Vessel detection

Jan Kybic

2020

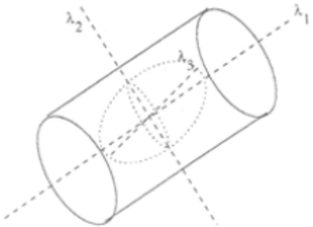
Frangi et al: Multiscale Vesel Enhancement Filtering. MICCAI 1998

Aim

- Enhancement and detection of vessels (tubular structures)
- Calculate *vesselness* (pointwise)

Quadratic approximation

$$L(\mathbf{x}_o + \delta\mathbf{x}_o, s) \approx L(\mathbf{x}_o, s) + \delta\mathbf{x}_o^T \nabla_{o,s} + \delta\mathbf{x}_o^T \mathcal{H}_{o,s} \delta\mathbf{x}_o$$

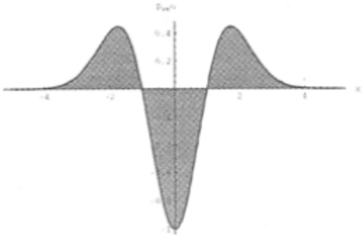


Gaussian derivatives

$$\frac{\partial}{\partial x} L(\mathbf{x}, s) = s^{\gamma} L(\mathbf{x}) * \frac{\partial}{\partial x} G(\mathbf{x}, s)$$

where the D-dimensional Gaussian is defined as:

$$G(\mathbf{x}, s) = \frac{1}{\sqrt{(2\pi s^2)}^D} \; e^{-\frac{\|\mathbf{x}\|^2}{2s^2}}$$



$$\delta\mathbf{x}_o^T \mathcal{H}_{o,s} \delta\mathbf{x}_o = (\frac{\partial}{\partial \delta\mathbf{x}_o})(\frac{\partial}{\partial \delta\mathbf{x}_o}) L(\mathbf{x}_o, s)$$

Eigenvalues

$$\mathcal{H}_{o,s}\hat{\mathbf{u}}_{s,k} = \lambda_{s,k}\hat{\mathbf{u}}_{s,k}$$

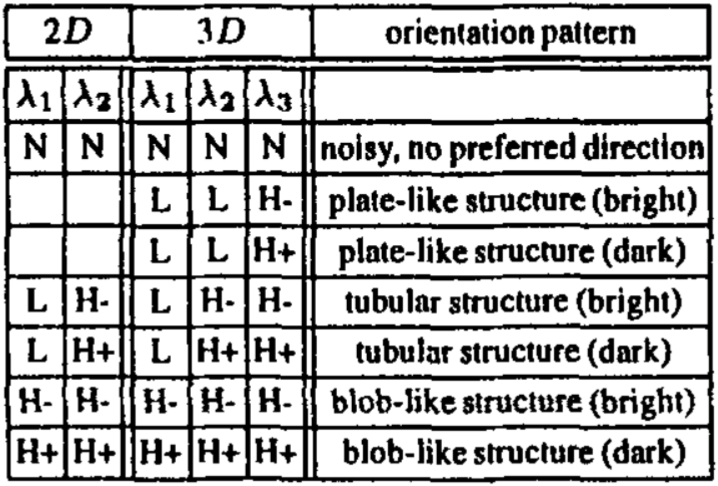
$$(|\lambda_1| \leq |\lambda_2| \leq |\lambda_3|).$$

$$\hat{\mathbf{u}}_{s,k}^{T}\mathcal{H}_{o,s}\hat{\mathbf{u}}_{s,k} = \lambda_{s,k}$$

To summarize, for an ideal tubular structure in a $3D$ image:

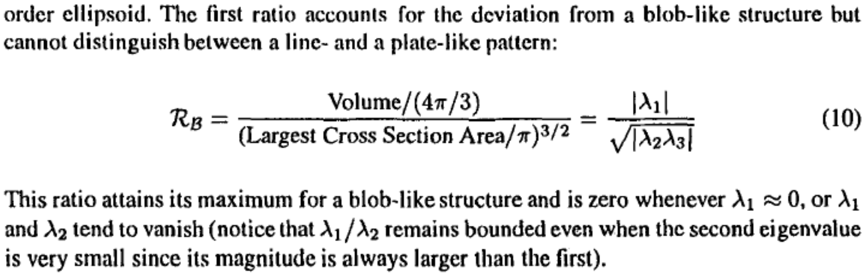
$$|\lambda_1| \approx 0$$
$$|\lambda_1| \ll |\lambda_2|$$
$$\lambda_2 \approx \lambda_3$$

Patterns



2D		3D			orientation pattern
λ_1	λ_2	λ_1	λ_2	λ_3	
N	N	N	N	N	noisy, no preferred direction
		L	L	H-	plate-like structure (bright)
		L	L	H+	plate-like structure (dark)
L	H-	L	H-	H-	tubular structure (bright)
L	H+	L	H+	H+	tubular structure (dark)
H-	H-	H-	H-	H-	blob-like structure (bright)
H+	H+	H+	H+	H+	blob-like structure (dark)

