STAG: Smart Tools and Applications in Graphics (2023)
K. Lupinetti, F. Banterle, N. Capece, and U. Erra (Editors)

An approach to the decomposition of solids with voids via Morse theory

Juan Pareja-Corcho[1] †, Diego Montoya-Zapata[1,2], Aitor Moreno[1], Carlos Cadavid[3], Jorge Posada[1], Ketzare Arenas-Tobon[2] and Oscar Ruiz-Salguero[2]

[1]Vicomtech Foundation, Basque Research and Technology Alliance, Spain
[2]CAD CAM CAE Laboratory, Universidad EAFIT, Colombia
[3]Mathematics and Applications, Universidad EAFIT, Colombia

Abstract

The decomposition of solids is a problem of interest in areas of engineering such as feature recognition or manufacturing planning. The problem can be stated as finding a set of smaller and simpler pieces that glued together amount to the initial solid. This decomposition can be guided by geometrical or topological criteria and be applied to either surfaces or solids (embedded manifolds). Most topological decompositions rely on Morse theory to identify changes in the topology of a manifold. A Morse function f is defined on the manifold and the manifold's topology is studied by studying the behaviour of the critical points of f. A popular structure used to encode this behaviour is the Reeb graph. Reeb graph-based decompositions have proven to work well for surfaces and for solids without inner voids, but fail to consider solids with inner voids. In this work we present a methodology based on the handle-decomposition of a manifold that can encode changes in the topology of solids both with and without inner voids. Our methodology uses the Boundary Representation of the solid and a shape similarity criteria to identify changes in the topology of both the outer and inner boundary(ies) of the solid. Our methodology is defined for Morse functions that produce parallel planar level sets and we do not consider the case of annidated solids (i.e. solids within other solids). We present an algorithm to implement our methodology and execute experiments on several datasets. Future work includes the testing of the methodology with functions different to the height function and the speed up of the algorithm's data structure.

CCS Concepts
• ***Computing methodologies*** $\rightarrow$ ***Shape analysis; Volumetric models; Mesh models;***

1. Introduction

The decomposition of a piece into smaller and simpler pieces is a problem of interest in many applications related to computer graphics: collision detection [MG09], feature detection [TTF04], real time animation [KS00], surface matching [LGQ09], manufacturing planning [GXZ*23], amongst others. Many decomposition (also called *segmentation*) algorithms, specially for surface-based models, are found in literature (see surveys in [APP*07; HW18]). A possible cause for this proliferation of approaches is that algorithms are tailored to the specific needs of the target application and there is no universal solution to the problem.

A decomposition algorith is characterized by its *decomposition criteria*, that is a common trait shared by all the subsets that make up the initial set (this is known as the *common base domain* [KZW12]). For most applications this common base can be of *geometrical*, *functional* or *topological* nature. Geometrical decompositions proceed considering only the geometric characteristics of the model and decompose the model into geometrically similar components. For the case of surfaces, common examples are the planarity-based and curvature-based segmentations [SSGH01; WY11]. Functional (also called *meaningful*) decompositions divide the model into *meaningful* components [HC*12], that is subsets that have a special significance in the interpretation of the model

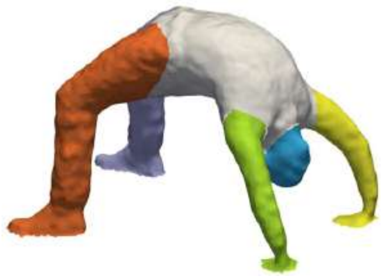


Figure 1: *Example of the decomposition of a piece into smaller and simpler constituent pieces.*

† Corresponding author. E-mail: jcpareja@vicomtech.org

DOI: 10.2312/stag.20231302

(e.g. the decomposition of the human body into head, torso, legs, arms). Topological decompositions divide the model by considerings its topological features [MG21; SJ17]. These approaches usually study the changes in the topology of a collection of level sets defined by a real-valued function $f : M \to \mathbb{R}$ on the piece M and rely heavily on Morse theory. Common examples of the topological approach are the methods based on the *Reeb graph* decomposition [SJ17; HDL16]. These methods are well defined for surface-based models but literature regarding solid models (specially when there is more than one boundary component) is rather scarce. We will examine these methodologies in the next section.

In this work we present an approach to a topological decomposition of solids (embedded three-dimensional manifolds with boundaries) with and without inner voids. Our methodology is defined for Morse functions that produce parallel planar level sets and we do not consider the case of annidated solids (i.e. solids within other solids). We must, however, first examine the principles of Morse theory on manifolds to show that current methods are unsuitable to deal with the decomposition of solids with more than one boundary component.

2. Morse theory on manifolds

Let M be a smooth manifold and $f : M \to \mathbb{R}$ a smooth function defined on M. We have the following definitions:

Level sets: The level set of f on M at a value $a \in \mathbb{R}$ is $\{p \in M : f(p) = a\}$. We denote the level set of f at value a as $\Pi_{f,a}$. Connected parts of a level set are called *level set components*.

Critical points and values: A critical point of the function f is a point p where $dF(p) = 0$. A real c is called a critical value of f if the pre-image $f^{-1}(c)$ contains a critical point of f. Additionally, a critical point is said to be non-degenerate if it's Hessian is non-degenerate.

Morse functions: Function f is said to be a Morse function if it satisfies the following conditions:

1. all critical points of f are non-degenerate.

$$\forall p \in \mathcal{M} : \nabla_{\mathcal{M}} f(p) = 0 \to \det(\mathbf{H}_{\mathcal{M}} f(p)) \neq 0 \tag{1}$$

2. for all pairs (p,q) of different critical points of f, $f(p) \neq f(q)$.

Morse lemma: Assume p to be a non-degenerate critical point of f. There is a *local* coordinate system $\{X_1,\dots,X_n\}$ on a neighborhood N_p of p, such that on N_p:

$$f(X_1,\dots,X_n) = -X_1^2 - \cdots - X_k^2 + X_{k+1}^2 + \cdots + X_n^2 \tag{2}$$

In this formulation, k is called the index of f at p. The Morse lemma introduces a way to classify critical points of a Morse function defined on a manifold and therefore opens the door to the classification of the manifold itself by studying the evolution of these critical points. In the case of two-dimensional manifolds (surfaces), the Morse lemma classifies the critical points of a Morse function in three kinds: minimum ($k = 0$), saddle ($k = 1$) and maximum ($k = 2$). See Figure 2. A three-dimensional manifold has four types of non-degenerate critical points: minimum ($k = 0$), 1-saddle ($k = 1$), 2-saddle ($k = 2$) and maximum ($k = 3$). Critical points (and their index) can be identified by their effects on the level set sequence of f: minimum points create contours *ex nihilo*, saddle points join or separate contours and maximum points eliminate contours.

