Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Supplementary Material

OXFORD

Supplementary Material

DiversePathsJ: Diverse Shortest Paths for Bioimage Analysis

Virginie Uhlmann [1,*], Carsten Haubold [2,*], Fred Hamprecht [2] and Michael Unser [1]

[1]École Polytechnique Fédérale de Lausanne (EPFL), Biomedical Imaging Group, 1015 Lausanne, Switzerland and
[2]Heidelberg University, IWR/HCI, 69120 Heidelberg, Germany.

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

1 Theoretical Formulations

1.1 Shortest Path with the Viterbi Algorithm

The Viterbi algorithm is a widely-used dynamic programming (DP) algorithm for solving the problem of finding the shortest path between a source and a target point. It relies on a trellis graph G composed of the set of nodes

$$\{(\nu, x_\nu)|1 \leq \nu \leq N,\ x_\nu \in \mathcal{X}\} \cup \{\sigma, \tau\}, \tag{1}$$

where σ is the source node, τ the target node, and x_ν the state of node ν, which belong to the set of labels $\mathcal{X}$ of size L. The resulting graph is therefore composed of $NL + 2$ vertices, as illustrated in Figure 1a. Edges can be described by the set

$$\{((\nu, x_\nu),(\nu+1, x_{\nu+1}))|1 \leq \nu \leq N,\ x_\nu \in \mathcal{X}\} \cup \{(\sigma, (1, x_1))|x_1 \in \mathcal{X}\} \cup \{((N, x_N), \tau)|x_N \in \mathcal{X}\}. \tag{2}$$

We denote the full state vector for the complete graph as $\mathbf{x} = \{x_\nu : 1 \leq \nu \leq N\}$.

The energy function associated to the graph is composed of two types of elements: $\theta_\nu(x_\nu)$, the cost for assigning the label x_ν to node ν, and $\theta_{\nu,\nu+1}(x_\nu, x_{\nu+1})$, the cost for assigning labels x_ν and $x_{\nu+1}$ to the two neighboring nodes ν and $\nu + 1$, respectively. For $1 \leq \nu < N$, going from node (ν, x_ν) to $(\nu + 1, x_{\nu+1})$ costs $\theta_{\nu+1}(x_{\nu+1}) + \theta_{\nu,\nu+1}(x_\nu, x_{\nu+1})$. Then, going from σ to $(1, x_1)$ costs $\theta_1(x_1)$, and from (N, x_N) to τ costs nothing. The shortest path problem can then be expressed as an energy minimization task with the objective

$$\min_{\mathbf{x}} E(\mathbf{x}) = \min_{\mathbf{x}} \sum_{\nu=1}^{N} \theta_\nu(x_\nu) + \sum_{\nu=1}^{N-1} \theta_{\nu,\nu+1}(x_\nu, x_{\nu+1}). \tag{3}$$

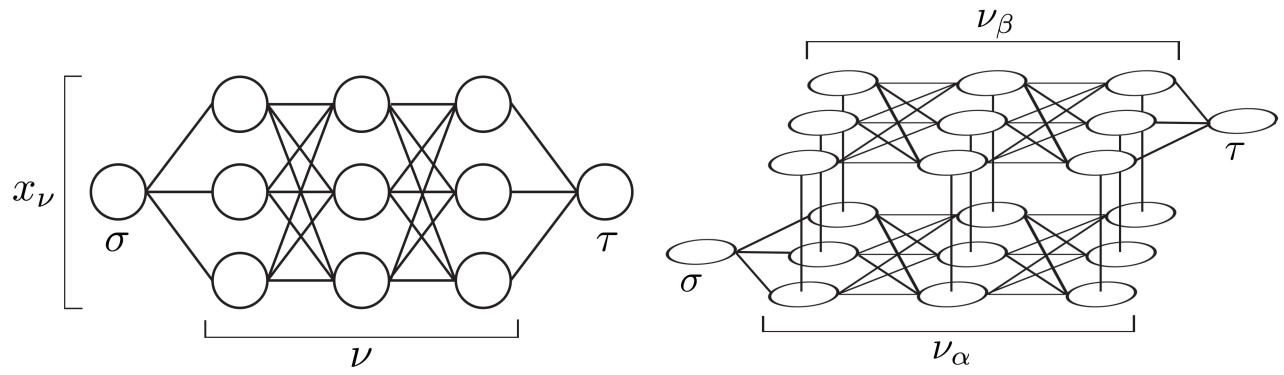
To run DP, one successively computes the cost of optimal solutions of subproblems of increasing size. Whenever a node ν knows the cost from its incoming edge, it can compute its own energy $E_\nu(x)$ for the first ν nodes in the sequence. The global solution is therefore obtained by solving smaller subproblems. The corresponding recursive equations are given by

