Published in final edited form as:
Int J Comput Geom Appl. 2007 ; 17(3): 261–296. doi:10.1142/S0218195907002331.

THE LAYERED NET SURFACE PROBLEMS IN DISCRETE GEOMETRY AND MEDICAL IMAGE SEGMENTATION*

XIAODONG WU[†],
Dept. of Electrical and Computer Engineering, Dept. of Radiation Oncology, University of Iowa, Iowa City, Iowa 52242, USA, xiaodong-wu@uiowa.edu

DANNY Z. CHEN[‡],
Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN 46556, USA, chen@cse.nd.edu

KANG LI[§], and
Dept. of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA, kangl@cmu.edu

MILAN SONKA[¶]
Dept. of Electrical and Computer Engineering, University of Iowa, Iowa City, IA 52242-1595, USA, sonka@engineering.uiowa.edu

Abstract

Efficient detection of multiple inter-related surfaces representing the boundaries of objects of interest in d-D images ($d \geq 3$) is important and remains challenging in many medical image analysis applications. In this paper, we study several *layered net surface (LNS)* problems captured by an interesting type of geometric graphs called *ordered multi-column graphs* in the d-D discrete space ($d \geq 3$ is any constant integer). The LNS problems model the simultaneous detection of multiple mutually related surfaces in three or higher dimensional medical images. Although we prove that the d-D LNS problem ($d \geq 3$) on a general ordered multi-column graph is NP-hard, the (special) ordered multi-column graphs that model medical image segmentation have the self-closure structures and thus admit polynomial time exact algorithms for solving the LNS problems. Our techniques also solve the related *net surface volume (NSV)* problems of computing well-shaped geometric regions of an optimal total volume in a d-D weighted voxel grid. The NSV problems find applications in medical image segmentation and data mining. Our techniques yield the first polynomial time exact algorithms for several high dimensional medical image segmentation problems. Experiments and comparisons based on real medical data showed that our LNS algorithms and software are computationally efficient and produce highly accurate and consistent segmentation results.

[*] This research was supported in part by an NIH-NIBIB research grant R01-EB004640. A preliminary version of this paper was presented at the 16th Annual International Symposium on Algorithms and Computation (ISAAC), 2005.

Communicated by Li Zhang, Guest Editor

[†] The research of this author was supported in part by a faculty start-up fund from the University of Iowa, and in part by a seed grant award from the American Cancer Society through an Institutional Research Grant to the Holden Comprehensive Cancer Center, the University of Iowa, Iowa City, Iowa, USA. Part of the work of this author was done in the Dept. of Computer Science, the University of Texas – Pan American, Edinburg, TX, USA.

[‡] The research of this author was supported in part by the National Science Foundation under Grants CCR-9988468 and CCF-0515203.

[§] Part of the work of this author was done in the Dept. of Electrical and Computer Engineering, University of Iowa, Iowa City, IA.

[¶] This research was supported in part by NIH-NHLBI research grants R01-HL063373 and R01-HL071809.

Keywords

Surface detection; ordered multi-column graphs; minimum-cost closed sets; geometric optimization; medical image segmentation

1. Introduction

In this paper, we study the *layered net surface* (LNS) problems and their extensions in discrete geometry in the d-D space ($d \geq 3$ is any constant integer). These problems arise in d-D medical image segmentation and other applications.

Image segmentation, a central problem in medical image analysis, aims to define accurate boundaries for the objects of interest captured by image data. 3-D image segmentation for improved medical diagnosis promises to revolutionize the current medical imaging practice which is inherently 2-D. Although intensive research has been done on 2-D image segmentation in several decades, efficient and effective 3-D image segmentation still poses one of the major challenges in image understanding. In common practice, to identify surfaces representing the boundaries of the sought 3-D objects, first 2-D image slices are more or less analyzed independently; then the 2-D results are stacked together to form the 3-D segmentation output. This approach has some inherent limitations — the most fundamental one stems from the lack of contextual slice-to-slice information when analyzing a sequence of consecutive 2-D images. Performing the segmentation directly on a 3-D image can produce a more consistent segmentation result, yielding 3-D surfaces for object boundaries instead of a set of individual 2-D contours. However, most 2-D image segmentation approaches are difficult to extend to a direct 3-D segmentation, not to mention higher dimensional cases (e.g., a time-series of 3-D images).

We present novel techniques for the LNS problems, i.e., for a simultaneous segmentation of multiple inter-related surfaces in three and higher dimensional medical images. This is an important step towards achieving a practical 3-D surface detection tool since many surfaces in medical images appear in mutual relations. A number of medical imaging problems can benefit from an efficient method for simultaneous detection of multiple inter-related 3-D surfaces.[11,18,23,24,25,35,19]

We use vascular MR images of femoral arteries to exemplify the segmentation of multiple inter-related surfaces and the underlying constraints. Figure 1(a) is a schematic cross-sectional anatomy of a diseased artery, and Figure 1(b) shows a 2-D cross-sectional vascular MR image of a human femoral artery specimen. A 3-D MR artery image consists of a sequence of such 2-D cross-sections. To detect the vascular layer structures shown in Figure 1(a), we first perform a polar resampling in each 2-D slice along the centerline of the vessel (see Figure 1 (b)), and then embed the resampled 2-D geometric space into a 3-D grid $\mathscr{G}(\mathbf{x}, \mathbf{y}, \mathbf{z})$, as in Figure 1(c). This process is called “unfolding”.[22] Each of the sought surfaces in $\mathscr{G}(\mathbf{x}, \mathbf{y}, \mathbf{z})$ for the object boundaries (e.g., lumen, internal and external elastic lamina, etc) then contains exactly one voxel in every column of $\mathscr{G}(\mathbf{x}, \mathbf{y}, \mathbf{z})$ that is parallel to the $\mathbf{z}$-axis. Finally, the “unfolded” surfaces in $\mathscr{G}(\mathbf{x}, \mathbf{y}, \mathbf{z})$ are segmented. Some geometric constraints on these surfaces should be satisfied by the segmentation. In this example, pairs of the sought surfaces must be non-crossing and within a given range of distances apart. Since many anatomical structures are smooth, the segmented surfaces must be sufficiently “smooth”. Generally speaking, the smoothness is related to the surface curvature and means that an object boundary cannot change abruptly.

The simultaneous detection of multiple inter-related surfaces has been studied by the medical image analysis community for a long time. For the 2-D case, there are several satisfactory results.[23,22,25,2,29] However, little work has been done on the three and higher dimensional cases. Previous attempts[26,27,8] on extending graph-search based segmentation methods for the 2-D case to identifying even a single optimal surface in 3-D medical images either made the methods computationally intractable or traded their ability to achieve global optima for computational efficiency. Motivated by this segmentation problem, Wu and Chen[30] introduced the optimal net surface problems and presented efficient polynomial time exact algorithms for them. But, the algorithms in the paper[30] can detect only one optimal surface in 3-D. An implementation of their algorithms and experimental validation based on real 3-D medical images were presented in the paper.[15] More recently, Li *et al*.16 extended the approach[15] to segmenting multiple inter-related surfaces in 3-D. However, their new method does not incorporate the very important region information (e.g., homogeneity) for the surface detection.

Modeling the simultaneous detection of multiple inter-related surfaces in high dimensional medical images, we introduce the *layered net surface* (LNS) problems on an interesting type of geometric graphs, called *ordered multi-column graphs*, embedded in the d-D discrete space for $d \geq 3$ (to be defined in Section 2). We further extend the LNS problems to a more general ordered multi-column graph. Motivated by segmenting anatomical structures with a relatively regular geometric shape, such as the left ventricles, kidneys, livers, and lungs, we also study several *net surface volume* (NSV) problems, which aim to find well-shaped regions of an optimal "volume" in a d-D weighted voxel grid. These well-shaped geometric regions are closely related to monotonicity and convexity in d-D discrete spaces (Section 6). Our main results in this paper are summarized as follows.

- We develop an efficient algorithm for solving the LNS problem on an interesting type of ordered multi-column graphs in polynomial time, by formulating it as computing a minimum-cost closed set in a vertex-weighted directed graph.
- We prove that the LNS problem on a general ordered multi-column graph is NP-hard. However, the (special) ordered multi-column graphs that model medical image segmentation applications have additional properties, and the LNS problem on such graphs is polynomially solvable.
- We extend our LNS technique to solving the NSV problems of computing several classes of optimal well-shaped geometric regions in a d-D weighted voxel grid. These NSV problems arise in data mining,[9,7] image segmentation,[2] and data visualization.[1] The classes of regions that we study can be viewed as generalizations of some of the pyramid structures in the paper.[7]
- We apply our polynomial time LNS algorithms to segmenting multiple inter-related object boundaries in 3-D medical images. Our method can be easily extended to higher dimensional image segmentation. We also show implementation and experimental results of our algorithms.

Our LNS approach was inspired by Wu and Chen's algorithms[30] for solving the optimal net surface problems. However, instead of searching for a single optimal surface as in the paper,[30] our algorithms concurrently identify multiple optimal inter-related surfaces. Our approach is based on a characterization of the self-closure structures of the underlying graphs and a sophisticated graph transformation scheme, in order to enforce the constraints of the LNS problems and to ensure the optimality of the solutions. The extension of our LNS approach to the NSV problems is quite elegant. To compute a sought optimal d-D region, we exploit the monotonicity property of the target region and transform in a nontrivial fashion the NSV problem to the LNS problem.

The rest of the paper is organized as follows. In Section 2, we formally define the layered net surface (LNS) problems. The modeling of medical image segmentation as our optimal LNS problems are discussed in Section 3. Our algorithm for solving the LNS problem is presented in Section 4. We prove the NP-hardness of the LNS problem on a general ordered multi-column graph in Section 5 and show the problem on a special ordered multi-column graph is polynominal-time solvable. Section 6 gives extensions of our LNS techniques to solving the NSV problems. Finally, our implementation and experimental results are shown in Section 7.

2. The Layered Net Surface (LNS) Problems

A *multi-column graph* $G = (V, E)$ embedded in the d-D discrete space is defined as follows. For a given undirected graph $B = (V_B, E_B)$ embedded in $(d - 1)$-D (called the *net model*) and an integer $\kappa > 0$, G is an undirected graph in d-D *generated by* B and κ. For each vertex $v = (x_0, x_1, \ldots, x_{d-2}) \in V_B$, there is a sequence $Col(v)$ of κ vertices in G corresponding to v; $Col(v) = \{(x_0, x_1, \ldots, x_{d-2}, k) \mid k = 0, 1, \ldots, \kappa-1\}$, called the v-*column* of G. We denote the vertex $(x_0, x_1, \ldots, x_{d-2}, k)$ of $Col(v)$ by v_k. If an edge $(v, u) \in E_B$, then we say that the v-column and u-column in G are *adjacent* to each other. For each vertex $v_k \in Col(v)$, v_k has edges in G to a non-empty list of consecutive vertices in every adjacent u-column $Col(u)$ of $Col(v)$, say $u_{k'}$, $u_{k'+1}, \ldots, u_{k'+s}$ $(s \geq 0)$; we call $(u_{k'}, u_{k'+1}, \ldots, u_{k'+s})$, in this order, the *edge interval* of v_k on $Col(u)$, denoted by $I(v_k, u)$. For an edge interval I, we denote by $Bottom(I)$ (resp., $Top(I)$) the d-th coordinate of the first (resp., last) vertex in I (e.g., $Bottom(I(v_k, u)) = k'$ and $Top(I(v_k, u)) = k' + s$ in the above example).

Two adjacent columns $Col(v)$ and $Col(u)$ in G are said to be in *proper order* if for any two vertices v_k and v_{k+1} in $Col(v)$, $Bottom(I(vk, u)) \leq Bottom(I(v_{k+1}, u))$ and $Top(I(v_k, u)) \leq Top(I(v_{k+1}, u))$, and if the same holds for any two vertices u_k and u_{k+1} of $Col(u)$ on $Col(v)$. The corresponding edge $(v, u) \in E_B$ is called a *proper edge*. If all pairs of adjacent columns in G are in proper order, then we call G a *properly ordered* multi-column graph (briefly, a *properly ordered graph*). Figures 2(a)–2(b) show a net model and a properly ordered graph.

Note that in medical image segmentation, the boundaries of the target objects (e.g., organs) are often sufficiently “smooth”. The smoothness constraint on the sought surfaces is modeled by the proper ordering of the edges in a multi-column graph G, that is, the edges connecting each vertex v_k in G to every adjacent column $Col(u)$ of $Col(v)$ form a vertex interval on $Col(u)$, and such vertex intervals for any two adjacent columns of G are in proper order.

A *net surface* in G (also called a *net*) is a subgraph of G defined by a function $\mathcal{N}: V_B \rightarrow \{0,1, \ldots, \kappa - 1\}$, such that for every edge $(v, u) \in E_B$, $(v_{k'}, U_{k''})$, with $k' = \mathcal{N}(v)$ and $k'' = \mathcal{N}(u)$, is also an edge in E. For simplicity, we denote a net by its function $\mathcal{N}$. Intuitively, a net $\mathcal{N}$ in G is a special mapping of the $(d - 1)$-D net model B to the d-D space, such that $\mathcal{N}$ “intersects” each v-column of G at exactly one vertex and $\mathcal{N}$ preserves all topologies of B. $\mathcal{N}$ can be viewed as a functional “surface” of B in d-D defined on the $(d - 1)$-D space in which B is embedded.

Given two integers L and U, $0 < L < U$, two nets $\mathcal{N}_1$ and $\mathcal{N}_2$ of a properly ordered graph G are said to be (L, U)-*separate* if $L \leq \mathcal{N}_2(v) - \mathcal{N}_1(v) \leq U$ for every vertex $v \in V_B$. Roughly speaking, $\mathcal{N}_1$ and $\mathcal{N}_2$ do not cross each other and are within a specified range of distance to each other. Figure 2(c) shows two net surfaces. For a given set of $l - 1$ integer parameter pairs $\{(L_i, U_i) \mid 0 < L_i < U_i, 1 \leq i < l\}$, $l \geq 2$, we consider l net surfaces $\mathcal{NS} = \{\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_l\}$ in G such that $\mathcal{N}_{i+1}$ is “on top” of $\mathcal{N}_i$ (i.e., $\forall\ v \in V_B$, $\mathcal{N}_{i+1}(v) > \mathcal{N}_i(v)$), and $\mathcal{N}_i$ and $\mathcal{N}_{i+1}$ are (L_i, U_i)-separate ($i = 1,2, \ldots, l - 1$). Then, these l net surfaces partition the vertex set V of G into $l + 1$ disjoint subsets R_i, with $R_0 = \{v_k \mid v \in V_B, 0 \leq k \leq \mathcal{N}_1(v)\}$, $R_i; = \{v_k \mid v \in V_B, \mathcal{N}_i(v) < k \leq \mathcal{N}_{i+1}(v)\}$ for $i = 1,2, \ldots, l -1$, and $R_l = \{v_k \mid v \in V_B, \mathcal{N}_1(v) < k \leq \kappa\}$ (see Figure 2(d)).

Motivated by medical image segmentation,[22,5,6,36,19] we assign costs to every vertex of G as follows. Each vertex $V_k \in V$ has an *on-surface cost* $b(v_k)$, which is an arbitrary real value. For each region R_i $(i = 0,1, \ldots, l)$, every vertex $v_k \in R_i \subseteq V$ is assigned a real-valued *in-region cost*

$c_i(v_k)$. The on-surface cost of each vertex in G is inversely related to the likelihood that it may appear on a desired net surface, while the in-region costs $C_i(\cdot)$ ($i = 0,1, \ldots, l$) measure the inverse likelihood of a given vertex preserving the expected regional properties of the partition $\{R_0, R_1, \ldots, R_l\}$. Both the on-surface and in-region costs for image segmentation can be determined by using simple low-level image features.[22,36,33,19]

The **layered net surface (LNS)** problem seeks l net surfaces $\mathcal{NS} = \{\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_l\}$ in G such that the total cost $\alpha(\mathcal{NS})$ induced by the l net surfaces in $\mathcal{NS}$, with

$$\begin{aligned}\alpha(\mathcal{NS}) = & \sum_{i=1}^{l} b(\mathcal{N}_i) \\ & + \sum_{i=0}^{l} c_i(R_i) \\ & = \sum_{i=1}^{l} \sum_{u \in V(\mathcal{N}_i)} b(u) \\ & + \sum_{i=0}^{l} \sum_{u \in R_i} c_i(u),\end{aligned}$$

is minimized, where $V(H)$ denotes the vertex set of a graph H.

In fact, our algorithmic framework is general enough for the cases in which each vertex has *only* an on-surface cost, *only* in-region costs, or both. We illustrate our unified approach as an example for the case where each vertex has both the on-surface and in-region costs.

In this paper, we also consider the layered net surface problem on a more general ordered multi-column graph, defined as follows. Note that any two adjacent columns of a *properly* ordered multi-column graph are in proper order. We now define the *reverse order* on two adjacent columns $Col(v)$ and $Col(u)$ in a d-D multi-column graph $G = (V, E)$ generated by a $(d - 1)$-D net model $B = (V_B, E_B)$: If for any two vertices V_k and V_{k+1} in $Col(v)$, $Bottom(I(v_k, u)) \geq Bottom(I(v_{k+1}, u))$ and $Top(I(v_k, u)) \geq Top(I(v_{k+1}, u))$, and if the same holds for any two vertices u_k and u_{k+1} of $Col(u)$ on $Col(v)$, then we say that $Col(u)$ and $Col(v)$ are in *reverse order*. The corresponding edge $(v, u) \in E_B$ is called a *reverse edge*. If every two adjacent columns in G are in either proper order or reverse order, then we call G a d-D *ordered* multi-column graph. Further, for two (L, U)-separate nets $\mathcal{N}_1$ and $\mathcal{N}_2$ in G, if two adjacent columns $Col(v)$ and $Col(u)$ are in reverse order, then $L \leq \mathcal{N}_1(v) - \mathcal{N}_2(v) \leq U$ and $L \leq \mathcal{N}_2(u) - \mathcal{N}_1(u) \leq U$. We prove that the LNS problem on a general d-D ordered multi-column graph ($d \geq 3$) is NP-hard (see Section 5).

