Fast Computation of Tunnels in Corneal Collagen Structure

Jia Chen
University of California, Irvine
jiac5@uci.edu

James Jester
University of California, Irvine
jjester@uci.edu

M. Gopi
University of California, Irvine
gopi@ics.uci.edu

ABSTRACT

The collagen fiber organization plays an important role in controlling the mechanical strength of the extracellular matrix tissues and in the cornea is thought to control tissue shape and refractive power. In the case of the cornea, collagen fibers run in orthogonal directions, interweaving, branching and anastomosing, that can be imaged using second harmonic-generated signals. But the complexity of the structure makes it difficult for identifying tunnels using previous methods. In the proposed work, we apply an efficient graph based algorithm to find fundamental cycles, including both tunnels and non-tunnels. After tightening the cycle and decoupling composite cycles, we classify the resulting fundamental cycles based on their geometric properties and their relationship between each other. The utility of our tunnel computation and location algorithm is demonstrated on various example cornea structures from various animals.

CCS CONCEPTS

• **Computing methodologies → Shape analysis**;

KEYWORDS

topological analysis, fiber organization, tunnel computation

ACM Reference Format:
Jia Chen, James Jester, and M. Gopi. 2018. Fast Computation of Tunnels in Corneal Collagen Structure. In *CGI 2018: Computer Graphics International 2018, June 11–14, 2018, Bintan Island, Indonesia.* ACM, New York, NY, USA, Article 4, 9 pages. https://doi.org/10.1145/3208159.3208175

1 INTRODUCTION

The cornea is the outermost part of the eye, covering the iris, pupil and anterior chamber. The cornea plays a dual role both as a physical barrier to maintain the ocular integrity, and as a positive meniscus lens to focus light into the retina, so it must be transparent and tough. As transparency is the most important requirement for the cornea, it does not have blood vessels. How collagen structure defines corneal biomechanics and regulates shape is unknown, and insights might provide better therapeutic strategies to correcting refractive errors, including myopia, hyperopia, astigmatism, and ectasia. Therefore, it is of great importance to study the collagen fiber organization inside the cornea. The advent of nonlinear high resolution microscopy (NLO-HRMac) imaging makes it possible to capture the tiny structure of collagen fiber (approximately $1\mu m$ in diameter), as shown in Fig. 1. Previous works [13] [15] [2] [17] have studied fiber interconnectivity by summarizing the statistics such as density of fibers, angles between fibers etc. in the images. However, the 2D analysis cannot distinguish grooves, pockets, protrusions, cavities, and pores and tunnels, and moreover, the 2D analysis results cannot be applied to 3D mechanical and optical simulation. The high complexity and the large amount of noise makes it difficult to analyze the interconnectivity of the collagen fibers in 3D representation. In this paper, we study the interconnectivity in 3D space by identifying tunnel loops.

Topological analysis on geometric data set has proved to work well on high-dimensional, incomplete and noisy data sets [1] . However, there are many difficulties in applying it on complex low dimensional data, such as corneal structure. (1) Computation of topological attributes involving the construction of simplicial complex, Morse complex, homology groups etc., is expensive or even cannot be run on very large data sets. (2) The topological features, such as Betti number, persistence, etc. draw insights on global information but lack relation with local geometry, thus is not intuitive to the non-expert users. In this paper, we supplement the existence of tunnels, which is a topological feature, with a geometrically sensitive representation using edge loop for each tunnel.

Main contributions

The following are the main contributions of this paper:

- We present a weighting method specially designed for corneal structure to find all the fundamental cycles, which prefers finding tunnels, thus have high likelihood of including all tunnels.
- We propose a tightening method, which, instead of searching for shortest length loops, refine the fundamental cycles to make them good for visualization, mechanical and optical simulation.
- We analyze the properties of the refined fundamental cycles, and based on their geometric properties and relationship between each other, we cluster and label them into tunnels and non-tunnels.

2 RELATED WORK

A lot of research efforts have been devoted on corneal collagen structure analysis. [2] demonstrates region growing based method to segment fibers from 2D corneal image slices. [15] examines the arrangement and spacing in rabbit corneal structure. [17] proposes an automated method to measure the collagen fiber angle distribution relative to the corneal surface. All of these works analyze 3D structure based on 2D image slice, but a lot of information is lost in the conversion from 3D space to 2D image representation.

CGI 2018, June 11–14, 2018, Bintan Island, Indonesia

ACM ISBN 978-1-4503-6401-0/18/06…$15.00
https://doi.org/10.1145/3208159.3208175

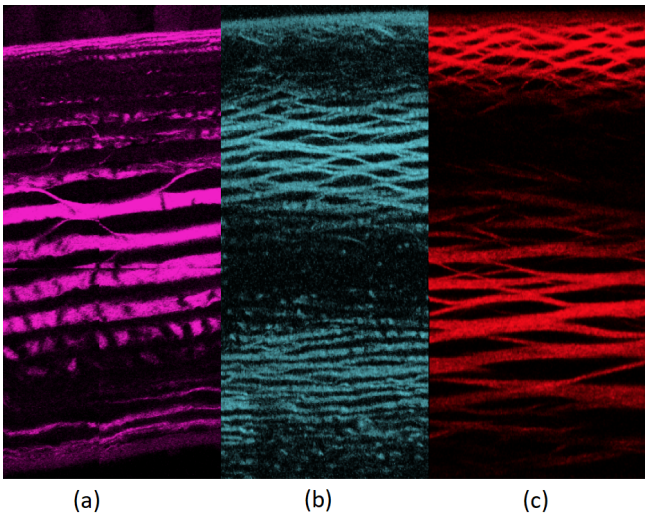

Figure 1: Corneal images slices captured using NLO-HRMac. (a) bull frog (b) chicken (c) hawk.

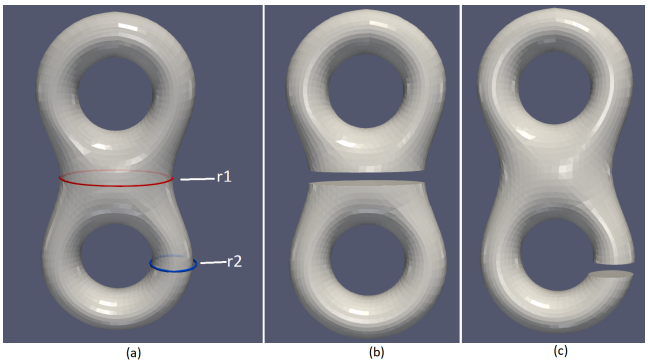

Figure 2: Fundamental cycle and non-fundamental cycle. Both r1 and r2 are non-trivial loops (a), but r1 is a separating loop (b) while r2 is not (c). So r2 is a fundamental cycle, while r1 is not.

