Reconstructing Loopy Curvilinear Structures Using Integer Programming

Engin Türetken[1] * Fethallah Benmansour[1] Bjoern Andres[2] Hanspeter Pfister[2] Pascal Fua[1]

[1]Computer Vision Laboratory (EPFL), CH-1015 Lausanne, Switzerland
{engin.turetken,fethallah.benmansour,pascal.fua}@epfl.ch

[2]Visual Computing Group (Harvard University), MA-02138 Cambridge, US
{bandres,pfister}@seas.harvard.edu

Abstract

We propose a novel approach to automated delineation of linear structures that form complex and potentially loopy networks. This is in contrast to earlier approaches that usually assume a tree topology for the networks.

At the heart of our method is an Integer Programming formulation that allows us to find the global optimum of an objective function designed to allow cycles but penalize spurious junctions and early terminations. We demonstrate that it outperforms state-of-the-art techniques on a wide range of datasets.

1. Introduction

Networks of curvilinear structures are abundant both in natural and man made systems. They appear at all possible scales, ranging from nanometers in Electron Microscopy images of neurons and meters in aerial images of roads to petameters in dark-matter arbors binding massive galaxy clusters. As a result, their automated reconstruction has been one of the earliest topics addressed by computer vision scientists. Yet, full automation remains elusive when the image data is noisy and the structures exhibit complex morphology.

Recently, there has been renewed interest in the reconstruction of tree-like structures and significant progress has been achieved by formulating the problem as one of optimizing a global objective function [26, 25]. However, in practice, many interesting networks, such as those formed by the roads and blood vessels depicted by the first row of Fig. 1, are not trees since they contain cycles. In the latter case, they are created by capillaries connecting the arteries to the veins. Furthermore, even among those that really are trees, such as the neurites of Fig. 1, the imaging resolution is often so low that the branches appear to cross, thus introducing several spurious cycles that can only be recognized as such once the whole structure has been recovered. In fact, this is widely reported as one of the major sources of error [28, 8, 4, 29, 26, 7] and a number of heuristics have been proposed to avoid spurious connections in the presence of such cycles [26, 29, 8].

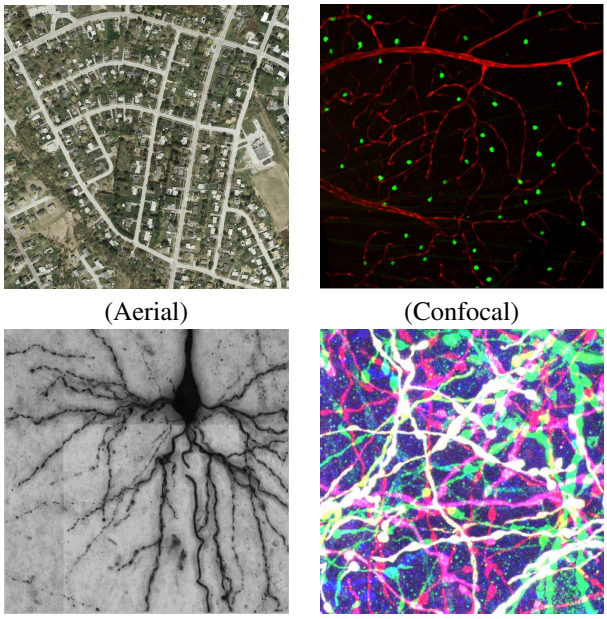
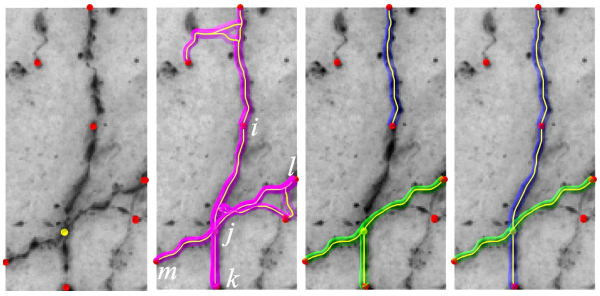


Figure 1. 2D and 3D datasets used to test our approach. (Aerial) Aerial image of roads. (Confocal) Maximum intensity projection of a confocal image stack of blood vessels, which appear in red. (Brightfield) Minimum intensity projection of a brightfield stack of neurons. (Brainbow) Maximum intensity projection of a brainbow [17] stack. As most images in this paper, they are best visualized in color.

Fig. 2 depicts a typical case of this nature. We will show that, in such cases, it is more effective to relax the tree constraint and to build loopy networks by penalizing the formation of spurious junctions and early branch terminations. More specifically, we first select evenly spaced

*This work was supported in part by the Swiss National Science Foundation.

1063-6919/13 $26.00 © 2013 IEEE
DOI 10.1109/CVPR.2013.238



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

Figure 2. Enforcing tree topology results in creation of spurious junctions. (a) Image with two crossing branches. The dots depict sample points (seeds) on the centerlines found by maximizing a tubularity measure. In 3D, these branches may be disjoint but the z-resolution is insufficient to see it and only a single sample is found at their intersection, which we color yellow. (b) The sample points are connected by geodesic paths to form a graph. (c,d) The final delineation is obtained by finding a subgraph that minimizes a global cost function. In (c), we prevent sample points from being used more than once, resulting in an erroneous delineation. In (d), we allow the yellow point to be used twice, and penalize early terminations and spurious junctions, which yields a better result.

voxels that are very likely to belong to the curvilinear structures of interest. We treat them as vertices of a graph and connect those that are within a certain distance of each other by geodesic paths [5] whose quality we assess on the basis of local image evidence. We then look for a subgraph that maximizes a global objective function that combines image-based and geometry-based terms with respect to which edges of the original graph are active.

