Topology-controlled Reconstruction of Multi-labelled Domains from Cross-sections

ZHIYANG HUANG, Washington University in St. Louis
MING ZOU, Washington University in St. Louis
NATHAN CARR, Adobe Systems
TAO JU, Washington University in St. Louis

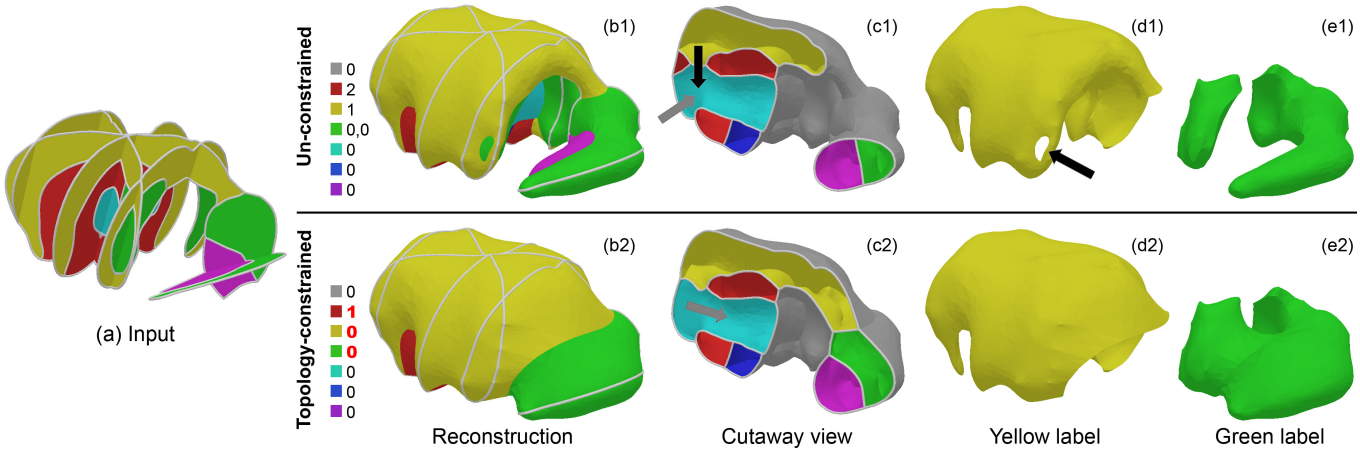


Fig. 1. Given several multi-labeled planes depicting the anatomical regions of a mouse brain (a), reconstruction without topology control (b1) leads to redundant handles for the red and yellow labels (black arrows in c1, d1) and disconnection for the green label (e1). Our method (b2) allows the user to prescribe the topology such that the red label has one tunnel (gray arrow in c2), the yellow label has no tunnels (d2), and the green label is connected (e2). The legends in (b1,b2) report, for each label in the reconstruction, the genus of each surface component bounding that label (e.g., "0,0" means two surfaces each with genus 0). User-specified constraints are colored red.

In this work we present the first algorithm for reconstructing multi-labeled material interfaces the allows for explicit topology control. Our algorithm takes in a set of 2D cross-sectional slices (not necessarily parallel), each partitioned by a curve network into labeled regions representing different material types. For each label, the user has the option to constrain the number of connected components and genus. Our algorithm is able to not only produce a material interface that interpolates the curve networks but also simultaneously satisfy the topological requirements. Our key innovation is defining a space of topology-varying material interfaces, which extends the family of level sets in a scalar function, and developing discrete methods for sampling distinct topologies in this space. Besides specifying topological constraints, the user can steer the algorithm interactively, such as by scribbling. We demonstrate, on synthetic and biological shapes, how our algorithm opens up new opportunities for topology-aware modeling in the multi-labeled context.

CCS Concepts: • **Computing methodologies** → **Mesh models**;

This work is supported by the National Science Foundation, under grants IIS-084607 and IIS-1302200, and a gift from Adobe Inc.

DOI: http://dx.doi.org/10.1145/3072959.3073644

Additional Key Words and Phrases: surface reconstruction, material interfaces, topology, contour interpolation

ACM Reference format:
Zhiyang Huang, Ming Zou, Nathan Carr, and Tao Ju. 2017. Topology-controlled Reconstruction of Multi-labelled Domains from Cross-sections. *ACM Trans. Graph.* 36, 4, Article 76 (July 2017), 12 pages.
DOI: http://dx.doi.org/10.1145/3072959.3073644

1 INTRODUCTION

Computational modeling of multi-labeled domains arises in many disciplines, such as biomedicine (e.g., organs made up of multiple anatomical regions) and mechanical engineering (e.g., machine pieces made up of blocks of different materials). Such domains are often represented by the non-manifold network of surfaces that partition the domain into labeled sub-domains. This network, known as the *material interface*, is widely used in applications including geometric processing, physical simulations, and manufacturing.

To be useful for applications, a material interface has to meet certain correctness criteria. Most importantly, the material interface should be *geometrically valid* (i.e., free of holes and intersections), so that it defines a proper partitioning of the domain into disjoint sub-domains. Furthermore, some applications are also sensitive to the *topology* of the surface. For example, fluid simulation within one or more sub-domains can be adversely affected if the surfaces

bounding these sub-domains fail to have an expected number of connected components or genus. Extraneous components or genus can also be detrimental for many geometric processing tasks, such as mesh simplification and surface parameterization.

Topology control has been extensively studied in the context of modeling two-labeled domains, where the material interfaces are closed manifold surfaces. However, to date, no such control has been seen in modeling domains containing three or more labels in the absence of a template (see review in Section 2). Ensuring correct topology in a multi-labeled context is arguably more challenging, because the topology of different labels are intertwined: modifying the topology of one label may affect the topology of several other labels. The intertwining makes it difficult even for humans to manually fix topological errors on a complex material interface without introducing geometric errors (e.g., the mouse brain in Figure 1 (b1-e1)).

In this paper, we present a novel algorithm for enforcing topological constraints when reconstructing multi-labeled material interfaces. Our algorithm is designed for inputs consisting of cross-sections of the subject. Such inputs often arise in biomedicine, where experts delineate boundaries of anatomical regions on 2D slices of a 3D medical image (e.g., MRI, CT). Cross-sectional inputs can also be found in other disciplines, such as material science (e.g., sectioned micrographs) and geology (e.g., seismic images). To model a multi-labeled domain, each cross-section contains a curve network that partitions the plane into labeled regions. Our method gives the user the option to specify the desired topology, in terms of number of connected components and genus, for any subset of the labels. The output is a geometrically valid material interface that interpolates the curve networks while meeting the topological requirements (Figure 1 (b2-e2)).

Our key contribution is a novel definition, called *interface sets*, that gives rise to not one, but a *space* of topology-varying material interfaces. Our definition mimics the level sets, which is a family of closed manifold surfaces defined by a scalar function and parameterized by a scalar value. Similarly, the interface sets are non-manifold surface networks defined by a vector function and parameterized by a vector value. We analyze the topological events in the multi-variate space of interface sets, which are much more complex than those in the univariate family of level sets, and propose a simple and effective method for sampling distinct topologies of interface sets.

Using the interface sets, we extend the recently introduced topology-controlled algorithm of Zou et al. [2015] from two-labeled domains to multiple labels. Our algorithm proceeds in two stages. First, within each cell bounded by the cross-section planes, we define a suitable vector function and enumerate interface sets with different topologies. Each topology is also given a score that measures its likelihood. Next, we perform combinatorial optimization to select one topology per cell so that the overall reconstruction satisfies the user-given topological constraints while the total score is maximized.

In addition to specifying components and genus, the user can steer the method in interactive ways. The user may browse and select from the list of topologies computed by our method for each cell. If a desired topology does not exist in our computed list, we offer a sketching interface whereby the user can easily create new topologies. These user inputs guide the algorithm towards a more satisfactory reconstruction. We demonstrated our algorithm and tool on both simple synthetic inputs and non-trivial biological data sets (e.g., Figures 1, 8, 9).

1.1 Contributions

To the best of knowledge, our method is the first for material interface reconstruction that offers topology control without the use of any templates. Our main contributions are:

(1) Defining a multi-variate space of material interfaces, analyzing its topological structure in the discrete setting, and developing a topology sampling method (Section 3).
(2) Extending the topology-controlled reconstruction algorithm of Zou et al. [2015] from two to multiple labels (Section 4).
(3) Developing interactive tools for refining the surface topology (Section 5).

While our method is designed for cross-sectional inputs, we believe some of our contributions (particularly the method of interface sets) can benefit topology-aware modeling from other input types, such as point clouds or labeled medical images.

2 RELATED WORKS

We briefly review the three bodies of work that are closed to ours, namely modeling multi-labeled domains, topology control in modeling two-labeled domains, and reconstruction from cross-sections.

2.1 Modeling multi-labeled domains

Typical representations of multi-labeled domains include (regional or global) implicit functions [Feng et al. 2010; Kim 2010; Losasso et al. 2006; Mitchell et al. 2015; Saye 2015; Yuan et al. 2012; Zhao et al. 1996; Zheng et al. 2006] and volume fractions [Ahn and Shashkov 2007; Anderson et al. 2008, 2010; Bonnell et al. 2003]. While implicit function representations are often based on level sets, current works typically utilize a single level as the underlying function evolves (e.g., during a simulation), which creates a univariate family of domains. We are not aware of any work that explores a multi-variate space of material interfaces.

