}$$

It is clear that the case of $i=j$ and $k=l$ reduces the corresponding covariance term to 0. Besides those, there are 3 cases to consider:

1. Distinct i,j,k,l :In this case $\operatorname{cov}\left((x_i-x_j)(y_i-y_j),(x_k-x_l)(y_k-y_l)\right)=0$ by independence.

2. Distinct i,j in $\operatorname{cov}\left((x_i-x_j)(y_i-y_j),(x_i-x_j)(y_i-y_j)\right)$:

   There are $2n(n-1)$ terms fulfilling this case.

$$\begin{aligned}&\operatorname{cov}\left((x_i-x_j)(y_i-y_j),(x_i-x_j)(y_i-y_j)\right) \\ &\quad= \mathbb{E}\left((x_i-x_j)^2(y_i-y_j)^2\right) - \left(\mathbb{E}(x_i-x_j)(y_i-y_j)\right)^2 \\ &\quad= 2(c_2+\sigma_X^2\sigma_Y^2+2c_1^2) - 4c_1^2 \\ &\quad= 2c_2 + 2\sigma_X^2\sigma_Y^2\end{aligned}$$

3. Distinct i,j,k in $\operatorname{cov}\left((x_i-x_j)(y_i-y_j),(x_k-x_j)(y_k-y_j)\right)$: There are $4n(n-1)(n-2)$ terms fulfilling this case. After expressing each term with respect to difference from the mean, expanding out the products and some manipulation, we get

$$\mathbb{E}\left((x_i-x_j)(y_i-y_j)(x_k-x_j)(y_k-y_j)\right) = 3c_1^2 + c_2$$

which yields

$$\begin{aligned}\operatorname{cov}\left((x_i - x_j)(y_i - y_j), (x_k - x_j)(y_k - y_j)\right) &= (3c_1^2 + c_2) - 4c_1^2 \\ &= c_2 - c_1^2\end{aligned}$$

Summing up the weighted contributions of each case gives the desired result. □

A.2 Moments of the uniform distribution

In this section we list some useful statistics of the random variable $S \sim \text{Uniform}(-r, r)$ with the distribution $f_S(s) = \frac{1}{2r}$ in $[-r, r]$ and zero elsewhere. In the next section we will then apply these statistics to the curve noise model of (2.9).

It can be easily verified that:

$$\mu_S \triangleq \mathbb{E}(S) = \int_{-r}^{r} s \frac{1}{2r} ds = 0 \tag{A.16}$$

$$d_2(S) = \mathbb{V}(S) = \int_{-r}^{r} s^2 \frac{1}{2r} ds = \frac{r^2}{3} \tag{A.17}$$

$$\mathbb{E}(S^3) = \int_{-r}^{r} s^3 \frac{1}{2r} ds = 0 \tag{A.18}$$

$$d_4(S) = \mathbb{E}(S^4) = \int_{-r}^{r} s^4 \frac{1}{2r} ds = \frac{r^4}{5} \tag{A.19}$$

Let a new random variable $U = S^2$. Then it follows that:

$$\mu_U \triangleq \mathbb{E}(U) = \mathbb{E}(S^2) = \frac{r^2}{3} \tag{A.20}$$

$$\mathbb{E}(U^2) = \mathbb{E}(S^4) = \frac{r^4}{5} \tag{A.21}$$

$$d_2(S^2) = \mathbb{V}(U) = \mathbb{E}(U^2) - (\mathbb{E}(U))^2 = \frac{4}{45} r^4 \tag{A.22}$$

$$\mathbb{E}(U^3) = \int_{-r}^{r} s^6 \frac{1}{2r} ds = \frac{r^6}{7} \tag{A.23}$$

$$\mathbb{E}(U^4) = \mathbb{E}(S^8) = \frac{r^8}{9} \tag{A.24}$$

$$d_4(S^2) = \mathbb{E}((U - \mu_U)^4) = \frac{16}{945} r^8 \tag{A.25}$$

Let a new random variable $V = S^3$. It similarly follows that:

$$\mu_V \triangleq \mathbb{E}(V) = \mathbb{E}(S^3) = 0 \tag{A.26}$$

$$d_2(S^3) = \mathbb{E}(V^2) = \mathbb{E}(S^6) = \frac{r^6}{7} \tag{A.27}$$

$$\mathbb{V}(V) = \mathbb{E}(V^2) - (\mathbb{E}(U))^2 = \frac{r^6}{7} \tag{A.28}$$

$$\mathbb{E}(V^3) = \mathbb{E}(S^9) = 0 \tag{A.29}$$

$$d_4(S^3) = d_4(V) = \mathbb{E}(V^4) = \mathbb{E}(S^{12}) = \frac{r^{12}}{13} \tag{A.30}$$

Note also that:

1. Effect of Scaling:
$$\mathbb{E}((\alpha X)^n) = \alpha^n \mathbb{E}(X^n) \tag{A.31}$$
for a constant α.

2. Effect of Gaussian noise on variance:
$$d_4(X + \eta) = d_4(X) + 6\sigma^2 d_2(X) + 3\sigma^4 \tag{A.32}$$
for normally distributed noise $\eta \sim \mathcal{N}(o, \sigma^2)$ using the fact the odd moments of η are zero and that $\mathbb{E}(\eta^4) = 3\sigma^4$.

3. Effect of Gaussian noise on covariance:
$$\text{cov}(X + \eta_X, Y + \eta_Y) = \text{cov}(X, Y) \tag{A.33}$$
for zero-mean independent noise η_X and η_Y.

4. Effect of Gaussian noise on $c_2(X, Y)$:
$$c_2(X + \eta_X, Y + \eta_Y) = c_2(X, Y) + \sigma_0^2(\sigma_X^2 + \sigma_Y^2) + \sigma_0^4 \tag{A.34}$$
for zero mean independent identically distributed noise η_X and η_Y that are independent of X, Y and have variance σ_0^2.