Elongation

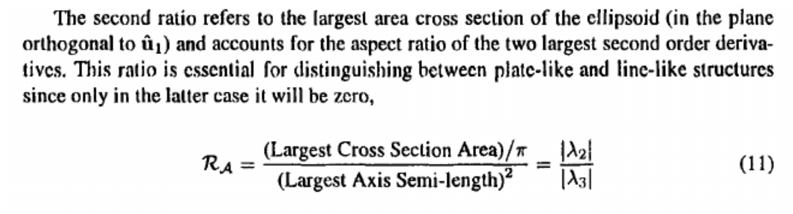


order ellipsoid. The first ratio accounts for the deviation from a blob-like structure but cannot distinguish between a line- and a plate-like pattern:

$$\mathcal{R}_{\mathcal{B}} = \frac{\text{Volume}/(4\pi/3)}{(\text{Largest Cross Section Area}/\pi)^{3/2}} = \frac{|\lambda_1|}{\sqrt{|\lambda_2 \lambda_3|}} \tag{10}$$

This ratio attains its maximum for a blob-like structure and is zero whenever $\lambda_1 \approx 0$, or λ_1 and λ_2 tend to vanish (notice that λ_1/λ_2 remains bounded even when the second eigenvalue is very small since its magnitude is always larger than the first).

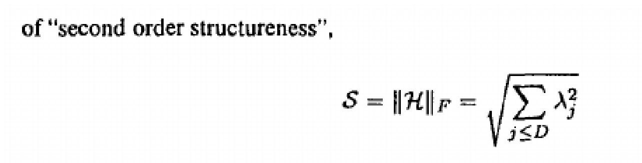
Aspect ratio



The second ratio refers to the largest area cross section of the ellipsoid (in the plane orthogonal to $\hat{\mathbf{u}}_1$) and accounts for the aspect ratio of the two largest second order derivatives. This ratio is essential for distinguishing between plate-like and line-like structures since only in the latter case it will be zero,

$$\mathcal{R}_{\mathcal{A}} = \frac{(\text{Largest Cross Section Area})/\pi}{(\text{Largest Axis Semi-length})^2} = \frac{|\lambda_2|}{|\lambda_3|} \qquad (11)$$

“Structureness”



of “second order structureness”,

$$\mathcal{S} = \|\mathcal{H}\|_F = \sqrt{\sum_{j \leq D} \lambda_j^2}$$

low for no structure (background)

Vesselness in 3D

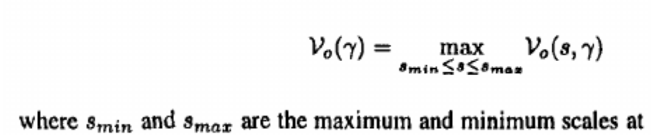
$$\mathcal{V}_o(s) = \begin{cases} 0 & \text{if } \lambda_2 > 0 \text{ or } \lambda_3 > 0, \\ \left(1 - \exp\left(-\frac{\mathcal{R}_{\mathcal{A}}{}^2}{2\alpha^2}\right)\right) \exp\left(-\frac{\mathcal{R}_{\mathcal{B}}{}^2}{2\beta^2}\right) \left(1 - \exp\left(-\frac{\mathcal{S}^2}{2c^2}\right)\right) & \end{cases} \tag{13}$$

$\alpha = 0.5,\ \beta = 0.5,\ c = \frac{1}{2} \max \|H\|_F$

Vesselness in 2D

$$\mathcal{V}_o(s) = \begin{cases} 0 & \text{if } \lambda_2 > 0, \\ \exp\left(-\frac{\mathcal{R}_{\mathcal{B}}{}^2}{2\beta^2}\right)\left(1 - \exp\left(-\frac{\mathcal{S}^2}{2c^2}\right)\right) & \end{cases}$$

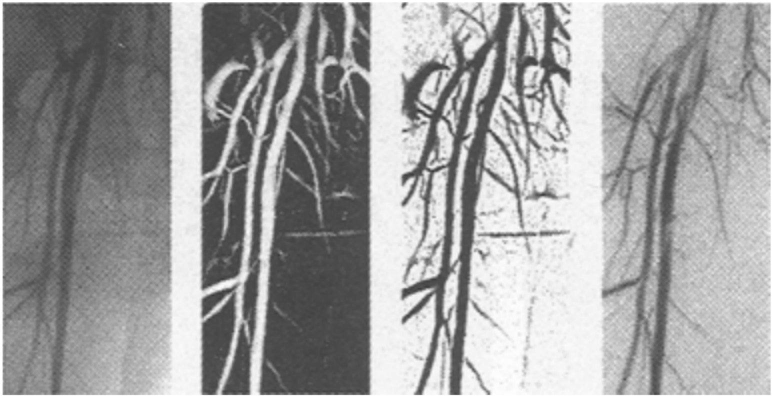
Multiscale



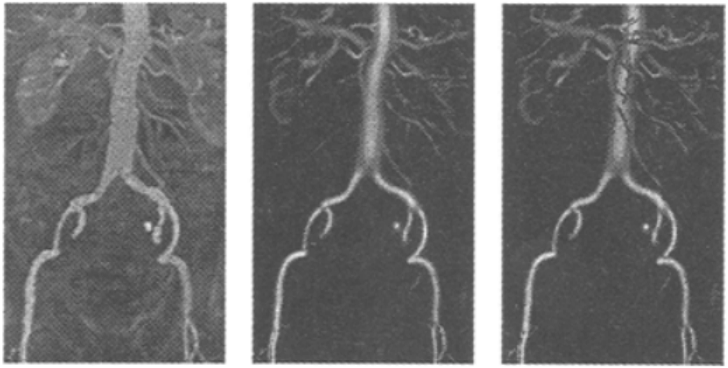
$$\mathcal{V}_o(\gamma) = \max_{s_{min} \leq s \leq s_{max}} \mathcal{V}_o(s,\gamma)$$