Many reconstruction methods are capable of creating geometrically valid material interfaces. The majority of these methods are based on iso-contouring [Anderson et al. 2008; Bertram et al. 2005; Dillard et al. 2007; Feng et al. 2010; Haitham Shammaa et al. 2010; Ju et al. 2002; Qian and Zhang 2011; Yuan et al. 2012; Zhang et al. 2007], a few perform mesh surgeries [Brakke 1992; Da et al. 2014], and others further address the quality of elements (e.g., triangles and tetrahedra) using Delaunay meshing [Boltcheva et al. 2009; Bronson et al. 2014; Dey et al. 2012; Faraj et al. 2016; Pons et al. 2007] or particle diffusion [Meyer et al. 2008]. However, none of these methods offers explicit control over the topology. While topological errors can be avoided by fitting or evolving a template shape with the correct topology [Waggoner et al. 2015], these methods are limited to the availability of templates.

2.2 Topology-aware modeling of two-labeled domains

Numerous methods have been developed to fix topological errors on a closed manifold surface (see survey [Attene et al. 2013]). The

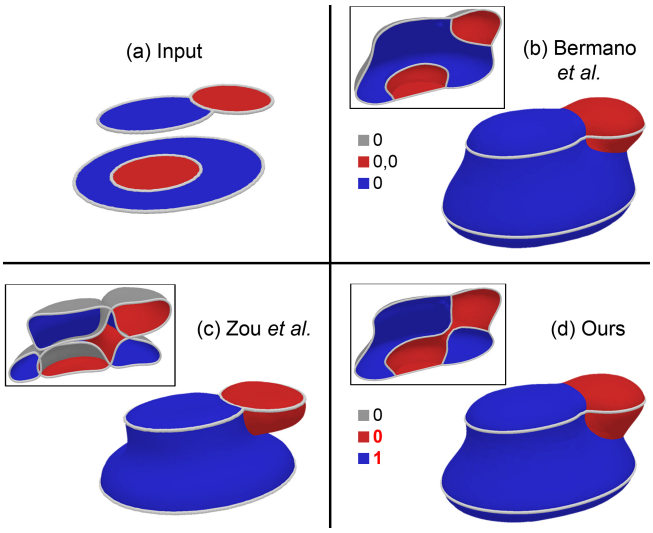


Fig. 2. Comparing the topology-oblivious multi-labeled method of Bermano et al. [2011] (b), the topology-constrained two-labeled method of Zou et al. [2015] (c), and our topology-constrained multi-labeled method (d) on two cross-sections with three labels (a). Cutaway views are shown in inserts. Legends report the per-component genus for each label (red numbers are constrained).

vast majority of these methods are concerned with the removal of redundant topological handles, while some also address connected components [Ju et al. 2007; Nooruddin and Turk 2003]. Another class of methods directly reconstruct a topologically correct model from raw inputs, such as a point cloud [Sharf et al. 2006, 2007; Yin et al. 2014], a collection of cross-section curves [Zou et al. 2015], or a grayscale volume [Bazin and Pham 2007; Zeng et al. 2008]. These methods are guided by prescribed genus [Sharf et al. 2006; Zou et al. 2015], interactive inputs [Sharf et al. 2007; Yin et al. 2014], or an existing template [Bazin and Pham 2007; Zeng et al. 2008].

2.3 Reconstruction from cross-sections

Surface reconstruction from cross-sectional curves has been extensively studied in computer graphics in the past few decades. We refer readers to recent works [Bermano et al. 2011; Zou et al. 2015] for more expansive reviews. Our method is closest to the class of methods that extracts the surface as the level set of an implicit function [Bermano et al. 2011; Herman et al. 1992; Liu et al. 2008; Turk and O'Brien 1999; Zou et al. 2015]. Most notably, Bermano et al. [2011] uses multiple implicit functions, one for each label, to extract multi-labeled material interfaces. However, except [Zou et al. 2015], all of these methods consider a single level in the implicit function(s) and leave no room for topology control. For the simple input shown in Figure 2 (a), applying the method of Bermano et al. [2011] results in two components of the red label (Figure 2 (b), see cutaway view in the insert). This would be undesirable if the user wants a single connected red structure that tunnels through the blue structure.

The only method that offers topology-controlled reconstruction was introduced recently by Zou et al. [2015], upon which our method extends. Within each cell bounded by the planes, this method explores multiple, topologically different level sets of an implicit function. A naïve way to extend this method to handle multiple labels is by reconstructing each label independently with the desired topology and combining the reconstructed surfaces. This is exemplified in Figure 2 (c), where red and blue labels are reconstructed by Zou's method respectively with genus 0 and 1. However, as seen in the cutaway view, the combination of two reconstructions results in jarring conflicts and intersections. In contrast, our extension of Zou's work creates a geometrically valid material interface with the desired genus for both labels (Figure 2 (d)).

3 INTERFACE SETS

Level sets have played fundamental roles in existing methods [Sharf et al. 2006, 2007; Zou et al. 2015] to provide topology control in modeling two-labeled domains. These methods take advantage of several unique features of level sets. First, given a scalar function, any level set is guaranteed to be geometrically valid (i.e., a closed manifold). Second, the collection of all level sets is parameterized along a single "level" axis and can be easily explored. Third, the level sets have a rich topological variety, and extensive studies are available on the topological evolution of the level set with the level [Edelsbrunner and Harer 2009; Milnor 1963].

To enable topology control in the context of multi-labeled modeling, we introduce a space of material interfaces that possesses similar features as level sets. Given a vector function, we define a space of *interface sets* such that each interface set is a geometrically valid material interface. The space is parameterized by a vector value (as opposed to a scalar level in level sets) and can be systematically explored. Lastly, this space reduces to the family of level sets in the special case of two labels, and it contains a even richer variety of non-manifold topologies in the case of three or more labels.

We start by defining interface sets and discussing their properties in the continuous setting (Section 3.1). Building upon classical works on level set topology, we then characterize the topological variations of interface sets in a discrete setting (Section 3.2). Finally, we propose a simple and effective scheme for sampling the large variety of interface set topologies (Section 3.3). In the next section, the sampling scheme will be utilized in our reconstruction algorithm to produce candidate local topologies from cross-sectional inputs.

3.1 Definition and properties

Our definition builds on an existing implicit definition of material interfaces, which has been used by various researchers [Feng et al. 2010; Losasso et al. 2006; Yuan et al. 2012]. In this definition, a n-labeled domain is represented by a vector-valued function $\vec{f}(\vec{x}) = \{f_1(\vec{x}), f_2(\vec{x}), \ldots, f_n(\vec{x})\}$, where $\vec{x}$ is a point in d-dimensional space and each f_i is a continuous scalar function. Intuitively, $f_i(\vec{x})$ describes the "prominence" of the i-th label at $\vec{x}$. Each point is then assigned the *most prominent* label(s), that is,

$$Labels(\vec{x}) = \arg \max_{i=1,\ldots,n} f_i(\vec{x}). \quad (1)$$

The material interface consists of all points whose label assignment is not unique (i.e., two or more labels share the greatest prominence).

This material interface is guaranteed to be geometrically valid, as it divides the space into regions with unique labels.

To be able to define not just one, but a parameterized set of material interfaces, we introduce an *offset* vector $\vec{c} = \{c_1, \ldots, c_n\}$ to the definition described above. This offset vector plays the role of the "level" in defining the level sets in a scalar function. More precisely, $\vec{c}$ is *added* to the vector function $\vec{f}$ before evaluating the labels. That is, the labeling is now *parameterized* by $\vec{c}$ as

$$Labels(\vec{c}, \vec{x}) = \arg \max_{i=1,\ldots,n} (f_i(\vec{x}) + c_i). \quad (2)$$

The *interface set at offset* $\vec{c}$ consists of all points $\vec{x}$ whose label assignment is not unique, that is, $|Labels(\vec{c}, \vec{x})| \neq 1$.

We can visualize interface sets in $d = 2$ dimensions intuitively as superimposed terrain maps (Figure 3). Imagine that each f_i is the height map of a 3D terrain over the 2-dimensional domain, and that each terrain has a unique color (Figure 3 (a)). Given an offset vector $\vec{c}$, we shift each i-th terrain vertically by the amount c_i and superimpose the shifted terrains (Figure 3 (b,c,d) middle). The labeling function $Labels(\vec{c}, \vec{x})$ is precisely the picture of the superimposed terrains taken from above, and the interface set is where two or more terrains meet in this picture (Figure 3 (b,c,d) bottom).

It is easy to see that every interface set is a geometrically valid material interface, since it divides the domain into regions carrying unique labels (i.e., $Labels(\vec{c}, \vec{x})$). In particular, the interface set at the zero offset $\vec{c} = 0$ is the material interface defined in the first paragraph and used in previous works.

