ETH zürich

Towards expert-aware computer vision algorithms in medical imaging

PhD examination presentation by Dmitry Laptev

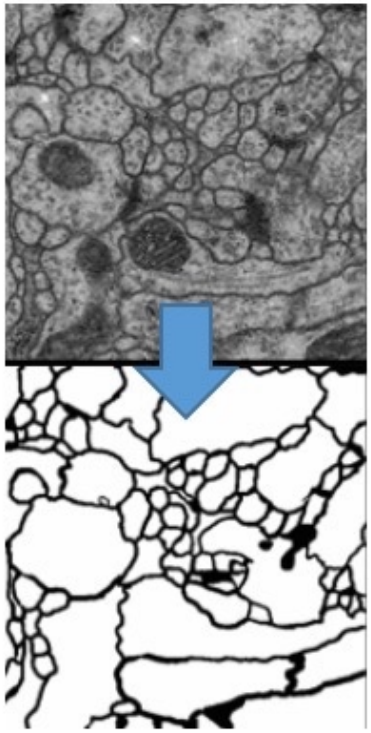
Towards expert-aware computer vision algorithms in medical imaging

Towards expert-aware **computer vision** algorithms in medical imaging

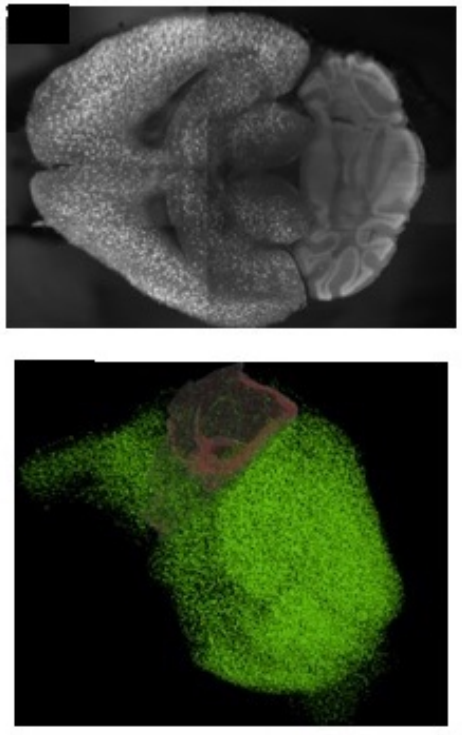
- Problems:
  - Segmentation / classification,
  - Image enhancement / restoration;
- Techniques:
  - Feature engineering and learning,
  - Random forests, decision jungles,
  - Max-flow / min-cut, optical flow, registration,
  - Convolutional neural networks.

Towards expert-aware computer vision algorithms in **medical imaging**

Electron microscopy membrane segmentation



Optical microscopy on cleared brains

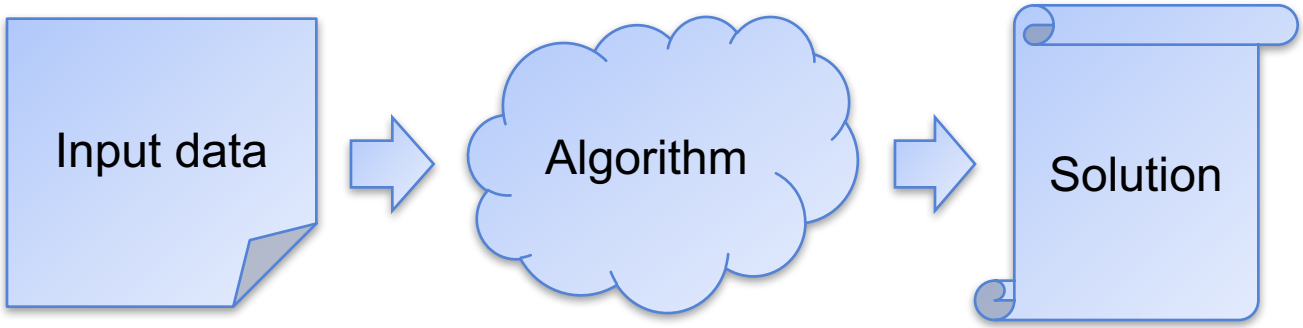


But also other non-medical datasets

Towards **expert-aware** computer vision **algorithms** in medical imaging

Incorporate strong prior knowledge from field experts into modern algorithmic pipelines

Towards **expert-aware** computer vision **algorithms** in medical imaging



- Different types of expert prior knowledge:
  - useful known properties of the input data,
  - expert approach to solve the problem,
  - known statistics of the output objects.

Why expert-aware approach?

- Unrepresentative data sets.
  - Some ambiguities cannot be resolved without additional structural information (3D problem in 2D).
- Limited amount of labelled data.
  - Annotations can be very expensive limiting the power of supervised learning methods.
- Limited number of samples.
  - Data collection can be very expensive (invasive tissue sampling, growing a subject animal).

The structure of the presentation

- Anisotropic data segmentation / enhancement.
  - Expert-mimicking method to resolve ambiguities in electron microscopy and video data.
- Parameter tuning in a weakly-supervised setting.
  - Employing known biological properties of the solution for amyloid plaque estimation.
- Transformation-invariance.
  - Enforcing algorithm invariance to the variation of the input data to allow efficient use of available samples.

List of publications

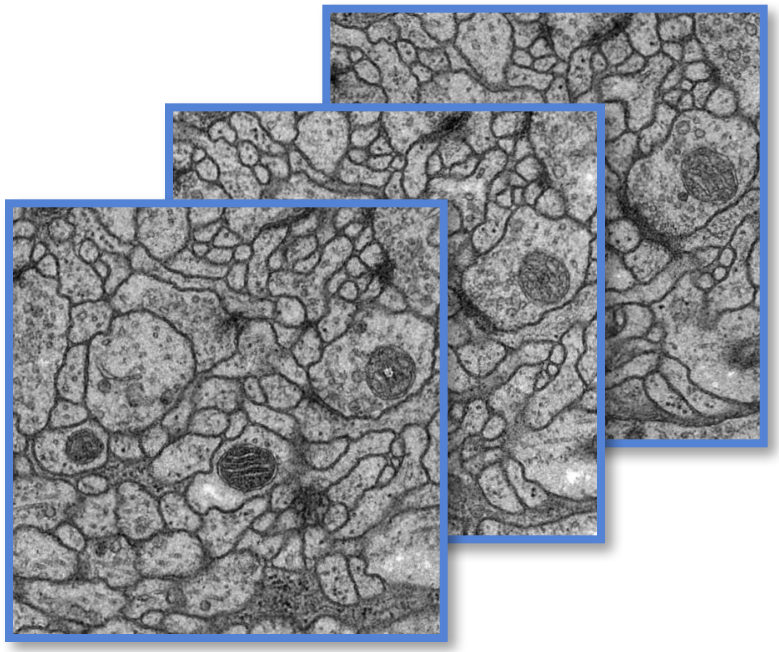
- Dmitry Laptev, Alexander Vezhnevets, Sarvesh Dwivedi, Joachim M. Buhmann. Anisotropic ssTEM image segmentation usingdense correspondence across sections. Medical Image Computing and Computer-Assisted Intervention – MICCAI 2012, pages 323–330, 2012.
- Dmitry Laptev, Alexander Vezhnevets, Joachim M. Buhmann. SuperSlicing frame restoration for anisotropic ssTEM. IEEE 11th International Symposium on Biomedical Imaging–ISBI 2014, pages 1198–1201, 2014.
- Dmitry Laptev, Joachim M. Buhmann. SuperSlicing frame restoration for anisotropic ssTEM and video data. ECML 2014 Neural Connectomics workshop – NCW ECML 2014, JMLR Workshop and Conference Proceedings 46, pages 93–103, 2015.
- Dmitry Laptev, Joachim M. Buhmann. Convolutional Decision Trees for feature learning and segmentation. German Conference on Pattern Recognition – GCPR 2014, Pattern Recognition 8753, pages 95–106, 2014.
- Dmitry Laptev, Joachim M. Buhmann. Transformation-Invariant Convolutional Jungles. IEEE International Conference on Computer Vision and Pattern Recognition – CVPR 2015, pages 3043–3051, 2015.
- Ignacio Arganda-Carreras, Srinivas C. Turaga, Daniel R Berger, Dan Cireşan, Alessandro Giusti, Luca M. Gambardella, Jürgen Schmidhuber, Dmitry Laptev, Sarvesh Dwivedi, Joachim M. Buhmann, et al. Crowdsourcing the creation of image segmentation algorithms for connectomics. Frontiers in Neuroanatomy 9, 2015, article 142.
- Dmitry Laptev, Nikolay Savinov, Joachim M. Buhmann, Marc Pollefeys. TI-Pooling: transformation-invariant pooling for feature learning in Convolutional Neural Networks. IEEE/CVF International Conference on Computer Vision and Pattern Recognition – CVPR 2016.
- Dmitry Laptev, Daniel Kirschenbaum, Joachim M. Buhmann, Adriano Aguzzi. Biologically-motivated priors for non-parametric plaque analysis in mouse brains. *The paper in preparation.*
- Daniel Kirschenbaum, Oliver Bichsel, Dmitry Laptev, Michael B. Smith, Fabian F. Voigt, Joachim M. Buhmann, Adriano Aguzzi. Rapid electrophoretic tissue clearing and molecular labelling in whole mount mouse brain. *The manuscript in preparation.*

Anisotropic data

Idea: to employ expert approach to resolve ambiguities

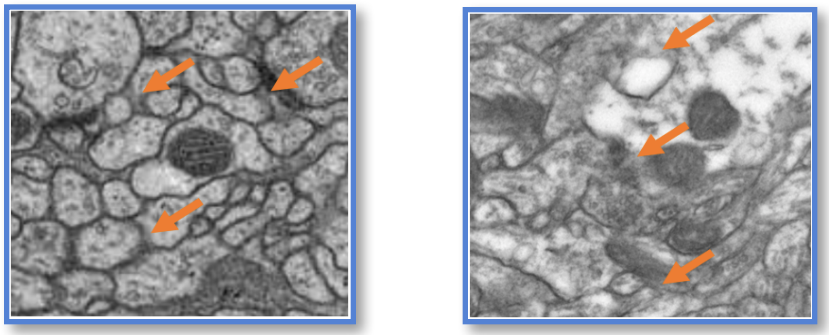
Anisotropic data definition

- A collection of sequential images (a stack), in which the resolution across one dimension of the stack is much lower than the resolution of the other two dimensions.

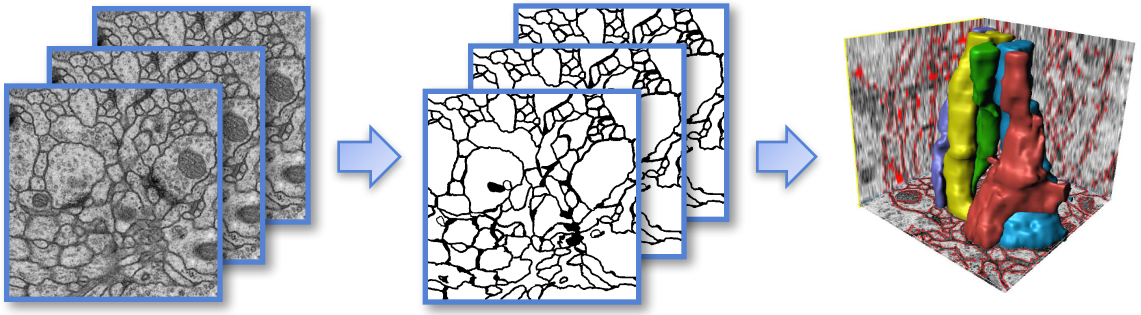


Serial section transmission electron microscopy

- Highly anisotropic volume (resolution is $4 \times 4 \times 50$nm).
- Some neuronal structures are blurred due to anisotropy.

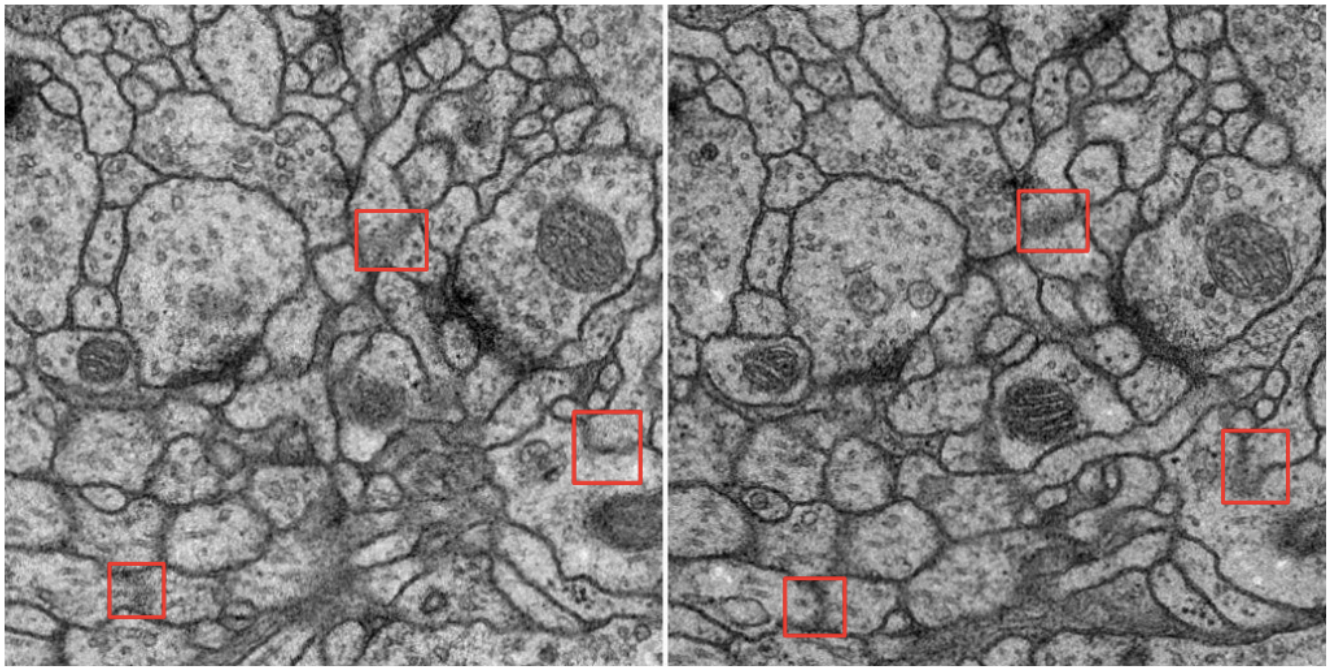


- Enables neuronal geometry reconstruction for *Connectomics*.



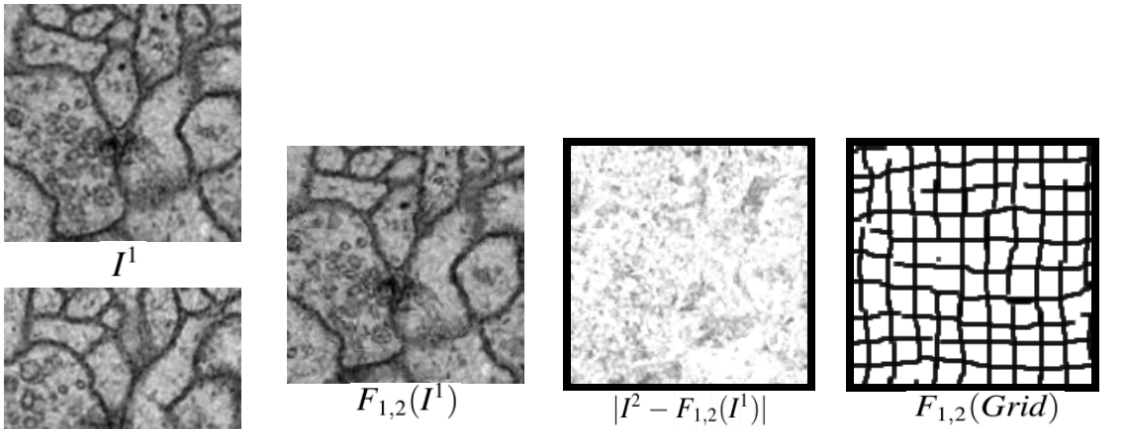
Expert approach

- Resolve ambiguities using neighboring sections.
  - A step from local appearance to global appearance (from 2D to 3D).
  - Find corresponding regions and combine information from them.



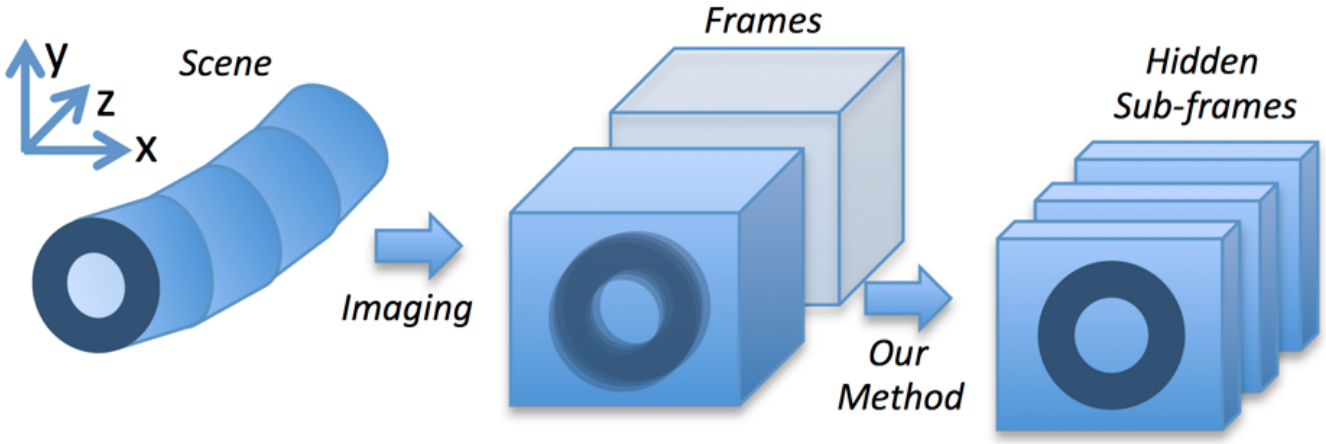
Correspondence problem

- Image registration:
  - find pixels in two images that correspond to the same biological structure.
- We use either *SIFT-flow* or *Optical-flow* methods:
  - given two images I^1 and I^2, find the warping $F_{1,2}$ between them.



Anisotropic data restoration: SuperSlicing

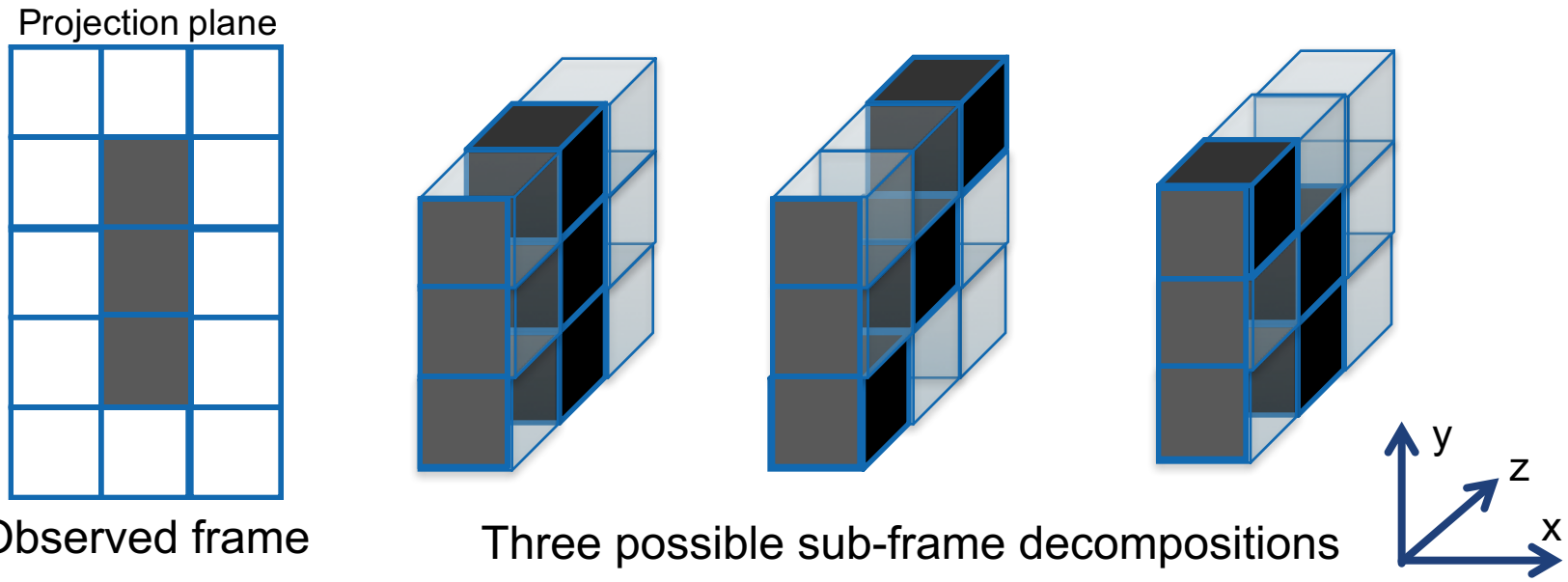
- Bringing the information from multiple slices is useful by itself, but the structure can be blurred in multiple sections.
- Can we restore the averaged away details?



- Idea: decompose the observed frame into “hidden sub-frames”

Idea of SuperSlicing

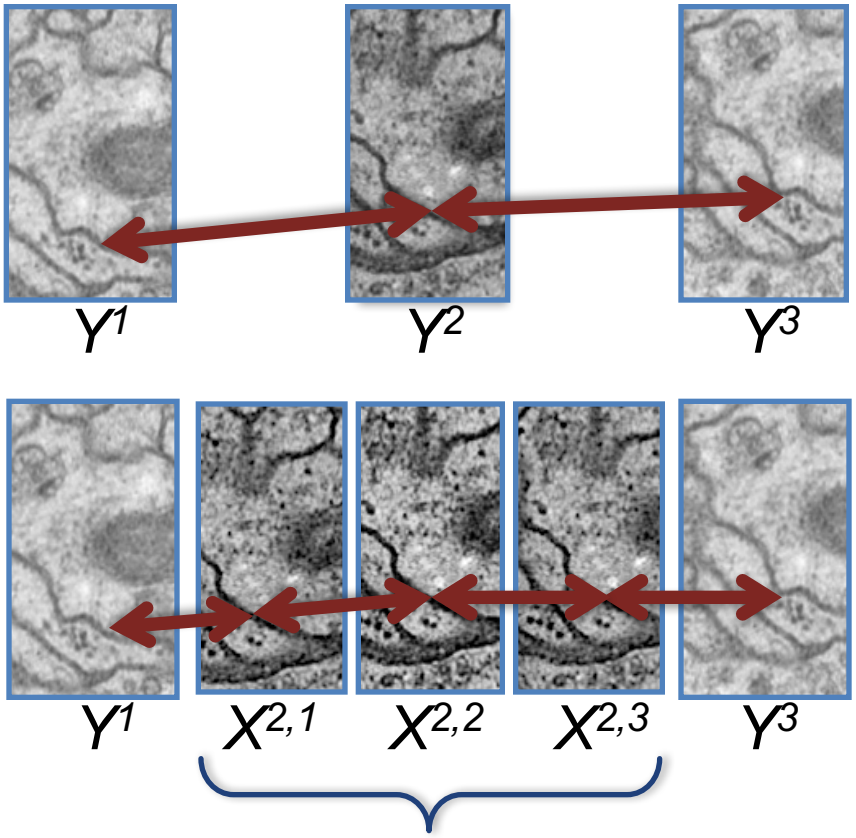
- Very ill-posed problem: multiple possible solutions.



