Surface Detection using Round Cut

Vedrana A. Dahl, Anders B. Dahl and Rasmus Larsen
Department of Applied Mathematics and Computer Science
Technical University of Denmark
Lyngby, Denmark
{*vand,abda,rlar*}*@dtu.dk*

***Abstract*—We propose an iterative method for detecting closed surfaces in a volumetric data, where an optimal search is performed in a graph build upon a triangular mesh. Our approach is based on previous techniques for detecting an optimal terrain-like or tubular surface employing a regular grid. Unlike similar adaptations for triangle meshes, our method is capable of capturing complex geometries by iteratively refining the surface, where we obtain a high level of robustness by applying explicit mesh processing to intermediate results. Our method uses on-surface data support, but it also exploits data information about the region inside and outside the surface. This provides additional robustness to the algorithm. We demonstrate the capabilities of the approach by detecting surfaces of CT scanned objects.**

***Keywords*-image processing; image analysis; object detection;**

I. Introduction

Volumetric surface detection is important for a range of applications including computer assisted diagnosis in medical image analysis and measuring structures in material science. Surface detection techniques based on graph cut have shown very good results by providing an optimal solution and having low computational complexity. In this paper we extend the optimal net surface detection originally suggested by Wu and Chen [1]. We model the surface using a triangular mesh and employ mesh processing techniques to improve robustness of the approach.

Optimal net surface detection via a graph search was first used for finding terrain-like and tubular surfaces [2]. Such surfaces are well represented using a rectangular grid which also naturally provides a necessary neighbourhood structure for the search lines. In the case of terrain-like surfaces a search is performed along vertical lines, and for tubular surfaces the search is in radial direction. The graph is built on a set of samples along the search lines in a way that limits the roughness of the solution. The optimality of the solution is defined in terms of a volumetric cost function derived from the data.

A grid-based representation is however not suited for modeling other geometries like spherical objects, because the grid deforms when projected to a sphere. To detect spherical objects a search can be performed along lines from a single seed point through the vertices of a regular polyhedron. Such an approach, suggested by Egger et al. [3] and Wang and Beichel [4], allows for finding boundaries of objects in the so-called *star domain* which might be sufficient in many applications [5].

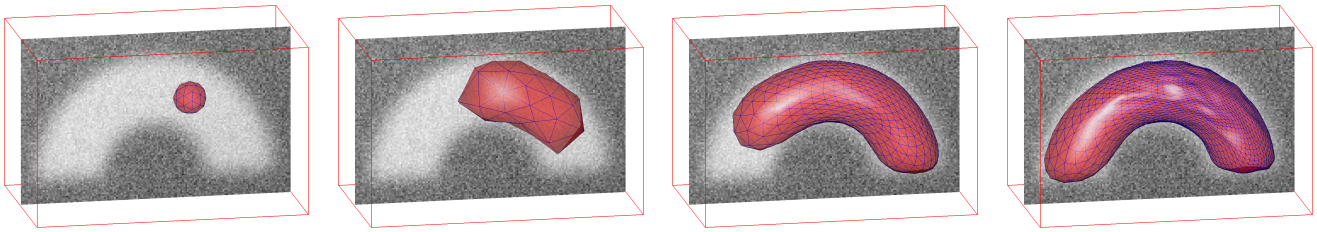
A mesh based technique that utilizes a rough initial surface has been suggested in Li et al. [6] and similar methods have been used in [7], [8], [9]. The search lines in those approaches are inferred from the initial surface, which might lead to problems in the case of noisy data. To overcome this problem curved search lines have been proposed in [10], [11], [12].

Net surface detection has seen a growing use within medical image segmentation [13], [14] where it has been further extended in a number of ways. This includes multiple interrelated surfaces that can be detected simultaneously [2]. The cost function, originally defined only in terms of on-surface costs, has also been extended to incorporate regional information [15], [16], [7]. In addition, the roughness has been allowed to vary across the surface [17], and soft shape constraints have been introduced [18].

The Round Cut algorithm is a novel approach which combines explicit mesh processing with a layered net surface detection. Our approach incorporates the following steps: (*a*) a polyhedron based initialization, (*b*) a layered surface detection with on-surface and in-region cost functions, and (*c*) an explicit mesh processing including mesh subdivision and mesh smoothing. Explicit mesh processing makes it possible to use our method for an iterative surface refinement by repeating steps (*b*) and (*c*). Consequently, our method differs from similar approaches in being able to detect details and objects beyond the star domain (i.e. surfaces that cannot be reached by rays from a seed point) even though the mesh is initiated at a point and not close to the surface, see Fig. 1.

II. Method

Our method extends previous work on the optimal net surface problem using graph search originally suggested by Wu and Chen [1]. To make this paper self-contained we first review the graph construction for finding an optimal solution to the net surface problem and a few relevant extensions of the original algorithm in Section II-A. Then in Section II-B we describe how mesh-based surface search is performed and we explain the elements of our proposed iterative search refinement.



(a) (b) (c) (d)

Figure 1. Example of the iterative surface detection. (a) the mesh is initialized as a polyhedron and in (b-d) the boundary iteratively adapts to the data.