where s_{min} and s_{max} are the maximum and minimum scales at

Results X-rays


contrast X-ray, vesselness, inverted, contrast-reference

3D Gd/DTPA MRA



original, vesselness, closest vessel projection

Scale selection

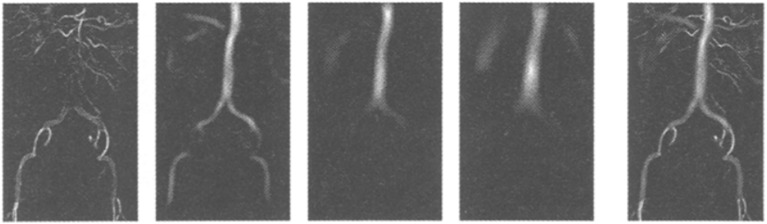
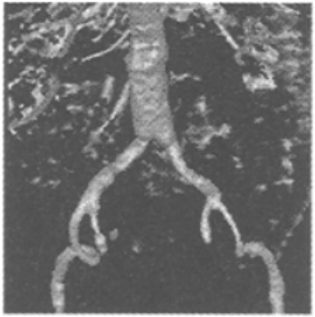
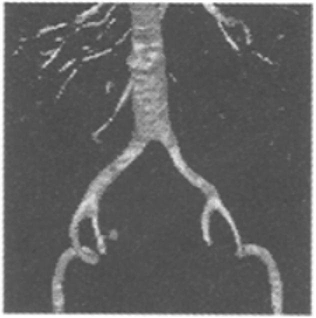


Fig. 4. The first four images show the vesselness obtained at increasing scales. The last image is the result after the scale selection procedure.

Volume rendering





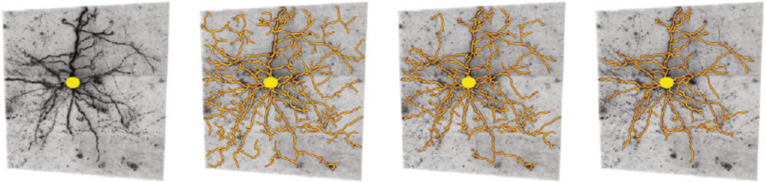
volume rendering - direct, based on vesselness

Türetken et al: Automated Reconstruction of Dendritic and Axonal Trees by Global Optimization with Geometric Priors. Neuroinformatics 2011

Aim and key techniques

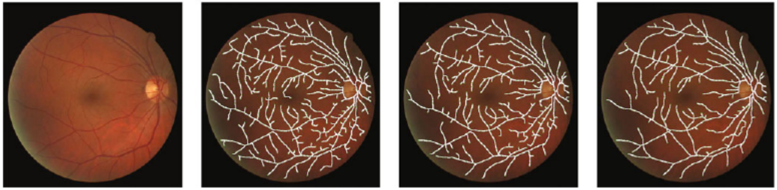
- Detect and segment tree (linear) structures (dendritic, vascular, bronchial...)
- Machine learning for filament detection
- k-MST to find connections (edges)

Neuron example



original, MSTs, optimum k, regularized reconstruction

Retina example

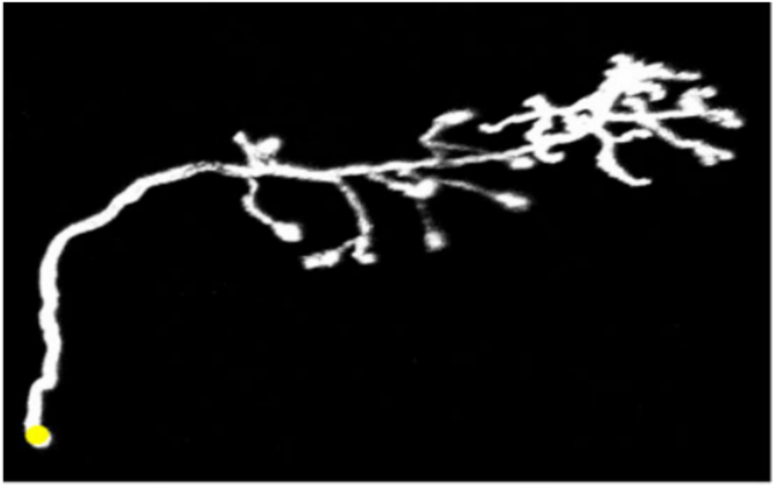

original, MSTs, optimum k, regularized reconstruction

Method overview

- Compute tubularity
- Find anchor (seed) points
- Compute paths between anchor points
- Compute *k*-MST - span *k* edges
- Select the best tree