Interestingly, if the ordered multi-column graph is generated by a special net model, which also models key medical image segmentation problems(see Section 3), the LNS problem is polynomially solvable. The special net model $B = (V_B, E_B)$ is defined in the following way. First, remove from B all reverse edges; the remaining B is a set $\mathcal{CC}$ of connected components with proper edges only. Then, contract each connected component of $\mathcal{CC}$ into a single vertex. Finally, for each (removed) reverse edge $(v, u) \in E_B$, say, v in $C' \in \mathcal{CC}$ and u in $C'' \in \mathcal{CC}$ ($C' = C''$ is possible), add an edge between the contracted vertices of C' and C''. The resulting graph is called the *p-contracted graph* of B. The **bipartite LNS** (BLNS) problem is defined on a d-D ordered multi-column graph with a net model B whose p-contracted graph is *bipartite*. For the BLNS problem, we naturally assume that each vertex in G has only an on-surface cost.

3. Modeling Medical Image Segmentation

This section discusses the modeling of medical image segmentation problems. Numerous medical applications can benefit from simultaneous detection of multiple optimal interacting surfaces representing terrain-like or tubular structures, which can be modeled as our optimal LNS problems. If we also consider the motion of these anatomical structures, then we have a time-series of 3-D images, forming a 4-D image.

It is easy to embed a 3-D image with a terrain-like object into the 3-D space. Let $\mathcal{I}(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \{(x, y, z) \mid 0 \le x < X, 0 \le y < Y, 0 \le z < Z\}$ be a 3-D image of size $X \times Y \times Z$, where X, Y, and Z denote the image size in the **x**, **y**, and **z** dimensions, respectively. Each desired terrain-like boundary surface contains exactly one voxel $\mathcal{I}(x, y, z)$ in each *column* of $\mathcal{I}(\mathbf{x}, \mathbf{y}, \mathbf{z})$ that is parallel to the **z**-axis. Since many anatomical structures are smooth, one may expect the resulting surfaces to be sufficiently “smooth”. The smoothness constraints guarantee the surface continuity in 3-D. Specifically, we use two *smoothness parameters*, Δ_x and Δ_y, to specify the maximum allowed change in the **z**-coordinate of a feasible surface along each unit distance change in the **x** and **y** directions, respectively. Note that here we assume a 4-neighbor adjacency (the modeling described below can be easily extended to other adjacency settings). In addition, assume that we are looking for l boundary surfaces and any two adjacent desired surfaces $\mathcal{S}_i$ and $\mathcal{S}_{i+1}$ are within a certain range of distances apart (e.g., for all $0 \le x < X$ and $0 \le y < Y$, $\delta_i^l \le \mathcal{S}_{i+1}(x, y) - \mathcal{S}_i(x, y) \le \delta_i^u$). A cost $c(x, y, z)$ is assigned to each voxel $\mathcal{I}(x, y, z)$ such that the cost is inversely related to the likelihood that a desired surface would contain the voxel, which is usually determined by using simple low-level image features.[22] The multiple surface detection problem seeks l feasible surfaces in $\mathcal{I}$ whose total sum of voxel costs is minimized.

This (terrain-like) surface detection problem can be modeled as a layered net surface problem on a properly ordered graph $G = (V, E)$, as follows. The net model of G is a 2-D $X \times Y$ grid B. Each voxel $\mathcal{I}(x, y, z)$ corresponds to exactly one vertex $v(x, y, z)$ in G. Thus, any grid point (x, y) in B is associated with a *column* $Col(x, y)$ of Z vertices of G, i.e., $Col(x, y) = \{v(x, y, z)| z = 0,1, \ldots, Z - 1\}$. Then, for every vertex $v(x, y, z) \in V$ with $x < X - 1$ (resp., $y < Y - 1$), there is an edge to vertex $v(x + 1, y, z')$ (resp., $v(x, y + 1, z'))$ for any z' such that $0 \le z' < Z$ and $|z' - z| \le \Delta_x$ (resp., $|z' - z| \le \Delta_y$). These edges are used to enforce the smoothness constraint of the target surfaces along the x-dimension (resp., y-dimension). Each vertex $v(x, y, z)$ has an on-surface cost equal to the cost $c(x, y, z)$ of the corresponding voxel $\mathcal{I}(x, y, z)$. It is straightforward to show that G is a properly ordered graph defined on the net model B. We are then looking for l net surfaces $\{\mathcal{N}_1, \mathcal{N}_2 \ldots, \mathcal{N}_l\}$ in G such that $\mathcal{N}_i$, and $\mathcal{N}_{i+1}$ are (δ_i^l, δ_i^u)-separate for $i = 1,2, \ldots, l - 1$. Hence, this surface detection problem is modeled as a layered net surface problem on the properly ordered graph G. Note that in this application, we do not consider the region-based information for the segmentation though it is easy to be incorporated into G.

To segment a tubular structure in a 3-D image $\mathcal{P}$, we need to first “unfold” it, and then embed it into the 3-D **xyz**-space to obtain an image $\mathcal{I}(\mathbf{x}, \mathbf{y}, \mathbf{z})$. In common medical practice, there are two useful unfolding approaches: One is based on the transversal cross-sections (TCS) of the tubular object; the other is based on the sagittal cross-sections (SCS) (see Figure 3(a)).

In the TCS-based approach, as illustrated in Figure 1, each transversal cross-section is unfolded by performing a polar resampling with respect to the center point O, and is embedded into an **xz**-plane, called an **xz**-*slice* of $\mathcal{I}$. Hence, any two adjacent columns on a same **xz**-slice satisfy the smoothness constraints. Note that on each **xz**-slice, the last column is treated as being adjacent to the first column, thus satisfying the smoothness constraints as well. Additionally, the corresponding columns on any two adjacent **xz**-slices also need to meet the smoothness constraints in the **y**-direction. Next, by applying the same modeling to $\mathcal{I}(\mathbf{x},\mathbf{y},\mathbf{z})$ as for the terrain-like object, we transform this problem into that of computing multiple optimal interacting net surfaces in a properly ordered graph G, whose net model B is a 2-D grid with a wraparound in the **x**-dimension.

In the SCS-based method, each sagittal cross-section of the tubular structure, which consists of a sequence of profiles from all transversal cross-sections at the same angle, corresponds to a **yz**-slice of $\mathcal{I}$. Figure 3(b) shows a transversal cross-section of a **3-D** intravascular ultrasound vessel image, in which each angle defines a sagittal cross-section. The sagittal cross-section corresponding to angle θi in Figure 3(b) is illustrated in Figure 3(c) and is embedded as a **yz**-slice of $\mathcal{I}$ at $x = \theta i$ (see Figure 3(d)). Note that the centerline of the tubular object is on every sagittal cross-section, thus forming the so-called ***O-plane*** in $\mathcal{I}$. The O-plane splits a desired tubular surface in the original image $\mathcal{P}$ into two sub-surfaces, the upper sub-surface and lower sub-surface, as shown in Figure 3(d). To model this problem, we decompose the image $\mathcal{I}$ along the O-plane into two sub-images $\mathcal{I}_1$ and $\mathcal{I}_2$ (see Figure 3(e)), each of which is used for searching the upper sub-surfaces and the lower sub-surfaces, respectively. One may search for the upper sub-surfaces and the lower sub-surfaces in $\mathcal{I}_1$ and $\mathcal{I}_2$ separately, by using our algorithm for the LNS problem. The net models B_1 and B_2 for $\mathcal{I}_1$ and $\mathcal{I}_2$, respectively, are a 2-D grid with only proper edges. However, the upper (resp., lower) surface on the last sagittal cross-section (i.e., at $x = \theta_{X-1}$ in Figure 3(e)) needs to be smoothly connected with the lower (resp., upper) surface on the first sagittal cross-section (i.e., at $x = \theta_0$ in Figure 3(e)). To enforce this smoothness constraint, we introduce a reverse edge between the corresponding vertices on the last column of B_1 (resp., B_2) and the first column of B_2 (resp., B_1) (see Figure 3(f)), thus forming a net model B. Obviously, the p-contracted graph of B is bipartite. Hence, we model this segmentation problem as solving a BLNS problem.

4. Algorithm for the Layered Net Surface (LNS) Problem

This section gives our polynomial time algorithm for the layered net surface problem on a d-D properly ordered graph $G = (V, E)$. We first exploit the self-closure structure of the LNS problem, and then model it as a minimum-cost closed set problem based on a nontrivial graph transformation scheme.

A *closed set* $\mathcal{C}$ in a directed graph with arbitrary vertex costs $w(\cdot)$ is a subset of vertices such that all successors of any vertex in $\mathcal{C}$ are also contained in $\mathcal{C}$.[20,14] The *cost* of a closed set $\mathcal{C}$, denoted by $w(\mathcal{C})$, is the total cost of all vertices in $\mathcal{C}$. Note that a closed set can be empty (with a cost zero). The minimum-cost closed set problem seeks a closed set in the graph whose cost is minimized.

4.1. Overview of the LNS algorithm

Our LNS algorithm is based on a sophisticated graph transformation scheme, which enables us to simultaneously identify $l > 1$ optimal inter-related net surfaces as a whole by computing a minimum closed set in a weighted directed graph G' that we transform from G. The algorithm uses the following three main steps.

Step 1: Graph Construction—Build a vertex-weighted directed graph $G' = (V', E')$, which contains l vertex-disjoint subgraphs $G'_i = (V'_i, E'_i)$. Each subgraph G'_i is used to search for the i-th net surface in $\mathcal{I}$. The separation constraints of the net surfaces are enforced by introducing a subset of directed edges between any two adjacent subgraphs, G'_i and G'_{i+1} ($i = 1,2,\ldots, l-1$). The construction of the graph G (see Section 4.3) hinges on the self-closure structure exploited in Section 4.2.

Step 2: Computing a Minimum-cost Closed Set—Compute a minimum-cost non-empty closed set $\mathcal{C}$*in G', which can be done by formulating it as computing a minimum s-t cut in an edge-weighted directed graph transformed from G'.

Step 3: Net Surfaces Reconstruction—The set of l optimal net surfaces is reconstructed from the minimum-cost closed set $\mathcal{C}$* with each net surface being specified by $C^* \cap V_i'$.

4.2. The self-closure property of the LNS problem

Our algorithm for the LNS problem hinges on the following observations about the self-closure structure of any feasible LNS solution. Recall that in a set of l feasible net surfaces $\mathcal{NS}$ = {$\mathcal{N}_1$, $\mathcal{N}_2$, …, $\mathcal{N}_l$} in G, $\mathcal{N}_{i+1}$ is “on top” of $\mathcal{N}_i$, for each $i = 1,2,\ldots,l-1$.

For a vertex $v_k \in V$ (i.e., $v \in V_B$ and $0 \le k < \kappa$) and each adjacent column $Col(u)$ of $Col(v)$ (i.e., $(v,u) \in E_B$), the *lowest-neighbor* of v_k on $Col(u)$ is the vertex in $Col(u)$ with the smallest d-th coordinate that has an edge to v_k in G (i.e., the vertex in $Col(u)$ with the smallest d-th coordinate that can possibly appear together with v_k on a same feasible net surface in G). In Figure 4(a), the lowest-neighbor of v_2 (resp., u_2) on $Col(u)$ (resp., $Col(v)$) is u_2 (resp., v_1).

Given the surface separation constraints, we define below the *upstream* and *downstream* vertices of any vertex in G, to help characterize the spatial relations between feasible net surfaces in G. For every vertex $v_k \in V$ and $1 \le i < l$ (resp., $1 < i \le l$), the *i-th upstream* (resp., *downstream*) vertex of v_k is v_{k+L_i} (resp., $v_{\max\{0,k-U_{i-1}\}}$) if $k + L_i < \kappa$ (resp., $k - L_{i-1} \ge 0$). Intuitively, if $v_k \in \mathcal{N}_i$, then the i-th upstream (resp., downstream) vertex of v_k is the vertex in $Col(v)$ with the smallest d-th coordinate that can be on $\mathcal{N}_{i+1}$ (resp., $\mathcal{N}_{i-1}$).

We say that a vertex v_k is *below* (resp., *above*) a net surface $\mathcal{N}_i$ if $\mathcal{N}_i(v) > k$ (resp., $\mathcal{N}_i(v) < k$), and denote by $LO(\mathcal{N}_i)$ the subset of all vertices of G that are on or below $\mathcal{N}_i$. For every vertex $v_k \in LO(\mathcal{N}_i)$, consider its lowest-neighbor $u_{k'}$ on any adjacent $Col(u)$ of $Col(v)$. Let $r = \mathcal{N}_i(v)$ and $u_{k''}$ be the lowest-neighbor of v_r on $Col(u)$ ($v_r \in Col(v)$ is on $\mathcal{N}_i$). Then by the definition of net surfaces, $k'' \le \mathcal{N}_i(u)$. Since $k \le \mathcal{N}_i(v)$, we have $k' \le k''$ due to the proper ordering. Thus, $k' \le \mathcal{N}_i(u)$, and further, $u_{k'} \in LO(\mathcal{N}_i)$. Hence, we have the following observation.

Observation 1—For any feasible net surface $\mathcal{N}_i$ in G, if a vertex v_k is in $LO(\mathcal{N}_i)$, then every lowest-neighbor of v_k is also in $LO(\mathcal{N}_i)$.

Observation 1 characterizes the self-closure property of every set $LO(\mathcal{N}_i)$. However, our task is more involved since the l net surfaces in $\mathcal{NS}$ are inter-related. We thus need to further examine the closure structure between the $LO(\mathcal{N}_i)$’^{S}. Consider any vertex $v_k \in LO(\mathcal{N}_i)$. Since $k \le \mathcal{N}_i(v)$, the i-th upstream vertex of $v_{\mathcal{N}_i(v)}$ is “above” the i-th upstream vertex of v_k (i.e., $\mathcal{N}_i(v) + L_i \ge k + L_i$). In the meanwhile, $\mathcal{N}_{i+1}(v) \ge \mathcal{N}_i(v) + L_i$ (by the definition of the i-th upstream vertex). Hence, the i-th upstream vertex of v_k, v_{k+L_i}, is in $LO(\mathcal{N}_{i+1})$. Using a similar argument, the i-th downstream vertex of v_k, $v_{\max\{0,k-U_{i-1}\}}$ is in $LO(\mathcal{N}_{i-1})$. Thus, the observation below follows.

Observation 2—Given any set $\mathcal{NS}$ = {$\mathcal{N}_1$, $\mathcal{N}_2$,… $\mathcal{N}_l$} of l feasible net surfaces in G, the i-th upstream (resp., downstream) vertex of each vertex in $LO(\mathcal{N}_i)$ is in $LO(\mathcal{N}_{i+1})$ (resp., $LO(\mathcal{N}_{i-1})$), for every $1 \le i < l$ (resp., $1 < i \le l$).

Observations 1 and 2 show an important self-closure structure of the LNS problem, which is crucial to our LNS algorithm and suggests a connection between our target problem and the minimum-cost closed set problem.[20,14] In our LNS approach, instead of directly searching for an optimal set of l net surfaces, $\mathcal{NS}^* = \left\{\mathcal{N}_1^*, \mathcal{N}_2^*, \ldots, \mathcal{N}_l^*\right\}$, we look for l optimal subsets of vertices in G, $LO(\mathcal{N}_1^*) \subset LO(\mathcal{N}_2^*) \subset \ldots \subset LO(\mathcal{N}_l^*)$, such that each $LO(\mathcal{N}_i^*)$ uniquely defines the net surface $\mathcal{N}_i^* \in \mathcal{NS}^*$.

4.3. The graph transformation scheme

This section presents the construction of the vertex-weighted directed graph $G' = (V', E')$ from the d-D properly ordered graph $G = (V, E)$, which enables us to simultaneously identify $l > 1$ optimal inter-related net surfaces as a whole by computing a minimum closed set. This construction crucially relies on the self-closure structure shown in Section 4.2.

The graph G' contains l vertex-disjoint subgraphs $\{G'_i=(V'_i, E'_i)|i=1, 2, \ldots, l\}$; each G'_i is for the search of the i-th net surface $\mathcal{N}_i$. $V'=\cup_{i=1}^{l} V'_i$ and $E'=\cup_{i=1}^{l} E'_i \cup E'_s$. The surface separation constraints between any two consecutive net surfaces $\mathcal{N}_i$ and $\mathcal{N}_{i+1}$ are enforced in G' by a subset of edges in E'_s, which connect the corresponding subgraphs G'_i and G'_{i+1}.

We first show the construction of every $G'_i=(V'_i, E'_i)$ $(i = 1,2, \ldots,l)$ from G. For G'_i, each vertex v_k in G corresponds to exactly one vertex $v_k^i \in V'_i$. Note that G'_i is used for the search of $LO(\mathcal{N}_i)$ that defines the i-th net surface $\mathcal{N}_i$. If vertex $v_k \in Col(v)$ appears on $\mathcal{N}_i$, then all vertices of $Col(v)$ "below" v_k (i.e., those vertices of $Col(v)$ whose d-th coordinate is $\leq k$) are in $LO(\mathcal{N}_i)$. Thus, for each $v \in V_B$ of the net model B and $k = 1,2,\ldots,\kappa - 1$, vertex v_k^i has a directed edge (v_k^i, v_{k-1}^i), forming a *chain* $v_{\kappa-1}^i \to v_{\kappa-2}^i \to \cdots \to v_0^i$ in G'_i for $Col(v)$, denoted by $Ch_i(v)$. These chains help capture the closure property of $LO(\mathcal{N}_i)$. Two chains $Ch_i(v)$ and $Ch_i(u)$ in G'_i are said to be *adjacent* if their corresponding columns $Col(v)$ and $Col(u)$ in G are adjacent. Next, we put directed edges between any two adjacent chains in G'_i to ensure the feasibility of the net surface $\mathcal{N}_i$. For every vertex v_k of each $Col(v)$ in G and its lowest-neighbor $u_{k'}$ on each adjacent $Col(u)$ of $Col(v)$, we put into E'_i a directed edge from $v_k^i \in Ch_i(v)$ to $u_{k'}^i \in Ch_i(u)$. The role of this edge is to ensure that if $v_k \in \mathcal{N}_i$, then $\mathcal{N}_i$ must contain a vertex in $Col(u)$ that is no "lower" than $u_{k'}$ (i.e., $\mathcal{N}_i(u) \geq k'$). This is because a feasible $\mathcal{N}_i$ containing v_k can only contain a vertex in the edge interval $I(v_k,u)$ of v_k on $Col(u)$, i.e., $I(v_k,u) = (u_{k'}, u_{k'+1}, \ldots, u_{k'+s})$ $(s \geq 0)$. It might appear that adding this edge to G'_i might not do the job. But, based on the proper ordering in G, we are able to prove that our graph transformation well enforces the feasibility of the net surfaces (see Lemmas 4 and 5). Figure 4(a) shows the part of the graph G associated with an edge $(v,u) \in E_B$, and Figure 4(b) illustrates the corresponding construction in G'_i. This completes the construction of each G'_i.