A. Optimal Surface Search

In the original formulation by Wu and Chen [1] the *optimal net surface* problem in 3D is described as follows. Given discrete volumetric data $I(x,y,z)$, $x \in X = \{1,\dots,x_{\max}\}$, $y \in Y = \{1,\dots,y_{\max}\}$, $z \in Z = \{1,\dots,z_{\max}\}$ we consider terrain-like surfaces $S : X \times Y \to Z$, i.e. surfaces which intersect each (x,y)-column of voxels exactly once at a voxel of height $S(x,y)$. To limit surface roughness, a constraint is imposed such that the height at neighbouring columns may not change more than a certain *inter column shift*

$$|S(x,y) - S(x-1,y)| \le \Delta_x, \quad x \neq 1 \tag{1}$$

and

$$|S(x,y) - S(x,y-1)| \le \Delta_y \quad y \neq 1 \quad . \tag{2}$$

Voxel intensities are used to define an on-surface cost function $c_{\text{on}}(x,y,z)$ which takes a small value where data supports the surface. Hereby the total cost of the surface is the cost of all its voxels

$$c_{\text{on}}(S) = \sum_x \sum_y c_{\text{on}}(x,y,S(x,y)) \quad , \tag{3}$$

and an optimal net surface is the surface with the minimum-cost among all terrain-like surfaces satisfying the shift constraint. The polynomial time solution presented in [1] transforms the optimal net surface problem into that of finding a *minimum-cost closed set* in a node-weighted directed graph[1]. This is transformed into a problem of finding a *minimum-cost s-t cut* in a related arc-weighted directed graph [19]. We start by describing the *latter* of the two transformations.

1) Minimum closed set to minimum graph cut: The minimum-cost s-t cut algorithm of Boykov and Kolmogorov [19] is a well known tool and has been used in many image segmentation tasks. The optimal net surface problem presented here is also ultimately solved using the minimum-cost s-t cut algorithm, but with a very different graph representation than that in [19]. In the following paragraph we describe and exemplify how a minimum cost closed set problem in a node-weighted graph can be transformed to a minimum cost s-t graph cut problem. In the rest of this paper we will explain the surface detection problem only in terms of minimum-cost closed set, and will not look back at the details of finding the ultimate solution using s-t cut.

A set of nodes in a directed graph with no arcs towards the rest of a graph is called a *closed set*. In a node-weighted graph $G = (N, A)$ consisting of weighted nodes N and directed arcs A, a cost of a set is the total cost of its nodes. The *minimum-cost* closed set has minimal possible cost of all closed sets [1]. To transform a minimum-cost closed set problem to minimum-cost s-t cut problem, a related arc-weighted graph $\tilde{G} = (\tilde{N}, \tilde{A})$ is constructed as follows. Terminal nodes are added to G

$$\tilde{N} = N \cup \{s,t\} \quad , \tag{4}$$

and the arc set is extended with terminal arcs

$$\tilde{A} = A \cup A_{\text{st}} \quad . \tag{5}$$

The source is linked to all nodes with negative weight, while all nodes with nonnegative weight are linked with the sink

$$A_{\text{st}} = \{\langle s,n\rangle,\ w(n) < 0\} \ \cup\ \{\langle n,t\rangle,\ w(n) \ge 0\} \quad , \tag{6}$$

where w is a weight of a node in $n \in N$. Each terminal arc is assigned a weight which is an absolute value of the weight of the involved node, while the internal arcs are assigned infinite weight

$$\tilde{w}(a) = \begin{cases} \infty & a \in A \\ |w(n)| & a = \langle s,n\rangle \text{ or } a = \langle n,t\rangle \end{cases} \quad . \tag{7}$$

It can be shown [1] that the source set of the solution for the minimum-cost s-t problem in $\tilde{G}$ is a minimum closed set of G.

2) Optimal surface to minimal closed set: When transforming the optimal net surface problem into a minimum-cost closed set problem, a graph is constructed on top of the volumetric data. The feasibility of the surface is handled by the graph arcs, while the optimality of the solution is ensured by the node weights. The idea is to create a correspondence between a surface and a voxel set below the surface. The fundamentals of the approach are explained below while the details can be found in [1], [2].

[1]We will use the term *node* and *arc* for the base entities of the graph, and will reserve the terms *vertex* and *edge* for the entities of a triangle mesh.

We will describe the construction of a graph $G = (N, A)$ containing weighted node set N and directed arc set A. Graph nodes represent the volume voxels, so we write $N = \{n(x, y, z)\}$ for the nodes and $w(x, y, z)$ for the corresponding node weights. We assume $x \in X, \quad y \in Y, \quad z \in Z$, unless otherwise stated. An arc from the node n to the node n' is denoted as $\langle n, n' \rangle$.

The nodes of the lowest possible surface $n(x, y, 1)$, called *the base set*, are connected in a grid-like manner by arcs A_B, running in both directions and drawn from each node to its (up to) four neighbours

$$A_B = \{\langle n(x, y, 1),\, n(x', y', 1)\rangle\,,\ |x - x'| + |y - y'| = 1\} \quad . \tag{8}$$

To guarantee that the base set always is included in a minimum closed set, the cost of the base set should be negative. This can be obtained by assigning an arbitrary negative weight, e.g. $w(x, y, 1) = -1$, to all base nodes.