We can also show that interface sets reduce to level sets in the case of $n = 2$ labels. In this case, any level set of a scalar function can be reproduced by some interface set of a vector function, and vice versa. Specifically, the level set of a scalar function f at any level c is identical to the interface set of the vector-valued function $\vec{f}(\vec{x}) = \{f(\vec{x}), 0\}$ at offset $\vec{c} = \{0, c\}$. Conversely, the interface set of a vector-valued function $\vec{f}(\vec{x}) = \{f_1(\vec{x}), f_2(\vec{x})\}$ at any offset $\vec{c} = \{c_1, c_2\}$ is the same as the level set of the scalar function $f(\vec{x}) = f_1(\vec{x}) - f_2(\vec{x})$ at level $c = c_2 - c_1$.

Note that an interface set is different from the intersection of n level sets, each defined by a scalar function f_i and a level c_i, as studied in multivariate topological data analysis [Carr and Duke 2014; Edelsbrunner and Harer 2002]. The interface set is generally a $(d-1)$-dimensional complex, regardless of the number of labels n, since it partitions the d-dimensional space into labelled regions. In contrast, the intersection of n level sets has a dimensionality of $d - n$, which is lower than that of the interface set and decreases as the number of labels increases. In the case of $n = 2$ labels in $d = 3$ dimensions, the intersection of level sets consists of one-dimensional curves, which are known as *fibers* and have found uses in visualization of bivariate data [Carr et al. 2015; Tierny and Carr 2017].

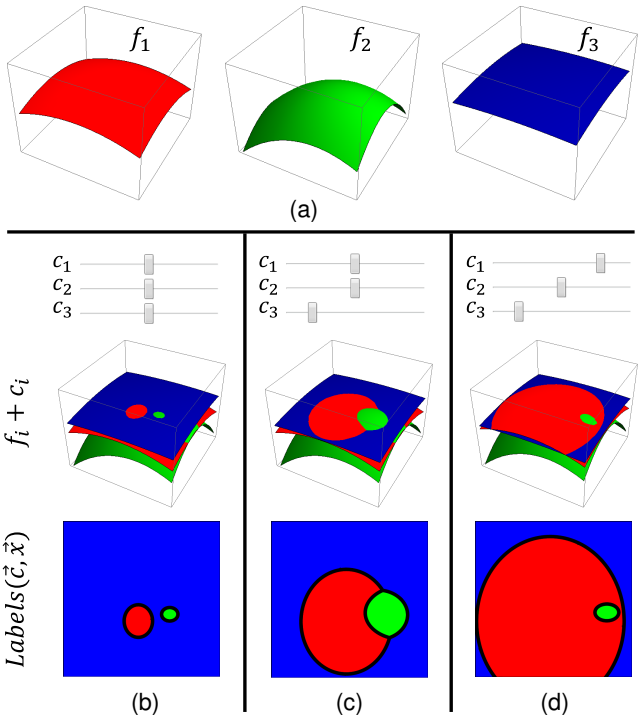


Fig. 3. Interface sets in 2D: (a) The input vector function $\vec{f} = \{f_1, f_2, f_3\}$, visualized as three height maps. (b,c,d): Three different choices of offsets $\vec{c} = \{c_1, c_2, c_3\}$ (top), superimposed height maps $f_i + c_i$ (middle), and the labeling (as color) and interface sets (as black curves) in the 2D domain (bottom).

3.2 Discrete topological variations

Topological evolution of the level set, as the level changes, is well-understood [Edelsbrunner and Harer 2009; Milnor 1963]. Topological changes are marked by local *topological events*, such as merging, splitting, and destruction or creation of components. These events take place at well-defined locations, known as *critical points*, which are identified by vanishing gradient of the scalar function. The function values at these locations, known as *critical values*, divide the range of levels into one-dimensional intervals, such that the level sets within one interval all share a common topology.

In contrast, topological evolution of the interface set, as the offset vector changes, is far more complex. First, there is a greater variety of topological events that involve non-manifold features of the surface (e.g., junction curves and points where more than three sheets meet). Second, the multi-variate nature of our "level" parameter - the offset vector - creates a complex topological landscape in the space of interface sets. Let's consider the n-dimensional space of all offset vectors, which we call the *offset space*. This space is made up of disjoint n-dimensional regions (as opposed to one-dimensional intervals in the case of level sets) such that interface sets within one region all share a common topology. We call such regions *topology pockets*, or *pockets* in short. Topological events take place when the offset vector moves from one pocket to an adjacent pocket in the offset space. We call the offset vectors that lie on the boundary of pockets the *critical offsets*. Unlike the critical values in a scalar

function, critical offsets in a vector function form continuous, $(n-1)$-dimensional complex in the offset space, which we call the *critical complex* [1].

Characterizing the critical offsets is key to understanding the topological evolution of the interface set. However, providing a complete, continuous characterization has proven to be a non-trivial task. As our goal is to develop algorithms for practical, discrete inputs, we perform our analysis in a discrete setting and leave the continuous characterization as a venue for future investigation.

3.2.1 Discretization. We consider the discrete input as a simplicial complex C in $\mathbb{R}^d$. Each vertex (i.e., 0-cell) v of C is associated with a vector $\vec{f}_v = \{f_{v,1}, \ldots, f_{v,n}\}$. A common way to construct a function is by piecewise linear (PL) interpolation within each cell of C. Such interpolation is particularly suited for analyzing level set topologies [Edelsbrunner and Harer 2009], since topological events of level sets are restricted to the vertices of C. However, we have observed that the topological events of interface sets in a PL vector function are no longer restricted to vertices of C. In fact, these events can take place in arbitrary locations in the domain. An example is shown in Figure 4 (a) for a 4-labeled domain in 2D; note that the yellow label disappears inside a triangle after the offset changes.

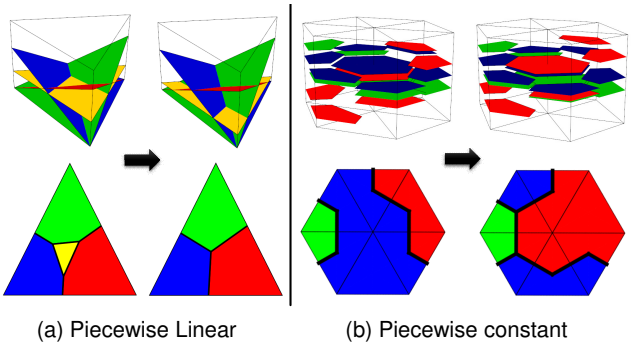


Fig. 4. (a) A topological change of interface sets in PL interpolation can take place in arbitrary locations: lowering the offset of the yellow label causes the yellow region to disappear inside the triangle. (b): A topological change in our PC interpolation is restricted to the dual of the input complex. The offseted functions are shown at the top and the interface sets are shown at the bottom.

To make the analysis of topological events simpler, we opt for a piecewise constant (PC) interpolation. We consider the dual complex of C, noted as C^*, which consists of k-cells that are dual to $(d-k)$-cells of C for $k = 0, \ldots, d$. The vertices of C^* lie at the barycenters of their dual d-cells in C (although the exact locations do not affect the topology analysis). We define the vector function $\vec{f}$ so that $\vec{f}(\vec{x}) = \vec{f}_v$ for any point $\vec{x}$ in the interior of a d-cell in C^* that is dual to a primary vertex v. Since $\vec{f}$ is discontinuous, we adjust the definition of the interface sets as *the union of all cells of C^* that are faces of two d-cells of C^* with different labels*. Examples of such interface sets are shown in Figure 4 (b). In contrast to PL interpolation, interface sets in PC interpolation are restricted to low-dimensional cells of C^*, which allows for simpler characterization of critical offsets (see below).

3.2.2 Critical offsets. In our PC interpolation, the interface set changes *only* when a vertex of complex C alters its label. We call the offsets that trigger these vertex label-changing events the *active* offsets. An active offset is critical if label-changing causes the topology of the interface set to change as well. We will first characterize the active offsets and then identify the subset that is critical.

Given a vertex v and any pair of labels $i, j \in \{1, \ldots, n\}$, there is a collection of offsets $\vec{c}$ at which v may switch its label between i and j. This collection is defined by a set of equality and inequality constraints:

$$\{\vec{c} \mid f_{v,i} + c_i = f_{v,j} + c_j > f_{v,k} + c_k, \forall k \neq i, j\} \tag{3}$$

Geometrically, this collection forms a bounded $(n-1)$-dimensional hyperplane in the n-dimensional offset space. There are C_n^2 such hyperplanes for each vertex v, and they together partition the offset space into n symmetric regions corresponding to the n possible labelling of v. Combining the hyperplanes over all vertices forms a piecewise linear complex in the offset space is the union of all active offsets. We call this complex the *active complex*.

To see this visually, take $n = 3$. Equation 3 defines a 2D half-plane in the 3D offset space. For each vertex v, there are $C_3^2 = 3$ such half-planes, one for each pairing of labels. The three half-planes share a common boundary line in the direction of $\{1, 1, 1\}$ passing through the point $\{-f_{v,1}, -f_{v,2}, -f_{v,3}\}$. Visually, they form a "triblade" around the line. The triblades associated with all vertices intersect to form a honeycomb-shaped active complex (Figure 5 (c) and cutaway in (d)).

