BIOINFORMATICS

Vol. 27 ISMB 2011, pages i239–i247
doi:10.1093/bioinformatics/btr237

Automatic 3D neuron tracing using all-path pruning

Hanchuan Peng*, Fuhui Long and Gene Myers

Janelia Farm Research Campus, Howard Hughes Medical Institute, Ashburn, VA 20147, USA

ABSTRACT

Motivation: Digital reconstruction, or tracing, of 3D neuron structures is critical toward reverse engineering the wiring and functions of a brain. However, despite a number of existing studies, this task is still challenging, especially when a 3D microscopic image has low signal-to-noise ratio (SNR) and fragmented neuron segments. Published work can handle these hard situations only by introducing global prior information, such as where a neurite segment starts and terminates. However, manual incorporation of such global information can be very time consuming. Thus, a completely automatic approach for these hard situations is highly desirable.

Results: We have developed an automatic graph algorithm, called the all-path pruning (APP), to trace the 3D structure of a neuron. To avoid potential mis-tracing of some parts of a neuron, an APP first produces an initial over-reconstruction, by tracing the optimal geodesic shortest path from the seed location to every possible destination voxel/pixel location in the image. Since the initial reconstruction contains all the possible paths and thus could contain redundant structural components (SC), we simplify the entire reconstruction without compromising its connectedness by pruning the redundant structural elements, using a new maximal-covering minimal-redundant (MCMR) subgraph algorithm. We show that MCMR has a linear computational complexity and will converge. We examined the performance of our method using challenging 3D neuronal image datasets of model organisms (e.g. fruit fly).

Availability: The software is available upon request. We plan to eventually release the software as a plugin of the V3D-Neuron package at http://penglab.janelia.org/proj/v3d.

Contact: pengh@janelia.hhmi.org

1 INTRODUCTION

Digital reconstruction, or tracing, of 3D neuron structures (Fig. 1) is critical toward reverse engineering the wiring and functions of a brain (Roysam *et al*., 2009; Peng *et al*., 2011b). A number of studies (e.g. Al-Kofahi *et al*., 2002, 2003; Abdul-Karim *et al*., 2005; Cai *et al*., 2008; Dima *et al*., 2002; Evers *et al*., 2005; Losavio *et al*., 2008; Meijering *et al*., 2004; Narro *et al*., 2007; Peng *et al*., 2010a, 2010b, 2011a; Rodriguez *et al*., 2009; Schmitt *et al*., 2004; Sun *et al*., 2009; Vasilkoski *et al*., 2009; Wearne *et al*., 2005; Weaver *et al*., 2004; Xie *et al*., 2010; Xiong *et al*., 2006; Yuan *et al*., 2009; Zhang *et al*., 2007, 2008, 2011) have been conducted to develop semi- or fully automatic neuron-tracing methods that would yield more efficient neuron reconstruction than the currently widely adopted manual reconstruction strategy. Most of these existing methods have used various structural components (SC), e.g. 3D spheres, ellipsoids, cylinders, lines segments or irregular compartments, to model a neuron's morphology (Fig. 1a–e). The most successful strategy among these algorithms is to build-up the reconstruction by incrementally adding more and more such SCs into the morphological modeling of a neuron. Good examples include image voxel scooping (Rodriguez *et al*., 2009), ray shooting (Wearne *et al*., 2005) and template matching (Zhao *et al*., 2011). These bottom-up local searching methods are suitable for 3D images that have ideally continuous neurite tracts and good signal-to-noise ratio (SNR).

However, precise digitization of the 3D morphological structure of a neuron acquired through various microscopy methods, such as 3D laser scanning microscopy, remains very problematic in practice. It is especially hard when an image has low SNR, and/or broken and fuzzy neurite segments that are due to the intrinsic punctuated neurite structures (e.g. synaptic boutons) or imperfections in sample preparation (Fig. 1f and g). Notably such datasets are common for the nervous systems of different animals. For instance, the punctuated and thus often broken neurites can be ubiquitously seen in the single-neuron images of *Drosophila melanogaster* (fruit fly) (Fig. 1f), *Caenorhabditis elegans* and mouse (Fig. 1g). The local search methods discussed above cannot easily handle these hard situations, as it is very difficult to cross these gaps (i.e. low signal regions).

One strategy to tackle this challenging situation is to combine both global and local cues. Global prior information, such as the starting and ending locations of neurite structures, will guide the finer-scale optimization using local image content. Such global priors of neurite structure can be supplied easily using a novel and highly effective 3D visualization system V3D (Peng *et al*., 2010b). The Graph-augmented Deformable model (GD) algorithm, which is a graph-augmented deformable model, can be used to trace the optimal paths from the starting location to each of the ending points automatically (Peng *et al*., 2010a). Then, the entire reconstruction can be assembled automatically by detecting branching points along the merging paths (Peng *et al*., 2010b). This method had been successfully applied to reconstructing neuronal connections in the most detailed mouse brain image atlas to date (Li *et al*., 2010). Unfortunately, when a neuron's structure is very complicated or the image quality is low, it may still be very time-consuming to input such simple global prior/guiding information (Zhang *et al*., 2008). Thus, a completely automatic approach for broken structures and low SNR images is highly needed.

We previously proposed to detect the tips/termini of a neuron automatically based on the spatial anisotropy of these terminal points (manuscript under review) or adaptive template matching (e.g. the AutoMarker function in the V3D system), followed by the GD-tracing. However, tip detection may not be accurate when the termini of a neuron do not form sharp tips, such as those terminated with synaptic boutons (Fig. 1f). Adaptive template matching may also mis-detect the irregularly shaped termini. Therefore, here, we propose a new method that iteratively prunes an over-reconstruction

*To whom correspondence should be addressed.

