Topological reconstruction of sampled surfaces via Morse theory

Franco Coltraro[a,b,*], Jaume Amorós[b,c], Maria Alberich-Carramiñana[a,b], Carme Torras[a]

[a] *Institut de Robòtica i Informàtica Industrial, CSIC-UPC, C/ Llorens i Artigas 4-6, 08028, Barcelona, Spain.*
[b] *Departament de Matemàtiques, Universitat Politècnica de Catalunya, Barcelona, Spain.*
[c] *Centre de Recerca Matemàtica, Barcelona, Spain*

Abstract

In this work, we study the perception problem for sampled surfaces (possibly with boundary) using tools from computational topology, specifically, how to identify their underlying topology starting from point-cloud samples in space, such as those obtained with 3D scanners. We present a reconstruction algorithm based on a careful topological study of the point sample that allows us to obtain a cellular decomposition of it using a Morse function. No triangulation or local implicit equations are used as intermediate steps, avoiding in this way reconstruction-induced artifices. The algorithm can be run without any prior knowledge of the surface topology, density or regularity of the point-sample. The results consist of a piece-wise decomposition of the given surface as a union of Morse cells (i.e. topological disks), suitable for tasks such as mesh-independent reparametrization or noise-filtering, and a small-rank cellular complex determining the topology of the surface. The algorithm, which we test with several real and synthetic surfaces, can be applied to smooth surfaces with or without boundary, embedded in an ambient space of any dimension.

Keywords: computational topology; Morse functions; surface reconstruction; point-clouds.

*Corresponding author
Email address: franco.coltraro@upc.edu (Franco Coltraro)

1. Introduction

The domestic manipulation of cloth with robots is an increasingly relevant problem because of the widespread presence of textiles in human environments, with promising applications ranging from dressing disabled people to automatic bed-making [[10, 13]]. Nowadays, it is relatively easy to obtain point-samples of an arbitrary and unknown garment that is presented before the robot, e.g. through the use of depth-cameras or 3D-scanners. Nevertheless, these points will have no known structure. Thus, if one wishes that the robot manipulate the garment, *reconstructing* and *recognizing* it from the point-cloud (i.e. to deduce its topology) becomes of great importance [[27]].

Arguably, the main challenge faced in the automated manipulation of cloth is the high number of deformation states that textiles can present [[25]]. In contrast to rigid body manipulation, where the dynamics of the manipulated object are very well understood [[26]], there is not one single physical model that can be considered best in terms of describing the dynamics of real textiles [[24]]. In any case, physical models of cloth behavior remain useful for developing planning and control strategies [[20, 4]]; as well as for generating the massive data required to train learning algorithms before their deployment and tuning in the real world [[19, 5]]. For all of these tasks it is crucial to have an accurate and fast-to-compute reconstruction of the garment to be controlled/simulated.

Naturally, because of the previous reasons, the reconstruction of a surface in Euclidean space from a point-sample on it is a question to which considerable attention has been given in the areas of Computer Graphics and Computational Geometry (see [[9]] for algorithms with mathematical guarantees and [[18]] for a survey of state-of-the-art methods). Common algorithms involve triangulating the cloud points, fitting local implicit functions or more recently applying learning (i.e. Neural Networks) methods. Nevertheless, to our knowledge almost all these algorithms disregard a direct topological study of the point-cloud. Moreover, most of them focus on reconstructing *watertight* surfaces (i.e. without boundary). Applying this kind of algorithms to point-clouds coming from garments can lead to incorrect results since textiles can be naturally realized as surfaces *with* boundary, which is coincidentally what the majority of physical models assume in order to simulate them. One of the reasons for this focus on reconstructing surfaces without boundary may be the challenge associated in detecting boundaries of point-cloud surfaces (see [[22]]): the problem is in general ill-posed since regions of the cloud with

low density could be mistaken for gaps, with boundaries in the underlying surface.

In this work, we introduce a new reconstruction algorithm that directly processes a point cloud to generate a cellular decomposition of the sampled surface. Our method produces a global piece-wise decomposition of the surface, consisting of a small number of parametrized, explicitly contractible, pieces. The algorithm does not rely on an intermediate triangulation or local implicit equations, thus avoiding any possible artifacts induced by such reconstructions. From this cellular decomposition, the surface topology can be intermediately determined. For instance, the cellular decomposition furnishes closed loops in any homology class having length which has the order of magnitude of the shortest possible loop in the class.

Furthermore, the algorithm is *topologically robust*, meaning that different samplings of the surface or noise, although giving slightly different decompositions, preserve the topological features of the sampled surface, i.e. we obtain the same underlying topology of the surface. Thus, our goal is not to propose a general-purpose surface reconstruction method that competes with state-of-the-art meshing algorithms in terms of geometric fidelity. Instead, our primary focus is on topologically faithful reconstructions, particularly for applications where capturing the correct topology is essential, such as robotic manipulation of textiles or objects with nontrivial topology (e.g. manipulating a t-shirt is completely different to handling pants: they have different topologies). Methods that prioritize geometric detail may inadvertently produce reconstructions with incorrect topology, particularly when input data is sparse or noisy. Our approach prioritizes preserving the correct topological structure, even if this entails some loss of surface detail.

To obtain this cellular decomposition, we present for the first time in literature (to our knowledge) an algorithm to determine how (discretized) Morse cells attach to each other. Furthermore, for the case of surfaces with boundary, we develop a novel graph-theoretical method to determine robustly boundary points of the cloud. Our decomposition algorithm was first briefly sketched by the authors in two proceedings papers [[1, 7]]. Here we expand greatly on that work to give full explanations on how to decompose and handle the very delicate case of surfaces with boundary.

The extracted topological features obtained with our method are expected to directly benefit downstream tasks in the robotic manipulation of cloth. For example, knowing the exact layout of 1-cells and boundary curves on a

reconstructed cloth can aid in segmentation: our Morse cell decomposition essentially segments the cloth into meaningful patches which are topologically disks. These patches could correspond to functional parts of a garment or simply provide a simplified model of the cloth's shape for higher-level reasoning. Similarly, identifying the garment's critical points and boundaries can improve automatic grasp planning; a robot could decide to grasp at a corner or along an edge, based on the cell complex, rather than treating the cloth as an amorphous cloud of points.

1.1. Contributions

In summary, the novel contributions of this work are:

1. The theoretical development of stratified Morse theory for surfaces with boundary with views towards applying it to point-clouds.

2. A robust identification of the neighbors of each point in the cloud that captures truthfully its local structure and allows the use of a discrete curvature filter to handle noisy sampled surfaces.

3. A novel recognition algorithm based on graph theory to determine robustly boundary points of the cloud.

4. The effective determination of critical points of a Morse function using level set sections which permits the computation of Morse cells and for the first time their attachment maps.

5. The reconstruction of several challenging point-clouds of surfaces with boundary and an algorithm to parameterize its 2-cells with flat patches.

1.2. Organization

The remainder of this paper is organized as follows: in Section 2 we review literature from computational and differential topology related to our method; then in Section 3 we explain and develop the theoretical machinery –coming from smooth Morse theory– needed to apply our algorithm successfully. In Section 4 we initiate the study of the point-cloud, giving it local structure by finding neighbors for each point, their tangent planes and the boundary curves. Section 5 deals with the computation of the Morse flow, critical points, Morse cells and how they attach to each other, and it is one of the main contributions of this work. Finally, in Section 6 we explain how

to parametrize the 2-cells by flat regions of the plane; and in Section 7 we present the reconstruction of five challenging point-clouds (four with boundary, one of them being a real 3D scan of a garment) using the presented algorithm.

2. Related work

Differential topologists have long focused on the problem of parametrizing manifolds through the use of Morse functions. A smooth map $f : M \to \mathbb{R}$ defined on a compact manifold without boundary is called *Morse* if it has only finitely many critical points, and at each of these points, the Hessian $H(f)$ is non-degenerate. Classical Morse theory (see [[16]]) demonstrates that a generic Morse function f induces two distinct decompositions of the manifold M via its gradient flow:

1. *As a CW complex* (see [[23]]): Each critical point of f, along with its unstable manifold for the vector field $-\nabla f$, defines a cell that is topologically a ball, whose boundary attaches to lower-dimensional cells (see Figure 1). This results in a global piece-wise parametrization of M, and a Morse-Smale complex, where the critical points of f serve as a basis, yielding the singular homology of M.

2. *As level sets*: The manifold M is foliated by the level sets $f^{-1}(c)$. For regular values of c, these level sets are submanifolds of M with codimension 1, and a diffeomorphism $f^{-1}(c_1) \cong f^{-1}(c_2)$ exists if no critical values of f fall between c_1 and c_2. When c crosses a critical value of f, the level set undergoes a surgery (see [[16]] and Section 3).

The strength of Morse theory lies in the fact that Morse functions, along with the Morse-Smale transversality conditions necessary for this analysis, are generic among $\mathcal{C}^2$ maps from M to $\mathbb{R}$. For example, a height function in a random direction in $\mathbb{R}^N$ is almost certainly a Morse-Smale function, with the probability that it is not being zero. Morse theory also generalizes to manifolds with boundaries via stratified spaces [[15]], as discussed in detail in this work.

The application of Morse theory directly to the sample point-cloud of a surface was first suggested by [[12, 28]], who proposed an algorithm for point-clouds with a known, uniform sampling density. Later, [[3]] proposed

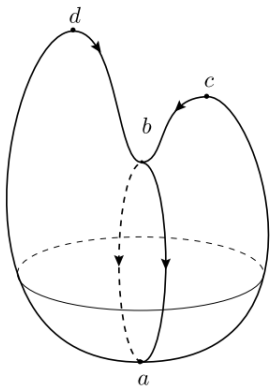


Figure 1: Critical points of the Morse-Smale function $f(x, y, z) = z$ on an example surface. The cellular decomposition is achieved by attaching two 2-cells containing d and c (the maxima) along the curve passing through b (the saddle point) and a (the minimum).

a Morse decomposition method for point-clouds sampling manifolds without boundary in any dimension. However, these approaches do not fully address issues such as cell parametrization or attachment maps, which are crucial in robotic applications where point-clouds of textiles may need to be filtered and down-sampled for simulation or control, as outlined in the introduction. We build on the gradient flows of [[12, 28]], but propose a novel method for identifying critical points and their Morse cells by examining level sections of these flows.

3. Preliminaries: smooth Morse theory for surfaces

In this section we present a summary of smooth Morse theory for surfaces with and without boundary. We first explain briefly the case without boundary, which encapsulates the main ideas of the field. For a summary of this part, including what is needed to run our algorithm see Section 3.3.

3.1. Morse theory for surfaces without boundary

Let $S \subset \mathbb{R}^N$ be a smooth compact surface without boundary. The goal is to decompose any given S as in Figure 1. In order to do that, we will compute its Morse-Smale complex, which in the case of a surface only consists of 0-cells (points), 1-cells (curves) and 2-cells (topological disks).

As explained before, a map $f : S \to \mathbb{R}$ is *Morse* if it is $\mathcal{C}^2$, has only finitely many critical points (i.e. points p where $\mathrm{d}_p f = \nabla f(p) = 0$), and at all of these its Hessian has rank 2.

Definition 1 (Morse data)**.** For each critical point $p \in S$, the Morse data are the pair of sets $(A(p), B(p))$ where

$$A(p) := \mathcal{B}(p) \cap f^{-1}\left([f(p) - \epsilon, f(p) + \epsilon]\right),$$

with $\mathcal{B}(p) \subset \mathbb{R}^N$ a closed ball around p of sufficiently small radius, the value of $\epsilon > 0$ is such that there are not more critical points of f in $f^{-1}\left([f(p) - \epsilon, f(p) + \epsilon]\right)$ and $B(p) := \mathcal{B}(p) \cap f^{-1}\left(f(p) - \epsilon\right)$. Notice that $B(p) \subseteq \partial A(p)$. See Figure 2 for examples of the sets A, B.

Theorem 1 (Main theorem of Morse theory)**.** *Let us now denote* $f_{\leq c} = f^{-1}\left((-\infty, c]\right), f_c = f^{-1}(c)$*. Then (see [[16]]) as* $c \in \mathbb{R}$ *increases:*

A *If* $c_1 < c_2$ *and there are no critical values of* f *in* $[c_1, c_2]$*, then* $f_{\leq c_1}$ *and* $f_{\leq c_2}$ *have the same topology (they are actually diffeomorphic).*

B *If there is a single critical point* $p \in S$ *such that* $c_1 < f(p) < c_2$ *then* $f_{\leq c_2}$ *is obtained, up to diffeomorphism, from* $f_{\leq c_1}$ *by attaching the cell* $A(p)$ *along* $B(p)$*, i.e.* $f_{\leq c_2} \simeq \left(f_{\leq c_1} \cup A(p)\right) / \sim$ *where the equivalence relation is given by identifying* $B(p) \subseteq f_{\leq c_1}$ *with points of* $\partial A(p) \supseteq B(p)$*.*

The previous theorem is useful because it allows us to deduce how to couple each type of critical point with the Morse cells that will give us the sought decomposition of the surface under study (e.g. Figure 1). Since $H(f)$ has rank 2 at p, we can only have three types of critical points (see Figure 2) according to their index (number of negative eigenvalues of $H(f)$):

1. Minima: $A(p)$ is homeomorphic to a disk and $B(p) = \emptyset$. In this case there is no surgery, the cell $A(p)$ just appears. This cell retracts to a point, the local minimum, which will be a 0-cell in the Morse-Smale complex.

