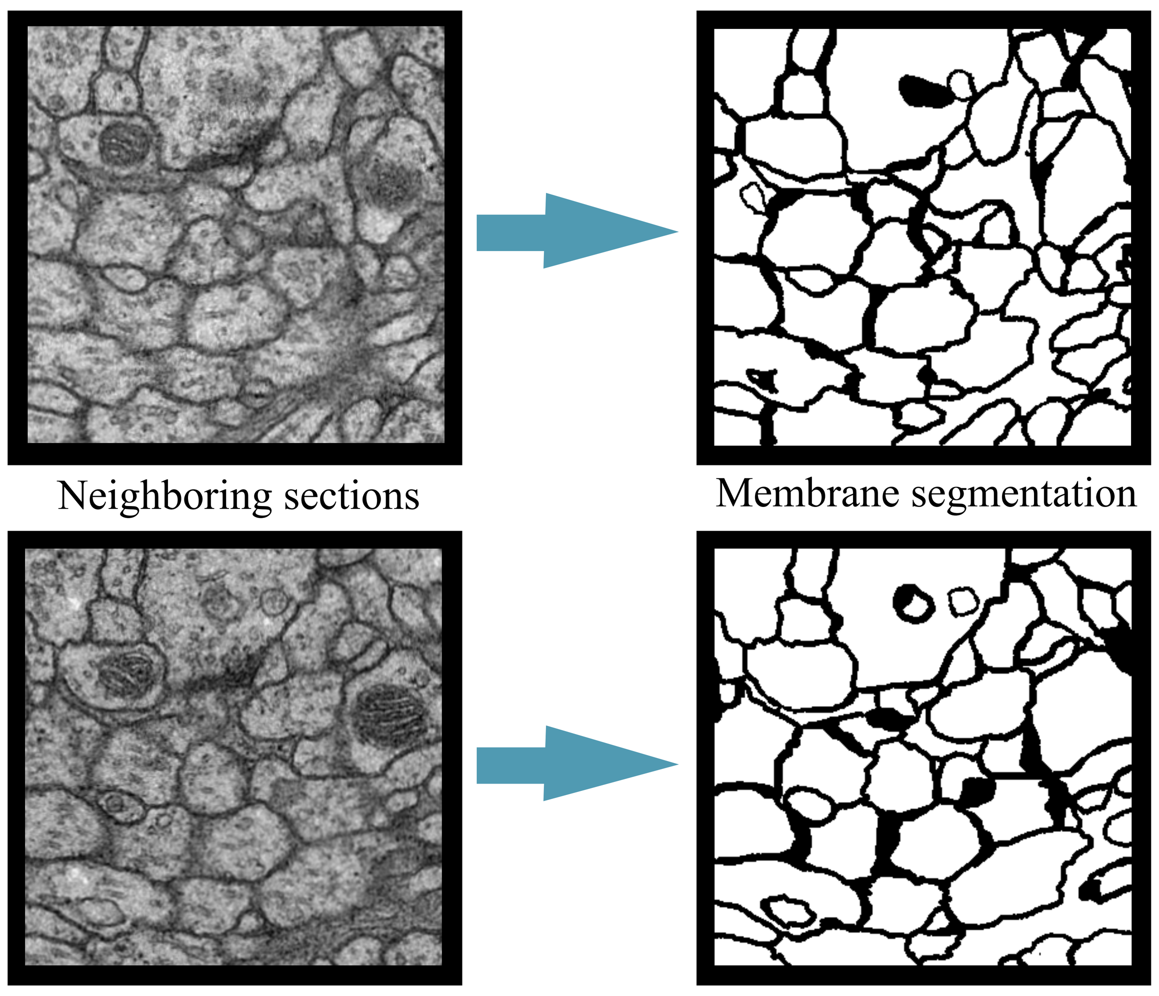
Anisotropic ssTEM Image Segmentation Using Dense Correspondence across Sections