A.3 Moments and Covariances

In this section we compute some useful moment and covariance identities relating the random variables X, Y and Z, which represent the observed point samples as a noisy function of the position S on the curve. We assume $S \sim \text{Uniform}(-r, r)$ implying the distribution

$f_S(s) = \frac{1}{2r}$ in $[-r, r]$ along the curve, and zero elsewhere. As stated earlier, we consider the observed quantities to be corrupted by independent identically distributed noise terms η_X, η_Y and η_Z normally distributed with variance σ_0^2. In what follows, we make use of some statistics of the uniform distribution derived in Appendix A.2.
First we consider $X = S + \eta_X$

$$\mu_X = \mathbb{E}(S + \eta_X) = \int_{-r}^{r} s\frac{1}{2r}ds = 0 \tag{A.35}$$

$$\boxed{d_2(X)} = \sigma_X^2 = \int_{-r}^{r} s^2\frac{1}{2r}ds + \sigma_0^2 = \frac{r^2}{3} + \sigma_0^2 \tag{A.36}$$

$$\tag{A.37}$$

$$\begin{aligned}\boxed{d_4(X)} &= \mathbb{E}(X + \eta_X - \mu_X)^4 = \mathbb{E}(S^4) + 6\mathbb{E}(S^2)\mathbb{E}(\eta_X^2) + \mathbb{E}(\eta_X^4) \\ &= \frac{r^4}{5} + 2\sigma_o^2 r^2 + 3\sigma_0^4\end{aligned} \tag{A.38}$$

Next we consider $Y = \frac{\kappa}{2}S^2 + \eta_Y$

$$\mu_Y = \frac{\kappa}{2}\mathbb{E}(S^2) = \frac{\kappa r^2}{6} \tag{A.39}$$

$$\begin{aligned}\boxed{d_2(Y)} = \sigma_Y^2 &= \left(\frac{\kappa}{2}\right)^2 d_2(S^2) + \sigma_0^2 \\ &= \frac{\kappa^2 r^4}{45} + \sigma_0^2\end{aligned} \tag{A.40}$$

$$\begin{aligned}\boxed{d_4(Y)} &= \left(\frac{\kappa}{2}\right)^4 d_4(S^2) + 6\sigma_0^2\left(\frac{\kappa}{2}\right)^2 d_2(S^2) + \mathbb{E}(\eta_Y^4) \\ &= \frac{\kappa^4 r^8}{945} + \frac{2}{15}\sigma_0^2\kappa^2 r^4 + 3\sigma_0^4\end{aligned} \tag{A.41}$$

We can also compute the first and second order covariance between X and Y:

$$\boxed{c_1(X, Y)} = \mathrm{cov}(X, Y) = \frac{\kappa}{2}\mathrm{cov}(S, S^2) = 0 \tag{A.42}$$

$$\begin{aligned}\boxed{c_2(X, Y)} &= c_2(S, \frac{\kappa}{2}S^2) + \sigma_0^2\left(d_2(S) + \left(\frac{\kappa}{2}\right)^2 d_2(S^2)\right) + \sigma_0^4 \\ &= \frac{11}{945}\kappa^2 r^6 + \sigma_0^2\left(\frac{r^2}{3} + \left(\frac{\kappa}{2}\right)^2\frac{4}{45}r^4\right) + \sigma_0^4 \\ &= \frac{11}{945}\kappa^2 r^6 + \sigma_0^2\left(\frac{r^2}{3} + \frac{\kappa^2}{45}r^4\right) + \sigma_0^4\end{aligned} \tag{A.43}$$

Next we consider $Z = \frac{\kappa\tau}{6}S^3 + \eta_Z$

$$\mu_Z = \mathbb{E}(\frac{\kappa\tau}{6}S^3 + \eta_Z) = 0 \tag{A.44}$$

$$\boxed{d_2(Z)} = \sigma_Z^2 = \left(\frac{\kappa\tau}{6}\right)^2 \int_{-r}^{r} s^6 \frac{1}{2r} ds + \sigma_0^2 = \left(\frac{\kappa\tau}{6}\right)^2 \frac{r^6}{7} + \sigma_0^2 \tag{A.45}$$

$$\begin{aligned} \boxed{d_4(Z)} &= \left(\frac{\kappa\tau}{6}\right)^4 d_4(S^3) + 6\left(\frac{\kappa\tau}{6}\right)^2 d_2(S^3) + \mathbb{E}(\eta_Y^4) \\ &= \left(\frac{\kappa\tau}{6}\right)^4 \frac{r^{12}}{13} + 6\left(\frac{\kappa\tau}{6}\right)^2 \frac{r^6}{7} + 3\sigma_0^4 \\ &= \left(\frac{\kappa\tau}{6}\right)^4 \frac{r^{12}}{13} + \frac{(\kappa\tau)^2}{42} r^6 + 3\sigma_0^4 \end{aligned} \tag{A.46}$$

We can also compute the first and second order covariance between Z and X and Y.

$$\boxed{c_1(X,Z)} = cov(X,Z) = \frac{\kappa\tau}{6} cov(S, S^3) = \frac{\kappa\tau}{30} r^4 \tag{A.47}$$

$$\boxed{c_1(Y,Z)} = cov(Y,Z) = \frac{\kappa^2\tau}{12} cov(S^2, S^3) = 0 \tag{A.48}$$

$$\begin{aligned} \boxed{c_2(X,Z)} &= c_2(S, \frac{\kappa\tau}{6}S^3) + \sigma_0^2\left(d_2(S) + d_2(\frac{\kappa\tau}{6}S^3)\right) + \sigma_0^4 \\ &= \left(\frac{\kappa\tau}{6}\right)^2 \mathbb{E}(S^8) + \sigma_0^2\left(\frac{r^2}{3} + \left(\frac{\kappa\tau}{6}\right)^2 \frac{r^6}{7}\right) + \sigma_0^4 \\ &= \left(\frac{\kappa\tau}{6}\right)^2 \frac{r^8}{9} + \sigma_0^2\left(\frac{r^2}{3} + \left(\frac{\kappa\tau}{6}\right)^2 \frac{r^6}{7}\right) + \sigma_0^4 \end{aligned} \tag{A.49}$$

$$\begin{aligned} \boxed{c_2(Y,Z)} &= c_2(\frac{\kappa}{2}S^2, \frac{\kappa\tau}{6}S^3) + \sigma_0^2\left(d_2(\frac{\kappa}{2}S^2) + d_2(\frac{\kappa\tau}{6}S^3)\right) + \sigma_0^4 \\ &= \left(\frac{\kappa^2\tau}{12}\right)^2 c_2(S^2, S^3) + \sigma_0^2\left(\frac{\kappa^2}{4} d_2(S^2) + \left(\frac{\kappa\tau}{6}\right)^2 d_2(S^3)\right) + \sigma_0^4 \\ &= \left(\frac{\kappa^2\tau}{12}\right)^2 \frac{68}{2079} r^{10} + \sigma_0^2\left(\frac{\kappa^2 r^4}{45} + \left(\frac{\kappa\tau}{6}\right)^2 \frac{r^6}{7}\right) + \sigma_0^4 \end{aligned} \tag{A.50}$$

A.4 Estimators of the covariance matrix

In this section, we compute the expected values for each entry in the covariance matrix using the results from Appendix A.3 as well as the identities for the variances of sample variance and sample covariance estimators derived in Appendix A.1.

