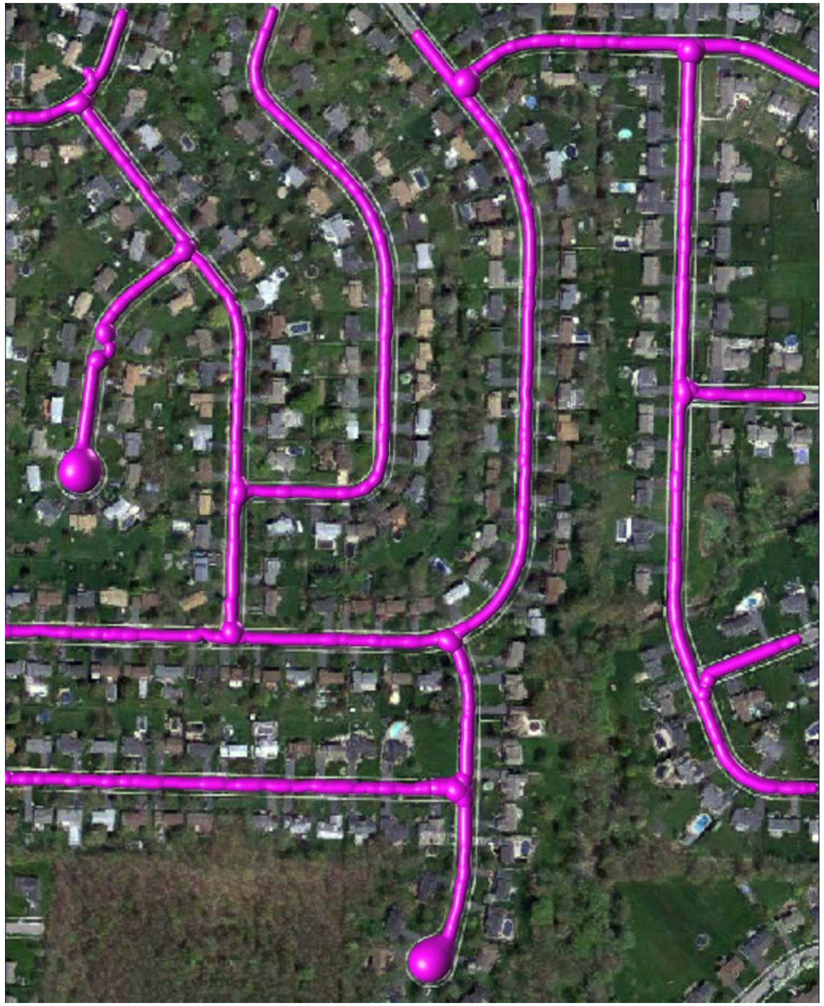
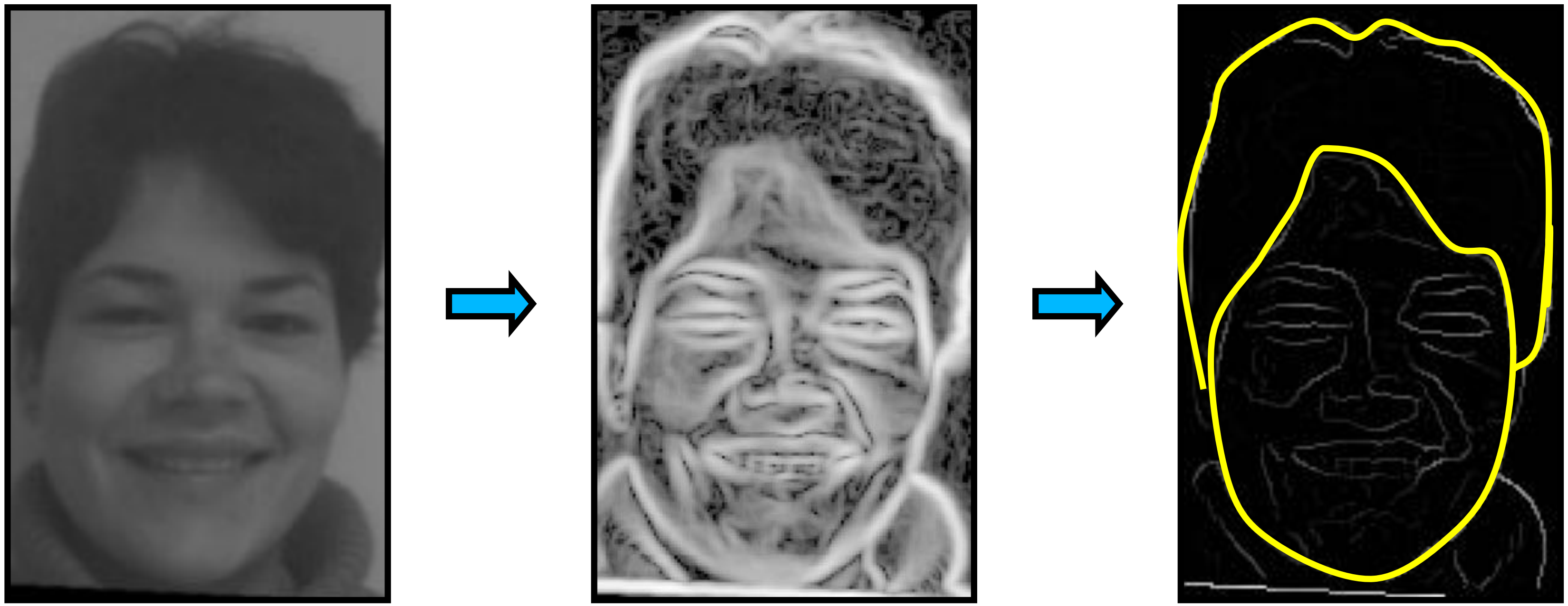
Delineation

- Dynamic Programming
- Deformable Models
- Hough Transform
- Graph Based Approaches

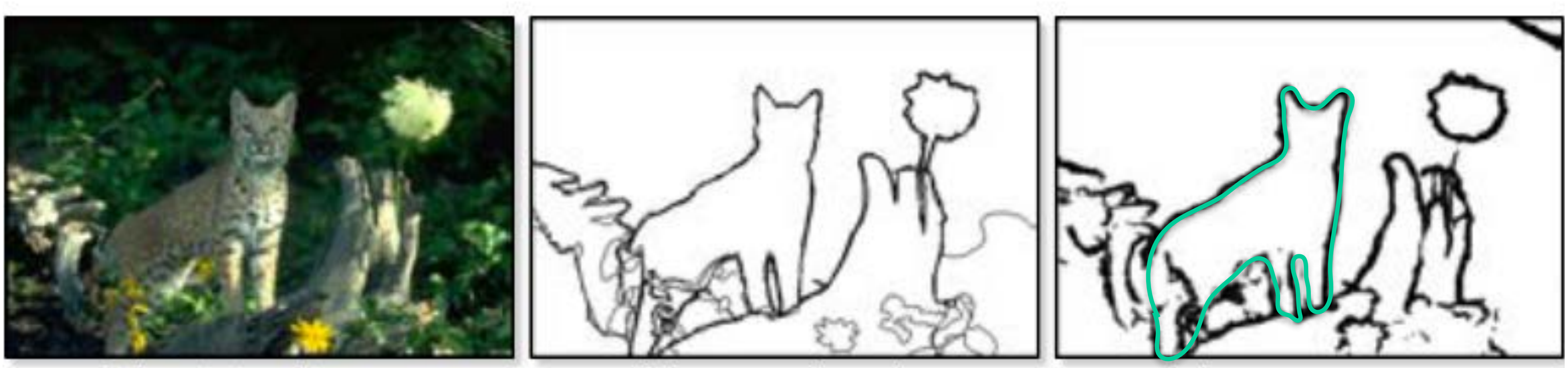


EPFL

From Gradients to Outlines



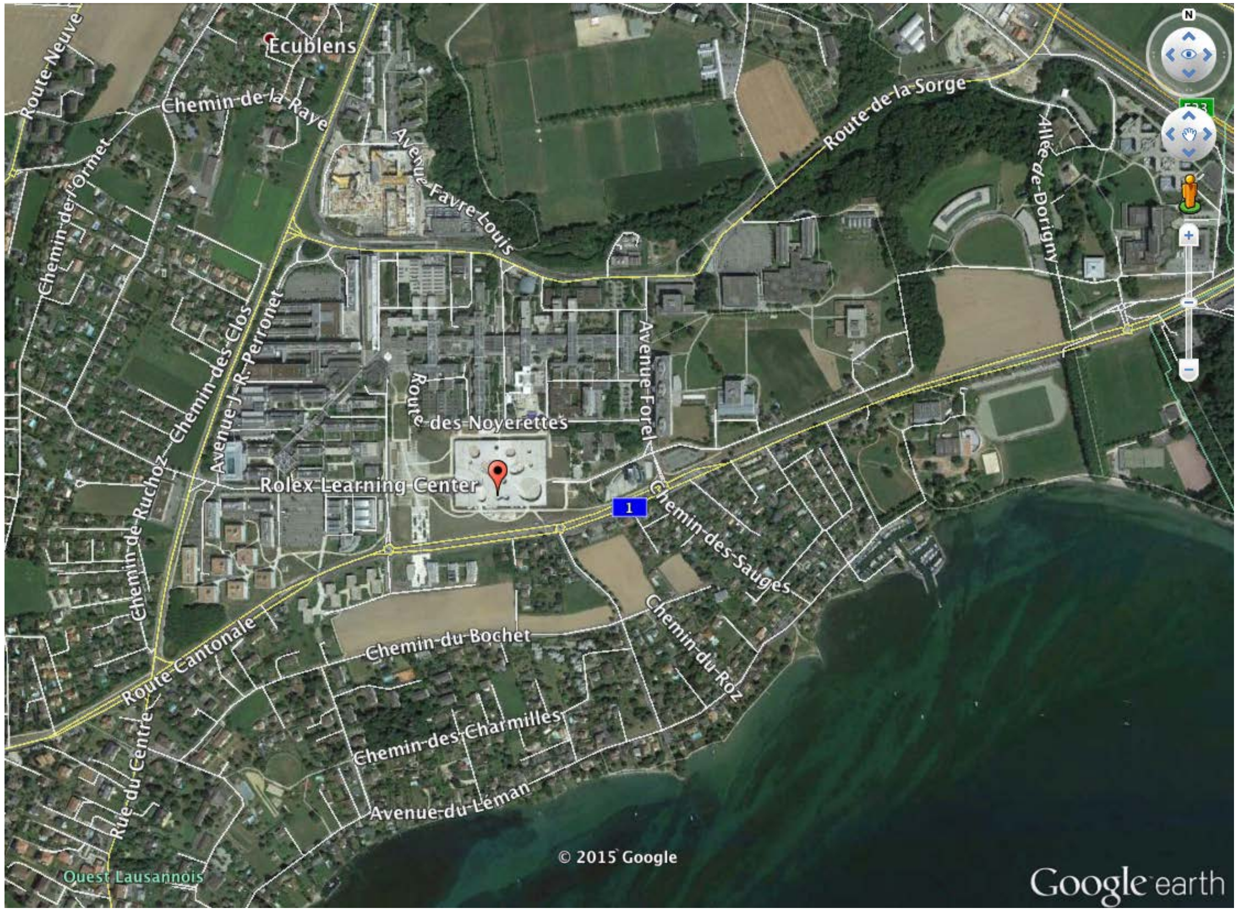
From Deep-Nets to Outlines



(a) original image (b) ground truth (c) HED: output

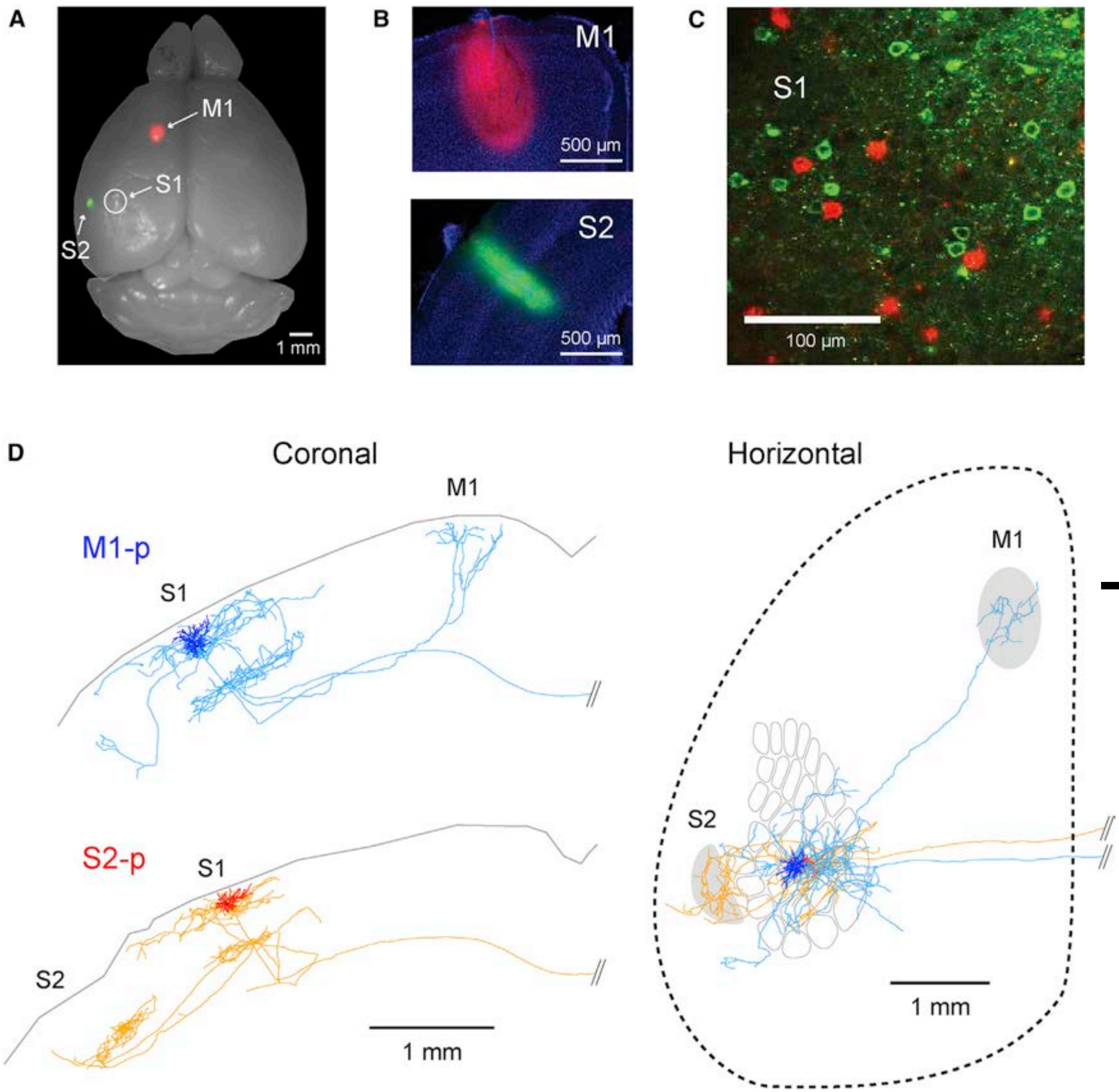
—> Still work to do!

Mapping and Overlays



Connectivity matters!

Connectomics



—> Topology needed

Techniques

Semi-Automated Techniques:

- Dynamic Programming
- Deformable Models

Fully Automated Techniques:

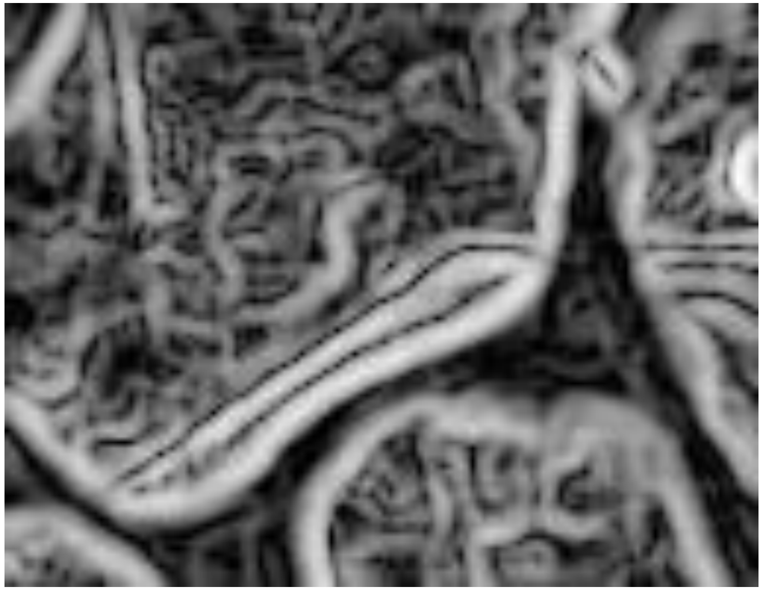
- Hough Transform
- Graph Based Approaches

Reminder: Canny Limitations

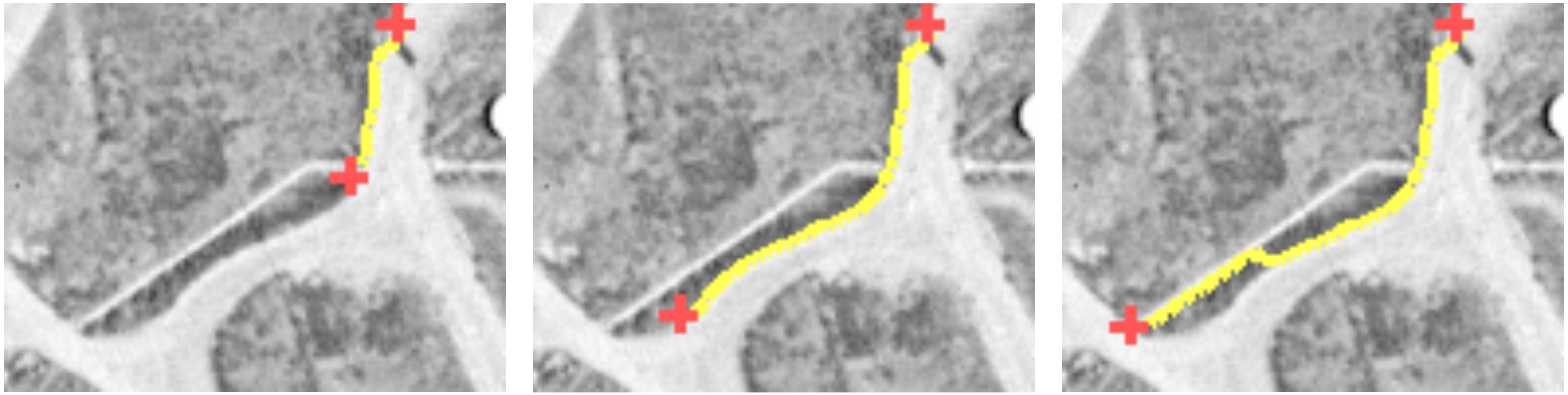
- There is no ideal value of σ!
- Deep nets can help but do not solve the problem.

Interactive Delineation

Image



Gradient

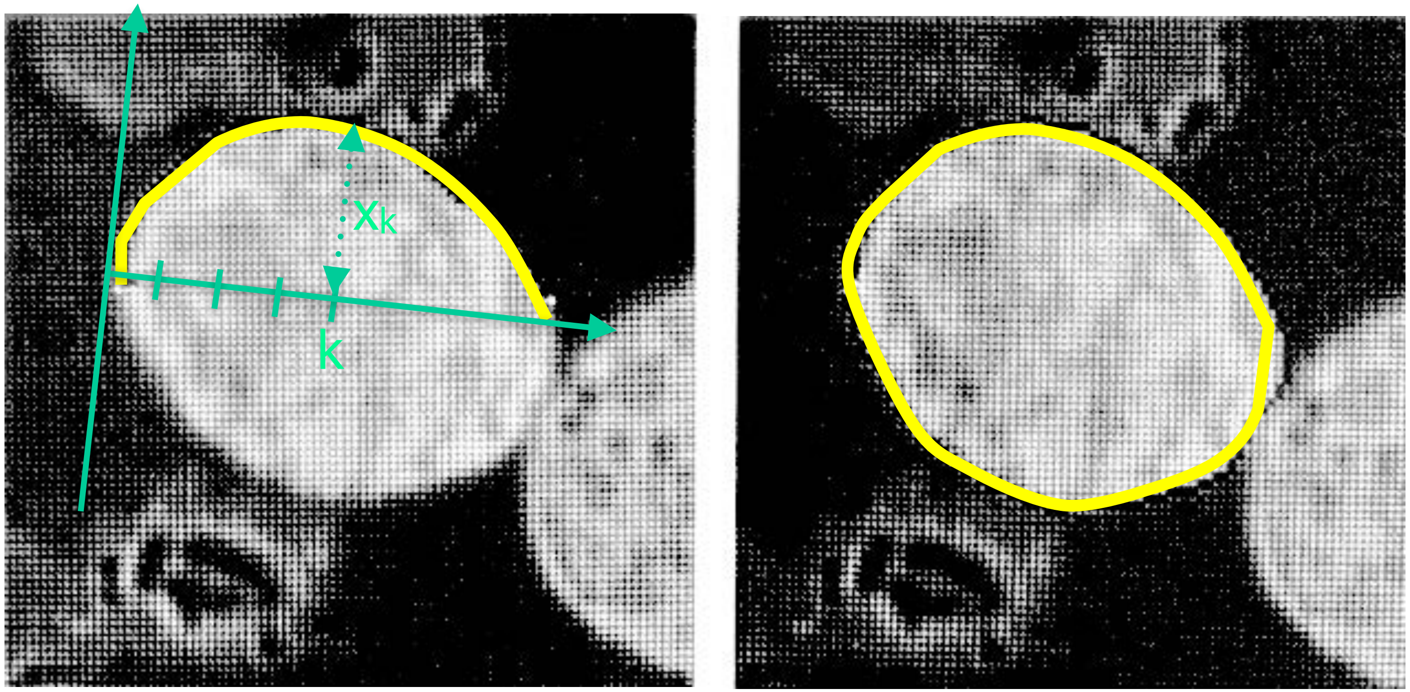


- The user provides the start and end points (red x).
- The algorithm does the rest (yellow line).

Live Wire in Action

Mortensen and Barrett, SIGGRAPH'95

1D Dynamic Programming



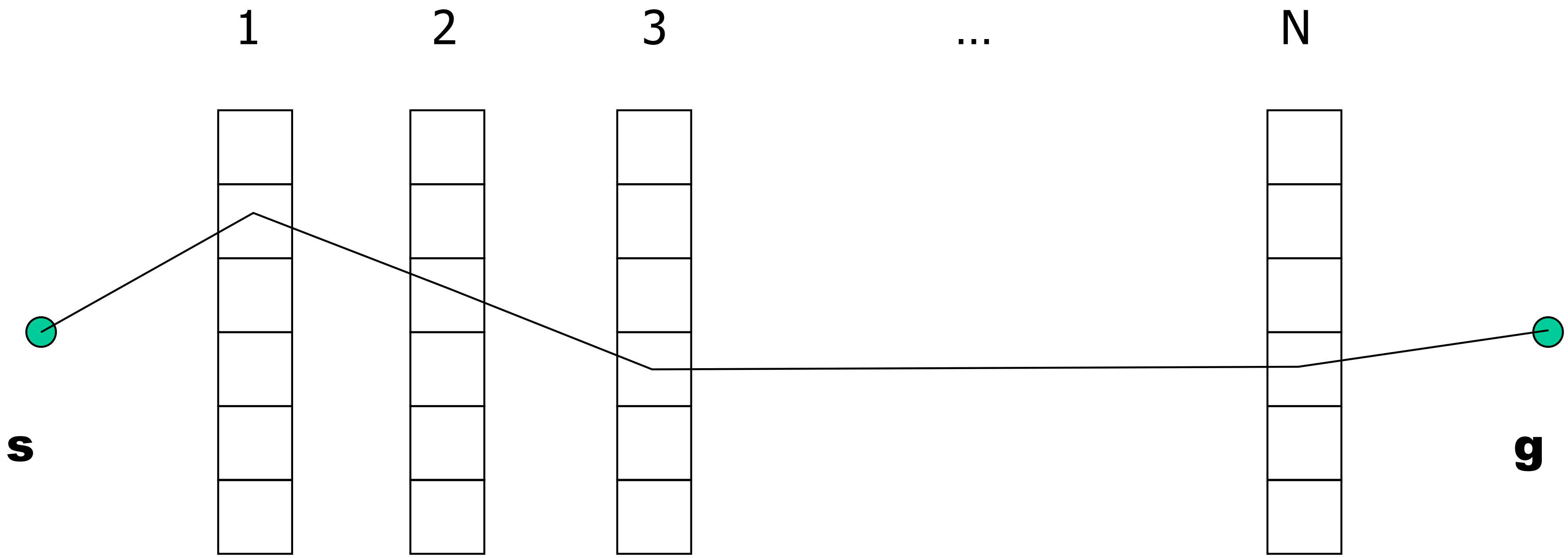
$$h(x_1, x_2, \ldots, x_n) = -\sum_{k=1}^{n} g(x_k) + \sum_{k=1}^{n-1} r(x_k, x_{k+1})$$

$$r(x_k, x_{k+1}) = diff(\phi(x_k), \phi(x_{k+1}))$$

where ϕ denotes the gradient orientation.

