Reconstructing Cell Complexes From Cross-sections

Scott E. Dillard[1,2], Dan Thoma[1] and Bernd Hamann[2]

[1] Materials Design Institute, Los Alamos National Laboratory
[2] Institute for Data Analysis and Visualization, Department of Computer Science, University of California, Davis

Abstract. Many interesting segmentations take the form of cell complexes. We present a method to infer a 3D cell complex from of a series of 2D cross-sections. We restrict our attention to the class of complexes whose duals resemble triangulations. This class includes microstructures of polycrystalline materials, as well as other cellular structures found in nature. Given a prescribed matching of 2D cells in adjacent cross-sections we produce a 3D complex spanning these sections such that matched 2-cells are contained in the interior of the same 3-cell. The reconstruction method considers only the topological structure of the input. After an initial 3D complex is recovered, the structure is altered to accommodate geometric properties of the dataset. We evaluate the method using ideal, synthetic datasets as well as serial-sectioned micrographs from a sample of tantalum metal.

1 Introduction

Cross-section imaging is a common technique used to study and analyze the structure of materials. A series of planar cross-sections through a specimen is generated and the cross-sections are used to reconstruct a 3D representation. A related, well-studied problem in computer graphics and visualization asks to construct a surface from a set of curves lying in the cross-sections. There have been many solutions suggested for the *two-phase* version of this problem, in which the reconstructed surface divides space into two "phases" and hence can be a manifold. For example, the zero-surface of a continuous scalar function divides the domain into two phases: points with negative values and points with non-negative values. In this paper we consider the *multiphase* generalization of this problem, in which the non-manifold separating surface divides space into multiple phases. This distinction in illustrated in Figure 1.

Solutions to the problem of constructing a surface from cross-sections can be divided into two types: *mesh-based* ones, operating on an irregular collection of vertices, edges and faces, and *voxel-based* ones, operating on a 3D array of sample points. See the papers by Nonato *et al.* [6] and Braude *et al.* [1] and the references therein for examples of mesh and voxel-based solutions, respectively. Previous solutions to the multiphase problem have all been of the voxel-based type. See the paper by Dillard *et al.* [3] and references therein for examples.

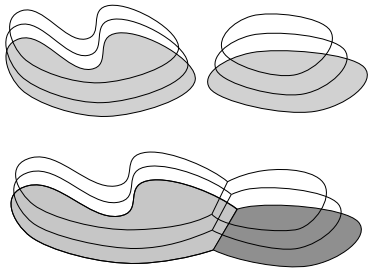


Fig. 1. In a two-phase segmentation (top) every point is incident on at most two regions. In a mult-phase segmentation (bottom) there are "triple points."

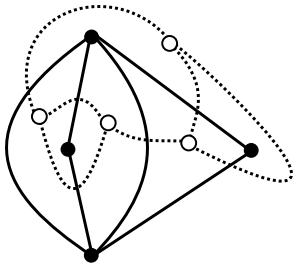


Fig. 2. A Δ-complex, in solid dots and lines, and its dual Δ^*-complex in hollow dots and dashed lines.

We present a novel mesh-based solution to the multiphase segmentation reconstruction problem. In voxel-based methods, topological and geometric properties are conflated, which is beneficial if one wants to optimize some geometric criterion (e.g., smoothness) without being hindered by topological constraints (e.g., genus). In the multiphase problem, however, direct control over reconstructed topology can be important. An example of this kind of control is given by Nonato *et al.* [6] in their mesh-based β-connection method, a solution to the two-phase reconstruction problem. Their method provides a user with control over the reconstruction topology through a parameter β. Higher values of β favor reconstructed surfaces of higher genus. Our method supports a similar type of control in the multiphase setting: The user can prescribe that two regions in two adjacent cross-sections should be path-connected, or not. A significant challenge in the mesh-based setting is the avoidance of self-intersections in the constructed surface, a problem that most voxel-based methods avoid by design.

The data we consider are serial-sectioned micrographs of polycrystalline material. Each phase represents a grain, a region of uniform crystal structure. The ability to prescribe connections between 2D cross-section regions is important when the imaging mode provides more information than just region boundary geometry. In the case of metal micrography, electron back-scatter diffraction (EBSD) measures crystallographic orientations, and we therefore prescribe a connection between two cross-section regions if their orientations are similar and they are relatively close to each other.

