Medical Image Analysis 18 (2014) 272–284

Contents lists available at ScienceDirect

Medical Image Analysis

journal homepage: www.elsevier.com/locate/media

3D actin network centerline extraction with multiple active contours

Ting Xu [a], Dimitrios Vavylonis [b], Xiaolei Huang [a,*]

[a] Department of Computer Science and Engineering, Lehigh University, Bethlehem, PA, USA
[b] Department of Physics, Lehigh University, Bethlehem, PA, USA

ARTICLE INFO

Article history:
Received 11 February 2013
Received in revised form 27 October 2013
Accepted 30 October 2013
Available online 16 November 2013

Keywords:
Cytoskeleton
Actin filaments
Curvilinear networks
Active contours
Normalized cuts

ABSTRACT

Fluorescence microscopy is frequently used to study two and three dimensional network structures formed by cytoskeletal polymer fibers such as actin filaments and actin cables. While these cytoskeletal structures are often dilute enough to allow imaging of individual filaments or bundles of them, quantitative analysis of these images is challenging. To facilitate quantitative, reproducible and objective analysis of the image data, we propose a semi-automated method to extract actin networks and retrieve their topology in 3D. Our method uses multiple Stretching Open Active Contours (SOACs) that are automatically initialized at image intensity ridges and then evolve along the centerlines of filaments in the network. SOACs can merge, stop at junctions, and reconfigure with others to allow smooth crossing at junctions of filaments. The proposed approach is generally applicable to images of curvilinear networks with low SNR. We demonstrate its potential by extracting the centerlines of synthetic meshwork images, actin networks in 2D Total Internal Reflection Fluorescence Microscopy images, and 3D actin cable meshworks of live fission yeast cells imaged by spinning disk confocal microscopy. Quantitative evaluation of the method using synthetic images shows that for images with SNR above 5.0, the average vertex error measured by the distance between our result and ground truth is 1 voxel, and the average Hausdorff distance is below 10 voxels.

1. Introduction

Analysis of protein fiber networks imaged by fluorescence microscopy reveals structural and dynamical properties important in understanding their biological functions. For instance, the structure and connectivity of collagen and fibrin networks are of crucial importance to their mechanical properties in the tissue (Storm et al., 2005; Piechocka et al., 2010; Stein et al., 2008). Actin filaments, the focus of this paper, form dynamic meshworks that organize in structures such as stress fibers, actin cables and contractile rings that are of fundamental importance for the cell (Pollard and Cooper, 2009). In many cases, these fibers are dilute enough to allow systematic analysis of the network topology and dynamics. Fig. 1(a) shows fluorescently-labeled actin filaments imaged by Total Internal Reflection Fluorescence Microscopy (TIRFM) in a study of actin polymerization *in vitro* (Fujiwara et al., 2007). In this experiment the filaments grew parallel to a glass slide by polymerization and intersected with each other as they elongate. Fig. 1(b) shows a 3D *in vitro* network of actin cables (bundles of actin filaments) imaged by confocal microscopy (Falzone et al., 2012). In Fig. 1(c), actin cables inside a yeast cell were imaged by spinning-disk confocal microscopy in 3D (Smith et al., 2010). Actin cables promote polarized cell growth by directing the transport of vesicles towards the growing cell tip. They are highly dynamic, changing their distribution inside the cell within minutes. During mitosis, actin reorganizes and forms a dynamic meshwork in the cell center (Fig. 1(d)). This meshwork condenses into a contractile ring whose constriction drives the separation of the cell into two daughters (Vavylonis et al., 2008; Pollard and Wu, 2010).

Image analysis of protein fiber networks has given insights into the function of biomolecular systems. To study the geometrical and mechanical properties of these networks quantitatively, an important step is to reliably extract their morphology. One of the earliest computer-assisted extraction and reconstruction methods, with human input playing a major role, was developed by Baradet et al. (1995) who analyzed fibrin networks. Semi- and fully-automatic identification of biopolymer fiber networks has since become an active research topic. One category of methods uses binarization followed by skeletonization. Wu et al. (2003) and Stein et al. (2008) reconstructed collagen fiber networks by tracing maximal ridges in the Euclidean distance maps of the binarized images. Mickel et al. (2008) extracted the skeleton of collagen fiber networks by distance-ordered homotopic thinning using the Chamfer Distance map. Herberich et al. (2010, 2011) extracted networks of microtubules and intermediate filaments by binarizing

* Corresponding author.
E-mail address: huang@cse.lehigh.edu (X. Huang).

1361-8415/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.media.2013.10.015

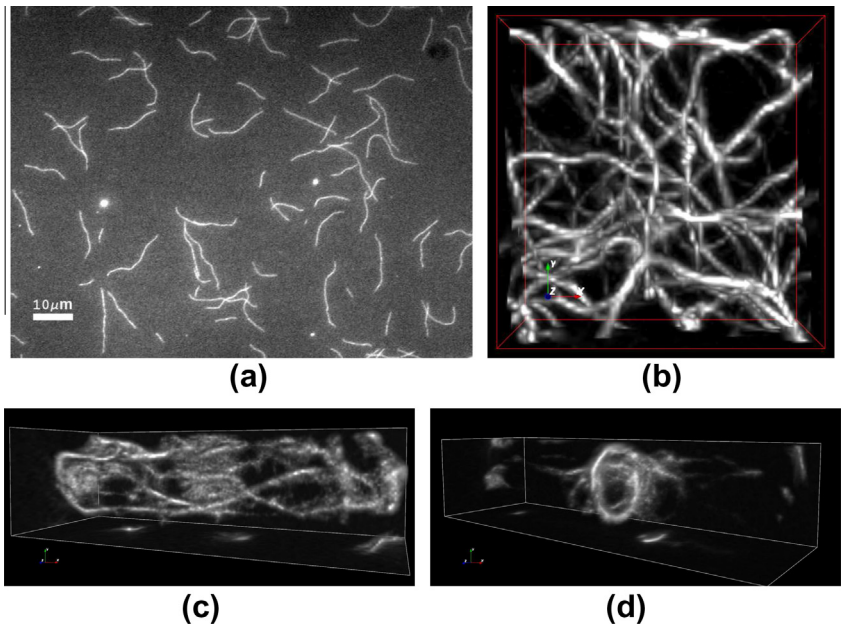

Fig. 1. Examples of biopolymer meshwork in 2D and 3D. (a) Intersecting actin filaments in one frame of a TIRFM time-lapse sequence (Fujiwara et al., 2007). Scale bar, 10 μm. (b) A 3D network of actin filaments cross-linked by α-actinin, reproduced from Supplementary Movie 2 of (Falzone et al., 2012). (c and d) 3D meshworks of actin cables labeled by GFP-CHD in a fission yeast cell, of which radius is 1.73 μm. Images were acquired at different times during the cell's life cycle. The cells were treated with CK-666 (Nolen et al., 2009) that disassembled actin patches and allowed clearer images of actin cable structures. Images are provided by Jian-Qiu Wu.

and thinning the image filtered by vessel enhancement filters, such as those based on Hessian analysis (Frangi et al., 1998; Sato et al., 1998). Basu et al. (2013) extracted the centerlines of actin meshwork by thinning a tubularity map using a constrained reverse diffusion-based method. In their method, centerline pixels are connected by finding Minimum Spanning Trees among nearby centerlines pixels. However, the network topology was not extracted. To remove artifacts caused by binarization, Beil et al. (2005) and Lück et al. (2010) further pruned the obtained skeletons, eliminating outliers such as open branches or loops. A more robust way of extracting these networks is to use template matching (Mayerich and Keyser, 2009; Rigort et al., 2012; Weber et al., 2012; Krauss et al., 2012), where prior knowledge about the target is incorporated into a database of 2D or 3D templates. These template-based detection methods are more selective and impose stronger constraints than simple intensity thresholding. Another form of template matching is to use Hough/Radon Transform to extract linear structures. Sandberg and Brega (2007) and Winkler et al. (2012) adopted localized Radon Transform to extract thin line network of microtubules and actin filaments. In (Rusu et al., 2012), actin filaments in cryo-ET data sets were segmented by a stochastic template-based search, which combines a genetic algorithm and a bidirectional expansion strategy. However, the method does not resolve the possible template overlap when tracing converging filament branches and the network junctions were left undetected.

Confocal microscopy images of polymer fiber networks usually suffer from low SNR. Active contour based methods increase segmentation robustness by incorporating prior information about the object shape. Unlike the previous image processing or template matching based methods, open curve parametric active contours (Kass et al., 1988; Berger and Mohr, 1990) explicitly model the linear nature of filaments, and have been successfully applied to the segmentation of actin filaments (Li et al., 2009a; Li et al., 2009b; Li et al., 2010; Smith et al., 2010), microtubules (Kong et al. (2005); El-Saban and Manjunath, 2005; El-Saban et al., 2006; Altinok et al., 2006; Nurgaliev et al., 2010), and axons (Wang et al. (2011)). These previous methods mostly segment individual filaments instead of extracting both geometry and topology of the entire curvilinear network. El-Saban and Manjunath (2005) used 2D open Snakes to track microtubules but their method does not handle the cases when microtubules cross each other. In their method, the extent of elongating or shrinking the Snakes relies on thresholded outputs of a line detector, which may cause over-/under-segmentation in case of large intensity variations in foreground and background regions. In contrast to the above methods based on parametric active contours, our proposed method is able to extract the topological information about complex 3D curvilinear networks with intersecting filaments. Furthermore, to handle intensity variations and address over-/under-segmentation, we defined a locally adaptive stretching force that can elongate Snakes properly based on local image contrast.

Different from parametric active contours, implicit active contours (Malladi et al., 1995; Caselles et al., 1997) have been used for better handling of topology changes when segmenting curvilinear structures such as bronchi (Lorigo et al., 2001), blood vessels (Vasilevskiy and Siddiqi, 2002), white matter tracts (Melonakos et al., 2008), and road networks (Rochery et al., 2006). Using implicit active contours for centerline extraction, Basu et al. (2007) evolved 2D open curves implicitly by deriving the medial axis from the zero level set, but it is not clear how this formulation can be extended to 3D. Moreover, as topology change is handled implicitly, additional steps are needed to detect junctions and retrieve topological information. In contrast, our method uses a coordinated set of parametric active contours to explicitly extract the centerlines and detect junctions as the models evolve to segment the network, eliminating the need for additional steps to retrieve topological information. Also, implicit active contours are usually more computationally expensive than parametric active contours. Combined with automatic initialization, our method is thus more efficient using the parametric active contour model.

To handle robustly corrupted data or anomalies, minimal path approaches (Hadjidemetriou et al., 2005; Sargin et al., 2007a,b) have also been used to extract the centerlines of curvilinear networks, as these methods can find the global optimal path between two points. These methods are usually interactive, requiring the user to specify the ends of a filament before extracting its

centerline. The method of Hadjidemetriou et al. (2005) requires manual input of one starting point for each microtubule in 2D. It would be cumbersome to make such methods adapt to 3D segmentation tasks, however, since it would be difficult for the user to specify ending points of filaments in a 3D curvilinear network, especially when the network is dense. In contrast, our method can segment 3D curvilinear networks without manual initialization. Another method that specifies initial seed points automatically was proposed by Sargin et al. (2007b) to trace intersecting curvilinear structures in 2D; this method does not explicitly detect the locations of junctions, however, and it has a constraint on the initial seed points which must not be on a gap or intersection.