This is similar in spirit to what is done in [25]. However, unlike in this earlier paper, we let graph vertices be used by several branches, thus allowing cycles as shown in Fig. 2(d), but introduce a regularization prior and structural constraints to limit the number of branchings and terminations. Unlike earlier graph-based delineation approaches, ours lets vertices be shared among branches while still allowing the recovery of the optimal tree from the resulting loopy subgraph when the result should be acyclical. We formulate the optimization as an Integer Program (IP), which is NP-hard in theory but for which we propose an effective formulation that delivers near-optimal solutions.

Our contribution is therefore the design of the constrained optimization problem that can be solved to optimality. We will use all four very significantly different datasets depicted by Fig. 1 to demonstrate that our approach consistently outperforms earlier ones.

In the remainder of this paper, we first briefly review related approaches. We then introduce our method, discuss its implementation, and present our results.

2. Related Work

There has recently been a resurgence of interest in automated delineation techniques [18, 10, 20, 14] because extracting curvilinear structures automatically and robustly is of fundamental relevance to many scientific disciplines. For example, it has been recognized that "the lack of powerful and effective computational tools to automatically reconstruct neuronal arbors has emerged as a major technical bottleneck in neuroscience research," as stated on the homepage of the DIADEM challenge [3]. Similar statements could be made about medical research involving the fine modeling of complex blood vessel structures, such as those in the lungs, or automated delineation of linear structures in aerial imagery databases.

Most automated approaches involve greedy strategies that start from a set of seed points, incrementally grow branches by evaluating a local tubularity measure—usually based on the Hessian and Oriented Flux matrices [23, 13, 15]—in the vicinity of the initial seeds [7, 27, 4, 2]. High tubularity paths are then iteratively added to the solution and their end points are treated as the new seeds from which the process can be restarted. Since the search typically involves processing only a fraction of the image data, these algorithms are computationally efficient . However, they are sensitive to imaging artifacts and noise since errors early in the growing process propagate, that can eventually result in large morphological mistakes.

By contrast, graph-based methods find seed points in the whole image or volume by evaluating the tubularity measure densely and finding its local maxima [12, 22, 27, 26, 25]. Although this is more computationally demanding, it can still be done efficiently in Fourier space or using GPUs [15, 16, 9]. The seed points are then connected by paths that follow local maxima of the tubularity measure. This results in a graph that forms an overcomplete representation of the underlying tree structure and the final step is to build a tree by selecting an optimal subset of the edges. This can be done by finding the Shortest Path Tree (SPT) [22], the Minimum Spanning Tree (MST) [12, 29, 27], the k-Minimum Spanning Tree (k-MST) [26], or a solution to the Minimum Arborescence Problem (MAP) [25].

Although efficient polynomial-time algorithms exist for both SPT- and MST-based formulations [22, 12, 29, 27], these approaches suffer from the fact that they must span *all* seed points, including some that might be false positives. As a result, their topology may be wrong and suboptimal post-processing procedures are required to eliminate spurious branches and, possibly, correct topological errors. The k-MST formulation [26] addresses this shortcoming by selecting an appropriate subset of points to be spanned. However, it relies on a dual cost function and a heuristic optimization algorithm [6] that does not guarantee optimality. Furthermore, all these spanning-tree approaches

evaluate the quality of an edge between two seed points by integrating a function of the tubularity measure along a connecting path. This quality measure usually fails to distinguish legitimate paths along faint curvilinear structures from those that are shortcuts between high-contrast structures. The MAP formulation [25] addresses these issues by using path classifiers to score the paths and introducing a Mixed Integer Programming approach to guaranteeing optimality of the resulting solution.

However, none of these approaches address the delineation problem for loopy structures. For the cases where spurious loops seem to be present, for example due to insufficient imaging resolution in 3D stacks or due to projections of the structures on 2D [24], some of the above-mentioned methods [26, 29, 8] attempt to distinguish spurious crossings from legitimate junctions by penalizing sharp turns and high tortuosity paths in the solutions and introducing heuristics to locally and greedily resolve the ambiguities. They do not guarantee global optimality and, as a result, easily get trapped into local minima, as reported in several of these papers. This is what our approach seeks to address by looking for the global optimum of a well-defined objective function.

3. Method

Our algorithm, like those of [12, 29, 27, 26, 25] starts by building a weighted graph designed to be an overcomplete representation for the underlying network of curvilinear structures, such as the one of Fig. 2(b). It then finds an optimal subgraph in terms of an appropriately designed objective function.

The major difference from these earlier approaches is that instead of constraining the subgraph to be a tree as in Fig. 2(c), we allow it to contain cycles, as in Fig. 2(d), and penalize spurious junctions and early branch terminations. In the Result Section, we will show that this yields improved results over very diverse datasets.