$$E_1(x) = \theta_1(x_1), \tag{4}$$

$$E_\nu(x) = \theta_\nu(x_\nu) + \min_{x_{\nu-1}} \left[\theta_{\nu-1,\nu}(x_{\nu-1}, x_\nu) + E_{\nu-1}(x)\right] \tag{5}$$

for $2 \leq \nu \leq N$. Once the intermediate energies $E_\nu(x)$ have been computed for all ν, the optimal solution of the global problem is obtained by taking $x^*_\tau = \arg\min_{x_\tau} E_\tau(x)$ at the target node τ and backtracking down the graph until the source node σ following

$$x^*_\nu = \arg\min_{x_\nu} \left[\theta_{\nu,\nu+1}(x_\nu, x^*_{\nu+1}) + E_\nu(x)\right]. \tag{6}$$



(a) (b)

Fig. 1. Example for $N = 3$ and $L = 3$ of (a) the standard Viterbi algorithm graph G, and (b) the resulting two-layers graph $\tilde{G}$ for solving M-shortest paths. The source node is denoted by σ and the target node by τ.

1.2 Diverse M-Shortest Paths

To search for diverse M-shortest paths with the Viterbi algorithm, we construct an auxiliary graph that holds the original graph (which we denote as layer α), a second copy (layer β), and additional edges to jump between layers. This allows us to find a collection of M-best solutions obeying a user-defined diversity constraint, which is encoded in the potentials of the *layer-jump-edges*. The more general formulation of our approach is presented in (Haubold *et al.*, 2017). Here, we focus on the subcase covering the particular class of shortest path applications we are interested in.

Let $\mathbf{x}^*$ be the best solution (*i.e.*, the shortest path) $\mathbf{x}^* = \arg\min_{\mathbf{x}} E(\mathbf{x})$ in graph G. We create the new layered graph $\tilde{G}$ from G with nodes

$$\{(\nu_\alpha, x_{\nu_\alpha}), (\nu_\beta, x_{\nu_\beta}) | 1 \leq \nu \leq N,\, x_\nu \in \mathcal{X}\} \cup \{\sigma, \tau\}, \tag{7}$$

and edges

$$\{((\nu_\alpha, x_{\nu_\alpha}), (\nu_\alpha + 1, x_{\nu_\alpha+1})), ((\nu_\beta, x_{\nu_\beta}), (\nu_\beta + 1, x_{\nu_\beta+1})) | 1 \leq \nu \leq N,\, x_\nu \in \mathcal{X}\} \tag{8}$$

$$\cup \{(\sigma, (1_\alpha, x_{1_\alpha})), ((N_\beta, x_{N_\beta}), \tau)\} \tag{9}$$

$$\cup \{((\nu_\alpha, x_{\nu_\alpha}), (\nu_\beta, x_{\nu_\beta})) | 1 \leq \nu \leq N\}. \tag{10}$$

Note that (7), (8) and (9) are simply duplicated nodes and edges from G, and that (10) corresponds to the inclusion of the layer jump edges. An example of resulting two-layers graph is illustrated in Figure 1b.

