Analysis of 3D objects at multiple scales: application to shape matching

Nicolas Mellado

N° d'ordre : 4685

THÈSE

présentée à

L'UNIVERSITÉ BORDEAUX 1

ÉCOLE DOCTORALE DE MATHÉMATIQUES ET D'INFORMATIQUE

par **Nicolas Mellado**

Pour obtenir le grade de

DOCTEUR

SPÉCIALITÉ : INFORMATIQUE

Analysis of 3D objects at multiple scales: application to shape matching

---

Soutenue le : 06/12/2012

Après avis des rapporteurs :

Pierre Alliez	Professeur
Niloy Mitra	Maitre de conference

Devant la commission d'examen composée de :

Pierre Alliez	Professeur	Rapporteur
Niloy Mitra	Maitre de conference	Rapporteur
Andrei Sharf	Maitre de conference	Examinateur
Pascal Guitton	Professeur	Examinateur
Christophe Schlick	Professeur	Directeur de Thèse
Patrick Reuter	Maitre de conference	Co-Directeur de Thèse

– 2012 –

Abstract

Over the last decades, the evolution of acquisition techniques yields the generalization of detailed 3D objects, represented as huge point sets composed of millions of vertices. The complexity of the involved data often requires to analyze them for the extraction and characterization of pertinent structures, which are potentially defined at multiple scales. Among the wide variety of methods proposed to analyze digital signals, the scale-space analysis is today a standard for the study of 2D curves and images. However, its adaptation to 3D data leads to instabilities and requires connectivity information, which is not directly available when dealing with point sets.

In this thesis, we present a new multi-scale analysis framework that we call the *Growing Least Squares* (GLS). It consists of a robust local geometric descriptor that can be evaluated on point sets at multiple scales using an efficient second-order fitting procedure. We propose to analytically differentiate this descriptor to extract continuously the pertinent structures in scale-space. We show that this representation and the associated toolbox define an efficient way to analyze 3D objects represented as point sets at multiple scales. To this end, we demonstrate its relevance in various application scenarios.

A challenging application is the analysis of acquired 3D objects coming from the Cultural Heritage field. In this thesis, we study a real-world dataset composed of the fragments of the statues that were surrounding the legendary Alexandria Lighthouse. In particular, we focus on the problem of fractured object reassembly, consisting of few fragments (up to about ten), but with missing parts due to erosion or deterioration. We propose a semi-automatic formalism to combine both the archaeologist's knowledge and the accuracy of geometric matching algorithms during the reassembly process. We use it to design two systems, and we show their efficiency in concrete cases.

Résumé

Depuis quelques années, l'évolution des techniques d'acquisition a entraîné une généralisation de l'utilisation d'objets 3D très dense, représentés par des nuages de points de plusieurs millions de sommets. Au vu de la complexité de ces données, il est souvent nécessaire de les analyser pour en extraire les structures les plus pertinentes, potentiellement définies à plusieurs échelles. Parmi les nombreuses méthodes traditionnellement utilisées pour analyser des signaux numériques, l'analyse dite *scale-space* est aujourd'hui un standard pour l'étude des courbes et des images. Cependant, son adaptation aux données 3D pose des problèmes d'instabilité et nécessite une information de connectivité, qui n'est pas directement définie dans les cas des nuages de points.

Dans cette thèse, nous présentons une suite d'outils mathématiques pour l'analyse des objets 3D, sous le nom de *Growing Least Squares* (GLS). Nous proposons de représenter la géométrie décrite par un nuage de points via une primitive du second ordre ajustée par une minimisation aux moindres carrés, et cela à pour plusieurs échelles. Cette description est ensuite derivée analytiquement pour extraire de manière continue les structures les plus pertinentes à la fois en espace et en échelle. Nous montrons par plusieurs exemples et comparaisons que cette représentation et les outils associés définissent une solution efficace pour l'analyse des nuages de points à plusieurs échelles.

Un défi intéressant est l'analyse d'objets 3D acquis dans le cadre de l'étude du patrimoine culturel. Dans cette thèse, nous nous étudions les données générées par l'acquisition des fragments des statues entourant par le passé le Phare d'Alexandrie, Septième Merveille du Monde. Plus précisément, nous nous intéressons au réassemblage d'objets fracturés en peu de fragments (une dizaine), mais avec de nombreuses parties manquantes ou fortement dégradées par l'action du temps. Nous proposons un formalisme pour la conception de systèmes d'assemblage virtuel semi-automatiques, permettant de combiner à la fois les connaissances des archéologues et la précision des algorithmes d'assemblage. Nous présentons deux systèmes basés sur cette conception, et nous montrons leur efficacité dans des cas concrets.

Contents

List of Figures

Chapter 1

Introduction

Scope of this research

Over the last decades, the evolution of acquisition techniques greatly simplified the digitization of surfaces of 3D objects. Starting from laser scans that produce range images from a given point of view, devices have been improved to move around the target (or to move the target in front of the device) to acquire all the sides of the object. Other methods, purely optical such as photogrammetry or shape-from-shading, reconstruct 3D objects by taking as input only sets of photos (see Figure 1.1 for an example). These evolutions provide more and more accessible approaches that can now be widely used by non-expert people without requiring expensive devices. Most often, the digitization is used to acquire accurately geometric details on the surface, producing data that contains various relevant information, from coarse structures to fine geometric details. However, the obtained data may also contain noise, due to variations in acquisition accuracy, and artifacts may appear (for example outliers and holes).

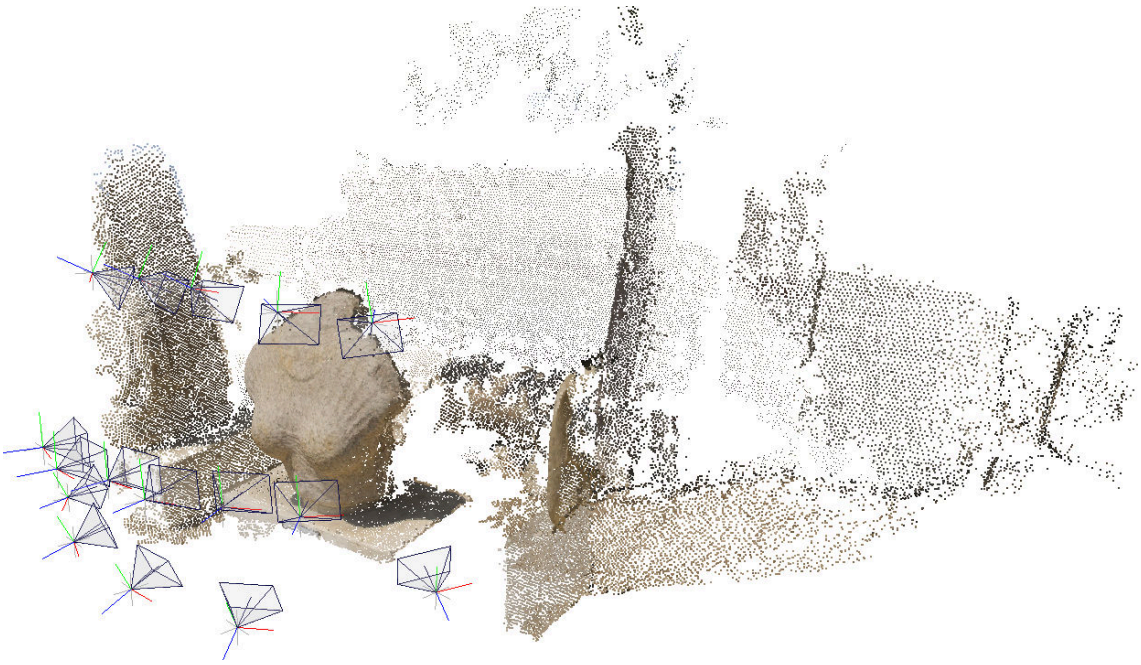


Figure 1.1: **Photogrammetry reconstruction**. Recent reconstruction techniques are able to reconstruct 3D objects consisting of millions of points by taking as input images of the real object from different points of view (shown as blue pyramids). Image generated with [DA12].

The ease of acquisition leads to a generalization of the use of 3D objects in various application fields, with different objectives, such as visualization, comprehension and analysis of object properties, matching, and retrieval. In addition to these final usages, the 3D data

are usually stored, transmitted and edited. The process involved in these different tasks should be *scalable*, which means it should handle any kind of data, independently of their complexity (within reasonable limits). Most often, the processing involves an intermediate step to compress or summarize the data. This step may produce an alternative data representation that facilitates the task to accomplish, associated with tools to analyze the object (e.g. noise detection, multi-scale feature extraction, and element comparison), or to edit it (e.g. noise removal, deformations, and hole filling).

The idea of the work presented in this thesis is to design an analysis method that is adapted to the different uses presented above, by respecting the 4 following criteria:

- **It should take as input the acquisition data directly**: the majority of acquisition techniques produce point sets, usually with noise, missing data, or acquisition artifacts: the analysis should be able to analyze them without any preliminary conversion preprocess.
- **It should have a good description power**: 3D objects usually contain various structures and patterns, defined at multiple scales (e.g. ridges and valleys, planar and constant curvature surfaces, or edges): the analysis should be able to characterize and disambiguate them.
- **It should be evaluated locally**: depending on the processing task, the 3D data might be analyzed in a preprocess on the entire object, or on the fly at specific locations: the analysis should have a local evaluation to support both approaches.
- **It should be versatile**: we do not make any assumption on the process that might use the analysis and the associated application context: the analysis should be parameterless, and it should produce a generic output associated with dedicated analysis and editing tools.

An acquired object is usually represented as a point set. One may be tempted to use remeshing techniques to convert the point set to a mesh, in order to use the wide variety of mesh-based analysis methods. However, in practice, this conversion step often takes a lot of time to treat the large amount of samples, and it even fails in some cases due to the input complexity. Furthermore, the remeshing process generates new additional connectivity information, and for efficiency reasons, it may require data simplification. According to our first criterion, this should be avoided to avoid an erroneous analysis.

The second criterion, the description power, is related to the potential richness of the acquired object's geometry, composed of numerous complex features defined at multiple scales. Consider for instance a relief ridge pitted with small cavities, as shown in Figure 1.2: the highlighted area inside one of the small pits belongs to a concave feature at a small scale, and to a convex feature at a larger scale. The analysis must be able to detect both of them, and furthermore, it has to describe them sufficiently accurately to disambiguate them. Another point is that we do not want to be restricted by any assumption on the object that will be analyzed, thus the analysis must describe a majority of shapes (such as ridges, valleys, edges, or planar areas) in arbitrary configurations.

The third criterion is related to the way the analysis is computed. For example, suppose that we want to compute slices on the bust shown in Figure 1.2, in order to perform some measurements. A solution could be to reconstruct a closed surface, which represents the entire object, and then compute its intersection with a slice plane. This approach may require extensive computation to reconstruct the entire surface. A better solution is to

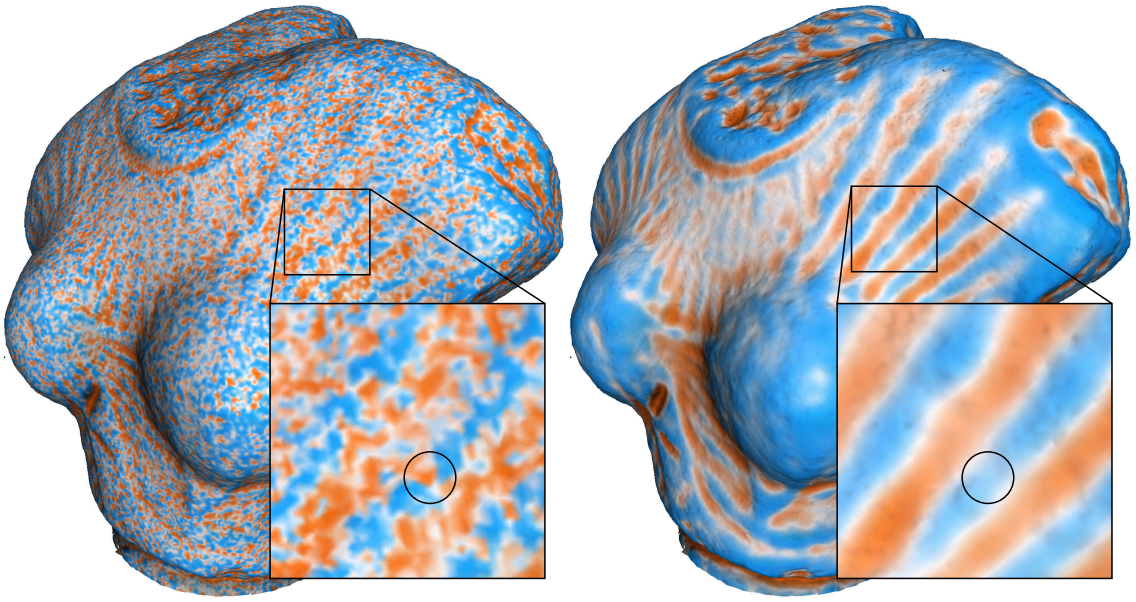


Figure 1.2: **Introduction to feature extraction on an acquired 3D object**. We display the curvature of the object as color on its surface, at a fine (left) and medium (right) scale. The highlighted area, inside one of the small pits, belongs respectively to a concave feature (in orange) and to a convex feature (in cyan).

reconstruct the surface only at locations where the plane hits the surface, which requires a reconstruction method that can be evaluated at specific locations.

Last but not least, the fourth criterion is that we do not want to propose an analysis method that is restricted to a specific application. Indeed, for example, it seems more promising to use one and the same analysis to compute slices on the object, to guide a reassembly process with other fragments, to use geometry-aware visualization techniques, or to define advanced feature selection tools. In this sense, the analysis should be parameter-less as far as possible along the pipeline. In the last resort, its output should be manipulated by tools, this time dedicated to a specific application.

We propose to illustrate the relevance of the aforementioned criteria through the reassembly of fractured objects. In particular, we deal with a real-world dataset composed of the fragments of the colossal statues that were surrounding the legendary Alexandria Lighthouse. Its specificity is that the fragments have been submerged under water level hundreds of years ago, and that they are now strongly eroded. For example, the features that could be used to guide the matching have often been faded and are thus hard to perceive. However, researchers in archeology are trained to detect the crucial features, thanks to their high-level knowledge and experience. Another difficulty is that most often, as in Alexandria, archeological fragments are too heavy to be manipulated by hand. A solution is to acquire them, and then let the experts align their virtual representations by means of efficient user interaction technique. However, even for archaeologists, it is quite difficult to obtain an accurate alignment because of interaction difficulties, position ambiguities, or surface interpenetrations. In the light of these observations, we propose to combine both the archaeologists' skills and the accuracy provided by the geometry processing during the reassembly process, by using a semi-automatic reassembly formalism. Hence, we understand the virtual reassembly as a permanent interplay between the user and the geometric analysis, via an efficient user interaction.

Thesis organization

The research presented in this thesis focuses on the elaboration of fundamental tools for the analysis of 3D objects. In Chapter 2, we present the theoretical background required to design such an analysis method.

In Chapter 3, we present concrete solutions that have been proposed before to characterize the geometry of 3D objects and to extract relevant information at multiple scales. We propose a classification of these methods, and we compare them regarding the aforementioned requirements. We then describe concrete guidelines for the elaboration of a suitable solution.

According to these guidelines, in Chapter 4, we provide a formal definition of our new analysis framework. We compare it to previous work, and we show how to use it in some application scenarios.

We focus more in-depth in Chapter 5 on the challenging virtual reassembly of broken objects with strongly deteriorated fragments. After analyzing some specific related work, we advocate for a semi-automatic solution that we formalize by a user-integrating semi-automatic interaction loop. We also present two new practical implementations of this formalism for pairwise matching, one based on tangible interaction, and the other one based on multi-touch interaction.

Chapter 2

Background

In this chapter, we present the theoretical and fundamental notions required for the analysis of the geometric properties of 3D objects. In Section 2.1, we start by defining the meaning of a 3D object in our context and the different ways to represent it. Then, in Section 2.2, we show how to analyze *continuous* 3D objects using differential geometry. Even though these concepts might seem familiar to some readers, this chapter also introduces the mathematical notations that will be used throughout this manuscript.

2.1 Representation of 3D objects

2.1.1 Definitions

One of the main objectives of this work is to analyze 3D objects that are considered as the digital representation of free-form physical objects. Before going further, it is necessary to ask the following questions:

1. How to properly represent the shape of a 3D object?
2. How to ensure that this digital representation refers to a correct physical object?

The three-dimensional representation of a *solid*[1] object can be defined in three ways: by considering *a*) the volume $\mathcal{V} \subset \mathbb{R}^3$ that is *inside* the object, *b*) the complement *outside volume* $\overline{\mathcal{V}}$ and *c*) the surface $\mathcal{S}$ that defines the boundary between the inside and outside. In practice, the solution *b)* is often not feasible: $\mathbb{R}^3$ is an infinite space, like $\overline{\mathcal{V}}$. Representing explicitly this infinite space with a discrete representation may not be possible. Any representation should rather be linked to the definition of $\mathcal{V}$ that can be used directly. In Computer Graphics, both the volume $\mathcal{V}$ and surface $\mathcal{S}$ representations are used. For the latter, it is necessary to ensure that $\mathcal{S}$ defines the boundary of a volume $\mathcal{V}$ by studying the properties of the surface $\mathcal{S}$.

A surface $\mathcal{S}$ is the boundary of a well-defined closed volume if the following three criteria are respected. First, the surface has to be *orientable*: it has to define an *inside* and *outside*, and thus enclose a volume (see examples of orientable and non-orientable surfaces in Figure 2.1). Second, it has to be *closed*: the presence of holes introduces ambiguities between *inside* and *outside*. Third, it has to be a *2D manifold*: each point on the surface must define a boundary between the *inside* and the *outside*. If the volume is degenerated into a point, a line, or a plane, its surface is locally not orientable and thus non-manifold. A surface with auto-intersections is also considered as non-manifold. Finally, if all these criteria are respected, the surface well represents a volume $\mathcal{V}$ and the associated 3D object: it can be called an *orientable continuous 2D manifold embedded in* $\mathbb{R}^3$ [BKP+10]. The

[1] The term *solid* means that we does not consider a potential empty space enclosed in the object that cannot be reached from the outside. For example, we ignore the empty space contained in a soccer ball, but we consider the volume contained in a hot air balloon as part of the outside volume.

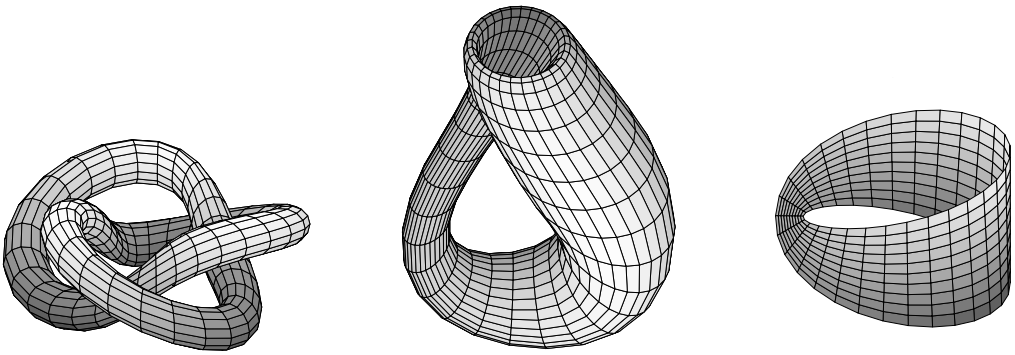


Figure 2.1: **Examples of parametric surfaces**. Left: the Knot surface is the only 2D-manifold of these examples. Middle and right: the Klein Bottle and the Mobius strip are *non-orientable*. The Klein bottle also contains self-intersections, whereas the Mobius strip is a bounded surface of genus 1 with only one edge.

term *continuous* refers to the definition of smooth surfaces with well defined derivatives. See, for example, the parametric 2D-manifold Knot surface in Figure 2.1. In practice, it is also necessary to deal with 2D manifolds with boundaries, for example surfaces with small holes, which can be easily filled to represent a volume. Common acquisition techniques often produce this kind of objects with holes that still refer to 3D objects.

In the next sections, for the sake of explanation and a better understanding, some examples are given for 2D objects that are defined by an area $\mathcal{A} \subset \mathbb{R}^2$ and a $1D$-manifold boundary curve. So in the following, we use the terms *3D surface* and *2D curve* to refer, respectively, to the surface of a 3D object and to the boundary of a 2D object. In both cases, we use the term *ambient space* (or *embedding space*) to refer to the space that contains the boundary (i.e. $\mathbb{R}^2$ or $\mathbb{R}^3$ in our case), whereas the term *embedded space* (or parameter space) refers to the manifold subspace (in our case included in $\mathbb{R}^{n-1}$). For an example, see Figure 2.2.

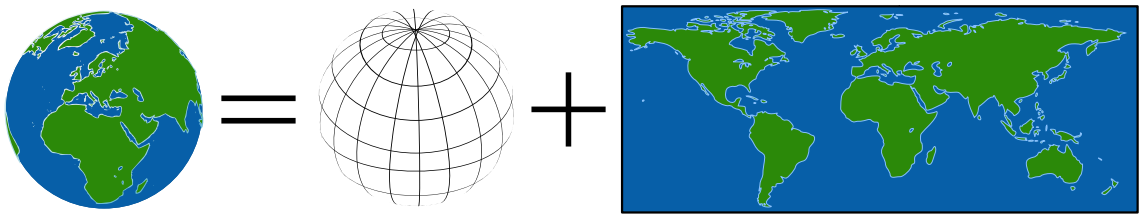


Figure 2.2: **Ambient and embedded space**. A 3D sphere is defined in an ambient space in $\mathbb{R}^3$, and defines an embedded space in $\mathbb{R}^2$. The relation between both spaces is usually represented as a map. In this example, we use a spherical mapping to associate to each position $[x, y, z]^T \in \mathbb{R}^3$ a parameter space coordinate $[u, v] \in \mathbb{R}^2$.

3D objects have to be analyzed in various applications, which all define specific constraints concerning acquisition techniques, memory requirements, processing, and scalability. In the next sections, we present two categories of representations and the associated properties: the *explicit* and the *implicit* representations. In a last section, we also introduce the discrete volumetric representation, that is used in some previous work.

2.1.2 Explicit representations

Explicit representations define the object boundary explicitly using either a parametric definition or by considering boundary samples, such as a set of points. In this case, the points do not define a closed surface as described above, but they are sufficient to characterize it in many applications (e.g. display or analysis).

Parametric surfaces A surface can be defined by a function that associates a position in a parameter space to a position in the ambient space. For instance, let us consider the following 2D parametric function

$$\mathbf{c} : [0, 2\pi] \rightarrow \mathbb{R}^2, u \rightarrow \begin{bmatrix} r\cos u \\ r\sin u \end{bmatrix}, \tag{2.1}$$

that defines a circle of radius r. Walking along the object's boundary is done by varying the parameter u. For the sampling of this circle, in order to obtain the required sampling in object space, an adapted discretization of the parameter space has to be found. The association of multiple parametric patches, such as NURBS, is commonly used in modeling software to generate free-form surfaces (e.g. manufactured objects).

Unorganized point set The minimalistic sample to characterize 3D surfaces is the triplet $\left[x, y, z\right]^T \in \mathbb{R}^3$ (respectively $\left[x, y\right]^T \in \mathbb{R}^2$ for 2D curves), called point or vertex, that defines a position on the boundary in the ambient space. Even though a point set does not strictly define a continuous closed surface, it is usually sufficient to represent it and analyze its properties. The positions can be characterized by some attributes, such as a color, or a normal vector $\mathbf{n}$ that defines the orientation of the local surface. Many analysis methods and processing algorithms require in addition a normal vector for each sample. However, the native output of most acquisition techniques, such as photogrammetry and laser scanning, are unorganized point sets without normal vectors. There are a variety of methods that have been proposed to robustly estimate them [AB98, PKG03, MN03, DG06, DLS05, Li210, BM12]; they define oriented point sets as $\{(\mathbf{q_i}, \mathbf{n_i}); \mathbf{q_i}, \mathbf{n_i} \in \mathbb{R}^3, \|\mathbf{n_i}\| = 1\}$. The unorganized point sets can be directly and efficiently rendered using splatting [LW85, ZPvBG01], and they are usually organized in spatial data structures that are adapted to specific queries. For instance, BSP trees [FKN80] are used for visibility tests and kd-trees [Ben75] for nearest neighbor queries.

Polygonal mesh Connectivity information can be associated to the vertex positions in order to create piecewise continuous surfaces. In this case, the object boundary is defined by a set of faces, composed of points and edges. These pieces of surface, which locally approximate the underlying surface, can be defined as triangles, quads, or more complex polygons. They can be used to map properties on the object, for example textures or vector fields, using local coordinates (usually called UV or texture coordinates). There are different data structures and algorithms that have been proposed to move on the surface defined by meshes [Dij59, Wei85]. Note that for unorganized point sets, due to the missing parametrization, moving on the surface is quite difficult.

Furthermore, using polygons as basic primitive for rendering (and especially triangles or quads) is very efficient because of the existing dedicated pipelines on GPUs. Meshes are

usually generated by modeling tools, or by estimating the connectivity of unorganized point sets.

A complete and up-to-date survey on polygonal mesh representations, their generation, and their processing can be found in [BKP+10]. We also refer to [KB04] for a discussion on point set and mesh representations for 3D object processing and rendering.

In this thesis, we focus on the analysis of acquired data. In order to avoid a preprocessing step to reconstruct a fully connected mesh (which may fail in some cases and take a lot of time), we strive to design methods that can directly operate on unorganized point sets as input.

2.1.3 Implicit representations

Implicit representations define a surface as the 0-isocontour

$$\mathscr{S} = \{\mathbf{p} \in \mathbb{R}^3; f(\mathbf{p}) = 0\} \tag{2.2}$$

of a given scalar field $f : \mathbb{R}^n \to \mathbb{R}$. The scalar field f is often chosen to represent the signed distance to the given surface. They are commonly used to simulate physical phenomena (e.g. water and smoke simulations [MST10, Che11], object deformations [BEB12]), or as input for volume rendering [HLSR08]. They can be expressed in different ways, either using analytical functions, or by using reconstruction methods.

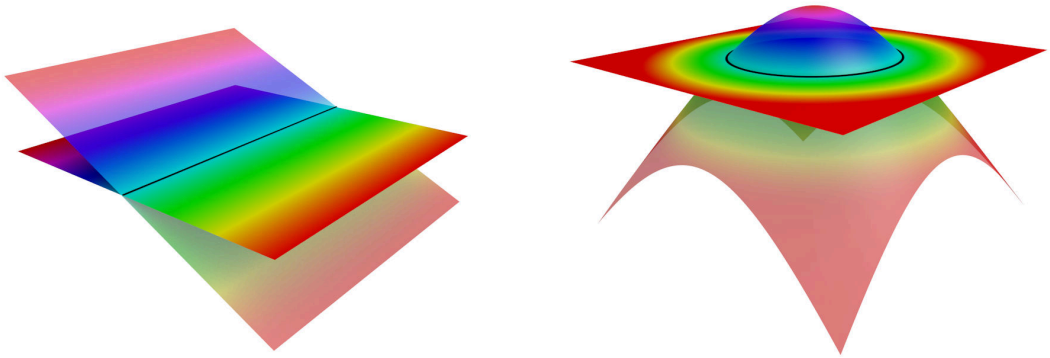


Figure 2.3: **2D scalar fields and implicit curves.** 2D implicit curves (in black) in $\mathbb{R}^2$ are defined as the 0-isocontour of an implicit scalar field, displayed in color on a plane as well as on a height field in $\mathbb{R}^3$ (the transparent shape). Left: A univariate scalar field produces a straight line in the 2D domain. Right: A bivariate quadratic scalar field defines a 2D circle.

Analytical implicit surfaces Analytical definitions are commonly used in 3D modeling [Bli82, BS91] for designing organic 3D objects, or for simulating fluid interaction. In this context, complex shapes are obtained by the composition of basic primitives. In our context, we are more interested by *algebraic* varieties that implicitly define curves or surfaces as the 0-set of a polynomial scalar field. For example, the explicitly defined 2D circle with radius r that we presented in Equation 2.1 can be defined by the following scalar field

$$f_c : \mathbb{R}^2 \to \mathbb{R}, (x, y) \to x^2 + y^2 - r^2. \tag{2.3}$$

The boundary $\mathscr{S}$ is then defined as $\mathscr{S} = \{\left[x, y\right]^T \in \mathbb{R}^2; f_c(x, y) = 0\}$. See Figure 2.3 for an example of a 2D line and a 2D circle.

Reconstruction methods The interesting properties of implicit surfaces motivated the development of methods that generate these scalar fields from explicit samples, usually from unstructured point set with or without normals. The idea is to find the best scalar field that interpolates or approximates the input samples while respecting some constraints on the 0-isosurface properties. For instance, this includes constraints on positions, gradient directions or magnitudes, or total curvature variation. Many techniques seek for a global solution of this problem. We can cite for instance techniques based on signed distance function estimation [HDD+92], radial basis functions [Fra82, CBC+01], level sets [ZOMK00], Poisson reconstruction [KBH06], or non-oriented variational formulation [CSAD04]. Using these approaches to design local descriptors requires a preprocessing step to compute the global solution, that is not compatible with our requirements: we want to characterize local geometric properties at random positions regardless of the object complexity. By using partition of unity [OBA+03], the global formulations can be turned into set of regional and independent problems: the object is split in several parts which are reconstructed independently and then merged to obtain the final object. However, this requires to evaluate and to blend with neighboring regions that again do not fulfill our random access requirement. A last class of methods are *Point Set Surfaces* [AK04] (PSS) that directly define implicit surfaces from point sets using *Moving Least Squares* [Lev98] (MLS) approximations.

Moving Least Squares MLS surfaces are defined by a purely local approach as follows: given an evaluation position $\mathbf{p}$, a weighted neighborhood of points $\mathscr{P}_t$ is collected with respect to a weighting function with support size t. Then, a primitive is fitted on this neighborhood through a least squares minimization of a robust norm, e.g. L_1. The evaluation point can then be projected onto the local surface approximation, where a new weighted neighborhood is collected from this new position, leading to a new fit and a new projected position. These steps are repeated until convergence [Lev03], and define a projection operator [AA04]. The reconstruction of smooth surfaces is achieved by *moving* the evaluation point, hence the name Moving Least Squares. The continuity of the resulting surface depends on the continuity of the weighting function [Lev98]. Furthermore, the approximation can be controlled by the support size t to produce smooth surfaces even in the presence of noise.

Concerning the primitive that is used to fit the neighborhoods, the first proposed methods express it relatively to a fitted reference plane, then fit a polynomial by linear minimization of the squared distance between the samples and the surface [Lev03, ABCO+03]. This procedure is not valid when fold-overs occur, because the neighborhood cannot be represented as a height function with respect to the reference plane. As a consequence, it is more robust to only consider the plane, fitted using normal information [AA04]. However, approximation problems still appear at large scales and with complex shapes. More recently, Guennebaud and Gross have proposed a fast algebraic hyper-sphere fitting procedure for oriented point sets [GG07], robust to sparse sampling, high curvature, and close sheet configurations. Independently of the fitted primitive, sharp features can also be supported using non-linear regression [OGG09].

We refer to [Blo97] for a more general introduction to implicit surfaces and to [Reu03, GP07] for a more in-depth presentation of the point set reconstruction.

2.1.4 Volumetric representations

The discretization of implicit functions is usually done by filling a regular grid that splits the ambient space into small sub-volumes, called *voxels* (for *vol*-ume *el*-ements). This concept reminds on 2D images that are defined by regular pixel grids. There are some surface analysis methods that are based on these representations (see Section 3.1) as extensions of 2D image algorithms. Unfortunately, voxel grids require a large amount of memory compared to other representations. Moreover, the strong impact of the choice of the grid properties (for example the cell size and orientation) make voxel grids complex to use in practice. Except specific applications that really need the volumetric information (such as medical imaging or simulation), common processing pipelines first convert a voxel grid to an explicit surface representation [NY06] prior to further processing. We refer to [BKP+10] for an overview of the conversion process from voxels to explicit surfaces.

2.2 Differential analysis

Independently of the chosen representation, 3D objects have geometric properties that are involved in various processing tasks such as analysis, editing, similarity detection, and matching. In this section, we introduce fundamental notions of differential geometry analysis. For the sake of clarity, we will first consider the case of 2D curves, and then extend them to 3D surfaces. Our objective is to define quantities on smooth surfaces (i.e. with well defined derivatives) that are invariant to the surface parametrization. This section is inspired by the Chapter 3 of [BKP+10]. We also refer to [BJ86, CG10] for more details on this topic.

2.2.1 Differential analysis for parametric manifolds

First-order differential analysis

The position of a given element u of the parametric space on a 2D curve can be expressed as a parametric function (generalization of Eq. 2.1) as

$$\mathbf{c} : \Omega \to \mathbb{R}^2, u \to \begin{bmatrix} x(u) \\ y(u) \end{bmatrix} \tag{2.4}$$

where $\Omega \subset \mathbb{R}$ is the parameter domain of $\mathbf{c}$.

The tangent vector is the first derivative of $\mathbf{c}$. In mechanics, it corresponds to the velocity vector (see Figure 2.4), and it is defined as $\mathbf{c}' : \Omega \to \mathbb{R}^2, u \to \left[x'(u), y'(u)\right]^T$.

This first-order invariant defines the distance between two points u_1 and u_2 in the embedded space. The length l of the path going from u_1 to u_2 can be computed as

$$l : \Omega \to \mathbb{R}, (u_1, u_2) \to \int_{u_1}^{u_2} \|\mathbf{c}'(u)\| \delta u. \tag{2.5}$$

l is an intrinsic property of the curve, called the curve length, that defines its intrinsic *metric*.

In the case of a 3D surface, the differentiation of $\mathbf{s}$ requires the definition of partial derivatives. Consider the parametric definition of a 2D manifold

$$\mathbf{s} : \Omega \rightarrow \mathbb{R}^3, (u,v) \rightarrow \begin{bmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{bmatrix}, \tag{2.6}$$

where $\mathbf{s}$ is continuously defined. The parametric space has 2 dimensions spanned by the tangent vectors $\mathbf{s}_u = \frac{\delta \mathbf{s}}{\delta u}$ and $\mathbf{s}_v = \frac{\delta \mathbf{s}}{\delta v}$. The manifold is assumed to have a *regular parametrization*, that means: $\mathbf{s}_u \times \mathbf{s}_v \neq 0$.

In order to compute the distance between elements on the surface in arbitrary directions, it is necessary to define the directional derivatives. Considering a 2D direction vector $\mathbf{w}$ in parametric space, it is possible to define the *Jacobian matrix* $\mathbf{J}$ to obtain the associated tangent vector $\mathbf{s}_\mathbf{w} = \mathbf{J}\mathbf{w}$ where

$$\mathbf{J} = \begin{bmatrix} \frac{\delta x}{\delta u} & \frac{\delta x}{\delta v} \\ \frac{\delta y}{\delta u} & \frac{\delta y}{\delta v} \\ \frac{\delta z}{\delta u} & \frac{\delta z}{\delta v} \end{bmatrix} = \begin{bmatrix} \mathbf{s}_u & \mathbf{s}_v \end{bmatrix}. \tag{2.7}$$

The matrix $\mathbf{J}$ encodes the impact of the parametrization on angular and distance measures, and consequently on area measures. This can be illustrated by the fact that the squared length of a vector is equal to the dot product with itself. The dot product is also related to the cosine of the angle between the two vectors. Now, let us consider the squared length of the tangent vector $\mathbf{s}_\mathbf{w}$:

$$\|\mathbf{s}_\mathbf{w}\|^2 = \mathbf{s}_\mathbf{w}^T \, \mathbf{s}_\mathbf{w} = (\mathbf{J}\mathbf{w})^T (\mathbf{J}\mathbf{w}) = \mathbf{w}^T \, \mathbf{G} \, \mathbf{w},$$

where $\mathbf{G}$ is the *metric tensor*, also named *first fundamental form*[1] of $\mathbf{s}$:

$$\mathbf{G} = \mathbf{J}^T\mathbf{J} = \begin{bmatrix} E & F \\ F & G \end{bmatrix} = \begin{bmatrix} \mathbf{s}_u^T\mathbf{s}_u & \mathbf{s}_u^T\mathbf{s}_v \\ \mathbf{s}_u^T\mathbf{s}_v & \mathbf{s}_v^T\mathbf{s}_v \end{bmatrix}. \tag{2.8}$$

The first fundamental form can thus be used to compute the length between the elements following a given path. Using eigen decomposition, it is also possible to compute the two orthogonal anisotropy directions (eigenvectors), and the associated anisotropy degrees (eigenvalues). In other words, the first fundamental form defines *intrinsic properties* of a surface, which are, as shown in Section 3.4.1, related to specific invariance properties.

Second-order differential analysis

Let us go back to a 2D curve $\mathbf{c}(u)$. The second derivative $\mathbf{c}''(u)$ measures the variations of the tangent vector, that is called, in mechanics, the acceleration. For example, consider a solid that moves in a 2D space, with an arbitrary constant velocity. When the solid moves on a straight line, there is no speed variation and its acceleration vector is null. Now suppose that a force is applied to the solid. We must ignore any component in the tangent direction (because we suppose that the tangent vector has a constant norm), so we should consider

[1] A common notation for the first fundamental form is $\mathbf{I}$. In this thesis, we prefer to use $\mathbf{G}$ to avoid confusion with the Identity Matrix.

this force as a vector orthogonal to the tangent direction. If it has a small norm, the solid should change its direction a bit: the greater the norm, the more the solid's trajectory is deviated. In geometry, we call this norm the curvature $\kappa : \Omega \rightarrow \mathbb{R}, u \rightarrow \|\mathbf{c}''(u)\|$. Figure 2.4 gives an example with a circular trajectory. As the acceleration vector is oriented in an orthogonal direction to the tangent vector, we have the relation:

$$\mathbf{c}''(u) = \kappa(u)\mathbf{n}(u) \tag{2.9}$$

where the unitary vector $\mathbf{n}$ is called the normal vector. If $\kappa(u)$ is positive or negative, the solid turns in one direction or in the other. By convention, the normal vector is directed to the outside of the object, so a negative value characterize the concavities, and positive values the convexities. In practice, this definition is valid even if the tangent vector has a variable norm along the curve.

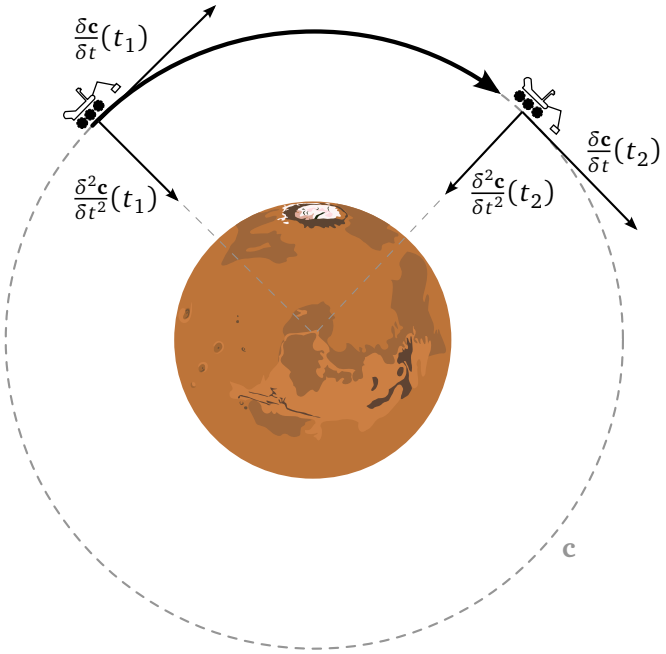


Figure 2.4: **Intuition on curvature: acceleration vector.** In mechanics, the variation of the velocity $\frac{\delta \mathbf{c}}{\delta t}$ of a given object is called the acceleration vector $\frac{\delta^2 \mathbf{c}}{\delta t^2}$. In geometry, the norm of this acceleration vector is called the curvature.