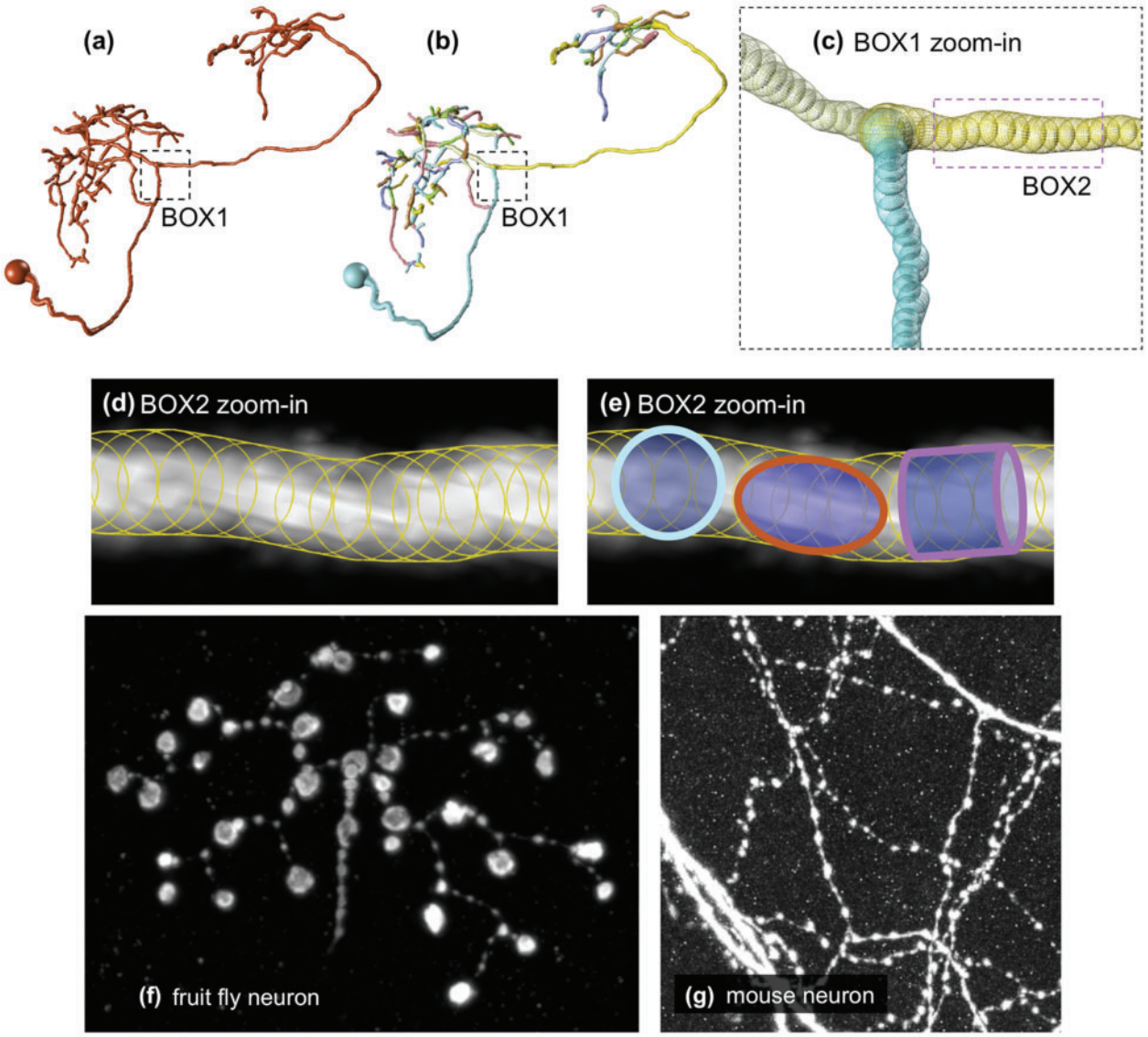

Fig. 1. Neuron tracing/reconstruction from images and SC examples of a reconstruction. (**a**) A 3D reconstruction of a fruit fly neuron. (**b**) The entire morphology model can typically be decomposed as individual segments (shown in different colors), which are connected at the branching points. Typically, each segment can be traced/reconstructed separately. (**c**) The zoom-in view of the BOX1 in (a) and (b). (**d**) Illustration of the modeling of image voxel information using a series of spherical SCs. The edge of image region (bright voxels) best matches to the aggregation of SCs. (**e**) Other types of SCs besides spheres, such as ellipsoids and cylinders, can be used in locally matching the image content and thus growing the reconstruction. (**f**) A maximum intensity project of a 3D stack of a fruit fly lamina neuron (courtesy of G. Rubin lab, Janelia Farm, HHMI). The neurites are highly punctuated, have high contrast in image intensity and appear to be broken. (**g**) Stained CA3 pyramidal neuron of a mouse brain region (courtesy of R. Tsien lab, Stanford University), where axonal varicosities make it hard to grow a reconstruction using local searching based on SCs.

of a neuron. We show its efficiency and convergence. We examine its performance using challenging 3D neuronal image datasets of different model organisms (e.g. fruit fly).

2 METHODS

2.1 APP: overview of the method

The goal of a neuron-tracing algorithm is to produce a reconstruction of a neuron's morphology as complete as possible, using a reasonably succinct model. To be precise, we define a reconstruction of a neuron as a set of topologically connected structure components that describe the 3D spatial morphology of this neuron in a 3D image. The structural component (SC, Fig. 1) is a loosely defined concept that could mean neuron branches, individual reconstruction node, individual voxels or other sub-structures contained in the neuron reconstruction. We call a reconstruction complete when all visible regions and their voxels in a neuron image are covered in this reconstruction (e.g. Fig. 1d). We call a reconstruction over-complete (i.e. an over-reconstruction) if this construction is complete, but some of its SCs are covered by other SCs and thus appear to be redundant. As mentioned in

Section 1, SCs are often spheres, ellipsoids or cylinders. SCs are represented as *reconstruction nodes*, each of which has a respective shape descriptor. In a reconstruction, graphs are often used to describe the topological relationship of all reconstruction nodes (and thus SCs). Such graphs can be coded in the SWC format (Cannon *et al.*, 1998). Typically, a neuron reconstruction is described as a tree graph, which has a root/seed reconstruction node, many leaf nodes, branching nodes and other inter-nodes. We follow the same convention here.

Our new neuron-tracing method is called all-path pruning (APP). It consists of two major steps, namely (i) producing an initial over-complete reconstruction (ICR) of a neuron and (ii) pruning its redundant SCs. We consider spherical SCs centered at every possible voxel location of a neuron. We design an efficient and reliable method to produce a complete reconstruction based on these SCs (Section 2.2). Then, we design a maximal-covering minimal-redundant (MCMR) subgraph-searching algorithm to automatically determine redundant SCs iteratively and remove them until no one can be further deleted (Section 2.3). In this way, we efficiently simplify the complete reconstruction using a minimal number of SCs, while maintaining the best covering of entire reconstructed neuron structure.

2.2 ICR of a neuron structure

Our method takes both a 3D neuron image I and a seed-location $P_s=(x_s,y_s,z_s)$ that is within the neuron region as inputs. The seed is often the soma or another big bright spot (in case soma has not been imaged) of the neuron, and can be detected automatically in many cases (it is beyond the discussion of this paper; thus, here we just assume its availability). Let us assume bright, but not dark, image voxels represent the neuron signal. To produce the initial over-complete reconstruction (ICR) of the neuron, we use the average intensity value, t_a, of the entire image as a global threshold to define the image 'foreground', that is, any voxel that has greater value than t_a is assumed to be part of the neuron, otherwise it belong to image background. Of note, in fluorescent images the amount of bright voxels is typically small; thus, t_a is often very low. Therefore, using t_a as the global threshold is very conservative and thus ensures all the possible signals are captured. After the global thresholding, we also optionally run a median filter with the $3\times3\times3$ window to remove noise.