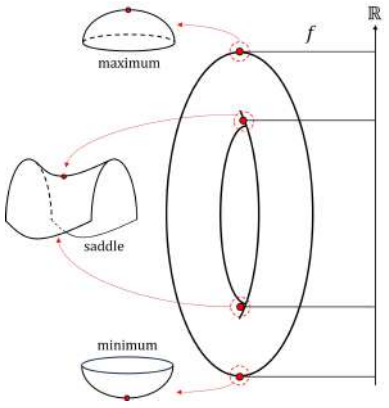


Figure 2: *Local shapes of the critical points of a Morse function f on the standing torus closed surface.*

In Figure 2, if we were to consider the standing solid torus (an embedded 3-manifold with boundary) instead of its boundary surface intuitively we notice that the points of interest (those where topology changes) are still located on the boundary surface of the solid, at least for the height function $f(x,y,z) = z$. This fact opens the door to analyze solids by considering only their boundary components and the relationships between them.

2.1. Handle decomposition of manifolds

Simply put, a compact manifold can be expressed as a sum of submanifolds called handles [Boh19]. Let M be a smooth manifold and $f : M \to \mathbb{R}$ be a Morse function defined on M. We define the following:

Lower level sets: The lower level set M_t of f in M at value t is the set $M_t = \{x \in M : f(x) \leq t\}$.

Theorem 1: For two reals a,b with $a < b$ if f has no critical values in the interval $[a,b]$, then the manifolds M_a and M_b are diffeomorphic.

The previous theorem implies that the topology of M_t does not change as parameter t passes through regular (non-critical) values of f.

Theorem 2: Let p be a critical point of f and $f(p) = t$ be its critical value. Let ε be a number small enough so that f does not have critical values in $[t-\varepsilon, t+\varepsilon]$. The manifold $M_{t+\varepsilon}$ can be obtained by *gluing* a handle (a contractible smooth manifold) to the manifold $M_{t-\varepsilon}$.

The previous theorem implies that the topology of M_t changes as t passes through a critical value of f. This change can be captured and incorporated into M_t by gluing a new piece called *handle*. Since a Morse function defined on a compact manifold admits only finitely many critical points [BGSF08], the topology of M_t changes a finite number of times and can be therefore decomposed by tracking the *handles* glued to the manifold as parameter t increases:

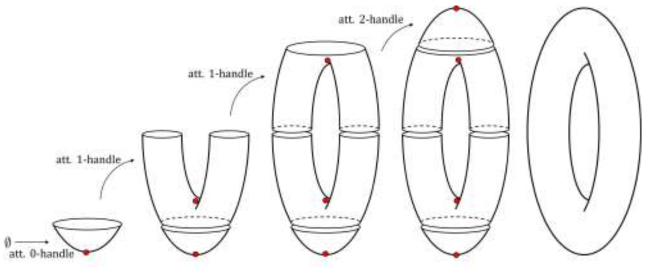


Figure 3: *Handle decomposition of the standing torus closed surface.*

Handle decomposition: A handle decomposition of a compact manifold M is a finite sequence of manifolds $W_0, \ldots, W_l$ such that:

1. $W_0 = \emptyset$,
2. W_l is diffeomorphic to M,
3. W_i is obtained from W_{i-1} by attaching a handle.

The necessary handle to obtain $W_i = M_{t+\varepsilon}$ from $W_{i-1} = M_{t-\varepsilon}$ depends on the topological change that took place in the interval $[t-\varepsilon, t+\varepsilon]$. In the case of closed surfaces this classification is straightforward as the handles are dictated by the values of the index of the critical point p with critical value $t \in [t-\varepsilon, t+\varepsilon]$. This means that to decompose a surface we only need to consider the attachment of 0-handles, 1-handles and 2-handles in a given sequence. See Figure 2 and 3 for an example. As we will detail later, the case for solids is not as simple. The reason for this is that, in the case of solids embedded in $\mathbb{R}^3$, changes in topology arise from both changes in the number of connected components in a level set and changes in the topology of the level set itself (i.e. changes in the genus of isosurfaces).

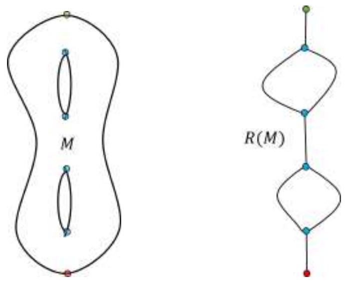
Most algorithms that effectuate a handle decomposition on closed surfaces exploit a data structure known as the *Reeb graph*. The Reeb graph represents the critical points of f and their indexes to encode the topology of a manifold. In the next section we briefly review the Reeb graph principles and its approaches to surfaces before dealing with the case of solids.

2.1.1. Reeb graphs

Reeb graph: Two points $p, q \in \mathcal{M}$ are equivalent if they belong to the same connected component of $f^{-1}(c)$ with $c = f(p) = f(q)$. The Reeb Graph of f, $R(f) = \mathbb{X}_{\sim}$, is the quotient space defined by this equivalence relation.

The nodes of the Reeb graph correspond to the critical points of f on M. Since f is Morse, every critical point of f with index 1 or 2 gives rise to a degree 1 node and every saddle of f gives rise to a degree 3 node. The nodes of $R(f)$ are points in which the topology of M changes. The arcs of $R(f)$ correspond to regions of M in which the topology of M does not change. See Figure 4.

In simple terms, Reeb graphs are obtained by contracting every level set component to a point and establishing the connectivity as dictated by the changes in the number of level set components between consecutive level sets. Numerous methodologies exist for the extraction of Reeb graphs of manifolds [TPT14]. Algorithms for the extraction of Reeb graphs in surfaces can be found



(a) (b)

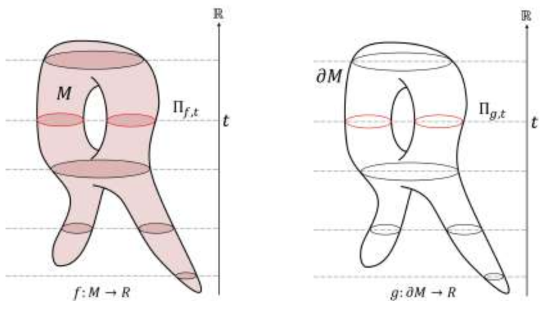
Figure 4: *Reeb graph of the standing double torus. (a) Manifold M. (b) Reeb graph of manifold M.*

in [SK91; BP12; CEH*03; Bia04; HR20]. The most efficient algorithms complexity-wise can be found in [Par12; HWW10]. Reeb graphs are popular tools to decompose a surface into its handle constituents [HDL16].