In case of 2D manifolds, the unit normal vector $\mathbf{n}$, orthogonal to the surface, is defined as $\mathbf{n} = \frac{\mathbf{s}_u \times \mathbf{s}_v}{\|\mathbf{s}_u \times \mathbf{s}_v\|}$. On a surface, we can measure the curvature at a location $[u, v]^T$ in any direction $\mathbf{w} = [u_w, v_w]^T$, by measuring the curvature of a curve going through the evaluation point in the direction $\mathbf{w}$ and lying on the surface. Since the normal vector is defined regardless this direction, we first have to measure if the variation of the tangent vector $\frac{\delta^2 \mathbf{s}}{\delta \mathbf{w}^2}$ in the direction $\mathbf{w}$ is aligned with this normal by considering the relation

$$-\frac{\delta^2 \mathbf{s}}{\delta \mathbf{w}^2}^T \mathbf{n} = \mathbf{w}^T \, \mathbf{II} \, \mathbf{w}, \tag{2.10}$$

where $\mathbf{II}$ is the *second fundamental form* defined as

$$\mathbf{II} = \begin{bmatrix} e & f \\ f & g \end{bmatrix} = \begin{bmatrix} \mathbf{s}_{uu}^T \mathbf{n} & \mathbf{s}_{uv}^T \mathbf{n} \\ \mathbf{s}_{uv}^T \mathbf{n} & \mathbf{s}_{vv}^T \mathbf{n} \end{bmatrix}, \tag{2.11}$$

with the partial derivatives

$$\mathbf{s}_{uu} = \frac{\delta^2 \mathbf{s}}{\delta u^2}, \quad \mathbf{s}_{uv} = \frac{\delta^2 \mathbf{s}}{\delta u \delta v}, \quad \mathbf{s}_{vv} = \frac{\delta^2 \mathbf{s}}{\delta v^2} \ . \tag{2.12}$$

We can see that the relation in Equation 2.10 is equal to 0 when the tangent variation vector is null or when it is orthogonal to $\mathbf{n}$, which means in both cases that the surface is a plane. When the tangent vector variations are oriented in the same direction as the normal vector (i.e., concavity), the computed value is negative, and on the opposite direction (i.e., convexity), it is positive. We thus have a similar behavior to the curvature of 2D curves, but not exactly: the measure is still related to the norm of the tangent vector, which may not be unitary. In order to compute the *normal curvature* κ_n in the direction $\mathbf{w}$, we have to normalize this relation by the surface metric (see Equation 2.8):

$$\kappa_n(\mathbf{w}) = \frac{\mathbf{w}^T \ \mathbf{II} \ \mathbf{w}}{\mathbf{w}^T \ \mathbf{G} \ \mathbf{w}} = \frac{e u_w^2 + 2 f u_w v_w + g v_w^2}{E u_w^2 + 2 F u_w v_w + G v_w^2}. \tag{2.13}$$

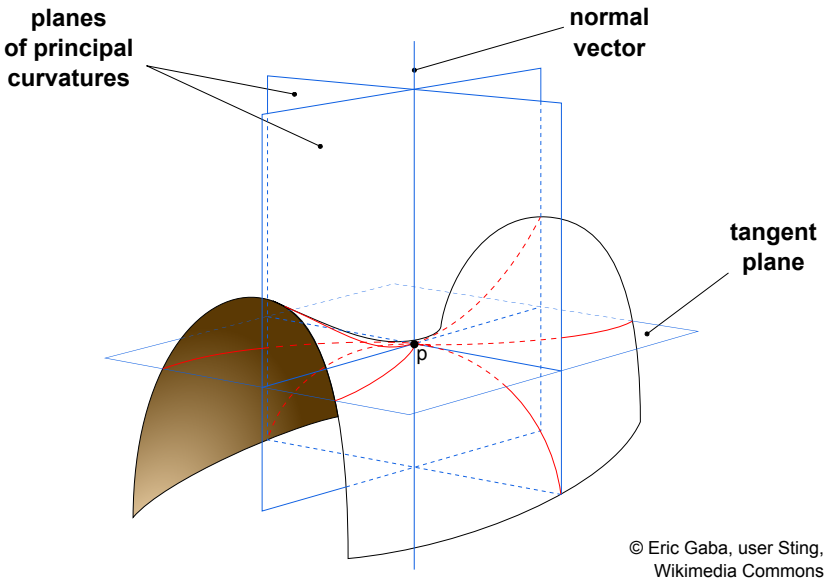


Figure 2.5: **Curvature Tensor.** Illustration of the curvature tensor for a saddle point $\mathbf{p}$: the normal curvature $k_\mathbf{n}(\mathbf{t})$ in direction t is expressed as a combination of the principal curvatures κ_1, κ_2, oriented in t_1, t_2 (defining a tangent plane). Image courtesy of Eric Gaba.

It can be seen in Equation 2.10 that κ_n has two extremal values κ_1 and κ_2, called principal curvatures that are defined in the orthogonal directions $\mathbf{w}_1$ and $\mathbf{w}_2$[1]. Hence, any value κ_n can be expressed as a linear combination of κ_1 and κ_2. A simple example is to consider the case of an isotropic point with $\kappa_1 = \kappa_2 = \kappa_n$, and thus $\|\mathbf{w}_1\| = \|\mathbf{w}_2\| = \|\mathbf{w}\|$ for any direction $\mathbf{w}$. In all cases, a common way to represent κ_n is to build the *curvature tensor* $\mathbf{C} = \mathbf{PDP}^{-1}$ with $\mathbf{P} = [\mathbf{w}_1, \mathbf{w}_2, \mathbf{n}]$ and $\mathbf{D} = diag[\kappa_1, \kappa_2, 0]$ (see Figure 2.5). It is also possible to compute the *mean curvature* H (represented by the green sphere in Figure 2.6) as

$$H = \frac{\kappa_1 + \kappa_2}{2}, \tag{2.14}$$

[1]Equation 2.10 is based on a dot product, that is itself based on a cosine function, which has extrema in orthogonal directions. See also Euler's theorem for a more formal justification [Eul67].

and the *Gaussian curvature* G as

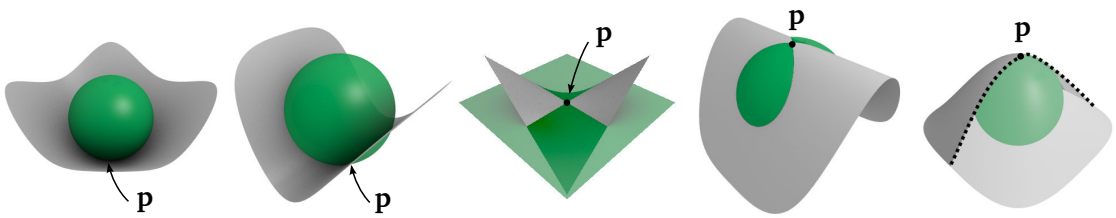
$$G = \kappa_1 \kappa_2. \tag{2.15}$$

Given a scalar function g defined on a manifold $\mathscr{S}$, one can define the Laplace-Beltrami operator $\Delta_{\mathscr{S}}$ as

$$\Delta_{\mathscr{S}} g = div \nabla g = g_{uu} + g_{vv}, \tag{2.16}$$

where the gradient of g is defined as $\nabla g = \left[g_u, g_v\right]^T$. This constitutes a common tool for the analysis of surfaces. For instance, when applied to the surface coordinates, this operator measures its variations, and it is related to the unsigned mean curvature: $\Delta_{\mathscr{S}} g = -2H\mathbf{n}$.

It can be shown that the Gaussian curvature can be computed directly from the first fundamental form [BJ86]. The Gaussian curvature defines an intrinsic property of the surface, like the Laplace-Beltrami operator. The other properties (κ_1, κ_2, H) that are linked to the second fundamental form are *extrinsic* in that they measure the way the surface is embedded in space. For example, consider a non-elastic paper sheet, with a straight line drawn between two points 10 centimeters apart. Folding the paper again and again does not change the length of the drawn line since it is related to the intrinsic properties of the sheet which are not altered by non-elastic transformations, as for example folding. On the other hand, the Euclidean distance between the two points is changed by folding, as well as the curvature or other extrinsic properties.



$\kappa_1 = \alpha$	$\kappa_1 = \alpha$	$\kappa_1 = \alpha$	$\kappa_1 = 0$	$\kappa_1 = -\alpha$
$\kappa_2 = \alpha$	$\kappa_2 = 0$	$\kappa_2 = -\alpha$	$\kappa_2 = -\alpha$	$\kappa_2 = -\alpha$
$G = \alpha^2$	$G = 0$	$G = -\alpha^2$	$G = 0$	$G = \alpha^2$
$H = \alpha$	$H = \frac{1}{2}\alpha$	$H = 0$	$H = -\frac{1}{2}\alpha$	$H = -\alpha$

Figure 2.6: **Mean and Gaussian Curvature** for the point **p**. H is represented as a green sphere and characterizes the mean of the tangent vector variation without considering the anisotropy direction. G represents the Gaussian curvature.

2.2.2 Differential properties of implicit surfaces

The implicit formulation does not offer the parametrization needed to express differential properties such as tangent vectors and curvature. In this subsection, we present the closed-form formula to compute the normal vector **n**, the principal, mean and Gaussian curvatures, and the Laplace-Beltrami (LB) operator for a surface expressed a the 0-isolevel of a given scalar-field $f(\mathbf{p})$, where $\mathbf{p} \in \mathbb{R}^3$. A more in-depth presentation of these formula can be found in [Gol05] for curvatures, and for the LB operator, as well as its relations with the standard parametric formulation in [BCOS01]. We denote by f_x the first partial derivatives of f.

Normal vector By construction, the scalar field has a gradient vector orthogonal to the surface, defined as $\nabla f = \left[f_x, f_y, f_z\right]^T$. The normal vector is thus defined as

$$\mathbf{n}(f) = \frac{\nabla f}{\|\nabla f\|}. \tag{2.17}$$

Curvature In order to compute the mean and Gaussian curvatures, we need to define the *Hessian matrix* $\mathbf{H}$ formed by the second-order partial derivatives of f

$$\mathbf{H} = \begin{bmatrix} f_{xx} & f_{xy} & f_{xz} \\ f_{yx} & f_{yy} & f_{yz} \\ f_{zx} & f_{zy} & f_{zz} \end{bmatrix}, \tag{2.18}$$

and its adjugate $\mathbf{H}^*$ as

$$\mathbf{H}^* = \begin{bmatrix} f_{yy}f_{zz} - f_{yz}f_{zy} & f_{yz}f_{zx} - f_{yx}f_{zz} & f_{yx}f_{zy} - f_{yy}f_{zx} \\ f_{xz}f_{zy} - f_{xy}f_{zz} & f_{xx}f_{zz} - f_{xz}f_{zx} & f_{xy}f_{zx} - f_{xx}f_{zy} \\ f_{xy}f_{yz} - f_{xz}f_{yy} & f_{yx}f_{xz} - f_{xx}f_{yz} & f_{xx}f_{yy} - f_{xy}f_{yx} \end{bmatrix}. \tag{2.19}$$

We can now define the Gaussian curvature

$$G = \frac{\nabla f \ \mathbf{H}^* \ \nabla f^T}{\|\nabla f\|^4}, \tag{2.20}$$

and the mean curvature

$$H = -div\ \mathbf{n}(f) = \frac{\nabla f \ \mathbf{H}^* \ \nabla f^T - \|\nabla f\|^2 \ Tr(\mathbf{H})}{2\|\nabla f\|^3}, \tag{2.21}$$

where $Tr(\mathbf{H}) = \sum_{i=1}^{3} h_{ii}$ is the trace of the Hessian matrix.

The principal curvatures κ_1 and κ_2 can be deduced from G and H as

$$\begin{aligned} \kappa_1 &= H + \sqrt{H^2 - G}\,, \\ \kappa_2 &= H - \sqrt{H^2 - G}\,. \end{aligned} \tag{2.22}$$

Laplace-Beltrami With the implicit formulation, the LB operator can be defined for any scalar function $h(x, y, z)$. Its gradient on the 0-isosurface ∇h_f can be computed as the projection of ∇h on the surface as

$$\nabla h_f = (\mathbf{I} - \mathbf{n}\mathbf{n}^T)\nabla h \tag{2.23}$$

The Laplace-Beltrami operator (already defined in Eq. 2.16 for the parametric representation) is then defined as

$$\Delta_f h = \frac{div\left(\nabla h_f \ \|\nabla f\|\right)}{\|\nabla f\|}. \tag{2.24}$$

When f is an Euclidean scalar field (i.e. the value of the scalar field represents the Euclidean distance to the 0-isosurface), the gradient is unitary by construction for any coordinate. In this case, we denote the associated function $\bar{f}$, yielding to the simplified implicit LB operator

$$\Delta_{\bar{f}} h = div\ \nabla h_{\bar{f}}. \tag{2.25}$$

Chapter 3

Multi-scale shape analysis

In the previous chapter, we have introduced common representations of 3D objects, as well as the differential analysis of their continuously defined surfaces $\mathscr{S}$. This chapter focuses on the practical methods that have been proposed to analyze discrete 3D surfaces $\mathbf{s}$. As explained in the introduction, this analysis is a preliminary step for various processing tasks in Computer Graphics. The chapter is divided into 4 sections, starting from general concepts and finishing by the description of specific methods and their comparison.

More precisely, in Section 3.1, we present an analysis pipeline shared by the majority of previous work. In Section 3.2, we present previous methods that define *geometric descriptors* as approximations or alternatives to differential quantities on non-smooth surfaces. In Section 3.3, we present methods for detecting relevant *features* in the geometric descriptions. We propose a classification of these approaches in order to compare them with each other. A final discussion in Section 3.4 summarizes advantages and limitations of previous work, thus motivating the technical choices for the design of our analysis framework.

3.1 Analysis pipeline

We call a *geometric descriptor* a value (or a set of values) that characterizes the object geometry for a given position $\mathbf{p}$ on the surface. The mean curvature is probably the simplest example. *Geometric descriptors* are often designed as discrete estimations of differential properties, discrete integration of surface properties, statistical neighborhood analysis, diffusion procedures, or regressions.

These descriptors can also be computed at different *scales* t. The motivation is to be able to measure properties that characterize geometries, starting from small patterns or surface macro-structures, to the overall object shape. Figure 3.1 shows 4 different ways to compute *geometric descriptors* at multiple scales. First, differential properties (or some approximations) can be computed on smoothed versions of the surface (Figure 3.1(a)): the more the object is smoothed, the coarser is the considered scale. Most of the *geometric descriptors* need to collect neighboring points to be computed. In this case, a distance threshold used to collect neighbors defines the scale, and can be expressed as geodesic (Figure 3.1(b)) of Euclidean (Figure 3.1(c)) distances. When dealing with polygonal meshes with a uniform sampling, such geodesic neighborhoods can be approximated by collecting the points in a n-ring, where n represents the number of edges between the evaluation point and the current ring. Finally, some approaches consider the volume defined by the intersection between the object interior volume and a ball $B_t(\mathbf{p})$ centered at the evaluation point with radius t (Figure 3.1(d)). We present in Section 3.2 the previous work that is the most relevant in our application case to compute such descriptors. We emphasis that we focus on local descriptors only, in opposite to global descriptors [ZH99, MDTK06] that build a unique description for the entire object and do not consider local properties.

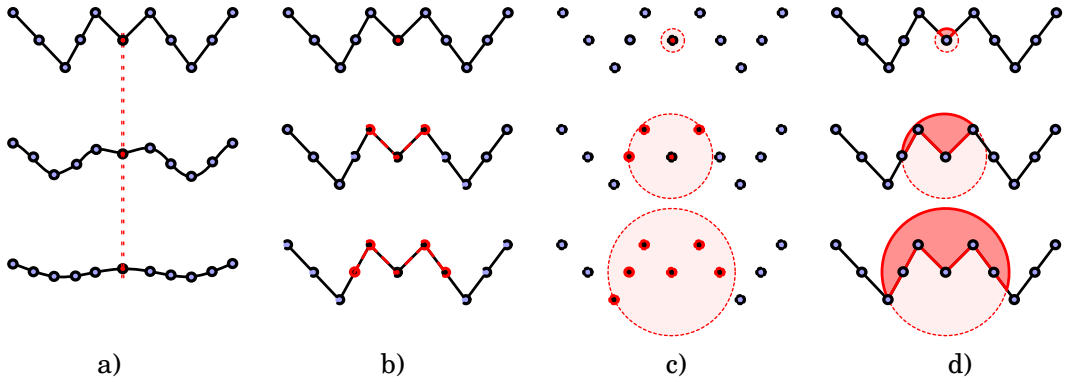


Figure 3.1: **Multi-scale neighborhood collection approaches.** (a) iterative smoothing, (b) n-ring, (c) Euclidean point, and (d) Euclidean volume/surface. Top to bottom: fine to coarse scales.

Once a dense geometric description has been computed on the surface (i.e., for all points $\mathbf{q}_i$ of the object), it is often desired to extract the relevant structures that characterize the object. This is addressed by what we call *feature detectors*, which usually analyze the variation or the distribution of the *geometric descriptors* along the object, possibly at multiple scales. We present these methods in Section 3.3.

Even though a lot of different analysis approaches have been proposed in the past years, they usually all rely on the same overall pipeline. First, they compute a *geometric descriptor* for each sample of the object, which is used in a second step to detect the relevant features. These features are then used for various applications, such as modeling, matching, or remeshing. Obviously, some pipeline variants have been proposed for specific contexts. For example, some approaches compute a first *geometric descriptor*, use it to extract features, then compute another descriptor to characterize them. In the following section, we present these descriptors and detectors as independently as possible from the application, in order to be able to compare them.

We also analyze the invariance properties of the proposed analysis pipelines, with respect to: *a*) the metric used to collect neighborhood information, *b*) the way to describe geometry, and *c*) the feature extraction method. This is fundamental for the design of the analysis pipeline, since the invariance of each step impacts the entire pipeline due to the sensitivity to transformations. For example, using intrinsic metrics associated to extrinsic descriptions is sensitive to isometric transformations.

Also note that in our application context, we want to be able to use our analysis for shape matching. As shown in Figure 3.2, this implies that the analysis approach has to handle bounded surfaces.

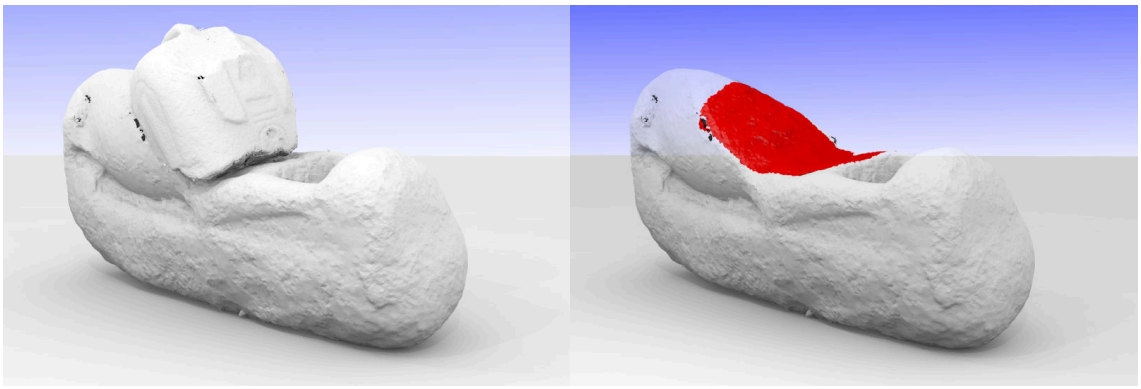


Figure 3.2: **Shape matching involves bounded surfaces.** In this example, the two fragments (left) have been matched using their contact surface, shown in red in the right image (for the bottom fragment). The geometry description used for the matching is performed only on the bounded surface, and it does not take into account unrelated features.

3.2 Geometric descriptors

3D object analysis is a topic that is widely studied in Computer Graphics, Computer Vision, and Computational Geometry. In the last decades, a lot of methods have been proposed to characterize 3D objects with different objectives and limitations. In this section, we present the descriptors that we consider most relevant to our application and research context. When it is possible, we refer the interested reader to more exhaustive surveys.

In the following subsections, we propose a classification of the *geometric descriptors* into 5 categories based on different mathematical tools: estimation of differential operators, neighborhood integration, statistical analysis, diffusion methods, and regression. In order to produce a flexible analysis, the descriptors have to be evaluable at arbitrary scales and positions. Hence, we focus on *multi-scale* methods, and not on *multi-resolution* methods [KBS00] such as wavelets [SLR+12]. Indeed, the latter are rather tailored for the data summarization at multiple resolutions for compression or editing purposes. Moreover, they characterize the global properties of the shape and not the local geometric details [OT03], and they might not be able to detect the relevance of the analyzed data. For this reason, multi-scale methods are preferred because they analyze each scale independently in order to produce stable descriptors of the relevant data, even though this involves some redundant computations.

We are particularly interested in descriptors that characterize the surface of an object, and not its global volumetric properties [LZSCO09, RBBK10, BS12, LKF12, GL12]. Indeed, partial shape matching involves surfaces with boundaries, as shown in Figure 3.2.

3.2.1 Differential operator approximations

The first category of *geometric descriptors* is based on the estimation of differential values on discrete surfaces. Differential approximations require the surface to be C^1-continuous (to compute the metric), or C^2-continuous (to compute the curvature). Since surfaces defined by discrete samples are not continuous, most of the methods presented below are based on

the discrete Laplace-Beltrami operator [Tau95, MDSB02, BS07] for triangle meshes (noted LB for convenience in the following), defined for any function g on the surface. Note that when g is defined as the point coordinates $\mathbf{q}_i$, Δg is related to the mean curvature. We refer to [HPW06, WMKG07] for an in-depth analysis of the LB approximation properties.

Differential approximations are sensitive to position noise, and they cannot distinguish between noise and features due to their purely local nature. To this end, they have to be evaluable at multiple scales, which is achieved by smoothing the signal using a given operator and then applying the LB operator. In the case of the Gaussian smoothing operator G_σ, where σ is the Gaussian standard deviation, this pipeline can be simplified to the convolution of *Laplacian of Gaussian* kernel (LoG)

$$\Delta\left(G_\sigma \otimes g\right) = \left(\Delta G_\sigma\right) \otimes g = LoG \otimes g. \tag{3.1}$$

This approach has been recently employed for voxel grids [WNK06, FBMB10, GW11] and meshes [ZHDQ08].

Another common solution is to use the *Difference of Gaussians* (DoG) to approximate the LB operator (see Figure 3.3). This has first been proposed for images [MH80], then for curves [MM86], and more recently for meshes [MKY01, ZBVH09, MFK+10] and voxel grids [SAS07, FBMB10]. It works in two steps: first, smoothing the data at two scales, and second, approximating the curvature at a point by the distance between its two smoothed locations. More formally, the DoG operator is applied to the surface $\mathbf{s}$ as

$$DoG_{\mathbf{s}} = \left(G_{\sigma_1} \otimes \mathbf{s}\right) - \left(G_{\sigma_2} \otimes \mathbf{s}\right). \tag{3.2}$$

Both the LoG and the DoG method do not require any parametrization as long as the smoothing operator is well defined.

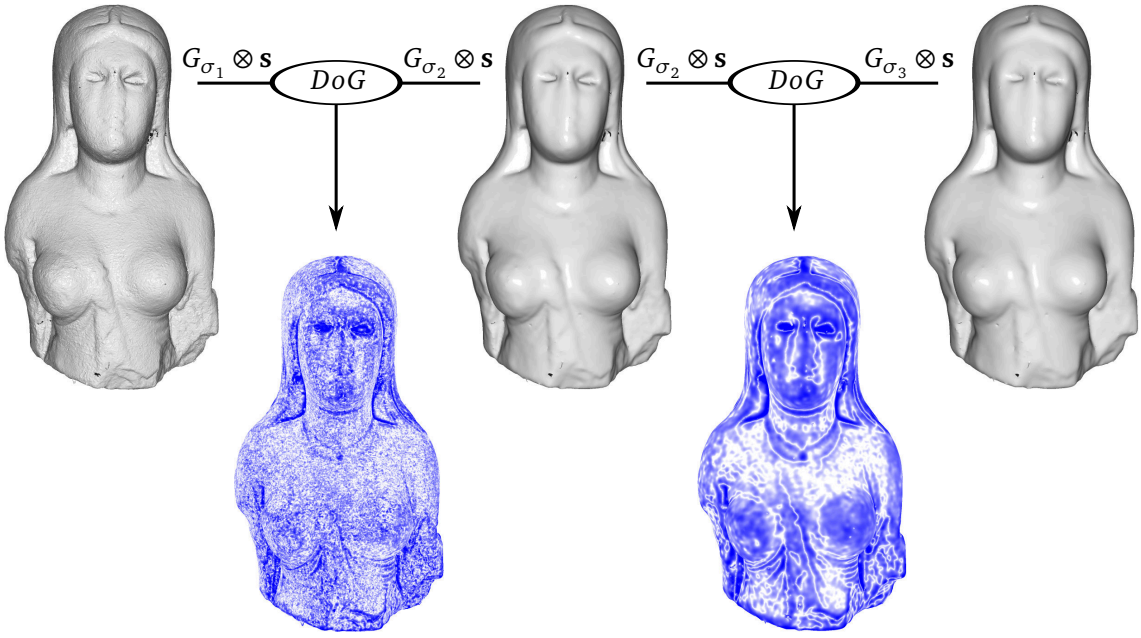


Figure 3.3: **Estimation of mean curvature using DoG.** Top: 3 levels of smoothing. Bottom: estimation of the mean curvature H using DoG in order to measure planar (white) and curved (blue) areas at different scales.

It is also possible to estimate curvature-like information more directly on polygonal meshes without the LB operator. For example, the mean curvature can be estimated form the average normal [IJ85, LG05], using simplex angles [HID95], or as the ratio between the geodesic and the Euclidean distance [GSCO07]. For a more detailed comparison of curvature estimators on triangle meshes, we refer the interested reader to the surveys [Pet02, GG06]. We will show in Section 3.2.5 that curvature-like values can also be estimated directly on point sets [CP03, CCSLT09, ZLCZ09] using regression techniques.

Irrespectively to the differential value estimation, the approaches presented above directly depend on the smoothing operator that might fail for some configurations. In order to avoid them, smoothing is replaced in the following with neighborhood collection approaches presented in Section 3.1.

3.2.2 Neighborhood integration

In practice, the approaches presented above are quite sensitive to noise: they are based on discrete samples to estimate the values that are in fact related to a continuous surface. Some methods regularize the estimated values by integrating the information contained in the area or volume neighborhood, defined as the intersection of the surface $\mathbf{s}$ and an Euclidean ball $B_t(\mathbf{p})$ of radius t centered around $\mathbf{p}$ (see Figure 3.1 (d)).

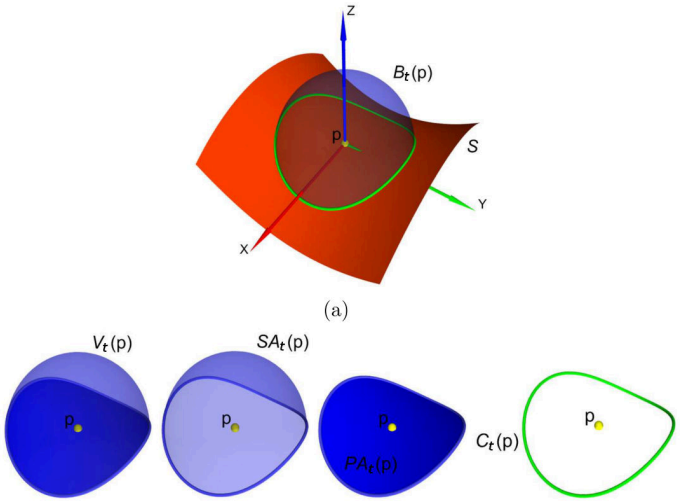


Figure 3.4: **Integral Invariants.** Neighborhood types generated from the intersection of the Euclidean ball and the surface $\mathbf{s}$. From left to right: ball neighborhood, spherical patch, surface patch, and spherical intersection. Image taken from [PHYK05].

A first type of approach is based on the *Integral Invariants* [PHYK05, GMGP05, YLHP06, MCH+06, PWHY09] that can be defined both for 2D curves and 3D surfaces. The key idea is to integrate the properties of various neighborhood types shown in Figure 3.4, such as the volume $V_t(\mathbf{p})$ defined by the intersection of the ball and the surface, the associated enclosing surface $SA_t(\mathbf{p})$ (for *Spherical Area*), the intersecting surface $PA_t(\mathbf{p})$ (for *Patch Area*), and the associated bounding 3D curve $C_t(\mathbf{p})$. For each of these neighborhoods, the authors present ways to approximate differential quantities, such as for example the mean curvature. Walter et al. propose a descriptor called SUSAN3D [WAL08], which can be seen as an extension of a 2D descriptor related to the volume invariant $V_t(\mathbf{p})$.

A second type of approach uses *geometrical moments* [CGR+04, XL06, XL08] that characterize statistical properties of the geometry. A moment $M_{\alpha\beta\gamma}$ is computed as

$$M_{\alpha\beta\gamma} = \int\int\int x^{\alpha} y^{\beta} z^{\gamma} \rho(x,y,z)\, \delta x\, \delta y\, \delta z, \tag{3.3}$$

where ρ is a density function that encodes the intersection between the Euclidean ball and the surface. This formulation provides a robust characterization at multiple orders of the surface.

In practice, all these integration methods can be very complex to use due to the high computational cost, especially at coarse scales where a large neighborhood has to be taken into account. Indeed, good approximations of the integrals require fine discretizations of volumes and surfaces as well as good estimations of intersections that might be complex to obtain for free-form surfaces.

3.2.3 Neighborhood distribution

Independently of the practical difficulties due to the estimation of differential quantities, a main limitation of these methods is that they cannot precisely characterize complex shapes. For example, a junction of 3 ridges, called a T-junction, may involve three different directions that cannot be described by the surface metric or the curvature tensor. Furthermore, some properties, like the surface roughness, have no particular differential meaning. In this subsection, we present techniques that represent the neighborhood as statistical distributions, by considering a surface patch $PA_t(\mathbf{p})$, a spherical intersection $C_t(\mathbf{p})$, or other custom neighborhoods.

Statistical analysis of $C_t(\mathbf{p})$

In the following, the neighborhood is parametrized along the arc-length ℓ of the boundary curve $C_t(\mathbf{p})$ from a reference position and the current neighbor. We recall that this neighborhood collection is usually approximated by n-rings when dealing with meshes.

A first type of information that can be expressed as distribution is the position of the neighbor points $\mathbf{q_i}$ surrounding $\mathbf{p}$. Chua and Jarvis proposed the *point signature* [CJ97], which considers a plane defined by $\mathbf{p}$ and the associated normal vector $\mathbf{n}$. Then, they measure height values between the surface $\mathbf{s}$ and the plane along $C_t(\mathbf{p})$, in order to build a 2D curve $h(\ell)$. Even though the *point signature* seems to be adapted to catch T-junctions or other complex anisotropic features, this representation is very sensitive to the estimation of the reference plane, and to position noise.

In order to reduce the sensitivity to position noise, some approaches focus on the analysis of normal vectors that can be robustly estimated even in the presence of noise [AB98, PKG03, MN03, DG06, DLS05, Li210, BM12]. Stein and Medioni [SM92] proposed to analyze the *local normal* defined as the normal vector of the current neighbor expressed relatively to $\mathbf{n}$ in a given tangent direction. Then, they build their descriptor in a similar way as the *point signature*, but by considering the local normal coordinates.

Statistical analysis of $PA_t(\mathbf{p})$

The main limitation of considering the spherical intersection curve $C_t(\mathbf{p})$ is that it contains only features that are across the neighborhood boundary, but completely ignores the ones that are enclosed in the Euclidean ball $B_t(\mathbf{p})$. This is addressed by considering the spherical patch $PA_t(\mathbf{p})$, that can be analyzed in ambient space, yielding an extrinsic characterization, or directly on the surface $\mathbf{s}$.

An extrinsic analysis of $PA_t(\mathbf{p})$ can be achieved using the *Shape Context* [BMP02, KNK03, FHK+04, WNK06], which is essentially a 3D histogram of the neighbor positions $\mathbf{q_i}$ (see Figure 3.5). A main limitation of this approach is the difficulty to find efficient and generic parameters to construct the histogram. A variant is to represent the neighborhood by spherical harmonics summarized using Principal Component Analysis (PCA) [FS06].

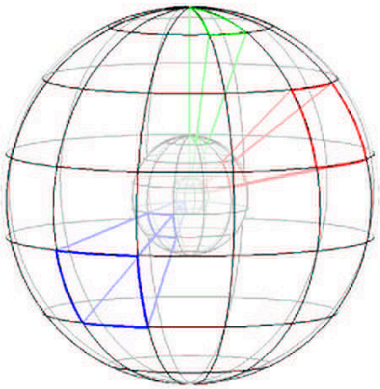


Figure 3.5: **Shape Context.** The volume neighborhood is segmented into a 3D histogram. Image taken from [FHK+04].

In order to measure properties directly on the surface, it is possible to build a *curvature map* [GGGZ05], which is an image where polar coordinates r, θ of the neighbors are mapped to image coordinates u, v. For each pixel, Yamani et al. measure a curvature value [YF99, YF02], while Zelinka and Garland use an association of position, normals, and tangent vectors [ZG04]. In [GGGZ05], Gatzke et al. propose a comparison of these approaches. The major problem of these methods is that they usually do not offer a summarization of the neighborhood, and thus they can be very memory-consuming when dealing with huge datasets at multiple scales. In [LG05], Li and Guskov summarize images using a re-parametrization step based on Discrete Cosine and Fourier Transforms.

In practice, the distribution analysis parameters can be very difficult to adjust: the relation between each parameter and their impact on the efficiency of the description is seldom intuitive. An alternative is to consider the approach proposed in [LVJ05, LLKR07, CCFM08], which uses DoG to compute per-vertex curvature, and then defines *mesh saliency* at a given scale using a linear combination of the neighborhood curvature values. This approach detects specific features with respect to perceptual notions, and thus it may not be adapted to all application cases.

Statistical analysis of custom neighborhoods

A last statistical descriptor is the *spin image* proposed in [JH99, DK12]. The idea is to build a view-dependent image of the object that associates to each pixel a measure of density, which is related to the number of points projected to each pixel. In this approach, the size of the neighborhood is defined by considering points that have a normal forming an angle with $\mathbf{n}$ inferior to a given threshold. This method works with point sets having a regular sampling: the variation of the visible density in the *spin image* is thus due to the variations of the local surface orientation after projection, which is related to the curvature. These approaches can be complex to use in some cases, in particular when there is no assumption on the viewing parameters. Furthermore, the neighborhood is defined by an angle threshold that could be hard to manipulate in case of a multi-scale analysis.

3.2.4 Diffusion maps

Recently, a class of approaches inspired by *diffusion maps* [CL06] emerged. These techniques were initially conceived for shape retrieval, or, more generally, for similarity detection under isometric deformations. The key idea is to characterize a 3D objects by considering its intrinsic properties. The surface metric can be seen as a diffusion flow on the surface:

$$\Delta g(\mathbf{p},t) = -\frac{\delta g(\mathbf{p},t)}{\delta t}, \tag{3.4}$$

also known as *heat diffusion equation*. Here, g is a scalar function defined on the surface that varies with respect to the time t. Intuitively, since the surface variation is related to the Laplace-Beltrami (LB) operator, the speed of diffusion directly depends on the surface metric. Hence, it is possible to define a function $k_t(\mathbf{p_1},\mathbf{p_2})$ that measures the amount of heat transferred from $\mathbf{p_1}$ to $\mathbf{p_2}$ during a time period $[t_0 : t]$, given an initial heat source at $\mathbf{p_1}$ that stops to emit heat at the initial time t_0. We call this function the *diffusion distance*.

The first approach that exploits this *diffusion distance* to design a *geometric descriptor* is the *average diffusion distance* [dGGV08] that measures the average of $k_t(\mathbf{p},\mathbf{q_i})$ over a region $\mathscr{R}$, where $\mathbf{q_i},\mathbf{p} \in \mathscr{R}$. The main problem of this approach is the link between the evaluation point descriptor and its region: when the segmentation changes, the descriptor becomes invalid. This limitation is by-passed by the *Heat Kernel Signature* (HKS) [SOG09]. The idea is to consider the multi-scale profile generated as $k_{\mathbf{p}}(t) = k_t(\mathbf{p},\mathbf{p})$, which encodes the remaining energy at a time t after a unitary heat impulsion at t_0 (see Figure 3.6 for an example on the dragon model). However, the initial HKS method suffers from normalization issues: for each time t, the values of $k_{\mathbf{p}}(t)$ have to be normalized by the maximum value on the whole object. This limitation is overcome by the *Scale-Invariant Heat Kernel Signature* (SI-HKS) [BK10, LBB11], which removes the dependency between $k_{\mathbf{p}}(t)$ and the scale factor s by using an s-dependent time shift. These methods motivated various studies on *diffusion geometry* [BB11, BBGO11, FW11, WBBP12]. Note also the recent work of Liang et al. that approximates the LB operator on point-sets using MLS [LLWZ12], and the extension of Zobel et al. to compute HKS on first-order tensor fields [ZRH11].

All these methods provide an elegant way to compute intrinsic surface descriptors, and they redefine the notion of scale by a time interval. Although this concept is well adapted to the initial problem (shape retrieval), it can be complex to adapt to shape matching: using a time parameter is not intuitive, and more importantly, the spatial region surrounding

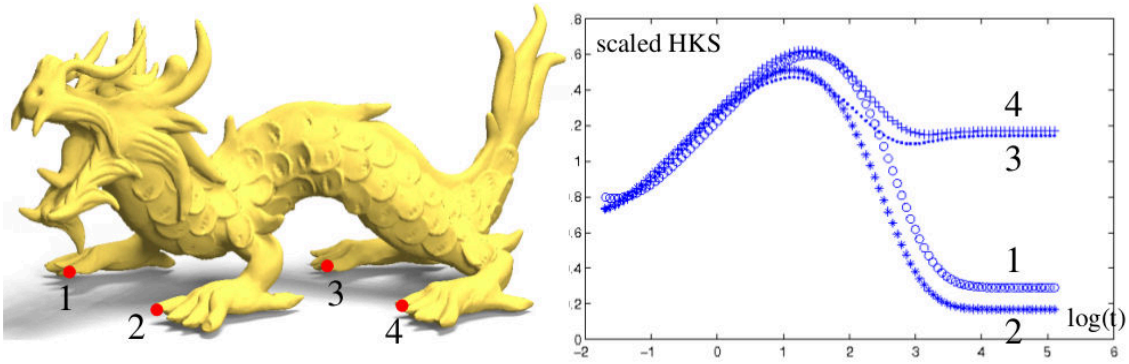


Figure 3.6: **Heat Kernal Signature.** Left: dragon model. Right: scaled HKS at points 1, 2, 3, and 4. All four signatures are close at low values for t, while high values for t separate the points on the front claws from those on back. Image taken from [SOG09].

p may have different sizes depending on local surface curvature properties. It is rather difficult to set this parameter in practice. Furthermore, solving the Heat Diffusion equation can be time consuming with huge 3D objects because it requires to solve an eigen-decomposition of the LB matrix, an $n \times n$ matrix containing weights for discretizing the LB operator [SOG09]. Usually, only the largest eigenvalues and eigenvectors can be computed in a reasonable amount of time. Since the eigen-decomposition characterizes the geometry from coarse (high eigenvalues) to fine scales (low eigenvalues), these methods are adapted to characterize coarse features and not small geometric details.

3.2.5 Regression

The previously discussed approaches are based on a direct analysis of explicit samples, such as point sets [BMP02], meshes [WMKG07], and Voronoï diagrams [LLWZ12]. In this subsection, we focus on regression-based approaches that use an intermediate continuous surface patch that approximates the local surface to characterize its properties. The methods presented in this subsection that use least square fitting approaches, are efficient and robust to the input representations. We can distinguish two kinds of approaches, depending on the order of the fitted primitive: first-order or second-order.

First-order primitive

Using plane fitting to characterize the geometry has been proposed in [PKG03]. The key idea is to analyze the eigen-decomposition of a covariance matrix representing the neighborhood, hence the name *Covariance Analysis* (CA). The $3x3$ covariance matrix $\mathbf{M}_{COV}$ that represents a weighted neighborhood surrounding **p** can be built as

$$\mathbf{M}_{COV} = \frac{1}{\sum_0^k w_i} \begin{bmatrix} \mathbf{q}_0 - \overline{\mathbf{p}} \\ \cdots \\ \mathbf{q}_k - \overline{\mathbf{p}} \end{bmatrix}^T \begin{bmatrix} w_0 & & 0 \\ & \ddots & \\ 0 & & w_k \end{bmatrix} \begin{bmatrix} \mathbf{q}_0 - \overline{\mathbf{p}} \\ \cdots \\ \mathbf{q}_k - \overline{\mathbf{p}} \end{bmatrix}, \tag{3.5}$$

where $\|\mathbf{q}_i - \mathbf{p}\| \leq t, w_i \in [0,1]$. Pauly et al. propose in [PKG03] to measure the *surface variation* using an eigen-decomposition of $\mathbf{M}_{COV}$. The idea is to measure if the input points

vary only in the two directions that define the fitted plane, using the ratio

$$\sigma(\mathbf{p}) = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2}, \lambda_0 \leq \lambda_1 \leq \lambda_2 \; ; \tag{3.6}$$

where λ_i are eigenvalues of $\mathbf{M}_{COV}$. The surface is assumed to be without noise, so this value is related to the mean curvature. Similar approaches compute anisotropy using $\mathbf{M}_{COV}$ [MBO10] or normal variation [LG05]. [MOG11] also use a covariance analysis, but with an alternative definition of the neighborhood using Voronoi cells. However, the latter approaches measure first-order quantities (normals) and estimate second-order quantities by comparisons. The main problem is that they cannot distinguish between variation due to noise and due to curvature. Hence, they require a specific tuning to handle non-smooth surfaces [MOG11].

Second-order primitive

As explained in Section 2.1.3, a second-order continuous surface patch can be estimated by first finding a planar parametrization of the neighborhood, then by expressing it as height-field, and finally by using a linear regression [FJ89, BSF02, CP03, GCO06, CPG09] to fit a bivariate quadric as

$$q(u,v) = au^2 + bv^2 + cuv + du + hv, \tag{3.7}$$

where u, v are vertex coordinates expressed in the support plane (see Figure 3.7). The curvature tensor $\mathbf{C}$ is obtained by differentiating this function q (see Section 2.2.1). Mian et al. [MBO10] also proposed a descriptor based on the averaging of H, k_1 and k_2 over the parametric patch.

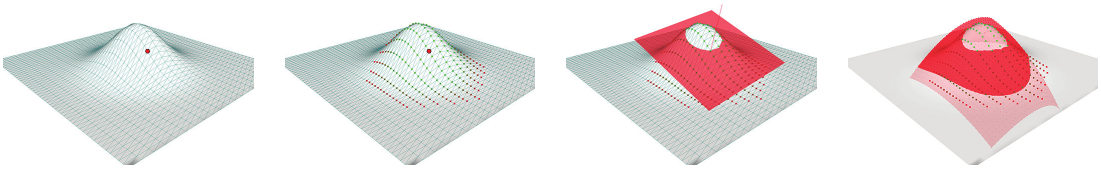


Figure 3.7: **Quadric fitting on the neighborhood expressed as a height field**. From left to right: evaluation point, neighborhood, reference plane, quadric. Image taken from [CPG09].