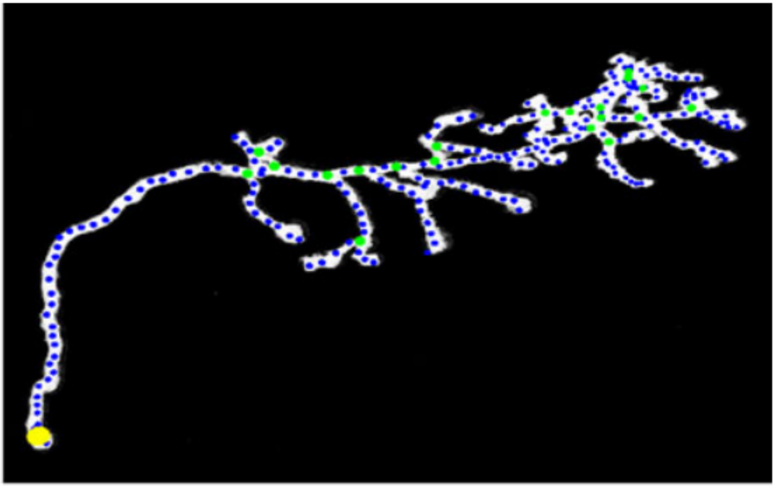
Example

Original image - olfactory fibers



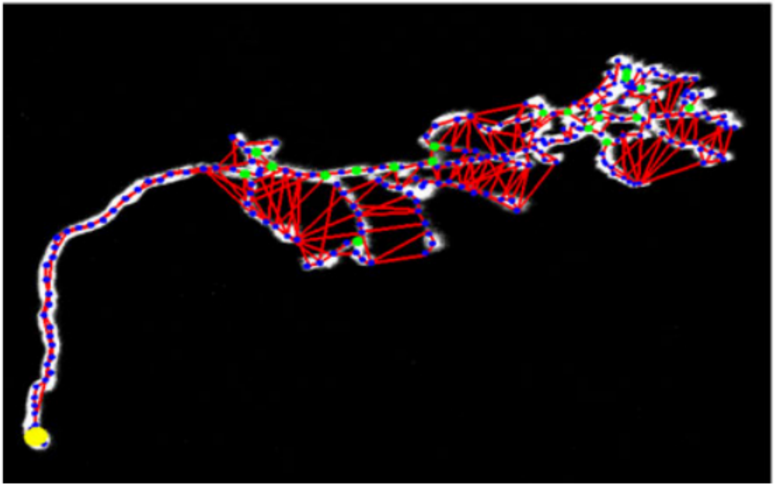
Example (2)

Seed points



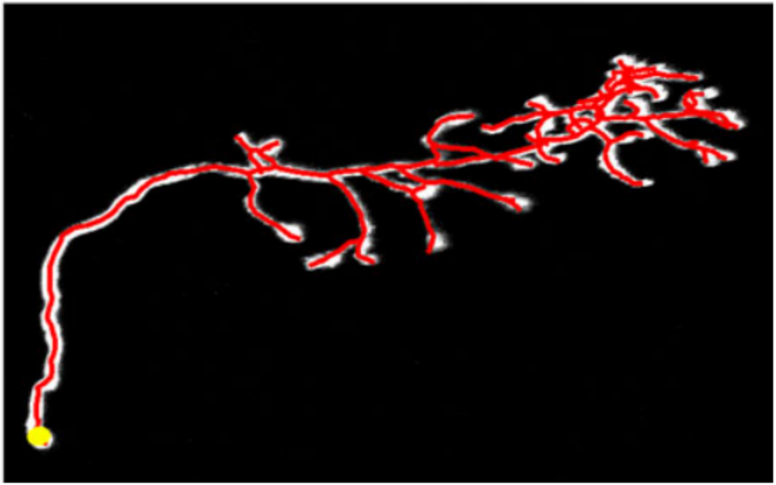
Example (3)

Neighborhood graph



Example (4)

Final tree



Tubularity measure

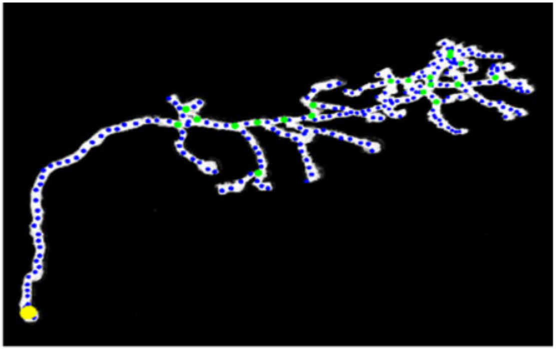
- features = steerable filter responses (e.g. Gabor), Hessian eigenvalues, different scales and orientations
- SVM classifier trained on expert labeled data.
  - trained on randomly sampled background locations
- tubularity

$$f_i = \max_{w,\phi} f(x_i, w, \phi)$$

$$p_i = \frac{1}{1 + e^{-(af_i+b)}}$$

Anchor points

- Threshold tubularity at $p = 0.5$
- Calculate skeleton
- Detect potential junction points $\rightarrow$double point
- Greedily assign remaining points



Linking anchor points

- Connect points to their neighbors (within distance)
- Dijkstra (minimum path using 26 pixel neighborhood) to minimize

$$d_{mn} \approx \int -\log p(s) ds$$

$$d_{mn} = \sum_{e_{ij}^I \in e_{mn}} -\log p_{ij} , \qquad \approx \sum_{e_{ij}^I \in e_{mn}} \int_0^{l_{ij}} -\log p \left(\frac{l_{ij} - s}{l_{ij}} x_i + \frac{s}{l_{ij}} x_j \right) ds$$

In practice, to avoid divisions by zero, we therefore take p_{ij} to be equal to $p_i^{l_{ij}}$ if $|p_i - p_j| \leq \epsilon$, and so that

$$\log(p_{ij}) = l_{ij} \frac{p_i(\log(p_i) - 1) + p_j(1 - \log(p_j))}{p_i - p_j} , \qquad (6)$$

otherwise. Note that this is consistent because when $p_j - p_i$ tends towards zero, $\log(p_{ij})$ defined in this manner tends towards $l_{ij} \log(p_i) = l_{ij} \log(p_j)$.