The nodes of each (x, y) column are linked by directed arcs pointing downwards

$$A_C = \{\langle n(x, y, z),\, n(x, y, z - 1)\rangle\,,\ z \neq 1\} \quad . \tag{9}$$

Those *intra-column* arcs, linking each column in a path pointing at the base set, ensure that the surface found as an upper envelope of a minimum closed set is a terrain-like surface. The shift constraint is enforced by adding slanted *inter-column* arcs

$$\begin{aligned} A_S = &\{\langle n(x, y, z),\, n(x', y, z - \Delta_x)\rangle\,, \\ &\qquad |x - x'| = 1, \quad z > \Delta_x\} \quad \cup \\ &\{\langle n(x, y, z),\, n(x, y', z - \Delta_y)\rangle\,, \\ &\qquad |y - y'| = 1, \quad z > \Delta_y\} \quad , \end{aligned} \tag{10}$$

which completes the arc set $A = A_B \cup A_C \cup A_S \quad .$

Finally, a cost

$$w(x, y, z) = c_{\mathrm{on}}(x, y, z) - c_{\mathrm{on}}(x, y, z - 1),\ z \neq 1 \tag{11}$$

is assigned to all nodes outside the base set. Due to cancelations, the w-cost of each closed set in a graph is equal (up to a constant, because of the arbitrary weight of the base set) to the c-cost of its upper envelope.

3) Layered surfaces: The optimal net surface approach is readily extended to multiple interrelated surfaces [2], allowing the detection of layered surfaces. The problem is now to find $k_{\max}$ surfaces $S_k(x, y), \quad k \in K = \{1, \ldots k_{\max}\}$ where, for example, the overlap between surface 2 and surface 3 is constrained by

$$\delta_{\mathrm{l}} \leq S_3(x, y) - S_2(x, y) \leq \delta_{\mathrm{u}} \quad . \tag{12}$$

Each of the $k_{\max}$ surfaces to be detected contributes with a sub-graph dedicated to a single surface, and constructed as described above. When put together, this results in a node set $N = \{n(x, y, z, k), \quad k \in K\}$. The input to a layered surface detection problem is a 4D cost $c_{\mathrm{on}}(x, y, z, k)$, taking small values where a volume data $I(x, y, z)$ supports the surface k.

To incorporate overlap constraints, sub-graphs are then connected by *inter-surface* arcs in such a way that the desired distance between the surfaces is ensured. For example, if an overlap between surface 2 and surface 3 is constrained by (12) the following set is needed

$$\begin{aligned} A_{L23} = &\{\langle n(x, y, z - \delta_l, 2),\, n(x, y, z, 3)\rangle\,, z > \delta_l\} \quad \cup \\ &\{\langle n(x, y, z, 3),\, n(x, y, z - \delta_u, 2)\rangle\,, z > \delta_u\} \ . \end{aligned} \tag{13}$$

Inter surface constraints can be used to prevent surface overlap, or to limit the distance between surfaces.

4) Tubular surfaces: Tubular surfaces can also be detected by unfolding the volume using cylindrical coordinate transform. In this case the graph columns correspond to the rays of radial samples in a volumetric data where each radial ray has four neighbours, two axial and two tangential. The inter column shift constraint is defined for both directions, and additional intra-column arcs have to be placed in a graph between the first and the last radial plane. So again, the base surface is a regular grid wrapped around the cylinder, and the optimal surface is found by adjusting the position of the grid vertices along the radial ray.

5) In-region cost: The cost function, originally pertaining only to where the data supports the surface, has been extended to incorporate *in-region cost*, i.e. knowledge about the region above, under or between the surfaces to be detected [15], [16]. This applies to $k_{\max}$ non-overlapping and ordered surfaces dividing the volume in $(k_{\max} + 1)$ regions, with the volume boundaries defining the first and the last region. The in-region cost is given by $c_{\mathrm{in}}(x, y, z, k), \quad k \in \{1, \ldots, k_{\max} + 1\}$ taking a small value where the volume data $I(x, y, z)$ supports the region above $k - 1$ and below k. The bottom and the top of the volume are used here as the 0-th and the $(k_{\max} + 1)$-th surface.

Using in-region costs only affects weights of the graph nodes, which are now $w_{\mathrm{in}}(x, y, z, k) = c_{\mathrm{in}}(x, y, z, k) - c_{\mathrm{in}}(x, y, z, k+1)$. To verify that this construction guarantees optimality consider a specific solution $S_k(x, y)$. Moving one surface $\hat{k}$ in one column $(\hat{x}, \hat{y})$ from its position $\hat{z} = S_{\hat{k}}(\hat{x}, \hat{y})$ to a one voxel higher position $\hat{z} + 1$ would move the voxel at position $(\hat{x}, \hat{y}, \hat{z} + 1)$ from a region $\hat{k} + 1$ to a region $\hat{k}$. This move would change the in-region cost of the entire solution with $c_{\mathrm{in}}(\hat{x}, \hat{y}, \hat{z}+1, \hat{k}) - c_{\mathrm{in}}(\hat{x}, \hat{y}, \hat{z}+1, \hat{k}+1)$ which is only favorable (i.e. negative) if the data supports the moved voxel being in region $\hat{k}$ stronger than in region $\hat{k}+1$. On-surface and in-region costs can be used separately or combined as a weighted sum.

B. Mesh Based Surface Search

It would be possible to detect surfaces of spherically or elliptically shaped objects, using a spherical transform. However, choosing suitable inter column shift constraints would be problematic, due to the big distortion of the grid

in polar regions. Instead, a mesh defining a more regular tiling of a sphere should be used as a base surface in this case. In [3] and [4] a polyhedron to set up a graph and surface is detected in a single optimization step.