In a previous study (Xu et al., 2011), we have shown how multiple Stretching Open Active Contours (SOACs) that elongate and merge with one another can be used to segment networks of actin filaments in 2D TIRFM images, such as in Fig. 1(a). In this article we present an extension and significant modification of the multiple SOACs method to address the more complex issues that arise in 3D and to propose mechanisms that make the approach robust to heavy image noise, non-uniform filament and background intensity, and anisotropic sampling as are typical in many 3D experimental images. We introduce new methods for local background intensity estimation and for sequential SOAC evolution and reconfiguration, and also present our validation results from an extensive set of experiments on images of actin meshworks.

Our overall strategy is to extract the whole curvilinear network at once using SOACs that are automatically initialized at image intensity ridges. The initial SOACs then evolve along the centerlines of curvilinear structures to extract the entire network. Because of their large number, the initial SOACs can cover every branch of the network but are redundant. We propose novel mechanisms that eliminate any redundancies by allowing the SOACs to merge, stop at junctions, and transform into closed curve forms. In Xu et al. (2011) all of the initial SOACs evolved simultaneously. Here we found that sequential evolution according to a specific set of rules (a process we refer to as "regulated sequential evolution") is a more robust method. After regulated sequential evolution, we maintain the smoothness of SOACs across junctions by carefully reconfiguring them with other SOACs. Using the resultant SOACs, the geometry and topology of the network can be analyzed, such as orientation, curvature, connectivity, and the relationship among filaments.

In summary, our main contributions are (1) A complete set of mechanisms to regulate SOACs' behavior to generate a clean and physical representation of network morphology by a set of SOACs; (2) A fully automatic initialization of multiple SOACs in 3D; (3) An adaptive stretching force enabling SOACs to elongate on filaments with a wide range of intensity levels.

2. Methods

In this section, we first introduce Stretching Open Active Contours (Section 2.1) with adaptive stretching forces and their automatic initialization (Section 2.2). We then introduce the regulated sequential evolution scheme to avoid overlap among SOACs (Section 2.3). The result of this evolution process is a network of active contours that connect with one another at junction points ("T-junctions," which are vertices of order 3) or else have an unconnected end corresponding to a dangling filament tip. The T-junctions would capture actin cytoskeletal networks that consist of branches off of mother filaments, such as those formed by the Arp2/3 complex (Pollard and Cooper, 2009). Actin networks may also contain crosses (vertices of order 4) when two filaments are linked by a cross-linker (Pelletier et al., 2003; Falzone et al., 2012). Actin bundles may also form star-shapes (vertices of order 5 and higher) (Vignjevic et al., 2003). While we do not account for vertices of order higher than 3 during evolution, we reconfigure the segmented filaments as a separate step at the end, for example by joining two nearby T-junctions into a cross-junction. Similarly, main filament and branches are reassigned at the end. A flow diagram of the proposed method is shown in Fig. 2.

2.1. Stretching open active contours

Stretching Open Active Contours (SOACs) (Li et al., 2009a) are open-ended parametric active contours with stretching forces applied at both ends of the open curve, so that the models can elongate while conforming to the desired image feature.

Mathematically, a SOAC in three-dimensional space is a curve $\boldsymbol{r}(s) = (x_0(s), x_1(s), x_2(s))^T,\ s \in [0, L]$ parameterized by arc length s, with L being its total length (Fig. 3(a)). It evolves by minimizing a contour energy functional $\mathcal{E}(\boldsymbol{r}) = \mathcal{E}_{int}(\boldsymbol{r}) + \mathcal{E}_{ext}(\boldsymbol{r})$. Minimizing the internal energy functional $\mathcal{E}_{int}(\boldsymbol{r})$ maintains the continuity and smoothness of the curve; minimizing the external energy functional $\mathcal{E}_{ext}(\boldsymbol{r})$ pushes the curve towards salient image features such as edges or ridges. $\mathcal{E}_{int}(\boldsymbol{r})$ is defined as

$$\mathcal{E}_{int}(\boldsymbol{r}) = \int_0^L \alpha(s)|\boldsymbol{r}'(s)|^2 + \beta(s)|\boldsymbol{r}''(s)|^2\,\mathrm{d}s \tag{1}$$

where $\alpha(s)$ and $\beta(s)$ are weights for controlling the tension and rigidity of the curve. The external energy functional $\mathcal{E}_{ext}(\boldsymbol{r})$ is composed of an image potential energy function $E_{img}(\boldsymbol{x})$ and a stretching energy function $E_{str}(\boldsymbol{r})$,

$$\mathcal{E}_{ext}(\boldsymbol{r}) = \int_0^L k_{img}E_{img}(\boldsymbol{r}(s)) + k_{str}E_{str}(\boldsymbol{r}(s))\,\mathrm{d}s \tag{2}$$

where k_{img} and k_{str} are weights for controlling the strength of the image and stretching forces, respectively. The image potential energy field $E_{img}(\boldsymbol{x})$ is the convolution between the input image $I(\boldsymbol{x})$ and a Gaussian kernel with standard deviation σ : $E_{img}(\boldsymbol{x}) = I(\boldsymbol{x}) * G_\sigma(\boldsymbol{x})$. Thus the image force exerted on the SOAC can be derived as:

$$\nabla E_{img}(\boldsymbol{r}(s)) = \nabla(I * G_\sigma)(\boldsymbol{r}(s)) = (I * \nabla G_\sigma)(\boldsymbol{r}(s)) \tag{3}$$

They point towards the center of bright intensity ridges (Fig. 3(b)).

The tangential stretching force $\boldsymbol{F}(\boldsymbol{r}(s))$ exerted at tips ($s = 0, L$) of a SOAC makes the curve elongate (Fig. 3(a)), and are defined as

$$\boldsymbol{F}(\boldsymbol{r}(s)) = \begin{cases} -\frac{\boldsymbol{r}'(s)}{|\boldsymbol{r}'(s)|} \cdot F(\boldsymbol{r}(s)), & \text{if } s = 0, \\ \frac{\boldsymbol{r}'(s)}{|\boldsymbol{r}'(s)|} \cdot F(\boldsymbol{r}(s)), & \text{if } s = L, \\ 0, & \text{if } 0 < s < L \end{cases} \tag{4}$$

where $F(\boldsymbol{r}(s))$ is the magnitude of the stretching force, which plays a critical role in determining how much a SOAC elongates. We will introduce a definition of $F(\boldsymbol{r}(s))$ in Section 2.1.1.

Combining the image and stretching forces, the overall external force field is:

$$\nabla\mathcal{E}_{ext}(\boldsymbol{r}(s)) = k_{img} \cdot (I * \nabla G_\sigma)(\boldsymbol{r}(s)) + k_{str} \cdot \boldsymbol{F}(\boldsymbol{r}(s)) \tag{5}$$

Minimizing the contour energy $\mathcal{E}(\boldsymbol{r})$ makes SOACs grow along the bright intensity ridges until the internal and external forces balance out and reach an equilibrium.

Next, we present the discrete representation of a SOAC and an iterative solution to curve evolution and convergence. A 3D SOAC can be represented as a linearly-ordered sequence of points, $\boldsymbol{r} = \{(x_{i,0}, x_{i,1}, x_{i,2})\},\ i = 0, \ldots, n-1$, with a uniform spacing δ. Let $\boldsymbol{X}_k = (x_{0,k}, \ldots, x_{i,k}, \ldots, x_{n-1,k})^T,\ k = 0, 1, 2$ be the vector containing all the kth-dimension coordinates of SOAC points, $\boldsymbol{X}_k^t$ at iteration t can be computed iteratively after deriving the Euler–Lagrange equation (Kass et al., 1988),

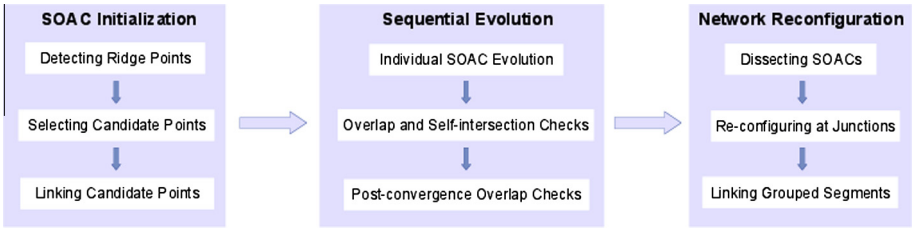

Fig. 2. Flow diagram of the proposed method, which consists of three main steps: SOACs initialization, sequential evolution and SOAC network reconfiguration.

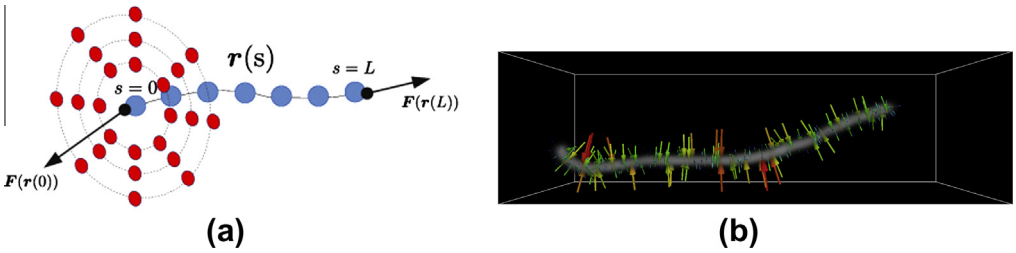

Fig. 3. (a) Illustration of a 3D Stretching Open Active Contour (SOAC). Two stretching forces $\boldsymbol{F}(\boldsymbol{r}(s))$ (arrows) are applied at two tips. The magnitude of $\boldsymbol{F}$ are determined by the local intensity contrast near tips, which are estimated by intensities at background sample points (red) and the intensity of the corresponding tip, all located on the plane $\boldsymbol{r}'(s)\cdot(\boldsymbol{x}-\boldsymbol{r}(s))=0, s=0,L$. (b) An illustration of the image force exerted on $\boldsymbol{r}(s)$. Arrows show samples of image gradient vectors for this synthetic meshwork image. Note that the vectors point toward the center of this bright intensity ridge.

$$\boldsymbol{X}_k^t=(\boldsymbol{A}+\gamma\boldsymbol{I})^{-1}(\gamma\boldsymbol{X}_k^{t-1}-\partial E_{ext}(\boldsymbol{X}_0^{t-1},\boldsymbol{X}_1^{t-1},\boldsymbol{X}_2^{t-1})/\partial x_k) \tag{6}$$

where $\boldsymbol{A}$ is the pentadiagonal banded matrix containing the internal continuity and smoothness constraints defined by (1). Since we use open curves, we introduce position and tangent discontinuity at two ends by setting $\alpha(0)=\alpha(L)=\beta(0)=\beta(L)=0$. $\boldsymbol{I}$ is the identity matrix, and γ is the viscosity coefficient that controls the step size for the dynamic evolution of the curve (Kass et al., 1988). The larger γ is, the smaller the step size will be. All SOACs are resampled to maintain the point spacing δ after each iteration. We considered a SOAC to be converged if every point drifts less than 0.05 voxels after 100 iterations.