The potentials $\tilde{\theta}$ are conserved for the nodes and edges duplicates. Within each layer, they are hence defined as

$$\tilde{\theta}_{\nu_\alpha} := \theta_\nu,\, \tilde{\theta}_{\nu_\beta} := \theta_\nu, \tag{11}$$

for $1 \leq \nu \leq N$, and as

$$\tilde{\theta}_{\nu_\alpha, \nu_\alpha+1} := \theta_{\nu,\nu+1},\, \tilde{\theta}_{\nu_\beta, \nu_\beta+1} := \theta_{\nu,\nu+1} \tag{12}$$

for $1 \leq \nu < N$. Between the two layers, new potentials are introduced as

$$\tilde{\theta}_{\nu_\alpha, \nu_\beta}(x_{\nu_\alpha}, x_{\nu_\beta}) := \begin{cases} \infty & \text{if } x_{\nu_\alpha} \neq x_{\nu_\beta} \\ \infty & \text{if } d(x_{\nu_\alpha}, x^*_\nu) < D \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

for $1 \leq \nu \leq N$, where d represents a distance to the previous solution x^*, for instance $d(x_{\nu_\alpha}, x^*_\nu) = |x_{\nu_\beta} - x^*_\nu|$.

In the two-layers graph, the DP rules are modified as

$$E_{1_\alpha}(x) = \theta_{1_\alpha}(x_{1_\alpha}), \tag{14}$$

$$E_{1_\beta}(x) = E_{1_\alpha}(x) + \tilde{\theta}_{1_\alpha, 1_\beta}(x_{1_\alpha}, x_{1_\beta}), \tag{15}$$

$$E_{\nu_\alpha}(x) = \tilde{\theta}_{\nu_\alpha}(x_{\nu_\alpha}) + \min_{x_{\nu_\alpha - 1}} \left[\tilde{\theta}_{\nu_\alpha - 1, \nu_\alpha}(x_{\nu_\alpha - 1}, x_{\nu_\alpha}) + E_{\nu_\alpha - 1}(x)\right] \tag{16}$$

$$E_{\nu_\beta}(x) = \min\left(E_{\nu_\alpha}(x) + \tilde{\theta}_{\nu_\alpha, \nu_\beta}(x_{\nu_\alpha}, x_{\nu_\beta}), \tilde{\theta}_{\nu_\beta}(x_{\nu_\beta}) + \min_{x_{\nu_\beta - 1}} \left[\tilde{\theta}_{\nu_\beta - 1, \nu_\beta}(x_{\nu_\beta - 1}, x_{\nu_\beta}) + E_{\nu_\beta - 1}(x)\right]\right). \tag{17}$$

The modification can be easily explained as follows. In the second layer β, an initial layer jump cost is first added (15). Then, the best path up to node ν_β is given by (17), which simply selects the shortest between paths jumping from the bottom layer α and paths already coming from layer β. The layer-jump-edges availability can then modulated based on the distance of a solution to the previous ones.

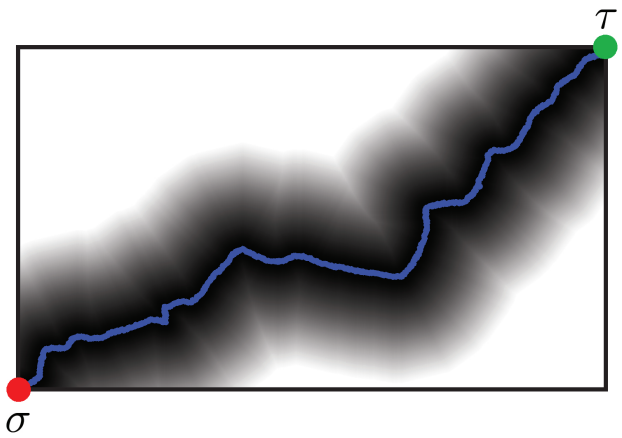


Fig. 2. Example of a diversity map for the introduction of diversity accumulation constraint in the M-shortest paths problem.

1.2.1 Diversity accumulation

So far, our formulation allows to modulate the layer-jump-edges availability based on the distance of a solution to the previous ones. A possible refinement is to not only enforce that the new solution is at least at one point distant enough from the previous one, but that it accumulates a given amount of diversity.