Essentially, the detection of terrain-like, tubular or spherical surfaces are special cases of a general framework based on any type of meshed surface. The search for the optimal surface is performed along the graph columns, which correspond to rays trough mesh vertices. Mesh edges provide a neighborhood structure when drawing intra-column edges needed to enforce the shift constraint. Not only does this allow us to initialize the algorithm for terrain-like, tubular or spherical surfaces, but it enables taking an arbitrary meshed surface, for example an initial rough object segmentation, as an input. The approach with pre-segmentation has been pursued in a number of applications [6], [8].

Still, this more general framework allows for another extension, which has not been reported before. Having triangle mesh as both input and output of an optimization step we can iteratively refine the surface. Refinement can be both in terms of mesh connectivity (inserting additional vertices to the mesh) or in terms of mesh geometry (moving mesh vertices closer to the surface). In particular, iterative approach allows us to detect objects beyond the star domain.

The method we implemented can detect layered closed surfaces while incorporating in-region costs. For this we use polyhedra based initialization from a given seed point. As an original approach, we can iteratively refine the optimal search, which allows us to detect surfaces beyond the star domain. A number of issues needs to be considered when using the iterative method since the lack of smoothness and the irregularity of the triangle mesh might compromise the robustness of the method. To circumvent those issues we, as another novel contribution, incorporate mesh processing steps in the iterative approach.

1) Polyhedron construction: The base surface for our algorithm is a triangle mesh on the unit sphere placed at the desired seed point. To define the base surface mesh we mimic the construction of a geodesic dome, starting with the regular icosahedron, a polyhedron with $F_0 = 20$ equilateral triangular faces and $V_0 = 12$ vertices. Finer resolution is obtained by dividing icosahedron faces in a desired geodesic dome frequency ν. This frequency represents a number of equal-length segments into which each icosahedron edge is divided, resulting in a division of each icosahedron face into ν^2 smaller equilateral triangular faces. Vertices added in the process are then projected out on the sphere. This produces a high quality triangle mesh on the unit sphere with all the faces being roughly equilateral. It is also a very regular mesh, with only 12 extraordinary points, since all but the original icosahedron vertices have valency 6. The resulting number of mesh vertices is $V = V_0 + \frac{F_0}{2}(\nu+1)(\nu-1)$, and even though V grows quadratically with ν, this allows us to choose suitable polyhedra resolution, unlike e.g. [3], where only two resolution settings are used.

2) Mesh based graph construction: In its general form, a graph search algorithm is described as follows. The input to the algorithm is an arbitrary meshed surface $M = (V, F)$, with a set of vertices $V = \{v_i, \quad i \in \{1, \ldots, n\}\}$ and a set of faces F. Faces imply a neighborhood relation for vertices, and we write $i \sim i'$ if vertices v_i and $v_{i'}$ belong to a face from F. A sequence of numbers $\mathbf{q} = (q_1, \ldots, q_m)$ defining sampling distances in the normal direction, typically equally spaced and symmetric around zero, is also required. Furthermore, an inter column shift constraint for each surface, and the overlap constraint between surfaces are needed.

Spatial sampling is performed along the vertex normals at the distances $\mathbf{q}$

$$U(i,j) = I(\mathbf{v}_i + q_j \mathbf{n}_i) \quad , \tag{14}$$

where $\mathbf{v}_i$ are spatial coordinates of mesh vertices, and $\mathbf{n}_i$ are normals of mesh vertices. Typically a trilinear interpolation is used to obtain voxel values outside the lattice points. The resulting *column-image* is a collection of normal samples, and each mesh vertex contributes with one image column.

In general we are not directly interested in voxel intensities $I(x, y, z)$ but will use them to calculate the on-surface and in-region cost functions. Depending on the problem at hand two approaches are possible. We can sample in the volumetric data and compute the costs based on the column image U. Alternatively, we can pre-calculate volumetric on-surface and/or in-region costs from data, and preform sampling similar to (14) in the cost volumes. The former approach is preferable when defining a cost function in terms of the gradient in the normal direction.

A graph $G = (N, A)$ needs to be constructed on top of the column-image U. We describe graph construction for a single surface. Graph nodes represent column-image pixels, so now we write $N = \{n(i,j)\}, \quad i \in \{1, \ldots, n\}, \quad j \in \{1, \ldots, m\}$. Intra-column arcs are drawn pointing downwards in U (inwards in I)

$$A_C = \{\langle n(i,j),\, n(i,j-1)\rangle\,, \quad j \neq 1\} \tag{15}$$

ultimately pointing at the base surface which consists of the nodes corresponding to the lowest (innermost) sample in each column. When drawing arcs between the columns, we utilize mesh vertex neighborhoods, so base surface nodes are connected by

$$A_B = \{\langle n(i,1),\, n(i',1)\rangle\,, \quad i \sim i'\} \quad . \tag{16}$$

In a similar manner, inter-column arcs are drawn by utilizing the neighborhood information

$$A_S = \{\langle n(i,j),\, n(i',j-\Delta)\rangle\,,\ i \sim i',\ j > \Delta\}\ , \tag{17}$$

where Δ is the inter column shift constant. Unlike the regular grid setting where we distinguish between Δ_x and Δ_y, a meshed surface has no intrinsic directions so now we have a *single* shift constraint.

The optimal graph search is now performed in the same manner as described before, and the output of the algorithm defines a normal displacement of mesh vertices.