- Need structural constraints.

Idea of SuperSlicing

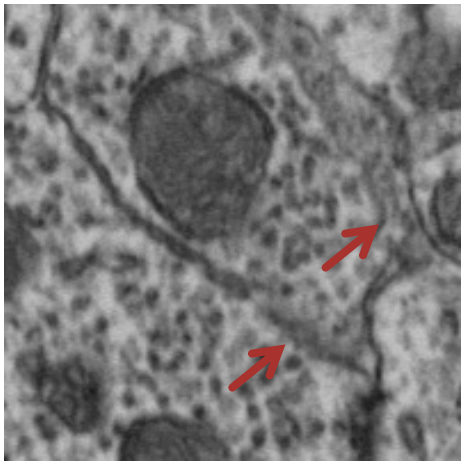
Y^1, Y^2, Y^3 – observed neighboring frames



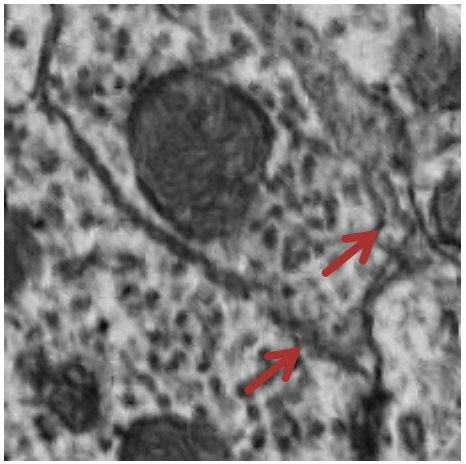
Hidden sub-frame decomposition of Y^2

- Find corresponding pixels in anisotropic frames by solving **registration task.**
- Interpolate correspondences between pixels in sub-frames.
- Corresponding pixels should have similar intensities.

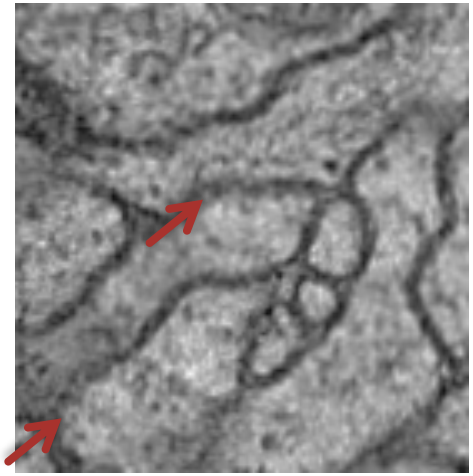
SuperSlicing experiments: ssTEM

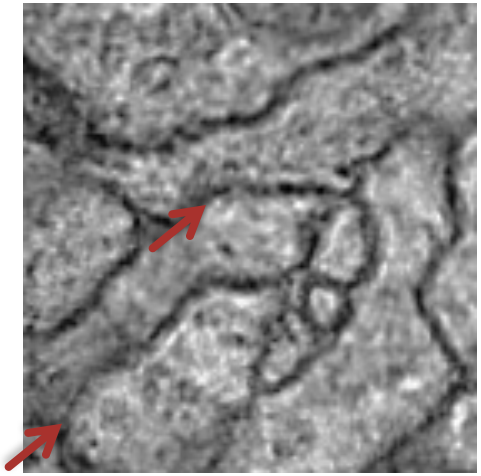


Anisotropic frames

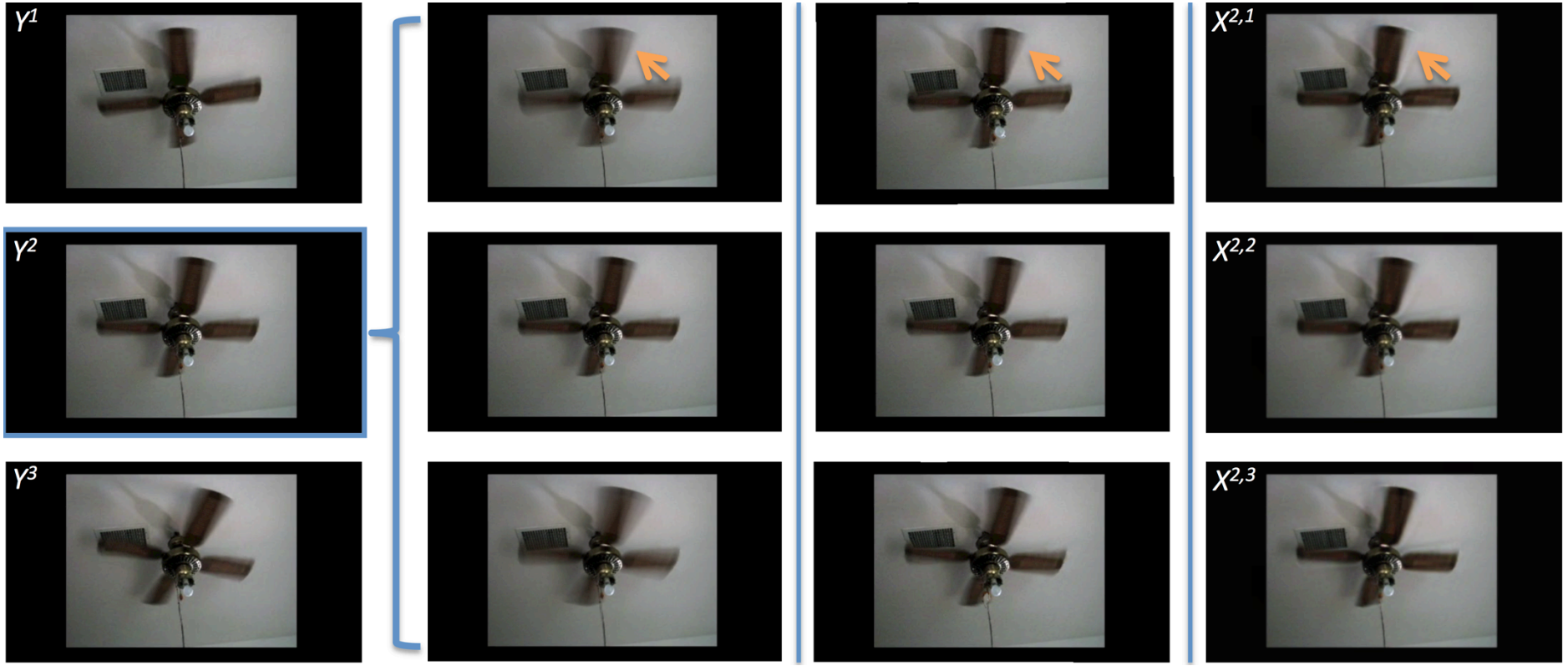


Reconstructed sub-frame



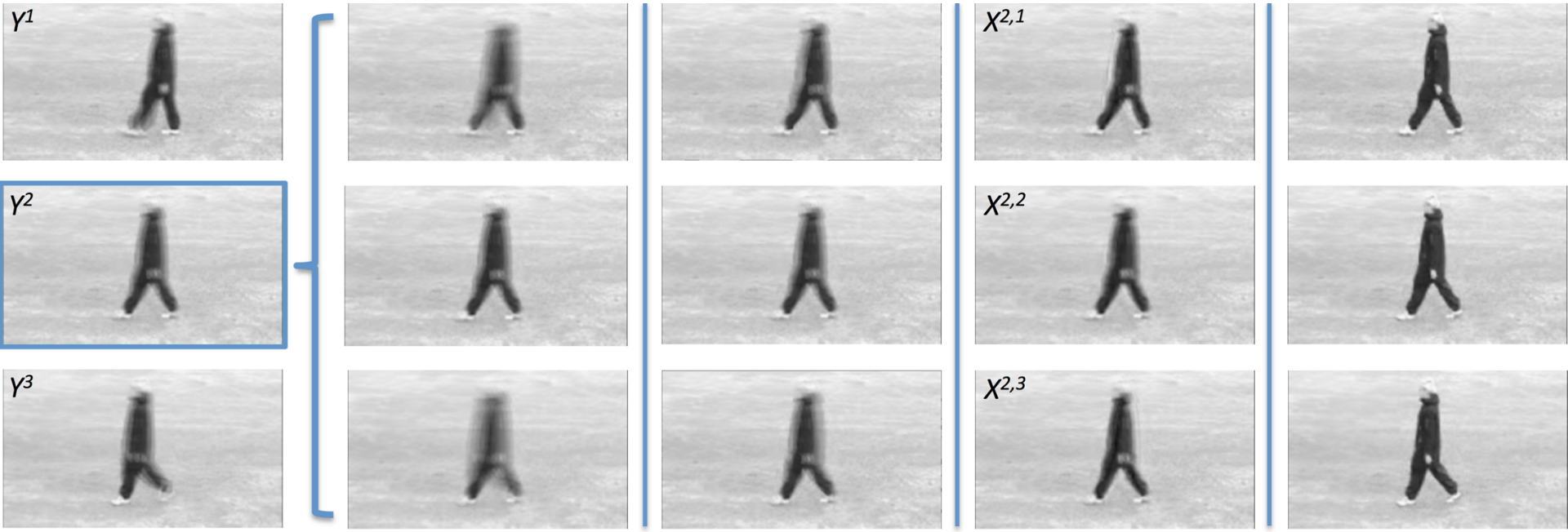


SuperSlicing experiments: video



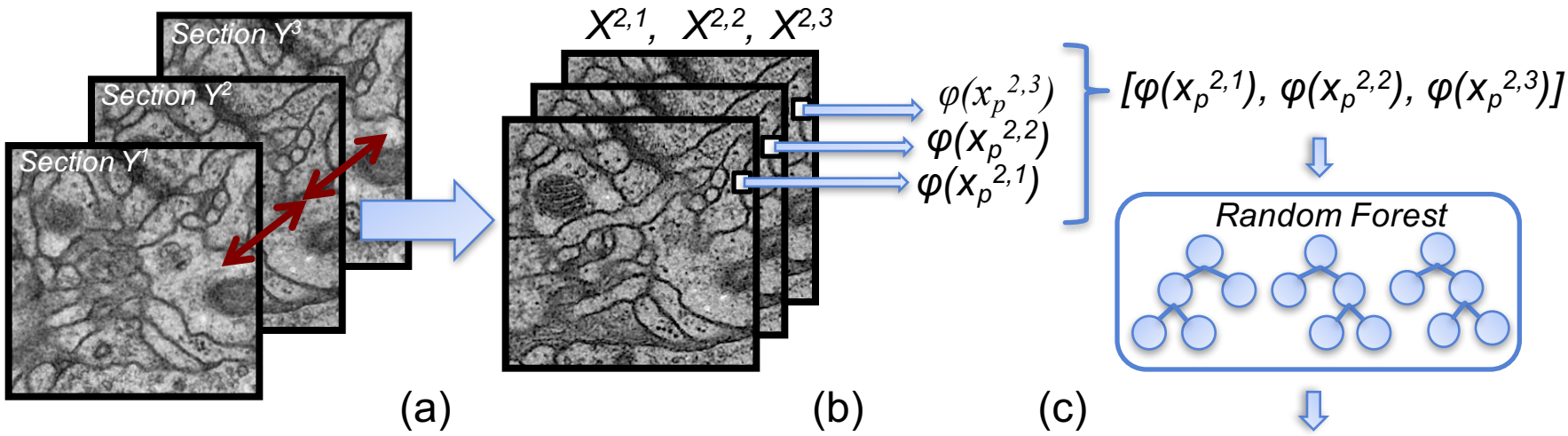
Three original frames | *Linear interpolation* | *Optical Flow warping* | *Ours*

SuperSlicing experiments: synthetic video



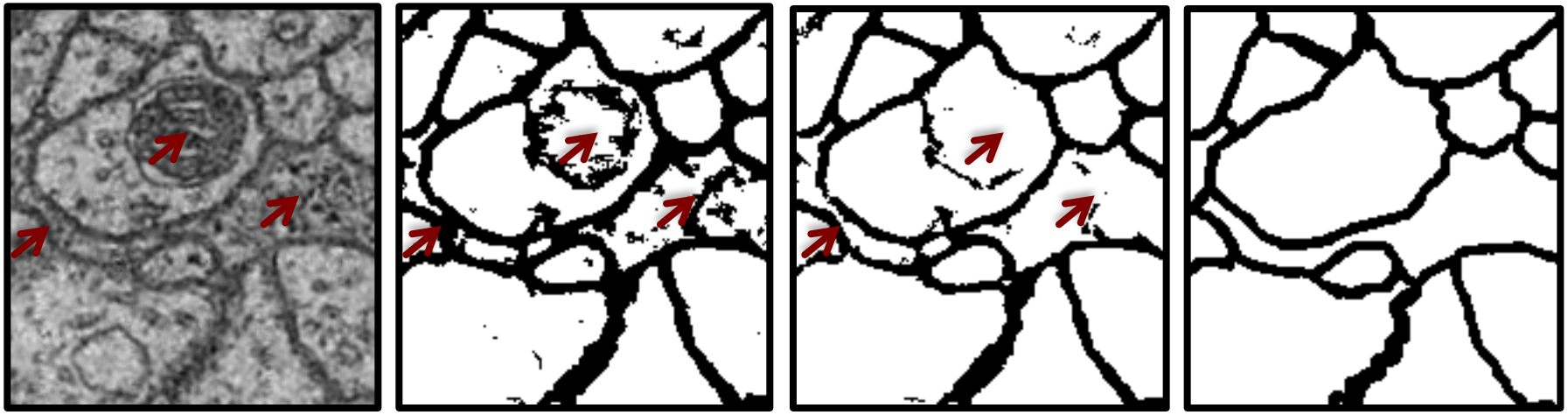
Synthetic frames | *Linear interpolation* | *Optical Flow warping* | *Ours* | *Original frames*

SuperSlicing for anisotropic data segmentation



a) Decompose anisotropic section with SuperSlicing
b) Extract feature vectors in sub-frames
c) Concatenate feature vectors and pass to a Random Forest classifier

SuperSlicing for anisotropic data segmentation



Original fragment *One-slice segmentation* *SuperSlicing segmentation* *Ground Truth*

Method	Warping error	
One section segmentation	$2.87*10^{-3}$	(17%)
Three section segmentation	$2.69*10^{-3}$	(11%)
SuperSlicing segmentation	$2.38*10^{-3}$	

Anisotropic data summary

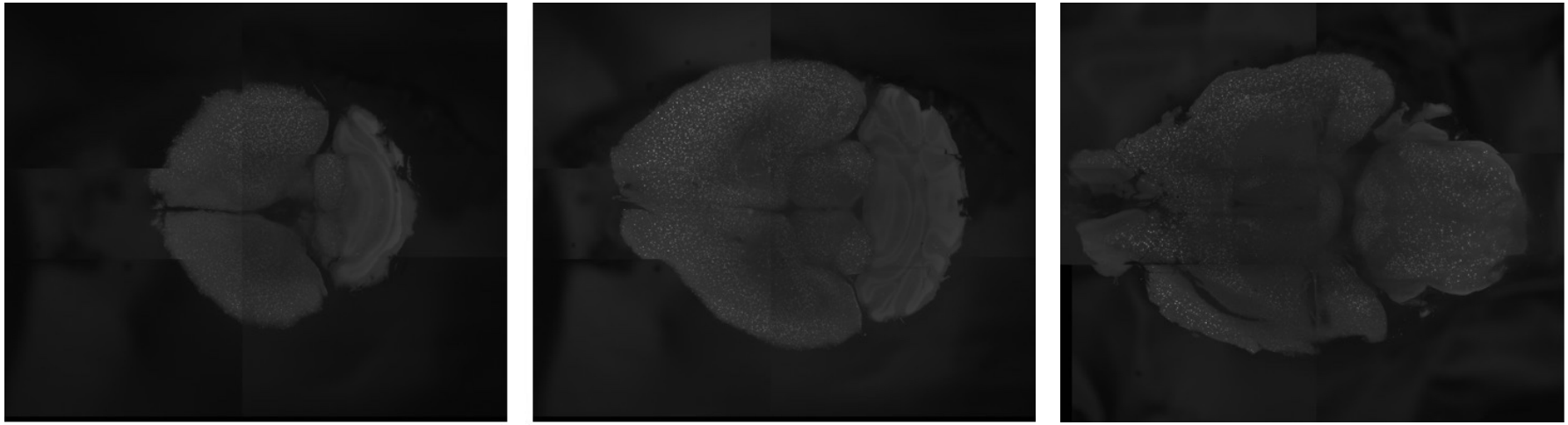
- Expert-mimicking approach:
  - Employ correspondences to resolve ambiguities.
- SuperSlicing:
  - Reconstruct hidden slices, not interpolate between them.
  - 10% better PSNR than with non-linear interpolation.
  - Good for visualization and further processing.
- Segmentation:
  - Concatenated feature vectors to combine information from slices.
  - 17% better accuracy when compared to one-slice segmentation.

Global biological priors

Idea: to employ the known properties of the output objects

Amyloid plaque distribution estimation

- Plaques in a 3D brain volume: compact regions of higher intensity.

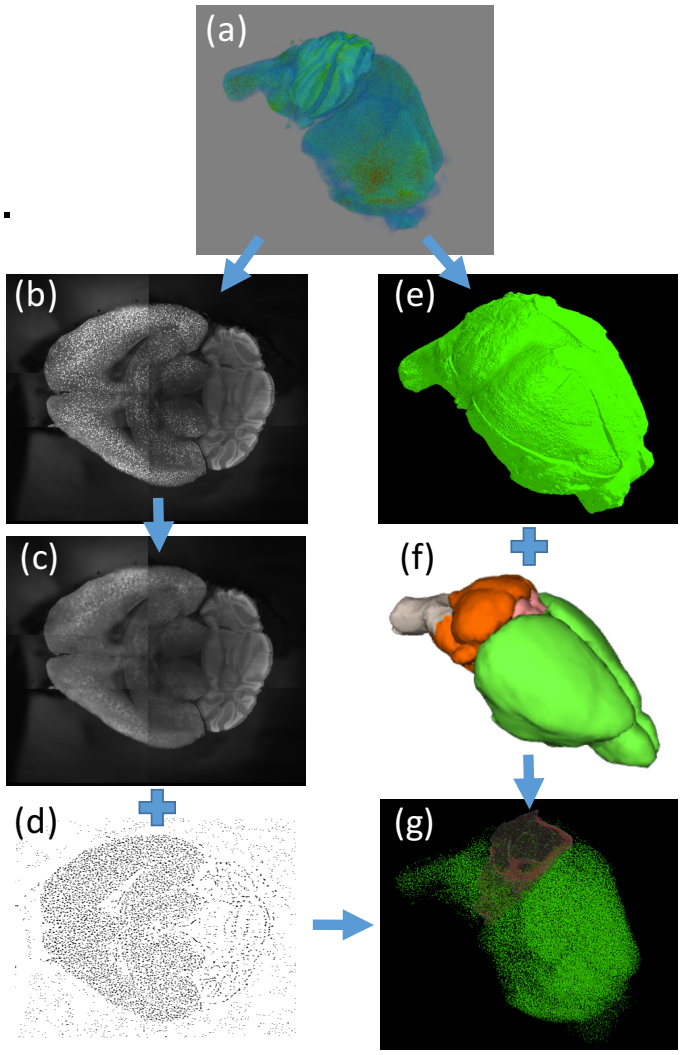


- Challenges:
  - large appearance variations within the volume;
  - weakly-supervised setting (no ground truth).

Plaque estimation pipeline

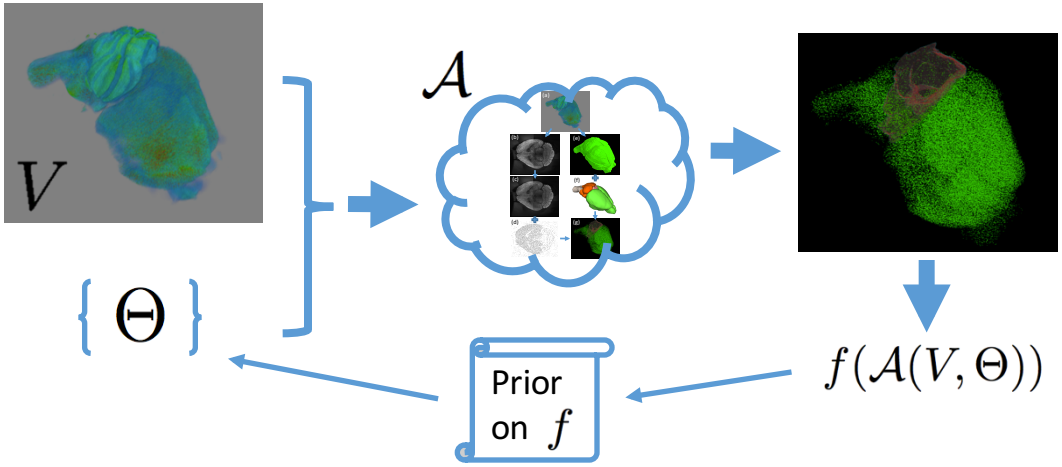
a) The brain volume imaged with SPIM.
b) Each slice is separated into:
c) background and
d) plaque candidates.
e) Volume segmentation.
f) Volume-atlas registration.
g) Resulting plaques in green and cerebellum highlighted in red.

- Algorithm parameters:
  - plaque candidate threshold,
  - segmentation smoothness.



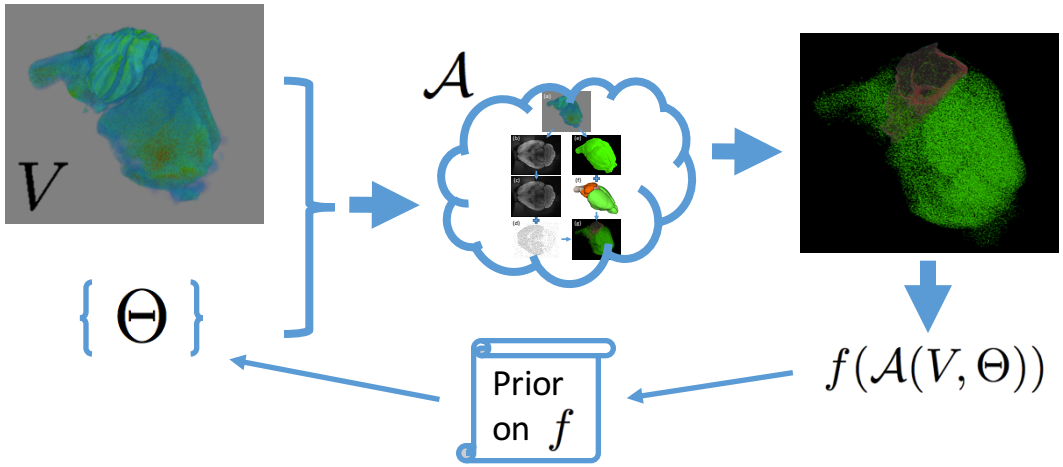
Feedback-loop for parameter tuning

- Idea: employ known biological properties of the output.
  - For example, plaques volume is known to be 6.5e-5 mm^3.