To do so, we introduce an additional map on the lower layer α that integrates over *diversity*, which is computed from a (possibly user-defined) image or metrics. A possible diversity map is shown in Figure 2. The availability of layer-jump-edges is now conditioned on the amount of collected diversity. In other words, a layer jump is allowed only when the summed diversity over the path exceeds a threshold.

Under this constraint, the optimization problem becomes

$$\min_{\mathbf{x}} E(\mathbf{x}) = \min_{\mathbf{x}} \sum_{\nu=1}^{N} \theta_\nu(x_\nu) + \sum_{\nu=1}^{N-1} \theta_{\nu,\nu+1}(x_\nu, x_{\nu+1}) \tag{18}$$

$$\text{s. t. } \sum_{\nu=1}^{N} \delta_\nu(x_\nu) + \sum_{\nu=1}^{N-1} \delta_{\nu,\nu+1}(x_\nu, x_{\nu+1}) > T, \tag{19}$$

and the modified DP update rules are given by

$$E_{1_\alpha}(x) = \theta_{1_\alpha}(x_{1_\alpha}), \tag{20}$$

$$E_{1_\beta}(x) = E_{1_\alpha}(x) + \tilde{\theta}_{1_\alpha,1_\beta}(x_{1_\alpha}, x_{1_\beta}) + c(x_1), \tag{21}$$

$$E_{\nu_\alpha}(x) = \tilde{\theta}_{\nu_\alpha}(x_{\nu_\alpha}) + \min_{x_{\nu_\alpha-1}} \left[\tilde{\theta}_{\nu_\alpha-1,\nu_\alpha}(x_{\nu_\alpha-1}, x_{\nu_\alpha}) + E_{\nu_\alpha-1}(x)\right] \tag{22}$$

$$E_{\nu_\beta}(x) = \min\left(c(x_\nu) + E_{\nu_\alpha}(x) + \tilde{\theta}_{\nu_\alpha,\nu_\beta}(x_{\nu_\alpha}, x_{\nu_\beta}), \tilde{\theta}_{\nu_\beta}(x_{\nu_\beta}) + \min_{x_{\nu_\beta-1}} \left[\tilde{\theta}_{\nu_\beta-1,\nu_\beta}(x_{\nu_\beta-1}, x_{\nu_\beta}) + E_{\nu_\beta-1}(x)\right]\right), \tag{23}$$

where c is a constraint function of the form

$$c(x_\nu) = \begin{cases} 0 & \text{if } \sum_{n=1}^{\nu} \delta_n(x_n) + \sum_{n=1}^{\nu-1} \delta_{n,n+1}(x_n, x_{n+1}) > T \\ \infty & \text{otherwise} \end{cases}. \tag{24}$$

2 Software

DiversePathsJ, our M-shortest path plugin for ImageJ (Abràmoff *et al.*, 2004) and Fiji (Schindelin *et al.*, 2012), requires the specification of the following inputs parameters through its GUI (Figure 3).

1. **Number of diverse paths.** Number M of diverse paths to be computed.
2. **Size of exclusion corridor.** Distance (in pixels) that any valid new solution must have to all previous ones, at least at one point.
3. **Accumulated diversity threshold.** Minimal amount of diversity (in pixels) that any new solution must accumulate with respect to all previous ones.
4. **Target type.** Type of object on which the shortest paths have to be searched (*bright on dark background* or *dark on bright background*).
5. **Processing filter.** Options for processing the input image and highlight features of interest prior to shortest path search. The options are *none* (no processing), *EDM* (Euclidean distance map, highlights the centerline of objects), *edge filter* (highlights edges), *ridge filter* (highlights ridges), and *external* (custom processing).