3) Iterative approach: When starting from a certain seed point we will sample in outgoing rays so the distances are usually defined as $\mathbf{q} = (1, 2, \ldots, q_{\max})$. However, the algorithm may take any meshed surface as an input and preform the optimal search along the normals. In particular, we might refine any rough surface by (re-)sampling only in a certain layer around the surface, e.g. by using sampling distances $\mathbf{q} = (q_{\min}, \ldots, -1, 0, 1, \ldots, q_{\max})$. Furthermore, since both the input and the output of the algorithm are triangle meshes, we may use the algorithm in an iterative manner where we increase the number of sampling rays while reducing their length.

4) Challenges: There are a number of issues that should be considered when using an arbitrary meshed surface as the input to the algorithm. The first issue is an applicability of the inter column shift constraint. Mesh edges are the basis for limiting the shift and in order for this constraint to be intuitive and applied equally across the surface, edge length should be roughly uniform. Furthermore, the shift is defined relative to the input mesh, so parameter Δ pertains to how much *additional* inter-column displacement is allowed and does not directly relate to the roughness of the final result. The second important issue is a sampling along the normal direction which can lead to serious problems. If the input mesh is not smooth, vertex normals may overlap within the sample distance leading to folding of the surface. To alleviate the problem [11] suggests sampling along the curved gradient vector flow lines. Instead, and to address both mentioned issues we decided to ensure that the input mesh is of a fair quality by incorporating mesh processing steps in our algorithm.

5) Mesh subdivision: Maintaining a smooth triangle mesh is essential for the performance of our algorithm. It is therefore important that the mesh subdivision increases the number of mesh vertices while preserving smoothness. In order to achieve that, two mesh subdivisions schemes have been used, butterfly [20] and Loop [21] subdivision. In both butterfly and Loop a triangle face is quadrisected by splitting its edges in two and connecting the three inserted vertices. Butterfly subdivision is an interpolating scheme. The positions of the inserted vertices are computed as linear combinations of existing vertices resulting in a smooth surface. The limit surface, obtained by repeated butterfly subdivision, is C^1 continuous everywhere but in the extraordinary points where it is C^0 continuous. Loop subdivision is an approximating scheme. Besides computing the positions of inserted vertices, the position of the original mesh vertices are updated in order to achieve higher smoothness. Loop subdivision generates a limit surface which is C^2 continuous everywhere but in the extraordinary points where it is C^1 continuous.

Figure 2. A photograph of hamster objects used for testing the Round Cut algorithm.

6) Mesh smoothing: The intermediate result of our iterative algorithm is smoothed in every step. This is to prevent mesh normals from overlapping while preparing for the next graph cut detection step. It is appropriate for the mesh smoothing to be rather aggressive, as it is essential for the robustness of the method and it only affects the temporary result. Still, we would like to avoid mesh shrinkage, which accompanies e.g. Laplacian smoothing. This is particulary important when detecting a surface beyond the star domain by growing the mesh from a seed point, where pulling the protrusions inwards would delay the algorithm. Taubin smoothing [22], which we employ, attenuates only high frequencies and may even boost the low frequencies, resulting in less shrinkage.

7) Pipeline: The Round Cut algorithm pipeline includes an initialization, a few refinement steps, and a final surface detection. Prior to the surface detection, volumetric data must be processed. This involves defining the transformation of the data into on-surface and/or in-region costs, e.g. by computing the spatial gradients or the probabilities of the voxels belonging to a region. During the initialization a polyhedron is set up at a seed point, which is either the center of the volume or user defined. The first graph cut surface detection is then performed, typically sampling over the entire volume. Each iterative steps consists of mesh subdivision followed by graph cut surface refinement which in turn is completed by mesh smoothing. The number of vertices grows exponentially when subdividing a mesh, so in general we need to preform only a few subdivision steps before reaching the desired mesh resolution. On the contrary, the number of required graph cut refinement steps might be large when the surface needs to evolve extensively from the seed point. We therefore allow for a number of refinement steps following each subdivision. Mesh smoothing is performed in each of the refinement steps. During