We now put directed edges into E'_s between G'_i and G'_{i+1}, to enforce the surface separation constraints. Based on Observation 2, if vertex $v_k \in \mathcal{N}_i$, then its i-th upstream vertex v_{k+L_i} must be on or below the net surface $\mathcal{N}_{i+1}$ (i.e., $v_{k+L_i} \in LO(\mathcal{N}_{i+1})$). Thus, for each vertex v_k^i with $k < \kappa - L_i$ on the chain $Ch_i(v)$ in G'_i, a directed edge is put in E'_s from v_k^i to $v_{k+L_i}^{i+1}$ on $Ch_{i+1}(v)$ in G'_{i+1} (see Figure 4(c)). Intuitively, these edges ensure that the net surface $\mathcal{N}_{i+1}$ must be at a distance of *at least* L_i "above" $\mathcal{N}_i$ (i.e., for each $v \in V_B$, $\mathcal{N}_{i+1}(v) - \mathcal{N}_i(v) \geq L_i$). On the other hand, each vertex v_k^{i+1} with $k \geq L_i$ on $Ch_{i+1}(v)$ has a directed edge in E'_s to $v_{k'}^i$ on $Ch_i(v)$ with $k' = \max\{0, k - U_\mathrm{i}\}$ (note that $v_{k'}$ in G is the $(i + 1)$-th downstream vertex of v_k), making sure that $\mathcal{N}_{i+1}$ must be at a distance of *no larger than* U_i "above" $\mathcal{N}_i$ (i.e., for each $v \in V_B$, $\mathcal{N}_{i+1}(v) - \mathcal{N}_i(v) \leq U_i$). Note that in this construction, any vertex v_k^i with $k \geq \kappa - L_i$ has no edge to any vertex on Ch_{i+1}(v), forming the *upper deficient vertex segment* $udv_i(v)$ of $Ch_i(v)$, and any vertex v_k^{i+1} with $k < L_i$ has no edge to any vertex on $Ch_i(v)$, forming the *lower deficient vertex segment* $ldv_{i+1}(v)$ of $Ch_{i+1}(v)$. Each of these vertices of G' is called a *deficient vertices*. This construction is applied to every pair of the corresponding chains of any two subgraphs G'_i and G'_{i+1}, for $i = 1, 2, \ldots, l - 1$.

Recall that we aim to compute a minimum-cost non-empty closed set in G', which can specify l optimal net surfaces in G. However, the graph G' constructed up to this point does not yet work for this purpose. When a closed set $\mathcal{C}$ in G' includes only deficient vertices of a chain in G', which is possible, it does not correspond to l feasible net surfaces. The reader may notice that for any deficient vertex in G', the corresponding vertex in G cannot lie on a feasible net surface. Hence, it might seem safe to remove all deficient vertices from G'. However, this removal of deficient vertices may in turn cause other vertices in G' to become deficient. (A vertex v_k^i on $Ch_i(v)$ is said to be *deficient* if either no vertex in G'_{i-1} or G'_{i+1} connects with it, or there exists an edge $(v, u) \in E_B$ but no vertex on $Ch_i(u)$ of G'_i is adjacent to it.) Moreover, we cannot simply delete all edges incident with the deficient vertices.

By utilizing the proper order of graph G, we develop a deficient vertex pruning scheme, as follows. Instead of removing the deficient vertices one by one, we grow the deficient vertex segments until no further changes can occur, and then safely remove all deficient vertex segments and their edges. We first consider the lower deficient vertex segment $ldv_i(v)$ of each chain $Ch_i(v)$ in G'. A key observation here is that when pruning the lower deficient vertex segments, if a vertex v_k^i on $Ch_i(v)$ is deficient, then all vertices $v_{k'}^i$ on $Ch_i(v)$ "below" v_k^i (i.e., with $k' \leq k$) are also deficient. Hence, the deficient vertices on every $Ch_i(v)$ always form a continuous segment and it is sufficient to consider the "topmost" deficient vertex v_t^i on $ldv_i(v)$ (i.e., $\forall v_k^i \in ldv_i(v), k \leq t$). For each column $Col(u)$ adjacent to $Col(v)$, recall that $I(v_t, u)$ denotes the edge interval of v_t on $Col(u)$ in G. If $I(v_t, u) \subseteq ldv_i(u)$, then the removal of $ldv_i(v)$ does not cause additional deficient vertices on $Ch_i(u)$. Otherwise, consider the edge interval $I(v_{t+1}, u)$ of v_{t+1} on $Col(u)$. Let q be the smallest d-th coordinate of the vertices in $I(v_{t+1}, u)$. Due to the proper order of G, no vertices v_k with $k > t$ have an edge connecting with $u_{q-1} \in Col(u)$. Thus, the vertex u_{q-1}^i is deficient since the edge $(v, u) \in E_B$ but the removal of all incident edges of the vertices in $ldv_i(v)$ leaves no edge connecting u_{q-1}^i with any vertex on $Ch_i(v)$ in G'_i. Let $u_{t'}^i$ be the topmost deficient vertex on $ldv_i(u)$. If $q - 1 > t'$, then the lower deficient vertex segment $ldv_i(u)$ now grows up to u_{q-1}^i. Note that for each vertex u_k^i with $k \geq q$ and $u_k \in I(v_t, u)$, if u_k is on a feasible surface, then the vertex of $Col(v)$ on that surface must be no lower than v_{t+1}; in fact, the edge (v_{t+1}, u_k) is in G. Thus, a directed edge from u_k^i to v_{t+1}^i is put in G'_i to enforce this relation. This pruning procedure is also applied to $Ch_i(v)$ with respect to $Ch_{i+1}(v)$ and $Ch_{i-1}(v)$. When no further changes are needed on the lower deficient vertex segments of G', we just remove all these segments and their incident edges. Actually, we can maintain a queue to keep the chains of G' with a changed lower deficient vertex segment. When eventually the queue is empty, it means that no lower deficient vertex segments will be further changed. The upper deficient vertex segments in G' are handled in a similar way. It is not hard to see that the following lemma holds.

Lemma 1—The deficient vertex pruning scheme runs in O(|E′|) time on G′ = (V′, E′).

We simply denote the graph thus resulted also by G'. Note that in G' if any chain $Ch_i(v) = \varnothing$, then there is no feasible solution to the LNS problem. In the rest of this section, we assume that the LNS problem has feasible solutions. Then, for every $v \in V_B$ and $i = 1, 2, \ldots, l$, let $\mu_i(v)$ and $\kappa_i(v)$ be the smallest and largest d-th coordinates of the vertices on the chain $Ch_i(v)$ of G'_i, respectively. We denote by Z_0 the set of the "lowest" vertices (i.e., with the smallest d-th coordinate) on every chains of G' and by H_0 the induced subgraph of Z_0 in G'. Our above construction of G' ensures the following lemma.

Lemma 2—(1) H_0 is a strongly connected component of G′. (2) Z_0 is a closed set in G′. (3) For any non-empty closed set $\mathcal{C}$ in G′, $Z_0 \subseteq \mathcal{C}$.

Up to this point, the graph construction establishes the connection between a closed set in G' and l feasible net surfaces in G. Our goal is to compute a non-empty minimum-cost closed set in G', which can specify l optimal nets in G. Thus, we need to further assign a cost $w(\cdot)$ to each vertex in G'.

Note that the cost of $\mathcal{NS}$ is $\alpha(\mathcal{NS})=\sum_{i=1}^{l} b(\mathcal{N}_i)+\sum_{i=0}^{l} c_i(R_i)$, where R_i's are the disjoint subsets of V partitioned by the nets in $\mathcal{NS}$. Further, recall that we are looking for $LO(\mathcal{N}_i)$ instead of $\mathcal{N}_i$ directly, by using the subgraph G'_i. Thus, we want to find a way to assign vertex costs to V'_i, so that for any feasible $\mathcal{N}_i$ in G, the cost $b(\mathcal{N}_i)$ can be "distributed" to those vertices of G'_i corresponding to $LO(\mathcal{N}_i)$ (i.e., $\sum_{v_k \in LO(\mathcal{N}_i)} w(v_k^i)=b(\mathcal{N}_i)$). We may go one step further to require that for any vertex $v_r \in Col(v)$ in G, the total cost of the vertices v_k^i on $Ch_i(v)$ of G'_i with $k \leq r$ (i.e., $\sum_{k \leq r} w(v_k^i)$) be equal to the cost of v_r. In addition, note that $R_0 = LO(\mathcal{N}_1)$, $R_i = LO(\mathcal{N}_{i+1}) - LO(\mathcal{N}_i)$ ($i = 1, 2, \ldots, l-1$), and $R_l = V - LO(\mathcal{N}_l)$, and consider the removal of the deficient vertices as well. Thus, we have the following vertex-cost assignment scheme for each subgraph G'_i ($i = 1, 2, \ldots, l$): For every $v \in V_B$,

$$w(v_k^i)=\begin{cases} b(v_k)+\sum_{j=0}^{k}[c_{i-1}(v_j)-c_i(v_j)] & \text{if } k=\mu_i(v), \\ [b(v_k)-b(v_{k-1})]+[c_{i-1}(v_k)-c_i(v_k)] & \text{for } k=\mu_i(v)+1,\ldots,\kappa_i(v). \end{cases} \tag{1}$$

This completes the construction of G'.

4.4. Computing optimal layered net surfaces for the LNS problem

The graph G' thus constructed allows us to find l optimal net surfaces in G, by computing a non-empty minimum-cost closed set in G'. In order to do that, below we prove the following facts: (1) Any closed set $\mathcal{C} \neq \emptyset$ in G' defines l feasible net surfaces in G whose total cost differs from that of $\mathcal{C}$ by a fixed value; (2) any set $\mathcal{NS}$ of l feasible net surfaces in G corresponds to a closed set $\mathcal{C} \neq \emptyset$ in G' whose cost differs from that of $\mathcal{NS}$ by a fixed value. Consequently, a non-empty closed set in G' with the minimum cost can specify l optimal net surfaces in G.

Given any closed set $\mathcal{C} \neq \emptyset$ in G', we define l feasible net surfaces, $\mathcal{NS} = \{\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_l\}$, in G, as follows. Recall that we search for each net $\mathcal{N}_i$, in the subgraph $G'_i=(V'_i, E'_i)$. Let $C_i=C \cap V'_i$. For each vertex $v \in V_B$, denote by $\mathcal{C}_i(v)$ the set of vertices of $\mathcal{C}_i$ on the chain $Ch_i(v)$ of G'_i. Based on the construction of G'_i, it is not hard to show that $\mathcal{C}_i(v) \neq \emptyset$. Let $r_i(v)$ be the largest d-th coordinate of the vertices in $\mathcal{C}_i(v)$. Define the function $\mathcal{N}_i$ as $\mathcal{N}_i(v) = r_i(v)$ for every $v \in V_B$.

First, we show that $\mathcal{NS}$ thus defined in G indeed consists of l feasible net surfaces. Lemmas 3 and 4 summarize the fact.

Lemma 3—Each $\mathcal{N}_i$ defined by $\mathcal{C}_i$ is a feasible net surface in G.

Proof—For each edge $(v, u) \in E_B$, let $k' = \mathcal{N}_i(v)$ and $k'' = \mathcal{N}_i(u)$. We need to prove that $(v_{k'}, u_{k''})$ is an edge in G. In G, let the edge interval $I(v_{k'}, u)$ of $v_{k'}$ on $Col(u)$ be $(u_p, u_{p+1}, \ldots, u_{p+s})$

and the edge interval $I(u_{k''}, v)$ of $u_{k''}$ on $Col(v)$ be $(v_q, v_{q+1}, \dots, v_{q+t})$ (see Figure 5(a)). If vertex $u_{k''}$ is in the edge interval $I(v_{k'}, u)$, then we are done. Otherwise, either $p > k''$ or $p + s < k''$. Consider the case with $p > k''$. By the observation that $\mathcal{C}_i, \neq \varnothing$ is a closed set in G'_i and the construction of G'_i, vertex $u^i_p \in C_i$, which contradicts to the fact that k'' is the largest d-th coordinate of the vertices in $\mathcal{C}_i(u)$. Thus, we only need to consider the case with $p + s < k''$ (see Figure 5(a)). By a similar argument, we have $q + t < k'$. Since there is an edge between vertices v_{q+t} and $u_{k''}$ in G ($v_{q+t} \in I(u_{k''}, v)$), vertex $u_{k''}$ thus is in the edge interval $I(v_{q+t}, u)$ of v_{q+t} on $Col(u)$. Hence, we have the largest d-th coordinate $Top(I(v_{q+t}, u))$ of vertices in $I(v_{q+t}, u)$ is no less than k''. Note that $k'' > p + s$ and $p + s$ is the largest d-th coordinate $Top(I(v_{k'}, u))$ of vertices in $I(v_{k'}, u)$. Thus, $Top(I(v_{q+t}, u)) > Top(I(v_{k'}, u))$, contradicting with the proper ordering of the graph G (since $k' > q + t$). Hence, the lemma follows.

Lemma 4—Any two adjacent net surfaces $\mathcal{N}_i$ and $\mathcal{N}_{i+1}$ in G (i = 1, 2,…, l − 1) defined by a closed set $\mathcal{C} \neq \varnothing$ in G′ are (L_i, U_i)-separate.

Proof—For each vertex $v \in V_B$, we need to prove that $L_i < \mathcal{N}_{i+1}(v) - \mathcal{N}_i(v) \leq U_i$. Let $k' = \mathcal{N}_i(v)$ and $k'' = \mathcal{N}_{i+1}(v)$. Denote by v_p the i-th upstream vertex of $v_{k'}$ By the construction of G', there is a directed edge from $v^i_{k'}$ to v^{i+1}_p, which indicates that v^{i+1}_p is in $\mathcal{C}_{i+1}(v) \subset \mathcal{C}$. Note that k'' is the largest d-th coordinate of vertices in $\mathcal{C}_{i+1}(v)$. Thus, we have $p \leq k''$ and $k'' - k' \geq p - k' \geq L_i$ (the definition of the upstream vertices). On the other hand, let v_q denote the i + 1-st downstream vertex of $v_{k''}$. Then, vertex $v^{i+1}_{k''} \in G'_{i+1}$ has a directed edge to $v^i_q \in G'_i$. Similarly, $q \leq k'$. From the definition of the downstream vertices, $k'' - q \leq U_i$. Hence, $k'' - k' \leq k'' - q \leq U_i$. This proves the lemma.

Then, we show that the cost $\alpha(\mathcal{NS})$ of $\mathcal{NS}$ thus defined by $\mathcal{C}$ differs by a fixed value from the total vertex cost $w(\mathcal{C})$ of $\mathcal{C}$. Note that in G'_i, if a vertex $v^i_q \in C_i(v)$, then all vertices in $\{v^i_k | v^i_k \in Ch_i(v), k \leq q\}$ are also in $\mathcal{C}_i(v)$. Hence, the total vertex cost of $\mathcal{C}_i(v)$ is $w(C_i(v)) = \sum_{k=\mu_i(v)}^{r_i(v)} w(v^i_k)$. Thus, we have

$$
\begin{aligned}
\alpha(\mathcal{NS}) &= \sum_{i=1}^{l} b(\mathcal{N}_i) + \sum_{i=0}^{l} c_i(R_i) = \sum_{i=1}^{l} \sum_{u \in \mathcal{N}_i} b(u) + \sum_{i=0}^{l} \sum_{u \in R_i} c_i(u) \\
&= \sum_{i=1}^{l} \sum_{v \in V_B} \left\{ b(v_0) + \sum_{k=1}^{\mathcal{N}_i(v)} [b(v_k) - b(v_{k-1})] \right\} \\
&\quad + \left(\sum_{i=1}^{l} \sum_{v \in V_B} \sum_{k=0}^{\mathcal{N}_i(v)} [c_{i-1}(v_k) - c_i(v_k)] + \sum_{v \in V_B} \sum_{k=0}^{\kappa-1} c_l(v_k) \right) \\
&= \sum_{i=1}^{l} \sum_{v \in V_B} \left\{ \underbrace{\left(b(v_0) + \sum_{k=1}^{\mu_i(v)} [b(v_k) - b(v_{k-1})] + \sum_{k=0}^{\mu_i(v)} [c_{i-1}(v_k) - c_i(v_k)] \right)}_{\text{the cost of the “lowest” vertex } v^i_q(q=\mu_i(v)) \text{ of } Ch_i(v) \text{ in } G'_i} \right. \\
&\quad \left. + \sum_{k=\mu_i(v)+1}^{r_i(v)} \underbrace{([b(v_k) - b(v_{k-1})] + [c_{i-1}(v_k) - c_i(v_k)])}_{\text{the cost of vertex } v^i_k \in Ch_i(v)} \right\} + \sum_{v \in V_B} \sum_{k=0}^{\kappa-1} c_l(v_k) \\
&= \sum_{i=1}^{l} \sum_{v \in V_B} \sum_{k=\mu_i(v)}^{r_i(v)} w(v^i_k) + \sum_{v \in V_B} \sum_{k=0}^{\kappa-1} c_l(v_k) = \sum_{i=1}^{l} \sum_{v \in V_B} w(C_i(v)) + \sum_{v \in V_B} \sum_{k=0}^{\kappa-1} c_l(v_k) \\
&= w(C) + \sum_{v \in V_B} \sum_{k=0}^{\kappa-1} c_l(v_k)
\end{aligned}
\tag{2}
$$

Note that the term $\sum_{v\in V_B}\sum_{k=0}^{\kappa-1} c_l(v_k)$, denoted by $c_l(V)$, is fixed and is the total sum of the l-th in-region costs of all vertices in G. From Lemmas 3 and 4 and Equation (2), the following lemma holds.