2.1.1. Magnitude of stretching force

Because of variations in both foreground and background intensity, the magnitude of stretching force, $F(\boldsymbol{r}(s))$ in Eq. (4), needs to be adaptive. It is determined by the contrast between the intensity at the SOAC tip $I(\boldsymbol{r}(s)|_{s=0,L})$ and the intensity of local background around the tip $I_{lb}(\boldsymbol{r}(s)|_{s=0,L})$, divided by $I(\boldsymbol{r}(s)|_{s=0,L})$:

$$F(\boldsymbol{r}(s))=\frac{I(\boldsymbol{r}(s))-I_{lb}(\boldsymbol{r}(s))}{I(\boldsymbol{r}(s))}=1-\frac{I_{lb}(\boldsymbol{r}(s))}{I(\boldsymbol{r}(s))},\qquad s=0,L \tag{7}$$

We found that an effective way to measure the background intensity around a SOAC tip is to sample points in its vicinity, as shown in Fig. 3(a). More specifically, points are sampled in a plane that passes through the tip point and has its normal direction aligned with the curve's tangential direction at the tip. The sample points are located on concentric circles of radii ranging between $[R_{near}, R_{far}]$ with a sampling step angle $\phi=\pi/4$; the common center of the circles is the tip ($\boldsymbol{r}(0)$ or $\boldsymbol{r}(L)$). For the estimated local background intensity we use the average intensity of all the sample points.

The above local contrast estimation makes the SOAC's stretching force adaptable to the local contrast of the filament. Unlike previous methods that relied on a global intensity contrast (Li et al., 2009a; Smith et al., 2010; Xu et al., 2011), Eq. (7) generates stretching force that is adaptable according to local intensity contrast around SOAC tips. This can avoid under-segmentation for dim filaments in low-intensity backgrounds or over-segmentation for bright filaments in high-intensity backgrounds.

2.2. Automatic initialization of multiple SOACs

Since we employ a large number of SOACs to extract the network structure, automatic initialization of SOACs is necessary to eliminate the need for tedious manual initialization. By locating "ridge points" that are brighter in intensity, we can initialize SOACs on the centerlines of curvilinear structures.

2.2.1. Ridge points detection

Since the image intensity of an actin filament is higher than the background, points on the centerlines of filaments tend to be ridge points that have a local intensity maximum along at least two axis directions. Thus a ridge point $\boldsymbol{x}$ can be detected by searching for the plus-to-minus sign change in the spatial derivatives of image $\widetilde{I}(\boldsymbol{x})$, obtained after convolving input image $I(\boldsymbol{x})$ with a Gaussian kernel of standard deviation σ (Chang et al., 2001). Let $\partial_k\widetilde{I}(\boldsymbol{x})=\frac{\partial(G_\sigma(\boldsymbol{x})*\ I(\boldsymbol{x}))}{\partial x_k}$ denote the image derivative along the kth axis direction. $\boldsymbol{x}$ is a ridge point in that direction if

$$\exists m>0:\begin{cases}\partial_k\widetilde{I}(\ldots,x_k-\lfloor m/2\rfloor,\ldots)>\tau\\ \partial_k\widetilde{I}(\ldots,x_k+\lceil m/2\rceil,\ldots)<-\tau\\ |\partial_k\widetilde{I}(\ldots,x_k+l,\ldots)|<\tau,\quad \text{for all } l\in(-\lfloor m/2\rfloor,\lceil m/2\rceil)\end{cases} \tag{8}$$

where $\tau>0$ is a threshold to control the significance of the ridge. Here m,l are integers and $m>0$. This defines a ridge point on a ridge of m voxels wide, depending on how uniform the intensity is across the ridge. Fig. 4(a) and the second row of Fig. 5 show detected ridge points in 2D TIRFM and 3D confocal microscopy images, respectively.

2.2.2. Identifying candidate SOAC points

A candidate SOAC point should be a ridge point in at least two axis directions. In 3D, we define a SOAC candidate point *along x* axis as a point that is a ridge point *in* both of the other two axis directions, namely, *y* and *z*. Similarly, a SOAC candidate point *along*

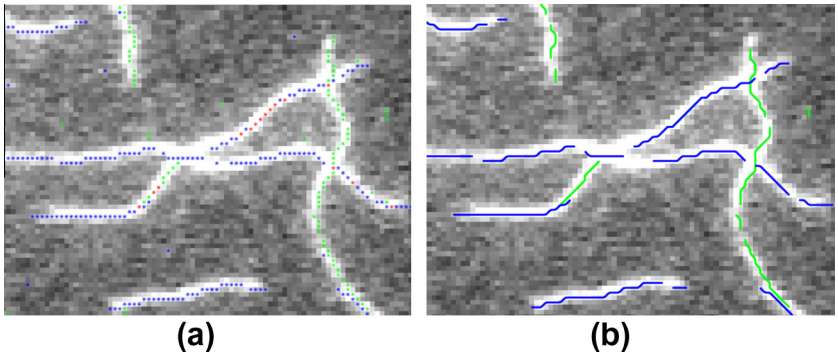

Fig. 4. Ridge points and initialized SOACs in a 2D TIRFM image. (a) Ridge points in x and y directions are labeled green and blue, respectively; the ones that are detected as a ridge point in both directions are labeled red. Ridge points in x are candidate points *along* y direction, and *vice versa*. (b) Initialized SOACs.

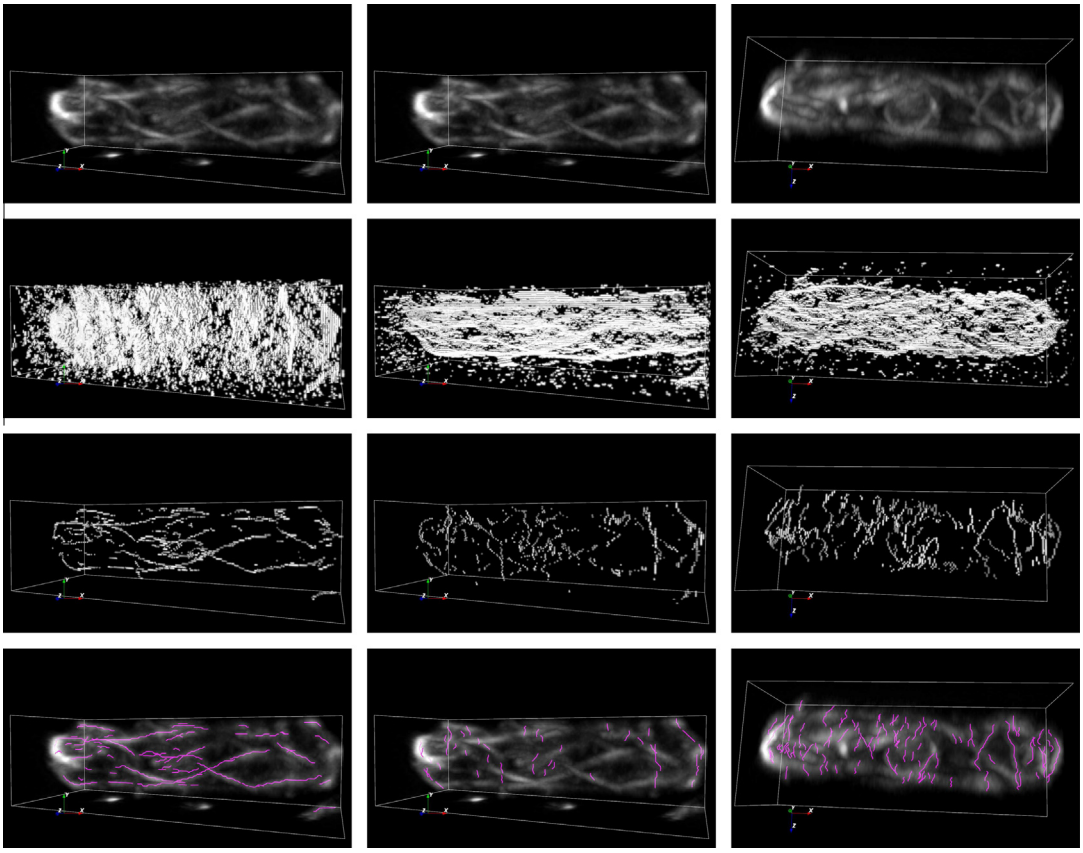

Fig. 5. SOAC initialization on a confocal microscopy image in Fig. 1(c) and (d). Image intensity is normalized to $[0, 1]$ followed by Gaussian smoothing with $\sigma = 1.0$. First row: Input image. Note the first two images share the same view point while the third one is from another view point. Second row: Detected ridge points ($\tau = 0.003$). Third row: SOAC candidate points generated from ridge points. Fourth row: SOACs initialized from candidate points. Columns from left to right correspond to initialization along x, y, and z axis directions, respectively. All images are rendered by Maximum Intensity Projection.

y is a ridge point in both x and z, and a SOAC candidate point *along* z is a ridge point in both x and y directions. The third row of Fig. 5 shows examples of identified candidate points in 3D. In 2D, candidate points *along* x are ridge points in y, and *vice versa* (Fig. 4(a)). Next, the candidate points are linked to form initial SOACs.

2.2.3. Linking candidate points to initial SOACs

In order to avoid ambiguities when grouping and linking SOAC candidate points, we initialize SOACs separately along each axis direction. That is, locally connected candidate points along x are linked to form an initial SOAC along x, and the process is repeated to form SOACs along y and z.

To start tracing an initial SOAC along an axis direction k, we first find an unused candidate point $\boldsymbol{x} = (\ldots, x_k, \ldots)$ that has the smallest kth coordinate. Then we repeatedly add a new point that has the kth coordinate incremented by 1, $\boldsymbol{x}' = (\ldots, x_k + 1, \ldots)$, and is 26-connected to $\boldsymbol{x}$, until no such unused candidate point can be found. Fig. 4(b) and the fourth row of Fig. 5 demonstrate the result of tracing initial SOACs along all three axis directions.

After the points on a SOAC are determined, we resample the SOAC with spacing δ. After resampling, if the resulting number of points on a SOAC is fewer than 5, the SOAC is not used (because the pentadiagonal banded matrix $\boldsymbol{A}$ requires SOACs having at least 5 points). The initial SOACs may be redundant but they will be

regulated to generate a concise and clear representation of the network structure after their subsequent evolution.

2.3. Sequential evolution of multiple SOACs

Initialized SOACs will evolve until convergence, one after another, in a sequential manner. We introduce a set of evolution regulation mechanisms to keep a SOAC free of overlap with others and itself.

2.3.1. Overlap checks

A brute-force strategy to detect overlap involves calculating the Euclidean distance from each point of an evolving SOAC to all converged SOACs: if the distance to a converged SOAC is smaller than a threshold, an overlap is detected. If this is done after each iteration, it would be too expensive and would dramatically slow the evolution process down. If, otherwise, we do overlap checking after all the SOACs have converged, then a lot of computation time would be wasted when SOACs extend onto structures that have already been extracted.