EPFL

1D Dynamic Programming



- N Locations
- Q Quantized values

→ Global optimum $O(NQ^2)$

1D Dynamic Programming

To find

$$\min_{x_i} h(x_1, x_2, \ldots, x_n)$$

where

$$h(x_1, x_2, \ldots, x_n) = \mathrm{r}(s, x_1) + \sum_{i=1}^{n-1} \mathrm{r}(x_i, x_{i+1}) + \mathrm{r}(x_n, g)$$

define

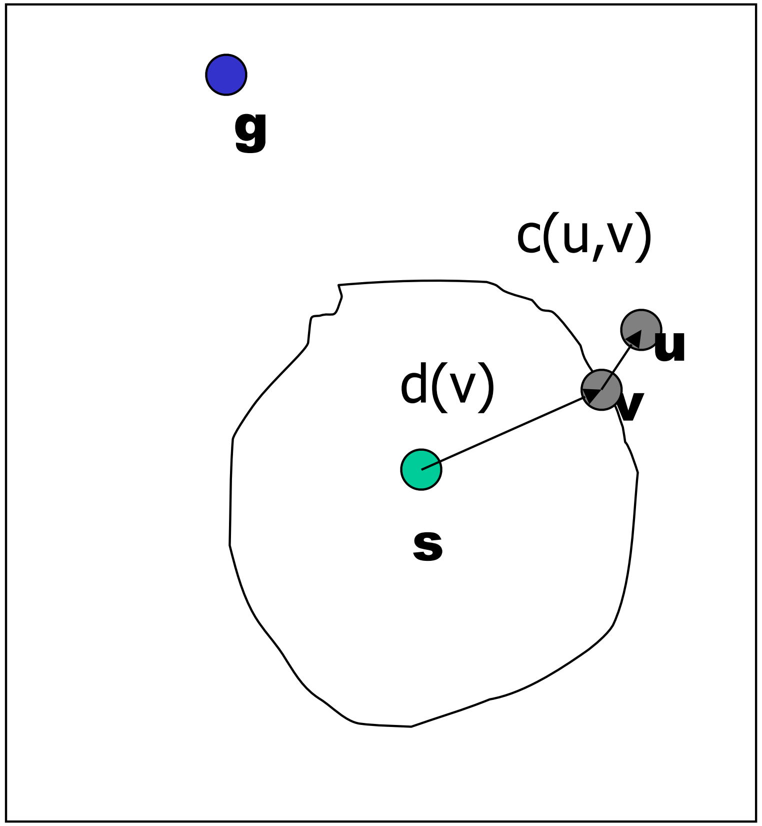
$$\begin{array}{lcl}
f_1(x_2) & = & \min_{x_1} (\mathrm{r}(s, x_1) + \mathrm{r}(x_1, x_2)) \\
f_2(x_3) & = & \min_{x_2} (\mathrm{r}(x_2, x_3) + f_1(x_2)) \\
\vdots & \vdots & \vdots \\
f_{n-1}(x_n) & = & \min_{x_{n-1}} (\mathrm{r}(x_{n-1}, x_n) + f_{n-2}(x_{n-1}))
\end{array}$$

$$\Rightarrow \min h(x_1, x_2, \ldots, x_n) = \min_{x_n} (\mathrm{r}(x_n, g) + f_{n-1}(x_n))$$

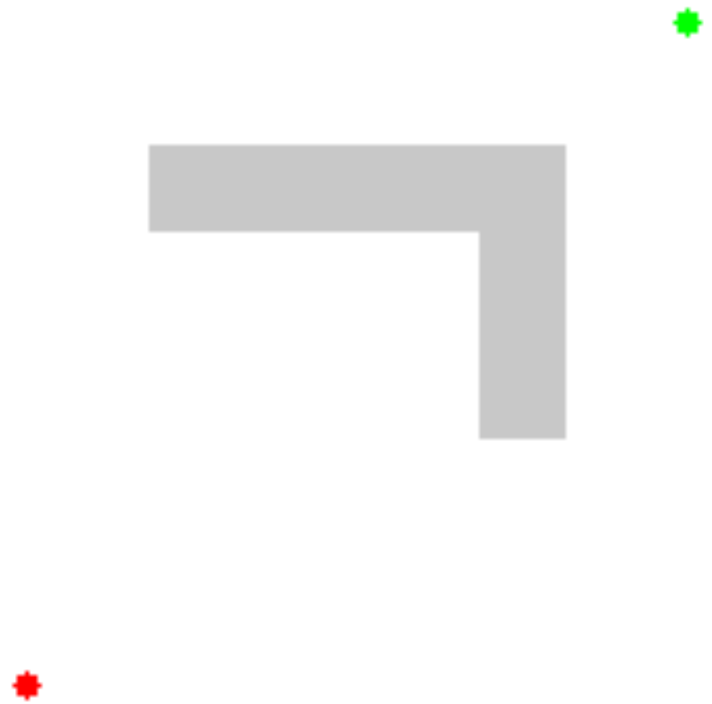
2D Dynamic Programming

Notations:

s	Start point
L	List of active nodes
c(u,v)	Local costs for link u -> v
d(v)	Total cost from s to v



Dijkstra Path Expansion



Open nodes represent "unvisited" nodes. Filled nodes are visited ones, with color representing the distance: The greener, the shorter the path. Nodes in all the different directions are explored uniformly, appearing more-or-less as a circular wavefront.

https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm

Dijkstra's Algorithm

Initialization :

$d(s) \leftarrow 0$ and $d(u) \leftarrow \infty$ for $u \neq s$

$T = \varnothing$

$v = s$

Loop until goal is reached :

$\mathrm{T} \leftarrow \mathrm{T} \cup \{\mathrm{v}\}$

for all $v \rightarrow u$ edges such that $u \notin T$

if $d(v) + c(v,u) < d(u)$

$d(u) \leftarrow d(v) + c(v,u)$

end

end

$\mathrm{v} = \mathrm{argmin}_{w \notin T} d(w)$

Maintain a sorted list of paths

EPFL

Live Wire

- Sorting is the expensive operation. Normally nlog(n), but can be reduced to log(n) if all costs are integer costs

- Local costs computed using gradient:

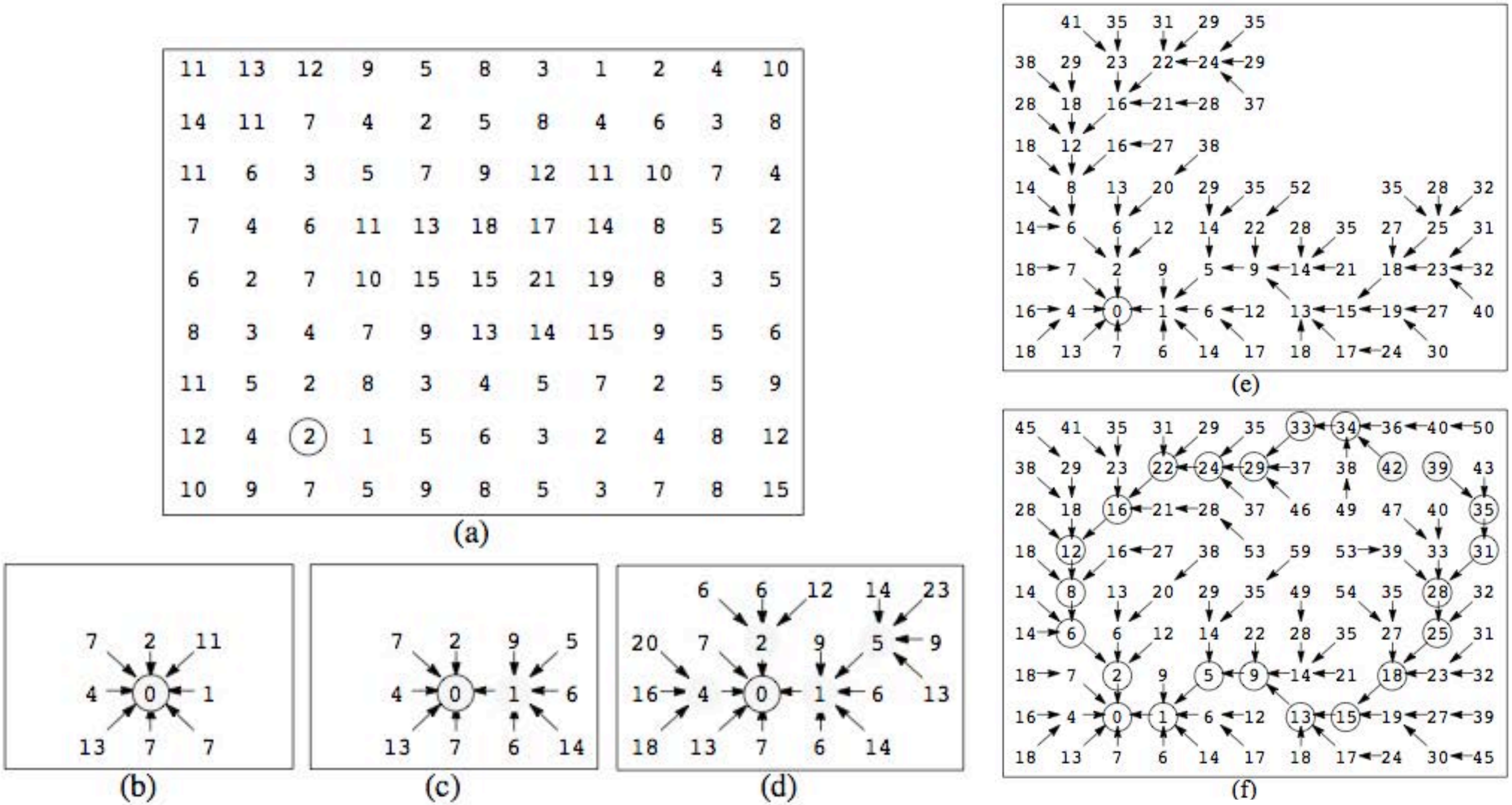
$$c(u,v) = 255 - \frac{1}{2}\,(g(u) + g(v))$$

- Diagonal penalized by multiplying cost of non diagonal edges by:

$$\frac{1}{\sqrt{2}} = \frac{5}{7}$$

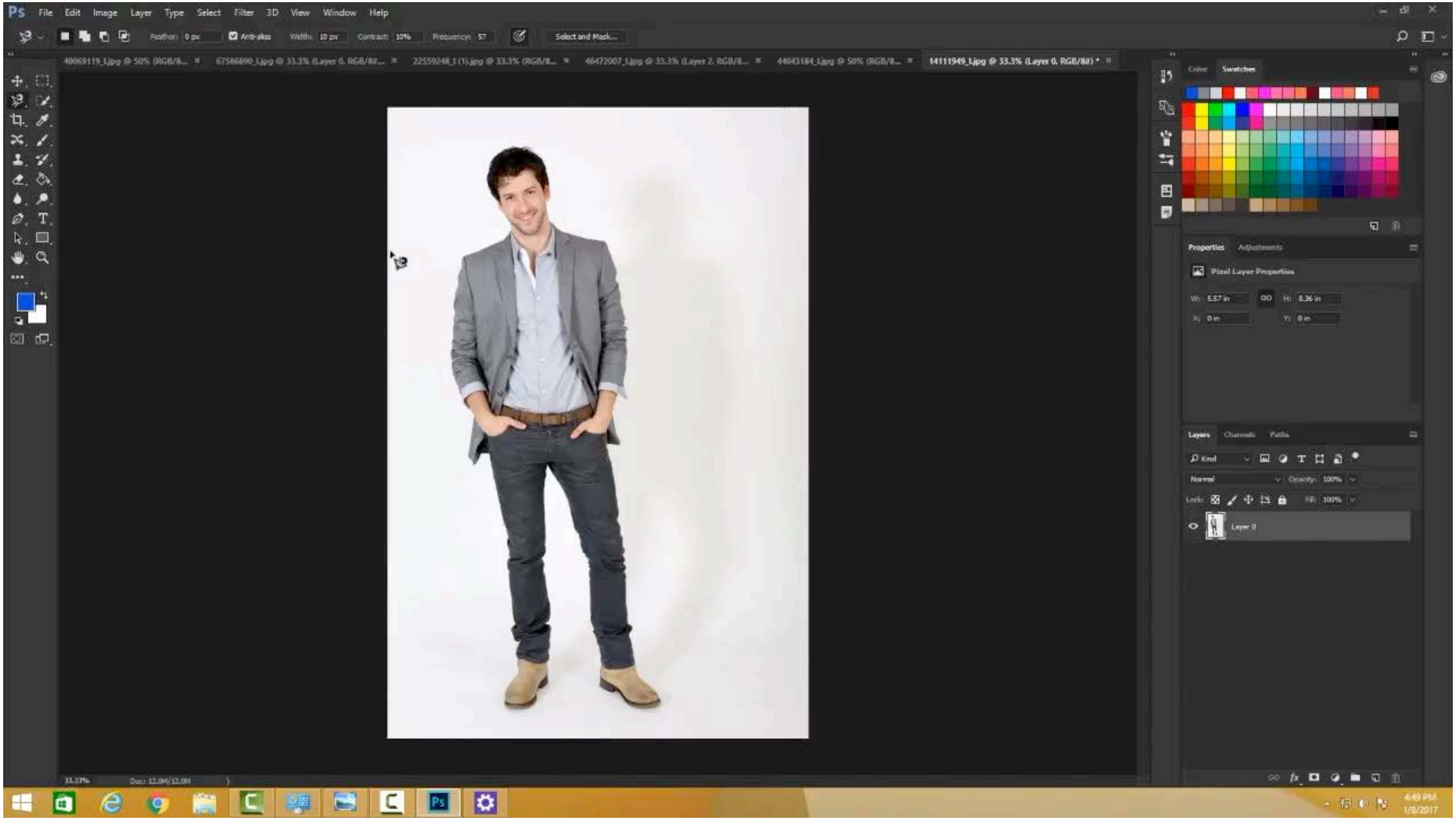
- Add a constant cost for each edge.

Cost Expansion



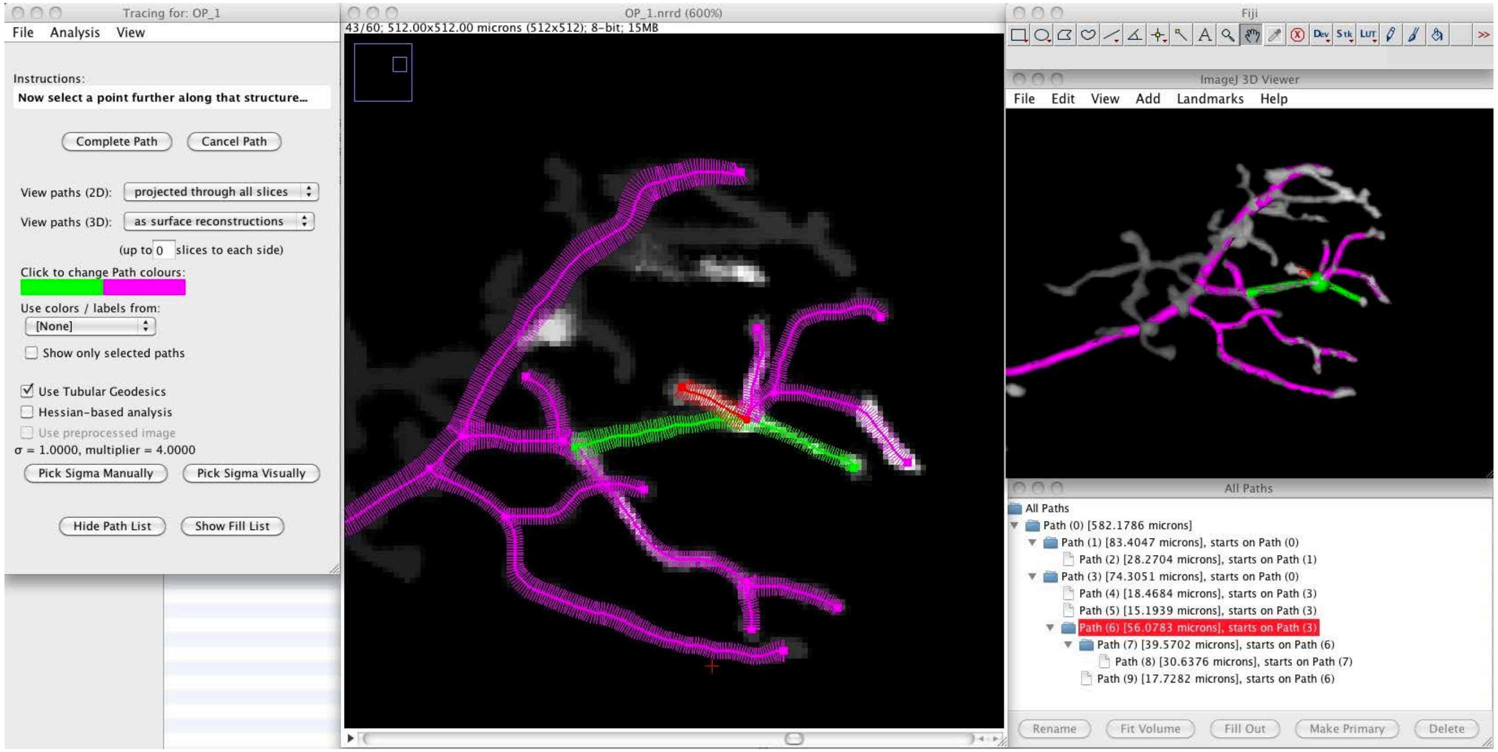
(a) Local cost map. (b) Seed point expanded. (c) 2 points expanded. (d) 5 points expanded. (e) 47 points expanded. (f) Completed cost path-pointer map with optimal paths shown from nodes with total costs 42 and 39.

Magnetic Lasso in Photoshop



Integrating the LiveWire into a powerful interface that allows a user to correct mistakes yields a useful too.

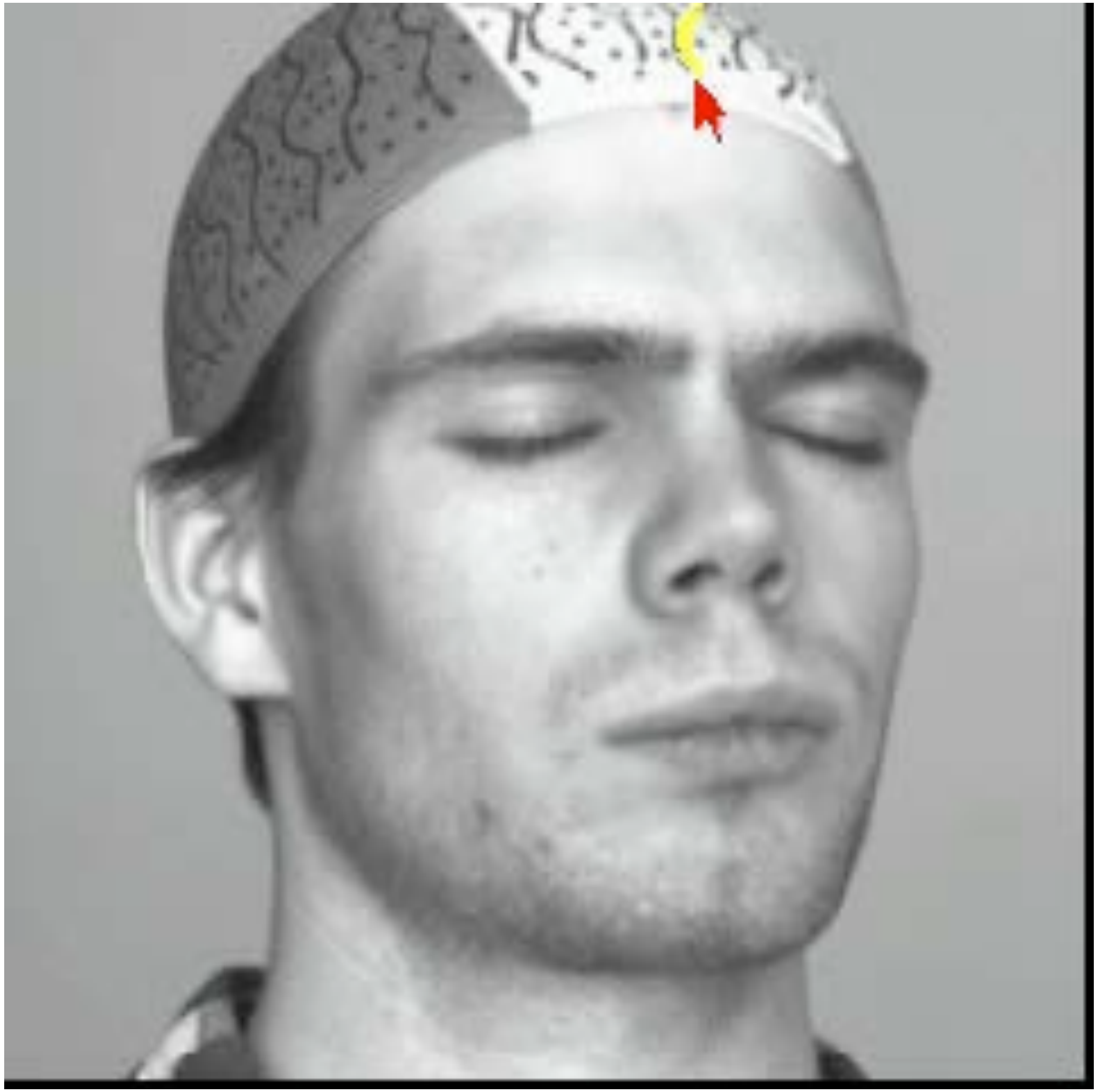
Tracing Neurons



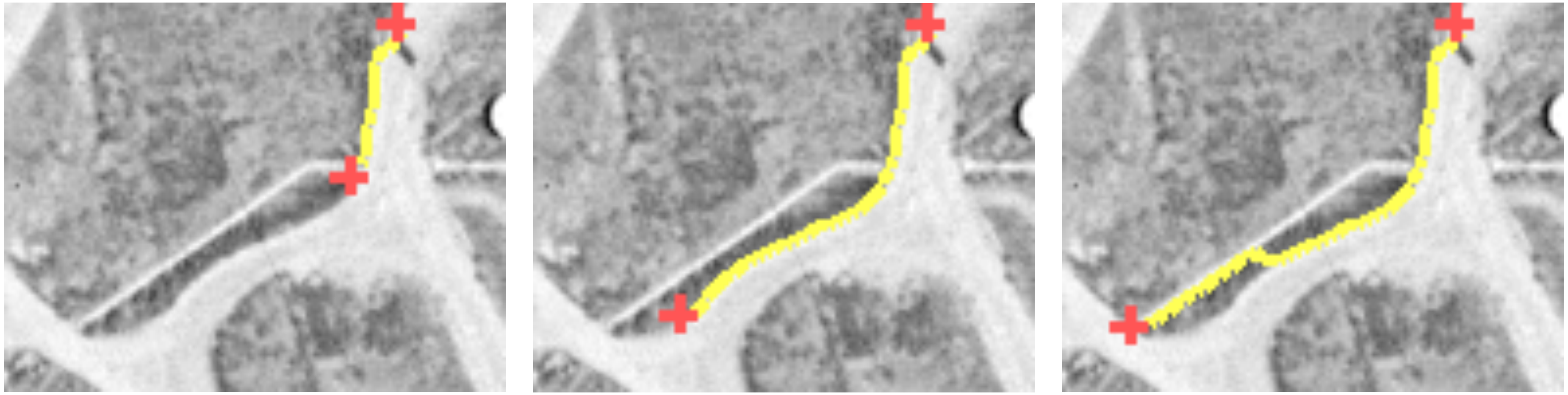
- In the biomedical world, images are 3D cubes of data.
- The approach extends naturally to tracking of 3D structures such as dendritic trees in the brain, blood vessels, etc ...

Face Image

Live Wire



Limitations



- The "optimal" path is not always the "best" one.
- Difficult to impose global constraints.
- The cost grows exponentially with the dimension of the space in which we work.

--> Must often look for local, as opposed to global, optimum using gradient descent techniques.

Techniques

Semi-Automated Techniques:

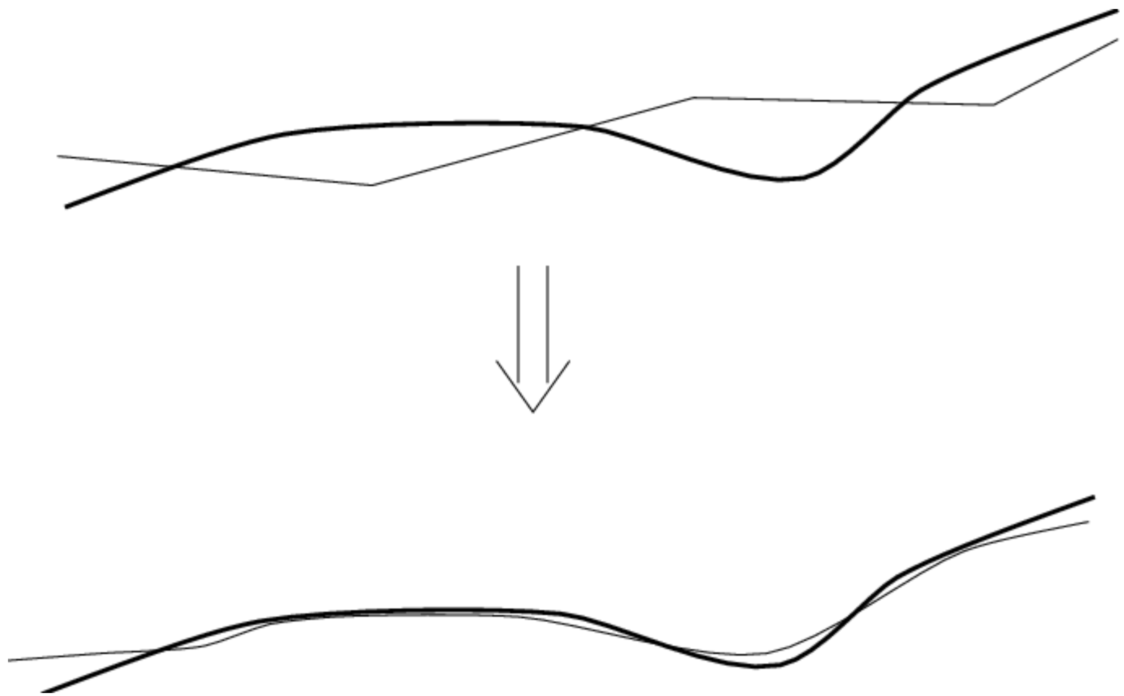
- Dynamic programming
- Deformable Models

Fully Automated Techniques:

- Hough transform
- Graph Based Approaches

Snakes

2—D Snake

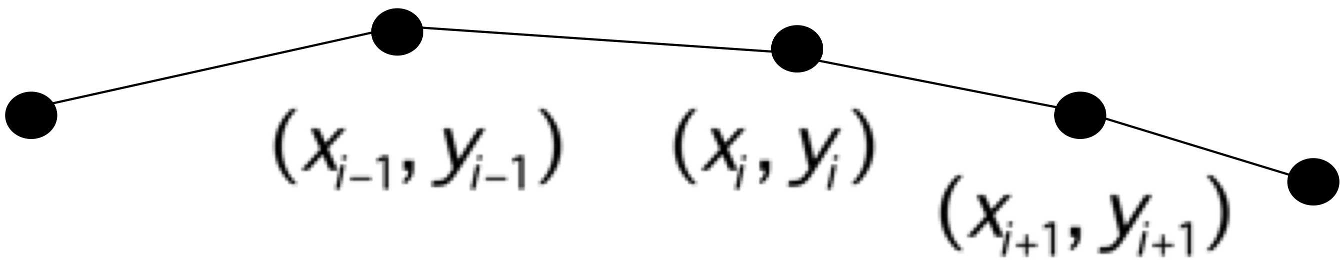


Deformable contours that

- Maximize the gradient along the curve;
- Minimize their deformation energy.