As said before, a limitation of considering the curvature tensor as a descriptor is that it only defines two directions of anisotropy, and thus it cannot characterize complex shapes such as T-junctions. In [ABG+12], Ammann et al. address it by analyzing the second-order curves fitted in multiple tangent directions instead of a second-order surface patch.

The main limitation of these approaches is due to the two-step fitting procedure: when the plane fitting fails, or when the neighborhood cannot be represented as a height field, the second fitted primitive is invalid. Moreover, plane fitting is not robust when the geometry represents complex shapes [GG07]. These issues can be fixed by using non-linear fitting of 3D quadrics [DB02, Pet02], but in practice using non-linear procedures is unstable, and they have a high computational cost.

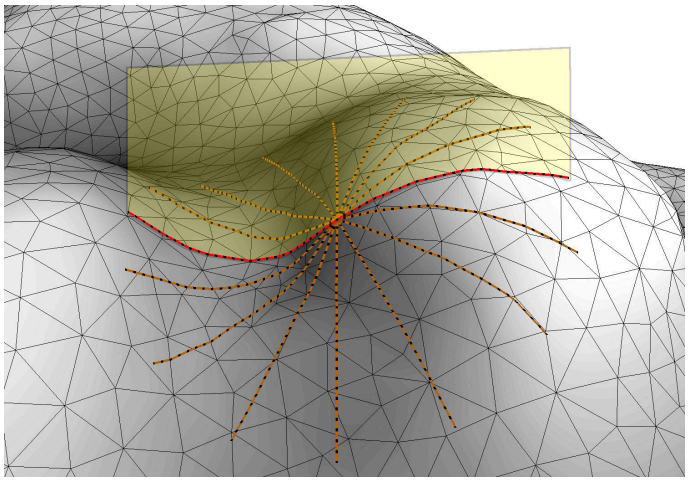


Figure 3.8: **Curve fitting on the neighborhood expressed as a height field**. 2D curves are fitted in multiple directions and then combined to describe the geometry. Image taken from [ABG+12].

3.3 Feature detectors

In this section, we introduce the notion of *features* together with some methods to detect them. We call a feature a point or a set of points that contains a relevant information for a given processing. In the literature, these points are called features, pertinents, or salients[1]. In the following, we use both feature and pertinent points with the same meaning interchangeably.

Feature detectors are often used as an intermediate step between a generic geometry characterization and a given specific process. They analyze results at one or multiple scales, and if possible, with one single descriptor. Using *feature detectors* produces two different types of output: 1) a dense characterization that associates a pertinence measure for each input sample, or 2) a sparse extraction of feature points depending on the inter-relation between samples.

Regarding these concepts, we present three types of *feature detectors* (illustrated in Figure 3.9):

a) *Space-varying detectors* detect features at a given scale and select a subset of samples with respect to the descriptor's spatial variation. They can possibly be evaluated at multiple scales, but each scale is analyzed individually.

b) *Scale-varying detectors* consider each sample independently within a scale interval, and they produce a dense multi-scale pertinence characterization in function of the descriptor's variation when scale changes.

c) *Scale-space detectors* extract features as a set of scale/position pairs by considering the descriptor's variation both in scale and in space.

We denote by $g(\mathbf{p}, t)$ the function that associates a descriptor value to a position $\mathbf{p}$ at a given scale t.

[1]Salient point: this term should be used carefully. Indeed, it originates from neuroscience and refers to elements that attract human attention in the visual field. Thus, the term saliency detection in Computer Graphics should be used only when it is based on perceptual studies.

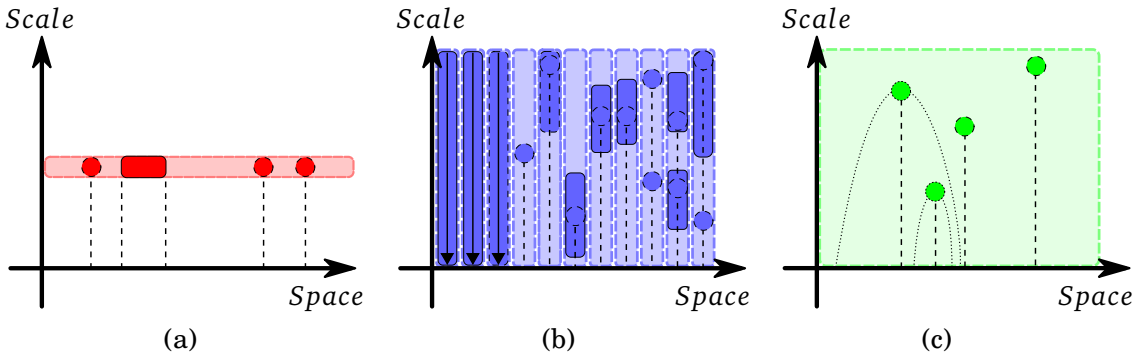


Figure 3.9: **Feature detector working space.** Scale-space representation of a 1D domain (2D curve), parametrized by the arc-length. We propose three types of feature detectors *(a) Space-varying detectors* that identify features at a given scale and select a subset of samples, *(b) Scale-varying detectors* that consider each sample independently within a scale interval and produce a dense multi-scale pertinence characterization, and *(c) Scale-space detectors* that extract features as a set of pairs scale/position.

3.3.1 Space-varying detectors

Here we consider the category of *feature detector* that takes into account the descriptor at a single scale. Of course, it is possible to consider *geometric descriptors* computed at various scales, but each of them are treated independently. As space-varying detectors work in the spatial domain, they can check the coherency between descriptors and thus extract features as sets of points (shown as a rectangular selection in Figure 3.9 (a)). We distinguish three classes of space-varying *feature detectors*. In the first one, features are detected as individual points, while in the two others, features are sets of points that are similar with respect to a given criterion: they belong to a same primitive, or they are linked to the same transformation. In the following, we denote by t_e an arbitrary scale used for the feature extraction.

Spatial extrema This first class detects features as individual points. The idea is very simple: the *feature detector* extracts points that are spatial extrema of $g(\mathbf{p}, t_e)$, or which have values below or above a given threshold. Extrema detection is used in [WB01] for the selection of salient points. In [SOG09], the extraction is done for a given arbitrary large scale in order to consider global structures instead of small geometric details. The detection of points belonging to crest lines is a common feature used in rendering and segmentation: it is performed by finding the extrema of k_1 [TG95, PAT00, Pet02]. In [TF95, Pet02], in order to select feature points, an adaptive threshold for the Gaussian curvature G is defined with respect to the mean curvature H. Another quite different and specific approach, proposed in [GMGP05], is based on *Shape Contexts* and selects feature points with most populated bins.

Primitive clustering This second class combines both the descriptor response and the relations between the evaluation points for the detection of features as sets of points. It is often used for segmentation. The idea is to find the set of points that best describes an underlying feature. For example, Gal and Cohen-Or [GCO06] proposed a region-growing

approach that selects feature regions by adding points that maximize a *saliency grade* function. Some more complex approaches, such as [ZZWC12], use a Laplacian matrix for variational segmentation: the error metric used to decide whether to add or not a point to a cluster depends on the dihedral angles that measure concavities and convexities.

Transformation clustering This third class uses clustering under transformations in order to select representative sets of points, by considering clusters either in transformation space [MGP06] (usually for symmetry detection), or in the spatial domain according to specific transformations. For example, the *area projection transform* proposed by [GL12] detects radial symmetry axes or points: at each scale, the surface is projected along its normal direction at a given distance. The relevant points in $\mathbb{R}^3$ are detected as the center of clusters formed by projected points, as shown in Figure 3.10.

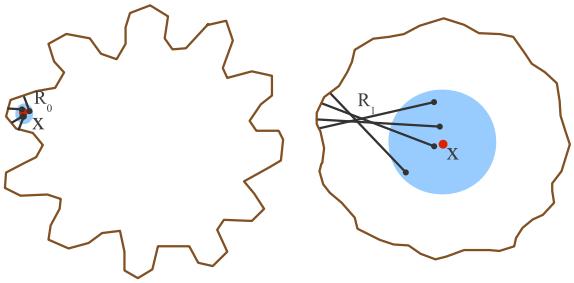


Figure 3.10: **Area projection transform**. Left: clusters corresponding to fine features are detected by a small projection distance R_0. Right: clusters corresponding to the global shape are detected by a larger projection distance, and then applied to a smoothed version of the object (other methods to handle coarse scales are proposed in the original paper). Image taken from [GL12].

Space-varying *feature detectors* extract features by analyzing spatial relations between points according to a *geometric descriptor* at a single scale. Their main limitation is that they require the users to choose a correct scale value. Indeed, the analysis can completely fail when the surface is not described at the right scale.

3.3.2 Scale-varying detectors

This second type of *feature detectors* is based on multi-scale *geometric descriptors*. The main idea is to characterize the geometry at multiple scales in order to generate a profile $g_{\mathbf{p}}(t) = g(\mathbf{p}, t)$ for each points. This profile is then summarized to reduce its memory consumption and to speed-up the comparison (for example, by comparing one scalar value instead of an array of scalars). We can distinguish two classes of scale-varying approaches depending on the ways the profile is summarized. The first one integrates descriptor values for the entire scale interval, and the second one extracts a pertinent scale, sometimes within a confidence interval, that is used as description.

Integration This class of approaches is represented as black arrows in Figure 3.9. It can be seen as a statistical analysis of the *geometric descriptor* response at multiple scales, since measures like mean and variance are involved. Hence, in [HFG+06], the authors describe surface roughness and sharpness, respectively given by the an integration of the

Integral Invariant $V_t(\mathbf{p})$, and by a measure of the variation of normals and positions in the scale interval. In the *mesh saliency* framework [LVJ05, LLKR07, CCFM08], the authors use non-linear normalization to combine saliency values at different scales. The main limitation is that combining values at multiple scales without taking into account the notion of pertinence may merge completely different features to an irrelevant description.

Pertinent scale This class of approaches uses $g_{\mathbf{p}}(t)$ to extract pertinent scales (the blue circle in Figure 3.9) by considering the way the descriptor changes during scale variation. Lindeberg proposes to extract a scale that corresponds to the strongest local maximum [Lin98] from the multi-scale signature. This approach is also used in [PKG03, MBO10]. Furthermore, [PKG03] propose to measure a pertinence associated to this extremum (the blue rectangles around the circles in Figure 3.9), by counting the number of scales that correspond to a descriptor value greater than a given threshold. The extrema detection requires a dense scale sampling to avoid to miss the pertinent scale. Furthermore, considering only one pertinent scale may not be sufficient to characterize complex surfaces.

3.3.3 Scale-space detectors

This last type of approaches combines the *geometric descriptor* analysis of both spatial and scale variation in order to detect features. Except the multi-scale extension of *area projection transform* [GL12], they are all based on the *scale-space theory* [Lin94], which defines the *scale-space*, a way to represent multi-scale descriptions, and the *scale-space analysis* to extract from this description a set of pertinent features. A scale-space can be filled with various descriptors. The most common and efficient construction is to consider the *curvature scale-space*, filled by computing the mean curvature at multiple scales using DoG. Once this is done, it is possible to compute the 0-*crossings* of the curvature by extracting, along the parametrization and for a given scale, the points where the curvature sign is changing. Using the mean curvature, these points are *inflexion points*. It is also possible to detect other kinds of feature by considering curvature γ-derivatives (for more details, see [Lin94]). Once this is done, the pertinent points can be detected by tracking these 0-crossings when scale varies, and extract the scale-space position where they annihilate, as shown in Figure 3.9 (green circles).

Even though this theory can be applied in various dimensions, it is best adapted to 2D curves that offer a regular one-dimensional parametrization space. The extension to 2D images yields the apparition of spurious 0-crossings due to the two-dimensional parameter space [Lin94, Rom09]. In this case, it is possible to detect features in the same way as the Scale Invariant Feature Transform (SIFT) algorithm [Low99] that extracts feature points as both scale and space 0-crossings (the two isolated green circles in Figure 3.9).

The scale-space can be adapted in two ways to deal with 3D objects: by considering voxel grids [SAS07, FBMB10] (and thus a three-dimensional parameter space) or local parametrizations on surfaces based on mesh connectivity [ZBVH09, MFK+10, DK12], which is the counterpart to image analysis. Some ameliorations have been proposed to improve surface smoothing [ZHL+09], or to use extrema of LoG both with voxels [WNK06, GW11] and surfaces [ZHDQ08]. Also note the recent work of [FW11] that uses the HKS to fill scale-space. Independently of the filling method, these approaches produce a sparse characterization of the shape. As shown in [Lin94], the 0-crossing can be instable regarding noise or input variation.

3.4 Discussion of previous work

In this chapter, we have presented some methods that are potentially usable in our context of shape matching for the characterization of 3D objects at multiple scales. In this section, we discuss more in-depth three concrete topics linked to our application: the meaning of *similarity*, the *scale invariance*, and the robustness to *topological changes*. These notions are then used to analyze previous work, and to motivate the work presented in the next chapter.

3.4.1 Application context

Similarity detection

Most of the time, objects are analyzed in order to detect *similarities*, for example for shape retrieval or matching. However in both cases, this term may refer to different meanings. For example, on the one hand, one might want to study whether two 3D surfaces represent the same human being, while on the other hand, one might want to decide whether two pieces of surfaces match exactly [vKZHCO11]. Independently of the application, a geometry analysis is most of the time used to characterize the surface and extract features. Then, a specific process exploits this description to perform various processing tasks: extraction of symmetry axes [MGP06], retrieval in databases [BBGO11], or fragment matching [HFG+06]. It is also possible to look for self-similarities, and detect coherent or regular structures or patterns on objects [MBB10, IT11a]. We identified two different meanings of *similarity* depending on the application.

In a first application class, for example for shape retrieval, two different geometries representing the same object have to be similar even though they correspond to different poses. In this case, a natural choice is to consider the intrinsic properties [SOG09, MBB10, BK10] that characterize the surface in a pose-invariant manner. More precisely, in the case of isometric deformations, the description has to be isometry-invariant. Otherwise, we have less guarantees on distance preservation, and the descriptions and their comparisons have to be almost isometry-invariant (or ϵ-isometry invariant, where ϵ is a tolerance threshold). This is for instance the case in the registration of scans of deformable objects.

In the second class, for example for shape matching, the detection of similar surfaces has to take into account both the object shape and the geometry details [HFG+06, IT11a]. Considering only intrinsic properties would not be pertinent since they might be similar even for incompatible surfaces. Hence, in this case, it is more relevant to focus on extrinsic properties.

Scale-invariance

The *scale-invariance* qualifies an analysis of a 3D object that provides an unvariable result even when an arbitrary scale factor is applied to the object. For example, extracting pertinent points of a given 3D object and its upscaled version should produce the same results. In practice, this is a difficult task because it often requires to define some thresholds that are strongly dependent on the evaluation scale.

A straightforward solution is to normalize the description values by a factor that depends on the size of the entire object. For instance, one can use the length of the object's bounding box diagonal as an approximation of the object size. Unfortunately, this approach introduces a dependency between the characterization of the surface and the size of the entire object, and this is not always desirable. Consider for example the comparison of two objects, the first one representing a human body, and the other one representing a finger of this human body. Even though the finger is locally exactly the same as the one on the complete body object, it is easy to see that the large difference of the bounding box size produces quite different normalizations and thus different descriptors.

Another solution is to normalize the description by the evaluation scale. The side effect is that two different features on the same object may have the same normalized descriptor at different scales, and thus they cannot be distinguished. In this case, it is necessary to compare the normalized descriptors only at the same evaluation scale. Furthermore, some methods define scale in another space than the description, and so the normalization cannot be coherent (for example Euclidean distances and distribution properties).

Topological changes

In practice, it is far from evident to use only intrinsic properties for shape retrieval when dealing with acquired data. Indeed, isometric or ϵ-isometric deformations applied on physical objects may produce topological changes on the acquired 3D object, and this invalidates local similarities based on intrinsic properties. This is illustrated in Figure 3.11 at the example of a human hand in three different poses. In a general context, without specific post-processing, the acquisition produces 3D objects with different topology in pose (a) compared to poses (b) and (c). We can imagine to build a *Self-Similarity Transformation Space* (SSTS) that associates to each pose a similarity value for a given neighborhood collection at a specific position with a given descriptor. Figure 3.11 (bottom row) illustrates this theoretical space for the specific position A at poses that are morphed from (a) to (b) and then to (c). The higher the value, the more similar is the pose compared to the initial position (in our case (a)). Using a geodesic or a diffusion collection may detect pose (a) to be different different to poses (b) and (c) because of the topological change, and introduce a discontinuity in the self-similarity measure. On the other hand, Euclidean collection may detect poses (a) and (b) to be similar, because of the small variation in ambient space (Euclidean distances and curvature). Hence, using diffusion with these objects in a context of shape retrieval requires to use topologically robust surface metrics to avoid false negatives [BBK+10].

3.4.2 Geometric descriptors

Requirements

The discussion of previous work established several requirements that the method we want to conceive should fulfill. Concerning the mathematical properties of 3D surfaces, the method should be related to the First or Second Fundamental Form in order to be able to capture local properties. However, shape matching refers to non-deformable objects that should be characterized by the way they are embedded in space, and hence an extrinsic characterization is preferable. As we want to characterize all scales, ranging from fine geometric details to overall shape properties, the proposed method must be usable with

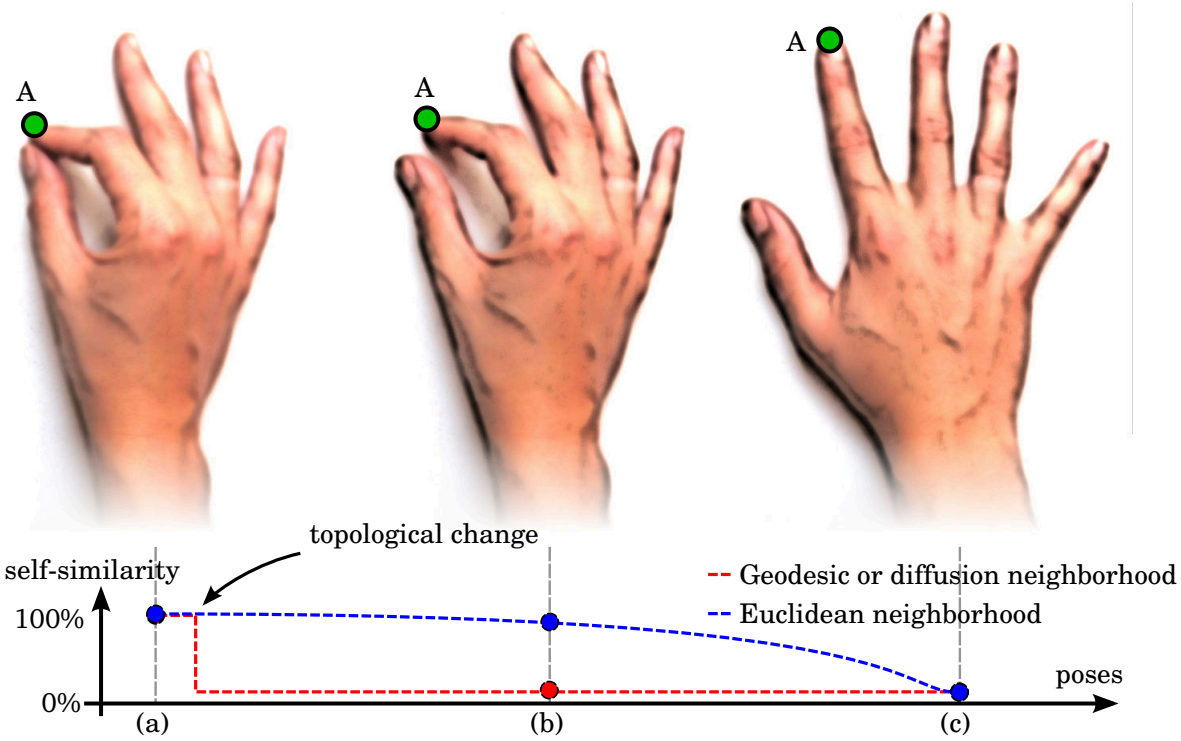


Figure 3.11: **Neighborhood collection and topological changes.** Top: human hand with three different poses. In a general context, the acquisition might produce 3D objects with different topology in pose (a) versus poses (b) and (c). Bottom: theoretical evolution of the similarity value at a specific position A between the input pose (a) and a morphing to (b) and (c) using geodesic/diffusion or Euclidean neighborhoods.

multi-scale neighborhood collection and smoothing. Another requirement is the usability: the analysis should work directly on acquired data, usually point sets, and should have as less as possible parameters to adjust in order to be usable by non-expert users.

Limitations of previous work

Among all presented methods to describe the geometry, only a few match our criteria. Indeed, discrete differential operators (Section 3.2.1) usually refer to the discrete Laplace-Beltrami (LB) operator which has been applied for meshes [Tau95, MDSB02, HPW06, WMKG07, BS07], and recently for point-sets [LLWZ12]. A common use is to compute this intrinsic operator to approximate the curvature. In our context, we strive for an extrinsic characterization, and thus approximating the LB operator to estimate the mean curvature (or an equivalent) seems to be hazardous. Direct methods that compute curvature have also been presented, but they usually rely on the connectivity of mesh representations, and they may be sensible to noise [Pet02, GG06]. Integration methods (see Section 3.2.2) produce more robust descriptors, but in practice, they are unusable at multiple scales with large objects due to the high computational cost, especially when dealing with point sets (due to the necessary step to approximate the surfaces or volumes and then compute intersections). However, in both cases, the use of differential quantities, such as the curvature, is not sufficient to characterize complex and anisotropic shapes, like T-junctions or sharp edges. This

limitation could be overcome using distribution analysis (see Section 3.2.3), but in practice this algorithm involves a variety of parameters that makes it complex to use. Hence, considering all these criteria, only two types of methods remain. First, diffusion methods (see Section 3.2.4) use the LB matrix to define multi-scale descriptors. Even though the proposed methods focus on the intrinsic characterizations of the shape, it might be possible to differentiate these descriptors and produce extrinsic invariants. However, although this topic seems to be promising, these methods can hardly be adapted to our context since we want to work directly with point sets. Hence, only regression-based methods (see Section 3.2.5) can theoretically be adapted: they deal with point sets, they are fully local and thus easily parallelizable, and they might lead to a continuous evaluation both in scale and space. The main limitations are due to the fitting process that often involves a plane primitive that is itself sensible to fold-overs and that might be unstable with complex shapes [GG07].

Another important topic is related to the unicity of the description with respect to shape variation. Indeed, there are some methods that first use a given descriptor to characterize the geometry, and then they extract features, but for the description, instead of using the latter descriptor, these methods use a different invariant [HFG+06, GCO06, WNK06, SAS07, ZHDQ08, ZBVH09, FBMB10, IT11b, MFK+10, MBO10, GW11, DK12]. This second step is either based on a 3D extension of the SIFT descriptor, or on a *geometric descriptor* that is specific to the application case and that might be hard to adapt for other contexts. This is usually due to the lack of description power of the descriptors used for the feature detection, which implies to use a more robust descriptor to characterize extracted complex features. Moreover, these methods might lead to redundant computations between the two descriptions.

3.4.3 Feature detectors

Requirements

Considering the *feature detectors*, we have presented three types: space-varying, scale-varying, and scale-space approaches. In our context, we want our method to handle huge and dense 3D objects with relevant information at multiple scales. For example, consider the illustration of a 2D curve in Figure 3.12 that is defined as a circle with a displacement pattern. This simple object contains few levels of detail: the pattern is regular on the object and refers to a single scale.

Limitations of previous work

When we consider space-varying approaches (see Section 3.3.1), the feature extraction produces different results with respect to the scale level used to compute the *geometric descriptor*. Setting this critical parameter requires additional knowledge about the object's properties and meaning. On the other hand, when the scale parameter is well chosen and corresponds to the displacement pattern, analyzing the spatial descriptor variation might produce a sufficiently dense shape characterization. Scale-varying approaches (see Section 3.3.2) can be used to automatically detect pertinent scales. In this case, the *geometric descriptors* are summarized by a single scalar value to permit automatic scale extraction and potentially pertinence measures. These approaches are thus less scale dependent: they still require the specification of a scale range: this can be achieved with a provided factor

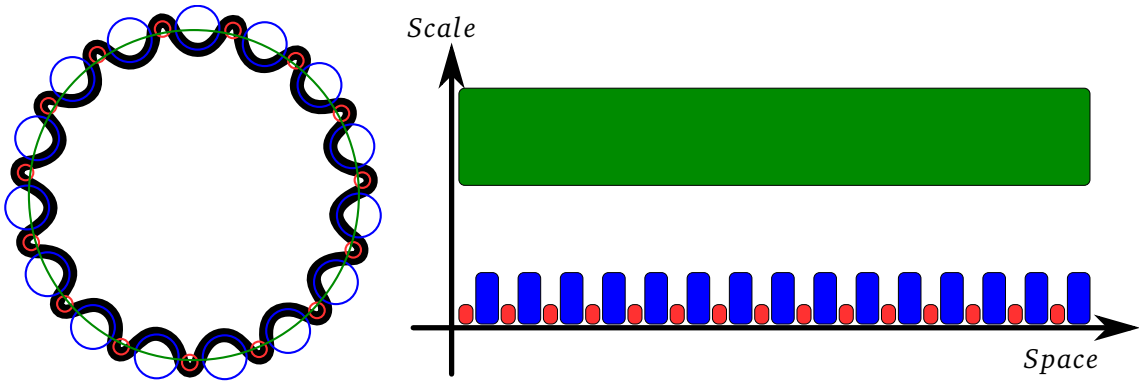


Figure 3.12: **Example of a multi-scale object.** Left: a 2D curve (in black) composed of a circle and a regular displacement pattern. Right: the scale-space locations that contain relevant information. The circle primitives that can be associated to each of these locations are shown as overlay on the 2D curve.

of the object's bounding box (the descriptor may not be sensible to these parameters). However, these methods process each sample independently, and thus they cannot ensure any spatial continuity or coherence despite a dense feature extraction. Furthermore, they also refer to a single scalar description that might not be compatible with some descriptors. The last type of *feature detectors* (see Section 3.3.3) analyzes both spatial and scale variations in order to produce a coherent bi-lateral extraction. However, they are based on the detection of 0-crossings in scale-space (with tracking or not) that have been proved unstable [Lin94]. Furthermore, these approaches produce a sparse characterization that is not sufficient in some applications as, for example, shape matching.

3.4.4 Design of our approach

All these limitations motivate the elaboration of a new *geometric descriptor* associated to a new dense multi-scale *feature detector*. In the next chapter, we present our new unified framework with a second-order regression-based descriptor and continuous feature extraction that is coherent in scale and space. We compare the descriptor to previous work in terms of robustness and the capacity to represent complex shapes. Moreover, we compare the feature extraction to scale-space analysis (with and without the tracking of 0-crossings) in terms of robustness to noise and to input variation, and to the capacity to detect various features.

Chapter 4

Growing least squares

In this chapter, we present the core theoretical part of this thesis: the elaboration of fundamental tools for the analysis of 3D objects represented by unorganized point sets. The objective is to define a unified framework to analyze 3D objects at multiple scales (Section 4.1). In particular, we address two theoretical topics. First, we define a new robust geometric descriptor that characterizes the geometry at three differential orders using an efficient algebraic hyper-sphere fitting procedure. Second, we propose a new scale-space analysis that continuously extracts features, even complex and anisotropic ones, as an alternative to state of the art approaches [Lin98, Low99]. In the two following sections, we discuss more in-depth the technical choices for the design of our analysis framework. We compare with previous work both our descriptor (Section 4.2) and our analysis (Section 4.3). Finally, in the last section, we present some results and promising preliminary studies for future work (Section 4.4).

4.1 Framework

In this section, we present the core of our method in a general manner by considering an ambient space of dimension d (with codimension 1). Note that we tested our framework for 2D curves ($d = 2$) and 3D surfaces ($d = 3$), and it probably requires more investigations in higher dimensions.

The key idea of our approach is to perform the scale-space analysis of 2D curves and 3D surfaces by means of continuous algebraic fits. More precisely, we propose to build a scale-space through least-square fits of a low-degree algebraic surface onto neighborhoods of continuously increasing sizes. In some sense, this can be seen as an adaptation of the Moving Least Squares formalism [Lev98] to continuously varying scales, hence the name "Growing Least Squares" (GLS). The use of an algebraic surface ensures robust fits even at large scales and yields a rich geometric descriptor with only a few parameters, called in the following the *GLS descriptor*. The continuity of the fitting process through scales provides for a stable and elegant analysis of geometric variations, under the name *GLS analysis*.

4.1.1 Scale-space via local regression

The first step of our approach is to characterize any point $\mathbf{p}$ of a manifold at any scale t by a low-degree algebraic surface that best approximates its neighborhood $\mathscr{P}_t$. In a discrete setting, our manifold is described by a set of points $\mathbf{q}_i \in \mathbb{R}^d$, and the neighborhood $\mathscr{P}_t$ consists of the set of data points contained in a ball of radius t centered at $\mathbf{p}$: $\mathscr{P}_t = \{\mathbf{q}_i ; \|\mathbf{q}_i - \mathbf{p}\| \leq t\}$. Inspired by recent work on MLS reconstruction [GG07], we use algebraic hyperspheres which have the advantage of being easy to fit in a robust manner, while providing second-order information with a minimal number of parameters. To this end, we assume

Related publication: [MB12]

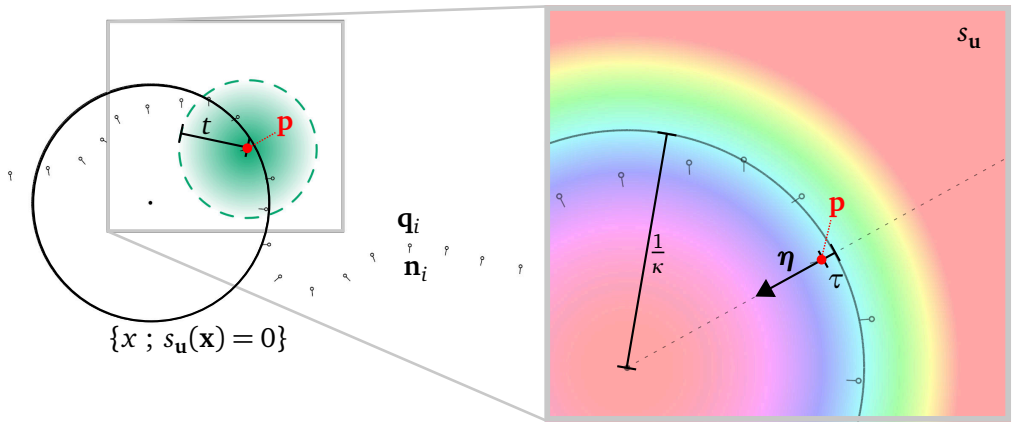


Figure 4.1: **Fitting and reparametrization.** The weight function (in green) around a point $\mathbf{p}$ is defined in a neighborhood of size t (dashed green line). The points $\mathbf{q}_i$ and normals $\mathbf{n}_i$ that belong to this neighborhood are fitted by an algebraic hyper-sphere $s_{\mathbf{u}}$; its 0-isosurface is shown in black, and its scalar field with a color code. This sphere is reparametrized by three geometric parameters: the mean curvature κ, the offset τ, and the gradient direction $\boldsymbol{\eta}$ at $\mathbf{p}$.

that each point $\mathbf{q}_i$ is equipped with a normal $\mathbf{n}_i \in \mathbb{R}^d$. In case normals are not provided, we refer to normal estimations methods, as done in previous work [AB98, PKG03, MN03, DG06, DLS05, Li210, BM12].

Fitting. An algebraic sphere is implicitly defined as the 0-isosurface of the following scalar field (see Figure 4.1):

$$s_{\mathbf{u}}(\mathbf{x}) \quad = \quad [1 \ \mathbf{x}^T \ \mathbf{x}^T\mathbf{x}] \cdot \mathbf{u}, \tag{4.1}$$

where $\mathbf{u} \in \mathbb{R}^{d+2}$, $\mathbf{u} = [u_c \ \mathbf{u}_\ell \ u_q]^T$ is the vector of the constant, linear, and quadratic parameters. In order to fit such a sphere to a set of neighborhood points $\mathscr{P}_t$, we use the fast fitting technique of Guennebaud et al. [GGG08]. First, $\mathbf{u}_\ell$ and u_q are computed by minimizing $\sum_i w_i(t)\|\nabla s_{\mathbf{u}}(\mathbf{q}_i) - \mathbf{n}_i\|^2$, where $\mathbf{q}_i \in \mathscr{P}_t$ and w_i is a *scale-dependent* weight function:

$$w_i(t) \quad = \quad \left(\frac{\|\mathbf{q}_i - \mathbf{p}\|^2}{t^2} - 1 \right)^2. \tag{4.2}$$

Second, the constant coefficient u_c is obtained by minimizing in a least square sense the algebraic distance to the samples: $\sum_i w_i(t)\|s_{\mathbf{u}}(\mathbf{q}_i)\|^2$. This minimization expresses the fitted primitive in a global frame. In order to be translation invariant, we define the neighborhood in a centered basis by defining $\bar{\mathbf{q}}_i = \mathbf{q}_i - \mathbf{p}$.

As shown in [GGG08], the two above-mentioned minimizations yield closed-form formula that can be re-expressed as

$$\begin{aligned} u_q &= \frac{1}{2} \frac{\sum w_i \bar{\mathbf{q}}_i^T \mathbf{n}_i \ - \ \sum \tilde{w}_i \bar{\mathbf{q}}_i^T \sum w_i \mathbf{n}_i}{\sum w_i \bar{\mathbf{q}}_i^T \bar{\mathbf{q}}_i \ - \ \sum \tilde{w}_i \bar{\mathbf{q}}_i^T \sum w_i \bar{\mathbf{q}}_i} \\ \mathbf{u}_\ell &= \sum \tilde{w}_i \mathbf{n}_i - 2u_q \sum \tilde{w}_i \bar{\mathbf{q}}_i \\ u_c &= -\mathbf{u}_\ell^T \sum \tilde{w}_i \bar{\mathbf{q}}_i \ - \ u_q \sum \tilde{w}_i \bar{\mathbf{q}}_i^T \bar{\mathbf{q}}_i \end{aligned} \tag{4.3}$$

where $\tilde{w}_i$ is the normalized weight of the sample $\mathbf{q}_i$: $\tilde{w}_i = w_i / \sum_j w_j$.

Normalization. Contrary to [GG07, GGG08], our goal is not to reconstruct a surface from a point set, but instead to analyze the underlying shape at multiple scales. To this end, we want to assign a *unique* and *meaningful* geometric descriptor for any choice of the point $\mathbf{p}$ and the scale t. A straightforward solution would be to use the center $\mathbf{c}$ and radius r of the hyper-sphere. Unfortunately, this leads to degenerated cases when the surface is locally planar: in particular, $\mathbf{c}$ becomes undefined.

We thus rather consider the scalar field itself $s_{\mathbf{u}}$ as a geometric descriptor. However, there exists an infinity of scalar fields (based on scalar multiples of $\mathbf{u}$) that correspond to the same hyper-sphere. To solve this issue and consistently pick a unique solution, we use Pratt's normalization [Pra87]: its basic idea is to constrain the scalar field to have a unitary gradient vector on the 0-isosurface, yielding:

$$\hat{\mathbf{u}} = [\hat{u}_c \ \hat{\mathbf{u}}_\ell \ \hat{u}_q]^T \quad = \quad \mathbf{u}/\sqrt{\|\mathbf{u}_\ell\|^2 - 4u_c u_q}. \tag{4.4}$$

This choice has the additional advantage to make algebraic distances near-Euclidean for points close to the 0-isosurface.

Reparametrization. After the normalization, we obtain a scalar field $s_{\hat{\mathbf{u}}}$ for which a geometric interpretation is far from evident. First, $\hat{u}_c$ and $\hat{\mathbf{u}}_\ell$ do not correspond to any measurable physical quantity. Second, all $d+2$ parameters are still interdependent, since the normalization binds them together with: $\|\hat{\mathbf{u}}_\ell\|^2 - 4\hat{u}_c\hat{u}_q = 1$. We propose an alternative parametrization of the scalar field (illustrated in Figure 4.1). Intuitively, its parameters consist of: the algebraic offset distance τ between the evaluation point $\mathbf{p}$ and the 0-isosurface; the unit normal $\boldsymbol{\eta}$ of the scalar field at $\mathbf{p}$; and the signed curvature κ of the hyper-sphere. When the fitting degenerates to a plane, τ represents the distance from the origin to the plane, $\boldsymbol{\eta}$ its normal, and κ vanishes (see Figures 4.3 and 4.4 for 2D and 3D illustrations, respectively).

Formally, the geometric parameters are given by:

$$\tau = s_{\hat{\mathbf{u}}}(\mathbf{p}) \ ; \quad \boldsymbol{\eta} = \frac{\nabla s_{\hat{\mathbf{u}}}(\mathbf{p})}{\|\nabla s_{\hat{\mathbf{u}}}(\mathbf{p})\|} \ ; \quad \kappa = 2\hat{u}_q. \tag{4.5}$$

In practice, since we use a centered basis, we can compute τ as:

$$\tau = s_{\hat{\mathbf{u}}}(0) = [1\ 0\ 0] \cdot \hat{\mathbf{u}} = \hat{\mathbf{u}}_c, \tag{4.6}$$

and $\boldsymbol{\eta}$ as:

$$\boldsymbol{\eta} = \frac{\hat{\mathbf{u}}_\ell}{\|\hat{\mathbf{u}}_\ell\|}, \tag{4.7}$$

$\hat{\mathbf{u}}_\ell$ being the gradient of the scalar field at the center of the basis.

As said before, thanks to Pratt's normalization, the offset τ provides a close approximation to the Euclidean distance between $\mathbf{p}$ and the 0-isosurface. The normal parameter $\boldsymbol{\eta}$ provides the direction to the point on the hyper-sphere that is closest to $\mathbf{p}$. The curvature parameter simply corresponds to the inverse of the hyper-sphere radius r, and has the advantage of behaving continuously when passing through a locally planar surface, while r tends towards infinity.

Note that with these parameters, the scalar field can no longer be expressed as a linear combination of monomials, but it can be retrieved by

$$s_{\hat{\mathbf{u}}}(\mathbf{x}) = s_{\tau,\boldsymbol{\eta},\kappa}(\mathbf{x}) = \tau + (1 + 2\tau\kappa)^{\frac{1}{2}}\boldsymbol{\eta} \cdot (\mathbf{x} - \mathbf{p}) + \frac{\kappa}{2}(\mathbf{x} - \mathbf{p})^2. \tag{4.8}$$

Fitness. Once reparametrized, the scalar field fitted from $\mathscr{P}_t$ yields a univocal geometric descriptor, invariant to rigid transformations. However, a given geometric descriptor can be associated to a space of generator neighborhoods. Considering the sphere as a whole instead of, for instance, local curvature only, already permits to significantly reduce the size of these spaces. It is interesting to remark that they can be further reduced by looking at the fitness φ that shows how close the $\mathbf{q}_i$ are to the fitted scalar field $s_{\mathbf{u}}$. We define this additional parameter by $\varphi = \sum_i w_i(t) \nabla s_{\mathbf{u}}(\mathbf{q}_i) \cdot \mathbf{n}_i / \sum_i w_i(t)$, where w_i is the same weighting function as before. Using the fitting equations (Eq. 4.3), it can be shown that for this fitting procedure, φ boils down to Pratt's norm for $\mathbf{u}$: $\varphi = \|\mathbf{u}_\ell\|^2 - 4u_c u_q$. Note that by construction, φ is dimension-less, scale-invariant, and varies in the $[0,1]$ range, with $\varphi = 1$ meaning a perfect alignment between the fitted scalar field and the input normals. As illustrated in Figure 4.2, this typically permits to disambiguate between surfaces that locally have the same geometric description, but depart from a pure algebraic sphere.

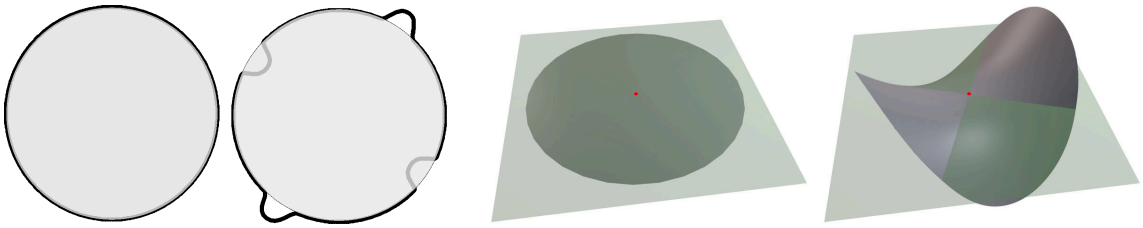


Figure 4.2: **The fitness parameter** φ helps disambiguate two identical fits, e.g., a smooth versus a bumped neighborhood (left), or a flat versus a saddle configuration (right).

The chain of operations described in this section augments an arbitrary point $\mathbf{p}$ at an arbitrary scale t with a geometric descriptor that characterizes the data points $\mathbf{q}_i \in \mathscr{P}_t$, as illustrated in Figure 4.3(a-b) for a 2D curve, and Figure 4.4(a-b) for a 3D surface. It thus describes an elegant method for building a fully continuous scale-space from sampled manifolds in arbitrary dimensions, providing meaningful surface information in the form of the τ, $\boldsymbol{\eta}$ and κ geometric parameters, and a fitness parameter φ that further helps disambiguate similar descriptors. Another handy property of our geometric descriptor is that negating its parameters yields the *complement* descriptor, which is equivalent to the descriptor of the same surface with an opposite orientation. One may be tempted to use it to only track 0-crossings of the curvature κ for instance, as in previous work. Instead, we show in the next section that the scale-space analysis can also be performed in a fully continuous manner.