To detect overlaps efficiently, we adopt a balanced strategy as follows. We divide SOAC overlaps into two types. The first type are those with an overlapping portion that starts from a SOAC's tip point; these are usually introduced by SOAC elongation. The second type are those with an overlapping portion that does *not* include any SOAC tip points; these are usually caused by multiple SOACs' body drifting toward the same nearest bright intensity ridge. For the sake of efficiency, our strategy detects and eliminates overlaps of the first type during each SOAC's evolution in order to timely avoid redundant iterations for a SOAC to evolve onto a structure that has already been extracted by another converged SOAC. Meanwhile, since the first type of overlaps occur only at the evolving SOAC's tip points, we essentially check overlap that starts from two tips only. Overlaps of the second type are more expensive to detect since every point on a SOAC will need to be checked; therefore checks for this type of overlap are only done once after a newly converged SOAC is obtained.

Next we give details on overlap detection of both types. To facilitate discussion, we denote the current evolving SOAC by $\boldsymbol{r}_{ev}(s), s \in [0, L_{ev}]$, where s is the arc length parameter for SOAC $\boldsymbol{r}_{ev}$ and L_{ev} is the total length of $\boldsymbol{r}_{ev}$. We compare $\boldsymbol{r}_{ev}$ against those already-converged SOACs to check for overlap; let us denote such a converged SOAC by $\boldsymbol{r}_j(s), s \in [0, L_j]$.

2.3.1.1. Overlap checks during evolution. After each iteration of evolution for $\boldsymbol{r}_{ev}$, following Eq. (6), the Euclidean distances from the two tips of $\boldsymbol{r}_{ev}$ to $\boldsymbol{r}_j$ is computed; if the distances are greater than a threshold $D_{min} = 2$ voxels, no overlap between $\boldsymbol{r}_{ev}$ and $\boldsymbol{r}_j$ is detected, otherwise, the tip of $\boldsymbol{r}_{ev}$ that overlaps with $\boldsymbol{r}_j$ is identified. Without loss of generality, suppose the tip that overlaps with $\boldsymbol{r}_j$ is the head of $\boldsymbol{r}_{ev}$, $\boldsymbol{r}_{ev}(s)|_{s=0}$. Starting from this tip, we proceed to check more inner points on $\boldsymbol{r}_{ev}$ until we reach a point P that has a distance to $\boldsymbol{r}_j$ greater than the threshold D_{min}, as shown in Fig. 6. It is possible that the tail tip of $\boldsymbol{r}_{ev}$ is reached and we still have not found such a point P; this means that all points of $\boldsymbol{r}_{ev}$ overlap with $\boldsymbol{r}_j$, thus $\boldsymbol{r}_{ev}$ is redundant and is deleted. If a point P is found before reaching the tail tip (like the case in Fig. 6), we delete the overlapping part on $\boldsymbol{r}_{ev}$, starting from the overlapping tip to the first non-overlapping point P, and then find the point Q on $\boldsymbol{r}_j$ that is the closest to P. We then make Q the new head tip for $\boldsymbol{r}_{ev}$.

It is possible that Q is one of the end tip points of $\boldsymbol{r}_j$; in this case, we check the angle between the tangent vector of $\boldsymbol{r}_{ev}$ at Q and the tangent vector of $\boldsymbol{r}_j$ at Q, and if the angle is greater than a direction threshold θ indicating good continuity between $\boldsymbol{r}_{ev}$ and $\boldsymbol{r}_j$ at Q, $\boldsymbol{r}_{ev}$ will be concatenated with $\boldsymbol{r}_j$ and the merged SOAC will evolve again. Otherwise, if the angle is smaller than the threshold thus

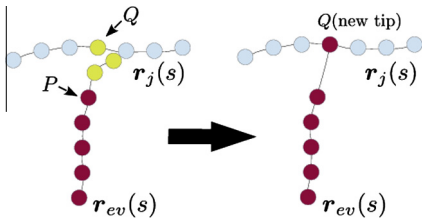

Fig. 6. Illustration of checking SOAC collision during evolution. (Left) The head tip (yellow) of an evolving SOAC $\boldsymbol{r}_{ev}$ collides with an already-converged SOAC $\boldsymbol{r}_j$. The overlapping part on $\boldsymbol{r}_{ev}$ is identified, and P is the first point on $\boldsymbol{r}_{ev}$ that does not overlap with $\boldsymbol{r}_j$ (having a distance from $\boldsymbol{r}_j$ greater than the threshold D_{min}). Q is the point on $\boldsymbol{r}_j$ that is the closest to point P. (Right) Deletion of the overlapping part, and making Q the new head tip for $\boldsymbol{r}_{ev}$. After forming the T-junction, $\boldsymbol{r}_{ev}(s)$ will be resampled (not shown).

$\boldsymbol{r}_{ev}$ and $\boldsymbol{r}_j$ are not to be concatenated, or if Q is not a tip point of $\boldsymbol{r}_j$ (e.g. as shown on the right side in Fig. 6), a corner- or a T-junction is formed at the intersection point Q.

Once a junction is formed, we set the stretching force at that tip for $\boldsymbol{r}_{ev}$ to be zero, and pin down the tip at the junction and preventing $\boldsymbol{r}_{ev}$ from stretching further. It is possible that the junction is actually a cross-junction where two filaments intersect. Although not allowing the evolving SOAC to stretch across the junction, it is expected that if the filament continues across the junction, the remaining part will be extracted by another SOAC that is initialized on that part of the filament.

For images containing filament loops (e.g., Fig. 1(d)), a SOAC may intersect with itself. After a SOAC's each iteration, we check if there are two SOAC points on the curve that are sufficiently apart along the curve length but are spatially very close, i.e. $\|\boldsymbol{r}_{ev}(s)|_{s=l_1} - \boldsymbol{r}_{ev}(s)|_{s=l_2}\|_2 < D_{min}$ and $|l_1 - l_2| > \kappa$ where κ is a threshold typically set to be one tenth of the curve length. If so, we divide $\boldsymbol{r}_{ev}$ into 3 sub-SOACs which correspond to the parts of $\boldsymbol{r}_{ev}(s)$ with $s \in [0, l_1), s \in [l_1, l_2), s \in [l_2, L_{ev}]$, respectively. The three sub-SOACs will re-evolve on their own, and the second sub-SOAC will be a closed contour.

2.3.1.2. Post-convergence overlap check. After convergence of the current SOAC $\boldsymbol{r}_{ev}$, we check whether its body overlaps with any other already-converged SOACs. This is the second type of overlap, which features an overlapping part that does not include any tip point of $\boldsymbol{r}_{ev}$, thus would not have been detected during the SOAC's evolution. If an overlap is detected in the middle of $\boldsymbol{r}_{ev}$'s body (i.e. no tips included), $\boldsymbol{r}_{ev}$ is dissected into several sub-SOACs; any sub-SOAC that overlaps with another SOAC will be discarded, and any remaining non-overlapping sub-SOACs will be treated as new independent SOACs and will evolve again on their own and allowed to form new T-junctions. As discussed above, this post-convergence body overlap check is necessary in that the body of a SOAC may drift away from the original intensity ridge that initialized the SOAC. This happens when the SOAC was initialized on some faint curvilinear structure but drifted toward some other brighter structures nearby during its evolution.

2.3.1.3. Sequential vs. simultaneous evolution. Here we add some notes comparing the proposed regulated sequential evolution of multiple SOACs with simultaneous evolution of multiple SOACs. In our previous work Xu et al. (2011), all of the initialized SOACs were allowed to evolve simultaneously. Although we obtained good results with simultaneous evolution, we found that it sometimes causes improper merging among un-converged SOACs and complicates overlap detection. In this work, by separating different overlap scenarios and leaving the more time-consuming body overlap check in a post-convergence step, sequential evolution is a faster and more robust method.

2.3.1.4. SOAC network reconfiguration. Reconfiguration of the converged SOAC network addresses the issue of filament continuity at junctions and extraction of network topology. The network of converged SOACs may not be topologically correct for three reasons. First, the sequential evolution is designed to form T- or corner-junctions only, without any cross-junctions or higher-degree junctions. Second, because SOACs evolve one after another, those SOACs that start early have a better chance to extend along multiple branches. When they elongate across intersections of branches, artificial corners may occur. For example, suppose we have a "T"-shape network composed of a horizontal filament and a vertical branch. If the first SOAC to evolve is initialized on the lower branch of "T," it may continue to grow onto one of the horizontal branches and exhibit an non-physical corner around the junction. Lastly, artificial "sharp turns" can also be introduced by the influence of strong external force field such as a very bright nearby branch.

To retrieve a more physical network topology, we use the fact that junctions in cytoskeletal filament networks typically involve straight fibers crossing one another or side branches forming off straight filaments. We use this smoothness constraint to compute the optimal branch-connection configuration at junctions using Normalized Cuts (Shi and Malik, 2000).

Normalized Cuts performs data clustering based on a graph partitioning algorithm. It minimizes a global criterion which measures both the total dissimilarity between different subgroups of a graph as well as the total similarity within the subgroups. We first compute the direction vectors $\{\boldsymbol{v}_i\}$ for each SOAC branch at a junction. Under the graph partitioning framework, the continuity (or similarity) among branches can be represented as a complete weighted graph $G = (V, E)$. The direction vectors $\{\boldsymbol{v}_i\}$ are nodes and the edge weight w_{ij} is the normalized angle between $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$, which specifies how continuous the two branches are across the junction. The affinity matrix $\boldsymbol{W} = \{w_{ij}\}$ encodes the mutual smoothness, where

$$w_{ij} = \begin{cases} \frac{1}{\pi}\arccos\left(\frac{|\boldsymbol{v}_i \cdot \boldsymbol{v}_j|}{\|\boldsymbol{v}_i\|\|\boldsymbol{v}_j\|}\right), & \text{if } i \neq j, \\ 1, & \text{if } i = j \end{cases} \tag{9}$$

Given the affinity matrix, the Normalized Cuts algorithm partitions all the branches into groups of two branches that are smooth and continuous across the junction.

Occasionally there may be cross-junctions (at which two straight filaments intersect) or even higher-degree junctions that are represented by multiple nearby T-junctions in the network (Fig. 7(a-b)). In order to reconfigure and detect such junctions, we first perform a connected component analysis to cluster T-junctions that are very close to each other to form a single higher-degree junction (Fig. 7(c)). Then we dissect the converged SOACs into segments at each junction (Fig. 7(d)), where we recursively apply the 2-way Normalized Cuts to obtain multiple groups of 2 smooth segments which are linked together afterwards to form new SOACs (Fig. 7(e)). Fig. 7(f) shows an example of the reconfigured SOACs and detected filament junctions.

2.4. Program output

The output of the above framework includes multiple SOACs, each representing an individual actin filament or cable in the cytoskeleton network. For each SOAC, we have densely sampled points on the curve as well as image intensity along the curve. From these we can compute distributions of orientation, curvature, and intensity along filaments. Besides the geometry and intensity information, we also have the network topology recording all the junction locations and the connectivity information among filaments.