2.1.2. The case of solids in $\mathbb{R}^3$

In the case of the handle decomposition of solids, there are two distinct situations that require distinct treatments: solids with *one* boundary component and solids with *more than one* boundary component. Solids with more than one boundary component are said to have *voids*.

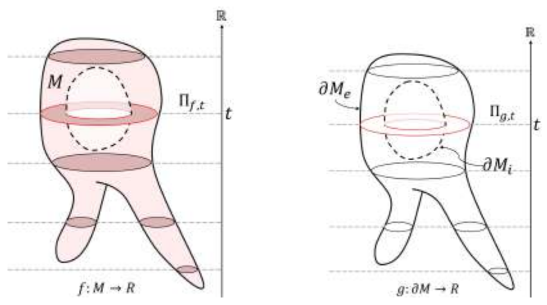
Let M be a solid (3-manifold with boundary) *without voids* in $\mathbb{R}^3$ and $f_M : M \to \mathbb{R}$ a Morse function defined on M. Consider only functions f that do not possess critical points in the interior of the solid. Let function g be the restriction $f|_{\partial M}$ of f to the boundary of M. If f is the height function, we get the scenario depicted in Figure 5. In this case, for every non-critical value t the level sets $\Pi_{f,t}$ and $\Pi_{g,t}$ are closely related. The level set components of $\Pi_{g,t}$ consist of closed curves on the boundary ∂M. These curves on ∂M bound regions of M that match with the level set components of $\Pi_{f,t}$. This means that the number of level set components in $\Pi_{f,t}$ and $\Pi_{g,t}$ is the same and therefore the topology of M can be studied by studying the topology of its boundary ∂M.



(a) (b)

Figure 5: *Relationship between solid level sets and boundary level sets for a solid without voids. (a) Solid level sets. (b) Boundary level sets.*

This *unequivocal relationship* between the solid level sets and the boundary level sets is used to produce decompositions of a solid by considering its boundary surface [DN11; TSK98; ACA*19; BGSF08]. Some problems may arise when the solid has tunnels with axis not parallel to the slicing plane. This is resolved by [SJ17] by combining the Reeb graph defined on the boundary surface with the Reeb graph defined on the volume itself. With this approach they are able to obtain a decomposition of the solid into solid handles.



(a) **(b)**

Figure 6: *Relationship between solid level sets and boundary level sets for a solid with voids. (a) Solid level sets. (b) Boundary level sets.*

Now consider M to be a solid *with voids* as the one depicted in Figure 6. In these solids there are critical values at which the topological genus of the isosurface changes without modifying the number of connected components in the level sets [BGSF08; SJ15]. Let $f_M : M \to \mathbb{R}$ be a Morse function defined on M. Let function g be the restriction $f|_{\partial M}$ of f to the boundaries of M. Notice that in this case the boundary ∂M of M is composed by two distinct surfaces: an interior boundary and an exterior boundary, which we will denote as ∂M_i and ∂M_e respectively. Consider now any non-critical value t and the solid level sets $\Pi_{f,t}$ and the boundary level sets $\Pi_{g,t}$. The level set components of $\Pi_{g,t}$ (curves on the boundary) do not bound a level set component of $\Pi_{f,t}$ in a one-to-one correspondence. Notice that both components of $\Pi_{g,t}$ bound the same component of $\Pi_{f,t}$, one from the inside and the other from the outside. This means that the unequivocal *component to component* relationship between the solid level sets and the boundary level sets is lost and therefore additional information is required to study these phenomena.

Some authors build Reeb-like graphs for these kind of solids by considering two orthogonal Morse functions defined on the solid and augmenting the Reeb graph with additional information regarding the topology of each level set [SJ15]. This approach, however, does not produce a decomposition of the solid into submanifolds.

If M is a solid *without voids* then it can be decomposed by considering the handle decomposition of its boundary surface. M can be reconstructed by attaching together *solid* versions of the 0-, 1- and 2-handle manifolds defined for 2-manifolds. The presence of genus-related critical values means that if M is a solid *with voids* then to produce a handle decomposition of M we need to augment the set of possible handles with pieces that consider such genus-related transitions. In this work we present an approach to the decomposition of solid with and without voids by considering the Boundary Representation (B-Rep) of the solid with voids.

3. Methodology

In the same fashion as in Section 2.1, the problem of the handle decomposition of solids can be stated as:

Solid decomposition: Given a solid M (embedded 3-manifold with boundary) and a slicing function f defined on M, find the sequence of manifolds $W_0, \ldots, W_k$ such that

1. $W_0 = \emptyset$,
2. W_k is homeomorphic to M,
3. W_i is obtained from W_{i-1} by attaching a solid handle,
4. Each solid handle attached incarnates a critical point of f in ∂M.

The stated problem requires us to find the sequence of solid handles such that attached together they form a solid that resembles the topological structure of the solid M. Notice that the problem statement requires that the critical values of f are unique (i.e. only one topological change is allowed between consecutive level sets). This means that for any two critical points p, q of f then $f(p) \neq f(q)$. Therefore, it is always possible to order the critical points of f in ascending order $f(p_0) < \ldots < f(p_k)$. To approach this problem we must first define the set of solid handles that materialize topological transitions between level sets.

3.1. Solid handles and level set operations

Figure 7 shows the set of solid handles that represent topological changes on the level set sequence of a solid. Figure 8 shows the effect of these handles on the level sets component population. The shape of these handles is dictated by the local shape of the function f near its critical points.

Consider the external boundary surface ∂M_e and an internal boundary surface ∂M_i of the solid M. The solid 0-, 1- and 2-handles (Figures 7a, 7b and 7c respectively) are the volume versions of the local shape of the critical points on surface ∂M_e. The void 0-, 1- and 2-handles (Figures 7d, 7e and 7f respectively) are the volume versions of the local shape of the critical points on surface ∂M_i. The solid and void 1-handles shown in Figure 7, considered from bottom to top, perform the union operation. These handles can be considered *upside down* to perform the inverse operation (separation). When this is the case, we call them the *inverse* 1-handle (solid or void). This fact is important because the decomposition sequence must be able to tell apart union 1-handles from separation 1-handles.

