Optimal Topological Cycles and Their Application in Cardiac Trabeculae Restoration

Pengxiang Wu[1], Chao Chen[2], Yusu Wang[3], Shaoting Zhang[4], Changhe Yuan[2]
Zhen Qian[5], Dimitris Metaxas[1], and Leon Axel[6]

[1] Rutgers University, New Brunswick, NY
[2] CUNY Queens College, Flushing, NY
[3] Ohio State University, Columbus, OH
[4] University of North Carolina at Charlotte, Charlotte, NC
[5] Piedmont Heart Institute, Atlanta, GA
[6] NYU School of Medicine, New York, NY

Abstract. In cardiac image analysis, it is important yet challenging to reconstruct the trabeculae, namely, fine muscle columns whose ends are attached to the ventricular walls. To extract these fine structures, traditional image segmentation methods are insufficient. In this paper, we propose a novel method to jointly detect salient topological handles and compute the optimal representations of them. The detected handles are considered hypothetical trabeculae structures. They are further screened using a classifier and are then included in the final segmentation. We show in experiments the significance of our contribution compared with previous standard segmentation methods without topological priors, as well as with previous topological method in which non-optimal representations of topological handles are used.

Keywords: topology data analysis, trabeculae, cardiac, segmentation, homology localization

1 Introduction

The interior of a human cardiac ventricle is filled with fine structures including the papillary muscles and the *trabeculae*, i.e., muscle columns of various width whose both ends are attached to the ventricular wall (Figure 1). Accurately capturing these fine structures is very important in understanding the functionality of human heart and in the diagnostic of cardiac diseases. These structures compose 23% of left ventricle (LV) end-diastolic volume in average and thus is critical in accurately estimating any volume-based metrics, e.g., ejection fraction (EF) and myocardial mass; these measures are critical in most cardiac disease diagnostics. A detailed interior surface model will also be the basis of a high quality ventricular flow simulation [10], which reveals deeper insight into the cardiac functionality of patients with diseases like hypokinesis and dyssynchrony.

With modern advanced imaging techniques, e.g., Computed Tomography (CT), we can capture details within cardiac ventricles (Fig. 1(left)). However,

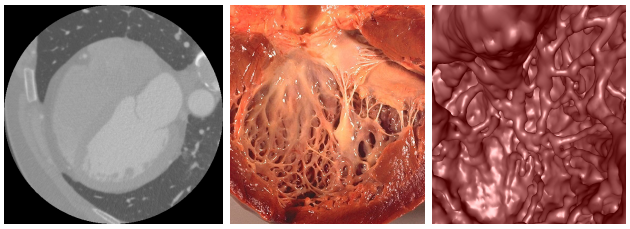


Fig. 1. Left: our input CT image. Middle: interior of LV [8]. Right: our result (a 3D triangle mesh) successfully captures the trabeculae (viewed from the valve).

most state-of-the-art cardiac analysis methods [17, 16], although very efficient, can not accurately capture these complex structures. The challenge is twofold. First, large variation of geometry and intensity of trabeculae makes it difficult to distinguish them from noise. Second, most segmentation models employ global priors, which tend to work against fine structures; the smoothness prior tends to simplify the model and thus remove any fine structures. The shape prior, e.g., the active shape model (ASM), tends to use an average shape and thus remove most fine-scale geometric details.

We exploit novel global information which is more suitable for the extraction of trabeculae, namely, the *topological prior*. A trabeculae is naturally a *topological handle*; both of its ends are attached to the wall, while the intermediate section is freely mobile. Gao *et al.* [9] proposed a topological method that explicitly computes topological handles which are salient compared with their surrounding regions. The saliency is measured based on the theory of *persistent homology* [7] and can be computed efficiently. These handles are further filtered using a classifier and are included in the final segmentation. However, this method fails to provide an ideal description of each detected handle. The generated non-optimal descriptions carry noisy geometric information and will hurt the performance of the classifier and segmentation module down the pipeline.

In this paper, we propose a new topological method that not only detects salient topological handles, but also finds the *ideal description* of each handle. Observe that at the end-diastole state, the heart is maximally relaxed and the trabeculae are maximally stretched out. Therefore, we argue that an ideal description of a topological handle should be geometrically concise; being generally straight rather than wiggling freely. Roughly speaking, a topological handle is

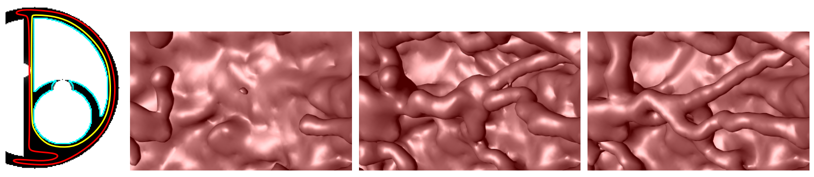


Fig. 2. Left: a topological handle and its representative cycles. The yellow one is the shortest and best describes its geometry. Middle-left: segmentation result without topological prior. Most trabeculae are missed due to the smoothness prior. Middle-right: result with topological prior but without optimal cycles [9]. The reconstructed trabeculae wiggles. Right: our result, using topological prior and optimal cycles generates straight trabeculae.

an equivalent class of loops that can be continuously deformed into each other (Figure 2(left)). Any of these loops can be used to represent the handle. We propose to compute the shortest one as it gives us the most concise description. In real data (Figure 2), we observe that computing optimal loops generates straight trabeculae as desired (Fig. 2(right)), while previous topological method using non-optimal loops generates wiggling trabeculae (Fig. 2(middle-right)).

