Pearling: 3D interactive extraction of tubular structures from volumetric images

B. Whited[1], J. Rossignac[1], G. Slabaugh[2], T. Fang[2], and G. Unal[2]

[1] Georgia Institute of Technology, Graphics, Visualization and Usability Center, Atlanta, GA 30332
[2] Siemens Corporate Research, Intelligent Vision and Reasoning Department, Princeton, NJ 08540

Abstract. This paper presents Pearling, a novel three-dimensional approach to the interactive segmentation and modeling of tubular structures from a volumetric image. Given a user-supplied initialization, Pearling extracts runs of pearls (balls) from the image, where each pearl is specified by a center position and radius. The runs are combined into a graph, with bifurcations and possibly loops. By treating each pearl's center and radius parameters as a control point in 4D space, a continuous tubular model is defined via subdivision. We show that Pearling is both computationally efficient and flexible, providing a convenient mechanism for fast, interactive segmentation of a portion of interest in a tubular network.

1 Introduction

We propose a new approach, called *Pearling*, for the interactive extraction of user-selected tubular sub-structures directly from 3D (three-dimensional) images. Each year, hundreds of millions MRI and CT scans are performed. Many of these are needed to support diagnosis or analysis of cardiovascular or pulmonary connections, where physicians examine specific subsets of tubular structures. The availability of higher level models of tubular structures assists physicians in performing quantitative measurements, diagnosis, surgical planning [1], hemodynamic analysis, and follow-up studies to assess the progress or remission of disease. However, extracting clean 3D structures from raw images is very time consuming and often requires tedious user intervention and editing.

We propose an approach, demonstrated in Figure 1 (a) through (k), where the user guides the automated construction of a 3D model of the desired structure directly from the 3D dataset, not going through slice contours or relying on global segmentation. Our initial evaluation suggests that the approach will significantly improve clinical workflow because it eliminates delays between user actions (that select which undesired branches should be trimmed or where new branches that are missing should be grown) and automatic updates of the resulting 3D model. In addition, we optionally provide the ability to input end-points, such that Pearling does not waste time segmenting unwanted branches. As a result, an