--> Interactive tools for contour detection that can be generalized to handle sophisticated models

Polygonal Approximation



Weighting coefficient

$$E = -\frac{\lambda}{N+1}\sum_{i=0}^{N} G(x_i, y_i) + \frac{1}{N}\sum_{i=1}^{N}\left((2x_i - x_{i-1} - x_{i+1})^2 + (2y_i - y_{i-1} - y_{i+1})^2\right)$$

Average gradient

Average sum of squared 2nd derivatives

$\approx$

Average sum of square curvature

Matrix Notation

$$
\begin{aligned}
E &= E_G + 1/2X^tKX + 1/2Y^tKY \\
X &= [x_1, \ldots, x_N]^t \\
Y &= [y_1, \ldots, y_N]^t \\
K &= \begin{bmatrix}
. & . & . & . & . & . & . & . & . & . & . & . & . \\
. & . & 1 & -4 & 6 & -4 & 1 & . & . & . & . & . & . \\
. & . & . & 1 & -4 & 6 & -4 & 1 & . & . & . & . & . \\
. & . & . & . & 1 & -4 & 6 & -4 & 1 & . & . & . & . \\
. & . & . & . & . & 1 & -4 & 6 & -4 & 1 & . & . & . \\
. & . & . & . & . & . & 1 & -4 & 6 & -4 & 1 & . & . \\
. & . & . & . & . & . & . & . & . & . & . & . & .
\end{bmatrix}
\end{aligned}
$$

Local Optimum

$$
\begin{aligned}
\frac{\delta E}{\delta X} &= \frac{\delta E_G}{\delta X} + KX = 0 \\
\frac{\delta E}{\delta Y} &= \frac{\delta E_G}{\delta Y} + KY = 0
\end{aligned}
$$

But K is not invertible!

Dynamics

Embed curve in a viscous medium and solve at each step:

$$0 = \frac{\partial E}{\partial X} + \alpha \frac{dX}{dt} = \frac{\partial E_G}{\partial X} + KX + \alpha \frac{dX}{dt}$$

$$0 = \frac{\partial E}{\partial Y} + \alpha \frac{dY}{dt} = \frac{\partial E_G}{\partial Y} + KY + \alpha \frac{dY}{dt}$$

Iterating

At every step:

$$
\begin{aligned}
0 &= \frac{\delta E_G}{\delta X} + KX_t + \alpha(X_t - X_{t-1}) \quad \Rightarrow \quad (K + \alpha I)X_t = \alpha X_{t-1} - \frac{\delta E_G}{\delta X} \\
0 &= \frac{\delta E_G}{\delta Y} + KY_t + \alpha(Y_t - Y_{t-1}) \quad \Rightarrow \quad (K + \alpha I)Y_t = \alpha Y_{t-1} - \frac{\delta E_G}{\delta Y}
\end{aligned}
$$

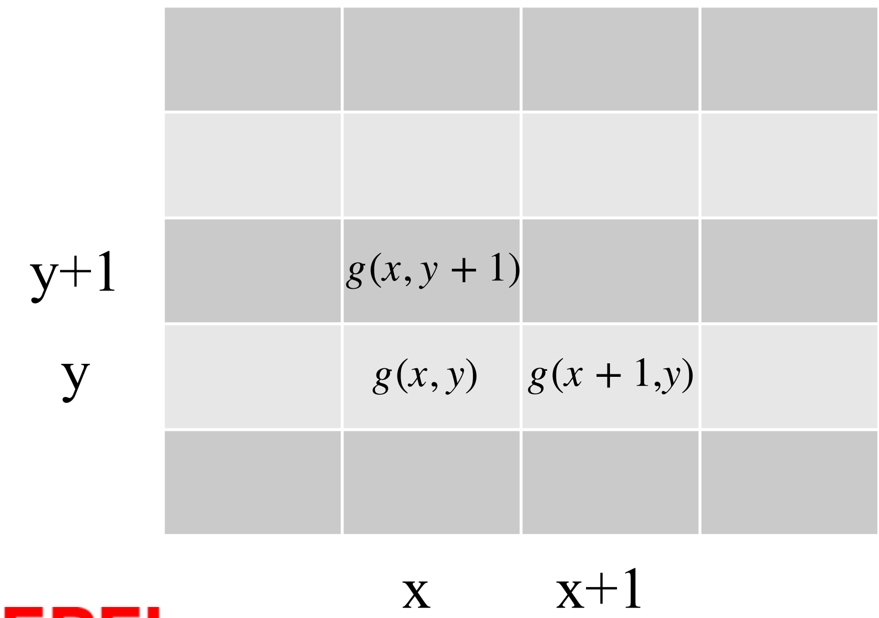
→ Solve two linear equations at each iteration.

Derivatives of the Image Gradient

$$E_G = -\frac{l}{N}\sum_{i=1}^{N} G(x_i, y_i)$$

$$\frac{\partial E_G}{\partial X} = \begin{bmatrix} \frac{\partial E_G}{\partial x_1} & \cdots & \frac{\partial E_G}{\partial x_N} \end{bmatrix} \quad , \quad \frac{\partial E_G}{\partial Y} = \begin{bmatrix} \frac{\partial E_G}{\partial y_1} & \cdots & \frac{\partial E_G}{\partial y_N} \end{bmatrix}$$

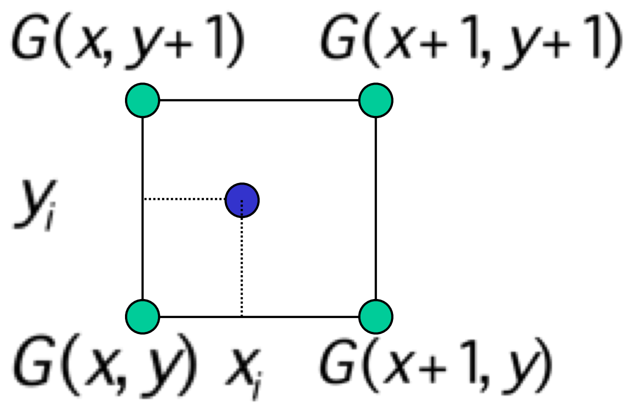
$$\frac{\partial E_G}{\partial x_i} = -\frac{l}{N}\frac{\partial G}{\partial x_i}(x_i, y_i) \quad , \quad \frac{\partial E_G}{\partial y_i} = -\frac{l}{N}\frac{\partial G}{\partial y_i}(x_i, y_i)$$



- We have values of g for integer values of x and y.
- But x_i and y_i are not integers.

—> We need to interpolate.

Bilinear Interpolation



$$p = x_i - x$$

$$q = y_i - y$$

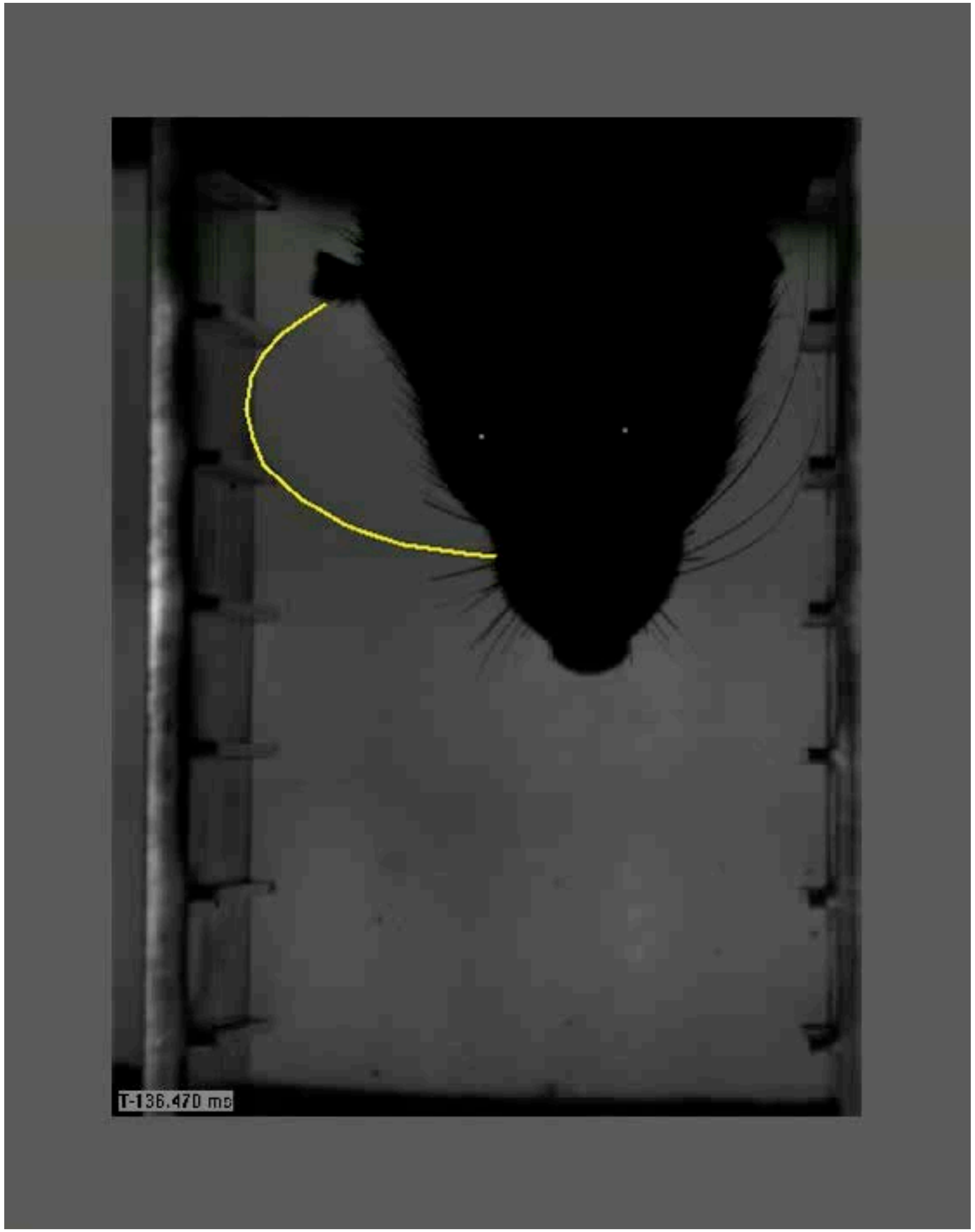
$$G(x_i, y_i) = (1-p)(1-q)G(x,y) + (1-p)qG(x,y+1) + p(1-q)G(x+1,y) + pqG(x+1,y+1)$$

$$\frac{\partial G}{\partial x_i} = (1-q)(G(x+1,y) - G(x,y)) + q(G(x+1,y+1) - G(x,y+1))$$

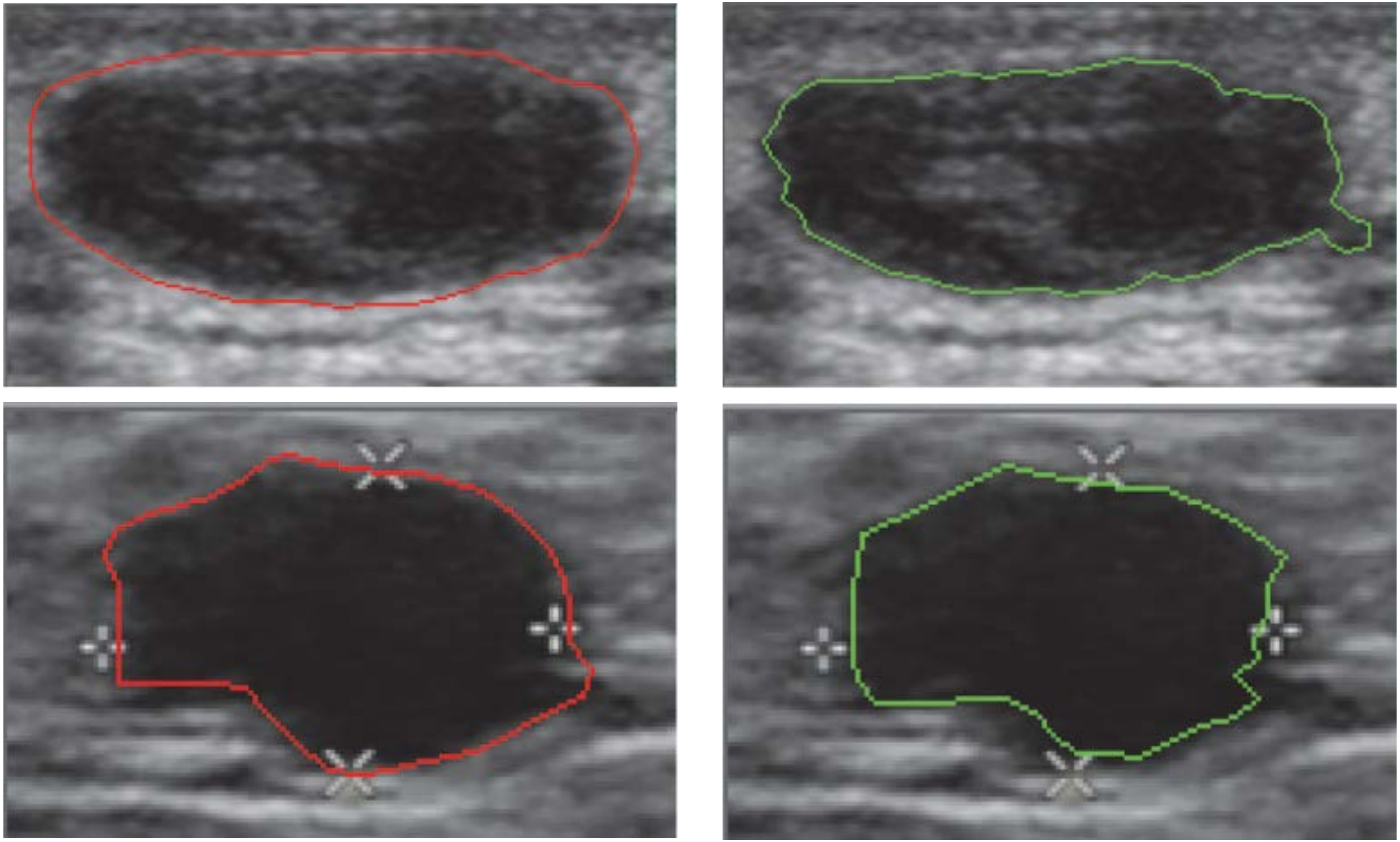
$$\frac{\partial G}{\partial y_i} = (1-p)(G(x,y+1) - G(x,y)) + p(G(x+1,y+1) - G(x+1,y))$$

EPFL

Open and Closed Snakes



Cysts Tumors in Ultrasound Images



Drawn by the physician. Refined by the Computer.

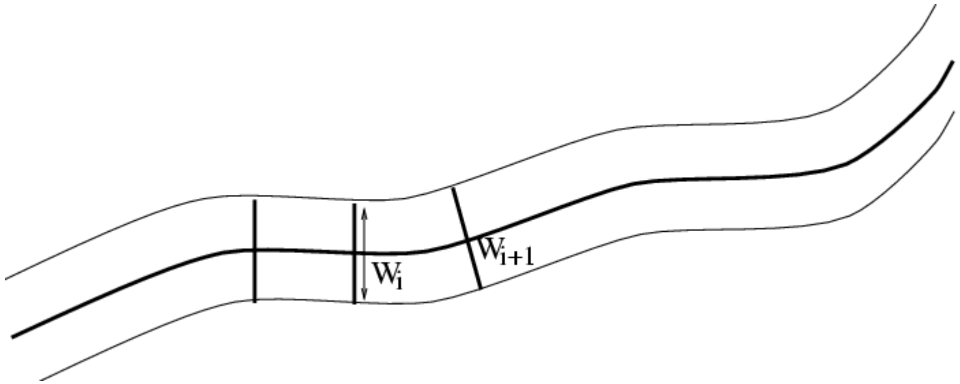
Luo et al., 2017

Network Snakes

--> Updated field boundaries.

Butenuth, PhD 2008

Ribbon Snakes



$$E = E_G + 1/2 X^t K X + 1/2 Y^t K Y + 1/2 W^t K_W W$$

$$W = [w_1, \ldots, w_N]^t$$

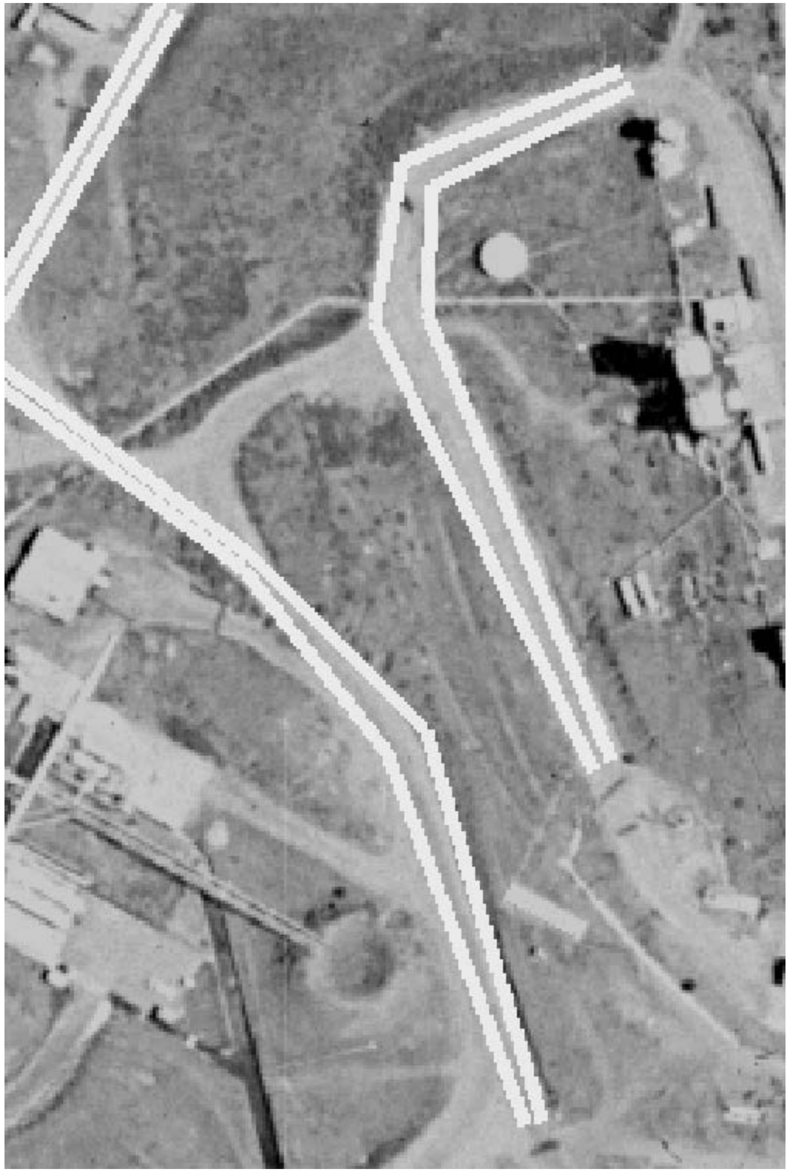
$$K_W = \begin{bmatrix} . & . & . & . & . & . & . & . & . & . & . \\ . & . & . & -1 & 2 & -1 & . & . & . & . & . \\ . & . & . & . & -1 & 2 & -1 & . & . & . & . \\ . & . & . & . & . & -1 & 2 & -1 & . & . & . \\ . & . & . & . & . & . & . & . & . & . & . \end{bmatrix}$$

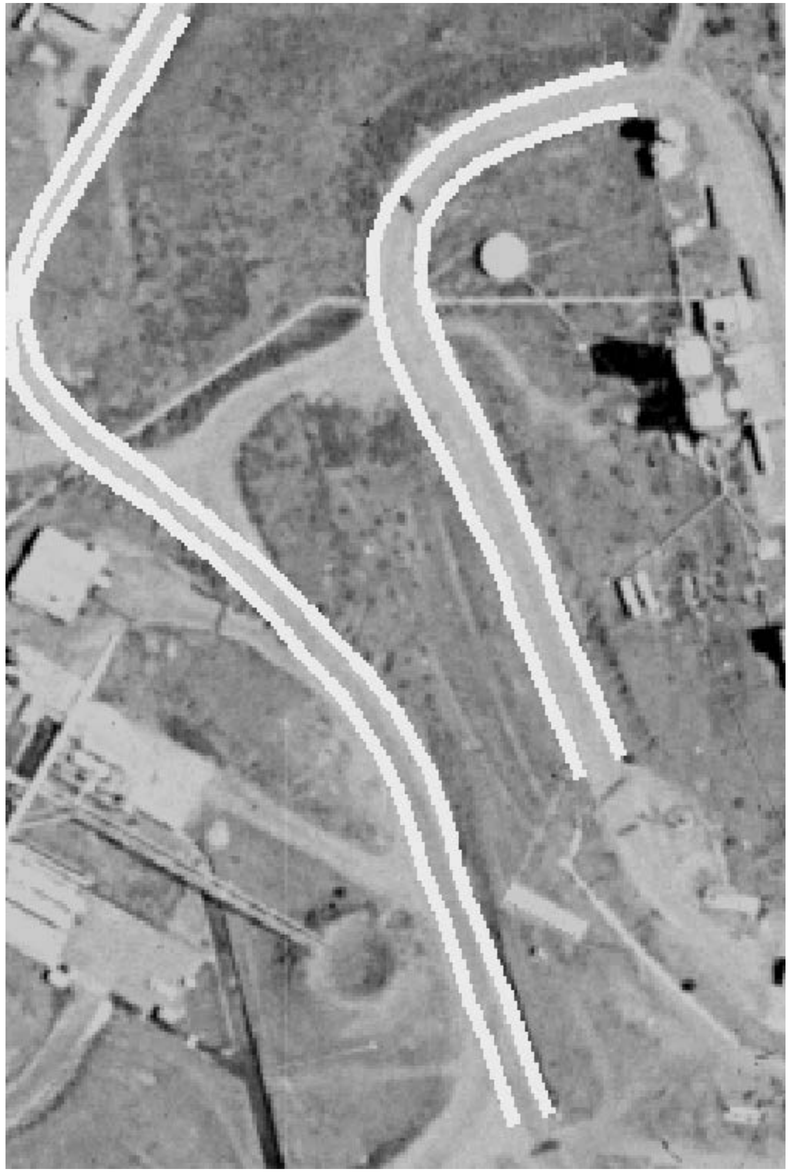
Dynamics Equations

$$
\begin{aligned}
(K + \alpha I) X_t &= \alpha X_{t-1} - \frac{\delta E_G}{\delta X} \\
(K + \alpha I) Y_t &= \alpha Y_{t-1} - \frac{\delta E_G}{\delta Y} \\
(K + \alpha I) W_t &= \alpha W_{t-1} - \frac{\delta E_G}{\delta W}
\end{aligned}
$$

→ Solve three linear equations at each iteration.