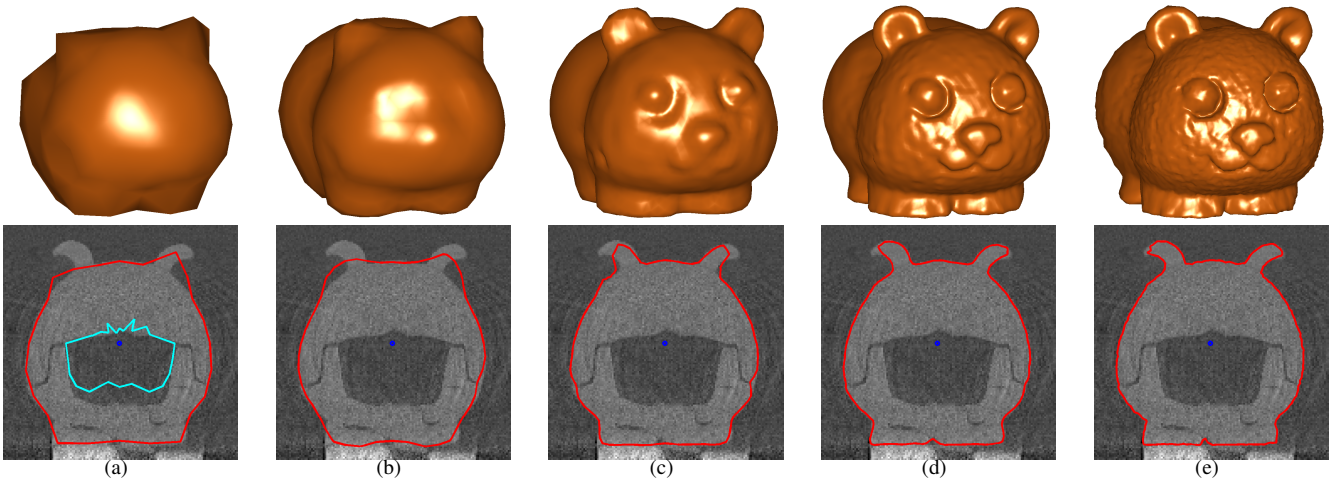


Figure 3. Detecting a surface of a hollow hamster object consisting of four parts fitted together. The top row shows surfaces, while the bottom row shows a slice of the volumetric CT data displaying coronal plane together with the segmentation contours. The blue dot is the initialization point, (a) shows initial rough surfaces defining outer and inner object boundary, (b-d) are three iterations of subdivision and surface refinement followed by smoothing, and (e) shows the result after the final graph cut surface refinement.

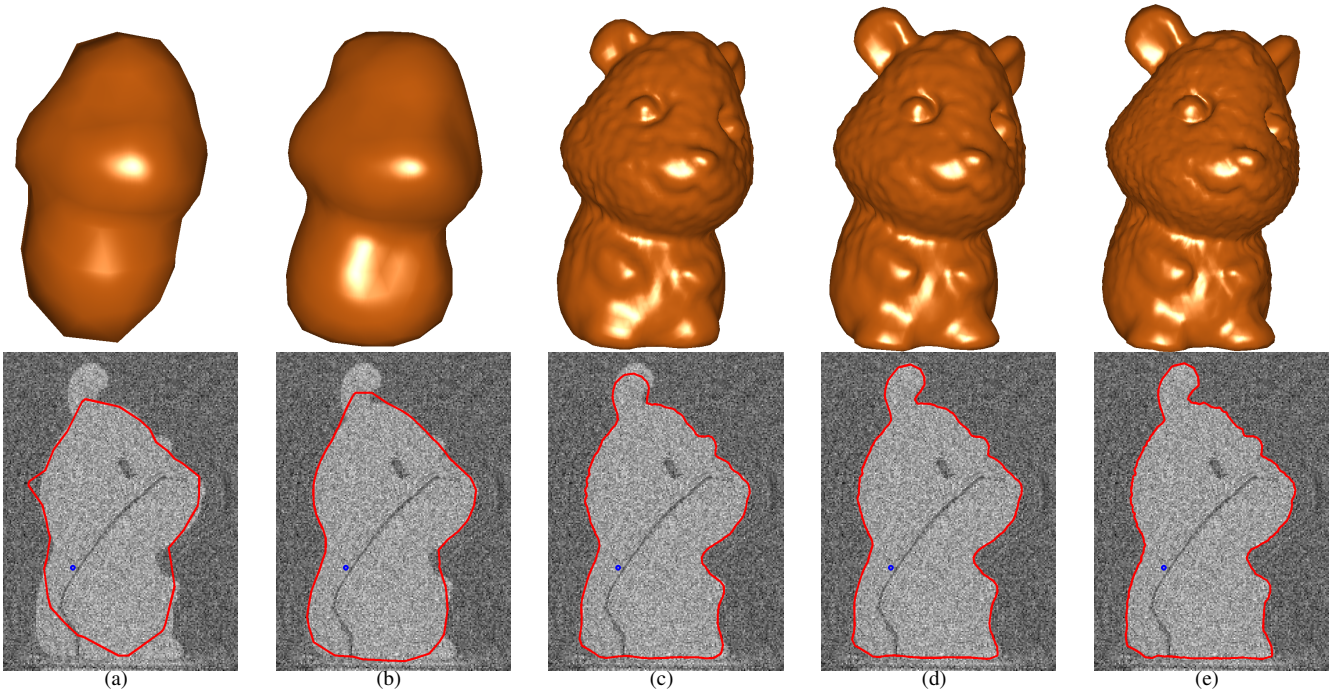


Figure 4. Detecting a surface of another hamster object, with the same five steps as in Fig. 3. Only the outer object boundary is detected and a sagittal plane is shown.

refinements the sampling depth is reduced as we do not consider positions far from the current surface. Likewise, the shift constraint is reduced as we get closer to the desired surface. Finally, the algorithm ends with a single graph cut surface detection which carries out the final adjustment of the surface.

8) Source code: Round Cut is written in MATLAB and relies on a publicly available implementation of the graph cut algorithm developed by Boykov and Kolmogorov [19]. Round Cut source code, together with the volumetric data and the MATLAB scripts for producing the results shown in this paper are available at http://people.compute.dtu.dk/vand.

III. RESULTS

We tested the Round Cut algorithm by segmenting micro-CT scans of two hamster objects shown in Fig. 2. These were chosen because they feature challenging geometry with small details like ears and tails. To make a surface detection

task more challenging, we added 2.5% additive Gaussian noise to the volumetric data.

Fig. 3 shows a segmentation of a hollow object, where a layered surface model was applied to initially locate the inner and the outer boundary. Fig. 4 shows a similar segmentation of a taller hamster object. The segmentation results illustrate the geometric flexibility of the Round Cut algorithm. Our approach detects surfaces outside the star domain with high precision. Detailed features like the ears or eyes of the hamster objects are well recovered as well, despite the single point initialization.