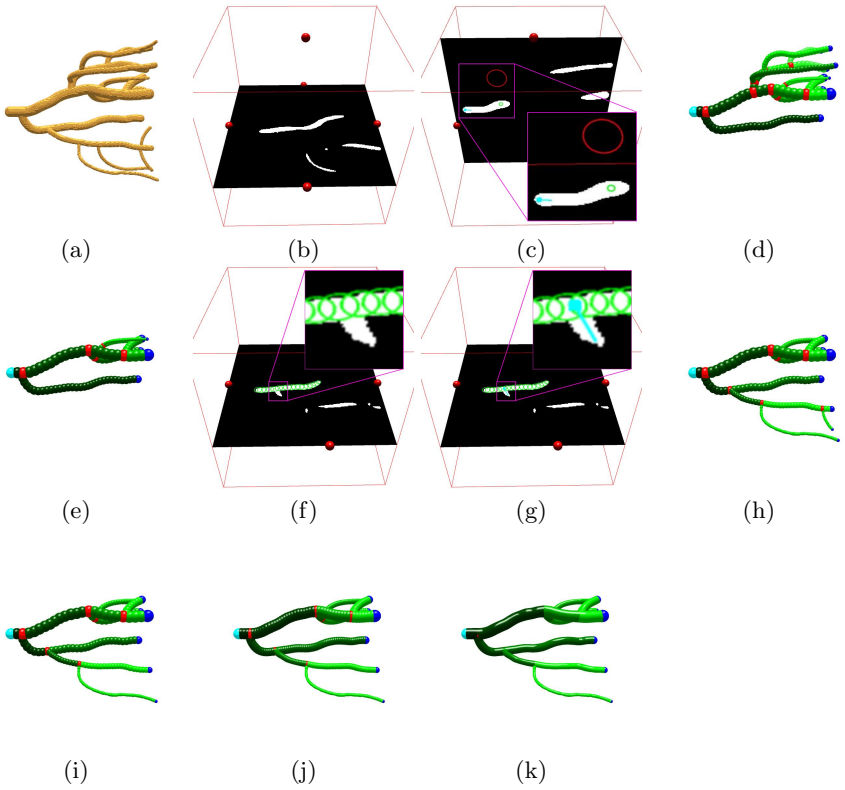

Fig. 1. (a) Segmentation obtained through global thresholding. (b) A horizontal slice through the volume. (c) A different slice in which the operator has marked selected examples of good (green) and bad (red) material and has identified the seed and initial growth direction (cyan). (d) Initial tree extracted by Pearling in 1.23 seconds with the seed-pearl (cyan), branching-pearls (red), leaf-pearls (blue), and control-pearls (green). (e) By clicking on the base of undesired branches, the operator has trimmed the tree. (f) By sliding the cross-section along the spine, the operator has discovered a missing branch. (g) The direction of the desired branch is indicated in cyan. (h) New portion added to the tree at the desired location. (i) A small branch of the new portion removed. (j) Branches after 1 refinement step. (k) Branches after 5 refinements. The whole process took the operator 95 seconds.

operator may interactively construct an accurate model of exactly the desired portion of a tubular structure in a couple of minutes or less.

The output produced by Pearling is a high-level 3D model of the desired portion of the tubular structure. For simplicity, our 3D model will be called a tree (even though it is an oriented graph, that may contain closed loops). The tree has a seed (root) and is divided into branches and bifurcations. Each bifurcation is a ball. Each branch is the region swept by a ball of varying radius and is

defined by a smooth four-dimensional piecewise parametric curve that specifies the spine (three-dimensional curve) and local thickness (radius variation along the spine).

1.1 Related Work

Tubular surface segmentation and modeling is a wide-ranging subject with numerous applications. Kirbas and Queck [2] provide a recent survey of techniques for vessel segmentation. They conclude that there is no single vessel segmentation approach that is robust, automatic, and fast, that successfully extracts the vasculature across all imaging modalities and different anatomic regions. Following their conclusion, we step away from total automation and instead focus on interactive segmentation, striving for an effective symbiosis between parsimonious user guidance through simple mouse clicks and interactive system responses that expand or trim the 3D structure recovered so far or show it in the context of the original input data. Other vessel segmentation approaches in the medical imaging literature include [3], who model vessel segments using superellipsoids, and [4], who implement co-dimension two level set flows. While elegant approaches, due the significant computational requirements of these techniques, they are not practical for interactive segmentation and modeling. Fast methods such as [5] perform the segmentation on 2D slices made in a plane orthogonal to the vessel centerline or intensity maxima on 2D slices [6] and then extract 3D geometry by connecting the 2D results. Unlike these methods, our segmentation approach is based fully in 3D, by defining forces that act on a pearl to position it in the vessel.

2 Overview

The proposed Pearling approach involves four phases, summarized below.

2.1 Initialization

The operator uses an interactive viewer to explore the volume of interest using the mouse to orient the view and to continuously slide or reorient a cross-section through it (Fig. 1-(b)). The operator may click on any cross-section, to gather samples of good and bad material (Fig. 1-(c)). Good material represents the intensities inside the desired tubular structure; bad material represents the intensities on the boundary of the vessel or outside it. These initial intensities will be used by Pearling to compute histograms, which are Parzen-windowed [7] to produce probability distributions denoted as $p_g(I)$ and $p_b(I)$, for the good and bad material respectively. Then, by clicking on a slice and dragging the mouse in a desired direction, the operator identifies the seed and suggests the general direction in which the tree is to be grown (Fig. 1-(c)). Also, the operator may choose endpoints for the segmentation, which are input as single-points.

2.2 Growth

Pearling automatically grows the tree (portion of the tubular structure) one branch at a time, identifying bifurcations and making local decisions automatically as to which branches to follow. In our approach, we have three global parameters: T_h, which controls the desired height of the tree, $r_{\min}$, which is a minimum pearl radius, and G_{target}, which will be defined later. Because the effect of changing these parameters may be shown immediately, these parameters may be adjusted easily for each application, and even, if desired for each model.

The branches of the tree connect the seed to bifurcations and to leaves. All are represented by a graph whose nodes are balls of possibly different radii, called *pearls*. The seed, each bifurcation, and each leaf is a pearl. Furthermore, each branch may contain a string (ordered set) of control pearls. A leaf-pearl has one incident edge. A control-pearl has two. A bifurcation-pearl has three or more. Pearling starts at the seed-pearl and attempts to grow a branch, adding one control-pearl at a time and deciding whether it is a leaf-pearl, a control-pearl, or a bifurcation pearl. It recursively follows all branches at bifurcation-pearls and stops at leaf-pearls. Pearling also detects loops where the next control pearl of the current branch reaches an older pearl in the tree and ensures that the contact is a bifurcation pearl. The result is a graph whose nodes are pearls and whose edges define the connectivity.

