STRAIGHTENING WORM IMAGES

Hanchuan Peng, Fuhui Long, and Eugene W. Myers
Janelia Farm Research Campus, Howard Hughes Medical Institute, Ashburn, VA.

ABSTRACT

C. elegans, a roundworm in soil, is widely used in studying animal development and aging, cell differentiation, etc. Recently, high-resolution fluorescence images of *C. elegans* have become available, introducing several new image analysis applications. One problem is that worm bodies usually curve greatly in images, thus it is highly desired to straighten worms so that they can be compared easily under the same canonical coordinate system. We develop a Worm Straightening Algorithm (WSA) using a cutting-plane restacking method, which aggregates the linear rotation transforms of a continuous sequence of cutting lines/planes orthogonal to the "backbone" of a worm to best approximate the nonlinearly bended worm body. We formulate the backbone as a parametric form of cubic spline of a series of control points. We develop two minimum-spanning-tree based methods to automatically determine the locations of control points. Our experimental methods show that our approach can effectively straighten both 2D and 3D worm images.

1. INTRODUCTION

Caenorhabditis elegans (*C. elegans*) is a nematode (roundworm), which lives in soil. As shown in the huge resource at [1], this animal has been used to study many biological problems such as animal development and aging. In recent years, high-resolution multiplex fluorescence 2D/3D images for the entire worm have been generated more and more routinely. Because worms usually have different curved shapes and other morphological features, an automatic worm straightening technique is highly desired for both biology and engineering reasons. Biologically it is easier to compare features of various worms if these worms are straight from anterior to the posterior. Practically, in term of the bounding box, a straightened worm (Fig. 1 (c)) usually occupies a far less image area/volume than the original curved worm in the raw image (Fig. 1 (a)). Thus, worm straightening will usually shrink down the image file size dramatically, saving a huge amount of storage space and I/O time in file accessing.

Several characteristics of worm images complicate this problem. Two major challenges are the following.

- First, because specific features of a worm will be measured in these images, the straightened worm should have the least possible deformation of cell morphology, intensity, and relative cell locations, compared to the original worm in the raw image.
- Second, typically the fluorescence staining will light up only a selected set of cells instead of the entire worm. The skin/cuticle of the worm is usually unstained. This makes it difficult to estimate the bending parameters directly.

The application of worm straightening *per se* is new. We have developed in the following a Worm Straightening Algorithm (WSA) to straighten both 2D and 3D worm images. WSA is currently one of the major image analysis tools in our project on digital worm cell atlas.

2. OVERALL SCHEME OF WSA

The ultimate goal of WSA is that it should minimize the loss of information in the geometrical transform in straightening. On top of this, it would be very useful to have an efficient algorithm to straighten a worm even its boundary is not well defined.

In our approach, the crucial observation about the worm images is that a worm can be treated as a manifold, which is globally nonlinear (i.e. curved), but locally smooth. This is seen in Fig. 1 (a). Thus we achieve the above goals by stitching all local rigid rotation transforms to approximate the globally nonlinear geometrical warping transform that curves the worm. As at any local point the transform is simply a rigid rotation without even scaling, the resolution of the input image can be maximally preserved.

The remaining question is how to pile up all these local linear transformations together. As shown in Fig. 1(b), we detect the backbone of a worm (§3), generate a series of 1-pixel-spaced cutting planes/lines orthogonal to the backbone, and restacking all the re-sampled/interpolated data on these cutting-planes (Fig.1(c)). In this way, the error of the global nonlinear transform is the sum of errors of all local linear transforms. As every local transform is simply a rigid rotation, and the error is merely a function of the interpolation error when a straight line is rotated in space, the overall warping error is minimized naturally.

Within the above framework, the key is how to detect the meaningful backbone of a worm. We elaborate this in §3.