3. Validation on simulated and experimental images

We validate the method using simulated and experimental images of actin cable meshwork in fission yeast imaged by spinning disk confocal microscopy. The method was implemented in C++ along with Insight Toolkit (ITK),[1] Visualization Toolkit (VTK)[2] and Qt.[3] The program was tested on a PC workstation with Intel Xeon 3.00 GHz and 4 Gb memory. The source code of the method is freely available upon request; please see http://www.cse.lehigh.edu/~idealab/soax.html for updates on the status of the project.

3.1. Tests on simulated images

The goal of testing on 3D simulated images is to quantitatively evaluate the accuracy and robustness of our method. We obtained segmentation results of real confocal microscopy images of fission yeast actin structures labeled by GFP-CHD, using a semi-automatic tool (Smith et al., 2010), and use these results to generate synthetic meshwork images. The manual segmentation results themselves also serve as ground truth when evaluating the automated segmentation results from our program. These results, like SOACs, are represented as open curves consisting of sequences of points.

3.1.1. Generation of simulated images

We collected 13 confocal microscopy images with their corresponding manual segmentation results, 9 of which are images of actin cable meshworks and the rest contain actin contractile ring structures. From the manual result of each image, we generate 14 synthetic images with different noise levels. Thus the test bed contains $13 \times 14 = 182$ noisy synthetic images (see Fig. 8 for samples).

Here we describe in detail how a synthetic image is generated. *In order to generate synthetic images that closely simulate the real experimental images we used, we estimated the mean intensity of background regions in 13 experimental images. The estimated background intensity is around 230. The first step in generating a synthetic image is to start with an image with zero background intensity. Then, intensities of ground-truth centerline voxels from a corresponding real experimental image, minus the estimated average background intensity value 230, are used to set foreground intensities in the generated image. Next, we apply a Point Spread Function (PSF) using a Gaussian kernel with* $\sigma_f = (1.5, 1.5, 4.32)^T$ *on this "centerline image". The maximum intensity of the blurred image is rescaled to match the maximum intensity of the corresponding real experimental image minus 230. In the last step, we add a background Gaussian noise with* $\mu_b = 230$, *and* $\sigma_b = 2, 4, 6, \ldots, 26, 28$.

3.1.2. Results and evaluation

We measure the statistics of vertex error and Hausdorff distance between resultant SOACs $\{\boldsymbol{r}_c\}$ from our program and the ground truth $\{\boldsymbol{r}_{gt}\}$ used to generate the synthetic images. Here we treat all computed SOAC points $\{\boldsymbol{x}_c\}$ as one set R_c and all ground truth points $\{\boldsymbol{x}_{gt}\}$ as another set R_{gt}. The vertex error is defined similar to "error per face" in (Narayanaswamy et al., 2010),

$$d_V(R_c, R_{gt}) = \frac{1}{2|R_c|}\sum_{\boldsymbol{x}_c \in R_c}\min_{\boldsymbol{x}_{gt} \in R_{gt}}\|\boldsymbol{x}_c - \boldsymbol{x}_{gt}\| + \frac{1}{2|R_{gt}|}\sum_{\boldsymbol{x}_{gt} \in R_{gt}}\min_{\boldsymbol{x}_c \in R_c}\|\boldsymbol{x}_{gt} - \boldsymbol{x}_c\| \tag{10}$$

and the Hausdorff distance between R_c and R_{gt} is

$$d_H(R_c, R_{gt}) = \max\{\max_{\boldsymbol{x}_c \text{in} R_c}\min_{\boldsymbol{x}_{gt} \in R_{gt}}\|\boldsymbol{x}_c - \boldsymbol{x}_{gt}\|, \max_{\boldsymbol{x}_{gt} \text{in} R_{gt}}\min_{\boldsymbol{x}_c \in R_c}\|\boldsymbol{x}_{gt} - \boldsymbol{x}_c\|\} \tag{11}$$

[1] www.itk.org. ITK is used for routine image processing procedures.
[2] www.vtk.org. VTK is used to visualize SOAC evolution.
[3] qt-project.org. Qt is used to assist with building the user interface.

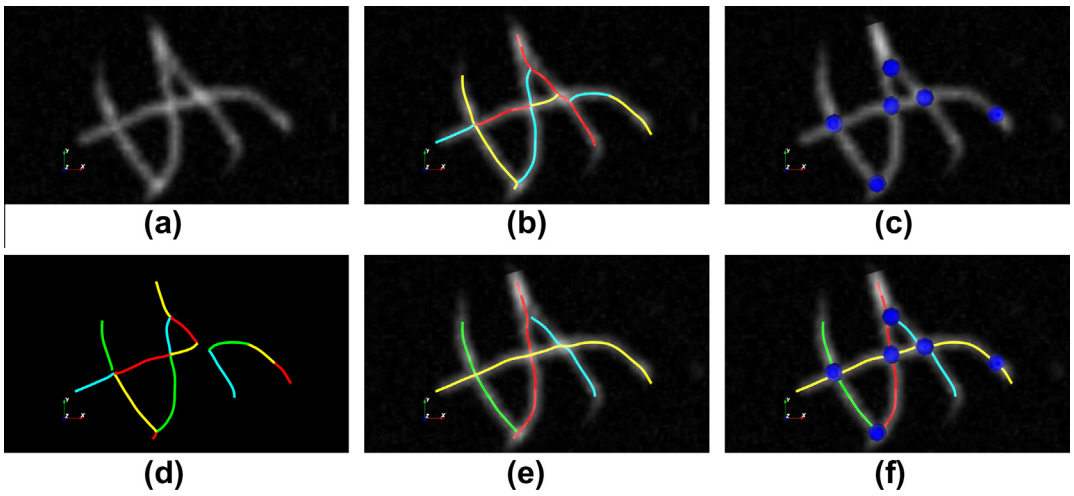

Fig. 7. Network reconfiguration on a 3D synthetic image rendered by Maximum Intensity Projection. (a) One part of the 3D image. (b) Converged SOACs. Different colors indicate different SOACs. (c) Cluster nearby T-junctions into a single higher-degree junction (blue spheres). Note the rightmost junction is not a false positive as there is a branch perpendicular to this image plane (not shown). (d) Dissect converged SOACs into segments (shown in different colors) at junctions. (e) New SOACs formed by reconfiguration and linking of grouped segments. (f) Reconfigured SOACs and detected junctions.

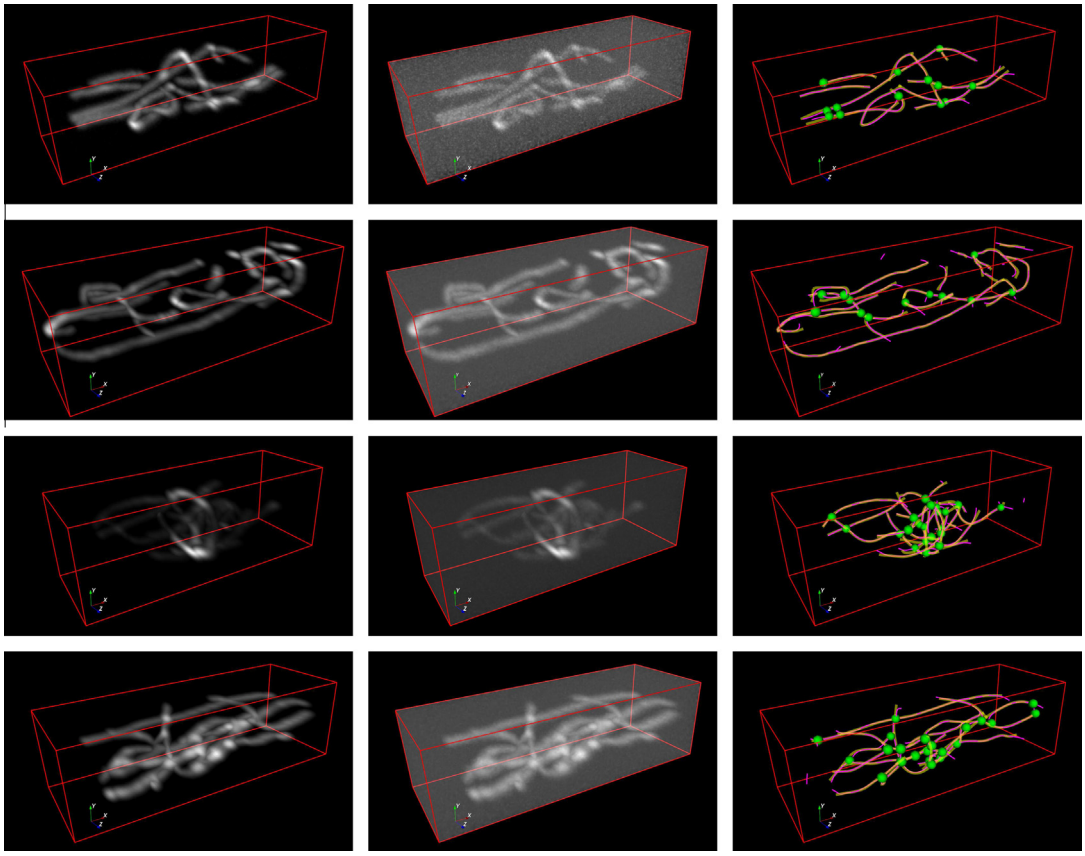

Fig. 8. Samples of clean image, noisy image and resultant SOACs extracted from noisy image overlaid with ground truth SOACs. Left column: 4 clean synthetic images with various foreground intensities copied from corresponding real confocal microscopy images. Middle column: 4 generated noisy images corrupted by additive Gaussian noise ($\sigma_b = 12$). Right column: resultant SOACs extracted from noisy images using our program (translucent purple) overlaid with ground truth (light yellow). Junctions are shown by green spheres. Images are rendered by Maximum Intensity Projection; red lines show boundary of the image volume.

We compute the mean and standard deviation of vertex error and of Hausdorff distance and plot them against the Signal-to-Noise Ratio (SNR) of images. The SNR is defined as

$$SNR = \frac{\mathrm{E}(I_s(\boldsymbol{x}) - \mu_b)}{\sigma_b} \tag{12}$$

where the numerator is the mean intensity of foreground voxels subtracted by mean intensity of background noise, and the denominator σ_b is the standard deviation of the background Gaussian noise. Here we define foreground to be voxels that are within $2\sigma_f$ (the standard deviation of the Gaussian PSF) to the centerline of filaments.