In the remainder of this section, we introduce our Integer Programming approach to finding an optimal and potentially loopy subgraph.

3.1. Formulation

Our approach to constructing over-complete graphs such as the one of Fig. 2(b) is similar to that of [25]. For each pixel or voxel, we first estimate its likelihood of being on the centerline of a curvilinear structure whose radius is within a given range, using a tubularity measure similar to those discussed in Section 2. We then find regularly spaced local maxima, which will serve as the nodes of our graph, and connect all those that are within a given distance from each other. The paths are obtained by minimizing a geodesic distance in (N+1)-D scale space—N spatial dimensions and the radius—as described in [5].

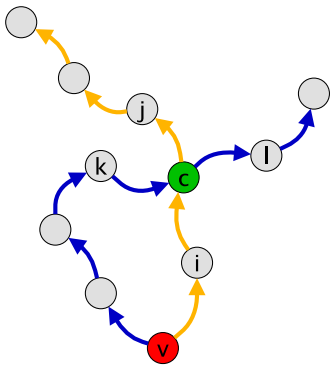


Figure 3. A loopy graph with a single root vertex v (in red). Allowing vertex c (in green) to be used by the two different branches (denoted by blue and yellow arrows) produces a loopy solution instead of a tree. Describing the crossing in terms of edge pairs $\{i, c, j\}$ and $\{k, c, l\}$ being active and all other edge pairs containing c, such as $\{k, c, j\}$, being inactive makes it possible to eventually recover the tree topology nevertheless.

This produces a graph $G = (V, E)$, whose vertices $V = \{v_i\}$ represent the seed points and pairs of oppositely directed edges $E = \{e_{ij} = (v_i, v_j), e_{ji} = (v_j, v_i)\}$ the paths linking them. Algorithms [12, 29, 27, 26, 25] that rely on this kind of formulation can all be understood as maximizing an *a posteriori* probability given image evidence and geometric priors. Most of them do so by selecting a subset of these edges that define a cycle-free subgraph. Disallowing cycles prevents vertices from being shared by separate branches, as is required for successful reconstruction cases such as the one of Fig. 2.

Conversely, allowing such crossings produces cyclic graphs such as the one shown in Fig. 3. However, in some cases, such as when delineating the neural structures of the *Brightfield* and *Brainbow* images of Fig. 1, we know that the underlying structure truly is a tree whose topology we will eventually want to recover. In the case of Fig. 3, this means that we need to be able to distinguish the one branch from the other. One approach would be to first recover the subgraph defined by the active edges and then attempt to assess its topology. However, to consistently enforce geometric constraints on branches even at junctions, we do both simultaneously by reasoning in terms of whether consecutive pairs of edges belong to the final delineation or not.

Concretely, in the case of Fig. 3, edge pairs (e_{ic}, e_{cj}) and (e_{kc}, e_{cl}) should belong but neither (e_{ic}, e_{cl}) nor (e_{kc}, e_{cj}). Similarly, consider the vertices labeled i,j,k,l, and m in the graph of Fig. 2(b). In the delineation of Fig. 2(d), edge pairs (e_{ij}, e_{jk}) and (e_{mj}, e_{jl}) are both active and vertex j belongs to both branches.

To formalize this, let $F = \{e_{ijk} = (e_{ij}, e_{jk})\}$ be the set of consecutive edge pairs in G, $\mathbf{X} = \{X_{ijk}\}$ the vector of binary random variables denoting whether edge pairs $\{e_{ijk}\}$

truly belong to the underlying curvilinear structure, and $\mathbf{x} = \{x_{ijk}\}$ the corresponding vector of indicator variables. We will say that e_{ijk} is active in the solution if $x_{ijk} = 1$. Given the graph and the image evidence I, we look for the optimal subgraph as the solution of

$$\begin{aligned}\mathbf{x}^* &= \underset{\mathbf{x}\in\mathcal{P}_c}{\operatorname{argmax}}\ P(\mathbf{X}=\mathbf{x}|I,G)\ , \\ &= \underset{\mathbf{x}\in\mathcal{P}_c}{\operatorname{argmax}}\ P(I,G|\mathbf{X}=\mathbf{x})P(\mathbf{X}=\mathbf{x})\ , \\ &= \underset{\mathbf{x}\in\mathcal{P}_c}{\operatorname{argmin}} -\log(P(I,G|\mathbf{X}=\mathbf{x})) - \log(P(\mathbf{X}=\mathbf{x})), \quad (1)\end{aligned}$$

where $\mathbf{x}$ belongs to the set $\mathcal{P}_c$ of binary vectors that define feasible subgraphs, as defined in the following section. In other words, we seek to minimize the sum of two negative log-likelihood terms, which we evaluate as follows.

As we will show in the appendix, assuming conditional independence of the image evidence given the true values of the random variables X_{ijk}, the first log likelihood term of Eq. 1 can be rewritten as

$$-\log(P(I,G|\mathbf{X}=\mathbf{x})) = \sum_{e_{ijk}\in F} w_{ijk}x_{ijk}, \quad (2)$$

where w_{ijk} is a cost term that accounts for the quality of the geodesic paths associated with the edge pair e_{ijk}. We use the path classification approach of [25] to compute it and give the details of the computation in the appendix. As discussed in Section 2, we have found it more effective at distinguishing legitimate paths from spurious ones than more standard methods, such as those that integrate a tubularity measure along the path.