Furthermore, mechanical and optical simulation requires 3D representation, which the 2D image analysis cannot provide. Thus, a method analyzing the corneal structure in 3D space is required.

Fundamental cycles on a surface mesh are non-separating loops that do not cut the model into more components (Fig 2). A number of algorithms have been proposed to compute fundamental cycles on surfaces. [18] uses Reeb Graph to identify small handles and remove them for topology repair. [5] computes tunnel and handle loops using topological persistence, thus requires tetrahedral meshing which itself is not triavial for large model. [4] constructs Reeb graph from the mesh to avoid tetrahedral tessellation, thus considerably speeding up the process of finding fundamental cycles. [20] utilizes a graph built from the medial axis, however, the construction of accurate medial axis for complex model remains a challenge. [7] presents a graph based algorithm to find fundamental cycles aligned with the principal curvature directions of a surface. This approach, based on classical spanning tree contruction, is much more efficient than the other methods.

Another set of works explore to extract fundamental cycles from volumetric meshes. Those methods construct simplicial comlex or Morse complex from the volume, and locate the fundamental cycles using persistent homology. [14] determines the Morse complex of a two or three-dimensional grayscale digital image, applies it to analyze complex structures such as 3D foam. [11] defines two geometric measures that enrich the Betti numbers of a binary volume. [21] extends hololology cycle localization problem to any topological space. The key of the approach is to blow up the space into local pieces, and the persistence barcode localizes the topology of the original space with respect to the cover. Since these volume based methods depend on the construction of simplicial or Morse complex, which itself is non trivial, they are not applicable when the volume is as complicated as the corneal structure.

Approaches studying the structure of tunnels, or similarly pores, cavaties, have been proposed in various domains. [9] uses α-hulls to identify small tunnels that are not accessible by a user-defined ball rolling on the surface. [12] applies Morse theory to segment a surface mesh into simple pieces called "pants", which, up to topology, are genus orientable surfaces each with 3 boundary components, however, although they are related, the "pants" are not directly useful for locating the tunnels. [19] and [16] study the tunnels in protein structure, and trace the pathway from the internal cavity to the protein surface. Whie the tunnels in protein are mostly of a similar size, tunnels in corneal structure vary a lot in size as shown in Fig. 1, which is hard to trace. Therefore, the same strategy cannot be applied to cornea.

3 OVERVIEW

The input of our algorithm is a manifold mesh extracted from corneal image slices. The output is a set of edge loops each representing a tunnel. The main steps of the algorithm are as follows. (1) A graph algorithm with carefully designed weighting function is applied to the manifold mesh, and it finds all the fundamental cycles, including tunnels and non-tunnels. (2) The cycles found might be geometrically bad, e.g., they may be unnecessarily long or winding, or they may have undesired orientation. We refine the local geometry as well as the orientation of the fundamental cycle. (3) The fundamental cycles may be a result of combining more than one tunnel/handle loop. After cycle refinement, we resolve these cases by breaking the fundamental cycles into component fundamental cycles. (4) We then classify all the component cycles into tunnels and non-tunnels based on their geometric properties and their relationship between each other. Algo. 1 gives an overview of the algorithm.

4 FUNDAMENTAL CYCLES

A *loop* on surface M is a closed path. A loop γ is *trivial* if γ alone bounds a subset of M, or in other word, it can be continuously retracted into a point. Otherwise, the loop is non-trivial. A fundamental cycle on M is a class of non-trivial, non-separating loops that does not split M into multiple components, as shown in Fig. 2. It is known that a closed surface of genus g has $2g$ fundamental

Input : A closed triangular mesh M with genus g
Output: A set of g tunnel loops of M
Step 1: Compute $2g$ fundamental cycles including tunnels and non-tunnels (Section 4);
Step 2: Tighten the identified fundamental cycle (Section 5);
Step 3: Decouple the cycles which are mixed with each other (Section 6);
Step 4: Classify the cycles into g tunnels and g non-tunnels (Section 7);

Algorithm 1: Algorithm overview.

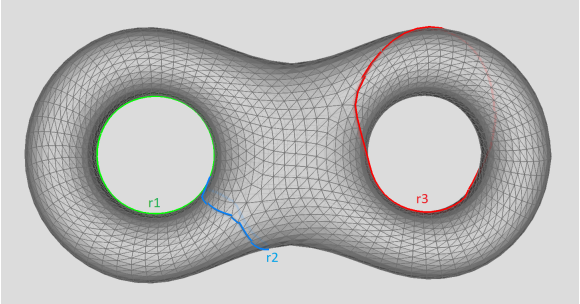

Figure 3: Example fundamental cycles on a double torus. r1 is a tunnel loop. r2 is a handle loop. r3 is neither.

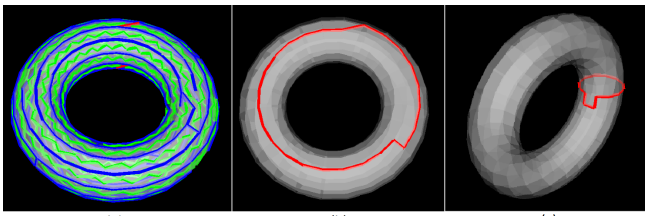

Figure 4: Spanning trees of the primal and dual graph. (a) The spanning tree for primal graph is colored as blue, and the spanning tree for dual graph is colored as green. The two leftover edges are colored as red. When adding the leftover edges into the spanning tree of the primal graph, a loop is formed for each of the leftover edges. For torus, one of the formed loops is tunnel loop (b), the other is handle loop (c).

cycles, and the surface can be cut into a disk by cutting it along loops belonging to these $2g$ fundamental cycles.

Two special kinds of fundamental cycles are *tunnel loops* and *handle loops* [5]. A connected closed surface M embedded in R^3 partitions R^3 into two regions I and O. I is called the *interior* of M and O is called the *exterior* of M. A loop γ is a *handle loop* if it is trivial in O but non trivial in M. A loop γ is a *tunnel loop* if it is trivial in I and non trivial in M. All handles and tunnels are fundamental cycles, but not all fundamental cycles are handles or tunnels. For example, in Fig. 3, $\gamma 1$ is a tunnel loop, $\gamma 2$ is a handle loop, and $\gamma 3$ is neither.