Optimal tree - image term

$$
\begin{aligned}
P(\mathbf{f}|\mathbf{T}=\mathbf{t}) &= \prod_{e_{mn}\in E} P(\mathbf{f}_{mn}|T_{mn}=t_{mn}) & (11)\\
&= \prod_{e_{mn}\in E} P(\mathbf{f}_{mn}|T_{mn}=1)^{t_{mn}} \\
&\quad \times P(\mathbf{f}_{mn}|T_{mn}=0)^{(1-t_{mn})} & (12)\\
&\propto \prod_{e_{mn}\in E} \left[\frac{P(\mathbf{f}_{mn}|T_{mn}=1)}{P(\mathbf{f}_{mn}|T_{mn}=0)}\right]^{t_{mn}} & (13)
\end{aligned}
$$

$$
= \prod_{e_{mn}\in E} \left[\prod_{e^{I}_{ij}\in e_{mn}} \frac{p_{ij}}{1-p_{ij}}\right]^{t_{mn}}, \qquad (17)
$$

...

Optimal tree - image term (2)

$$F_i(\mathbf{t}) = -\log P(\mathbf{f}|\mathbf{T}=\mathbf{t}) = \sum_{e_{mn}\in E} c_{mn}\, t_{mn}, \qquad (18)$$

$$\text{where}\;\; c_{mn} = \sum_{e^I_{ij}\in e_{mn}} -\log \frac{p_{ij}}{1-p_{ij}}.$$

Optimal tree - geometric term

- Graph must be a tree
- Consistent widths
- Consistent orientations

$$-\log P(\mathbf{T}=\mathbf{t}|\mathbf{\Phi},\mathbf{\Theta}) = \sum_{e_{ri}\in E_r} b_{ri}\, t_{ri} + \sum_{\substack{e_{om}\in E \\ e_{mn}\in E\setminus E_r}} a_{omn} t_{mn} t_{om} \ , \tag{19}$$

where

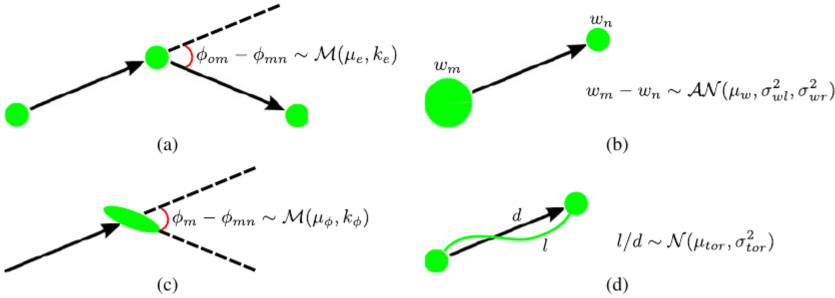
$$b_{ri} = -\log P(T_{ri}=1|\mathbf{\Phi}_r, \mathbf{\Phi}_i, \mathbf{\Theta}) \ , \tag{20}$$

$$a_{omn} = -\log P(T_{mn}=1|T_{om}=1, \mathbf{\Phi}_{omn}, \mathbf{\Theta}).$$

where Φ are width and orientation estimates

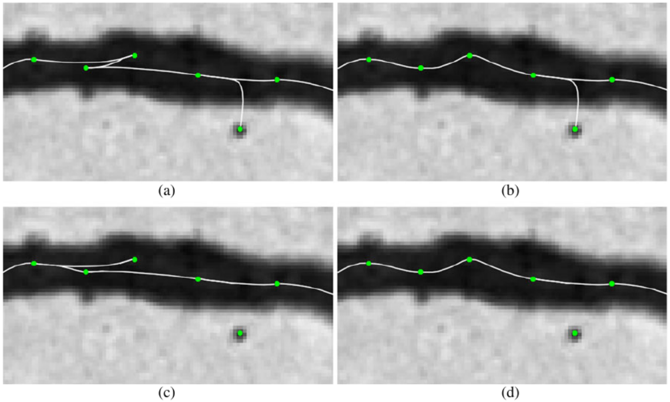
Geometric term modeling

- edge direction similarity (rate of turn) - von Mises distribution (circular normal)
- width consistency - asymetric Gaussian
- orientation consistency - von Mises
- tortuosity - l/d, Gaussian


$\phi_{om} - \phi_{mn} \sim \mathcal{M}(\mu_e, k_e)$
(a)
w_n
w_m
$w_m - w_n \sim \mathcal{AN}(\mu_w, \sigma^2_{wl}, \sigma^2_{wr})$
(b)
$\phi_m - \phi_{mn} \sim \mathcal{M}(\mu_\phi, k_\phi)$
(c)
d
l
$l/d \sim \mathcal{N}(\mu_{tor}, \sigma^2_{tor})$
(d)