The second log likelihood term in Eq. 1 is a prior term that penalizes unwarranted bifurcations or terminations. We model it as a Bayesian network with latent variables $M_{ij} = \sum_{e_{mij}\in F} X_{mij}$ and $O_{ij} = \sum_{e_{ijn}\in F} X_{ijn}$, which denote the true number of incoming and outgoing edge pairs into or out of edge e_{ij}. Furthermore, let

- $p^t = P(O_{ij} = 0|M_{ij} = 1)$ be the prior probability that a branch terminates at edge e_{ij},
- $p^c = P(O_{ij} = 1|M_{ij} = 1)$ be the prior probability that a branch continues at edge e_{ij},
- $p^b = P(O_{ij} = 2|M_{ij} = 1)$ be the prior probability that a branch bifurcates at edge e_{ij},

Assuming that the edge pairs $\{e_{ijn}\}$ are independent of the other edge pairs, given the true state the edge e_{ij}, and by inspecting the probability of each admissible event, namely termination, continuation or bifurcation at edge e_{ij}, the prior term $-\log(P(\mathbf{X}=\mathbf{x}))$ can be rewritten as

$$-\sum_{e_{ij}\in E}\left[\sum_{e_{mi}\in E}\log(p^t)x_{mij} + \sum_{e_{jn}\in E}\log\left(\frac{p^c}{p^t}\right)x_{ijn} + \sum_{e_{jn}\in E}\sum_{\substack{e_{jk}\in E\\ k<n}}\log\left(\frac{p^b p^t}{(p^c)^2}\right)x_{ijn}x_{ijk}\right], \quad (3)$$

which we derive in the appendix. In short, minimizing the negative log likelihood of Eq. 1 amounts to minimizing, with respect to the indicator variables $\mathbf{x}$, the criterion

$$\sum_{e_{ijk}\in F} a_{ijk}x_{ijk} + \sum_{e_{ijk},e_{ijn}\in F} b_{ijkn}x_{ijk}x_{ijn}\ , \quad (4)$$

which is the sum of the linear and quadratic terms of Eqs. 2 and 3 and whose a_{ijk} and b_{ijkn} coefficients are obtained by summing the respective terms.

However, not all choices of binary values for the indicator variables give rise to a connected subgraph that represents a plausible delineation. The above minimization must therefore be carried out subject to a set of constraints that we introduce next.

3.2. Constraints

Our images may contain several disconnected structures. To avoid having to process them sequentially in a greedy manner, which may result in some branches being "stolen" by the first structure we reconstruct and therefore a suboptimal solution, we connect them all. Assuming we are given a set R of seed vertices, one for each structure of interest, we create a virtual root vertex v_v and connect it to each $v_r \in R$ by zero cost edge pairs containing all other vertices to which v_r is connected.

We now define four sets of constraints to ensure that the solutions to the minimization problem of Eq. 4 are such that seed vertices are not isolated, branches are edge-disjoint, potential crossovers are consistently handled, and all active edge pairs are connected.

Non-isolated Seeds: We require the seed vertices $v_r \in R$ to be connected to at least one vertex other than the virtual root v_v and to have no incoming edge other than e_{vr}. We write this as

$$\begin{aligned}&\sum_{e_{ri}\in E} x_{vri} \geq 1,\ \ \forall v_r \in R\ , \quad (5)\\ &\sum_{e_{ijr}\in F} x_{ijr} + \sum_{e_{irj}\in F: v_i\neq v_v} x_{irj} = 0,\ \ \forall v_r \in R\ .\end{aligned}$$

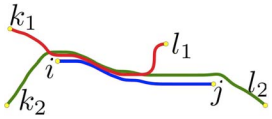
Disjoint Edges: For each edge $e_{ij} \in E$, we let at most one edge pair be active among all those that either contain e_{ij} or sufficiently overlap with it. We do the first by preventing the number of active incoming edge pairs into an edge to be more than one. Second, we treat edges that overlap more than a certain fraction of their radius as being the same edge for the purpose of this constraint. Let t_{ij} denote

the geodesic path corresponding to edge e_{ij}. We write

$$\sum_{e_{kl}\in C(e_{ij})} \sum_{\substack{e_{mk}\in E:\\ v_m\neq v_l}} x_{mkl} \leq 1, \quad \forall e_{ij}\in E : v_i \neq v_v,$$

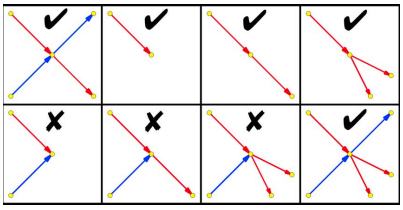
$$C(e_{ij}) = \left\{ e_{kl}\in E \mid \begin{smallmatrix}(t_{kl}\subset t_{ij})\vee\\ (l(t_{ij}\cap t_{kl})>\alpha\bar{r}(t_{ij}\cap t_{kl})\wedge\\ l(t_{ij})<l(t_{kl}))\end{smallmatrix} \right\}, \tag{6}$$

where $l(.)$ and $\bar{r}(.)$ denote the length and mean radius of a path respectively. α is a constant value that determines the allowed extent of the overlap between the geodesic paths of the edges. It is set to 5 in all our experiments. In the example depicted by the figure above, among all the edge pairs incoming to the edges e_{ij}, $e_{k_1l_1}$ and $e_{k_2l_2}$, only one can be active in the final solution.