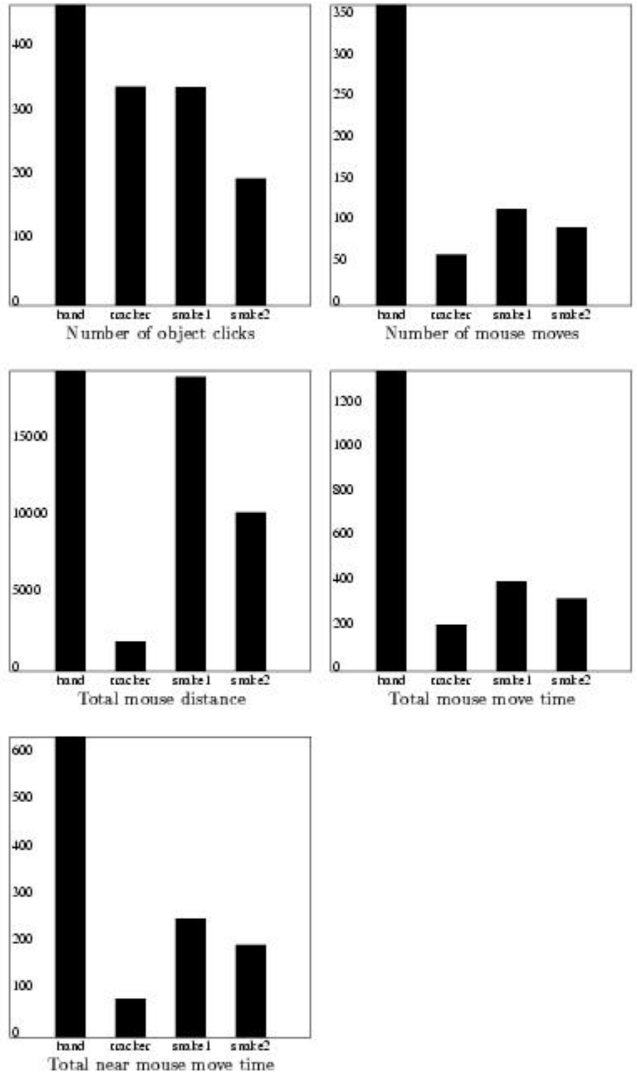
Delineating Roads





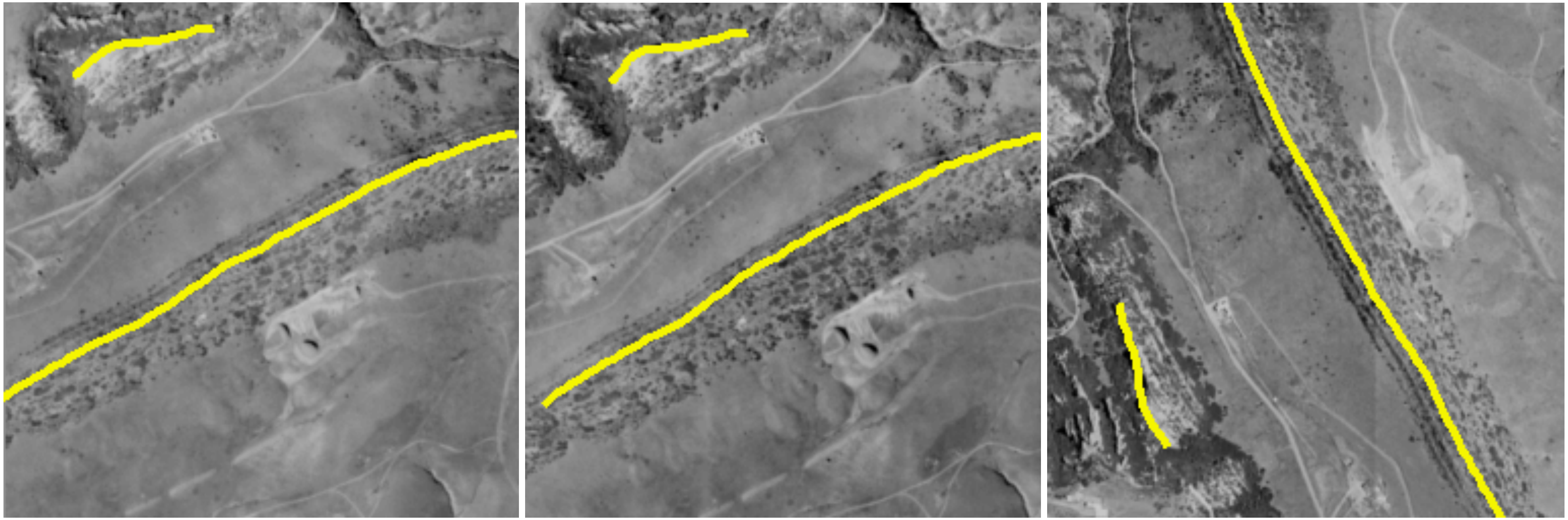
Delineating Roads

Evaluation

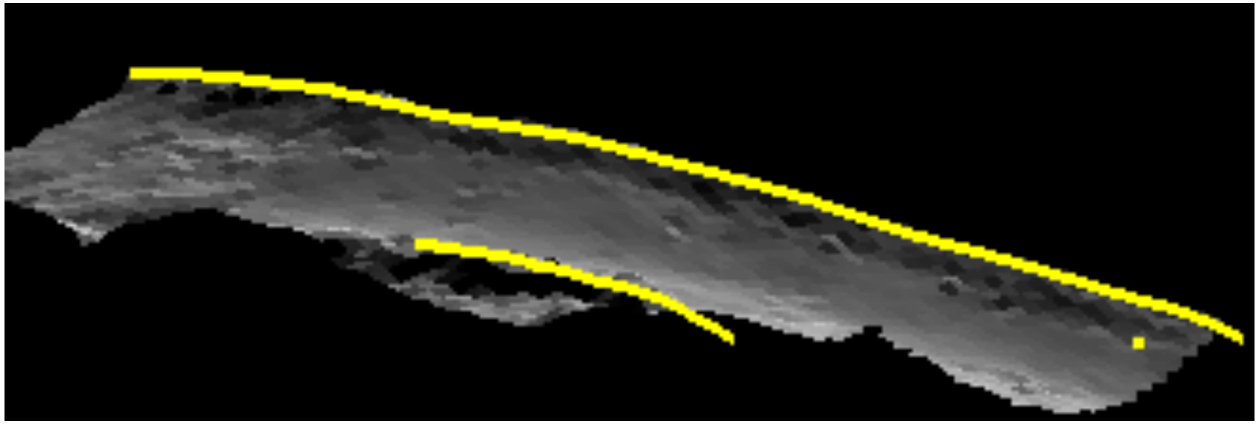


It takes far fewer clicks to trace the roads using semi-automated tools than doing entirely by hand.

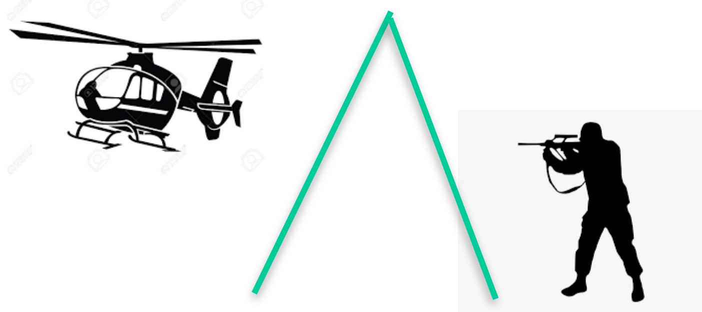
Modeling a Ridge Line in 3D



Three different views

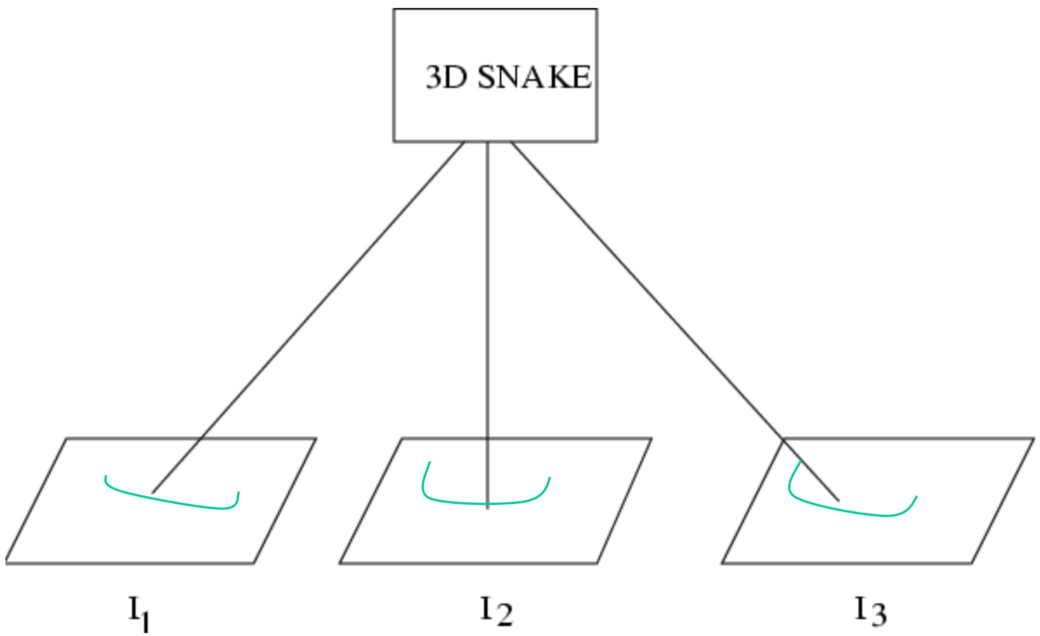


Synthetic side view.

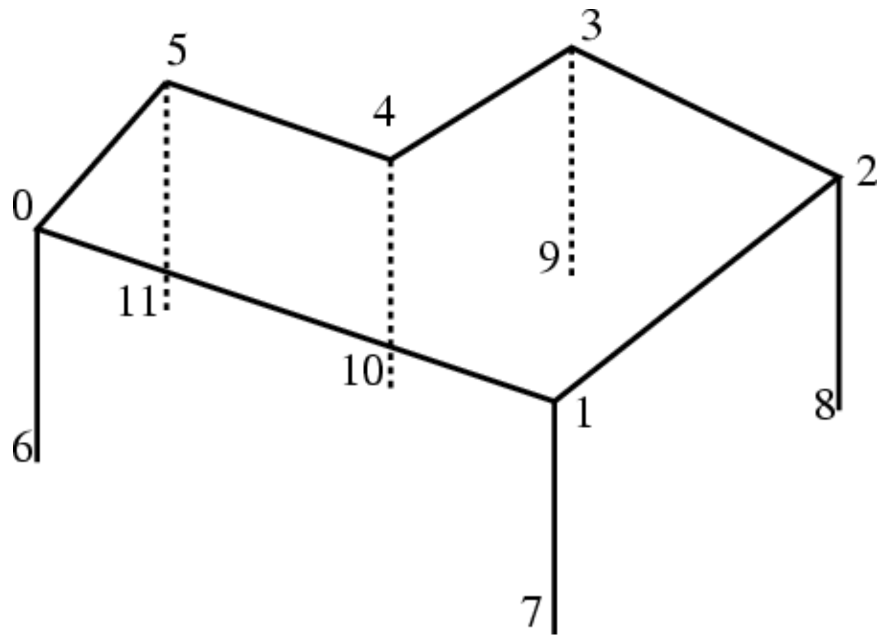


Modeling a Building in 3D

3D Snakes



Smooth 3—D snake



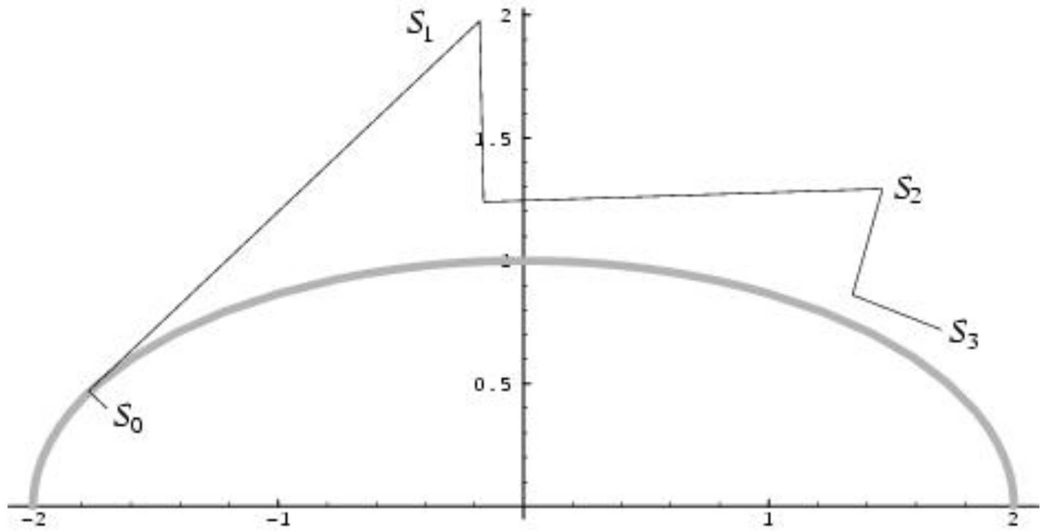
Rectilinear 3—D snake

Dynamics Equations

$$
\begin{aligned}
(K+\alpha I)X_t &= \alpha X_{t-1} - \frac{\delta E_G}{\delta X} \\
(K+\alpha I)Y_t &= \alpha Y_{t-1} - \frac{\delta E_G}{\delta Y} \\
(K+\alpha I)Z_t &= \alpha Z_{t-1} - \frac{\delta E_G}{\delta Z}
\end{aligned}
$$

→ Solve three linear equations at each iteration.

Constrained Optimization



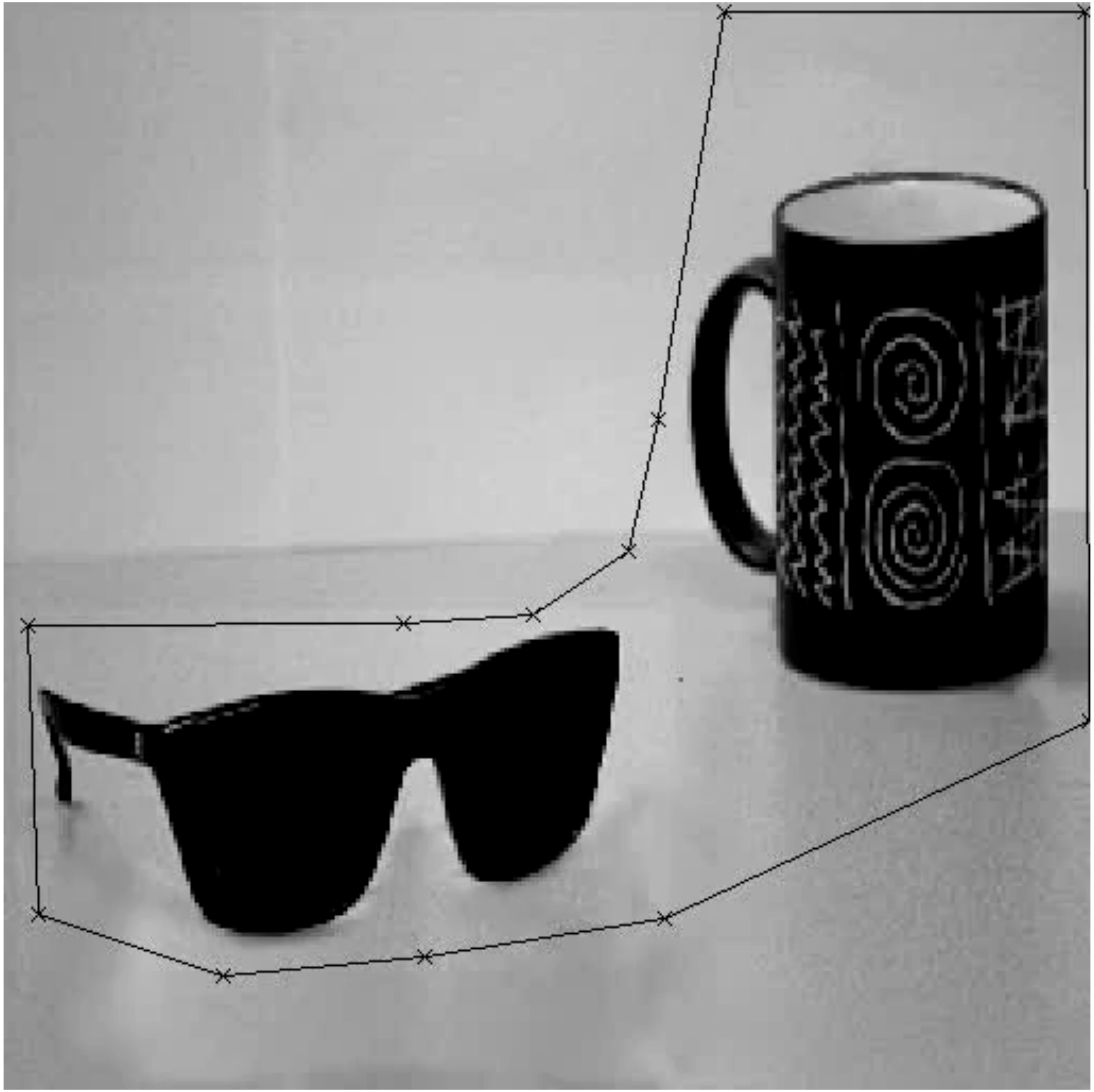
- Minimize F(S) subject to C(S) = 0

Site Modeling (1996)

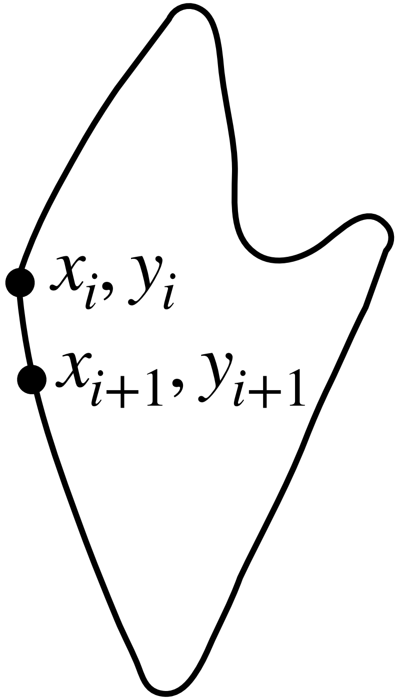
Site Modeling (2019)

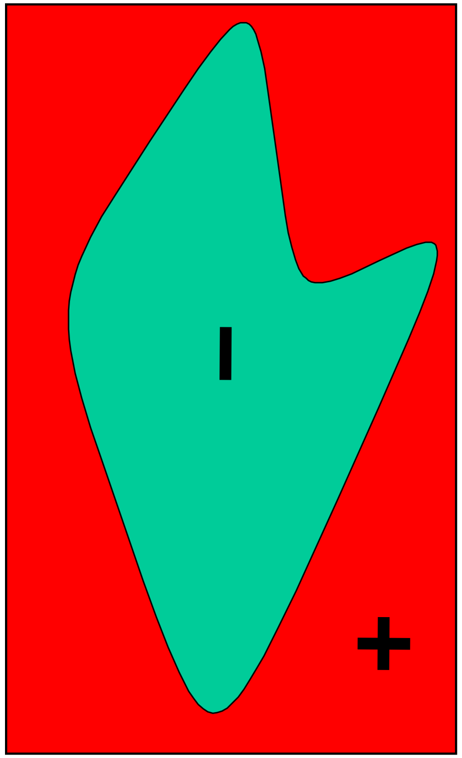
Site Modeling (2025)

Level Sets



Implicit vs Explicit





$z = \Phi(x, y)$,

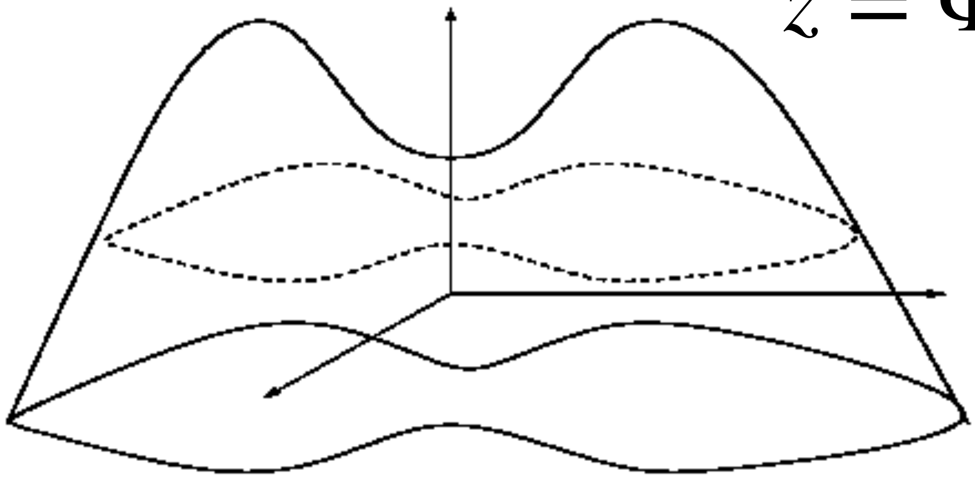
$z > 0$ outside,

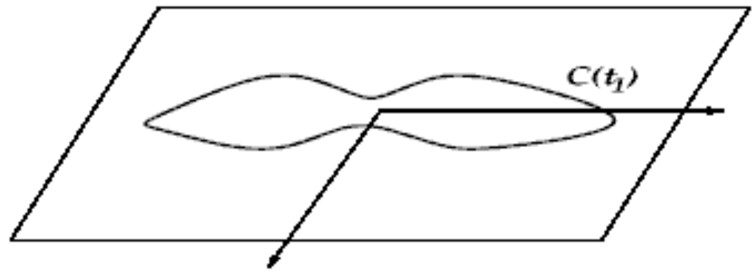
$z < 0$ inside,

—> Consider the curve as the zero level set of a surface.

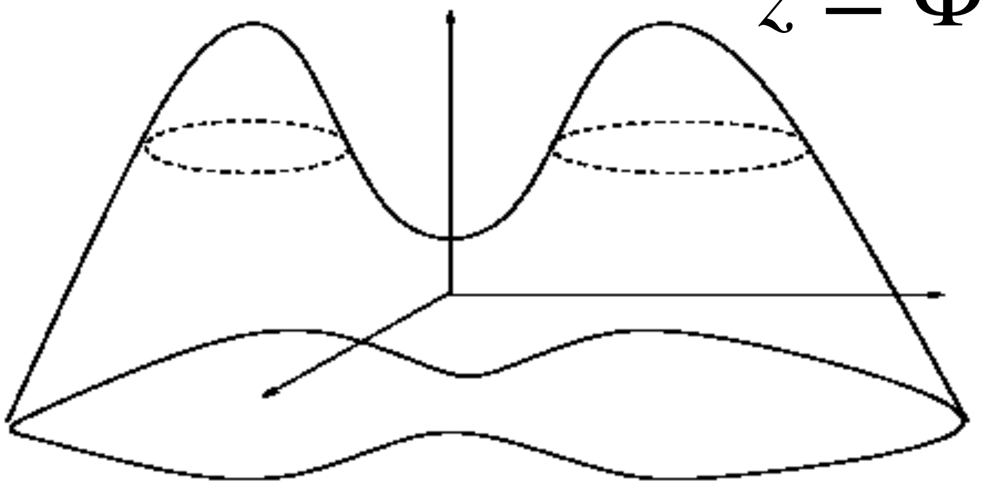
Topology Changes are Possible

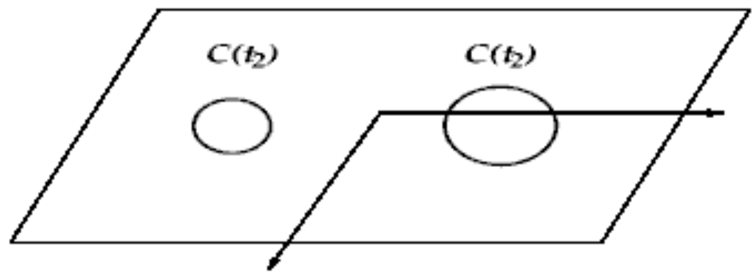
$$z = \Phi(x, y, t_1)$$





$$z = \Phi(x, y, t_2)$$





Curve Evolution

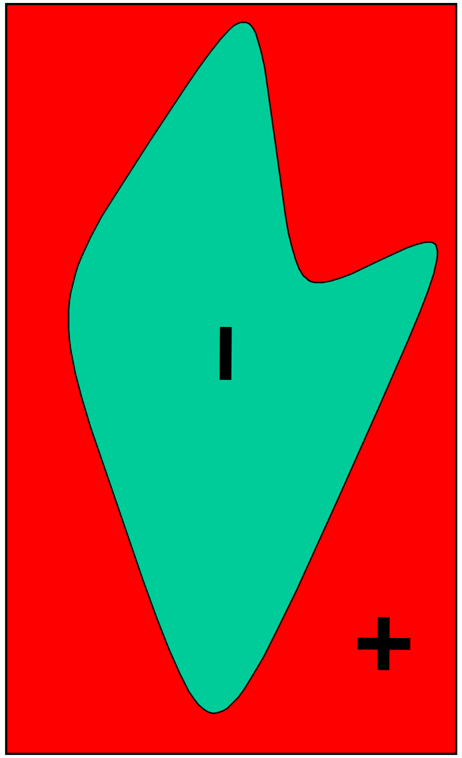
Consider the curve as the zero level set of the surface:

$$z = \Phi(x, y, t)$$

Evolution equation:

$$\begin{aligned} 0 &= \Phi_t + \beta(\kappa)\,|\nabla\Phi| \\ \text{where } \kappa &= \frac{\Phi_{xx}\Phi_y^2 - 2\Phi_{xy}\Phi_x\Phi_y + \Phi_{yy}\Phi_x^2}{\Phi_x^2 + \Phi_y^2} \end{aligned}$$

curvature



$\beta(\kappa)$ is the speed at which the surface deforms.

Level Set Smoothing

Smoothing occurs when $\beta(\kappa) = -\kappa$

Desirable properties:

- Converges towards circles.
- Total curvature decreases.
- Number of curvature extrema and zeros of curvature decreases.

Relationship with Gaussian smoothing:

- Analogous to Gaussian smoothing of boundary over the short run, but does not cause self-intersections or overemphasize elongated parts.
- Can be implemented by Gaussian smoothing the characteristic function of a region.