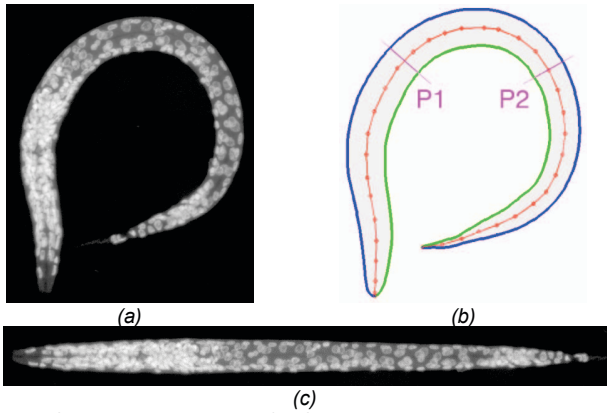

Fig. 1. Schematic illustration of WSA. (a) A curved worm image. (b) The detected backbone (red), the respective control points (red dots) and orthogonal cutting lines/planes (purple lines P1 and P2). Also shown is the worm boundary (two edges in blue and green), which is used in the BDB+ algorithm (§3) but is unnecessary for the BDB⁻ algorithm (§3). (c) The straightened worm image.

3. BACKBONE DETECTION

We present two algorithms, BDB^{-} and BDB^{+}, in the following to detect the backbone for a worm image without and with boundaries, respectively. It is natural to define the smooth backbone as a cubic-

spline over a train of control points (i.e. red dots in Fig. 1(b)). Hence, the backbone detection problem is equivalent to determining the locations of control points. This parametric form also allows marching along the backbone in any spacing (e.g. 1-pixel a step) to produce the entire sequence of orthogonal cutting lines/planes for restacking.

3.1 BDB⁻: Backbone Detection without Worm Boundary

For worm images where only cells are lit up using fluorescence labels, it is often hard to define clear worm boundary, except the several cases discussed in §3.2. The backbone detection problem is thus how to find a smooth mid-axis through the scattered worm cells. This problem is similar to detecting the principal curve of a set of scattered points. However, the existing principal curve detection methods (e.g. [2][4]) cannot be employed directly, because worm cells with various morphologies, intensities, and sizes can hardly be simplified using points such as cell centers.

In our algorithm for *backbone detection without worm boundary* (BDB⁻), intuitively we define backbone control points as those span a smooth curve (i.e. the backbone) and have the least sum of distances to the nearest stained cells. In the following we address three issues: (1) how to give an ordering to these control points, (2) how to compute the distance between control points to worm body without cell segmentation, and (3) how to evolve the locations of control points.

3.1.1 Initialization of Control Points

In BDB⁻, we initialize the set of control points, denoted as Ω, by randomly selecting a small group of pixel locations (e.g. 30 pixels) within the worm body region. This can be done by randomly selecting pixels with high intensities.

In an earlier work [3], we have considered the minimum weight spanning tree (MST) in image clustering, a data distribution exploration problem. It can be noted that the longest path of an MST, i.e. the diameter, gives a rough approximation of the skeleton of data distribution. Hence in BDB⁻ we use the diameter of the MST to give an ordering to all control points from one end (e.g. anterior or posterior) of the worm body to the other end (e.g. posterior or anterior). Each control point is a vertex of in the graph, and each edge linking two vertexes has a weight that is the distance between the two vertexes. Starting from a full graph of all control points, finding the MST can be done in $O(|\Omega|^2)$ time and the diameter can be detected using two passes of the breadth-first-search [7]. Note some control points will be pruned if they fall in short MST branches. This feature makes our algorithm insensitive to the number of initial control points.