Tunnels and handles may be computed using 3D tetrahedralization, directly following their definition on interior/exterior spaces [5] [10], or computed from Reeb graph which represents the skeleton of the structure [4]. However, both these methods suffer from the complexity of the cornea structure. [7] is based on tree-cotree decomposition of the surface M, thus is computationally efficient for analyzing corneal fibers. The original method guides the construction of the fundamental cycles along the principal curvatures of the mesh, but for corneal structure, the noisy surface makes the computation of curvature directions unreliable. We refine this method to use a weighting function that is more suitable for the mesh reconstructed from the microscope image stack, as well as the orthogonal structure of the fiber directions. This change results in very efficient computation of fundamental cycles that is more relevant to the application. We then classify the computed cycles into tunnels and non-tunnels.

Fundamental Cycle Computation: The tree-cotree decomposition method decomposes the edges on the mesh into three non-intersecting sets (1) edges in the spanning tree T_0 of the *primal graph*, which has vertices on mesh as graph nodes and the edges on the mesh as graph edges (2) edges in the primal graph corresponding to the graph edges in the spanning tree T_1 of the *dual graph*, which has faces of mesh as nodes, and face-face connectivity as graph edges (3) $2g$ leftover edges in the primal graph that are neither in (1) nor in (2) above. When $2g$ leftover edges are introduced into the spanning tree T_0, it will create $2g$ fundamental cycles, shown in Fig. 4. Edge weights are assigned to the primal graph to direct the construction of T_0 which forms the core of the fundamental cycles. Although we cannot directly control which edges are left over, and hence which edges in T_0 will be a part of fundamental cycles, we may adjust the weight values for the construction of T_1 to get the preferred leftover edges. To get short fundamental cycles, the edge weight in the dual graph is set to be the hop-distance in T_0 between the vertices of its corresponding primal edge. With these edge weights, constructing the maximum spanning tree of the dual graph as T_1 will increase the likelihood of the getting the desired leftover edges, and hence the fundamental cycles.

Choice of edge weights: In a typical case, the slices of corneal tissue are taken along the fiber direction and hence the tunnel cycles can be assumed to be the shortest as seen in an image slice. [Any deviation from this assumption is corrected later using our orientation correction method.] The microscopic slices are stacked in the Z-direction, and hence the to retain the loops within the same slice, we assign high cost to the edges between the vertices in two different slices, and low cost for edges in the same slice, so that the minimum spanning tree T_0 will choose the edges in the same slices when possible. Thus our edge weight

$$w_{ij} = \begin{cases} \alpha, & \text{if i and j are in the same slice} \\ \beta, & \text{otherwise} \end{cases} \quad (1)$$

where α is a user specified small number, while β is a user specified large number. Since the edge weights are favored to find tunnels, other cycles that are found, although fundamental cycles, may not be the shortest, nor may be handles. In the following two steps, we tighten the cycles, as well as find appropriate number of accurate tunnels from the set of fundamental cycles.

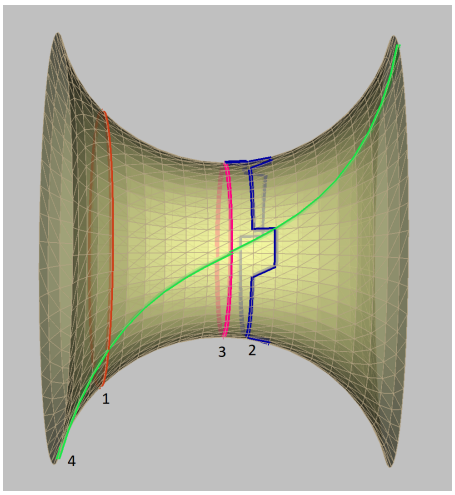

Figure 5: Good and bad fundamental cycles shown on a section of a torus. Cycle 3 is the loop with the shortest path, which is optimal for this tunnel. Cycle 2 contains unneccessary coutours, so is not desired. Cycle 4 does not have any local unnecessary contours, but has a tilt orientation, which is also not good for mechanical and optical simulation. Cycle 1, though longer than the optimal cycle 3, expresses almost the same information to the viewers, thus we consider it also as good.

5 CYCLE TIGHTENING

Each fundamental cycle computed is a representative of other cycles that are topologically equivalent to it. The weighting function introduced in the last section prefers representative loops in the same slice and with shorter distance, but artifacts may still exist, e.g., the loop might be unnecessarily long or winding. So we further refine and tighten the loop.

A few previous works explore tightening the loops. [5] applies "geodesic size" to control the order of adding triangles into filtration. This method works well on persistent homology based methods, but may not be extended to the methods which do not have the process of filtration. [10] deforms the loops along the gradient of the distance field, thus suffers from local minimum and also may not work well on noisy surfaces. [6] enumerates all the fundamental cycles, ordering them by their length, and then find the independent loops greedily. However, the enumeration of all canonical loops makes it only feasible for small data set. [4] constructs shortest path trees at "base points", and finds independent handle loops using "annotations". All of those methods seek for loops with shortest length, and thus searching is inevitable. Note that as this searching problem is thought to be NP-hard in general[4] , the aforementioned methods all apply heuristics and do not guarantee to find the actual optimal loops. For our application, as shown in Fig. 5, a "good" tunnel does not have to be the shortest regarding the sum of the edge lengths. Instead, a good tunnel for visualization and mechanical simulation (1) shoud not have unnecessary contours, e.g., cycle 2 in Fig. 5 (2) should not have tilted orientation, as cycle 4 in Fig. 5. Therefore, in order to tighten the initial fundamental cycles into such defined

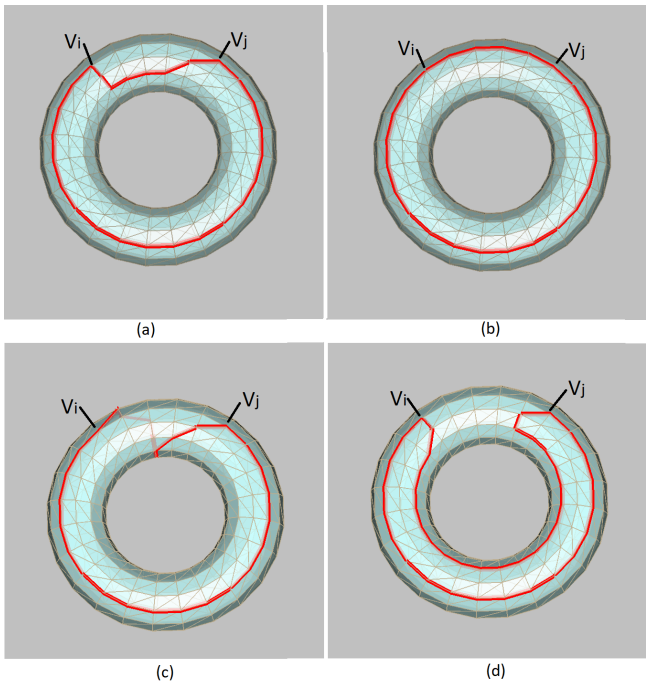

Figure 6: Homotopy equivalence. Given a cycle r1 in (a), r2 in (b) is homotopy equivalent to r1, while cycles r3 (c) and r4 (d) are not as they cannot be continuously deformed to r1. While tightening cycles, we have only homotopy equivalent cycles in the search space.

good ones, instead of searching for the shortest length loops, we iteratively apply refinements as follows.