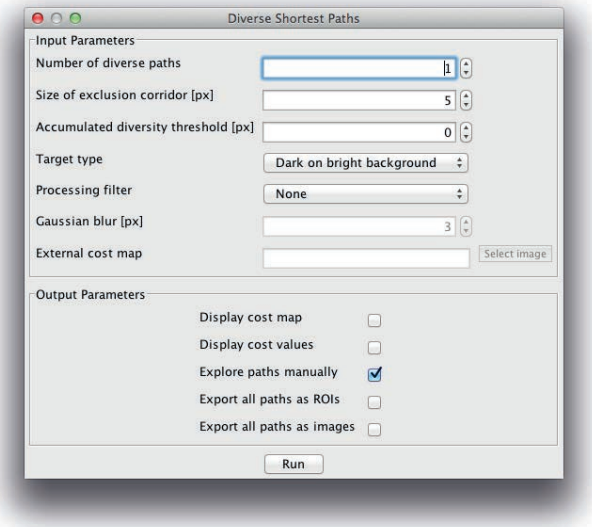


Fig. 3. General User Interface of the DiversePathJ plugin.

6. **Gaussian blur.** This option is only available when the *edge* and *ridge* processing filters are chosen. It should be set as the approximate size (in pixels) of the edge or ridge object to be highlighted.
7. **External cost map.** Path to a file path containing a processed version of the input image. This option is only available when the *external* processing filter is chosen. It allows for the user to input a custom cost map to be used by the Viterbi algorithm.

The cost function implemented in DiversePathsJ is very general to remain flexible enough to accommodate a wide range of use-cases. The cost between two successive nodes is computed as a convex combination of a data-fidelity term (the integral of the cost map image under the edge linking the nodes) and a regularization term (the distance between their states). Paths of low cost therefore tend to follow dark areas on the cost map image and to remain locally smooth.

The user can select any of the following different outputs.

1. **Cost map.** Displays the underlying cost map image used by the Viterbi path in a new image window.
2. **Cost values.** Displays the value of the cost of each computed path in ImageJ/Fiji's log window.
3. **Manual exploration of cost map.** By default, displays the first shortest path as an overlay on the original input image. Upon clicking on the right (resp. left) arrows of the keyboard, the overlay is updated with the second (resp. Mth) shortest path. The complete collection of computed paths can thus be explored interactively. Upon clicking on the space bar, the currently displayed path is validated and exported as a Polyline ROI into ImageJ/Fiji's ROI Manager.
4. **Paths as ROIs.** Exports the M computed paths as M Polyline ROIs into ImageJ/Fiji's ROI Manager.
5. **Paths as images.** For each computed path, creates and displays the path as overlayed on a copy of the original image. For M path, this option therefore creates M new image windows.

The software, a complete user-manual, a video tutorial and test data are available on the webpage of the plugin[1]. DiversePathsJ can be run in macro mode and can thus easily be integrated as parts of larger analysis pipelines.

3 Experiments

3.1 Protocol

For all provided results, the experimental procedure is as follows.

1. Open the image,
2. Select two end-points via ImageJ/Fiji's *multi-point* selection tool,
3. Launch DiversePathsJ from the *Plugins* menu,
4. Set plugin settings in DiversePathsJ GUI,
5. Run the plugin.

The initial points and software settings for the two experiments presented in the manuscript are given below.