2. Saddles: $A(p)$ is a quadrilateral (homeomorphic to a disk) and $B(p)$ consists of two segments. Two opposite sides of $\partial A(p)$ are identified with $B(p)$ (see Figure 2). The attachment of the cell $A(p)$ along $B(p)$ is homotopy-equivalent to the attachment of a 1-cell, namely the medial axis of $A(p)$ to the middle points of $B(p)$.

3. Maxima: $A(p) = D$ is homeomorhic to a disk and $B(p) = \partial D$ is its boundary. The attachment map identifies $\partial A(p)$ with $B(p)$. This surgery adds a 2-cell to the complex.

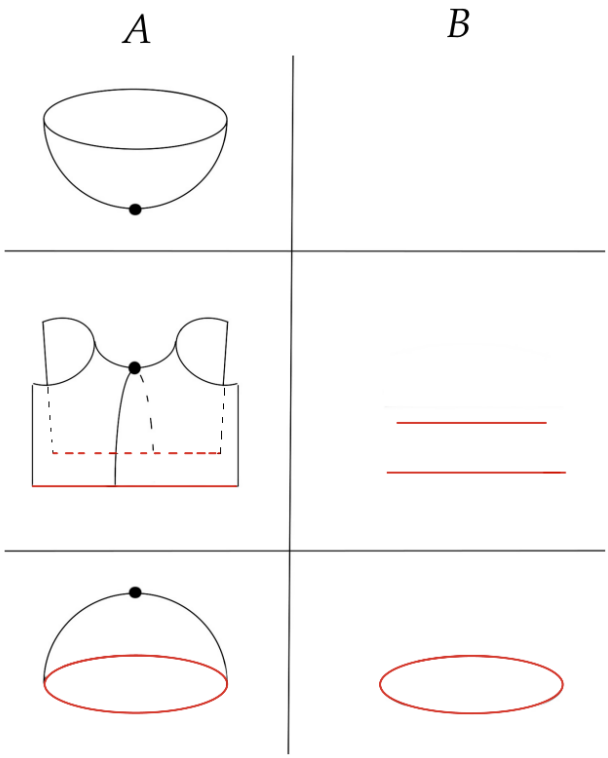


Figure 2: Three types of critical points p (from top to bottom: minima, saddles and maxima) and their Morse data $(A(p), B(p))$ for surfaces without boundary. In red we depict the identifications made between $A(p)$ and $B(p)$.

In summary, minima generate 0-cells of the complex, saddle points 1-cells and maxima 2-cells. For instance, in Figure 1 there is one 0-cell (point a), one 1-cell (the closed curve passing trough a and b) and two 2-cells (one containing d and the other c).

3.2. Morse theory for surfaces with boundary

Morse theory also extends to manifolds with boundary via stratified spaces [[15]]. Since stratified Morse theory is less well known and in [[15]] no explicit construction is given for manifolds with boundary, in this short section and in the Appendix A we give more details on how the boundaries affect the cellular decomposition and the transition between level sets using the terminology presented in the previous section for the case without boundary.

Definition 2 (Morse function)**.** We say that a $\mathcal{C}^2$ map $f : S \to \mathbb{R}$ defined on a surface S with boundary ∂S is *Morse* if

1. it is Morse in the interior of S,
2. its restriction to ∂S, $g := f_{|\partial S}$ is also a Morse function,
3. there are no saddle points of f located at ∂S.

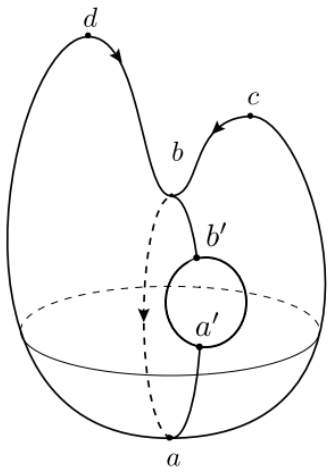


Figure 3: Critical points of the Morse-Smale function $f(x, y, z) = z$ on an example surface with boundary. Notice that now we have a boundary maximum b' and a boundary minimum a' since we have removed a disk.

Notice that now we can have critical points of $f_{|\partial S}$ in the boundary curves ∂S that are not critical points of f in the whole of S. Those are only 2 new cases which we call *boundary* minima and maxima (since ∂S is a curve, there can not be saddle points of g, see Figure 3). These critical points will be responsible for generating new Morse cells depending on whether they are also local maxima or minima of the full surface or not (see Section 3.3).

3.3. Construction of the Morse-Smale complex

We now make a summary of the effect of each critical point on the cell complex once we have made the appropriate deformation retracts. Recall that 0-cells are points, 1-cells are curves and 2-cells are topological disks. Boundary curves of ∂S will be part of the complex as 1-cells.

1. Interior maximum: attach a 2-cell to the 1-cells as before.
2. Interior saddle: attach a 1-cell to the 0-cells as before or to a point of the boundary (this point becomes a new 0-cell).
3. Interior minimum: add a 0-cell to the skeleton as before.
4. Local maximum of S located in ∂S: attach a 1-cell to a 0-cell (on the boundary) and attach a 2-cell to the 1-cells.
5. Boundary maximum (not of S): attach a 1-cell to a 0-cell (on the boundary).
6. Local minimum of S located in ∂S: add a 0-cell to the skeleton (this point will be on the boundary).
7. Boundary minimum (not of S): add a 0-cell (boundary point) to the skeleton and attach a 1-cell to a 0-cell or to a point of the boundary (this point becomes a new 0-cell).

In order to construct the complex, we first add all the 0-cells (all interior, boundary minima and possibly some points of the boundary), then the 1-cells (the boundary curves, the 1-cells corresponding to each saddle point and the 1-cells joining boundary minima to local minima or boundary points) and finally add the 2-cells corresponding to each local maximum.

Remark 1. A delicate point that we have not discussed yet, but will be crucial for our algorithm, is how to determine which cells attach to which cells, and to find explicitly the attachment maps. This problem will be addressed in detail for point-clouds in Section 5.

4. Structure of the point-cloud

Before we can define the flow of a Morse function on a given point-cloud X, we will need to give it a local structure. This is done by finding local neighbors to each point, which allows us to estimate tangent planes to X and will be very important later to recognize boundary points.

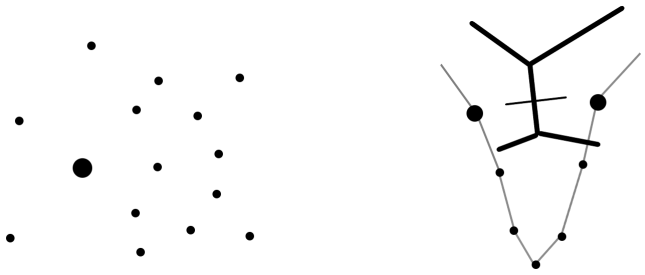


Figure 4: Typical problems associated to k-nearest neighbors (left) and Voronoi neighbors (right). On the left, most closest points to v (in bold) are clustered at one side of it. On the right, vertices that are too far apart from each other belong to neighboring cells.

4.1. Neighbors identification

The first step is the identification of a set of neighbors of each point v in the cloud $X \subset \mathbb{R}^N$. There are two classical approaches:

1. k-nearest neighbors (KNN): given a value for k and a point v, the k nearest points $\{v_1, \dots v_k\}$ with respect to the Euclidean distance are declared as its neighbors. This is quite efficient to compute but runs into problems when the point-cloud has irregular densities and k is not big enough, e.g. when all the closest points to v are clustered at one side of it and do not *enclose* the vertex (see Figure 4, left). Hence, the result is highly sensitive to the k one chooses.

2. Voronoi-Delaunay neighbors: Voronoi's cellular decomposition of the point cloud is performed, and the points v_i belonging to neighboring cells (i.e. those connected to v by an edge in the Delaunay triangulation of the cloud) are then declared as neighbors of v. This has the virtue of enclosing the vertex v even with irregular densities, but it can be expensive to compute and may produce neighbors too apart from each other (see Figure 4, right).

Therefore we merge these two criteria, and declare two points as neighbors when (i) each point is among the k-nearest neighbors of the other and (ii) their Voronoi cells in the decomposition of the ambient space $\mathbb{R}^N$ induced by X are adjoining. In order to be efficient, inspired by *sphere packing theory* [[14]], we first choose a $k = 12$ to compute the k-nearest points and then

only keep as neighbors the vertices that are connected to v in the Delaunay triangulation of these few points. Finally, the relationship of neighborhood is made symmetric by reciprocating neighboring relationships where needed. The neighbors of v will be denoted by $\mathrm{Neigh}(v)$. This produces a (locally non-planar) graph of neighbors, which gives an idea of the local structure of X, but which will be in general very complicated. In Figure 5 we can see the KNN neighbors, the Voronoi-Delaunay neighbors and our mixed criterion method computed for some example points of a given cloud.

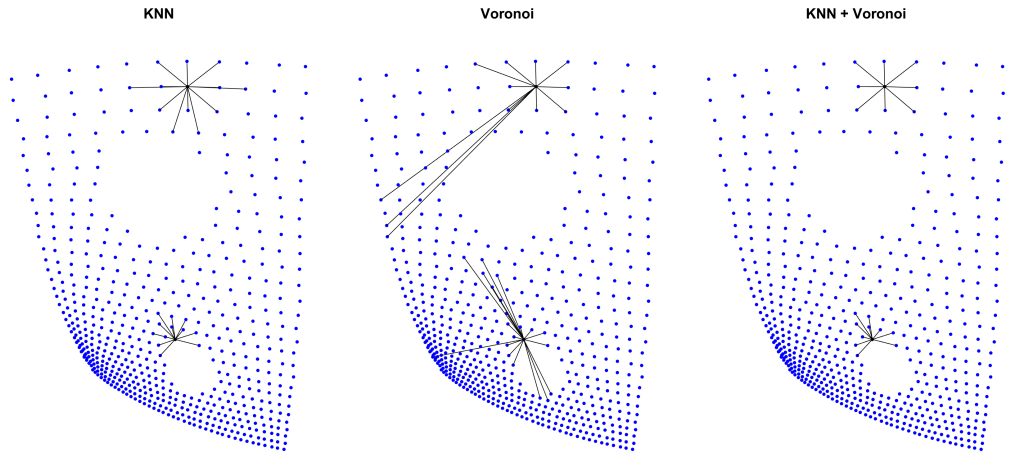


Figure 5: KNN graphs are computationally efficient, but very sensitive to the k one chooses (left, $k = 12$). The Voronoi-Delaunay criterion mitigates this issue by having no parameters and ensuring that neighbors enclose each point in a more uniform way, but often introduce long edges that do not reflect local connectivity well (middle). By combining both approaches, our mixed construction reduces the dependence on k of KNN while improving the topological correctness of the neighborhood graph (right).

4.2. Discrete curvature filter

In order to avoid the proliferation of critical points of the Morse-Smale function that in turn would generate a very high number of Morse cells, once we have identified neighbors of each point, we apply a *discrete-curvature filter* based on the combinatorial Laplacian of the graph of neighbors to the point-cloud. This means that we substitute each point v for a weighted average of its position and the location of its neighbors:

$$h(v) = \alpha v + (1-\alpha)\frac{1}{|\operatorname{Neigh}(v)|}\sum_{v_i \in \operatorname{Neigh}(v)} v_i.$$

When applied a small number of times this filter defines a bijection between the original cloud and the filtered one, which preserves the topology of the underlying surface (see [[8]] for a thorough discussion of topology preserving curvature flows applied to triangle meshes). Hence, the decomposition we find for the filtered case will still be valid and topologically accurate for the original cloud. The application of this filter is not always needed, being most relevant when the point-clouds present a lot of noise or a high level of local variability.