The estimate of the covariance matrix is then given by:

$$\begin{aligned}\hat{M}_n &= \begin{bmatrix} \sum_i (x_i - \bar{X}_n)^2 & \sum_i (x_i - \bar{X}_n)(y_i - \bar{Y}_n) & \sum_i (x_i - \bar{X}_n)(z_i - \bar{Z}_n) \\ \sum_i (x_i - \bar{X}_n)(y_i - \bar{Y}_n) & \sum_i (y_i - \bar{Y}_n)^2 & \sum_i (y_i - \bar{Y}_n)(z_i - \bar{Z}_n) \\ \sum_i (x_i - \bar{X}_n)(z_i - \bar{Z}_n) & \sum_i (y_i - \bar{Y}_n)(z_i - \bar{Z}_n) & \sum_i (z_i - \bar{Z}_n)^2 \end{bmatrix} \\ &= \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{12} & M_{22} & M_{23} \\ M_{13} & M_{23} & M_{33} \end{bmatrix}\end{aligned} \tag{A.51}$$

The first and second moments of these quantities follow as:

1. M_{11} :

$$\mathbb{E}(M_{11}) = \mathbb{V}(X_i) = \mathbb{V}(S_i) + \sigma_0^2 = \frac{r^2}{3} + \sigma_0^2 \tag{A.52}$$

$$\begin{aligned}\mathbb{V}(M_{11}) &= \frac{d_4(X_i)}{n} - \frac{(n-3)}{n(n-1)}\sigma_X^4 \\ &= \frac{1}{5n}\left(r^4 + 10\sigma_0^2 r^2 + 15\sigma_0^4\right) + \frac{(n-3)}{9n(n-1)}\left(r^2 + 3\sigma_0^2\right)^2\end{aligned} \tag{A.53}$$

2. M_{12} :

$$\mathbb{E}(M_{12}) = \mathrm{cov}(X, Y) = \frac{\kappa}{2}\mathbb{E}(S^3) = 0 \tag{A.54}$$

and

$$\mathbb{V}(M_{12}) = \frac{1}{n}\left(\frac{11}{945}\kappa^2 r^6 + \sigma_0^4 + \sigma_0^2(\frac{r^2}{3} + \frac{\kappa^2 r^4}{45})\right) + \frac{\left(\frac{r^2}{3} + \sigma_0^2\right)\left(\frac{\kappa^2 r^4}{45} + \sigma_0^2\right)}{n(n-1)} \tag{A.55}$$

3. M_{13} :

$$\mathbb{E}(M_{13}) = \mathrm{cov}(X, Z) = \frac{\kappa\tau}{6}\mathbb{E}(S^4) = \frac{\kappa\tau}{30}r^4 \tag{A.56}$$

and

$$\begin{aligned}\mathbb{V}(M_{13}) =& \frac{1}{n}\left(\left(\frac{\kappa\tau}{6}\right)^2 \frac{r^8}{9} + \sigma_0^2\left(\frac{r^2}{3} + \left(\frac{\kappa\tau}{6}\right)^2 \frac{r^6}{7}\right) + \sigma_0^4\right) \\ &+ \frac{1}{n(n-1)}\left(\frac{r^2}{3} + \sigma_0^2\right)\left(\left(\frac{\kappa\tau}{6}\right)^2 \frac{r^6}{7} + \sigma_0^2\right) \\ &+ \frac{(n-2)}{n(n-1)}\left(\frac{\kappa\tau}{30}\right)^2 r^8\end{aligned} \tag{A.57}$$

4. M_{22} :

$$\mathbb{E}(M_{22}) = \mathbb{V}(Y_i) = \frac{\kappa^2}{4}\mathbb{V}(S^2) + \sigma_0^2 = \frac{\kappa^2}{45}r^4 + \sigma_0^2 \tag{A.58}$$

and

$$\begin{aligned}\mathbb{V}(M_{22}) &= \frac{d_4(Y)}{n} - \frac{(n-3)}{n(n-1)}\sigma_Y^4 \\ &= \frac{1}{n}\left(\frac{\kappa^4 r^8}{945} + \frac{2}{15}\sigma_0^2\kappa^2 r^4 + 3\sigma_0^4\right) + \frac{(n-3)}{n(n-1)}\left(\frac{\kappa^2 r^4}{45} + \sigma_0^2\right)^2\end{aligned} \tag{A.59}$$

5. M_{23} :

$$\mathbb{E}(M_{23}) = \operatorname{cov}(Y, Z) = \frac{\kappa^2\tau}{18}(\mathbb{E}(S^5) - \mathbb{E}(S^2)\mathbb{E}(S^3)) = 0 \tag{A.60}$$

and

$$\begin{aligned}\mathbb{V}(M_{23}) =& \frac{1}{n}\left(\left(\frac{\kappa^2\tau}{12}\right)^2 \frac{68}{2079}r^{10} + \sigma_0^2\left(\frac{\kappa^2 r^4}{45} + \left(\frac{\kappa\tau}{6}\right)^2\frac{r^6}{7}\right) + \sigma_0^4\right) \\ &+ \frac{1}{n(n-1)}\left(\frac{\kappa^2 r^4}{45} + \sigma_0^2\right)\left(\left(\frac{\kappa\tau}{6}\right)^2\frac{r^6}{7} + \sigma_0^2\right)\end{aligned} \tag{A.61}$$

6. M_{33} :

$$\mathbb{E}(M_{33}) = \mathbb{V}(Z_i) = \left(\frac{\kappa\tau}{6}\right)^2\frac{r^6}{7} + \sigma_0^2 \tag{A.62}$$

and

$$\mathbb{V}(M_{33}) = \frac{1}{n}\left(\left(\frac{\kappa\tau}{6}\right)^4\frac{r^{12}}{13} + \frac{(\kappa\tau)^2}{42}r^6 + 3\sigma_0^4\right) + \frac{(n-3)}{n(n-1)}\left(\left(\frac{\kappa\tau}{6}\right)^2\frac{r^6}{7} + \sigma_0^2\right)^2 \tag{A.63}$$

Observe that the estimator for sample covariance matrix may be expressed as the sum of the matrix of its expected value and a matrix of random variables as

$$\hat{M} = \tilde{M} + Q, \tag{A.64}$$

where $\tilde{M} = \mathbb{E}(M)$ is a symmetric matrix with elements given by the equations (A.52), (A.54), (A.56), (A.58), (A.60) and (A.62), and Q is a symmetric *perturbation matrix* of random variables each with mean 0 and variance given by the equations (A.53) (A.55), (A.57), (A.59), (A.61), and (A.63).