The growth process also checks whether each new pearl intersects an endpoint that was input by the operator. Once an endpoint is reached, the intersecting branch stops growing. After all endpoints have been found, the entire growth process stops, and all branches which do not end at an input endpoint are trimmed away, leaving only the desired section structure. This allows for the operator to isolate specific sections of the tubular surface prior to the segmentation, reducing the running time and the post-segmentation trimming phase.

The duration of the growth phase depends on the total number of pearls and on their size, but the growth typically happens in realtime. For the example demonstrated in Fig. 1-(d), Pearling grew about 300 pearls with a radius of roughly 5 voxels in one second, and the total structure was grown in 1.23 seconds.

2.3 Editing

Any pearl may be selected by a mouse click. Through a key-press and mouse-click combination, the operator may trim the portion of the tree starting at the clicked pearl. Hence, with a few clicks, all undesired branches may be removed in seconds (Fig. 1-(e)). The operator may toggle between seeing the tree or seeing any of the axis-aligned cross-sections through the selected (Fig. 1-(f)). If a cross-section reveals a missing branch, the operator may click and drag on the cross-section (Fig. 1-(g)) to indicate the position and direction of a new growth process for adding a portion of the tree (Fig. 1-(h)). Small unwanted branches produced by this new growth may be easily removed (Fig. 1-(i)).

2.4 Refinement

Pearling generates runs of control-pearls, which can be used as control points of a curve in the four dimensional space of the (x, y, z, r) coordinates defining the center location and the radius of the pearls. This control polygon defines a continuous (piecewise-cubic) model of each branch, which may be trivial approximated by iterating Catmull-Rom [8] refinements (Fig. 1-(j)).

3 Computing the next pearl

The process of computing the next pearl involves two steps. In step 1 we compute its position and radius. In step 2, we decide whether it is a bifurcation-pearl, a control-pearl, or and end-pearl and compute the starting positions for the dependent pearls, if any. Step 2 includes the detection of closed loops.

3.1 Step 1: Position and radius of the next pearl

In this step, we analyze the voxels inside the pearl. Each voxel can impart a force on the pearl, and the forces are integrated and applied to the pearl, causing its position and radius to change. The integrals in our approach provide robustness to noise. The following equations are presented in a continuous form for elegance, but the conversion to a discrete form is trivial. We estimate the overall goodness G of pearl P_i as the ratio of bad to total material as

$$G = \frac{\int_{\mathbf{x}\in P_i} \phi(\mathbf{x}) d\mathbf{x}}{\int_{\mathbf{x}\in P_i} d\mathbf{x}}, \tag{1}$$

where $\mathbf{x}$ is a voxel center, and

$$\phi(\mathbf{x}) = \begin{cases} 1, & \text{if } p_b(I(\mathbf{x})) > p_g(I(\mathbf{x})) \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

is an indicator function that equals one when the voxel's intensity has a higher probability of belonging to the bad region than the good region.

While computing the goodness, we compute a weighted average $\mathbf{F}$ of repelling forces, defined as

$$\mathbf{F}(\mathbf{c}_i, r_i) = \frac{9}{\pi r_i^3} \int_{\mathbf{x}\in P_i} \phi(\mathbf{x})(\mathbf{c}_i - \mathbf{x}) \left(1 - \frac{\|\mathbf{c}_i - \mathbf{x}\|^2}{r_i^2}\right) d\mathbf{x} \tag{3}$$

where r_i is the radius of the ith pearl, and $\mathbf{c}_i$ is its center. The repelling force of a voxel has direction from the pixel towards the center of the pearl and has a magnitude that is proportional to the weight (measure of badness) of the scalar field value at the voxel and the inverse of its squared distance to the center. This means that pixels that are closer to the center of the pearl apply more force.

The factor $\frac{9}{\pi r_i^3}$ was computed as follows: imagine a pearl intersected through its center by a plane, dividing the pearl into equal "good" and "bad" regions. In

this case, the term $\frac{9}{\pi r_i^3}$ ensures that the magnitude of the offset is equal to r_i, which would move the pearl to be tangent to the intersecting plane and entirely in the "good" region. This factor is important in the fast convergence of the pearl position.

