INEXACT GRAPH MATCHING FOR NEURON TRACKING IN CALCIUM IMAGING

K. V. Busum[1], *W. D. Tracy*[2], *L. He*[2], *S. Gulyanon*[3], *and G. Tsechpenakis*[1]

[1]Computer and Information Science Department, Indiana University-Purdue University Indianapolis, USA;

[2] Department of Biology, Indiana University, USA;

[3]Data Science and Innovation Program, Faculty of Science and Technology, Thammasat University, Thailand;

ABSTRACT

In this work, we propose the novel neuron tracking method that handles the issue in calcium images using the inexact graph matching over a set of graphs, each representing the neuron morphology at a time step. The match is computed by minimizing the graph edit distance that incorporates the tissue deformation, local image feature, neuronal morphology, and local characteristics of neurons.

***Index Terms*—** neuron tracking, calcium images, graph matching, graph edit distance, A*-beamsearch

1. INTRODUCTION

In this neuron tracking problem [1], we are given the calcium image stack sequence, $\mathcal{I} = \{\mathcal{I}^{(t)}\}$, where $t = 0, 1, \ldots, T$. To make the problem feasible, we require the initial morphology of the neuron, $\mathcal{G} = \{V_{\mathcal{G}}, E_{\mathcal{G}}\}$.

2. DEFINITIONS

A *graph* is denoted by $G = \{V, E\}$, where V is the set of nodes (also called *vertices*) and $E \subset V \times V$ (or $E \subset [V]^2$) is the set of edges (also known as lines) of graph G.

The problem of *inexact graph matching* (IGM) or error-tolerant/correcting GM is finding the mapping of nodes between two graphs that optimizes a certain affinity or distortion criterion, where nodes do not need to have a correspondence in another graph.

A commonly used measurement of affinity criterion is the *graph edit distance* (GED) [2]. A graph can be transformed to another one by a finite sequence of graph edit operations. Each operation has a cost, which is defined differently in various algorithms. GED, $\delta(G_i, G_j)$, is defined by the least-cost graph edit operation sequence, P, that is needed to transform one graph, $G_i = \{V_i, E_i\}$, into another, $G_j = \{V_j, E_j\}$.

$$\delta(G_i, G_j) = \min_{P} \sum_{(u \to v) \in P} c(u \to v) \qquad (1)$$

This work is supported by NSF/DBI [#1252597]: 'CAREER: Modeling the structure and dynamics of neuronal circuits in the *Drosophila* larvae using image analytics' awarded to G. Tsechpenakis.

where nodes $u \in V_i$ and $v \in V_j$ are a correspondence; and $c(u \to v)$ is the cost of the graph edit operation that maps node u to v, which depends on the affinity between u and v.

3. METHOD

In this work, we tackle the neuron tracking problem over calcium images as the IGM problem computed by minimizing the GED.

Given the calcium image sequence, first convert every frame into the "frame" graph whose nodes are superpixels. Second, compute the "template" graph, which is the subgraph of the "frame" graph corresponded to the neuron trace, using the IGM. Finally, recover the trace by aligning every dendrite to the image cues, and repeat throughout the sequence (fig. 1).

3.1. Creating "Frame" Graph

Due to the imaging system limitation, the maximum intensity projection of calcium image stack, $I_t = f(\mathcal{I}^{(t)})$, is used instead of the raw input volume.

To convert a frame into a graph, first compute the superpixel segmentation over I_t using techniques like Linear Spectral Clustering Superpixel [3, 4][1] or simple linear iterative clustering (SLIC). The result is the set of superpixels V_t.

Let the "frame" graph be $G_t = \{V_t, E_t\}$, where the existence of edges E_t between superpixels are computed using Delaunay triangulation over the centroids of superpixels in V_t.

[1]`https://jschenthu.weebly.com/projects.html`, `https://www.mathworks.com/matlabcentral/fileexchange/65684-linear-spectral-clustering-superpixel`

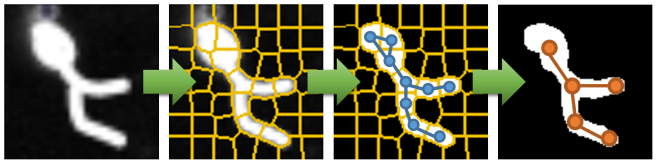

Fig. 1. From left to right. Input image, the "frame" graph, the "template" graph, and the aligned neuron trace. Yellow lines show the boundary of superpixels. Blue color shows the "template" graph and orange colors shows the neuron trace result.