3.1.2 Distance between Control Points and the Worm Body

We consider an aggregation approach to compute the distance between control points and the worm body. In fluorescence images a cell is indicated by the group of photons recorded at the respective pixel locations of this cell. The whole worm is thus indicated by the photon distribution of the entire worm body region, including all visible worm pixels in an image. Thus we define the distance between any control point $Q \in \Omega$ and the worm body as the sum of distances between Q and all photons that are closest to Q than any other control points. Because the number of photons at a pixel location J is apparently proportional to the pixel intensity, denoted as I_J, we define the distance between control point Q and a pixel J as the sum of Euclidean distances between Q and all photons at pixel J:

$$D(Q,J) = I_J \cdot \|Q - J\| \tag{1}$$

Denote the entire set of worm body pixel as W, thus the distance of the control point set Ω and W is defined as the sum of distances of all individual $Q_k \in \Omega$ (k = 1, …, $|\Omega|$) and the respective subsets $\Theta(W, Q_k) \in W$, called the influence-zone of Q_k, which is the set of pixels in W that have Q_k as their closest control point:

$$D(\Omega,W) = \sum_{k=1}^{|\Omega|} D(Q_k,W) = \sum_{k=1}^{|\Omega|} \sum_{J \in \Theta(W,Q_k)} D(Q_k,J) \tag{2}$$

$$L(J) = \arg\min_{k=1,\ldots,|\Omega|} D(Q_k,J) \tag{3}$$

$$\Theta(W,Q_k) = \{L(J) = k \mid J \in W\} \tag{4}$$

3.1.3 Adjusting Control Points

The objective function in Eq. (2) can be viewed as an external energy term. We can further constrain the algorithm to find a backbone that is as smooth and short as possible. These additional requirements can be formulated as internal energy terms defined by the distances between adjacent data points on the backbone. Due to space limitation, we omit the details of derivation; the formula to adjust the locations of control points is:

$$Q_k^{new} \leftarrow \frac{\alpha\left(\frac{1}{N_k}\sum_{J \in \Theta(W,Q_k)} (I_J \cdot J)\right) + \beta(Q_{k-1} + Q_{k+1}) + \gamma\left(\frac{3}{2}Q_{k-1} + \frac{3}{2}Q_{k+1} - \frac{1}{2}Q_{k+2} - \frac{1}{2}Q_{k-2}\right)}{\alpha + 2\beta + 2\gamma} \tag{5}$$

where $N_k = \sum I_J$ ($J \in \Theta(W,Q_k)$), and α, β, γ (typically α=1, β=0.5, γ=0.5) are three positive coefficients controlling the weights of the external energy and internal energy terms.

3.1.4 BDB⁻ Algorithm

The BDB⁻ algorithm works as follows:

(a) Initialize control points using the MST method.
(b) Find the influence-zones of control points using Eqs. (3) and (4).
(c) Update the coordinates of control points using Eq. (5).
(d) Check if control points have significant change or displacement. If yes, go to step (b). Otherwise go to step (e).
(e) Use cubic spline to generate a smooth backbone curve of the control points.

The major advantage of BDB⁻ is it works for grayscale images without clear worm boundary. It also has a loose constraint of the initialization of control points. BDB⁻ can be applied to both 2D and 3D images.

3.2 BDB⁺: Backbone Detection with Worm Boundary

In a few cases it is possible to define a meaningful worm boundary in images. For example, due to auto-fluorescence a worm body region may have higher intensity than the image background. The worm body may be more visible when all Z-sections in a 3D worm image stack are overlaid together. As *C. elegans* can only bend in the dorsal-ventral direction and is limited to bend in other directions, summing up all Z-sections simplifies 3D straightening as a 2D problem. Worm boundary can also be extracted in many non-fluorescence images of bright/dark fields, phase contrast, etc. They are often 2D. Thus we develop the following BDB⁺ algorithm for *backbone detection with clear worm boundary*.

We assume the 2D worm boundary is already extracted. The problem is how to find the best backbone given the worm boundary. We define backbone control points as those have equal distances to the worm boundary and span a smooth curve. This is similar to the "morphological skeleton" of an elongated shape. This skeleton can be detected using a Blum's medial axis transform [6]. Image morphological operations such as thinning can generate similar results (for a review see [5]). However, the skeleton found in this way is often broken and needs post-processing such as skeleton linking.