2 Definitions

Many of the terms are familiar from simplicial complexes, with subtle yet important differences in definition. A d-simplex $\Delta^d \subset \mathbb{R}^d$ is the convex hull of $d+1$ affinely independent vertices. A face of a simplex is the $(d-1)$-simplex obtained by removing a vertex. Let $\partial\Delta^d$ be the union of all faces of Δ^d, and $\mathring{\Delta}^d = \Delta^d \setminus \partial\Delta^d$. Let the sequence $(v_0 \dots v_d)$ be an ordering of the vertices of Δ^d, then the ordering of the ith face is $(-1)^i(v_0 \dots \hat{v_i} \dots v_d)$ where the "hat" indicates that v_i is removed and a negative coefficient swaps the first two vertices of

the sequence. Let X be a topological space and let $\sigma : \partial\Delta^d \to X$ be a continuous map. A new space $Y = X \cup_\sigma \Delta^d$ is obtained from X by *attaching a cell* to X, giving Y the quotient topology of $X \cup \Delta^d/\sim$, where $y \sim \sigma(y)$ for all $y \in Y$. A d-dimensional *cell complex*, or just *d-complex*, is a topological space constructed by attaching cells of dimension no greater than d. A **Δ-complex** is a collection of maps $\sigma_i : \Delta^d \to X$ such that:

1. The restriction $\sigma_i|\mathring{\Delta}^d$ is injective, and each point of X is in the image of exactly one such restriction. We say that σ_i is a *d-cell*, and let $\mathring{\sigma}_i$ denote the restriction. Referring to σ_i in the context of a set refers to its range.
2. Each restriction of σ_i to one face of Δ^d is one of the maps $\sigma_j : \Delta^{d-1} \to X$, identifying the face of Δ^d with Δ^{d-1} by the linear homeomorphism that preserves the ordering of vertices. We say that σ_i is *attached* to σ_j.
3. The restriction $\sigma_i|\partial\Delta^d$ is injective.
4. A set $A \subset X$ is open iff $\sigma_i^{-1}(A)$ is open in Δ^d for each i.

Conditions 1, 2 and 4 follow Hatcher [4]. We impose condition 3 to prevent degeneracies such as loops. This class of complexes contains simplicial complexes, but is broader. For instance, we may have multiple 1-cells each incident on the same pair of 0-cells. Informally, one may think of a Δ-complex as a triangulation with curved edges and faces.

Two cells σ_i and σ_j are *incident* on each other if $\sigma_i \cap \sigma_j \neq \emptyset$. Two d-cells are *adjacent* if there exists a $(d-1)$-cell and a $(d+1)$-cell on which they are both incident. Let all 0-cells be incident on a single (-1)-cell, and if σ_i and σ_j are of maximal dimension, let them be adjacent only if they are incident on a common $(d-1)$ cell. Since we are concerned here only with 2-complexes and 3-complexes we make the following abbreviations: A *vertex* is a 0-cell. An *edge* is a 1-cell. A *face* is a 2-cell, and henceforth *only* a 2-cell. A *tetrahedron* is a 3-cell of a Δ-complex. A 2-complex that is also a Δ-complex is a 2Δ-complex, and a 3Δ-complex is defined analogously. The *valence* of a vertex is the number of edges that are incident on it.

Let $\mathcal{C}$ be a cell complex. $\mathcal{C}'$ is a *subcomplex* of $\mathcal{C}$ if $\mathcal{C}'$ is a cell complex and $\mathcal{C}' \subseteq \mathcal{C}$. The *closure* of a set of cells is the smallest subcomplex containing it. Two d-complexes $\mathcal{C}$ and $\mathcal{C}'$ are *isomorphic* if there exist bijections $M_k : \mathcal{C}_k \to \mathcal{C}'_k$, $0 \leq k \leq d$, such that if cells $a, b \in \mathcal{C}$ are adjacent then so are $M_k(a)$ and $M_k(b)$. Two d-complexes $\mathcal{C}$ and $\mathcal{C}^*$, are *duals* if there exist bijections $D_k : \mathcal{C}_k \to \mathcal{C}_{d-k}$, $0 \leq k \leq d$ such that if $a, b \in \mathcal{C}$ are adjacent then so are $D_k(a)$ and $D_k(b)$. For example, in a pair of dual 3-complexes, tetrahedra of one complex are mapped to vertices of the other, and faces to edges. Our method is restricted to a certain class of segmentations because it exploits the structure of the dual complex. In particular, the dual must be a Δ-complex. Correspondingly, we call the complex that represents the segmentation a Δ^*-complex. An example of dual Δ and Δ^*-complexes is shown in Figure 2.