APPENDIX B

Matrix Perturbation

This appendix summarizes some known results from matrix perturbation theory that analyze the effect of numerical errors (or perturbations) on the solution of eigenvector and generalized eigenvector problems. The purpose of this summary is to provide sufficient background to understand the origin of the error bounds in equation (2.26) stated in Section 2.3 which is central to the finite-sample error analysis used in Chapter 2. It is also to state its extension to the generalized eigenvector problem that is relevant to initializing the constrained regression problem of Chapter 3 using the HEIV estimator of (3.28).

An elaborate derivation of the theorems used to obtain the formulate we are interested in is well beyond the scope of this appendix and is the subject of several textbooks [20, 56, 100]. We will restrict ourselves to simply presenting some general theorems and their implications, and invoke these theorems to study problems characteristic of geometric fitting. We refer the interested reader to [100] for a detailed exposition.

B.1 Perturbation of eigenvectors

In our study of geometric fitting problems, the desired model parameters are often estimated as the principal (or sometimes, the minimal) eigenvector of a coefficient matrix. For many problems of interest, is possible to construct the coefficient matrix so that in the ideal scenario of an infinite amount of noise free data, the estimated models parameters are the true values of the system. However in practice, as shown in Chapter 2, the coefficient matrices are formed from a finite number of noisy observations whose effect is to perturb the estimated model parameters away from their true values.

In this section, we will look at eigenvector problems of the form

$$A\mathbf{e} = \lambda\mathbf{e},$$

where A is a real symmetric $n \times n$ positive semidefinite matrix, $\mathbf{e}$ is an eigenvector and λ is an eigenvalue of the system.

Note that the reason for restricting our discussion to real symmetric positive semidefinite matrices is intentional and appropriate for the geometric fitting problem. It may be easily seen from the construction of the matrices corresponding to our A, namely the scatter matrix M in (2.10) of Chapter 2, and the $S(\theta)$ matrix in (3.29) and (3.22) of Chapter 3, that they are guaranteed to be symmetric positive semidefinite.

A slightly way to express the eigenvector problem is as

$$AE = EL, \tag{B.1}$$

where

- $E = [\,\mathbf{e}_1\ \mathbf{e}_1\ \cdots\ \mathbf{e}_n\,]$ are the eigenvectors of A stacked columnwise, and
- $L = \mathrm{diag}\,([\,\lambda_1\ \lambda_1\ \cdots\ \lambda_n\,])$ is a diagonal matrix of the eigenvalues.

The ordering of eigenvalues is assumed to be $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ without loss of generality. We will also denote the set of eigenvalues of a matrix, say A, by $\mathcal{L}(A)$.

It will be useful to partition the columns of E as $[\,E_1\ E_2\,]$. Correspondingly, the diagonal matrix L may also be partitioned into diagonal sub-blocks as $L = \mathrm{diag}(\lambda_1, L_2)$.

The range of the columns of any non-empty partition, say E_1 is denoted by $\mathcal{R}(E_1)$. Each partition of matrix E forms an *invariant subspace* of A having the property that $\mathcal{R}(AE_1) \subset \mathcal{R}(E_1)$.

In what follows we will be interested in a symmetric positive semidefinite matrix Q that perturbs the coefficient matrix A as $\tilde{A} = A + Q$, and study its effect on the the original leading eigenvector $\mathbf{e}_1$. We will denote the corresponding principal eigen vector-value pair of the perturbed matrix $\tilde{A}$ as $\tilde{\mathbf{e}}_1$ and $\tilde{\lambda}_1$ respectively.

The following theorem by C. Davis and W. M. Kahan [29] is one incarnation of what is referred to as the '$\sin\Theta$" theorem. The theorem is so called because it bounds the sum of squares of the sines of the canonical angles between an invariant subspace of A and an approximation of that subspace.

Theorem 2. *(Originally in [29], and appearing as Theorem V.3.4 in [100])*
Let matrix A have the spectral decomposition

$$\begin{bmatrix} E_1 & E_2 \end{bmatrix}^T A \begin{bmatrix} E_1 & E_2 \end{bmatrix} = \boldsymbol{diag}(L_1, L_2) \tag{B.2}$$

where $[\, E_1 \; E_2 \,]$ *is unitary with* $E_1 \in \mathbb{R}^{n \times k}$.

Consider any matrix $Z \in \mathbb{R}^{n \times k}$ *having orthonormal columns, and any symmetric real matrix* D *of order* k. *Let us define a residual matrix* R *as*

$$R \triangleq AZ - ZD \tag{B.3}$$

If

$$\hat{\delta} = \min |\mathcal{L}(L_2) - \mathcal{L}(D)| > 0 \tag{B.4}$$

then

$$\sin \|\Theta\left(\mathcal{R}(E_1), \mathcal{R}(Z)\right)\|_F \leq \frac{\|R\|_F}{\hat{\delta}}. \tag{B.5}$$

where $\Theta\left(\mathcal{R}(E_1), \mathcal{R}(Z)\right)$ *is the matrix of canonical angles between the two subspaces.*

Thus, this theorem allows us to upper bound between the subspace spanned by the columns of matrix E_1 and the subspace spanned by other matrix Z that is "close" to E_1.

Since the partition of E, as well as Z and D are arbitrary, we may use this theorem to assess the effect of a perturbation to the matrix A on the estimated eigenvector as follows.

Let $E_1 = \mathbf{e}_1$ so that $E = [\, \mathbf{e}_1 \; E_2 \,]$ and so $L_1 = \lambda_1$. Let us now assign the matrices Z and D as $Z = \tilde{\mathbf{e}}_1$ and $D = \tilde{\lambda}_1$.

Then the implication of the above theorem is that the sine of the angle between the true eigenvector $\mathbf{e}_1$ and the approximate eigenvector $\tilde{\mathbf{e}}_1$ may be bounded as

$$\sin \angle(\mathbf{e}_1, \tilde{\mathbf{e}}_1) \leq \frac{\|R\|_F}{\hat{\delta}}. \tag{B.6}$$

With this bound and the relations that precede it, we may proceed to try and estimate the terms $\hat{\delta}$ and $\|R\|_F$ using available quantities.