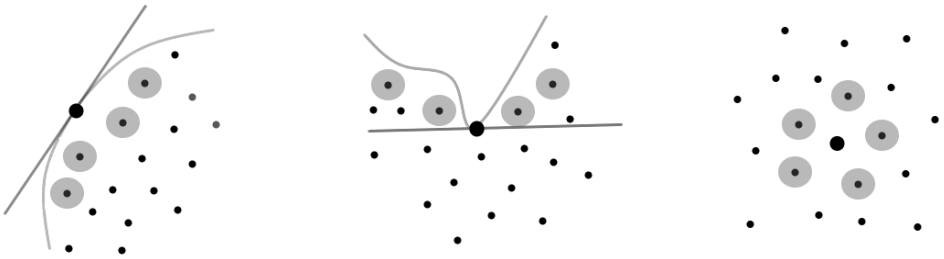
4.3. Tangent space estimation

This task is performed through *Principal Component Analysis* [[17]]: if the point v and all its neighbors $\operatorname{Neigh}(v) = \{v_1, \dots v_k\}$ were co-planar we would have that for every i: $\langle v\vec{v}_i, n_j\rangle = 0$, where n_j are all the normal vectors to the surface at v (recall that in general we are in $\mathbb{R}^N$) and $v\vec{v}_i = v_i - v$. Since in general this will not be the case, we find the n_j's by minimizing the function $\sum_j \sum_{i=1}^k \langle v\vec{v}_i, n_j\rangle^2$. This is equivalent to finding the regression plane through v in the least squares sense, and it can be done efficiently by means of a *singular value decomposition* of the matrix with vectors $v\vec{v}_i$ as rows: the right singular vectors corresponding to the 2 largest singular values define the tangent plane, whereas the rest give us the normal directions.

4.4. Boundary recognition

Once we have an estimation of the tangent spaces, in principle a boundary point of the surface can be easily identified because after orthogonally projecting it and its neighbors on its tangent plane, they cluster in a half-space (see the first panel of Figure 6). Nevertheless this method is difficult to implement robustly (see the second panel of Figure 6).

In order to obtain a robust detection, the idea will be to declare points as lying on the boundary only when the projections do not enclose the point. In points with high curvature where the tangent plane may not be perfectly estimated (or equivalently the normal vectors; for a discussion of this phenomenon, see [[2]]) the previous method can give false positives (see Figure 7 lower panel, left). In order to overcome this difficulty, we will also project the point and all its neighbors in the tangent planes estimated for the neighbors.



(points projected in a tangent plane)

Figure 6: In principle, a boundary point can be identified easily because after projecting it and its neighbors (in gray in the figure) on the tangent plane, they cluster in a half-space (defined by the gray line in the figure) of $\mathbb{R}^2$ (left panel). Nevertheless, this is not always the case for every boundary point (middle panel). To overcome this, we declare a point as being on the boundary when none of the plane projections of it and its neighbors in all their respective tangent planes enclose the point (right panel).

We will build a graph for every projection and only declare v as boundary point when none of the graphs enclose v (see Figure 7 lower panel, right).

Now we proceed to describe our method in detail: let $v \in X$, and T_vS be the estimated tangent plane at v. We follow the following steps:

1. Given $\{v_1, \dots v_k\}$ the neighbors of $v_0 := v$ we project the $k+1$ points on the planes $T_{v_j}S$ for $j = 0, \dots k$. These projections will be denoted by $\pi_j(v_i)$.

2. We create a plane graph G_j (not necessarily planar) with the projected points for every $j = 0, \dots k$, where we add an edge between $\pi_j(v_i)$ and $\pi_j(v_l)$ only when v_i, v_l are themselves neighbors in the cloud and $i, l \neq 0$.

3. We declare the vertex v_0 as a boundary point only when none of the plane graphs G_j *encloses* $\pi_j(v_0)$ for every projection to $T_{v_j}S$.

Remark 2. We say that that a plane graph G (not necessarily planar: edges can cross) does not enclose a point when the point is not inside any of the simple cycles contained in the graph.

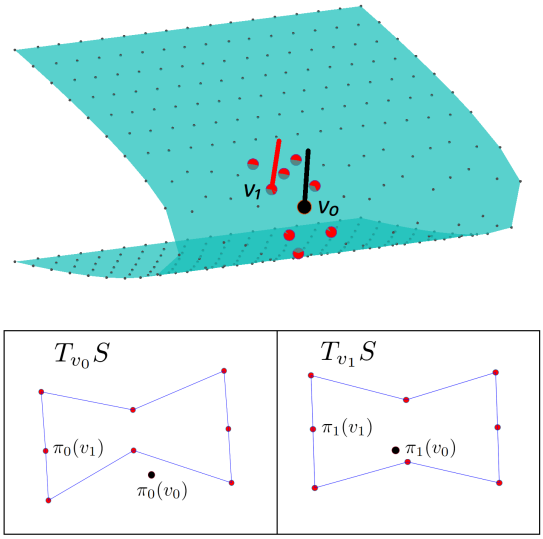


Figure 7: In interior points with high curvature the projections of the neighbors (plotted red) of a point v_0 (plotted black) into the tangent plane of the point may not enclose said point (lower panel, left). In order to avoid false positives we also project the point and its neighbors into the tangent planes estimated for the neighboring points (lower panel right, depicted for a neighbor v_1). In the example point-cloud we also plot the estimated normal vector to the tangent planes corresponding to v_1 (in black) and to v_0 (in red).

Remark 3. In point-clouds with irregular densities the previous algorithm can still give false positives because of small boundary holes. In that case clusters of boundary points with small diameter (i.e. clusters contained in small balls) can be discarded.

4.5. Reconstruction of curves

Curve reconstruction from a point-cloud sample is used for boundary parametrization, and later for level set reconstruction when we intersect the gradient flow graph with level hyperplanes. We employ a variant of the NN-Crust algorithm described in [[9]], which can be called KNN-Crust: we limit the search of edges to the k-nearest neighbors on the curve, typically with $k = 9$. This speeds up the computation, and makes the parametrization more robust in the presence of noise or non-uniformity of the point-cloud. The rest of the reconstruction follows as in [[9]]. Thus, from now on, we assume

that boundary curves of the cloud (if present) have been reconstructed and parametrized.

5. Morse function and Morse cells

Once we have a notion of neighborhoods in the cloud, we are able to define Morse functions, and more importantly their flows. This in turn allows us to compute critical points and from them the Morse cells that form the skeleton of the topological decomposition of the point-cloud. The final step will be to study how these cells attach to each order, in order to recover the full topology of the underlying surface.

5.1. Morse function and its flows

We define $f : X \to \mathbb{R}$ as the height function $f(v) = \langle v, \nu \rangle$ where ν is a fixed unitary direction vector. We try several ν randomly until all the $\{f(v)\}_{v \in X}$ are different and the number of local maxima of f (the future number of 2-cells) is small. In the presence of boundaries we also choose the direction ν that minimizes the number of critical points at the boundaries. This keeps the quantity of boundary 1-cells small.

Then, following [[12, 28]] we define the **upward flow** of f as the function $\mathrm{Up} : X \to X$ such that

$$\mathrm{Up}(v) = \mathrm{argmax}_{v_k \in \mathrm{Neigh}(v),\ f(v_k) > f(v)} \frac{f(v_k) - f(v)}{||v_k - v||}. \tag{1}$$

Analogously, the **downward flow** is the function $\mathrm{Down} : X \to X$ defined by

$$\mathrm{Down}(v) = \mathrm{argmin}_{v_k \in \mathrm{Neigh}(v),\ f(v_k) < f(v)} \frac{f(v_k) - f(v)}{||v_k - v||}. \tag{2}$$

When $f(v) > f(v_k)$ for every $k \in \mathrm{Neigh}(v)$, we have a local maximum and write $\mathrm{Up}(v) = v$. Likewise, when $f(v) < f(v_k)$ for every $k \in \mathrm{Neigh}(v)$, we have a local minimum and $\mathrm{Down}(v) = v$.

Since we have identified and parametrized the boundary curves of X, we can define the two flows $\partial\mathrm{Down}(\cdot)$ and $\partial\mathrm{Up}(\cdot)$ analogously (considering that each boundary point has two natural neighbors in ∂X) so that they send boundary points to boundary points. Then we can easily compute boundary maxima (resp. minima). Notice that in general these points do not need to be local maxima (resp. minima) of the full cloud X.

5.2. Hyperplane sections and computation of critical points

In order to compute the critical points (minima, saddles and maxima) of a given Morse function f, we perform n level set intersections at equally spaced levels $c_i = c_0 + i \cdot h$, where $c_n = \max f(X)$ and $c_0 = \min f(X)$. The separation h among levels is the size threshold over which we will be able to detect topological features. For surfaces with boundaries, however, it may be preferable to choose levels that are not equally spaced, as discussed in Remark 4. This process involves intersecting the oriented graph G_{down}, with vertices X and edges defined by $\text{Down}(\cdot)$ (i.e., two vertices $v, w \in X$ share an edge if $w = \text{Down}(v)$), along with the boundary curves, with the hyperplanes $H_c = p \in \mathbb{R}^N : p \cdot \nu - c = 0$ (see Figure 8). These intersections $\Gamma(c) = G_{\text{down}} \cap H_c$ yield plane point-samples of one-dimensional curves (possibly with boundary, i.e., intervals whose endpoints correspond to boundary points of X as identified in Section 4.4), which we can reconstruct and parametrize as described in 4.5. By construction, we can trace the correspondence of points between different levels, $\Gamma(c_{i+1})$ and $\Gamma(c_i)$, using the flow $\text{Down}(\cdot)$. Changes in the number or topological type of the connected components of these level sets indicate that we have crossed critical points of f. We will analyze each possible scenario in detail.

In order to get the correspondence of points between the levels $\Gamma(c_{i+1})$ and $\Gamma(c_i)$ using $\text{Down}(\cdot)$, we may need to apply the flow more than once depending on the size h of the jump between level sets that we have defined. The explicit correspondence is obtained as follows: for each node v, we consider the trajectory given by the sequence $v^n = \text{Down}(v^{n-1})$ where $v^0 = v$ and then we intersect this polygonal curve with the planes $H_{c_{i+1}}$ and H_{c_i} respectively.

Remark 4. When the surface has non-empty boundary we select level sections that are not necessarily equispaced, so that between two consecutive (boundary) maxima or minima of f there is always a regular level set (without any critical value) in between. Notice that we have an estimation of the approximate location of (boundary) maxima or minima by virtue of studying the flows $\text{Down}(\cdot), \partial\text{Down}(\cdot), \text{Up}(\cdot)$ and $\partial\text{Up}(\cdot)$ as described in Section 5.1.

Case without boundary

When the underlying surface S of the point-cloud X has no boundary, the reconstructed curves $\Gamma(c)$ are all homeomorphic to the disjoint union of $\mathbb{S}^1$ and hence only 3 changes may happen:

1. *Passing a maximum*: when going down from $\Gamma(c_{i+1})$ to $\Gamma(c_i)$ a new connected component appears (see Figure 8, left). These new points can **not** be reached from points of $\Gamma(c_{i+1})$ using the flow Down$(\cdot)$.

2. *Passing a minimum*: when going down from $\Gamma(c_{i+1})$ to $\Gamma(c_i)$ an existing connected component disappears (see Figure 8, right). These points do **not** go to points of $\Gamma(c_i)$ using the flow Down$(\cdot)$.

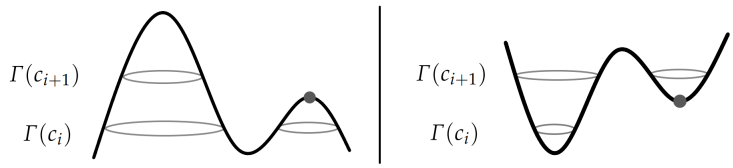


Figure 8: *Left*: Passing a maximum point when following the downwards flow: a connected component appears. *Right*: Passing a minimum when following the downwards flow: an existing connected component disappears.

3. *Passing a saddle*: when going down from $\Gamma(c_{i+1})$ to $\Gamma(c_i)$ a connected component appears or disappears (see Figure 9). Points on any component of $\Gamma(c_{i+1})$ have images in every component of $\Gamma(c_i)$ using the flow Down$(\cdot)$.

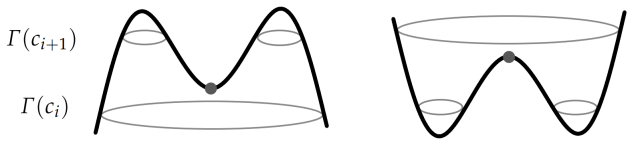


Figure 9: Passing a saddle point when following the downwards flow: a connected component appears or disappears.

Case with boundary

When the underlying surface S of the point-cloud X has a boundary ∂S, the reconstructed curves $\Gamma(c)$ are homeomorphic to a union of $\mathbb{S}^1$ and closed intervals $[0, 1]$. These intervals always join two points of ∂S, introducing an equivalence relation in $\partial S \cap H_c$. Changes in pairings of boundary points

when going from one hyperplane section to the next are important to detect saddle points (see Figure 10, fifth case of the last row). Therefore, we have the three previous cases, plus two new ones, since now we can have minima and maxima at the boundaries. Saddle points can not be located at the boundaries since that would violate the third condition of Definition 2 as already explained. We will make a distinction between (local) minima (resp. maxima) of the whole point-cloud, and boundary minima (resp. maxima) which are points that are critical when we restrict f to the boundary curves, but not in the whole surface.