- Given a parametrized algorithm $\mathcal{A}$, tune the parameters such that the output statistic corresponds to the prior.

Feedback-loop for parameter tuning



Lemma 1. *Let the algorithm $\mathcal{A}$ depend on the parameter set Θ. Also assume we have formulated a property f of the output of $\mathcal{A}$ as a statistic $f(\mathcal{A}(\Theta))$. If the function $f(\mathcal{A}(\theta))$ is smooth and strictly monotonic for every $\theta \in \Theta$, then the binary search procedure is able to identify the value of the parameters θ such that $f(\mathcal{A}(\Theta))$ is equal to the desired prior on f up to any tolerance level within a finite amount of steps.*

Global biological priors summary

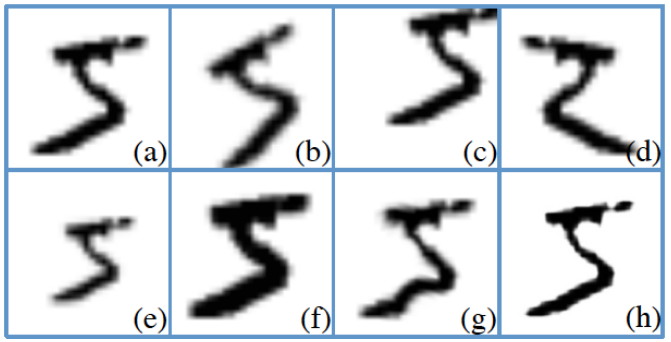
- Biologically-motivated approach:
  - Employ known biological properties of the output structures to tune the parameters of the algorithm.
- Feedback-loop for parameter tuning:
  - Enabling technology in a weakly-supervised setting.
  - Generally applicable and theoretically justified.
- Senile plague analysis:
  - Whole brain 3D volumetric estimation.
  - No manual parameter tuning resulting in less subjective pipeline.

Transformation-invariance

Idea: to employ the intrinsic properties of the input data

Transformation-invariance

- Nuisance variations in the data:
  - Natural images: illumination, camera view-point, projections;
  - Medical data: rotations, shifts, deformations, imaging artefacts.
- Non-nuisance variations:
  - Important to the solution.
  - Size of the cancerous cell,
  - Orientation of a digit (6 vs. 9).



- Goal: develop computer-vision algorithms robust to transformations defined by an expert in the field.

Transformation-invariance methods

- Transformation-invariant features (SIFT, RIFT):
  - Only allow simple transformations and simple algorithms.
- Spatial Transformer Networks:
  - Learning transformations, not incorporating the known ones.
  - Introduces additional layer of complexity.
- Multiple instance learning (multi-column networks):
  - Transformation-invariant algorithm, but not features.
- Augmentation (state of the art):
  - Relying on the power of the algorithm to learn a solution for every transformation. Requires more flexible models.

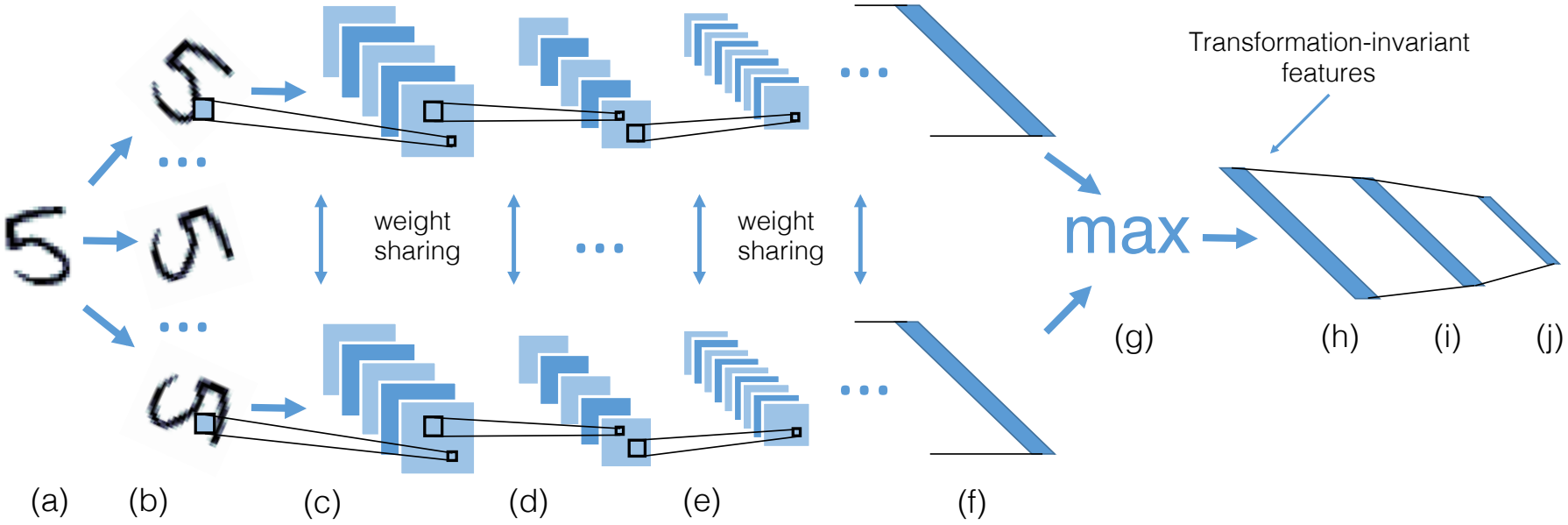
TI-pooling (transformation-invariance pooling)

- Idea:
  - Take a feature and make it transformation-invariant.
- Given:
  - a feature $f_k(x)$ of an input image x,
  - a set Φ of transformations ϕ.
- Formulate a new transformation-invariant feature $g_k(x)$:

$$g_k(x) = \max_{\phi \in \Phi} f_k(\phi(x))$$

Lemma 2. The feature of the image *x* defined above is transformation-invariant if the set Φ of all possible transformations forms a group (axioms of closure, associativity, invertibility and identity).

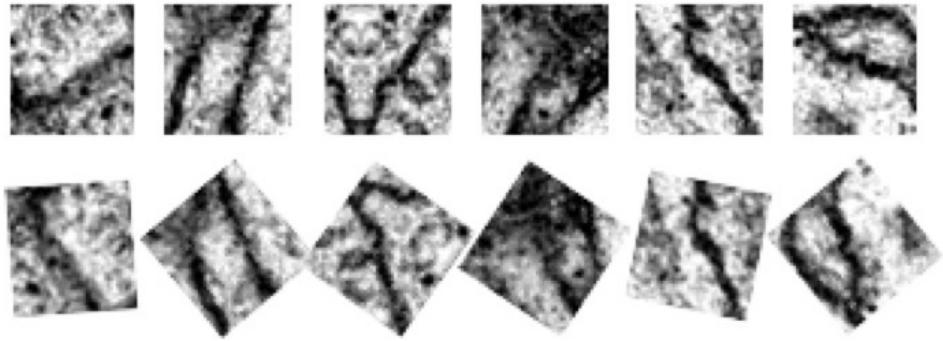
TI-pooling with convolutional neural networks



- Siamese networks share parameters => no parameter increase.
- Efficient gradient computations => back-propagation works.

TI-pooling properties

- Learns on ”canonical samples”:
  - No need to learn different features for different orientations.
  - I.e. training on the most representative samples.



Original patches are often oriented similarly for learning, this orientation is considered ***canonical****.*

- Faster convergence than augmentation.
- Same accuracy with smaller models.

TI-pooling experiments

Mnist-rot-12k

Method	Error, %
ScatNet-2	7.48
PCANet-2	7.37
TIRBM	4.2
TI-POOLING (ours)	**1.2**

Half-rotated MNIST

Method	Error, %
FCN	2.1
CNN	1.2
STN (general)	0.8
STN (affine)	0.7
TI-POOLING (ours)	0.8

Neuronal segmentation

Method	Error, %
MIL over CNN	8.9
CNN with augmentation	8.1
TI-POOLING - dropout	**7.4**
TI-POOLING + dropout	**7.0**

TI-pooling summary

- Intrinsic properties of the data/imaging:
  - Employ known nuisance variations and make the algorithm invariant to them for better data usage.

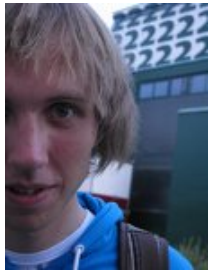
- TI-pooling:
  - Guaranteed to learn transformation-invariant features for any arbitrary set of expert-defined transformations.
  - Allows to simplify the complexity of the network.
  - Converges faster and more robustly than without TI-pooling.
  - Achieves state of the art results in multiple benchmarks.

Conclusion

- Anisotropic data:
  - SuperSlicing – an expert-mimicking approach to resolve ambiguities through correspondences between slices/frames.
  - Novel neuronal structure segmentation pipeline.
- Global biological priors:
  - Feedback-loop for parameter tuning, which employs biologically-motivated priors known from experts in the field.
  - Parameter-free non-subjective pipeline for plaque detection.
- Transformation-invariance:
  - TI-pooling to make the algorithm robust to nuisance variations in the data and make the learning process more efficient.

Gratitude



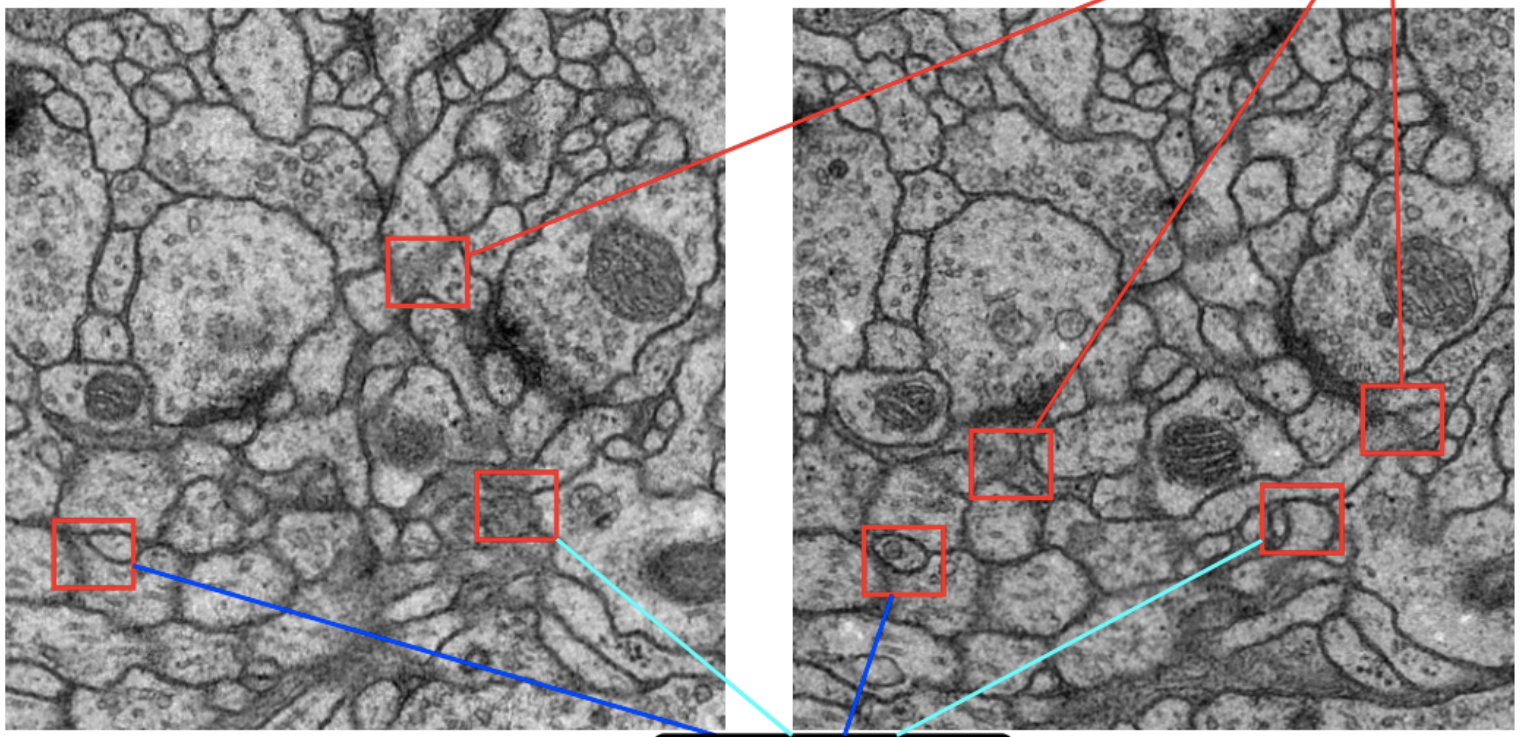




Appendix A
Anisotropic data

Anisotropic data challenges

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

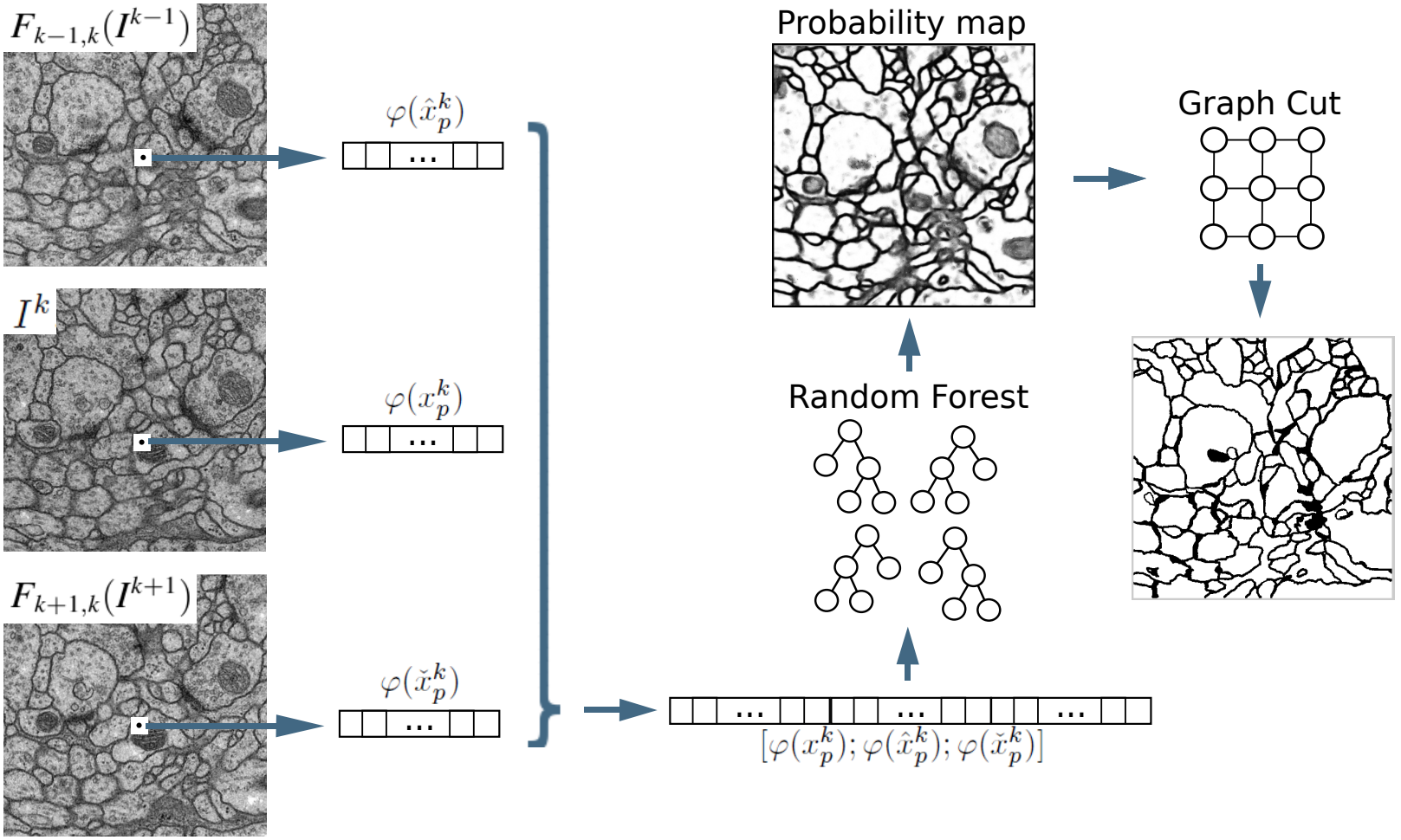


Neighboring slices are highly different one from another

Anisotropic data segmentation

- Let I^{k-1}, I^k and I^{k+1} be neighboring sections,
- x_p^k is a pixel in section k,
- $\varphi(x_p^k)$ - feature vector in pixel x_p^k.
- Suppose we are given non-linear warpings operator
  - $F_{k-1,k}$: $F_{k-1,k}(x_p^{k-1}) = \hat{x}_p^k$
  - $F_{k+1,k}$: $F_{k+1,k}(x_p^{k+1}) = \check{x}_p^k$.
- Then we can create an **extended feature vector by concatenation** $[\varphi(x_p^k); \varphi(\hat{x}_p^k); \varphi(\check{x}_p^k)]$.

Anisotropic data segmentation



Anisotropic data segmentation: results

- ISBI 2012 neuronal segmentation challenge.
  - 3.6% and 6.4% more accurate when using dense correspondence.
  - Second-best result overall, orders of magnitude faster than competitors.

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

SIFT-flow functional

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

$s(x_p)$ 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.

Features

Features over different windows with different parameters:

- 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.

Graph cut segmentation

$Y = \{y_p\}$ - binary labelling is found by minimizing the energy with max-flow/min-cut computation:

$$E(Y) = \sum_{p=1}^{N} E_{rf}(y_p) + \lambda_s \sum_{(p,q)\in\epsilon} E_s(y_p, y_q) + \lambda_{gf} \sum_{p=1}^{N} E_{gf}(y_p) + \lambda_{gc} \sum_{(p,q)\in\epsilon} E_{gc}(y_p, y_q), \quad (2)$$

$E_{rf}(y_p)$ 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.

Graph cut smoothing term

Penalizes for discontinuities in the segmentation for neighbored pixels of similar intensities:

$$E_s(y_p, y_q) = \exp\left(-\frac{(i(x_p) - i(x_q))^2}{2\sigma_s^2}\right)\frac{\delta(y_p, y_q)}{D(x_p, x_q)}, \qquad (3)$$

where $\delta(y_p, y_q)$ is a Kronecker function that equals 0 if $y_p = y_q$ and 1 otherwise.

Graph cut gradient flux term

$$E_{gf}(y_p) = \begin{cases} \max(0, F(x_p)) & \text{if } y_p = 1 \\ -\min(0, F(x_p)) & \text{if } y_p = 0, \end{cases} \quad (4)$$

where $F(x_p)$ denotes a gradient flux,
$F(x_p) = \sum_{x_q:(x_p,x_q)\in\epsilon} < u_{x_p,x_q}, v_{x_p} >$, u_{x_p,x_q} represents a unit vector pointing from pixel x_p to the neighboring pixel x_q and vector v_{x_p} corresponds to the gradient vector at pixel x_p

Graph cut good continuation term

$$E_{gc}(y_p, y_q) = | < v_{x_p}, u_{x_p,x_q} > | \exp\left(-\frac{(i(x_p) - i_m)^2}{2\sigma_{gc}^2}\right) \frac{\delta_{\rightarrow}(y_p, y_q)}{D(x_p, x_q)}, \quad (5)$$

The variable i_m encodes the average gray value of membrane pixels and σ_{gc} is estimated as the variance of these gray values. The factor $\delta_{\rightarrow}(y_p, y_q) = 1$ for $y_p = 1, y_q = 0$ and equals 0 for all other cases.

SuperSlicing algorithm

- Notation
  - Y^n — observed frames, $n \in [1, \ldots, N]$,
  - y_p^n — pixel p of the frame Y^n,
  - $i(y_p^n)$ — the intensity of pixel y_p^n,
  - $\epsilon(x_p^n)$ — a set of neighbors of pixel x_p^n,
  - Ω — a set of given correspondences,
  - $X^{n,l}$ — hidden sub-frames decomposition of Y^n, $l \in [1, \ldots, L]$.