We have compared our algorithm with a single-step approach, similar to [3], [4], by initializing the Round Cut algorithm with high resolution, but without refinement. In our comparison we made sure that the number of vertices in both resulting meshes is the same. The noisy data was challenging for a single-step approach, so in comparison we used noise-free data from higher quality scans. As shown in Fig. 5 and Fig. 6 the single step approach is not able to handle parts outside the star domain due to the straight search lines, and only part of the object are segmented correctly. Furthermore, a large column shift is needed to reach an elongated object, which results in a higher degree of noise in the final segmentation compared to Round Cut.

A few parameters need to be chosen to obtain an optimal performance. This includes an initial polyhedron frequency which governs the resolution of the final mesh. We have chose to set the initial polyhedron frequency to $\nu = 4$. Together with three subdivision steps this results in a mesh with 10242 vertices. Initially the inter-column shift is set to 20 and is reduced to 10, 5 and 2 after each subdivision to reflect the reduction in vertex distances when new vertices are added. The third subdivision step is followed by three refinements without subdivision. Each graph cut is completed by 5 iterations of Taubin smoothing, with a scale factor 0.5 and a band pass frequency 0.2. The final surface refinement step also allows for a shift of 2 voxels.

The cost function used for all shown results is an in-region cost calculated from the volume V using a Gaussian probability distribution $f(v, \mu, \sigma)$ evaluated at each voxel with voxel intensity v. The cost is

$$c_{\text{in}} = \frac{1 - f(V(x,y,z), \mu_k, \sigma_k)}{\sum_{k'=1}^{2}(1 - f(V(x,y,z), \mu_{k'}, \sigma_{k'}))} \qquad (18)$$

where mean μ_k and standard deviation σ_k of inside (k=1) and outside (k=2) were estimated beforehand.

IV. Conclusion

In this paper we have presented the Round Cut surface detection algorithm. This algorithm extends the optimal net surface detection method for meshes. We apply an iterative mesh refinement which allows us to detect objects beyond the star domain. Each surface detection step is solved by an efficient graph cut based algorithm which guarantees the optimal solution within the search lines. We have experimentally tested the algorithm on micro-CT scanned test objects, and we show how high precision segmentation is obtained, also in regions not reachable from the initial seed point.

Explicit mesh processing is crucial for the performance of our method. While it improves robustness of our method, self intersections might still occur if the displacement in the normal direction is big with respect to the curvature of the mesh. This leads to the following drawback of our method. Aggressive smoothing required by the method will

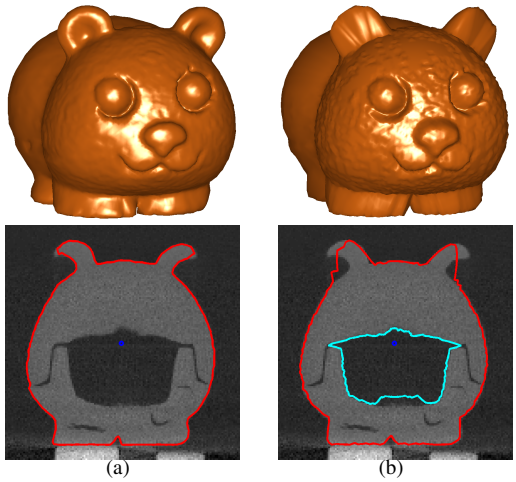


Figure 5. Comparing Round Cut (a) with a single step approach (b) by detecting a surface of a hamster object from a noise-free scan.

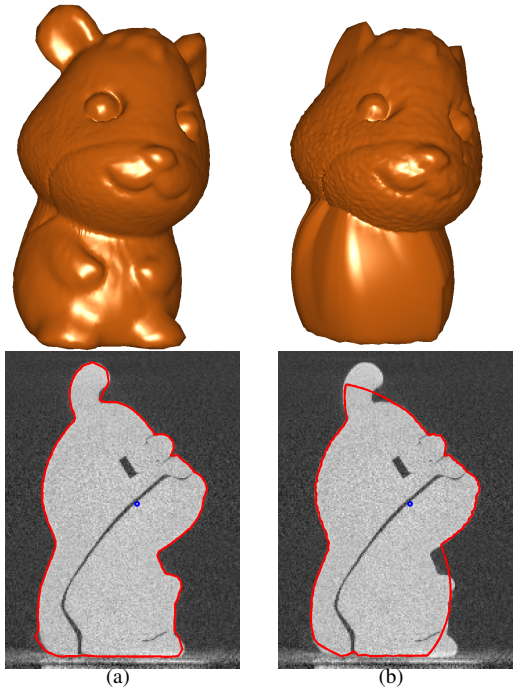


Figure 6. Another comparison of the Round Cut (a) and a single-step approach (b).

in each step pull the surface away from the data, especially at protuberances. As a result, to detect an elongated surface, quite a few iterations might be required – in each iteration a surface approaches the data in the graph cut surface refinement step, but is immediately after pulled back by smoothing.

9) Future work: To alleviate the convergence problem, we plan to incorporate mesh optimization in each iterative step of Round Cut. Having a mesh of a higher quality, with well shaped triangles, will make a calculations of normals less sensitive to noise. This will reduce the need for aggressive smoothing and will allow a faster convergence.