Our technical contribution is threefold. First, we formulate a new optimization problem, i.e., for each salient topological handle detected by persistent homology theory, compute its best geometric description (the shortest cycle). Second, we prove a theorem (Theorem 1) as the foundation of the computation of the desired optimal cycles. An algorithm based on homology localization theory [1, 2] is proposed accordingly. Third, we propose a new A$*$ search strategy to solve the optimization problem efficiently in practice. The heuristic function of the search strategy is designed based on insights into the problem.

In the last step of our system, accurate geometric features from these optimal loops are used to select correct topological handles as trabeculae and to compute a high quality final segmentation. Figure 3 illustrates the pipeline of our topological method. We validate our method by comparing with segmentation without topological priors and the topological method without geometric optimization [9]. The ground truth of our data is acquired manually. See Section 4 for more details.

Contributions. Our contributions, i.e., the formulation and computation of optimal cycles of persistent homology classes, are also novel and important to the topology data analysis community. While many algorithms have been proposed to compute persistent homology [4, 6], the computation of optimal cycles representing them have never been tackled. The methodology proposed in this paper will have broad applications in various persistent-homology-based data analytics [3, 11, 14, 13].

2 Background

Within this paper, we assume the image domain, $\Omega \subseteq \mathbb{R}^3$, is discretized into a cubical complex, K_Ω, i.e., a collection of vertices, edges, squares and cubes. Each vertex corresponds to the center of a voxel. Each edge connects adjacent vertices. Squares are convex hulls of adjacent four vertices. Cubes are convex hulls of adjacent eight vertices. Any portion of the image domain can be approximated by a subcomplex $K \subseteq K_\Omega$. In particular, we are interested in a *sublevel set*, i.e., the part whose image value is no greater than a given threshold, $f_t^{-1} = \{x \in \Omega \mid f(x) \leq t\}$, in which $f : \Omega \to \mathbb{R}$ is the image (intensity) function. A sublevel

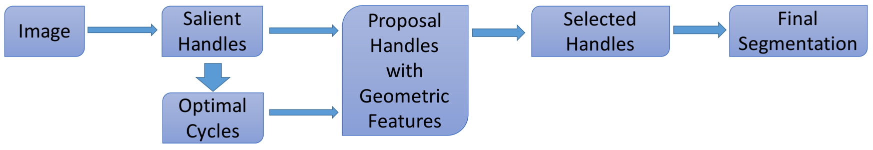


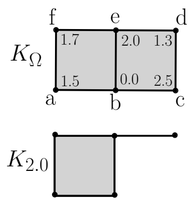
Fig. 3. The flow of our method.

set can be approximated as the complex, $K_t \subseteq K_\Omega$, which includes all elements which fall completely within f_t^{-1}. See Figure 4(a) for an example complex in 2D and one of its sublevel sets (at threshold 2.0). Different sublevel sets have different topology. Next, we introduce how the topology of a sublevel set K_t is defined. Afterwards, we introduce how topology of different sublevel sets are combined to recover the intrinsic structure of the image.

Homology. We focus on one-dimensional homology over $\mathbb{Z}_2$ field. For a general setting, please refer to a standard algebraic topology textbook [12]. Assume a given complex, K, we call its vertices, edges, squares and cubes the *0-, 1-, 2-* and *3-cells*. Any set of p-cells form a *p-chain*. Fixing an index of all p-cells, a p-chain can be written as an n_p dimensional binary vector, in which n_p is the number of p-cells in K. All p-chains constitute an n_p dimensional vector space over modulo-2 addition, called the *chain group*, denoted by $\mathsf{C}_p(K)$. The boundary of a p-cell is a $(p-1)$-chain comprising all the $(p-1)$-cells bounding it. Putting the boundaries of all p-cells together form a $n_{p-1} \times n_p$ matrix called the p-th *boundary matrix*, denoted by ∂_p (Figure 4(b) and 4(c)). The boundary of a p-chain, c, is the modulo-2 sum of all the boundaries of c's elements, and can be written as the product $\partial_p c$. The set of all p-dimensional boundaries (p-boundaries), called the p-th boundary group, is the image of the $(p+1)$-th boundary operator, $\mathsf{B}_p(K) = \operatorname{im} \partial_{p+1}$.