Next, we create an undirected graph $G=(V,E)$ on the image foreground, where the set of graph vertices V stands for image voxels and the set of graph edges E encodes a geodesic metric function defined below. A pair of vertices, which correspond to image voxels at locations $v_0=(x_0,y_0,z_0)$ and $v_1=(x_1,y_1,z_1)$, have an undirected edge between them when and only when the two vertices correspond to immediate spatial neighbors, i.e. their spatial coordinates simultaneously satisfy $|x_0-x_1|\le1$, $|y_0-y_1|\le1$ and $|z_0-z_1|\le1$ (and of course at two different locations). The edge weight is defined as,

$$e(v_0,v_1)=\|v_0-v_1\|\left(\frac{g_I(v_0)+g_I(v_1)}{2}\right), \quad (1)$$

where the first term is Euclidian distance of the two vertices, and $g_I(.)$ in the second term constrains that the edge weight between bright voxels will have smaller value than that between dark voxels. $g_I(.)$ has the following form,

$$g_I(p)=\exp\left(\lambda_I\left(1-I(p)/I_{\max}\right)^2\right), \quad (2)$$

where $I(p)$ is the voxel intensity value at location p and $I_{\max}$ is the maximum intensity of the entire image I, respectively. We use the exponent of squared inverse intensity to accentuate the voxel intensity that represents signal. λ_I is a positive coefficient to control the contribution of this term. We choose $\lambda_I=10$ (other big values like 20 lead to similar results in experiments).

As our geodesic metric function outputs only the positive value, we then use Dijkstra algorithm (Dijkstra, 1959) to find the shortest path from the seed-location P_s to every other vertex in G. Since G contains only edges of adjacent voxel-vertices, it is highly sparse. Thus, the time complexity to solve the geometric shortest path is $O(N\log N)$ (N is the number of vertices in G). In practice, this step often takes only a few seconds to run for a $512\times512\times128$ image stack on a current laptop computer.

In the resultant shortest path map, we search vertices that have no child, and call such a set as the *leaf set*. Obviously, we can back trace a path from every leaf vertex to the seed location, which is also called the root vertex. All these paths share many common pieces. We organize the entire solution as a tree graph. Since this step detects all ordered paths from the root to every image foreground voxels, and thus include no false negative, we call this tree graph the all-path reconstruction, which is an ICR.

This automatic ICR method is similar to the graph step in our earlier GD (Peng *et al*., 2010a) and V3D-Neuron 1.0 (Peng *et al*., 2010b). Differently, in GD we have the prior information of the starting and ending locations of a neurite tract, whereas here we use all possible foreground voxels as ending locations of a neuron. Since the underlying graph G remaining the same, the initial reconstruction has the same computational complexity compared to V3D-Neuron 1.0.

2.3 Pruning a reconstruction: MCMR subgraph algorithm

We treat each vertex/node in the ICR as a SC. Then, we take three steps in the next three sub-sections to prune the redundant SCs, while maintaining the overall coverage of all SCs. The entire strategy is called maximal-covering minimal redundant (MCMR) subgraph search.

2.3.1 Dark-leaf pruning (DLP) We have used a very conservative global threshold, t_a, which is the mean intensity of the entire image, to define the image foreground. The reason is that to capture all possible paths in the neuron, we have to maximize any potential connectivity of any bright image regions that may be a neuron area connecting to the seed location. This threshold is often so low that a number of the 'foreground' voxels are not visible to human eyes. In an ICR, these invisible regions correspond to many redundant branches.

Therefore, we use another threshold, t_v, which defines the lowest 'visible' voxel intensity. We choose $t_v=30$ (assume we have an 8-bit image, of which the maximal intensity is 255). Then, because the reconstruction is a tree graph, we iteratively remove all leaf nodes whose respective voxel intensity is below t_v; until no more leaf node can be detected. In this way, we maximize the connectivity of different regions, while reduce the structural complexity of the reconstruction.

2.3.2 Covered-leaf pruning (CLP) To effectively prune leaf nodes, we estimate the covering radius of each remaining reconstruction node. Such a radius could be estimated using the image distance transform. However, 3D distance transform is sensitive to noise and irregular foreground border. Thus, we develop a more robust method. We define a radius-adjustable sphere centered at a reconstruction node, and then enlarge the radius gradually until 0.1% of the image voxels within this sphere are darker than the global threshold t_a (i.e. average voxel intensity of the entire image). In most cases, the choice of 0.1% makes an estimated boundary of a neurite have clear contrast to the image background. Indeed, the threshold 0.1% is the same as used as the default value in the V3D-Neuron 1.0 system (Peng *et al.*, 2010b), which has been increasingly used by the neuron reconstruction community. We treat each of the reconstruction nodes, along with its estimated radius, i.e. the covering range, as an SC.

For two reconstruction nodes a and b (or SCs, equivalently), we say a is significantly covered by b if they satisfy the following condition:

$$\Omega(a\cap b)/\Omega(a)\ge0.9, \quad (3)$$

where $\Omega(.)$ is an operator for computing the volume (or mass, see below) of the occupied region of an SC of this reconstruction node. In our case, since each SC is a sphere, Equation (3) tells how much volume of a sphere another SC occupies. As the voxel intensity can be uneven, even the overlapping volume is the same, the case that a is bright and b is dark will be quite different from the case that both a and b are bright. Indeed, intuitively the latter case

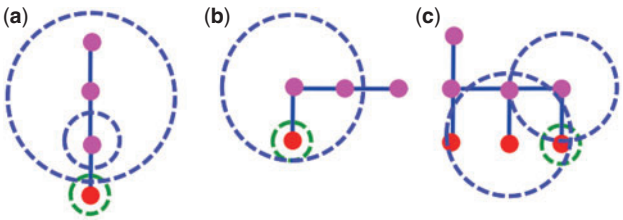

Fig. 2. Different covering situations of reconstruction nodes. Red and purple dots: leaf and non-leaf nodes, respectively. Green and blue circles: covering ranges for leaf and non-leaf nodes. (**a**) Keep; (**b**) prune; and (**c**) prune.

appears to be more heavily overlapped. Hence, we redefine $\Omega(.)$ as the mass-computing operator for the respective SC region. As a result, the more bright voxels have been covered, the more significant the covering is.

A reconstruction node may be covered jointly by multiple others. Let $S = \{b, c, \ldots\}$ as a series of reconstruction nodes, we say it covers a reconstruction node a if the following condition is satisfied:

$$\Omega(a \cap S) \big/ \Omega(a) = \Omega(a \cap (b \cup c \cup \ldots)) \big/ \Omega(a) \geq 0.9, \quad (4)$$

where $\Omega(.)$ is the mass-computing operator for the occupied region of the SC of a.

In an ICR, there are many redundant reconstruction nodes that highly overlap with each other. Obviously, a reconstruction node that is covered by others may be removed, without influencing the completeness of the reconstruction. However, which reconstruction nodes should be removed first? In general, this is a highly combinatorial subset selection problem, which is hard. However, one remarkable advantage of producing the ICR is that since we already have the ordered connectivity of all reconstruction nodes, we can design the following linear-time method to remove these redundant nodes.

We note that (i) when no node covers a leaf node, this leaf node should always be kept (Fig. 2a) and (ii) if a leaf node is covered by another node (Fig. 2b) or several other nodes jointly (Fig. 2c), this leaf node can be safely pruned. Therefore, we check all the leaf nodes in the reconstruction, and remove those being significantly covered by other remaining nodes. We iterate this pruning process until no more leaf node can be pruned.