Homotopy equivalent cycles: Let's recall that each fundamental cycle we find is a representative of a class of equivalent cycles. Two cycles are *homotopy equivalent* if one of them can be continuously deformed into the other [8]. On the triangular mesh, we discretize this relationship by two rules: (1) if cycle r_i is different from r_j by only one triangle, then they are equivalent to each other (2) if r_i is equivalent to r_j, and r_i is equivalent to r_k, then r_j is equivalent to r_k.

During cycle tightening, we need to ensure that the tightened cycle is homotopy equivalent to the original cycle. Each step of the cycle tightening carefully replaces the edges in the original cyle with alternative ones. For example, in Fig. 6, when replacing the edges between vertices i and j with an arbitrary path from i to j, the modified loop may not be homotopy equivalent to the original cycle. In order to avoid the cases such as (c) and (d) in Fig. 6, we restrict the searching of alternative edges as follows (1) In each step, we search alternative edges only in *one-ring neighborhood* of the current cycle, which is composed by all incident faces or edges and neighboring vertices of the current cycle. This is to avoid case (c) in Fig. 6, as the path in one ring neighborhood is impossible to create new winding around the surface. (2) The one-ring neighborhood of a fundamental cycle has three edge loops, and we label them with consistent direction, as shown in Fig. 7. In order to avoid the case

(d) in Fig. 6 , the alternative path is required to be consistent with the labelled edge direction. (3) Additionally, in the alternative path, no vertex is allowed to appear more than once.

Path cost definition: Given a fundamental cycle, we seek for a better alternative from all the homotopy equivalent cycles in its one ring neighbor. As discussed earlier, we consider a cycle as good when it has no uncessary contours, and correct in orientation. The uncessary contours can be avoided using sum of edge lengths as path cost. However, as shown in Fig. 9, even if the path is the shortest in the search space, it cannot guarantee the desired orientation. We refine the orientation of the tunnel loops based on a simple observation: when a cycle is tilted, in the neighborhood of the vertex v_i on the cycle, there exist a vertex that is nearer to the centroid C of the cycle than v_i, as shown in Fig 9. Therefore, we apply the distance between the vertex and the centroid as a measure for orientation correction. Combining these two together, for a cycle γ we have the path cost:

$$\phi_\gamma = \alpha \sum_{v_i \in \gamma} |\overrightarrow{v_i C}| + \sum_{e_i \in \gamma} |e_i| \tag{2}$$

where e_i is the ith edge in the cycle, and α is a user specified coefficient which keeps the balance between edge lengths and orientation correctness.

Cycle searching as a scheduling problem: We iteratively search for a better alternative cycle and replace the original one with it. The search space gets big when the length of the original cycle is long, thus enumeration is not feasible. To simplify the problem, we consider another problem first: given a vertex v_i in the one ring neiborhood, what is the shortest non-trivial loop which passes through v_i? As shown in Fig. 8, we flatten the one ring neighborhood, then the region can be considered as three assembly lines with v_i as their starting and ending points, when a job is on the assembly lines, it has to follow the labelled dicrection but they can switch between the lines freely along the undirected edges. We define the cost the same way as in eq. 2, and the shortest schedule is corresponding to the optimal path in original one ring neighborhood. To choose v_i which is the starting and ending point of the schedule, we greedily pick the vertex that is nearest to the centroid in the one ring. As scheduling problem can be solved by dynamic programming in $O(n)$ time [3], the searching in each step can be done in $O(n)$ time, where n is the number of vertices in the original fundamental cycle. Note that since we search for alternative cycles only in one ring neighbor, the algorithm does not find the globally shortest path in most cases. But as discussed earlier global shortest path searching is a NP-hard problem, and our goal is not shortest path cycle but non-contouring and orientation correct cycle which is good for visualization and mechanical simulation.

6 DECOUPLING COMPOSITE FUNDAMENTAL CYCLES.

Our fundamental cycle identification algorithm finds 2g fundamental cycles, which may be (1) tunnels, (2) handles, and (3) ones that are neither tunnel nor handle. For the third class, as it can be considered as a composition of tunnels and handles in topology, we call it a *composite cycle*. The composite cycles are not desired, they may produce visually unpleasant results, and furthermore, if we do not decouple them into pure tunnels and handles, we may miss tunnels which are expected to be found. Fig 10 shows the procedure for decoupling the composite cycles into tunnels and handles.

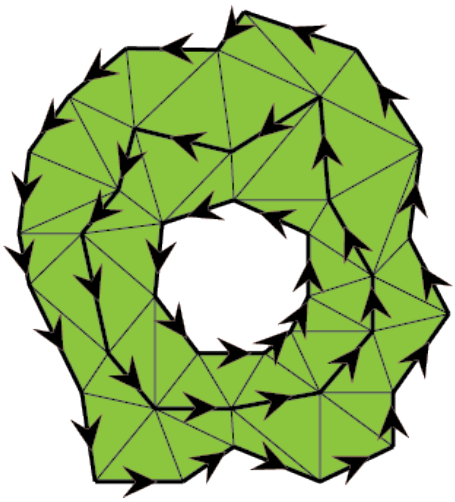

Figure 7: Searching for better alternative cycle in one ring neighborhood. The one ring neighborhood is composed by all the vertices adjacent to the original fundamental cycle. The alternative cycle should be consistent with the labelled directions, and have no self-intersections.

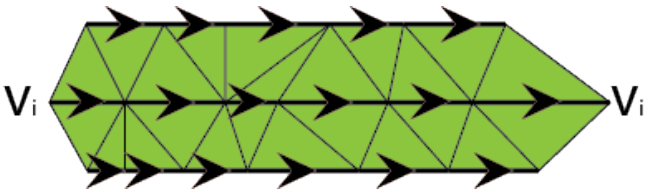

Figure 8: If we consider v_i as starting and ending point, and flatten the one ring neighborhood, the searching problem becomes a scheduling problem with three assmbly lines. The job can be switched between assembly lines along the undirected edges, and the schedule with the lowest cost is corresponding to the optimal alternative cycle.