SuperSlicing algorithm

- Energy minimization problem

An average of the hidden frames should give observed frame

$$E(X^{n,1},\dots,X^{n,L}) = \sum_{y\in Y^n}\left(i(y) - \frac{1}{L}\sum_{l=1}^{L} i(x_p^{n,l})\right)^2$$

Smoothness across all the correspondences

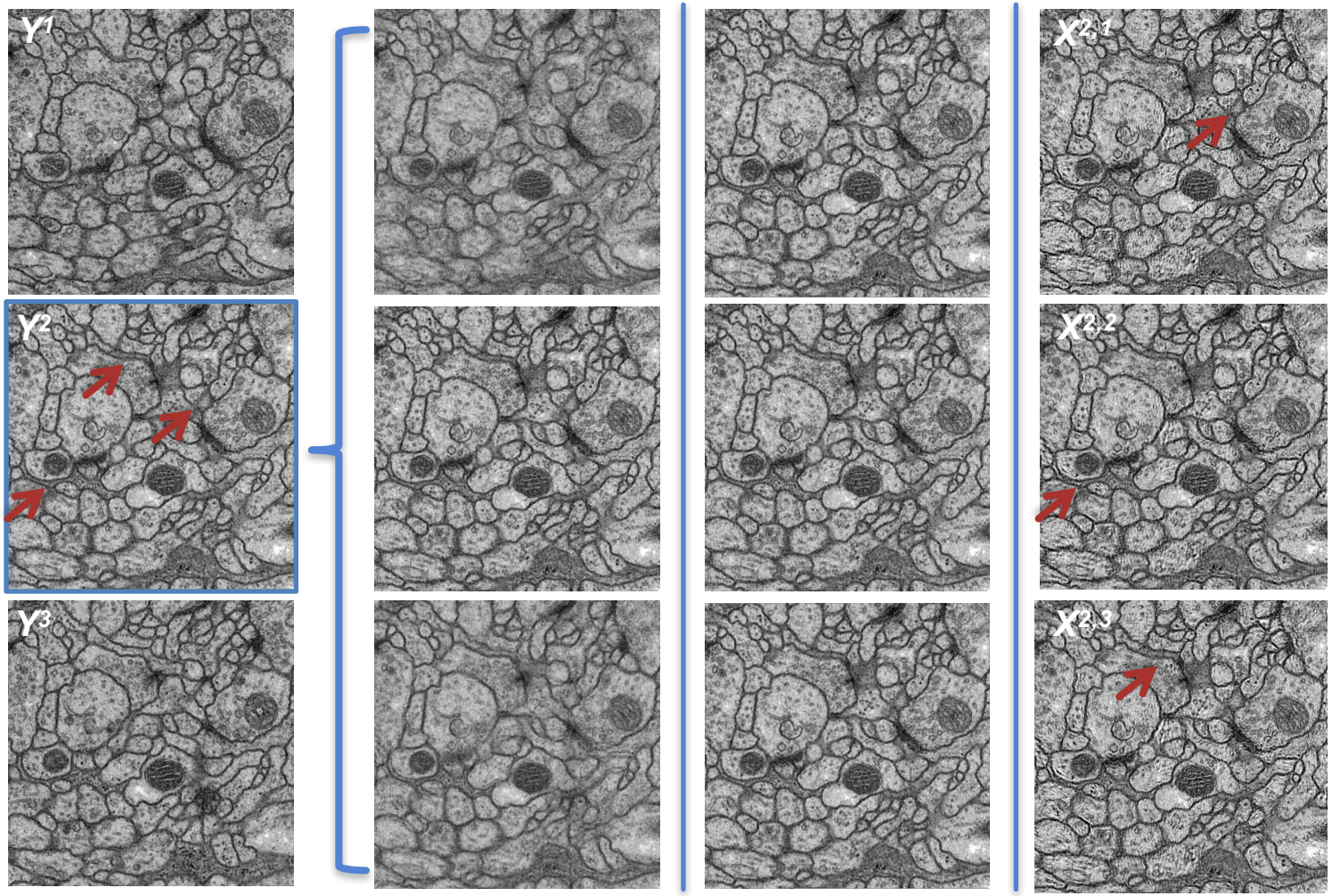
$$+\lambda \sum_{(\hat{x}_p^{n,l},\hat{x}_q^{n,l+1})\in\Omega}\left(\sum_{x\in\epsilon(\hat{x}_p^{n,l})} w(x,\hat{x}_p^{n,l})i(x) - \sum_{x\in\epsilon(\hat{x}_q^{n,l+1})} w(x,\hat{x}_q^{n,l+1})i(x)\right)^2$$

Smoothness within the frame

$$+\gamma \sum_{\substack{x_p^{n,l};x_q^{n,l}\in\epsilon(x_p^{n,l})\\ l=1,\dots,L}}\left(i(x_p^{n,l}) - i(x_q^{n,l})\right)^2$$

Quadratic programming: global optimum

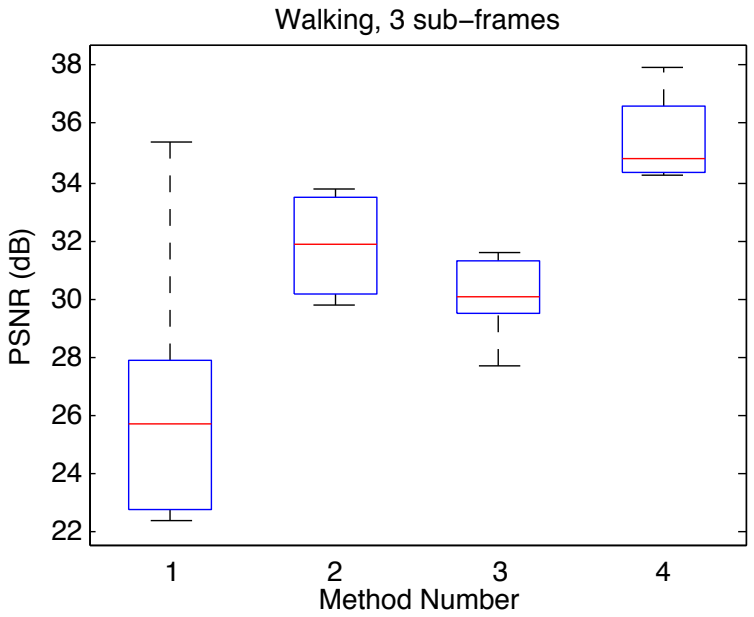
SuperSlicing ssTEM experiments

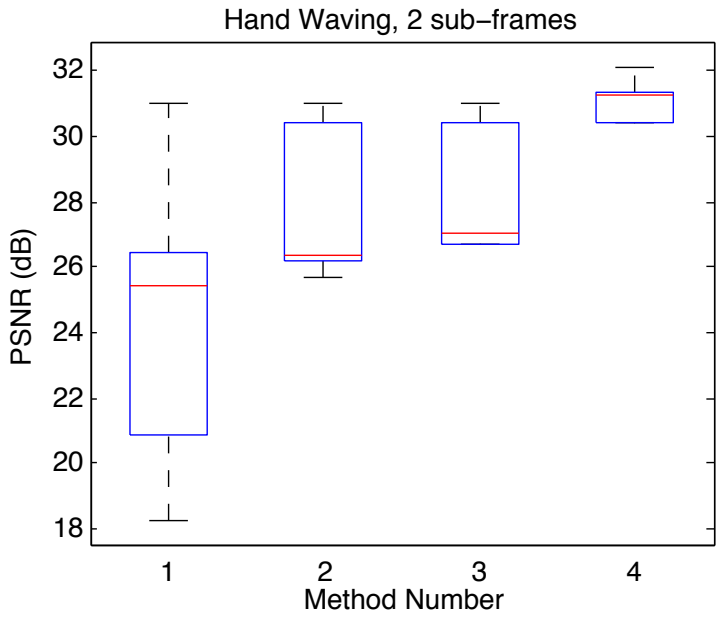


SuperSlicing video experiments

Synthetic frames | *Linear interpolation* | *Optical Flow warping* | *Ours* | *Original frames*

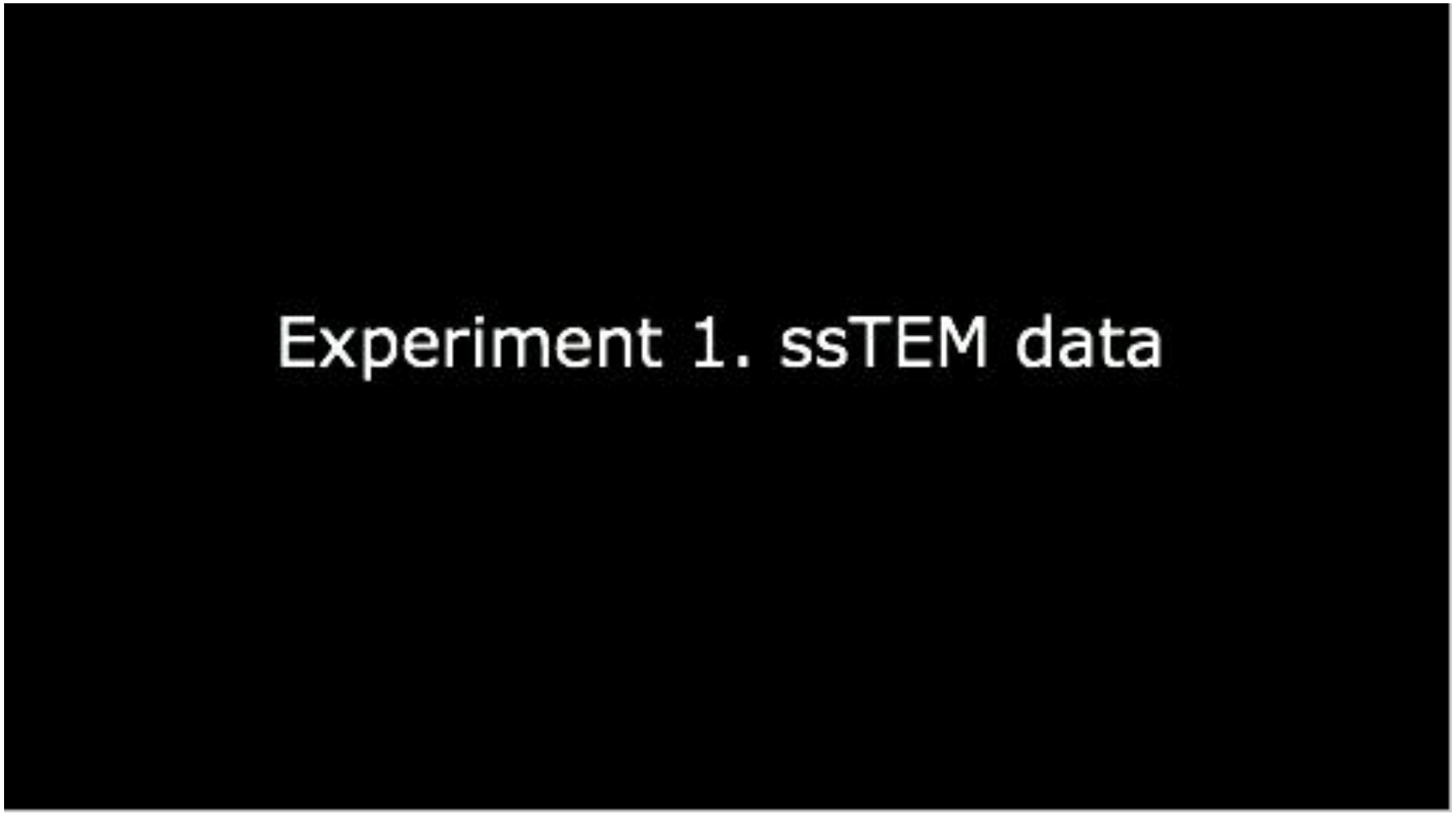
SuperSlicing video experiments





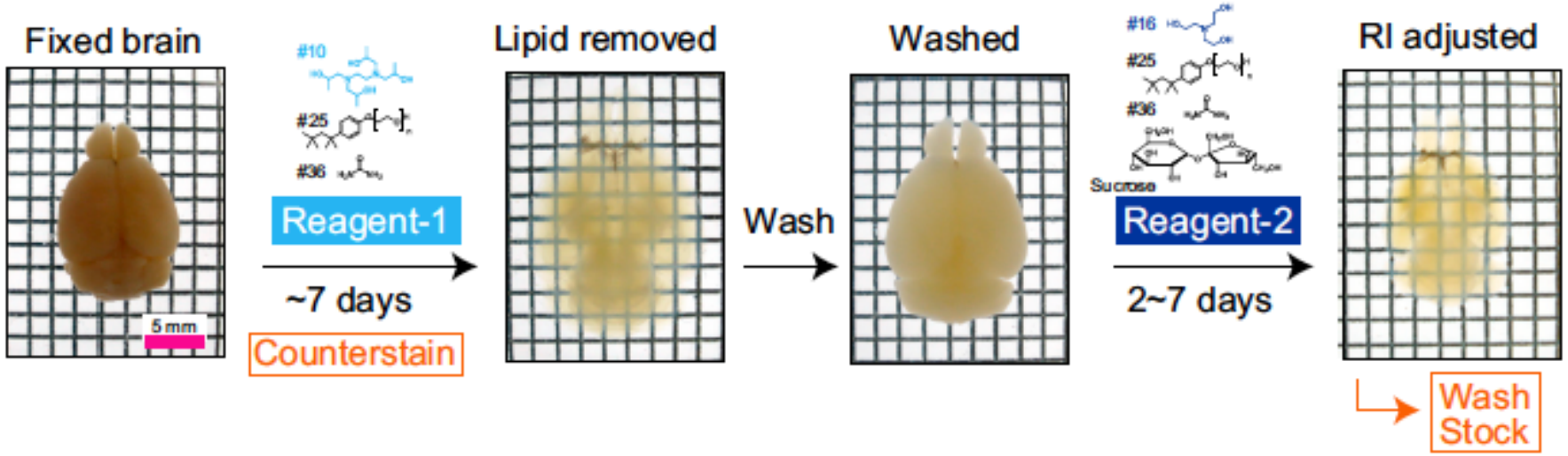
Significantly more consistent reconstruction!

SuperSlicing experiments



Appendix B
Cleared brain project

Clarity / Crystal imaging



- Brains are made transparent by removing the lipids from it
- Some structures can be highlighted with special markers
- UZH group made this process ~20 times faster

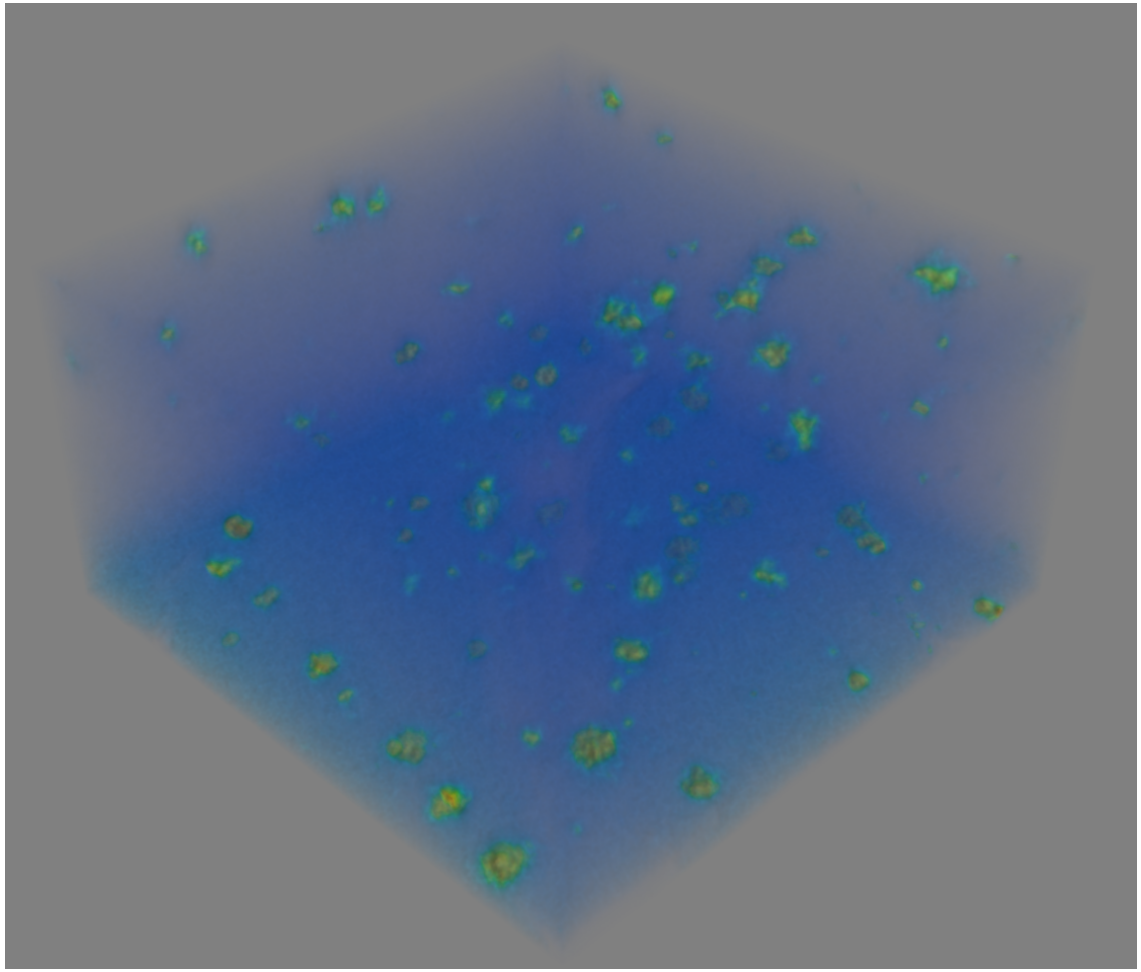
Datasets we have

- Focus on senile plaques: the further the Alzheimer decease progressed – the more plaques there are

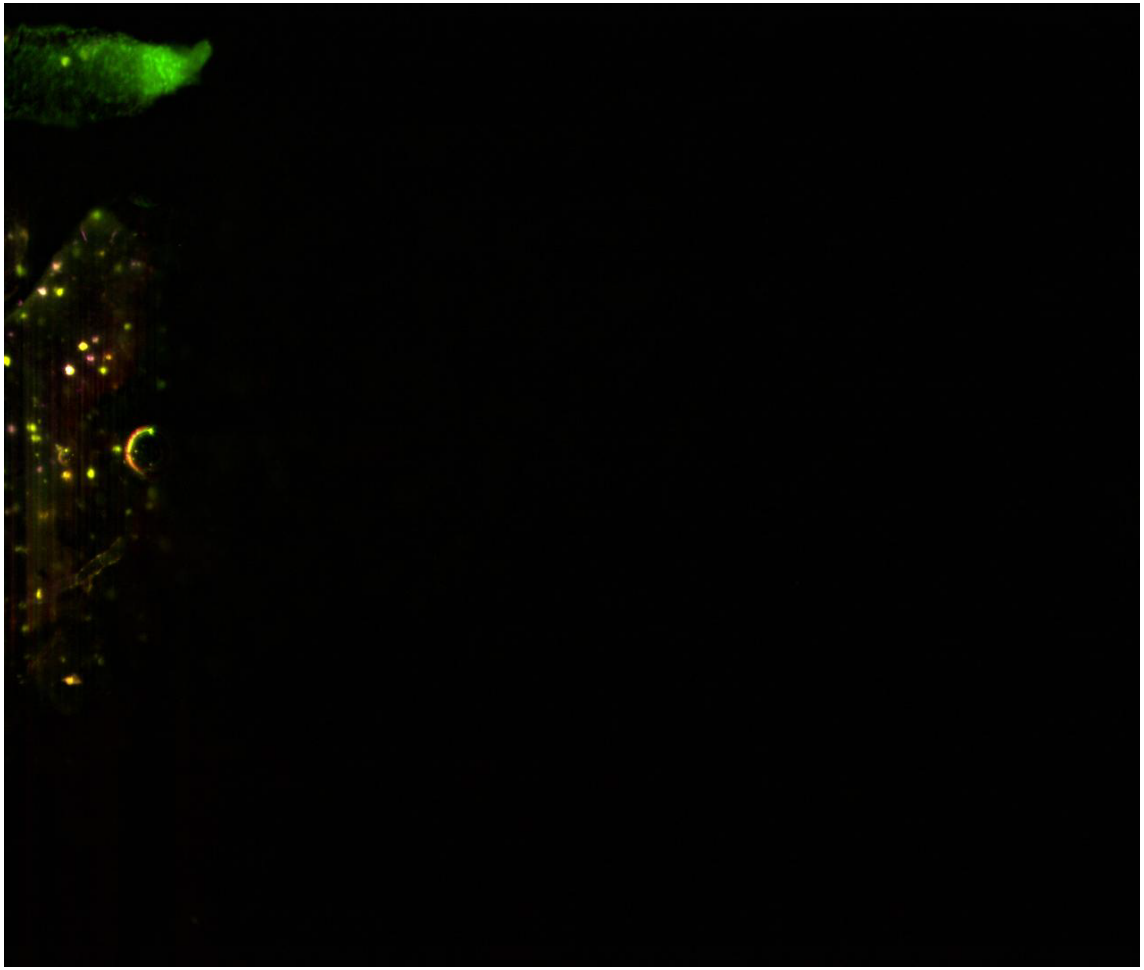
- Up to whole brain mouse optical scans
- Resolution from 2x up to 20x
- Image sizes: e.g. 2287 x 1542 x 210 x 5

- Task: count plaques automatically to estimate Alzheimer progression (to evaluate possible treatments)

Datasets we have (20x, small, slow)



Datasets we have (5x, ¼ brain, slow)



Datasets we have (2x, whole brain, fast)

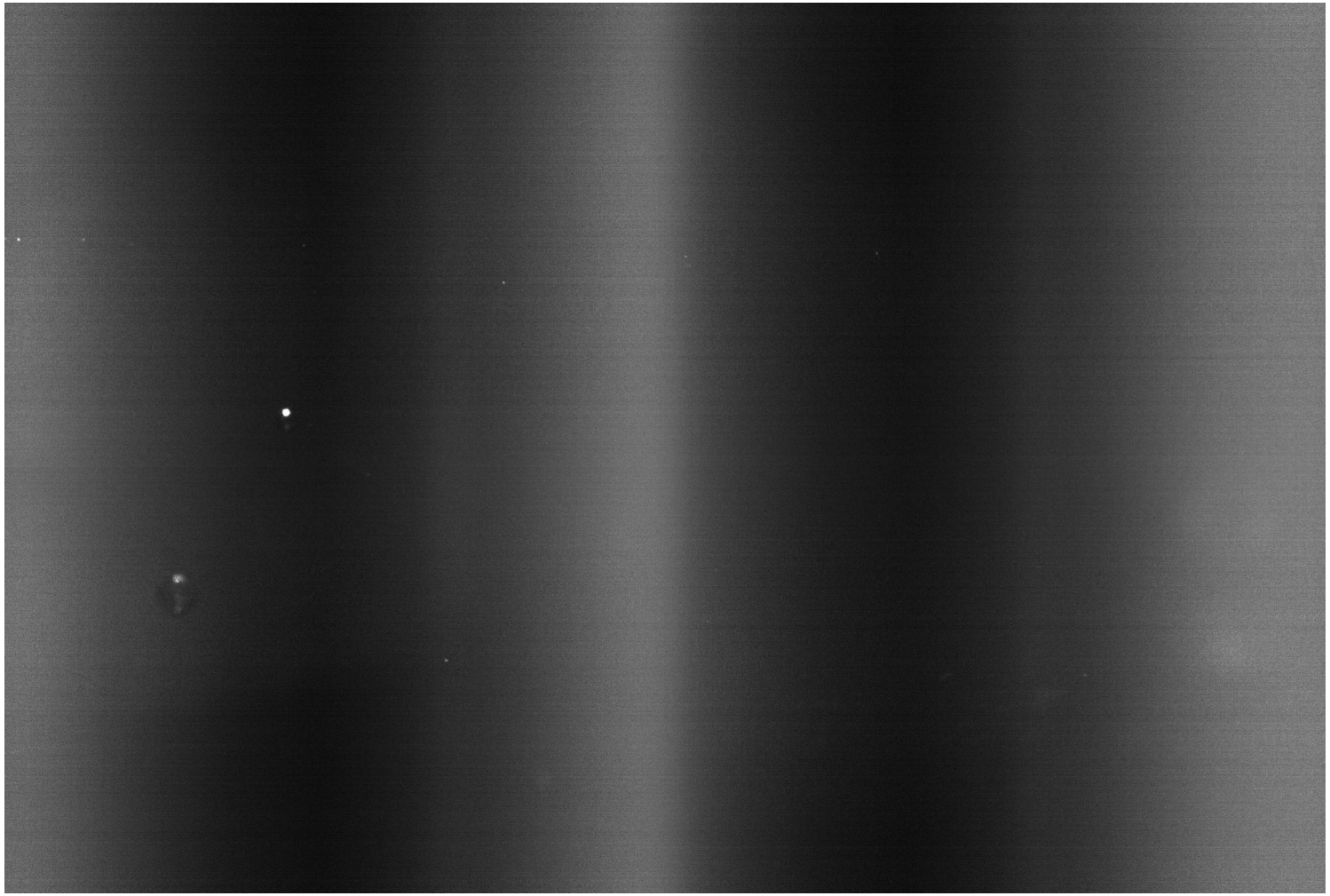


Image processing 20x

- Small region, almost no artifacts, some background noise, uniform across the volume