To make this process as efficient as possible, we first scan the entire reconstruction, where all dark leaf nodes have been removed. We create a look-up table to record all the 3D spatial locations of reconstruction nodes. Then, for each node a, we create an empty covering list C_a, which would store the identities of other nodes that cover the node a. Next, we scan the entire reconstruction again and for every node b, we compute whether or not b's covering range (determined by its radius) includes any other nodes' (e.g. the node a) spatial location. If yes, then we put b in a's covering list, which is sorted (from large to small) by the radius. The sorting is automatically done as we are growing each of the nodes' covering list. It can be seen that since each node is only covered by a local neighborhood, the process to determine the covering list needs only linear time. The actual pruning as discussed in the preceding paragraph needs only a few times of scanning of currently remaining leaf nodes of the reconstruction; thus, the overall time complexity of this algorithm is linear time. Also note that this pruning algorithm guarantees to converge, as there are a limited number of nodes in the reconstruction. When we prune more and more leaf nodes, this process must stop when no leaf is covered by other reconstruction nodes.

2.3.3 Inter-node pruning (INP) and refinement After covered-leaf node pruning, the reconstruction is already succinct, in the sense that all neuron regions have been reached by a minimum number of leaf nodes. However, we could further reduce the complexity of the reconstruction, without compromising its connectedness and completeness, by removing the redundant inter-nodes that connect leaf nodes to branching nodes or the root.

We start from every leaf node a, and for its immediate parent node b, which at the same time is not a branching node or the root, we check if b is significantly covered by a, based on the criterion in Equation (5), where the operator is the same as in Equation (3).

$$\Omega(a \cap b) \big/ \Omega(a) \geq 0.1 \quad (5)$$

When the condition of Equation (5) is met, then this inter-node b is pruned and a's parent is updated as b's original parent. If b is not pruned, then we check if b's non-branching-node parent, denoted as c, would be pruned based on the coverage relation of c and b. For each leaf node, we iterate this process until a branching point or the root is reached. For each branching point, we do the same thing until its parent branching point or the root is reached. In this way, we can prune all redundant inter-nodes. Of note, in Equation (5) we have used a different threshold from Equation (3). This is because overall as few as possible reconstruction nodes should be used to maintain the coverage of the whole neuron region. Thus, the fewest number of leaf nodes should be pruned and the greatest number of inter-node should be pruned.

All the reconstruction nodes, except the root, in the simplified reconstruction have integer spatial co-ordinates, which correspond to the image voxel vertexes we initially use (Section 2.2). Thus individual segments of the reconstruction may not appear to be entirely smooth, which is more intuitive to human observation. Hence, we can use the gradient descent based deformable curve optimization (Peng *et al*., 2008, 2010a) to refine the locations of all inter-nodes. This is indeed equivalent to running the mean shift algorithm for every inter-node in a segment until convergence, with respect to a regularization constraint that the path is least bended.

The inter-node refinement step has a similar effect as treating all leaf nodes as neuron termini and then running our previous V3D-Neuron tracing method (Peng *et al*., 2010b).

2.4 Comparison to related methods

Optimal pruning of a neuron reconstruction is a topic that has not been well studied. Intuitions such as removing short branches via thresholding the absolute branch length were briefly mentioned in a few papers (e.g. Rodriguez *et al*., 2009), but we are not aware of any carefully designed algorithms on this topic. Our APP and MCMR methods fill this niche. An APP, as an integration of MCMR and the ICR, is designed to produce a compact neuron reconstruction that captures all image signals, and then to effectively overcome many problems of the existing local searching schemes in dealing with the broken or punctuated neuron structures, or low SNR images.

An APP is efficient, due to that it uses the topology of the initial reconstruction to constrain the search space. This reduces a combinatorial search to linear search, without compromising the connectedness and completeness of the entire reconstruction. This idea may be useful in general in other data analysis applications.

Image thinning is an image morphological operation that can be used to produce a skeleton of an image object. In principle, we can view image thinning as a pruning process. However, image thinning is sensitive to the contour as well as its orientation of the image object, which can be easily affected by image noise. As a result, in an image thinning result, one often finds unexpected branches. Differently, an APP is robust to the local contour shape.

Another related method is the principal curve (Hastie, 1994) and the principal skeleton (Qu and Peng, 2010). These methods are essentially regression models or deformable curves defined based on a series of control points, regularized by some kind of smoothness constraint. Besides the difference of the designing principles of these methods to APP, our experience is that APP is less sensitive to local minima and does not need a prior shape model to handle arbitrary neuron morphology.

APP can be further enhanced by considering a shape context model of branches extracted from a proof-corrected database of neuron morphology (Peng *et al.*, 2011a), so that the resultant system would be able to produce