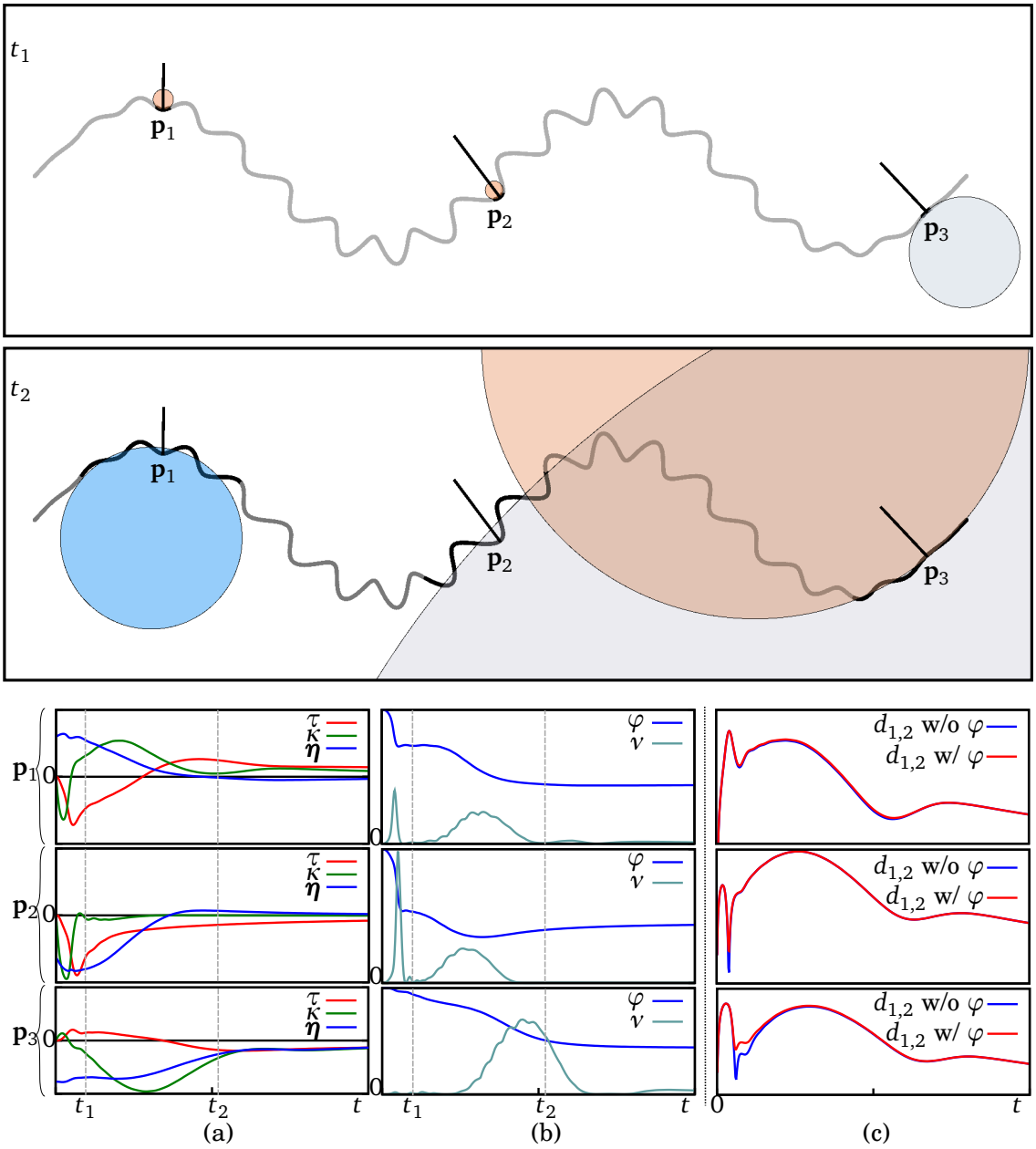


Figure 4.3: **2D analysis of a sinus curve.** Top: the analysis of a synthetic 2D curve composed of two sinusoids of different frequencies is illustrated at three different points. We show their geometric descriptor at two scales t_1, t_2. Bottom: (a) the geometric descriptor parameters are visualized for all scales with one point per row. In (b), we display their geometric variations and fitness: note that the third point has a more stable structure at small scales since the magnitude of the high-frequency component is low at its location. In (c), we display the dissimilarity measures for all pairs of points. Observe how the use of the fitness parameter helps disambiguate between the two types of inflexion points at intermediate scales. In all plots, the scale sampling is quadratic.

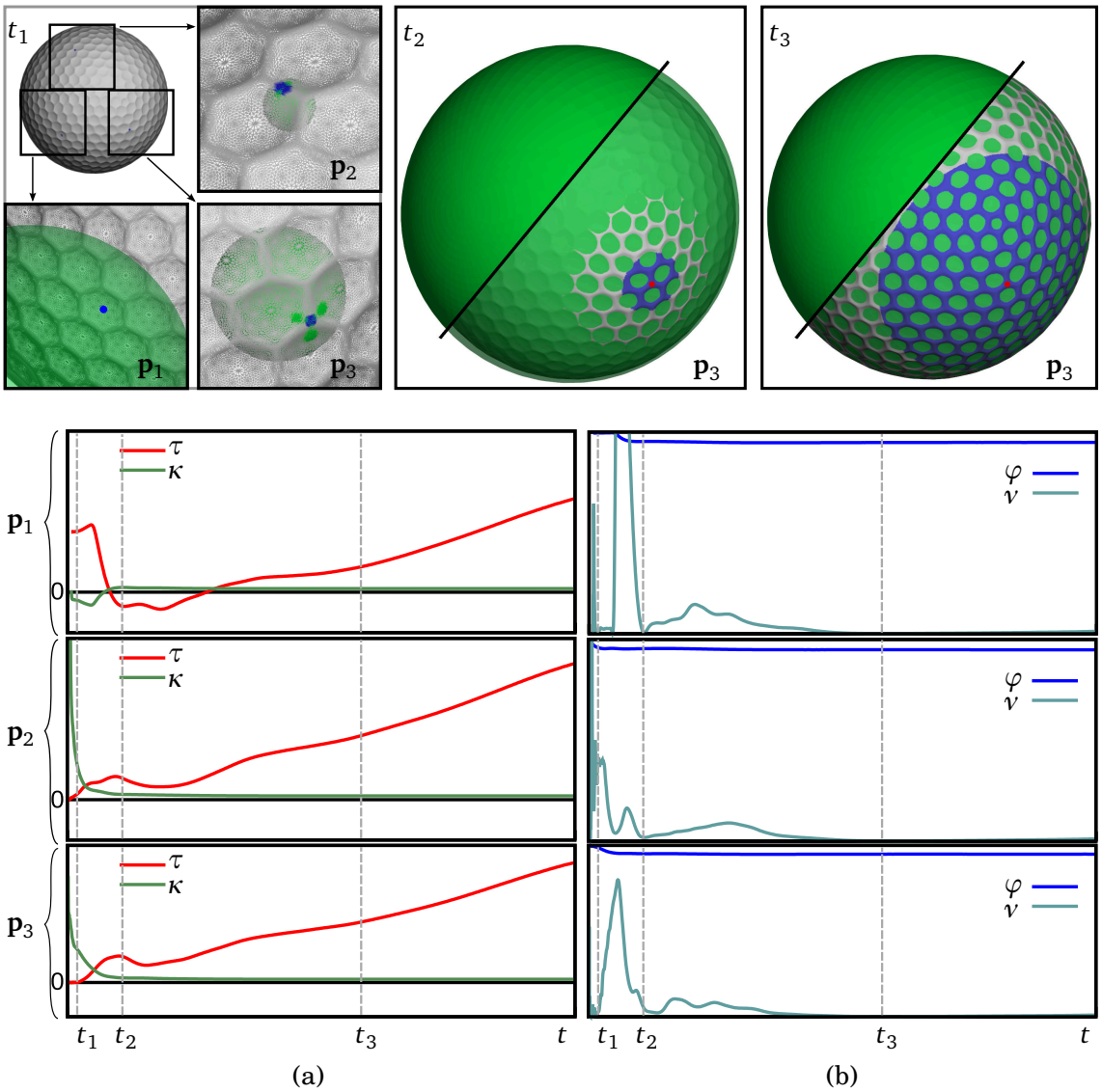


Figure 4.4: **3D analysis of a golfball object.** Top: the analysis of a synthetic 3D golfball object is illustrated at three different points: a concavity $\mathbf{p}_1$, an edge $\mathbf{p}_2$, and a junction $\mathbf{p}_3$. This is shown at a small scale t_1 in (left), where we see that the three points (with their neighborhoods in blue) have quite dissimilar geometric descriptors (the fitted sphere is shown in transparent green). However, at medium and large scales t_2 and t_3 (middle and right), all three points converge to the same global sphere. Bottom: this is best observed in (a), where we display the geometric parameters (except for $\boldsymbol{\eta}$), and in (b), where all three points converge together. In all plots, the scale sampling is quadratic.

4.1.2 Continuous scale-space analysis

The main purpose of analyzing a spatial signal in scale-space is to track its variations at increasing scales to discover its geometric structure. A classical approach consists in tracking invariants in the form of 0-crossings of a *spatial* derivative of the signal, and then find locations in scale-space where they get annihilated [Lin94]. Another approach is to extract 0-crossings of both *spatial* and *scale* derivatives (named SIFT-like approaches below) [Low99].

As mentioned in Section 3.3.2, the tracking of 0-crossings leads to many shortcomings, especially when trying to deal with manifolds. Most importantly, it restricts the analysis to a subset of locations in scale-space while requiring a parametrization.

We propose a different approach to discover the multi-scale structure of a manifold. Our key insight is to observe that, in general, a pertinent scale for $\mathbf{p}$ is one where its geometric descriptor exhibits *minimal variation* when the neighborhood size increases. This suggests that, at such scales, the parameters of our descriptor do not crucially depend on scale, but rather indicate stable geometric properties of the manifold. In this section, we focus on the derivation of such a general geometric variation. We will show its relevance in Section 4.3 and how it could be exploited in Chapter 5.

One may think that the curvature parameter κ is the one mostly involved in geometric variations. Figure 4.5 shows counter examples where either τ or $\boldsymbol{\eta}$ have significant influence. We thus compute the variations of all 3 geometric parameters and combine them in a natural fashion to yield a geometric variation function $\nu(\mathbf{p}, t)$ that describes the scale-space structure of the input manifold.

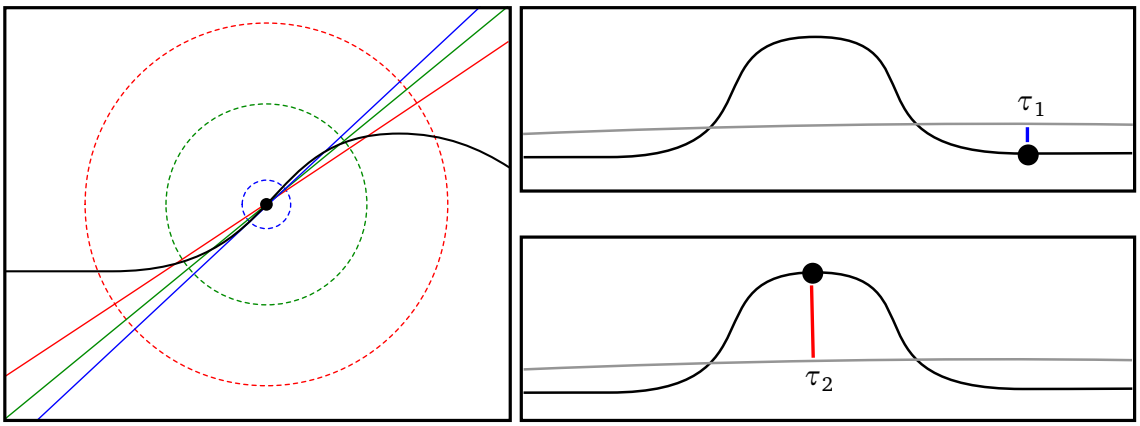


Figure 4.5: **Variations of η (left) and τ (right).** Left: hyper-spheres fitted for the black point of a dense 2D curve (in black) with various support sizes (shown as dashed spheres). The three configurations yield planes with different orientations represented by η (shown in blue, green, and red). Right: At a large scale, two evaluation points are characterized by similar fitted spheres (in gray), but with different values of τ, which captures the offset due to the bump.

Scale derivatives. The variation of the *geometric descriptor* parameters that we are looking for are simply given by their partial derivatives along the scale dimension at $(\mathbf{p}, t)$: $\frac{\delta\tau}{\delta t}$, $\frac{\delta\eta}{\delta t}$, and $\frac{\delta\kappa}{\delta t}$. We emphasize that we are interested in the fitted hyper-sphere geometric variations only, hence the fitness φ does not play a role here. Since we make use of a local

regression that is both continuous and given in closed-form, these derivatives are easily computed analytically, provided the weight functions themselves are differentiable. Their computation does not yield any difficulty, and simply involves differentiating the chain of equations presented in Section 4.1.1, i.e. from top to bottom: weighting (Eq. 4.2), fitting (Eq. 4.3), normalization (Eq. 4.4) and reparametrization (Eq. 4.5). These steps are described in Appendix A.

Geometric variation. The geometric variation function $v(\mathbf{p}, t)$ is obtained by a weighted squared sum of these partial derivatives. In order not to introduce any bias, a special care has to be taken in the choice of these weights. In particular, we propose the following weighting scheme:

$$v(\mathbf{p}, t) \quad = \quad \alpha_v \left(\frac{\delta \tau}{\delta t} \right)^2 + \beta_v \left(t \frac{\delta \boldsymbol{\eta}}{\delta t} \right)^2 + \gamma_v \left(t^2 \frac{\delta \kappa}{\delta t} \right)^2 , \tag{4.9}$$

which has the fundamental advantages to yield a dimension-less and scale-invariant measure. Indeed, let us for instance choose meters m for the unit of length. Thanks to our intuitive reparametrization of Section 4.1.1, we have τ in m, the unit-less $\boldsymbol{\eta}$, and κ in m^{-1}. Moreover, by construction it is reasonable to expect to have τ to be mostly comprised in $[-t, t]$, κ in $[-\frac{1}{t}, \frac{1}{t}]$, while $\|\boldsymbol{\eta}\| = 1$. Therefore, a reasonable choice is to scale the parameters $(\tau, \boldsymbol{\eta}, \kappa)$ by $(1/t, 1, t)$, respectively, in order to get scale-invariant and unit-less quantities of the same order of magnitude. Finally, in order to compensate for the differentiation over the scale t that is in m, it is natural to multiply by the scale t, thus leading to the scaling factors $(1, t, t^2)$ of Equation 4.9. We propose to use $\alpha_v = \beta_v = \gamma_v = 1$ in order to naturally give equal importance to each parameter, and we plan to study these weighing parameters in future work.

The function $v(\mathbf{p}, t)$ is one of the the key contributions of this framework. It provides a continuous description of pertinent scales for any point $\mathbf{p}$, and it is robust to small changes in the input. This is to contrast with previous approaches that rely on the annihilation of extremal points (0-crossings) and may lead to altogether different structures when the input changes slightly, as shown in Section 4.3.1. Most importantly, our approach is the first to identify *multiple* pertinent scales for individual points on manifolds, as shown in the examples in Figures 4.3(c) and 4.4(d) for the 2D and the 3D case, respectively. This opens the door to many new applications, for which we outline a few examples in Section 4.4.

4.1.3 Pairwise dissimilarity in scale-space

For a variety of applications, it is also interesting to compare a pair of arbitrary scale-space locations $(\mathbf{p}_a, t_a)$ and $(\mathbf{p}_b, t_b)$. In this context, it is crucial to provide a measure that is invariant to similarity transformations. Invariance to translation is readily available because the geometric parameters are defined relative to their fitted points. Since our descriptor is isotropic, invariance to rotation is achieved by aligning the respective unit normals $\boldsymbol{\eta}_a$ and $\boldsymbol{\eta}_b$, which amounts to ignore these parameters. Finally, scale invariance is obtained using the weights derived in the previous paragraph, and we thus define the dimension-less dissimilarity function $d_{a,b}$ by

$$d_{a,b} \quad = \quad \left(\frac{\tau_a}{t_a} - \frac{\tau_b}{t_b} \right)^2 + \left(t_a \kappa_a - t_b \kappa_b \right)^2 + \left(\varphi_a - \varphi_b \right)^2 \tag{4.10}$$

Here, the fitting errors φ_a and φ_b help disambiguate between similar descriptors that may correspond to different surfaces. Recall that this dissimilarity function $d_{a,b}$ is dimensionless, scale-invariant, and of the same order of magnitude as the other quantities since it varies between 0 and 1.

In practice, we propose to use a per-scale dissimilarity function $d_{\mathbf{p},\mathbf{q}}$ to compare arbitrary positions $\mathbf{p}$ and $\mathbf{q}$ at scale $t = t_1 = t_2$:

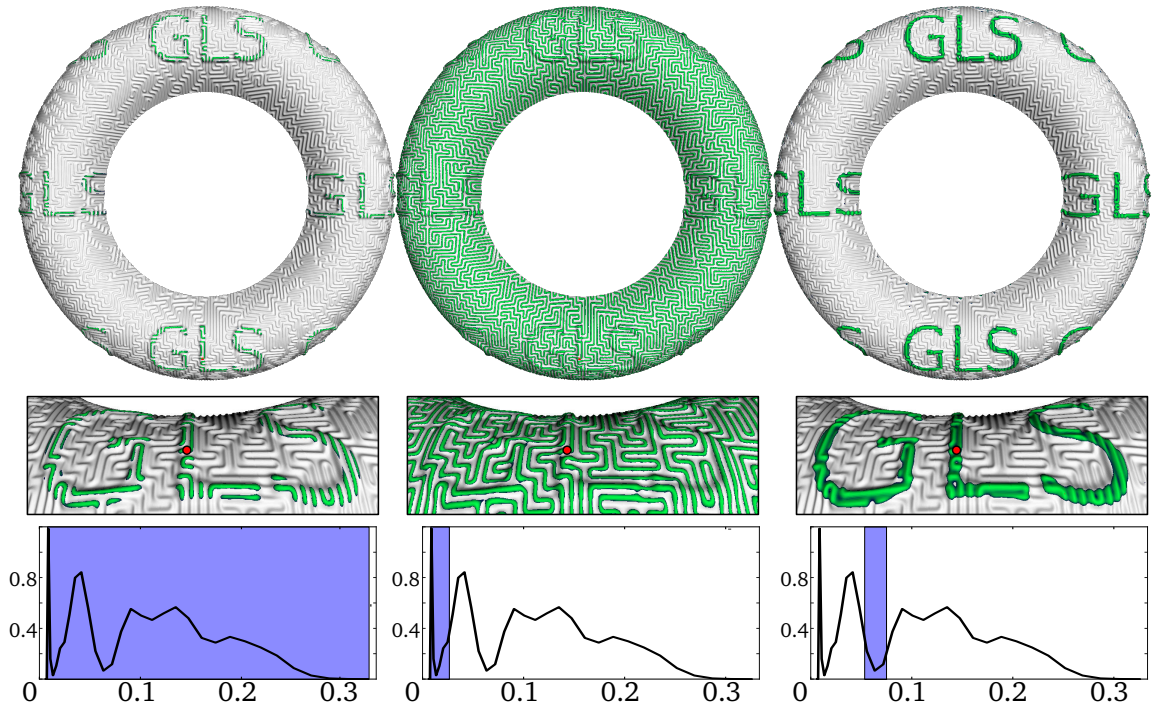
$$d_{\mathbf{p},\mathbf{q}}(t) \quad = \quad \frac{\left(\tau_a - \tau_b\right)^2}{t^2} + t^2\left(\kappa_a - \kappa_b\right)^2 + \left(\varphi_a - \varphi_b\right)^2 \tag{4.11}$$

This definition is more adapted to usual application cases, and it is illustrated in Figure 4.3(d), where we compare three pairs of points at multiple scales. We present some other examples in Section 4.2, where we combine geometric variation and dissimilarity measurements at multiple scales.

Multi-scale dissimilarity. The ability to evaluate local surface similarities is the cornerstone of many matching or registration techniques. However, the notion of multi-scale similarity hides an important question: which scales should actually be taken into account? One may think that taking all scales at once is a natural solution. However, as shown in Figure 4.6(a), this might not always be the case, at least for some applications. The question becomes delicate as soon as multiple scales are nested in a same object: one may want to find similarities at the smallest of these scales, or perhaps in other situations at the largest scale.

Our approach permits to make this choice in a continuous manner. For example, consider the following simple picking tool: the user selects a point on a surface, and the system finds all similar points on the same object, given a rough scale prior. The basic idea is to combine the global prior with the local geometric variation to compute a per-scale dissimilarity, which is then integrated over the scales t_i to yield a multi-scale dissimilarity $D_{\mathbf{p},\mathbf{q}} = \sum_i d_{\mathbf{p},\mathbf{q}}(t_i)h(t_i)$, where h is a normalized weighting function over scales that defines the global prior. In our example, we use a simple box filter for h.

Figure 4.6(b-c) visualizes $D_{\mathbf{p},\mathbf{q}}$ as a function of $\mathbf{q}$ (in blue) for a given point $\mathbf{p}$ (in red) located both on a small ridge and on the 'L' of the GLS acronym. By varying the global prior, our method identifies unambiguously either one detail layer or the other. Although our approach is based on an isotropic regression, this result demonstrates that anisotropic features at pertinent scales are properly extracted, even junctions and corners. This is due to the fact that an anisotropic feature can be considered as a set of isotropic descriptors at different locations. We address this topic in more details in Section 4.4.2. Moreover, this is done irrespective of the shape of the base surface, since the letters GLS are extracted similarly on different locations of the torus.



(a) All scales (b) Fine scales (c) Coarse scales

Figure 4.6: **Multi-scale similarity.** Top and middle rows: For a selected point (in red), similar points are selected (in green) via our dissimilarity measure. The similarity is computed for each vertex and interpolated per fragment during the rendering. Bottom row: the type of selected feature depends on a user-controlled global prior (shown as a blue box), which is locally refined by our geometric variation. In (a), all scales are selected. In (b), only the fine displacement pattern emerges. In (c), the large-scale GLS letters are properly segmented (GLS torus: 500k pts, 20 scales, 42sec).

4.2 Comparison with previous work: GLS Descriptor

In this section, we demonstrate the benefits of our scale-space construction and compare it to standard approaches. We first compare our mean curvature estimation with Discrete Curvature (DC) estimation for triangle meshes [MDSB02] and with two-step quadric fitting for point sets [FJ89, BSF02, CP03, GCO06, CPG09]. Then, we only consider methods that, like ours, avoid fold-over issues, do not require any parametrization and provide valid approximations of curvature at all scales: DoG mean curvature estimation [ZBVH09] and Covariance Analysis (CA) [PKG03].

Most of the comparisons of this section are done on 2D curves for two reasons, 1) we want to visualize the different scale-spaces in a comprehensive manner, and 2) we want to permit comparisons with DoG and 0-crossings. We display curvature as color in scale-space or directly on 3D objects, with orange colors for concavities and blue colors for convexities.

4.2.1 Multi-scale mean curvature

An important property of our method is that is based on the estimation of differential quantities at multiple scales (such as curvature). In this section, we compare our curvature measure κ with previous methods in order to highlight its stability and relevance in critical cases.

Two-step quadric fitting. In a first test, shown in Figure 4.7, we compare our method to quadric fitting [FJ89, BSF02, CP03, GCO06, CPG09], which estimates curvature using a two-step fitting procedure: first, estimating a plane, and then, fitting a quadric on the generated height data. This method is commonly used to characterize point sets at multiple scales. In order to have comparable estimations, the fit is performed on 2D data (a 1D manifold with only one curvature value) and without fold-overs (which would invalidate the height profile assumption). In practice, we use a first-order regression to compute the reference plane, and then we use a linear least square minimization on the resulting height data to get a polynomial representation of the quadric as $f(u) = au^2 + bu + c$. The curvature is computed as $\kappa_f(u) = \frac{2a}{\left(1+(2au+b)^2\right)^{\frac{3}{2}}}$.

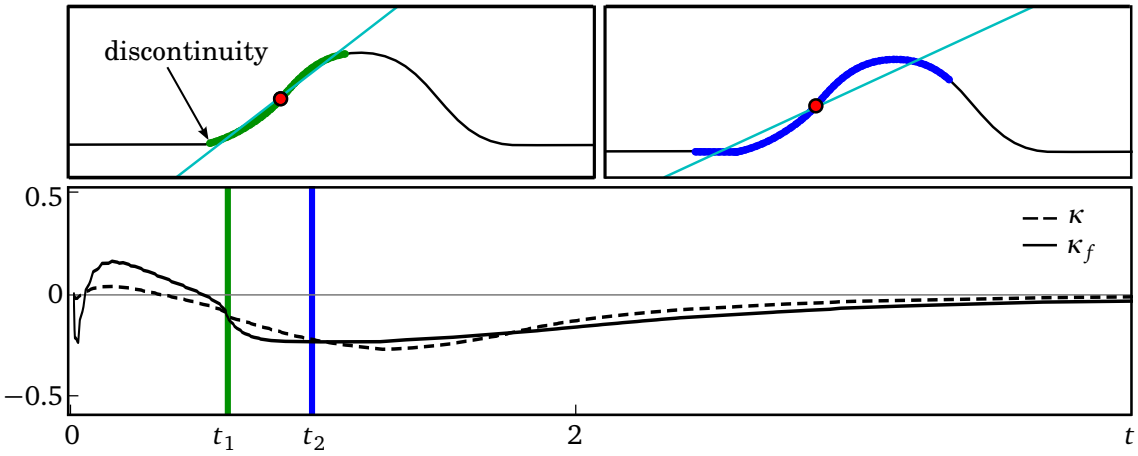


Figure 4.7: **Estimation of curvature on a 2D curve.** Top: fitted plane and the associated neighborhood used to compute curvature at scales t_1 and t_2. Bottom: comparison of the curvature profiles between the two-step fitting (black curve) and our method (dashed curve), for a point close to an inflexion point, by using a constant weighting function.

Figure 4.7 shows curvature profiles for a point $\mathbf{p}$ close to an inflexion point. At small scales (ranging from 0 to t_1), we can expect that the curvature values are close to 0 due to smooth variations of curvature in space. We can see that our descriptor gives a better result by producing small and stable values in this scale interval, probably because it takes into account the normal during the fitting procedure. The variation of κ_f can also be explained by various orientation changes for the fitted plane in the interval. At t_1, the apparition of a discontinuity in the neighborhood produces a jump in κ_f, resulting in additional instabilities in the fitting of the quadric. Note that we use the same weighing function for the plane, the quadric, and the hyper-sphere fitting. At coarse scales (after t_2), both methods produce similar values, although κ converges more quickly to 0, corresponding to a hypothetical global plane.

This simple example shows that two-step quadric fitting cannot be used to compute curvature at multiple scales for inter-scale comparison, because there is no warranty that a variation of measured curvature corresponds to a transition between different features at multiple scales on the geometry. Indeed, quadric fitting is not robust enough and generates false variations on near-planar areas. A last limitation is that it is not possible to directly use the quadric parameters as descriptor, in contrast to our method, because it is expressed in a basis that is potentially varying in scale and space. At the opposite, our method is expressed in a local basis centered around the evaluation point.

Discrete Curvature. In a second test, we compare κ to Discrete Curvature (DC), which uses connectivity information to compute mean curvature[1]. As input, we use a mesh of the slightly noisy Isis model that was generated using photogrammetry, and that contains macro structures representing the grain of the rock (see Figure 4.8, left).

DC and κ both detect similar coarse and fine features, and even sharp edges and smooth regions. However, our method produces a smooth curvature function on the surface, whereas DC produces some artifacts because of the variation of the connectivity information (valence, holes). Note also the sharp curvature changes on smooth features (at the Isis' chin). Different results would be achieved by using different smoothing operators and mesh-based curvature estimators. In any case, our approach is much easier to use since it does not involve a separated smoothing operator, since it does not require any connectivity information.

DoG and Covariance Analysis In a third test, we compare κ to DoG and Covariance Analysis (CA) by comparing the scalar-valued scale-space obtained using the three methods. All our visualizations of the scale-space for 2D curves use the arc-length ℓ and the scale t for the horizontal and vertical axis, respectively. Figure 4.9 (a) compares our approach to DoG and CA for the same smooth sinus-like curve as Figure 4.3. In this example, all methods manage to identify the two signals, although CA cannot distinguish between concavities and convexities, and DoG fails to capture important variations at curve borders. A more complex example (Figure 4.9 (b)) shows that the different methods exhibit more different behaviors. In particular, DoG does not reach convergence at large scales: it keeps on introducing meaningless structures. In contrast, our curvature κ produces a much smoother and more coherent scale-space, even though it does not rely on input point connectivity.

[1] We use Meshlab 1.3.0 [Cig12] and the plugin Discrete Curvatures.

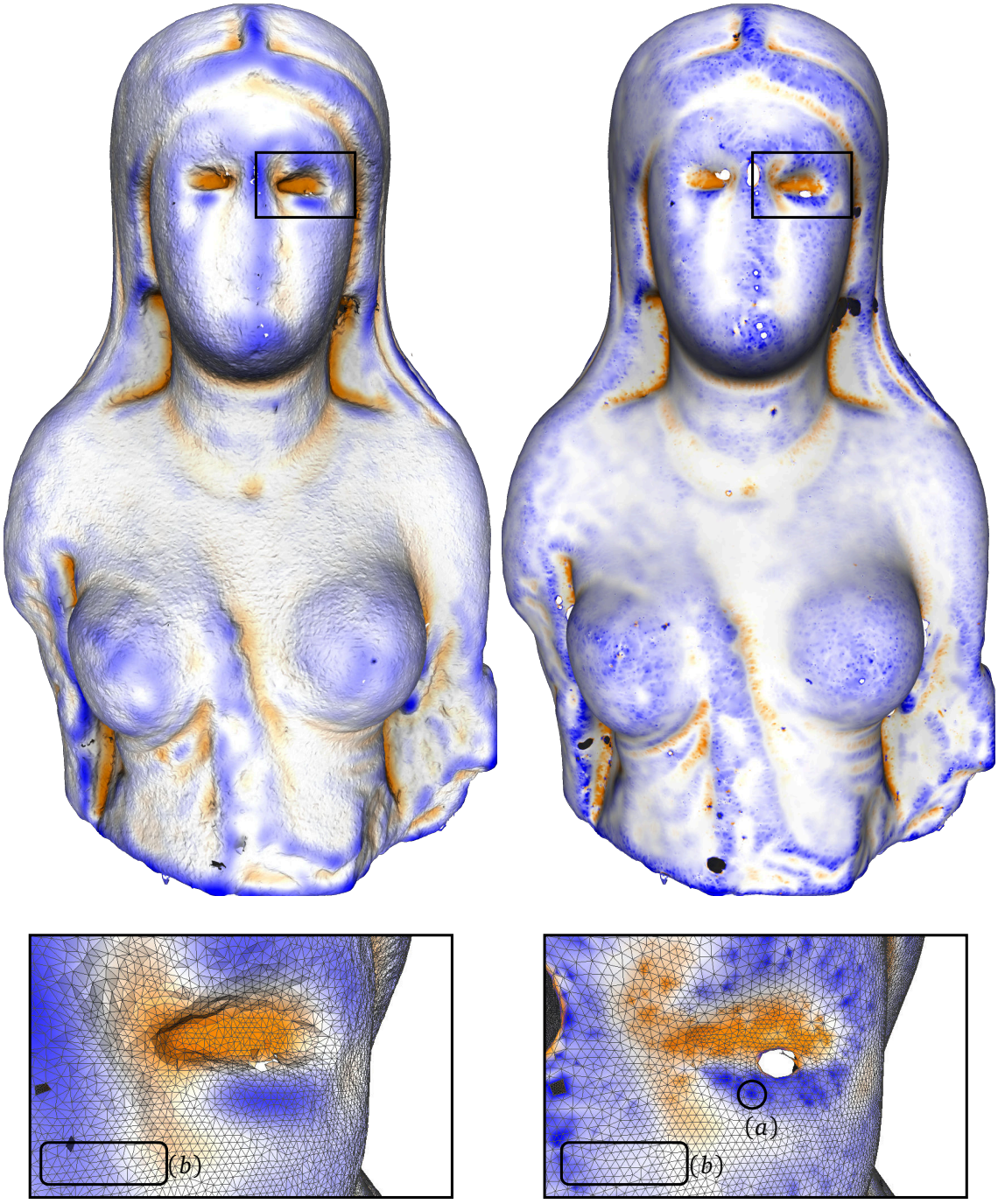


Figure 4.8: **Estimation of curvature on an acquired 3D surface at medium scale.** Left: 3D object generated using photogrammetry, with noise and important macro-structures, colored using κ computed at medium scale ($t = 15$). Right: Smoothed version (Laplacian smoothing with 15 iterations), colored using Discrete Curvature (DC). Both methods detect similar features (e.g. sharp edges, anisotropy), while DC produces artifacts depending on vertex valence (a) and non-smooth transitions between planar and curved areas (b).

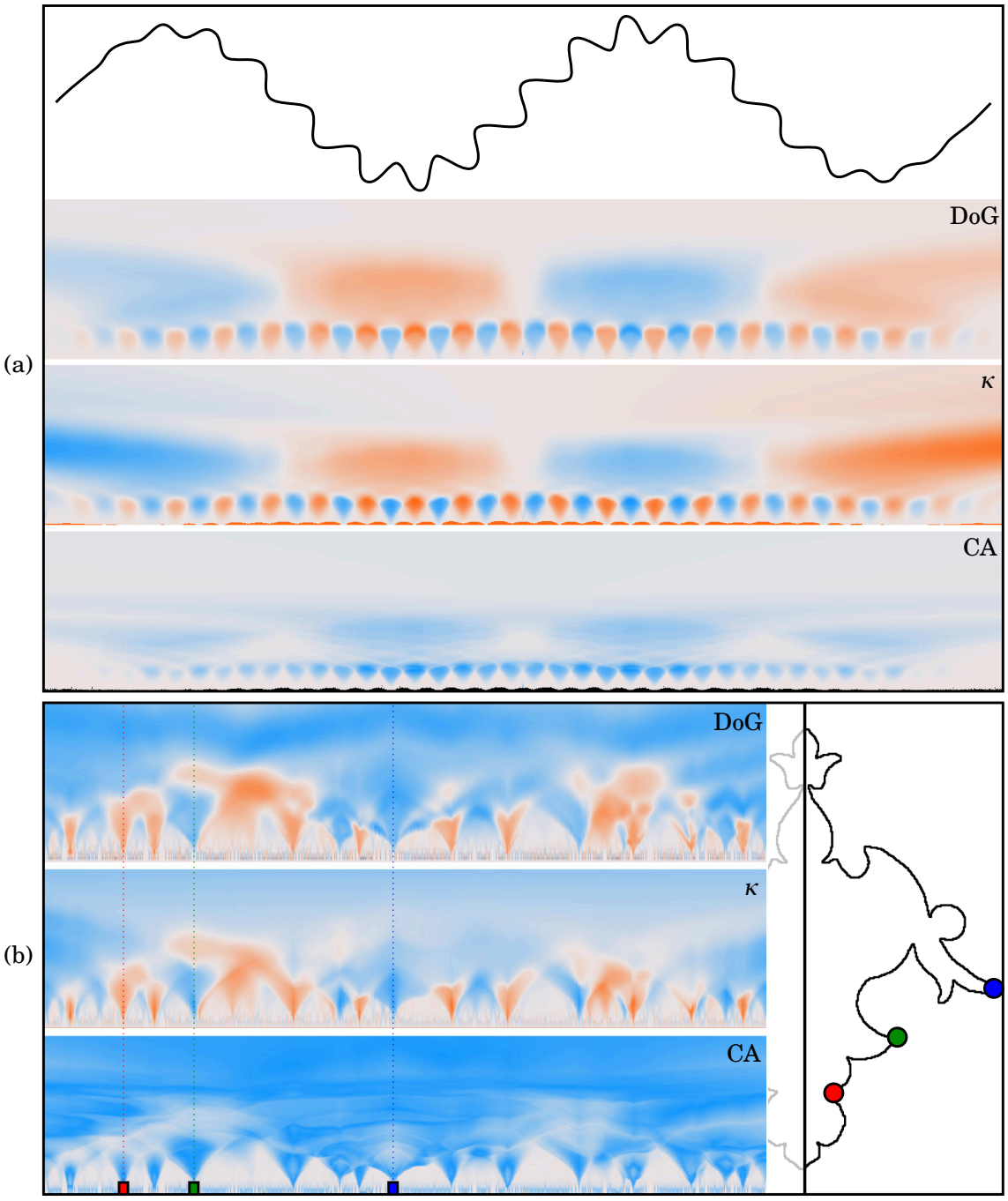


Figure 4.9: **Curvature scale-spaces**. From top to bottom: DoG, ours, and CA. In (a), we show curvatures for the sinus-based function shown on top. In (b), we show curvatures for the symmetric curve shown on the right. Compared to DoG, our approach exhibits less noise at small scales and a better convergence at large scales. CA does not make any difference between convex and concave regions, which are merged together.

4.2.2 Robustness to noise

In this section we first compare κ to CA and DoG on noisy 2D curves. Then, we show the stability of our descriptor and some similarity queries on noisy 3D objects.

Curvature estimation Figure 4.10 compares our approach to DoG and CA for a smooth 2D curve that is locally corrupted by noise only on positions (A), both on normals and positions (B), and on normals only (C). We display κ and DoG 0-crossings, as well as the 0-crossings of CA variation (measured along scale) in order to highlight the impact of noise. Considering CA, we observe that the position noise produces 0-crossings at small scales

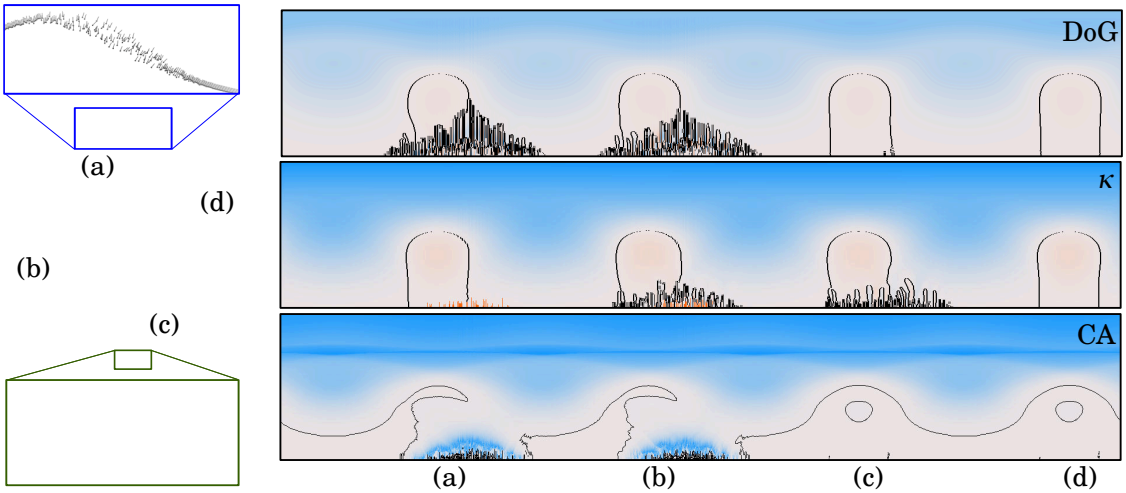


Figure 4.10: **Robustness to noise: 2D curve.** Left: Input 2D curve corrupted with noise on positions (A), both positions and normals (B), and normals only (C). The source code to generate this curve is given in Appendix B. Right: Scale-spaces filled with Difference of Gaussian, κ, and Covariance Analysis (from top to bottom). For DOG and κ, the black curves represent the 0-crossing, and for CA, they represent the 0-crossing of its variation in scale.

and influences the descriptor at large scales. The DoG is also sensitive to position noise and produces very noisy curvatures from fine to medium scales, but it well detects coarse features. Both methods are insensitive to normal noise because the normals are not taken into account in the computation. In contrast, our method is not very sensitive to position noise, while normal noise has an influence at small scales. The small orange areas at fine scales (visible with position noise) correspond to undefined fitting due to an insufficient number of points in the neighborhood.

κ is robust to position noise when the normals are well defined. In practical cases, this can be achieved by using a robust normal estimation. On the contrary, both DoG and CA are sensitive to position noise and require a smoothing step to deal with noisy acquired data.

Descriptor stability In Figure 4.11, we show an example of the robustness to noise of our descriptor for a 3D surface. We generate a random displacement on the vertex positions and recompute the normal vectors (we use a random noise of a magnitude of 0.2% of the object size[1]), and then we compare the results with the original surface. The curvature is

[1]We use Meshlab 1.3.0 [Cig12] and the plugin Random vertex displacement.

displayed as vertex color for a small scale ($t = 1\%$ of the object size): we observe that the results are similar in both cases, and that the fine features are well characterized. A zoomed area focuses on a flat anisotropic part that is well captured in both cases. Considering the descriptor profiles, we can see that the three components present some variations at fine scales due to noise, but produce quite similar profiles at medium and large scales.

These two last examples show that noise at small scales has only a slight impact on the GLS descriptor at large scales, even with normal perturbation.