Lemma 5—Any closed set $\mathcal{C} \neq \emptyset$ in G′ specifies l feasible net surfaces in G whose total cost differs from that of $\mathcal{C}$ by a fixed value $c_l(V)$.

Next, we argue that any l feasible net surfaces, $\mathcal{NS} = \{\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_l\}$ in G correspond to a closed set $\mathcal{C} \neq \emptyset$ in G'. Based on the construction of G', every vertex v_k on the net $\mathcal{N}_i$ corresponds to a vertex v_k^i in G_i' (v_k^i is not a deficient vertex). We construct a closed set $\mathcal{C}_i \neq \emptyset$ in G_i' for each net $\mathcal{N}_i$, as follows. Initially, let $\mathcal{C}_i = \emptyset$. For each vertex $v \in V_B$, we add to $\mathcal{C}_i$, the subset $C_i(v)=\left\{v_k^i | k \le \mathcal{N}_i(v)\right\}$ of vertices on $Ch_i(v)$ of G_i'. Let $C=\cup_{i=1}^{l} C_i$. We prove that $\mathcal{C}$ is a closed set in G'.

Lemma 6—Any l feasible net surfaces, $\mathcal{NS} = \{\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_l\}$, in G define a closed set $\mathcal{C}$ in G ′.

Proof—For each vertex $v_p^i \in C$, due to the construction of the graph G', the only successors of v_p^i are: (1) v_{p-1}^i (if $p \ge \mu_i(v)$) on the same chain $Ch_i(v)$; (2) one vertex u_q^i on every adjacent chain $Ch_i(u)$ of $Ch_i(v)$ in G_i';(3) one vertex $v_{p'}^{i+1}$ on the chain $Ch_{i+1}(v)$ in G_{i+1}' (if $i < 1$); and (4) one vertex $v_{p''}^{i-1}$ on the chain $Ch_{i-1}(v)$ in G_{i-1}' (if $i > 1$).

<u>**Case (1):**</u> Obviously, $v_{p-1}^i \in C_i \subseteq C$ due to the construction of $\mathcal{C}$.

<u>**Case (2):**</u> Consider each adjacent chain $Ch_i(u)$ of $Ch_i(v)$ in G_i'. Let $r = \mathcal{N}_i(v)$ and $t = \mathcal{N}_i(u)$. We have $r \ge p$ since r is the largest d-th coordinate of vertices of $Ch_i(v)$ that we put in $\mathcal{C}_i$. Since all vertices u_k^i with $\mu_i(u) \le k \le t$ are in $\mathcal{C}_i$, if $q \le t$, then we are done. Hence, we assume that $q > t$ (see Figure 5(b)). Since u_q^i is a successor of v_p^i, the smallest d-th coordinate $Bottom(I(v_p, u))$ of the vertices in the edge interval $I(v_p, u)$ of v_p on $Col(u)$ is q (i.e., $Bottom(I(v_p, u)) = q$). Note that the edge (v_r, u_t) is on the net surface $\mathcal{N}_i$. Thus, $u_t \in I(v_r, u)$ and $Bottom(I(v_r, u)) \le t$. Hence, we have $Bottom(I(v_p, u)) = q > t \ge Bottom(I(v_r, u))$, a contradiction to the proper ordering property of the graph G (since $r \ge p$). Consequently, $q \le t$ and $u_q^i \in C_i \subseteq C$.

<u>**Case (3):**</u> Let $r = \mathcal{N}_i(v)$ and $t = \mathcal{N}_{i+1}(v)$. Then, $r \ge p$. We need to prove $t \ge p'$. Since the successor of v_p^i on $Ch_{i+1}(v)$ is $v_{p'}^{i+1}$, $v_{p'}$ is the vertex in $Col(v)$ with the smallest d-th coordinate that can be on $\mathcal{N}_{i+1}$ (v_p is on the net $\mathcal{N}_i$). Note that $p' - p \ge L_i$, but p' may not equal to $p + L_i$ due to the deficient vertex pruning operations. Now assume that the successor of v_r^i on $Ch_{i+1}(v)$ is $v_{r'}^{i+1}$. We conclude that $r' \ge p'$. Otherwise, since $r' - r \ge L_i$ and $r \ge p$, we have $p' - p > r' - p \ge r' - r \ge L_i$ (see Figure 5(c)). Hence, vertex $v_{r'}$ is the vertex in $Col(v)$ with the smallest d-th coordinate that can be on $\mathcal{N}_{i+1}$, a contradiction. Thus, $r' \ge p'$. Since $v_r \in \mathcal{N}_i$ and $v_t \in \mathcal{N}_{i+1}$, $t \ge r'$. This indicates that $t \ge p'$.

<u>**Case (4):**</u> A similar argument as for Case (3) can be applied, which proves that $v_{p''}^{i-1} \in C$.

In conclusion, for each vertex $v_p^i \in C$, all successors of v_p^i are in $\mathcal{C}$. Thus, the lemma holds.

Using a similar argument as in Equation (2), we can show that $w(\mathcal{C}) = \alpha(\mathcal{NS}) - c_l(V)$. Together with Lemma 6, the following lemma follows.

Lemma 7—Any set $\mathcal{NS}$ of l feasible net surfaces in G defines a closed set $\mathcal{C} \neq \emptyset$ in G′ whose cost differs from that of $\mathcal{NS}$ by a fixed value.

By Lemmas 5 and 7, we compute a minimum-cost closed set $\mathcal{C}^* \neq \varnothing$ in G', which specifies l optimal net surfaces in G. However, the minimum closed set $\mathcal{C}^*$ in G' can be empty (with a weight zero), and when this is the case, $\mathcal{C}^* = \varnothing$ gives little useful information on G'. Fortunately, our careful construction of G' still enables us to overcome this difficulty. If the minimum closed set in G' is empty, then it implies that the weight of every non-empty closed set in G' is non-negative. To obtain a minimum *non-empty* closed set in G', we do the following: Let M be the total weight of vertices in Z_0; pick an arbitrary vertex $u \in Z_0$ and assign a new weight $w(u) - M - 1$ to u. We call this a *translation operation* on G'. From Lemma 2, $Z_0 \neq \varnothing$ is a closed set in G' and is a subset of any non-empty closed set in G'. Further, observe that the total weight of vertices in the closed set Z_0 (after a translation operation on G') is negative. This implies that any minimum closed set in G' (after a translation operation on G') cannot be empty. Also based on Lemma 2, we have the following lemma.

Lemma 8—For a non-empty closed set $\mathcal{C}$ in G′, let w($\mathcal{C}$) denote the total weight of $\mathcal{C}$ before any translation operation on G′. Then after a translation operation, the weight of $\mathcal{C}$ is w($\mathcal{C}$) − M − 1.

Now, we can simply find a minimum closed set $\mathcal{C}^*$ in G' after performing a translation operation on G'. Based on Lemma 8, $\mathcal{C}^*$ is a minimum *non-empty* closed set in G' before the translation.

As in the paper,[20,14,30] we find a minimum closed set $\mathcal{C}^* \neq \varnothing$ in G' by formulating it as computing a minimum *s*-*t* cut in a weighted directed graph G″ transformed from G', with $|G''| = O(|G'|)$. Note that G' has $O(l \cdot n)$ vertices and $O(l \cdot n \cdot \frac{m_B}{n_B})$ edges, where $n = |V|$ is the number of vertices in G, and $n_B = |V_B|$ and $m_B = |E_B|$ for the net model B. The deficient vertex pruning procedure takes $O(l \cdot n \cdot \frac{m_B}{n_B})$ time. By using the minimum *s*-*t* cut algorithm in the paper,[12] we obtain the following result.

Theorem 1

The LNS problem can be solved in $O(l^2 n^2 \frac{m_B}{n_B} \log(\frac{l \cdot n \cdot n_B}{m_B}))$ time.

5. Algorithm for the Bipartite LNS (BLNS) Problem

In this section, we consider the layered net surface problem on a *d*-D ($d \geq 3$) *ordered* multi-column graph. We prove that the LNS problem on such a multi-column graph is NP-hard and give a polynomial time algorithm for solving the BLNS problem. Recall that the BLNS problem is defined on an ordered multi-column graph such that the *p*-contracted graph of its net model *B* is bipartite.

Theorem 2

The optimal LNS problem on a d-D ordered multi-column graph (d ≥ 3) is NP-hard.

Proof—We can prove that the LNS problem on a general *d*-D ordered multi-column graph is NP-hard, by reducing to it the minimum vertex cover problem that is known to be NP-complete.[10]

Given an undirected graph $B = (V_B, E_B)$, the minimum vertex cover problem seeks a *vertex cover* $V' \subseteq V_B$ such that for each edge $(v, u) \in E_B$, at least one of v and u is in V' and $|V'|$ is minimized. This problem is reduced to the optimal LNS problem, as follows. First, embed the graph B into the 2-D space. Next, a 3-D ordered multi-column graph $G = (V, E)$ is constructed. For each vertex $v = (x, y) \in V_B$, there are two corresponding vertices in V, $v_0 = (x, y, 0)$ and $v_1 = (x, y, 1)$. Assign a cost 0 to v_0 and a cost 1 to v_1. For every edge $(v, u) \in E_B$, v_0 connects with u_1 in G and v_1 connects with both u_0 and u_1. Obviously, the graph G thus generated is an ordered multi-column graph. Actually, any two adjacent columns in G are in reverse order. We simply assume $l = 1$. Note that a vertex $v \in V_B$ is in a vertex cover V' of B if and only if $v_1 \in V$ is on a net surface in G, and the cost of the net surface equals the cardinality of the vertex cover. Hence, Theorem 2 holds.

The NP-hardness proof of the LNS problem actually shows that on a d-D ordered multi-column graph ($d \geq 3$) generated by a net model with only *reverse* edges, the LNS problem is NP-hard. On the other hand, Section 4 shows that on any d-D ordered multi-column graph generated by a net model whose edges are all *proper*, the LNS problem is polynomially solvable. Hence, it is desirable to examine net models in order to explore the polynomial solvability of the LNS problem. We next consider the LNS problem on a d-D ordered multi-column graph $G = (V, E)$ with a special net model $B = (V_B, E_B)$ whose p-contracted graph is bipartite.

Observe that if the p-contracted graph of the net model $B = (V_B, E_B)$ is a bipartite one, then essentially, we can partition V_B into two disjoint subsets Q and $\bar{Q}$ ($V_B = Q \cup \bar{Q}$), such that each reverse edge of B connects one vertex in Q with one in $\bar{Q}$, but both vertices of each proper edge of B are either in Q or in $\bar{Q}$. Based on this observation, we can construct a properly ordered graph $G' = (V', E')$ such that the optimal solution to the LNS problem in G' defines an optimal solution to the BLNS problem in G.

Each vertex in G defines one and exactly one vertex in G'. For every vertex v_k in G, if $v \in Q$, then it corresponds to vertex v'_k in G'; otherwise, if $v \in \bar{Q}$, then it corresponds to vertex $v'_{\kappa-1-k}$ in G' (see Figure 6). The cost of each vertex in G' is equal to the on-surface cost of its corresponding vertex in G. We denote by $Col(v')$ the column of κ vertices in G' that are associated with $v \in V_B$. Then, edges in G' are introduced. For each edge (v_k, u_j) in G, there are three cases: (1) if both v and u are in Q, then put in G' an edge (v'_k, u'_j); (2) if both v and u are in $\bar{Q}$, then add in G' an edge ($v'_{\kappa-1-k}, u'_{\kappa-1-j}$); and (3) if $v \in Q$ and $u \in \bar{Q}$, then introduce in G $'$ an edge ($v'_k, u'_{\kappa-1-j}$). Figure 6(b) illustrates the construction of the properly ordered graph G $'$ from the ordered graph G in Figure 4(a).

Lemma 9—The graph G′ = (V′, E′) constructed from G is a properly ordered multi-column graph.

Proof—Consider any edge $(v, u) \in E_B$ of the net model B. If (v, u) is a proper edge, obviously, the columns $Col(v')$ and $Col(u')$ in G' are in proper order. We thus only investigate the case that (v, u) is a reverse edge. Without loss of generality (WLOG), assume $v \in Q$ and $u \in \bar{Q}$. In G, each vertex V_k has an edge interval $I(v_k, u) = \{u_p, u_{p-1}, \ldots, u_{p-s}\}$ ($s \geq 0$) on $Col(u)$. From the construction of G', vertex v'_k connects with an interval of vertices on $Col(u')$, that is, $\{ u'_{\kappa-1-p}, u'_{\kappa-p}, \ldots, u'_{\kappa-1-p+s}\}$.

We now consider the edge intervals of two adjacent vertices v'_k and v'_{k+1} on $Col(u')$. Let $I(v'_k, u')=\{u'_p, u'_{p-1}, \ldots, u'_{p-s}\}$ ($s \geq 0$) and $I(v'_{k+1}, u')=\{u'_q, u'_{q-1}, \ldots, u'_{q-t}\}$ ($t \geq 0$). This indicates that the edge interval $I(v_k, u)$ of v_k on $Col(u)$ is $\{u_{\kappa-1-p}, u_{\kappa-p}, \ldots, u_{\kappa-1-p+s}\}$ and the edge interval $I(v_{k+1}, u)$ of v_{k+1} on $Col(u)$ is $\{u_{\kappa-1-q}, u_{\kappa-q}, \ldots, u_{\kappa-1-q+t}\}$. Since $Col(v)$ and $Col(u)$ are in reverse order, we have $\kappa - 1 - p \geq \kappa - 1 - q$ and $\kappa - 1 - p + s \geq \kappa - 1 - q + t$. Alternatively, $p \leq q$ and $p - s \leq q -$ t, that is, $Top(I(v'_k, u')) \leq Top(I(v'_{k+1}, u'))$ and $Bottom(I(v'_k, u')) \leq Bottom(I(v'_{k+1}, u'))$.

On the other hand, consider any two adjacent vertices u'_k and u'_{k+1}, and their edge intervals on $Col(v')$. Let($I(u'_k, v')=\{v'_p, v'_{p-1}, \ldots, v'_{p-s}\}$ (s ≥ 0) and $I(u'_{k+1}, v')=\{v'_q, v'_{q-1}, \ldots, v'_{q-t}\}$ ($t \geq 0$). Note that vertex u'_k (resp., u'_{k+1}) in G' corresponds to $u_{\kappa-1-k}$ (resp., $u_{\kappa-2-k}$) in G. The edge interval of $u_{\kappa-1-k}$ on $Col(v)$ is $\{v_p, v_{p-1}, \ldots, v_{p-s}\}$ and the edge interval of $u_{\kappa-2-k}$ on $Col(v)$ is $\{v_q, v_{q-1}, \ldots, v_{q-t}\}$. Due to the inverse order between $Col(v)$ and $Col(u)$, we have $p \leq q$ and $p - s \leq q - t$, that is, $Top(I(u'_k, v')) \leq Top(I(u'_{k+1}, v'))$ and $Bottom(I(u'_k, v')) \leq Bottom(I(u'_{k+1}, v'))$.

Hence, *Col(v′)* and *Col(u′)* are in proper order. This proves the lemma.

We show in the following that the optimal solution to the LNS problem in G' defines an optimal solution to the BLNS problem in G.

Lemma 10—Any l feasible net surfaces in G′ specify l feasible net surfaces in G with the same total cost, and vice versa.

Proof—By Lemma 9, G' is a properly ordered graph. Let $\mathcal{NS}'=\{\mathcal{N}'_1, \mathcal{N}'_2, \ldots, \mathcal{N}'_l\}$ be a feasible solution to the LNS problem in G'. For each net surface $\mathcal{N}'_i$, we define a function $\mathcal{N}_i$ in G, as follows. Consider every vertex $v \in V_B$. If $v \in Q$, then $\mathcal{N}_i(v)=\mathcal{N}'_i(v')$; otherwise, if $v \in \bar{Q}$, then $\mathcal{N}_i(v)=\kappa - 1 - \mathcal{N}'_i(v')$. The construction of G' from G immediately indicates that each $\mathcal{N}_i$ is a feasible net surface in G and any two adjacent net surfaces $\mathcal{N}_i$ and $\mathcal{N}_{i+1}$ satisfy the separation constraints. Furthermore, the total cost of $\mathcal{N}_i$ equals to the total cost of $\mathcal{N}'_i$. Hence, $\mathcal{NS} = \{\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_l\}$ is a feasible solution to the BLNS problem in G, and $\alpha(\mathcal{NS}) = \alpha(\mathcal{NS}')$. A similar argument can show that any l feasible nets in G define l feasible net surfaces in G' with the same total cost.

Note that G' has the same size as G (i.e., |V′| = |V|and $|E'| = |E|$), and by Theorem 1 an optimal set of l nets in G' can be computed in $O(l^2 n^2 \frac{m_B}{n_B} \log(\frac{l \cdot n \cdot n_B}{m_B}))$ time. Based on Lemma 10, Theorem 3 follows.

Theorem 3—The general BLNS problem can be solved in $O(l^2 n^2 \frac{m_B}{n_B} \log(\frac{l \cdot n \cdot n_B}{m_B}))$ time, where l is the number of sought net surfaces, n = |V|, $m_B = |E_B|$, and $n_B = |V_B|$.

6. Algorithms for the Net Surface Volume (NSV) Problems

This section presents our algorithms for several **optimal net surface volume (NSV)** problems. Specifically, instead of looking for multiple inter-related net surfaces as in Section 4, for a given *d-D* voxel grid $\Gamma = [0..N-1]^d$ of $n = N^d$ cells, with each cell $\mathbf{x}(x_0, x_1, \ldots, x_{d-1}) \in \Gamma$ having an arbitrary real "volume" value $vol(\mathbf{x})$, we seek multiple surfaces that enclose a well-shaped region $R \subseteq \Gamma$, such that the *volume* $vol(R)$ of R, $vol(R) = \Sigma_{\mathbf{x} \in R}{}^{vol(\mathbf{x})}$, is minimized (or maximized). Note that even the case of the NSV problem on finding an optimal simple polygon in a weighted 2-D grid is in general NP-hard.[2,28] The NSV problems in higher dimensions are so under-explored that few known methods actually address them. Interestingly, our LNS approach can be generalized to solving the optimal NSV problems on several nontrivial and useful classes of geometric regions in polynomial time. Recently, Chen *et al.*[7] studied certain more restricted classes of regions than our classes.