We also plan to extend the Round Cut algorithm with an adaptive mesh subdivision, only in areas where the resolution is small and/or where curvature is high. As a result, the protrusions will not suffer from the lack of vertices and the inter-column shift will have a uniform effect across the surface.

REFERENCES

[1] X. Wu and D. Z. Chen, "Optimal net surface problems with applications," in *Automata, Languages and Programming*. Springer, 2002, pp. 1029–1042.

[2] K. Li, X. Wu, D. Z. Chen, and M. Sonka, "Optimal surface segmentation in volumetric images-a graph-theoretic approach," *TPAMI*, vol. 28, no. 1, pp. 119–134, 2006.

[3] J. Egger, M. H. Bauer, D. Kuhnt, B. Carl, C. Kappus, B. Freisleben, and C. Nimsky, "Nugget-cut: a segmentation scheme for spherically-and elliptically-shaped 3d objects," in *DAGM Pattern Recognition*. Springer, 2010, pp. 373–382.

[4] Y. Wang and R. Beichel, "Graph-based segmentation of lymph nodes in ct data," in *Advances in Visual Computing*. Springer, 2010, pp. 312–321.

[5] S. Steger, N. Bozoglu, A. Kuijper, and S. Wesarg, "Application of radial ray based segmentation to cervical lymph nodes in ct images," *TMI*, vol. 32, no. 5, 2013.

[6] K. Li, S. Millington, X. Wu, D. Z. Chen, and M. Sonka, "Simultaneous segmentation of multiple closed surfaces using optimal graph searching," in *Information Processing in Medical Imaging*. Springer, 2005, pp. 406–417.

[7] Y. Yin, X. Zhang, R. Williams, X. Wu, D. D. Anderson, and M. Sonka, "Logismos – layered optimal graph image segmentation of multiple objects and surfaces: cartilage segmentation in the knee joint," *TMI*, vol. 29, no. 12, pp. 2023–2037, 2010.

[8] S. Sun, G. McLennan, E. A. Hoffman, and R. Beichel, "Model-based segmentation of pathological lungs in volumetric ct data," in *Proc. of Third International Workshop on Pulmonary Image Analysis*, 2010.

[9] K. Lee, R. K. Johnson, Y. Yin, A. Wahle, M. E. Olszewski, T. D. Scholz, and M. Sonka, "Three-dimensional thrombus segmentation in abdominal aortic aneurysms using graph search based on a triangular mesh," *Computers in biology and medicine*, vol. 40, no. 3, pp. 271–278, 2010.

[10] Y. Yin, Q. Song, and M. Sonka, "Electric field theory motivated graph construction for optimal medical image segmentation," in *Graph-Based Representations in Pattern Recognition*. Springer, 2009, pp. 334–342.

[11] C. Bauer, S. Sun, and R. Beichel, "Avoiding mesh folding in 3d optimal surface segmentation," in *Advances in Visual Computing*. Springer, 2011, pp. 214–223.

[12] J. Petersen, M. Nielsen, P. Lo, Z. Saghir, A. Dirksen, and M. De Bruijne, "Optimal graph based segmentation using flow lines with application to airway wall segmentation," in *Information Processing in Medical Imaging*. Springer, 2011, pp. 49–60.

[13] M. K. Garvin, M. D. Abràmoff, R. Kardon, S. R. Russell, X. Wu, and M. Sonka, "Intraretinal layer segmentation of macular optical coherence tomography images using optimal 3-d graph search," *TMI*, vol. 27, no. 10, pp. 1495–1505, 2008.

[14] J. Petersen, P. Lo, M. Nielsen, G. Edula, H. Ashraf, A. Dirksen, and M. de Bruijne, "Quantitative analysis of airway abnormalities in ct," in *SPIE Medical Imaging*. International Society for Optics and Photonics, 2010, pp. 76 241S–76 241S.

[15] M. Haeker, X. Wu, M. Abràmoff, R. Kardon, and M. Sonka, "Incorporation of regional information in optimal 3-d graph search with application for intraretinal layer segmentation of optical coherence tomography images," in *Information Processing in Medical Imaging*. Springer, 2007, pp. 607–618.

[16] X. Wu, D. Z. Chen, K. Li, and M. Sonka, "The layered net surface problems in discrete geometry and medical image segmentation," *International Journal of Computational Geometry & Applications*, vol. 17, no. 03, pp. 261–296, 2007.

[17] M. Haeker, M. D. Abràmoff, X. Wu, R. Kardon, and M. Sonka, "Use of varying constraints in optimal 3-d graph search for segmentation of macular optical coherence tomography images," in *MICCAI*. Springer, 2007, pp. 244–251.

[18] Q. Song, J. Bai, M. Garvin, M. Sonka, J. Buatti, and X. Wu, "Optimal multiple surface segmentation with shape and context priors," *TMI*, vol. 32, no. 2, pp. 376–386, 2013.

[19] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *TPAMI*, vol. 26, no. 9, pp. 1124–1137, 2004.

[20] N. Dyn, D. Levine, and J. A. Gregory, "A butterfly subdivision scheme for surface interpolation with tension control," *ACM transactions on Graphics (TOG)*, vol. 9, no. 2, pp. 160–169, 1990.

[21] C. Loop, "Smooth spline surfaces over irregular meshes," in *Computer Graphics and Interactive Techniques*. ACM, 1994, pp. 303–310.

[22] G. Taubin, "A signal processing approach to fair surface design," in *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. ACM, 1995, pp. 351–358.