Step 1. Detection of the composite cycles: We observe that, if the triangulation is dense enough, a handle and a tunnel, after tightening of both cycles, will share no more than an edge. If two intersecting fundamental cycles share more than one edge after cycle tightening, one of them has to be a composite cycle as shown in Fig. 10. If the triangulation is sparse, a tunnel and a handle may share more than one edge, and hence we may be looking for a composite cycle. But the following steps in the algorithm can handle such false-positives in our classification.

Step 2. Replacing shared edges: We replace the shared edges between cycles with the part that is not shared. We label the edges in the two cycles into three group (A) the edges in cyle 1 but not in cycle 2; (B) the edges in cycle 2 but not in cycle 1; (C) the edges in both cycle 1 and 2. We may represent the original cycles we found

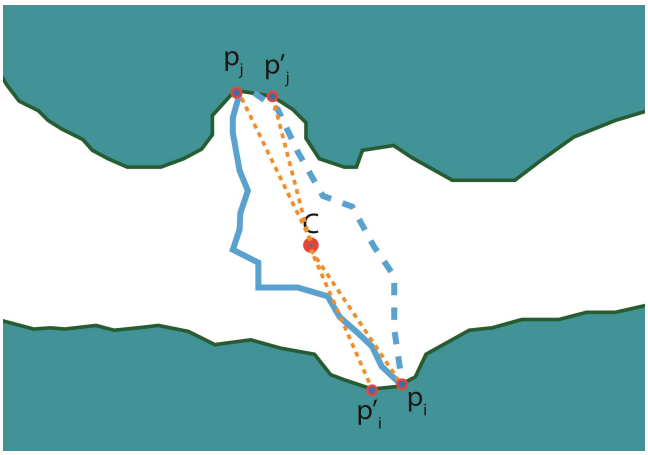

Figure 9: Orientation refinement. If we consider only path in one ring neighborhood, the local shortest paths chosen may have undesired orientation. By computing the distance from vertices to the centroid of the cycle, the orientation is estimated.

as cycle 1=$A + C$, cycle 2=$B + C$. Then if $|A| < |B|$, then we modify cycle 2 to $A + B$, and similarly if $|A| > |B|$, we modify cycle 1 to $A + B$.

Step 3. Tightening.: After replacing the shared edges, we apply the cycle tightening introduced in last section on the modified cycles. We iteratively run the whole procedure until no further change is made to any cycle.

Remark: It is possible that a mixed cycle does not have any intersection with any other fundamental cycle, and thus our algorithm cannot detect this case. However, such occurence seem rare and we have not observed in our experiments.

7 TUNNEL/NON-TUNNEL CLASSIFICATION

The tree-cotree decomposition algorithm determines a basis of the manifold mesh with $2g$ fundamental cycles. However, the algorithm cannot tell if a specific fundamental cycle is a tunnel or a non-tunnel. A direct approach is to check if the cycle can be retracted into a point in the exterior space, but it requires the 3D tessellation of the space. Instead, as the tunnels and non-tunnels show different geometric fetures, the local geometry is brought in to classify the cycles into tunnels and handles. As shown in Fig. 11, for tunnels, the vertex normals tend to point towards the center of the cycle, while for handles, the vertex normals tend to point away from the center. Therefore, we characterize the relationship between the vertex normals and the centroid of the fundamental cycle, and define the *tunnelness measure* for each fundamental cycle as:

$$t = \frac{1}{V} \sum_{i=0}^{V} \overrightarrow{v_i C} \cdot \vec{n_i} \tag{3}$$

where V is the number of mesh vertices in the cycle, v_i is a vertex in the cycle, n_i is the surface normal at v_i, and C is the centroid of the vertices of the cycle. Fig. 12 shows the tunnelness distribution of the fundamental cycles found by our algorithm.

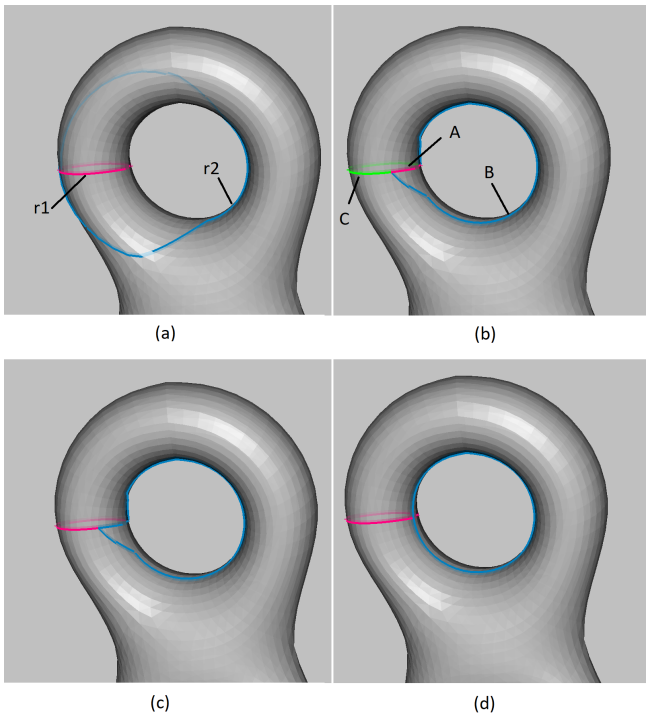

Figure 10: Decoupling composite fundamental cycles. (a)The fundamental cycle identification algorithm may find cycles which are neither tunnels or handles, such as r2. (b) After tightening, this composite cycle shares common edges with adjacent fundamental. r1 is composed by A and C, and r2 is composed by B and C (c) The composite cycle can be decoupled by replacing r2=B+C with r2=A+B (d) The shape of decoupled cycle can be further refined.