3.2. Initial "Template" Graph

The trace describes the neuron morphology at the structural level, while the input image stacks describe the neuron at the appearance level. We propose the "template" graph that bridges between two levels of representations.

The "template" graph at time $t = 0$, $G'_0 = \{V'_0, E'_0\}$, is the subgraph of G_0 corresponded to the initial morphology $\mathcal{G}$. V'_0 is the set of patches overlapping with $\mathcal{G}$,

$$V'_0 = \{\, v \mid v \text{ overlaps } \mathcal{G}, v \in V_0 \,\} \tag{2}$$

E'_0 is the set of edges between adjacent nodes in V'_0,

$$E'_0 = \{\, (u,v) \mid (u,v) \in E_0;\ u,v \in V'_0 \,\} \tag{3}$$

3.3. Tree Search Algorithm

A tree (state space) search algorithm [5] is used to find the GED because it is simple, allows the many-to-many mapping function, and makes no assumptions about the problem. Tracking is done by computing the GED between the previous "template" graph and the current "frame" graph, $\delta(G'_{t-1}, G_t)$. The tree state space consists of a number of possible mappings and the corresponding cost (fig. 2). The algorithm terminates when every node in G'_{t-1} has at least one correspondence in G_t.

A state is produced by adding a new correspondence to the parent state. Let ψ be a function that determines, for a given state/mapping p, a set of possible new correspondences that can be added to form new child states.

$$\psi(p) = \{\check{V}_{p,t-1} \times \check{V}_{p,t}\} \cup \{\check{V}_{p,t-1} \times V_{p,t}\} \cup \{V_{p,t-1} \times \check{V}_{p,t}\} \tag{4}$$

where $\check{V}_{p,t}$ is the set of nodes that are not currently mapped and are adjacent to the set of mapped nodes $V_{p,t}$, $\check{V}_{p,t} = \{\, v \mid (u,v) \in E_t, u \in V_{p,t}, v \notin V_{p,t} \,\}$.

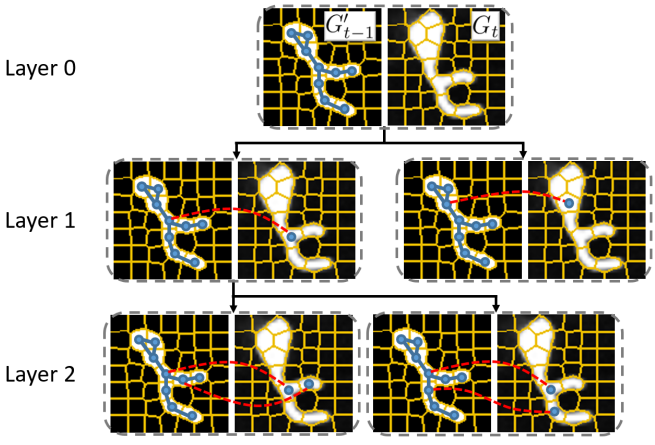

Fig. 2. Tree state space. The root of the state space represents the mapping with no correspondences. The first layer of states represents mappings with one correspondence, and so on. Red lines show pairs of nodes that are a correspondence.

3.4. Coarse Tracking

Let P be the graph edit operation sequence with the optimal GED; and $P(V_t)$ denotes the set of nodes V_t mapped by the sequence P. Then, the "template" graph at time t is defined by $G'_t = \{V'_t, E'_t\}$, where

$$V'_t = P(V_t) \tag{5}$$

And E'_t is the set of edges between adjacent nodes in V'_t,

$$E'_t = \{\, (u,v) \mid (u,v) \in E_t;\ u,v \in V'_t \,\} \tag{6}$$

3.5. Fine Tracking: Aligning Dendrites

Consider a set of vertices, $V_d \subset V_{\mathcal{G}}$, that constitutes a dendrite/branch d in the neuron trace. The nodes corresponding to the dendrite d in the "template" graph at time $t = 0$ is

$$V'_{d,0} = \{\, v \mid v \text{ overlap with } V_d, v \in V_0 \,\} \tag{7}$$

For time t,

$$V'_{d,t} = P(V'_{d,t-1}) \tag{8}$$

The alignment between dendrite at consecutive time step is done by registering patches from $V'_{d,t-1}$ to $V'_{d,t}$ using techniques like free-form deformation (FFD) [6].

4. OPTIMIZATION

A tree search algorithm in Section 3.3 that explores the whole state space is inefficient because the size of the state space is $|V'_{t-1}||V_t|$ and many states have no meaningful interpretation, e.g., mappings that produce unconnected dendrites.

In this work, the computation of GED is improved by using the A*-beamsearch [7] with node grouping, that only explores the most likely and meaningful states and allows many-to-many mapping solution.