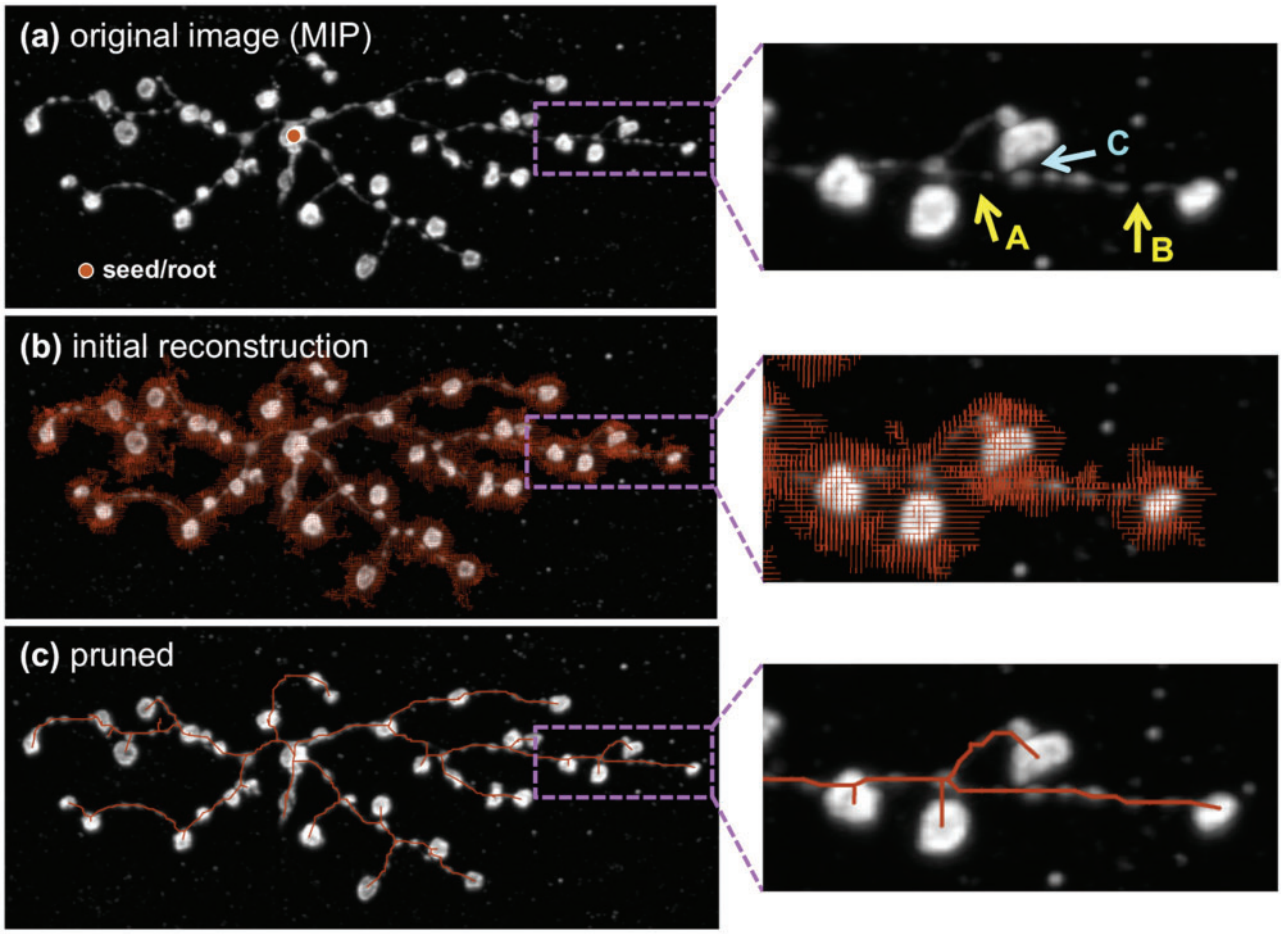

Fig. 3. APP result. (**a**) A 3D image stack of a fruit fly neuron. MIP: maximum intensity projection. Arrows A and B: two locations where the intensity of the axon is very low, thus the local search using SCs would be difficult. Arrow C: a location where the bright bouton is close to another neurite tract, so that simple local SC searching can easily produce a wrong morphology. (**b**) ICR (red) of this neuron, where the neuron region is fully covered by reconstruction nodes. The topological connection relationship and order of these nodes are also determined in this reconstruction. (**c**) Pruning of the reconstruction nodes (thus also the respective SCs and redundant branches) yield a compact by representative reconstruction (red), which is still complete (the radii of the nodes are not shown for clearer illustration).

probabilistically more probable branches, instead of connecting regions purely based on intensity. This can be useful for tracing multiple neurons that have been co-stained in the same image.

3 EXPERIMENTS

We first investigated APP's pruning performance by visually inspecting it in 3D and checking its convergence and ability to simplify complete reconstructions (Sections 3.1 and 3.2). We also evaluated APP's robustness and consistency (Section 3.3) and compared it to other competitive methods (Section 3.4). We then applied APP to tracing neurons of model animals (Section 3.5).

3.1 Visual inspection of pruning results

Figures 3 and 4 show the pruning results step by step for one of the challenging fruit fly neurons we used in the experiments of subsequent sections. This lamina neuron has clear punctuated sites (boutons), where the stained synaptic proteins are so enriched that the inter-bouton segments of the neuron appear very dark and discontinuous in the 3D image stack. Previously, we tried to reconstruct it using 3D cylinder matching method (Zhao *et al.*, 2011), but encountered problems at locations of low intensity gaps, as shown in Figure 3a arrows A and B.

Figure 3b shows the ICR has a full coverage of all meaningful neurite regions. Yet, it contains 94 741 reconstruction nodes. After the MCMR pruning, a major amount of branches were successfully removed. The final reconstruction (Fig. 3c) has only 418 reconstruction nodes. The reconstruction has a very meaningful coverage of every bouton.

In details, the dark-leaf pruning (DLP) (Fig. 4a) effectively removed all the invisible segments in the initial reconstruction (Fig. 3b), without compromising the connectedness of the entire structure. After this step only 22 903 reconstruction nodes were kept.

For each bouton, the covered-leaf pruning (CLP) (Fig. 4b) successfully removed all the branches, except the most representative one. The number of remaining nodes is 1391.

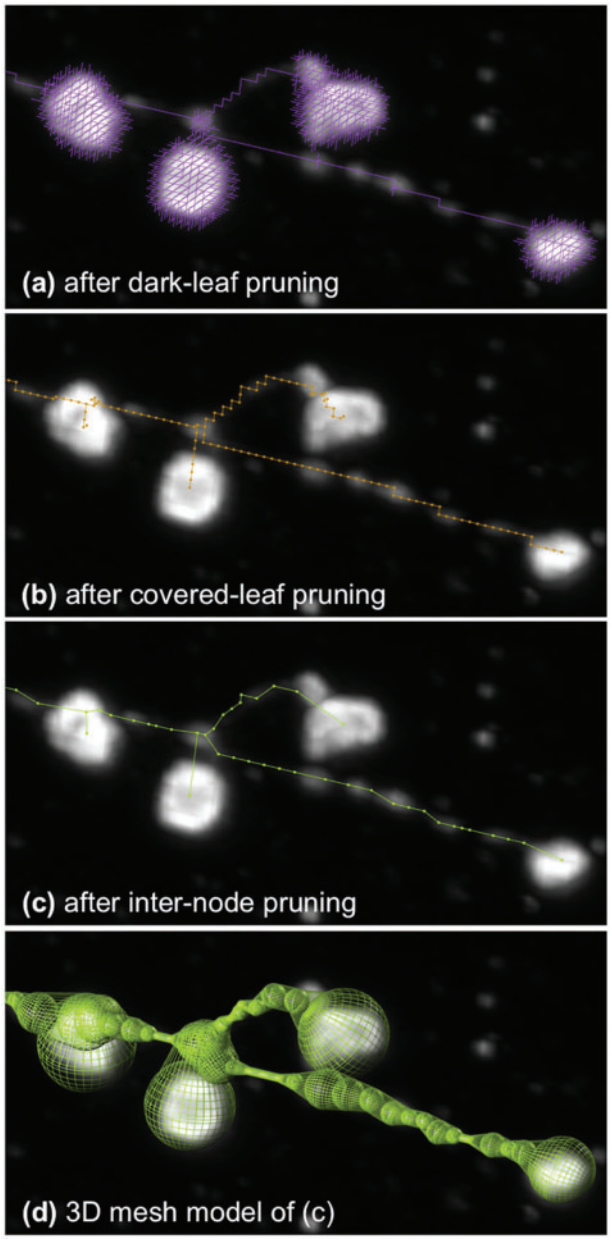

Fig. 4. Detailed MCMR pruning process of the zoom-in area of Figure 3. The 3D display is tilted from the frontal projection in Figure 3b so that both the 3D structure of the reconstruction and its spurs are visible. (**a**) The remaining complete reconstruction (magenta) after pruning all dark leaf nodes repeatedly. (**b**) The remaining reconstruction after iterative CLP. (**c**) The reconstruction after an INP. (**d**) The mesh of the complete space-filling model of the final reconstruction in (c).