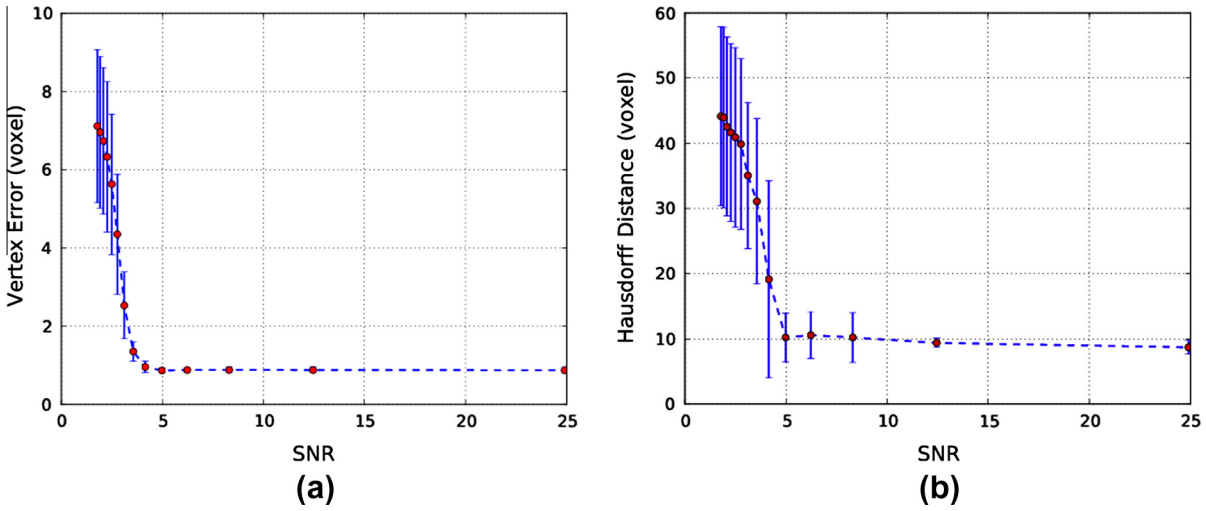

Fig. 9. Statistics of vertex error (a) and Hausdorff distance (b) versus image Signal-to-Noise-Ratio (SNR) for experiments on 182 simulated images.

Table 1
Parameters for SOAC initialization and evolution used in experiments on synthetic images. The intensities of all images are scaled to be in the range [0, 1].

γ	α	β	k_{img}	k_{str}	σ	τ	D_{min}	θ	R_{near}	R_{far}
2	0.01	0.1	1	0.5	1	0.005	2.0	$2\pi/3$	3	5

Fig. 9 shows the plots (a) vertex error d_V vs. SNR, and (b) Hausdorff distance d_H vs. SNR. One can see that our algorithm performs well even on very noisy images; the vertex error is around one voxel when the SNR is greater than 4. The Hausdorff distances between our result and ground truth are around 10 voxels when the SNR is greater than 5. The increase in false positives and false negatives becomes an issue when SNR is less than 4. The estimated SNR values for real experimental images we used range from 4.85 to 6.52, with a mean of 5.72. Some sample segmentation results from our program are displayed visually in Fig. 8. The experimental parameter settings are listed in Table 1.

3.1.3. Tests for rotational sensitivity

We also tested the rotational sensitivity of our method. We took the set of 182 simulated images and rotated them by 45 degrees around the z-axis. We run our algorithm on the rotated images and extracted centerlines. The extraction results were compared against the rotated centerlines extracted from the original simulated images. We measured the difference between the two results using vertex error and Hausdorff distance. As shown in Fig. 10, the disparity measured by vertex error between results on rotated simulated images and rotated results on original simulated images is about 0.5 voxels when the SNR is greater than 5, and Hausdorff distance is below 10 voxels. Rotational sensitivity increases when the SNR is less than 5. When the SNR drops even further, both measurements first drastically increase and then drop; the drop is because both results have a large number of closely spaced false positives when the SNR gets very low.

3.2. Tests on spinning disk confocal microscopic images

We also tested our algorithm directly on real images of fission yeast cells expressing fluorescent protein GFP-CHD that attaches to the sides of actin filaments.[4] These cell images acquired by spinning disk confocal microscopy show actin cables in non-dividing cells (Fig. 1(c)), actin contractile rings and medial actin cables in dividing cells (Fig. 1(d)). The voxel spacing is 0.0694 μm in the x and y directions and 0.2 μm in the z direction. The images were first interpolated using the sinc function with Lanczos window to make the voxel spacing equal to 0.0694 μm in all directions.

Fig. 11 shows results on segmenting 3D networks of actin cables. The parameter settings for our algorithm in these tests are shown in Table 2. Since it is still an open question how to get reliable ground truth for these types of real experimental images, we first evaluate our results qualitatively by visually inspecting the results against the image and counting the number of false negatives and false positives, for four different experimental images (left column of Fig. 11). If a filament in an image is not segmented by any SOAC, then it counts as a false negative; if a SOAC does not appear to segment any filament, then it counts as a false positive. The number and the percentage of false positives and false negatives against the total number of SOACs are shown in Table 3.

To get quantitative evaluation on the accuracy of our result, we also compare the automated segmentation results with manual segmentation using the same disparity measures as for the synthetic images. We used a semi-automatic tool proposed in (Smith et al., 2010) and asked an expert to use the tool to manually delineate actin cables that he can identify. Note that the manual segmentation is often incomplete, missing some cables that are near cell boundary or have faint intensity. In comparison, our algorithm was able to extract some cables that are present in the image but are missed by manual segmentation. For quantitative comparison, we selected all SOACs that have a corresponding manually segmented curve for the same filament, and calculated the vertex error and Hausdorff distance between the automatic SOACs and the manual curve counterparts (Table 4).

Fig. 12 shows the result on an image of dividing yeast cells that assemble a contractile actin ring. Owing to the mechanism for self-intersection detection (described in "Overlap Checks during Evolution" step, Section 2.3.1), a closed-form SOAC (highlighted in Fig. 12(b)) is formed to capture the ring structure in the center of the cell. In other scenarios, a closed-form SOAC could also form through reconfiguration, where re-grouped segments form a loopy SOAC.

We also tested our algorithm on a 2D TIRFM image of actin filaments growing on a glass slide (Fig. 1(a)) and a 3D network of actin filaments cross-linked by α-actinin (Fig. 1(b)). The result is

[4] The images were provided by I-Ju Lee and Jian-Qiu Wu at the Ohio State University.

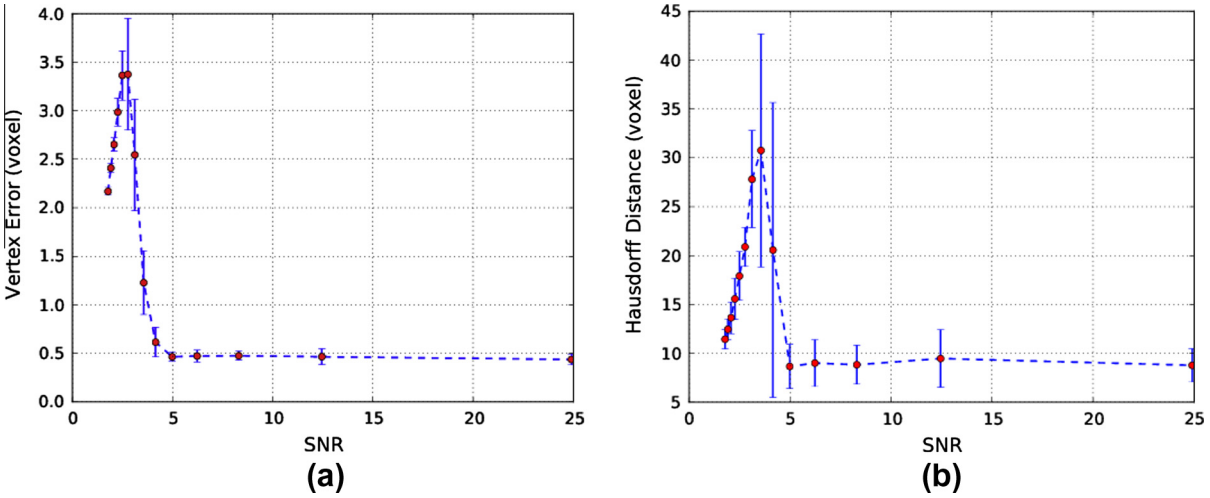

Fig. 10. Disparities between results on rotated simulated images and rotated results on original simulated images versus SNR. (a) Vertex error, (b) Hausdorff distance.

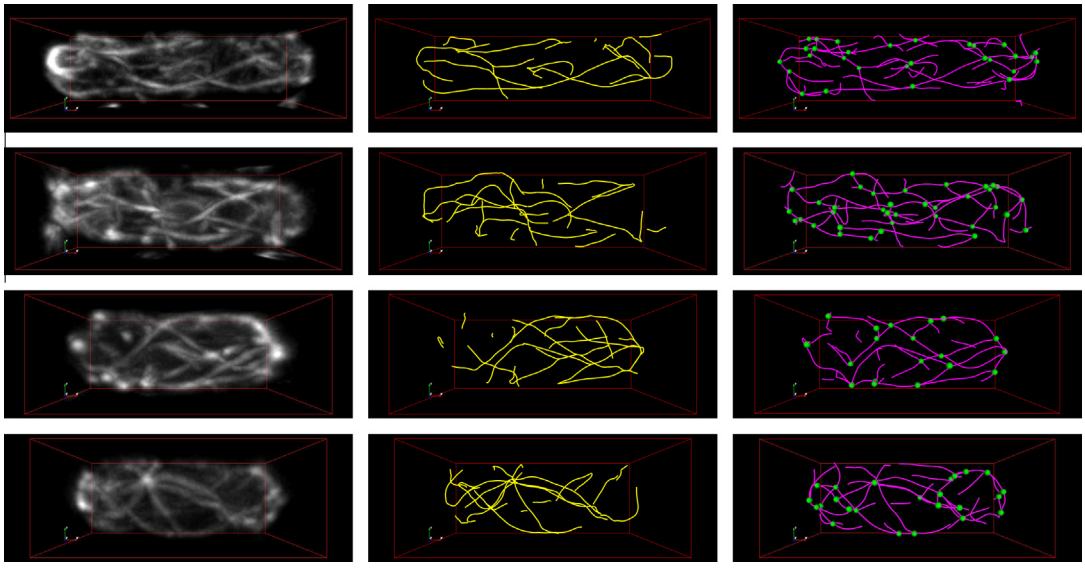

Fig. 11. Results on confocal microscopy images of actin cables in yeast cells labeled by GFP-CHD and treated with CK-666 (Nolen et al., 2009). Left column: Four experimental images rendered by Maximum Intensity Projection along z direction. Middle column: Manual segmentation with the semi-automatic tool in (Smith et al., 2010). Right column: Results using our proposed method with junctions shown by green spheres.

Table 2
Parameters for SOAC initialization and evolution used in experiments on real experimental images. The intensities of all images are scaled to be in the range $[0, 1]$.

γ	α	β	k_{img}	k_{str}	σ	τ	D_{min}	θ	R_{near}	R_{far}
2	0.01	0.4	1	0.7	0.5	0.01	1.0	$2\pi/3$	3	5

Table 3
Number of false negatives (FN) and false positives (FP) counted by inspecting the results (right column in Fig. 11) against the image (left column in Fig. 11). The total number of resultant SOACs for each image is shown in the fourth column (# SOACs). The percentage of (FP + FN) is shown in the last column.

Image no.	FP	FN	# SOACs	%(FP + FN)
1	6	4	41	24.4
2	4	2	34	17.6
3	2	4	22	27.3
4	1	2	26	11.5

Table 4
Vertex error and Hausdorff distance between automated segmentation results and manual segmentation results on 4 different experimental images.