Shape Recovery

$$\textbf{Evolution equation:} \quad 0 = \Phi_t + \beta(\kappa)\,|\nabla\Phi|$$

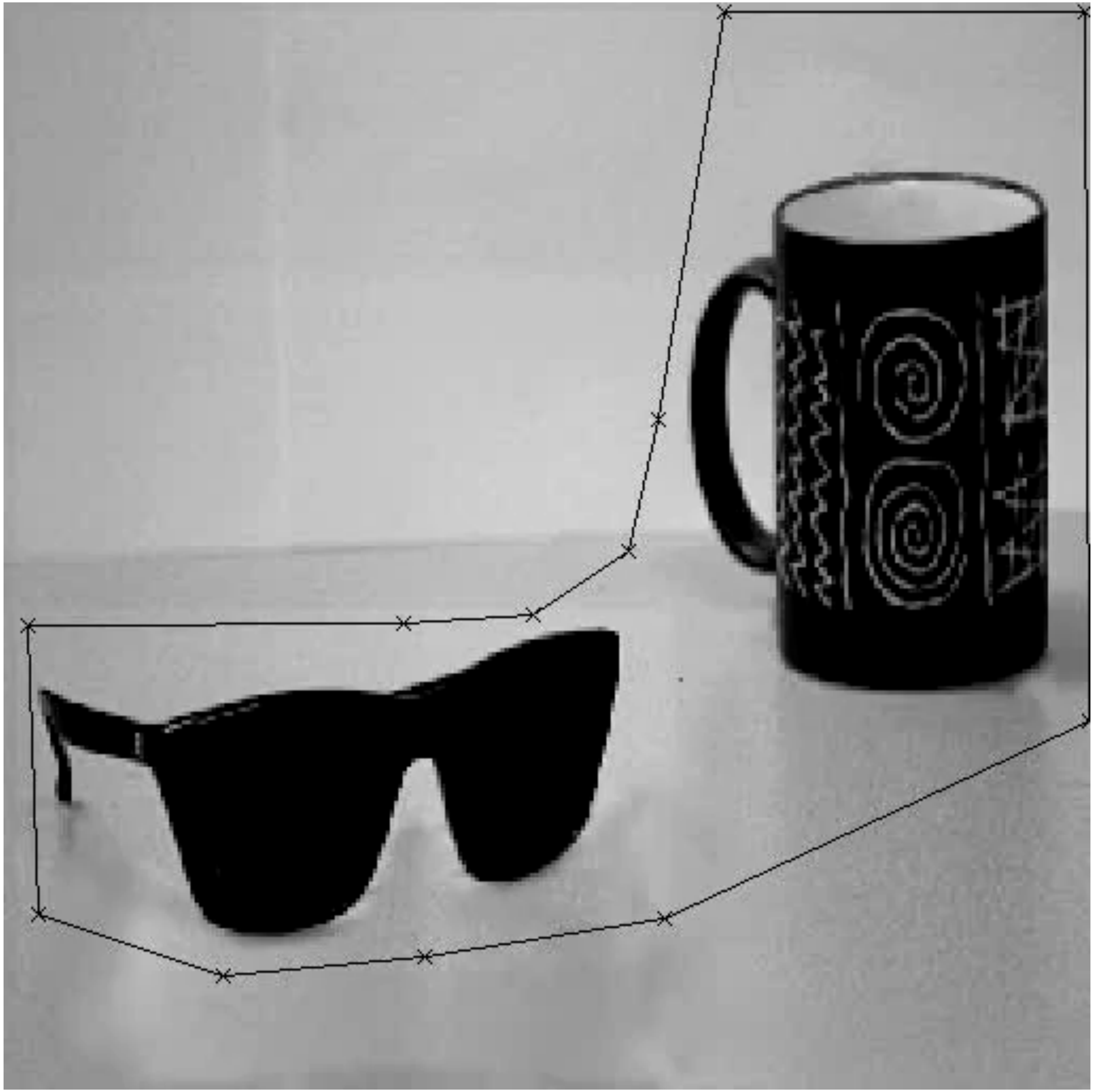
$$\begin{aligned} \text{where: } \beta(\kappa) &= k_I(1-\epsilon\kappa) \\ k_I &= \frac{1}{1+\nabla I} \end{aligned}$$

→ Expansion stops at the boundaries.

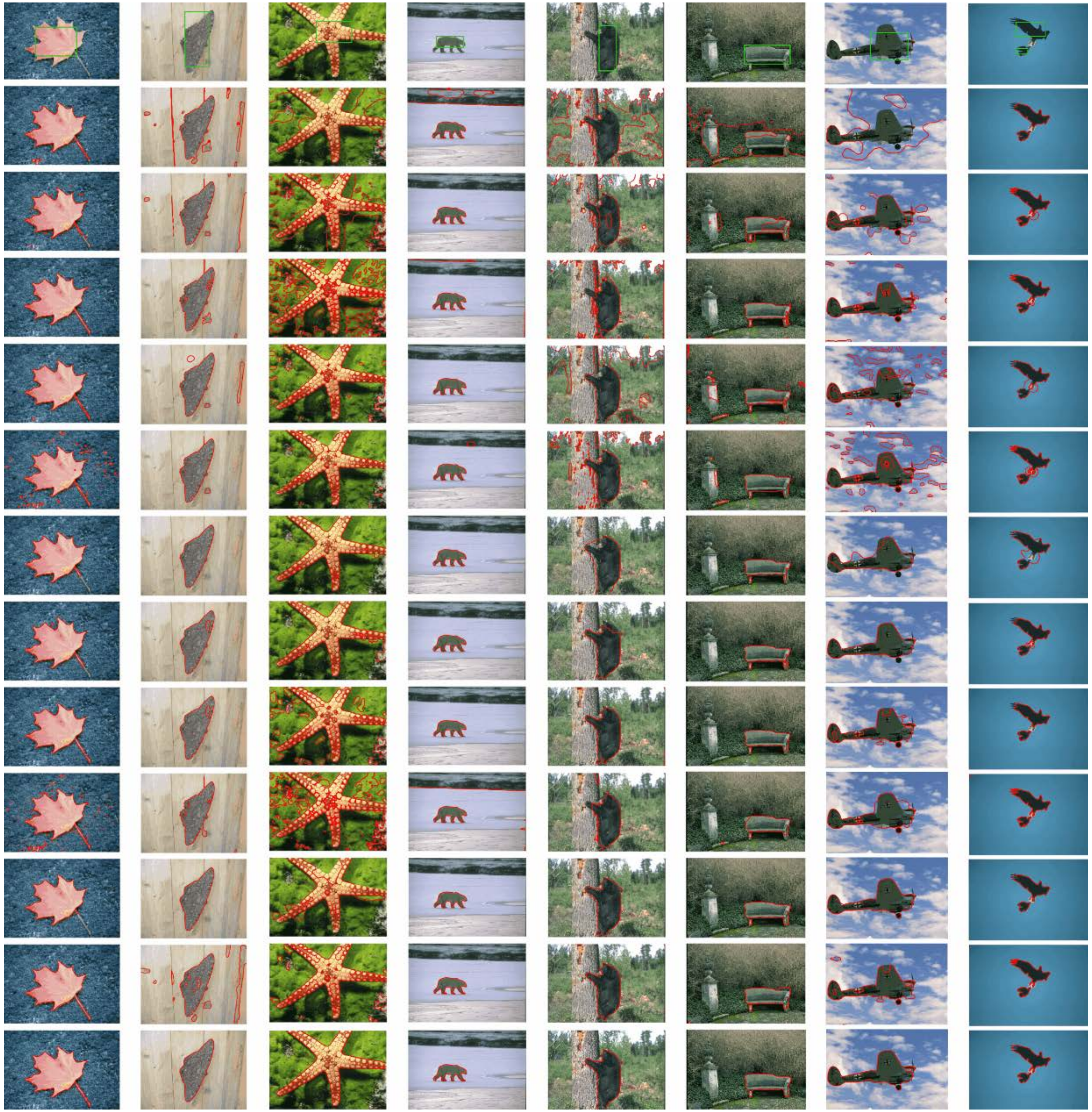
EPFL

Level Sets

Level Sets



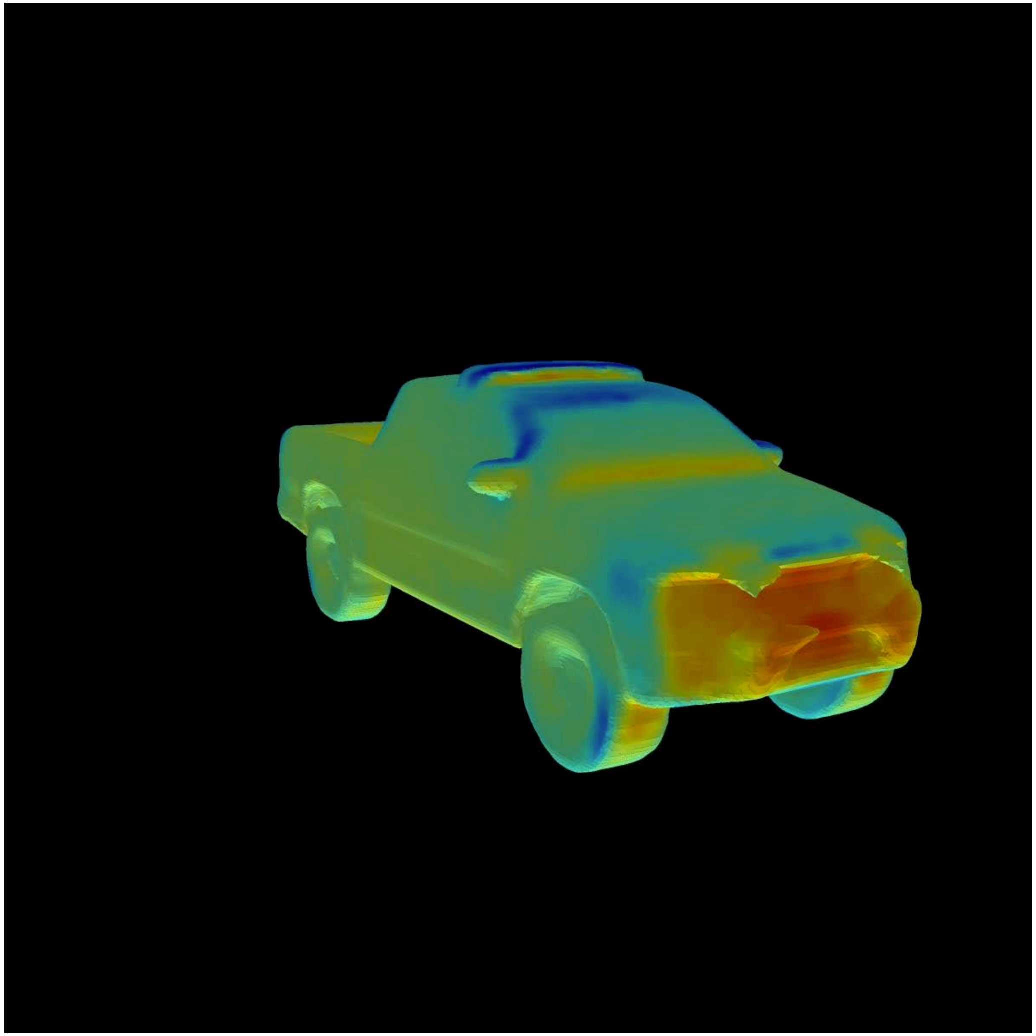
Newer Implementations



Use a DNN to compute the energy function.

Deep Implicit Surfaces

Changing the topology

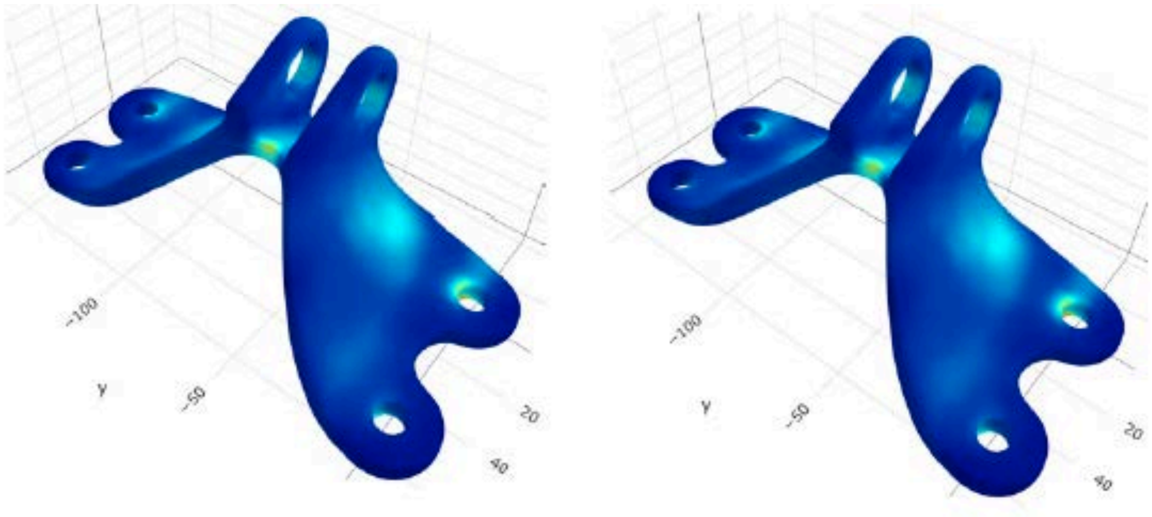


Minimizing the drag

EPFL

Guillard et al, PAMI'24

Designing Brackets



- Predicting accurate stress and displacement on new geometries.
- Finding the best compromise between strength and weight.

Techniques

Semi-Automated Techniques:

- Dynamic programming
- Deformable Models

Fully Automated Techniques:

- Hough transform
- Graph Based Approaches

Finding Lines

Input:

- Canny edge points.
- Gradient magnitude and orientation.

Output:

- All straight lines in image.

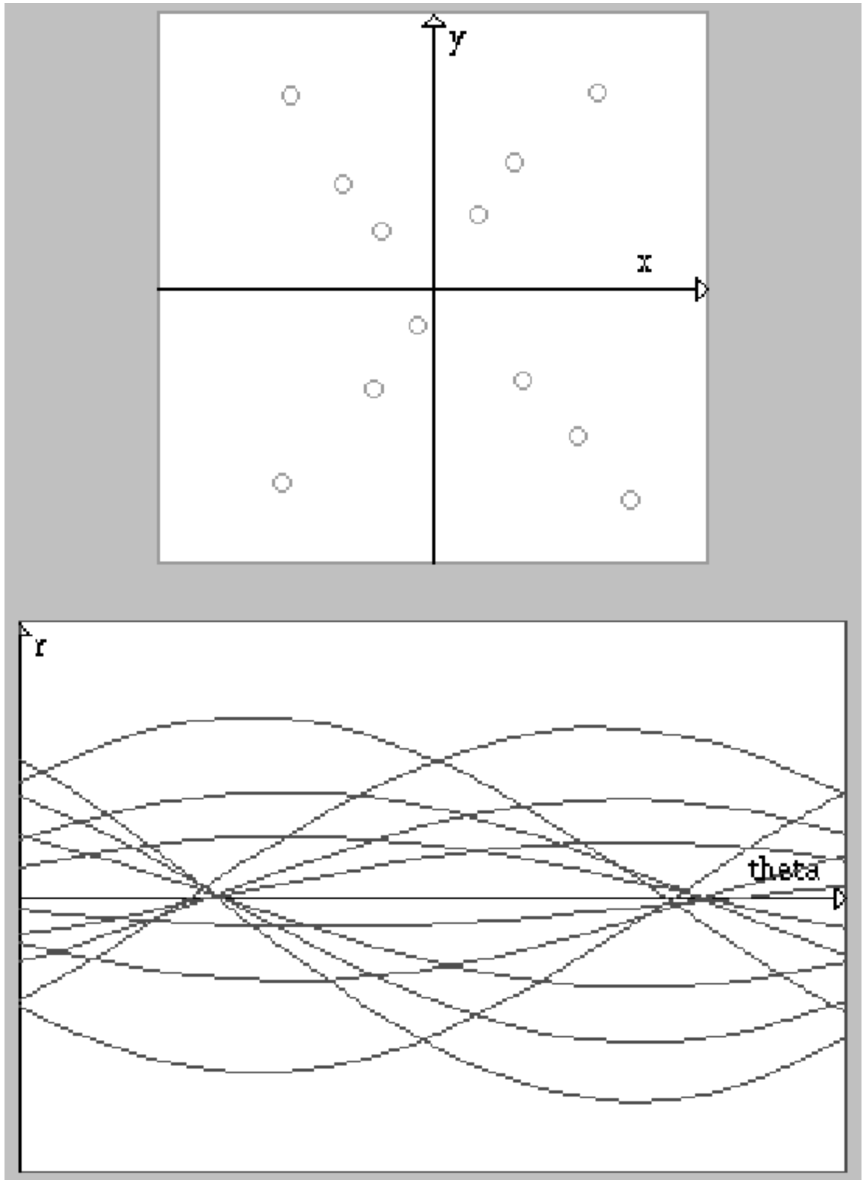
Hough Transform

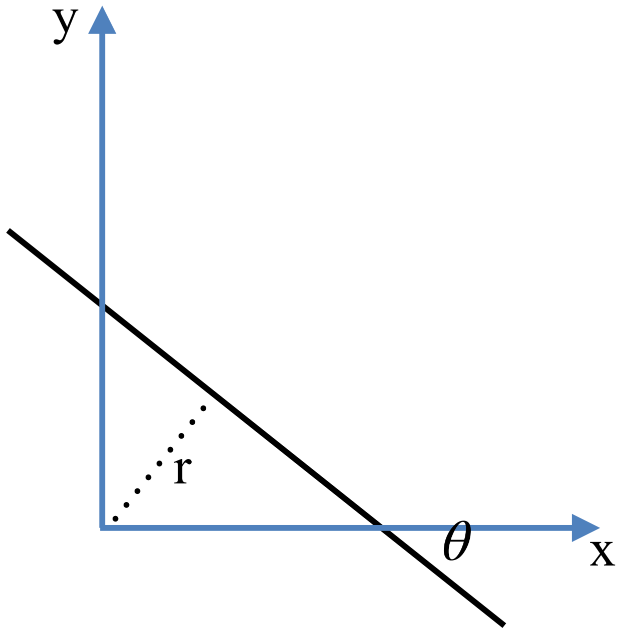
Given a parametric model of a curve:

- Map each contour point onto the set of parameter values for which the curves passes through it.

- Find the intersection for all parameter sets thus mapped.

Voting Scheme





$$x\cos(\theta) + y\sin(\theta) = r\ ,\ \ 0 \leq \theta \leq \pi$$

Synthetic Lines

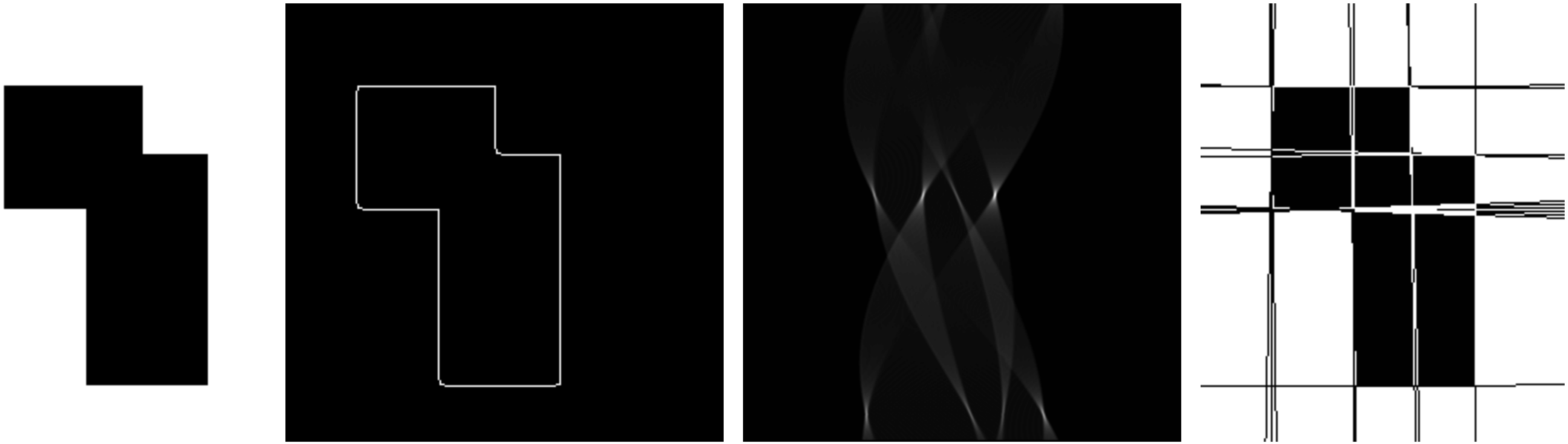
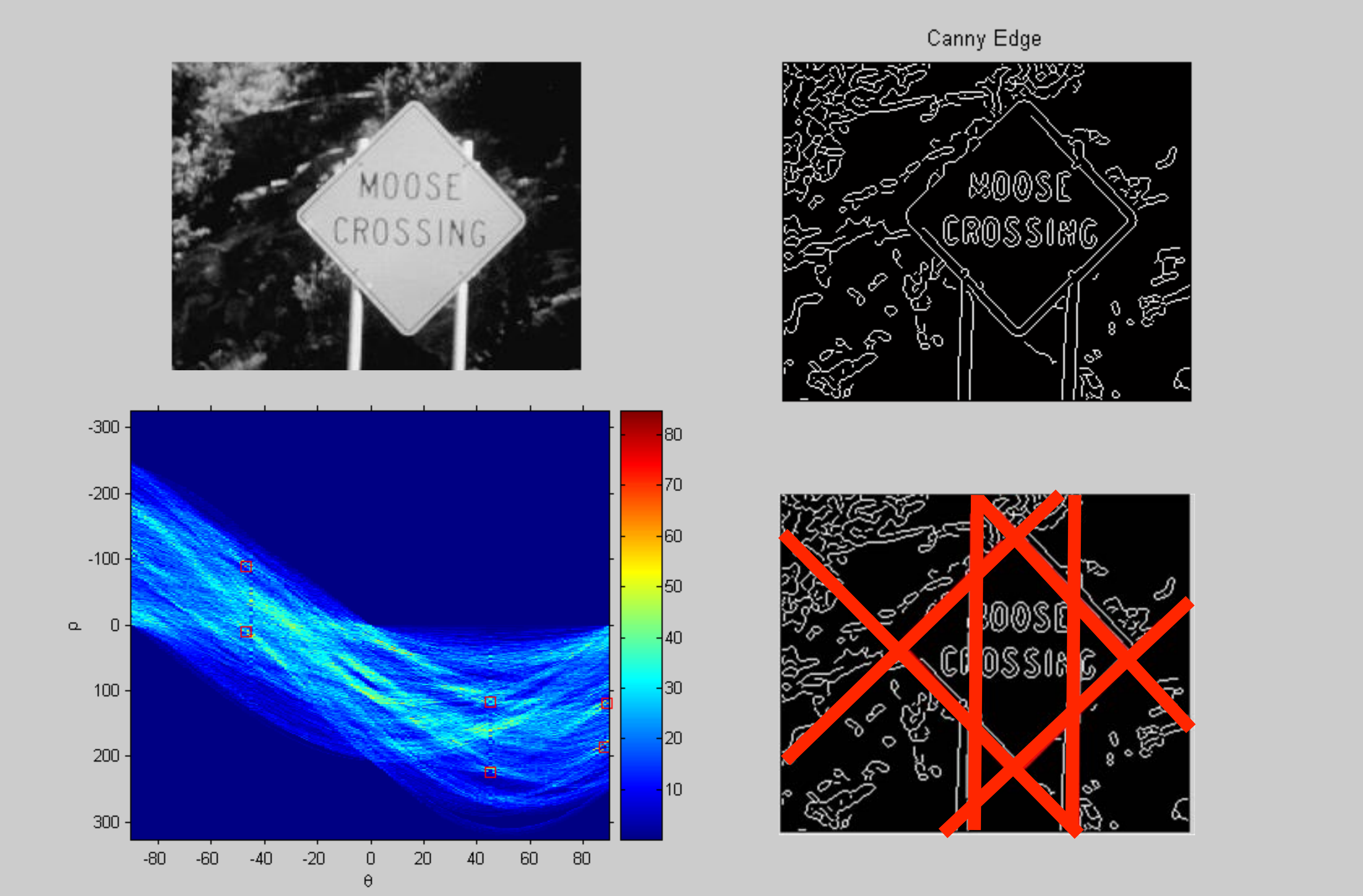


Image Contours Accumulator Lines

Once the contour points are associated to individual lines, you can perform least squares fitting.