We adjust the radius striving towards a prescribed goodness, G_{target} and then displace the pearl according to $\mathbf{F}$, but cap the displacement so that its surface remains sufficiently close to the center of the previous pearl. We repeat the process until G is sufficiently close to G_{target}. In practice, this process takes about 20 iterations per pearl and Pearling can grow about 300 new control pearls per second. Through trial and error, we have selected the target ratio for G_{target} to be 25%, which works sufficiently well for all the examples throughout this paper.

3.2 Step 2: Pearl classification and bifurcation directions

To establish whether the current pearl is an end-, control-, or bifurcation-pearl, we proceed as follows.

First, a check is done to see if the current pearl contains an endpoint. If so, the pearl is designated an end-pearl. Otherwise, we use a set of rays emanating from uniform samples on the pearl's surface along the normals to sample the scalar field. These samples are simply the vertices of a pre-existing triangle mesh representation of a geosphere which is scaled to fit each pearl. Let $\mathbf{v}$ be a sampling point on the pearl's surface. We sample the scalar field along the segment $\mathbf{c}_i + r_i t(\mathbf{v} - \mathbf{c}_i)$, where $t \in [1, 2]$. A segment is classified as good if all its samples lie in the good region; otherwise it is classified as bad. Next, we identify the connected components of the samples $\mathbf{v}$ on good segments and for each connected component, we identify a "center" of the connected component as the average of the $\mathbf{v}$s corresponding to the segments within the component. To prevent backtracking, we discard the connected component that is closest to the previous (parent) pearl. If the current pearl has no remaining connected component, it is an end-pearl. If it has one remaining connected component, it is a control pearl. Otherwise, it is a bifurcation pearl. We use the remaining connected component(s) as initial center(s) for the subsequent pearl(s). Finally, to detect cycles, we can simply do sphere-intersection tests. If an intersection is found between two pearls that are not directly connected, then the intersected pearl is labeled as a bifurcation.

4 Evaluation

Although we plan to conduct a formal evaluation in a clinical setting, the process is complicated and falls far beyond the scope of the present manuscript. Hence, we include here only the results of a preliminary evaluation.

We have evaluated the pearling approach on a variety of datasets. In Fig. 2, a rotational angiography scan of a head with an aneurysm, a CT scan of the bronchial tubes and lungs, and a contrast-enhanced MRA of pulmonary arteries

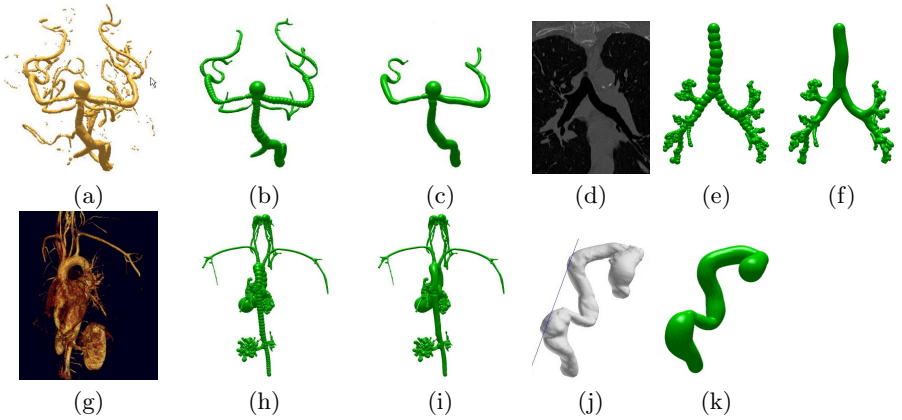

Fig. 2. (a) Thresholded volume rendering of the original angiography dataset (b) Initial result of the pearling algorithm (328 pearls, 1.1 seconds) (c) Result of user editing and refinement (9 deletions, 30 seconds) (d) Slice of a chest CT scan showing bronchial tubes (e) Result of the initial run of pearling (638 pearls, 1.5 seconds) (f) Result of user editing and refinement rendered within the volume itself (2 deletions, 10 seconds) (g) Detailed volume render of pulmonary arteries (h) Initial pearling result (i) Refined pearling result (j) Aorta segmention obtained via level set methods (10 minute execution on reduced volume) (k) Pearling result using an endpoint and no trimming (1.5 second execution time on full-resolution image)

are all shown as examples. In Fig. 2a, a thresholded volume rendering of the original angiography dataset is shown. Then, in Fig. 2b, we show the initial result of the Pearling algorithm which contains 328 pearls and took 1.1 seconds. Finally, in Fig. 2c, we show the result of user editing where the goal was to trim everything away except the main vessel, the aneurysm, and the two major branches leading from that junction. This interaction took about 30 seconds total and involved nine deletions.