For those curvilinear structures that are inherently trees, these constraints make their recovery from the resulting subgraph possible by starting from the terminal vertices and following the active edge pairs along the paths that lead to the root vertices.

Crossover Consistency: A potential crossover in G is a vertex, which is adjacent to at least four other vertices and whose in- and out-degrees are greater than one. A consistent solution containing such a vertex v_p is then defined as the one, in which branches do not terminate at v_p if its in-degree in the solution is greater than one. The figure to the left illustrates consistent configurations denoted by a swoosh and inconsistent ones denoted by a cross. We express this as



$$\sum_{\substack{e_{ki}\in E:\\ v_k\neq v_j}} x_{kij} + \sum_{\substack{e_{lm}\in E:\\ v_l\neq v_q, v_l\neq v_j}} x_{lmq} - \sum_{\substack{e_{ju}\in E:\\ v_u\neq v_i}} x_{iju} \leq 1 \ , \tag{7}$$

$$\forall e_{ij}\in E \quad \forall e_{mq}\in C(v_j) : v_m \neq v_i$$

$$C(v_j) = \{e_{nk}\in E \mid v_k = v_j \vee (v_j \in t_{nk}, v_n \neq v_j, v_k \neq v_j)\}$$

These constraints are only active when dealing with structures that inherently are trees, such as the neural structures of Fig. 1. For inherently loopy ones, such as the roads and blood vessels, we deactivate them to allow creation of junctions that are parts of legitimate cycles.

Connectedness: We require all the active edge pairs to be connected to the virtual root v_v. An edge pair e_{ijk} is said to be connected if there exists a path in G, starting at v_v and containing e_{ijk}, along which all the edge pairs are active.

Let y^l_{ij} $(i \neq l)$ be a non-negative continuous flow variable that denotes the number of distinct directed paths in the solution, from the virtual root v_v to vertex v_l, that traverse the edge e_{ij}. This gives rise to the following constraint set

$$\sum_{e_{vj}\in E} y^l_{vj} = \sum_{e_{il}\in E} y^l_{il}, \quad \forall v_l \in V\setminus\{v_v\} \ , \tag{8}$$

$$\sum_{e_{vj}\in E} y^l_{vj} \leq deg^-(v_l), \quad \forall v_l \in V\setminus\{v_v\} \ , \tag{9}$$

$$\sum_{e_{ij}\in E} y^l_{ij} = \sum_{e_{jk}\in E} y^l_{jk}, \quad \forall v_j, v_l \in V\setminus\{v_v\} : v_j \neq v_l \ , \tag{10}$$

$$y^l_{il} \geq x_{ilk}, \quad \forall e_{ilk}\in F \ , \tag{11}$$

$$y^l_{il} = \sum_{e_{ki}\in E} x_{kil}, \quad \forall e_{il}\in E : v_i \neq v_v \ , \tag{12}$$

$$y^l_{ij} \leq deg^-(v_l) \sum_{e_{ki}\in E} x_{kij}, \quad \begin{smallmatrix}\forall e_{ij}\in E: v_i\neq v_v,\\ \forall v_l\in V\setminus\{v_v,v_i,v_j\}\end{smallmatrix} \ , \tag{13}$$

where $deg^-(.)$ is the in degree of a vertex. The first two constraints guarantee that the amount of flow outgoing from virtual vertex v_v to true vertex v_l is equal to the incoming flow to v_l, which must be smaller than the in degree of v_l. The following constraint imposes conservation of flow at intermediate vertices. Finally, the last three constraints bind the flow variables to the binary ones, ensuring that a connected network formed by the non-zero flow variables is also connected in the active edge pair variables. Note that, since we are looking for possibly cyclic subgraphs, there can be multiple paths incoming to a vertex. Hence, unlike the flow variables of [11] that are bounded by one, the ones defined here have no upper bounds.

3.3. Optimization

Minimizing the objective function of Eq 4 subject to the constraints described above is NP-Hard. Nevertheless, its solution can be closely approximated using the branch and cut algorithm implemented in a publicly available library [1]. We produced all the results described in the result section by running this code. The optimization algorithm always converged to the global optimum using a very small solution gap ($1e^{-7}$).

4. Results

In this section, we first describe briefly the four datasets of Fig. 1, which we used to validate our approach. We then present our results on the first two, which contain cyclic networks, and finally on the next two, which contain true trees but whose optical resolution is so poor that they look like cyclic graphs. We show that our approach outperforms the state-of-the-art in both cases.