The inter-node pruning (INP) knocked out another 973 redundant nodes (Fig. 4c). The final reconstruction (Fig. 4d) is very compact, but effectively covers all interesting neuron regions. Remarkably, for each bouton in this image stack, the MCMR algorithm puts one and only one reconstruction node to represent it, and there is no inter-node between a bouton-representing leaf node and its parent branching node.

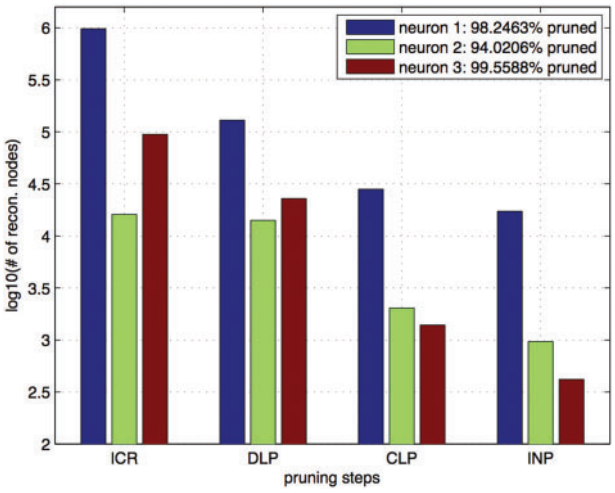

Fig. 5. Number of remaining reconstruction nodes after each step of the MCMR pruning. Three confocal images of fruit fly neurons were used. Neuron 1 (courtesy of A. Chiang lab): $1024 \times 1024 \times 56$ voxels, voxel XYZ size $0.215\,\mu\text{m} \times 0.215\,\mu\text{m} \times 1\,\mu\text{m}$. Neuron 2 (courtesy of G. Jefferis lab): $512 \times 512 \times 60$ voxels, voxel size ratio $Z/XY = 3.03$. Neuron 3 (courtesy of G. Rubin lab): $963 \times 305 \times 29$ voxels, voxel size ratio Z/XY = 1.

3.2 Convergence and compression rate

Figure 5 summarizes APP's performance for three fruit fly neuron images we used. They are typical images produced in three different labs using different protocols, which were applied to different brain areas. The image sizes also vary significantly. Especially, the levels of image brightness, contrast and noise are all different for these three images.

In spite of the difference among images, in all cases, APP converges quickly. When the image is bright (neuron 2), DLP may not remove as many nodes as in the other images. The covered-leaf node pruning and INP steps always have an order of magnitude of reduction of the total reconstruction node number.

Evidently, the MCMR pruning greatly reduces the complexity of the reconstructions. Figure 5 shows that 94.0–99.6% of reconstruction nodes can be removed without compromising the completeness of the reconstructions.

3.3 Robustness and consistency

We tested the robustness and consistency of APP using two experiments. We chose the image used in Figure 4 as it is a hard case that we did not find another automatic method that would be able to produce a meaningful reconstruction.

First, we traced the same neuron image multiple times, each from a different seed location, and compared the similarity of the resultant reconstruction. We computed the 'spatial distance' (SD) between any pair of reconstructions, say R_1 and R_2, by averaging the reciprocal minimal spatial distances of their reconstruction nodes (Peng *et al.*, 2010b). A larger distance means the greater discrepancy between R_1 and R_2. When some nodes have spatial distance >2 voxels, this discrepancy is visible. Thus, such a spatial distance is called substantial SD (SSD). The percentage of SSD nodes in a pair of reconstruction, SSD%, is also a good measure of the visible difference of two reconstructions.

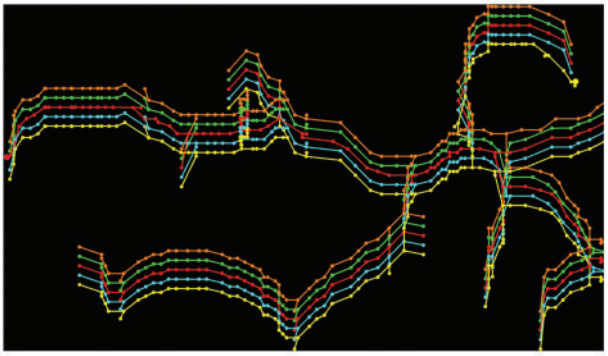

Fig. 6. Consistent reconstructions produced for the same neuron from different seed locations. The final skeletons of reconstructions are intentionally displaced for better visualization. Different colors indicate different reconstructions. Dots: reconstruction nodes. Root/seed nodes are bigger dots. The major difference of these reconstructions happens around the locations of their seed locations.

In our experiment, we traced it 20 times from 20 randomly selected, and spatially distant seeds. The reconstructions are very similar (Fig. 6). The number of reconstruction nodes range from 390 to 403, with mean ± standard deviation = 398 ± 3.34 voxels. The average SD score between R_1 and the 19 other reconstructions is 0.215 voxels over the entire structure. The average SSD% score is only 2.79%. For these visible distinct regions, the average SSD score is only 3.0 voxels. Therefore, the overall structures traced from APP are consistent.

In the second experiment, we added different levels of 'deletion' noise into the same image and thus produced an even more challenging image for reconstruction (remember the original image is already very challenging). We reconstructed multiple times from the same seed location and tested the robustness of APP.

Figure 7 shows that with as much as 75% of visible voxels have been randomly removed from this image (Fig. 7b), APP is still able to produce an overall meaningful reconstruction (Fig. 7c), which has a similar skeleton with the reconstruction obtained from the noise-free image. Most distinct areas of these reconstructions happen at the bouton regions, where most bright voxels are deleted and thus this area has a number of fragments. Interestingly, most of the skeleton regions remain very similar. This demonstrates the robustness of APP.

Table 1 shows the distance scores between the original reconstruction and each of these noisy reconstructions. When 25% to 75% bright voxels have been removed, there are 32–40% reconstruction nodes have visible spatial differences. However, these nodes enrich at the bouton region, but not the major neuron branching points (Fig. 7). The SSD% score jumps up when 90% of image voxels are darkened; this is reasonable—there is nothing in this noisy image to trace.