With this set of handles we characterize the possible Morse transitions between consecutive level sets. Notice that these handles are related exclusively to the shape of the critical points on the boundaries of M, therefore allowing us to characterize the topology of M by studying the evolution of contours both on the external and internal boundary surfaces.

3.2. Algorithmic approach

In this section we present the pseudocode to identify and classify solid handles from a level set sequence. Consider a solid M (possibly with voids) with external boundary surface B_0 and possible

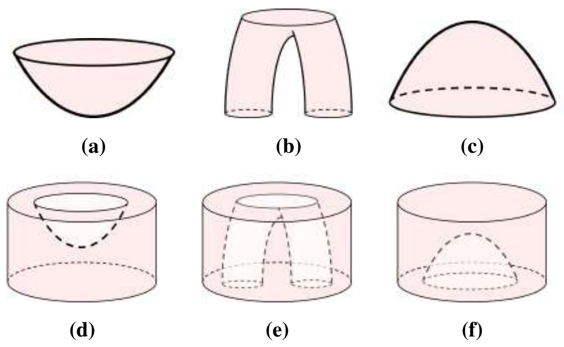


Figure 7: *Solid handle set. (a) Solid 0-handle. (b) Solid 1-handle. (c) Solid 2-handle. (d) Void 0-handle. (e) Void 1-handle. (f) Void 2-handle.*

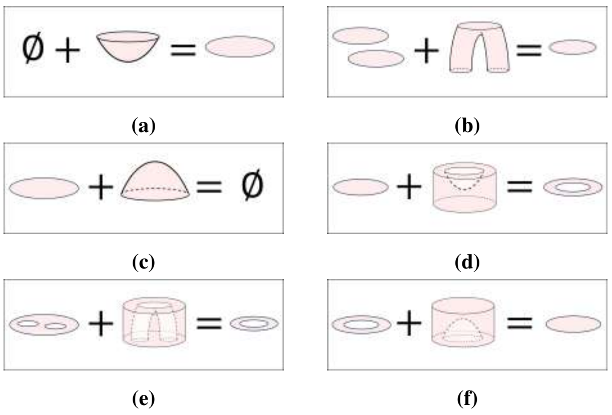


Figure 8: *Solid handle effect on the level sets component population.*

internal boundary surfaces $B_1,\ldots,B_k$. The methodology to materialize the solid handle decomposition of M goes as follows:

1. *Define a slicing*: Set a function f on $B_0,\ldots,B_k$ to obtain a sequence of level sets Π. Notice that each individual level set Π_i can have contours coming from one or more boundary surfaces $B_0,\ldots,B_k$.
2. *Define inclusion relationships*: build a forest F_i for level set Π_i such that first-level nodes are contours $C_i \in B_0$ coming from the external boundary B_0 and second-level nodes are contours $C_i \subset B_k$ coming from internal boundaries $B_1,\ldots,B_k$.
3. *Calculate contour mappings*: match every contour $C_i \in \Pi_i$ in a level set to a 2D similar contour $C_k \in \Pi_{i+1}$ in the next level set. If a contour in $C_i \in \Pi_i$ cannot be mapped to a contour in Π_{i+1}, or viceversa, a *contour to void* mapping is produced. If two contours $C_i, C_j \in \Pi_i$ are matched to the same contour in Π_{i+1} (*double* mapping), a topological event took place.
4. *Event classification*: the apparition of *contour to void* mappings or *double* mappings means a topological change took place (a contour disappeared, appeared *ex nihilo* or merged with a nearby contour). These cases are further processed to determine which solid handle can correctly model the change that took place.
5. *Decomposition of the solid*: once the topological events have been correctly identified with a solid handle, a decomposition of the solid is produced.

3.2.1. Contour orientation and inclusion

The slicing function f defines a level set sequence Π that contains contours (curves) from boundary surfaces $B_0,\ldots,B_k$. Additionally, the slicing should record the information about the origin of each contour (whether it comes from an internal or an external surface). This is straightforward (by storing the normal vector with respect to the surface) if the Boundary Representation is well defined. In the fashion of [MRC*20], the orientation of each contour is defined by the projection of the surface normal vector to the slicing plane. Consider the slicing plane intersects a triangle $t \in M$, then:

$$\vec{n}_{xy}(t) = Proj_{xy}(\vec{n}(t)) \tag{3}$$

A connected component of a level set $f^{-1}(c)$ can be: (i) a closed 1-manifold (closed curve), (ii) a 0-manifold (point) or (iii) a 2-manifold region (the triangle itself when it coincides with the slicing plane). In the degenerate cases (ii and iii) no orientation can be produced and these components can be safely ignored as they will appear in a non-degenerate case in following level sets. Every contour must have *coherent* normal behaviour if the mesh is well defined (i.e. all projected normal vectors must point outwards or inwards with respect to the polygon defined by the closed curve of the contour). The contour forest F_i for a level set Π_i should be constructed from an algorithm like Algorithm 1. This contour forest contains the information about the inclusion relationships between contours.

Algorithm 1 Build contour forest F_i for every level set Π_i

```
$\Pi_{ext} \leftarrow external(C \in \Pi_i)$
$\Pi_{int} \leftarrow internal(C \in \Pi_i)$
$F_i \leftarrow$ new Forest
for all contour $C_k \in \Pi_{ext}$ do
    $F_i \leftarrow$ addParent($C_k$)
end for
for all contour $C_k \in \Pi_{int}$ do
    for all contour $C_j \in \Pi_{ext}$ do
        included $\leftarrow$ testInclusion($C_k, C_j$)
        if included then
            $F_i \leftarrow$ addChild($C_k$)
            break
        end if
    end for
end for
```

Algorithm 1 creates the contour forest of a level set assuming external contours as parent nodes (since no external contour can contain other external contour) and internal contours as child nodes. The inclusion test between two polygons (defined by their contours) is a very well studied problem and numerous solutions exist [FTU95]. Every forest constructed this way will have depth no larger than 1. This is because no internal contour can contain other contour as we are not considering the case of annidated solid bodies. Figure 9 shows an example of the contour forest of a level set with several contours.