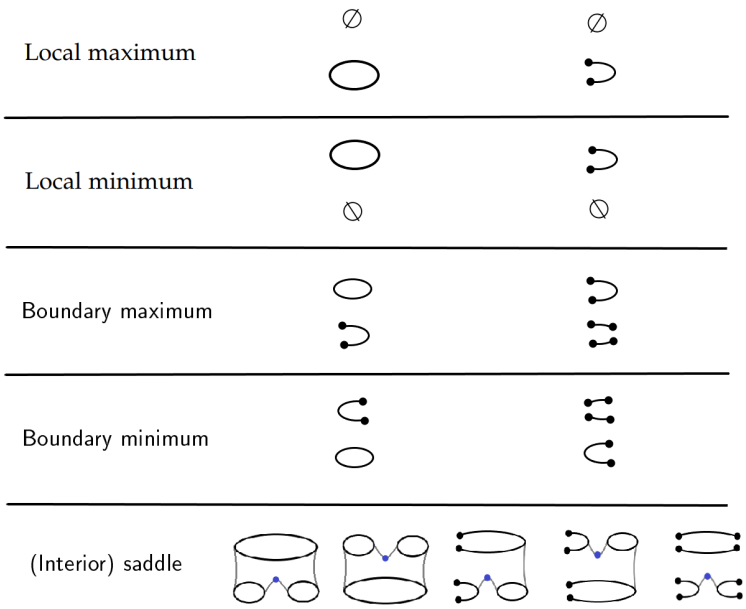


Figure 10: Different local level set transformations: the reconstructed curves $\Gamma(c)$ are homeomorphic to a union of $\mathbb{S}^1$ and closed intervals $[0, 1]$.

In Figure 10 we can see a summary of the generic (i.e. non-degenerate) level set transformations when the level crosses that of a particular type of critical point. More in detail, we can have the following 5 cases:

1. *Passing a local maximum*: this is similar as before, when going down from $\Gamma(c_{i+1})$ to $\Gamma(c_i)$ a new connected component $\mathbb{S}^1$ or $[0, 1]$ whose points can **not** be reached from $\Gamma(c_{i+1})$ using Down$(\cdot)$ appears.

2. *Passing a local minimum*: similarly as before, when going down from $\Gamma(c_{i+1})$ to $\Gamma(c_i)$ an existing connected component $\mathbb{S}^1$ or $[0,1]$ whose points do **not** go to $\Gamma(c_i)$ using $\mathrm{Down}(\cdot)$ disappears.

3. *Passing a boundary maximum*: when going down from $\Gamma(c_{i+1})$ to $\Gamma(c_i)$ either a $[0,1]$ opens in two intervals or a $\mathbb{S}^1$ splits into a $[0,1]$.

4. *Passing a boundary minimum*: when going down from $\Gamma(c_{i+1})$ to $\Gamma(c_i)$ either a $[0,1]$ closes to be a $\mathbb{S}^1$ or two $[0,1]$ merge into one interval.

5. *Passing a saddle*: this is the most complex case, when going down from $\Gamma(c_{i+1})$ to $\Gamma(c_i)$ a connected component $\mathbb{S}^1$ or $[0,1]$ appears or disappears. We can also have a case where the number of connected components stays the same but in passing from $\Gamma(c_{i+1})$ to $\Gamma(c_i)$ the pairing of boundary points of ∂S given by the $[0,1]$ component changes. Points on any component of $\Gamma(c_{i+1})$ have images in every component of $\Gamma(c_i)$ using the flow $\mathrm{Down}(\cdot)$.

5.3. Identification of Morse cells

In this section we explain how to identify all the *discrete* Morse cells of the complex that give us a cellular decomposition of the point-cloud.

0-*cells*

They correspond to local minima or to boundary minima of f, we already found them; they are the fixed points of $\mathrm{Down}(\cdot)$ and $\partial\,\mathrm{Down}(\cdot)$ respectively.

1-*cells*

There are three cases that generate the 1-cells of the skeleton:

1. Boundaries: when S has non-empty boundary, the boundary curves are also part of the 1-skeleton. We already parametrized those. Each boundary maximum gives rise to a 1-cell that attaches to a 0-cell (point) located at the boundaries.

2. Saddle points: there are one-dimensional curves that go from one local minimum m_1 or boundary point to another (not necessarily distinct) local minimum m_2 or boundary point passing through the saddle point.

3. Boundary minima: there are 1-cells in the complex not associated to a saddle point or to boundary curves. In this last case they always connect a boundary minimum to a local minimum or boundary point of the cloud. In order to compute this 1-cell we simply flow down every boundary minimum using Down$(\cdot)$.

Remark 5. When 1-cells introduced by saddle points or boundary minima end at points of the boundary ∂S, or of preexisting 1-cells, which are not the ends of its original 1-cell and thus not part of the 0-skeleton, the original 1-cell is split into 2 by the meeting point, which is labeled as a new 0-cell.

Computation of saddles and their 1*-cells*

We now explain how to compute saddle points and afterwards their associated 1-cells. Recall that we know when we are in the presence of a saddle: when going from $\Gamma(c_{i+1})$ to $\Gamma(c_i)$ a connected component appears, disappears or there is a change of bordism pairing of boundary points. In any case, all points of $\Gamma(c_{i+1})$ have images in $\Gamma(c_i)$ using the flow Down$(\cdot)$. Therefore, there exist four points A, B, C, D in $\Gamma(c_{i+1})$ that can be paired by proximity (i.e. are neighbors in the level set curves) as $\{A, B\}$ and $\{C, D\}$ whose images by Down$(\cdot)$ are now paired differently as $\{A', C'\}$ and $\{B', D'\}$ (see Figure 11).

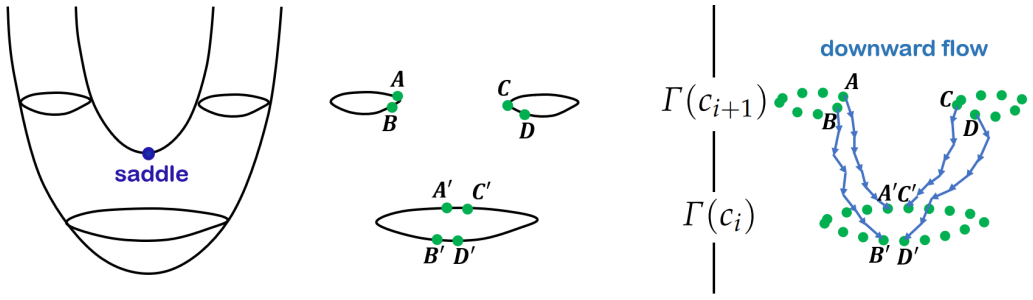


Figure 11: Change in level set topology when crossing a saddle point in a surface (left) and point-cloud (right): note the change in neighbors among the 4 marked points after applying the flow Down$(\cdot)$.

One branch of the 1-cell is obtained by taking the average of the two orbits generated by Down$(\cdot)$ starting from $\{A', C'\}$. The other branch is approximated in the same manner but starting from $\{B', D'\}$. This averaging

operation generates new points not previously in the cloud. All these new points are added to the cloud, declaring as neighbors of s the 8 points used for its computation, and for each new point of the branches its two neighbors in the 1-cell plus the two points that were used for its computation. The flows $\mathrm{Down}(\cdot)$ and $\mathrm{Up}(\cdot)$ are simply defined at those points by following the newly created trajectories down or up (except the saddle points which are fixed points of both flows). This ensures that this newly defined curve (the 1-cell) is invariant by both flows.

2-*cells*

There is one 2-cell for each local maximum of f (recall that boundary maxima only generate 1-cells, i.e. the boundary curves), we already found them (they are also given by the fixed points of $\mathrm{Up}(\cdot)$), but we will now explain how to deduce which points of the cloud correspond to each 2-cell (or each maximum). The idea is simply to flow up every point $v \in X$ by $\mathrm{Up}(\cdot)$ and see at which maximum it ends up. In symbols this means that we consider the limit of the sequence $v^n = \mathrm{Up}(v^{n-1})$ where $v^0 = v$ when $n \to +\infty$ (this limit exists because $f(v^n) \geq f(v^{n-1})$ by definition and f is bounded being X a finite collection of points and hence compact). It can happen that the sequence $\{v^n\}_{n=1}^{+\infty}$ converges to a saddle s. This can only be allowed for points of the 1-cells, but must be corrected for the rest, otherwise v is not assigned to any 2-cell. When a point v ends at a saddle s but it is not part of the 1-cell, we look at the neighbors $\mathrm{Neigh}(v)$ and see in which maximum they finished. The most common maximum among the neighbors is selected as the corrected destination of v and hence its 2-cell is deduced. In case the neighbors of v also converge to a saddle s and not to a maximum, it may be necessary to also consider the limit of their own neighbors.

5.4. Attachment maps of the Morse cells

Now that we have the skeleton of S, i.e. the $0, 1, 2$-cells, we encounter a delicate problem: figuring out the attachment maps between the cells. We already know how the 1-cells attach to the 0-cells. We now discuss how 2-cells attach to 1-cells. We must figure out which 1-cells are the boundary of the 2-cells, in which order, orientation and how many times they appear: once (when they are part of the boundary of S or when they attach to a different 2-cell) or twice (when they will be, formally, two different sides of the 2-cell that are identified).

Smooth case

We first describe the process for the smooth case (without boundary) and then explain how to adapt it to point-clouds. Notice that the 2-cell corresponding to each maximum $\hat{m}$, let us call it $D_{\hat{m}}$, is the set of points that converge to $\hat{m}$ under the flow $+\nabla f$ (in Dynamical System terminology, its stable manifold). The main idea to deduce how the boundary of $D_{\hat{m}}$ attaches to the 1-cells is to choose a simple closed curve around each maximum and flow it down by $-\nabla f$ until it reaches the 1-cells. To achieve this properly, we must perturb $-\nabla f$ so that in a neighborhood of the 1-cells the gradient flow is *transversal* (i.e. not parallel) to the 1-cells. This is needed since otherwise the points of $D_{\hat{m}}$ would converge to minima under the downwards flow $-\nabla f$. In order to obtain this perturbed $-\nabla \tilde{f}$ we consider an arbitrary small tubular neighborhood around the 1-cells and a normal vector field to the 1-cells. These two flows are joined smoothly in the tubular neighborhood with the aid of a partition of unity (i.e. bump functions, see [[16]]).

Discrete case

In the discrete case we choose a simple (i.e. without self-intersections) closed curve of neighbors around each maximum (a *crown*) and flow it by $\mathrm{Down}(\cdot)$ until it reaches the 1-cells. We do this in a way such that when a point of the curve has neighbors that are in the 1-cell, we stop following the flow $\mathrm{Down}(\cdot)$ and match the point in question to the point in the 1-cell which results in the most negative downwards slope. This method would work perfectly if simple closed curves were preserved by $\mathrm{Down}(\cdot)$; since this is not the case, we must *repair* the curve every time we flow it down. There are three main situations:

1. *Splitting of points*: it may happen that two points w_1 and w_2 were neighbors but $\mathrm{Down}(w_1)$ and $\mathrm{Down}(w_2)$ are not. In that case we define the sub-graph of neighbors given by points

$$\{v \in X : \ f(v) \leq \max\left[f(\mathrm{Down}(w_1)), f(\mathrm{Down}(w_2))\right]\},$$

   and compute the shortest path joining $\mathrm{Down}(w_1)$ to $\mathrm{Down}(w_2)$ (taking as distance of a path that given by the edges of the graph).

2. *Collapse of points*: it may happen that two points w_1 and w_2 have the same image $\mathrm{Down}(w_1) = \mathrm{Down}(w_2)$. In that case we simply delete one of the repeated points from the curve.

3. *Creation of spikes*: it may happen that one of two neighboring points w_1 and w_2 has image $\text{Down}(w_2) = w_1$. This could happen with more than one point at the time. In that case we simply delete the *spiky* points (i.e. $\text{Down}(w_2) = w_1$) in the new curve.

The final problem we face is that the flowing crown may arrive at a stage where it is stationary by the downward flow but some points of it have not reached the 1-cells. In that case we pair each unmatched point of the crown with the closest (as measured by the edges of the neighboring relationship of the cloud) point of the 1-cells.

We now have a matching between the initial crown and the 1-cells (although not a bijection). But this is enough to deduce which 1-cells are the boundary of $D_{\hat{m}}$ (only the 1-cells that are reached), in which order (this can be deduced since the crown is parametrized), the orientation (again thanks to the parametrization) and how many times they appear (once or twice, depending on the matching). This information completely determines the boundary operator of the Morse-Smale complex and hence the homology of the surface (see [[16]]).