A *p-cycle* is p-chain with zero boundary. The set of all p-cycles, called the *cycle group*, is the kernel of the boundary operator, $\mathsf{Z}_p(K) = \ker \partial_p$. A boundary is a cycle. But the opposite is not necessarily true. The boundary group is a subspace of the cycle group. The quotient space of the latter over the former is called the *homology group*, $\mathsf{H}_p := \mathsf{Z}_p/\mathsf{B}_p$. Each element in the homology group, called a *homology class*, is an equivalent class of cycles whose differences are boundaries of high-dimensional patches. Picking any element in a class h, $z_h \in h$, we can formally write h as $h := [z_h] = \{z_h + b \mid b \in \mathsf{B}_p\}$. In this case, we call z_h the *representative cycle* of h, and h the homology class of z_h.

Persistent homology. Given an image function defined over the image domain Ω, simply selecting a single sublevel set and measuring its topology is insufficient.



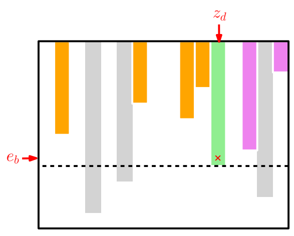
(a)

$$\left[\begin{array}{c|ccccccc} & ab & be & bc & de & cd & af & ef \\ \hline b & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ d & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ a & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ f & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ e & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ c & 0 & 0 & 1 & 0 & 1 & 0 & 0 \end{array}\right]$$

(b)

$$\left[\begin{array}{c|cc} & abef & bcde \\ \hline ab & 1 & 0 \\ be & 1 & 1 \\ bc & 0 & 1 \\ de & 0 & 1 \\ cd & 0 & 1 \\ af & 1 & 0 \\ ef & 1 & 0 \end{array}\right]$$

(c)



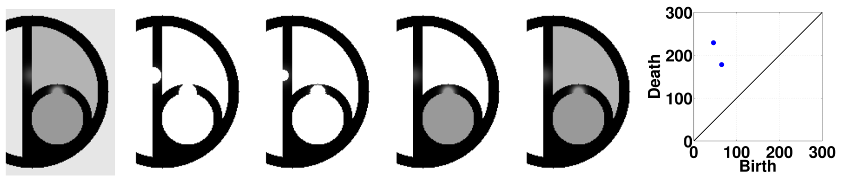
(d)

Fig. 4. Computation of persistent homology. (a), a 2D cubical complex with given function values on all vertices. We also draw the sublevel set at value 2.0. (b) and (c), the boundary matrices of 1D and 2D, whose columns and rows are sorted according to the function value. (d), A schematic illustration of a reduced boundary matrix. The row and column of each pivot (red cross) give the coordinates of one persistence dot.

Instead, we need to jointly consider sublevel sets of all different thresholds. If we increase a threshold t continuously from $-\infty$ to $+\infty$, the sublevel set f_t^{-1} grows from an empty set to the whole image domain, Ω. During the process, different topological structures will be born and dies. In Figure 5, we observe two topological handles. The longer one is born at threshold b_1 and dies at d_1. The shorter one is born at b_2 and dies at d_2. We capture the thresholds at which each structure is born and dies, called its *birth* and *death* time. Intuitively, the birth time is the maximal value along a handle and death time is the maximal value inside the handle. Their difference, called the *persistence*, measures the saliency of the handle.

These structures can be recorded as 2D dots, using their birth and death as the x and y coordinates. The persistence of a dot/structure is its distance from the diagonal ($x = y$). Figure 5(f) shows the point set, called the *persistence diagram*, of the function in Figure 5(a). The theory of persistent homology [7] provides a principled definition of these topological structures and an efficient algorithm to detect all of them, in spite of their shapes and scales. The detected salient topological structures are provably robust to noise [5], and thus reveal intrinsic structures of the function. In general, a topological structure could be a connected component, a handle, or a void (a thickened sphere). In this paper, we focus on one-dimensional structures, i.e., handles.

Computation. Recall the sublevel sets of different thresholds are approximated by intermediate complexes $K_t \subseteq K_\Omega$. If we sort all rows and columns of the boundary matrices of K_Ω according to the function values of the corresponding cells, these sorted boundary matrices also encode the boundary operators of all intermediate complexes; the boundary matrices of any complex $K_t \subseteq K_\Omega$ are the upper-left submatrices of those of K_Ω. To compute the persistence homology, we reduce all boundary matrices of K_Ω using a column-wise Gaussian elimination with special constraints: no columns can be swapped and we only add columns from left to right. Finally, reading the pivot entries of the *reduced matrices* give us the persistence diagram; the row and column of each pivot entry correspond to the birth and death times of each dot in the persistence diagram. See Figure 4(d) for an illustration of the reduced matrix. This algorithm has a cubical complexity, but is very efficient in practice.