A bijection f is a *homeomorphism* if both f and f^{-1} are continuous. If such f exists, its domain and range are *homeomorphic*. A Δ-complex is a *manifold* if for every vertex v, the union of the interiors of cells incident on v (the *star*

of v) is homeomorphic to the open unit ball $\{x : \|x\| < 1\} \subset \mathbb{R}^d$. The relevant implication is that every $(d-1)$-cell in a manifold d-complex is incident on two d-cells. A complex is a *manifold with boundary* if the star of every vertex is homeomorphic to the open unit ball or the half-ball $\{x : x_1 \geq 0, \|x\| < 1\} \subset \mathbb{R}^d$, and its *boundary* is the closure of those $(d-1)$-cells that are incident on only one d-cell. A Δ-complex is a *sphere* if it is homeomorphic to the standard sphere $\{x : \|x\| = 1\} \subset \mathbb{R}^d$, and a *ball* is homeomorphic to $\{x : \|x\| \leq 1\}$.

A *homotopy* between two continuous functions $f, g : X \to Y$ is a continuous function $h : [0,1] \times X \to Y$, such that $h(0,x) = f(x)$ and $h(1,y) = g(y)$. Denote the existence of h by $f \simeq g$. X and Y are *homotopy equivalent* if $f \circ g \simeq \mathrm{id}_Y$ and $g \circ f \simeq \mathrm{id}_X$, where $\mathrm{id}_X(x) = x \in X$. A space is *contractible* if it is homotopy equivalent to a point. Crucially, if $\mathcal{S}$ is a contractible subcomplex of $\mathcal{C}$, then the complex $\mathcal{C}'$ obtained by replacing $\mathcal{S}$ with a vertex is homotopy equivalent to $\mathcal{C}$ [4]. Because homotopy equivalence is transitive, we may also replace $\mathcal{S}$ by any other contractible complex.

2.1 Edge Flip

An important operator for 2Δ-complexes is the *edge flip*, which modifies a complex by replacing two adjacent faces. Let face f be attached to edges s, r and q, spanning vertices a, d and c. Similarly, face g is attached to edges r, t and p spanning vertices a, c and b. This is shown in Figure 3(a). We flip the edge r by first removing f and g, then removing r, then reattaching r' to b and d, then reattaching faces f' and g' to edges r', p, q and r', s, t, as shown in Figure 3(b). The flip *consumes* f and *produces* f'.

In a 2D simplicial complex, a flip must not be performed if the vertices to be connected by the new edge are already connected. Doing so collapses the space because every simplex is uniquely determined by its vertices. This constraint is overly restrictive because a Δ-complex admits multiple edges between the same pair of vertices. Let the *apex* of a triangular face with respect to an edge e be the vertex of that face which is not incident on e. In a manifold 2Δ-complex, we allow an edge e to flip if the apexes of its two incident faces, f and g, are not the same vertex. If this is the case, then the closure of $\{f, g\}$ is contractible and the complex resulting from the flip is homotopy equivalent to the original. If the two apexes of f, g are the same, then flipping e creates a loop. While not immediately changing the topology of the complex, loops complicate further operations such as edge contractions, so we avoid them altogether. An example of a non-flippable edge is shown in Figure 3(c), labeled r.

2.2 Edge Contraction

An *edge contraction* modifies a 3Δ-complex by merging two adjacent vertices. We decompose the edge contraction into three operations that contract a 1-cell, some 2-cells and some 3-cells. Collapsing the initial edge removes the edge e and one vertex. The resulting complex is not a Δ-complex: every face that was incident on e is now a 2-cell incident on only two edges. To restore the

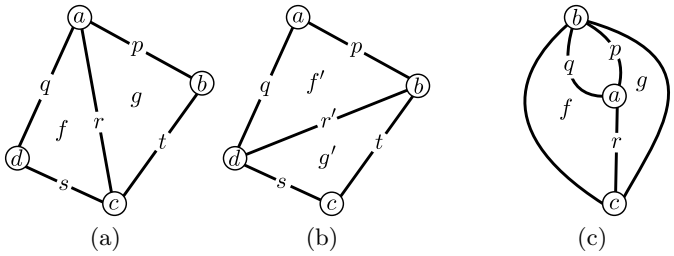


Fig. 3. (a) and (b) show the process of flipping edge r. In (c), edge r is not flippable. The apexes of f and g with respect to r are both b, so flipping r would create a loop.