From the above assignment, the value of δ in the denominator may be lower bounded as

$$\begin{aligned} \hat{\delta} &= \min |\mathcal{L}(L_2) - \tilde{\lambda}_1| \\ &\geq \min |\mathcal{L}(L_2) - \lambda_1| \\ &= \lambda_1 - \lambda_2 \end{aligned} \tag{B.7}$$

which is simply the spectral gap δ_A of matrix A. Note that the inequality arises from Q being a positive semi-definite matrix.

The numerator may be upper bounded as

$$\begin{aligned} R \triangleq AZ - ZD &= A\tilde{\mathbf{e}}_1 - \tilde{\mathbf{e}}_1\tilde{\lambda}_1 \\ &= (\tilde{A} - Q)\tilde{\mathbf{e}}_1 - \tilde{\lambda}_1\tilde{\mathbf{e}}_1 \\ &= -Q\tilde{\mathbf{e}}_1, \end{aligned} \tag{B.8}$$

from which $\|R\|_F \leq \|Q\|_F$ arises out the property of Frobenius norms.

Thus we obtain the desired relation

$$\sin\angle(\mathbf{e}_1, \tilde{\mathbf{e}}_1) \leq \frac{\|Q\|_F}{\delta_A} \tag{B.9}$$

B.2 The Generalized eigenvector problem

In this section, we will look at eigenvector problems of the form

$$A\mathbf{e} = \lambda B\mathbf{e},$$

where A and B are real symmetric $n \times n$ positive semidefinite matrices, $\mathbf{e}$ is an eigenvector and λ is an eigenvalue of the system. Again, our reason for this restriction may be justified from inspecting (3.22) and (3.23) from Chapter 3. We will confine ourselves to presenting only the main results for the case of positive definite matrices and again refer the interested reader to [100] for details.

The concept of an invariant subspace in the case of real symmetric matrices extends to *eigenspaces* in the case of generalized eigenvector problems. If (A, B) form a matrix pair having non-zero eigenvalues, then the subspace formed by columns of E is an eigenspace if

$$\dim(AE + BE) \leq \dim(E) \tag{B.10}$$

where dim denotes the dimensionality of the subspace.

The theorem below considers the perturbed system $\tilde{A} = A + Q_A$ and $\tilde{B} = B + Q_B$ and the estimates of its corresponding eigenspaces.

Theorem 3. *(Originally in [101], and appears as Theorem VI.3.9 in [100])*
Let A and B have the spectral decomposition

$$\begin{bmatrix} \mathbf{e}_1 & E_2 \end{bmatrix}^T A \begin{bmatrix} \mathbf{e}_1 & E_2 \end{bmatrix} = \begin{bmatrix} \lambda_{A1} & 0 \\ 0 & L_{A2} \end{bmatrix} \tag{B.11}$$

$$\begin{bmatrix} \mathbf{e}_1 & E_2 \end{bmatrix}^T B \begin{bmatrix} \mathbf{e}_1 & E_2 \end{bmatrix} = \begin{bmatrix} \lambda_{B1} & 0 \\ 0 & L_{B2} \end{bmatrix}, \tag{B.12}$$

where the columns of $[\,\mathbf{e}_1\ E_2\,]$ *are the generalized eigenvectors of* (A, B).

Define

$$\hat{\delta} \triangleq \min\{\chi(\lambda, \tilde{\lambda}) : \lambda \in \mathcal{L}(\lambda_{A1}, \lambda_{B1}), \tilde{\lambda} \in \mathcal{L}(\tilde{L}_{A2}, \tilde{L}_{B2})\} > 0 \tag{B.13}$$

where the chordal distance $\chi(\lambda, \mu)$ *between two eigenvalues* λ *and* μ *is defined as*

$$\chi(\lambda, \mu) = \frac{\|\lambda - \mu\|}{\sqrt{1+\lambda^2}\sqrt{1+\mu^2}}, \tag{B.14}$$

Then

$$\sin\angle(\mathbf{e}_1, \tilde{\mathbf{e}}_1) \leq \frac{\sqrt{\|A^2 + B^2\|_2}}{\gamma(A, B)\gamma(\tilde{A}, \tilde{B})} \frac{\sqrt{\|Q_A \mathbf{e}_1\|_F^2 + \|Q_B \mathbf{e}_1\|_F^2}}{\hat{\delta}} \tag{B.15}$$

where the Crawford number $\gamma(A, B)$ *is defined as*

$$\gamma(A, B) \triangleq \min_{\|\mathbf{e}\|_2 = 1} \sqrt{(\mathbf{e}^T A \mathbf{e})^2 + (\mathbf{e}^T B \mathbf{e})^2} \tag{B.16}$$

As done in the previous section, we can lower bound the terms in the denominator and upper bound the terms in the numerator to yield

$$\sin\angle(\mathbf{e}_1, \tilde{\mathbf{e}}_1) \leq \frac{\sqrt{\|A^2 + B^2\|_2}}{\gamma(A, B)^2} \frac{\sqrt{\|Q_A\|_F^2 + \|Q_B\|_F^2}}{\min(\delta_A, \delta_B)} \tag{B.17}$$

Bibliography

[1] 3dcafe. `http://www.3dcafe.com`. 1

[2] Large geometric models archive at georgia institute of technology. `http://www.cc.gatech.edu/projects/large_models/`. 1

[3] Stanford 3D scanning repository. `http://graphics.stanford.edu/data/3Dscanrep`. 1

[4] Turbosquid. `http://www.turbosquid.com`. 1

[5] Velodyne lidar systems. `http://www.velodyne.com/lidar`. 1

[6] A. Adamson and M. Alexa. Approximating and intersecting surfaces from points. In *Proc. of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 230–239, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association. 13

[7] A. Adamson and M. Alexa. Anisotropic point set surfaces. In *Afrigaph '06*, pages 7–13, New York, NY, USA, 2006. ACM Press. 13

[8] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva. Point set surfaces. In *Proceedings of the Conference on Visualization (VIS)*, pages 21–28, Washington, DC, USA, 2001. IEEE Computer Society. 13

[9] N. Amenta and M. Bern. Surface reconstruction by voronoi filtering. *Discrete Computational Geometry*, 22(4):481–504, 1999. 14

[10] N. Amenta, M. Bern, and M. Kamvysselis. A new voronoi-based surface reconstruction algorithm. In *Proc. of the Eurographics/ACM SIGGRAPH*, 1998. 14