We follow the idea in §3.1, using the diameter of an MST to initialize all the control points of a backbone curve.

3.2.1 Initialization of Control Points

Let's consider a graph defined on image pixels. In a simple formulation, we produce an undirected graph $G_{object} = (V, E)$, where the vertex set V = {all pixels in the worm region} and E denotes the set of edges between vertexes. A pair of pixel vertexes have an edge between them iff the shortest path (i.e. straight line segment) between them does not cross any non-worm-body pixel. It can be seen that the diameter of the MST of G_{object} goes through the worm body from anterior to posterior. A smoothed diameter of this MST is a good approximation of the worm backbone. As G_{object} is not very sparse, the complexity to find the MST is $O(N^2)$, where $N=|V|$. Note this complexity is very high due to the usually big N, the number of pixels in the worm body region.

One way to reduce the complexity is to generate a sparse graph, called G_{mesh}, by only retaining edges of immediately adjacent pixel vertexes (i.e. they are within a 2×2 neighborhood) and dropping all other edges. This mesh graph covers the entire worm body. The complexity to finding the MST from G_{mesh} can be reduced to $O(|E|+N\ln N)$ [7]. Note this case is similar to image thinning.

We consider a different way to accelerate the algorithm by using only a small portion of pixel vertexes randomly sampled from V. Assume it has rN pixels (e.g. r = 1%). The edge weight between them is defined as their Euclidean distance. Because the worm body forms a manifold in the space, it is possible that two pixel vertexes that are geodesically far away on the worm body could have small Euclidean distance. We disallow this situation by defining the distance of two pixel vertexes to be infinity if the straight line-segment between them intersects any non-worm-body pixels. We call the new graph G_{random}, which is a random sub-graph of G_{object}. G_{random} is not necessary to be sparse, however, the complexity to find its MST is $O(r^2N^2)$, about 10,000 times smaller than the case of G_{object} if r = 1%.

3.2.2 Adjusting Control Points

We repeatedly adjust the coordinates of control points. As we already detect the MST diameter, for each of its end points we search the closest boundary point. Thus we can find two boundary points that indicate two boundary segments B_L and B_R, to the left and right sides of the diameter, respectively. Let Q denote a control point. We seek for the nearest points in these two sets to Q:

$$i_L^* = \arg\min_{i=1,\ldots,|B_L|} D(Q, b_L^i) \quad (6)$$

$$i_R^* = \arg\min_{i=1,\ldots,|B_R|} D(Q, b_R^i) \quad (7)$$

where i_L^* and i_R^* are the indexes of the respective best matching boundary pixels. We next adjust the x and y coordinates of the control point as the mean values of both matching pixels:

$$Q^* \leftarrow \left\{ \tfrac{1}{2}[x(i_L^*) + x(i_R^*)],\ \tfrac{1}{2}[y(i_L^*) + y(i_R^*)] \right\} \quad (8)$$

We iterate Eqs. (6), (7) and (8) until the location of Q converges. Typically this occurs very rapidly in several loops.

Due to the random initialization, the above method may output unevenly spaced control points. We thus sort the control points using their orders on this line graph. Next, we delete a point that is too close to its intermediate anterior point, i.e. the distance is smaller than a preset threshold. Finally, for each pair of adjacent control points, we add one more control point in between them, with the initial coordinate being the average of these two points. The location is then best adjusted using the control point refinement algorithm above. In this way, we can generate a pretty much evenly spaced control points along the worm backbone.

Every time after the backbone control points are adjusted, we can re-find the best left and right boundary segments B_L and B_R. As a result, BDB$^+$ is able to detect the exact anterior and posterior ends of a worm body. In contrast, the morphological skeleton and thinning methods usually have imperfect results at these locations.

4. EXPERIMENTS

We evaluate WSA using both 2D and 3D images. In the following, we first compare results of several methods, then show the significance in straightening 3D images using WSA. Due to space limitation, we only show partial results.