(a) Intensity (b) K_{b_1} (c) K_{b_2} (d) K_{d_2} (e) K_{d_1} (f) Diagram

Fig. 5. Sublevel sets in persistent homology. (a) Synthetic intensity function. (b)-(e) Sublevel sets at time $b_1 < b_2 < d_2 < d_1$. We also show the intensity inside the sublevel sets. (f) the persistence diagram, with two dots corresponding to the two handles. At time b_1 and b_2, the long handle and the short one are created. At time d_2 and d_1 the short handle and the long one are destroyed.

3 Method

The overall flow of our method is illustrated in Figure 3. First, we extract salient topological handles from the image based on the theory of persistent homology. Second, we compute optimal loops to represent salient handles. Geometric features are extracted from each optimal loop and are associated to the corresponding topological handle. The combined information is fed to a classifier which selects good handles. Finally, the selected handles are combined with standard image segmentation techniques to generate the final segmentation result.

It remain to solve the optimization problem for each topological handle. In Section 3.1, we formalize the problem. In Section 3.2, we explain an algorithm to solve the problem. This algorithm, although polynomial, is inefficient in both time and space. To develop a practical system, we propose an efficient A* search algorithm (Section 3.3). The heuristic function in the search is based on insights into the topological computation. Due to space constraints, some of the proofs are simplified or omitted. More details will be found in a technical report accompanying this paper.

3.1 Optimal Loops for Persistent Homology

We compute an optimal cycle to represent a topological handle detected by persistent homology, corresponding to a dot in the persistence diagram. Each handle corresponds to a family of homology classes that are created and destroyed at the same birth and death time. At the birth time, several homology classes are created, and only some of them are destroyed at the death time of the handle. We need to identify all these homology classes and find an optimal cycle representing either of them. This is different from a traditional homology localization problem [1, 2], in which we compute the optimal cycle of a fixed homology class.

Problem 1 (Optimal Representative Cycle of a Persistence Dot) *Given a persistence dot $p = (b, d)$, compute the shortest element in the set of all cycles of K_b that are created at time b and become boundaries at time d.*

Next, we prove our main theorem (Thm. 1), which provides an algebraic formulation for the space of all representative cycles of a persistence dot, denoted by $RepCycles(p)$. This result is the foundation of the algorithms we propose in the following sections.

Denoted by $\widehat{\partial_2}$ the reduced 2D boundary matrix (Figure 4(d)). Without loss of generality, we assume only one square has the function value d and one edge has the function value b, denoted as e_b. Let z_d be the corresponding column, whose pivot is at the row of e_b. Let Z_- be the matrix consisting of all columns before column z_d whose pivots are before e_b. Let Z_+ be the matrix consisting of all columns after d whose pivots are before e_b. In Figure 4(d), we select a fixed dot and its corresponding column z_d (green). All orange columns form Z_- and all purple columns constitute Z_+. We have the following theorem.

Theorem 1 (Main) *The set of representative cycles of $p = (b, d)$ are the sum of z_d and linear combinations of columns in Z_-, formally, $RepCycles(p) = \{z_d + [Z_-]x, \forall x\}$.*

Proof. Since edge e_b is the only new edge in K_b, any new cycle created at b contains this edge. The following lemmas are straightforward.

Lemma 2 *Any cycle in $RepCycles(p)$ contains the edge e_b.*

Lemma 3 *When the image domain Ω is Euclidean and has trivial topology, columns $[Z_-, z_b, Z_+]$ constitute a basis of the cycles in complex K_b.*

Based on these lemmas, we prove our theorem as follows. The set of all columns in $[Z_-, z_d, Z_+]$ form a basis of all cycles of the complex K_b, $\mathsf{Z}_1(K_b)$. Since all cycles created at time b has to contain the edge e_b, the space of all cycles of K_b created at time b are in the format of $z_d + [Z_-, Z_+]x$, $\forall x$. However, only columns in Z_- are on the left hand side of z_d, and thus are boundaries in K_d. Therefore, the cycles that are created at b and become boundaries at d are in the format of $z_d + [Z_-]x$, $\forall x$. □

For a more general case, when Ω is not trivial in topology, we can extend the lemma by adding additional columns corresponding to essential cycles, i.e., cycles representing the intrinsic topological handles of Ω.

3.2 OptTopoDij: The First Algorithm

In this section, we introduce our first algorithm to compute the optimal cycle within $RepCycles(p)$. Our algorithm is based on an algebraic annotation technique, which assigns all edges of a given complex different vectors. The annotation satisfies the property that if we walk along any cycle z and add up all the annotations of the edges, the sum immediately certifies whether the cycle belongs to the desired group of cycles $RepCycles(p)$. This technique provides the basis of an efficient algorithm to search through $RepCycles(p)$ for the optimal one despite its exponential size. First, we introduce the annotation technique in details.