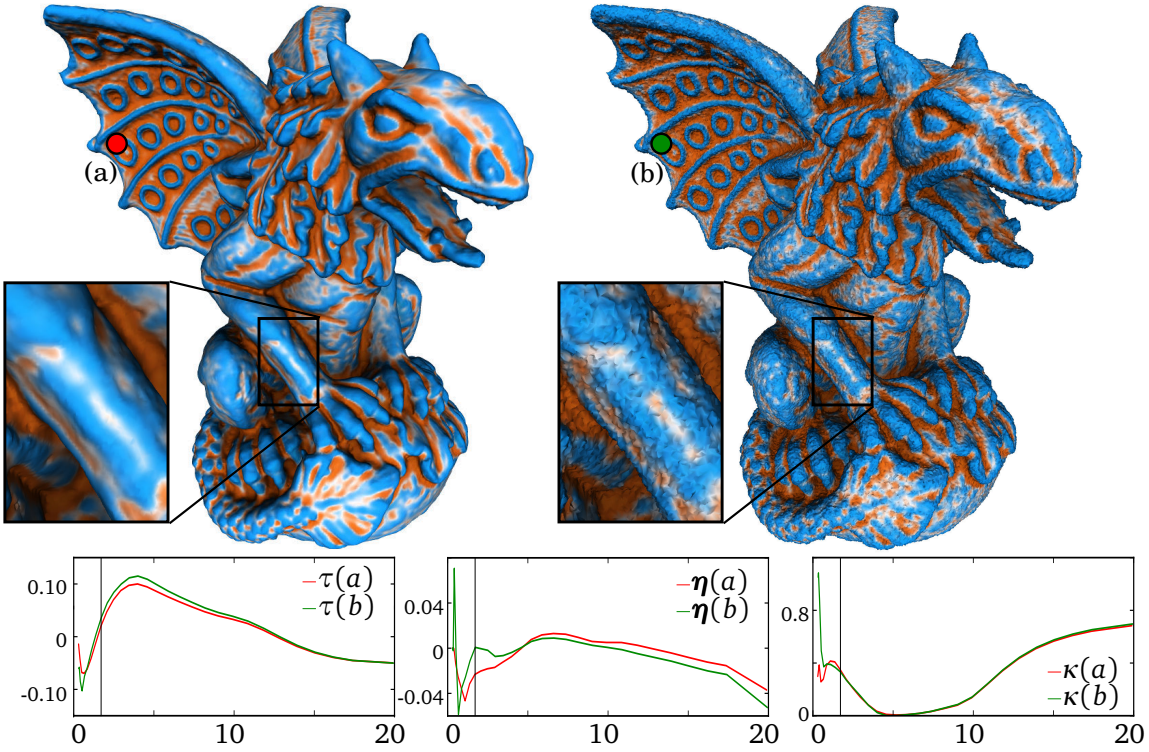


Figure 4.11: **Robustness to noise: 3D surface.** Top left: original Gargoyle mesh. Top right: corrupted version: application of random vertex displacement of magnitude 0.1% of the bounding box, with normal re-estimation. The color code represents the curvatures κ at scale $t = 1.7$. Bottom: the descriptor profiles for points (a) (clean) and (b) (noisy). For $\boldsymbol{\eta}$, we show the variation of the dot product with the original normal.

4.3 Comparison with previous work: GLS Analysis

In this section, we discuss the benefit of our scale-space analysis and compare it to other approaches, such as 0-crossing tracking (ZCT) [Lin94] and SIFT point extraction [Low99]. As explained in the previous work chapter, ZCT is very efficient with 2D curves, but it becomes unstable in higher dimensions. Consequently, and for the sake of a fair comparison, we compare our method on 2D curves. Concerning SIFT, we extract pertinent points as the extrema of an 8-neighborhood in scale-space. We compare the methods in terms of robustness to input variation. We also propose a study of the relationship between our geometric variation v and curvature 0-crossings.

4.3.1 Stability to input variation

In Figure 4.12, we compare our analysis to ZCT and SIFT at a curve where we introduced input changes of different sizes in order to point out an important limitation of previous methods.

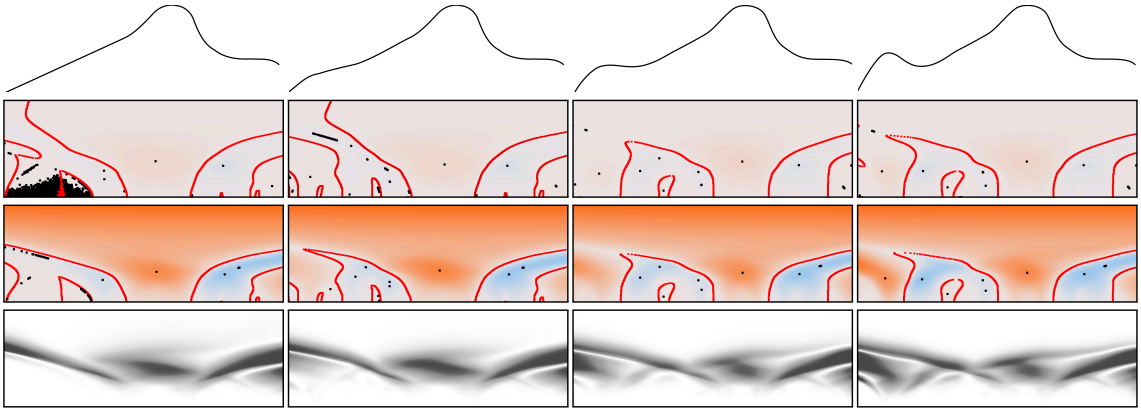


Figure 4.12: **Stability to input variation.** From top to bottom. First row: a convex bump with a gradually varying magnitude is added to the left side of a simple open curve. Second row: DoG scale-space with 0-crossings (in red) and SIFT points (in black). Third row: κ scale-space with 0-crossings (in red) and SIFT points (in black). Bottom row: geometric variation ν scale-space.

First, κ produces a more stable scale-space for ZCT than DoG. Indeed, lots of instabilities are due to vertical 0-crossing profiles that reach the top of the scale-space or annihilate at medium scales. This behavior is well attenuated because of the natural convergence of our descriptor to a global sphere at coarse scales. Furthermore, there is no reason to identify a precise location as a representative of such a large scale, because this scale corresponds to an extremely smoothed version of the shape. SIFT extraction also benefits from this behavior and does not produce invalid points at coarse scales in contrast to DoG scale-space analysis. It is also more stable at small and medium scales: in our example, the extracted set differs less from one input to another (concerning κ). A last advantage of using κ is that it reduces oscillations at planar areas that would generate a large set of invalid SIFT points (we refer to Section 4.2.1 for more details on this topic).

Let us now consider the behavior of ZCT and SIFT in comparison with ν. When the input curve is modified so that one of its bumps is slightly more prominent, then the corresponding set of 0-crossings and SIFT points changes abruptly, suggesting that the structure of the curve has changed in similar respects. On the contrary, our geometric variation scale-space evolves continuously to reflect the more subtle structure change implied by this slight modification. In particular, intermediate scales around the bump region are no longer considered persistent when the more prominent bump appears. Indeed, increasing the amplitude of the bump has the effect of "breaking the structure" of the slope on the left side of the curve.

4.3.2 Relationship to 0-crossing

Figure 4.13 shows a layering of 0-crossings on top of our geometric variation scale-space. There are three types of differential invariants: 0-crossings of κ (in red) its spatial derivative κ' (in green), and SIFT points (in black). For convenience, we denote the corresponding scale-space points C and C', respectively. First, note that all annihilation events for both C and C' occur in regions of low geometric variation, but there are other such regions which are never reached by either C or C'. For example, we show in blue in Figure 4.13 a small convexity and a planar area at coarser scales that are not detected by SIFT. This suggests that the tracking of extrema actually leads to a *subset* of pertinent scale-space locations according to ν. It is also interesting to note that the points in C (respectively C') seem to be attracted towards regions of high (respectively low) variation, a tendency that we plan to study in future work.

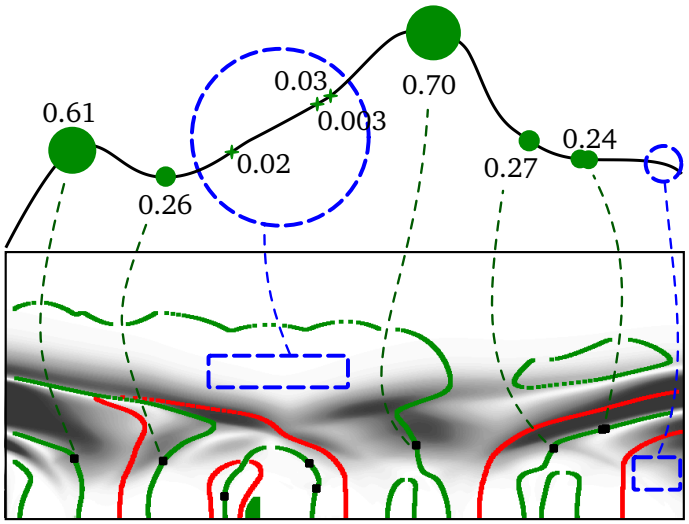


Figure 4.13: **Relationship to 0-crossings.** The geometric variation ν for the last curve in Figure 4.12 is displayed with 0-crossings of κ (in red), its curvilinear derivative κ' (in green), and its SIFT points (in black) layed on top. For each extracted point, we show a circle on top of the input curve with a radius corresponding to the associated value of ν; a cross represents the smallest points. In blue, we represent some features and the associated scale-space locations that are not detected by SIFT.

Our second observation concerns the SIFT points. As they are extracted as pairs of position and scale, they can be associated to a geometric variation value. Figure 4.13(top) shows these points as circles with a radius proportional to ν. A human observer may see that the relevance of each feature point seems to be coherent with the values of ν. We also remark that, most of the time, a pertinent scale associated to a given location is preceded by low values of ν followed by high values of ν, which characterizes the transition from one feature to another. We plan to analyze this behavior more in-depth in future work in order to extract pertinent scales (see Section 4.4.3).

4.4 Results and perspectives

In this section, we present concrete application scenarios of our GLS framework as well as qualitative and quantitative results. In particular, we first present performance and implementation details. Then, we present some case-studies focusing on the characterization of complex features (anisotropy and saddle points) and on the extraction of pertinent scales. For each of these topics, we present previous work and their limitations that motivated us to propose a more robust analysis. Finally, we present preliminary studies and results of our framework for further applications, such as expressive rendering and surface reconstruction.

4.4.1 Performances and implementation details

Implementation details In our system, a frequent computation is the neighborhood collection for a given evaluation point, since the fitting at scale t is performed by collecting all the neighbors within the distance t of the current point. In order to speed-up the neighbor queries, we compute a kd-tree on the CPU in a preprocess. We then transfer and use it on the GPU for an efficient parallel computation of the descriptor at multiple locations. This makes the complexity of our algorithm quadratic with respect to t for a 3D surface in worst cases. We implemented our approach both on the CPU (with multi-threading on all 8 cores of an Intel I7 3.40Ghz) and on the GPU (using CUDA on an NVidia GTX 580). As expected, the GPU implementation outperforms the CPU version. For example, in Figure 4.19(c), the analysis takes only 6.3s on the GPU versus 221s on the CPU. We show in Table 4.14 the reported timings corresponding to our CUDA implementation for the models used in this chapter, which include the computation time for both the descriptor and its scale derivative. Since we use a maximum scale size that depends on the bounding box diagonal length, the timings are strongly dependent on the object density. Finally, transferring the descriptor values from the GPU to the CPU takes a negligible time (0.015329 seconds for the Isis headpiece model).

Object	Figure(s)	Nb vert.	Nb scales	Time (s)
Armadillo	4.16, 4.19	173k	20	6.3
Isis bust 1	1.1, 1.2, 4.15, 4.22	246k	20	15.3
Gargoyle	4.11, 4.20	250k	50	16.77
Golfball	4.4, 4.22	491k	20	16.21
Headstone	4.18	554k	20	27.66
Torus	4.6	583k	20	41.82
Isis bust 2	3.3, 4.8	861k	20	61.4
Isis headpiece	4.17	2698k	20	822

Figure 4.14: **Computation timings.** We computed the descriptor value for all points at multiple scales on the GPU. The fine scales was chosen according to the point set density (in order to select around 10 neighbors), and the coarse scale is 33% of the object's bounding box diagonal.

The actual bottleneck of our approach is the neighborhood search at successive scales. We believe that a huge speed-up could be achieved by iteratively smoothing the object and/or using an adaptive sampling of the input point cloud for the computation of the larger scales [PKG06].

4.4.2 Description of complex shapes

As explained in Section 4.1, our description is based on the fitting of an algebraic hyper-sphere, which is a purely isotropic primitive. In this section, we show that this description can be used to detect anisotropic features and saddle-like points by taking advantage of multiple scales. Moreover, we see that this description can be used for similarities queries.

Anisotropy A first way to detect anisotropy is to understand anisotropic features as a set of isotropic descriptors disposed along paths on the surface. For an example, see the similarity query for a point on an anisotropic feature of an acquired 3D object in Figure 4.15. In the top left image, we show the curvature value that gives an intuition of the object's shape. In the middle and right images, we show the result of the similarity query, and we emphasize the anisotropic feature and the underlying path that characterizes it. At the bottom, we show three fitted hyper-spheres that stay similar all along the path. The small variation of κ in the middle image (the bigger sphere) is due to a local stretch of the feature.

This example shows that our descriptor can be used to detect anisotropic features even though the descriptor itself is isotropic.

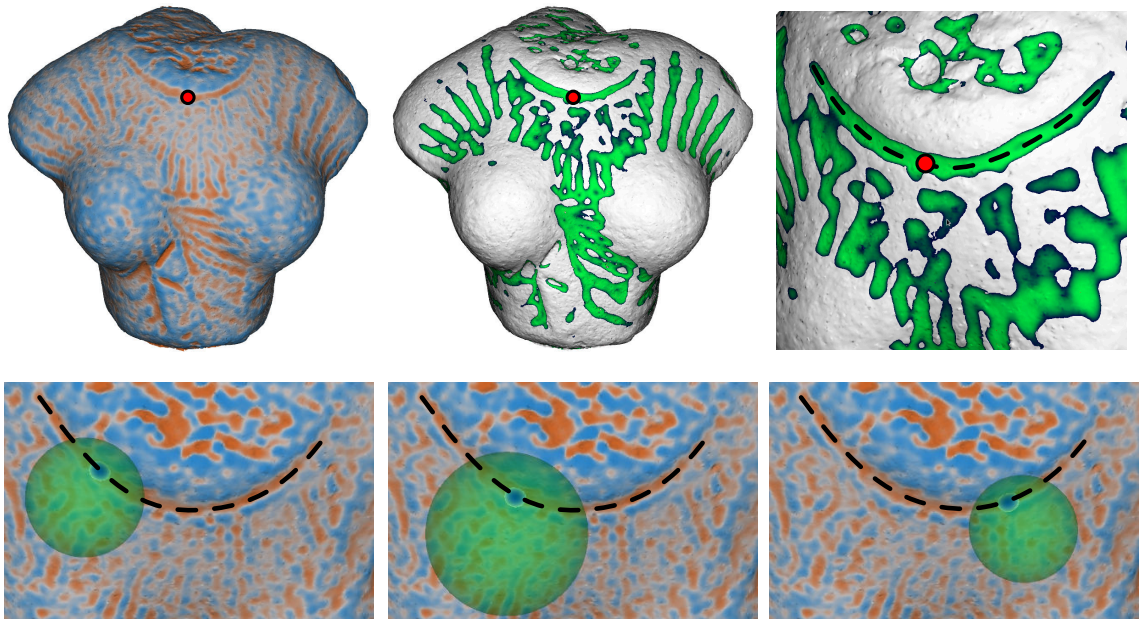


Figure 4.15: **Anisotropy detection with the isotropic descriptor.** Top: similarity query for the selected vertex (in red). From left to right: curvature, query result, and zoom on the anisotropic feature. The dashed curve represents the path that characterizes the feature. Bottom: isotropic primitives that describe the feature.

Saddle points Another type of feature that our descriptor cannot capture directly by itself are saddle and saddle-like points. These points have principal curvatures with opposite sign corresponding to a negative Gaussian curvature. Our purely isotropic descriptor cannot characterize these points, since the mean curvature is not sufficient to disambiguate these saddle points. For example, a *perfect* saddle point has a mean curvature value equal

to 0, like a planar surface. The fitness parameter cannot be used neither in this case, because it is also purely isotropic by definition.

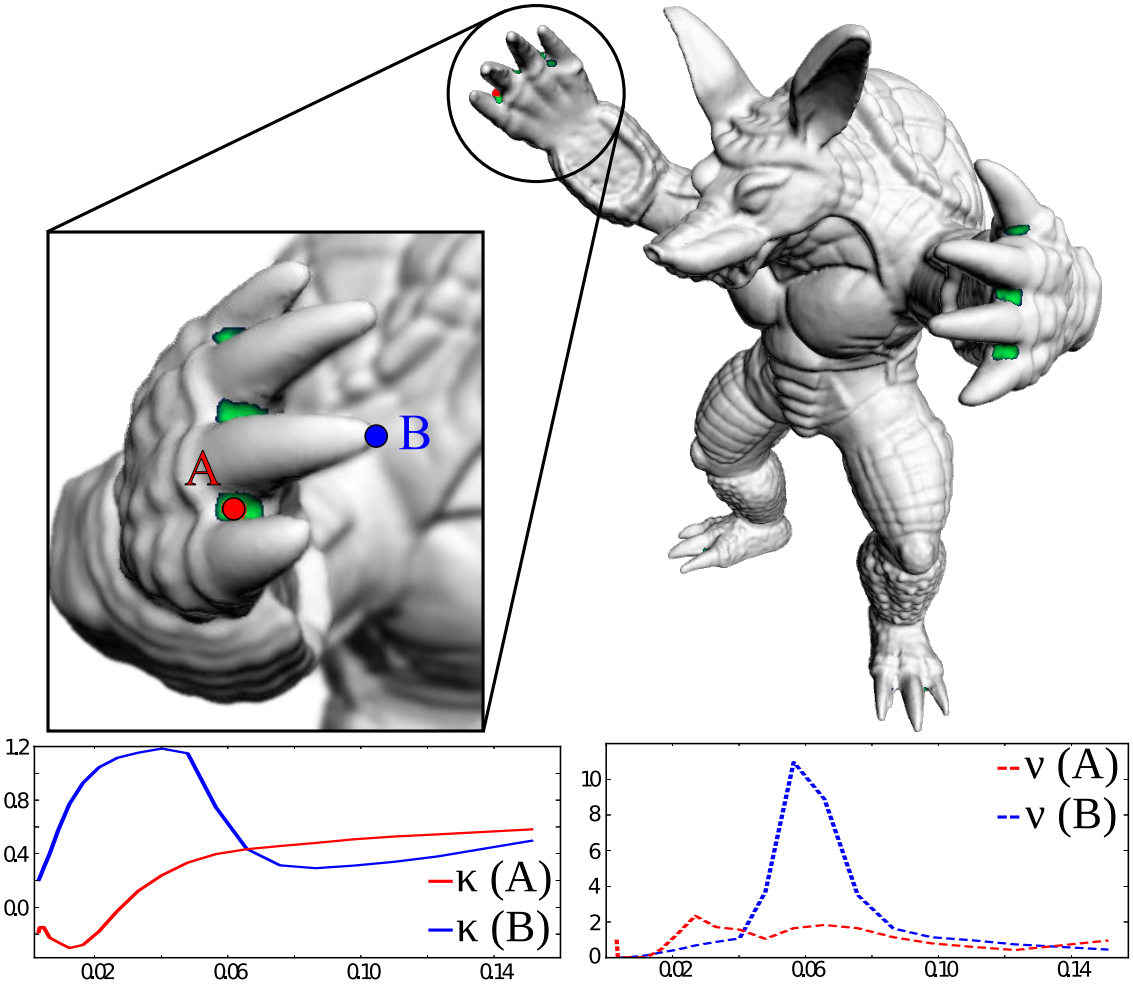


Figure 4.16: **Saddle-like ambiguity** Top: given a saddle-like point (A-red), similar points are identified and shown in green. Bottom: given the saddle point A and a convexity point (B-blue), we show the profiles of the curvature κ (left), and the geometric variation ν (right).

However, in practice, it is rare to have *perfect* saddle points on 3D objects (with exactly opposed principal curvatures), especially with acquired data, since the probability to have a point sample exactly at the saddle center is very small. For example, in Figure 4.16, we show some results of a similarity query with our descriptor for a saddle-like point on the Armadillo model. It can be observed that similar saddle-like points are correctly selected without any false positive. This is because the multi-scale profile of this point characterizes the properties of the surrounding shape. In comparison with the saddle-like point, we also show the profile of κ and ν for a convex feature at a nearby point B located on the extremity of finger. On the one hand, at coarse scales, both descriptors are similar ($t > 0.1$) and refer to the hand of the Armadillo. On the other hand, at small scales, the saddle-like point produces a specific profile. This specific profile was used to identify all the similar saddle-like points (shown in green in the figure).

Perspectives The two previous examples have shown that our hyper-sphere descriptor can be used to disambiguate different complex shapes. Of course, we have to define and perform more formal studies besides these examples. More precisely, we want to address more in-depth the analysis of the descriptor space variation.

As shown in Figure 4.15, anisotropic features may be described by considering successive unique descriptors disposed along a path. We propose to analyze the variation of our descriptor in space in order to be able to detect such paths and to characterize anisotropic features. In the same way that we measure variations when scale varies, we propose to differentiate our descriptor in space (see the Appendix A for more details). The first step is to differentiate our weighting function (defined in Equation 4.2) using partial spatial derivatives

$$\frac{\delta w_i}{\delta x}(t) = \frac{-4x(\overline{\mathbf{q}}_i)\left(\frac{x(\overline{\mathbf{q}}_i)^2-1}{t}\right)}{t}, \tag{4.12}$$

where $x(\overline{\mathbf{q}}_i)$ is the x coordinate of the current point $\mathbf{q}_i$ expressed in the centered basis. Using a similar definition for the other coordinates, we differentiate each parameter and thus build the 3x5 Jacobian matrix

$$\hat{\mathbf{J}}_i(t) = \begin{bmatrix} \alpha\frac{\delta\tau}{\delta x} & \alpha\frac{\delta\tau}{\delta y} & \alpha\frac{\delta\tau}{\delta z} \\ \beta\frac{\delta\boldsymbol{\eta}}{\delta x}t & \beta\frac{\delta\boldsymbol{\eta}}{\delta y}t & \beta\frac{\delta\boldsymbol{\eta}}{\delta z}t \\ \gamma\frac{\delta\kappa}{\delta x}t^2 & \gamma\frac{\delta\kappa}{\delta y}t^2 & \gamma\frac{\delta\kappa}{\delta z}t^2 \end{bmatrix}, \tag{4.13}$$

where α, β and γ are user-specified weighting factors. Note that in contrast to the similarity measure that discards the parameter $\boldsymbol{\eta}$ (see Equation 4.10), the differentiation is performed in the same local frame, which leaves the comparison with $\boldsymbol{\eta}$ valid.

The meaning of these derivatives is one of the first topics that we want to address in future work. Indeed, we differentiate in space τ, η, and κ that characterize the surface at, respectively, first, second, and third orders. As said previously, τ represents the distance from the evaluation point to the fitted surface. When the evaluation location is moved, τ can be seen as an approximation of a height function to the implicit surface, which plays the role of a smoothed base surface. Hence, $\delta\tau$ seems to represent the gradient of this height function on the surface . The value $\delta\eta$ measures the variation of a fitted normal vector, representing an approximation of the normal curvature. The last parameter $\delta\kappa$ measures an approximation of the mean curvature variation, an information that is usually used in scale-space analysis [Lin94], or for screen-space curve extraction [VVC+11]. The third order differentiation of the surface is also studied in [KST08] to extract demarcating curves on the analyzed object. It is also interesting to see that our formulation provides three different approximations of the mean curvature: κ, the mean of the normal curvatures $\delta\eta$, and the second-order differentiation of the APSS implicit surface [GG07]. It would be interesting to compare the associated properties with respect to other mean curvature estimation techniques [FJ89, ZBVH09].

We also propose to use our spatial differentiation to build a hyper-sphere variation tensor $\mathbf{T}_i(t)$ that measures the variations of our descriptor on the surface as follows:

$$\mathbf{T}_i(t) = \hat{\mathbf{J}}_i(t)^T\hat{\mathbf{J}}_i(t). \tag{4.14}$$

We project this tensor on the surface and then perform an eigen-decomposition. This defines an oriented surface flow that associates to each position on the surface a vector in the

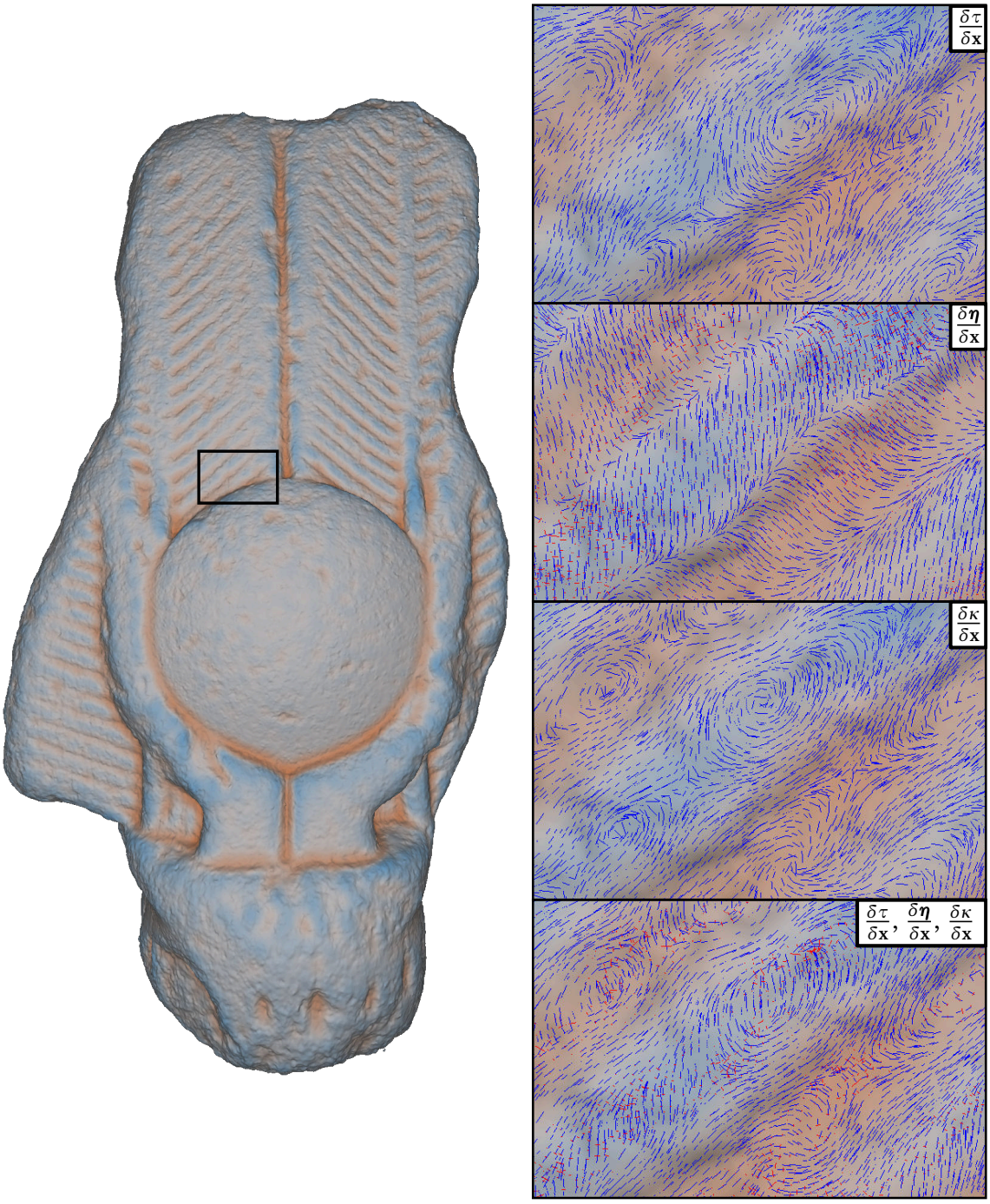


Figure 4.17: **Hyper-sphere variation surface flow.** Left: the Isis headpiece 3D object consisting of 2.7 millions points. Right: zoomed area, with flows generated from the hyper-sphere variation tensor (see Eq. 4.14) with first (blue) and second (red) eigenvectors, the length corresponding to the associated eigenvalues. From top to bottom: we use different values of $[\alpha, \beta, \gamma]$ in order to measure the contribution of each spatial derivatives $\frac{\delta\tau}{\delta\mathbf{x}}$, $\frac{\delta\boldsymbol{\eta}}{\delta\mathbf{x}}$, $\frac{\delta\kappa}{\delta\mathbf{x}}$ and their uniform combination (respectively $[1,0,0]$, $[0,1,0]$, $[0,0,1]$ and $[1,1,1]$).

direction of the strongest variation (first eigenvector of $\mathbf{T}_i(t)$), with a length corresponding to the associated eigenvalue. By adjusting the factors α, β, and γ in Equation 4.13, it is possible to generate a flow for each parameter individually, or for a combination of these parameters (the blue lines in Figure 4.17). We also show the second eigenvector in this figure (in red), which indicates local ambiguities when both eigenvalues are similar. We can see that each parameter produces quite different flows, and that their combination should be investigated more deeply.

The second topic that we plan to address is the definition of the family of shapes that can be characterized by our framework. Indeed, we have shown that different shapes produce quite different *signatures* when we consider both the descriptor itself and its variations (in scale and in space). We have also shown in our preliminary studies that non-spherical shapes can be disambiguated by our framework. In the future, we want to analyze these *signatures* formally and experimentally. This analysis is necessary to better understand the robustness properties of our descriptor for the analysis of complex shapes, noise, and deformations.

Finally, we also plan to work on the potential applications that might benefit from our analysis. We believe that the versatility of our approach is clearly an advantage in various application cases. For example, in Cultural Heritage, our multi-scale similarity measure may be used to detect engraved patterns on surfaces given the geometry of a potential engraving tool, while the associated flow characterizes the underlying engraved drawing, as shown in Figure 4.18. We also propose to use our descriptor in the context of shape retrieval. Indeed, the approximation of principal curvatures using the spatial derivatives of η could be used to compute approximations of the Gaussian Curvature, which is an intrinsic shape descriptor. This measure could be used directly in classical shape retrieval methods [TCF09, BBGO11]. In this case, we have to take care that using an Euclidean neighborhood collection implies sensitivity to non-rigid transformations. In contrast, isometry invariance requires to define a geodesic distance for PSS [RDSK06] and its derivatives in order to compute $\delta\tau$.

4.4.3 Pertinence detection

In order to deal with complex manifolds, many geometry processing applications rely on a preliminary feature extraction step that identifies a subset of salient points to consider for further processing.

Continuous pertinence detection in space Instead of relying on individual points, we propose the continuous feature function $f_v(\mathbf{p}) = \int \bar{v}(\mathbf{p}, t)dt$, where $\bar{v}$ is a smooth remapping of v that disregards too strong geometric variations (we use a *tanh* function to this end). The intuition behind this formula is that points for which f_v is small are subject to nearly no geometric variations across scales (e.g., $f = 0$ on a sphere), whereas points with a high f indicate that the geometric descriptor has important variations and thus might refer to pertinent points, as explained above.

This is illustrated on complex curves and surfaces in Figure 4.19 using a color gradient. Note how the most pronounced features (in red) may correspond to different concave and/or convex regions. This is desirable because there is often no *a priori* on the sign of the curvatures of features. Note also that regions with repetitive small details are not considered as important, whereas isolated or more prominent shape details emerge as key features of the object.

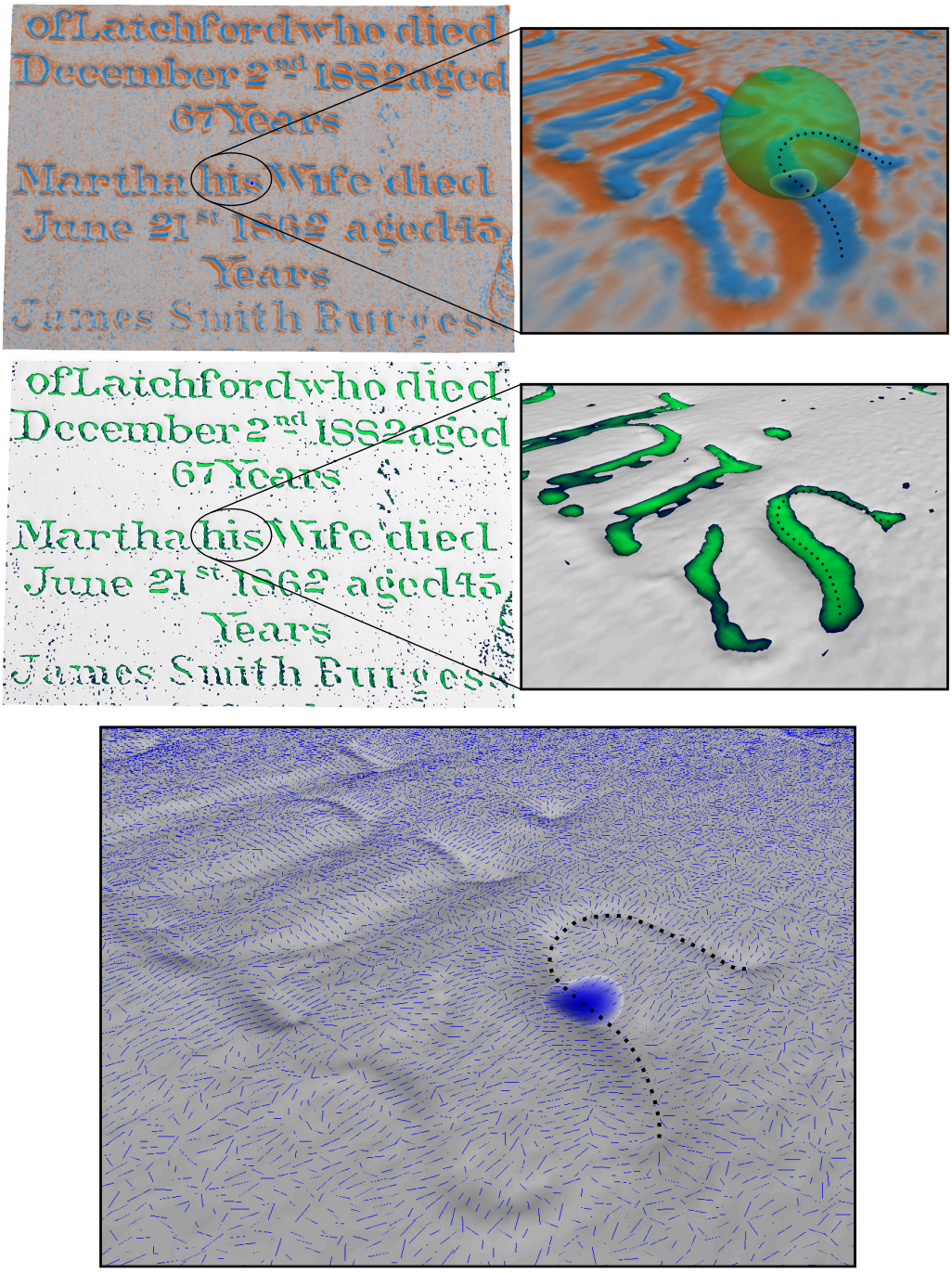
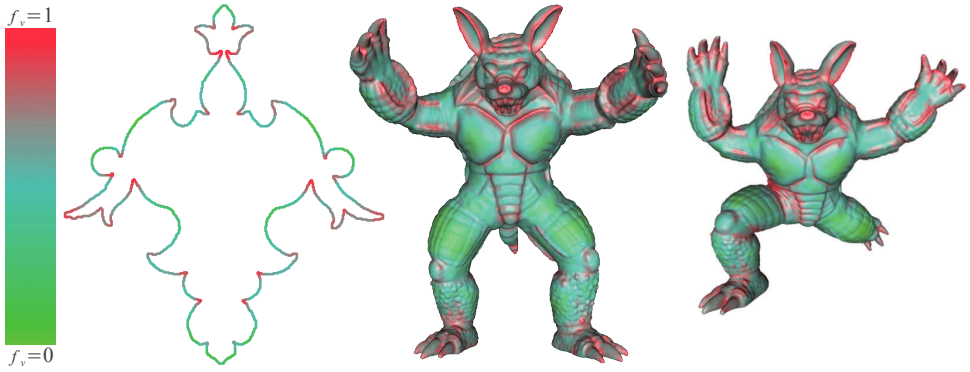


Figure 4.18: **Characterization of engraved text.** Top: the geometric descriptor characterizes the geometry of the potentially used engraving tool. Middle: result of the similarity detection - the geometric descriptor is coherent along the path. Bottom: the flow defined by $\delta\kappa$ is orthogonal to the shape of the engraved text. In all images, the path shown as dashed curve has been extracted manually.



(a) (b) 5k pts, 1000k scales, 3s (c) 173k pts, 20 scales, 6.3s

Figure 4.19: **Continuous features** are displayed using a color gradient (a), on a symmetric 2D closed curve (b), and on 2 variants of the Armadillo 3D object (c). Features of varying shapes and sizes (in red) are properly selected in all the examples.

Influence of noise As shown in Section 4.2.2, our descriptor produces robust profiles even in presence of noise. In Figure 4.20, we show a similar test with profiles of ν and f_ν. Unsurprisingly, ν is impacted only at small scales and stays stable at medium and coarse scales. On the contrary, the continuous pertinence variation is a bit disturbed, and some regions have low values despite the presence of noise: the red area becomes green in the figure, as can be seen at the wings. This means that the presence of noise reduces the total amount of descriptor variation when scale varies. This counterintuitive behavior can be explained by observing the ν profiles in Figure 4.20 (bottom row). We can observe a local maximum around the scale t_1 that corresponds to the transition between two successive features (in this example from the circular small pattern to the global wing shape). In case of noise, this transition between successive features is smoothed, which reduces the value of the local maximum, and as a consequence the quantity measured by our pertinence measure.

Perspectives This sensitivity to noise indicates that our continuous pertinence measure is not well designed to be sufficiently robust. We propose to still analyze the ν profile in order to detect stable intervals followed by a local maximum that characterizes the transition to a feature at coarser scales. This phenomenon seems to be a common rule: we can observe it in all profiles of ν of this document, as for example in Figures 4.3, 4.4, 4.16 and 4.20. The position of the major SIFT points in Figure 4.13 seems also to be coherent with this definition, because they are all followed by a local maximum of ν at coarser scales.

We propose to detect this phenomenon by convolving the multi-scale profile of ν with a Gaussian kernel, and we obtain $\bar{\nu}_\sigma(t) = G_\sigma \otimes \bar{\nu}(t)$, where $\bar{\nu}$ is a normalization function that maps ν into the range $[0:1]$. Then, we detect a pertinent scale as the maximum of

$$\mu(t) = (1 - \bar{\nu}_\sigma(t - \Delta_t)) \,.\, \bar{\nu}_\sigma(t + \Delta_t), \tag{4.15}$$

where Δ_t is an offset value. For example, we show this automatic scale detection for different points on the synthetic golfball object, and on the acquired Isis headpiece in Figure 4.22. In our first experiments, small values of σ (0.1 in the figure) produce strong maxima of $\mu(t)$ that are easily detectable automatically. The value of this maximum seems to be an interesting criterion for the classification of pertinent scales. In our example, the detected

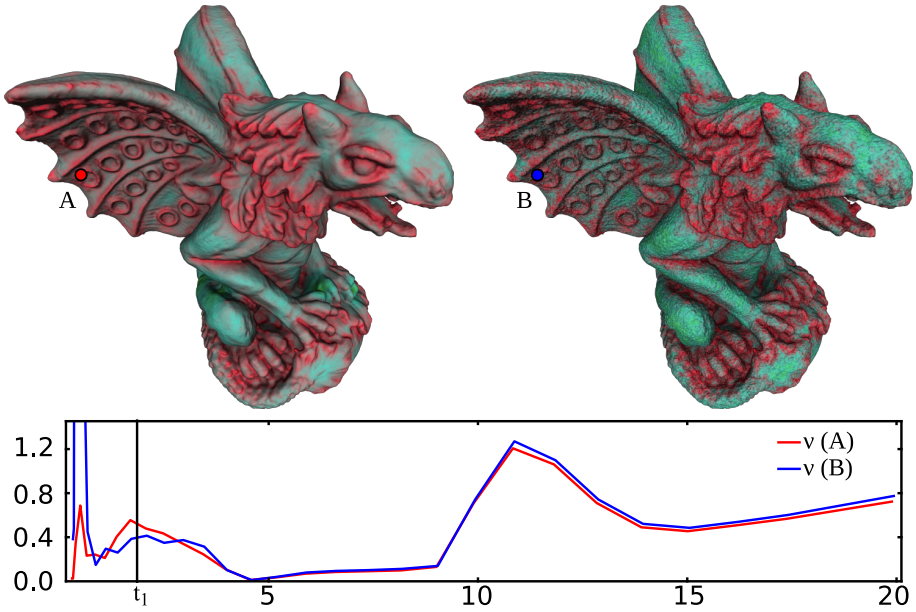


Figure 4.20: **Robustness to noise: ν.** Top: continuous feature detection. Bottom: multi-scale profile of ν for the points A (clean object, red curve) and B (noisy object, blue curve). Note the bump at t_1.

scales seem to be visually coherent with each feature's scale. However, we highlight the fact that, in practice, the detected scale is usually a little bit greater than the "real" scale of the feature. This is due to our weighting function that continuously introduces new features in the neighborhood when the scale varies. Hence, neighbor points that are close to the neighborhood edge should have a small weight and thus a small influence on the fit: our weighting function contains 90% of its energy in $[0 : 0.6]$ (see Figure 4.21). Therefore, the "real" pertinent scale should be in the interval $[0.6t_\mu : t_\mu]$, where t_μ is the extracted scale. Regarding our experiment, its exact location also clearly depends on the strongest variation between the two successive features: important transitions are detected earlier than small ones (by earlier we mean with smaller values of t). We also emphasis that our fitting procedure involves normal vectors, which incorporates information of their local neighborhood, and thus rely on a neighborhood size slightly larger than t.