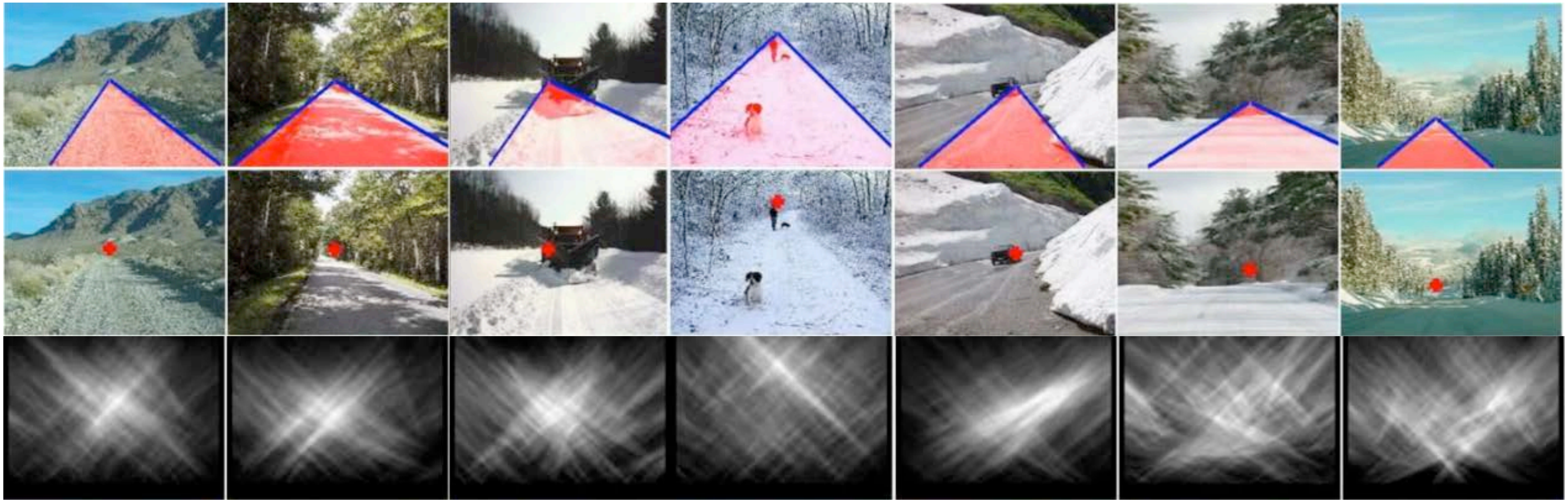
Real Lines



Canny Edge
MOOSE
CROSSING
ρ
θ
-300
-200
-100
0
100
200
300
-80
-60
-40
-20
0
20
40
60
80
10
20
30
40
50
60
70
80

Road Lines

Road Edges

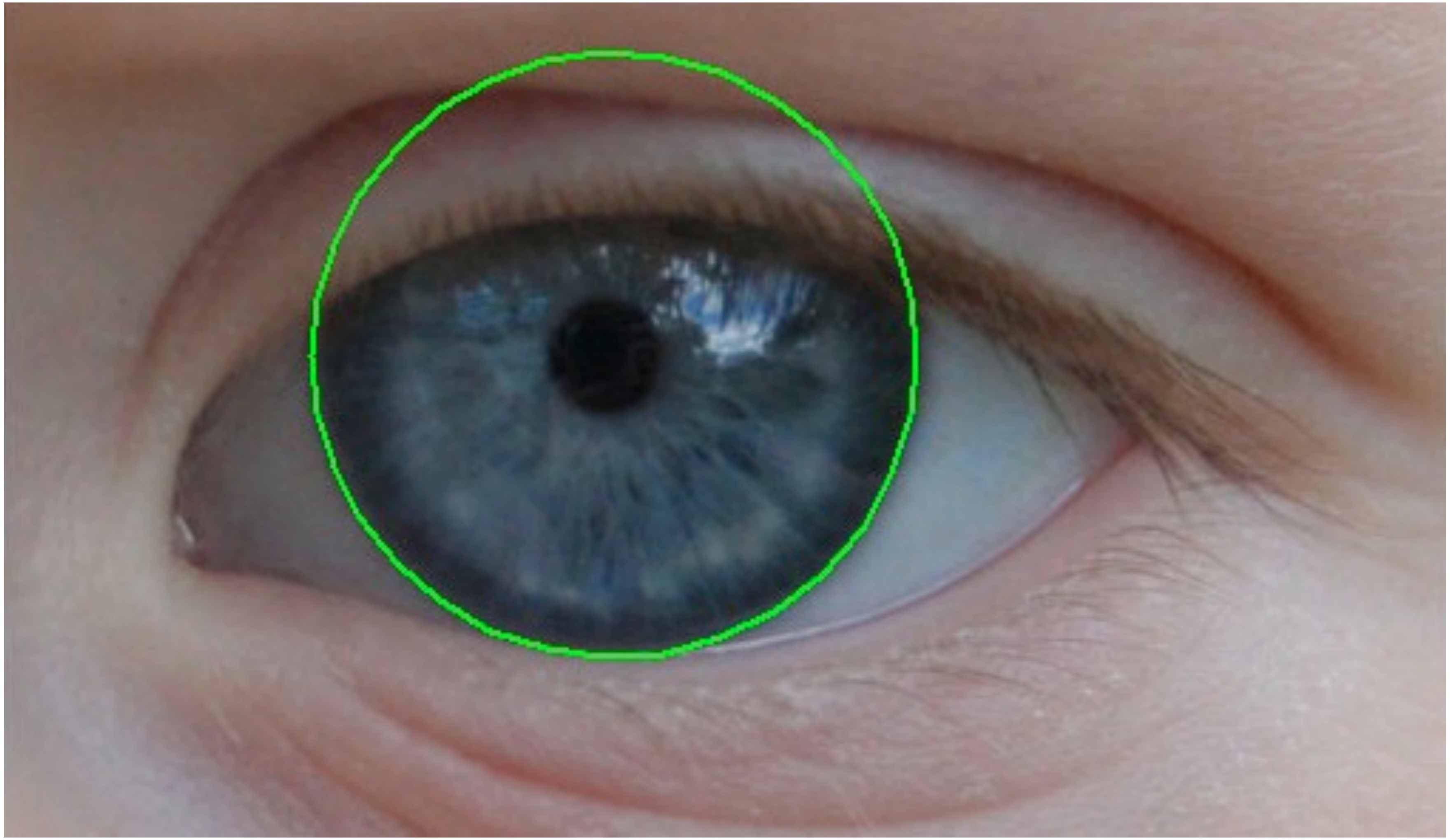


Kong et al. CVPR'09

Generic Algorithm

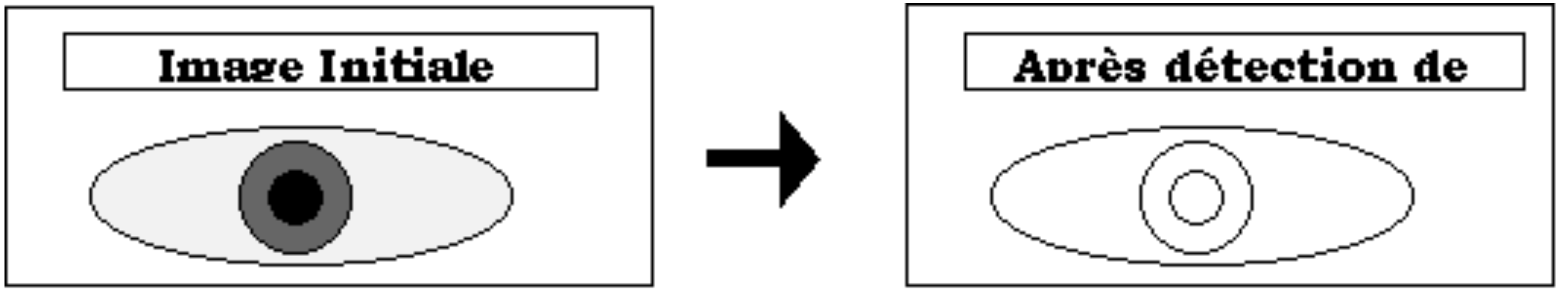
- Quantize parameter space with 1 dimension per parameter.
- Form an accumulator array.
- For each point in the gradient image such that the gradient strength exceeds a threshold, increment appropriate element of the accumulator.
- Find local maxima in the accumulator.

Iris Detection



Occlusions

In theory:



In practice:

Circle Detection

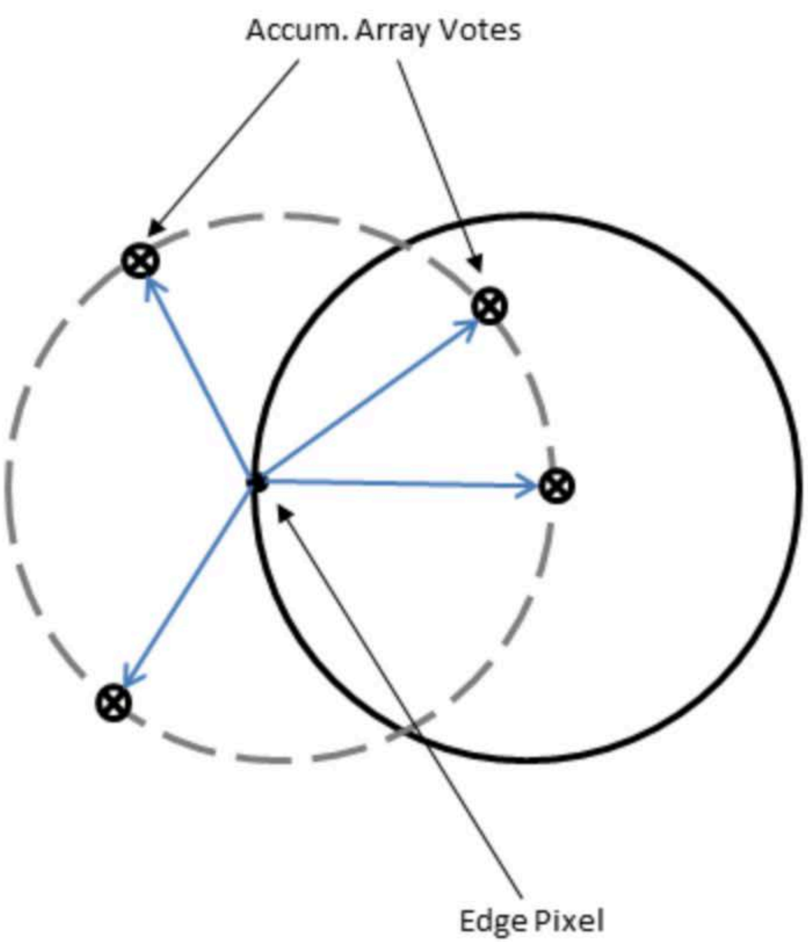
Circle of equation:

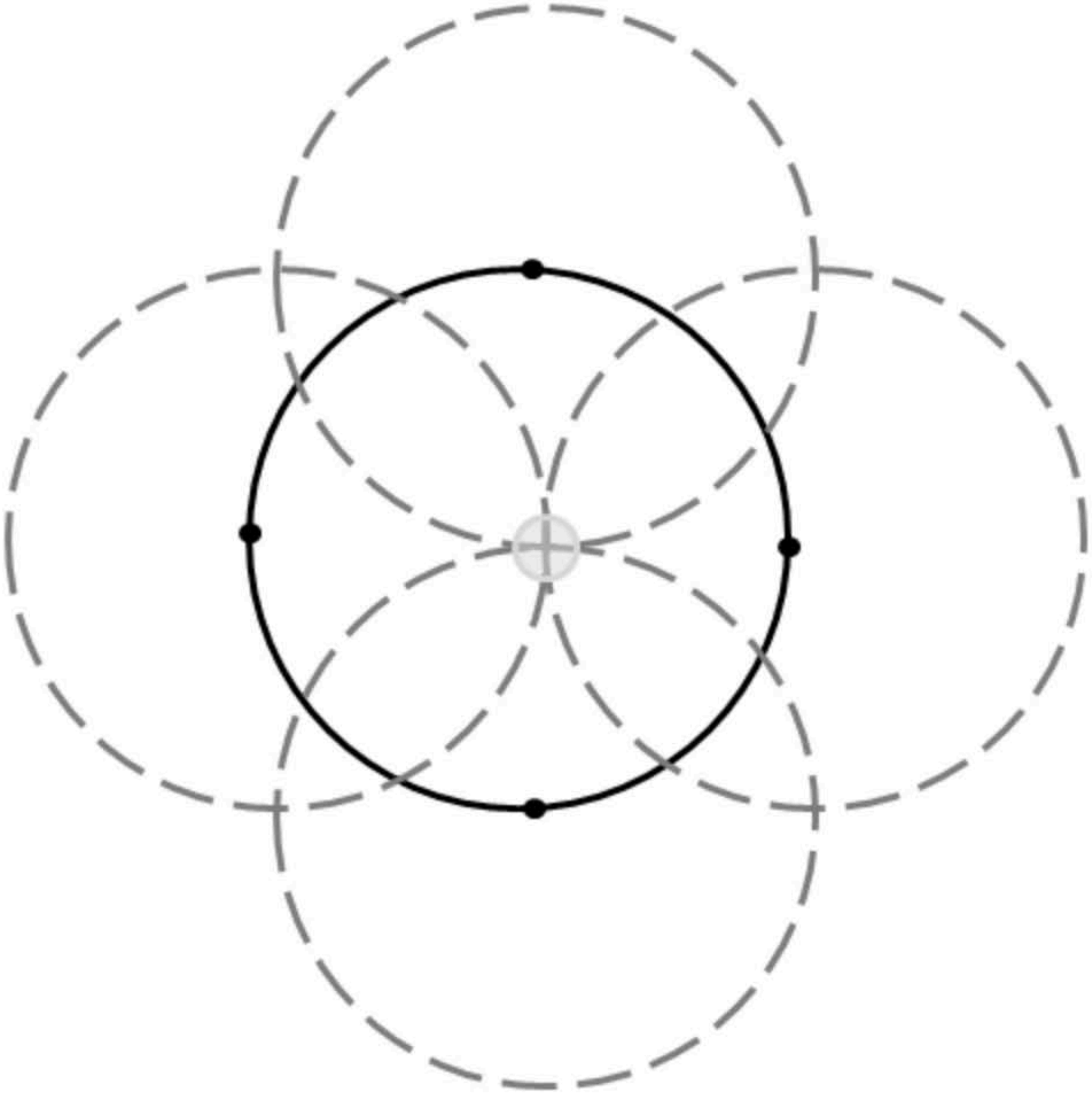
$$
\begin{aligned}
x &= x_0 + r\cos(\theta) \\
y &= y_0 + r\sin(\theta)
\end{aligned}
$$

Therefore:

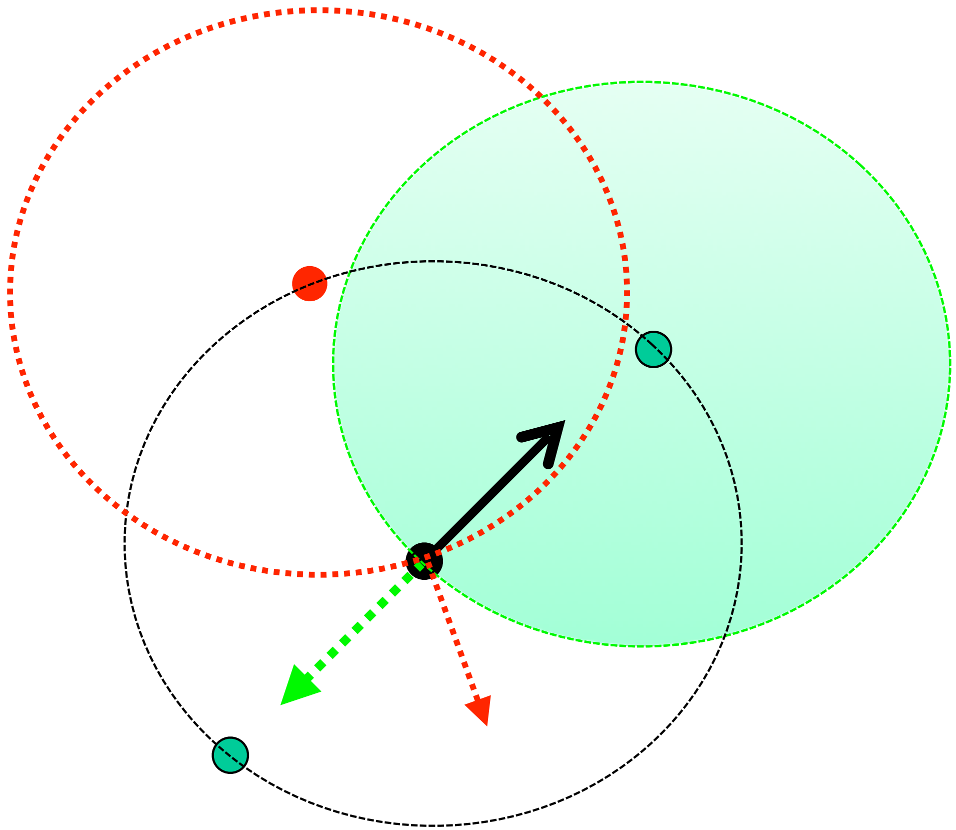
$$
\begin{aligned}
x_0 &= x - r\cos(\theta) \\
y_0 &= y - r\sin(\theta)
\end{aligned}
$$

Plausible Circles





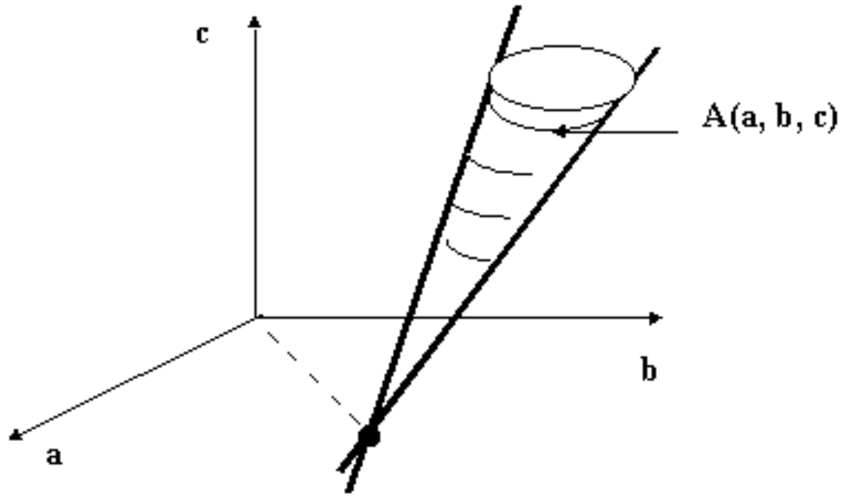
Gradient Orientation



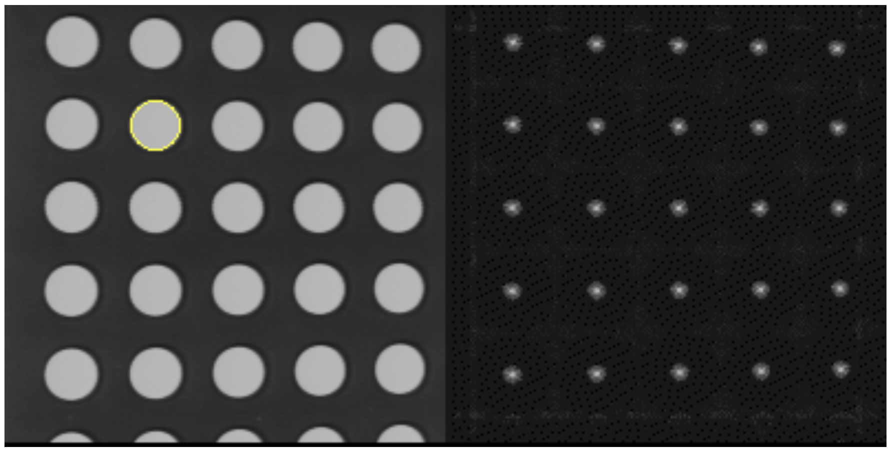
Can vote either along the entire circle or only at two points per value of the radius.

Simple Image

Voting scheme:

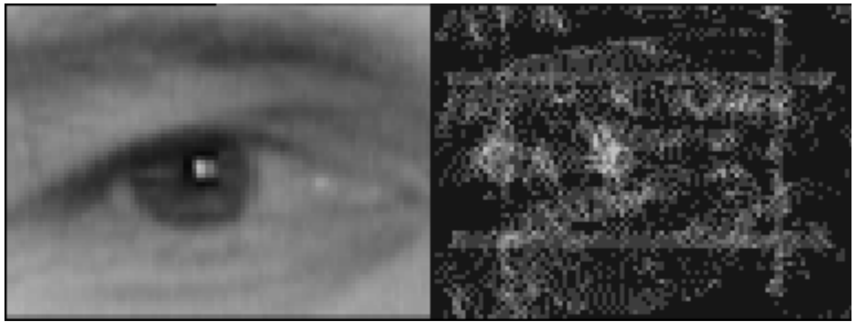


Result:

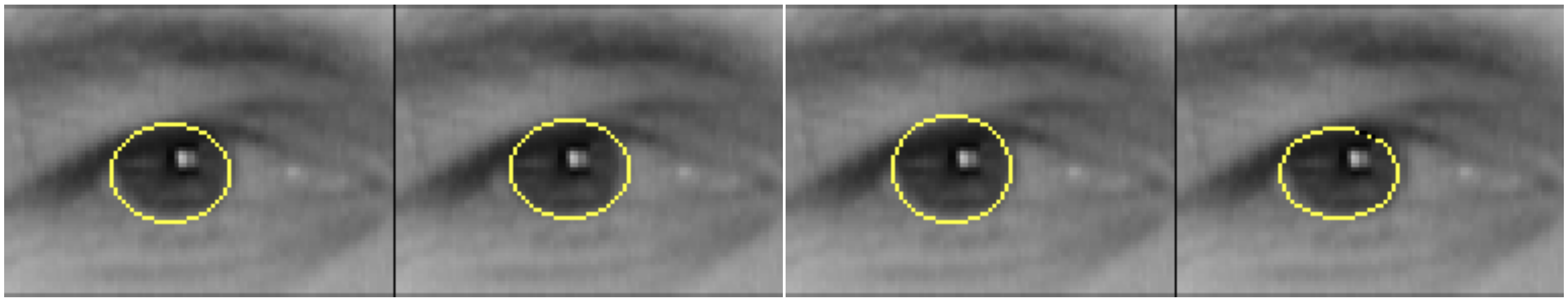


Eye Image

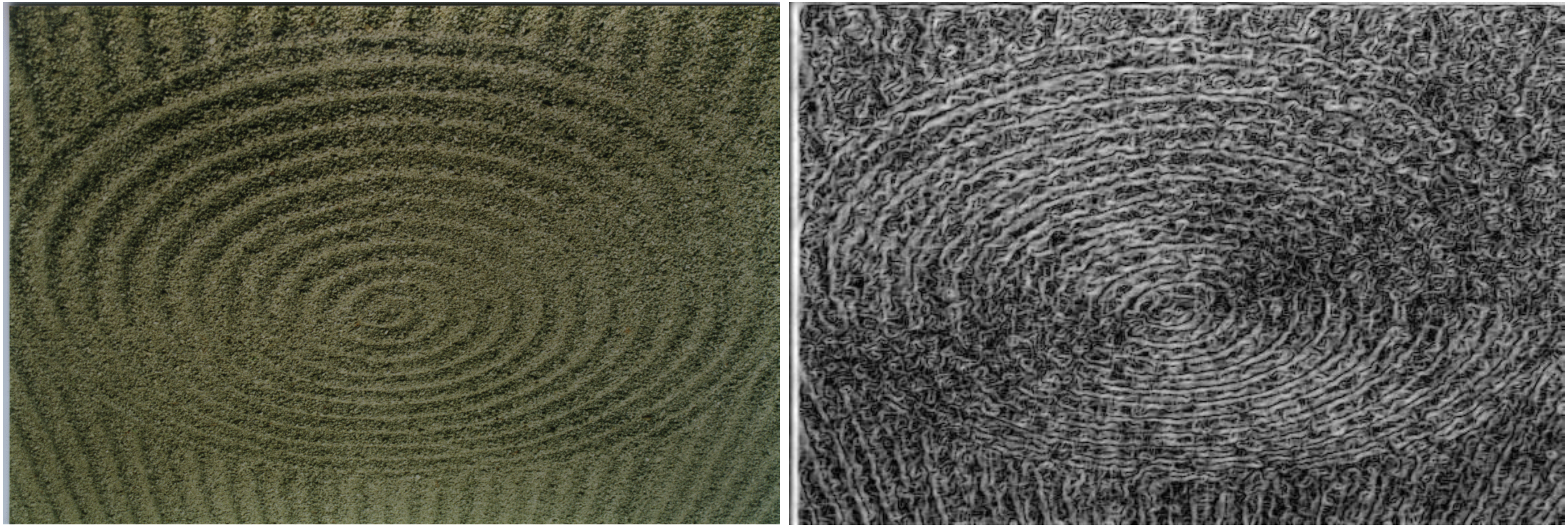
Image and accumulator:



Best four candidates:



Ellipses



Ellipse Detection

Ellipse of equation:

$$\begin{aligned} x &= x_0 + a\cos(\theta) \\ y &= y_0 + b\sin(\theta) \end{aligned}$$

Therefore:

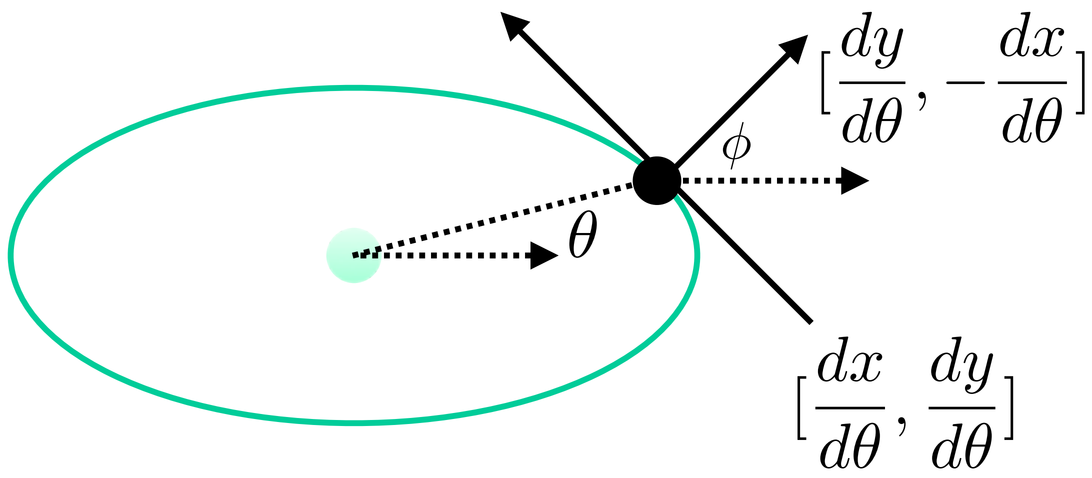
$$\begin{aligned} x_0 &= x - a\cos(\theta) \\ y_0 &= y - b\sin(\theta) \end{aligned}$$