Image no.	Vertex error	Hausdorff distance
1	2.607	29.240
2	2.193	18.170
3	2.019	18.147
4	2.501	19.053

shown in Fig. 13. Visual comparison between our results and the images shows that our method extracts the geometry and topology of these curvilinear networks accurately in the presence of foreground and background intensity variations.

In terms of efficiency, our 3D network structure extraction method is much more efficient than the single-SOAC, semi-automatic tool in (Smith et al., 2010). Using the single-SOAC method, it typically took 20–30 min for a human to segment the

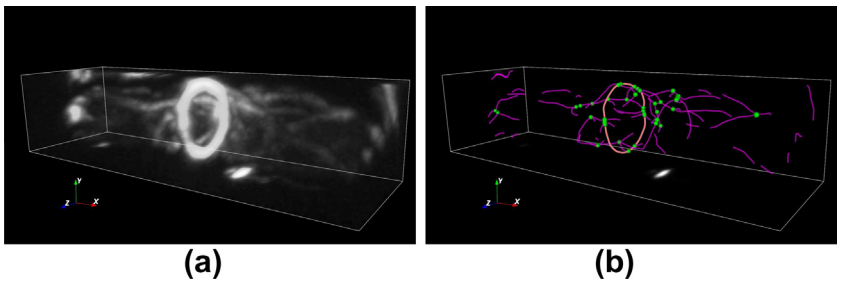

Fig. 12. Result on confocal microscopy images of dividing fission yeast cells that assemble a contractile actin ring. (a) Image rendered by Maximum Intensity Projection. (b) Segmentation result using our method; the closed-form SOAC capturing the ring is highlighted.

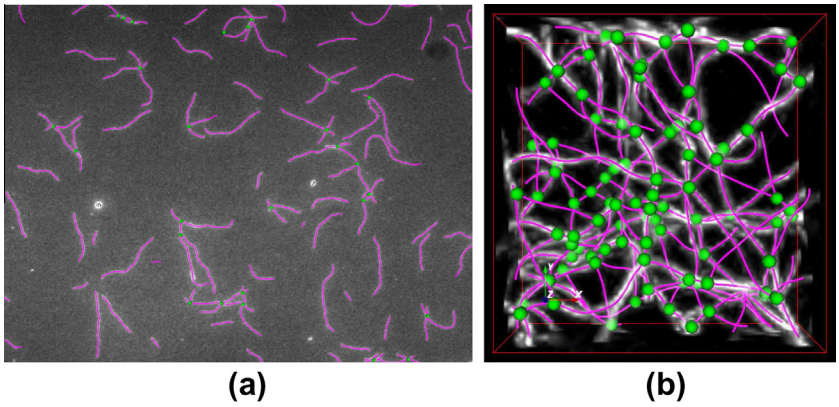

Fig. 13. Results on a 2D TIRFM image of actin filaments growing on a glass slide (a) and a 3D network of actin filaments cross-linked by α-actinin (b). Both figures show the segmentation result overlaid on top of the original image. The original images without overlay can be seen in Fig. 1(a) and Fig. 1(b) (Falzone et al., 2012).

cytoskeletal network from a 3D image like the ones shown in Fig. 11. By our method, the time is reduced by more than 10-fold, since our algorithm completes a segmentation in under 2 min. Furthermore, our method also outputs information about the topology of the extracted network.

4. Discussion

In the proposed method, we exploit the curvilinear nature of SOACs and use them to extract curvilinear structures in a network. By detecting and reconfiguring T-junctions, our method not only corrects non-physical sharp corners in SOACs but also obtains topology information about the extracted network including the connectivity pattern among actin filaments, the position and degree of junctions, and other measurements that can be computed from these.

Compared to other methods that use open Snakes for tracing filamentous structures such as microtubules in cellular electron tomography (Nurgaliev et al., 2010) and neurons in confocal microscopy images (Wang et al., 2011), the novel aspects in our work include a 3D ridge point based initialization scheme, a principled way to exert adaptive stretching forces at a snake's tips, regularization mechanisms to resolve SOAC collision and overlap issues, and also graph-cut based reconfiguration of a SOAC network which re-groups SOAC segments at junctions according to physical constraints. Thus, our method is applicable to segmentation of general curvilinear network structures with complex topology whereas previous methods were often tested on filamentous structures that are mostly parallel to each other or form simple tree-like topology (e.g. microtubules and neurons).

The computational efficiency of our algorithm depends on the quantity of initialized SOACs. On a PC workstation with 3.00 GHz CPU and 4 Gb RAM, the program completes the extraction of a 3D cytoskeletal network from a synthetic image of size $200 \times 60 \times 60$ voxels in 20 s; it takes longer (usually less than 90 s) to segment a real confocal microscopy image of similar size, since more SOACs are initialized on noisy experimental images. We expect that run time will decrease significantly with a parallel implementation.

In our proposed method, SOACs are always initialized on the intensity ridges. The key parameter to this initialization scheme is τ, which controls the number of initialized SOACs. Initialization fails only due to inappropriate τ, which may cause initialization in the background (τ is too small) or no initialization on a foreground filament (τ is too large). In the first case, the SOACs initialized in the background will shrink then be deleted because the stretching force is near zero in the uniform background area. Failing to extract a filament is rare in our experiments, since as long as one section of a filament receives a SOAC initialization, the SOAC can elongate to extract the whole filament.

Several mechanisms in our framework that facilitate a concise network representation help reduce segmentation time. First, detection of collision and redundancy prevents a SOAC from elongating to a region already covered by other SOACs. Second, by differentiating two types of overlap, we keep the overhead of regulation low: we need only check overlap from end points after each iteration of evolution and postpone the more expensive body overlap check to after convergence. Third, we can also sort the initial SOACs based on their length and let those longer ones evolve first to reduce the total evolution time.

Parameter tuning is often needed when different types of experimental images need to be analyzed. The most critical two parameters are τ (Eq. (2)) and k_{str} (Eq. (8)). τ influences the quantity of initialized SOACs and is chosen empirically. In a rescaled image

with intensity values in $[0,1]$, depending on the contrast between background and foreground filaments, the appropriate value of τ can range from 10^{-4} to 0.1. If τ is large, fewer image points will be considered as a ridge point thus fewer SOACs will be initialized. In our experiments, we have a set of τs that work well for the experimental images. For instance, $\tau \in [0.005, 0.015]$ for the actin network images we tested on. Given a new type of experimental image, one can adjust this parameter in a semi-automatic fashion. k_{str} acts as an additional global parameter that controls the final magnitude of stretching force. One can use a larger k_{str} if the image is under-segmented and a smaller value if SOACs are over-elongated.

As discussed in Section 2.1.1, the radius range $[R_{near}, R_{far}]$ specifies the range where local background intensities are sampled. They are empirically chosen based on the size and density of filaments which depend on the type of image and the point spread function (PSF) of the imaging process. Specifically, the targeted filaments in our experimental images of actin cables have radii $\leqslant 3$ voxels, so we set $R_{near} = 3$ to set apart the foreground and background region. R_{far} is influenced by the density of the curvilinear structures in the image. We do not want nearby foreground voxels to be included for background intensity estimation, so a small value is preferred for R_{far} when the curvilinear network is dense. Keeping R_{far} not too large makes the intensity estimation local. For the type of actin cable images used, we set $R_{far} = 5$ voxels, since the distance between filaments are usually greater than this value when they are not crossing each other. When they do cross, a small magnitude of stretching force is usually generated because areas near the junction are mostly occupied by foreground filaments. This makes the snake stop elongating at junctions but this works fine as the sequential evolution step of our algorithm discourages a SOAC to go across junctions in the first place.

The viscosity coefficient γ, controlling the step size, is set to 2 for the rescaled image where the intensity ranges from 0 to 1, in all the experiments. γ is inversely proportional to the step size: the larger γ is, the smaller step size will be. For typical images that have sufficiently high SNR to allow visual recognition of filamentous structures, we found any $\gamma \geqslant 2$ can converge without any stability issue if we re-initialize the model after each iteration. Since larger step size can give quicker convergence, we favor smaller γ here. However, we found $\gamma < 2$ which gives larger step size, could cause oscillation around the solution. Therefore, we set $\gamma = 2$ for all the experiments and resample the contour after each iteration of evolution to maintain a uniform spacing, so that the overlap check *during* evolution is accurate.

Our proposed method is semi-automatic in the sense that parameter tuning is often needed for different types of experimental images. Manual testing in search of appropriate set of parameters is often needed for each *type* of images. However, the set of parameters can be fixed for a certain type of experimental images which share the same image characteristics. One only needs to tune parameters for one example of a new type of images, and then can utilize these same parameters for batch segmentation of all such images without any human intervention.

5. Conclusions

We have proposed and implemented a semi-automated method to extract the skeleton of actin meshworks in 3D confocal microscopy images. The method is also generally applicable to other images of curvilinear networks with low SNR. Validation on synthetic images shows the robustness of the proposed method; experiments on real experimental images show the potential of our method in extracting 3D cytoskeletal networks accurately and efficiently.

Acknowledgements

We thank Dr. Jian-Qiu Wu and I-Ju Lee for providing images used in our analysis. We thank Dr. Hongsheng Li, Dr. Tian Shen, Dr. Eddy Yusuf, Feng-Ching Tsai and Dr. Gijsje Koenderink for their help in system implementation and evaluation. This work is supported by NIH Grant R01GM098430.

References

Altinok, A., El-Saban, M., Peck, A., Wilson, L., Feinstein, S., Manjunath, B.S., Rose, K., 2006. Activity analysis in microtubule videos by mixture of hidden markov models. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 1662–1669.

Baradet, T., Haselgrove, J., Weisel, J., 1995. Three-dimensional reconstruction of fibrin clot networks from stereoscopic intermediate voltage electron microscope images and analysis of branching. Biophysical Journal 68, 1551–1560.

Basu, S., Dahl, K., Rohde, G., 2013. Localizing and extracting filament distributions from microscopy images. Journal of Microscopy 250, 57–67.

Basu, S., Mukherjee, D., Acton, S., 2007. Implicit evolution of open ended curves. In: IEEE International Conference on Image Processing, 2007 (ICIP 2007), pp. 261–264.

Beil, M., Braxmeier, H., Fleischer, F., Schmidt, V., Walther, P., 2005. Quantitative analysis of keratin filament networks in scanning electron microscopy images of cancer cells. Journal of Microscopy 220, 84–95.

Berger, M.O., Mohr, R., 1990. Towards autonomy in active contour models. In: Proceedings of the 10th International Conference on Pattern Recognition, 1990. IEEE, pp. 847–851.

Caselles, V., Kimmel, R., Sapiro, G., 1997. Geodesic active contours. International Journal of Computer Vision 22, 61–79.

Chang, S., Kulikowski, C., Dunn, S., Levy, S., 2001. Biomedical image skeletonization: a novel method applied to fibrin network structures. Studies in Health Technology and Informatics 84, 901–905.

El-Saban, M., Altinok, A., Peck, A., Kenney, C., Feinstein, S., Wilson, L., Rose, K., Manjunath, B., 2006. Automated tracking and modeling of microtubule dynamics. In: 3rd IEEE International Symposium on Biomedical Imaging: Nano to Macro, 2006, pp. 1032–1035.