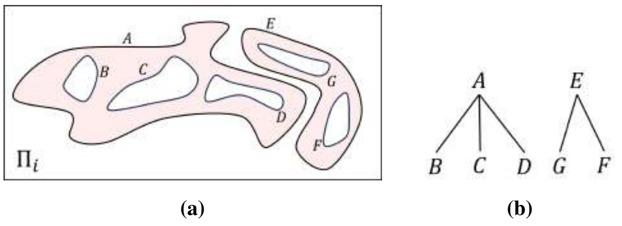


Figure 9: *Contour forest F_i for level set Π_i. (a) Example level set Π_i. (b) Resulting contour forest F_i.*

3.2.2. Calculation of mapping groups

The mapping groups track the existence of contours as the function f evolves. This amounts to, given a contour $C_k \in \Pi_i$, find a contour $C_j \in \Pi_{i+1}$ that most resembles a feasible geometrical and topological evolution of C_k. Algorithm 2 describes our approach. We use the information from the contour forests from the previous step to avoid unnecessary tests between contours and use a similarity criteria to match two contours. Figure 10 shows an example of a contour mapping. Notice that:

1. External contours in Π_i need only be tested against external contours in Π_{i+1}. No *external to internal* match can exist.
2. Let $C_1 \in \Pi_i$ be an external contour containing an internal contour $C_2 \in \Pi_i$. Let $C_a \in \Pi_{i+1}$ be the matching contour of C_1 in Π_{i+1}. Internal contour $C_2 \in \Pi_i$ can only produce matchings with contours that are contained by C_a. This reduces the number of matching operations needed.

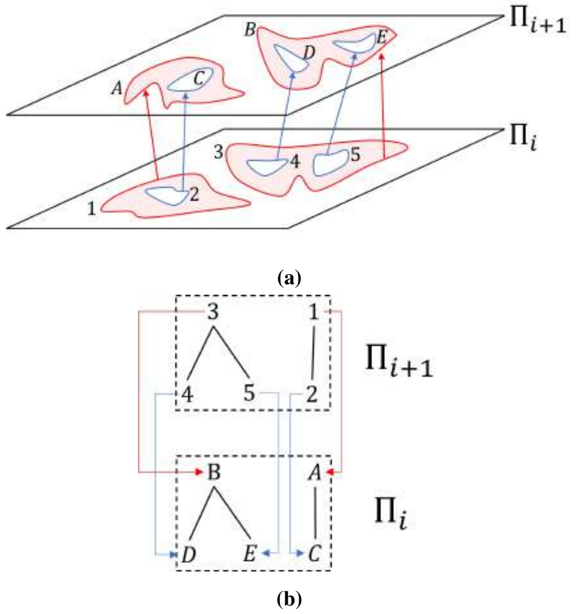


Figure 10: *Example of contour mapping. (a) Level set sequence (Π_i, Π_{i+1}). (b) Maps produced between the forests.*

Algorithm 2 builds the contour mappings between consecutive level sets. First, all external contours of level set Π_i are tested against external contours of level set Π_{i+1} and mappings are produced according to the similarity criteria (explained later). If a contour in Π_i can't find a match in Π_{i+1} a *contour to void* mapping is produced. Every time a contour in Π_{i+1} is added to a map it is marked as already mapped. After all combinations are tested, any contour in Π_{i+1} left unmarked is added to a *void to contour* mapping. For every internal contour in Π_i, we seek the parent external contour in Π_i and find its correspondent external contour in Π_{i+1}. Then the internal contour in Π_i is tested against the children of the correspondent external contour in Π_{i+1}.

The core of the mapping algorithm is the shape similarity test. The goal of this test is to calculate the degree of geometric resemblance between two contours. This is done by considering the orthogonal projection of the contours to a common plane and calculating a similarity index (Figure 11). Consider contours $C_j \in \Pi_i$ and

Algorithm 2 Calculation of mappings between Π_i and Π_{i+1}

```
$\Pi_{ext,i} \leftarrow external(C \in \Pi_i)$
$\Pi_{int,i} \leftarrow internal(C \in \Pi_i)$
$\Pi_{ext,i+1} \leftarrow external(C \in \Pi_{i+1})$
$\Pi_{int,i+1} \leftarrow internal(C \in \Pi_{i+1})$
for all contour $C_j \in \Pi_{ext,i}$ do                ▷ External matchings
    match ← False
    for all contour $C_k \in \Pi_{ext,i+1}$ do
        match ← testSimilarity($C_k, C_j$)
        if match then
            newMap($C_j, C_k$)
            markAsMatched($C_k$)
        end if
    end for
    if not match then
        newMap($C_j, \emptyset$)
    end if
end for
if any $C_k \in \Pi_{ext,i+1}$ not matched then
    newMap($\emptyset, C_k$)
end if
for all contour $C_j \in \Pi_{int,i}$ do                ▷ Internal matchings
    $C_{prnt}$ ← getParentContour($C_j$)
    $C_m$ ← getMatchInMap($C_{prnt}$)
    $\Pi_{m,i+1}$ ← getChildrenContours($C_m$)
    match ← False
    for all $C_k \in \Pi_{m,i+1}$ do
        match ← testSimilarity($C_k, C_j$)
        if match then
            newMap($C_j, C_k$)
            markAsMatched($C_k$)
        end if
    end for
    if not match then
        newMap($C_j, \emptyset$)
    end if
end for
if any $C_k \in \Pi_{ext,i+1}$ not matched then
    newMap($\emptyset, C_k$)
end if
```

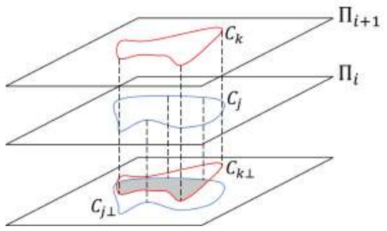


Figure 11: *Projection of contours to calculate the similarity index.*

$C_k \in \Pi_{i+1}$. Their respective orthogonal projections to a common parallel plane are denoted as $C_{j\perp}$ and $C_{k\perp}$. The similarity index is calculated as:

$$similarity(C_j, C_k) = \min\left(\frac{Area\left(C_{j\perp} \cap C_{k\perp}\right)}{Area(C_{j\perp})}, \frac{Area\left(C_{j\perp} \cap C_{k\perp}\right)}{Area(C_{k\perp})}\right) \tag{4}$$

Two contours in consecutive level sets are said to have a real relation if their projections intersect in a significant portion (more than a threshold value). In Equation 4 two quantities are calculated: the portion of C_j inside C_k and the portion of C_k inside C_j. The similarity index is defined as the smallest of those two quantities. Some authors define the similarity index as the largest of the two quantities [LSY*14] but this results in a less strict test and false positives can be wrongly matched. The intersection approach has proven to be stable and robust [RCG*05]. The canonical approach to establish connectivity between contours of subsequent level sets is the distance criteria [SK91]. This approach establishes a weighted distance function between contours in different level sets and searches for the maximum value of the function for each contour. The similarity approach advantages the distance criteria because the it is independent of the distance between the level sets $d(\Pi_i, \Pi_{i+1})$. This independence is important to preserve the robustness of the matching when changing the slicing density.