Gradient Orientation

For each ellipse point:

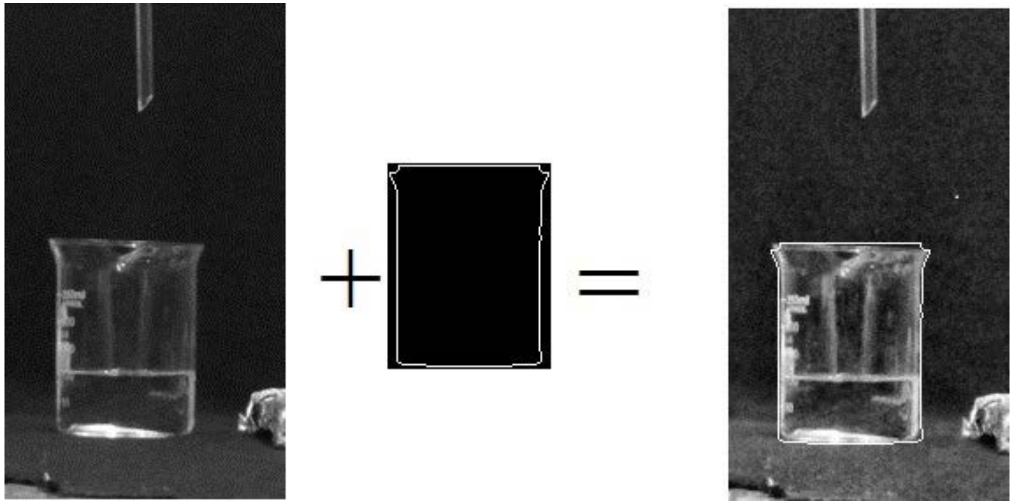
$$\frac{dx}{d\theta} = -a\sin(\theta)$$
$$\frac{dy}{d\theta} = \quad b\cos(\theta)$$

$$\phi = \operatorname{atan}\left(-\frac{dx}{d\theta}, \frac{dy}{d\theta}\right)$$
$$= \operatorname{atan}(a\sin(\theta), b\cos(\theta))$$
$$= \operatorname{atan}\left(\frac{a}{b}\tan(\theta)\right)$$
$$\tan(\phi) = \frac{a}{b}\tan(\theta)$$
$$\Rightarrow \theta = \operatorname{atan}\left(\frac{b}{a}\tan(\phi)\right)$$



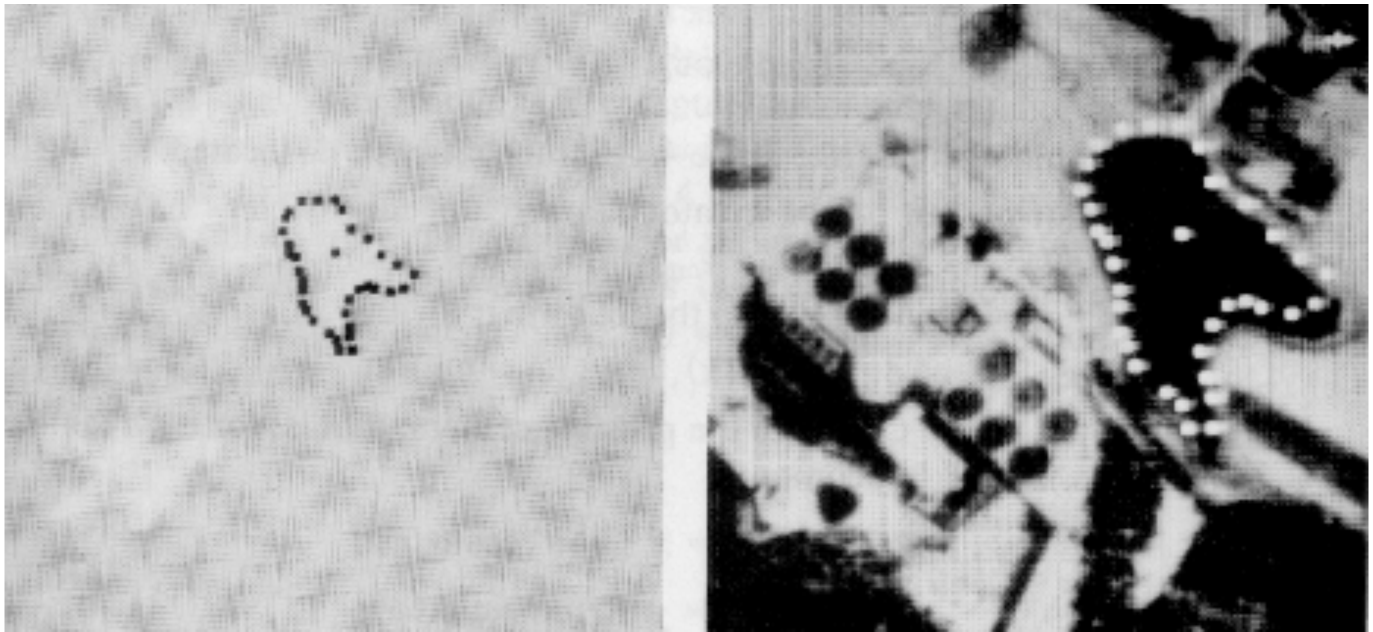
The accumulator need only be incremented for this θ.

Generalized Hough

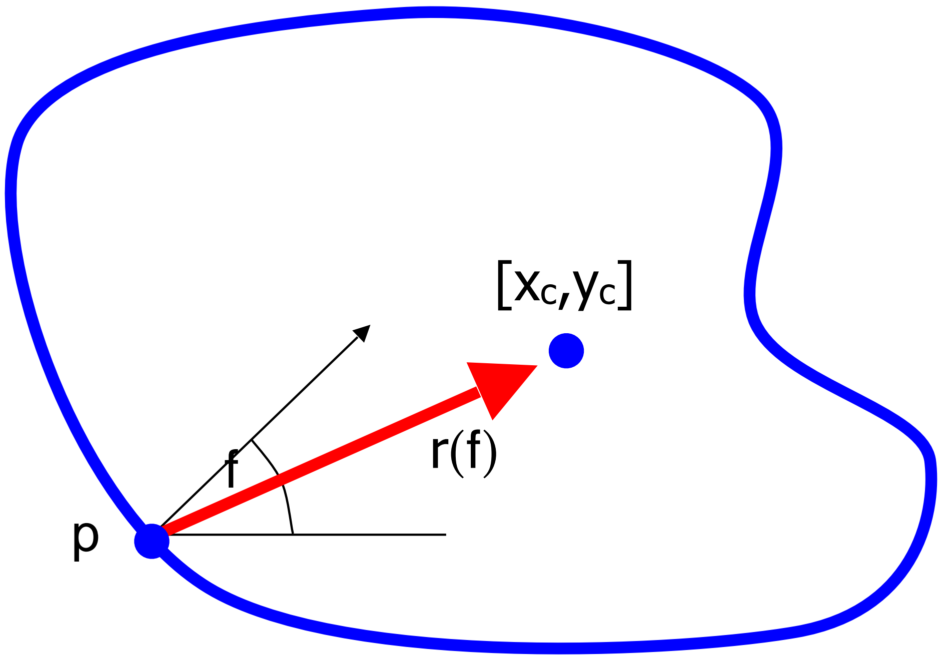


Finding a becher ...

... or a lake.



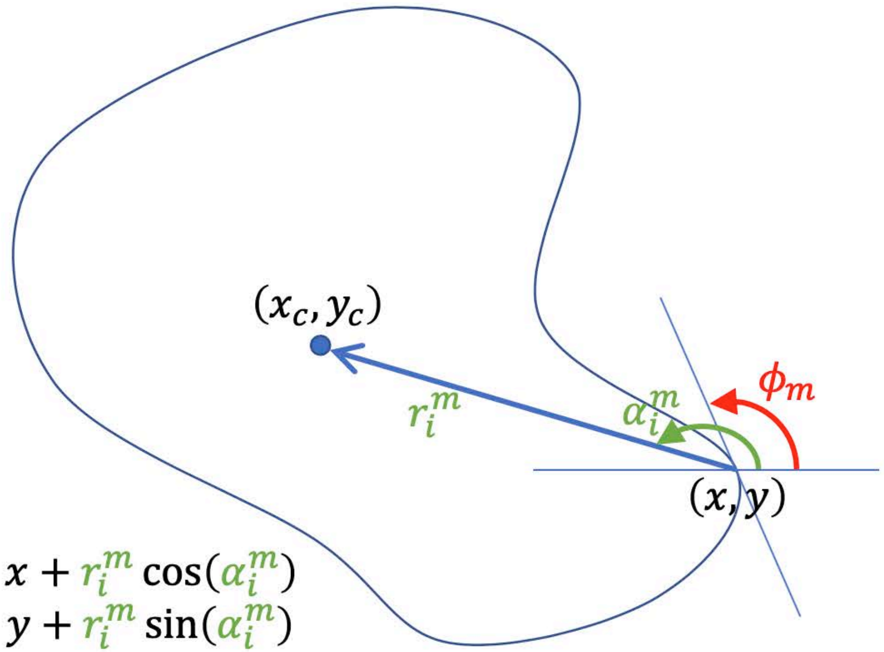
Generalized Hough



- We want to find a shape defined by its boundary points in terms of the location of a reference point $[x_c,y_c]$.
- For every boundary point p, we can compute the displacement vector $r = [x_c,y_c] - p$ as a function of local gradient orientation f.

R-Table

ϕ	$R(\phi_i)$
ϕ_1	$(r_1^1, \alpha_1^1), (r_2^1, \alpha_2^1)$,................................ $(r_{n1}^1, \alpha_{n1}^1)$
ϕ_2	$(r_1^2, \alpha_1^2), (r_2^2, \alpha_2^2)$, ...$(r_{n2}^2, \alpha_{n2}^2)$
..	....
ϕ_m	$(r_1^m, \alpha_1^m), (r_2^m, \alpha_2^m)$,.........$(r_i^m, \alpha_i^m)$........... $(r_{nm}^m, \alpha_{nm}^m)$
..	....
ϕ_M	$(r_1^M, \alpha_1^M), (r_2^M, \alpha_2^M)$, ... $(r_{nM}^M, \alpha_{nM}^M)$



$$x_c = x + r_i^m \cos(\alpha_i^m)$$
$$y_c = y + r_i^m \sin(\alpha_i^m)$$

Set of potential displacement vectors r,a given the boundary orientation f.

--> Generalized template matching.

Algorithm

1. Make an R-table for the shape to be located.

2. Form an accumulator array of possible reference points initialized to zero.

3. For each edge point,

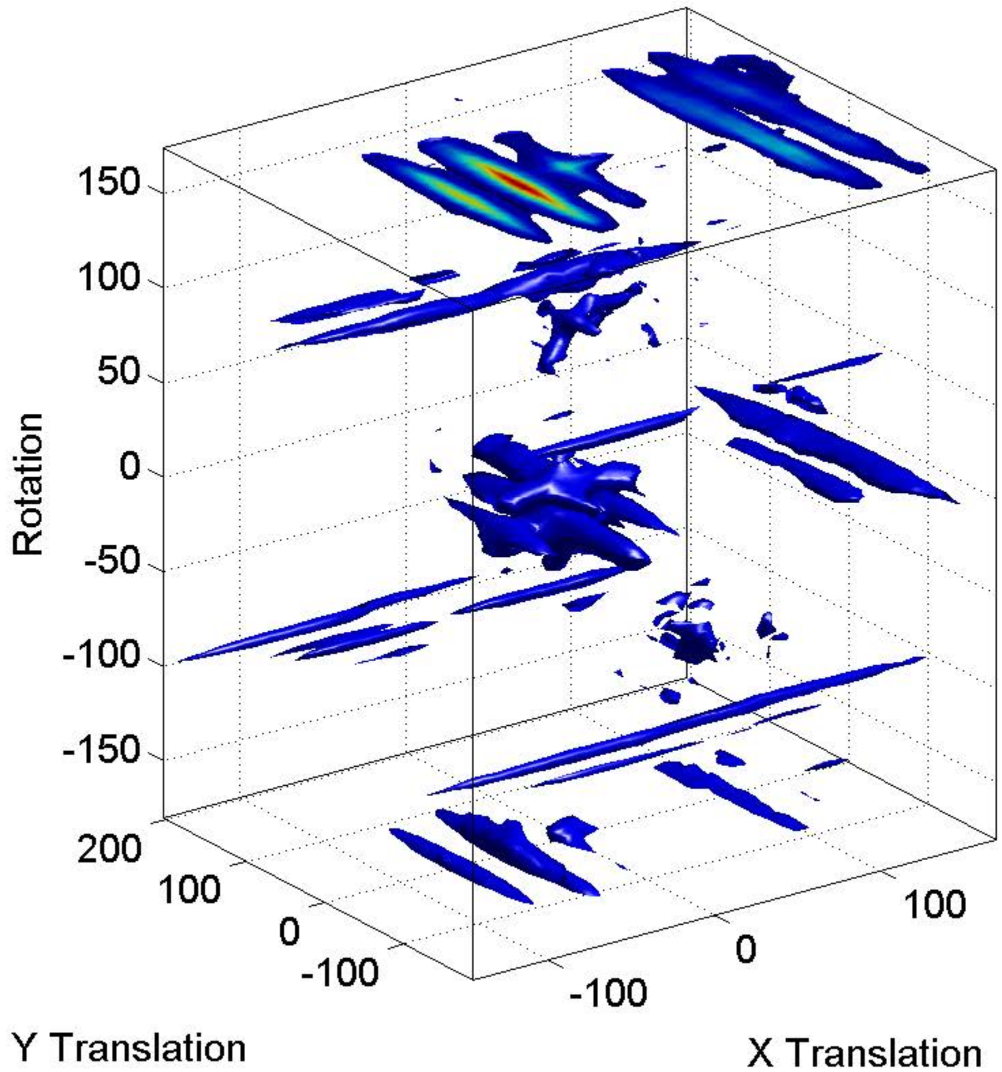
   - Compute the possible centers, that is, for each table entry, compute

$$
\begin{aligned}
x &= x_e + r_\phi \cos\left(\theta(\phi)\right) \\
y &= y_e + r_\phi \sin\left(\theta(\phi)\right)
\end{aligned}
$$

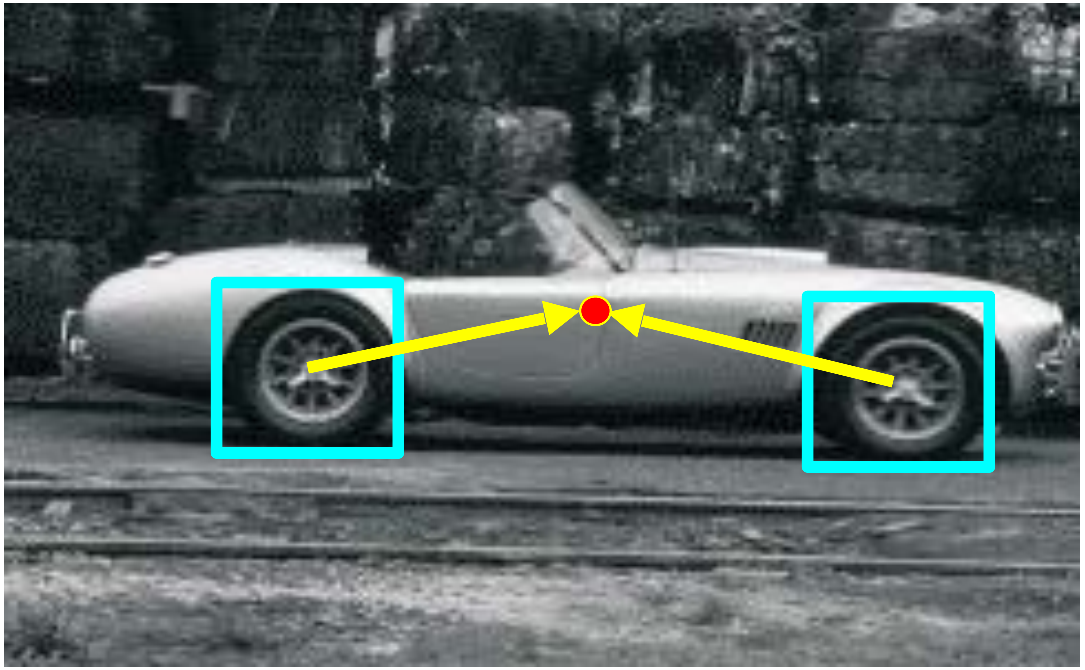
   - Increment the accumulator array

Real-Time Hough

Accumulator



From Delineation To Detection



Training image

Visual codeword with displacement vectors

Instead of indexing displacements by gradient orientation, index by "visual codeword".

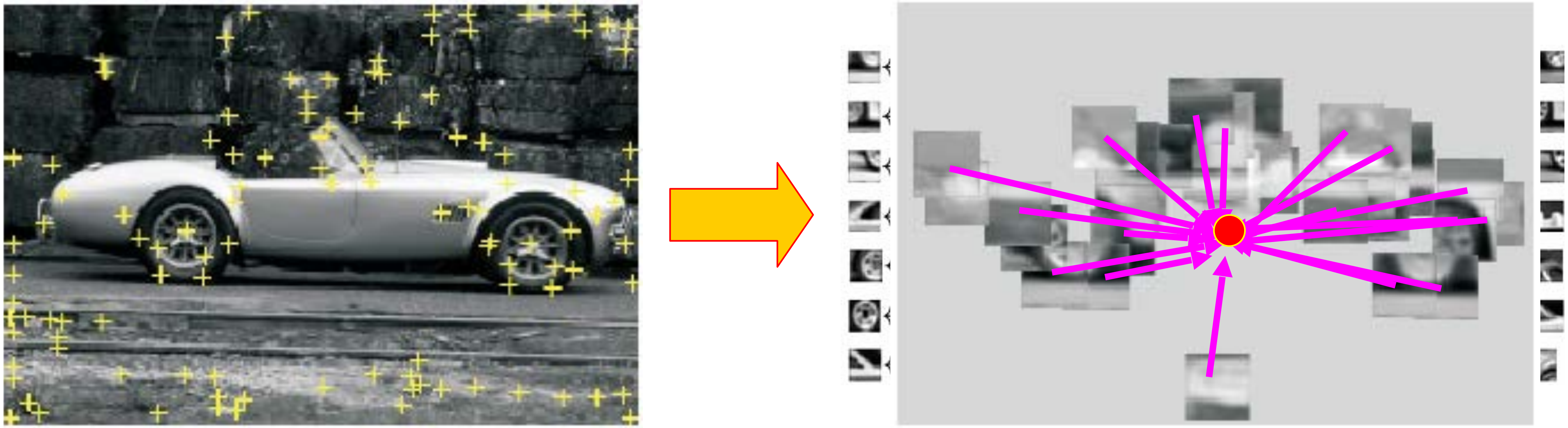
From Delineation To Detection

Test image

Instead of indexing displacements by gradient orientation, index by "visual codeword".

Leibe & al. , ECCV'04

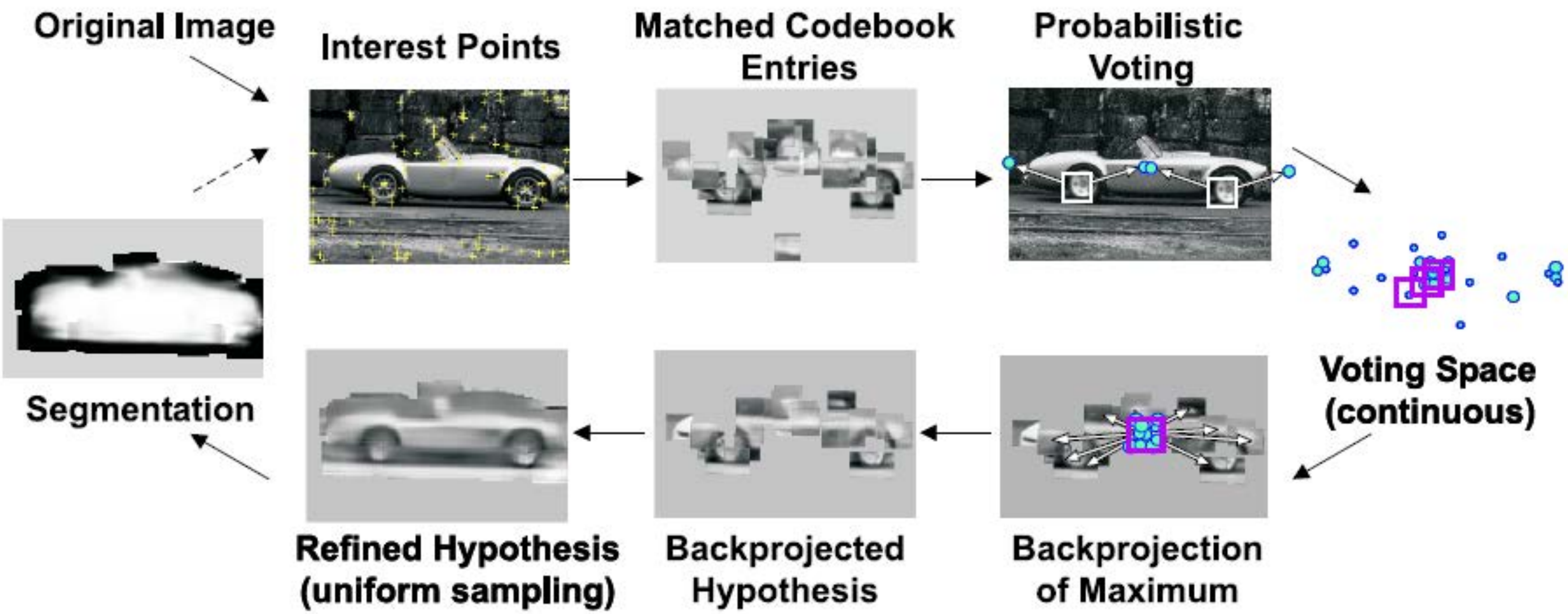
Optional: Training



1. Use clustering to build codebook of patches around extracted interest points.
2. Map the patch around each interest point to closest codebook entry.
3. For each codebook entry, store all positions it was found, relative to object center.

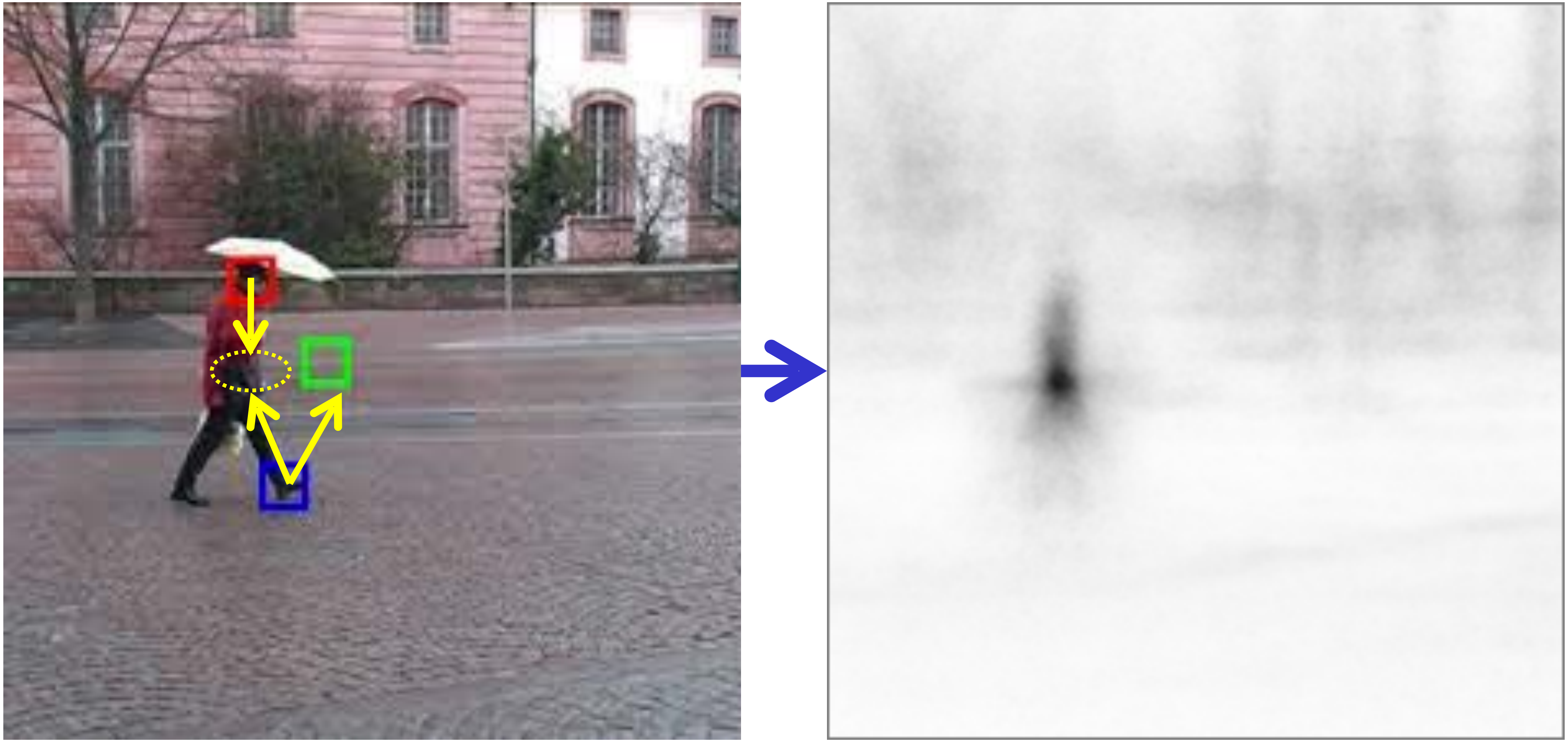
--> Build an R table.