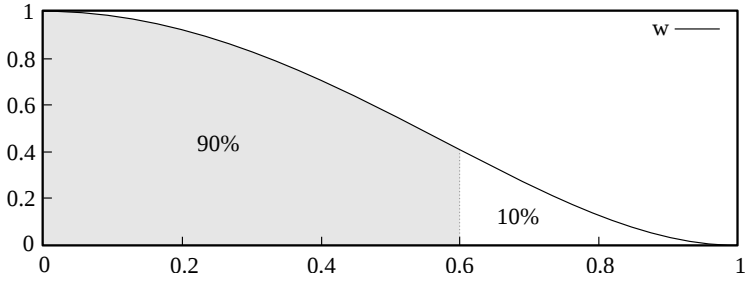


Figure 4.21: **Weighting function.** 90% of its energy is in the interval $[0 : 0.6]$.

A more formal analysis of these behaviors is an interesting future work. We also plan to analyze a special effect produced by our fit: the *drift*. Indeed, the fact that we use a fitting procedure allows our primitive to drift from the initial evaluation position to a feature in the neighborhood. This behavior is clearly an advantage of our framework, because it produces

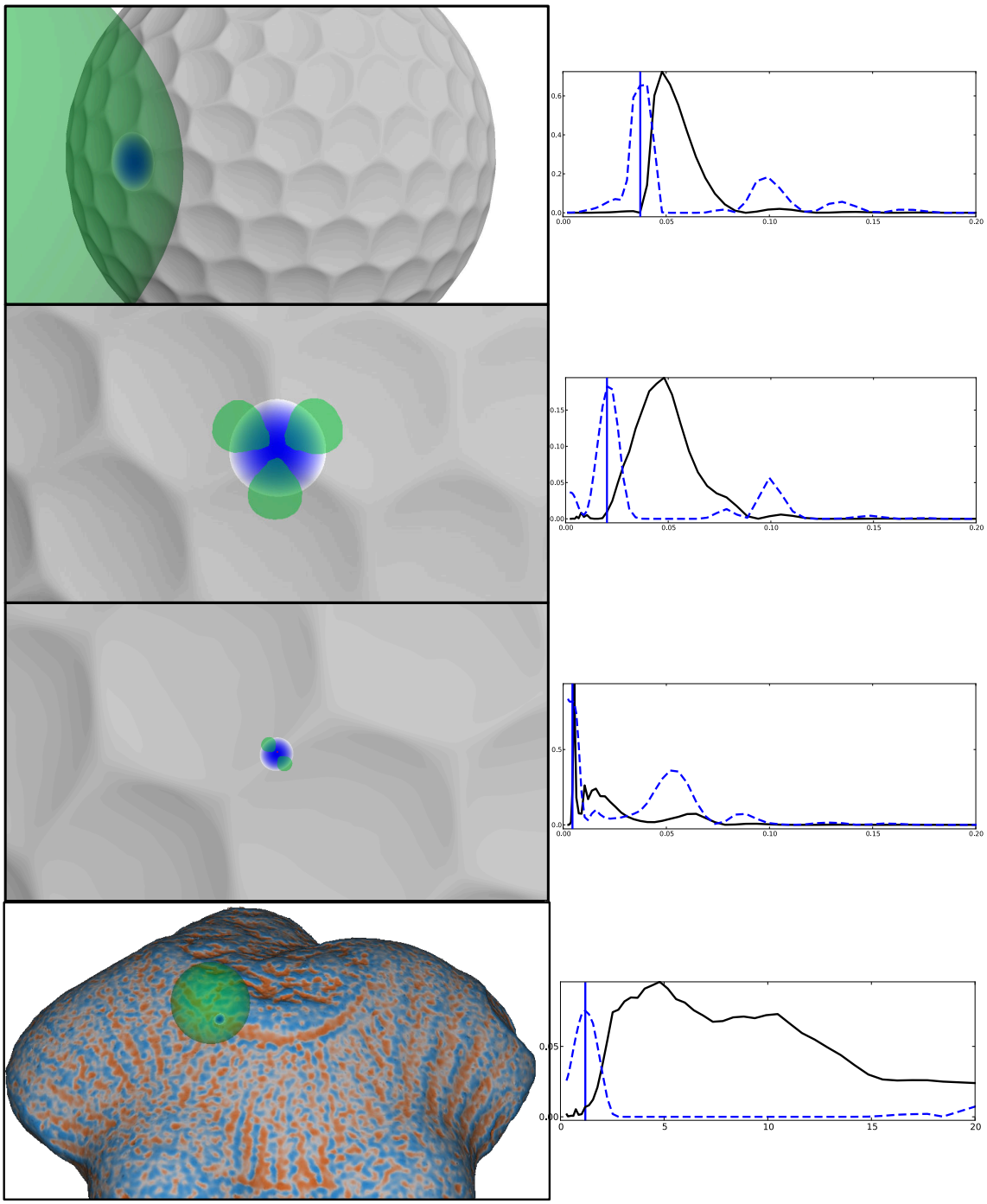


Figure 4.22: **Pertinent scale detection using μ.** Right: pertinent scale detection (vertical bar) using strongest maxima of the function μ (in blue) applied to the geometric variation v (in black). Left: we show on the 3D object the weighting function (in blue) and the fitted primitive (in green) for the detected scale.

similar descriptor values for points at different locations but referring to the same feature for a given scale interval. It can be seen as a snapping effect: the primitive snaps as long as possible to a feature when the evaluation position of the scale varies, and then the position changes radically to a new one. For scale variations, it produces the pattern detected in the Equation 4.15. Considering space variations, we can see them as sort of scale offsets, as illustrated in Figure 4.23: the farthest a point from the center of the feature, the earlier (in scale) it is detected: we call this effect a drift. We show an illustration of this drift in Figure 4.23 by adding the detected pertinent scales as an overlay on the ν scale-space (shown as a black curve). In future work, we plan to analyze its effect and the ways to take it into account in our multi-scale similarity metric.

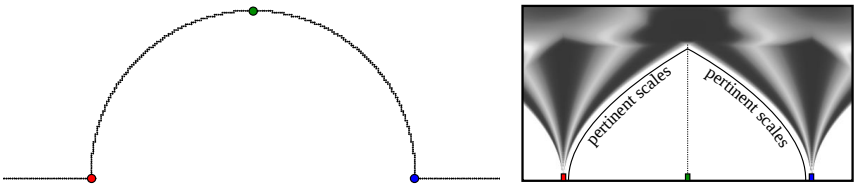


Figure 4.23: **Drift effect.** Left: input 2D curve with three points that characterize the boundaries (red and blue) and the center (green) of the feature. Right: ν scale-space. The black curve represents potential pertinent scales detected using Equation 4.15. These pertinent scales are drifting in scale-space.

4.4.4 Other applications

Expressive rendering Among the wide variety of methods proposed to render a 3D object, we can distinguish the expressive rendering approaches. The idea is to express information about the 3D scene through the rendering, such as the object geometric properties [Ver10].

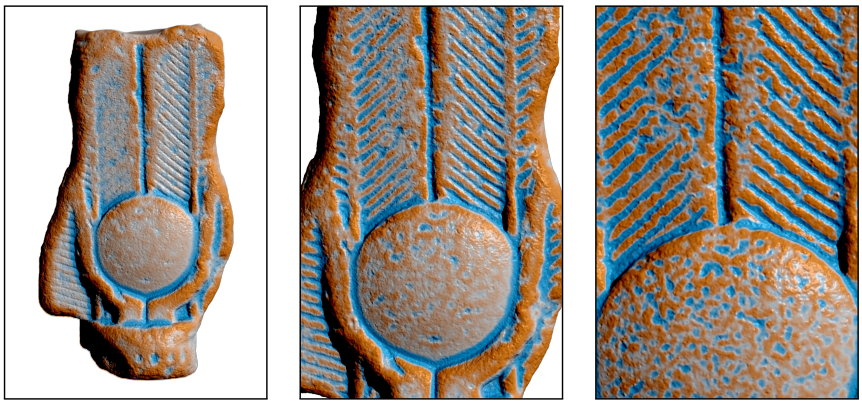


Figure 4.24: **Multi-scale screen space mean curvature estimation.** Visualization of $\check{\kappa}$, the curvature screen-space variant of our descriptor. We use a constant scale of about 25 pixels. The detected features depend only on the viewpoint.

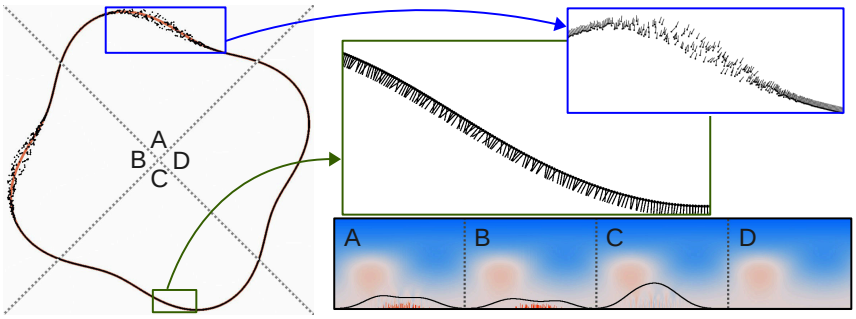
Expressive rendering is usually based on the understanding of how the perceived characteristics of a 3D scene (such as shading, color, or silhouettes) influence our perception of their properties (such as lights, materials, or shapes), in order to improve their perception.

The development of these approaches often requires to compute the object's properties in screen-space from a given viewpoint. Our fitting procedure can be adapted to compute a screen-space GLS descriptor, denoted by $\left[\check{\tau}, \check{\eta}, \check{\kappa}\right]$. In Figure 4.24, we show $\check{\kappa}$ that is computed using the Modo raytracing engine [Lux12]. We fit our primitive with a fixed neighborhood size in screen-space (about 25 pixels). Depending on the zoom factor, we identify coarse (left) to fine (right) features.

In future work, we plan to use our pertinent scale detection to select the best screen-space neighborhood size for each pixel with respect to a given viewpoint. The associated description can then be used as input for expressive rendering in order to highlight specific features. For example, $\check{\kappa}$ can be used with Radiance Scaling [VPB+10], a state-of-the-art method that uses a screen-space curvature estimation for surface enhancement.

Adaptive bandwidth for surface reconstruction When a point sampled manifold is corrupted with spatially varying noise as shown in Figure 4.25, it is not appropriate to reconstruct the signal at a single global scale. Instead, we must find a spatially varying scale that is locally adapted to the amplitude of the noise, also called adaptive bandwidth [WSS09]. However, care must be taken not to over-estimate this minimum scale, otherwise pertinent manifold structures may disappear.

Thanks to our geometric variation v, we can fill a scale-space for the study of 2D curves. We propose to use it to adaptively estimate the appropriate bandwidth: intuitively, we only have to find the smallest pertinent scale for each point $\mathbf{p}$. As a proof of concept, we propose a simple top-down heuristic that works well for smooth objects as follows. We detect the noise in a coarse-to-fine fashion: for each point we select the first scale where dv/dt is greater than a given threshold (we use 0.01 in this example) and use it as local bandwidth. We then regularize the result with a spatial smoothing defined along the curve. The resulting adaptive bandwidth is used as a variable support size to reconstruct a smooth curve, shown in orange in Figure 4.25 for different types of noise.



(a) Noisy 2D point set (b) Extracted bandwidth

Figure 4.25: **Adaptive bandwidth:** (a) 2D point set corrupted with noise on: normals and positions (A), positions (B), and normals (C). D is without noise. The reconstructed curve (shown in orange) is obtained using an adaptive bandwidth extracted from our continuous scale-space representation, shown in (b) as a black curve.

We will study the extension of this method to more complex objects in future work. A first promising test might be to use the pertinent scale heuristic proposed in Equation 4.15, in order to select the first pertinent scale that is locally coherent on the object.

4.5 Conclusion

We have presented a novel approach to scale-space analysis based on local regression that automatically and robustly characterizes the stable structure of 2D curves and 3D surfaces in a continuous manner. A strength of this approach is that it is entirely independent of any targeted application. Indeed, our continuous geometric variation scale-space may be interpreted and processed differently by different types of applications, as demonstrated in Section 4.4. Our solution considerably improves previous work by defining a new *geometry descriptor* and a novel alternative to multi-scale *feature detectors*.

First of all, our descriptor can be evaluated fully continuously in both space and scale, it is robust to noise (Figures 4.10, 4.11 and 4.20), it does not require any parametrization or connectivity, and it naturally deals with manifold borders (see Figure 4.8). Designed as a reparametrization of a fitted algebraic hyper-sphere, it approximates differential invariants at 3 different orders, yielding a robust extrinsic shape characterization. We compared our descriptor to previous work in Section 4.2. The DoG method could be adapted to handle point sets and manifold borders using local regression [LLWZ12]. However, DoG would still have to be applied to data points with geodesic neighborhoods, which are not extrinsic, often difficult to obtain, and it would provide only curvature measurements. Our approach identifies a more complete geometric descriptor while only requiring an Euclidean neighborhood with a reasonable number of point samples (at least about 5 in 2D and 12 in 3D). Quadric fitting is another alternative to compute curvature, but it is not adapted to scale-space analysis since it requires a local planar parametrization which is not globally coherent and not robust to fold-overs (see Section 4.2.1). In contrast, our method avoids these shortcomings by employing a direct second-order fitting procedure.

Secondly, we presented a new alternative to classical scale-space analysis based on the analytical differentiation of our descriptor when scale varies. This continuous evaluation avoids many shortcomings of pure 0-crossing tracking, a state of the art tool based on a discrete detection of stabilities in scale-space (see Section 4.3). The resulting *geometric variation* can be evaluated continuously and efficiently during the fitting procedure, it is robust and evolves continuously when the input changes, and it is able to detect multiple pertinent scales for a given evaluation point. This measure can be used to compare points and to detect underlying features at multiple scale ranges (see Figures 4.6, 4.15, 4.16), or to extract continuous feature areas on the object (see Figure 4.19).

A limitation of our approach comes from the choice of an isotropic regression: surface anisotropy is not explicitly identified. However, as shown in Section 4.4.2, our method is able to detect indirectly complex anisotropic structures or saddle-like shapes. Indeed, anisotropic features can be understood as a set of isotropic descriptors disposed along paths on the surface, while saddle-like shapes are disambiguated by their specific multi-scale descriptor profile. In future work, we plan to study spatial variations of our geometric descriptor, in a way similar to what is classically done with MLS (see preliminary results in Section 4.4.2). This would permit to explicitly identify direction fields on manifolds and the scales at which they are pertinent. We also plan to study the different shape types that can be characterized using both the spatial and scale variation. Our pipeline can also be tuned to measure intrinsic surface properties that are crucial for some applications. In this chapter, we have only considered shape geometry, and we ignored additional information such as color attributes. In future work, we plan to extend our weight functions to reject neighbor points with dissimilar attributes, yielding a *non-linear* version of our continuous scale-space analysis. We have also observed oscillations in v for input manifolds with regu-

lar structures (see Figures 4.3 and 4.4). For some applications, the oscillations might lead to false positives, which could be detected through a frequential analysis of ν. We have also identified a *drift* effect due to our fitting procedure. More globally, we plan to study deeply the meaning of each of our descriptor's parameters as well as their derivatives in order to anticipate such behaviors. Finally, we believe that these fundamental studies could lead to the development of new analysis tools adapted to various application cases.

Chapter 5

Semi-automatic matching

In the previous chapter, we have presented the elaboration and the study of theoretical tools for the multi-scale analysis of 3D objects. The work presented in this chapter was done in a concrete application-oriented research context: Cultural Heritage (CH). Together with researchers in archaeology, we especially focused on the problem of fractured object reassembly consisting of few fragments (up to about ten), but with missing parts due to erosion or deterioration.

We start by presenting the scope of this research in Section 5.1. More precisely, we recall previous work on 3D surface registration, being a key ingredient of many pairwise fragment matching algorithms. We then present concrete existing systems that have been proposed to reassemble archaeological objects. We finish this section by presenting our concrete dataset composed of fragments of the statues surrounding the legendary Alexandria lighthouse in Egypt, and we show that our waterworn data introduces specific constraints that require adapted matching procedures.

In order to address this reassembly problem, we have elaborated a theoretical semi-automatic formalism. The key idea is to design reassembly systems that combine both the expert user knowledge and the matching algorithms' speed and accuracy. The efficiency of the method is determined by both the geometry analysis and the robustness of the geometric matching, while its versatility is based on the knowledge of the final expert users and their possibilities to guide the reassembly. To this end, as the user takes into account additional information that is not easily representable as input for algorithms (e.g. historical texts or iconographies), he or she should be able to control in real-time the reassembly process using a dedicated interaction technique. This concept gives the user the possibility to test various hypotheses and to guide a digital reassembly process that is similar to manual reassembly methods, which are still used in CH.

In Section 5.2, we present our formalism for semi-automatic matching, using a real-time interaction loop between the user and the system. Then, we validate the feasibility of this formalism by two practical implementations for pairwise matching, one based on tangible interaction, and the other one based on multi-touch interaction. In order to produce setups that correspond to the requirements of the experts, we developed both implementations in close collaboration with researchers in archaeology and specialists of digitalization and virtual restoration of CH content. We also collaborated with researchers in human-computer interaction (HCI), and more precisely 3D user interaction (3DUI), for the design of appropriate interaction techniques. Considering the matching itself, we have adapted standard approaches to our context.

In Section 5.3, we present the results of informal studies of both implementations for the pairwise reassembly, and we conclude with preliminary work on the possibilities to combine our semi-automatic formalism with our multi-scale geometry analysis. Indeed, the versatility of the latter makes it usable during all steps of the involved pipeline. For example, it

Related publications: [MRB09, MRS09, MRS10, RMG+11]

provides the user with high-level information about the object's geometry at multiple scales, and it indicates relationships between surface points during the matching. We show how to use our multi-scale geometry analysis in order to help the user in selection tasks, to pre-process input data for matching algorithms, and to highlight features that can make sense for the user.

All this work is a part of a complete pipeline that we put up, ranging from the fragment digitalisation to their virtual reassembly. We even manufactured one of these reassembled objects for a temporary museum exhibition in Paris: an Isis statue that once was at the entrance of the Alexandria lighthouse.

5.1 Scope

5.1.1 3D surface registration

The registration of 3D surfaces has originally been studied for the alignment of partial views of a same object generated by range scans. The basic idea is to compute a transformation (rotation and then translation) to align both sheets of surfaces by minimizing a given energy. Today, the majority of fragment matching systems uses registration algorithms to align the fragments' contact surfaces (see Section 5.1.2), and in particular, the *Iterative Closest Point* (ICP) algorithm [BM92]. The idea is to *iteratively* minimize the Euclidean distance between two surfaces, by considering pairs of points that associate to each element in a range image its *closest* point in the other range image. The original ICP version was designed for fully overlapping surfaces, without noise and with exact correspondences between the two objects. There are some variants that have been proposed to deal with partially overlapping surfaces with different sampling [TL94, BS97, CSSK02], or to produce more robust solutions using color information [GRB94, UGR04], or geometric descriptors under the name *Iterative Corresponding Point* [RL01, Jos02] (we use this meaning for ICP in the following). The improvement of hardware and acquisition procedures also yielded the apparition of multi-view registration [CM91, BDL95, MSY96, DWJM98, Pul99] and real-time ICP variants [RHHL02].

The main limitation of the ICP is its dependence on the relative initial pose of the input surfaces. Indeed, the iterative minimization finds a transformation that corresponds to a local minima of a given energy that is close to the input position (in transformation space). Some work has been proposed to find the initial pose automatically [SJG05, SDG07, PGBP10], but the problem of local minima still remains: the algorithm's convergence for successful registration cannot be guaranteed without an a priori knowledge about the overlap area and the object's shape [BKW+08]. We propose to use this strong dependence to give the user a lot of control on the matching process, as shown in Section 5.2.

More recently, some approaches have been proposed to find a global solution for the registration problem. For example, German et al. [GBPP07] propose to automatically align an acquired range image to a known 3D object in a reverse engineering context. Liebelt et al. [LSS08] use a voting procedure in transformation space by considering rendered images of the object from different locations. Both approaches refer to a specific context with associated knowledge, and can be hard to adapt to fragment matching. We refer to the up-to-date survey of van Kaick et al. on shape correspondence [vKZHCO11] for a more in-depth presentation on this topic.

5.1.2 Archaeological object reassembly

The reassembly of digital fragments is a problem composed of two tasks: the rough alignment with the regularization of multiple pairwise relations, and the fine pairwise matching of fragments. Whereas the latter is based on registration algorithms, the rough alignment is usually achieved by graph optimization techniques. We briefly present this task in a first paragraph, and then we present more in-depth the systems that have been proposed to reassemble archaeological objects or monuments. Since this problem is very complex, some methods rely on specific data and heuristics to simplify it. Nevertheless, some versatile approaches have been proposed for the digital reassembly of fractured objects in CH, using manual, automatic, or semi-automatic concepts.

Rough alignment This step concerns the measurement and improvement of the global coherence between pairwise fragment relations to reassemble objects. Based on graph optimization, these techniques are essential for the reassembly process, but they are not specific to this problem and are widely studied in other fields of Computer Science. In brief, the rough alignment minimizes errors on a graph that represents the fragments as nodes and the matching relations as weighted edges, according to a provided metric. This can be used as pre-processing [TFK+09], or directly during the pairwise matching process as a local regularization (by using a subpart of the graph) [CWA+01, KK01], global regularization [CBR+11], or mixed approaches [ON12]. Usually, local approaches suffer from error accumulation along the graph.

Pairwise matching problem simplification The pairwise matching of fractured fragments is a complex topic, which requires to detect fractured surfaces on fragments, and to find potential compatible contact surfaces on other objects and match them. In order to tackle this complexity, several methods have been proposed to deal with specific data by reducing the dimensionality of the problem. For example, planar and thin artifacts (e.g. fresco fragments) can be considered only by their fractured surface, encoded as a 2D curve [Wol90, KK01], or more robustly as a ribbon with thickness information. The matching is then simplified to finding the optimal relative rotation [BTFN+08] between fragments (see Figure 5.1). The information coming from the top face can also be used to find more robust matches [KTNL05, CBR+11, FSTF+11]. Some other methods have also been proposed to reconstruct broken potteries and other rotationally symmetric objects [CWA+01, WOC03, WC04, KS04].

3D pairwise matching The reassembly of 3D objects without an a priori knowledge has been addressed in three different ways, depending on the user influence on the system.

A first approach presents the problem as an interaction task [RRC+07, RRC+10]. The authors propose a bi-manual tangible interaction technique to manipulate two fragments with their hands by using electromagnetically tracked props. This approach gives the user the possibility to move and rotate both fragments (two times six degrees of freedom) in order to find a coherent assembly. Some field studies with archaeologists have shown that this virtual manual approach is well adapted to the fragment reassembly task, and that it is coherent with the standard real world approach used by professionals [RRC+10]. Taking into account the expert knowledge is crucial in order to avoid to produce geometrically coherent reassemblies with no archaeological meaning. However, this purely manual approach can

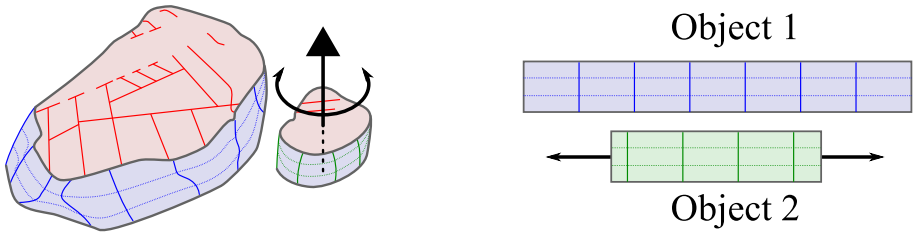


Figure 5.1: **2.5D matching of planar thin artifacts**. Left: a fresco fragment can be considered as a 2D ribbon (in blue and green) defining the fractured surface as heightfield (hence the name 2.5D). Right: the matching consists in finding a rotation around the axis defined by the fresco plane, corresponding to a translation in the ribbon domain. Visible patterns (in red on the left) can also be used to check the top surface continuity between fragments.

only reasonably produce rough alignments because it is not associated to any automatic geometry processing.

A second approach tackles the problem in an opposite direction by proposing a purely automatic reassembly system [HFG+06]. The fragments are segmented, and fractured faces are extracted by analyzing the surface roughness using *integral invariants* (see Section 3.2.2) for pairwise matching. This approach seems promising since the reconstruction seems coherent, but it has been tested only on data with perfect matches (without missing data or weatherworn effects) and with a small numbers of fragments, which is often not the case in for many real datasets involved in Cultural Heritage.

In a third type of approach, the authors propose intermediate solutions based on semi-automatic problem solving. The idea is to simplify the global process by letting the user perform one of the following steps:

1. select the relevant fragments,
2. segment fragments, extract contact surfaces or features,
3. provide pairwise relations between fragments, surfaces, or features.

The high-level knowledge usually involved to perform these tasks, most of the time by the archaeologists, can be hard to represent for the virtual reassembly without semi-automatic approaches.

One instance of this approach is the method proposed by Oxholm et al. [ON12] to reassemble thin artifacts that do not refer to fresco fragments or rotationally symmetric objects. The user has to accept or reject an extracted 2D curve representing the fragment boundary in order to ensure that it is a part of the currently reconstructed object, and in order to avoid invalid segmentations (items 1 and 2). Another example is the method presented in [TFK+09] for the digital anastylosis[1] of the Octagon monument in Turkey. The authors present a complete system to acquire and reconstruct numerous fragments automatically. Then, they provide the user with the possibility to specify the relations between the fragments, and to provide a rough initial position for the matching (items 1 and 3). The efficiency of the method described in this paper with an example on a real dataset validates the

[1] Archaeological term for *the reassembly of ruined monuments or other artifacts from remaining fragments in an archaeologically responsible way (with use of modern materials when needed)*, Definition by "http://en.wiktionary.org/wiki/anastylosis"

process and confirms the efficiency of semi-automatic methods elaborated in collaboration with archaeologists.

5.1.3 Application context

Archaeological context This work has been produced in collaboration with the archaeological research team of the CEAlex[1], a research center that studies, besides others, the Lighthouse of Alexandria, one of the Seven Wonders of the Ancient World. Badly damaged by several earthquakes and submerged under the Mediterranean sea hundreds of years ago, the fragments have been eroded by water and currents. Whereas some fragments are today exhibited in museums, the majority might never be raised due to the required infrastructure and the cost of the operation. Concerning the few raised fragments, in Figure 5.2, we show two parts of the colossal Isis statue, originally around 12 meters tall and weighting tens of tons (5 tons for the small headpiece). Concerning the reassembly, the fragment manipulation requires to use cranes or oxygen balloons underwater, so the archaeologists generally rather work with underwater measurements and drawings to test their hypotheses.

Acquisition By taking into account the different constraints, such as the size and the weight of the fragments and the fact that they are underwater, we defined an acquisition protocol in collaboration with ArcheoVision[2], a French team specialized in the acquisition, preservation and storage of Cultural Heritage content. In a preliminary mission, we went to Egypt to see the acquisition sites and to design procedures and tools to acquire the fragments. We were diving the underwater archeological site in order to understand the involved difficulties: stacks of fragments weighting several tons, the complexity of their manipulation and the necessity to put them in a clean area for proper acquisition (without other fragments and composed of sand), lack of visibility and presence of floating objects or fishes that disturb the acquisition. During our trip, we have started to adapt a photogrammetry reconstruction method [FP10] with Patrick Reuter[*] and Bruno Dutailly[†]. We used it to treat, during the night, the photographs taken during the day by Pascal Mora[‡], guided by the archaeologist Isabelle Hairy[§]. In a week, 49 fragments, of which 27 are underwater, have been photographed, and several fragments have been reconstructed in low resolution. Back to France, the reconstruction method has been improved by Elric Delord[¶], yielding the software SynAps [DA12] that is able to generate point sets with around one million of points, with per-point color information. The resulting dataset have been used in our experiments to test and design the interaction techniques, the geometric matching, and the advanced visualization. Several archaeologists have also used our methods for other research topics.

Virtual reassembly For the virtual reassembly, on the one hand, the automatic approaches presented in Section 5.1.2 cannot be used due to the extrem degradation of the

[1] *Centre d'Études Alexandrines*, USR 3134, Egypt.
[2] Archeovision UPS SHS 3D 3551, France. archeovision.cnrs.fr
[*] Inria - Univ. Bordeaux - IOGS - CNRS, France. Contact
[†] PFT3D Archeovision UPS SHS 3D 3551 - LAPP - PACEA UMR5199, France. Contact
[‡] PFT3D Archeovision UPS SHS 3D 3551 - ArcheoTransfert, France. Contact
[§] CEAlex USR 3134, Egypt. Contact
[¶] PFT3D Archeovision UPS SHS 3D 3551, France. Contact

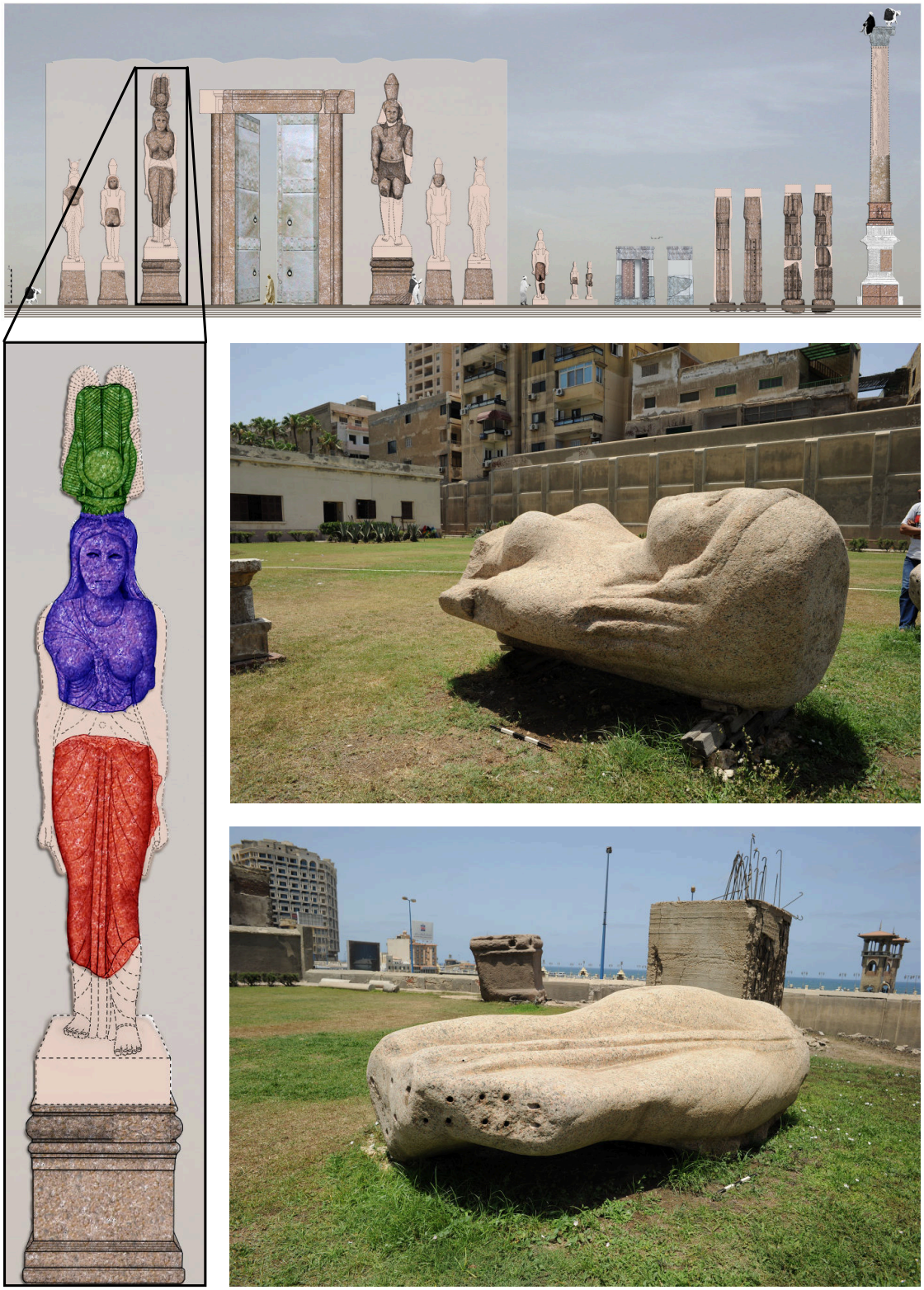


Figure 5.2: **Fragments of statues surrounding the Lighthouse of Alexandria**. Top: illustration of the Colossal Statues surrounding the Pharos (illustration by Isabelle Hairy). Left: zoom of the colossal Isis statue, broken in three fragments. Note the important erosion of each fragment that alter the fractured surfaces and erase smallest details. Right: examples of some photographs of the Isis fragments used for the photogrammetry reconstruction.

fragments. An important fact is that the erosion is not uniform over the surface, and some parts might be completely smoothed, while others still have fine patterns. This local surface smoothness prevents from systematically using fine details or detecting fractured surfaces via automatic processes. Another problem is that sharp edges are used to segment the fragments and to extract the contact surfaces. In our case, it can be very hard to distinguish between small patterns and smoothed edges, which makes segmentation algorithms unusable.

Some researchers in archeology are convinced that the lighthouse and the surrounding statues have been conceived according to specific mathematical regularities (see for example [Hai06]). These assumptions about the entire object can be crucial for the reassembly process (such as dimensions and proportions), and lead to successful reassemblies even though only a few fragments still remain.

The difficulties inherent to our dataset and the precise expert knowledge needed to reassemble the fragments induced us to use a semi-automatic approach for the matching. The methods proposed in previous work cannot be directly used since they require perfect matches [ON12], or since they use specific heuristics adapted to the architectural content [TFK+09], such as for instance the detection of planar surfaces. In the next section, we present a generic semi-automatic formalism and two concrete implementations, one based on tangible user interaction (see Section 5.2.2) and multi-touch surface interaction (see Section 5.2.3).

5.2 Semi-automatic matching

5.2.1 Semi-automatic formalism

The central idea of this formalism is to increase the efficiency of the reassembly process by integrating real-time geometry-driven matching algorithms into the user interaction loop. We call this idea the *semi-automatic reassembly interaction loop* (Figure 5.3), and there are three involved prerequisites.

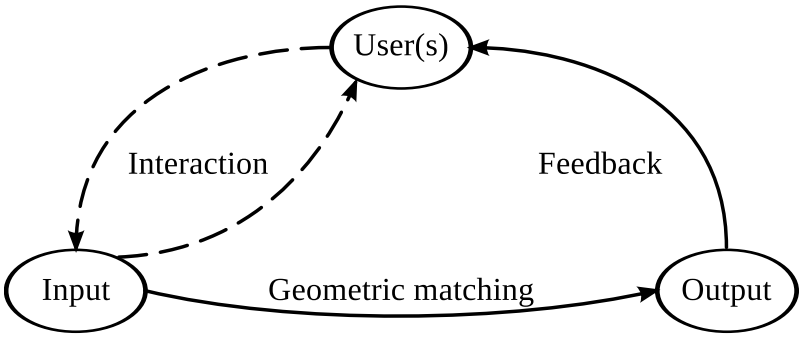


Figure 5.3: **The semi-automatic interaction loop.** We propose to enforce the standard interaction loop (dashed lines) by adding a real-time geometric matching whose input is defined by the user via the interaction technique, and whose output is the feedback to the user.

Interaction First, we require an efficient *interaction technique* so that the expert user can express prior knowledge on the matching input. It should allow him or her to remain

concentrated on the actual archaeological task and involved high-level knowledge, without being distracted from difficulties to interact with the system.

Geometric matching Second, we require a *geometric matching* algorithm that computes the locally optimal match with respect to the user-specified input. Of course, this should be done in real-time to ensure interactivity. Moreover, the matching output has to be coherent over time: in order to avoid annoying popping artifacts, slight changes in the input should not affect the matching output, except in areas of transition to different local best matches.

Feedback Third, we have to provide *feedback* from the geometric matching to make it easy for a user to validate the match, or to refine the input. The information transmitted to the user can be linked to the fragments' properties (for a better comprehension of the input) or to the matching result (for a better comprehension of the output).

This general principle makes it possible to use different approaches for each of the three prerequisites. In the following, we show two independent implementations of our idea, and the concrete choices that we have made for the interaction and feedback prerequisites. In order to be able to compare both of them, we use an unique geometric matching kernel, based on a real-time variant of the ICP, described in Section 5.2.4. For each implementation, we present the way we transform the user commands to an ICP input.

5.2.2 Semi-automatic reassembly with tangible interaction

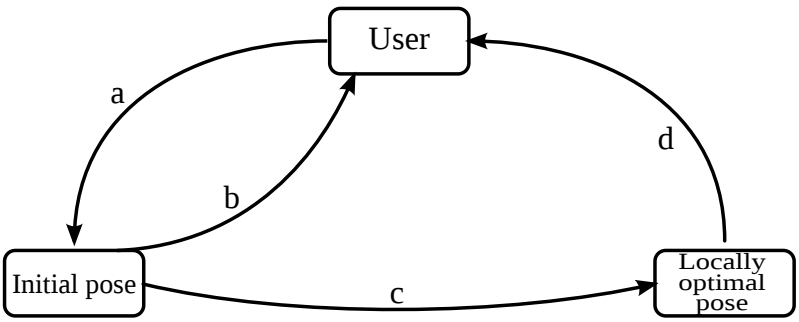


Figure 5.4: **The semi-automatic interaction loop for the tangible interaction technique**. (a) Specification of the initial pose using the bimanual tangible user interface. (b) Real-time visualization of the initial pose. (c) Real-time geometric matching. (d) Visualization of both the locally optimal match and the global matching error.

Interaction technique In this section, we propose a first solution for pairwise semi-automatic reassembly where the expert user can directly specify the relative positions and orientations of fragments that are used as input for the matching algorithm. For an efficient user interaction of this two times six degrees of freedom task, we rely on previous work dealing with bimanual tangible user interfaces [RRC+07]: as can be seen in Figure 5.5, in each hand, the user manipulates an electromagnetically tracked prop (items 1 and 2), and the translations and rotations are directly mapped to the corresponding virtual fragments on the display (items 3 and 4). For each side, a foot pedal (items 5 and 6) is used to activate

the tracking of the props, like "clutching" a gear in a car. The involved semi-automatic interaction loop is illustrated in Figure 5.4.

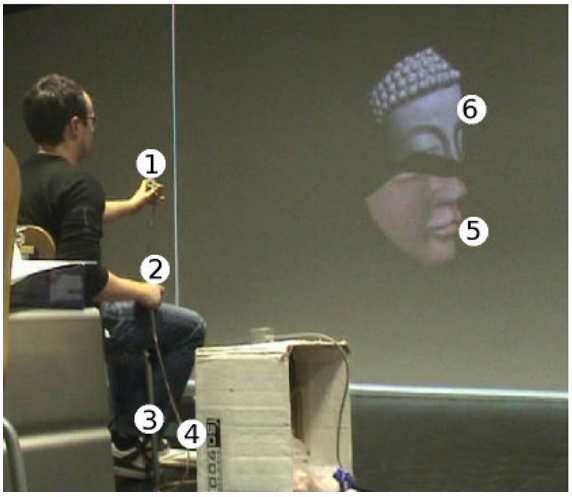


Figure 5.5: **The bimanual tangible user interface.**

Geometric Matching As explained in Section 5.1.1, the required input to the ICP is the initial relative pose of each object. It is thus straightforward to use with the bimanual tangible user interface by considering the poses specified by the user in real-time.

Feedback Once the locally optimal match with respect to the user-specified initial pose has been determined, we have to provide feedback about it. We use a visual semi-transparent 3D representation of the matching result, as can be seen in Figure 5.6. Based on this visual feedback, the user can evaluate whether the matching result is geometrically plausible and coherent with his intent, and either validate the proposition, or specify a new initial pose. As can be seen in Figure 5.6, we also provide a graphical indicator of the current ICP error; we use the root mean square (RMS) error of the matching of the contact surfaces.

Note that with this visual feedback, the reassembly "snaps" to the locally best corresponding match, like a magnet that sticks the fragments together. This information is provided as long as the provided pose is within a distance threshold, and, of course, it changes when the user-specified pose is closer to a different local minimum. A similar "snap" metaphor has proven to be efficient in 2D vector graphics applications, where imaginary grid lines at a coarse spacing help to precisely align 2D objects despite a roughly aligned input.