Annotation. Let $g' = \text{card}(Z_-)$ and $g = \text{card}(Z_+)$ be the numbers of columns in Z_- and Z_+, respectively. Due to Lemma 3, any cycle of K_b can be written as a linear combination of columns of $Z = [Z_-, z_d, Z_+]$, formally, $z = [Z]\widehat{y_z}$. We call $\widehat{y_z}$ the *coordinate* of z w.r.t. the cycle basis Z. The length of $\widehat{y_z}$ is $g' + g + 1$. We are particularly interested in the coordinate of the cycle z w.r.t. z_d and Z_+. We call the last $g + 1$ entries of $\widehat{y_z}$ z's coordinate w.r.t. $[z_d, Z_+]$, or simply z's *coordinate*. Denote by y_z such coordinate.

Definition 1 (Annotation). *An annotation is a mapping from edges of K_b to $(g+1)$-dimensional binary vectors, $\alpha : \mathcal{E}_{K_b} \to \mathbb{Z}_2^{g+1}$, satisfying the following property. For any cycle of K_b, z, summing the annotations of its edges gives the coordinate of z, formally, $\sum_{e \in z} \alpha(e) = y_z$. For any one-dimensional chain, whether it is a cycle or not, we say its annotation is the sum of the annotations of its edges, $\alpha(c) = \sum_{e \in c} \alpha(e)$.*

An annotation can be computed by the following algorithm. First, we compute a spanning tree of K_b, T. Any edge that does not belong to T, called a *sentinel edge*, form a unique cycle with the tree. We compute the coordinate of such cycle,

z_e, by solving the equation $Zy = z_e$ and keeping the last $g+1$ entries of the solution. Note that z_e is both a cycle and a binary vector in the reduced matrix $\widehat{\partial}_2$. The coordinate is used as the annotation of the corresponding sentinel edge, e. Any edge in the spanning tree has 0 annotation. Computing the annotations for all edges can be achieved in matrix multiplication time $O(n^\omega)$, in which n is the size of K_b. Now we are ready to introduce the main algorithm.

Algorithm. Our algorithm constructs a new graph $\widehat{\mathcal{G}}$ based on the given complex and the computed annotation, so that the optimal cycle problem is equivalent to the shortest path problem in the new graph $\widehat{\mathcal{G}}$, and thus can be solved using Dijkstra's algorithm.

Denote by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ the underlying graph of the subcomplex K_b. We construct a new graph, $\widehat{\mathcal{G}} = (\widehat{\mathcal{V}}, \widehat{\mathcal{E}})$ as follows. The vertices of the new graph is 2^{g+1} many copies of the original vertex set, $\widehat{\mathcal{V}} = \mathcal{V} \times \mathbb{Z}_2^{g+1}$. Vertices in each copy correspond to a different possible annotation. Next, we add 2^{g+1} copies of edges into the new graph as follows. For each edge (u,v) with annotation $\alpha(u,v)$ in $\mathcal{G}$, for each vertex $(u,\beta) \in \widehat{\mathcal{V}}$, we add an edge connecting (u,β) and $(v, \beta + \alpha(u,v))$.

Recall the $e_b = (u_b, v_b)$ is the new edge in K_b. By Lemma 2, all cycles in $RepCycles(p)$ contain e_b. The following theorem gives us an algorithm. The proof is omitted due to space constraints.

Theorem 4 *In graph $\widehat{\mathcal{G}}$, assign infinite weight to any copies of the critical edge e_b, $((u_b, *), (v_b, *))$, and weight one to all other edges. Computing the desired shortest cycle is equivalent to computing the shortest path from $(u_b, 0)$ to (v_b, β_0), in which β_0 is the annotation with one at the first entry and zero at the rest entries, $(1, 0, \cdots, 0)^T$.*

In summary, our algorithm is as follows. Compute edge annotations for all edges. Construct the graph $\widehat{\mathcal{G}}$ based on the annotations. In $\widehat{\mathcal{G}}$, compute the shortest path from $(u_b, 0)$ to (v_b, β_0) using Dijkstra's algorithm. The complexity is $O(n^\omega + 2^g gn \log n)$. Under certain mild assumptions, we can show that the algorithm is polynomial as g is upperbounded by a constant c_θ. More details can be found in the technical report.

3.3 OptTopoA*: An Improved Algorithm

Our algorithm, although polynomial, is exponential to the number of salient structures, c_θ. In particular, the constructed graph $\widehat{\mathcal{G}}$ has a size of $O(2^{c_\theta} n)$. The running time of the Dijkstra's algorithm on $\widehat{\mathcal{G}}$ is $O(c_\theta 2^{c_\theta} n \log n)$. In practice, both the time and space complexity can be too large when c_θ is relatively large.