Δ-complex property we contract these 2-cells, but this still does not create a Δ-complex: every tetrahedron that was incident on e is now a 3-cell incident on only two faces, resembling a triangular "pillow." We finally contract these 3-cells to restore the Δ-complex property. This process is shown in Figure 4.

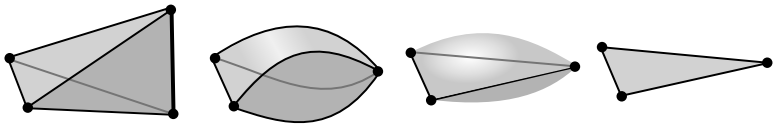


Fig. 4. An edge contraction.

This operation is decidedly different from the edge contraction operation for simplicial complexes. For that operation, the condition to preserve the topology of the complex is much stricter [2]. In Δ-complexes the condition is more relaxed: As with edge flipping, we forbid edge contractions that create loops. If vertices u and v are both incident on the same pair of edges, then neither of those edges may be contracted. As a consequence of condition 3 in the definition of a Δ-complex, the closure of every cell is a contractible subcomplex, and thus the intermediate complexes during the edge contraction process are all homotopy equivalent. Avoiding the creation of loops suffices to preserve condition 3. To see that the result is a Δ-complex, note that every tetrahedron incident on the contracted edge is turned into a valid face, and every incident d-cell, $d < 3$, is removed. This holds only as long as $\mathcal{C}'$ has enough vertices to satisfy the definition of a Δ-complex.

3 Algorithm

The problem is defined as follows: Let K_1 and K_2 be two 2-sphere Δ-complexes, and let P be a prescribed matching between their vertices, such that each vertex is matched with at most one other vertex, and that vertex is in the other complex. These complexes are the duals of cross-sections of a segmentation, and their vertices represent the regions or "phases" of that segmentation. The output is

a 3-sphere Δ-complex $\mathcal{C}$ that contains subcomplexes $\mathcal{C}_1$ and $\mathcal{C}_2$ isomorphic to K_1 and K_2, respectively. Additionally, for vertices u and v, $u \in K_1$, $v \in K_2$, if $(u, v) \in P$, then the isomorphisms $M_i : K_i \to \mathcal{C}$ map u and v to the same $v' \in \mathcal{C}$. In other words, $\mathcal{C}$ connects K_1 to K_2 and unifies matched vertices.

K_1 and K_2 are first combined to create a single Δ-complex K. To do so, nest K_1 inside K_2, then find two faces, f_1 and f_2, from K_1 and K_2, respectively, and connect them with a ball 3Δ-complex containing f_1 and f_2 on its boundary, like a triangular prism. Call this prism F. If possible, F should be chosen so that its three pairs of vertices each match under P. Next, two new vertices are created, v_1 and v_2, and each v_i is connected to K_i by a cone V_i, which is a ball 3Δ-complex containing v_i as a vertex and K_i as its boundary. The cone V_i lies on the side of K_i *opposite* F. The result, $V_1 \cup V_2 \cup F = B$ is a 3-ball. The boundary of this ball, A, is a 2-sphere formed by faces of K_i and F. Figure 5 shows a cut-away diagram of this construction.

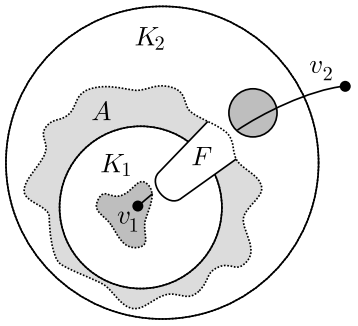
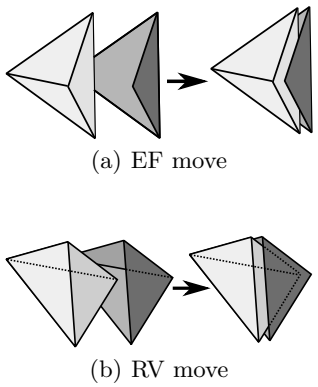


Fig. 5. The initial Δ-complex. Nested spheres K_1 and K_2 are connected by a solid prism F. Vertex v_1 is connected to the interior of K_1 by solid cone, and likewise v_2 is connected to the exterior of K_2. The union of these cones and F forms a ball, and the complement of this is a sphere bounded by the surface A.