3.4 Comparison to manual, semi-automatic and automatic neuron-tracing methods

We have compared APP to a number of tracing tools, including: (i) manual tracing (V3D-Neuron 1.0's Pinpoint-N-Link method); (ii) semi-automatic tracing (V3D-Neuron 1.0's 1-point-to-N-point

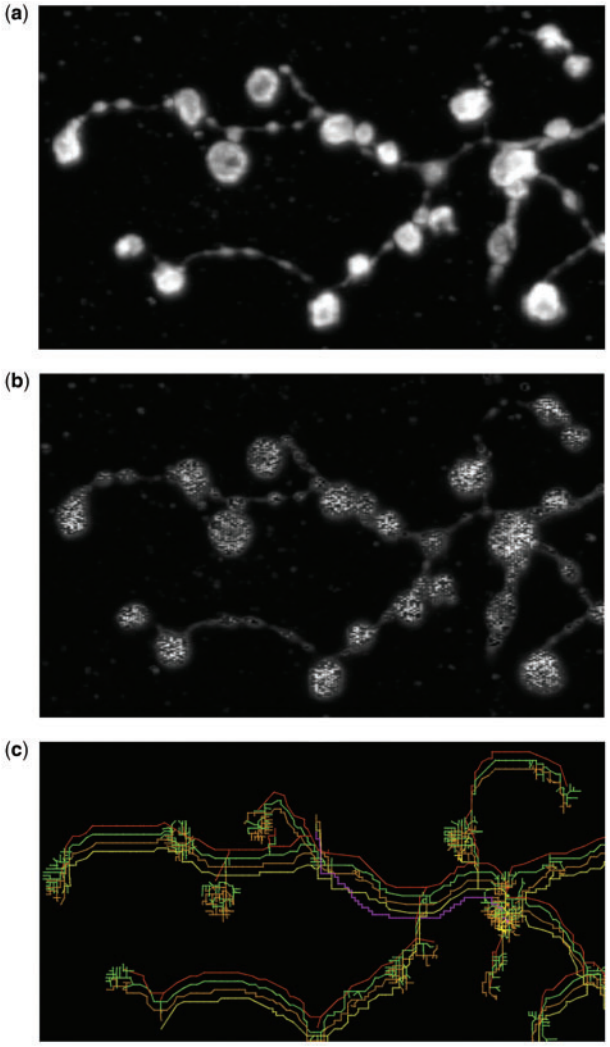

Fig. 7. Reconstructions produced for the same original neuron image, but contaminated by different levels of noise. (**a**) A part of the original image, where there are some dark regions that are hard for tracing. (**b**) A noise image where bright voxels are randomly deleted. As a result, the bouton region becomes fragmented. The noise introduced is a random deletion of q% of bright voxels (intensity >50). In this sub-figure, $q = 75$. (**c**) The final skeletons of reconstructions are intentionally displaced for better visualization. Different colors indicate different reconstructions. Red: the reconstruction from noise-free image. Green, orange, yellow and magenta: q is 25, 50, 75 and 90, respectively. When $q = 90$, most signals of the image have been removed. See summary result in Table 1 as well.

Table 1. Distance scores of the reconstructions of a noise-free image and several images, where different levels of noises were added

Score	25%	50%	75%	90%
SD	1.912	2.041	4.320	63.53
SSD	3.781	4.024	9.458	66.24
SSD%	31.9	35.2	40.1	81.7

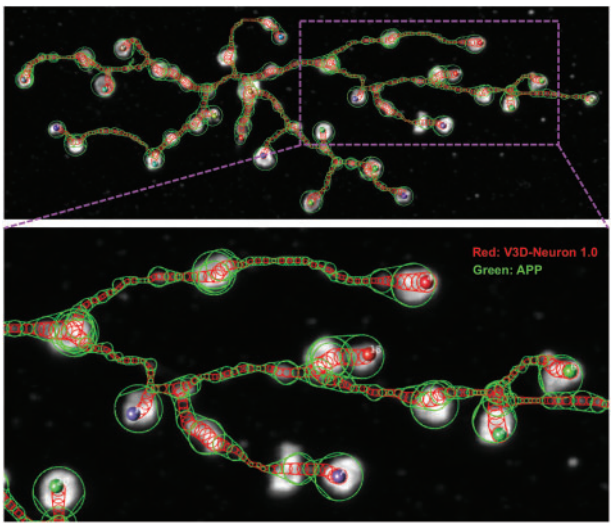

Fig. 8. Two reconstructions produced by APP (green) and V3D-Neuron 1.0's 1-point-to-N-point automatic tracing function (red). The contours of the spherical structural components of reconstruction nodes are overlaid on top of the original image (gray scale).

tracing, and V3D-Neuron 2.0's Paint-N-Trace method); and (iii) automatic tracing (Wearne *et al*., 2005; Zhao *et al*., 2011).

Compared to APP, both manual and semi-automatic reconstruction methods are slow. Typically, they take a few minutes to trace simple structures and 20–30 min for a relatively complicated structure. The respective results are still not error-free in many cases. On the other hand, APP is fast, but is limited by the resolution of an image. This drawback is shared by other automatic methods we have tested. However, APP has the advantage to be able to trace in discontinued areas, which is a major problem for the methods we have tested. Indeed, for all images we have shown in the previous sections, these automatic methods failed to produce meaningful results.

With the prior ending points available, V3D-Neuron 1.0's 1-point-to-N-point tracing is a powerful semi-automatic method to reconstruct neuron morphology. For the datasets used in the previous sections, we compared the fully automatic APP with V3D-Neuron 1.0, using the same seed location. Figure 8 shows that both methods produce very similar skeletons (SD = 0.84 voxels, SSD = 3.55 voxels, SSD% = 7.6%). This means APP is as good as the semi-automatic method, without any manual guidance.

A closer examination shows that APP can estimate the diameter of reconstruction nodes more precisely than V3D-Neuron 1.0. This is because V3D-Neuron 1.0 only back-trace the paths from predefined ending points, where due to various local image noise the diameter estimation may be imperfect. Then, because V3D-Neuron 1.0 also smoothes the diameters along the paths, this results in an overall under-estimation of the diameters of reconstruction nodes. On the contrary, APP considers all possible paths and thus always uses the reconstruction nodes that have the large diameter (equivalently, the radius mentioned in Section 2) to cover other nodes with smaller diameters. Thus, the diameter estimation is more accurate. Of note, generally APP's result has a little bit over-estimation of the visually optimal diameter of each reconstruction node. This is more obvious when the image object (e.g. a bouton) is of irregular shape.

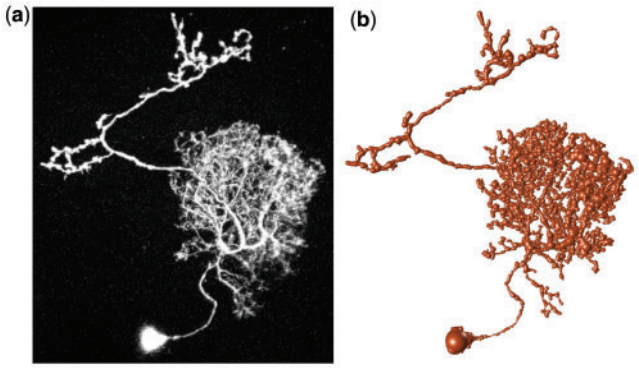

Fig. 9. An example of traced neurons that have complicated morphology. (**a**) The original image (courtesy of A. Chiang lab). (**b**) The reconstruction produced by APP.

3.5 Application in model animals

We have applied APP to tracing a number of neurons of model animals, including fruit fly and mouse. Due to the limitation of space, we show only one example in Figure 9, which is a neuron with very complicated arborization. It is fair to say this neuron is so difficult that the dense arbor may not be trace-able even by hand.