Optimizing the image term

- for $0 < k < N$ build a k-MST minimizing $\sum_{e_{mn} \in E} d_{mn} t_{mn}$
- k-MST built by an Ant colony optimization (ACO) method
  - "pheromones" to mark useful edges

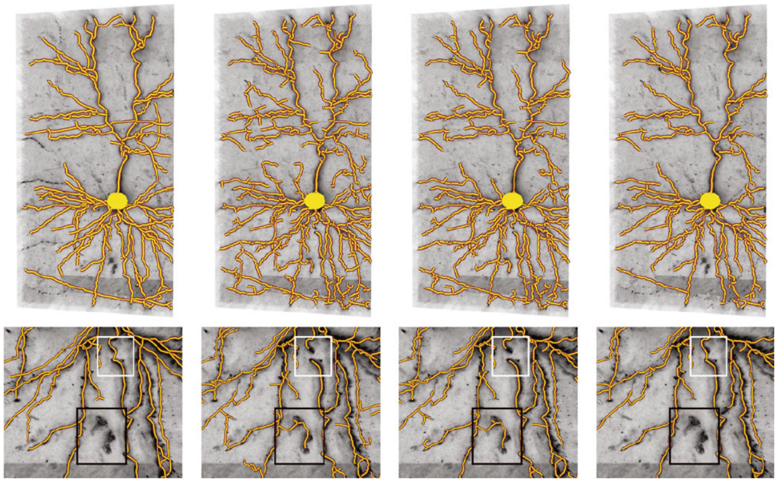


(a) local log-likelihood ratios, (b) total log-likelihood, (c) geometric priors, (d) geometric priors and log-likelihood

Algorithm improvements

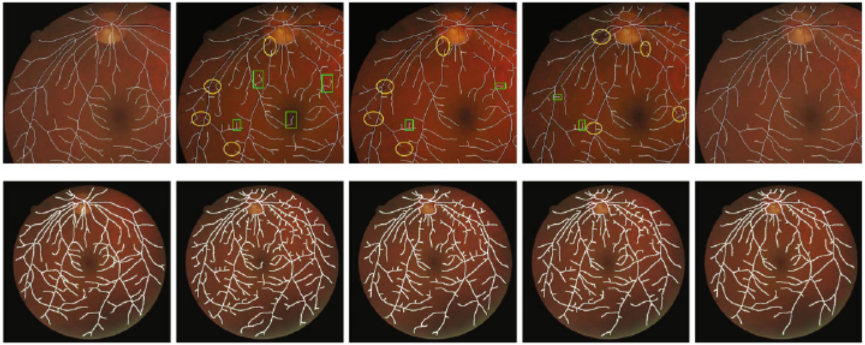
- take into account pairwise geometric terms for edge weights
- pheromone values assigned to pairs of edges
- crossover handling by neighborhood structure
- branching factor limit
- create n_a trees by ACO, choose the best, minimizing the primary objective (log likelihood ratios)

Example results - rat brains



(a) GT, (b) MST, (c) without geometric constraints, (d) with geometric constraints

Retina images

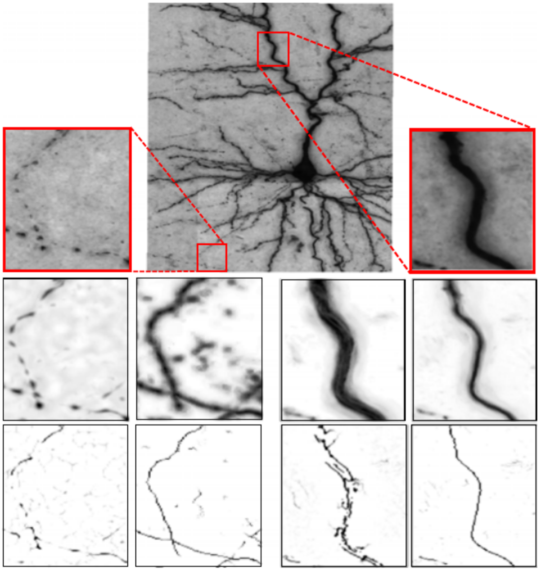


Sironi et al: Multiscale Centerline Detection. IEEE PAMI 2016

Key ideas

- Learned filters
- Centerline detection as a regression problem

Example



Regression target

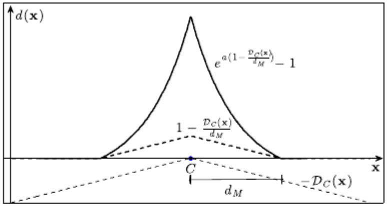


Fig. 3. The function d in the case of $\mathbf{x} \in \mathbb{R}$. If a centerline point is located in C, the function we want to learn is obtained from the distance transform $\mathcal{D}_C$, after thresholding and scaling. The vertical axis has been scaled for visualization purposes.

$$d(\mathbf{x}) = \begin{cases} e^{a(1-\frac{\mathcal{D}_C(\mathbf{x})}{d_M})}-1 & \text{if } \mathcal{D}_C(\mathbf{x}) < d_M, \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