4.1. Datasets and Path Classification

We evaluated our approach on the four different datasets depicted by Fig. 1 and described in more detail below:

- *Aerial*: Aerial images of loopy road networks. We used 21 grayscale versions of these images for training and 8 for testing.
- *Confocal*: Two image stacks of direction selective retinal ganglion cells were acquired with a confocal microscope. We used a portion of one to train our path classifier and both for testing. We only considered the red channel of these stacks since it is the only one used to label the blood vessels.
- *Brightfield*: Six image stacks were acquired by brightfield microscopy from biocytin-stained rat brains. We used three for training and three for testing.
- *Brainbow*: Neurites were visualized by targeting mice primary visual cortex using the brainbow technique [17] so that each neuronal structure has a distinct color. We used one image stack for training and three for testing.

Many roads of the *Aerial* dataset are partially occluded by trees while the images from the other datasets are very noisy, making the delineation task challenging in all cases. Fig. 4 depicts some of our results on these datasets and we provide additional ones as supplementary material.

To obtain these results, we had to estimate the cost terms w_{ijk} of Eq 2. To this end, we used the path classifier of [25], which operates on histograms of gradient deviation features and was designed for grayscale images. To adapt it to the *Brainbow* color images, we first converted the stacks into the CIELAB space and clustered their voxels using the K-Means algorithm. For each cluster, we then computed a normalized gray scale image whose voxel values are inversely proportional to the color distance between the original voxel and the cluster mean. This results in K gray scale images (K is set to 50 in all our experiments) and each voxel is associated to the one its cluster corresponds to. For a pixel along any given path, the gradient features are computed on this associated image. The result is that only gradients corresponding to pixels with a similar color are taken into account.

4.2. Roads and Blood Vessels

The roads and blood vessels of the *Aerial* and *Confocal* datasets form graphs in which there are many real cycles. In the case of the blood vessels, this is because there are capillaries that connect the arteries to the veins and irrigate the cells along the way.

As can be seen in the first row of Fig. 4, the road networks are recovered almost perfectly in spite of the occlusions. The only errors are driveways that are treated as very short roads and a few roads dead-ending because the connecting path to the closest junction is severely occluded. The first one could be addressed by introducing a semantic threshold on short overhanging segments while the latter would require a much more sophisticated semantic understanding. We supply results on the four remaining test images as supplementary material.

The quality of the blood vessel delineations depicted by the second row is much harder to assess on the printed page but becomes clear when looking at the rotating volumes that we supply as supplementary material.

	BRBW1	BRBW2	BRBW3	BRF1	BRF2	BRF3
HGD-QMIP [25]	0.3692	0.5118	0.4016	0.6114	0.4263	0.6551
L-QMIP	**0.8327**	**0.6897**	**0.7848**	**0.7282**	**0.5122**	**0.7391**

Table 1. DIADEM [3] scores for our results (L-QMIP) and those of [25] (HGD-QMIP) for the delineations of 3 *Brainbow* stacks and the 3 *Brightfield* ones. They are denoted by BRBWi and BRFi, respectively. Our scores are higher and therefore better.

	BRF1				BRF2				BRF3			
NARAY [21]	0.56	0.65	0.88	0.99	0.64	0.64	0.91	0.99	0.55	0.61	0.87	0.99
L-QMIP	**0.14**	**0.33**	**0.86**	**0.76**	**0.18**	**0.32**	**0.89**	**0.88**	**0.14**	**0.32**	**0.80**	**0.67**

Table 2. NetMets [19] scores for our results (L-QMIP) and those obtained with the code made publicly available by the DIADEM challenge winners [21] (NARAY). The NetMets software outputs four numbers for each trial, geometric False Positive Rate, geometric False Negative Rate, connectivity False Positive Rate, and connectivity False Negative Rate. We list them here in that order and ours are lower and therefore better.

4.3. Neural Structures

The neurites of the *Brightfield* and *Brainbow* datasets form tree structures without cycles. However, because of the low z-resolution, branches that really are disjoint appear to cross.

Because the ground truth tracings are trees, we were able to compute the DIADEM scores [3] for the four delineations depicted by the bottom two rows of Fig. 4. They are listed in Table 1 along with those results obtained by solving the Mixed Integer Program advocated in [25], which is not designed to prevent cycles. Since we use the same algorithm to assess the quality of the paths in both cases, the main difference between the two approaches is that ours allows vertices to be used more than once and favors creation of cycles through the additional cost terms and constraints, while the other does not, which results in a substantial performance gain.

We also evaluated the curvelet transform based algorithm of [21] on the *Brightfield* dataset. We used the publicly available code by the winners of the DIADEM challenge. Since the code does not allow the user to provide a set of root nodes, the DIADEM score of its output cannot be computed. However, the same group recently made available a