Attachment maps for surfaces with boundaries

When the surface S has a boundary ∂S, the boundaries of the 2-cells are no longer only the 1-cells derived from the saddle points, but also possibly curves of ∂S. We distinguish several cases depending on the type of maximum we have:

1. Interior maxima (not belonging to ∂S): as before we flow down a crown around the maximum until it reaches a point that is either a neighbor of a boundary point or of a 1-cell. In this way we obtain a matching between the boundary of the 2-cell and the 1-cells and boundary curves of S.

2. Boundary maxima (but not a local maximum of S): they do not intervene in the attachment maps of 2-cells to 1-cells (although they give rise to a 1-cell in the boundary).

3. Local maxima (located at ∂S): we consider a *semi-crown* around the maximum $\hat{m}$, meaning that we take an arc of the boundary centered at $\hat{m}$ and complete it with neighboring interior nodes so that we obtain a closed simple curve. Then we flow down this semi-crown in such a

way that the arcs of the boundary stay in the boundary (we use the restricted flow $\partial\,\mathrm{Down}(\cdot)$) and the interior nodes of the curve flow down normally by $\mathrm{Down}(\cdot)$. As before we flow down the semi-crown until each point of it reaches a node that is either a neighbor of a boundary point or of a 1-cell.

For an example of this process in action, see the Supplementary Video 1.

5.5. Algorithm for the topological decomposition of point-clouds

To finish this section, we describe our full algorithm in pseudo-code (see Algorithm 1), referencing the relevant sections needed to carry out the necessary calculations. The only inputs we need are: the point-cloud X, the number of neighbors k to consider and a unitary direction ν. As the final result we get a decomposition consisting of 0-cells $\mathcal{C}_0$ (points), 1-cells $\mathcal{C}_1$ (polygonal curves) and 2-cells $\mathcal{C}_2$ (collection of points); together with their attachment maps $g_1 : \mathcal{C}_1 \to \mathcal{C}_0$, $g_2 : \mathcal{C}_2 \to \mathcal{C}_1$ telling us which cells are the boundary of other cells.

Algorithm 1 Reconstruction of point-cloud surfaces

Require: X, k, ν

1: $\mathrm{Neigh} \leftarrow \mathrm{neighbors}(X, k)$ ▷ (Section 4.1)
2: $X \leftarrow \mathrm{curvatureFilter}(X)$ ▷ (Section 4.2)
3: $\mathrm{Normals} \leftarrow \mathrm{tangentPlanes}(X, \mathrm{Neigh})$ ▷ (Section 4.3)
4: $\partial X \leftarrow \mathrm{boundaryPoints}(X, \mathrm{Neigh}, \mathrm{Normals})$ ▷ (Section 4.4)
5: $\partial S \leftarrow \mathrm{parametrizeCurves}(\partial X)$ ▷ (Section 4.5)
6: $\mathrm{Up}, \mathrm{Down} \leftarrow \mathrm{MorseFlows}(X, \mathrm{Neigh}, \nu)$ ▷ (Section 5.1)
7: $\mathcal{P} \leftarrow \mathrm{criticalPoints}(X, \partial S, \mathrm{Down}, \nu)$ ▷ (Section 5.2)
8: $\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2 \leftarrow \mathrm{MorseCells}(X, \mathcal{P}, \mathrm{Down}, \mathrm{Up})$ ▷ (Section 5.3)
9: $g_1, g_2 \leftarrow \mathrm{attachmentMaps}(X, \mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2, \mathrm{Down})$ ▷ (Section 5.4)
10: **return** $\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2, g_1, g_2$

6. Parametrization of the 2-cells

Once we have each Morse cell and their attachment maps identified, the last problem we face is how to parametrize the 2-cells, i.e. finding a flat domain $D \subset \mathbb{R}^2$ and a map $\phi : D \to X$, such that each $\phi(D) \subset X$ corresponds to one of the 2-cells found before. We will further require D to be a convex polygon and that ∂D is isometric to $\phi(\partial D)$. Once we have a parametrization

of each 2-cell, since they attach well (their boundaries are the 1-cells), we have a full piece-wise parametrization of X. In order to find D and ϕ, we follow two steps:

1. Take any convex polygon in the plane whose boundary ∂D is isometric to the boundary of the 2-cell (e.g. a rectangle).

2. Obtain a correspondence of interior points of D mapping to the cloud points in the 2-cell: for each interior point v in the 2-cell we want to find an interior point $x = \phi^{-1}(v)$ in D. We assume that each point p_i of D (including ∂D) is a barycentric combination of its neighbors (as given by the neighboring relationship of the cloud) with Tutte's weights (all coefficients are equal to $\frac{1}{|\operatorname{Neigh}(v)|}$) or weights inverse to the distance, as proposed by Floater (see [[11]]). This results in a sparse linear system, whose solution gives us the coordinates of interior points of D mapping to the cloud.

Once we have these points, we can extend the parametrization to the whole interior of D by taking its Delaunay triangulation with the newly found vertices, and interpolating linearly for the images. Then we can re-mesh (or even quadrangulate) the polygon D if desired.

Remark 6. We may wish ∂D to also have the same number of sides of $\phi(\partial D)$ (i.e. the different 1-cells), their length and their order (these are known in advance and will be determined by the attachment map of the 2-cell to the 1-cells). In that case denoting the sides' lengths by $l_1, \ldots, l_d$; we can obtain a convex polygon $D \subset \mathbb{R}^2$ whose sides are $l_1, \ldots, l_d$ by finding the polygon with maximal area and predetermined sides $l_1, \ldots, l_d$ inscribed in a circumference. To do so, an optimization problem is solved in order to find the circumference in question. By a result of Bramagupta [[21]], this is done by finding the radius that makes the polygon's interior angles add up to 2π. Once we have D, we obtain a bijection between ∂D and the points of the corresponding 1-cells in X.

7. Results

In this section we reconstruct different sampled surfaces, with and without boundary which pose various challenges to the presented algorithm. Six of

Surface	$\lvert\partial S\rvert$	0-cells	1-cells	2-cells	$\mathcal{X}(S)$
Knotted torus	0	2	4	2	0
Ellipsoidal vest	3	5	8	2	-1
Dumbbell	0	2	2	2	+2
Turbine fan blade	4	7	10	1	-2
Cow head	1	7	9	3	+1
Algebraic rosette	4	12	16	4	-1 / +1*
Tank top	4	6	9	1	-2
Scanned pants	3	4	6	1	-1

Table 1: Reconstructed point-cloud samples and their computed topologies.

Note: $\lvert\partial S\rvert$ denotes the number of connected components of the boundary and $\mathcal{X}(S)$ the Euler characteristic computed as #0-cells - #1-cells + #2-cells.
*In the case of the Rosette $\mathcal{X}(S)$ is computed separately for each connected component.

the point-clouds are synthetic whereas the two are real 3D scans, one of an actual textile: a pair of pants.

A summary of the results of applying our algorithm to the all point-clouds can be seen in Table 1. There we display the number of reconstructed curves of the boundary and the Morse cells found. This information is enough to characterize topologically surfaces with boundary, since with it we can compute the Euler characteristic of the surface [[16, 23]].

Knotted torus

Figure 12 shows our algorithm applied to a point-sample from a torus embedded in $\mathbb{R}^3$ along a (2,3)-toric knot. The algorithm correctly detects 2 local maxima, 2 local minima and 4 saddle points for the height function depicted in the figure. Out of a point cloud of 30 000 points, a decomposition of the surface into 8 Morse cells is found (two 0-cells: the minima, four 1-cells associated to the saddle points and two 2-cells, one for each maximum). On the bottom of Figure 12 we display the level set curves $\Gamma(c) = G_{\text{down}} \cap H_c$ (see Section 5.2). Notice that for this point-cloud we have all possible local transformations of level-set curves for surfaces without boundary (see Figure 10).

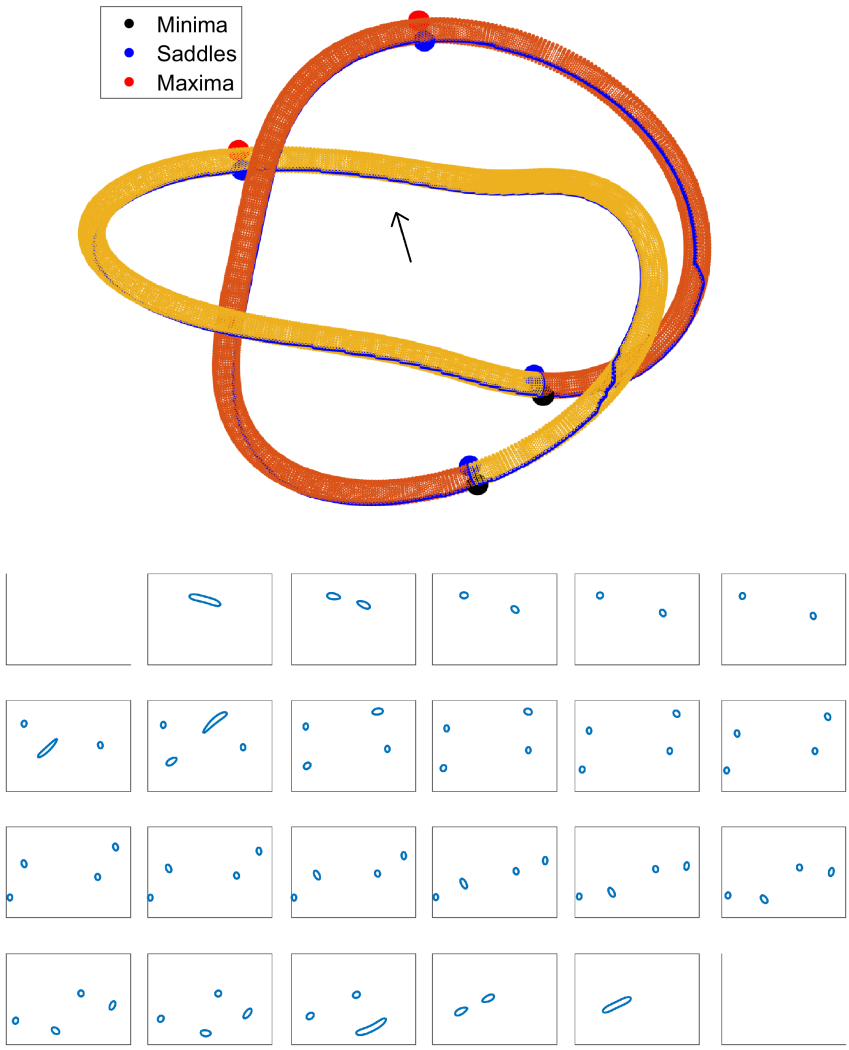


Figure 12: A sampled knotted torus: the black line is the direction of the height function; local maxima, resp. minima, are painted red, resp. black; saddle points are painted blue; their 1-cells are outlined in blue. On the bottom we plot the level set curves.

Ellipsoidal vest

Figure 13 shows our algorithm applied to a sample of 36 000 points from an embedded vest in $\mathbb{R}^3$. The point-cloud was generated by cutting sections of an ellipsoid to create a surface with the same topology as an open vest. After successfully detecting and parametrizing the boundary, the algorithm

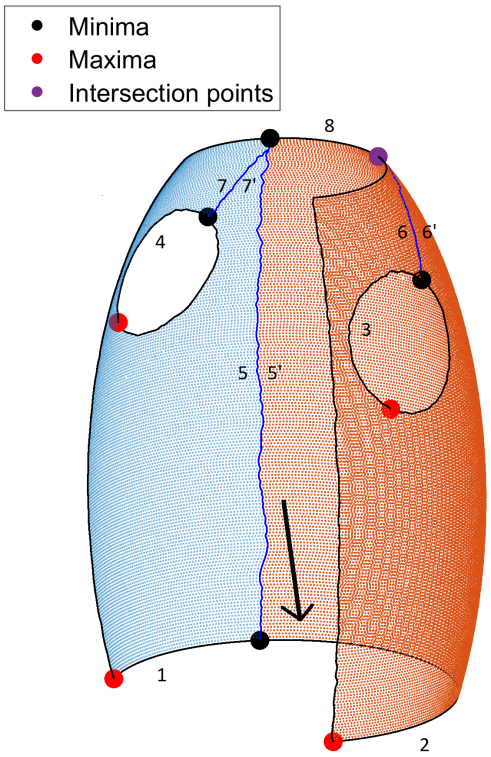


Figure 13: A sampled vest: the black line is the direction (downwards in this case) of the height function; maxima are painted in red, minima in black; 1-cells corresponding to boundary minima are outlined in blue and the boundary curves in black. One purple point where a 1-cell meets a boundary curve is added to the decomposition. The numbers correspond to the different formal 1-cells that, when identified (e.g. 7 with 7'), reconstruct the entire surface from 2 pieces homeomorphic to disks.

correctly identifies 2 local maxima, 2 boundary maxima, 1 local minimum, and 3 boundary minima for the height function (pointing downwards) shown in the figure. All critical points are located at the boundary. Additionally, one new point (shown in purple) is introduced where a 1-cell intersects a boundary curve (see Remark 5).