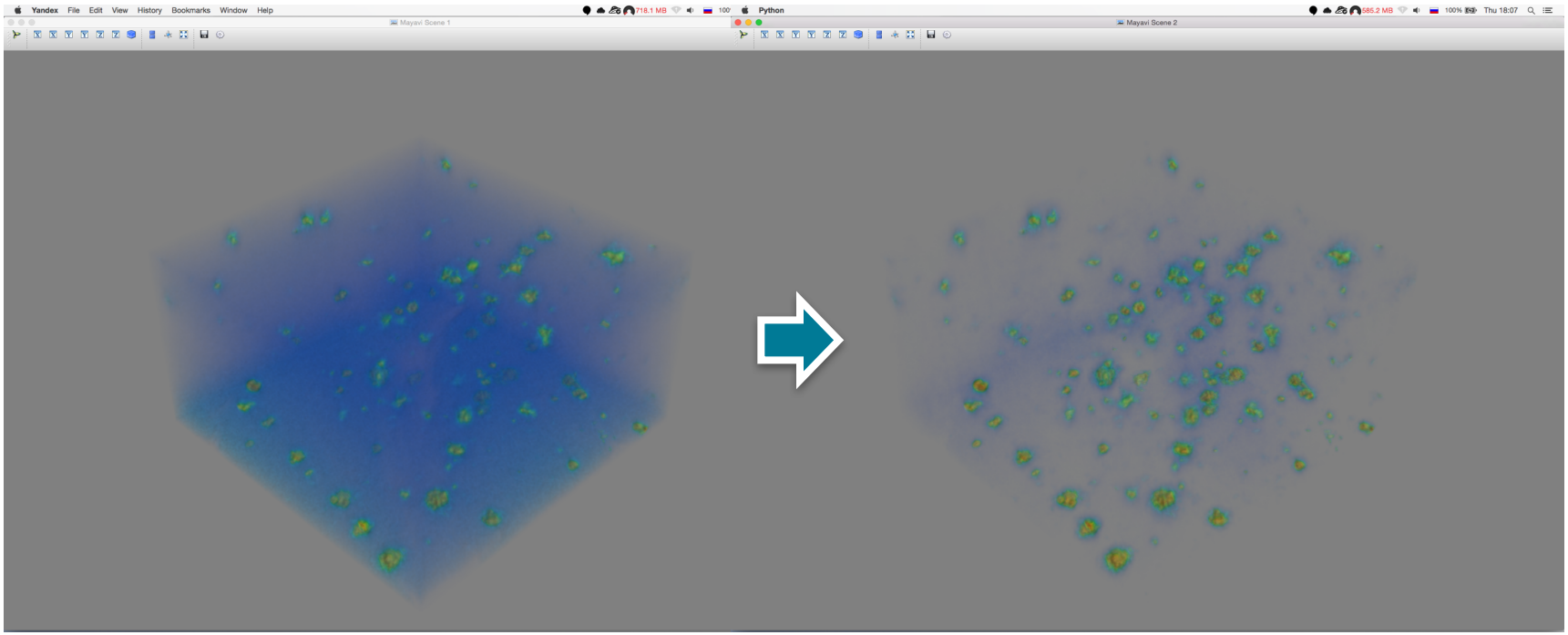
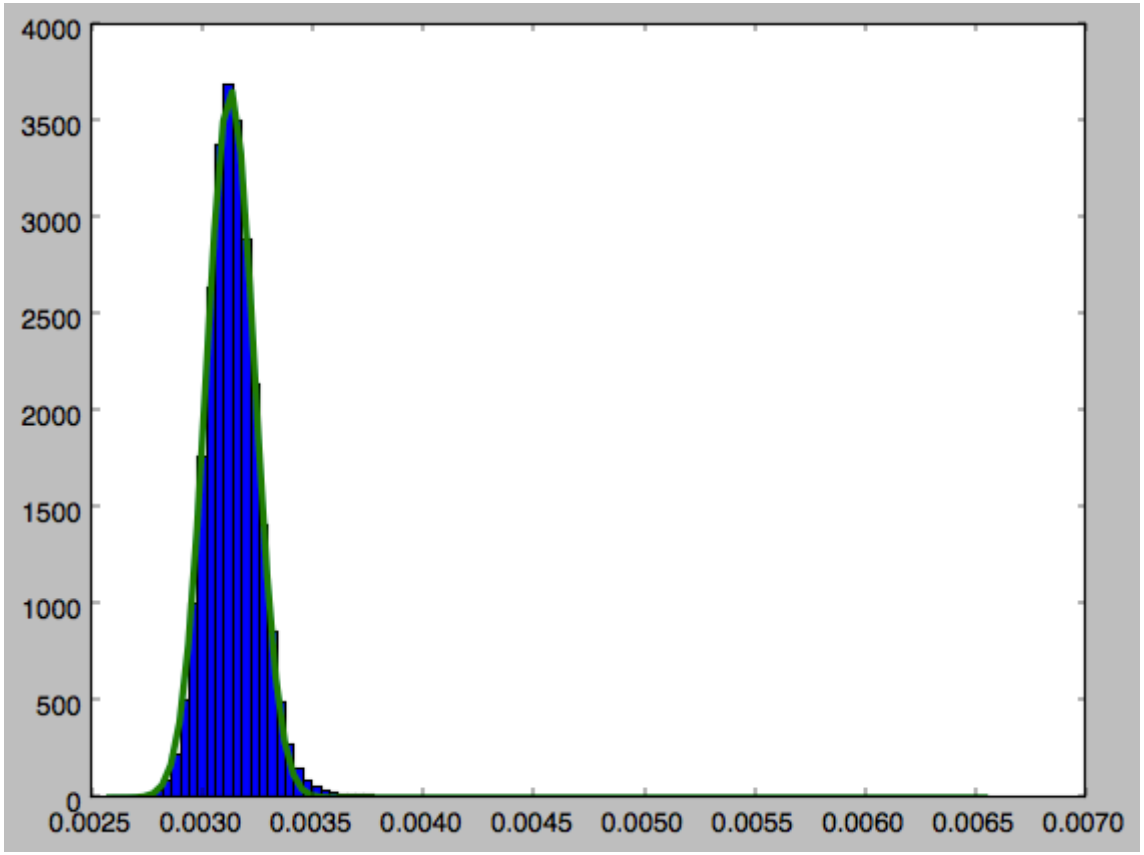


Image processing 20x



Voxel intensity distribution

Image processing 20x

- Background-foreground modeling

- Gaussian + Laplace mixture
  - EM-algorithm
- Gaussian (robust) + everything else
  - μ = sample median
  - $\Sigma = (q_{84\%} - q_{16\%}) / 2$
- Second one is simpler and works better

Image processing 20x

- Gaussian kernel density estimation

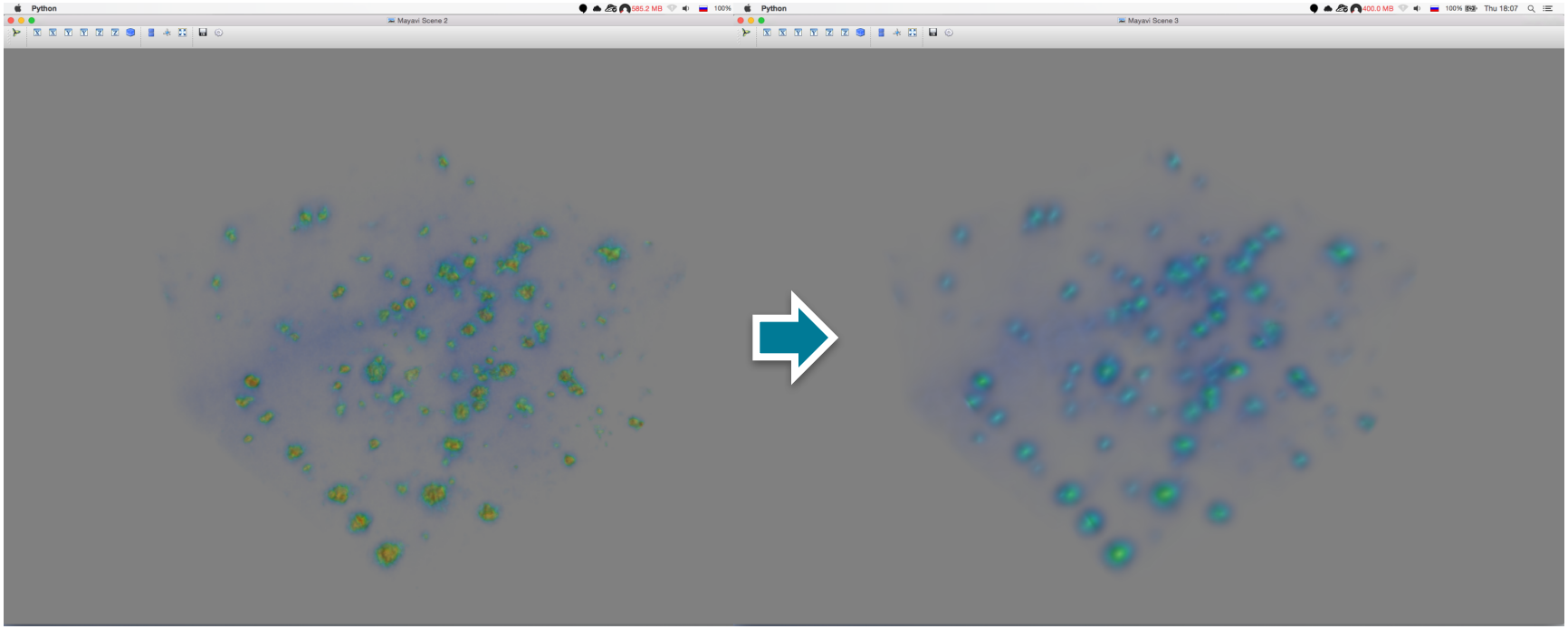


Image processing 20x

- Finding local peaks, counting, size estimation

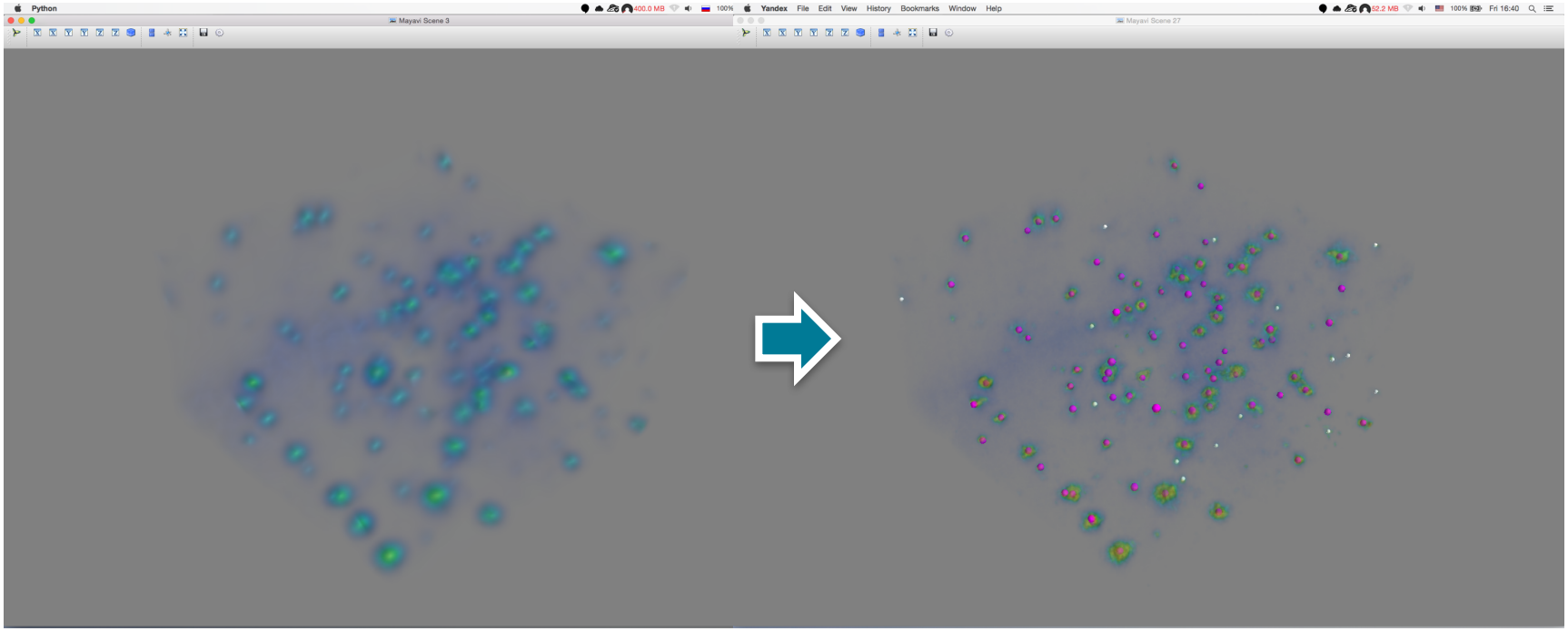


Image processing 20x

- Resulting size histogram is biologically accurate

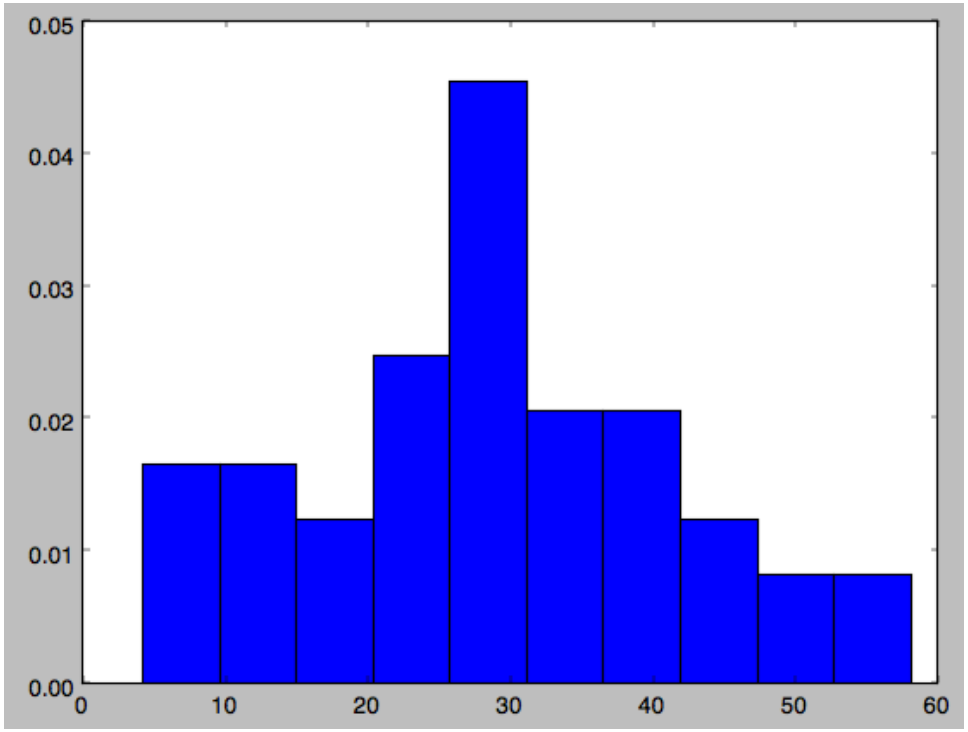


Image processing 2x

- Lots of variation within one image – no global statistics

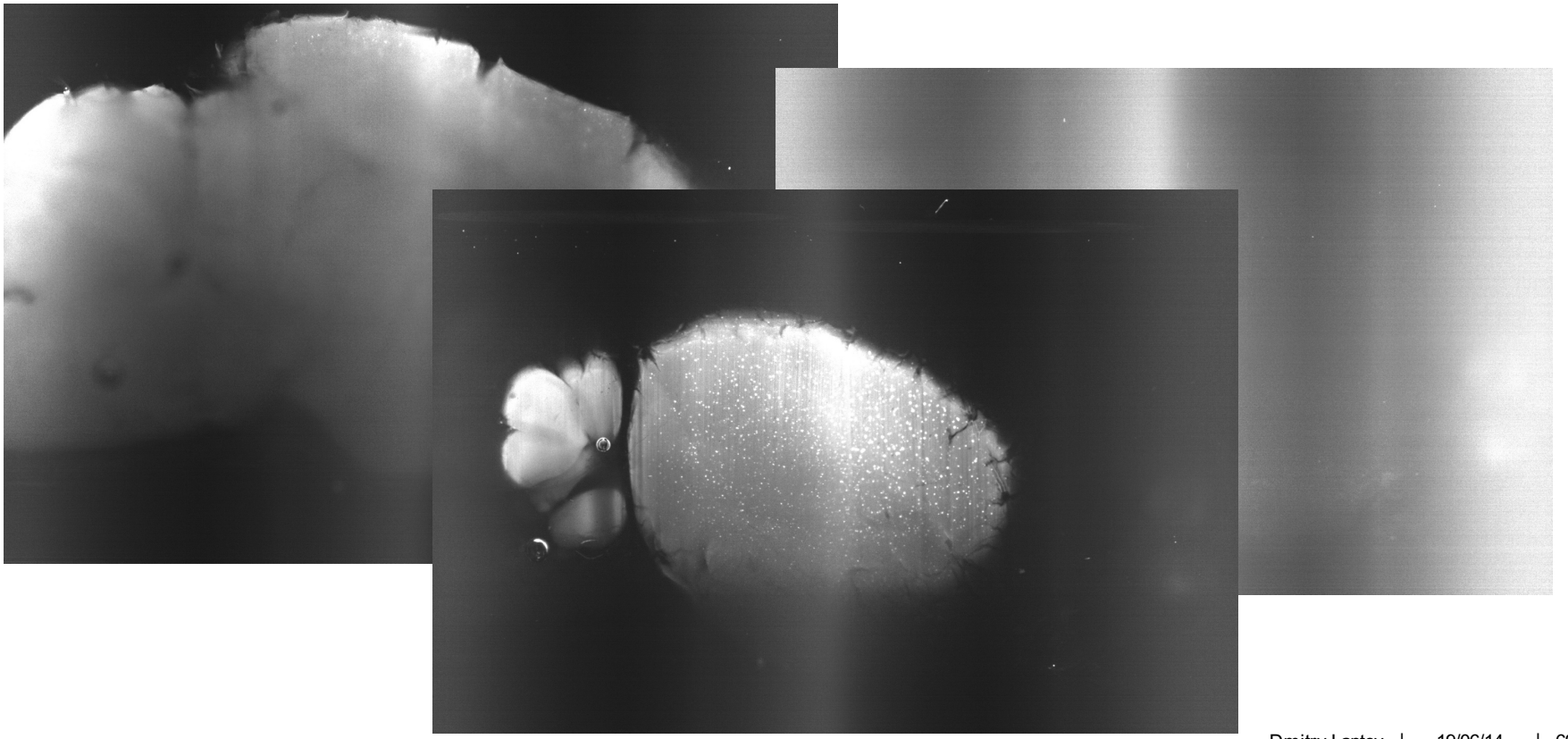


Image processing 2x

- Local statistics: local percentile within a disk
- E.g.: > 90% local percentile shows plaques
- Favor regions of almost-flat intensity

- 60% local percentile is a background
- See intensity comparing to the background:
- E.g.: “logit(fg / bg – 1)“ shows plaques probability