In this section, we propose a heuristic search method to solve the problem more efficiently. We use the A* algorithm so that we do not have to explicitly construct the whole graph, $\widehat{\mathcal{G}}$, and explore all its vertices. Instead, we only exploit a vertex when necessary, based on a heuristic function. Although the worst case complexity is not better, an A* algorithm often leads to better performance in practice, given a well designed heuristic function.

Recall our goal is to find the shortest path from $\widehat{u_b} = (u_b, 0)$ to $\widehat{v_b} = (u_v, \beta_0)$ within the graph $\widehat{\mathcal{G}}$. At any intermediate vertex $\widehat{w}$, we have the cost of the partially completed path from $\widehat{u_b}$ to $\widehat{w}$, $\mathrm{COST}(\widehat{w})$, and a heuristic function estimating the cost from $\widehat{w}$ to $\widehat{v_b}$, $\mathrm{HEU}(\widehat{w})$. At every iteration, a new node with the minimal estimated total cost $\mathrm{COST}(\widehat{w})+\mathrm{HEU}(\widehat{w})$ is selected and expanded, until the target $\widehat{v_b}$ is reached. A* is guaranteed to find the global optimum.

It remains to define a good heuristic function, $\mathrm{HEU}(\widehat{w})$, which is a lowerbound of the true shortest distance between $\widehat{w}$ and the target $\widehat{v_b}$. Let β be the difference of the annotations of $\widehat{w}$ and $\widehat{v_b}$. The true shortest path from $\widehat{w}$ to $\widehat{v_b}$ is the shortest path connecting w and v_b under the constraint that its annotation is β. Formally, the true shortest distance from $\widehat{w}$ to $\widehat{v_b}$ is $\min_{\gamma\in\Gamma(w,v_b):\alpha(\gamma)=\beta}\mathrm{card}(\gamma)$, in which $\Gamma(w,v_b)$ is the space of all paths connecting w and v_b in the original graph $\mathcal{G}$. Note that we are trying to avoid directly computing such distance. Instead, we approximate it using the following heuristic function.

$$\begin{aligned}\mathrm{HEU}(\widehat{w}) &= \max(\mathrm{HEU}_1(\widehat{w}),\mathrm{HEU}_2(\widehat{w}),\cdots,\mathrm{HEU}_{g+1}(\widehat{w})),\text{ in which}\\ \mathrm{HEU}_i(\widehat{w}) &= \mathrm{argmin}_{\gamma\in\Gamma(w,v_b):\alpha(\gamma)_i=\beta_i}\mathrm{card}(\gamma).\end{aligned}$$

Here $\alpha(\gamma)_i$ and β_i are the i-th entries of the annotation of γ and the difference annotation β, respectively. Intuitively, the cost of $\widehat{w}$ is the optimal length of paths from w to v_b with a fixed annotation β. We construct $g+1$ heuristic function, $\mathrm{HEU}_i(\widehat{w})$, each of which is the shortest distance with paths whose i-th bit annotation is equal to β_i. Since each such heuristic function is a lowerbound of the cost function, taking their maxima is still a valid heuristic and a tighter lowerbound of the cost function.

Finally, we explain how to compute these heuristic functions. For $\mathrm{HEU}_i(\widehat{w})$, consider the set of all edges in the original graph $\mathcal{G}$ whose i-th bit annotation is one, called $\mathcal{E}_i$. A path, γ, has the i-th bit of its annotation being one if and only if it contains an odd number of edges from $\mathcal{E}_i$. So depending on whether β_i is one or zero, we compute the shortest path from w to v_b with either odd or even number of edges in $\mathcal{E}_i$. This can be achieved by constructing two copies of the original graph, G, and add an edge connecting vertices across copies if it belongs to $\mathcal{E}_i$ and connecting vertices within a same copy if it does not belong to $\mathcal{E}_i$. A Dijkstra's algorithm on this graph gives $\mathrm{HEU}_i(\widehat{w})$ as desired.

4 Experiments

We validate the proposed method on synthetic and real cardiac images. We compare two versions of our methods and two baselines: (1) *RegComp*: a classic segmentation method without topological prior, region competition [18]; (2) *NaiveTopo*: the naive topological method without optimal representative cycle computation [9]; (3) *OptTopoDij*: our topological method which computes the optimal representative cycles by constructing the graph $\widehat{\mathcal{G}}$ explicitly and running Dijkstra's algorithm; (4) *OptTopoA**: our topological method which computes the optimal cycle using A* search. RegComp is implemented in ITK-SNAP [15]. All other three are implemented using C++, and are run on a computer with Intel(R) Xeon(R) CPU E5-1660v3 and 32GB RAM on Windows 8.1.

Synthetic experiment 1: non-optimal vs. optimal cycles. We compute persistent homology of the synthetic image in Figure 5(a) and generate cycles to represent the long handle. In Figure 2(left), we show the cycles generated by the baseline topological method, NaiveTopo (in cyan color), and by our method OptTopoDij (in yellow). This example shows that the quality of generated cycle can be very unsatisfying without the proposed optimization algorithm.