El-Saban, M.A., Manjunath, B.S., 2005. Tracking curvilinear structures using active contours and application to microtubule videos. Technical Report, <http://vision.ece.ucsb.edu/publications/saban_icip2005.pdf>.

Falzone, T.T., Lenz, M., Kovar, D.R., Gardel, M.L., 2012. Assembly kinetics determine the architecture of α-actinin crosslinked F-actin networks. Nature Communications 3, 861.

Frangi, A., Niessen, W., Vincken, K., Viergever, M., 1998. Multiscale vessel enhancement filtering. Medical Image Computing and Computer-Assisted Interventation–MICCAI'98, 130–137.

Fujiwara, I., Vavylonis, D., Pollard, T.D., 2007. Polymerization kinetics of ADP- and ADP-Pi-actin determined by fluorescence microscopy. Proceedings of the National Academy of Sciences 104, 8827–8832.

Hadjidemetriou, S., Toomre, D., Duncan, J., 2005. Segmentation and 3d reconstruction of microtubules in total internal reflection fluorescence microscopy (tirfm). In: Duncan, J., Gerig, G. (Eds.), Medical Image Computing and Computer-Assisted Intervention MICCAI 2005, Lecture Notes in Computer Science, vol. 3749. Springer, Berlin Heidelberg, pp. 761–769.

Herberich, G., Ivanescu, A., Gamper, I., Sechi, A., Aach, T., 2010. Analysis of length and orientation of microtubules in wide-field fluorescence microscopy. In: Goesele, M., Roth, S., Kuijper, A., Schiele, B., Schindler, K. (Eds.), Pattern Recognition, Lecture Notes in Computer Science, vol. 6376. Springer, Berlin Heidelberg, pp. 182–191.

Herberich, G., Windoffer, R., Leube, R., Aach, T., 2011. 3D segmentation of keratin intermediate filaments in confocal laser scanning microscopy. In: 2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBC, pp. 7751–7754.

Kass, M., Witkin, A., Terzopoulos, D., 1988. Snakes: active contour models. International Journal of Computer Vision 1, 321–331.

Kong, K.Y., Marcus, A., Hong, J.Y., Giannakakou, P., Wang, M., 2005. Computer assisted analysis of microtubule dynamics in living cells. In: 27th Annual International Conference of the Engineering in Medicine and Biology Society, 2005 (IEEE-EMBS 2005), pp. 3982 –3985.

Krauss, P., Metzner, C., Lange, J., Lang, N., Fabry, B., 2012. Parameter-free binarization and skeletonization of fiber networks from confocal image stacks. PLoS ONE 7, e36575.

Li, H., Shen, T., Smith, M., Fujiwara, I., Vavylonis, D., Huang, X., 2009a. Automated actin filament segmentation, tracking and tip elongation measurements based on open active contour models. In: IEEE International Symposium on Biomedical Imaging: From Nano to Macro, 2009 (ISBI '09), pp. 1302–1305.

Li, H., Shen, T., Vavylonis, D., Huang, X., 2009b. Actin filament tracking based on particle filters and stretching open active contour models. In: Yang, G.Z., Hawkes, D., Rueckert, D., Noble, A., Taylor, C. (Eds.), Medical Image Computing and Computer-Assisted Intervention MICCAI 2009, Lecture Notes in Computer Science, vol. 5762. Springer, Berlin Heidelberg, pp. 673–681.

Li, H., Shen, T., Vavylonis, D., Huang, X., 2010. Actin filament segmentation using spatiotemporal active-surface and active-contour models. In: Jiang, T., Navab, N., Pluim, J., Viergever, M. (Eds.), Medical Image Computing and Computer-Assisted Intervention, MICCAI 2010, Lecture Notes in Computer Science, vol. 6361. Springer, Berlin Heidelberg, pp. 86–94.

Lorigo, L., Faugeras, O., Grimson, W., Keriven, R., Kikinis, R., Nabavi, A., Westin, C.F., 2001. Curves: curve evolution for vessel segmentation. Medical Image Analysis 5, 195–206.

Lück, S., Sailer, M., Schmidt, V., Walther, P., 2010. Three-dimensional analysis of intermediate filament networks using sem tomography. Journal of Microscopy 239, 1–16.

Malladi, R., Sethian, J., Vemuri, B., 1995. Shape modeling with front propagation: a level set approach. IEEE Transactions on Pattern Analysis and Machine Intelligence 17, 158–175.

Mayerich, D., Keyser, J., 2009. Hardware accelerated segmentation of complex volumetric filament networks. IEEE Transactions on Visualization and Computer Graphics 15, 670–681.

Melonakos, J., Pichon, E., Angenent, S., Tannenbaum, A., 2008. Finsler active contours. IEEE Transactions on Pattern Analysis and Machine Intelligence 30, 412–423.

Mickel, W., Münster, S., Jawerth, L., Vader, D., Weitz, D., Sheppard, A., Mecke, K., Fabry, B., Schröder-Turk, G., 2008. Robust pore size analysis of filamentous networks from three-dimensional confocal microscopy. Biophysical Journal 95, 6072–6080.

Narayanaswamy, A., Dwarakapuram, S., Bjornsson, C., Cutler, B., Shain, W., Roysam, B., 2010. Robust adaptive 3-D segmentation of vessel laminae from fluorescence confocal microscope images and parallel GPU implementation. IEEE Transactions on Medical Imaging 29, 583–597.

Nolen, B., Tomasevic, N., Russell, A., Pierce, D., Jia, Z., McCormick, C., Hartman, J., Sakowicz, R., Pollard, T., 2009. Characterization of two classes of small molecule inhibitors of Arp2/3 complex. Nature 460, 1031–1034.

Nurgaliev, D., Gatanov, T., Needleman, D.J., 2010. Chapter 25 – automated identification of microtubules in cellular electron tomography. In: Cassimeris, L., Tran, P. (Eds.), Microtubules: In Vivo, Methods in Cell Biology, vol. 97. Academic Press, pp. 475–495.

Pelletier, O., Pokidysheva, E., Hirst, L.S., Bouxsein, N., Li, Y., Safinya, C.R., 2003. Structure of actin cross-linked with α-actinin: a network of bundles. Physical Review Letters 91, 148102.

Piechocka, I., Bacabac, R., Potters, M., MacKintosh, F., Koenderink, G., 2010. Structural hierarchy governs fibrin gel mechanics. Biophysical Journal 98, 2281–2289.

Pollard, T., Wu, J., 2010. Understanding cytokinesis: lessons from fission yeast. Nature Reviews Molecular Cell Biology 11, 149–155.

Pollard, T.D., Cooper, J.A., 2009. Actin, a central player in cell shape and movement. Science 326, 1208–1212.

Rigort, A., Günther, D., Hegerl, R., Baum, D., Weber, B., Prohaska, S., Medalia, O., Baumeister, W., Hege, H.C., 2012. Automated segmentation of electron tomograms for a quantitative description of actin filament networks. Journal of Structural Biology 177, 135–144.

Rochery, M., Jermyn, I., Zerubia, J., 2006. Higher order active contours. International Journal of Computer Vision 69, 27–42.

Rusu, M., Starosolski, Z., Wahle, M., Rigort, A., Wriggers, W., 2012. Automated tracing of filaments in 3d electron tomography reconstructions using sculptor and situs. Journal of Structural Biology 178, 121–128.

Sandberg, K., Brega, M., 2007. Segmentation of thin structures in electron micrographs using orientation fields. Journal of Structural Biology 157, 403–415.

Sargin, M., Altinok, A., Kiris, E., Feinstein, S.C., Wilson, L., Rose, K., Manjunath, B., 2007a. Tracing microtubules in live cell images. In: 4th IEEE International Symposium on Biomedical Imaging: From Nano to Macro, 2007 (ISBI 2007), pp. 296–299.

Sargin, M., Altinok, A., Rose, K., Manjunath, B., 2007b. Tracing curvilinear structures in live cell images. In: IEEE International Conference on Image Processing, 2007 (ICIP 2007), pp. 285–288.

Sato, Y., Nakajima, S., Shiraga, N., Atsumi, H., Yoshida, S., Koller, T., Gerig, G., Kikinis, R., 1998. Three-dimensional multi-scale line filter for segmentation and visualization of curvilinear structures in medical images. Medical Image Analysis 2, 143–168.

Shi, J., Malik, J., 2000. Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 22, 888–905.

Smith, M.B., Li, H., Shen, T., Huang, X., Yusuf, E., Vavylonis, D., 2010. Segmentation and tracking of cytoskeletal filaments using open active contours. Cytoskeleton 67, 693–705.

Stein, A.M., Vader, D.A., Jawerth, L.M., Weitz, D.A., Sander, L.M., 2008. An algorithm for extracting the network geometry of three-dimensional collagen gels. Journal of Microscopy 232, 463–475.

Storm, C., Pastore, J., MacKintosh, F., Lubensky, T., Janmey, P., 2005. Nonlinear elasticity in biological gels. Nature 435, 191–194.

Vasilevskiy, A., Siddiqi, K., 2002. Flux maximizing geometric flows. IEEE Transactions on Pattern Analysis and Machine Intelligence 24, 1565–1578.

Vavylonis, D., Wu, J.Q., Hao, S., O'Shaughnessy, B., Pollard, T.D., 2008. Assembly mechanism of the contractile ring for cytokinesis by fission yeast. Science 319, 97–100.

Vignjevic, D., Yarar, D., Welch, M.D., Peloquin, J., Svitkina, T., Borisy, G.G., 2003. Formation of filopodia-like bundles in vitro from a dendritic network. The Journal of Cell Biology 160, 951–962, <http://jcb.rupress.org/content/160/6/951.full.pdf+html>.

Wang, Y., Narayanaswamy, A., Tsai, C.L., Roysam, B., 2011. A broadly applicable 3-D neuron tracing method based on open-curve snake. Neuroinformatics 9, 193–217.

Weber, B., Greenan, G., Prohaska, S., Baum, D., Hege, H.C., Mller-Reichert, T., Hyman, A.A., Verbavatz, J.M., 2012. Automated tracing of microtubules in electron tomograms of plastic embedded samples of Caenorhabditis elegans embryos. Journal of Structural Biology 178, 129–138, Special Issue: Electron Tomography.

Winkler, C., Vinzenz, M., Small, J.V., Schmeiser, C., 2012. Actin filament tracking in electron tomograms of negatively stained lamellipodia using the localized radon transform. Journal of Structural Biology 178, 19–28.

Wu, J., Rajwa, B., Filmer, D.L., Hoffmann, C.M., Yuan, B., Chiang, C., Sturgis, J., Robinson, J.P., 2003. Automated quantification and reconstruction of collagen matrix from 3D confocal datasets. Journal of Microscopy 210, 158–165.

Xu, T., Li, H., Shen, T., Ojkic, N., Vavylonis, D., Huang, X., 2011. Extraction and analysis of actin networks based on open active contour models. In: 2011 IEEE International Symposium on Biomedical Imaging: From Nano to Macro, pp. 1334–1340.