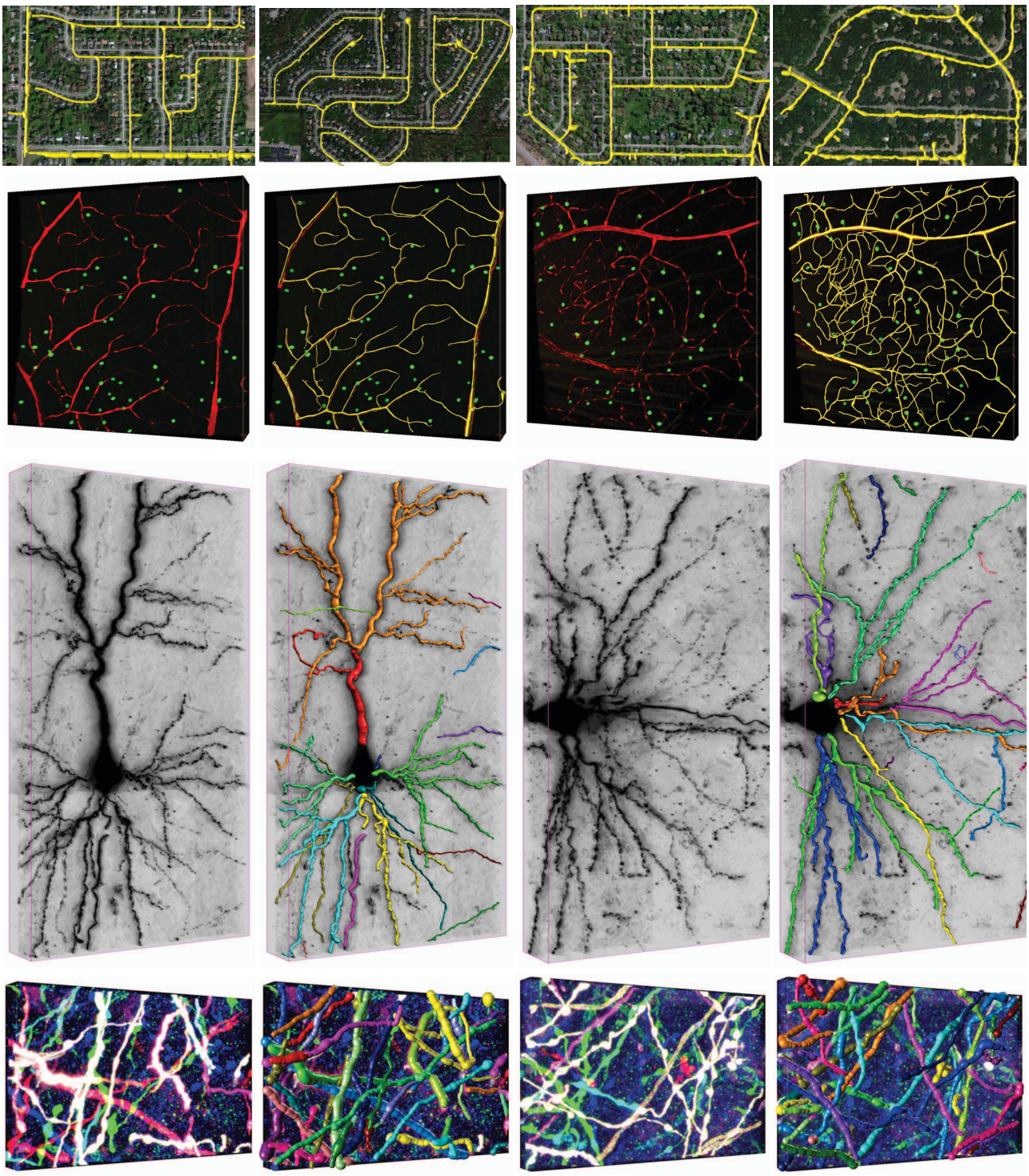


Figure 4. Delineation results, best viewed in color. **Top Row:** Four road images with final delineations shifted and overlaid to allow comparisons. **Bottom Rows:** For each dataset, two minimal or maximal projections and overlaid delineation results. Each connected curvilinear structure network is shown in a distinct color.

software package to evaluate reconstructions based on the NetMets [19] measure. Similar to the DIADEM metric, this measure takes as input the reconstruction, the corresponding ground truth tracings and a sensitivity parameter

σ, which is set to twice the minimum image spacing in all our experiments.

We evaluated this measure on both their result and ours and report the outcome in Table 2, which shows that our approach brings about a very significant improvement.

5. Conclusion

We have presented a graph-based approach to delineating complex linear structures in 2D images and 3D image stacks. Unlike most earlier ones, it explicitly handles the fact that they may be cyclic and builds graphs in which vertices may belong to more than one branch. This results in a substantial performance increase.

However the geometric constraints we impose are still relatively local since they bear on consecutive edge pairs. In future work, we will focus on imposing more global ones.

References

[1] Gurobi Optimizer. http://www.gurobi.com/. 5

[2] K. Al-Kofahi, S. Lasek, D. Szarowski, C. Pace, G. Nagy, J. Turner, and B. Roysam. Rapid Automated Three-Dimensional Tracing of Neurons from Confocal Image Stacks. TITB, 6(2):171–187, 2002. 2

[3] G. A. Ascoli, K. Svoboda, and Y. Liu. Digital Reconstruction of Axonal and Dendritic Morphology Diadem Challenge, 2010. http://diademchallenge.org/. 2, 6

[4] E. Bas and D. Erdogmus. Principal Curves as Skeletons of Tubular Objects - Locally Characterizing the Structures of Axons. Neuroinformatics, 9(2-3):181–191, 2011. 1, 2