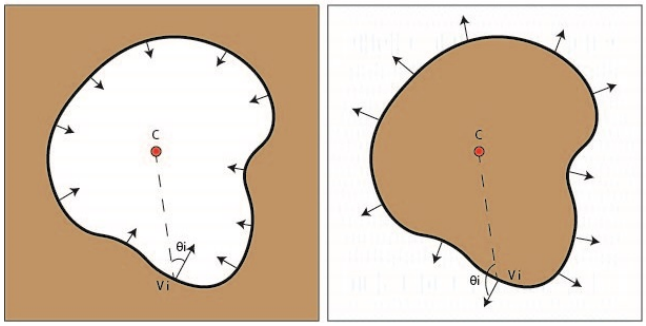

Figure 11: Vertex normals in a tunnel and in a handle. (a) The cross section of a tunnel. (b) The cross section of a handle. The boundary surfaces of (a) and (b) are exactly the same. However, the vertex normals of a tunnel mostly point towards the centroid of the handle, and the vertex normals of a handle mostly point away from the centroid.

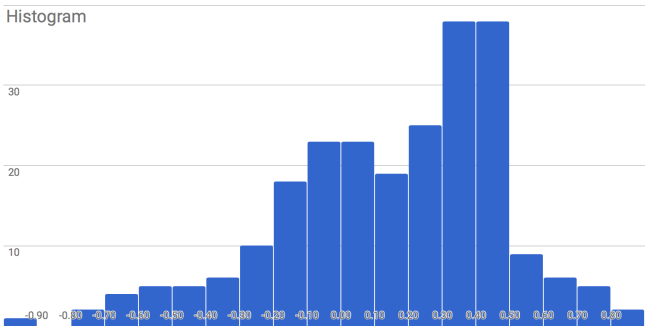

Figure 12: Tunnelness distribution. The tunnels should have high tunnelness values, and non-tunnels should have low tunnelness values. But as the fundamental cycle identification algorithm prefers tunnels, it may find low quality non-tunnel loops which have relatively high tunnelness value. Therefore, thresholding with tunnelness value is not a safe approach, and we need to bring in bipartite graph.

We also observe that a tunnel never intersects with another tunnel, and a handle never intersects with another handle. Therefore, if two fundamental cycles intersect with each other, then the two cycles are tunnel/handle, tunnel/composite, composite/composite, or handle/composite. Since our edge weighting functions to identify fundamental cycles prefer and compute tunnels accurately, we can assume that we only have two kinds of cycles intersections - tunnel/handle, and tunnel/composite. Given the tunnelness measure of each cycle and the intersection relationship between cycles, the problem is to classify tunnels from non-tunnels, such that the resulting sets have intersections only between two sets, and not within a set. In other words, the classification results in a bipartite graph in which each fundamental cycle is a node in the graph, and an edge exists in the graph between nodes i and j, if their corresponding fundamental cycles intersect. For each connected component in the graph, we run a bipartite graph algorithm on it. For example in Fig. 13, in the connected component (a, c, d, g, i), the bipartite algorithm classifies the nodes into two groups, nodes a, c, d should be in the same group, say A, while g and i should in the other group, say B. To determine if group A or B are tunnels, we calculate the average tunnelness values for A and B, and the one with higher average tunnelness value is a tunnel group, and all its members are tunnels. Accordingly, the members of the other group are all labelled as non-tunnels. After processing all the connected components, there might still be isolated nodes left, such as node b, f and h in Fig. 13, which do not intersect with any other fundamental cycle, we sort them by their tunnelness values, and determine their class based on how many tunnels are expected, for example, if k tunnels are not identified in this stage, then we select the fundamental cycles with top k tunnelness values as tunnels. Note that for a manifold with genus g, we expect g tunnels.

8 EXPERIMENTS

We apply our algorithm on a series of data sets, including synthesized high genus meshes and meshes extracted from corneal NLO-HRMac image slices. The synthesized meshes are smooth, but

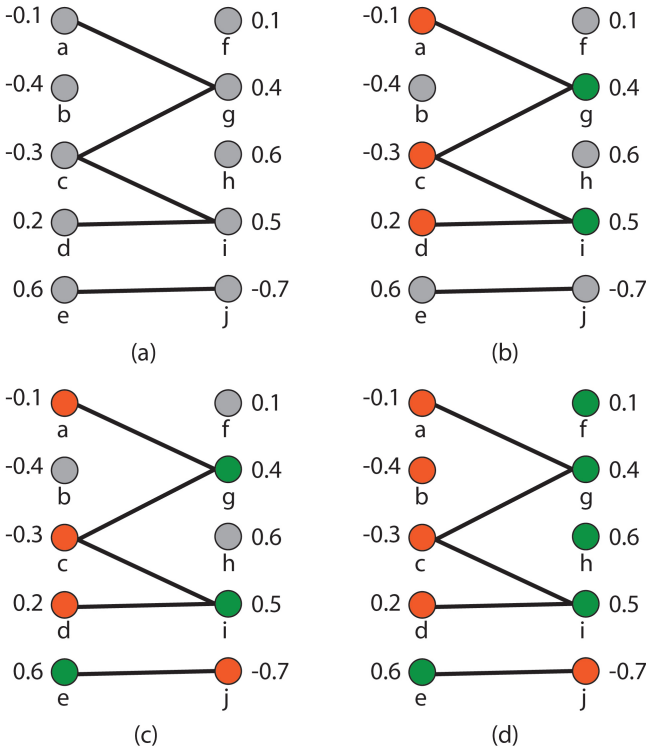

Figure 13: Tunnel/non-tunnel classification using bipartite graph. (a) Graph with each fundamental cycle as a node, and the edges represent intersections between cycles. The tunnelness values are shown beside each node.(b) The largest connected component is partitioned into two classes using bipartite graph algorithm, based on average tunnelness value in each class, we label one of them as tunnel and the other as handle (c) For the second largest connected component, we apply the same procedure (d) The remaining cycles are sorted by their tunnelness values, and we choose the threhold to classify them based on how many tunnels have not been labelled yet.

with high genus. The meshes extracted from image slices are from cornea structures of dog, hawk and chicken. The sizes of the meshes are shown in Table 1. Due to the complexity of our data sets, the 3D tetrahedralization is very expensive, thus the 3D tessellation based methods, [5] [10] are not feasible for failing in our data sets. [7] is efficient to work on large data set, but does not have cycle refinement. The only previous work that can work on large size data sets while has cycle refinement is [4]. In this paper, we extensively compare our algorithm and [4], and evaluate the algorithm from the quality of the identified tunnels, the performance, and the robustness of the algorithm.

Quality of the found tunnels: Fig. 16 shows the tunnels found by our algorithm and by [4]. As [4] seeks for shortest path loop, the tunnels found by [4] are generally smaller in size. But for visualization and mechanical simulation purpose, the result of ours is comparable with theirs.