D. Laptev, A. Vezhnevets, S. Dwivedi, J. M. Buhmann

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

1. Membrane segmentation

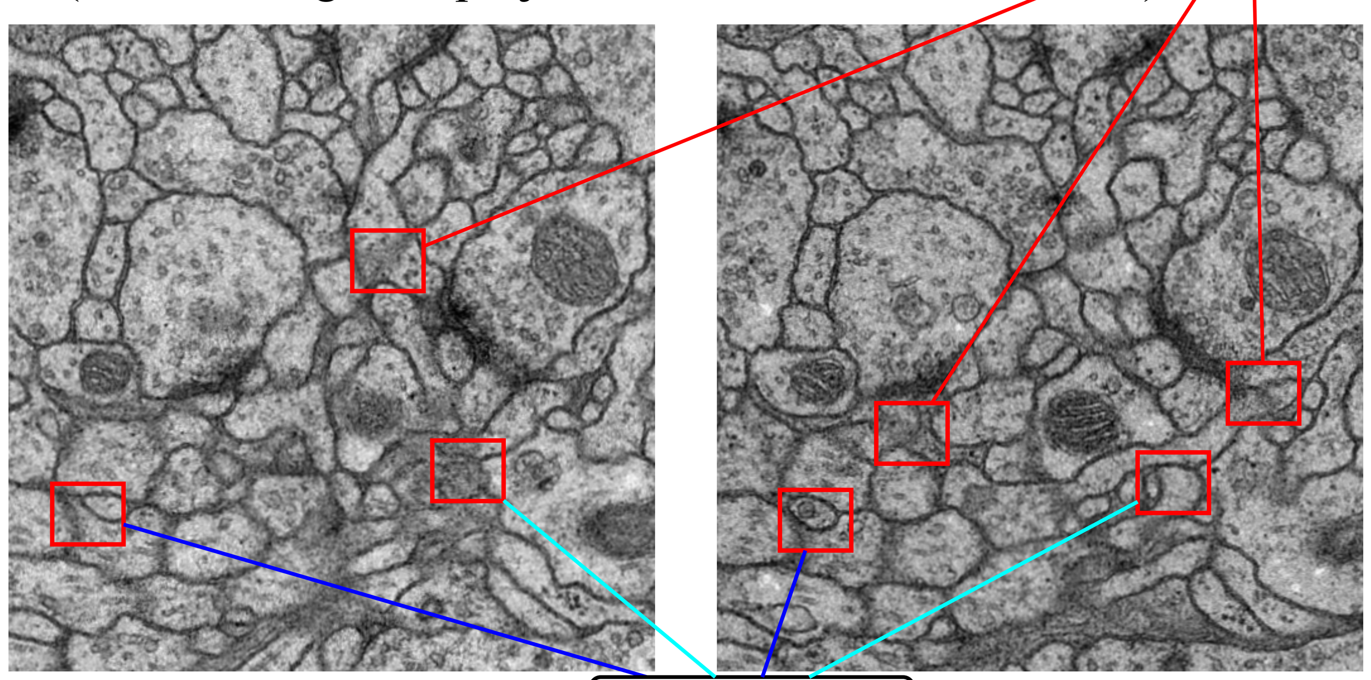
Task: automatic membrane segmentation for *anisotropic stacks* of ssTEM brain tissue



Sections from a ssTEM of the Drosophila first instar larva ventral nerve cord (VNC), imaged at a resolution of 4x4x50 nm/pixel (Cardona, A. et al. in PLoS Biol. 10, 2010)

2. Problems

- Membranes that are not orthogonal to cutting plane are blured (as each image is a projection of the whole thick slice)