4.1 Comparison of Several Methods

One key part of worm straightening is the detection of meaningful backbone of a worm. Thus we show the comparison here.

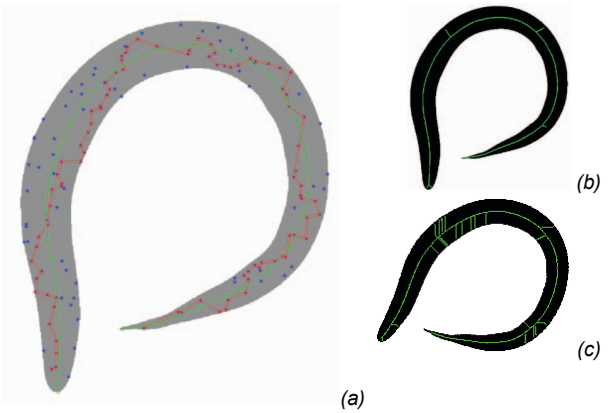

Fig. 2. Backbone detection results using different methods. (a) Our WSA result, where the backbone (green curve and dots) evolves from the MST diameter (red line) produced for a random sub-graph, whose vertexes (150 blue and red dots) are randomly sampled from the entire set of pixel vertexes on the worm body (>80,000 pixels in this image). (b) Morphological image skeleton (c) Morphological image thinning for the image rotated 30°.

Fig. 2 (a) shows the result of our WSA BDB$^+$ method. We initialize the control points randomly and identify the longest path, i.e., diameter, of the MST that spans these points. Then the points on the diameter are adjusted rapidly to converge to a stable backbone. BDB$^+$ is stable for various conditions such as randomly selected initial control points and orientation of the worm body. Its computational complexity is proportional to the number of control

points used and the length (pixel number) of the worm body boundary.

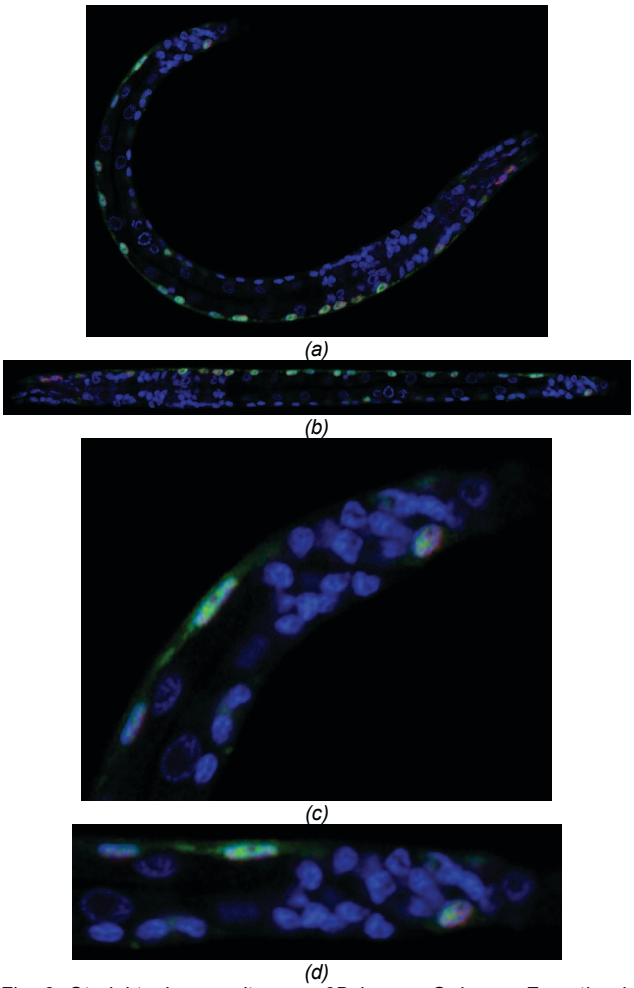

Fig. 3. Straightening results on a 3D image. Only one Z-section is shown. (a) The original worm (25% of the real size) (b) The straighten worm (20% size) (c) Tail region of the original worm (80% size) (d) Tail region of the straightened worm (80% size).