Then, a decomposition of the surface into 15 Morse cells is found (five 0-cells, eight 1-cells and two 2-cells). In order to deduce how the two 2-cells attach and which 1-cells are their boundary, we apply the curve flow explained in Section 5.4. This process in action for one of the 2-cells can be visualized in the Supplementary Video 1. Thus, we deduce how the cells

attach with each other (e.g. the 1-cell number 5 appears on both 2-cells and it is precisely one of the curves where they attach, see Figure 13). From this we recover the entire topology of the vest.

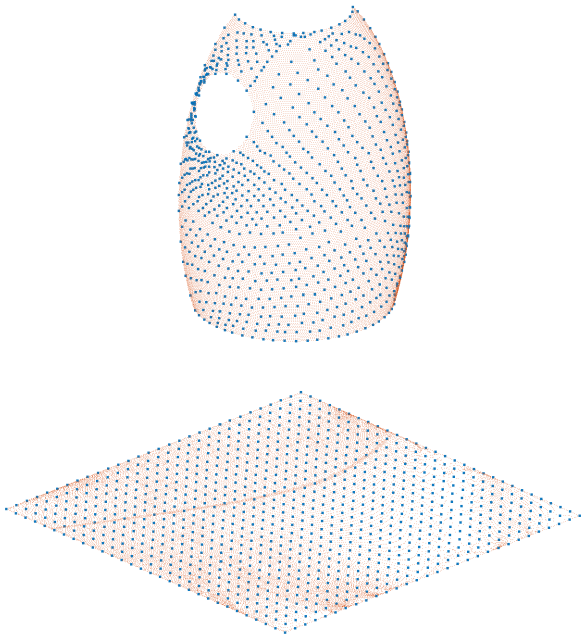
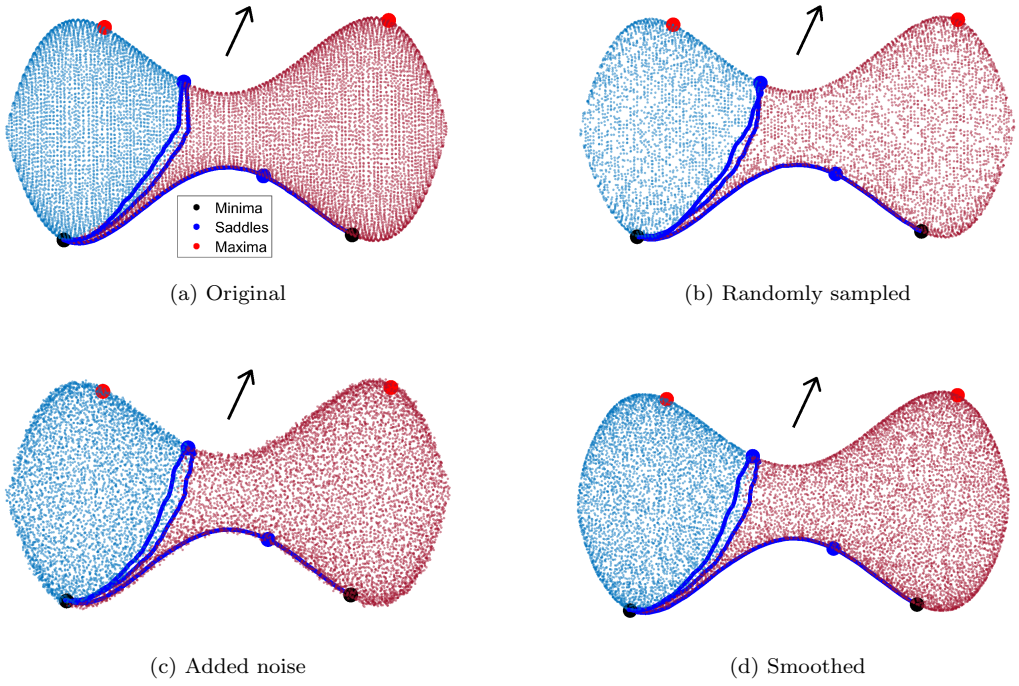


Figure 14: Parametrization by a rectangle of the rightmost 2-cell of Figure 13. The bounding 1-cells are mapped isometrically to a flat rectangle and then interior points are obtained using the neighboring relationships of the cloud. The 2-cell consisting of 27 000 points (shown in red in Figure 13) is down-sampled interpolating linearly to a cloud of 900 points (shown here in blue in the parametrizing rectangle and in the 2-cell).

In Figure 14 we show a parametrization by a rectangle of one of the 2-cells of the vest (the red one on the right in Figure 13). This is done as explained in Section 6: the bounding 1-cells are mapped isometrically to a flat rectangle (notice that we consider the 1-cells number 6 and 6' as different) and then interior points are obtained using the neighboring relationships of the cloud (taking care of removing neighbors at opposite sides of the 1-cell number 6). Finally, the 2-cell consisting of 27 000 points (shown in red in Figure 14) is down-sampled using the parametrization and interpolating linearly to a cloud of 900 points (shown in blue in Figure 14).



(a) Original

(b) Randomly sampled

(c) Added noise

(d) Smoothed

Figure 15: Sampled dumbbells: the black line is the direction of the height function; maxima are painted in red, minima in black; 1-cells corresponding to saddles are outlined in blue. In the four figures we can see our algorithm applied to different point-clouds of the same underlying surface with varying amounts of uniformity or noise.

Dumbbell

Figure 15(a) shows our algorithm applied to a point-sample from a dumbbell (without boundary) embedded in $\mathbb{R}^3$. The algorithm correctly detects 2 local maxima, 2 local minima and 2 saddle points for the height function depicted in the figure. Out of a point cloud of 12 000 points, a decomposition of the surface into 6 Morse cells is found (two 0-cells: the minima, two 1-cells associated to the saddle points and two 2-cells, one for each maximum).

This example cloud allows us to study the robustness of the decomposition algorithm with respect to different amounts of noise. Notice that noise in the location of the points can be decomposed in two components: tangential noise and normal noise to the underlying surface. Tangential noise is equivalent to moving the point samples along the surface and hence can be seen as resampling. In Figure 15(b) we show our algorithm applied to a random resampling of the original cloud which reduces the number of points to 7000.

As can be seen in the figure, even though the cloud is no longer uniform, the result is the exactly the same topological decomposition that we had before.

On the other hand, normal noise is more challenging to handle. As mentioned in Section 4.2 what we do in this case is to use a discrete curvature filter which smooths the point-cloud and allows the application of the reconstruction method, see Figure 15(d). When applied a small number of times this filter defines a bijection of the original cloud and the filtered one and hence the decomposition we find is still valid for the original noisy sample shown in Figure 15(c).

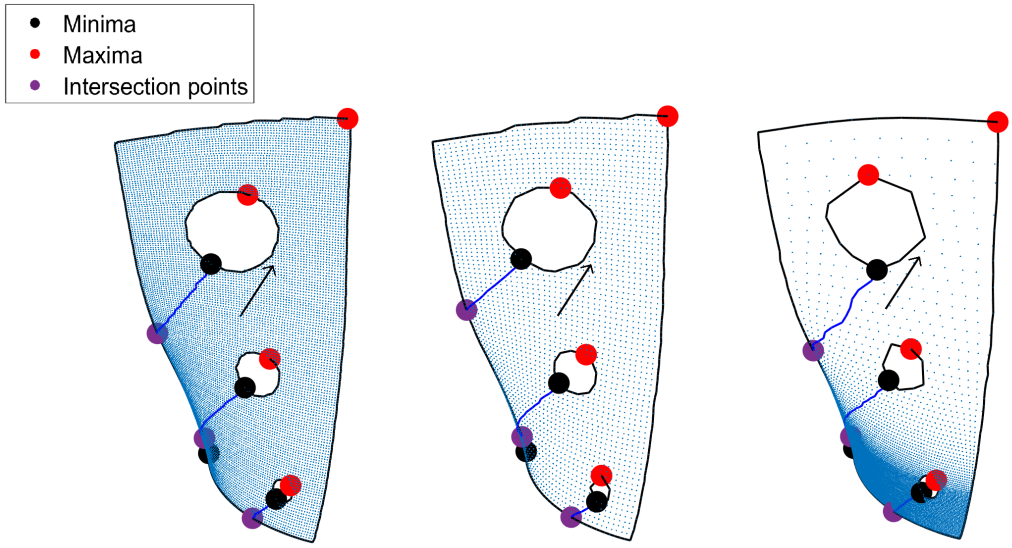


Figure 16: A 3D-scan of an aircraft turbine engine fan blade: the black line is the direction of the height function; maxima are painted red, minima in black; 1-cells are outlined in blue and the boundary curves in black. The purple point where the 1-cells meet is added to the decomposition. Notice that even with 3 different sampling densities (from left to right 11000, 3000 and 7000 points) our reconstruction is robust and we get the same topological decomposition.

Turbine engine fan blade

Figure 16 shows our algorithm applied to 3 point-clouds with different densities (approx. 3000, 7000 and 11000 points) coming from a 3D-scan of a real aircraft turbine engine fan blade, 640 mm long and 300 mm wide. For more details on how the data was collected and processed, we refer the reader to [[22]].

After detecting and parametrizing the boundary successfully, the algorithm correctly detects 1 local maxima, 3 boundary maxima, 1 local minima and 3 boundary minima for the height function depicted in the figure (the same for all 3 samples). All critical points are located at the boundary. Moreover, 3 new points (shown in purple) are added where the 1-cells meet each other or the boundary curves (see Remark 5). Then, a decomposition of the surface into 12 Morse cells is found (seven 0-cells, ten 1-cells and one 2-cell). With these example clouds we highlight the robustness of our algorithm when it faces non-uniform distributions of data points (this time for a surface with boundary).

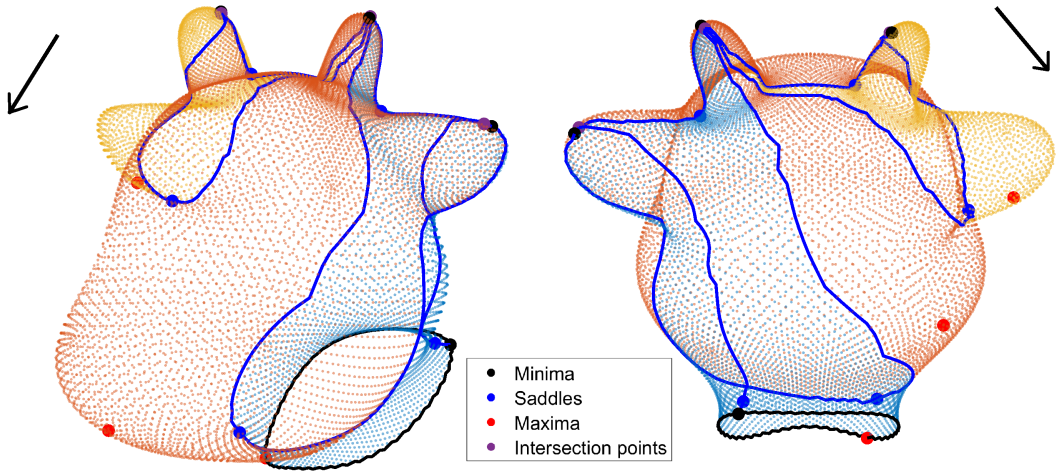


Figure 17: Two views of a sampled cow head: the black line is the direction (downwards in this case) of the height function; maxima are painted in red, minima in black; 1-cells corresponding to boundary minima and saddles are outlined in blue and the boundary curves in black. Three purple points where a 1-cell meets a boundary curve are added to the decomposition. The high variation in curvature of the cow's horns and ears causes the appearance of several saddle points.

Cow head

Figure 17 shows two views our algorithm applied to a sample of 13 000 points from an embedded cow-head in $\mathbb{R}^3$. After successfully detecting and parametrizing the boundary curve in the neck, the algorithm correctly identifies 3 local maxima, 3 local minima, 1 boundary minimum and 5 saddle points for the height function (pointing downwards) shown in the figure.

Additionally, 3 new points (shown in purple) are introduced where 1-cells intersect each other (see Remark 5). Then, a decomposition of the surface into 19 Morse cells is found (seven 0-cells, nine 1-cells and three 2-cells). Notice that the high variations in curvature of the cow's horns and ears cause the appearance of several saddle points which test the detection capabilities of the algorithm.