4.1. A* Search

The A* or the best-first search algorithm [8] finds an optimal path from a state space based on a heuristic function, which estimates the expected costs of the best route from the root through the current state (partial solution) to a leaf state (complete solution). At each step during tree state space traversal, the most promising state — the lowest heuristic cost — from the set of child states is chosen. Hence, at any state p, A* algorithm selects the path that minimizes the following cost:

$$f(p) = g(p) + h(p) \tag{9}$$

where $g(p)$ is the cost of the optimal path from the root to the current state p and $h(p)$ is the estimated cost from state p to a leaf state. Here, $h(p)$ is defined by the average cost of $g(p)$ times the number of remaining unmapped nodes.

4.2. Beam Search

In A* search, we may end up expanding all successor nodes in the search tree. In beam search, only a fixed number, b called *beam width*, of states in each layer are explored. We pick the b states with the lowest costs from eq. 9.

This means only those states with the most promising partial mappings are explored. Since a graph edit operation sequence between similar graphs has lower cost than the one between dissimilar graphs [7].

5. COST FUNCTION

The cost function, $c(u \to v)$, of the GED $\delta(G'_{t-1}, G_t)$ takes a correspondence and returns a single value measuring its similarity based on the tissue deformation, local image feature, neuronal morphology, and local characteristics of neurons.

Edge comparison is not meaningful here since edge captures the relationship between nodes but the correspondent nodes are not guaranteed to be the same. However, we still need the neighborhood information that edges have. Thus, we aggregate the edge descriptors and embed them in the node descriptor. Next, we describe the cost function between two nodes (one-to-one function) and between a node and a group of nodes (many-to-one/one-to-many functions)

5.1. Node to Node

Average intensity of any node $v_i \in V_t$ is defined by,

$$I_t(v_i) = \frac{1}{|S(v_i)|} \sum_{x \in S(v_i)} I_t(x) \tag{10}$$

where $S(v_i)$ is the set of pixels in the superpixel corresponded to node v_i.

Average eigenvector from Frangi filter [9] indicates the neurite orientation,

$$F(v_i) = \frac{1}{|S(v_i)|} \sum_{x \in S(v_i)} \hat{\mathbf{u}}_{x,1} \tag{11}$$

where Let $\lambda_{s,x,k}$ denote the eigenvalue corresponding to the k-th normalized eigenvector $\hat{\mathbf{u}}_{s,x,k}$ of the Hessian matrix of the image $\mathcal{H}_{x,k}$ all computed at scale s on pixel x.

$$\lambda_{x,k} = \max_{s_{min} \le s \le s_{max}} \lambda_{s,x,k} \tag{12}$$

will be the eigenvalue with the k-th smallest magnitude ($\lambda_{x,1} < \lambda_{x,2}$) and $\hat{\mathbf{u}}_{x,k}$ are their corresponding eigenvectors.

Relation to neighbors measures the intensity similarity between a node and its neighborhood. To compute, first find the location L_{ij} and magnitude M_{ij} based on the average intensity similarity for each neighbor v_j of v_i, $(v_i, v_j) \in E_t$,

$$\begin{aligned} L_{ij} &= C(v_j) - C(v_i) \\ M_{ij} &= 1 - \|I_t(v_j) - I_t(v_i)\| \end{aligned} \tag{13}$$

where $C(v_i)$ is the centroid of the superpixel corresponded to node v_i denoted by the xy-coordinate, (x_i, y_i).

Then, a histogram is created to encode the relation to neighboring superpixels, similar to the orientation histogram in SIFT [10]. In this histogram, the 360 degrees of L_{ij} direction are broken into 8 bins (each 45 degrees). The amount added to the bin depends on the magnitude projected in the bin's direction.

To achieve rotation independence, make the orientation relative to the keypoint's orientation by subtracting the keypoint's rotation from each orientation. The keypoint is the bin with the highest magnitude.

To achieve illumination dependence, threshold the magnitudes that are too big.