Since our reconstruction algorithm is concerned with the topology of individual labels, we define a topological event in this paper as when there is a change in either the number of connected components or genus of the surfaces that bound a particular label. To this end, we can use the same criteria for critical points in a PL scalar function [Edelsbrunner and Harer 2009] to identify topological events in any given label. Specifically, recall that the *star* of a cell σ in a complex consists of all cells that contain σ as a face, and the *link* consists of all faces of cells in the star that are disjoint from σ. Given a labeling of the vertices by an offset vector, we define the *i-link* of vertex v as the union of cells in v's link that contain only vertices with label i. Switching the label of v between i and another label triggers a topological change of label i if the i-link of v is not contractible to a single point.

As an example, we use the criteria to analyze the label-changing event in Figure 4 (b). Since the label of the center vertex changes from blue to red, we focus on the blue-link and the red-link of that vertex. The former consists of one edge (at the bottom) and a vertex (at top-left), which is not contractible, while the latter consists of a single edge (at top-right), which is contractible. Hence the blue label experiences a topological change (a splitting).

An active offset $\vec{c}$ satisfying Equation 3 is critical if *either the i-link or j-link of v is not contractible*. Geometrically, the critical complex, made up of the union of all critical offsets, forms a sub-complex of

[1] The effective dimension of offset space is $n-1$, since the interface set only depends on the relative difference between components of the offset vector. Similarly, the critical complex is an "extrusion" of a $(n-2)$-dimensional complex (Figure 5(c)).

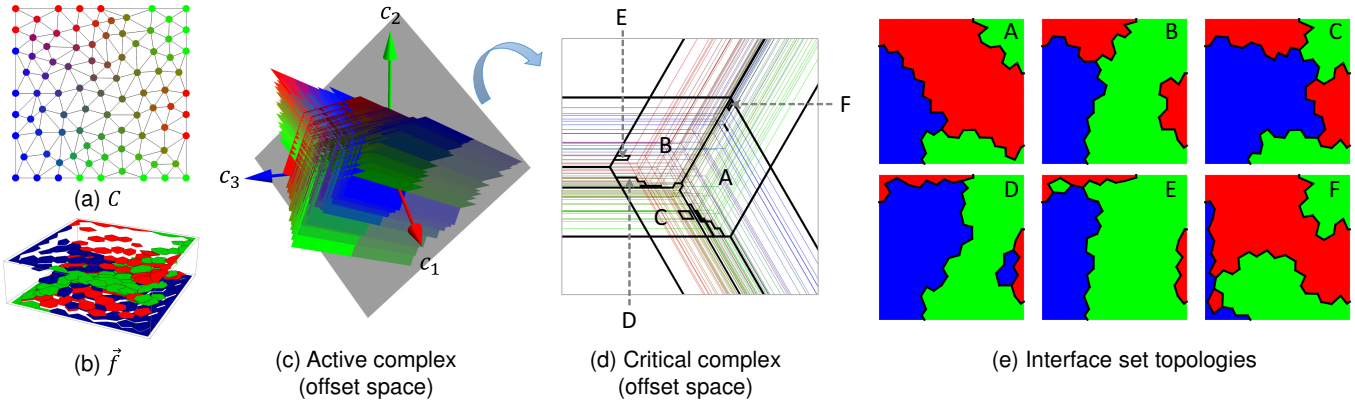


Fig. 5. Topology analysis of 3-labeled interface sets. (a) A complex C where each vertex is colored by the associated 3-vector. (b) The piecewise constant vector function shown as one height map for each label. (c) The active complex in the 3D offset space consisting of one triple half-plane ("triblade") for each vertex of C. (d) A cutaway of the offset space (by the gray plane in (c)) showing the active complex (colored lines) and the critical complex (black lines). (e) Example interface sets in different pockets (see pocket labeling in (d)).

the active complex. As the active complex is piecewise linear, so is the critical complex, which divides the offset space into polyhedral pockets. An example of the critical complex for $n = 3$ labels is shown in Figure 5 (d) on a cross-section of the offset space, and (e) shows topologically distinct interface sets in different pockets.

Note that the per-label topological events that we detect do not cover all possible ways in which the topology of the interface set can change. For example, as we will show in Section 7, the connectivity of the non-manifold junction curves can change without altering the topology of any individual label. It would be interesting in the future to define other types of topological events, which would lead to a more refined critical complex. This would potentially allow a reconstruction algorithm to have finer control over the non-manifold topology of the material interface.

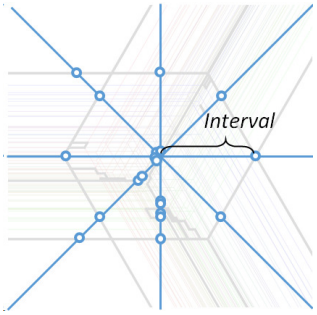
3.2.3 Topology sampling. A direct way to enumerate distinct topologies of interface sets is to explicitly construct the critical complex in the offset space. However, this can be computationally expensive. Our calculations, confirmed by experiments, show that the active complex has a complexity of $O(V^{n-1})$ where V is the number of vertices in the input domain C and n is the number of labels. A brute-force algorithm that first constructs the active complex and then prunes non-critical parts would be impractical.

To tame the complexity, we opt for an approximate, sampling-based approach. The motivating observation is that the more transient topologies tend to correspond to smaller pockets in the offset space. For example, tiny pockets (D,E,F) in the offset space of Figure 5 (d) correspond to topologies that contain small isolated components, as shown in (e). We therefore argue that a regular sampling scheme in the offset space would have a greater chance of finding the more stable topologies, because they are more likely to be captured by larger pockets.

A naïve point sampling scheme is to use regular lattice points in the offset space. However, to form a good coverage, more than a few points would be needed for each dimension of the space, and the total number of samples can still be significant as the dimension of the offset space (i.e., the number of labels) grows.

To reduce the sample count without sacrificing the coverage, we use 1-dimensional *rays* as samples. By intersecting a ray with the critical complex, we can compute *intervals* along each ray within which the interface sets share a common topology, much in the same way as how the level set topologies are enumerated. Unlike point samples, each ray effectively represents an infinite number of point samples (in one direction), and hence a relatively small number of rays are needed.

As our reconstruction algorithm is mostly interested in material interfaces at offsets close to zero, we shoot rays in a radial pattern from the origin of the offset space (see insert). Accounting for the one redundant dimension of the offset space, the rays all lie in hyperplane orthogonal to the 1-vector $\{1, \ldots, 1\}$. To create a pseudo-uniform distribution, we form rays connecting the origin with points on a regular lattice in that hyperplane centered at the origin with $(2b+1)$ points on each of its side, where b is a user-specified small integer. This creates up to $(2b+1)^{n-1}$ rays. In our tests, we found that $b = 1$ offers a reasonable sampling of topologies while keeping the overall execution time low even for large n (e.g., 6-8).



Intersecting a ray with the critical complex can be implemented easily and executed efficiently (i.e., in polynomial time). We start by computing the intersection between a parametric equation of the ray and a hyperplane of the active complex defined by Equation 3, which involves solving a linear equation with one variable and checking the solution against a set of linear inequalities. The intersecting offset, if found, is further

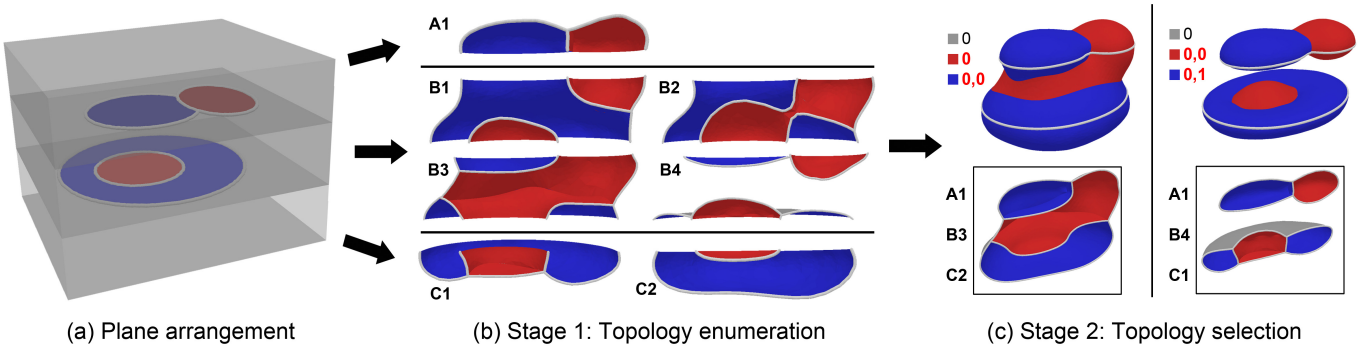


Fig. 6. Reconstruction algorithm: starting from the plane arrangement (a, showing 3 cells divided by the two cross-section planes), our algorithm first enumerates and scores topologies of interface sets within each cell (b, interface sets in each cell are ordered by decreasing scores), then one topology is selected per cell to achieve the topological constraints while maximizing the total score (c, showing solutions under two sets of constraints marked in red; letters by the cutaway views at the bottom are choices of cell topologies).

checked for criticality using the link-based criteria described earlier, which operates in the local neighborhood of a vertex in the input complex C. Repeating these computations for all hyperplanes and sorting the resulting critical offsets produces the desired intervals. The total complexity of the algorithm is $O(H \log H + L * n * H)$ where L is the maximum number of cells in the link of a vertex and $H = VC_n^2$ is the number of hyperplanes. For a fixed dimensionality of the input (3 in our case), this complexity is polynomial in both the number of vertices (V) as well as in the number of labels (n).