Synthetic experiment 2: scalability. Having established the necessity of computing the optimal cycles. We show how important the A* search strategy is in practice. We run our two topological methods, OptTopoDij and OptTopoA*, on synthetic examples with increasing topological complexity, measured by the number of handles. To compute the optimal cycles, we need to construct the graph $\widehat{\mathcal{G}}$, whose size is exponential to the number of handles. In Figure 6(a)(b), we show the memory and time consumption of the two methods. See Figure 6(c) for example input images (each has 400×400 pixels). While the expense of OptTopoDij increases exponentially as we increase the number of handles in the input image, the expense of OptTopoA* stays linear.

Real experiments. We validate our method on a real patient dataset consisting of six cardiac CT images at the end-diastolic state (512×512×320 voxels with spacial resolution from $0.3mm$ to $0.5mm$). For each image, we compute the persistence diagram (Figure 6(d)) and then compute the optimal cycle for each salient persistence dot (persistence $\geq$ 80, see Figure 6(e)). The running time is about seven minutes and the memory is about six to ten GB. This is about the same expense as the topological method without optimal cycle computation, thanks to the high efficiency of the A* search strategy.

After representative cycles of salient handles are extracted, their geometric features, e.g., birth time, death time, relative positions in the ventricle, etc., are used to train a linear SVM classifier, which selects the handles to be included in the final segmentation. We run a six-fold testing; use handles from five images for training and handles from the remaining image for testing, repeat for six times. The prediction accuracy of the classifier is 85.49%±5.25%. The prediction accuracy are similar for both NaiveTopo and OptTopoA*. The reason is the geometric features we are extracting are too simple. Therefore the geometric information of our optimal cycles are not fully leveraged.

However, the optimal cycles do improve our final segmentation quality as they avoid generating wiggling trabeculae. For each handle selected by the classifier, we add the corresponding cycles (thickened) into the standard segmentation

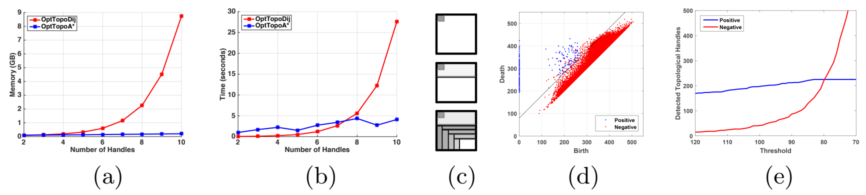


Fig. 6. (a) and (b): memory and time of synthetic data with different numbers of handles. (c): three example input images with different numbers of handles. (d): the persistence diagram of a real heart image. (e): the relationship between the persistence threshold and accuracy.

result to generate the final mesh. We report the distance error of the result mesh from the ground truth. The performance of RegComp, NaiveTopo and OptTopoA* is reported below. See Figure 7 and 2 for qualitative comparisons.

	RegComp	NaiveTopo	OptTopoA*
Distance from GT	0.1862 ± 0.7503	0.1258 ± 0.3810	0.0097 ± 0.1701

The ground truth of our data is extremely difficult to obtain. To train the classifier, we visually inspect each topological handle and decide whether it is a trabeculae based on the intensity function in the surrounding area. To obtain the ground truth mesh, we first use selected handles to generate the initial mesh, and then fine-tune the mesh manually.

Discussion. We observe that without topological priors, most trabeculae are missed. Using topological method but without optimal cycles, the final mesh tends to have wiggling handles. Using our method, the optimal cycles restore trabeculae with high quality mesh. But the classifier may still make mistakes, leading to missing trabeculae in the final segmentation. In the future, we plan to extract more detailed geometric and appearance features based on the optimal cycles, in order to further improve the classifier performance. Our ambitious goal is to develop general purpose topological features for the analysis of images of complex systems, e.g., cardiac trabeculae, neurons, cells, etc.