Similarly, **relation to mapping** indicates the proximity between v_i and mapped nodes. This helps distinguish nodes along ambiguous regions like dendrites since points along the dendritic line segment have similar properties. It is a histogram of mapped neighbors' orientation weighted by the inverse of distance H_{ij}, where $v_j \in V_{p,t}$.

$$H_{ij} = \frac{1}{\|L_{ij}\|} \tag{14}$$

Smooth deformation keeps the change in deformation low over space. For $u_i^{t-1} \in \check{V}_{p,t-1}$ and $v_i^t \in \check{V}_{p,t}$, the deformation of the correspondence i is defined as,

$$d_i = C(u_i^{t-1}) - C(v_i^t) \tag{15}$$

For $(u_i^{t-1}, u_j^{t-1}) \in E'_{t-1}$ and $(v_i^t, v_j^t) \in E_t$, where u_i^{t-1} and v_i^t are correspondence we are considering; $u_j^{t-1} \in V_{p,t-1}$ and $v_j^t \in V_{p,t}$ are the adjacent correspondences we found earlier. Then, the change in deformation at correspondence i is:

$$D_i = \sum_{d_j} \|d_j - d_i\| \tag{16}$$

Hence, the node descriptor $\alpha(v_i)$ is defined as the tuple of 19 numbers: average intensity $I_t(v_i)$, the orientation of the average eigenvector $F(v_i)$ in radian, smooth deformation T_i, and the values in both histograms' bins.

The cost of substituting node $(u, v) \in \{\check{V}_{p,t-1} \times \check{V}_{p,t}\}$ is defined based on cosine similarity as follows:

$$c(u \to v) = 1 - \frac{\sum_{i=1}^{N} \alpha_i(u)\alpha_i(v)}{\sqrt{\sum_{i=1}^{N} \alpha_i^2(u)}\sqrt{\sum_{i=1}^{N} \alpha_i^2(v)}} \tag{17}$$

5.2. Node to Group of Nodes

The node grouping is needed for finding many-to-many relationship between frames in an efficient way, similar to [5].

The cost of substituting node $(u, v) \in \{\check{V}_{p,t-1} \times V_{p,t}\}$ is defined based on cosine similarity as before but the labeling

function of u must take into account other correspondences of v.

Let $\hat{u}$ is the set of nodes correspondence to v, $\hat{u} = \{\, w \mid (w \rightarrow v) \in p \,\} \cup \{u\}$. Then, $\alpha(\hat{u})$ denotes the group-of-node descriptor. **Average intensity** and **average eigenvector** of a group of nodes are defined as,

$$\begin{aligned} I_t(\hat{u}) &= \frac{1}{|S(\hat{u})|} \sum_{x \in S(\hat{u})} I_t(x) \\ F(\hat{u}) &= \frac{1}{|S(\hat{u})|} \sum_{x \in S(\hat{u})} \hat{\mathbf{u}}_{x,1} \end{aligned} \tag{18}$$

The **relation to neighbors** and **relation to mapping** are quantified from the neighbors of the group of nodes, ignoring the internal edges within the group. Using the centroid of the group of nodes and edges between a group of nodes and other nodes.

The cost of substituting node $(u, v) \in \{V_{p,t-1} \times \check{V}_{p,t}\}$ follows the same suit.

6. RESULTS

7. CONCLUSION

8. REFERENCES

[1] S. Gulyanon, L. He, W.D. Tracey, and G. Tsechpenakis, "Neurite tracking in time-lapse calcium images using MRF-modeled pictorial structures," *ISBI*, pp. 1564–1568, 2018.

[2] X. Gao, B. Xiao, D. Tao, and X. Li, "A survey of graph edit distance," *Pattern Anal Appl*, vol. 13, no. 1, pp. 113–129, 2010.

[3] Z. Li and J. Chen, "Superpixel segmentation using linear spectral clustering," *CVPR*, pp. 1356–1363, 2015.

[4] J. Chen, Z. Li, and B. Huang, "Linear spectral clustering superpixel," *IEEE Trans Image Process*, vol. 26, no. 7, pp. 3317–3330, 2017.

[5] P. Morrison and J.J. Zou, "Inexact graph matching using a hierarchy of matching processes," *Computational Visual Media*, vol. 1, no. 4, pp. 291–307, 2015.

[6] D. Rueckert, L.I. Sonoda, C. Hayes, D.L.G. Hill, M.O. Leach, and D.J. Hawkes, "Nonrigid registration using free-form deformations: Application to breast MR images," *IEEE Trans Med Imag*, vol. 18, no. 8, pp. 712–721, 1999.

[7] M. Neuhaus, K. Riesen, and H. Bunke, "Fast suboptimal algorithms for the computation of graph edit distance," *S+SSPR*, pp. 163–172, 2006.

[8] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[9] A.F. Frangi, W.J. Niessen, K.L. Vincken, and M.A. Viergever, "Multiscale vessel enhancement filtering," *MICCAI*, pp. 130–137, 1998.

[10] D.G. Lowe, "Object recognition from local scale-invariant features," *ICCV*, vol. 99, no. 2, pp. 1150–1157, 1999.