(a) EF move

(b) RV move

Fig. 6. The two boundary operations used to modify the active surface. The EF move effectively flips an edge, and the RV move removes a vertex.

The algorithm fills in the space bounded by A, turning B into a 3-sphere. This is accomplished by attaching tetrahedra to A in one of two ways. In each case, B is extended by one additional tetrahedron, and A is updated to track the boundary of B. A loses vertices in the process, until it eventually reduces to one of two 4-vertex configurations, at which point the algorithm terminates. This is ensured because A remains a 2-sphere throughout the process. All operators take place on the surface A so one can think of A as the "active surface." The two operations on A are the following: An *EF* operation adds a tetrahedron c to B by attaching it to two adjacent faces in A. After updating A to track the new boundary, we see that the edge between these faces is flipped. An *RV* operation

attaches c to three mutually adjacent faces of A, effectively removing a vertex from the boundary of B. These operations are illustrated in Figure 6.

Edge flips, vertex removals and edge contractions are applied in a goal-directed way. We call a vertex who has no matching a "loner," and likewise, two matched vertices "mates." The goal is to remove every loner and contract an edge between every mated pair. To remove a loner w, we perform EF moves until the valence of w is three, then remove w by an RV move. If u and v are a mated pair then we use EF moves to flip all edges along a shortest path from u to v through the faces of A. Doing so is always possible, and causes u and v to become adjacent via a shared edge which is subsequently contracted, producing a new loner vertex.

Let a *face path* of a 2Δ-complex be a path between vertices u and v consisting of a sequence of faces f_i, $0 \leq i \leq n$, such that u is incident on f_0 and v is incident on f_n, and each f_i is adjacent to f_{i+1}. A face path is *shortest* if no other face path between u and v has fewer faces. When we refer to "the edges of a face path" we mean a sequence of edges e_i such that f_i and f_{i+1} are both incident on e_i, i.e., the edges one crosses when walking along the path. By the *length* of a face path we mean the number of edges, which is one less than the number of faces. In the proofs below, all complexes are assumed to be 2-spheres without loops.

Lemma 1. *The first edge of a shortest face path between distinct, non-adjacent vertices u and v is not incident on u.*

Proof. Let u, w_1, w_2 be the vertices of f_0. If $e_0 = uw_1$, then u is also incident on f_1 and the path is not shortest. By the same logic, $e_0 \neq uw_2$, leaving only $e_0 = w_1w_2$. □

Lemma 2. *The first edge of a shortest face path between distinct, non-adjacent vertices is always flippable.*

Proof. By Lemma 1, u is incident on f_0 and not incident on e_0, so u is the apex of f_0 with respect to e_0. If e_0 is not flippable, then u is also the apex of f_1. If u is then incident on both faces, and no shortest face path from u to any other vertex passes through e_0. □

We call a face path *flippable* if the first edge is flippable, and after flipping it the remaining path is flippable or empty. The previous lemma implies that shortest face paths are flippable.

Lemma 3. *Let e_i, $0 \leq i \leq n$, $n \geq 1$, be the edges of a flippable face path between distinct, non-adjacent vertices u and v. Flipping e_i in order of increasing i results in a Δ-complex in which u and v are adjacent.*

Proof. Let $n = 1$, then flipping e_0 immediately connects u and v. Now let e_i, $0 \leq i \leq k$ be the edges of a face path between x and u of length k. Assume that by flipping each e_i in sequence, a face s is created with u and x incident on s. Let w be the apex of s with respect to edge ux, let t be the other face on edge

ux, and let y be the apex of t with respect to ux. (By assumption $y \neq u$.) After flipping edge ux, y is made adjacent to u. The length of the face path between y and u was $k+1$, so the claim follows from induction on k. □

Lemma 4. *Any sequence of l flips, transforming 2Δ-complex $\mathcal{C}_0$ to $\mathcal{C}_l$, and causing vertices u, v not adjacent in $\mathcal{C}_0$ to become adjacent in $\mathcal{C}_l$, defines a face path between u, v in $\mathcal{C}_0$ of length at most l.*