We should note that even our method is able to produce the reasonably meaningful reconstruction (Fig. 9) quickly, the result is still not perfect. One cause is that the voxel resolution of this image, especially along the *Z*-direction, is not high enough. The anisotropic property of image voxels makes it hard to estimate the radii of neuron reconstruction nodes, and thus reduces the reconstruction accuracy of our method. It may also be interesting to consider non-spherical structure components to enhance our pruning algorithm.

4 CONCLUSION

We have developed an automatic APP method to trace the 3D structure of a neuron. This method is fast. We show that this method is robust to image noise and locations of seeds. It is able to produce results comparable to the semi-automatic tracing methods and also produce automatic reconstructions for neurons of complicated morphology.

ACKNOWLEDGEMENTS

We thank Tzumin Lee lab, Gerry Rubin lab and Richard Tsien lab, Ann-Shyn Chiang lab and Greg Jefferis lab for the test data.

Funding: Howard Hughes Medical Institute.

Conflict of Interest: none declared.

REFERENCES

Abdul-Karim,M. *et al*. (2005). Automatic selection of parameters for vessel/neurite segmentation algorithms. *IEEE T. Image Process.*, **14**, 1338–1350.

Al-Kofahi,K. *et al.* (2002) 'Rapid automated three-dimensional tracing of neurons from confocal image stacks.' *IEEE Trans. Inf. Technol. Biomed.*, **6**, 171–187.

Al-Kofahi,K. *et al.* (2003) 'Median-based robust algorithms for tracing neurons from noisy confocal microscope images.' *IEEE T. Inf. Technol. B.*, **7**, 302–317.

Cai,H. *et al*. (2008) Using nonlinear diffusion and mean shift to detect and connect cross-sections of axons in 3D optical microscopy images. *Med. Image Anal.*, **12** 666–675.

Cannon,R.C. *et al.* (1998) An on-line archive of reconstructed hippocampal neurons. *J. Neurosci. Methods*, **84**, 49–54.

Dijkstra,E.W. (1959) 'A note on two problems in connexion with graphs.' *Numer. Math.*, **1**, 269–271.

Dima,A. *et al*. (2002) Automatic segmentation and skeletonization of neurons from confocal microscopy images based on the 3-D wavelet transform. *IEEE T. Image Process.*, **11**, 790–801.

Evers,J. *et al*. (2005) Progress in functional neuroanatomy: precise automatic geometric reconstruction of neuronal morphology from confocal image stacks. *J. Neurophysiol.*, **93** 2331–2342.

Hastie T. (1994) Principal curves and surfaces. Ph.D. Thesis. Stanford University, 1994.

Li,A. *et al*. (2010) Micro-optical sectioning tomography to obtain a high-resolution atlas of the mouse brain. *Science*, **330**, 1404–1408.

Losavio,B. *et al*. (2008) Live neuron morphology automatically reconstructed from multiphoton and confocal imaging data. *J. Neurophysiol.*, **100**, 2422–2429.

Meijering,E. *et al.* (2004) 'Design and validation of a tool for neurite tracing and analysis in fluorescence microscopy images.' *Cytometry*, **58A**, 167–176.

Narro,M. *et al*. (2007). NeuronMetrics: software for semi-automated processing of cultured neuron images. *Brain Res.*, **1138** 57–75.

Peng,H. *et al*. (2008) 'Straightening Caenorhabditis elegans images.' *Bioinformatics*, **24** 234–242.

Peng,H. *et al.* (2010a) Automatic reconstruction of 3D neuron structures using a graph-augmented deformable model. *Bioinformatics*, **26**, i38–i46.

Peng,H. *et al.* (2010b) 'V3D enables real-time 3D visualization and quantitative analysis of large-scale biological image datasets.' *Nat. Biotechnol.*, **28**, 348–353.

Peng,H. *et al.* (2011a) Proof-editing is the bottleneck of 3D neuron reconstruction: the problem and solutions. *Neuroinformatics*, DOI: 10.1007/s12021-010-9090-x.

Peng,H. *et al*. (2011b) BrainAligner: 3D registration atlases of Drosophila brains. *Nature Methods* [Epub ahead of print; DOI: nmeth.1602].

Qu,L. and Peng,H. (2010) A principal skeleton algorithm for standardizing confocal images of fruit fly nervous systems. *Bioinformatics*, **26,** 1091–1097.

Rodriguez,A. *et al*. (2009) Three-dimensional neuron tracing by voxel scooping. *J. Neurosci. Methods*, **184,** 169–175.

Roysam,B. *et al.* (2009) 'The central role of neuroinformatics in the national academy of engineering's grandest challenge: reverse engineer the brain.' *Neuroinformatics*, **7**, 1–5.

Schmitt,S. *et al*. (2004) New methods for the computer-assisted 3-D reconstruction of neurons from confocal image stacks. *NeuroImage*, **23,** 1283–1298.

Sun,C. *et al*. (2009) Fast linear feature detection using multiple directional non-maximum suppression. *J. Microsc.*, **234,** 147–157.

Vasilkoski,Z. *et al*. (2009) Detection of the optimal neuron traces in confocal microscopy images. *J. Neurosci. Methods*, **178,** 197–204.

Wearne,S.L. *et al.* (2005) 'New techniques for imaging, digitization and analysis of three-dimensional neural morphology on multiple scales.' *Neuroscience*, **136**, 661–680.

Weaver,C. *et al*. (2004) Automated algorithms for multiscale morphometry of neuronal dendrites. *Neural Comput.*, **16,** 1353–1383.

Xie,J. *et al.* (2010) Automatic neuron tracing in volumetric microscopy images with anisotropic path searching. *Med. Image Comput. Comput. Assist. Interv.*, **6362** (Pt II), 472–479.

Xiong,G. *et al*. (2006) Automated neurite labeling and analysis in fluorescence microscopy images. *Cytometry Part A*, **69**, 494–505.

Yuan,X. *et al*. (2009) MDL constrained 3-D grayscale skeletonization algorithm for automated extraction of dendrites and spines from fluorescence confocal images. *Neuroinformatics*, **7**, 213–232.

Zana,F. and Klein,J.-C. (2001) Segmentation of vessel-like patterns using mathematical morphology and curvature evaluation. *IEEE T. Med. Imaging*, **11**, 1111–1119.

Zhang,Y. *et al*. (2008) 3D axon structure extraction and analysis in confocal fluorescence microscopy images. *Neural Comput.*, **20**, 1899–1927.

Zhang,Y. *et al.* (2009) Detection of retinal blood vessels based on nonlinear projections source. *J. Signal Process. Syst.*, **55**, 103–112.

Zhang,Y. *et al.* (2007) 'Automated neurite extraction using dynamic programming for high-throughput screening of neuron-based assays.' *NeuroImage*, **35**, 1502–1515.

Zhao,T. *et al.* (2011) Automated reconstruction of neuronal morphology based on local geometrical and global structural models. *Neuroinformatics* [Epub a head of print; DOI 10.1007/s12021-011-9120-3].