4 RECONSTRUCTION ALGORITHM

We now describe our algorithm for reconstructing material interfaces from cross-section inputs. The input to our method consists of a collection of possibly non-parallel planes in 3D, each partitioned into labeled regions by a network of curves. Without additional input, our algorithm produces a geometrically valid material interface that interpolates the curve networks.

The user have the option to specify the desired topology of the subject. For complex subjects made up of many labels (e.g., Figures 1,8,9), the user may not have the knowledge of the precise topology of all labels. Also, the topology of some labels may not matter in downstream applications. To be able to handle these practical situations, we let the user choose *any (possibly empty) subset* of the labels whose topology need to be constrained. For each constrained label, the user specifies the desired number of connected surface components that bound that label as well as the genus for each component. Note that a connected 3D region with an interior cavity ("bubble") counts as two separate surface components.

Our algorithm adopts the classical divide-and-conquer paradigm for cross-section-based reconstruction. We consider the partitioning of the 3D space into convex polyhedral cells by the input planes (known as the *arrangement* in computational geometry). The reconstruction problem is reduced to creating a surface within each cell that interpolates the curves on the boundary of the cell. What differentiates our algorithm from the majority of existing methods is that we create not one, but a collection of surfaces within each cell that differ in topology. These surfaces give rise to a space of topologically different overall reconstructions, among which one that matches the user-specified topology is chosen. Our method proceeds in two stages, which are illustrated on a simple example in Figure 6:

(1) **Enumeration:** For each cell of the arrangement, compute a set of topologically distinct material interfaces that all interpolate the curve network on the cell's boundary. Assign a score to each material interface that measures its likelihood. (Figure 6 (b), Section 4.1)
(2) **Selection:** Select one material interface per cell so that the overall reconstruction matches the user-given topology constraints while the sum of the scores is maximized. (Figure 6 (c), Section 4.2)

Our method generalizes the same two-stage framework of Zou et al. [2015] from closed, manifold surfaces to multi-labeled material interfaces. Besides using the newly developed interface sets for topology enumeration (Section 3), we made several extensions in their framework to address the challenges associated with multiple labels. In the enumeration stage, Zou designed a scalar *indicator* function whose level sets interpolate the curve loops on the cell's boundary. We extend it to a vector function whose interface sets interpolate the boundary curve networks. In the selection stage, Zou uses a region-growing dynamic programming algorithm, which we extend to simultaneously track the topology of multiple labels.

We next detail the two stages while highlighting our extensions over Zou's work. The result of these two stages is a topologically correct reconstruction made up of interface sets within the arrangement cells. Since our interface sets are defined on the dual of a tetrahedral complex, they have jagged appearances. We improve the geometry of the reconstruction in a post-process using the method in [Liu et al. 2008] which creates a refined and fair material interface that still interpolates the input curve networks.

4.1 Enumeration

We consider a polyhedral cell Ω in the arrangement whose boundary $\partial\Omega$ is partitioned by a curve network U into regions with up to n labels. To enumerate material interfaces within Ω, we define a vector

function $\vec{f}$ over Ω and use the techniques developed in the previous section to sample topologically distinct interface sets of $\vec{f}$.

The function $\vec{f}$, as well as the range of the offsets, need to be carefully chosen so that the interface sets interpolate U. In addition, the interface sets should depict a natural extension of U into the interior of Ω. In the special case of two labels (outside/inside), Zou et al. [2015] defines a *harmonic* indicator function that evaluates to 1 (resp. 0) at any point on $\partial\Omega$ that is labeled outside (resp. inside). Harmonic function offers a natural interpolation of the boundary values. This particular function also has the desirable property that any level set at a level in the range $(0, 1)$ interpolates U on $\partial\Omega$.

Extending Zou's scalar function to $n > 2$ labels, we define a vector indicator function $\vec{f} = \{f_1, \ldots, f_n\}$ such that each f_i is a harmonic function and, for any point $\vec{x}$ on $\partial\Omega$, $f_i(\vec{x}) = 1$ if $\vec{x}$ has label i and $f_i(\vec{x}) = 0$ otherwise. It is easy to verify that the interface set of $\vec{f}$ interpolates the curve network U at an offset $\vec{c} = \{c_1, \ldots, c_n\}$ where each c_i lies in the range $[0, 1)$. Figure 5 (a) is an example of such a function in a triangulated 2D cell. As seen in Figure 5 (e), interface sets at different offset vectors within the $[0, 1)$ range (which projects to the center hexagonal region in Figure 5 (d)) touch the cell's boundary at the same locations.

We compute $\vec{f}$ and enumerate its interface sets on a tetrahedralization of Ω. To ensure consistence among neighboring cells, tetrahedral meshing is performed once over the entire 3D domain, constrained by the planes as well as vertices and edges of the curve networks (we use Tetgen [Si 2007]). The harmonic functions are computed on the edge graph of the tetrahedral mesh inside Ω, as in [Zou et al. 2015], which results in a vector $\vec{f}_v$ associated with each vertex v. We then invoke the ray-sampling algorithm (Section 3.3) to compute intervals of offsets within the range $[0, 1)$. For each interval that does not contain the zero offset, we extract the interface set at the offset vector in the midpoint of that interval.

We also extend the scoring method in [Zou et al. 2015] to assess the likelihood of an interface set. The key idea is to treat each harmonic function f_i as the probability distribution of label i. Given an offset vector $\vec{c}$, which gives rise to the labeling $Labels(\vec{c}, \vec{x})$ for any point $\vec{x} \in \Omega$, we consider the joint probability of all labeled points,

$$h(\vec{c}) = \sum_v w(v) \log(f_{v, Labels(\vec{c}, v)}) \tag{4}$$

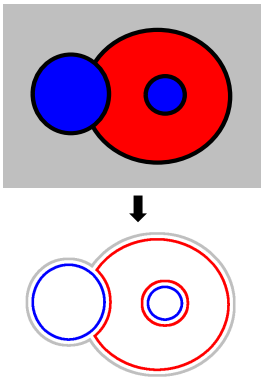
where the summation is over all interior vertices v in the tetrahedralization of Ω, and $w(v)$ measures the total volume of the tetrahedra incident on v.

Note that many interface sets may have the same topology. This can be caused by the same pocket in the offset space being sampled by multiple rays or even multiple times by the same ray. Also, different pockets may correspond to the same interface set topology. For each distinct topology (in terms of the number of connected components and genus for each label), we keep only the interface set with the highest score and remove the rest.

4.2 Selection

We first briefly review the combinatorial optimization method of Zou et al. [2015]. In the context of two-labeled modeling, their algorithm aims to find a closed surface with a user-specified genus from enumerated topologies within each cell. The algorithm is *optimal*, in that the output is guaranteed to have the highest total score among all possible combinations of cell topologies that match the target genus. The basic idea is to grow a *known volume* (KV), which is a union of a subset of the cells, while keeping track of the top-scored solution (i.e., a choice of topology per cell) for each possible surface topology within the KV. The KV is grown by merging with one adjacent cell at a time. Each merging computes the solutions of the new KV from those in the old KV as well as the enumerated topologies in the merged cell. When the KV is grown to the entire domain, the algorithm outputs the top-scored solution that matches the target genus.

We present a simple extension of the algorithm to handle multiple labels. In a nutshell, we treat the problem of creating a n-labeled material interface as creating n overlapping closed surfaces. We encode the curve network as a set of overlapping closed loops, in which each input curve segment is duplicated (see insert). Accordingly, we encode the topology of an interface set in a cell as the topology of a collection of manifold surfaces whose boundaries are these loops. Feeding this representation into Zou's algorithm allows simultaneous tracking of the topology of all labels as the KV is grown.



Although optimal, the algorithm can have a high complexity due to the possibly large number of topologies being tracked, which may grow significantly with the number of labels. We adopt three strategies to curb the space of topologies. The first two strategies follow those in [Zou et al. 2015], while the last one is unique to our multi-labeled context. First, we remove any interface set from the enumeration stage if its topology is deemed too complex, such as containing some surface with non-zero genus in the cell. Second, we remove any intermediate solutions during KV growing that already have higher genus or number of components than the given topology constraints. Third, if the user only constrains a subset of the labels, we only keep intermediate solutions that differ in the topology of those constrained labels. This last strategy effectively makes the complexity of the algorithm depend only on the number of constrained labels.

5 USER INTERACTION

Besides specifying topological constraints, we offer two ways for a user to interact with our algorithm and refine the solution. First, our method produces a ranked list of possible topologies within each cell (see Section 4.1). The user can browse through the list and pick any favorable topology. The user-selected topology will be treated as hard constraint during optimization. This interaction can be useful when the algorithm creates a solution that meets the topological constraints but exhibits undesirable local connectivity. For example, given the input in Figure 7 (a), the automatic solution under genus-0 constraint for the red label places the branching of

the red label in the upper cell (c). The user decides that the branching should take place in the lower cell, and she selects the corresponding topology from our ranked list in that cell (d, left). Incorporating this information, our algorithm then produces another genus-0 solution with the desired branching location (d, right).