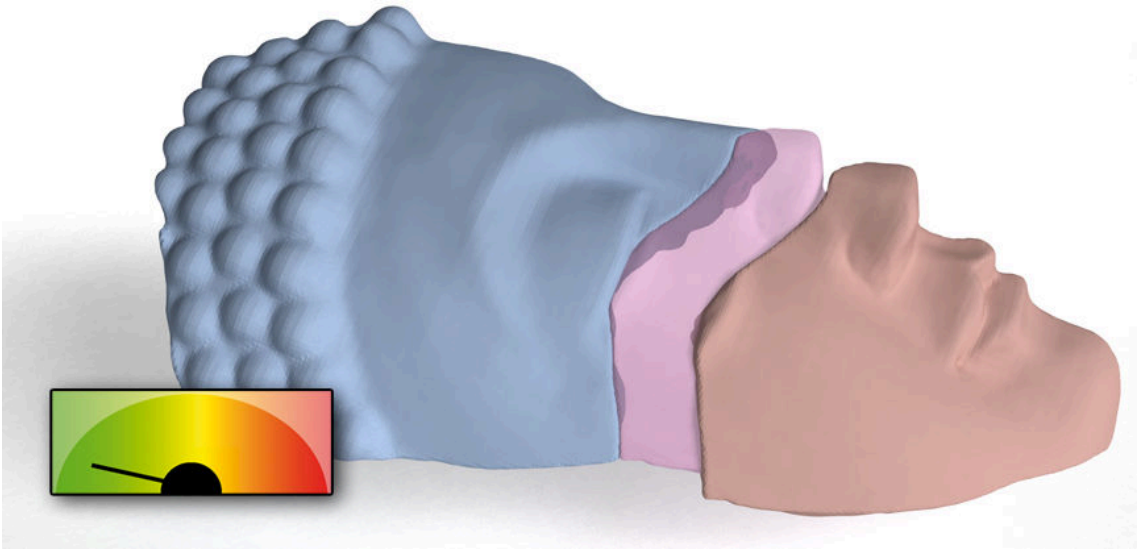


Figure 5.6: **Feedback of the tangible setup.** The system informs the user about the matching output by displaying a transparent view of the locally optimal match according to the given initial position. It also provides quantitative information about the match by displaying the error of the last ICP iteration.

5.2.3 Semi-automatic reassembly with multi-touch interaction

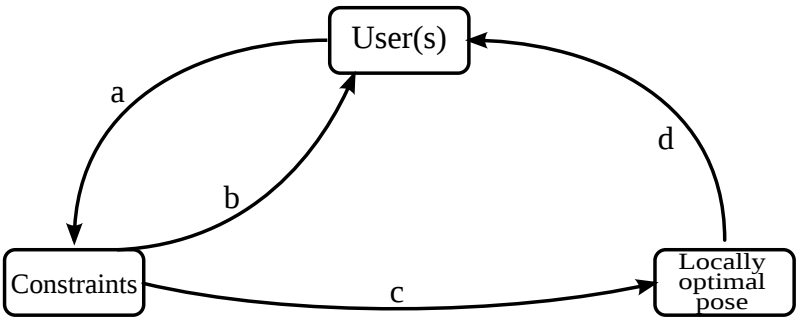


Figure 5.7: **The semi-automatic interaction loop for the multi-touch interaction technique**. The semi-automatic interaction loop for the multi-touch interaction. (a) Fragment selection (top window), fragment manipulation and specification of the constraints (bottom right and left windows). (b) Real-time visualization of the fragments and constrains (bottom right and left windows). (c) Real-time geometric matching. (d) Visualisation of the best local match (result window).

Interaction technique In this section, we propose a second solution for the pairwise semi-automatic reassembly using a multi-touch tactile interface. Instead of letting the expert user directly specify the relative positions and orientations of potential reassemblies, we rather let him reason about corresponding vertex pairs of the fragments. This approach has been elaborated in collaboration with Aurelie Cohe[‖], during her PhD about 3D user interaction. For an illustration of our solution, consider Figure 5.8.

[‖]Inria - Univ. Bordeaux - CNRS, France. Contact

Figure 5.8: **Multi-touch interaction**. Top: the selection window. Bottom left and right: the manipulation windows. Bottom middle: the result window.

First, the expert user can browse through the set of acquired fragments and identify potential pairwise reassembly candidates (Figure 5.8, top). The selected fragments are displayed in two separate manipulation windows (Figure 5.8, bottom left and right). Then, the user specifies multiple reassembly constraints on the two fragments by tipping on corresponding positions. The best match is calculated in real-time and shown in the visualization window, thus providing feedback to the user (Figure 5.8, bottom middle). The constraints of both fragments can be readjusted steadily and simultaneously thanks to the multi-touch tactile interaction, and the best matching result is calculated on-the-fly. Again, the semi-automatic reassembly interaction loop applies, as illustrated in Figure 5.7.

Geometric Matching The best geometric match has to be computed from the user-specified reassembly constraints. Since the relative initial positions and orientations are not specified directly, we have integrated an intermediate step for their determination: the constraints are considered as pairs, and a rigid transformation can easily and efficiently be calculated using quaternion-based minimization [BM92], when the user provide at least three constraint pairs. Then, we use our optimized ICP variant in order to compute the best local geometric match.

Feedback We provide two kinds of visual feedback to the user as is illustrated in Figure 5.8. First, the pairs of the associated areas of the two objects are recalled, so that the user can continuously modify the constraints if necessary. Second, the result of the geometric matching is displayed in a separate window. As a consequence, the user can inspect simultaneously the contact surface, and see the pairwise constraints and the result of the matching. Note that we could provide a graphical indicator of the local error of the ICP algorithm as before.

5.2.4 Real-time matching kernel

Since the 3D objects of the fragments are acquired either by range scanners or by photogrammetry, they are usually defined as huge and noisy unstructured point sets. However,

the interaction loop involved in our semi-automatic matching algorithm (see Figure 5.3) requires our algorithm to operate in real-time, and to robustly align the two surface sheets even in the presence of noise.

According to Rusinkiewicz's fast ICP variants [RL01], an iteration is composed of six steps that are likely to be optimized: data selection, pairwise vertex matching, weighting pairs, rejecting pairs, computing an error, and minimizing the error. In the following, we show how we optimize the first four steps for a more efficient reassembly. In particular, we will focus on the vertex pair selection as well as on their weighting and rejection.

Efficient vertex pair selection For an efficient alignment of the two surface sheets, we first downsample the vertices of the fragments to a few tens of thousands of vertices [TL94, RL01] in a preprocess. For further optimization, we only need to select the vertices that are present on the potential contact surface, while rejecting all the others. In our semi-automatic reassembly method, the relative initial positions and orientations of the two fragments are directly specified by the user, or estimated from the reassembly constraints, so that the potential contact surface is already roughly aligned. As a consequence, we can consider that the vertices of both contact surface sheets are close to each other when their distance is less than a user-specified threshold distance d_T.

For the efficient detection of contact surfaces and closest vertex queries, we construct a kd-tree for each of the two fragments in a preprocess. Furthermore, inspired by classical collision detection algorithms [Lin93], we mix the kd-tree with a bounding sphere hierarchy: for every node of the kd-tree, we determine a sphere that encloses all vertices of the child nodes.

Consequently, in the real-time interaction loop, for every user-specified initial pose, we can efficiently determine all the vertices of the contact surface by intersecting the two hierarchies recursively (see the bold spheres b_s and b'_s in Figure 5.9). This pruning is slightly different compared to collision detection since we do not only want to obtain the sphere intersections, but also all vertices where the distance is less than d_T. Our solution is conservative: we increase the sphere radii by $\frac{d_T}{2}$ to capture all necessary vertices, but also the vertices at a distance greater than d_T (see the dotted spheres in Figure 5.9).

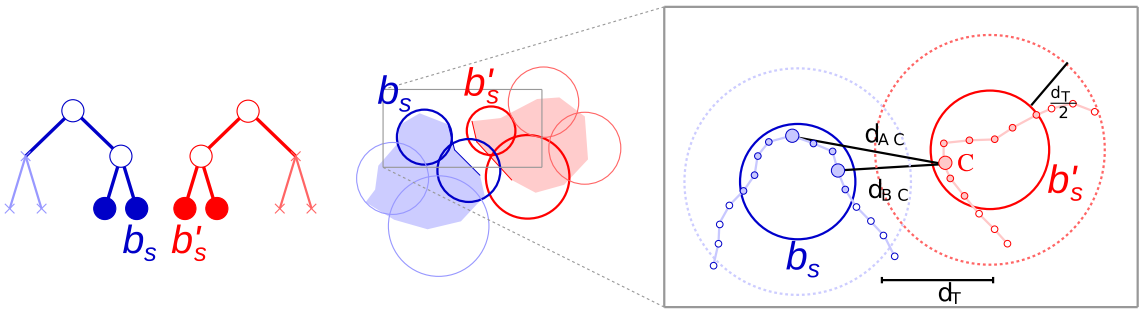


Figure 5.9: **Bounding sphere hierarchy.** Left: The two bounding sphere hierarchies. Right: Determining the contact surface by conservative pruning of the two hierarchies: even though $d_{BC} < d_T$, the sphere intersection does not detect it. By increasing the sphere radii by $\frac{d_T}{2}$, the intersection is successfully determined, but more distant vertices are detected as well ($d_{AC} > d_T$).

In order to obtain the closest vertex pairs, for every vertex of the potential contact surface

of the first fragment, we efficiently determine the closest vertex of the second fragment by using the kd-tree.

Vertex pair weighting and rejection Recall that once all the closest vertex pairs are conservatively determined, we have to reject all the pairs with a distance d greater than d_T in order to consider only the contact surface. Moreover, we have to determine the influence of each pair. Following Pulli [Pul99], we take into account normal coherence in addition to vertex distance: pairs with normals of opposite direction should have a greater influence (marked in green in Figure 5.10) compared to other neighboring pairs of the contact surface (marked in yellow in Figure 5.10). Pairs where the dot product of the normals η is greater than an orientation threshold o_T should not be taken into account at all. The result is a better convergence of the ICP algorithm.

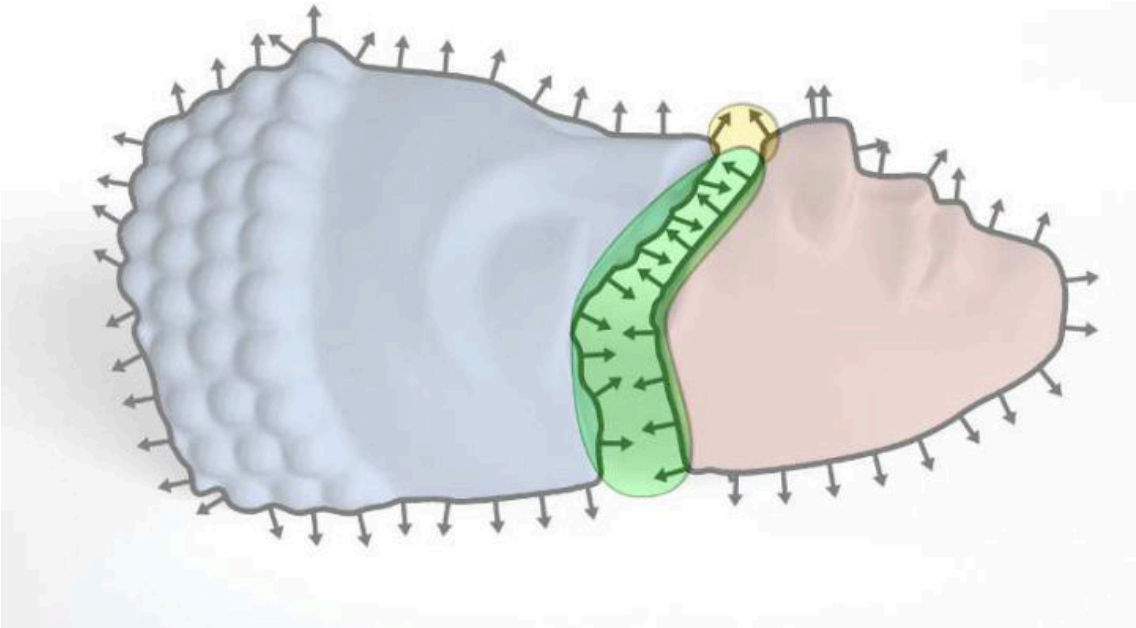


Figure 5.10: **Normal coherence**. Illustration of the normal orientation of the fractured surfaces. In the green area, compatible normals are oriented in opposite directions, while the yellow area contains orthogonal normals that are rejected by our weighting function.

For an efficient interaction, we must ensure that the locally optimal match evolves coherently over time. Indeed, when the user slightly changes the initial relative pose or the paired constraints of the two fragments, the matching should only vary gradually in order to prevent from popping artifacts when some vertex pairs oscillate around the distance threshold d_T or the normal coherence threshold o_T. Therefore, we propose to smooth both the distance of the vertex pairs and their associated normals with a decay function $f(x) = (x^2-1)^2$. This results in the weighting functions f_d for the vertex distance d (Equation 5.1) and f_n for the normal coherence (Equation 5.2). A zero weight corresponds to a pair rejection.

$$\begin{cases} f_d(d) = \left(\left(\frac{d}{d_T}\right)^2 - 1\right)^2 & \text{if} \quad 0 \le d \le d_T \\ f_d(d) = 0 & \text{otherwise} \end{cases} \tag{5.1}$$

$$\begin{cases} f_n(\eta) = \left(\left(\frac{1+\eta}{1-o_T}\right)^2 - 1\right)^2 & \text{if} \quad -1 \le \eta \le -o_T \\ f_n(\eta) = 0 & \text{otherwise} \end{cases} \tag{5.2}$$

We combine both weighting criteria in the following separable weighting function where $\alpha \in [0;1]$ is a parameter that can be tuned by the user in order to use only normal filtering ($\alpha = 0$) or per-distance weighting ($\alpha = 1$), while using $\alpha = 0.5$ gives both criteria the same influence .

$$B_f(d,\eta,\alpha) = \frac{e^{\left(-\frac{\alpha}{f_d(d)}\right)} * e^{\left(-\frac{1-\alpha}{f_n(\eta)}\right)}}{e^{-1}} \tag{5.3}$$

5.3 Results and perspectives

We now show results for the choices that we made for each of the prerequisites of the semi-automatic interaction loop: the interaction technique, the matching algorithm, and the feedback provided to the user. All the results are based on informal user studies of both interaction techniques, as well as on our own reassembly experience on a variety of fractured objects. We also present perspectives for each topic that we plan to work on in the future.

5.3.1 Interaction

Results Concerning the bimanual tangible interaction, we have observed a good acceptance and an efficient manipulation after a short learning process. It allows the user to directly manipulate the fragments with tracked props. This direct and continuous metaphor is invisible for the user, and so he or she can remain concentrated on the pairwise reassembly. However, we identified two major drawbacks. First, the lack of haptic feedback and the absence of collision detection is sometimes irritating during the fragment manipulation. Second, the efficient interaction with tangible props is interrupted when the user has to interact with the system for other tasks than the pairwise matching, such as the selection of a new fragment out of the database, or the tuning of matching parameters.

This was our major motivation to find an alternative solution, and so we developed the multi-touch interaction. Here, the interaction with the global interface (such as the fragment selection and parameter tuning), and the pairwise specification of the assembly constraints are both based on the same input device. The user remains concentrated on the task throughout the entire multi-piece reassembly, and not solely during the pairwise matching. Moreover, multi-touch interaction enables fast selection [FWSB07]. However, touch screens are considered imprecise for the selection of small targets [FWSB07]. In our application, this limitation is overcome by the use of a geometric matching algorithm that regularizes the rough constraint relations by an accurate registration.

Let us now compare both interaction techniques. In the tangible interaction, the fragment manipulation is natural because the user's reassembly gestures are similar to their real-world counterparts. The movements are continuous and aim faithful visualization, whereas the "snapping" discretely emphasis the geometrically plausible user inputs through the transparent object rendering. On the other hand, in the multi-touch technique, constraints are set directly in a discrete manner, which avoids a lot of the gestures to get the objects aligned. The assembly is updated continuously at each constraint readjustment and thus provides a better precision.

Perspectives In future work, we plan to improve the proposed interaction techniques in order to provide the user with more possibilities to input information for the matching process. For example, in our dataset, waterworn fragments cannot match anymore due to the eroded parts. We plan to add the possibility to define new constraints such as the continuity between surfaces and features instead of only using the contact surface. This is easy to set up with the multi-touch interaction. In a preliminary study on the Isis statue, we have used our analysis to highlight anisotropic features on the fragments, as shown in Figure 5.11, left. In collaboration with archaeologists, we used this information, displayed as color on each object, to visually align them (see Figure 5.11, middle). Once the reassembly was validated and compared to its theoretical properties (e.g. the total height of the Isis statue), we transmitted the reassembled 3D object to a french company [Nea12] specialized in digital object's physical reproduction. They fabricated each fragment individually in polystyrene with projected resin at scale $1:5$, and then reproduced the material by painting the surface. Once assembled and stabilized with a metallic support, the resulting reassembled statue has been exhibited in the French National Maritime Museum (see Figure 5.11, right). The temporary exhibition, named *Phares*[1] presents the history of lighthouses, and especially the last results of the archaeological studies on the Alexandria Pharos thanks to the fragment acquisitions and the virtual reassembly.

This first experiment validates the use of continuity constraints between fragments and motivates us to further investigate their use for the reassembly. This work also offer interesting opportunities from an acquisition technique point of view. Indeed, we now have both the fabricated and digital versions of three acquired fragments, that we plan to use as datasets to test the accuracy of the developed acquisition technique. Furthermore, the process involved during this experiment for the reproduction of the material properties is also interesting. Acquisition methods often produce a per-vertex color information, that can be stored in a texture, which is usually sufficient to represent the object appearance when dealing with digital representations. In the case of the fabrication of physical object, this representation is not anymore sufficient, and requires the development of new methods to acquire and reproduce materials.

Another interesting direction that we plan to work on is the development of a tool for advanced selection, which operates using our multi-scale analysis. Indeed, with the current system, when a user picks a point on the 3D object, it can refer to different structures at multiple scales. An interactive tool that lists pertinent scales (see Section 4.4.3) and the associated similar points around the selection should produce more accurate constraint definitions. When the user selects a set of points, we can use the coherency between these points to detect automatically the underlying feature. Another option is to use our descriptor variation flow to help the user extract an anisotropic feature as a 3D curve (see Sec-

[1] Exposition Phares *(Lighthouses)*, from March 7th to November 4th 2012, in Paris, France

Figure 5.11: **Fragment reassembly using continuity constraints**. Left: Definition of continuity constraints on features (dashed, in red) and surfaces (dashed, in green). Middle: Virtual reassembly of the Isis statue. Right: Manufactured statue (scale 1 : 5).

tion 4.4.2). Finally, a similar approach can be used to propose the user the structure on the second object that can match a currently selected feature on the first one. This set of tools gives an a priori indication of the fragments' compatibilities, and helps define an accurate input configuration for the matching, which we believe will improve its convergence.

5.3.2 Geometric matching

Results In this work we have focused on the definition of the interaction loop and on the definition of concrete setups. We have adapted a standard ICP to operate in real-time and ensure coherent matching output with respect to successive user inputs. However, our basic implementation has difficulties to converge when objects interpenetrate at the initial pose due to the high number of closest vertices that are not at the contact surface.

Note also that for every pair of fragments, there are three parameters that have to be adjusted beforehand which confers flexibility to the technique. However, it can sometimes be tedious to correctly choose these parameters, especially for noisy and eroded objects. This is a direct consequence of using the ICP algorithm for matching, as it only takes into account positions and normals.

Perspectives The difficulty of using this kind of parameters motivates the use of an advanced geometry analysis in order to adapt them automatically. In future work, we plan to use our multi-scale analysis directly during the matching process. First, we may use it for corresponding point selection and pair weighting in the ICP. Indeed, our multi-scale descriptor can be used as criterion for the selection of corresponding points, while the multi-scale dissimilarity measure (see Equation 4.11) seems to be a valid candidate for pairs weighting. A second possibility is to use our multi-scale analysis with a multi-resolution variant of the ICP [JH02]. This can be used to perform an adaptive sampling of the points depending of their detected pertinence (see Section 4.4.3), or this might help to choose the best scales for the generation of the different resolutions.

Another interesting topic is the incorporation of continuity constraints in the matching process. Indeed, most of the methods usually minimize the distance between points, which is not feasible for some real dataset with missing parts, like ours. We plan to work on the possibility to add curve and surface continuity constraints in our matching process. For example, we plan to minimize high-order derivatives of the completed surface represented by the pairwise assembly, most probably by using an implicit formulation that is compatible with our analysis framework. We emphasis that the objective is not complete the missing parts, as in [HT10], but to ensure the coherency and alignment between fragments while letting the missing parts empty.

5.3.3 Feedback

Results Concerning the *feedback* provided to the user, the informal user study shows that the visual feedback helps analyze the position of the best match and reason about its plausibility. Moreover, the graphical indicator allows the user to rapidly detect whether the local matching of the contact surfaces converges well. By a combination of both visual feedbacks, the user has a complete understanding of the global and local coherence of the matching.

The multitouch setup uses multiple views to provide the user with the possibility to see, at the same time, both the matching result and the contact surface with the specified constraints.

Anyway in both cases, the use of flat displays to manipulate relatively to each others multiple 3D objects introduces an ambiguity on their depths. This problem has been observed many times in our experiments. Despite our experiments with immersive setups [MRB09], we rather plan to address this problem by adding visual indicators in the 3D scene, such as drop shadows.

Perspectives For efficient pairwise matching, it is often important to study fine details on the surface of the fragments, either for comparing the contact zone, or for identifying continuities on the profile. In order to increase the perception of fine details, we use an expressive rendering technique [VPB+10] to emphasize the curvature of the surface (see Figure 5.12). In future works we plan to use our multi-scale analysis to adapt the curvature scale on the surface, depending on the local pertinent scale, and moreover on the scale of the user-selected features. This analysis can be expressed integrally in screen space during the rendering, as shown in Section 4.4.4.

Figure 5.12: **Detail enhancement**. Top: classical lit sphere shading. Bottom: Radiance Scaling [VPB+10].

In the case of eroded objects, we also plan to introduce a visualization of the probability of alignment of preliminary selected features, in the same way that we display a transparent matched object in the tangible setup. This could assist the user to check the continuity of features between both fragments. We emphasis that the goal is not to perform a shape completion, because this is only possible by combining archaeological knowledge with geometric reasoning. Finally, the local coherence could further be improved by showing multiple local error indicators, for example directly on the surface, instead of only one global indicator.

5.4 Conclusion

In this chapter, we have presented a semi-automatic interaction loop for the real-time reassembly of fractured archaeological objects. This interaction loop consists of an efficient interaction technique, a real-time matching algorithm, and a way to provide the user with a visual feedback about the best match and its associated error. We consider the user as the key operator of our approach: his or her knowledge and capacity to integrate semantic information in the reassembly process are used to increase the performance of the matching. We presented two concrete solutions for the efficient interaction technique, one based on bimanual user interaction, and the other one based on a multi-touch interface.

Our first results of informal user studies of both solutions show that they are capable of assisting an expert user in real-time during the pairwise matching of downsampled 3D fragments. Although our algorithm is optimized with spatial data structures, it could further be accelerated by a better exploitation of the system resources (e.g. multithreading or GPU computation).

In future work, we would like to address both the improvement of the user interaction, and the efficiency of the geometric matching. Concerning the bimanual tangible interaction, we will study the possibilities to integrate haptic feedback, and concerning the multi-touch interaction, we strive to let the user specify higher order constraints. In order to improve the geometric matching, we plan to integrate higher order derivatives and a multi-scale matching technique in the semi-automatic reassembly process.

For reassembly problems with a large number of fragments, we also believe that an a priori analysis of the all the fragments for salient feature detection at different scales could be used to identify potential matching candidates and their initial relative poses that can then be validated and refined by expert users by means of visual feedback. This first selection could reduce the number of potential matching candidates and thus make sequential fragment matching more efficient.

As a conclusion, this work on pairwise matching defines guidelines to elaborate semi-automatic fragment matching systems. We strongly believe that its association with our multi-scale analysis yields to the development of efficient solutions for concrete applications with complex datasets.

Chapter 6

Conclusion

In this thesis, we have presented a new analysis framework for the multi-scale geometric analysis of 3D objects represented as point sets, and a semi-automatic formalism for the digital reassembly of fractured objects from strongly deteriorated fragments.

Growing least squares

In Figure 6.1, we show a synthetic view of the multi-scale analysis framework presented in Chapter 4, which works both on 2D curves and 3D surfaces. Starting from an algebraic hyper-sphere fitting procedure, defined implicitly by a scalar field, we have proposed a robust geometric descriptor, composed of three parameters:

- The τ parameter that encodes the offset between the evaluation point and the fitted hyper-sphere,
- the η parameter that represents the direction of the scalar field gradient and that, in practice, corresponds to the normal vector, and
- the κ parameter that is the mean curvature of the fitted surface at the evaluation point.

The resulting descriptor characterizes the surface at three differential orders, providing a more robust description than the standard curvature. We have shown that our approach is more stable than previous curvature estimations, even in the presence of noise, for both 2D curves and 3D objects.

We propose an alternative to standard scale-space analysis by analytically differentiating our descriptor along the scale dimension. We then use the result to continuously capture the fitted sphere stabilities both in scale and space, via our geometric variation ν. We show how to use this analysis to define a dissimilarity function d that compares descriptors at multiple scales and at different locations. We exhibit the relevance of this comparison at locations that belong to multiple features at different scales, by detecting independently each of these features. Despite the use of an isotropic fitting procedure, we have shown the potential of our method to disambiguate and extract even anisotropic features.

Our analysis is efficiently performed using massively parallel GPGPU programming. As a consequence, it is able to analyze point sets consisting of millions of points in a few minutes. It is also parameter-less, and may be evaluated either for the entire object, or at specific locations if required.

We have also presented preliminary studies to extend our analysis framework (the dashed items in Figure 6.1).

A first topic that we plan to pursue is the explicit extraction of *pertinent scales* for a given location. Starting from our geometric variation ν, we have defined a Gaussian convolution operator μ that produces a strong local maximum for scales being at the end of a stability

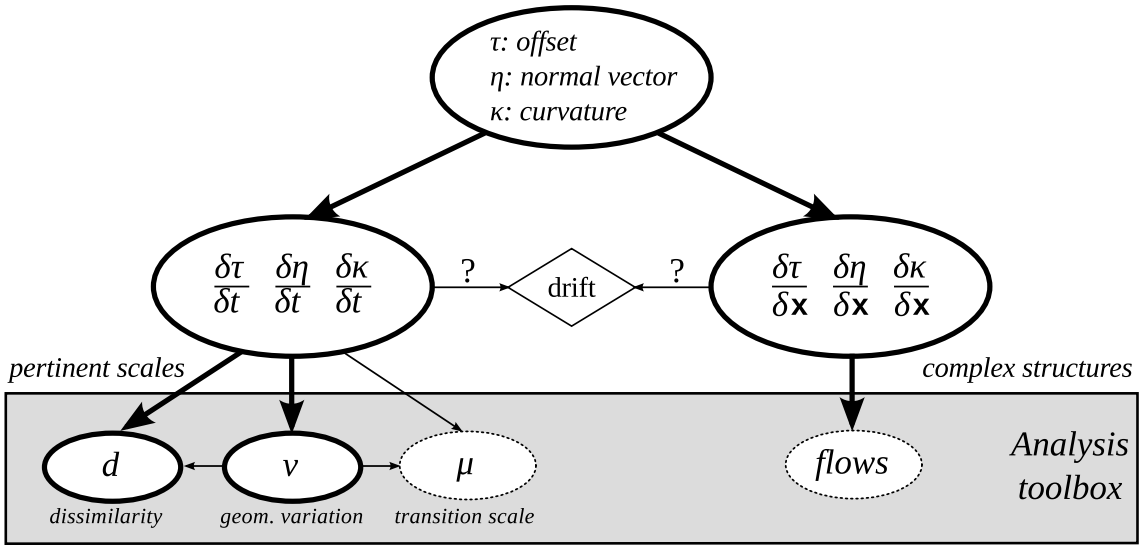


Figure 6.1: **Synthetic view of the GLS framework**. In bold: items presented in this thesis. Dashed: future work directions.

interval and followed by important variations. This behavior has been detected in all of our experiments, and seems correlated with the scales detected by the SIFT method in the case of the 2D curves analysis. We plan to study this behavior more deeply in future work, and to use the convolution operator μ in concrete application cases, such as finding an adaptive bandwidth for surface reconstruction.

We have also observed that our fitting procedure involves a drift effect. This means that it takes into account the strongest nearby feature. The result is that the description of the features is drifting in scale-space: the farthest a point from the center of the feature, the earlier (in scale) it is detected by μ. This behavior strongly motivates a study of the descriptor variations when the evaluation point is moved. Indeed, a better comprehension of the drift effect could help to anticipate it in order to produce a more robust point dissimilarity measure.

In a way similar to the analysis of the scale variations of our descriptor, we plan to work on its spatial differentiation in order to explicitly extract *complex structures*. We have shown that the involved derivatives can be computed analytically. Starting from the partial derivatives of τ, η, and κ, we have proposed to build variation flows, one for each parameter independently, or by combining them. These variations are related to other differential properties: a gradient function defined on the surface (using $\delta\tau$), the principal curvatures (using $\delta\eta$), and the mean curvature variation (using $\delta\kappa$).

Semi-automatic matching

The semi-automatic interaction loop presented in Chapter 5 defines the guidelines to elaborate reassembly systems for objects that are broken into a few fragments. The key idea is to combine archaeological knowledge and algorithmic accuracy. Starting from a standard interaction loop, we give the user the possibility to define some input to guide the reassembly by an adapted interaction technique. Then, a real-time geometric matching kernel combines this information and its incorporated analysis to match the fragments. The system

provides feedback generated from the output of the matching process to help refine the input, or validate the reassembly. This concept has been elaborated in a close collaboration with researchers in archeology.

We presented two implementations of this loop. A first implementation uses a tangible interaction technique. The idea is to manipulate by hand two fragments using tracked props, in order to define an input position for the matching process. We detect the potential contact surface of each of the two fragments in function of their respective distance, by using an acceleration technique inspired from collision detection. The second implementation is based on the definition of reassembly constraints by using a multi-touch surface. These constraints define both the relative input positions and the set of points that are taken as input for the matching process, and their relation.

In both implementations of the system, we use the same matching kernel, based on a real-time ICP variant. We combine both the normal and position criteria to filter the input points, and we ensure temporal coherence by using a smooth weighting function. This is required to avoid popping effects in case of slight changes of the user input. Considering the feedback provided to the user, we display the aligned configuration in a dedicated visualization window or with alpha-blending, respectively for the multi-touch and the tangible setups. We also display the ICP error thanks to a dedicated graphical indicator. We have shown the relevance of both the tangible and multi-touch setups by the results of informal user studies, and by our own experiments.

We plan to improve our concrete implementations of this formalism by working on its three components: the interaction technique, the matching, and the feedback.

The multi-touch setup has been developed in collaboration with CHI researchers. We plan to perform more formal studies on its relevance, for example by trying to find answers to the following questions:

- Is the multiview setup a good solution to manipulate two fragments at the same time?
- How to efficiently define multiple constraints, such as constraints on the continuity and on the contact surface?
- Which metaphor should be used for the multi-scale feature selection?

We also plan to study the influence of haptic feedback for the tangible setup.

Considering the matching process, we plan to work on the integration of new constraints in the minimization process. Indeed, it is not relevant to consider pairwise distance minimization between fragments, because the original contact surfaces have been eroded. A promising direction is to add continuity constraints between surface sheets or features represented as 3D lines.

In order to improve the feedback provided to the user, we plan to add some local indicators in the 3D scene, in order to highlight the fragments' shapes, the user input (e.g. the selected features and defined constraints), and the matching output (e.g. the local minimization error).

Future research directions

The perspectives brought by this thesis are exciting because they propose to combine, on the one hand, a theoretical and general analysis framework, and on the other hand, practical systems specifically designed for a given application field: Cultural Heritage.

We plan to integrate our multi-scale analysis framework in the interaction loop. Indeed, we believe that our semi-automatic conception is naturally compatible with the GLS framework: the parameter-less part of the framework generates a low level analysis that may then assist the user to perform specific tasks involved in the matching process. For example, our analysis toolbox could help to perform advanced multi-scale feature selection, select adapted input samples or define continuity constraints for the reassembly of eroded fragments. The matching algorithm itself may benefit from the analysis, for example by using the descriptor during the corresponding point research, with respect to a local user-specified evaluation scale. It may also be used as a basis to adaptively re-sample fragments in a multi-resolution ICP scheme [Jos02]. The flexibility of this approach is that our analysis is associated with multiple tools, yielding new degrees of freedom to parametrize the reassembly. A last topic is the improvement of the accuracy and the quantity of information that can be transmitted from the system to the user. For example, we plan to use new multi-scale expressive rendering techniques to help the user better understand the fragment's geometry [VPB+10]. This could be done independently for each fragment, but we believe that a challenging topic is to transmit information that characterizes the geometry with respect to the current reassembly hypothesis.

Besides the fragment reassembly, we also plan to use our multi-scale analysis to help archaeologists analyze virtual objects. For example, they usually need to compute slices on objects: our approach could be used to compute an adaptive bandwidth in order to reconstruct the 2D slice with an implicit reconstruction [GG07]. Another interesting topic is related to orthophotos generated from planar surfaces. We plan to extract engraved patterns at multiple scales from our variation flow, and potentially characterize the shape of the tool that was originally used for the engraving.

Our analysis can also be used in other application cases. For example, in some methods of 2D curves morphing, a first step is to find correspondences between the initial and the final poses [BBA09]. We plan to use our multi-scale dissimilarity to define correspondences at different scales, and thus provide a control on which features should be interpolated.

The MLS formulation is also an advantage for the development of point-based editing tools [ZPKG02]. We plan to use our GLS analysis to detect pertinent scales and adaptively decompose a 3D object in multiple surface approximation layers, a powerful multi-scale representation for point-based modelling [PKG06].

Finally, we are convinced that the work presented in this thesis represents a unified and versatile framework for the development of various powerful tools to analyze, edit, and visualize point-based objects regardless of their complexity.

Index

Appendix A

GLS closed-form fitting differentiation

In this appendix, we present the successive steps that are required to differentiate our descriptor with respect to the scale t: differentiation of the weighting function (Eq. 4.2), closed-form fitting formula (Eq. 4.3), normalization (Eq. 4.4) and reparametrization (Eq. 4.5).

Weighting

$$\frac{\delta w_i}{\delta t}(t) = \frac{4\|\mathbf{p}_i - \mathbf{p}\|^2}{t^3}\left(\frac{\|\mathbf{p}_i - \mathbf{p}\|^2}{t^2} - 1\right), \tag{A.1}$$

Closed-form fitting formula

Starting from the Equation 4.3, we can define

$$\begin{aligned} u_q &= \frac{1}{2}\frac{X}{Y} \\ \frac{\delta u_q}{\delta t} &= \frac{1}{2}\frac{Y * \frac{\delta X}{\delta t} - X * \frac{\delta Y}{\delta t}}{Y * Y} \end{aligned}$$

with

$$\begin{aligned} X &= \sum w_i \overline{\mathbf{q}}_i^T \mathbf{n}_i - \sum \tilde{w}_i \overline{\mathbf{q}}_i^T \sum w_i \mathbf{n}_i \\ Y &= \sum w_i \overline{\mathbf{q}}_i^T \overline{\mathbf{q}}_i - \sum \tilde{w}_i \overline{\mathbf{q}}_i^T \sum w_i \overline{\mathbf{q}}_i \\ \frac{\delta X}{\delta t} &= \sum \frac{\delta w_i}{\delta t} \overline{\mathbf{q}}_i^T \mathbf{n}_i - \left(\sum \frac{\delta w_i}{\delta t} \overline{\mathbf{q}}_i^T \sum \tilde{w}_i n_i + \sum \tilde{w}_i \overline{\mathbf{q}}_i^T \sum \frac{\delta w_i}{\delta t} n_i - \sum \frac{\delta w_i}{\delta t} \sum \tilde{w}_i \overline{\mathbf{q}}_i^T \sum \tilde{w}_i \mathbf{n}_i\right) \\ \frac{\delta Y}{\delta t} &= \sum \frac{\delta w_i}{\delta t} \overline{\mathbf{q}}_i^T \overline{\mathbf{q}}_i - 2\sum \frac{\delta w_i}{\delta t} \overline{\mathbf{q}}_i^T \sum \tilde{w}_i \overline{\mathbf{q}}_i - \sum \frac{\delta w_i}{\delta t} \sum w_i \overline{\mathbf{q}}_i^T \sum w_i \overline{\mathbf{q}}_i \end{aligned}$$

We ca now define

$$\begin{aligned} \frac{\delta \mathbf{u}_\ell}{\delta t} &= \frac{\sum \frac{\delta w_i}{\delta t} n_i - 2\left(u_q \sum \frac{\delta w_i}{\delta t} \overline{\mathbf{q}}_i + \frac{\delta u_q}{\delta t} \sum w_i \overline{\mathbf{q}}_i\right) - \sum \frac{\delta w_i}{\delta t} \mathbf{u}_\ell}{\sum w_i} \\ \frac{\delta u_c}{\delta t} &= -\frac{\left(\frac{\delta \mathbf{u}_\ell}{\delta t}\right)^T \sum w_i \overline{\mathbf{q}}_i + \frac{\delta u_q}{\delta t} \sum w_i \overline{\mathbf{q}}_i^T \overline{\mathbf{q}}_i + \mathbf{u}_\ell^T \sum \frac{\delta w_i}{\delta t} \overline{\mathbf{q}}_i + u_q \sum \frac{\delta w_i}{\delta t} \overline{\mathbf{q}}_i^T \overline{\mathbf{q}}_i + u_c \sum \frac{\delta w_i t}{\delta t}}{\sum w_i}. \end{aligned}$$

Normalization

We can differentiate the Pratt norm as $\sqrt{2\frac{\delta \mathbf{u}_\ell}{\delta t}^T \mathbf{u}_\ell - 4\frac{\delta u_c}{\delta t}u_q - 4u_c\frac{\delta u_q}{\delta t}}$ and thus differentiate the Equation 4.4 as

$$\frac{\delta \hat{\mathbf{u}}}{\delta t} = \begin{bmatrix} \frac{\delta \hat{u}_c}{\delta t} & \frac{\delta \hat{\mathbf{u}}_\ell}{\delta t} & \frac{\delta \hat{u}_q}{\delta t} \end{bmatrix}^T = \frac{\|\mathbf{u}_\ell\|^2 - 4u_c u_q \frac{\delta \mathbf{u}}{\delta t} - \frac{1}{2}\mathbf{u}\frac{2\frac{\delta \mathbf{u}_\ell}{\delta t}^T \mathbf{u}_\ell - 4\frac{\delta u_c}{\delta t}u_q - 4u_c\frac{\delta u_q}{\delta t}}{\sqrt{\|\mathbf{u}_\ell\|^2 - 4u_c u_q}}}{2\frac{\delta \mathbf{u}_\ell}{\delta t}^T \mathbf{u}_\ell - 4\frac{\delta u_c}{\delta t}u_q - 4u_c\frac{\delta u_q}{\delta t}}.$$

Reparametrization

We can now compute our descriptor. We can first differentiate Equation 4.6:

$$\frac{\delta \tau}{\delta t} = s_{\frac{\delta \hat{\mathbf{u}}}{\delta t}}(0) = \frac{\delta \hat{\mathbf{u}}_c}{\delta t}. \tag{A.2}$$

The, in order to compute $\boldsymbol{\eta}$, we have to differentiate Equation 4.7 to obtain:

$$\frac{\delta \boldsymbol{\eta}}{\delta t} = \frac{\frac{\delta \hat{\mathbf{u}}_\ell}{\delta t}}{\|\hat{\mathbf{u}}_\ell\|} - \frac{\hat{\mathbf{u}}_\ell \hat{\mathbf{u}}_\ell^T \frac{\delta \hat{\mathbf{u}}_\ell}{\delta t}}{\|\hat{\mathbf{u}}_\ell\|^3} \tag{A.3}$$

Finally, the last parameter derivative is given by:

$$\frac{\delta \kappa}{\delta t} = 2\frac{\delta \hat{u}_q}{\delta t}. \tag{A.4}$$

Appendix B

Noisy 2D curve generation

Source code used to generate 2D curves in Figures 4.10 and 4.25.

```
#!/usr/bin/env python
"""
This 2D curve is disturbed using noise on
- positions,
- normals+positions,
- normals

Generated curve is printed as
x y nx ny
 """
__author__      = "Nicolas Mellado"
__email__       = "nicolas.mellado@inria.fr"

import random
import math as m

#parameters
nbElement       = 1440
radius          = 100
noiseMax        = 0.10 * radius
normalNoiseMax = 0.2
sinusDispl      = 0.12 * radius

# constants
pi          = 3.14159265358979323846
pi2         = 2*pi
angleInRadian = 2.0*pi/float(nbElement)

coord = []

# generate inital shape
for i in range (nbElement):
  angle = i*angleInRadian

  # combinaison of a circle
  x  = m.cos(angle) * radius
  y  = m.sin(angle) * radius
  nx = m.cos(angle)
  ny = m.sin(angle)

  # and a sinus as displacement
  displ  = m.sin(4.*angle) * sinusDispl
  x     +=  nx * (displ)
  y     +=  ny * (displ)

  coord += [x,y]

# create noise: position, position+normals, normals
for i in range (nbElement-1):
  # get coordinate back
  x  = coord[2*i]
  y  = coord[2*i+1]
  nx  = coord[2*(i+1)+1] - coord[2*i+1]
  ny  = - (coord[2*(i+1)] - coord[2*i] )

  for noiseType in range (3):
    #setup
```

```
    sift        = (3+noiseType)*((nbElement-1) / 4.0)
    posNoise    = noiseType > 0
    normalNoise = noiseType < 2

    # compute weight in function of current angle
    angle = i*angleInRadian + sift
    while angle >= pi2:
      angle -= pi2
    w = (angle-pi) / (pi)
    w *= 4.0

    # apply noise only where it's needed
    if(w<1 and w >-1):
            w = w * w - 1
      w = w * w * w * w * w * w * w * w * w

            # displacement
      if(posNoise):
        noise = w * ( 2.0 * noiseMax * float(random.random()) - noiseMax)
        x     +=  nx * (noise)
        y     +=  ny * (noise)

      # normal noise
      if(normalNoise):
        nx += w * ( 2.0 * normalNoiseMax * float(random.random()) - normalNoiseMax)
        ny += w * ( 2.0 * normalNoiseMax * float(random.random()) - normalNoiseMax)

        nnorm = nx*nx + ny*ny
        nx    /= nnorm
        ny    /= nnorm
  print x, y, nx, ny
```

Bibliography

[AA04] Marc Alexa and Anders Adamson. On normals and projection operators for surfaces defined by point sets. In *In Eurographics Symp. on Point-Based Graphics*, pages 149–155, 2004.