In Fig. 2d a slice of the chest CT scan is shown within our pearling environment. Then, in Fig. 2e, we show the result of the initial run of pearling. This result contains 638 pearls and executed in 1.5 seconds. Even though there are almost twice as many pearls as the angiography, the running times are similar because most of the pearls in this example are extremely small and converge faster. Finally, in Fig. 2f, we show the result of refinement, producing a smooth structure.

The next example was chosen to show the result of pearling on a more detailed dataset. In Fig. 2g, a volume rendering of the dataset is shown, which includes the aorta, heart, kidneys and pulmonary arteries. In Fig. 2h, we show the result of pearling, which contains 10186 pearls and took 67 seconds to complete. Then, in Fig. 2i, the result of the refinement is shown, which creates smooth tubular structures.

In such a complex example, if the operator only needs to examine a subset of the image, an endpoint should be used to tell pearling to explicitly stop. The result in Fig. 2k shows such an example of an aorta. No final trimming was needed for this result, and the execution took 1.5 seconds on the raw 512x512x459 volume. The result shown in Fig. 2j is from a level-set approach, which required 10 minutes of running time on a reduced volume of size 256x256x229, because the full size volume combined with temporary structures would not fit in the RAM of our test machine. Note that the Pearling result is generated in over two orders of magnitude less time.

5 Conclusion

In this paper we presented Pearling, which interactively constructs 3D models of tubular shapes and represents them using a discrete set of Pearls and their connectivity. We have demonstrated that Pearling is computationally efficient and well suited to user interactivity. This interactivity affords user guidance of the segmentation in a particular direction as well as user correction of errant segmentation results in realtime. The overarching objective of our research is not total automation, but rather an effective tool for quickly creating and inspecting segmentations of a particular subset of interest.

References

1. Sharma, S., Goudy, S., Walker, P.G., Panchal, S., Ensley, S., Kanter, K., Tam, V., Fyfe, D., Yoganathan., A.P.: In Vitro Flow Experiments for Determination of Optimal Geometry of Total Cavopulmonary Connection for Surgical Repair of Children with Functional Single Ventricle. Journal American College of Cardiology **27** (1996) 1264–1269
2. Kirbas, C., Quek, F.: A Review of Vessel Extraction Techniques and Algorithms. ACM Computing Surveys **36**(2) (2004) 81–121
3. Tyrrell, J.A., di Tomaso, E., Fuja, D., Tong, R., Kozak, K., Jain, R., Roysam, B.: Robust 3-D Modeling of Vasculature Imagery Using Superellipsoids. IEEE Trans. on Medical Imaging **26**(2) (2007) 223–237
4. Lorigo, L.M., Faugeras, O.D., Grimson, W.E.L., Keriven, R., Kikinis, R., Nabavi, A., Westin, C.F.: CURVES: Curve Evolution for Vessel Segmentation. Medical Image Analysis **5** (2001) 195–206
5. Wink, O., Niessen, W., Viergever, M.A.: Fast delineation and visualization of vessels in 3-d angiographic images. IEEE Trans. on Medical Imaging **19**(4) (2000) 337–346
6. Szymczak, A., Tannenbaum, A., Mischaikow, K.: Coronary vessel cores from 3D imagery: a topological approach. In: Medical Imaging 2005: Image Processing. Proceedings of the SPIE. Volume 5747. (2005) 505–513
7. Parzen, E.: On estimation of a probability density function and mode. Ann. Math. Stat. **33** (1962) 1065–1076
8. Catmull, E., Rom, R.: A Class of Local Interpolating Splines. In: Computer Aided Geometric Design. Academic Press, New York (1974) 317–326 Barnhill, R.E. and R.F. Riesenfeld, Editors.