6.1. The weakly watershed-monotone regions and watershed-monotone shells

We consider two classes of regions, called *weakly watershed-monotone regions* and *watershed-monotone shells*, defined as follows. For any integers $0 \leq i < d$ and $0 \leq c < N$, let $\Gamma_i (c)$ denote the hyperplane $x_i = c$ in the domain of Γ, i.e., $\Gamma_i (c)$ is orthogonal to the x_i-axis and consists of all voxels of Γ whose x_i-coordinate is c. A region R in Γ is said to be x_i*-monotone* if for any line l parallel to the x_i-axis, the intersection $R \cap l$ *is* either empty or a continuous segment. Further, we say that R is *watershed-monotone with respect to* $\Gamma_i (c)$ if (1) R is x_i-monotone, and (2) for any line l orthogonal to $\Gamma_i (c)$, if the intersection $R \cap l$ ne; $\varnothing$, then $R \cap l$ intersects a voxel of $R \cap \Gamma_i (c)$. (Intuitively, the intersection of R and $\Gamma_I(c)$ is equal to the projection of R onto $\Gamma_i (c)$, and is like a “watershed” of R.) If for every $i = 0,1,\ldots, d - 1$, R is watershed-monotone to a $\Gamma_i (c_i)$ for an integer $0 \leq c_i < N$, then we say that R is *watershed-monotone*. In Figure 7(a), the region R is watershed-monotone to both $\Gamma_0 (c_0)$ and $\Gamma_1 (c_1)$, and thus R is watershed-monotone. A region $R \subseteq \Gamma$ is *weakly watershed-monotone* if R is watershed-monotone to every axis in a set of $d - 1$ axes of Γ and is monotone (but need not be watershed-monotone) to the remaining axis (e.g., see Figure 7(b)). Clearly, watershed-monotone regions are a subclass of weakly watershed-monotone regions.

Suppose R is watershed-monotone with respect to some $\Gamma_i (c_i)$, for each $i = 0,1,\ldots, d - 1$; then it is easy to see that $\cap_{i=0}^{d-1}\Gamma_i(c_i) \neq \varnothing$. A voxel in $\cap_{i=0}^{d-1}\Gamma_i(c_i)$ is called a *kernel voxel* of R. Our second region class is called the *watershed-monotone shells*. For any two watershed-monotone regions R_1 and R_2 such that R_1 and R_2 have a common kernel voxel **c** and $R_2 \subseteq R_1$, the region R in Γ bounded between R_1 and R_2, i.e., $R = R_1 - R_2$, is a *watershed-monotone shell* (e.g., see Figure 7(c)).

6.2. Modeling the optimal NSV problem for weakly watershed-monotone regions

In this section, we extend our approach for the LNS problem to solving the optimal NSV problem on weakly watershed-monotone regions. WLOG, we assume that a voxel $\mathbf{c}(c_0, c_1,\ldots, c_{d-2}, c_{d-1}) \in \Gamma$ is given and the target weakly watershed-monotone region R is x_{d-1}-monotone (but not necessarily watershed-monotone to any Γd_{-1}(c)) and is watershed-monotone to $\Gamma_i(c_i)$ for each $i \in \{0,1,\ldots, d - 2\}$.

Denote by $\mathcal{D}$ the projection of Γ onto the first $d - 1$ dimensions, i.e., $\mathcal{D} = [0..N - 1]^{d-1}$. A surface S' in Γ is said to be x_i*-monotone* if for any line l parallel to the x_i-axis, the intersection of S' and l either is empty or is in one single voxel. Observe that the boundary surface S of a weakly watershed-monotone region R in Γ can be split into two surface pieces, the “upper” boundary surface S^u and the “lower” boundary surface S^l, both being x_{d-1}-monotone. We let S^u consist of the voxels forming the upper boundary of R ($S^u \subseteq R$), and S^l consist of the voxels that are immediately below the voxels on the lower boundary of R ($S^l \not\subseteq R$), with respect to $\Gamma_{d-1}(0)$. Obviously, the domain of S^u and S^l may be only a subset of $\mathcal{D}$, denoted by $\mathcal{D}_R$. To obtain R, it is sufficient to compute S^u and S^l. Note that we may view $\mathcal{D}$ as an implicit net model and Γ as an implicit ordered multi-column graph.

Our approach in Section 4 can simultaneously identify multiple inter-related net surfaces that are defined on the *entire* domain $\mathcal{D}$. However, the domain $\mathcal{D}_R$ of S^u and S^l may just be a proper subset of $\mathcal{D}$. Besides, here we optimize a different “volume” criterion. We overcome these difficulties as follows. First, we “extend” (or “spread”) each of S^u and S^l to become an x_{d-1}-monotone surface defined on the entire $\mathcal{D}$, denoted by S_e^u and S_e^l, respectively (e.g., Figures 8 (a) and 8(b)). Interestingly, based on the monotonicity of R, we can manage to extend S^u and S^l in such a way that $S_e^u - S^u = S_e^l - S^l$. This implies that for any voxel $\mathbf{x} \in \mathcal{D} - \mathcal{D}_R$, the voxel of S_e^u defined on **x** must be the same as the voxel of S_e^l defined on **x**. (Recall that $LO(S')$ denote the region in Γ consisting of all voxels on or below a (net) surface S'.) Then, the volume *vol*

(R) of the sought region R is equal to $vol(LO(S_e^u)) - vol(LO(S_e^l))$. Hence, we look for two such extended surfaces S_e^u and S_e^l with $vol(LO(S_e^u)) - vol(LO(S_e^l))$ minimized; further, both S_e^u and S_e^l must be of a "good" shape to form a weakly watershed-monotone region.

Next, we construct two properly ordered graphs $G_1= (V_1, E_1)$ and $G_2 = (V_2, E_2)$, which are for searching for S_e^l and S_e^u, respectively. S_e^l and S_e^u are both defined on the following net model $B = (V_B, E_B)$. Each voxel $\mathbf{x} \in \mathcal{D}$ corresponds to exactly one vertex $u_{\mathbf{x}} \in V_B$ of B. For every two adjacent voxels of $\mathcal{D}$, $\mathbf{x} = (x_0, x_1,\ldots, x_{d-2})$ and $\mathbf{y} = (y_0, y_1, \ldots, y_{d-2})$, along any dimension (i.e., $\sum_{i=0}^{d-2}|x_i - y_i|=1$), we put an edge $(u_{\mathbf{x}}, u_{\mathbf{y}})$ in B.

Before we start constructing G_1 and G_2 to incorporate the watershed-monotonicity, we need to introduce some notation. For a voxel $\mathbf{x}(x_0, x_1,\ldots, x_{d-1}) \in \Gamma$, let $\mathbf{x}(i, val)$ denote the voxel in Γ whose i-th coordinate is val and the remaining coordinate values are the same as those of voxel $\mathbf{x}$. The i-th *column* $Col(\mathbf{x}, i)$ of voxel $\mathbf{x}$ consists of all voxels $\{\mathbf{x}(i, k) \mid k = 0,1,\ldots, N - 1\}$ in Γ. Two columns $Col(\mathbf{x}, d - 1)$ and $Col(\mathbf{y}, d - 1)$ are said to be *adjacent in the p-th dimension* ($p = 0,1,\ldots, d–2$), if $x_i = y_i$ for every $i \notin \{p, d - 1\}$ and $|x_p - y_p| = 1$.

We first construct graph $G_1 = (V_1, E_1)$. Each voxel $\mathbf{x} \in \Gamma$ defines exactly one vertex $v_{\mathbf{x}}^1$ in G_1 (if the voxel in Γ is denoted by $\mathbf{x}(i, val)$, we then denote by $v_{\mathbf{x}}^1(i, val)$ the corresponding vertex in G_1). Each vertex $v_{\mathbf{x}}^1 \in V_1$ is assigned a cost $c(v_{\mathbf{x}}^1)= - vol(\mathbf{x})$. We next introduce edges into G_1. For each dimension i ($i = 0,1,\ldots, d - 2$), consider every pair of $(d - 1)$-st columns $Col(\mathbf{x}, d - 1)$ and $Col(\mathbf{y}, d - 1)$ adjacent in the i-th dimension. WOLG, we assume that $y_i = x_i + 1$. Recall that we want to compute a weakly watershed-monotone region R with respect to a given voxel $\mathbf{c}(c_0, c_1, \ldots c_i,\ldots, c_{d-1})$ in Γ. Two cases are distinguished. (1) $x_i < C_i$: Note that for any two voxels $\mathbf{x}(d - 1, k')$ and $\mathbf{y}(d - 1, k'')$ on S_{e}^l, we have $k' \geq k''$. Thus, for every $0 \leq k < N$, let vertex $v_{\mathbf{x}}^1(d-1,k)$ connect with each vertex $v_{\mathbf{y}}^1(d-1,k')$ with $k' \leq k$. (2) $x_i \geq c_i$: For any two voxels $\mathbf{x}(d - 1, k')$ and $\mathbf{y}(d - 1, k'')$ on S_{e}^l, we have $k' \leq k''$. Hence, for every $0 \leq k < N$, we connect vertex $v_{\mathbf{y}}^1(d-1,k)$ with each vertex $v_{\mathbf{x}}^1(d-1,k')$ with $k' \leq k$. The construction of G_2 is similar to that of G_1. But the ways of putting edges and assigning vertex costs in G_2 are different. Every vertex $v_{\mathbf{x}}^2(d-1,k)$ in G_2 connects vertices $v_{\mathbf{y}}^2(d-1,k')$ with $k' \leq k$ if $v_{\mathbf{x}}^2(d-1,k)$'s corresponding vertex $v_{\mathbf{x}}^1(d-1,k)$ in G_1 connects vertices $v_{\mathbf{y}}^1(d-1,k'')$ with $k'' \geq k$ in G_1; and vice versa. In addition, each vertex $v_{\mathbf{x}}^2$ in G_2 is assigned a cost $c(v_{\mathbf{x}}^2)=vol(\mathbf{x})$. Lemma 11 immediately follows from the graph constructions.

Lemma 11—The graphs G_1 and G_2 thus obtained are both properly ordered graphs on the net model B.

Suppose we are given any two feasible net surfaces $\mathcal{N}_1$ in G_1 and $\mathcal{N}_2$ in G_2, with $\mathcal{N}_2$ "above" $\mathcal{N}_1$ (i.e., $\forall u_{\mathbf{x}} \in V_B$, $\mathcal{N}_2(u_{\mathbf{x}}) \geq \mathcal{N}_1(u_{\mathbf{x}})$). We can specify a region R in Γ using $\mathcal{N}_1$ and $\mathcal{N}_2$, which actually correspond to S_e^l and S_e^u respectively in Γ, as follows. Initially, $R = \varnothing$. For every voxel $\mathbf{x} \in \mathcal{D}$, if $\mathcal{N}_2(u_{\mathbf{x}}) > \mathcal{N}_1(u_{\mathbf{x}})$, then we put in R all voxels $\mathbf{x}(d - 1, k)$ with $\mathcal{N}_2(u_{\mathbf{x}}) \geq k > \mathcal{N}_1(u_{\mathbf{x}})$. The next lemma shows that the region R thus obtained is aweakly watershed-monotone in Γ.

Lemma 12—Any feasible net surfaces $\mathcal{N}_1$ in G_1 and $\mathcal{N}_2$ in G_2 with $\mathcal{N}_2$ above $\mathcal{N}_1$ define a weakly watershed-monotone region R in Γ with vol(R) = c(LO($\mathcal{N}_1$)) + c(LO($\mathcal{N}_2$)).

Proof: First, by the construction of R, it is easy to see that R is x_{d-1}-monotone. Hence, below we show that R is watershed-monotone to every $\Gamma_i(c_i)$ for $i = 0, 1, \ldots, d - 2$, i.e., R is x_i-

monotone and the intersection of R with Γ_i (c_i) equals the projection of R to Γ $_i(c_i)$. We prove this by contradiction.

Assume that there exists an $h \in \{0,1,\ldots, d-2\}$ such that R is not x_h-monotone. Then, there is a voxel $\mathbf{a}(a_0,\ldots, a_{h-1}, c_h, a_{h+1},\ldots, a_{d-1}) \in \Gamma_h$ (c_h) such that the intersection of R with the *h-th* column *Col* ($\mathbf{a}$, h) of $\mathbf{a}$ has at least two disconnected voxel segments, denoted by I_0, $I_1,\ldots,$ I_q along the h-th dimension, in this order. Let $\mathbf{a}$ (h, le_j) and $\mathbf{a}(h, re_j)$ be the two end voxels of segment I_j. For any two segments I_j and I_{j+1}, the *gap segment* I_g consists of all voxels $\mathbf{a}$ (h, k) in between I_j and I_{j+1} with $re_j < k < le_{j+1.}$ Considering the possible position of I_g in Γ, I_g can be "above" S_e^u (i.e., $I_g \subseteq \Gamma - LO(S_e^u)$), or $I_g \subseteq LO(S_e^l)$; further, I_g can be to the "left" (i.e., $le_{j+1} \leq c_h$), the "right" (i.e., $re_j \geq c_h$) of Γ $_h(c_h)$, or intersect Γ $_h(c_h)$ (i.e., $re_j < c_h < le_{j+1}$). Hence, there are six cases for the possible position of I_g. In one such case, for instance, I_g is to the left of Γ_h (c_h) and is a subset of $\Gamma - LO(S_e^u)$. The voxel $\mathbf{a}(h,\mathrm{re}_j) \in R$ implies that the vertex $v_{\mathbf{a}}^2(h, re_j) \in LO(\mathcal{N}_2)$. Due to the construction of G_2, all vertices $v_{\mathbf{a}}^2(h,k)$ in G_2 with $re_j < k \leq c_h$ are in $LO(\mathcal{N}_2)$, that is, $I_g \subseteq LO(S_e^u)$. Hence, the only possibility for which $I_g \not\subseteq R$ is when $I_g \subseteq LO(S_e^l)$, implying that the vertex $v_{\mathbf{a}}^1(h, re_j+1) \in G_1$ which corresponds to voxel $\mathbf{a}(h, re_j +$ $1) \in I_g$ is in $LO(\mathcal{N}_1)$. By the construction of G_1, all vertices $v_{\mathbf{a}}^1(h,k)$ with $re_j \geq k \geq 0$ are in LO $(\mathcal{N}_1)$. Hence, the voxel segment I_j is not in R, a contradiction. The other five cases can be handled similarly. Therefore, for every $i = 0, 1, \ldots, d-2$, R is monotone to Γ_i (c_i).

Then, we argue that for each $i = 0,1,\ldots, d-2$, the intersection of R with Γ_i (c_i) equals the projection of R to Γ $_i(c_i)$. Assume otherwise, i.e., there exists a voxel $\mathbf{a} \in R$ whose projection $\mathbf{a}(h, c_h)$ on Γ $_h$ (c_h) is not in R. Note that $\mathbf{a} \in R$ indicates that its corresponding vertex $v_{\mathbf{a}}^2 \in G_2$ is in $LO(\mathcal{N}_2)$, and its corresponding vertex $v_{\mathbf{a}}^1 \in G_1$ is not in $LO(\mathcal{N}_1)$. Thus, the vertex $v_{\mathbf{a}}^2(h,c_h)$ of G_2 is also included in $LO(\mathcal{N}_2)$ based on the construction of G_2. But, $v_{\mathbf{a}}^2(h,c_h) \in LO(\mathcal{N}_2)$ and $\mathbf{a}(h, c_h) \notin R$ imply that the vertex $v_{\mathbf{a}}^1(h,c_h)$ of G_1 is in $LO(\mathcal{N}_1)$. Hence, all vertices $v_{\mathbf{a}}^1(p,k)$ with $0 \leq k < N$ are in $LO(\mathcal{N}_1)$, which means $a \notin R$, a contradiction. Thus, R is watershed-monotone to every $\Gamma_i(c_i)$ for $i = 0,1,\ldots, d-2$.

Finally, we calculate the volume $vol(R)$ of the region R.

$$\begin{aligned} vol(R) = & \sum_{\mathbf{x}\in R} vol(\mathbf{x}) \\ & = \sum_{\mathbf{x}\in LO(S_e^u)} vol(\mathbf{x}) \\ & - \sum_{\mathbf{x}\in LO(S_e^l)} vol(\mathbf{x}) \\ & = c(LO(\mathcal{N}_1)) \\ & + c(LO(\mathcal{N}_2)). \end{aligned}$$

This proves the lemma.

6.3. Computing a weakly watershed-monotone region of minimum total volume

Based on Lemma 12, we need to compute two such net surfaces $\mathcal{N}_1^*$ and $\mathcal{N}_2^*$ with the total cost $c(LO(\mathcal{N}_1^*))+c(LO(\mathcal{N}_2^*))$ minimized. As in Section 4, we formulate this problem as computing a minimum-cost closed set in a vertex-weighted directed graph G′ constructed from $G_1$and G_2.

The directed graph $G' = (V', E')$ contains two vertex-disjoint subgraphs $G'_1=(V'_1, E'_1)$ and $G'_2=(V'_2, E'_2)$, which are used to search for $\mathcal{N}_1^*$ and $\mathcal{N}_2^*$, respectively. G'_1 (resp., G'_2) is constructed from G_1(resp., G_2) using the same approach as that in Section 4.3 for building G'_i from G. The cost $w(\cdot)$ of each vertex in G'_1 (resp., G'_2) is set to be the cost of its corresponding vertex in G_1 (resp., G_2). Further, note that $\mathcal{N}_2^*$ must be above $\mathcal{N}_1^*$. To ensure this, a set E'_r of directed edges connecting each vertex in G'_1 to the corresponding vertex in G'_2 is introduced. Thus, $V'=V'_1 \cup V'_2$ and $E'=E'_1 \cup E'_2 \cup E'_r$ (see Figure 9 for an example).