Gradient boost

$$\varphi(f(\mathbf{x}, I)) = \sum_{k=1}^{K} \alpha_k h_k(f(\mathbf{x}, I)),$$

- squared loss $L(d_i, \varphi(f_i)) = (d_i - \varphi(f_i))^2$
- weak learners *h*- 250 regression trees of depth 2

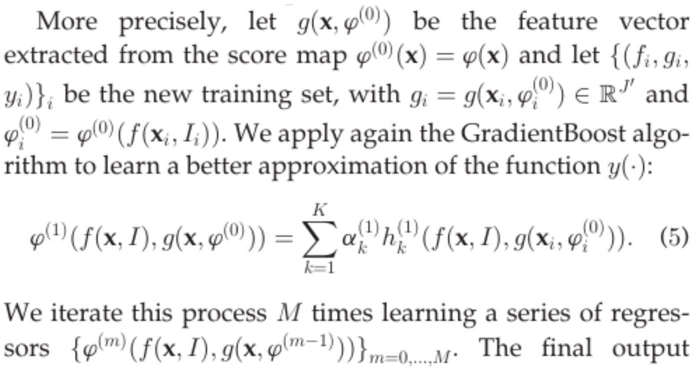
Image features

- Unsupervised learning of sparse convolutional features

$$f(\mathbf{x}, I) = \{(f_j * I)(\mathbf{x} + \mathbf{t})\}_{j,\mathbf{t}},$$

- Rotating learned filters
- Approximated by separable filters

Auto context



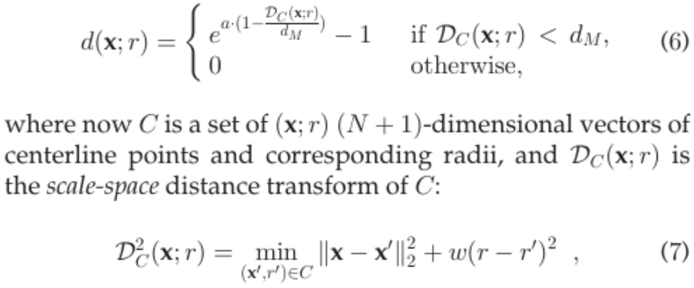
More precisely, let $g(\mathbf{x}, \varphi^{(0)})$ be the feature vector extracted from the score map $\varphi^{(0)}(\mathbf{x}) = \varphi(\mathbf{x})$ and let $\{(f_i, g_i, y_i)\}_i$ be the new training set, with $g_i = g(\mathbf{x}_i, \varphi_i^{(0)}) \in \mathbb{R}^{J'}$ and $\varphi_i^{(0)} = \varphi^{(0)}(f(\mathbf{x}_i, I_i))$. We apply again the GradientBoost algorithm to learn a better approximation of the function $y(\cdot)$:

$$\varphi^{(1)}(f(\mathbf{x}, I), g(\mathbf{x}, \varphi^{(0)})) = \sum_{k=1}^{K} \alpha_k^{(1)} h_k^{(1)}(f(\mathbf{x}, I), g(\mathbf{x}_i, \varphi_i^{(0)})). \quad (5)$$

We iterate this process M times learning a series of regressors $\{\varphi^{(m)}(f(\mathbf{x}, I), g(\mathbf{x}, \varphi^{(m-1)}))\}_{m=0,\dots,M}$. The final output

- random training subset to prevent overfitting, $M = 2$

Scale-space distance transform



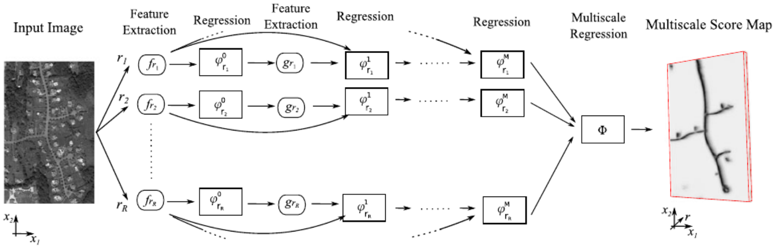
$$d(\mathbf{x};r) = \begin{cases} e^{a\cdot(1-\frac{\mathcal{D}_C(\mathbf{x};r)}{d_M})} - 1 & \text{if } \mathcal{D}_C(\mathbf{x};r) < d_M, \\ 0 & \text{otherwise,} \end{cases} \qquad (6)$$

where now C is a set of $(\mathbf{x};r)$ $(N+1)$-dimensional vectors of centerline points and corresponding radii, and $\mathcal{D}_C(\mathbf{x};r)$ is the *scale-space* distance transform of C:

$$\mathcal{D}_C^2(\mathbf{x};r) = \min_{(\mathbf{x}',r')\in C} \|\mathbf{x}-\mathbf{x}'\|_2^2 + w(r-r')^2 \ , \qquad (7)$$

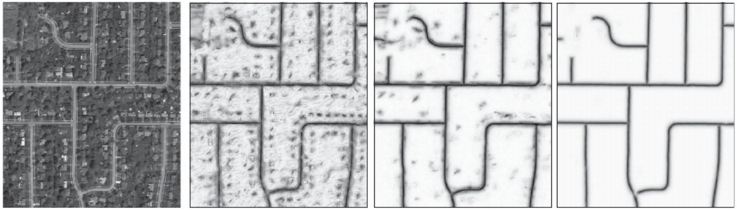
- small set of radii
- scaled and normalized regressor