[1] http://bigwww.epfl.ch/algorithms/diversepathsj

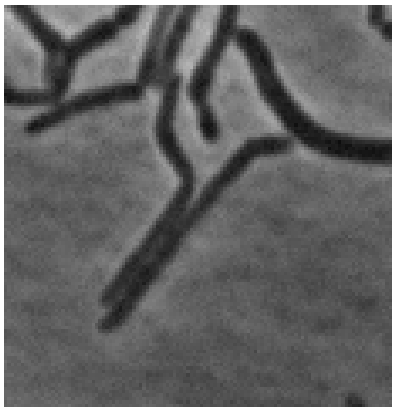
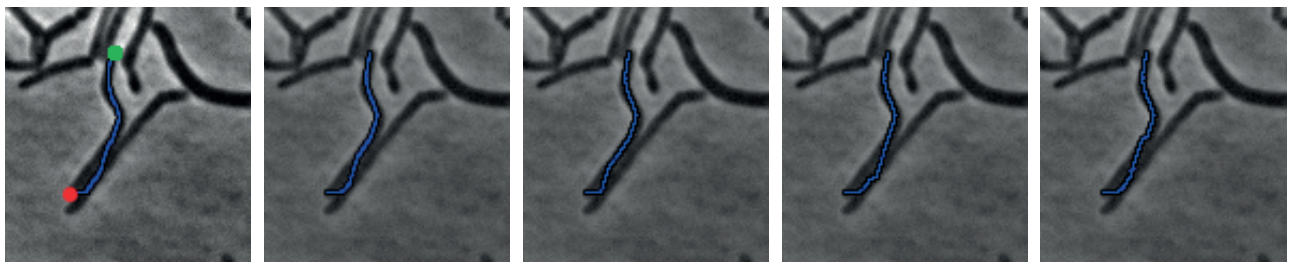
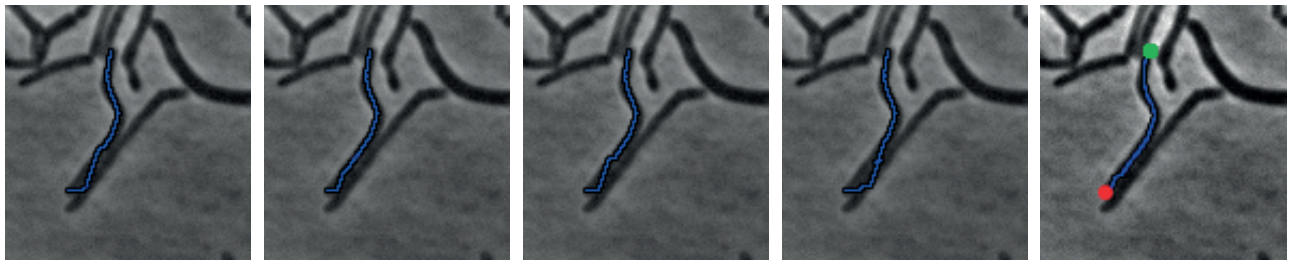


Fig. 4. Phase-contrast microscope image of mycobacteria. Image courtesy of the Laboratory of Microbiology and Microsystems (UPKIN), EPFL, Switzerland.



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



(f) (g) (h) (i) (j)

Fig. 5. Incorrect (a) first to (i) ninth shortest paths, and correct (j) tenth shortest path for the mycobacteria image.

3.2 Software Settings

3.2.1 Mycobacteria image

The images are courtesy of the Laboratory of Microbiology and Microsystems (UPKIN), EPFL, Switzerland.

- **Image format:** 115×119 pixels 8-bits gray-scale .tif image (Figure 4)
- **Endpoint coordinates:** $(28, 97)$—$(84, 41)$ (botton bacteria) and $(29, 86)$—$(50, 21)$ (top bacteria)
- **Number of diverse paths:** 10
- **Size of exclusion corridor:** 1
- **Accumulated diversity threshold:** 10
- **Target type:** Dark on bright background
- **Processing filter:** Ridge filter
- **Gaussian blur:** 3

The medial axis of the bottom-most bacteria is found as the first shortest path between source point $(28, 97)$ and target point $(84, 41)$. The medial axis of the top one is found as the tenth shortest path between source point $(29, 86)$ and target point $(50, 21)$ (Figures 5j). The first to ninth shortest paths are wrong due to the low contrast between touching bacterias. Since the image resolution is quite poor (115×119), the Viterbi paths gets trapped in many local minima, as seen in Figures 5a to 5i. Since the desired path lies very close to the unwanted ones, relying on a too wide exclusion corridor makes the algorithm miss the correct solution. Imposing a given amount of accumulated diversity however allows obtaining a proper medial axis. The user can swiftly scan through the proposed solution using the arrow keys.