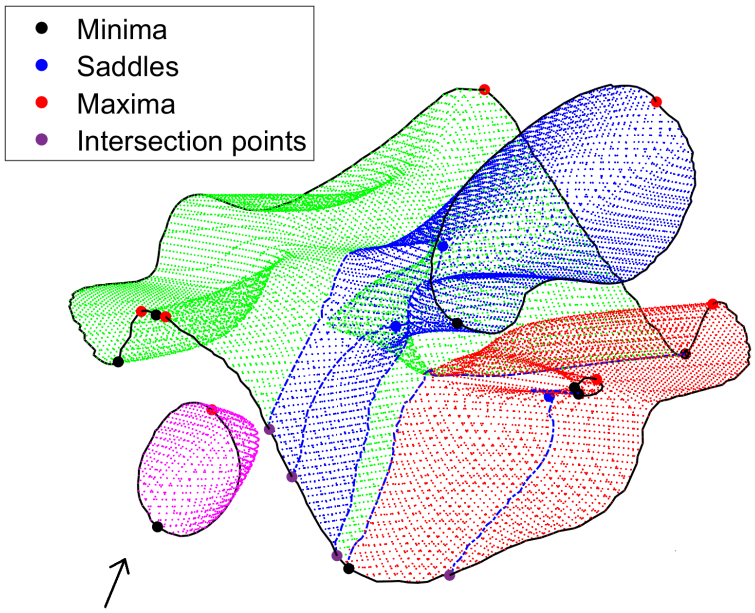


Figure 18: A sampled algebraic surface: the black line is the direction of the height function; maxima are painted red, minima in black; 1-cells are outlined in blue and the boundary curves in black. The purple points where the 1-cells meet are added to the decomposition.

Algebraic rosette

Figure 18 shows our algorithm applied to a cloud of 23 000 points coming from the zeros of a random and sparse polynomial of degree 8 in 3 variables. This means that each point (x, y, z) of the cloud satisfies an implicit equation of the form $\sum_{a,b,c=0}^{8} q_{abc} x^a y^b z^c = 0$ where most of the coefficients $q_{abc} \in \mathbb{Q}$ are zero and chosen at random for $a, b, c \in \{0, 1, \ldots, 8\}$. After detecting and parametrizing the boundary successfully, the algorithm correctly finds

a decomposition of the surface into 32 Morse cells: twelve 0-cells, sixteen 1-cells and four 2-cells. See the Supplementary Video 2 for a 3D view of the decomposition were all Morse cells and critical points can be appreciated.

Moving tank top

Figure 19 shows our algorithm applied to a cloud of 13 000 points coming from the simulation of a moving tank top with the cloth model described in [6]. After detecting and parametrizing the boundary successfully, the algorithm correctly finds a decomposition of the surface into 16 Morse cells: six 0-cells, nine 1-cells and one 2-cell. This point-cloud tests the robustness of our algorithm in several ways: it has very long and intricate boundary curves, a lot of local variation in mean curvature and a highly non-uniform distribution of points.

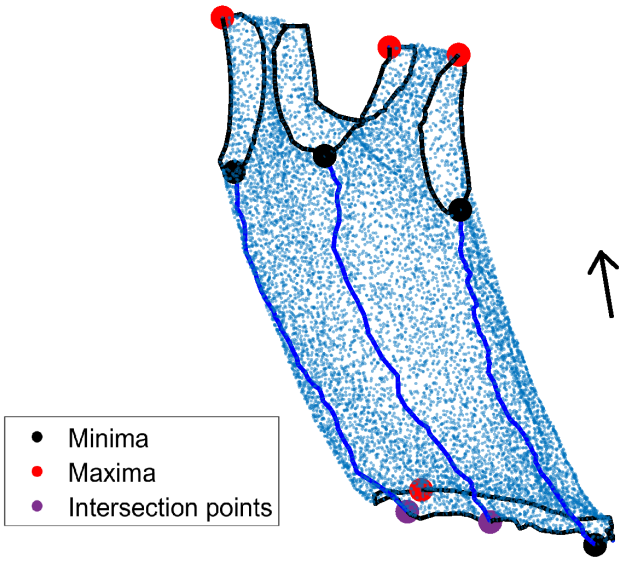


Figure 19: A sampled tank top: the black line is the direction of the height function; maxima are painted red, minima in black; 1-cells are outlined in blue and the boundary curves in black. The purple points where the 1-cells meet are added to the decomposition.

3D-scan of pants

Figure 20 illustrates our algorithm applied to a 3D scan of a pair of real jeans. The scan was conducted using an *Artec Eva* professional handheld 3D scanner while a person was wearing the garment.

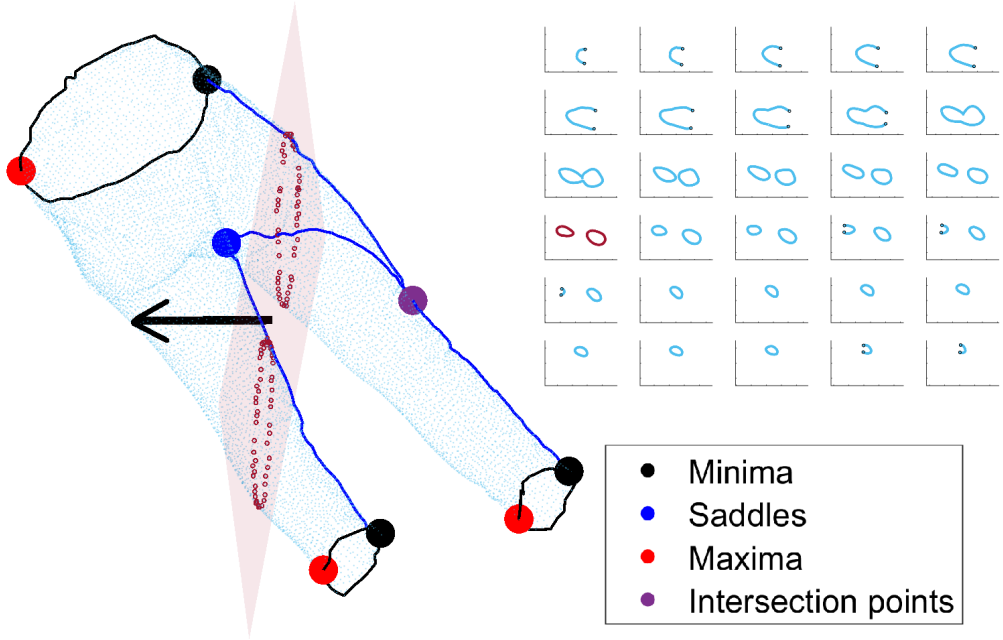


Figure 20: A 3D-scan of real pants: the black line is the direction of the height function; maxima are painted red, minima in black, saddle points are painted blue; 1-cells are outlined in blue and the boundary curves in black. The purple point where the 1-cells meet is added to the decomposition. On the bottom we plot the level-set curves obtained by intersecting the surface with planes perpendicular to the height function, with one such example curve shown in red.

The 3D scan was processed with the proprietary software *Artec Studio* to generate a point-cloud sample. To apply our algorithm and reduce irrelevant details and noise, we down-sampled the initial cloud of over 500,000 points to 11,000 using a *box-grid filter*. This filter works by computing an axis-aligned bounding box for the entire point-cloud and dividing it into grid boxes. The points within each grid box are merged by averaging their positions.

Despite the down-sampling, the cloud still retains significant detail (e.g., wrinkles), which increases the number of critical points and consequently, the number of Morse cells. After calculating the neighbors for each point as described in Section 4.1, we apply the *discrete-curvature filter* from Section 4.2. As detailed there, this involves replacing each point v with a weighted average of its position and that of its neighbors (using $\alpha = 0.4$ and applying the filter 8 times). This filter establishes a bijection between the original

cloud and the filtered one, preserving the topology of the underlying surface. Therefore, the decomposition we obtain from the filtered case remains valid and topologically accurate for the original cloud, as shown in Figure 20.

The algorithm successfully detects 1 local maximum, 2 boundary maxima, 2 local minima, 1 boundary minimum, and 1 saddle point for the height function depicted in Figure 20. A decomposition of the surface into 9 Morse cells is obtained: four 0-cells (the minima and an additional point due to the intersection of original 1-cells), six 1-cells (associated with the saddle points, boundary minima, and boundary curves), and one 2-cell corresponding to the local maximum. At the bottom of Figure 20, we display the level set curves $\Gamma(c) = G_{\text{down}} \cap H_c$ (see Section 5.2). Notice that for this point-cloud, several of the potential local transformations of level-set curves for surfaces with boundaries, as shown in Figure 10, are encountered.

8. Conclusions, limitations and further work

In this work, we addressed the problem of reconstructing surfaces with unknown topology from point-clouds. We introduced an algorithm capable of reconstructing a surface S by obtaining a Morse cellular decomposition of it from a set of sampled points. The algorithm can be applied to surfaces in $\mathbb{R}^N$, for any ambient dimension N, whether the surface has a boundary or not. We demonstrated the versatility of our approach by reconstructing several surfaces, both with and without boundaries, each presenting distinct challenges. For instance, the *rosette* is an algebraic surface which has two distinct connected components, very large and very tiny boundary curves and moreover nontrivial curvature everywhere which causes the appearance of many critical points. On the other hand, the *pants* were obtained as a 3D-scan of a real pair of jeans and thus its point-cloud presented wrinkles, noise and an irregular density distribution of points. For all surfaces a global piecewise decomposition was found, with a small number of pieces in the examples examined.

From the cellular decomposition the topology of the surface was computed easily (see Table 1). Moreover, the Morse-Smale complex defined by the cells provides loops realizing any homology class. In the point-clouds we studied, while not necessarily minimal, the length of these loops is often of the same order of magnitude as that of the shortest representative in the homology class. The algorithm is *topologically robust*, i.e. it captures the topological features of the sampled surface with a size greater than the average distance

between sample points. To obtain this decomposition, we presented a novel method to determine how (discrete) Morse cells attach to each order (see Figure 13). Furthermore, for the case of surfaces with boundary, we developed a new graph theoretical method to determine robustly boundary points of the cloud.

8.1. Limitations of the proposed algorithm

Although our algorithm performs well across a range of real and synthetic examples, it has some limitations which we now discuss. First, while the method is resilient to in-plane noise and irregular sampling densities, it is more sensitive to normal noise, i.e. noise perpendicular to the surface. To mitigate this, we can apply a discrete curvature filter, but if the noise is too large, the resulting cloud may no longer represent a sample of any underlying smooth surface, compromising the validity of the (smooth) Morse decomposition: in practice we can no longer reconstruct the curves obtained in the level set sections and hence we can not detect the critical points of the given Morse function. Furthermore, in regions with high variation in mean curvature, especially in surfaces with rapidly changing geometry, we may require a greater number of level set intersections to correctly capture saddle points. This increases the computational cost and resolution requirements of the cloud, particularly when fine geometric features are present.

8.2. Further work

Further work involves studying more deeply the parametrization of the 2-cells by flat regions of the plane. If a given 2-cell is too far away from being flat (e.g. because of high curvature) the parametrization method described in Section 6 may produce interior points in the region with non-uniform densities and thus it may be better to subdivide the cell into smaller and flatter pieces and afterwards parametrize each piece independently. On the other hand, as already commented in Remark 6, it may be interesting to use flat regions whose boundary is a polygon where its sides are mapped isometrically to each of the bounding 1-cells, which in turn has the advantage of allowing the gluing of the resulting parametrizations of the 2-cells into a global piece-wise defined and continuous parametrization of the entire surface.

We also plan to test the reconstruction algorithm with more *real* point-clouds of various textiles (e.g., those obtained via a 3D scanner). This would enable the creation of a reconstructed textile database that could later be simulated using a physical cloth model. In addition, we aim to extend the

point-cloud reconstruction algorithm for surfaces to higher-dimensional manifolds by iterating the hyperplane sections, reconstructing the manifold from lower-dimensional slices. A compelling example of this would be the study of real algebraic varieties of any dimension, where point-cloud samples can be derived from their equations and refined as needed. We hope this will allow our method to detect a Whitney stratification, leading to the Morse cellular decomposition of the variety, as in [[15]], through entirely numerical means.

9. Acknowledgments

This research work was funded by the European Commission - NextGenerationEU, through Momentum CSIC Programme: Develop Your Digital Talent. Franco Coltraro is staff hired under the Generation D initiative, promoted by Red.es, an organization attached to the Ministry for Digital Transformation and the Civil Service, for the attraction and retention of talent through grants and training contracts, financed by the Recovery, Transformation and Resilience Plan through the European Union's Next Generation funds. F. Coltraro was also partially supported by the ClothIRI (CSIC 202350E080) project and the RobIRI 2021-SGR-00514 AGAUR project. Jaume Amorós and Maria Alberich-Carramiñana have been partially supported by the projects PID2019-103849GB-I00 and PID2023-146936NB-I00 financed by the Spanish State Agency MICIU/AEI/10.13039/501100011033 and by ERDF/EU, and by the GEOMVAP 2021-SGR-00603 AGAUR project.