To deal with the case that the desired topology is not found in the computed list, we developed a sketching tool whereby the user can create arbitrarily complex topologies within a cell. As shown in Figure 7 (e, left), the user is presented with a cutaway view of the cell on a plane that she can manipulate, and she can scribble on that plane where a particular label should be present. The labeled scribbles are incorporated by our algorithm to update the vector function in that cell, by treating the scribble points as fixed label constraints (similar to points on the cell boundary). A new set of interface set topologies are then enumerated (the top-scored one is shown in (e, middle)) and used for optimization (e, right). Sketching can be particularly useful when our algorithm fails to find a solution satisfying the topological constraints, due to the limited set of topologies explored by the algorithm.

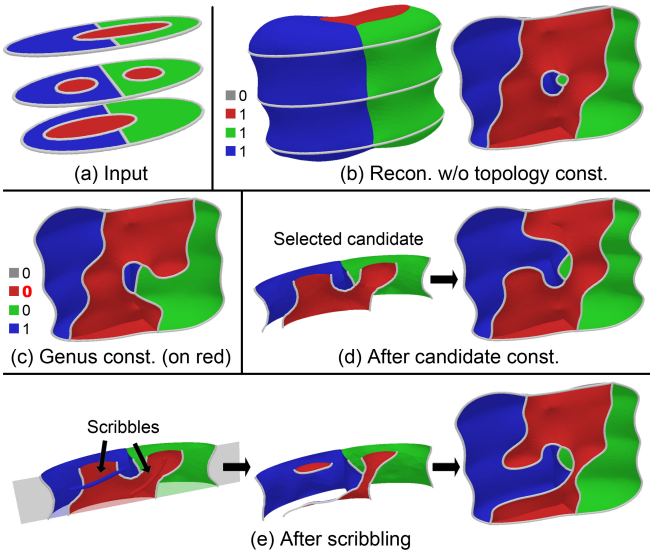


Fig. 7. User interactions: a 4-labeled input (a) and reconstructions without topological constraints (b), with genus-0 constraint on the red label (c), after the user picks a different topology in one of the cells (d), and after the user adds scribbles in that cell (e). All reconstructions are shown in cutaway views as their exterior shapes are similar to that in (b).

6 RESULTS

We test our algorithm and tool on several non-trivial biological examples. These examples contain a large number of labels (6 or more) that interact with each other in complex ways. Two of the examples (mouse brain and liver) are also demonstrated in the accompanying video.

In the mouse brain example in Figure 1, reconstruction without any topology constraints leads to errors (e.g., extraneous components and tunnels) in 3 of the 7 labels (b1-e1). Constraining the topology of these labels results in a satisfactory solution and no new topological errors were introduced to the un-constrained labels (b2-e2).

A more complex scenario is shown on a liver example in Figure 8. Unconstrained reconstruction produces several obvious errors, including two extra components for the green label and an extra tunnel for the outside label (see arrows in (b)). Based on prior knowledge, we also know that the turquoise label should form a "shell" that wraps around the blue and purple labels. In the unconstrained reconstruction, however, both blue and red labels are exposed to the outside, and together they create a tunnel for the turquoise label (see the dotted line in the insert of (b)). After running our algorithm with topology constraints on green, turquoise and outside labels, a new error occurs for the unconstrained blue label - it breaks into two components (see arrows in the insert of (c)). A topologically correct reconstruction is created after adding constraints for both blue and red labels (d). Finally, we added scribbles in two cells to create a more natural branching structure for the green label while keeping the same topological constraints (e) (the same effect can be achieved if we select an alternative topology in one of the cells and scribble in the other cell; see the accompanying video).

Lastly, we demonstrate our algorithm on a chicken heart data set made up of 13 parallel slices containing 8 labels. As shown in Figure 9 (a), this input is particularly challenging as some labels (e.g., light-green) weave through others. Without topological constraints, the reconstruction contains numerous errors, which are all resolved after adding constraints on 5 labels. Figure 9 (d) examines two of these labels, showing the removal of an extra tunnel for the orange label (d1-d3) and the connection of two components for the light-green label (d4-d6) as a result of adding the constraints.

6.1 Performance

The performance of our method depends on many aspects of the input. Besides the number of labels, slices and constraints, the amount of topological ambiguity (as manifested by the number of enumerated topologies) within each cell can also significantly affect the performance. The examples in the paper (Figures 1, 8, 9) exhibit a range of characteristics along these axes, as shown in Table 1.

Table 1 reports the timings of the two stages of our algorithm on these examples. Our tool was implemented in C++ and run on a MacBook Pro with 2.5 GHz Intel Core i7 and 16GB RAM. The first stage, topology enumeration, is by far the most time-consuming stage. In practice, we run this stage only once and invoke the selection stage repeatedly to optimize for different topological constraints. The only exception is that, after a sketching interaction is performed in one of the cells, the enumeration needs to be re-run for that cell. Timing of the enumeration stage is in fact dominated not by ray-shooting (Section 3.3), but by processing the intervals returned by ray-shooting (e.g., extracting interface sets, computing their genus and connected components, and scoring). Timing of the selection stage is sensitive to the number of topologies enumerated in each cell.

7 CONCLUSION AND DISCUSSION

We introduce an algorithm for reconstructing multi-labeled material interfaces that allows the user to explicitly prescribe the topology

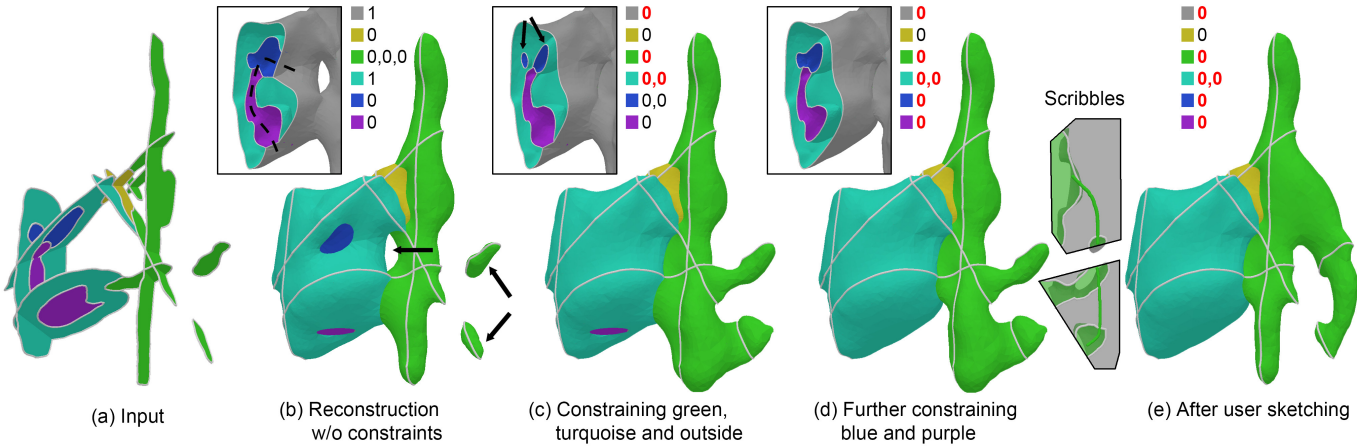


Fig. 8. The Liver data set (a) and reconstructions without topological constraints (b), with some (c) and more (d) constraints, and after applying scribbles (e). Cutaway views are shown in the inserts, and the legends report the per-component genus for each label (constrained genus are in red). Arrows point to topological issues, where the solution does not meet users expectation. See detailed explanations in Section 6.

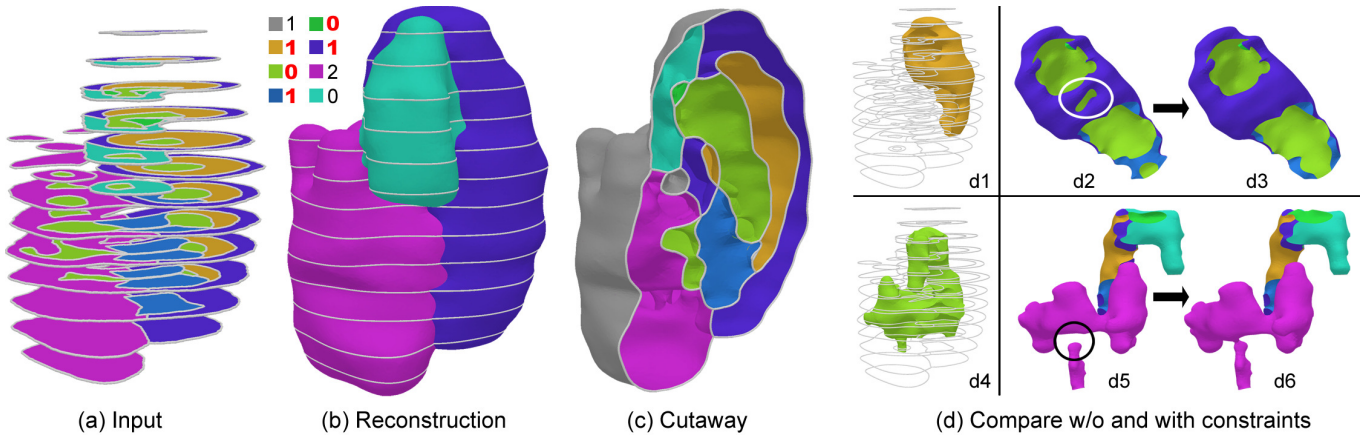


Fig. 9. The chicken heart data set (a) and reconstruction with topological constraints on 5 labels (b, cutaway view in c). (d) compares the reconstruction of orange (top) and light-green (bottom) labels without and with topological constraints. The surfaces in (d2,d3,d5,d6) are colored by labels of adjacent sub-domains to reveal the intertwining of labels.