[11] N. Amenta, S. Choi, T. K. Dey, and N. Leekha. A simple algorithm for homeomeorphic surface reconstruction. In *Proc. 16th ACM Annual Symposium on Computational Geometry*, pages 213–222, 2000. 14

[12] S. Arya, D. M. Mount, N. S. Netenyahu, R. Silverman, and A. Wu. An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. *Journal of the ACM*, 45(6):891–923, 1998. 94

[13] K. Barrett. Degenerate polynomial forms. *Communications in numerical methods in engineering*, 15(5), 1999. 56

[14] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003. 93, 94

[15] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, 24(4), 2002. 6, 7, 69

[16] Y. Bengio, O. Delalleau, N. Le Roux, J.-F. Paiement, P. Vincent, and M. Ouimet. Learning eigenfunctions links spectral embedding and kernel PCA. *Neural Computation*, 16(10):2197–2219, 2004. 97

[17] Y. Bengio, J.-F. Paiement, P. Vincent, O. Delalleau, N. L. Roux, and M. Ouimet. Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, volume 16. MIT Press, 2004. 97

[18] M. Bernstein, V. de Silva, J. Langford, and J. Tenenbaum. Graph approximations to geodesics on embedded manifolds. Technical report, Stanford University, 2000. 94

[19] P. J. Besl and R. C. Jain. Invariant surface characteristics for 3D object recognition in range images. *Comput. Vision Graph. Image Process.*, 33(1):33–80, 1986. 2

[20] R. Bhatia. *Matrix Analysis*. Springer, 1996. 113

[21] M. Botsch, M. Spernat, and L. Kobbelt. Phong splatting. In *Eurographics Symposium on Point-Based Graphics*, pages 25–32, 2004. 13

[22] J. Bunch and C. Nielsen. Updating the singular value decomposition. *Numerical Mathematics*, 31(2):131–152, 1978. 49

[23] O. Carmichael, D. Huber, and M. Hebert. Large data sets and confusing scenes in 3-d surface matching and recognition. In *Proceedings of the Second International Conference on 3D Digital Imaging and Modeling (3DIM)*, pages 358–367, October 1999. 8

[24] M. A. Carreira-Perpinán and R. S. Zemel. Proximity graphs for clustering and manifold learning. In *Neural Information Processing Systems (NIPS)*, pages 225–232, 2004. 37, 78, 93, 95

[25] F. Cazals, J. Giesen, and M. Yvinec. Delaunay triangulation based surface reconstruction: a short survey. Research Report 5394, INRIA, 2004. 14

[26] F. Cazals and M. Pouget. Estimating differential quantities using polynomial fitting of osculating jets. In *Proc. of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, pages 177–187, 2003. 14

[27] W. Chojnacki, M. J. Brooks, A. van den Hengel, and D. Gawley. FNS, CFNS and HEIV: A unifying approach. *Journal of Mathematical Imaging and Vision*, 2005. 58, 60

[28] F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997. 30, 31, 75

[29] C. Davis and W. M. Kahan. The rotation of eigenvectors by a perturbation III. *SIAM Journal on Numerical Analysis*, 7(1):1–46, 1970. 114

[30] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proc. of the Eurographics/ACM SIGGRAPH*, 1999. 3, 72

[31] T. K. Dey and S. Goswami. Provable surface reconstruction from noisy samples. In *Proc. 20th ACM Annual Symposium on Computational Geometry*, pages 330–339, 2004. 14

[32] M. do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice Hall, 1976. 33

[33] H. Edelsbrunner, D. Kirkpatrick, and R. Seidel. On the shape of a set of points in the plane. *IEEE Transactions on Information Theory*, 29:551–559, 1983. 14

[34] H. Edelsbrunner and E. P. Mücke. Three-dimensional alpha shapes. In *ACM Transactions on Graphics*, volume 13, pages 43–72, 1994. 14

[35] A. Elgammal, R. Duraiswami, and L. Davis. Efficient kernel density estimation using the fast gauss transform with applications to color modeling and tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 25(11):1499–1504, 2003. 96, 97

[36] D. Eppstein. Offline algorithms for dynamic minimum spanning tree problems. In *Workshop on Algorithms and Data Structures*, pages 392–399, 1991. 49

[37] D. Eppstein, G. L. Miller, and S.-H. Teng. A deterministic linear time algorithm for geometric separators and its applications. In *Proceedings of the ninth annual symposium on Computational geometry (SCG)*, pages 99–108, New York, NY, USA, 1993. ACM. 49

[38] P. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision (IJCV)*, 59(2):167–181, 2004. 93, 94

[39] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 264–271, 2003. 7

[40] A. Frome, D. Huber, R. Kolluri, T. Bulow, and J. Malik. Recognizing objects in range data using regional point descriptors. In *Proc. European Conf. on Computer Vision (ECCV)*, May 2004. 9, 69

[41] T. Funkhouser, P. Min, M. Kazhdan, J. Chen, A. Halderman, D. Dobkin, and D. Jacobs. A search engine for 3D models. In *Proc. of the Eurographics/ACM SIGGRAPH*, volume 22, pages 83–105, 2003. 1

[42] P. E. Gill, G. H. Golub, W. Murray, and M. A. Saunders. Methods for modifying matrix factorizations. *Mathematics of Computation*, 28(126):505–535, 1974. 49

[43] J. Goodman and J. O. Rourke, editors. *Handbook of Discrete and Computational Geometry*, chapter 30. CRC press, 2nd edition, 2004. 14

[44] S. Gumhold, X. Wang, and R. McLeod. Feature extraction from point clouds. In *Proceedings of 10th International Meshing Roundtable*, pages 293–305, 2001. 9, 93, 94

[45] T. Hastie and C. Loader. Local regression: Automatic kernel carpentry. *Statistical Science*, 8(2):120–129, 1993. 54

[46] M. Hein, J. Audibert, and U. von Luxburg. From graphs to manifolds - weak and strong pointwise consistency of graph laplacians. In *Proc. of the 18th Conference on Learning Theory (COLT)*, pages 470–485, 2005. 75, 76

[47] G. Hetzel, B. Leibe, P. Levi, and B. Schiele. 3D object recognition from range images using local feature histograms. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 394–399, 2001. 2

[48] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. *Computer Graphics*, 26(2):71–78, 1992. 14, 19, 48

[49] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. In *Proc. of the Eurographics/ACM SIGGRAPH*, pages 71–78, 1992. 54, 93

[50] J. Huang, A. Lee, and D. Mumford. Statistics of range images. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 324–331, 2000. 2