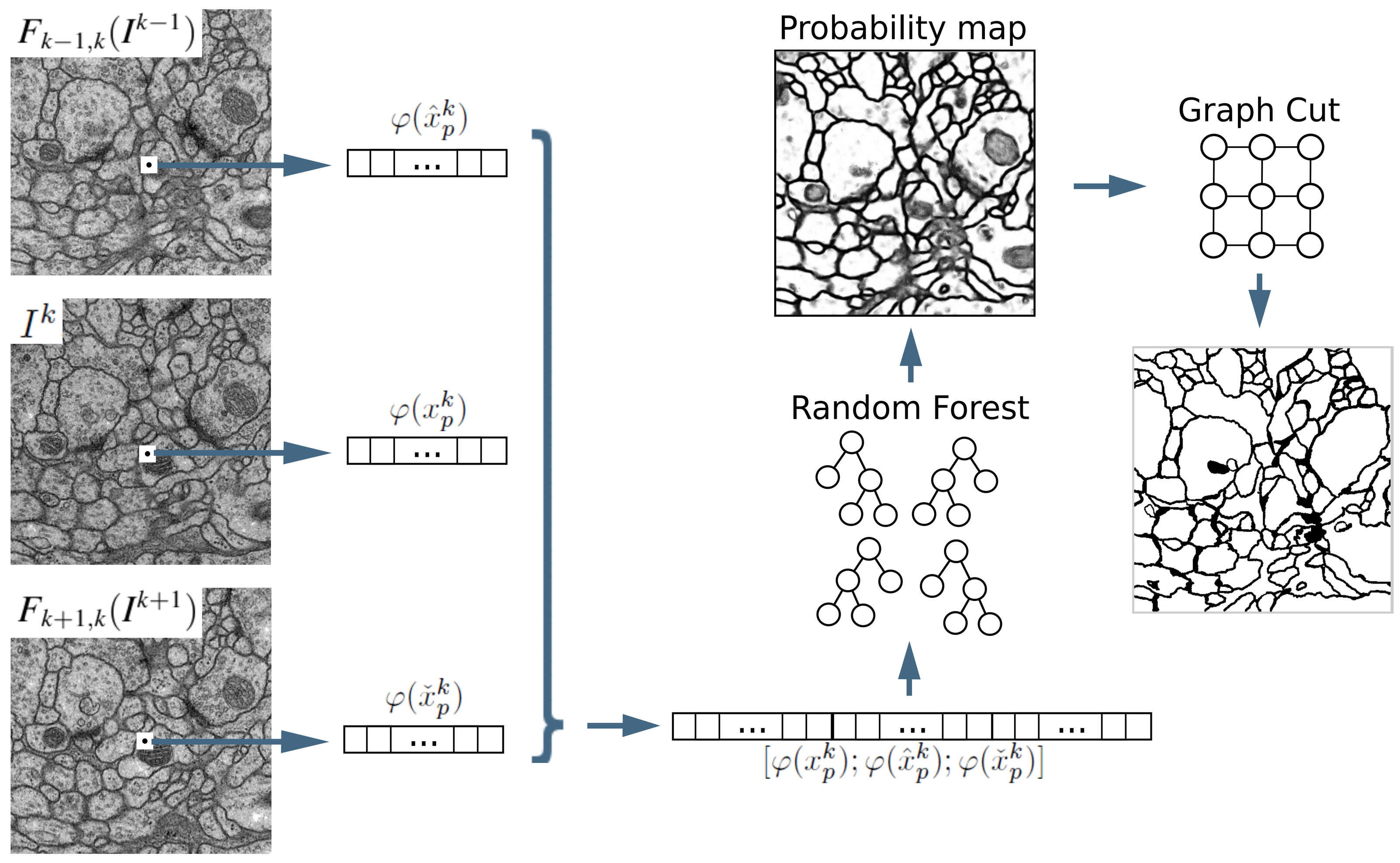


- Neighboring slices are highly different one from another (data is highly anisotropic), so we cannot use methods developed for isotropic data

Key idea

- Approximately find corresponding regions in neighboring sections
- Incorporate the context from neighboring sections

3. Proposed method



(warping images; feature extraction) (concatenation; training RF) (segmentation)

Notation

- I^{k-1}, I^k and I^{k+1} – neighboring sections,
- x_p^k – pixel in section k,
- $\phi(x_p^k)$ – feature vector in pixel x_p^k.
- Non-linear warpings operators:
  - $F_{k-1,k}$: $F_{k-1,k}(x_p^{k-1}) = \hat{x}_p^k$
  - $F_{k+1,k}$: $F_{k+1,k}(x_p^{k+1}) = \check{x}_p^k$.
- Extended feature vector: $[\phi(x_p^k); \phi(\hat{x}_p^k); \phi(\check{x}_p^k)]$.

- In order to incorporate the context from neighboring sections, we need to find warping operator F between sections
- We do it by finding Global Dense Correspondence with SIFT Flow (Ce Liu et al. in ECCV 2008)

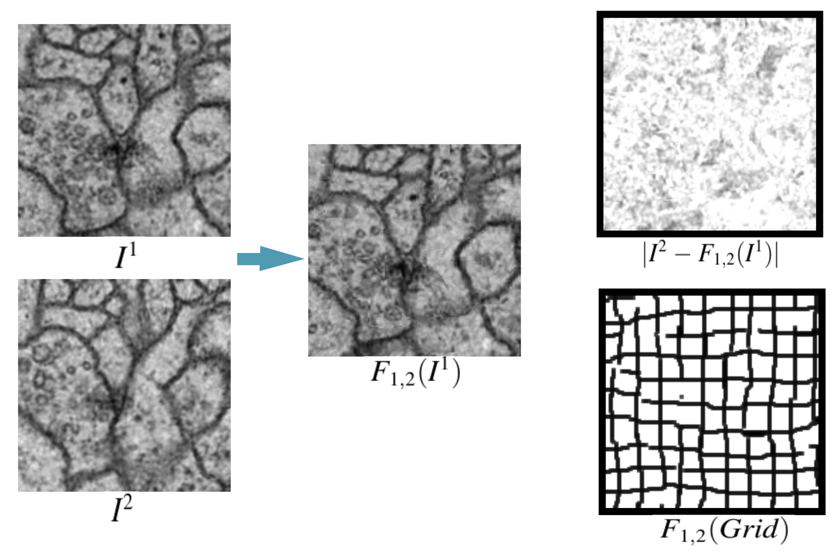
Non-linear warping F is found by searching for a minimum of SIFT flow energy:

$$E(F_{k-1,k}) = \sum_{p=1}^{N} \min\left(\|s(x_p^k) - s(F_{k-1,k}(x_p^{k-1}))\|, t\right) + \sum_{p=1}^{N} \gamma D(x_p^{k-1}, F_{k-1,k}(x_p^{k-1})) + \sum_{(p,q)\in\epsilon} \min\left(\alpha D(F_{k-1,k}(x_p^{k-1}), F_{k-1,k}(x_q^{k-1})), d\right),$$

$s(x_p)$ – sift descriptor in pixel x_p;

$D(x_p, x_q)$ – distance between pixels x_p and x_q;

t, γ, α, d – model parameters.