Then, given any non-empty closed set $\mathcal{C}$ in G', we can define two net surfaces $\mathcal{N}_1$ and $\mathcal{N}_2$ in G_1 and G_2, as in Section 4.4. $\mathcal{N}_1$ and $\mathcal{N}_2$ have the following property.

Lemma 13—Any non-empty closed set $\mathcal{C}$in G′ specifies two net surfaces $\mathcal{N}_1$ in G_1 and $\mathcal{N}_2$ in G_2 such that $\mathcal{N}_2$ is above $\mathcal{N}_1$ and the cost $w(\mathcal{C})$ of $\mathcal{C}$ is equal to $c(LO(\mathcal{N}_1)) + c(LO(\mathcal{N}_2))$.

By Lemmas 12 and 13, we have the following fact.

Lemma 14—Any non-empty closed set $\mathcal{C}$ in G′ specifies a feasible weakly watershed-monotone region R ⊆ Γ whose total volume is equal to the total cost of $\mathcal{C}$.

Now we show how to construct a closed set $\mathcal{C}$ in G' with the same total cost from a given weakly watershed-monotone region R of Γ. For simplicity, we use $\mathbf{x}(val)$ to denote a voxel $\mathbf{x}(d - 1, val)$ in Γ and $v'^{i}_{\mathbf{x}}(val)$ to denote the corresponding vertex of $\mathbf{x}(d - 1, val)$ in $G'_i(i=1, 2)$.

Lemma 15—For any weakly watershed-monotone region R of Γ, there is a corresponding closed set $\mathcal{C}$ in G′ with its cost $w(\mathcal{C}) = vol(R)$.

<u>**Proof:**</u> For a given weakly watershed-monotone region $R \subseteq \Gamma$, we define a closed set $\mathcal{C}$ in G′, as follows. First, we let $\mathcal{C}_s = \varnothing$. For each $\mathbf{x} \in \mathcal{D}$, if the intersection of R and the voxel set $(\mathbf{x}(0), \mathbf{x}(1),\ldots, \mathbf{x}(N - 1)\}$ (also denote by $Col(\mathbf{x}, d - 1)$) is not empty, say, consisting of voxels $\mathbf{x}(lw), \mathbf{x}(lw + 1), \ldots, \mathbf{x}(up)$, with $0 \le lw \le up < N$, then we put into $\mathcal{C}_s$ the vertex $v'^{2}_{\mathbf{x}}(up)$ of G'_2 and the vertex $v'^{1}_{\mathbf{x}}(lw - 1)$ of G'_1 if $lw > 0$. The sought closed set $\mathcal{C}$ is the closure in G' generated by $\mathcal{C}_s$. We next prove the cost $w(\mathcal{C})$ equals $vol(R)$.

For each $\mathbf{x} \in \mathcal{D}$, we distinguish two cases: $Col(\mathbf{x}, d - 1) \cap R \neq \varnothing$ and $Col(\mathbf{x}, d - 1) \cap R = \varnothing$.

Case (1)—$Col(\mathbf{x}, d - 1) \cap R \neq \varnothing$. In this case, all vertices $v'^{2}_{\mathbf{x}}(k)$ of G'_2 with $0 \le k \le up$ and $v'^{1}_{\mathbf{x}}(k)$ of G'_1 with $0 \le k \le lw$ are included in $\mathcal{C}$. Let $Ch_1(\mathbf{x})$ (resp., $Ch_2(\mathbf{x})$) denote the vertices in G'_1 (resp., G'_2) corresponding to voxels in $Col(\mathbf{x}, d - 1)$. Next, we will show that no vertices $v'^{2}_{\mathbf{x}}(k)$ with $up < k < N$ and $v'^{1}_{\mathbf{x}}(k)$ with $lw \le k < N$ are in $\mathcal{C}$. Otherwise, assume that $v'^{2}_{\mathbf{a}}(t)=v'(a_0, a_1, \ldots, a_{d-2}, t)$ is the "top" vertex of $Ch_2(\mathbf{a})$ in $\mathcal{C}$ (i.e., for any vertex $v'^{2}_{\mathbf{a}}(k) \in Ch_2(\mathbf{a}) \cap C, k \le t$) and $t > up$. The constructions of G' and $\mathcal{C}_s$ indicate that there is no path from any vertex of G'_1 in $\mathcal{C}_s$ to a vertex in G'_2 whose corresponding voxel is in R. Thus, there must be a vertex $v'^{2}_{\mathbf{b}}=v'(b_0, b_1, \ldots, b_{d-1})$ of G'_2 in $\mathcal{C}_s$, from which $v'^{2}_{\mathbf{a}}(t)$ is reachable in G'_2 (i.e., there is a directed path in G'_2 from $v'^{2}_{\mathbf{b}}$ to $v'^{2}_{\mathbf{a}}(t)$) and the $(d - 1)$-th coordinate b_{d-1} of the voxel $\mathbf{b}$ is $\ge t$. By characterizing of the property of G′, we then can claim that $P: v'(b_0, b_1,\ldots, b_{d-2}, b_{d-1}) \rightarrow v'(a_0, b_1, \ldots, b_{d-2}, b_{d-1}) \rightarrow v'(a_0, a_1, b_2, \ldots, b_{d-2}, b_{d-1}) \rightarrow \cdots \rightarrow v'(a_0, a_1, \ldots, a_{d-2}, b_{d-1}) \rightarrow v'(a_0, a_1, \ldots, a_{d-2}, t)$ is a path from $v'^{2}_{\mathbf{b}}$ to $v'^{2}_{\mathbf{a}}(t)$ in G'_2. Moreover, due to the watershed-monotonicity, all corresponding voxels of the vertices on the path P, except the last one $(a_0, a_1, \ldots, a_{d-2}, t)$ (since R is not necessarily watershed-monotone to $\Gamma_{d-1}(c_{d-1})$), are in

R. Thus, $\mathbf{a}(b_{\mathrm{d}-1}) = (a_0, a_1, \ldots, a_{d-2}, b_{d-1}) \in R$, $\mathbf{a}(up) = (a_0, a_1, \ldots, a_{d-2}, up) \in R$, and the voxel $\mathbf{a}(t) = (a_0, a_1, \ldots, a_{d-2}, t)$ is in between $\mathbf{a}(b_{d-1})$ and $\mathbf{a}(up)$ (i.e., $b_{d-1} \geq t > up$). Since R is monotone to x_{d-1}-axis, $\mathbf{a}(t) \in R$, a contradiction. Hence, no vertices $v'^2_{\mathbf{a}}(k)$ with $up < k < N$ are in $\mathcal{C}$. Using a similar argument, we can show that no vertices $v'^1_{\mathbf{a}}(k)$ of G'_1 with $lw \leq k < N$ are in $\mathcal{C}$. Hence, we have $w(Ch_2(\mathbf{x}) \cap \mathcal{C}) + w(Ch_1(\mathbf{x}) \cap \mathcal{C}) = vol(Col(\mathbf{x}, d-1) \cap R)$.

Case (2)—$Col(\mathbf{x}, d-1) \cap R = \varnothing$. Let vertex $v'^1_{\mathbf{x}}(lw)$ be the "top" vertex in $(Ch_1(\mathbf{x}) \cap \mathcal{C})$ (i.e., for any $v'^1_{\mathbf{x}}(k) \in Ch_1(\mathbf{x}) \cap C$, $k \leq lw$). Then, the vertex $v'^2_{\mathbf{x}}(lw)$ of G'_2 is in $\mathcal{C}$. We want to show that for any vertex $v'^2_{\mathbf{x}}(k)$ of $Ch_2(\mathbf{x})$ in $\mathcal{C}$, $k \leq lw$. Assume that in $\mathcal{C}$ there exists a vertex $v'^2_{\mathbf{x}}(t)$ of $Ch_2(\mathbf{x})$ with $t > lw$. Since $Col(\mathbf{x}, d-1) \cap R = \varnothing$, no vertex of G'_2 in $\mathcal{C}$ can reach $v'^2_{\mathbf{x}}(t)$. Hence, there must exist a path in G' from a certain vertex $v'^1_{\mathbf{a}}$ (with the $(d-1)$-th coordinate a_{d-1} of the voxel $\mathbf{a}$ being $\geq t$) of G'_1 in $\mathcal{C}$ to $v'^2_{\mathbf{x}}(t)$. But this implies that the vertex $v'^1_{\mathbf{x}}(t)$ is also in $\mathcal{C}$, a contradiction. Thus, we have $w(Ch_2(\mathbf{x}) \cap \mathcal{C}) + w(Ch_1(\mathbf{x}) \cap \mathcal{C}) = vol(Col(\mathbf{x}, d-1) \cap R) = 0$.

Therefore, $w(\mathcal{C}) = \Sigma_{\mathbf{x} \in \mathcal{D}}\, w(Ch_2(\mathbf{x}) \cap \mathcal{C}) + w(Ch_1(\mathbf{x}) \cap \mathcal{C}) = \Sigma_{\mathbf{x} \in \mathcal{D}}\, vol\,(Col(\mathbf{x}, d-1) \cap R) = vol\,(R)$. Thus, the lemma follows.

Hence, we compute a minimum-cost non-empty closed set $\mathcal{C}^*$ in G', which is used to specify a weakly watershed-monotone region R^* in Γ with the minimum volume. Note that $|V'| = O(n)$ and $|E'| = O(d \cdot n)$. By using the minimum s-t cut algorithm in the paper,[12] we have the following result.

Theorem 4—The optimal weakly watershed-monotone region problem is solvable in $O(dn^2 \log \frac{n}{d})$ time.

6.4. Algorithm for computing an optimal watershed-monotone shell

This section presents our algorithm for solving the d-D optimal watershed-monotone shell problem ($d \geq 3$). We solve this problem by using the same framework as the one for the optimal weakly watershed-monotone region problem. However, the construction of the graph $G' = (V', E')$, which is directly constructed from Γ, is in a slightly different way to incorporate the "shell" structure of the target region R.

The graph G' for this problem contains two disjoint subgraphs $G'_1 = (V'_1, E'_1)$ and $G'_2 = (V'_2, E'_2)$, which are used to search for the inner and outer boundaries, S_I and S_O, of R, respectively. Note that in our algorithm for the optimal weakly watershed-monotone region problem, each subgraph is employed for searching one of the two pieces of the boundary surface of the target region. The geometric constraints between S_O and S_I are enforced by a set of edges between G'_2 and G'_1.

We first show the construction of G'_1. Each voxel $\mathbf{x} \in \Gamma$ corresponds to exactly one vertex $v'^1_{\mathbf{x}} \in V'_1$ with a cost of $-vol(\mathbf{x})$. For a given watershed kernel voxel $\mathbf{c}(c_0, c_1, \ldots, c_{d-1})$ the directed edges are introduced as follows. Consider every voxel $\mathbf{x}$ on each hyperplane $x_i = c_i = 0, 1, \ldots, d-1$). The i-th column $Col(\mathbf{x},i)$, as defined in Section 6.2, is divided into two sub-columns $Col_l(\mathbf{x},i)$ and $Col_u(\mathbf{x},i)$, such that $Col_l(\mathbf{x},i)$ (resp., $Col_u(\mathbf{x},i)$) consists of all voxels in $Col(\mathbf{x},i)$ whose i-th coordinates are no bigger (resp., smaller) than c_i. Then, for every $k = N-1, N-2, \ldots, c_i + 1$ (i.e., $\mathbf{x}(i,k) \in Col_u(\mathbf{x},i)$), the vertex $v'^1_{\mathbf{x}}(i, k)$ has a directed edge to $v'^1_{\mathbf{x}}(i, k-1)$ in E'_1; for every $k = 0, 1, \ldots, c_i - 1$ (i.e., $\mathbf{x}(i,k) \in Col_l(\mathbf{x},i)$), the vertex $v'^1_{\mathbf{x}}(i, k)$ has a directed edge to $v'^1_{\mathbf{x}}(i, k+1)$, as illustrated in Figure 10(a).

The construction of G'_2 is similar to that of G'_1. The only difference is the vertex-cost assignment scheme. In G'_2, each vertex $v'^2_{\mathbf{x}}$ associated with a voxel $\mathbf{x} \in \Gamma$ has a cost of $vol(\mathbf{x})$. Then, for every voxel $\mathbf{x} \in \Gamma$, we put a directed edge from the vertex $v'^1_{\mathbf{x}}$ of G'_1 to the vertex $v'^2_{\mathbf{x}}$ in G'_2.

We then compute a minimum-cost closed set $\mathcal{C}^* \neq \varnothing$ in G'. An optimal watershed-monotone shell in Γ can be specified using R^*, as follows. Initially, let $R^* = \varnothing$. For each voxel $\mathbf{x} \in \Gamma$, if the vertex $v'^2_{\mathbf{x}}$ of G'_2 is in $\mathcal{C}^*$, but $v'^1_{\mathbf{x}}$ of G'_1 is not, then put $\mathbf{x}$ into R^* (see Figure 10(b)). The correctness of our algorithm for this problem follows from a similar argument for the optimal weakly watershed-monotone region problem.

Lemma 16—(1) Any watershed-monotone shell $R \neq \varnothing$ in Γ corresponds to a non-empty closed set $\mathcal{C}$ in G′ whose cost is equal to the volume of R. (2) Any non-empty closed set $\mathcal{C}$ in G′ defines a watershed-monotone shell $R \neq \varnothing$ in Γ whose volume is equal to the cost of $\mathcal{C}$.

Note that the graph G' thus constructed has $2|\Gamma| = 2n$ vertices and $O(dn)$ edges. By using the minimum *s-t* cut algorithm in the paper,[12] we have the following theorem.

Theorem 5—An optimal watershed-monotone shell in a d-D voxel grid Γ ($d \geq 3$) can be computed in $O(dn^2 \log \frac{n}{d})$ time.

7. Implementation and Experiments

To further examine the behavior and performance of our LNS algorithm, we implemented it in standard C++ templates. After the implementation, we extensively experimented with 3-D images on physical phantoms, human pulmonary CT data, vascular MR data, and intravascular ultrasound data, and compared with a previously validated slice-by-slice 2-D segmentation approach based on graph search techniques.[32] Over the years, graph search techniques have become one of the best understood and most utilized 2-D medical image segmentation tools.[22] Our LNS program was tested on an AMD Athlon MP 2000+ Dual CPU workstation with 3.5GB memory running MS Windows XP. In our implementation, we only consider the on-surface cost of voxels.

The experiments showed that our LNS algorithm and software are computationally efficient and produce highly accurate and consistent segmentation results. The average execution times of our simultaneous l-surface ($l = 2,3$) detection algorithm on images of various sizes are shown in Table 1.

An accuracy assessment on images of physical phantom tubes revealed that the overall signed errors for the inner and outer diameters derived from the tube boundaries were (mean ± standard deviation) −0.36 ± 2.47% and −0.08 ± 1.35%, respectively. Figure 11 presents the segmentation results by single surface detection methods versus our LNS algorithm. The synthesized image consists of 3 identical slices stacked together to form a 3-D image. Our LNS approach outperforms both the single surface detection algorithm[30,15] and the 2-D MetaMorphs method.[13] In comparison, the current MetaMorphs implementation is unable to segment the outer contour.

Our LNS approach was tested on segmenting both the inner and outer airway wall surfaces in CT images, in which outer wall surfaces are very difficult to detect due to their blurred and discontinuous appearance and the presence of adjacent blood vessels. The CT images had a nearly isotropic resolution of $0.7 \times 0.7 \times 0.6mm^3$ and consisted of 500 – 600 image slices, with 512 × 512 pixels per slice. The currently used 2-D dynamic programming method works reasonably well for the inner wall segmentation but is unsuitable for the segmentation of the outer airway wall. Our new approach produces good segmentation results for both airway wall surfaces in a robust manner. Comparing to manual tracing on 39 randomly selected slices, our LNS technique yielded signed border positioning errors of −0.01 ± 0.15mm and 0.01 ± 0.17mm for the inner and outer wall surfaces, respectively. Figure 12 gives the comparisons on some airway wall segmentation results with a previous slice-by-slice 2-D graph-search based approach.[32] Typically, by taking advantage of the 3-D coherence information, our LNS method can follow the 3-D airway wall structures correctly even among small and blurred airway segments, while the 2-D method exhibits large perturbations, and is even completely lost on some image slices.

Our algorithm was also used to simultaneously identify four vascular wall surfaces — the lumen–intima surface, internal elastic lamina (IEL), external elastic lamina (EEL), and adventitia in 3-D MR image data (see Figures 1(a) and 13). Our method successfully identified the four specified surfaces in 44 out of all 48 image slices. In comparison with manual tracing, the mean signed surface positioning errors for the lumen, IEL, and EEL boundaries were (mean ± standard deviation) 0.44 ± 0.37 pixel, −0.29 ± 0.34 pixel, and 0.11 ± 0.31 pixel. Comparing to the results obtained by the 2-D graph-search based approach,[32] our method showed higher accuracy and 3-D consistency. Besides, the 2-D approach constantly requires the user to interactively define boundary points for guiding the border detection in difficult locations.

Our 3-D LNS method for intravascular ultrasound (IVUS) image segmentation demonstrated lower surface positioning errors as well as more robust performance indicated by the success rate in comparison to the 2-D graph-search based approach.[32] The IVUS images consisted of 1581 image frames approximately 0.5*mm* apart, with 384 × 384 pixels per frame and in-plane resolution of 0.3 × 0.3mm^2/pixel. Some results are given in Table 2.

We have shown that our LNS technique provides a powerful tool for medical image segmentation. However, with the resolution of medical imaging scanners continuously increasing at a rapid pace, the time complexity of our LNS algorithm may constrain its feasibility in segmentation of large image data sets. As future work, we can apply a hierarchical refinement method to alleviate this problem in practice. Several recent works [4,3,21,34,17,31] have demonstrated that multi-scale image segmentation using graph cuts can produce impressive segmentation results. We first solves the LNS problem on the coarsest level graph and then solve the problem at successive higher resolution but on a narrow banded graph derived from the optimal solution found at the previous coarser graph. This multi-scale approach makes it possible to achieve high quality segmentation results on large data sets with faster speed, thus allowing our LNS technique to be used in a wider range of medical applications.