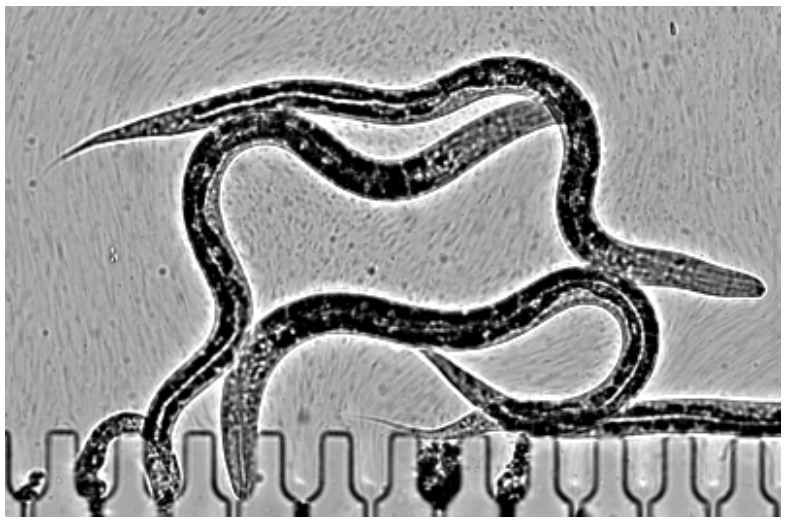
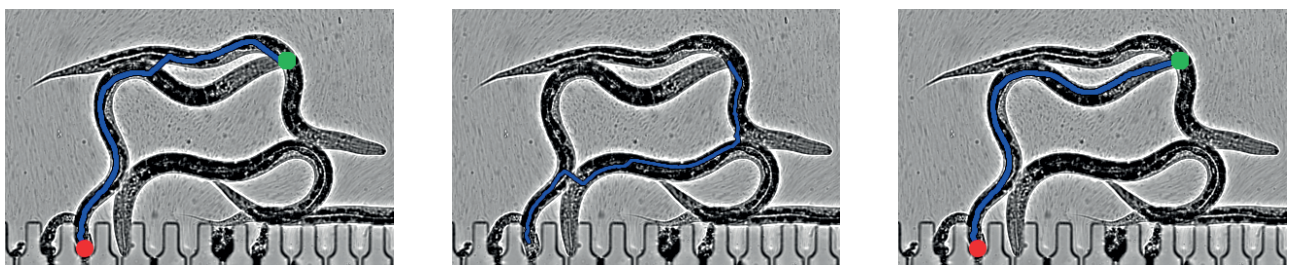


Fig. 6. Brightfield microscope image of C. elegans worms. Image courtesy of M. Cornaglia, Laboratory of Microsystems (LMIS2), EPFL, Switzerland.



(a) (b) (c)

Fig. 7. Incorrect (a) first and (b) second shortest paths, and correct (c) third shortest path for the worms image.

3.2.2 *C. elegans* image

The images are courtesy of M. Cornaglia, Laboratory of Microsystems (LMIS2), EPFL, Switzerland.

- **Image format:** 400×263 pixels 8-bits gray-scale .tif image (Figure 6)
- **Endpoint coordinates:** $(31, 75)$—$(390, 147)$ (top worm) and $(79, 240)$—$(286, 54)$ (bottom worm)
- **Number of diverse paths:** 10
- **Size of exclusion corridor:** 25
- **Accumulated diversity threshold:** 40
- **Target type:** Dark on bright background
- **Processing filter:** Ridge filter
- **Gaussian blur:** 6

The medial axis of the top-most worm is found as the first shortest path between source point $(31, 75)$ and target point $(390, 147)$. The medial axis of the bottom one is found as the third shortest path between source point $(79, 240)$ and target point $(286, 54)$ (Figures 7c). The first and second shortest paths are wrong due to neighboring worms, as seen in Figures 7a and 7b.

References

Abràmoff, M., Magalhães, P., and Ram, S. (2004). Image processing with imagej. *Biophotonics International*, **11**(7), 36–42.

Haubold, C. *et al.* (2017). Diverse M-best solutions by dynamic programming. In *Proc. 39th German Conf. Pattern Recognit. (GCPR'17)*, Basel, Switzerland.

Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., Preibisch, S., Rueden, C., Saalfeld, S., Schmid, B., Tinevez, J.-Y., White, D., Hartenstein, V., Eliceiri, K., Tomancak, P., and Cardona, A. (2012). Fiji: an open-source platform for biological-image analysis. *Nature methods*, **9**(7), 676–682.