In this case, the selected function is the height function $f(x,y,z) = z$. This choice results in perfectly planar and parallel level sets. Other choices for the Morse functions might produce non-planar level sets. Since our methodology relies on the orthogonal projection of the contours to a parallel plane, the proposed algorithm will fail to correctly grasp the geometry of non-planar level sets.

3.2.3. Solid handle decomposition

The calculated mapping groups are used to classify topological transitions between contours (Figure 8). When there is no topological change between level sets Π_i and Π_{i+1} all the contours in both level sets are present at most in one mapping group and there are no void mappings. When a topological change took place between Π_i and Π_{i+1} one (and only one) of the following situations is true:

1. A *contour to void* map is present.
2. A *void to contour* map is present
3. A contour in Π_i or Π_{i+1} is present in more than one map.

The presence of a *contour to void* map means that a contour disappeared in the transition between Π_i and Π_{i+1}. If the contour affected is a solid contour (i.e. coming from the external boundary of the solid) then this transition can be modeled by the solid 2-handle (Figure 7c). If it is a void contour (i.e. coming from an internal boundary of the solid) then this transition can be modeled by the void 2-handle (Figure 7f). This same reasoning applies to the case of a *void to contour* map that denotes the creation *ex nihilo* of a contour. This transition can be modeled either by the solid 0-handle (Figure 7a) or the void 0-handle (Figure 7d). If a contour is present in more than one map between the same pair of level sets it means a 1-handle transition took place and it can be modeled by the solid (Figure 7b) or void (Figure 7e) 1-handle or their inverse handles.

In our approach we execute a bottom to top sweep of the level set sequence to look for mappings that reveal topological changes and we record the identity of the solid handles. The resulting sequence of handles is a valid decomposition of the solid. For each identified pair of level sets (Π_i, Π_{i+1}) in which a topological transition took place, a handle distance d is chosen and the surface containing the transition contours is cut (parallel to the level sets) at a distance d and $-d$ from the level sets (Π_i, Π_{i+1}). After this, a simple flooding algorithm retrieves the decomposition both on the internal and external boundary surfaces. We make sure this chosen distance d is small enough so that it does not contains another pair of critical level sets.

4. Time complexity

To have a meaningful measurement of the complexity of the proposed algorithm let us look into the theoretical complexity of each one of its components. Our approach can be divided in four different processes: (a) slicing, (b) forest build, (c) contour matching, (d) surface retrieval. We will only consider the time complexity of the *forest build* algorithm and the *contour matching* algorithm as the other two processes are standard and well studied and their theoretical time complexity has been widely reported.

Consider Algorithm 1 with K being the number of level sets in the model, N being the maximum number of internal contours in a single level set and M the maximum number of external contours in a single level set. The worst-case complexity of Algorithm 1 is $O(K*N*M)$. In Algorithm 2, we have a worst-case complexity of $O(K(M^2+N^2))$. Therefore, the overall worst-case complexity of the algorithm in terms of the number of contours and level sets is:

$$O(K(NM+M^2+N^2)) \tag{5}$$

We believe it is more meaningful to consider complexity with respect to the number of contours and the density of the slicing than with respect to the size of the mesh given that the slicing reduces the mesh (whatever its size) to a number of contours that represent the topological changes of the solid and therefore its handle decomposition. Additionally, this consideration allows us to assess the algorithm's performance for applications in which the contour population does not come from a mesh (e.g. CT scans).

5. Experiments

In this section we present the experiments done to validate our approach. We tested our decomposition algorithm in Boundary Representations of several solids both with and without solids. Figure 12 shows the application of our methodology to solids without voids. We show that our methodology produces valid decompositions of these simple solids without voids similar to those produced by surface-based methodologies. Additionally, we present the equivalent solids of these datasets. The equivalent solids are the aggregation of all identified handles. By tracking the identified solid handles it can be seen that these solids are topologically equivalent to the dataset solids.

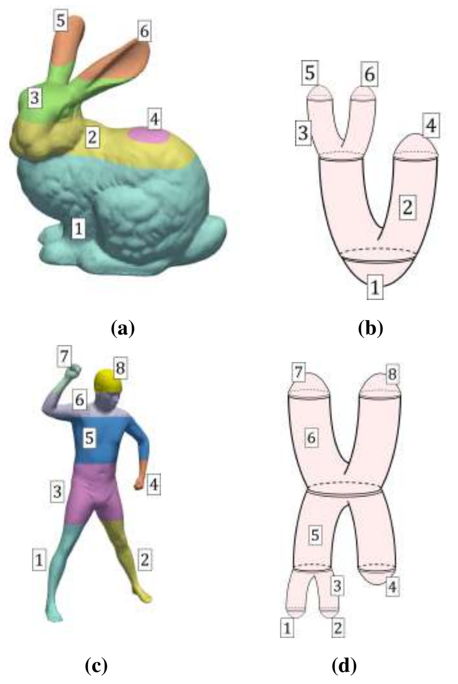


Figure 12: *Solid handle decomposition of solids without voids. (a) Solid bunny dataset. (b) Solid bunny's equivalent solid. (c) Solid man dataset. (d) Solid man's equivalent solid.*

Figures 13 and 14 show examples of the solid handle decomposition for solids *with* voids based on the CADNET dataset [MBM21]. These solids are pieces of interest in manufacturing applications and their handle decomposition is an important information necessary to optimize their additive manufacturing planning scheme. We show the cross-sectional views of the solid to highlight the fact that they have internal voids that need to be encoded in the decomposition. See, for example, handles 6 and 7 in Figure 13 and handles 4 and 5 in Figure 14: these handles encode the presence of an internal void in the solid. Figure 15 shows the application of our methodology to the *vertebrae* dataset. Figure 15c shows the resulting decomposition of the vertebrae with a detail on the inner void decomposition and their effect on the partition of the outer boundary.

An important disclaimer is that handle decompositions are sensitive to the choice of the Morse function used to obtain the level sets.