References

1. Amir, A.; Kashi, R.; Netanyalm, NS. Analyzing quantitative databases: Image is everything. Proc. 27th Int. Conf. Very Large Data Bases (VLDB),; Rome, Italy. 2001. p. 89-98.
2. Asano T, Chen DZ, Katoh N, Tokuyama T. Efficient algorithms for optimization-based image segmentation. Int J Comput Geom Appl 2001;11(2):145–166.
3. Barbu, A.; Zhu, S. Graph partition by Swendsen–Wang cuts. Proc. Int. Conf. Computer Vision; Nice, France. 2003. p. 320-327.
4. Benezit, F.; Cour, T.; Shi, J. Spectral segmentation with multi-scale graph decomposition. Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR); Jun. 2005 p. 1124-1131.
5. Boykov, Y.; Kolmogorov, V. Computing geodesics and minimal surfaces via graph cuts. Proc. Int. Conf. Computer Vision (ICCV); Nice, France. Oct. 2003 p. 26-33.
6. Chakraborty A, Staib H, Duncan J. Deformable boundary finding in medical images by integrating gradient and region information. IEEE Trans Med Imag 1996;15(6):859–870.

7. Chen, DZ.; Chun, J.; Katoh, N.; Tokuyama, T. Efficient algorithms for approximating a multi-dimensional voxel terrain by a unimodal terrain. Proc. 10th Ann. Int. Computing and Combinatorics Conf; Jeju Island, Korea. 2004. p. 238-248.

8. Frank, RJ.; McPherson, DD.; Chandran, KB.; Dove, EL. Computers in Cardiology. IEEE; Los Alamitos: 1996. Optimal surface detection in intravascular ultrasound using multi-dimensional graph search; p. 45-48.

9. Fukuda T, Morimoto Y, Morishita S, Tokuyama T. Data mining with optimized two-dimensional association rules. ACM Trans Datab Syst 2001;26:179–213.

10. Carey, MR.; Johnson, DS. Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman; San Francisco, CA: 1979.

11. Goldenberg R, Kimmel R, Rivlin E, Rudzsky M. Cortex segmentation: A fast variational geometric approach. IEEE Trans Med Imag 2002;21(2):1544–1551.

12. Goldberg AV, Tarjan RE. A new approach to the maximum-flow problem. J Assoc Comput Mach 1988;35:921–940.

13. Huang, X.; Metaxas, D.; Chen, T. MetaMorphs: Deformable shape and texture models. Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR); Jun. 2004 p. 496-503.

14. Hochbaum DS. A new-old algorithm for minimum-cut and maximum-flow in closure graphs. Networks 2001;37(4):171–193.

15. Li, K.; Wu, X.; Chen, DZ.; Sonka, M. Efficient optimal surface detection: Theory, implementation and experimental validation. Proc. SPIE's Int. Symp. Medical Imaging: Imaging Processing; San Diego, CA. 2004. p. 620-627.

16. Li, K.; Wu, X.; Chen, DZ.; Sonka, M. Globally optimal segmentation of interacting surfaces with geometric constraints. Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR); Jun. 2004 p. 394-399.

17. Lombaert, H.; Sun, Y.; Grady, L.; Xu, C. A multilevel banded graph cuts method for fast image segmentation. Proc. 10th IEEE Int. Conf. Computer Vision; Oct. 2005

18. MacDonald D, Kabani N, Avis D, Evans AC. Automated 3-D extraction of inner and outer surfaces of cerebral cortex from MRI. NeuroImage 2000;12(3):340–356. [PubMed: 10944416]

19. Paragios N. A variational approach for the segmentation of the left ventricle in cardiac image analysis. Int J Comput Vis 2002;46(3):223–247.

20. Picard JC. Maximal closure of a graph and applications to combinatorial problems. Manag Sci 1976;22:1268–1272.

21. Sharon, E.; Brandt, A.; Basri, R. Fast multiscale image segmentation. Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR); Jun. 2000 p. 70-77.

22. Sonka, M.; Hlavac, V.; Boyle, R. Image Processing, Analysis, and Machine Vision. 2. Brooks/Cole Publishing Company; Pacific Grove, CA: 1999.

23. Sonka M, Winniford MD, Collins SM. Robust simultaneous detection of coronary borders in complex images. IEEE Trans Medi Imag 1995;14(1):151–161.

24. Sonka M, Zhang X, Siebs M, Bissing MS, DeJong S, Collins SM, McKay CR. Segmentation of intravascular ultrasound images: A knowledge-based approach. IEEE Trans Med Imag 1995;14(4): 719–732.

25. Spreeuwers L, Breeuwer M. Detection of left ventricular epi- and endocardial borders using coupled active contours. Computer Assisted Radiology and Surgery. 2003

26. Thedens DR, Skorton DJ, Fleagle SR. A three-dimensional graph searching technique for cardiac border detection in sequential images and its application to magnetic resonance image data. Computers in Cardiology 1990:57–60.

27. Thedens DR, Skorton DJ, Fleagle SR. Methods of graph searching for border detection in image sequences with applications to cardiac magnetic resonance imaging. IEEE Trans Med Imag 1995;14 (1):42–55.

28. Woeginger GJ. Computing maximum valued region. Acta Cybernetica 1992;10:303–315.

29. Wu, X. Segmenting doughnut-shaped objects in medical images. Proc. 14th Ann. Int. Symp. Algorithms and Computation; 2003. p. 375-384.

30. Wu, X.; Chen, DZ. Optimal net surface problems with applications. Proc. 29th Int. Colloq. Automata, Languages and Programming (ICALP); 2002. p. 1029-1042.

31. Xu, N.; Bansal, R.; Ahuja, N. Object segmentation using graph cuts based active contours. Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR); Jun. 2003 p. 46-53.

32. Yang F, Holzapfel G, Schulze-Bauer C, Stollberger R, Thedens D, Bolinger L, Stolpen A, Sonka M. Segmentation of wall and plaque in in-vitro vascular MR image. Int J Cardiovasc Imag 2003;19(5): 419–428.

33. Yezzi, A.; Tsai, A.; Willsky, A. A statistical approach to snakes for bimodal and trimodal imagery. Proc. IEEE Int. Conf. Computer Vision; Corfu, Greece. 1999. p. 898-903.

34. Yu, SX. Segmentation using multiscale cues. Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR); Jun. 2004 p. 70-77.

35. Zeng X, Staib LH, Schultz RT, Duncan JS. Segmentation and measurement of the cortex from 3-D MR images using coupled surfaces propagation. IEEE Trans Med Imag 1999;18:927–937.

36. Zhu S, Yuille A. Region competition: Unifying snakes, region growing, and Bayes/MDL for multiband images segmentation. IEEE Trans Patt Anal Mach Intell 1996;18:884–900.

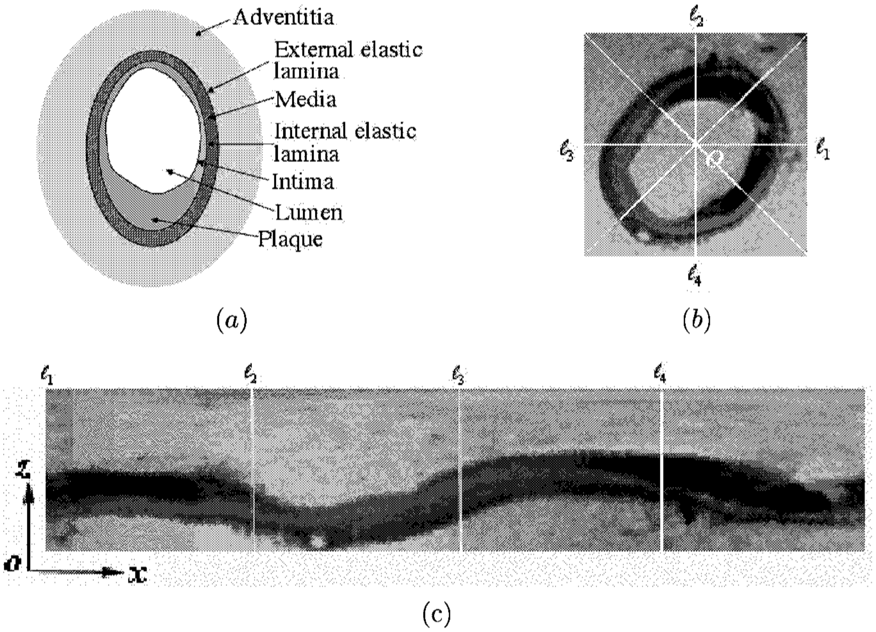

Fig. 1.

Illustrating the "unfolding" operations on the transversal cross-sections of vascular MR images of a human femoral artery specimen. (a) A schematic cross-sectional anatomy of a diseased artery. (b) Performing a polar resampling. (c) Each transversal cross-section is embedded as an **xz**-slice in the 3-D **xyz**-space.

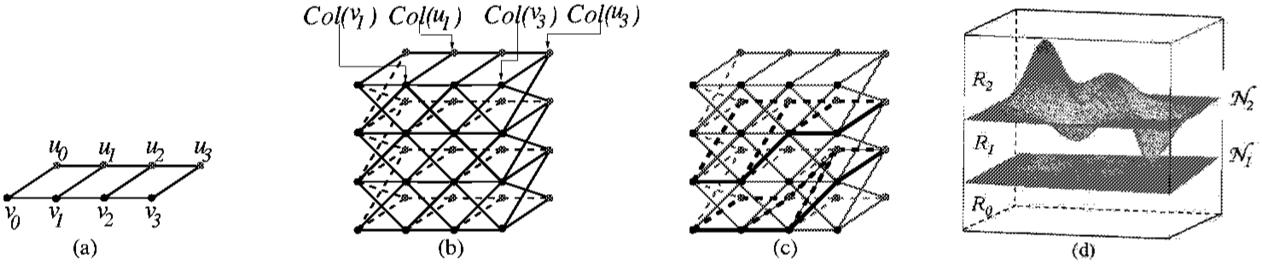

Fig. 2.

(a) A 2-D net model B. (b) A 3-D properly ordered multi-column graph G generated by B and $\kappa = 4$ (the edges between $Col(u_i)$ and $Col(u_{i+1})$, $i = 0,1, 2$, are symmetric to those between $Col(v_i)$ and $Col(v_{i+1})$, and the edges between $Col(v_j)$ and $Col(u_j)$, $j = 1, 2$, are symmetric to those between $Col(v_3)$ and $Col(u_3)$; all these edges are omitted for a better readability). (c) Two (l, 2)-separate net surfaces in G marked by heavy edges. (d) Two net surfaces divide the vertex set of G into three disjoint vertex subsets.

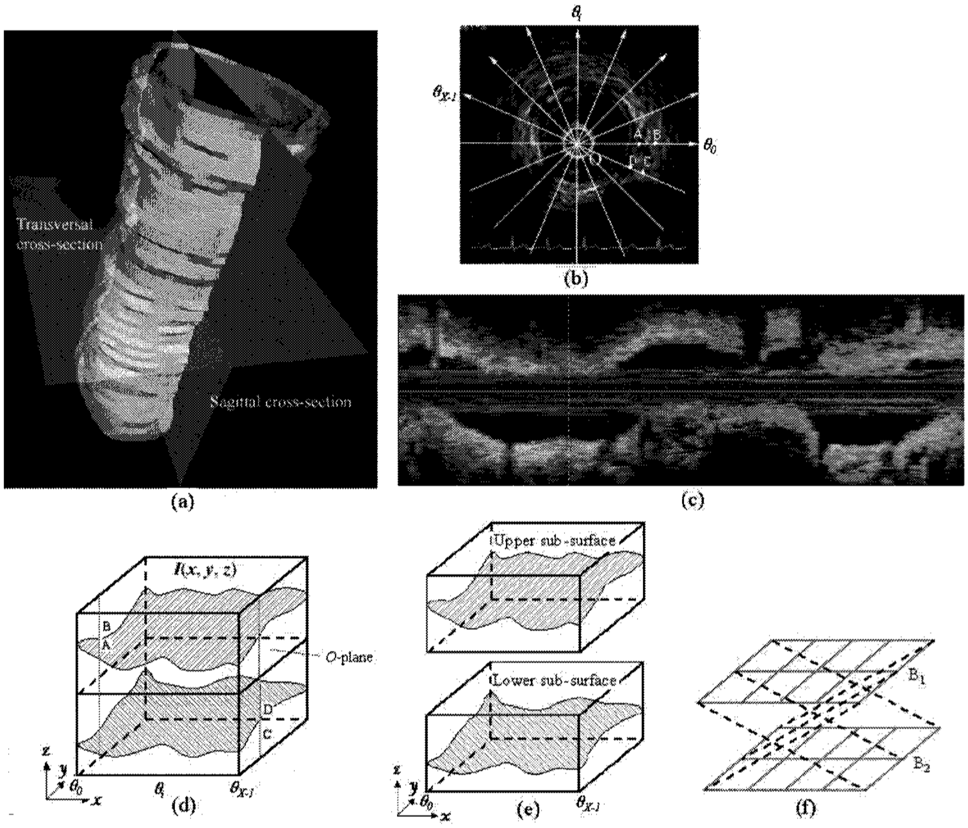

Fig. 3.

Illustrating the sagittal cross-section based unfolding method, (a) The sagittal and transversal cross-sections of a tubular object, (b) A transversal cross-section of a 3-D intravascular ultrasound vessel image. Resampling is performed at each angle for all transversal cross-sections to form a sagittal cross-section, (c) A sagittal cross-section of a 3-D intravascular ultrasound image at an angle θ_i. (d) A schematic unfolded 3-D image $\mathcal{I}(\mathbf{x}, \mathbf{y}, \mathbf{z})$. Each sagittal cross-section at an angle θ_i is embedded as a **yz**-slice of $\mathcal{I}$ at $x = \theta_i$. (e) Splitting $\mathcal{I}$ into two sub-images $\mathcal{I}_1$ and $\mathcal{I}_2$. (f) The net model B for $\mathcal{I}$.

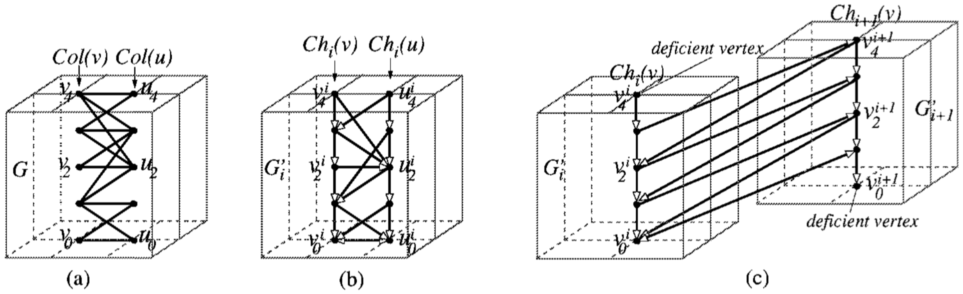

Fig. 4.

(a) Illustrating the proper ordering of the edges in *G*. (b) Constructing a subgraph G'_i from *G*. (c) Incorporating the surface separation constraints into the construction of G'_i and G'_{i+1} (with *Li* = 1 and U_i = 2).

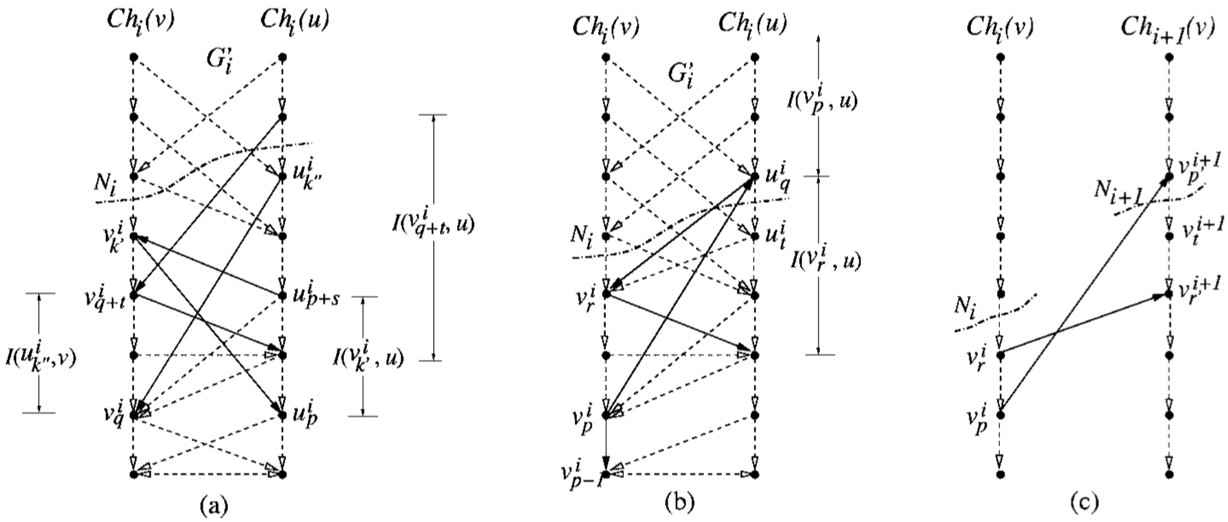

Fig. 5.

(a) Illustrating the proof of Lemma 3. (b) Illustrating Case (2) in the proof of Lemma 6. (c) Illustrating Case (3) in the proof of Lemma 6.

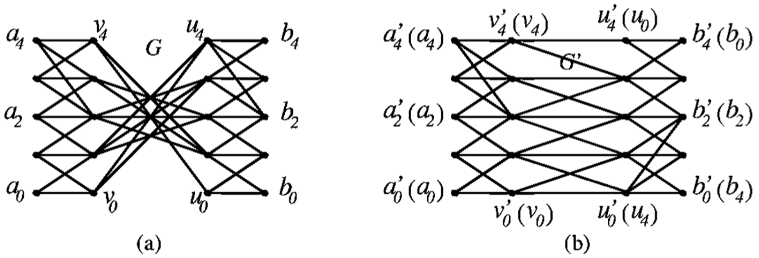

Fig. 6.