(input images) (warped image) (difference) (non-linear warping)

- We evaluate feature vectors in corresponding pixels
- Then concatenate them to get **extended feature vector**
- Train Random Forest classifier with these extended vectors

Pixel Features

- Mean, Minimum, Maximum, Median, Variance,
- Gaussian blur, Sobel filter, Hessian, Gradient,
- Bilateral, Lipschitz, Kuwahara, Gabor, Laplacian,
- SIFT descriptors, Radon-like features, Ray features,
- Line Filter Transform.

4. Graph Cut & Postprocessing

- We use graph cut segmentation to take into account the fact that the labels of the neighboring pixels are more likely to have the same value.
- Labelling Y is then estimated as argminimum of the following energy:

$$E(Y) = \sum_{p=1}^{N} E_{rf}(y_p) + \lambda_s \sum_{(p,q)\in\epsilon} E_s(y_p, y_q) + \lambda_{gf} \sum_{p=1}^{N} E_{gf}(y_p) + \lambda_{gc} \sum_{(p,q)\in\epsilon} E_{gc}(y_p, y_q)$$

$E_{rf}(y_p)$ negative log likelihood of a RF;
$E_s(y_p, y_q)$ smoothness term;
$E_{gf}(y_p)$ gradient flux term;
$E_{gc}(y_p, y_q)$ good continuation term.

Posprocessing

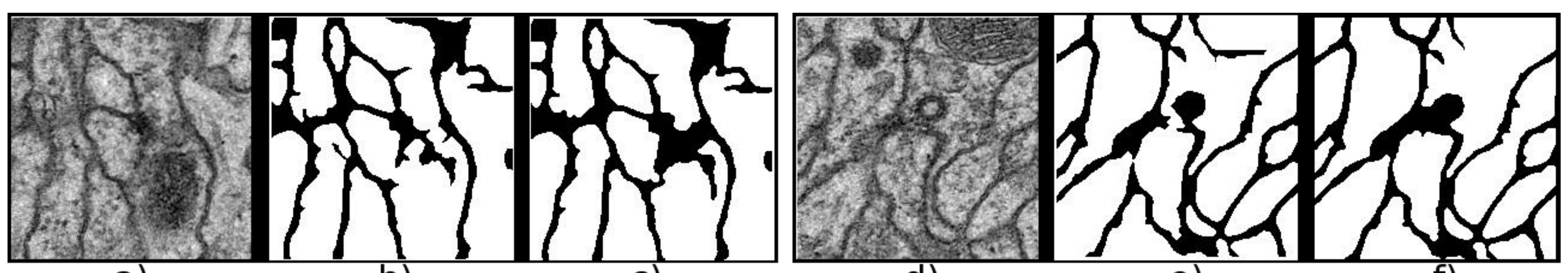
- Filtering connected regions based on their morphological properties: Area, Solidity, Euler Number and Eccentricity.
- Line Filter Transform:

$$LFT(x_p) = \max_{l \in L_r(x_p)} \int_{z \in l} y_z dz$$

Where $L_r(x_p)$ is a set of all the possible lines of the length r.

5. Results

Two examples of the resulted labelling. Original image: (a), (d), no context: (b), (e), our method (c), (f)



Results are based on ISBI 2012 challenge results "Segmentation of neuronal structures in EM stacks" (http://bit.ly/riGDUm).

Method	Pixel error	Warping error	Method	Pixel error	Warping error
Human	$6.7 * 10^{-2}$	$3.4 * 10^{-4}$	*IDSIA*	$6.0 * 10^{-2}$	$4.3 * 10^{-4}$
Dense *ETH*	$7.9 * 10^{-2}$	$6.2 * 10^{-4}$	*CSIRO*	$8.7 * 10^{-2}$	$6.8 * 10^{-4}$
Direct *ETH*	$8.0 * 10^{-2}$	$6.5 * 10^{-4}$	*Utah*	$1.3 * 10^{-1}$	$1.6 * 10^{-2}$
One slice *ETH*	$8.5 * 10^{-2}$	$6.4 * 10^{-4}$	*NIST*	$1.5 * 10^{-1}$	$1.6 * 10^{-2}$

Our method is **3.6** and **6.4%** more accurate in two different accuracy metrics than the algorithm with no context from other sections.