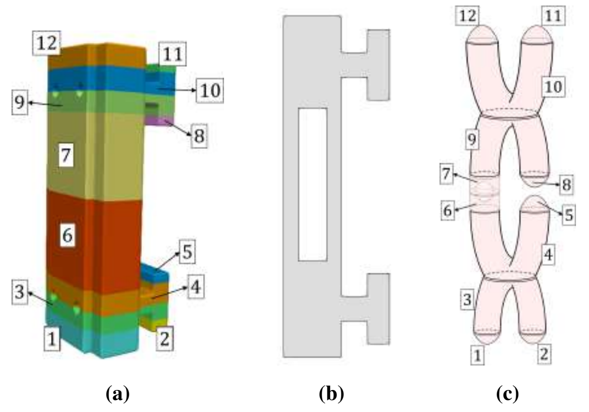


Figure 13: *Solid handle decomposition of solids with voids. (a) Void bracket dataset. (b) Cross-sectional view of the bracket. (c) Bracket's equivalent solid.*

Changes in the orientation of the slicing produce different decompositions of the solid. The analysis of the validity and performance of our method with different Morse functions (e.g. distance from center of mass) is still to be done. Despite this limitations, our main advantage with respect to other very fast and robust approaches to other decomposition methods (such as the one in [PSBM07]) is the adequate consideration of inner voids in the decomposition of the pieces.

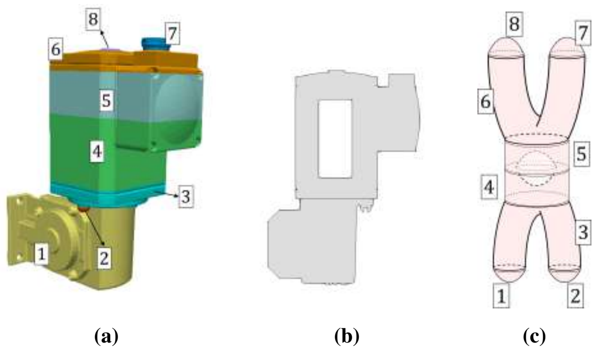


Figure 14: *Solid handle decomposition of solids with voids. (a) Void engine block dataset. (b) Cross-sectional view of the engine block. (c) Engine block's equivalent solid.*

6. Conclusion and Future Work

The decomposition of solids is a problem of interest in manufacturing and shape analysis. The problem can be stated as finding a set of smaller and simpler pieces that glued together amount to the initial solid. This decomposition can be guided by geometrical or topological criteria and be applied to either surfaces or solids (embedded manifolds). In this manuscript we reviewed the topological decomposition approaches which rely on Morse theory, in particular the Reeb graph-based ones. We show that Reeb graph-based decompositions have proven to work well for surfaces and for solids without inner voids, but fail to consider solids with inner voids. In this work

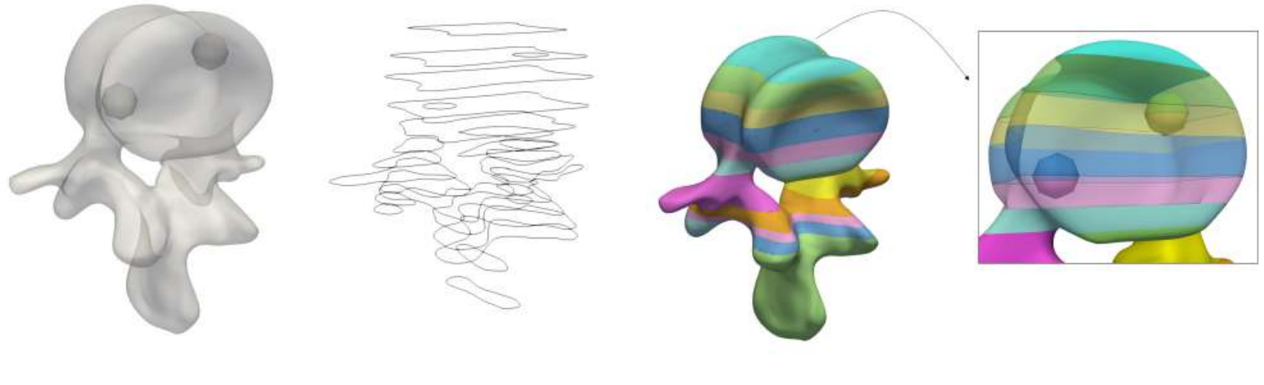


Figure 15: *Solid handle decomposition of solids with voids. (a) Vertebrae dataset. (b) Level set representation of vertebrae. (c) Decomposition of the vertebrae.*

we present a methodology based on the handle-decomposition of a manifold that can encode changes in the topology of solids both with and without inner voids. Our methodology uses the Boundary Representation of the solid and a shape similarity criteria to identify changes in the topology of both the outer and inner boundary(ies) of the solid. We present an augmented set of solid handles that allows the inclusion of internal voids in the encoding of the solid's topology. We describe an algorithm to implement our methodology and execute experiments on several datasets, including solids both with and without voids. The results show that our methodology can produce valid handle decompositions for solids both with and without inner voids. Our approach, however, as all approaches guided by Morse theory, are sensitive to the choice of the function f that defines the critical points in the manifold. Future research should include: (i) the evaluation of our method's viability and performance with Morse functions other than the height function, (ii) analysis of performance with bigger and more complex data, (iii) robustness to noise analysis.

References

[ACA*19] Araújo, Chrystiano, Cabiddu, Daniela, Attene, Marco, et al. "Surface2Volume: Surface segmentation conforming assemblable volumetric partition". *arXiv preprint arXiv:1904.10213* (2019) 4.

[APP*07] Agathos, Alexander, Pratikakis, Ioannis, Perantonis, Stavros, et al. "3D mesh segmentation methodologies for CAD applications". *Computer-Aided Design and Applications* 4.6 (2007), 827–841 1.

[BGSF08] Biasotti, S., Giorgi, D., Spagnuolo, M., and Falcidieno, B. "Reeb graphs for shape analysis and applications". *Theoretical Computer Science* 392.1 (2008). Computational Algebraic Geometry and Applications, 5–22. ISSN: 0304-3975 2, 4.

[Bia04] Biasotti, Silvia. "Reeb graph representation of surfaces with boundary". *Proceedings Shape Modeling Applications, 2004*. IEEE. 2004, 371–374 3.

[Boh19] Bohm, N. "Morse Theory and Handle Decompositions". *University of Chicago Mathematics REU* (2019) 2.

[BP12] Brandolini, Laura and Piastra, Marco. "Computing the Reeb Graph for Triangle Meshes with Active Contours." *ICPRAM (2)* 12 (2012), 80–89 3.

[CEH*03] Cole-McLaughlin, Kree, Edelsbrunner, Herbert, Harer, John, et al. "Loops in Reeb graphs of 2-manifolds". *Proceedings of the Nineteenth Annual Symposium on Computational Geometry*. New York, NY, USA: Association for Computing Machinery, 2003, 344–350 3.