(a) Illustrating the reverse ordering of G. Columns $Col(v)$ and $Col(u)$ are in reverse order, (b) Constructing a properly ordered graph G' from G (assume that $a, v \in Q$, $u, b \in \bar{Q}$, and $Q \cup \bar{Q} = V_B$). Each vertex in a parenthesis is a corresponding vertex in G.

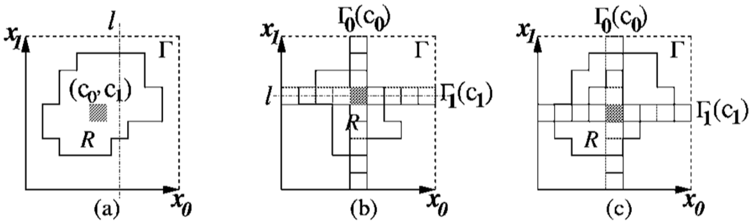

Fig. 7.

(a) A watershed-monotone region, (b) A weakly watershed-monotone region which is not watershed-monotone to any $\Gamma_1(c)$. (c) A watershed-monotone shell. (For a better readability, a 2-D pixel grid Γ is used.)

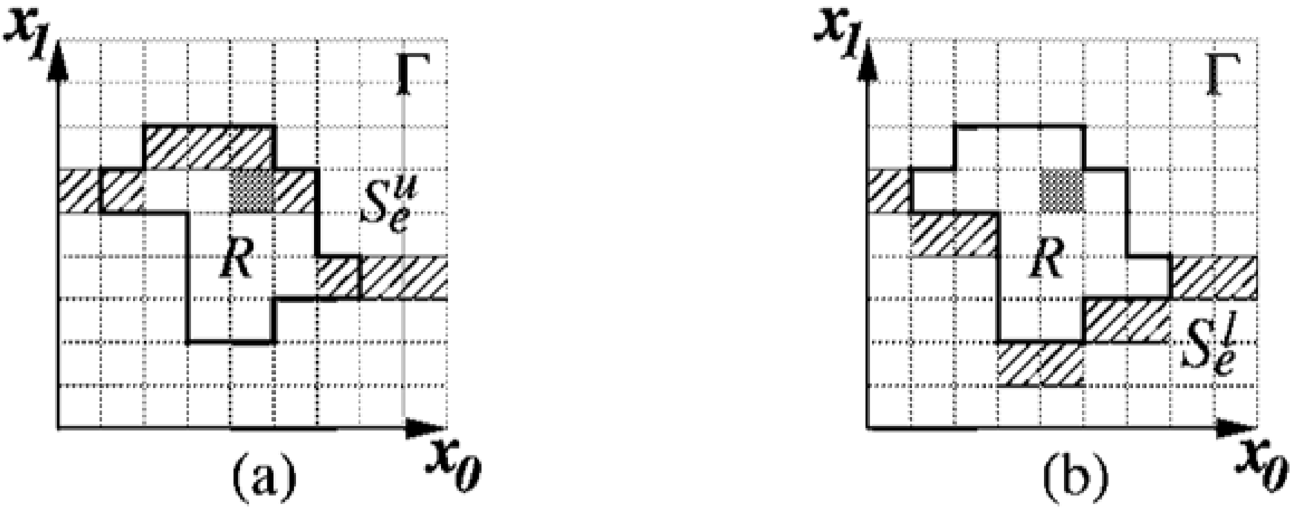

Fig. 8.

(a) The extended "upper" boundary surface S_e^u of a weakly watershed-monotone region R (consisting of the shaded pixels). (b) The extended "lower" boundary surface S_e^l of R.

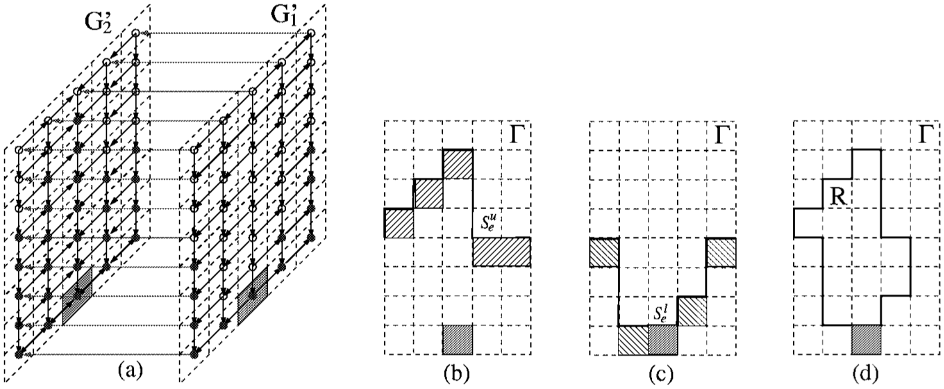

Fig. 9.

Computing an optimal weakly watershed-monotone region *R*. For a better readability, a 2-D pixel grid Γ is used, (a) Illustrating the construction of the graph G′. The shaded pixel is the given weak watershed kernel pixel **c**. Only a portion of the directed edges from G'_1 to G'_2 is shown. The solid vertices shown make up of a closed set $\mathcal{C}$ in G′. (b) The extended "upper" surface S_e^u defined by the vertices of G'_2 in $\mathcal{C}$. (c) The extended "lower" surface S_e^l defined by the vertices of G'_1 in $\mathcal{C}$. (d) A weakly watershed-monotone region *R* corresponding to the closed set $\mathcal{C}$.

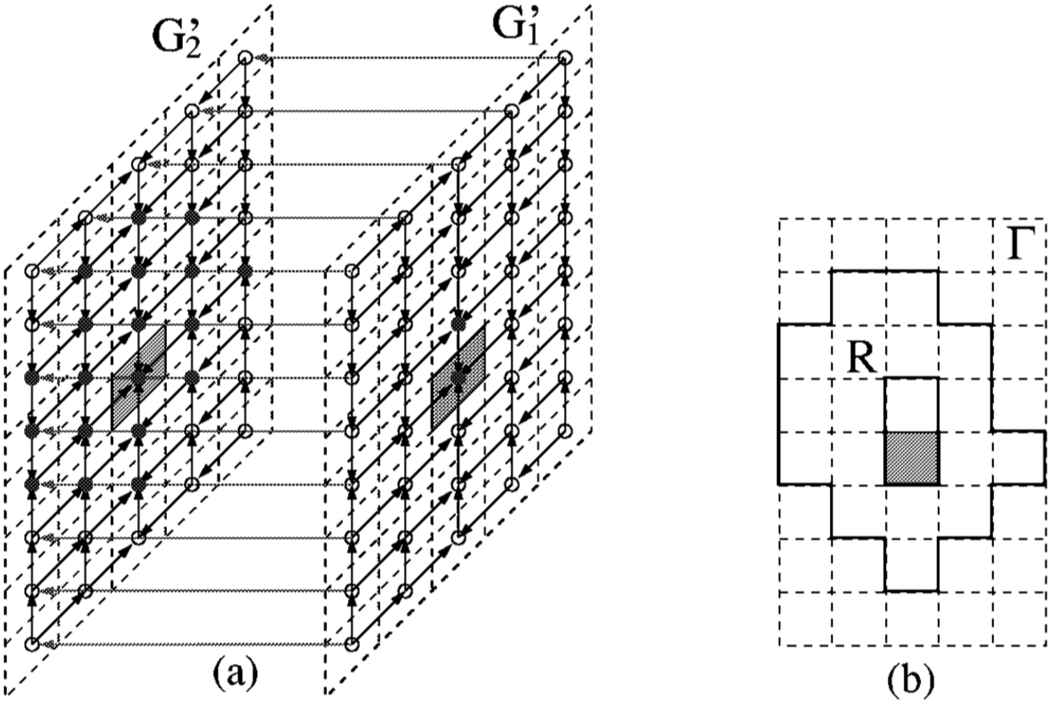

Fig. 10.

Computing an optimal watershed-monotone shell. For a better readability, a 2-D pixel grid Γ is used, (a) Illustrating the construction of the graph G'. The shaded pixel is the given watershed kernel pixel **c**. Only a portion of the edges from G'_1 to G'_2 is shown, (b) A watershed-monotone shell R corresponding to the closed set in G' consisting of all the solid vertices in (a).

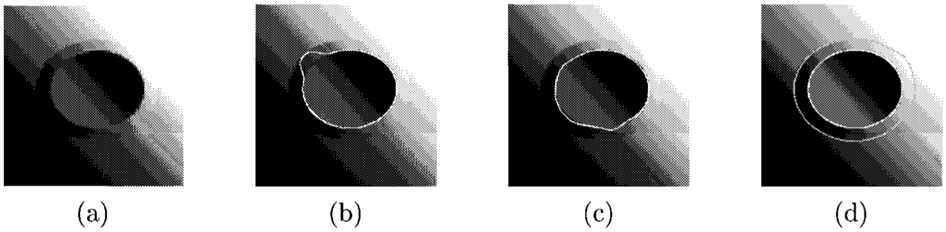

Fig. 11.

Single-surface methods versus our inter-related surface method, (a) Cross-section of the original synthesized image. (b) Single surface detection30,15 using a standard edge-based cost function, (c) MetaMorphs method13 segmenting the inner border in 2-D. (d) Double-surface segmentation obtained by our LNS approach.

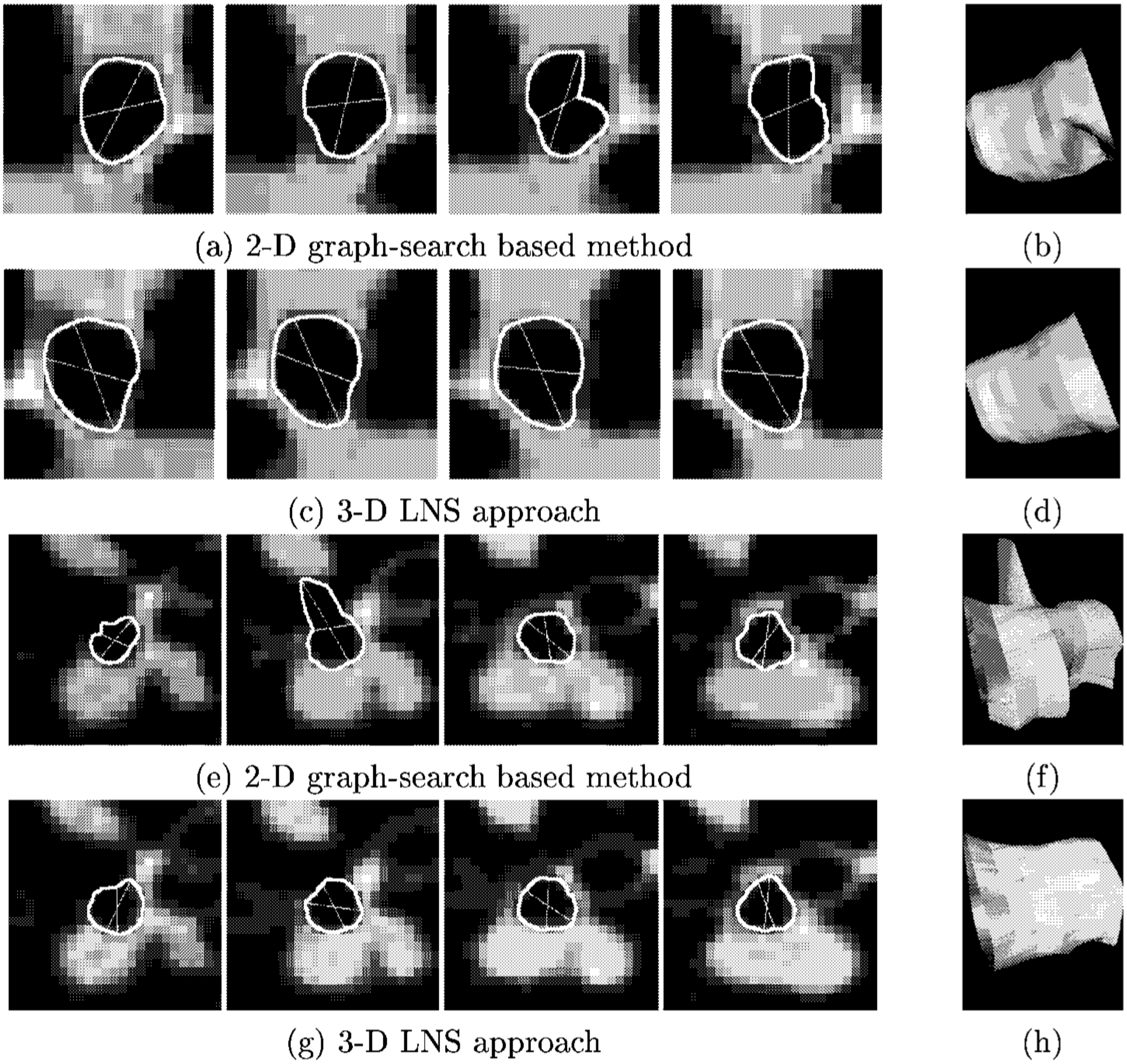
(a) 2-D graph-search based method
(b)
(c) 3-D LNS approach
(d)
(e) 2-D graph-search based method
(f)
(g) 3-D LNS approach
(h)

Fig. 12.

Comparisons on airway wall segmentation results. For a better readability of the 3-D surface renderings, only the single luminal surface results are shown. (a), (b), (e), and (f) are the results yielded by the slice-by-slice 2-D graph-search based approach on two different airway segments. Four consecutive slices and the 3-D surface rendering are shown for each airway segment (10 slices). (c), (d), (g), and (h) are the walls segmented by our 3-D LNS approach.

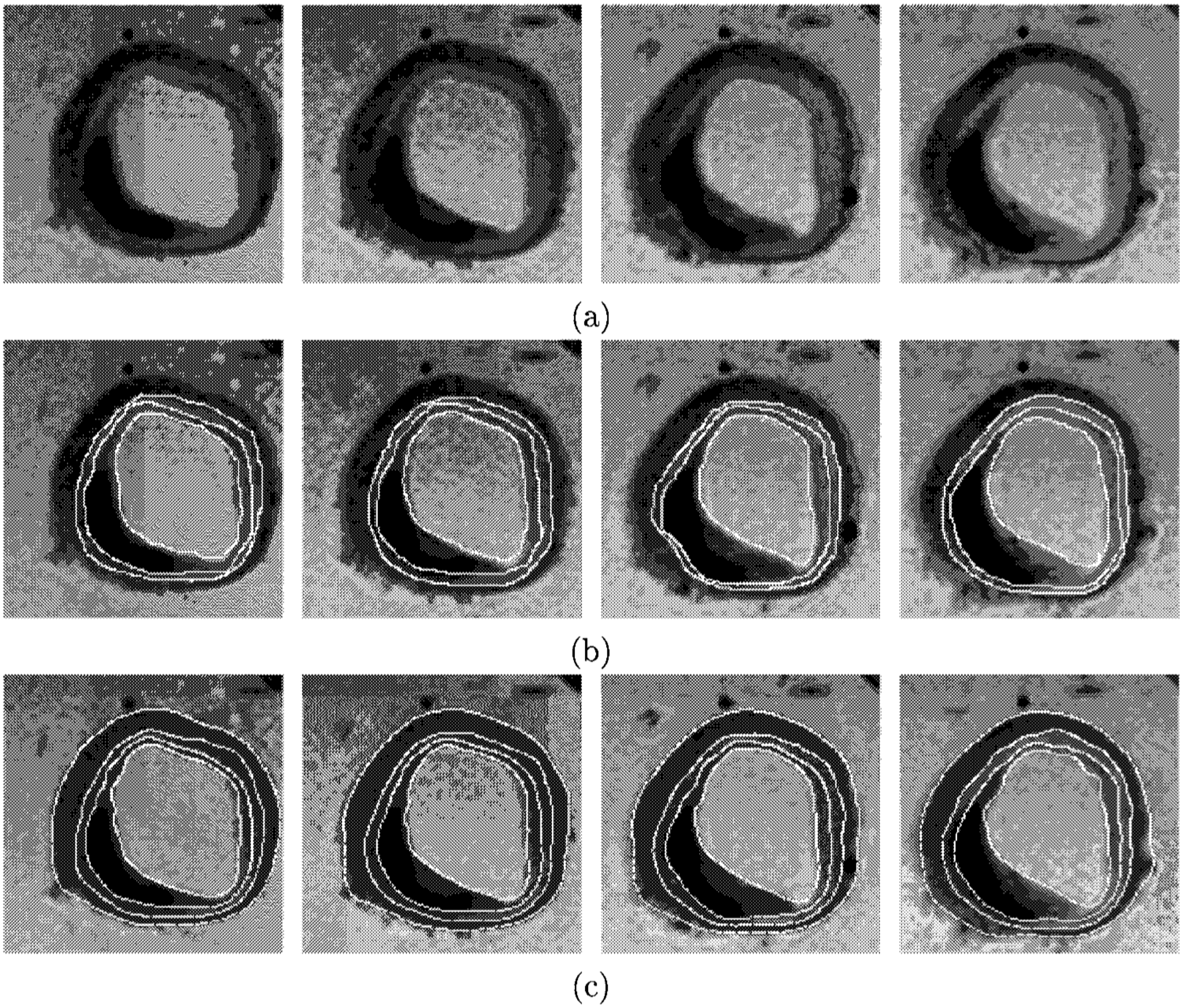

Fig. 13.
Results on MR vascular wall surface segmentation. (a) Four consecutive slices from the original image data. (b) Manually identified lumen, IEL, and EEL surfaces. The outermost adventitia surfaces are not shown. (c) Results produced by the program for our LNS algorithm.

Table 1

Average execution times (in seconds). Here (l denotes the number of sought surfaces.

Image Size	60^2×40	80^2×40	100^2×40	140^2×40	200^2×40
$l = 2$	3.3	6.1	16.2	85.8	376.1
$l = 3$	4.8	9.4	22.3	96.1	401.3

Table 2

Comparison results on intravascular ultrasound image segmentation between our 3-D LNS technique and the 2-D graph search approach[32].

Method	Success Rate	Positioning Error	Maximum Error
2-D graph search	68%	0.13±0.08mm	2.1mm
3-D LNS	82%	0.09±0.03mm	1.9mm