- That all is to be done in 3d

Image processing 2x: original

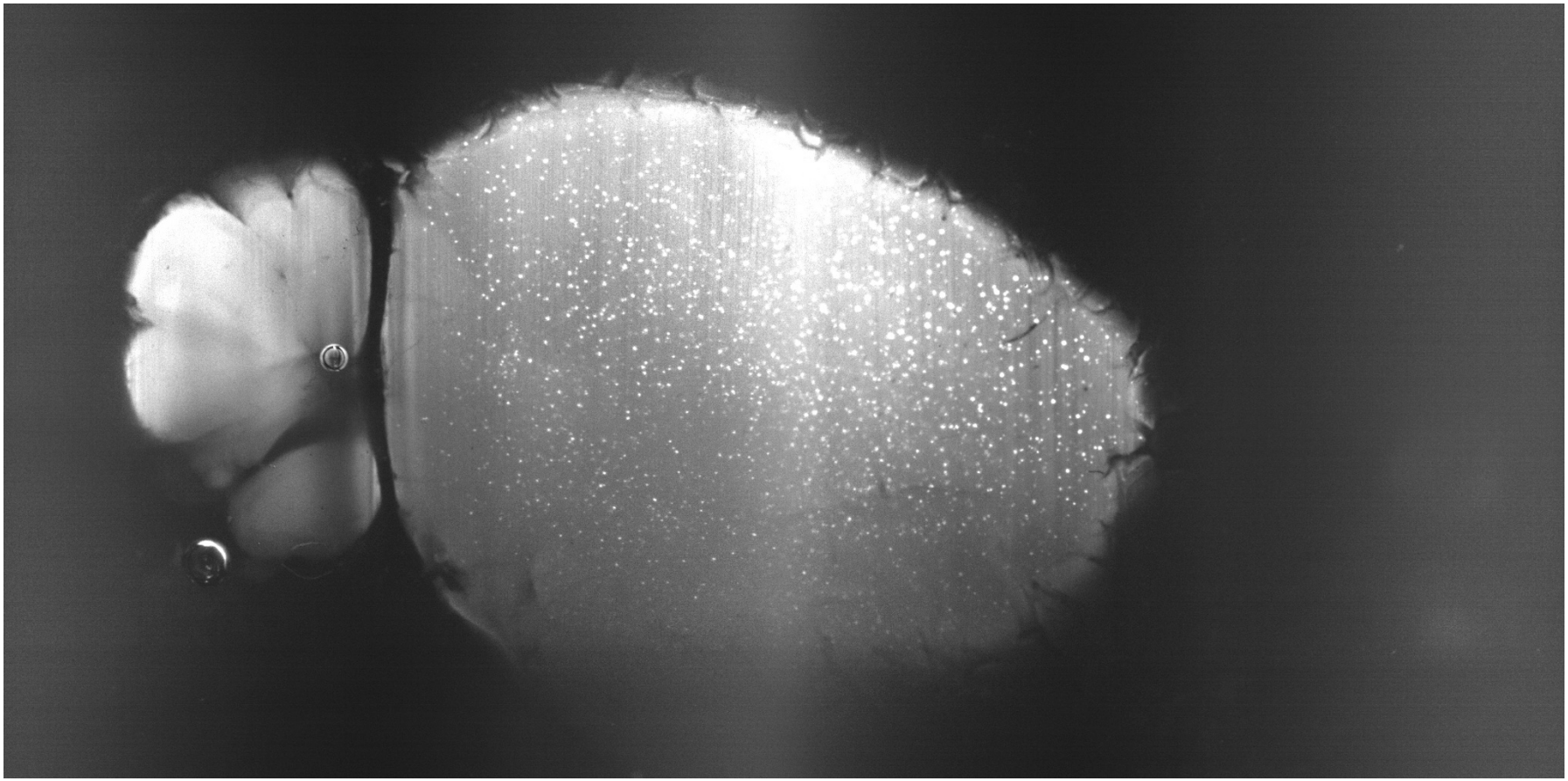


Image processing 2x: background

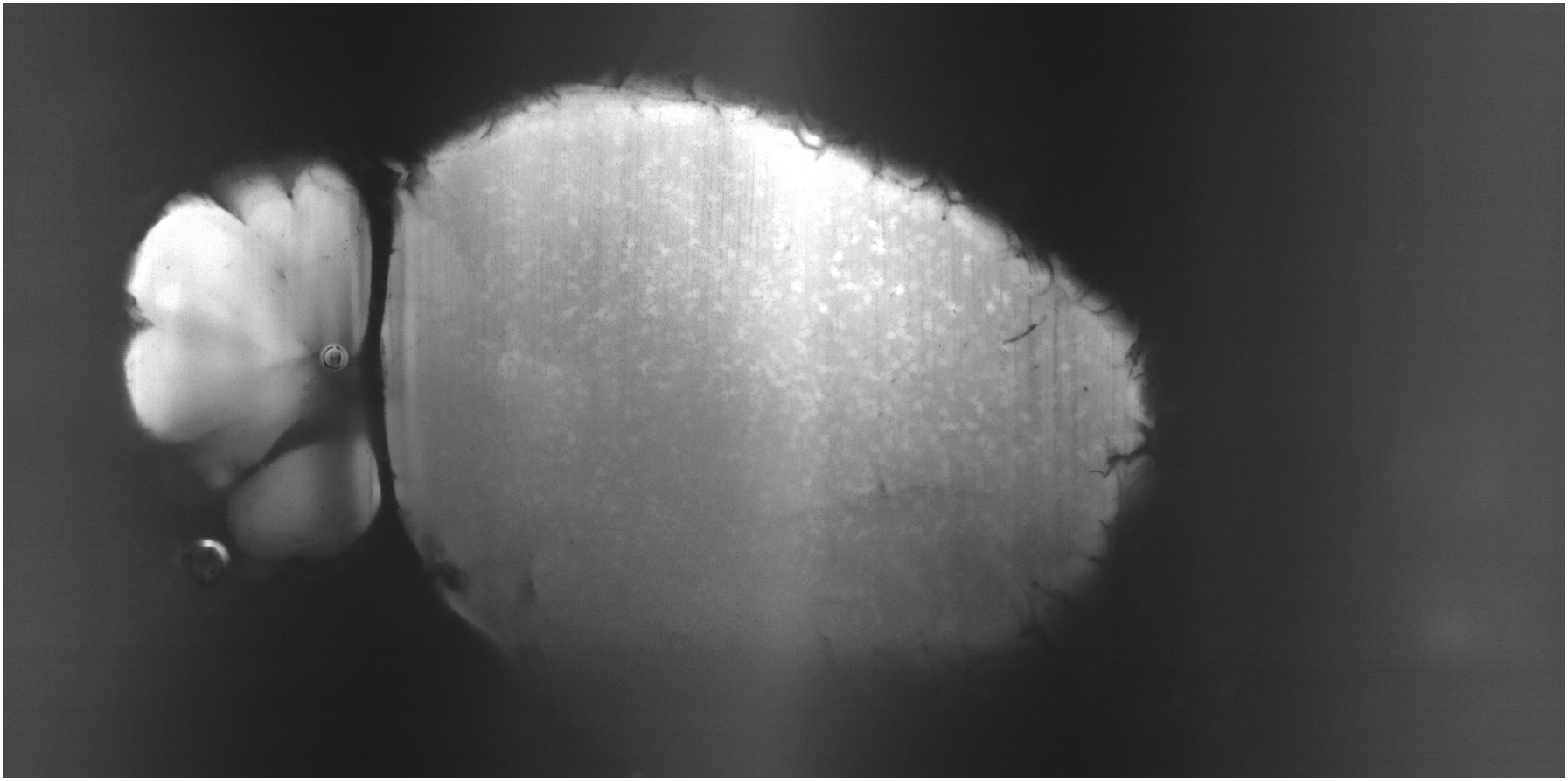


Image processing 2x: above 90%

Image processing 2x: plaques

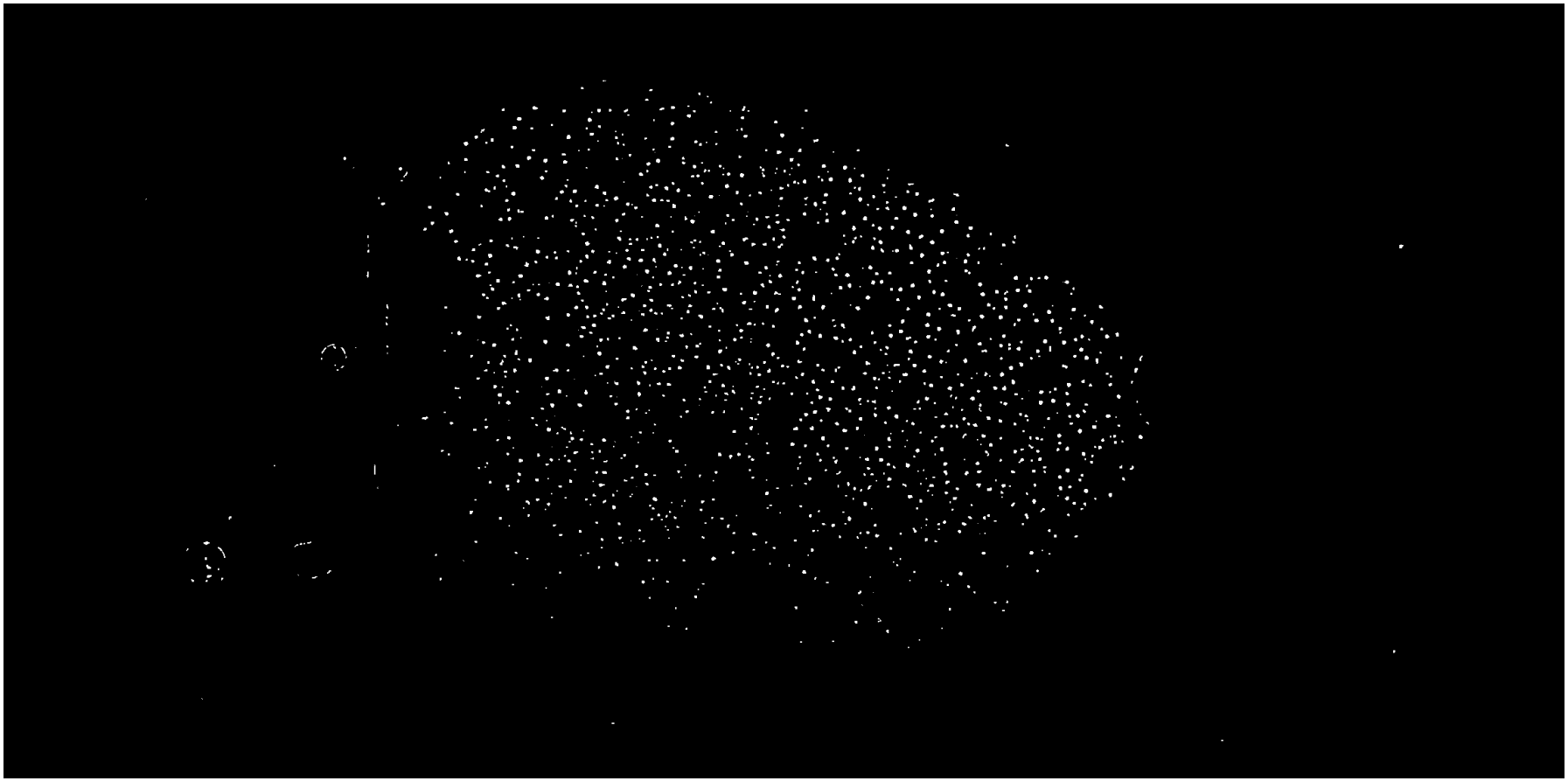


Image processing 2x: combining channels

- Different channels contain similar information
- Apply MRF for every voxel from all the channels

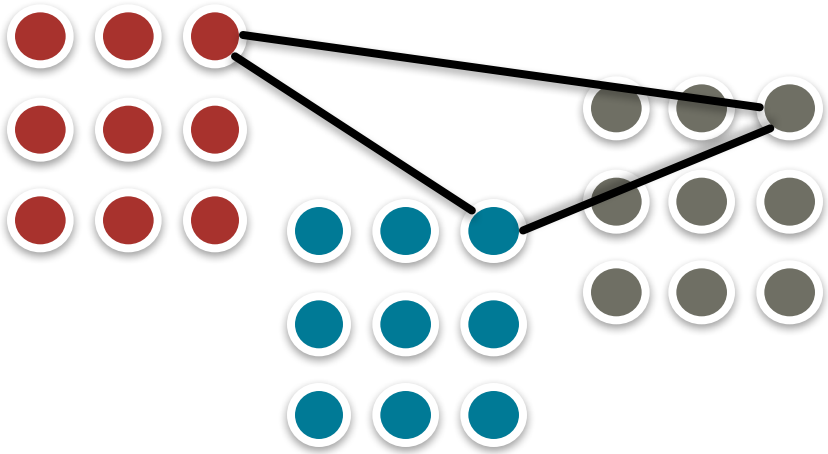


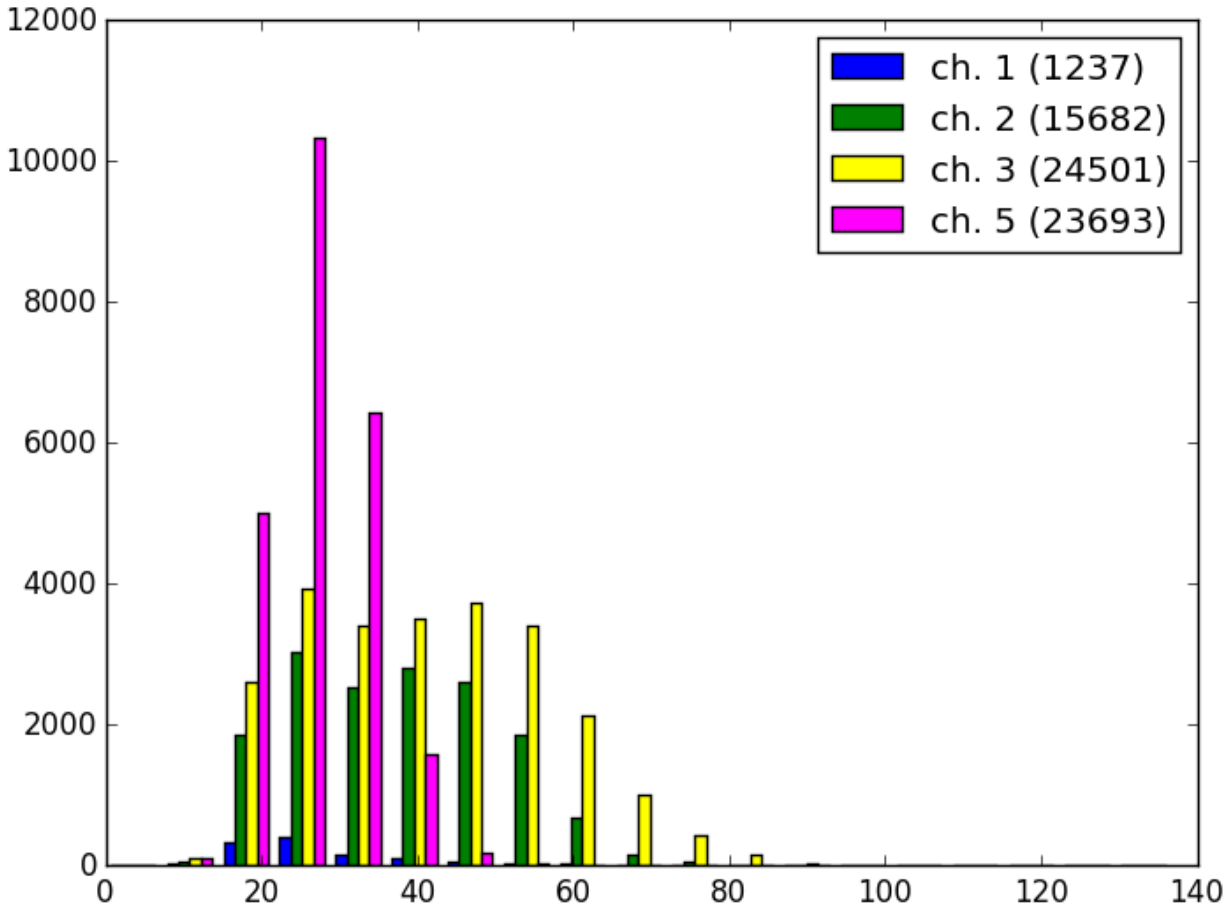
Image processing 2x

- Find all connected components in a binary image
- Filter out regions smaller than 10 micrometer in radius
- Visualize the plaques and all the statistics
- Repeat for different channels
- Repeat for different brains
- For verification use:
  - Plaque size distribution is know
  - Older brains should have more and larger plaques
  - Some channels are better than others
  - Best tool: your eyes and eyes of your collaborators

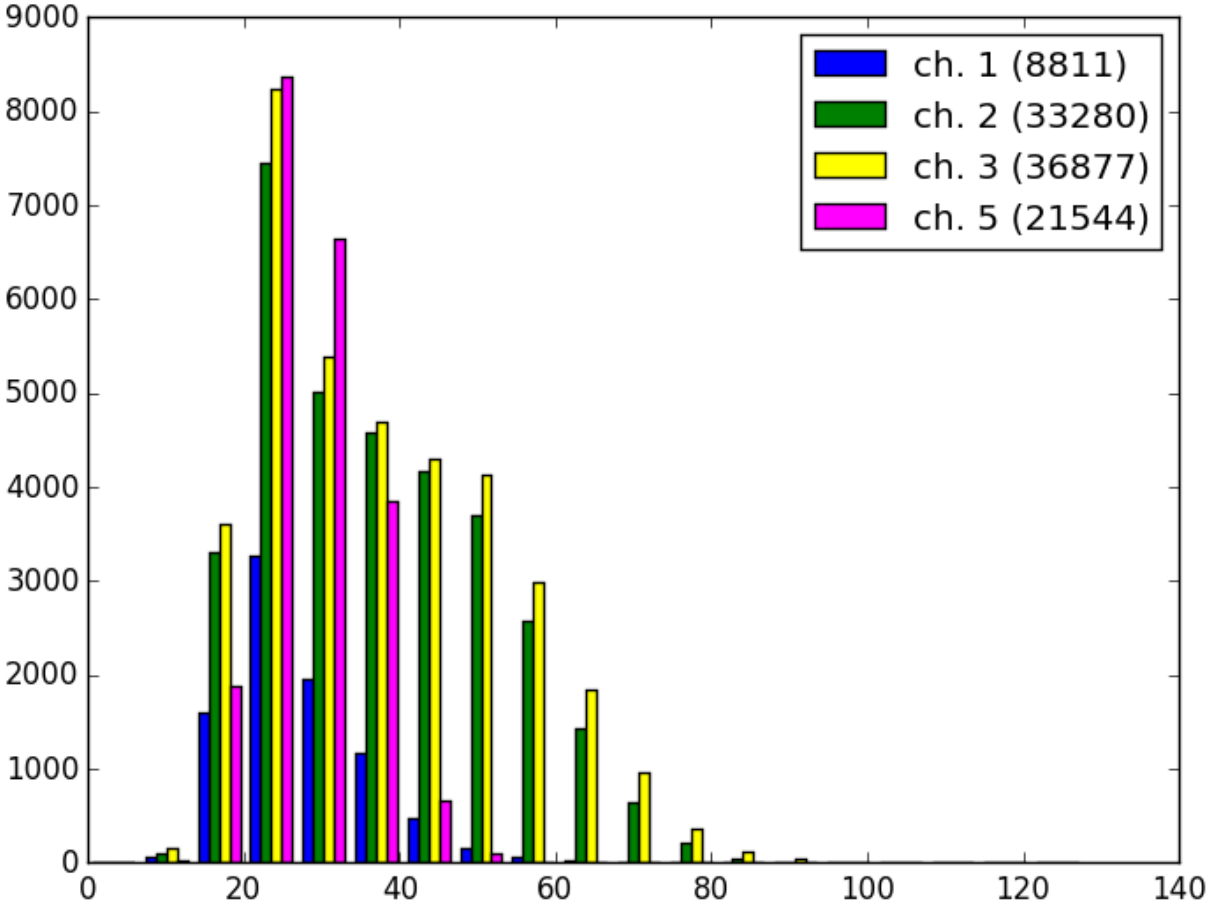
Image processing 2x

- Live result demo

Results: young brain histogram



Results: old brain histogram



Results: old vs. young brain comparison

	ch 1	ch 2	ch 3	ch 4
brain 2	3015	17259	27220	27788
brain 3	4582	19904	27635	15601
brain 4	2932	13824	21558	15657
brain 5	2254	12496	23229	23673
brain 6	3655	27290	32986	22842
brain 7	1826	22586	35925	38744
brain 8	10208	35268	40272	23206
brain 9	9844	69329	89953	87275

- Red color shows larger number of plaques, green color – smaller.
- Brains 2-5 are young, brains 6-9 are older.
- Channels 2 and 3 are similar.

Feedback-loop algorithm

Algorithm 2 Feedback-loop with binary search

Require: algorithm $\mathcal{A}$, parameter boundaries $\theta_{\min}, \theta_{\max}$,
property f, expected property value E, tolerance level ϵ.
repeat
  $\theta := \frac{1}{2}(\theta_{\min} + \theta_{\max})$
  if $\big(f(\mathcal{A}(\theta_{\min})) - E\big)\big(f(\mathcal{A}(\theta)) - E\big)$ **then**
    $\theta_{\min} := \theta$
  else
    $\theta_{\max} := \theta$
  end if
until $||f(\mathcal{A}(\theta)) - E||_2^2 < \epsilon$
return θ

Feedback-loop analysis

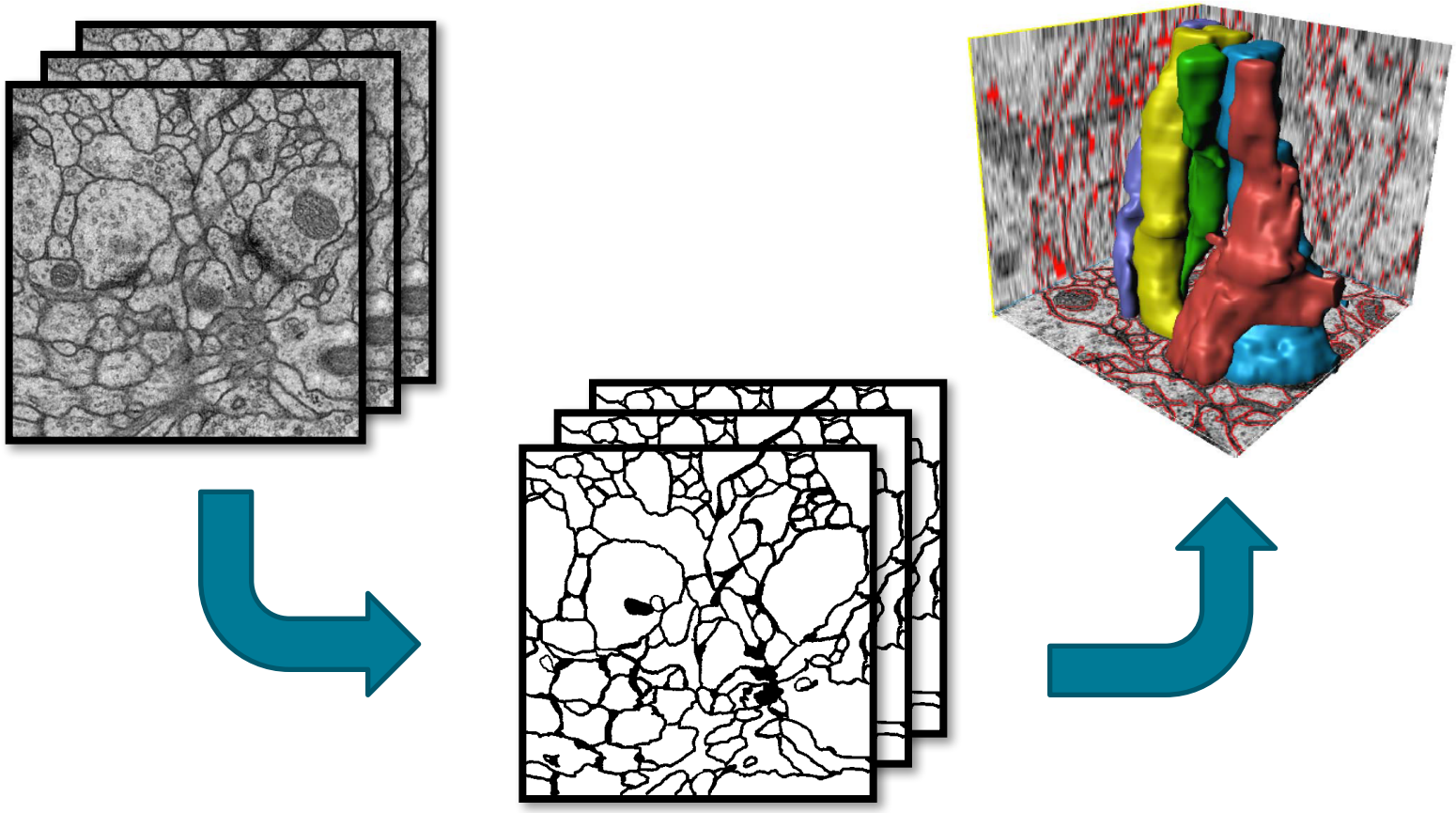
Method	Error
Trained on V_i, estimated on V_i	0.8% (training)
Trained on V_j, estimated on V_j	1.4% (training)
Trained on V_j, estimated on V_i	18% (validation)
Trained on V_i, estimated on V_j	34% (validation)
Ours estimated on V_i	12%
Ours estimated on V_j	15%

Table 3.1: Comparison of the proposed tuning-free approach with cross-validation parameter training. By not overfitting to one specific section, the proposed method achieves better generalization.

- More than three quarters of all the plaques are developing in neocortex.
- Young brains contain respectively 26859, 19602 and 34152 plaques.
- Old brains respectively 41924, 57292 and 50136 plaques.
- This result, together with manual result inspection, proves the reasonability.