Proof. Let S_i be a sequence of sets of faces, where S_l contains a face of C_l incident on u and v. Construct S_{i-1} from S_i as follows: Remove from S_i the faces produced by flip i. There are three cases where zero, one or two faces are removed. If one face r is removed, add to S_{i-1} the two faces p, q that are consumed by flip i. If S_i was a connected face path, then so is S_{i-1} because any face s adjacent to r in C_i is adjacent to one of p or q (which are themselves adjacent) in C_{i-1}, or s is consumed by the flip. If two faces are removed, S_i remains connected for the same reason. Thus, if S_l contains a single face, then all S_i are connected face paths. Vertices u and v are each incident on faces of S_i for all i, and therefore the path in S_0 connects them. At each iteration, the cardinality of S is increased by at most 1, so the cardinality of S_0 is at most $l+1$. □

Corollary 1. *If the shortest face path between u and v has length l, then no sequence of fewer than l flips can make u and v adjacent.*

The previous lemma and corollary allow us to guarantee that no mated pair of vertices become inadvertently connected by multiple edges, so that when the time comes to merge this pair the edge contraction is always possible. We define the cost of removing a vertex to be the number of EF moves required to remove it from the active surface. When removing a loner v the cost is $|3 - \text{valence } v|$, because the valence of v must first become three using EF moves before v can be removed with an RV move. When merging a pair of mated vertices u and v, the cost is the length of the face path between u and v.

Lemma 5. *If every mated pair of vertices is either non-adjacent or adjacent by one edge before an iteration, and the iteration performs the cheapest vertex removal, then the same is true after an iteration.*

Proof. Let c be the number of edge flips needed to remove the vertex. No face path of length $c-1$ or less exists between any pair of mated vertices, and thus by Corollary 1, performing $c-1$ flips does not cause any mated pair to become adjacent. After performing $c-1$ of the c flips needed to remove the vertex, it is still true that no other mated pair is adjacent. Flipping a single edge can only add one edge between any pair of non-adjacent vertices, implying that after the final flip every mated pair is still connected by at most one edge. □

If A starts with n vertices, then we perform exactly $n-4$ iterations. In each iteration we perform the cheapest move sequence to merge a mated pair or remove a loner. After $n-4$ iterations, what results is a Δ-complex in one of two configurations: either a tetrahedron or the complex shown in Figure 2, which is a tetrahedron that has had one edge flipped.

Lemma 6. *Any sphere 2Δ-complex with four vertices is isomorphic to either the boundary of a tetrahedron or the complex shown in Figure 2.*

Proof. If each edge is attached to a different pair of vertices, then the edges form a complete graph. The other possibility is that two edges, e and f, are attached to the same pair of vertices, u and v. Because the complex is a 2-sphere, the path from u along e to v and back along f to u separates the complex into two balls which must each contain one of the remaining vertices. □

The algorithm is summarized as follows:

1. Connect the two input spheres with a prism, and fill in the interior of each sphere with a cone. Initialize A to the boundary of the space between the spheres.
2. If A has four vertices, stop. If A is isomorphic to the complex in Figure 2, apply an EF operation to transform it into a tetrahedron.
3. Determine the cheapest sequence of EF and RV moves that either removes a loner or connects a mated pair.
   (a) If the cheapest operation is to remove a loner, then perform EF moves until valence is three, then perform an RV move.
   (b) If the cheapest operation is to connected a mated pair of vertices, then find a shortest face path between them. Flip those edges in sequence, then contract the resulting edge between the pair. Mark the newly merged vertex as a loner.
4. Update A and goto 2.

Main Theorem. The 3-sphere Δ-complex constructed by this algorithm contains subcomplexes isomorphic to K_1 and K_2, and this isomorphism identifies vertices of K_1 and K_2 that match under P.

Proof. After first constructing A, $\mathcal{C}$ already contains K_1 and K_2 as subcomplexes. No edge between two vertices of K_1 or two vertices of K_2 is ever contracted, nor any such face removed, so the output complex still contains these subcomplexes. Every matched pair of vertices from K_1 and K_2 is connected by flipping the edges of the shortest face path between them (Lemma 3.) Doing so does not create loops (Lemma 2) nor does it create multiple connections between any mated pairs (Lemma 5) and so the edge between the pair can always be contracted. Every iteration of the algorithm removes one vertex from A while maintaining that A is a 2-sphere Δ-complex, so when the algorithm terminates (Lemma 6) the complex is a 3-sphere.