[AB98] Nina Amenta and Marshall Bern. Surface reconstruction by voronoi filtering. pages 39–48, 1998.

[ABCO+03] Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T. Silva. Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 9(1):3–15, January 2003.

[ABG+12] Lucas Ammann, Pascal Barla, Gaël Guennebaud, Xavier Granier, and Patrick Reuter. Surface relief analysis for illustrative shading. *Comp. Graph. Forum*, 31(4):1481–1490, June 2012.

[AK04] Nina Amenta and Yong Joo Kil. Defining point-set surfaces. *ACM Trans. Graph.*, 23(3):264–270, August 2004.

[BB11] M.M. Bronstein and A.M. Bronstein. Shape recognition with spectral distances. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(5):1065 –1071, may 2011.

[BBA09] William Baxter, Pascal Barla, and Ken Anjyo. N-way morphing for 2d animation. *Computer Animation Virtual Worlds*, 20(2-3):79–87, June 2009.

[BBGO11] Alexander M. Bronstein, Michael M. Bronstein, Leonidas J. Guibas, and Maks Ovsjanikov. Shape google: Geometric words and expressions for invariant shape retrieval. *ACM Trans. Graph.*, 30(1):1:1–1:20, February 2011.

[BBK+10] Alexander M. Bronstein, Michael M. Bronstein, Ron Kimmel, Mona Mahmoudi, and Guillermo Sapiro. A gromov-hausdorff framework with diffusion geometry for topologically-robust non-rigid shape matching. *Int. J. Comput. Vision*, 89(2-3):266–286, September 2010.

[BCOS01] Marcelo Bertalmío, Li-Tien Cheng, Stanley Osher, and Guillermo Sapiro. Variational problems and partial differential equations on implicit surfaces. *Journal of Computational Physics*, 174(2):759 – 780, 2001.

[BDL95] Gérard Blais and Martin D. Levine. Registering multiview range data to create 3d computer objects. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(8):820–824, August 1995.

[BEB12] Tyson Brochu, Essex Edwards, and Robert Bridson. Efficient geometrically exact continuous collision detection. *ACM Trans. Graph.*, 31(4):96:1–96:7, July 2012.

[Ben75] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, September 1975.

[BJ86] Paul J. Besl and Ramesh C. Jain. Invariant surface characteristics for 3d object recognition in range images. *Comput. Vision Graph. Image Process.*, 33(1):33–80, January 1986.

[BK10] M.M. Bronstein and I. Kokkinos. Scale-invariant heat kernel signatures for non-rigid shape recognition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1704 –1711, june 2010.

[BKP+10] Mario Botsch, Leif Kobbelt, Mark Pauly, Pierre Alliez, and Bruno Lévy. *Polygon Mesh Processing*. AK Peters / CRC Press, September 2010.

[BKW+08] Michael D. Breitenstein, Daniel Küttel, Thibaut Weise, Luc J. Van Gool, and Hanspeter Pfister. Real-time face pose estimation from single range images. In *CVPR'08*, pages –1–1, 2008.

[Bli82] James F. Blinn. A generalization of algebraic surface drawing. *ACM Trans. Graph.*, 1(3):235–256, July 1982.

[Blo97] Jules Bloomenthal. *Introduction to Implicit Surfaces, First Edition (The Morgan Kaufmann Series in Computer Graphics)*. Morgan Kaufmann, August 1997.

[BM92] Paul J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(2):239–256, February 1992.

[BM12] Alexandre Boulch and Renaud Marlet. Fast and robust normal estimation for point clouds with sharp features. *Comp. Graph. Forum*, 31(5):1765–1774, August 2012.

[BMP02] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(4):509–522, April 2002.

[BS91] Jules Bloomenthal and Ken Shoemake. Convolution surfaces. *SIGGRAPH Comput. Graph.*, 25(4):251–256, July 1991.

[BS97] Raouf Benjemaa and Francis Schmitt. Fast global registration of 3d sampled surfaces using a multi-z-buffer technique. In *Image and Vision Computing*, pages 113–120, 1997.

[BS07] Alexander I. Bobenko and Boris A. Springborn. A discrete laplace–beltrami operator for simplicial surfaces. *Discrete Comput. Geom.*, 38(4):740–756, December 2007.

[BS12] Matthew Berger and Claudio T. Silva. Nonrigid Matching of Undersampled Shapes via Medial Diffusion. *Computer Graphics Forum*, 31(5):1587–1596, 2012.

[BSF02] Kim L. Boyer, Ravi Srikantiah, and Patrick J. Flynn. Saliency sequential surface organization for free-form object recognition. *Comput. Vis. Image Underst.*, 88(3):152–188, 2002.

[BTFN+08] Benedict J. Brown, Corey Toler-Franklin, Diego Nehab, Michael Burns, David Dobkin, Andreas Vlachopoulos, Christos Doumas, Szymon Rusinkiewicz, and

Tim Weyrich. A system for high-volume acquisition and matching of fresco fragments: Reassembling Theran wall paintings. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 27(3), August 2008.

[CBC+01] Jonathan C. Carr, Richard K. Beatson, Jon B. Cherrie, Tim J. Mitchell, W. Richard Fright, Bruce C. McCallum, and Tim R. Evans. Reconstruction and representation of 3D objects with radial basis functions. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques - SIGGRAPH '01*, pages 67–76, 2001.

[CBR+11] Antonio García Castañeda, Benedict J. Brown, Szymon Rusinkiewicz, Thomas A. Funkhouser, and Tim Weyrich. Global consistency in the automatic assembly of fragmented artefacts. In Franco Niccolucci, Matteo Dellepiane, SebastiÃąn PeÃśa Serna, Holly E. Rushmeier, and Luc J. Van Gool, editors, *VAST*, pages 73–80. Eurographics Association, 2011.

[CCFM08] U. Castellani, M. Cristani, S. Fantoni, and V. Murino. Sparse points matching by combining 3d mesh saliency with statistical descriptors. *Computer Graphics Forum*, 27(2):643–652, 2008.

[CCSLT09] F. Chazal, D. Cohen-Steiner, A. Lieutier, and B. Thibert. Stability of curvature measures. In *Proceedings of the Symposium on Geometry Processing*, SGP '09, pages 1485–1496, Aire-la-Ville, Switzerland, Switzerland, 2009. Eurographics Association.

[CG10] Chong Chen and Xu Guoliang. Construction of geometric partial differential equations for level sets. *Journal of Computational Mathematics*, 28(1):105, January 2010.

[CGR+04] U. Clarenz, M. Griebel, M. Rumpf, M. A. Schweitzer, and A. Telea. Feature sensitive multiscale editing on surfaces. *Vis. Comput.*, 20(5):329–343, July 2004.

[Che11] A Multigrid Fluid Pressure Solver Handling Separating Solid Boundary Conditions. *IEEE Transactions on Visualization and Computer Graphics*, 1(8):83–90, 2011.

[Cig12] Paolo Cignoni. MeshLab. http://meshlab.sourceforge.net/, 2012 (accessed September 25, 2012).

[CJ97] Chin Seng Chua and Ray Jarvis. Point signatures: A new representation for 3d object recognition. *Int. J. Comput. Vision*, 25(1):63–85, October 1997.

[CL06] Ronald R. Coifman and Stéphane Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5–30, July 2006.

[CM91] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image Vision Comput.*, 10(3):145–155, 1991.

[CP03] F. Cazals and M. Pouget. Estimating differential quantities using polynomial fitting of osculating jets. In *Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, SGP '03, pages 177–187, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[CPG09] Gregory Cipriano, George N Phillips, and Michael Gleicher. Multi-scale surface descriptors. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1201–8, 2009.

[CSAD04] David Cohen-Steiner, Pierre Alliez, and Mathieu Desbrun. Variational shape approximation. *ACM Trans. Graph.*, 23(3):905–914, August 2004.

[CSSK02] D. Chetverikov, D. Svirko, D. Stepanov, and Pavel Krsek. The trimmed iterative closest point algorithm. In *Proceedings of the 16 th International Conference on Pattern Recognition (ICPR'02) Volume 3 - Volume 3*, ICPR '02, pages 30545–, Washington, DC, USA, 2002. IEEE Computer Society.

[CWA+01] David B. Cooper, Andrew Willis, Stuart Andrews, Jill Baker, Yan Cao, Dongjin Han, Kongbin Kang, Weixin Kong, Frederic F. Leymarie, Xavier Orriols, Senem Velipasalar, Eileen L. Vote, Martha S. Joukowsky, Benjamin B. Kimia, David H. Laidlaw, and David Mumford. Assembling virtual pots from 3d measurements of their fragments. In *Proceedings of the 2001 conference on Virtual reality, archeology, and cultural heritage*, VAST '01, pages 241–254, New York, NY, USA, 2001. ACM.

[DA12] Elric Delord and ArcheoTransfert. SynAps. http://synaps.cnrs.fr, 2012 (accessed September 25, 2012).

[DB02] Ioannis Douros and Bernard Buxton. Three-Dimensional Surface Curvature Estimation using Quadric Surface Patches. In *Scanning 2002*, May 2002.

[DG06] Tamal K. Dey and Samrat Goswami. Provable surface reconstruction from noisy samples. *Comput. Geom. Theory Appl.*, 35(1):124–141, August 2006.

[dGGV08] Fernando de Goes, Siome Goldenstein, and Luiz Velho. A hierarchical segmentation of articulated bodies. In *Proceedings of the Symposium on Geometry Processing*, SGP '08, pages 1349–1356, Aire-la-Ville, Switzerland, Switzerland, 2008. Eurographics Association.

[Dij59] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, December 1959.

[DK12] T. Darom and Y. Keller. Scale-invariant features for 3-d mesh models. *Image Processing, IEEE Transactions on*, 21(5):2758 –2769, may 2012.

[DLS05] T.K. Dey, G. Li, and J. Sun. Normal estimation for point clouds: a comparison study for a Voronoi based method. *Proceedings Eurographics/IEEE VGTC Symposium Point-Based Graphics, 2005.*, pages 39–46, 2005.

[DWJM98] Chitra Dorai, Gang Wang, Anil K. Jain, and Carolyn Mercer. Registration and integration of multiple object views for 3d model construction. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(1):83–89, January 1998.

[Eul67] Leonard Euler. Recherches sur la courbure des surfaces. *Memoires de l'academie des sciences de Berlin*, 16:119–143, 1767.

[FBMB10] Gregory Flitton, Toby Breckon, and Najla Megherbi Bouallagu. Object recognition using 3d sift in complex ct volumes. In *Proceedings of the British Machine Vision Conference*, pages 11.1–11.12. BMVA Press, 2010. doi:10.5244/C.24.11.

[FHK+04] Andrea Frome, Daniel Huber, Ravi Kolluri, Thomas Bülow, and Jitendra Malik. Recognizing objects in range data using regional point descriptors. In *European conference on computer vision*, pages 224–237, 2004.

[FJ89] P.J. Flynn and A.K. Jain. On reliable curvature estimation. In *Computer Vision and Pattern Recognition, 1989. Proceedings CVPR '89., IEEE Computer Society Conference on*, pages 110 –116, jun 1989.

[FKN80] Henry Fuchs, Zvi M. Kedem, and Bruce F. Naylor. On visible surface generation by a priori tree structures. *SIGGRAPH Comput. Graph.*, 14(3):124–133, July 1980.

[FP10] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(8):1362–1376, August 2010.

[Fra82] Richard Franke. Scattered data interpolation: tests of some methods. *Mathematics of Computation*, 38:181–181, 1982.

[FS06] T. Funkhouser and P. Shilane. Partial matching of 3d shapes with priority-driven search. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, SGP '06, pages 131–142, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[FSTF+11] Thomas Funkhouser, Hijung Shin, Corey Toler-Franklin, Antonio García Castañeda, Benedict Brown, David Dobkin, Szymon Rusinkiewicz, and Tim Weyrich. Learning how to match fresco fragments. *J. Comput. Cult. Herit.*, 4(2):7:1–7:13, November 2011.

[FW11] H. Fadaifard and G. Wolberg. Multiscale 3d feature extraction and matching. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2011 International Conference on*, pages 228 –235, may 2011.

[FWSB07] Clifton Forlines, Daniel Wigdor, Chia Shen, and Ravin Balakrishnan. Direct-touch vs. mouse input for tabletop displays. pages 647–656, 2007.

[GBPP07] M. Germann, M.D. Breitenstein, H. Pfister, and In Kyu Park. Automatic pose estimation for range images on the gpu. In *3-D Digital Imaging and Modeling, 2007. 3DIM '07. Sixth International Conference on*, pages 81 –90, aug. 2007.

[GCO06] Ran Gal and Daniel Cohen-Or. Salient geometric features for partial shape matching and similarity. *ACM Trans. Graph.*, 25(1):130–150, January 2006.

[GG06] Timothy Gatzke and Cindy Grimm. Estimating curvature on triangular meshes. *International Journal of Shape Modeling*, 12(1):1–29, June 2006. How best to compare curvature metrics on meshes.

[GG07] Gaël Guennebaud and Markus Gross. Algebraic point set surfaces. *ACM Trans. Graph.*, 26(3), 2007.

[GGG08] Gaël Guennebaud, Marcel Germann, and Markus Gross. Dynamic sampling and rendering of algebraic point set surfaces. *Computer Graphics Forum*, 27(2), 2008.

[GGGZ05] Timothy Gatzke, Cindy Grimm, Michael Garland, and Steve Zelinka. Curvature maps for local shape comparison. In *Proceedings of the International Conference on Shape Modeling and Applications 2005*, SMI '05, pages 246–255, Washington, DC, USA, 2005. IEEE Computer Society.

[GL12] Andrea Giachetti and Christian Lovato. Radial Symmetry Detection and Shape Characterization with the Multiscale Area Projection Transform. *Computer Graphics Forum*, 31(5):1669–1678, 2012.

[GMGP05] Natasha Gelfand, Niloy J. Mitra, Leonidas J. Guibas, and Helmut Pottmann. Robust global registration. In *Proceedings of the third Eurographics symposium on Geometry processing*, SGP '05, Aire-la-Ville, Switzerland, Switzerland, 2005. Eurographics Association.

[Gol05] Ron Goldman. Curvature formulas for implicit curves and surfaces. *Comput. Aided Geom. Des.*, 22(7):632–658, October 2005.

[GP07] Markus Gross and Hanspeter Pfister. *Point-based Graphics*, volume 28. Morgan Kauffman, 2007.

[GRB94] Guy Godin, Marc Rioux, and Rejean Baribeau. Three-dimensional registration using range and intensity information. volume 2350, pages 279–290. SPIE, 1994.

[GSCO07] Ran Gal, Ariel Shamir, and Daniel Cohen-Or. Pose-oblivious shape signature. *IEEE Transactions on Visualization and Computer Graphics*, 13(2):261–271, March 2007.

[GW11] Afzal Godil and Asim Imdad Wagan. Salient local 3d features for 3d shape retrieval. *CoRR*, abs/1105.2796, 2011.

[Hai06] Isabelle Hairy. Le phare d'alexandrie, concentré de géométrie. *La Recherche*, 394:44–50, 2006.

[HDD+92] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. *ACM SIGGRAPH Computer Graphics*, 26(2):71–78, July 1992.

[HFG+06] Qi-Xing Huang, Simon Flöry, Natasha Gelfand, Michael Hofer, and Helmut Pottmann. Reassembling fractured objects by geometric matching. *ACM Trans. Graphics*, 25(3):569–578, 2006.

[HID95] Martial Hebert, Katsushi Ikeuchi, and Hervé Delingette. A spherical representation for recognition of free-form surfaces. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(7):681–690, July 1995.

[HLSR08] Markus Hadwiger, Patric Ljung, Christof Rezk Salama, and Timo Ropinski. Advanced illumination techniques for gpu volume raycasting. In *ACM SIGGRAPH ASIA 2008 courses*, SIGGRAPH Asia '08, pages 1:1–1:166, New York, NY, USA, 2008. ACM.

[HPW06] Klaus Hildebrandt, Konrad Polthier, and Max Wardetzky. On the convergence of metric and geometric properties of polyhedral surfaces. *Geometriae Dedicata*, 123(1):89–112, 2006.

[HT10] Gur Harary and Ayellet Tal. 3d euler spirals for 3d curve completion. In *Proceedings of the 2010 annual symposium on Computational geometry*, SoCG '10, pages 393–402, New York, NY, USA, 2010. ACM.

[IJ85] D.J. Ittner and A.K. Jain. 3-d surface discrimination from local curvature measures. In *Computer Vision and Pattern Recognition, 1985. Proceedings CVPR '85., IEEE Computer Society Conference on*, pages 119–123, 1985.

[IT11a] Arik Itskovich and Ayellet Tal. Semantic 3d media and content: Surface partial matching and application to archaeology. *Comput. Graph.*, 35(2):334–341, April 2011.

[IT11b] Arik Itskovich and Ayellet Tal. Surface partial matching and application to archaeology. *Computers & Graphics*, 35(2):334–341, 2011.

[JH99] Andrew E. Johnson and Martial Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(5):433–449, May 1999.

[JH02] Timothée Jost and Heinz Hügli. A Multi-Resolution Scheme ICP Algorithm for Fast Shape Registration. In *3D Data Processing Visualization and Transmission*, pages 540–543, 2002.

[Jos02] Timothée Jost. *Fast Geometric Matching for shape registration*. PhD thesis, Université de Neuchatel, 2002.

[KB04] Leif Kobbelt and Mario Botsch. A survey of point-based techniques in computer graphics. *Comput. Graph.*, 28(6):801–814, December 2004.

[KBH06] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, SGP '06, pages 61–70, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[KBS00] Leif P. Kobbelt, Thilo Bareuther, and Hans-Peter Seidel. Multiresolution shape deformations for meshes with dynamic vertex connectivity. *Computer Graphics Forum*, 19(3):249–260, 2000.

[KK01] Weixin Kong and B.B. Kimia. On solving 2d and 3d puzzles using curve matching. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 2, pages II–583 – II–590 vol.2, 2001.

[KNK03] M. Körtgen, M. Novotni, and R. Klein. 3D shape matching with 3D shape contexts. In *In The 7th Central European Seminar on Computer Graphics*, 2003.

[KS04] Martin Kampel and Robert Sablatnig. On 3d mosaicing of rotationally symmetric ceramic fragments. *ICPR*, 2:265–268, 2004.

[KST08] M. Kolomenkin, I. Shimshoni, and A. Tal. Demarcating curves for shape illustration. *ACM transactions on Graphics, SIGGRAPH ASIA*, 27(5):151:1–9, 2008.

[KTNL05] David Koller, Jennifer Trimble, Tina Najbjerg, and Natasha Gelfand Marc Levoy. Fragments of the city: Stanford's digital forma urbis romae project. In

Proc. of the Third Williams Symposium on Classical Architecture, Journal of Roman Archaeology suppl., 2005.

[LBB11] Roee Litman, Alexander M. Bronstein, and Michael M. Bronstein. Diffusion-geometric maximally stable component detection in deformable shapes. *Computers & Graphics*, 35(3):549 – 560, 2011. Shape Modeling International (SMI) Conference 2011.

[Lev98] David Levin. The approximation power of moving least-squares. *Math. Comput.*, 67(224):1517–1531, October 1998.

[Lev03] David Levin. Mesh-independent surface interpolation. *Geometric Modeling for Scientific Visualization*, 3, 2003.

[LG05] Xinju Li and Igor Guskov. Multi-scale features for approximate alignment of point-based surfaces. In *Proceedings of the third Eurographics symposium on Geometry processing*, SGP '05, Aire-la-Ville, Switzerland, Switzerland, 2005. Eurographics Association.

[Li210] Robust normal estimation for point clouds with sharp features. *Computers & Graphics*, 34(2):94–106, 2010.

[Lin93] Ming Chieh Lin. *Efficient collision detection for animation and robotics*. PhD thesis, University of California, Berkeley, 1993. Chair-Canny,, John F.

[Lin94] Tony Lindeberg. *Scale-Space Theory in Computer Vision*. Norwell, MA, USA, 1994.

[Lin98] Tony Lindeberg. Feature detection with automatic scale selection. *Int. J. Comput. Vision*, 30(2):79–116, November 1998.

[LKF12] Tianqiang Liu, Vladimir G. Kim, and Thomas Funkhouser. Finding Surface Correspondences Using Symmetry Axis Curves. *Computer Graphics Forum*, 31(5):1607–1616, 2012.

[LLKR07] Yu-Shen Liu, Min Liu, Daisuke Kihara, and Karthik Ramani. Salient critical points for meshes. In *Proceedings of the 2007 ACM symposium on Solid and physical modeling*, SPM '07, pages 277–282, New York, NY, USA, 2007. ACM.

[LLWZ12] Jian Liang, Rongjie Lai, Tsz Wai Wong, and Hongkai Zhao. Geometric understanding of point clouds using laplace-beltrami operator. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 214 –221, june 2012.

[Low99] David G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2*, ICCV '99, pages 1150–, Washington, DC, USA, 1999. IEEE Computer Society.

[LSS08] Jörg Liebelt, Cordelia Schmid, and Klaus Schertler. Viewpoint-independent object class detection using 3d feature maps. In *Conference on Computer Vision & Pattern Recognition*, jun 2008.

[Lux12] Luxology. Modo 601. http://www.luxology.com/modo/, 2012 (accessed September 25, 2012).

[LVJ05] Chang Ha Lee, Amitabh Varshney, and David W. Jacobs. Mesh saliency. *ACM Trans. Graph.*, 24(3):659–666, July 2005.

[LW85] Marc Levoy and Turner Whitted. The use of points as a display primitive. Technical Report TR 85-022, University of North Carolina at Chapel Hill, 1985.

[LZSCO09] Rong Liu, Hao Zhang, Ariel Shamir, and Daniel Cohen-Or. A part-aware surface metric for shape analysis. *Computer Graphics Forum*, 28(2):397–406, 2009.

[MB12] Nicolas Mellado and Pascal Barla. Growing Least Squares for the Continuous Analysis of Manifolds in Scale-Space. *Computer Graphics Forum (Proceedings of SGP 2012)*, 31(5), 2012.

[MBB10] Niloy J. Mitra, Alex Bronstein, and Michael Bronstein. Intrinsic regularity detection in 3d geometry. In *Proceedings of the 11th European conference on computer vision conference on Computer vision: Part III*, ECCV'10, pages 398–410, Berlin, Heidelberg, 2010. Springer-Verlag.

[MBO10] A. Mian, M. Bennamoun, and R. Owens. On the repeatability and quality of keypoints for local feature-based 3d object retrieval from cluttered scenes. *Int. J. Comput. Vision*, 89(2-3):348–361, September 2010.

[MCH+06] Siddharth Manay, Daniel Cremers, Byung-Woo Hong, Anthony Yezzi, and Stefano Soatto. Integral invariants and shape matching. In Hamid Krim and Anthony Yezzi, editors, *Statistics and Analysis of Shapes*, Modeling and Simulation in Science, Engineering and Technology, pages 137–166. BirkhÃd'user Boston, 2006.

[MDSB02] Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H. Barr. Discrete Differential-Geometry Operators for Triangulated 2-Manifolds. In *VisMath '02*, Berlin, Germany, 2002.

[MDTK06] Blerim Mustafa, Danco Davcev, Vladimir Trajkovik, and Slobodan Kalajdziski. 3D Object Matching Using Spherical Mapping. *IECON 2006 - 32nd Annual Conference on IEEE Industrial Electronics*, pages 3450–3454, November 2006.

[MFK+10] C. Maes, T. Fabry, J. Keustermans, D. Smeets, P. Suetens, and D. Vandermeulen. Feature detection on 3d face surfaces for pose normalisation and recognition. In *Biometrics: Theory Applications and Systems (BTAS), 2010 Fourth IEEE International Conference on*, pages 1 –6, sept. 2010.

[MGP06] Niloy J. Mitra, Leonidas J. Guibas, and Mark Pauly. Partial and approximate symmetry detection for 3d geometry. *ACM Trans. Graph.*, 25(3):560–568, July 2006.

[MH80] D. Marr and E. Hildreth. Theory of edge-detection. In *Proc. of the Royal Society of London*, pages 187–217, 1980.

[MKY01] F. Mokhtarian, N. Khalili, and P. Yuen. Curvature computation on free-form 3-d meshes at multiple scales. *Comput. Vis. Image Underst.*, 83:118–139, 2001.

[MM86] Farzin Mokhtarian and Alan Mackworth. Scale-based description and recognition of planar curves and two-dimensional shapes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-8(1):34 –43, 1986.

[MN03] Niloy J. Mitra and An Nguyen. Estimating surface normals in noisy point cloud data. In *Proceedings of the nineteenth annual symposium on Computational geometry*, SCG '03, pages 322–328, New York, NY, USA, 2003. ACM.

[MOG11] Q. Merigot, M. Ovsjanikov, and L.J. Guibas. Voronoi-based curvature and feature estimation from point clouds. *Visualization and Computer Graphics, IEEE Transactions on*, 17(6):743 –756, june 2011.

[MRB09] Nicolas Mellado, Patrick Reuter, and Florent Berthaut. Analyse de l'influence des systèmes de visualisation immersif sur l'assemblage virtuel de fragments en archéologie. In *AFRV 2009 - 4eme Journées de l'Association Francaise de Réalité Virtuelle*, Lyon, 2009.

[MRS09] Nicolas Mellado, Patrick Reuter, and Christophe Schlick. Assemblage d'objets 3D Semi-Automatique basé géométrie en Archéologie. In *Actes des 22èmes Journées de l'Association Française d'Informatique Graphique (AFIG)*, pages 141–148, Arles, 2009.

[MRS10] Nicolas Mellado, Patrick Reuter, and Christophe Schlick. Semi-automatic geometry-driven reassembly of fractured archeological objects. In *Proceedings of VAST 2010: The 11th International Symposium on Virtual Reality, Archaeology and Cultural Heritage*, pages 33–38, Paris, 2010.

[MST10] A. McAdams, E. Sifakis, and J. Teran. A parallel multigrid poisson solver for fluids simulation on large grids. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '10, pages 65–74, Aire-la-Ville, Switzerland, Switzerland, 2010. Eurographics Association.

[MSY96] T. Masuda, K. Sakaue, and N. Yokoya. Registration and integration of multiple range images for 3-d model construction. In *Proceedings of the 1996 International Conference on Pattern Recognition (ICPR '96) Volume I - Volume 7270*, ICPR '96, pages 879–, Washington, DC, USA, 1996. IEEE Computer Society.

[Nea12] Neamedia. http://www.neamedia.fr/en/, 2012 (accessed September 25, 2012).

[NY06] Timothy S. Newman and Hong Yi. A survey of the marching cubes algorithm. *Computers & Graphics*, 30(5):854–879, October 2006.

[OBA+03] Yutaka Ohtake, Alexander Belyaev, Marc Alexa, Greg Turk, and Hans-Peter Seidel. Multi-level partition of unity implicits. *ACM Trans. Graph.*, 22(3):463–470, July 2003.

[OGG09] Cengiz Oztireli, Gaël Guennebaud, and Markus Gross. Feature Preserving Point Set Surfaces based on Non-Linear Kernel Regression. *Computer Graphics Forum*, 28(2):493–501, 2009.

[ON12] Geoffrey Oxholm and Ko Nishino. A flexible approach to reassembling thin artifacts of unknown geometry. *Journal of Cultural Heritage*, (0):–, 2012.

[OT03] Ryutarou Ohbuchi and Tsuyoshi Takei. Shape-similarity comparison of 3d models using alpha shapes. In *Proceedings of the 11th Pacific Conference on Computer Graphics and Applications*, PG '03, pages 293–, Washington, DC, USA, 2003. IEEE Computer Society.

[PAT00] Xavier Pennec, Nicholas Ayache, and Jean-Philippe Thirion. Handbook of medical imaging. chapter Landmark-based registration using features identified through differential geometry, pages 499–513. Academic Press, Inc., Orlando, FL, USA, 2000.

[Pet02] Sylvain Petitjean. A survey of methods for recovering quadrics in triangle meshes. *ACM Comput. Surv.*, 34(2):211–262, June 2002.

[PGBP10] In Kyu Park, Marcel Germann, Michael D. Breitenstein, and Hanspeter Pfister. Fast and automatic object pose estimation for range images on the gpu. *Mach. Vision Appl.*, 21(5):749–766, August 2010.

[PHYK05] Helmut Pottmann, Qi-Xing Huang, Yong-Liang Yang, and Stephan Kölp. Integral invariants for robust geometry processing. Technical report, Geometry Preprint Series, Vienna University of Technology, 2005.

[PKG03] Mark Pauly, Richard Keiser, and Markus H. Gross. Multi-scale feature extraction on point-sampled surfaces. *Computer Graphics Forum*, 22(3):281–290, 2003.

[PKG06] Mark Pauly, Leif P. Kobbelt, and Markus Gross. Point-based multiscale surface representation. *ACM Trans. Graph.*, 25(2):177–193, April 2006.

[Pra87] Vaughan Pratt. Direct least-squares fitting of algebraic surfaces. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '87, pages 145–152, New York, NY, USA, 1987. ACM.

[Pul99] Kari Pulli. Multiview registration for large data sets. In *Proceedings of the 2nd international conference on 3-D digital imaging and modeling*, 3DIM'99, pages 160–168, Washington, DC, USA, 1999. IEEE Computer Society.

[PWHY09] Helmut Pottmann, Johannes Wallner, Qi-Xing Huang, and Yong-Liang Yang. Integral invariants for robust geometry processing. *Comput. Aided Geom. Des.*, 26:37–60, 2009.

[RBBK10] Dan Raviv, Michael M. Bronstein, Alexander M. Bronstein, and Ron Kimmel. Volumetric heat kernel signatures. In *Proceedings of the ACM workshop on 3D object retrieval*, 3DOR '10, pages 39–44, New York, NY, USA, 2010. ACM.

[RDSK06] Mauro R. Ruggeri, Tal Darom, Dietmar Saupe, and Nahum Kiryati. Approximating geodesics on point set surfaces. In *Symposium on Point-Based Graphics (SPBG'06)*, pages 85–93, 2006.

[Reu03] Patrick Reuter. *Reconstruction and rendering of implicit surfaces from large unorganized point sets*. PhD thesis, 2003.

[RHHL02] Szymon Rusinkiewicz, Olaf Hall-Holt, and Marc Levoy. Real-time 3D model acquisition. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 21(3):438–446, July 2002.

[RL01] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the icp algorithm. In *Proceedings of the Third Intl. Conf. on 3D Digital Imaging and Modeling*, pages 145–152, 2001.

[RMG+11] Patrick Reuter, Nicolas Mellado, Xavier Granier, Isabelle Hairy, Robert Vergnieux, and Nadine Couture. Semi-automatic 3D Acquisition and Reassembly of Cultural Heritage: The SeARCH Project, July 2011.

[Rom09] B.M. Romeny. *Front-End Vision and Multi-Scale Image Analysis: Multi-scale Computer Vision Theory and Applications, written in Mathematica*. 2009.

[RRC+07] Patrick Reuter, Guillaume Rivière, Nadine Couture, Nicolas Sorraing, Loïc Espinasse, and Robert Vergnieux. Archeotui - a tangible user interface for the virtual reassembly of fractured archeological objects. In *Proc. of VAST 2007*. Eurographics, 2007.

[RRC+10] Patrick Reuter, Guillaume Riviere, Nadine Couture, Stéphanie Mahut, and Loïc Espinasse. ArcheoTUI - Driving virtual reassemblies with tangible 3D interaction. *Journal on Computing and Cultural Heritage (JOCCH 2010)*, 3(2):1–13, 2010. 13 pages, Article 4.

[SAS07] Paul Scovanner, Saad Ali, and Mubarak Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th international conference on Multimedia*, MULTIMEDIA '07, pages 357–360, New York, NY, USA, 2007. ACM.

[SDG07] R. Synave, P. Desbarats, and S. Gueorguieva. Automated trimmed iterative closest point algorithm. In George Bebis, Richard Boyle, Bahram Parvin, Darko Koracin, Nikos Paragios, Syeda-Mahmood Tanveer, Tao Ju, Zicheng Liu, Sabine Coquillart, Carolina Cruz-Neira, Torsten Müller, and Thom Malzbender, editors, *Advances in Visual Computing*, volume 4842 of *LNCS*, pages 489–498. Springer, 2007.

[SJG05] Limin Shang, Piotr Jasiobedzki, and Michael Greenspan. Discrete pose space estimation to improve icp-based tracking. In *Proceedings of the Fifth International Conference on 3-D Digital Imaging and Modeling*, 3DIM '05, pages 523–530, Washington, DC, USA, 2005. IEEE Computer Society.

[SLR+12] Satti Shahid M., Denis Leon, Florea Ruxandra, Jan Cornelis, Peter Schelkens, and Adrian Munteanu. Optimized Scalable Wavelet-Based Codec Designs for Semi-Regular 3D Meshes. In Dumitru Baleanu, editor, *Advances in Wavelet Theory and Their Applications in Engineering, Physics and Technology*, pages 567–592. Intech edition, 2012.

[SM92] F. Stein and G. Medioni. Structural indexing: Efficient 3-d object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:125–145, 1992.

[SOG09] Jian Sun, Maks Ovsjanikov, and Leonidas Guibas. A concise and provably informative multi-scale signature based on heat diffusion. In *Proc. of the Symposium on Geometry Processing*, pages 1383–1392, 2009.

[Tau95] G. Taubin. Curve and surface smoothing without shrinkage. In *Proc. of the Fifth International Conference on Computer Vision*, 1995.

[TCF09] Roberto Toldo, Umberto Castellani, and Andrea Fusiello. A bag of words approach for 3d object categorization. In *Proceedings of the 4th International Conference on Computer Vision/Computer Graphics CollaborationTechniques*, MIRAGE '09, pages 116–127, Berlin, Heidelberg, 2009. Springer-Verlag.

[TF95] Emanuele Trucco and Robert B. Fisher. Experiments in curvature-based segmentation of range data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:177–182, 1995.

[TFK+09] Barbara Thuswaldner, Simon Flöry, Robert Kalasek, Michael Hofer, Qi-Xing Huang, and Hilke Thür. Digital anastylosis of the octagon in ephesos. *J. Comput. Cult. Herit.*, 2(1):1–27, 2009.

[TG95] Jean-Philippe Thirion and Alexis Gourdon. Computing the differential characteristics of isointensity surface. *Comput. Vis. Image Underst.*, 61(2):190–202, March 1995.

[TL94] Greg Turk and Marc Levoy. Zippered polygon meshes from range images. In *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 311–318, New York, NY, USA, 1994. ACM.

[UGR04] Kazunori Umeda, Guy Godin, and Marc Rioux. Registration of range and color images using gradient constraints and range intensity images. pages 12–15, 2004.

[Ver10] Romain Vergne. *Communication expressive de la forme au travers de l'éclairement et du rendu au trait*. These, Université Sciences et Technologies - Bordeaux I, December 2010.

[vKZHCO11] Oliver van Kaick, Hao Zhang, Ghassan Hamarneh, and Daniel Cohen-Or. A survey on shape correspondence. *Computer Graphics Forum*, 30(6):1681–1707, 2011.

[VPB+10] Romain Vergne, Romain Pacanowski, Pascal Barla, Xavier Granier, and Christophe Schlick. Radiance Scaling for Versatile Surface Enhancement. In *I3D '10: Proc. symposium on Interactive 3D graphics and games*, 2010.

[VVC+11] Romain Vergne, David Vanderhaeghe, Jiazhou Chen, Pascal Barla, Xavier Granier, and Christophe Schlick. Implicit Brushes for Stylized Line-based Rendering. *Computer Graphics Forum*, 30(2):513–522, April 2011.

[WAL08] N. Walter, O. Aubreton, and O. Laligant. Salient point characterization for low resolution meshes. In *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pages 1512 –1515, oct. 2008.

[WB01] Kouki Watanabe and Alexander G. Belyaev. Detection of salient curvature features on polygonal surfaces. *Comput. Graph. Forum*, pages –1–1, 2001.

[WBBP12] Chaohui Wang, Michael Bronstein, Alexander Bronstein, and Nikos Paragios. Discrete minimum distortion correspondence problems for non-rigid shape matching. In Alfred Bruckstein, Bart ter Haar Romeny, Alexander Bronstein, and Michael Bronstein, editors, *Scale Space and Variational Methods*

in Computer Vision, volume 6667 of *Lecture Notes in Computer Science*, pages 580–591. Springer Berlin / Heidelberg, 2012.

[WC04] A.R. Willis and D.B. Cooper. Bayesian assembly of 3d axially symmetric shapes from fragments. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–82 – I–89 Vol.1, june-2 july 2004.

[Wei85] Kevin Weiler. Edge-based data structures for solid modeling in curved-surface environments. *IEEE Comput. Graph. Appl.*, 5(1):21–40, January 1985.

[WMKG07] Max Wardetzky, Saurabh Mathur, Felix Kälberer, and Eitan Grinspun. Discrete laplace operators: no free lunch. In *Proceedings of the fifth Eurographics symposium on Geometry processing*, SGP '07, pages 33–37, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.

[WNK06] Raoul Wessel, Marcin Novotni, and Reinhard Klein. Correspondences between salient points on 3d shapes. In L. Kobbelt, T. Kuhlen, T. Aach, and R. Westermann, editors, *Vision, Modeling, and Visualization 2006 (VMV 2006)*, pages 365–372. Akademische Verlagsgesellschaft Aka GmbH, Berlin, November 2006.

[WOC03] Andrew Willis, Xavier Orriols, and David B. Cooper. Accurately estimating sherd 3d surface geometry with application to pot reconstruction. *Computer Vision and Pattern Recognition Workshop*, 1:5, 2003.

[Wol90] H.J. Wolfson. On curve matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:483–489, 1990.

[WSS09] Hao Wang, C.E. Scheidegger, and C.T. Silva. Bandwidth selection and reconstruction quality in point-based surfaces. *IEEE TVCG*, 15(4):572 –582, 2009.

[XL06] Dong Xu and Hua Li. 3-d surface moment invariants. In *Proceedings of the 18th International Conference on Pattern Recognition - Volume 04*, ICPR '06, pages 173–176, Washington, DC, USA, 2006. IEEE Computer Society.

[XL08] Dong Xu and Hua Li. Geometric moment invariants. *Pattern Recogn.*, 41(1):240–249, January 2008.

[YF99] Sameh M. Yamany and Aly A. Farag. Free-form surface registration using surface signatures. *Computer Vision, IEEE International Conference on*, 2:1098, 1999.

[YF02] Sameh M. Yamany and Aly A. Farag. Surfacing signatures: An orientation independent free-form surface representation scheme for the purpose of objects registration and matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(8):1105–1120, August 2002.

[YLHP06] Yong-Liang Yang, Yu-Kun Lai, Shi-Min Hu, and Helmut Pottmann. Robust principal curvatures on multiple scales. In *Proc. of the Eurographics symposium on Geometry processing*, pages 223–226, 2006.

[ZBVH09] Andrei Zaharescu, Edmond Boyer, Kiran Varanasi, and Radu P. Horaud. Surface feature detection and description with applications to mesh matching. In

Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Miami Beach, Florida, June 2009.

[ZG04] Steve Zelinka and Michael Garland. Similarity-based surface modelling using geodesic fans. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, SGP '04, pages 204–213, New York, NY, USA, 2004. ACM.

[ZH99] Dongmei Zhang and Martial Hebert. Harmonic Maps and Their Applications in Surface Matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR '99)*, volume 2, 1999.

[ZHDQ08] Guangyu Zou, Jing Hua, Ming Dong, and Hong Qin. Surface matching with salient keypoints in geodesic scale space. *Comput. Animat. Virtual Worlds*, 19(3-4):399–410, September 2008.

[ZHL+09] Guangyu Zou, Jing Hua, Zhaoqiang Lai, Xianfeng Gu, and Ming Dong. Intrinsic geometric scale space by shape diffusion. *IEEE transactions on visualization and computer graphics*, 15(6):1193–200, 2009.

[ZLCZ09] X. Zhang, H. Li, Z. Cheng, and Y. Zhang. Robust curvature estimation and geometry analysis of 3d point cloud surfaces. *Journal of Information & Computational Science*, 6(5):1983–1990, 2009.

[ZOMK00] Hong-Kai Zhao, Stanley Osher, Barry Merriman, and Myungjoo Kang. Implicit and nonparametric shape reconstruction from unorganized data using a variational level set method. *Comput. Vis. Image Underst.*, 80(3):295–314, December 2000.

[ZPKG02] Matthias Zwicker, Mark Pauly, Oliver Knoll, and Markus Gross. Pointshop 3d: an interactive system for point-based surface editing. *ACM Trans. Graph.*, 21(3):322–329, July 2002.

[ZPvBG01] Matthias Zwicker, Hanspeter Pfister, Jeroen van Baar, and Markus Gross. Surface splatting. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 371–378, New York, NY, USA, 2001. ACM.

[ZRH11] Valentin Zobel, Jan Reininghaus, and Ingrid Hotz. Generalized heat kernel signature. *Journal of WSCG, International Conference on Computer Graphics, Visualization and Computer Vision*, pages 93 – 100, 2011.

[ZZWC12] Juyong Zhang, Jianmin Zheng, Chunlin Wu, and Jianfei Cai. Variational mesh decomposition. *ACM Trans. Graph.*, 31(3):21:1–21:14, June 2012.