[5] F. Benmansour and L. Cohen. Tubular Structure Segmentation Based on Minimal Path Method and Anisotropic Enhancement. IJCV, 92(2):192–210, 2011. 2, 3

[6] C. Blum and M. Blesa. Combining Ant Colony Optimization with Dynamic Programming for Solving the K-Cardinality Tree Problem. In Computational Intelligence and Bioinspired Systems, pages 25–33, 2005. 2

[7] A. Choromanska, S. Chang, and R. Yuste. Automatic Reconstruction of Neural Morphologies with Multi-Scale Graph-Based Tracking. Frontiers in Neural Circuits, 6(25), 2012. 1, 2

[8] P. Chothani, V. Mehta, and A. Stepanyants. Automated Tracing of Neurites from Light Microscopy Stacks of Images. Neuroinformatics, 9:263–278, 2011. 1, 3

[9] L. Domanski, C. Sun, R. Hassan, P. Vallotton, and D. Wang. Linear Feature Detection on Gpus. 2010. 2

[10] D. Donohue and G. Ascoli. Automated Reconstruction of Neuronal Morphology: An Overview. Brain Research Reviews, 67:94–102, 2011. 2

[11] C. Duhamel, L. Gouveia, P. Moura, and M. Souza. Models and Heuristics for a Minimum Arborescence Problem. Networks, 51(1):34–47, 2008. 5

[12] M. Fischler, J. Tenenbaum, and H. Wolf. Detection of Roads and Linear Structures in Low-Resolution Aerial Imagery Using a Multisource Knowledge Integration Technique. CVIP, 15(3):201–223, March 1981. 2, 3

[13] A. Frangi, W. Niessen, K. Vincken, and M. Viergever. Multiscale Vessel Enhancement Filtering. Lecture Notes in Computer Science, 1496:130–137, 1998. 2

[14] C. Kirbas and F. Quek. Vessel Extraction Techniques and Algorithms: A Survey. In Symposium on BioInformatics and BioEngineering, pages 238–245, 2003. 2

[15] M. Law and A. Chung. Three Dimensional Curvilinear Structure Detection Using Optimally Oriented Flux. In ECCV, 2008. 2

[16] M. Law and A. Chung. An Oriented Flux Symmetry Based Active Contour Model for Three Dimensional Vessel Segmentation. In ECCV, pages 720–734, 2010. 2

[17] J. Livet, T. Weissman, H. Kang, R. Draft, J. Lu, R. Bennis, J. Sanes, and J. Lichtman. Transgenic strategies for combinatorial expression of fluorescent proteins in the nervous system. Nature, 450(7166):56–62, 2007. 1, 6

[18] J. Lu. Neuronal Tracing for Connectomic Studies. Neuroinformatics, 9(2-3):159–166, 2011. 2

[19] D. Mayerich, C. Bjornsson, J. Taylor, and B. Roysam. Netmets: Software for Quantifying and Visualizing Errors in Biological Network Segmentation. BMC Bioinformatics, 13, 2012. 6, 7

[20] E. Meijering. Neuron Tracing in Perspective. Cytometry Part A, 77(7):693–704, 2010. 2

[21] A. Narayanaswamy, Y. Wang, and B. Roysam. 3-d image pre-processing algorithms for improved automated tracing of neuronal arbors. Neuroinformatics, 9(2-3):219–231, 2011. 6

[22] H. Peng, F. Long, and G. Myers. Automatic 3D Neuron Tracing Using All-Path Pruning. Bioinformatics, 27(13):239–247, 2011. 2

[23] Y. Sato, S. Nakajima, H. Atsumi, T. Koller, G. Gerig, S. Yoshida, and R. Kikinis. 3D Multi-Scale Line Filter for Segmentation and Visualization of Curvilinear Structures in Medical Images. MIA, 2:143–168, June 1998. 2

[24] J. Staal, M. Abramoff, M. Niemeijer, M. Viergever, and B. van Ginneken. Ridge Based Vessel Segmentation in Color Images of the Retina. TMI, 2004. 3

[25] E. Turetken, F. Benmansour, and P. Fua. Automated Reconstruction of Tree Structures Using Path Classifiers and Mixed Integer Programming. In CVPR, June 2012. 1, 2, 3, 4, 6

[26] E. Turetken, G. Gonzalez, C. Blum, and P. Fua. Automated Reconstruction of Dendritic and Axonal Trees by Global Optimization with Geometric Priors. Neuroinformatics, 9(2-3):279–302, 2011. 1, 2, 3

[27] Y. Wang, A. Narayanaswamy, and B. Roysam. Novel 4D Open-Curve Active Contour and Curve Completion Approach for Automated Tree Structure Extraction. In CVPR, pages 1105–1112, 2011. 2, 3

[28] Y. Wang, A. Narayanaswamy, C. Tsai, and B. Roysam. A Broadly Applicable 3D Neuron Tracing Method Based on Open-Curve Snake. Neuroinformatics, 9(2-3):193–217, 2011. 1

[29] T. Zhao, J. Xie, F. Amat, N. Clack, P. Ahammad, H. Peng, F. Long, and E. Myers. Automated Reconstruction of Neuronal Morphology Based on Local Geometrical and Global Structural Models. Neuroinformatics, 9:247–261, May 2011. 1, 2, 3