[DN11] Doraiswamy, Harish and Natarajan, Vijay. "Output-sensitive construction of Reeb graphs". *IEEE Transactions on Visualization and Computer Graphics* 18.1 (2011), 146–159 4.

[FTU95] Feito, F, Torres, Juan Carlos, and Urena, A. "Orientation, simplicity, and inclusion test for planar polygons". *Computers & Graphics* 19.4 (1995), 595–600 5.

[GXZ*23] Guo, Hongshuai, Xu, Jinghua, Zhang, Shuyou, et al. "Multi-orientation optimization of complex parts based on model segmentation in additive manufacturing". *Journal of Mechanical Science and Technology* 37.1 (2023), 317–331 1.

[HC*12] Ho, Tan-Chi, Chuang, Jung-Hong, et al. "Volume based mesh segmentation". *Journal of Information Science and Engineering* 28.4 (2012) 1.

[HDL16] Hajij, Mustafa, Dey, Tamal, and Li, Xin. "Segmenting a surface mesh into pants using Morse theory". *Graphical Models* 88 (2016), 12–21 2, 3.

[HR20] Hajij, Mustafa and Rosen, Paul. "An efficient data retrieval parallel Reeb graph algorithm". *Algorithms* 13.10 (2020), 258 3.

[HW18] He, Chen and Wang, Chunmeng. "A survey on segmentation of 3D models". *Wireless Personal Communications* 102 (2018), 3835–3842 1.

[HWW10] Harvey, William, Wang, Yusu, and Wenger, Rephael. "A randomized O (m log m) time algorithm for computing Reeb graphs of arbitrary simplicial complexes". *Proceedings of the twenty-sixth annual symposium on Computational geometry*. 2010, 267–276 3.

[KS00] Kanongchaiyos, PIZZANU and Shinagawa, YOSHIHISA. "Articulated Reeb graphs for interactive skeleton animation". *Multimedia Modeling: Modeling Multimedia Information and Systems*. Nagano, Japan: World Scientific, 2000, 451–467 1.

[KZW12] Kwok, Tsz-Ho, Zhang, Yunbo, and Wang, Charlie CL. "Constructing common base domain by cues from Voronoi diagram". *Graphical Models* 74.4 (2012), 152–163 1.

[LGQ09] LI, XIN, GU, XIANFENG, and QIN, HONG. "Surface mapping using consistent pants decomposition". *IEEE Transactions on Visualization and Computer Graphics* 15.4 (2009), 558–571 1.

[LSY*14] LIM, CHI WAN, SU, YI, YEO, SI YONG, et al. "Automatic 4D reconstruction of patient-specific cardiac mesh with 1-to-1 vertex correspondence from segmented contours lines". *PloS one* 9.4 (2014), e93747 7.

[MBM21] MANDA, BHARADWAJ, BHASKARE, PRANJAL, and MUTHUGANAPATHY, RAMANATHAN. "A Convolutional Neural Network Approach to the Classification of Engineering Models". *IEEE Access* 9 (2021), 22711–22723. DOI: 10.1109/ACCESS.2021.3055826 8.

[MG09] MAMOU, KHALED and GHORBEL, FAOUZI. "A simple and efficient approach for 3D mesh approximate convex decomposition". *2009 16th IEEE international conference on image processing (ICIP)*. IEEE. 2009, 3501–3504 1.

[MG21] MA, CHAO and GAO, YUNKAI. "Study on mesh segmentation of topology optimization results using Reeb graph". *2021 International Conference on Artificial Intelligence and Electromechanical Automation (AIEA)*. IEEE. 2021, 277–280 2.

[MRC*20] MEJIA-PARRA, DANIEL, RUIZ-SALGUERO, OSCAR, CADAVID, CARLOS, et al. "Level Sets of Weak-Morse Functions for Triangular Mesh Slicing". *Mathematics* 8.9 (2020) 5.

[Par12] PARSA, SALMAN. "A deterministic o (m log m) time algorithm for the reeb graph". *Proceedings of the twenty-eighth annual symposium on Computational geometry*. 2012, 269–276 3.

[PSBM07] PASCUCCI, VALERIO, SCORZELLI, GIORGIO, BREMER, PEER-TIMO, and MASCARENHAS, AJITH. "Robust On-Line Computation of Reeb Graphs: Simplicity and Speed". *ACM Transactions on Graphics* 26.3 (2007), 58. ISSN: 0730-0301 8.

[RCG*05] RUIZ, OSCAR E, CADAVID, CARLOS A, GRANADOS, MIGUEL, et al. "2D shape similarity as a complement for Voronoi–Delone methods in shape reconstruction". *Computers & Graphics* 29.1 (2005), 81–94 7.

[SJ15] STRODTHOFF, BIRGIT and JÜTTLER, BERT. "Layered Reeb graphs for three-dimensional manifolds in boundary representation". *Computers & Graphics* 46 (2015), 186–197 4.

[SJ17] STRODTHOFF, BIRGIT and JÜTTLER, BERT. "Automatic decomposition of 3D solids into contractible pieces using Reeb graphs". *Computer-Aided Design* 90 (2017), 157–167 2, 4.

[SK91] SHINAGAWA, YOSHIHISA and KUNII, TOSIYASU L. "Constructing a Reeb graph automatically from cross sections". *IEEE Computer Graphics and Applications* 11.06 (1991), 44–51 3, 7.

[SSGH01] SANDER, PEDRO V, SNYDER, JOHN, GORTLER, STEVEN J, and HOPPE, HUGUES. "Texture mapping progressive meshes". *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. 2001, 409–416 1.

[TPT14] THEOLOGOU, PANAGIOTIS, PRATIKAKIS, IOANNIS, and THEOHARIS, THEOHARIS. "A review on 3D object retrieval methodologies using a part-based representation". *Computer-Aided Design and Applications* 11.6 (2014), 670–684 3.

[TSK98] TAI, CHIEW-LAN, SHINAGAWA, YOSHIHISA, and KUNII, TOSIYASU L. "A Reeb graph-based representation for non-sequential construction of topologically complex shapes". *Computers & Graphics* 22.2-3 (1998), 255–268 4.

[TTF04] TAKAHASHI, SHIGEO, TAKESHIMA, YURIKO, and FUJISHIRO, ISSEI. "Topological volume skeletonization and its application to transfer function design". *Graphical Models* 66.1 (2004), 24–49 1.

[WY11] WANG, JUN and YU, ZEYUN. "Surface feature based mesh segmentation". *Computers & Graphics* 35.3 (2011), 661–667 1.