Fig. 2 also shows the comparison results of morphological image skeleton and thinning algorithms. Morphological operations will typically lead to skeletons that are branched (Fig. 2 (b) and (c)) or broken (not shown, as it is very not visually noticeable when displayed in small pictures like here). In addition, these morphological procedures are often sensitive to the orientation of a worm body, as exemplified in Fig. 2 (c), and cannot generate consistent results for various conditions of the same worm body. For computational complexity, these methods are often linearly to the number of image pixels. Post-processing such as branches removal, major-segment linking, etc, are usually needed to repair these results as meaningful backbones. Note that one possibility is to use the MST in BDB^+ to refine the results of image morphological operations.

Because BDB^- is the only method we are aware of that is able to detect the worm backbone without clear worm boundary, we compare it against BDB^+. Let's consider an image with clear boundary. If BDB^- works, then without considering the worm boundary it should generate a backbone reasonably close to the one produced by BDB^+, which considers the boundary explicitly. Here we omit the respective results due to the limited space.

4.2 Application of 3D Worm Image Straightening

Table 1. Comparison of 3D image-stack file sizes (bytes) before and after straightening (without compression).

Image #	*before*	*after*	*size reduction*	*average size reduction*
1	*358M*	*74M*	*79.33%*	*79.76%*
2	*352M*	*68M*	*80.68%*	
3	*393M*	*80M*	*79.64%*	
4	*412M*	*85M*	*79.37%*	

As a real application, we apply WSA to straighten high-resolution 3D fluorescence images.

The effectiveness of our method can be seen by the minimal loss of image information in straightening, as shown in Fig. 3. The original data has three color-channels, each for a different set of cells. The blue channel is DAPI staining of nuclear DNA molecules. The green is GFP and red is mCherry. It can be seen that the original worm body with scattered cells in Fig. 3 (a) can be successfully straightened to a meaningful rod-shape in Fig. 3 (b). This indicates that WSA is able to detect the globally nonlinear manifold and transform it into a canonical space. Because we use 1-pixel spacing between all cutting-planes to restack the entire worm, the resolution is the same as the original image, although due to the sub-pixel level interpolation the straightened worm may look slightly smoother than the original worm. Overall the loss of information is minimal, which can be seen in the example of the tail region. Our biologist collaborators cannot find any visible distortion of the signal such as the change of cell locations or intensity.

We have applied WSA to our on-going digital worm cell atlas project, which routinely produces large 3D worm image stacks. Table 1 demonstrates several examples of the image file sizes before and after straightening. On average WSA reduces nearly 80% of the image file size, which we think is significant for our project that will eventually produce thousands of these images.

ACKNOWLEDGEMENTS

We thank Stuart Kim and Xiao Liu for data preparation.

REFERENCES

[1] http://www.wormbase.org/.

[2] Hastie, T., "Principal curves and surfaces," Ph.D. Thesis, Stanford Univ., 1984.

[3] Peng, H.C., Long, F., Eisen, M.B., and Myers, E.W. "Clustering gene expression patterns of fly embryos," Proc. IEEE ISBI 2006, pp.1144-1147, Washington DC, April 6-9, 2006.

[4] B. Kégl, A. Krzyzak, T. Linder, and K. Zeger, "Learning and design of principal curves," TPAMI, 22(3), pp. 281-297, 2000.

[5] Jain, A.K., *Fundamentals of Digital Image Processing*, Prentice Hall, 1989.

[6] Blum, H., "A transformation for extracting new descriptions of shape," Symp. on Models for the Perception of Speech and Visual Form, Cambridge: MIT Press, 1964.

[7] Corman, T.H., Leiserson, C.E., and Rivest, R.L., Introduction to Algorithms, MIT, 2001.