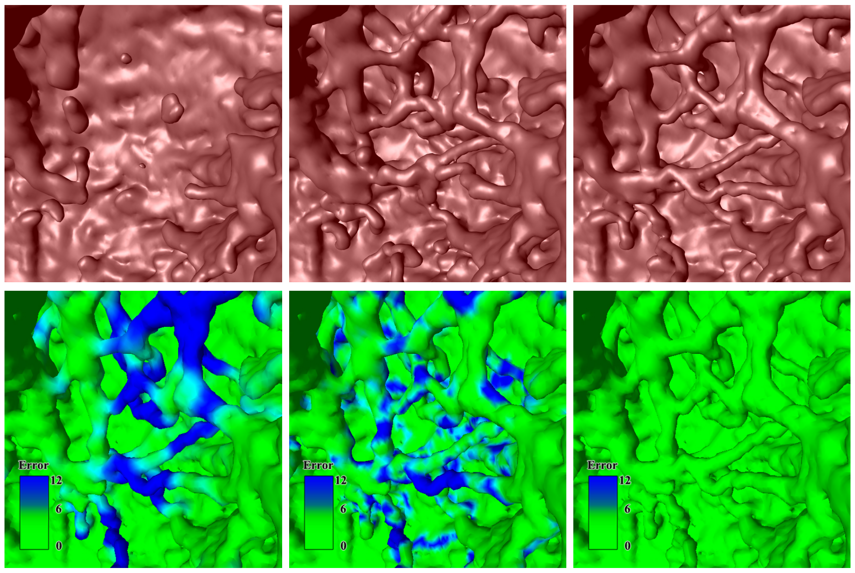


Fig. 7. Qualitative comparison. Top row: segmentation results of three methods, RegComp, NaiveTopo and OptTopoA*. Bottom row: the distance from the ground truth to the segmentation result, rendered on the ground truth mesh. Blue regions are the parts of the ground truth mesh that are not captured by the algorithms.

Acknowledgment. The research of Chao Chen is partially supported by the grants NSF PSC-CUNY 69844-00 47.

References

1. O. Busaryev, S. Cabello, C. Chen, T. K. Dey, and Y. Wang. Annotating simplices with a homology basis and its applications. In *Scandinavian Workshop on Algorithm Theory*, pages 189–200. Springer Berlin Heidelberg, 2012.
2. C. Chen and D. Freedman. Hardness results for homology localization. *Discrete & Computational Geometry*, 45(3):425–448, 2011.
3. C. Chen, D. Freedman, and C. H. Lampert. Enforcing topological constraints in random field image segmentation. In *CVPR*, pages 2089–2096, 2011.
4. C. Chen and M. Kerber. An output-sensitive algorithm for persistent homology. *Computational Geometry*, 46(4):435–447, 2013.
5. D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. Stability of persistence diagrams. *Discrete & Computational Geometry*, 37(1):103–120, 2007.
6. T. K. Dey, F. Fan, and Y. Wang. Computing topological persistence for simplicial maps. In *Proceedings of the thirtieth annual symposium on Computational geometry*, page 345. ACM, 2014.
7. H. Edelsbrunner and J. Harer. *Computational topology: an introduction.* Amer Mathematical Society, 2010.
8. J. Edwin P. Ewing. Gross pathology of idiopathic cardiomyopathy — Wikipedia, the free encyclopedia, 2016. [Online; accessed 09-December-2016].
9. M. Gao, C. Chen, S. Zhang, Z. Qian, D. Metaxas, and L. Axel. Segmenting the papillary muscles and the trabeculae from high resolution cardiac ct through restoration of topological handles. In *IPMI*, 2013.
10. S. Kulp, M. Gao, S. Zhang, Z. Qian, S. Voros, D. Metaxas, and L. Axel. Using high resolution cardiac CT data to model and visualize patient-specific interactions between trabeculae and blood flow. In *MICCAI*, LNCS, pages 468–475. 2011.
11. Y. Li, G. Ascoli, P. P. Mitra, and Y. Wang. Metrics for comparing neuronal tree shapes based on persistent homology. *bioRxiv*, page 087551, 2016.
12. J. R. Munkres. *Elements of algebraic topology*, volume 2. Addison-Wesley Menlo Park, 1984.
13. N. Singh, H. D. Couture, J. Marron, C. Perou, and M. Niethammer. Topological descriptors of histology images. In *Proceedings of the MICCAI Workshop on Machine Learning in Medical Imaging (MLMI)*, 2014.
14. E. Wong, S. Palande, B. Wang, B. Zielinski, J. Anderson, and P. T. Fletcher. Kernel partial least squares regression for relating functional brain network topology to clinical measures of behavior. In *Biomedical Imaging (ISBI), 2016 IEEE 13th International Symposium on*, pages 1303–1306. IEEE, 2016.
15. P. Yushkevich, J. Piven, H. Hazlett, R. Smith, S. Ho, J. Gee, and G. Gerig. User-guided 3d active contour segmentation of anatomical structures: significantly improved efficiency and reliability. *Neuroimage*, 31(3):1116–1128, 2006.
16. X. Zhen, H. Zhang, A. Islam, M. Bhaduri, I. Chan, and S. Li. Direct and simultaneous estimation of cardiac four chamber volumes by multioutput sparse regression. *Medical Image Analysis*, 2016.
17. Y. Zheng, A. Barbu, B. Georgescu, M. Scheuering, and D. Comaniciu. Four-chamber heart modeling and automatic segmentation for 3D cardiac CT volumes using marginal space learning and steerable features. *TMI*, 27(11):1668 –1681, nov. 2008.
18. S. Zhu, T. Lee, and A. Yuille. Region competition: unifying snakes, region growing, energy/Bayes/MDL for multi-band image segmentation. In *ICCV*, pages 416 –423, June 1995.