	#Slices	#Labels (const.)	#Tets.	Max #Topo. per cell	Stage 1 time	Stage 2 time
Fig 1	6	7(3)	91793	8	259s	11ms
Fig 8	5	6(5)	69819	18	212s	83ms
Fig 9	13	8(5)	188401	89	2702s	56s

Table 1. Running time of the two stages of our algorithm on the mouse brain (Fig 1), liver (Fig 8), and chicken heart (Fig 9). Also showing the number of constrained labels, number of tetrahedra, and maximum number of per-cell topologies.

of individual labels. Our key contribution is defining a novel space of material interfaces (as interface sets) that has a rich variety of topologies and allows for systematic exploration. Combined with interactive tools, our method was shown to be effective on non-trivial real-world data in the form of cross-sectional slices.

7.1 Limitations

Our work has several limitations that await further investigation and improvement. Perhaps the weakest aspect is the lack of theoretical analysis of the topological variation of interface sets in the continuous setting. We are already making progress in this direction, and our initial observation is that the criticality of interface sets, like the Jacobi set [Edelsbrunner and Harer 2002], is linked to certain geometric degeneracy of gradients of the scalar functions f_i. We expect such observation to lead to practical and robust algorithms for analyzing topological events in a piecewise linear interpolation.

Our method uses a number of approximating schemes to tame the complexity of topology enumeration, including using piecewise

constant interpolation and ray-sampling. As a result, some topologies could be missed. We would like to explore the practicality of a complete construction of the pockets in the offset space in the piecewise linear setting, perhaps limited to a small region around the origin of the offset space (i.e., the zero offset vector). Extensions of topology filtering methods [Gingold and Zorin 2006; Günther et al. 2014] from scalar functions to vector functions could also be useful for removing small pockets prior to the construction, therefore improving the efficiency.

While our current work focuses on optimizing topology, the geometry of our reconstruction can be further improved in a number of ways. As in [Zou et al. 2015], if an underlying image volume is available, one can create a reconstruction that is aligned to intensity edges in the volume by replacing the harmonic function in each cell with image-based random-walk probabilities. Some other ideas include using more sophisticated scoring function of interface set topologies that favor smoother geometry, and higher-order harmonic functions to improve the continuity of surface across cell boundaries.

7.2 Extensions

Our method opens up new opportunities for topology-aware modeling in the multi-labeled context. As our algorithm does not require the cells to be convex, it can be directly applied to non-planar, and even partial, cross-sections. In the future, we would like to explore similar methods for reconstruction from point cloud data and for fixing topological errors on existing material interfaces. In both cases, one can convert the input in a vector function and potentially apply our divide-and-conquer strategy to explore local topological variations in regions surrounding topological ambiguities.

Another direction for future extension, which we have already started to explore, is to offer topological controls at a finer level, such as over the *adjacency* among labels (i.e., whether and how two labels touch). Note that two material interfaces may share the same per-label topology (i.e., components and genus) but differ in their adjacency. Take the 5-labeled input of Figure 10 (a) for example, the two reconstructions in (c,d) both have genus-0 for each label but differ in how these labels touch each other. In particular, while the interface between the blue and green labels in (d) forms a continuous stripe, this interface is broken into several disconnected patches in (c) due to the touching of the other two labels (purple and yellow). Controlling adjacency can be important for applications (e.g., mesh simplification) that are sensitive to the non-manifold structure of the material interface, in addition to the topology of individual labels.

As a simple example for controlling adjacency, we can modify our algorithm to minimize the number of non-manifold *junction points*, where four or more labels meet (red balls in Figure 10 (c,d)), in addition to meeting the user-specified per-label topology constraints. This is done by expanding our criticality criteria of active offsets to also check for changes in the number of junction points (Section 3.2) and including the total number of junction points on an interface set as part of its score (Section 4.1). The modification creates the result in Figure 10 (d). We will continue to explore how our algorithm can be extended to offer more extensive and precise controls over label adjacency.

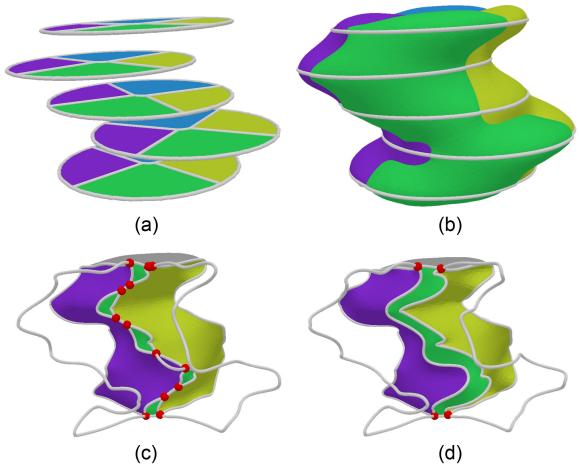


Fig. 10. Given a stack of 5-labeled slices (a) (only blue and green labels touch on each slice), reconstruction with genus-0 constraint on each label produces multiple patches of interface between the blue and green labels (c), whereas a further modification of the algorithm results in a contiguous interface (d) satisfying the same topology constraints. In (c,d) we only show the surfaces of the blue label colored by its adjacent labels, and junction curves and points are shown as grey wires and red balls. The exterior surface of the reconstruction in (d) is shown in (b).

REFERENCES

Hyung Taek Ahn and Mikhail Shashkov. 2007. Multi-material Interface Reconstruction on Generalized Polyhedral Meshes. *J. Comput. Phys.* 226, 2 (Oct. 2007), 2096–2132.

John C. Anderson, Christoph Garth, Mark A. Duchaineau, and Ken Joy. 2008. Discrete Multi-Material Interface Reconstruction for Volume Fraction Data. *Computer Graphics Forum (Proc. of Eurographics/IEEE-VGTC Symposium on Visualization 2008)* 27, 3 (2008).

J. C. Anderson, C. Garth, M. A. Duchaineau, and K. I. Joy. 2010. Smooth, Volume-Accurate Material Interface Reconstruction. *IEEE Transactions on Visualization and Computer Graphics* 16, 5 (2010), 802–814.

Marco Attene, Marcel Campen, and Leif Kobbelt. 2013. Polygon mesh repairing: An application perspective. *ACM Comput. Surv.* 45, 2 (2013), 15.

Pierre-Louis Bazin and Dzung L. Pham. 2007. Topology-Preserving Tissue Classification of Magnetic Resonance Brain Images. *IEEE Trans. Med. Imaging* 26, 4 (2007), 487–496.

Amit Bermano, Amir Vaxman, and Craig Gotsman. 2011. Online Reconstruction of 3D Objects from Arbitrary Cross-sections. *ACM Trans. Graph.* 30, 5, Article 113 (Oct. 2011), 11 pages. DOI: https://doi.org/10.1145/2019627.2019632

Martin Bertram, Gerd Reis, Rolf H. van Lengen, Sascha Köhn, and Hans Hagen. 2005. Non-manifold Mesh Extraction from Time-varying Segmented Volumes Used for Modeling a Human Heart. In *Proceedings of the Seventh Joint Eurographics / IEEE VGTC Conference on Visualization (EUROVIS'05)*. 199–206.

Dobrina Boltcheva, Mariette Yvinec, and Jean-Daniel Boissonnat. 2009. Feature preserving Delaunay mesh generation from 3D multi-material images. *Comput. Graph. Forum* 28, 5 (2009), 1455–1464.

Kathleen S. Bonnell, Mark A. Duchaineau, Daniel Schikore, Bernd Hamann, and Kenneth I. Joy. 2003. Material Interface Reconstruction. *IEEE Trans. Vis. Comput. Graph.* 9, 4 (2003), 500–511.

Kenneth A. Brakke. 1992. The surface evolver. *Experiment. Math.* 1, 2 (1992), 141–165. http://projecteuclid.org/euclid.em/1048709050

Jonathan R. Bronson, Joshua A. Levine, and Ross T. Whitaker. 2014. Lattice Cleaving: A Multimaterial Tetrahedral Meshing Algorithm with Guarantees. *IEEE Trans. Vis. Comput. Graph.* 20, 2 (2014), 223–237.