4 Discussion

An initial 2-sphere can be constructed from a segmented planar image by adding an additional region containing everything "outside" the image. The algorithm can be extended to multiple input cross-sections by turning each one into a 2-sphere and sequentially nesting these spheres like the layers of an onion, then

performing the algorithm on the cavity between each pair of consecutive layers. After the algorithm finishes, the cells representing the "outside" are removed, leaving a 3-ball to be embedded in $\mathbb{R}^3$. Most of this embedding is prescribed by the boundary curves of the cross-sections, but the vertices created by the algorithm (dual to tetrahedra of the Δ-complex) need to be placed. We place them by iteratively moving them toward the average position of nearby vertices. Placing them without self-intersections is a challenge.

A straightforward implementation of the algorithm has a run-time complexity bounded by $O(n^3)$ in the worst case, where n is the number of vertices in the initial complex. There are exactly $n-4$ iterations, each iteration having to determine the cheapest move. The cost of mating a pair is the length of the shortest face path between vertices, which can be determined by an $O(n)$ breadth-first search. There can be as many as $O(n)$ mated pairs, thus an upper bound on the running time of $O(n^3)$. In practice, the lengths of shortest face paths are much less than $O(n)$. There is usually a path of nearly constant length close to the interface between K_1 and K_2 in A. Further, if we have already found a shortest path of length l, we can cut future searches short. These two facts make the practical running time nearly linear for well-behaved inputs.

5 Results

To evaluate the 3D models produced by the algorithm, we have reconstructed two types of synthetic ideal cellular structures. Figure 7 shows the reconstruction of a polycrystal microstructure that has been simulated using the Potts model [5]. The algorithm was run on three different sets of cross-sections of the simulation lattice. One containing every plane of lattice sites, one containing every third plane, and one containing every fifth plane. The cross-sections were extracted from the lattice using marching-triangles, and subsequently simplified using an area-preserving polyline simplification method.

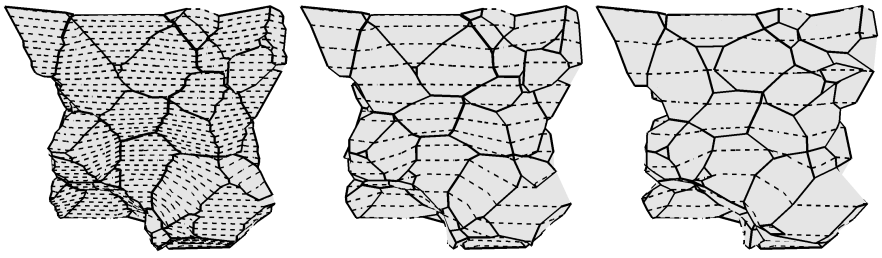


Fig. 7. Three reconstructions of a simulated microstructure on 42^3 cubic lattice. Some exterior grains have been removed for illustration. The solid lines are reconstructed grain edges and the dashed lines are input cross-sections. From left to right the distance between cross-sections is 1, 3 and 5 voxels.

Figure 8 shows the reconstruction of a group of cells called *truncated octahedra*. Each cell is bounded 14 faces, six squares and eight hexagons. Figure 8(a) shows the initial output of the algorithm, in which the reconstructed faces are not all squares and hexagons, a consequence of ambiguity. However this is a good starting point for further refinement, as the faces contain multiple polylines from adjacent cross-sections, which allows the estimation of a plane fitting the face, or a local region of it. Using this information we have modified the reconstructed complex to achieve the correct cell structure, shown in Figure 8(b).

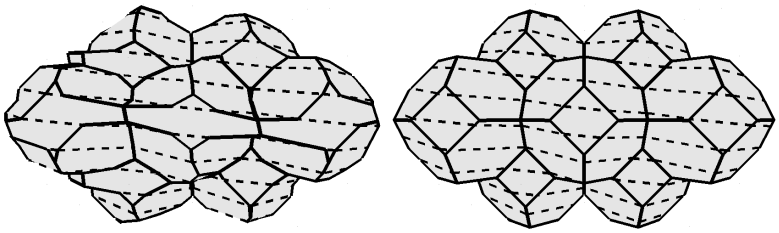


Fig. 8. Two reconstructions of a group of truncated octahedra. Reconstructed edges of the complex are drawn with solid lines and the input cross-sections are drawn with dashed lines. The left image shows the output of the algorithm as presented. The right image shows the complex after local topological modifications guided by the cell face geometry.