Appendix C
Convolutional Decision Trees

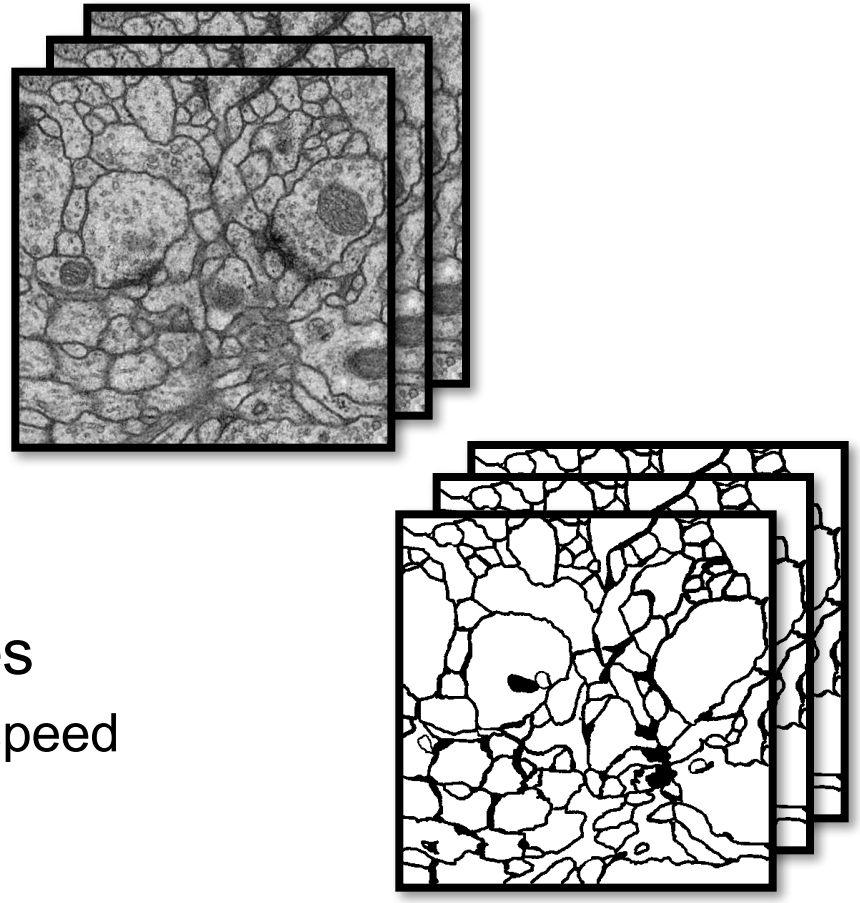
Connectomics*



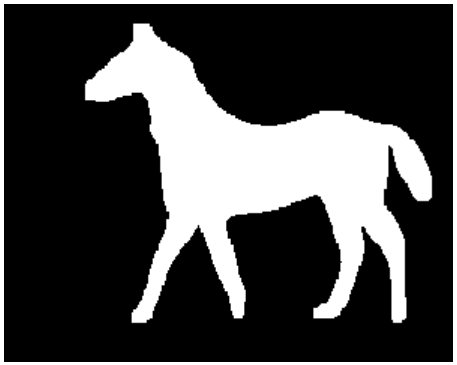
* Reconstruction and study of Connectome: a map of neuron connections

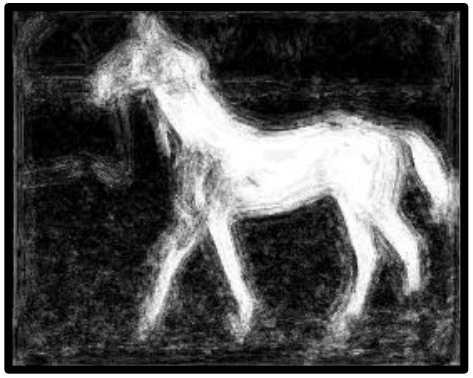
Connectomics

- Hard to automate
  - On a whole-brain scale
    it is simply necessary
- Good techniques are:
  - Require huge training sets
  - Very slow to train
  - Infeasible in one CPU
- Convolutional Decision Trees
  - Trade-off between quality and speed
  - Making Connectomics fast



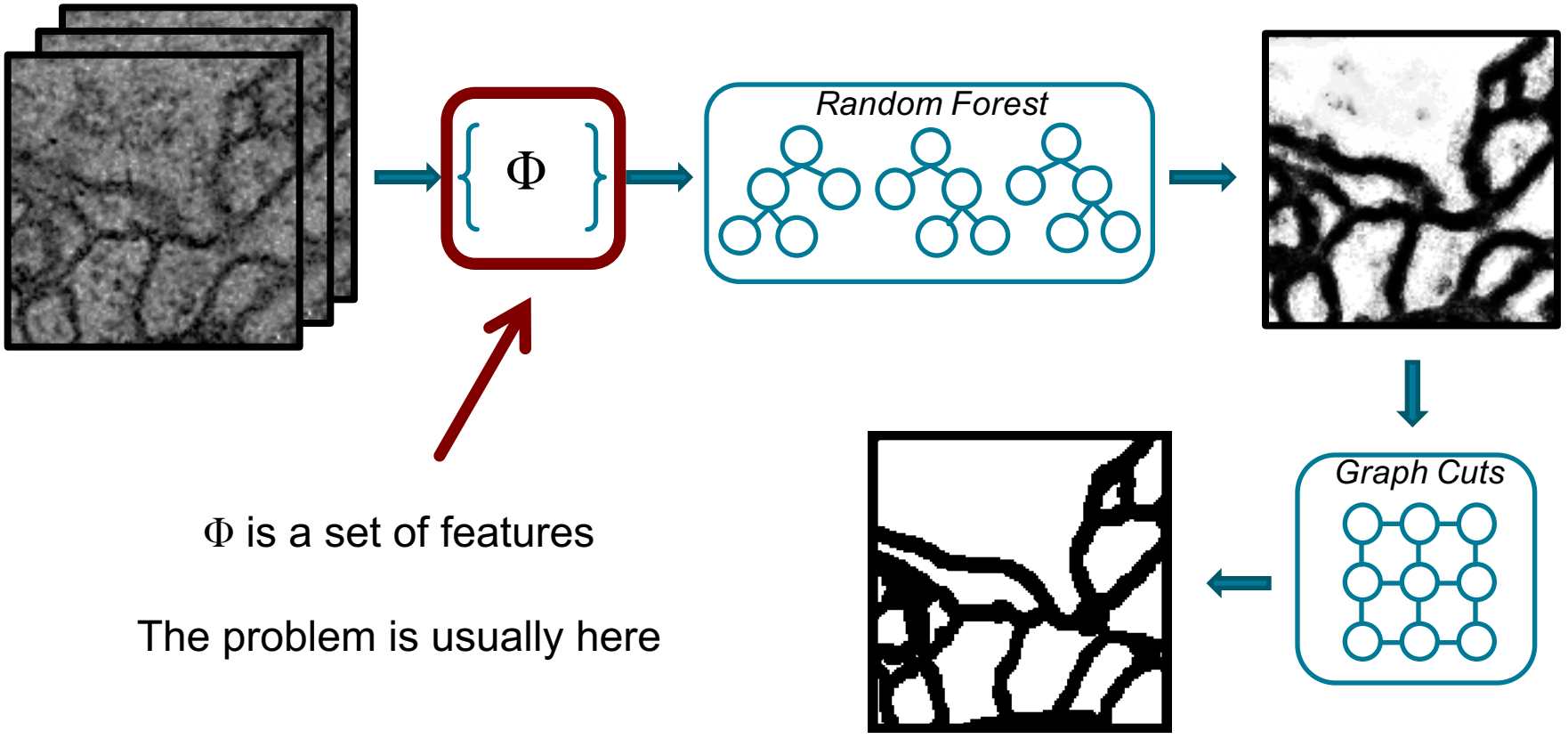
Other Computer Vision tasks





Segmentation tasks usually start with computing per-pixel probabilities

General pipeline

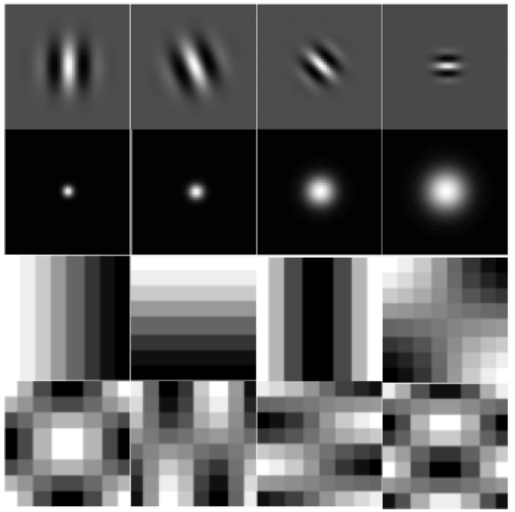


Φ is a set of features

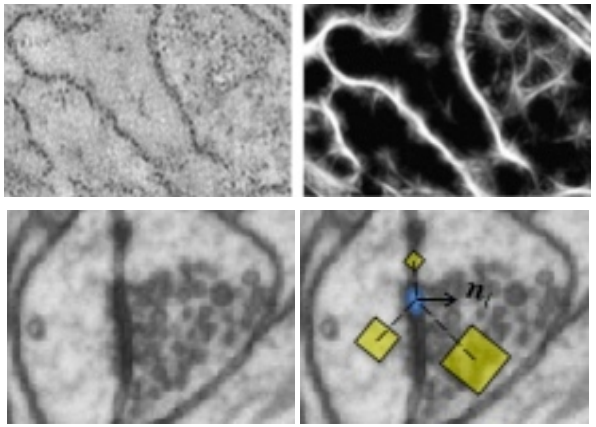
The problem is usually here

Static predefined features

- Generic features
  - HoG features,
  - Gaussian blur,
  - Sobel filters,
  - SIFT features



- Domain-specifically designed
  - Line Filter Transform
    for blood vessel segmentation,
  - Context Cue features
    for synapse detection



Feature learning

- Unsupervised feature learning
  - Bag of Visual Words,
  - Sparse Coding,
  - Autoencoders

+ no feature design
+ data-specific features
– not task-specific

- Supervised features learning
  - Sparse coding (quadratic),
  - KernelBoost,
  - Convolutional Neural Networks (CNN)
  - Convolutional Decision Trees

+ task-specific features
– either restricted class
– or very slow to train

Convolutional Decision Trees

- Learn informative features one by one
  - Tree split is a convolution with some kernel
  - Maximize the relaxed information gain
  - Introduce smoothness regularization
- Combine these features to form a decision tree
  - Grow the tree while the performance increases
  - Adjust the regularization while growing

Relaxed information gain

- Oblique decision trees notation:
  - $x_i \in \mathbb{R}^{w^2+1}, x_{i,1} \equiv 1$ - vectorized image patch
  - β - vectorized convolution kernel
  - Split predicate:

$$\phi(x_i, \beta) = [\beta^T x^i > 0]$$

- Relaxed split predicate:

$$\hat{\phi}_\alpha(x_i, \beta) = \frac{1}{1 + \exp(-\alpha \beta^T x_i)}$$

Relaxed information gain

- The notation of Information Gain is almost the same...

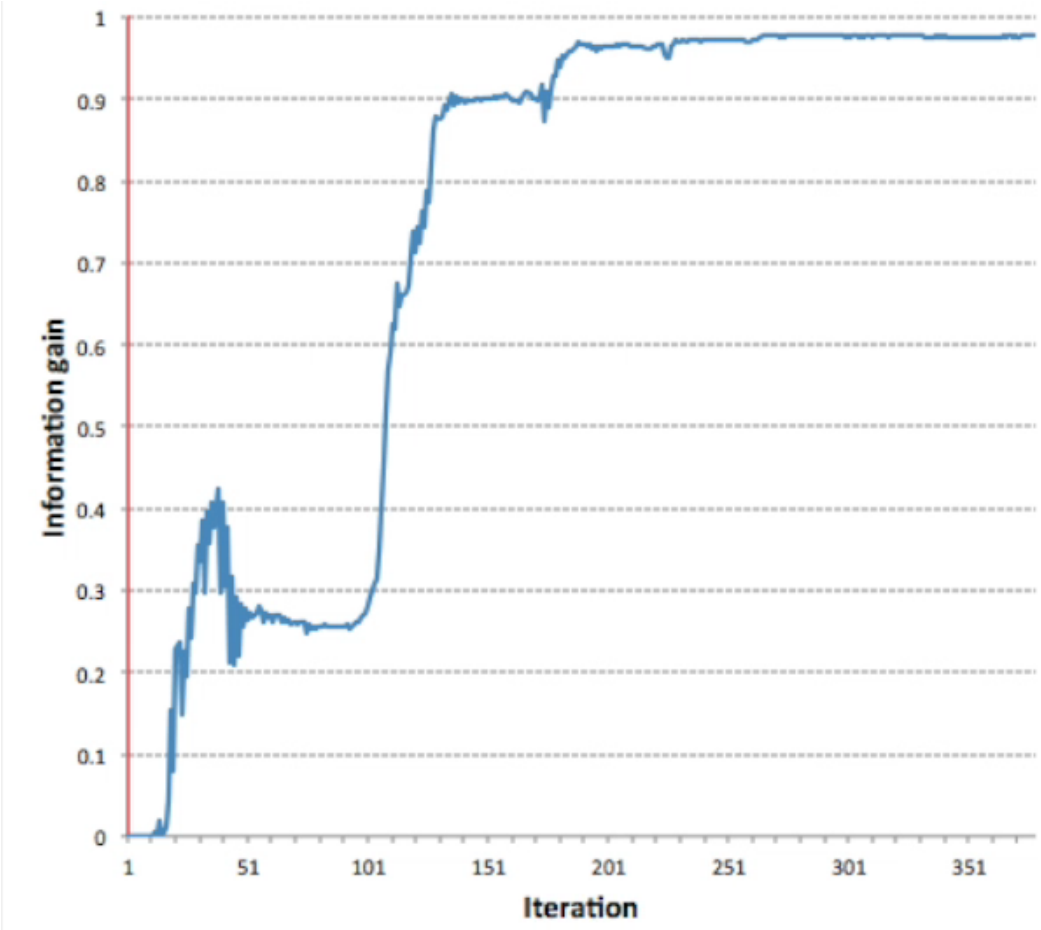
$$h_c = |\{i : y_i = c\}|, \forall c = 1, \ldots, C$$

$$h_c^{\text{left}} = \sum_{i:y_i=c} \hat{\phi}_\alpha(x_i, \beta), \quad h_c^{\text{right}} = h_c - h_c^{\text{left}}$$

$$\hat{\text{IG}}_\alpha = H(h) - \frac{\sum_i \hat{\phi}_\alpha(x_i,\beta)}{N} H(h^{\text{left}}) - (N - \frac{\sum_i \hat{\phi}_\alpha(x_i,\beta)}{N}) H(h^{\text{right}})$$

- ... but now *smooth*!

Relaxed information gain

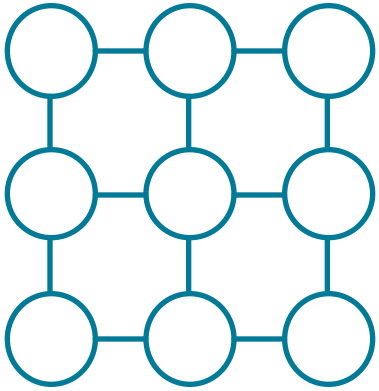


Regularized information gain

- Smoothness term Γ...

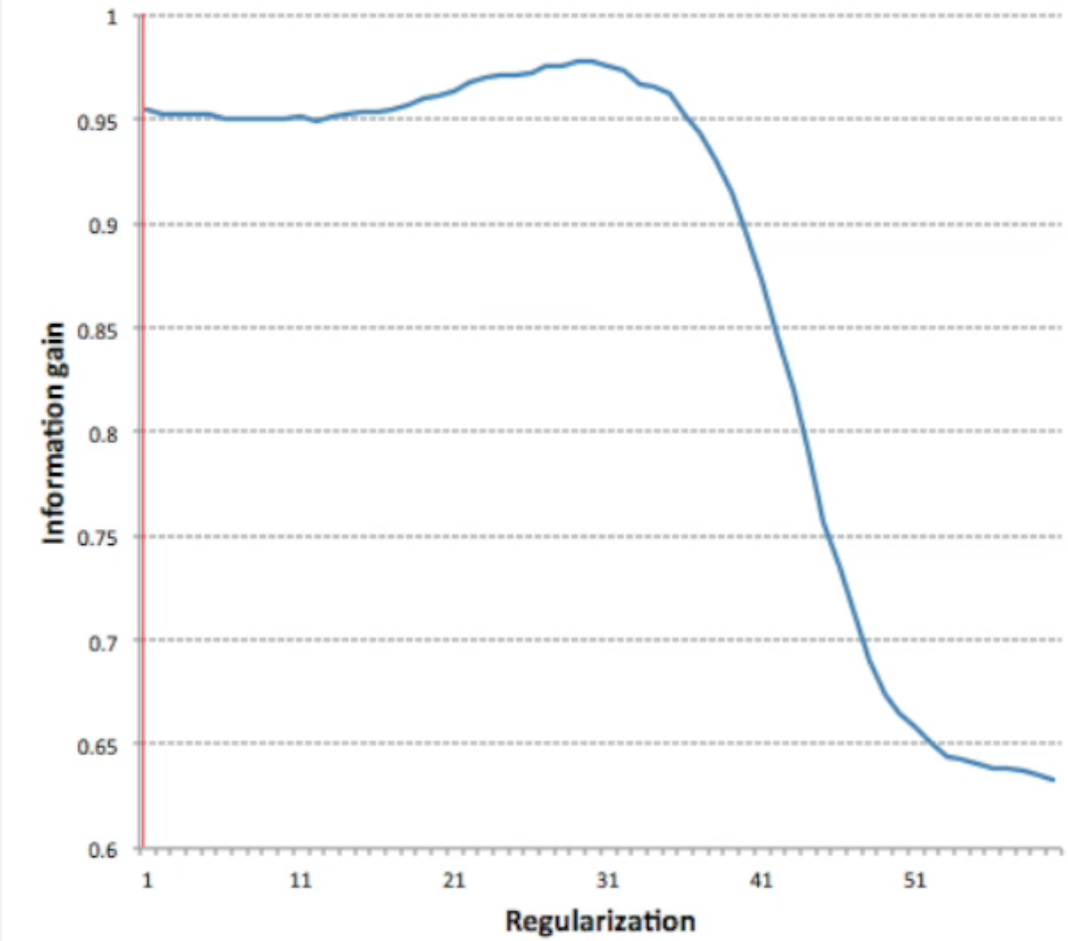
$$L_\alpha(\beta) = \mathrm{I}\hat{\mathrm{G}}_\alpha(\beta) - \lambda||\Gamma\beta||_2^2$$

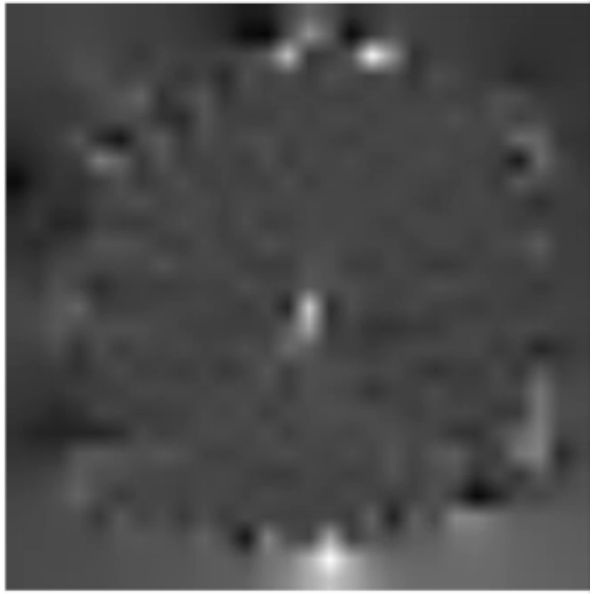
$$\beta_\alpha \in \arg\max_\beta L_\alpha(\beta),$$



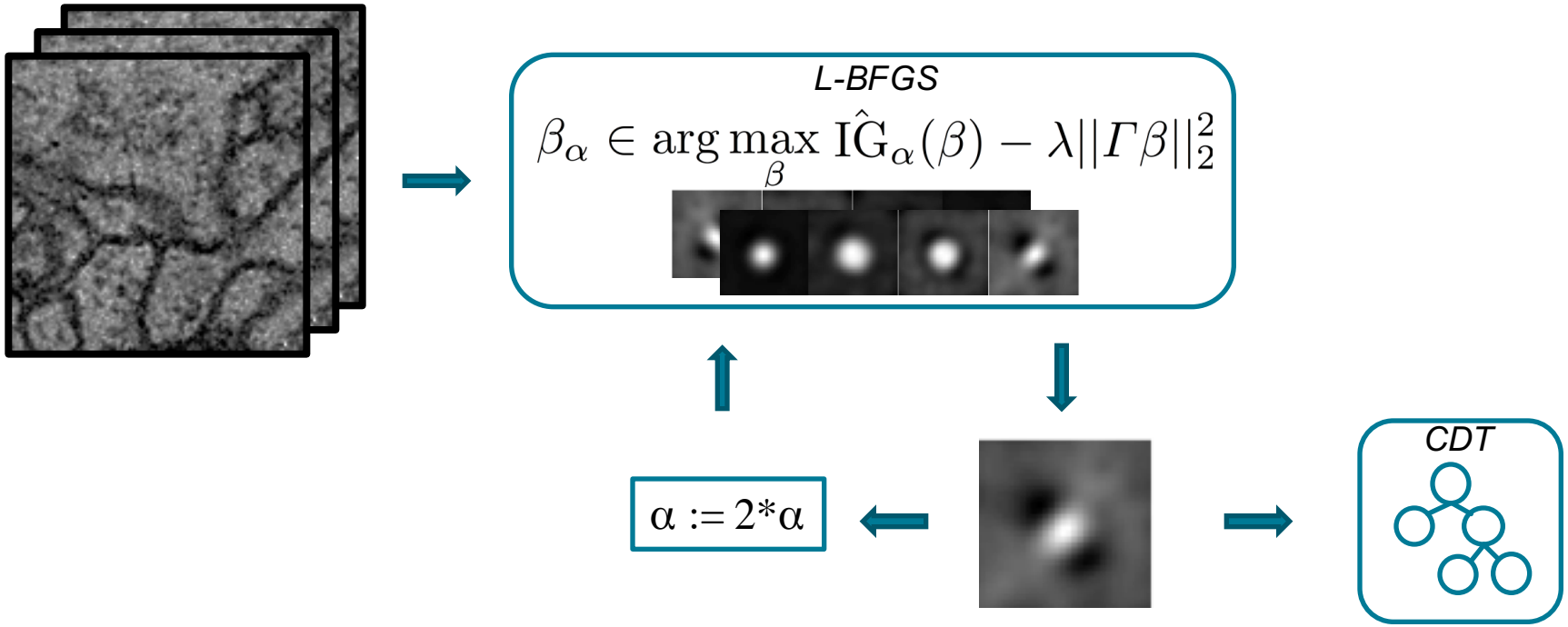
- ... makes the functional „*more convex*“

Regularized information gain





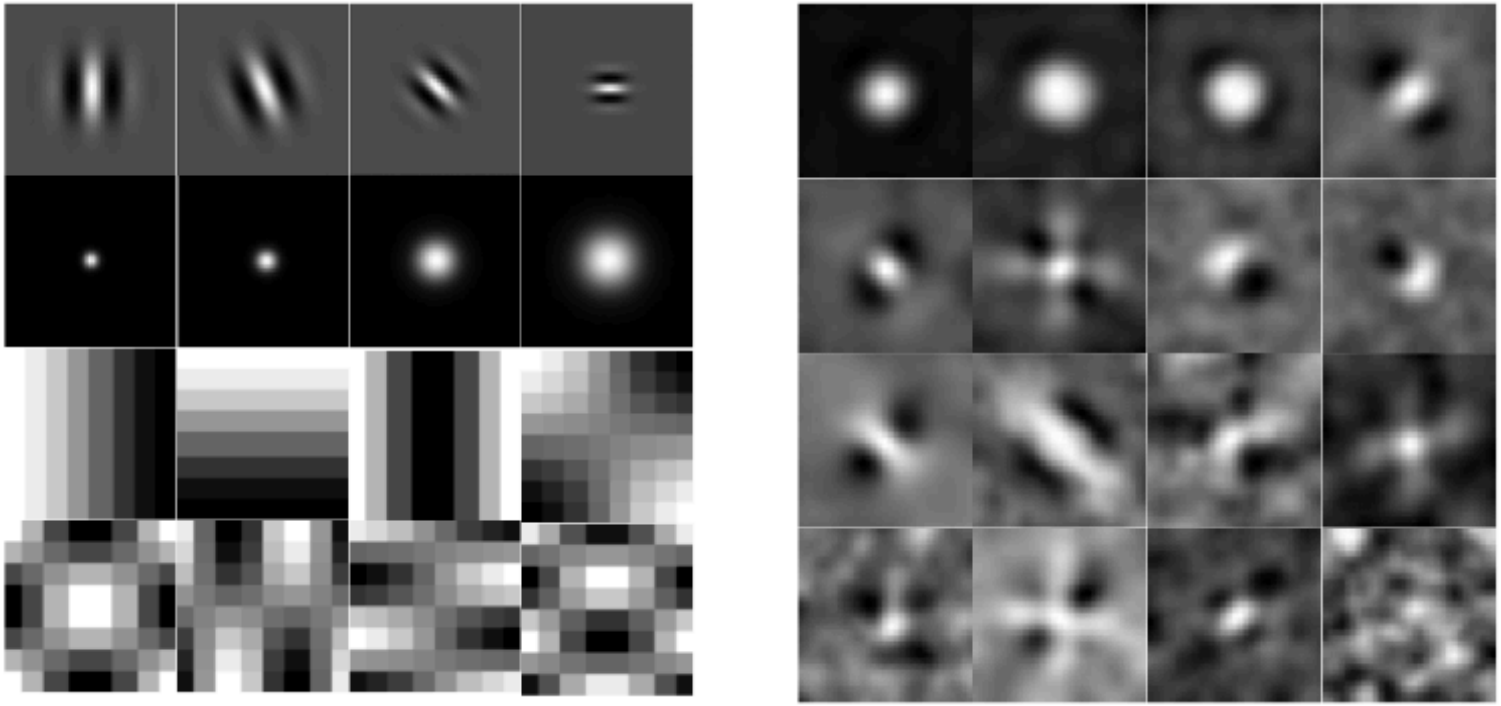
Learning one informative feature



$$\beta \in \arg\max_\beta \mathrm{IG}(\beta) = \lim_{\alpha \to +\infty} \arg\max_\beta \hat{\mathrm{IG}}_\alpha(\beta)$$