Flowchart



Input Image
Feature Extraction
Regression
Feature Extraction
Regression
Regression
Multiscale Regression
Multiscale Score Map
r_1
r_2
r_R
f_{r_1}
f_{r_2}
f_{r_R}
$\varphi^0_{r_1}$
$\varphi^0_{r_2}$
$\varphi^0_{r_R}$
g_{r_1}
g_{r_2}
g_{r_R}
$\varphi^1_{r_1}$
$\varphi^1_{r_2}$
$\varphi^1_{r_R}$
$\varphi^M_{r_1}$
$\varphi^M_{r_2}$
$\varphi^M_{r_R}$
Φ
x_1
x_2
r

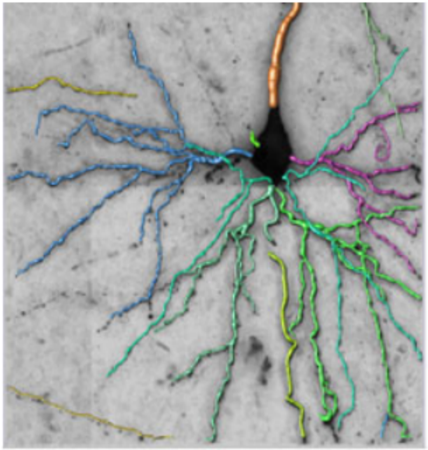
Example score maps



(a) Input Image I (b) Score map $\varphi^{(0)}$ (c) Score map $\varphi^{(M)}$ (d) Final approximation Φ

Example neuron delination

(using integer programming reconstruction)

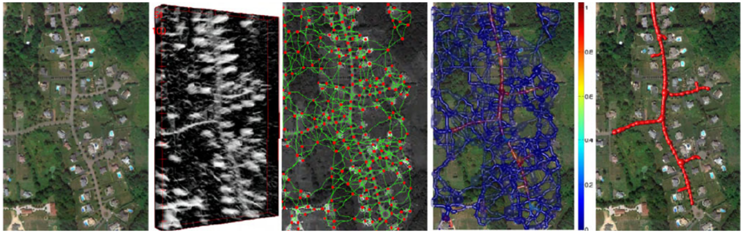


Türetken et al: Reconstructing Curvilinear Networks Using Path Classifiers and Integer Programming. IEEE PAMI 2016

Key ideas

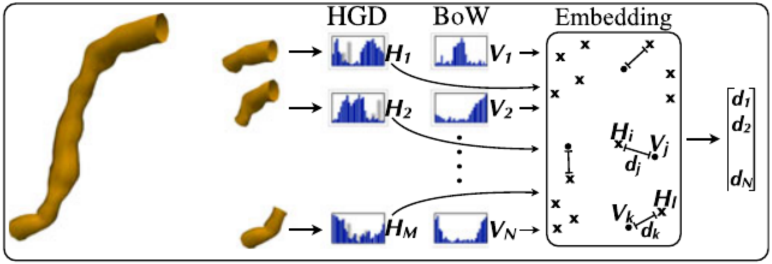
- Classifier-based edge weights
- Integer program formulation, leverage existing solvers
- Allow for non-tree topologies

Flowchart



Path weights

- geodesic tubular path maximizing tubularity in the space-scale coordinates
- for each segment - gradient strength and gradient symmetry histogram by angle
- BoW descriptor - distances to the codewords
- gradient boosted decision tree classifier



Quadratic program

$$\underset{\mathbf{t}\in\mathcal{T}(G)}{\operatorname{argmin}} \sum_{e_{ij},e_{jk}\in E} c_{ijk}\, t_{ij}\, t_{jk}. \qquad t_{ij}\in\{0,1\},$$

- connected tree - flow constraints.
- y^l_{ij}- if path to l traverses e_{ij},root(s) given

$$\sum_{j\in V\setminus\{r\}} y^l_{rj} \le 1, \qquad \forall l\in V\setminus\{r\},$$

$$\sum_{j\in V\setminus\{l\}} y^l_{jl} \le 1, \qquad \forall l\in V\setminus\{r\},$$

$$\sum_{j\in V\setminus\{i,r\}} y^l_{ij} - \sum_{j\in V\setminus\{i,l\}} y^l_{ji} = 0, \qquad \begin{array}{c}\forall l\in V\setminus\{r\},\\ \forall i\in V\setminus\{r,l\},\end{array}$$

$$y^l_{ij} \le t_{ij}, \qquad \forall e_{ij}\in E,\ l\in V\setminus\{r,i,j\},$$

$$y^l_{il} = t_{il}, \qquad \forall e_{il}\in E,$$

$$y^l_{ij} \ge 0, \qquad \forall e_{ij}\in E,\ l\in V\setminus\{r,i\},$$

Optimization

- Branch and cut *(implemented in Gurobi)*
  - solve without integer constraints
  - add a constraint violated by the current non-integer solution
  - branch and bounds: branch on a variable and solve the two subproblems, some subproblems may be pruned
- Prune or merge some edges based on weights
- Start with reduced set of constraints. Check for connectivity violations. Add constraint. Repeat.

Example results