[51] A. Hubeli and M. Gross. Multiresolution methods for nonmanifold models. *IEEE Transactions on Visualization and Computer Graphics*, 7(3):207–221, 2001. 55

[52] T. Iijima. Basic theory of pattern observation. In *Papers of Technical Group on Automata and Automatic Control, IECE*, 1959. 70, 73

[53] A. Johnson and M. Hebert. Using spin images for efficient object recognition in 3D scenes. *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, 21(5), 1999. 6, 7, 13, 43, 69

[54] T. R. Jones, F. Durand, and M. Desbrun. Non-iterative, feature-preserving mesh smoothing. In *Proc. of the Eurographics/ACM SIGGRAPH*, 2003. 3

[55] K. Kanatani. Statistical optimization for geometric fitting: Theoretical accuracy analysis and high order error analysis. In *21st Intl. Conf. on Image and Vision Computing New Zealand (IVCNZ)*, 2006. 17, 21, 22

[56] T. Kato. *Perturbation Theory for Linear Operators*. Springer, 1995. 113

[57] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz. Rotation invariant spherical harmonic representation of 3D shape descriptors. In *Symposium on Geometry Processing*, June 2003. 6, 69

[58] B. Kégl, A. Kryzak, T. Linder, and K. Zeger. Learning and design of principal curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 22(3):281–297, 2000. 23, 24

[59] L. Kobbelt. Discrete fairing. In *Proc. of the Eurographics/ACM SIGGRAPH*, 1998. 3, 72

[60] L. Kobbelt and M. Botsch. A survey of point-based techniques in computer graphics. In *Computers & Graphics*, volume 28, pages 801–814, 2004. 13

[61] J. Koenderink. The structure of images. *Biological Cybernetics*, 50:363–370, 1984. 71

[62] S. Lafon. *Diffusion maps and geometric harmonics.* PhD thesis, Yale University, 2004. 75, 76

[63] D. Langer and al. Imaging ladar for 3-d surveying and cad modeling of real world environments. *International Journal of Robotics Research*, 19(11), 2000. 41

[64] I.-K. Lee. Curve reconstruction from unorganized points. *Computer Aided Geometric Design*, 17(2):161–177, 2000. 14

[65] D. Levin. Mesh-independent surface interpolation. In *Geometric Modeling for Scientific Visualization*, pages 37–39, 2003. 54

[66] M. Levoy and T. Whitted. The use of points as a display primitive. Technical Report 85-022, University of North Carolina at Chapel Hill, 1985. 4, 13

[67] T. Lewiner, J. D. Gomez, H. Lopes, and M. Craizer. Curvature and torsion estimators based on parametric curve fitting. *Computers & Graphics*, 29(5):641–655, 2005. 13, 14, 19, 29, 36, 37, 41, 42

[68] X. Li and I. Guskov. Multi-scale features for approximate alignment of point-based surfaces. In *Symposium on Geometry Processing (SGP)*, 2005. 73

[69] Y. Li. On incremental and robust subspace learning. *Patter Recognition*, 37(7):1509–1518, 2004. 49

[70] T. Lindeberg. Feature detection with automatic scale selection. *Intl. Journal of Computer Vision (IJCV)*, 30(2):77–116, 1998. 69, 70, 71, 78

[71] D. G. Lowe. Object recognition from local scale-invariant features. In *Proc. Intl. Conf. on Computer Vision (ICCV)*, pages 1150–1157, 1999. 9

[72] D. G. Lowe. Distinctive image features from scale-invariant viewpoints. *Intl. Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004. 69, 71, 96

[73] D. J. C. MacKay. Gaussian processes for machine learning. In C. M. Bishop, editor, *Neural Networks and Machine Learning*, volume 168 of *NATO ASI*, pages 133–165. Springer, 1998. 49

[74] B. Matei and P. Meer. Estimation of nonlinear errors-in-variables models for computer vision applications. *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, 28:1537–1552, 2006. 17

[75] B. Matei and P. Meer. Estimation of nonlinear errors-in-variables models for computer vision applications. *IEEE Trans. PAMI*, 28(10):1537–1552, 2006. 53, 54, 58, 60, 62, 64, 65

[76] B. Matei, Y. Shan, H. Sawhney, Y. Tan, R. Kumar, D. Huber, and M. Hebert. Rapid object indexing using locality sensitive hashing and joint 3D-signature space estimation. *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, 28(7):1111 – 1126, July 2006. 9, 69

[77] G. Medioni and C. K. Tang. Inference of integrated surface, curve, and junction descriptions from sparse 3D data. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 20(11):1206–1223, 1998. 14, 23, 24, 48

[78] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In *VisMath*, 2002. 3

[79] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool. A comparison of affine region detectors. *Intl. Journal of Computer Vision (IJCV)*, 65(1/2):43–72, 2005. 7, 9, 69, 78, 82, 84

[80] J. Miller. *A 3D Color Terrain Modeling System for Small Autonomous Helicopters*. PhD thesis, Carnegie Mellon University, 2002. 41

[81] N. J. Mitra, A. Nguyen, and L. Guibas. Estimating surface normals in noisy point cloud data. *Special issue of Int. Journal of Computational Geometry and Applications*, 14(4):261–276, 2004. 15, 29, 30, 37, 48

[82] A. Y. Ng, A. X. Zheng, and M. Jordan. Link analysis, eigenvectors, and stability. In *Proc. of the Intl. Joint Conference on Artificial Intelligence (IJCAI)*, pages 903–910, 2001. 27

[83] J. Novatnack and K. Nishino. Scale-dependent 3D geometric features. In *Proc. Intl. Conf. on Computer Vision (ICCV)*, 2007. 73

[84] J. Novatnack, K. Nishino, and A. Shokoufandeh. Extracting 3D shape features in discrete scale-space. In *3D Processing, Visualization and Transmission (3DPVT)*, 2006. 73

[85] M. Pauly and M. Gross. Spectral processing of point-sampled geometry. In *Proc. of the Eurographics/ACM SIGGRAPH*, 2001. 13

[86] M. Pauly, M. Gross, and L. Kobbelt. Efficient simplification of point-sampled surfaces. In *Proc. IEEE Visualization*, 2002. 13

[87] M. Pauly, R. Keiser, and M. Gross. Multi-scale feature extraction on point-sampled surfaces. In *Eurographics*, 2003. 72, 84, 93, 94

[88] H. Pfister, M. Zwicker, J. van Baar, and M. Gross. Surfels: Surface elements as rendering primitives. In *Proc. of the Eurographics/ACM SIGGRAPH*, pages 335–342, 2000. 13