References

[1] M. Alberich-Carramiñana, J. Amorós, F. Coltraro, C. Torras, and M. Verdaguer. Morse cell decomposition and parametrization of surfaces from point-clouds. *Proceedings of XVII EACA 2022 (Encuentro Álgebra Computacional y Aplicaciones)*, pages 35–38, 2022.

[2] J. Cao, H. Chen, J. Zhang, Y. Li, X. Liu, and C. Zou. Normal estimation via shifted neighborhood for point cloud. *Journal of Computational and Applied Mathematics*, 329:57–67, 2018. The International Conference on Information and Computational Science, 2–6 August 2016, Dalian, China.

[3] F. Cazals, A. Roth, C. H. Robert, and M. Christian. Towards morse theory for point cloud data. *Research Report RR-8331, INRIA.*, page 37, 2013.

[4] A. Colomé and C. Torras. Dimensionality reduction for dynamic movement primitives and application to bimanual manipulation of clothes. *IEEE Transactions on Robotics*, 34(3):602–615, 2018.

[5] A. Colomé and C. Torras. *Reinforcement Learning of Bimanual Robot Skills.* Volume 134 of Springer Tracts in Advanced Robotics., 2020.

[6] F. Coltraro, J. Amorós, M. Alberich-Carramiñana, and C. Torras. An inextensible model for the robotic manipulation of textiles. *Applied Mathematical Modelling*, 101:832–858, 2022.

[7] F. Coltraro, J. Amorós, M. Alberich-Carramiñana, and C. Torras. Reconstruction of sampled surfaces with boundary via Morse theory. In J. Gimeno Sancho and M. Comino Trinidad, editors, *Spanish Computer Graphics Conference (CEIG)*. The Eurographics Association, 2023.

[8] K. Crane, U. Pinkall, and P. Schröder. Robust fairing via conformal curvature flow. *ACM Trans. Graph.*, 32(4), jul 2013.

[9] T. K. Dey. *Curve and Surface Reconstruction: Algorithms with Mathematical Analysis*. Cambridge Monographs on Applied and Computational Mathematics, 2006.

[10] A. Doumanoglou, J. Stria, G. Peleka, I. Mariolis, V. Petrik, A. Kargakos, L. Wagner, V. Hlavac, T.-K. Kim, and S. Malassiotis. Folding clothes autonomously: A complete pipeline. *IEEE Transactions on Robotics*, 32(6):1461–1478, 2016.

[11] M. S. Floater and K. Hormann. Parameterization of triangulations and unorganized points. In *Tutorials on Multiresolution in Geometric Modelling*, 2002.

[12] J. Gao, R. Sarkar, and X. Zhu. Morse-smale decomposition , cut locus and applications in sensor networks. *Pre-print*, 2008.

[13] I. Garcia-Camacho, M. Lippi, M. C. Welle, H. Yin, R. Antonova, A. Varava, J. Borràs, C. Torras, A. Marino, G. Alenyà, and D. Kragic. Benchmarking bimanual cloth manipulation. *IEEE Robotics and Automation Letters*, 5(2):1111–1118, 2020.

[14] T. Gensane. Dense packings of equal spheres in a cube. *The Electronic Journal of Combinatorics*, pages R33–R33, 2004.

[15] M. Goresky and R. MacPherson. *Stratified morse theory*. Springer Verlag, 1988.

[16] M. Hirsch. *Differential Topology*. Springer Verlag, 1976.

[17] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. In *Proceedings of the 19th annual conference on computer graphics and interactive techniques*, pages 71–78, 1992.

[18] Z. Huang, Y. Wen, Z. Wang, J. Ren, and K. Jia. Surface reconstruction from point clouds: A survey and a benchmark, 2022.

[19] R. Jangir, G. Alenya, and C. Torras. Dynamic cloth manipulation with deep reinforcement learning. *IEEE International Conference on Robotics and Automation (ICRA), Paris, France*, pages 4630–4636, 2020.

[20] Y. Li, Y. Yue, D. Xu, E. Grinspun, and P. Allen. Folding deformable objects using predictive simulation and trajectory optimization. *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 6000–6006, 12 2015.

[21] F. M. Maley, D. P. Robbins, and J. Roskies. On the areas of cyclic and semicyclic polygons. *Advances in Applied Mathematics*, 34(4):669–689, 2005.

[22] C. Mineo, S. G. Pierce, and R. Summan. Novel algorithms for 3d surface point cloud boundary detection and edge reconstruction. *Journal of Computational Design and Engineering*, 6(1):81–91, 2019.

[23] J. R. Munkres. *Elements of algebraic topology*. Addison-Wesley, 1984.

[24] A. Nealen, M. Müller, R. Keiser, E. Boxerman, and M. Carlson. Physically based deformable models in computer graphics. *Computer Graphics Forum*, 25(4):809–836, 2006.

[25] J. Sanchez, J.-A. Corrales, B.-C. Bouzgarrou, and Y. Mezouar. Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey. *The International Journal of Robotics Research*, 37(7):688–716, 2018.

[26] J. R. Taylor. *Classical Mechanics*. University Science Books, 2005.

[27] H. Yin, A. Varava, and D. Kragic. Modeling, learning, perception, and control methods for deformable object manipulation. *Science Robotics*, 6(54):eabd8803, 2021.

[28] X. Zhu, R. Sarkar, and J. Gao. Topological data processing for distributed sensor networks with morse-smale decomposition. *IEEE INFOCOM 2009*, pages 2911–2915, 2009.

Appendix A. Morse theory for surfaces with boundary

In this appendix we give more details on how Morse theory is applied to surfaces with boundary following the notions introduced in Section 3.2 and the tools developed in [[15]]. Since the boundary of a surface with boundary is itself a manifold, we will adopt a *Whitney stratification* dividing the surface into two **strata**: the first $E_1 = \partial S$ being the boundary and the second $E_2 = \text{int}(S)$ the interior. Each stratum will have as before Morse data, but this time apart from the sets $(A(p), B(p))$ defined as in Definition 1, which we will call *tangential data*, we will have also *normal data*.

Definition 3 (Tangential and normal data)**.** Let $p \in S$ be a critical point for some stratum E_i of the Morse function $f|_{E_i}$. Then

1. The tangential Morse data are the pair of sets (A_T, B_T) defined as in Definition 1 for $f|_{E_i}$.

2. The normal Morse data are the sets (A_N, B_N) given by

$$A_N(p) := \mathcal{N}(p) \cap f^{-1}\left([f(p) - \epsilon, f(p) + \epsilon]\right),$$
$$B_N(p) := \mathcal{N}(p) \cap f^{-1}\left(f(p) - \epsilon\right).$$

   where $\mathcal{N}(p) = S \cap D(p)$ is called the *normal slice* at p and $D(p) \subset \mathbb{R}^N$ is a sufficiently small closed disk around p of dimension $N - \dim(E_i)$ transversal to E_i at p and the value of $\epsilon > 0$ is such that there are no more critical points in $f^{-1}[f(p) - \epsilon, f(p) + \epsilon]$.

Notice that by construction $E_i \cap D(p) = \{p\}$, and as before $B_{N,T} \subseteq \partial A_{N,T}$.

Remark 7. When $p \in \text{int}(S)$ and $N = 3$ then $D(p)$ is homeomorphic to a piece of curve normal to S at p and hence $\mathcal{N}(p) = \{p\}$. Therefore $(A_N, B_N) = (p, \emptyset)$. It is not hard to see that this is also the case when $N > 3$.

We are now ready to state the main theorem of stratified Morse theory [[15]] (SMT theorem, pages 6-8).

Theorem 2 (Goresky-MacPherson)**.** *As $c \in \mathbb{R}$ increases two things can happen:*

A *If between $c_1 < c_2$ there are no critical points of f, then $f_{\leq c_1}$ and $f_{\leq c_2}$ are diffeomorphic.*

B *If there is a critical point $p \in S$ such that $c_1 < f(p) < c_2$, then $f_{\leq c_2}$ is obtained from $f_{\leq c_1}$ by performing a surgery around p with Morse data (A, B) diffeomorphic to the topological product*

$$(A_N, B_N) \times (A_T, B_T) = (A_N \times A_T, A_N \times B_T \cup B_N \times A_T),$$

i.e. $f_{\leq c_2} \simeq (f_{\leq c_1} \cup A) / \sim$ where the equivalence relation is given by identifying $B \subseteq f_{\leq c_1}$ with points of $\partial A \supseteq B$.

Remark 8. We already established that at interior critical points of S, we have $(A_N, B_N) = (p, \emptyset)$. Therefore in that case the above product is trivial and the attachment maps are the same ones described earlier.

The new cases occur when p is a critical point of $f|_{\partial S}$ lying on ∂S. Since ∂S is a one-dimensional curve, $p \in \partial S$ can only be a maximum or a minimum. Nevertheless, depending on whether p is also a local minima (resp. maxima) of f or just of $g = f|_{\partial S}$, we will have four different cases (see Figure A.21). Recall also that saddle points can not be located at ∂S by definition.

Remark 9. We will denote by $(\blacksquare, \sqcup, |\ |, \_)$ a quadrangular 2-cell, all its sides minus the top one, two lateral sides and the bottom side, respectively.

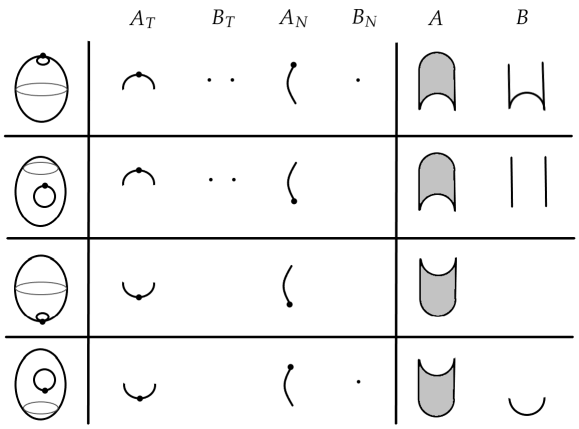


Figure A.21: Four types of critical points located at the boundary and their tangential and normal Morse data.

The four new types of critical points that we can have are:

Maxima of $f|_{\partial S}$: In this case A_T is a closed concave piece of curve (which we denote by $\cap$) and B_T two points, i.e. $(A_T, B_T) = (\cap, \,.\,.)$. We now have two sub-cases:

1. p is also a local maximum of f. Then A_N is a closed interval and B_T one point, i.e. $(A_N, B_N) = (\,|\,,.\,)$. Therefore the topological product is
$$(A, B) = (\cap, \,.\,.) \times (\,|\,,.\,) = (\blacksquare, \sqcup)\,,$$
and the three of the sides of ∂A where p is not present are attached to B. During this surgery a 1-cell (containing p) is attached to the boundary and a 2-cell to the interior (closing a void in the process).
2. p is not a local maximum of f (only of $f|_{\partial S}$). Then A_N is again a closed interval but B_T is empty, i.e. $(A_N, B_N) = (\,|\,, \emptyset)$. Then the Morse data is
$$(A, B) = (\cap, \,.\,.) \times (\,|\,, \emptyset) = (\blacksquare, \;|\,|\,)\,,$$
and the two opposite sides of ∂A where p is not present, attach to B. This surgery is equivalent to attaching a 1-cell to the boundary (but no 2-cell is attached to the interior).

Minima of $f|_{\partial S}$: In this case A_T is a closed convex piece of curve (which we denote by $\cup$) and B_T is empty, i.e. $(A_T, B_T) = (\cup, \emptyset)$. We again have two sub-cases:

1. p is also a local minimum of f. Then A_N is a closed interval and B_T is empty, i.e. $(A_N, B_N) = (\ |\ , \emptyset)$. Therefore the topological product is
$$(A, B) = (\cup, \emptyset) \times (\ |\ , \emptyset) = (\blacksquare, \emptyset)\,.$$
In this case there is no surgery, the cell A just appears. This cell retracts to a point, which will be a 0-cell in the cell complex.

2. p is not a local minimum of f (only of $f|_{\partial S}$). Then A_N is a closed interval and B_T is a point, i.e. $(A_N, B_N) = (\ |\ , .\ )$. The Morse data is
$$(A, B) = (\cup, \emptyset) \times (\ |\ , .\ ) = (\blacksquare, \_)\,.$$
and the opposite side of ∂A where p is, attaches to B. This surgery is equivalent to attaching a 0-cell to the boundary and a 1-cell to the interior.