Hamish Carr and David J. Duke. 2014. Joint Contour Nets. *IEEE Trans. Vis. Comput. Graph.* 20, 8 (2014), 1100–1113. http://dx.doi.org/10.1109/TVCG.2013.269

Hamish Carr, Zhao Geng, Julien Tierny, Amit Chattopadhyay, and Aaron Knoll. 2015. Fiber Surfaces: Generalizing Isosurfaces to Bivariate Data. *Comput. Graph. Forum* 34, 3 (2015), 241–250. http://dx.doi.org/10.1111/cgf.12636

Fang Da, Christopher Batty, and Eitan Grinspun. 2014. Multimaterial Mesh-based Surface Tracking. *ACM Trans. Graph.* 33, 4, Article 112 (July 2014), 11 pages.

Tamal K. Dey, Firdaus Janoos, and Joshua A. Levine. 2012. Meshing interfaces of multi-label data with Delaunay refinement. *Eng. Comput. (Lond.)* 28, 1 (2012), 71–82.

Scott Dillard, Dan Thoma, Bernd Hamann, and John Bingert. 2007. Construction of Simplified Boundary Surfaces from Serial-sectioned Metal Micrographs. *IEEE Transactions on Visualization & Computer Graphics* 13, undefined (2007), 1528–1535.

H. Edelsbrunner and J. Harer. 2002. Jacobi Sets of Multiple Morse Functions. In *Foundations in Computational Mathematics*. Cambridge University Press, 37–57.

Herbert Edelsbrunner and John L. Harer. 2009. *Computational Topology: An Introduction*. American Mathematical Society.

Noura Faraj, Jean-Marc Thiery, and Tamy Boubekeur. 2016. Multi-material Adaptive Volume Remesher. *Comput. Graph.* 58, C (Aug. 2016), 150–160.

Powei Feng, Tao Ju, and Joe D. Warren. 2010. Piecewise Tri-linear Contouring for Multi-material Volumes. In *Advances in Geometric Modeling and Processing, 6th International Conference, GMP 2010, Castro Urdiales, Spain, June 16-18, 2010. Proceedings*. 43–56.

Yotam I. Gingold and Denis Zorin. 2006. Controlled-topology filtering. In *Proceedings of the Tenth ACM Symposium on Solid and Physical Modeling 2006, Cardiff University, Wales, UK, June 6-8, 2006*. 53–61.

David Günther, Alec Jacobson, Jan Reininghaus, Hans-Peter Seidel, Olga Sorkine-Hornung, and Tino Weinkauf. 2014. Fast and Memory-Efficienty Topological Denoising of 2D and 3D Scalar Fields. *IEEE Trans. Vis. Comput. Graph.* 20, 12 (2014), 2585–2594. http://dx.doi.org/10.1109/TVCG.2014.2346432

M. Haitham Shammaa, Yutaka Ohtake, and Hiromasa Suzuki. 2010. Segmentation of Multi-material CT Data of Mechanical Parts for Extracting Boundary Surfaces. *Comput. Aided Des.* 42, 2 (Feb. 2010), 118–128.

Gabor T. Herman, Jingsheng Zheng, and Carolyn A. Bucholtz. 1992. Shape-Based Interpolation. *IEEE Comput. Graph. Appl.* 12, 3 (1992), 69–79. DOI: https://doi.org/10.1109/38.135915

Tao Ju, Frank Losasso, Scott Schaefer, and Joe D. Warren. 2002. Dual contouring of hermite data. *ACM Trans. Graph.* 21, 3 (2002), 339–346.

Tao Ju, Qian-Yi Zhou, and Shi-Min Hu. 2007. Editing the Topology of 3D Models by Sketching. *ACM Trans. Graph.* 26, 3, Article 42 (July 2007).

Byungmoon Kim. 2010. Multi-phase Fluid Simulations Using Regional Level Sets. *ACM Trans. Graph.* 29, 6, Article 175 (Dec. 2010), 8 pages.

Lu Liu, C. Bajaj, Joseph Deasy, Daniel A. Low, and Tao Ju. 2008. Surface Reconstruction From Non-parallel Curve Networks. *Comput. Graph. Forum* 27, 2 (2008), 155–163.

Frank Losasso, Tamar Shinar, Andrew Selle, and Ronald Fedkiw. 2006. Multiple Interacting Liquids. *ACM Trans. Graph.* 25, 3 (July 2006), 812–819.

Miriah D. Meyer, Ross T. Whitaker, Robert M. Kirby, Christian Ledergerber, and Hanspeter Pfister. 2008. Particle-based Sampling and Meshing of Surfaces in Multimaterial Volumes. *IEEE Trans. Vis. Comput. Graph.* 14, 6 (2008), 1539–1546.

J. Milnor. 1963. *Morse Theory*. Princeton Univ. Press, New Jersey.

Nathan Mitchell, Mridul Aanjaneya, Rajsekhar Setaluri, and Eftychios Sifakis. 2015. Non-manifold Level Sets: A Multivalued Implicit Surface Representation with Applications to Self-collision Processing. *ACM Trans. Graph.* 34, 6, Article 247 (Oct. 2015), 9 pages.

Fakir S. Nooruddin and Greg Turk. 2003. Simplification and Repair of Polygonal Models Using Volumetric Techniques. *IEEE Trans. Vis. Comput. Graph.* 9, 2 (2003), 191–205.

Jean-Philippe Pons, Florent Ségonne, Jean-Daniel Boissonnat, Laurent Rineau, Mariette Yvinec, and Renaud Keriven. 2007. High-Quality Consistent Meshing of Multi-Label Datasets. In *International Conference on Information Processing in Medical Imaging 2007*. Netherlands, 200. https://hal.archives-ouvertes.fr/hal-00488043

Jin Qian and Yongjie Zhang. 2011. Dual Contouring for Domains with Topology Ambiguity. In *Proceedings of the 20th International Meshing Roundtable, IMR 2011, October 23-26, 2011, Paris, France*. 41–60.

R. I. Saye. 2015. An Algorithm to Mesh Interconnected Surfaces via the Voronoi Interface. *Eng. with Comput.* 31, 1 (Jan. 2015), 123–139.

Andrei Sharf, Thomas Lewiner, Ariel Shamir, Leif Kobbelt, and Daniel Cohen-Or. 2006. Competing fronts for coarse–to–fine surface reconstruction. In *Eurographics*. Vienna, 389–398. http://www.mat.puc-rio.br/~tomlew/competing_fronts_eg.pdf

Andrei Sharf, Thomas Lewiner, Gil Shklarski, Sivan Toledo, and Daniel Cohen-Or. 2007. Interactive Topology-aware Surface Reconstruction. *ACM Trans. Graph.* 26, 3 (July 2007).

Hang Si. 2007. TetGen. A Quality Tetrahedral Mesh Generator and Three-Dimensional Delaunay Triangulator. (2007). http://tetgen.berlios.de

Julien Tierny and Hamish Carr. 2017. Jacobi Fiber Surfaces for Bivariate Reeb Space Computation. *IEEE Trans. Vis. Comput. Graph.* 23, 1 (2017), 960–969. http://dx.doi.org/10.1109/TVCG.2016.2599017

Greg Turk and James F. O'Brien. 1999. Shape transformation using variational implicit functions. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 335–342. DOI: https://doi.org/10.1145/311535.311580

J. Waggoner, Y. Zhou, J. Simmons, M. D. Graef, and S. Wang. 2015. Topology-Preserving Multi-label Image Segmentation. In *2015 IEEE Winter Conference on Applications of Computer Vision*. 1084–1091.

Kangxue Yin, Hui Huang, Hao Zhang, Minglun Gong, Daniel Cohen-Or, and Baoquan Chen. 2014. Morfit: Interactive Surface Reconstruction from Incomplete Point Clouds with Curve-driven Topology and Geometry Control. *ACM Trans. Graph.* 33, 6 (Nov. 2014), 202:1–202:12.

Zhan Yuan, Yizhou Yu, and Wenping Wang. 2012. Object-space Multiphase Implicit Functions. *ACM Trans. Graph.* 31, 4, Article 114 (July 2012), 10 pages.

Yun Zeng, Dimitris Samaras, Wei Chen, and Qunsheng Peng. 2008. Topology cuts: A novel min-cut/max-flow algorithm for topology preserving segmentation in N-D images. *Computer Vision and Image Understanding* 112, 1 (2008), 81–90.

Yongjie Zhang, Thomas J. R. Hughes, and Chandrajit L. Bajaj. 2007. Automatic 3D Mesh Generation for a Domain with Multiple Materials. In *Proceedings of the 16th International Meshing Roundtable, October 14-17, 2007, Seattle, Washington, USA, Proceedings*. 367–386.

Hong-Kai Zhao, T. Chan, B. Merriman, and S. Osher. 1996. A Variational Level Set Approach to Multiphase Motion. *J. Comput. Phys.* 127, 1 (1996), 179 – 195.

Wen Zheng, Jun-Hai Yong, and Jean-Claude Paul. 2006. Simulation of Bubbles. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '06)*. 325–333.

Ming Zou, Michelle Holloway, Nathan Carr, and Tao Ju. 2015. Topology-constrained surface reconstruction from cross-sections. *ACM Trans. Graph.* 34, 4 (2015), 128.