Model details				[4] (sec)			Our algorithm(sec)		
Data set	vertices	triangles	genus	step 1-5	tightening	total	cycle identification	tightening	total
synthesized_1	54,672	117,824	192	180	1,811	1,991	178	268	446
synthesized_2	114,765	231,138	404	N/A	N/A	N/A	561	878	1,439
dog	382,632	782,962	53	205	3,546	3,751	283	937	1,220
hawk	396,422	793,432	148	360	N/A	N/A	538	1,429	1,967
chicken	495,948	994,136	562	N/A	N/A	N/A	849	1,926	2,775

Table 1: Timing results of our algorithm and [4]'s. Synthesized_1 and synthesized_2 are smooth meshes randomly generated with large number of tunnels. The others are manifold meshes extracted from NLO-HRMac image slices. Tightening column includes time for both tightening and classification. "N/A" means the algorithm failed to finish in two hours after ten times trials.

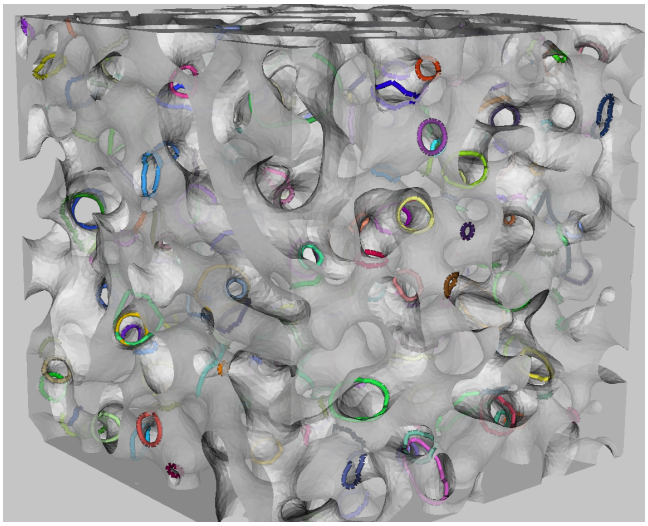

Figure 14: Tunnels found from a synthesized mesh with 404 randomly generated tunnels in various sizes.

Performance: Table 1 shows the timing result. [4]'s performance is greatly affected by the random direction along which it generates Reeb Graph, for fair comparison, we run the program for several times, and choose the shortest timing. For chicken and synthesized_2 data sets, [4] fails to identify the fundamental cycles as these meshes have high genus and complex structure, in which a good Reeb graph for identifying fundamental cycles is difficult to build.

Robustness: [4] is based on construction of Reeb graph, and the quality of the Reeb graph determines if all the fundamental cycles can be identified. As shown in Table 1, for chicken and synthesized_2 data sets, due the complexity and the high genus of the models, good Reeb graph cannot be constructed, thus [4] is unable to identify fundamental cyles on these data sets. Furthermore, [4] relies on inversion of a linking number matrix, and we observe that the numerical issue related to the matrix inversion may introduce excessively long tunnels. As every step of our algorithm is based on basic geometry operation, our algorithm does not have such issues.

9 CONCLUSION

We have introduced an algorithm to extract tunnel loops from a manifold mesh. The algorithm is designed to deal with the complex and noisy structure of corneal collagen fibers. We apply an efficient graph based algorithm to find all the fundamental cycles in the manifold mesh, with a weighting function preferring tunnels. On the computed fundamental cycles, we present algorithms to tighten cycles and resolve cases of composite cycles. To identify the tunnels from fundamental cycles which include both tunnels and non-tunnels, we present a classification algorithm based on the fundamental cycles' geometric properties and their relationship between each other. Finally, we demonstrate the results of our algorithm on corneal structures from various animals, and the algorithm is applicable on various patterns of fiber organization.

ACKNOWLEDGMENTS

We would like to thank the reviewers for their valuable feedback. We are grateful to Professor Ali Mohraz and Todd Thorson for providing the synthesized model in Fig. 14.

REFERENCES

[1] Gunnar Carlsson, Afra Zomorodian, Anne Collins, and Leonidas J Guibas. 2005. Persistence barcodes for shapes. *International Journal of Shape Modeling* 11, 02 (2005), 149–187.
[2] Jia Chen, James Jester, and M Gopi. 2016. Robust segmentation of corneal fibers from noisy images. In *Proceedings of the Tenth Indian Conference on Computer Vision, Graphics and Image Processing*. ACM, 58.
[3] Thomas H Cormen. 2009. *Introduction to algorithms*. MIT press.
[4] Tamal K Dey, Fengtao Fan, and Yusu Wang. 2013. An efficient computation of handle and tunnel loops via Reeb graphs. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 32.
[5] Tamal K Dey, Kuiyu Li, Jian Sun, and David Cohen-Steiner. 2008. Computing geometry-aware handle and tunnel loops in 3D models. In *ACM Transactions on Graphics (TOG)*, Vol. 27. ACM, 45.
[6] Tamal K Dey, Jian Sun, and Yusu Wang. 2011. Approximating cycles in a shortest basis of the first homology group from point data. *Inverse Problems* 27, 12 (2011), 124004.
[7] Pablo Diaz-Gutierrez, David Eppstein, and Meenakshisundaram Gopi. 2009. Curvature aware fundamental cycles. In *Computer Graphics Forum*, Vol. 28. Wiley Online Library, 2015–2024.
[8] Albrecht Dold. 2012. *Lectures on algebraic topology*. Vol. 200. Springer Science & Business Media.
[9] Jihad El-Sana and Amitabh Varshney. 1997. Controlled simplification of genus for polygonal models. In *Visualization'97., Proceedings*. IEEE, 403–410.
[10] Xin Feng and Yiying Tong. 2013. Choking loops on surfaces. *IEEE transactions on visualization and computer graphics* 19, 8 (2013), 1298–1306.
[11] Aldo Gonzalez-Lorenzo, Alexandra Bac, Jean-Luc Mari, and Pedro Real. 2016. Two measures for the homology groups of binary volumes. In *International Conference on Discrete Geometry for Computer Imagery*. Springer, 154–165.