This modification process is the subject of ongoing research, so we only sketch the method here. Notice in Figure 8(b) that not every cell edge intersects a cross-section. We group such ambiguous edges into clusters. In the dual, this edge cluster is a cluster of Δ-tetrahedra bounded by non-ambiguous Δ-faces. The interior of the cluster is deleted and then re-tessellated using the following heuristic: Non-ambiguous Δ-faces are associated with lines defined by the triple points from consecutive cross-sections, and during re-tessellation we favor creating Δ-tetrahedra whose faces are associated with lines that come closest to intersecting. Not every edge of the Δ^*-complex intersects enough cross-sections to define a line, but if there are a sufficient number of non-ambiguous edges and faces then their gemoetry can be used to resolve topological ambiguities.

Figure 8(b) was created by placing the unconstrained vertices—those not lying in a cross-section—to minimize the sum of squared distances to the planes of incident faces. This strategy produces accurate reconstructions if ambiguities are correctly resolved, but can cause severe self-intersections if they are not.

Figure 9 shows the reconstruction of a sample of tantalum. The sample was subjected to impact and consequently it exhibits deformed polycrystal grains and a number of small and large voids. There are 1976 cells in total. The running time of the main portion of the algorithm was approximately 30 seconds on a computer with a 2.1 GHz processor. The cross-sections were observed using an EBSD microscope, at a spacing of $25\mu m$. A larger portion of this same dataset

has been previously reconstructed using a voxel-based method [3]. The cross-sections in Figure 9 are sparse, and the planar boundary curves are complex, so there are some self-intersections.

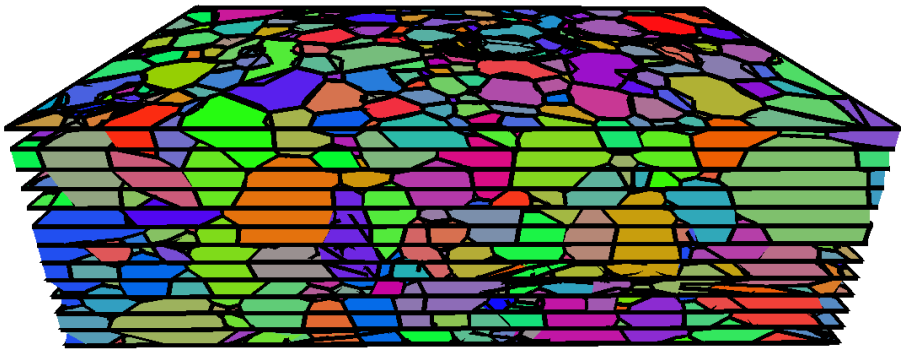


Fig. 9. A portion of a reconstruction of 13 cross-sections of a sample of shocked tantalum. The input cross-sections are indicated by horizontal lines.

Conclusion

We have presented an algorithm for reconstructing a 3D cell complex from a series of 2D cross-sections. Given any matching of regions between cross-sections, the algorithm produces a cell complex that connects matched regions. Self-intersections remain an outstanding issue. The constructed complex is guaranteed to have simple topology, but embedding it in $\mathbb{R}^3$ may not be straightforward. Indeed since any matching between cross-sections is permitted, it is quite easy to produce intractably tangled cell complexes. The embedding the reconstructed complex in $\mathbb{R}^3$ remains an area of future research.

References

1. I. Braude, J. Marker, K. Museth, J. Nissanov, and D. Breen. Contour-based surface reconstruction using MPU implicit models. *Graphical Models*, 69(2):139–157, 2007.
2. T. Dey, H. Edelsbrunner, S. Guha, and D. Nekhayev. Topology preserving edge contraction. *Publ. Inst. Math.(Beograd)(NS)*, 66(80):23–45, 1999.
3. S. Dillard, J. Bingert, D. Thoma, and B. Hamann. Construction of Simplified Boundary Surfaces from Serial-sectioned Metal Micrographs. *IEEE Transactions on Visualization and Computer Graphics*, pages 1528–1535, 2007.
4. A. Hatcher. *Algebraic Topology*. Cambridge University Press, 2002.
5. E. Holm and C. Battaile. The computer simulation of microstructural evolution. *JOM*, 53(9):20–23, 2001.
6. L. Nonato, A. Cuadros-Vargas, R. Minghim, and M. De Oliveira. Beta-connection: Generating a family of models from planar cross sections. *ACM Transactions on Graphics (TOG)*, 24(4):1239–1258, 2005.