[89] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes in C*. Cambridge University Press, Cambridge, UK, 2nd edition, 1992. 54

[90] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006. 49

[91] V. Raykar and R. Duraiswami. *Large Scale Kernel Machines*, chapter The Improved Fast Gauss Transform with applications to machine learning. MIT Press, 2007. 96, 97

[92] V. C. Raykar and R. Duraiswami. Fast large scale gaussian process regression using approximate matrix-vector products. In *Learning Workshop, Peurto Rico*, 2007. 49, 50

[93] R. C. Reilly. Mean curvature, the laplacian, and soap bubbles. *The American Mathematical Monthly*, 89(3), 1982. 75

[94] S. Roweis and L. T. Saul. Non-linear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000. 93

[95] S. Rusinkiewicz and M. Levoy. QSplat: A multiresolution point rendering system for large meshes. In K. Akeley, editor, *Proc. of the Eurographics/ACM SIGGRAPH*, pages 343–352, 2000. 13

[96] M. Schlattmann. Intrinsic features on surfaces. In *Central European Seminar on Computer Graphics*, 2006. 72

[97] S.Fleishman, D. Cohen-Or, and C. T.Silva. Robust moving least-squares fitting with sharp features. In *Proc. of the Eurographics/ACM SIGGRAPH*, 2005. 54, 63

[98] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 22(8):888–905, 2000. 93

[99] O. Sorkine. Differential representations for mesh processing. *Computer Graphics Forum*, 25(4), 2006. 3

[100] G. W. Stewart and J.-G. Sun. *Matrix Perturbation Theory*. Academic Press, 1990. 27, 113, 114, 116

[101] J.-G. Sun. Perturbation analysis for the generalized singular value decomposition. *SIAM Journal on Numerical Analysis*, 20:611–625, 1983. 116

[102] R. Szeliski and D. Tonnesen. Surface modeling with oriented particle systems. *Computer Graphics*, 26(2):185–194, 1992. 13

[103] J. B. Tanenbaum, V. de Silva, and J. C. Langford. A global geometric framework for non-linear dimensionality reduction. *Science*, 290:2319–2323, 2000. 93, 94

[104] C. K. Tang, G. Medioni, P. Mordohai, and W. S. Tong. First order augmentations to tensor voting for boundary inference and multiscale analysis in 3D. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 26(5):594–611, 2004. 14, 15, 23, 24, 48

[105] G. Taubin. An improved algorithm for algebraic curve and surface fitting. In *Intl. Conf. on Computer Vision*, 1993. 52

[106] G. Taubin. A signal processing approach to fair surface design. In *Proc. of the Eurographics/ACM SIGGRAPH*, 1995. 3, 72

[107] G. Taubin, T. Zhang, and G. Golub. Optimal surface smoothing as filter design. In *Proc. European Conf. on Computer Vision (ECCV)*, 1996. 3

[108] B. Triggs. A new approach to geometric fitting. In *Proc. Intl. Conf. on Computer Vision (ICCV)*, 1998. 17

[109] J. Tuley, N. Vandapel, and M. Hebert. Analysis and removal of artifacts in 3-d ladar data. In *Proc. of the IEEE Conference on Robotics and Automation*, pages 2203 – 2210, April 2005. 5

[110] R. Unnikrishnan, J.-F. Lalonde, N. Vandapel, and M. Hebert. Scale selection for the analysis of point-sampled curves. In *Third International Symposium on 3D Processing, Visualization and Transmission (3DPVT 2006)*, June 2006. 15, 63

[111] N. Vandapel, D. Huber, A. Kapuria, and M. Hebert. Natural terrain classification using 3D ladar data. In *Proc. of the IEEE Conference on Robotics and Automation*, volume 5, pages 5117 – 5122, April 2004. 3

[112] A. Verroust and F. Lazarus. Extracting skeletal curves from 3d scattered data. *The Visual Computer*, 16(1):15–25, 2000. 93

[113] R. Vidal, Y. Ma, and S. Sastry. Generalized principal component analysis (GPCA). *IEEE Trans. Pattern Analysis and Machine Intelligence*, 27(12):1945–1959, 2005. 52, 62

[114] C. Walder, B. C. Lovell, and P. J. Kootsookos. Algebraic curve fitting support vector machines. In *Digital Image Computing Techniques and Applications*, pages 693–702, 2003. 49, 50

[115] C. Walder, B. Schölkopf, and O. Chapelle. Implicit surfaces with globally regularised and compactly supported basis functions. In *Advances in Neural Information Processing Systems*, 2007. 13, 49

[116] M. P. Wand and M. C. Jones. *Kernel Smoothing*. Chapman & Hall, 1994. 57

[117] J. Wang, M. M. Oliveira, and A. E. Kaufman. Reconstructing manifold and non-manifold surfaces from point clouds. In *Proc. IEEE Visualization*, 2005. 54

[118] L. Wasserman. *All of Statistics*. Springer, 2004. 25

[119] J. Weickert, S. Ishikawa, and A. Imiya. Linear scale-space has first been proposed in Japan. *Journal of Mathematical Imaging and Vision*, 10(3):237–252, 1999. 73

[120] H. Wendland. *Scattered Data Approximation*. Cambridge University Press, 2004. 65

[121] T. Willmore. *Riemannian Geometry*. Clarendon Press, 1993. 33

[122] A. Witkin. Scale-space filtering. In *IJCAI*, volume 2, pages 1019–1022, 1983. 71

[123] G. Xu. Convergent discrete laplace-beltrami operators over triangular surfaces. In *GMP*, pages 195–204, 2004. 3

[124] C. Yang, R. Duraiswami, and L. Davis. Efficient kernel machines using the improved fast gauss transform. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2004. 96

[125] L. Yang. Building k edge-disjoint spanning trees of minimum total length for isometric data embedding. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 27(10):1680–1683, 2005. 95

[126] X. Zhu. *Semi-Supervised Learning with Graphs*. PhD thesis, Carnegie Mellon University, 2005. 93

[127] M. Zwicker, M. Pauly, O. Knoll, and M. Gross. Pointshop 3D: An interactive system for point-based surface editing. In *Conf. on Computer Graphics and Interactive Techniques*, pages 322–329, 2002. 13, 14, 19, 48

[128] M. Zwicker, J. Räsänen, M. Botsch, C. Dachsbacher, and M. Pauly. Perspective accurate splatting. In *Proceedings of Graphics Interface (GI)*, pages 247–254, 2004. 13