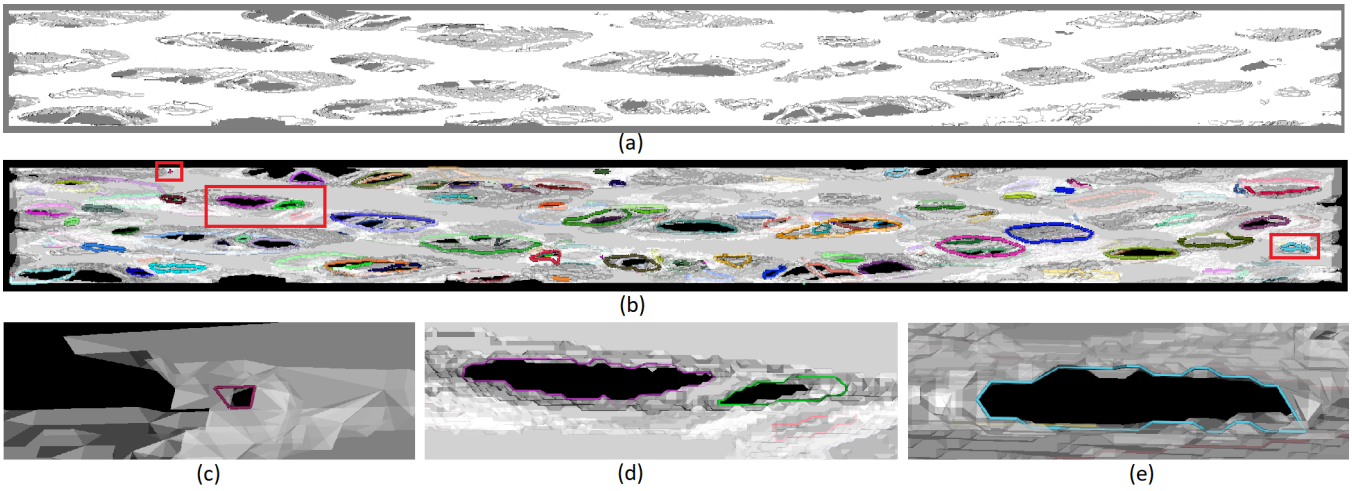

Figure 15: Tunnels found by our algorithm from hawk data set. (a) manifold mesh extracted from corneal images. (b) tunnels found by our algorithm (c-e) zoom in view of selected regions, captured from a different point of view for visualizing nearby geometry.

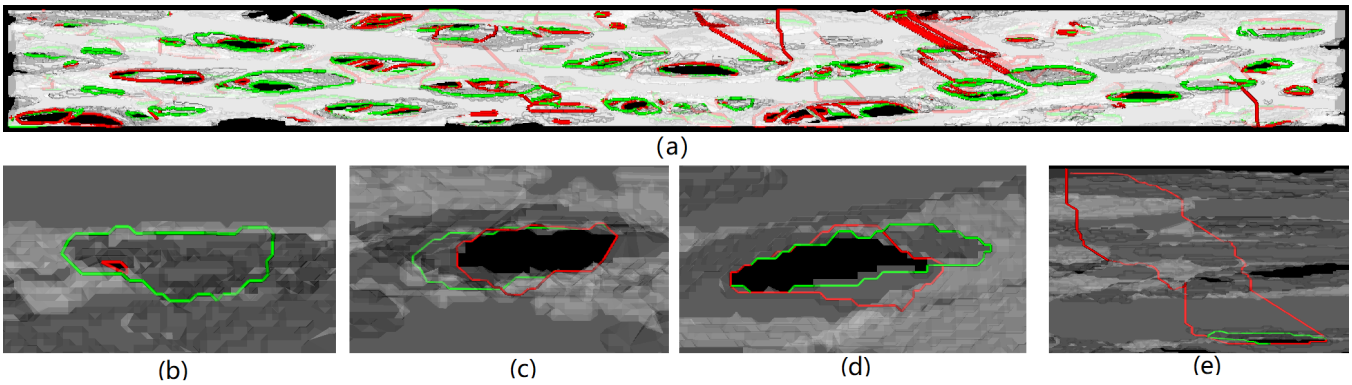

Figure 16: (a) Our vs. [4]'s results on hawk data set, ours are colored as green while [4]'s are colored as red. (b-d) Zoom in views of selected tunnels. Some of the tunnels found by our algorithm are larger in size compared to [4], but this does not have big negavtive impact on visualization and mechanical simulation purpose. (e) Due to numerical issues of matrix inversion, [4] may introduce excessively long tunnels, while our algorithm is stable in all cases.

[12] Mustafa Hajij, Tamal Dey, and Xin Li. 2016. Segmenting a surface mesh into pants using Morse theory. *Graphical Models* 88 (2016), 12–21.
[13] Steven J Petsche and Peter M Pinsky. 2013. The role of 3-D collagen organization in stromal elasticity: a model based on X-ray diffraction data and second harmonic-generated images. *Biomechanics and modeling in mechanobiology* 12, 6 (2013), 1101–1113.
[14] Vanessa Robins, Peter John Wood, and Adrian P Sheppard. 2011. Theory and algorithms for constructing discrete Morse complexes from grayscale digital images. *IEEE Transactions on pattern analysis and machine intelligence* 33, 8 (2011), 1646–1658.
[15] Sara M Thomasy, Vijay Krishna Raghunathan, Moritz Winkler, Christopher M Reilly, Adeline R Sadeli, Paul Russell, James V Jester, and Christopher J Murphy. 2014. Elastic modulus and collagen organization of the rabbit cornea: epithelium to endothelium. *Acta biomaterialia* 10, 2 (2014), 785–791.
[16] Neil R Voss and Mark Gerstein. 2010. 3V: cavity, channel and cleft volume calculator and extractor. *Nucleic acids research* 38, suppl_2 (2010), W555–W562.
[17] Moritz Winkler, Golroxan Shoa, Yilu Xie, Steven J Petsche, Peter M Pinsky, Tibor Juhasz, Donald J Brown, and James V Jester. 2013. Three-dimensional distribution of transverse collagen fibers in the anterior human corneal stromacorneal collagen fiber angle quantification. *Investigative ophthalmology & visual science* 54, 12 (2013), 7293–7301.
[18] Zoë Wood, Hugues Hoppe, Mathieu Desbrun, and Peter Schröder. 2004. Removing excess topology from isosurfaces. *ACM Transactions on Graphics (TOG)* 23, 2 (2004), 190–208.
[19] Eitan Yaffe, Dan Fishelovitch, Haim J Wolfson, Dan Halperin, and Ruth Nussinov. 2008. MolAxis: efficient and accurate identification of channels in macromolecules. *Proteins: Structure, Function, and Bioinformatics* 73, 1 (2008), 72–86.
[20] Qian-Yi Zhou, Tao Ju, and Shi-Min Hu. 2007. Topology repair of solid models using skeletons. *IEEE Transactions on Visualization and Computer Graphics* 13, 4 (2007).
[21] Afra Zomorodian and Gunnar Carlsson. 2008. Localized homology. *Computational Geometry* 41, 3 (2008), 126–148.