* we refer to the paper for all the details

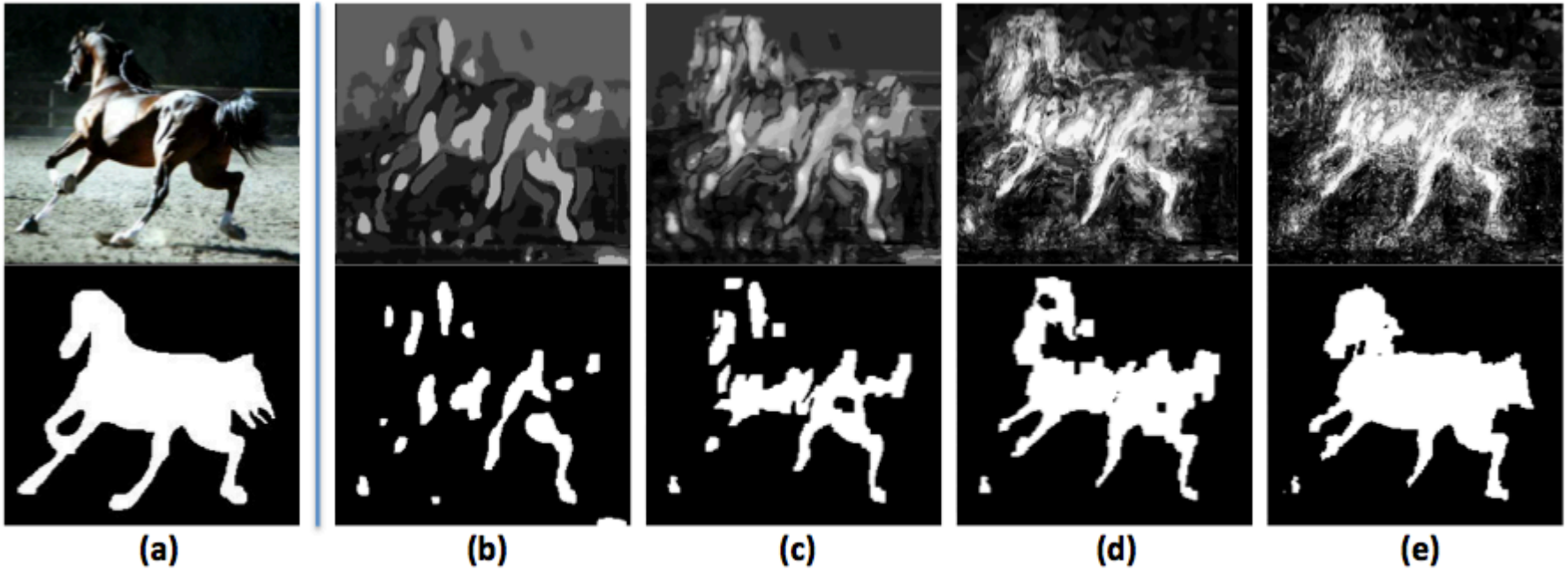
Examples of learned features



Left: static generic features; right: features learned with CDT

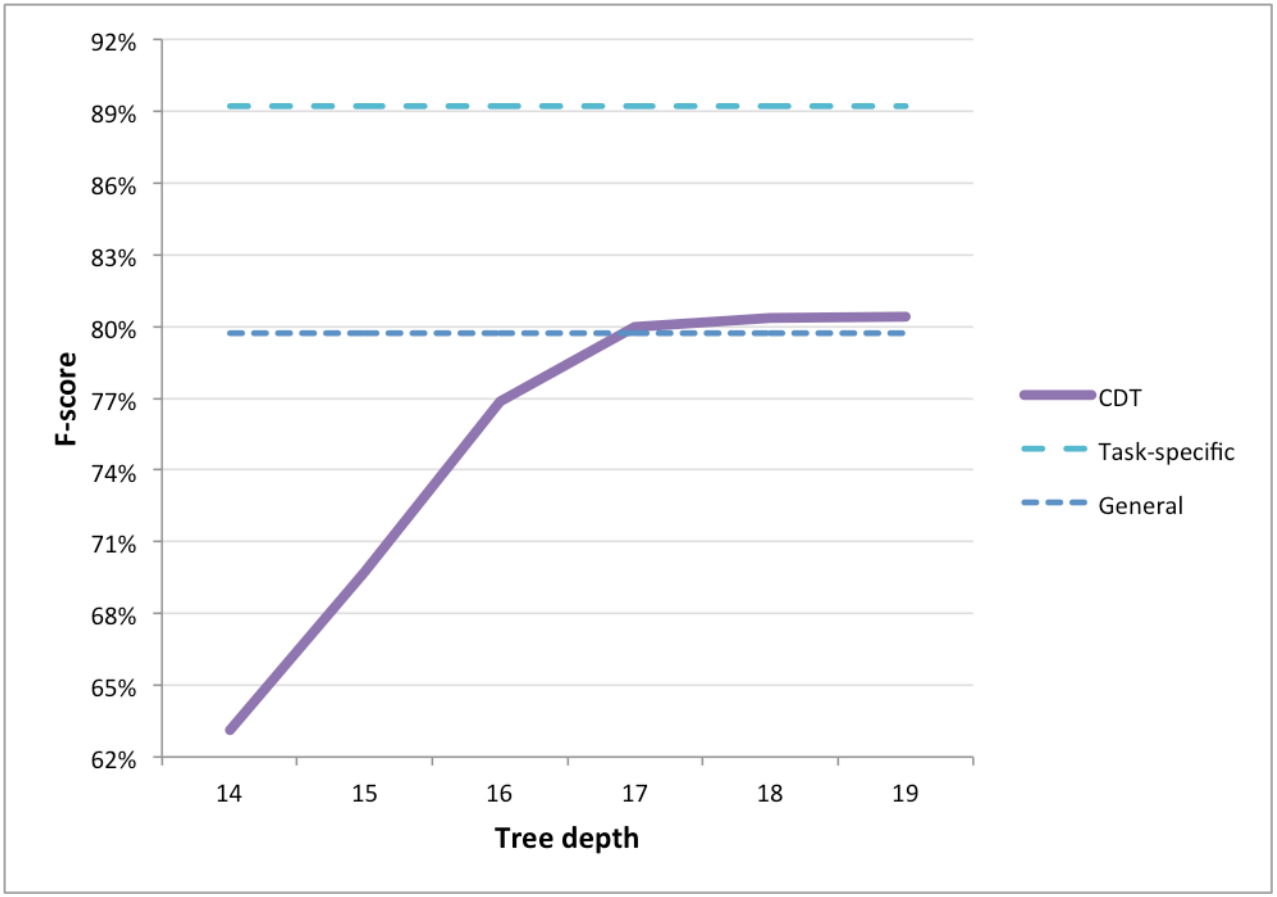
Features are interpretable: there are edge detectors, curvature detectors, etc.

Results: Weizmann Horse dataset



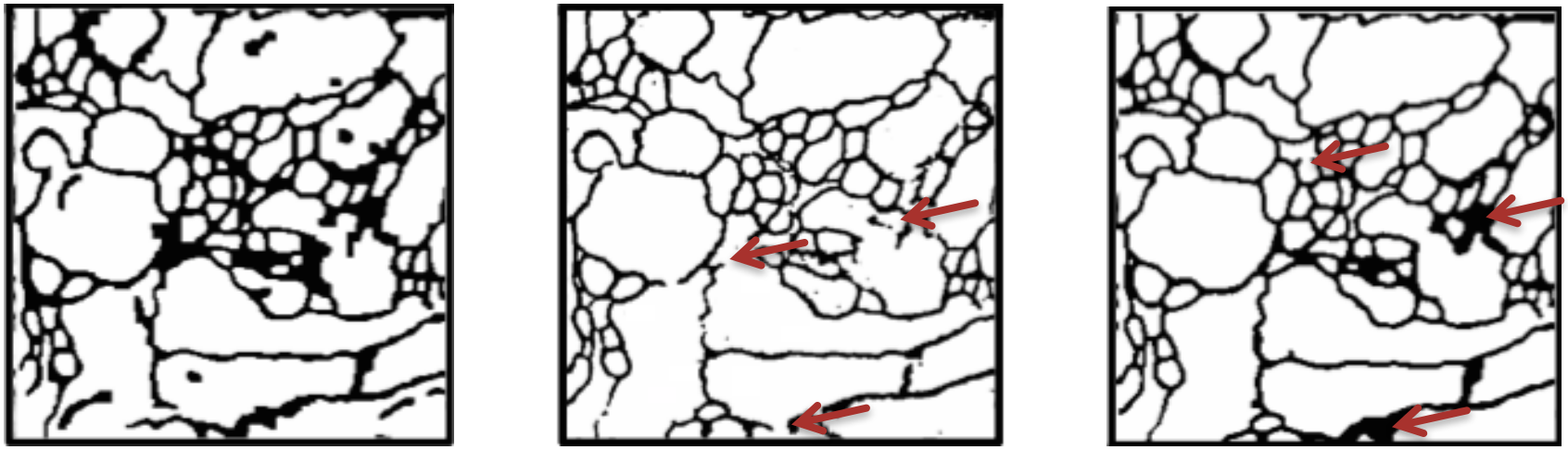
(a): ground truth; (b) – (e): results for different tree depth
(top): per-pixel probabilities; (bottom): graph-cut segmentation

Results: Weizmann Horse dataset



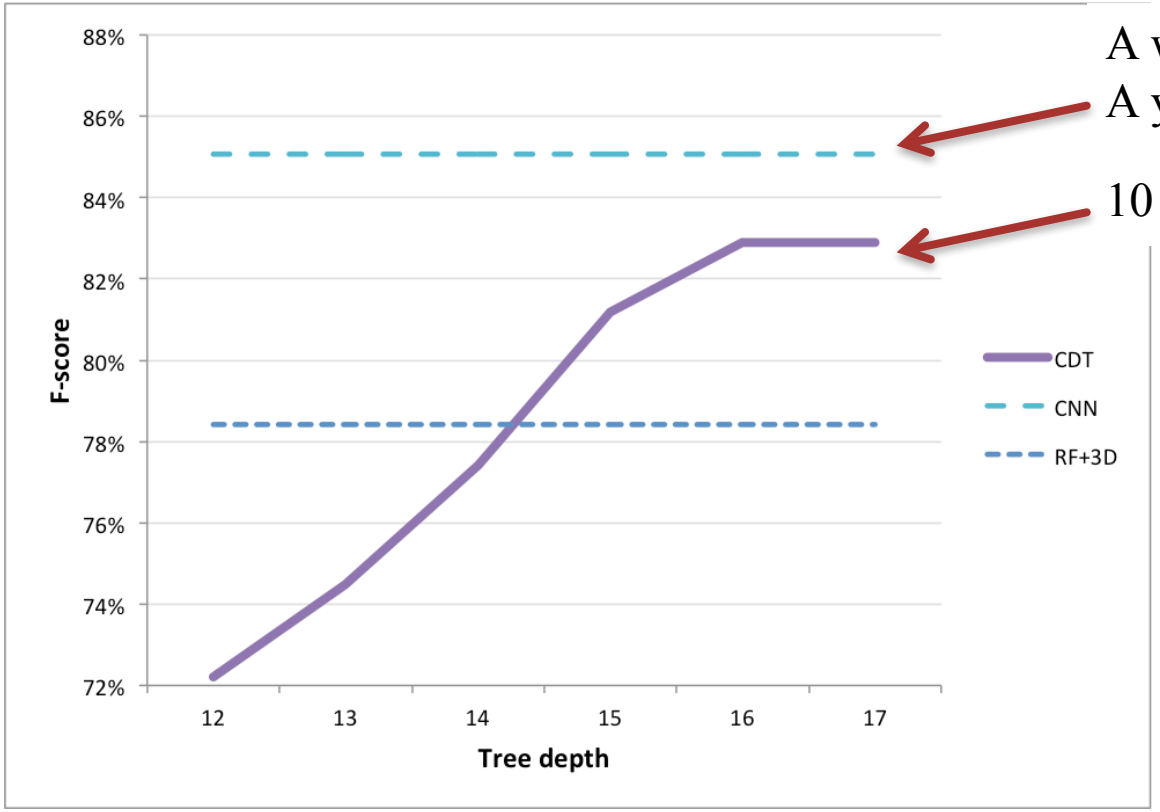
CDT is better than any general local technique, but worse than task-specific

Results: Drosophila VNC segmentation



From left to right: results of anisotropic RF, CDT and CNN

Results: Drosophila VNC segmentation



CDT is 4.5% better than the second-best technique and only 2.2% worse than CNN
CDT requires 10 hours training in 1 CPU, while CNN requires a week in GPU-cluster

Summary

- Convolutional Decision Trees:
  - Much faster (over-night experiment)
  - Require no special hardware
  - Trade-off between the speed and the quality

- Consist of three main components:
  - Relaxed information gain
  - Strictly convex regularization term
  - Iterative optimization algorithm

Appendix D
Transformation-invariant convolutional jungles

Feature formulation

Transformation-invariant feature parameterized by θ:

$$f_\theta(x) = \max_{\phi \in \Phi} \theta^T \phi(x)$$

Vectorized image (patch)

Transformations considered

Convolution kernel (parameters)

Lemma 1. The feature of the image $\mathbf{X}$ defined above is transformation-invariant if the set Φ of all possible transformations forms a group[1], i.e. satisfies the axioms of closure, associativity, invertibility and identity.

Feature Learning: looking for the best parameters θ.

Kernel gradient regularization

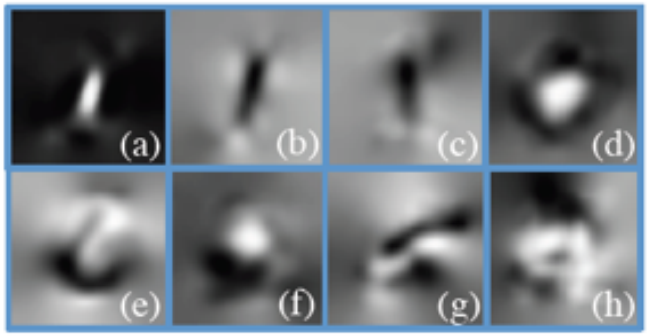
$$\theta = \arg\min_\theta E(\theta) = \arg\min_\theta \quad \lambda ||\Gamma\theta||_2^2 + \sum_{i:\ y_i=c_1 \text{ or } y_i=c_2} (f_\theta(X_i) + [y_i = c_1] - [y_i = c_2])^2$$

[•] equals 1 if • is true and 0 otherwise

Two classes to split

The problem is convex, but not continuously differentiable.

Feature examples



Examples of learned features for membrane segmentation. Features (a)–(c) detect direct membranes, (d) denotes the contrast of the center pixels comparing to the surroundings, (e) and (f) detect corners and curvatures (non-straight membranes), (g) and (h) – high-frequency features.

Transformation-invariant convolutional jungles

Oblique decision trees:
- use multivariate, not univariate splits.

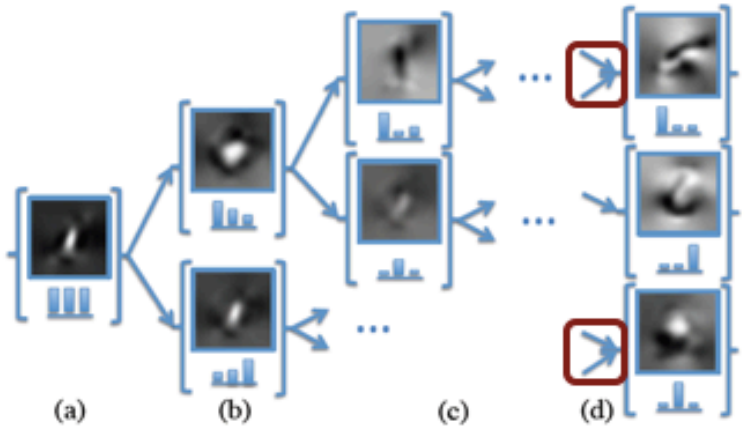
Decision jungles:
- merge nodes when max width is achieved.

---

1. Given a dataset and a pair of classes, we can find feature parameters θ;
2. θ defines a predicate that splits the data:

$$l_1 = \{i : f_\theta(X_i) > 0\} \quad l_2 = \{i : f_\theta(X_i) \leq 0\}$$

3. Selecting new pairs of classes, we can recursively build a tree from l_1 and l_2
4. When maximum width M is achieved, merge similar subsets of the dataset.



(a) shows the root node (the whole dataset is an input). Feature parameters θ are learned and the dataset is split in two subsets (input for two other nodes) (b). The algorithm proceeds until the maximum width M is achieved (c). Then some of the data subsets close in distribution can be joined together (d).

TICJ experiments

1. Neuronal segmentation.

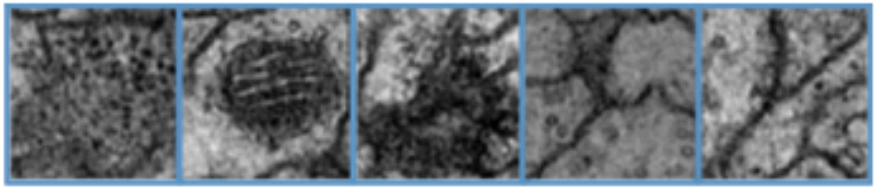
Data:
- ISBI 2012 challenge.

Two classes:
- membrane vs. neuron.

Images:
- 60 images 512x512,
- 31x31 patches.

Invariances:
- rotations (24 angles).

TICJ results:
- same F-score as for CNN,
- 100 times faster than CNN.



Example patches from the dataset: sometimes very blurred and unclear.

Method	1–F-score	Time
RF+3d	7.9	unknown
CDT	6.8	8h (CPU)
CNN	**6.0**	7d (GPU)
TICJ (ours)	**6.0**	**4h (CPU)**

Best teams results: Random Forest with hundreds of features, Convolutional Decision Trees, Deep Neural Networks.

TICJ experiments

2. Face recognition

Data:
- Yale face database,
- very small (165 images).

15 classes for:
- 15 individuals,
- 11 images per person.

Images:
- cropped version 32x32.

Invariances:
- small shifts, rotations,
- illumination-invariance.

TICJ results:
- best with no external data.

Example faces of one person from the dataset.

Method	Error, %
Cai et al.	18.3
Cai et al. (u)	14.7
Hua et al.	13.2
Shan et al.	**8.2**
TICJ	**12.9**

TICJ outperforms every team except one. Shan et al. shows significantly better results but uses additional training data.

TICJ summary

Transformation-Invariant Convolutional Jungles:

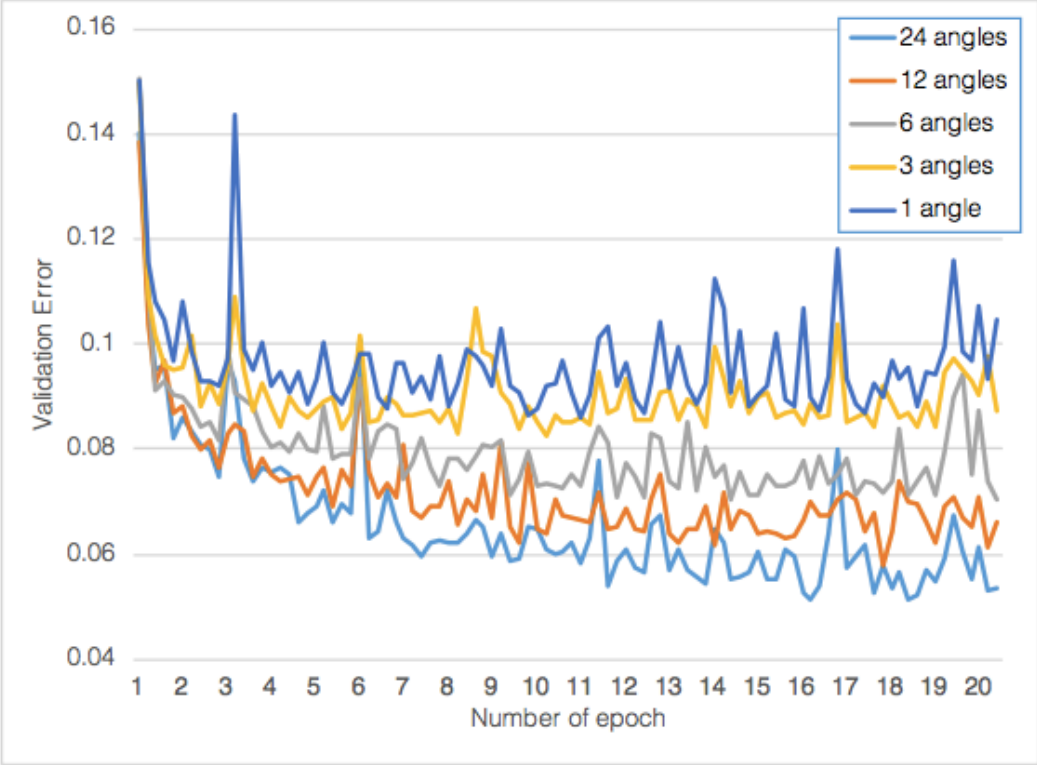
- much faster than deep neural networks (4 hours vs. 7 days),
- in some tasks can match their performance,
- require no special hardware (but can also benefit from GPU),
- can work with small datasets (165 images in Yale dataset).

Consist of three main components:

- max-pooling over transformations (ensures invariance),
- strictly convex regularization term (produces smooth kernels),
- decision jungles algorithm (prevents overfitting and speeds up).

Appendix E
TI-pooling

TI-pooling convergence



- The more angles we sample for a set Φ – the better results are achieved (time vs. accuracy trade-off).
- Fewer canonical positions needs to be handled by the learning algorithm, unlike augmentation.