Optional: Testing



1. Given test image, extract patches, match to codebook entry.
2. Cast votes for possible positions of object center.
3. Search for maxima in voting space.
4. Extract weighted segmentation mask based on stored masks for the codebook occurrences.

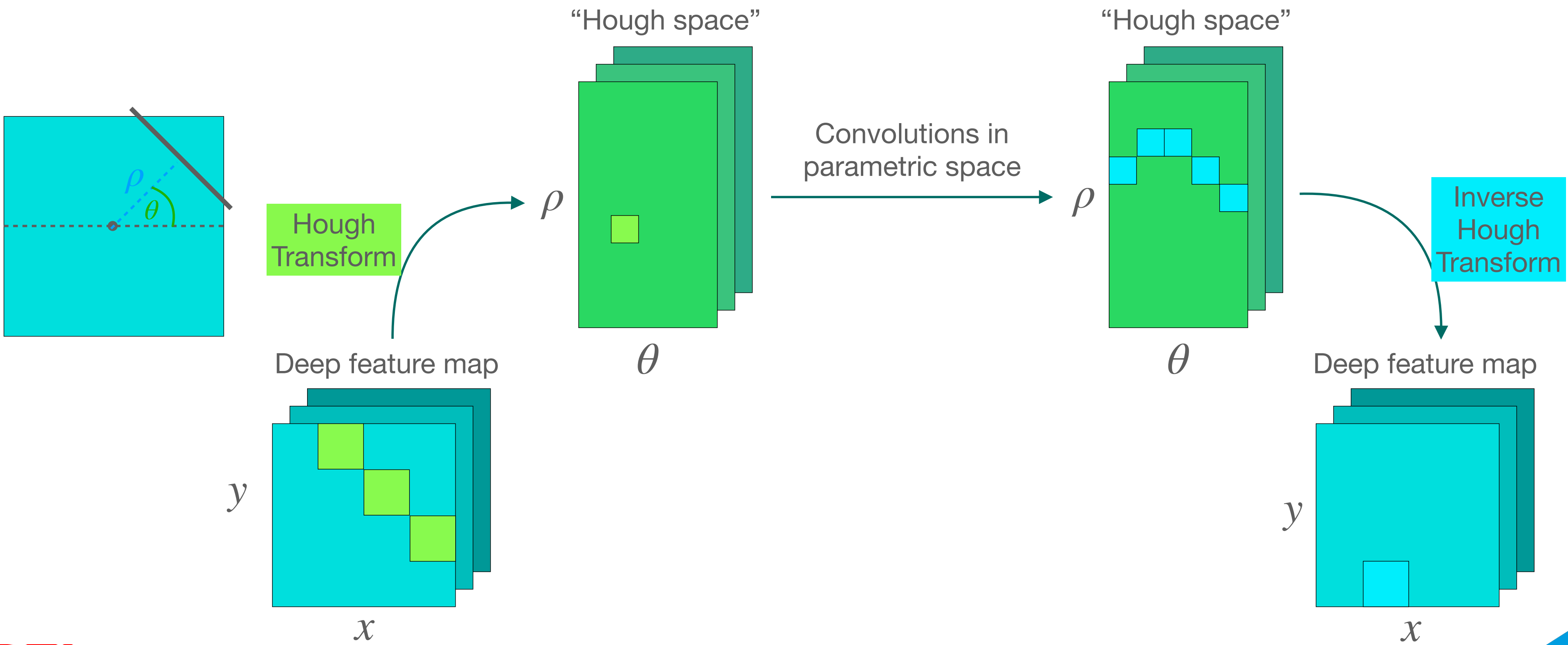
Pedestrian Detection



Gall & al. , PAMI'11

Occlusion Handling

Deep Hough Transform



Zhao, PAMI'21

Limitations

Computational cost grows exponentially with the number of model parameters:

→Only works for objects whose shape can be defined by a small number of parameters.

→Approach is robust but lacks flexibility.

Techniques

Semi-Automated Techniques:

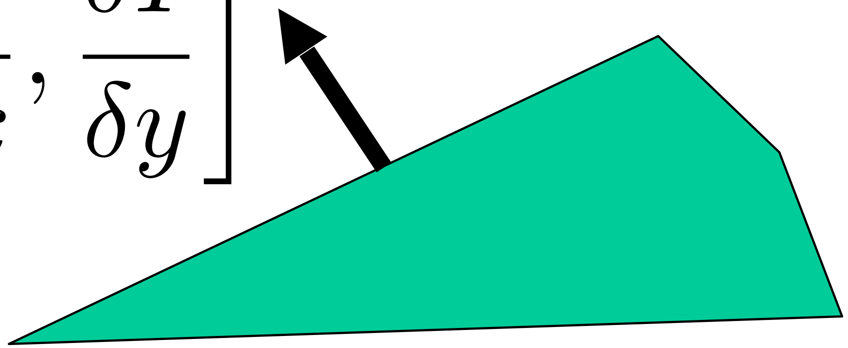
- Dynamic programming
- Deformable Models

Fully Automated Techniques:

- Hough transform
- Graph Based Approaches

Magnitude and Orientation

$$\left[\frac{\delta I}{\delta x}, \frac{\delta I}{\delta y}\right]$$

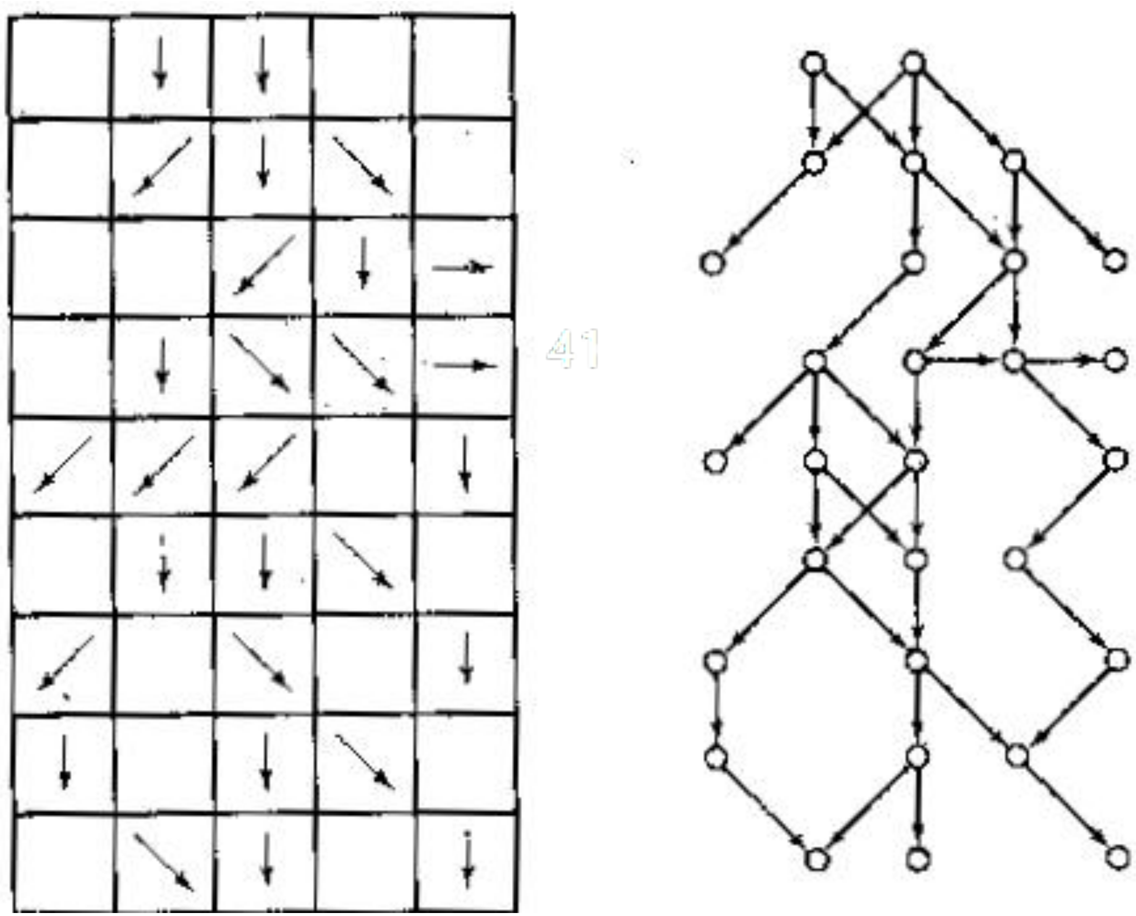


$$\text{Contrast: } G = \sqrt{\frac{\delta I}{\delta x}^2 + \frac{\delta I}{\delta y}^2}$$

$$\text{Orientation: } \Theta = \arctan\left(\frac{\delta I}{\delta y}, \frac{\delta I}{\delta x}\right)$$

Minimum Spanning Tree

Image modeled as a graph:



-> Generate minimal distance graph O(N log(N) algorithm)

Delineation 1998

Detect road centerlines

Find generic paths

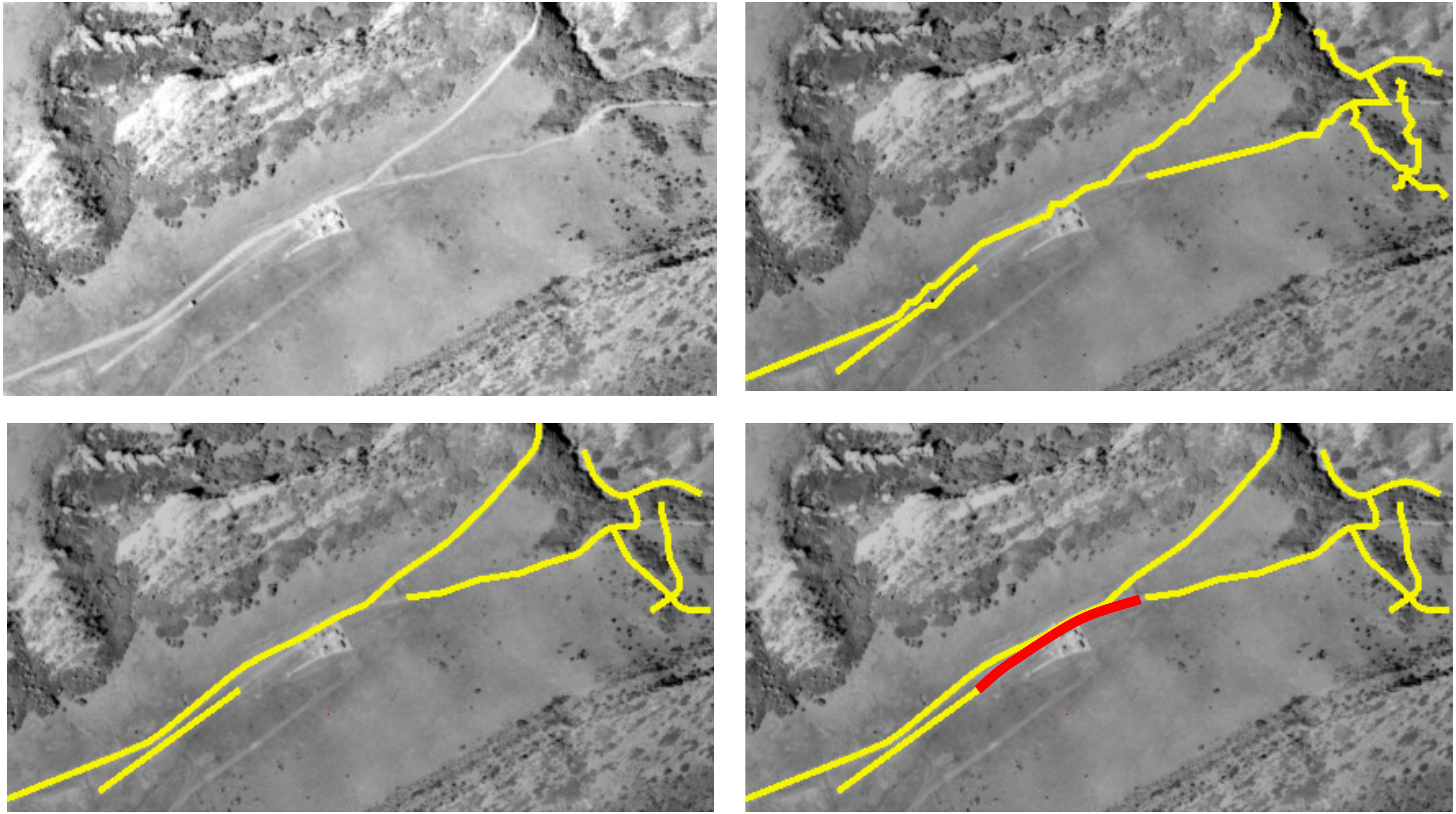
Apply semantic filter

Find road widths

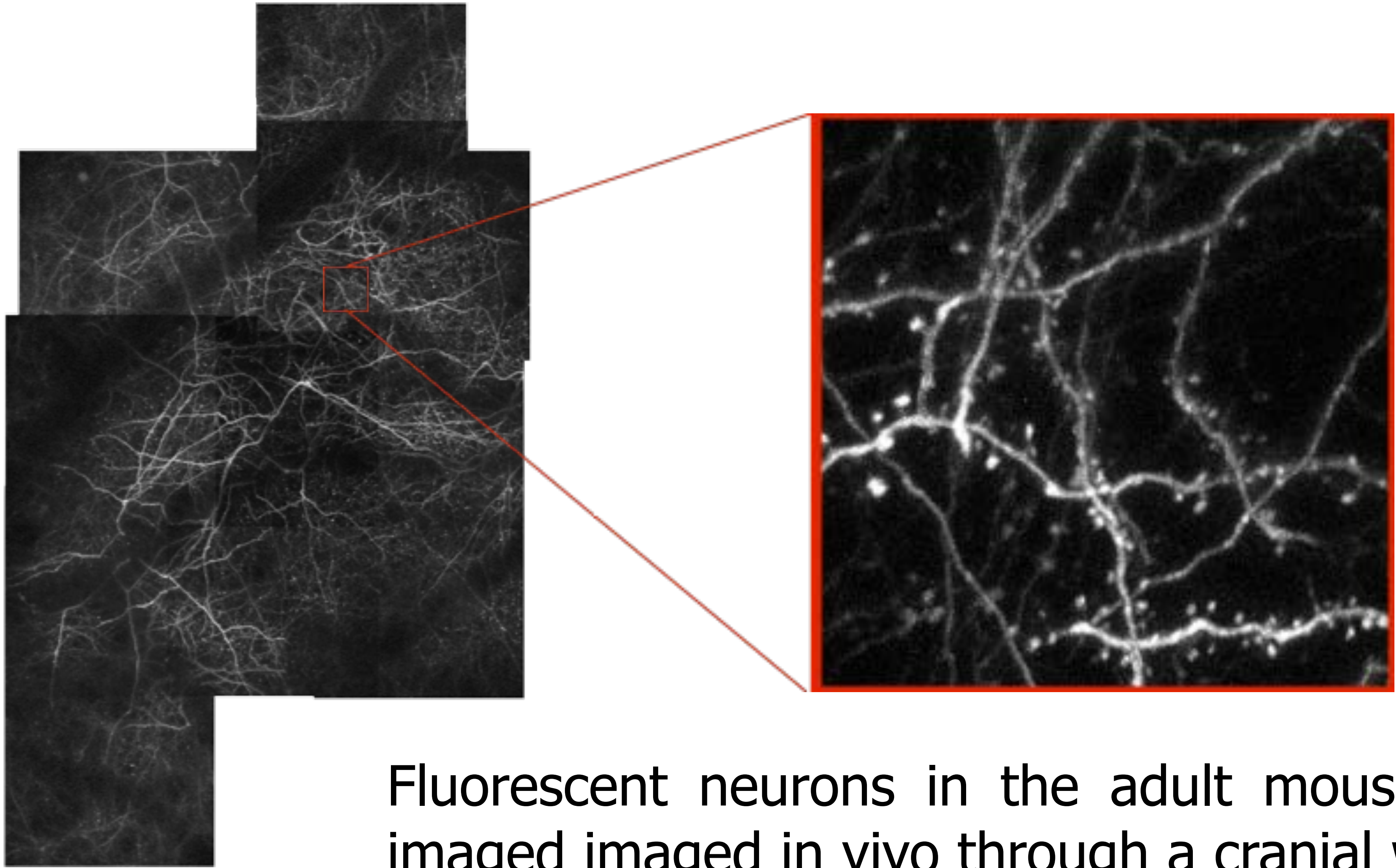
Fischler & Heller, 1998.

From Image To Roads

Road Editing

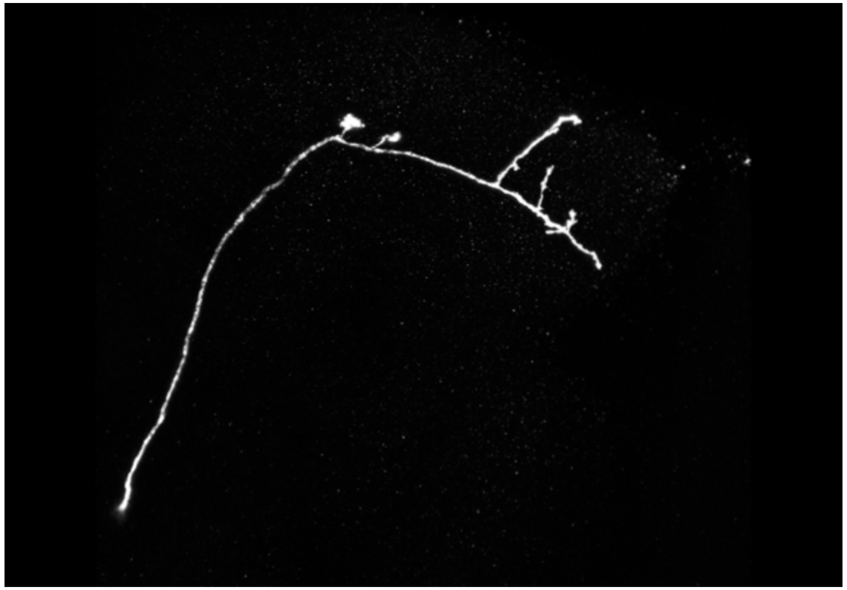


Dendrites And Axons

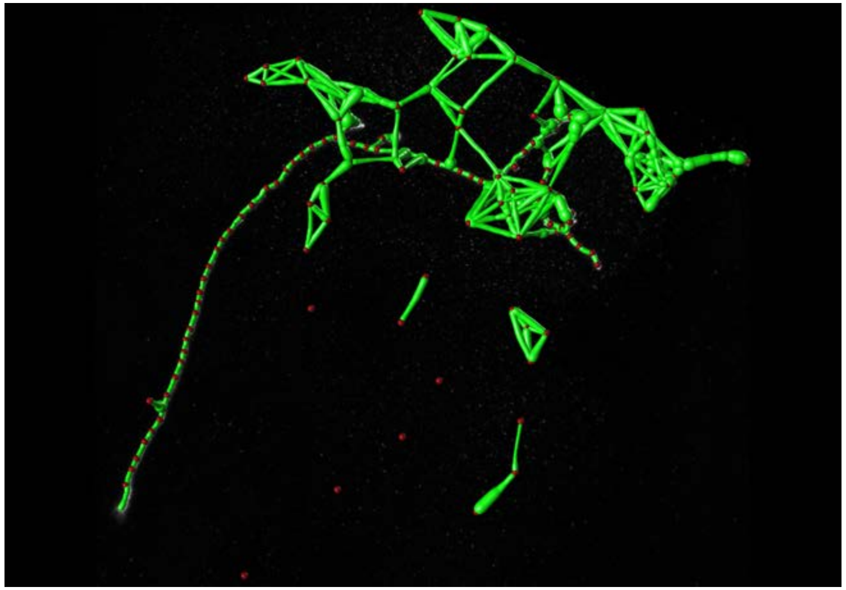


Fluorescent neurons in the adult mouse brain imaged imaged in vivo through a cranial window using a 2-photon microscope.

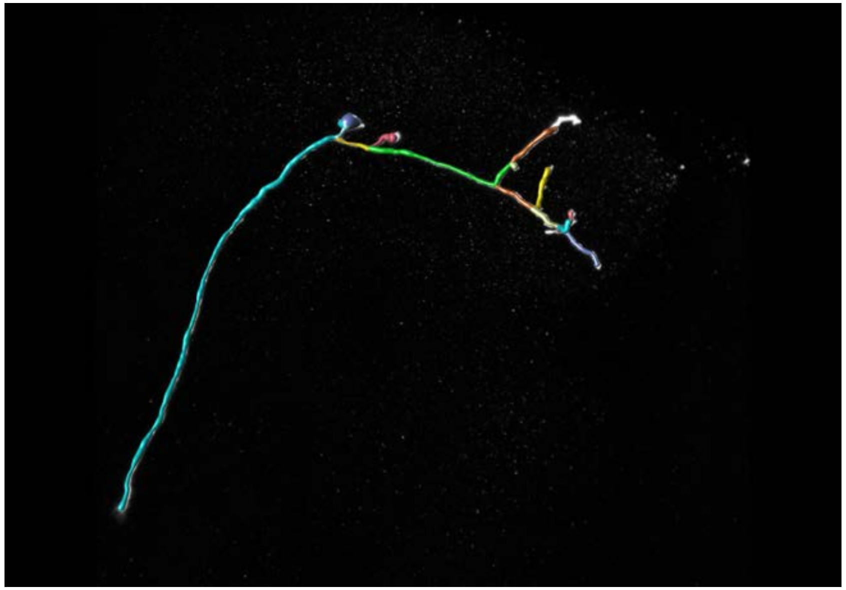
Delineation 2012: Neurites …



Filtered Image



Graph



Maximum Likelihood Subtree

.. and Roads

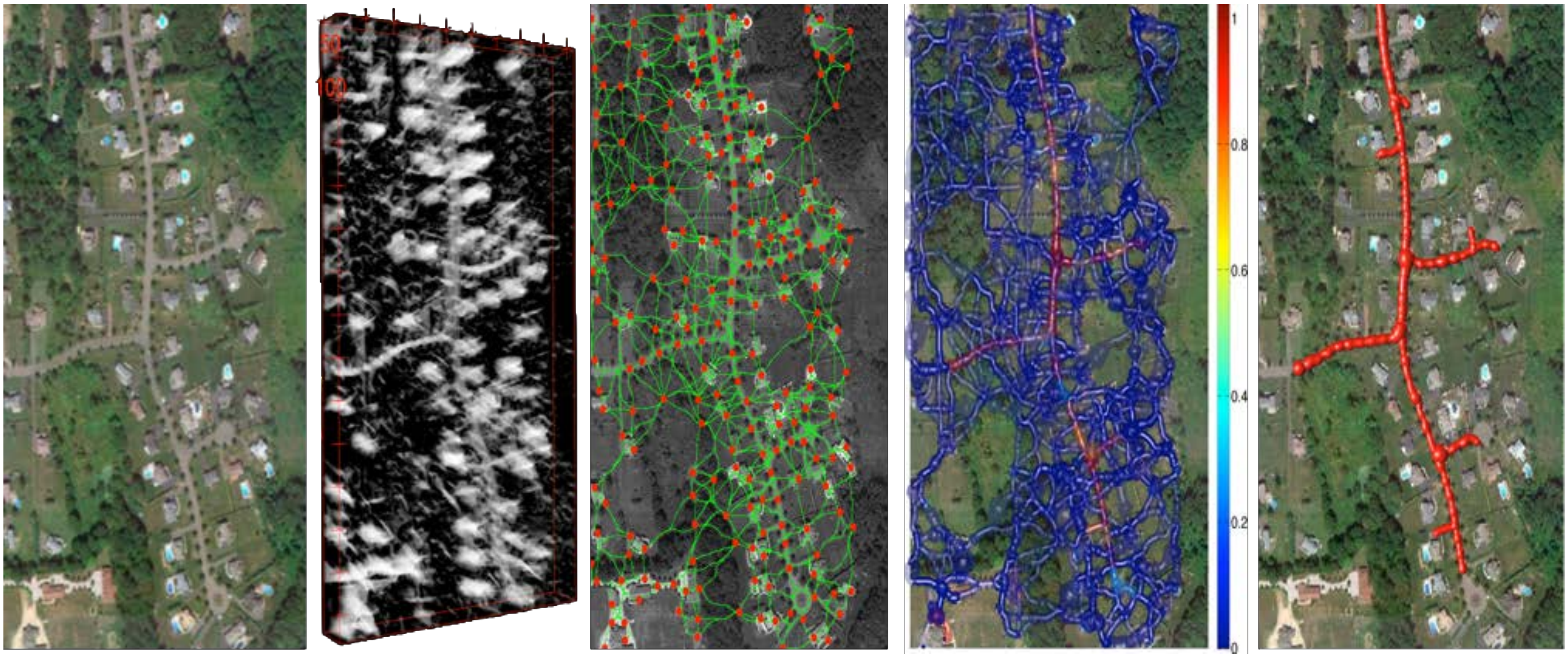
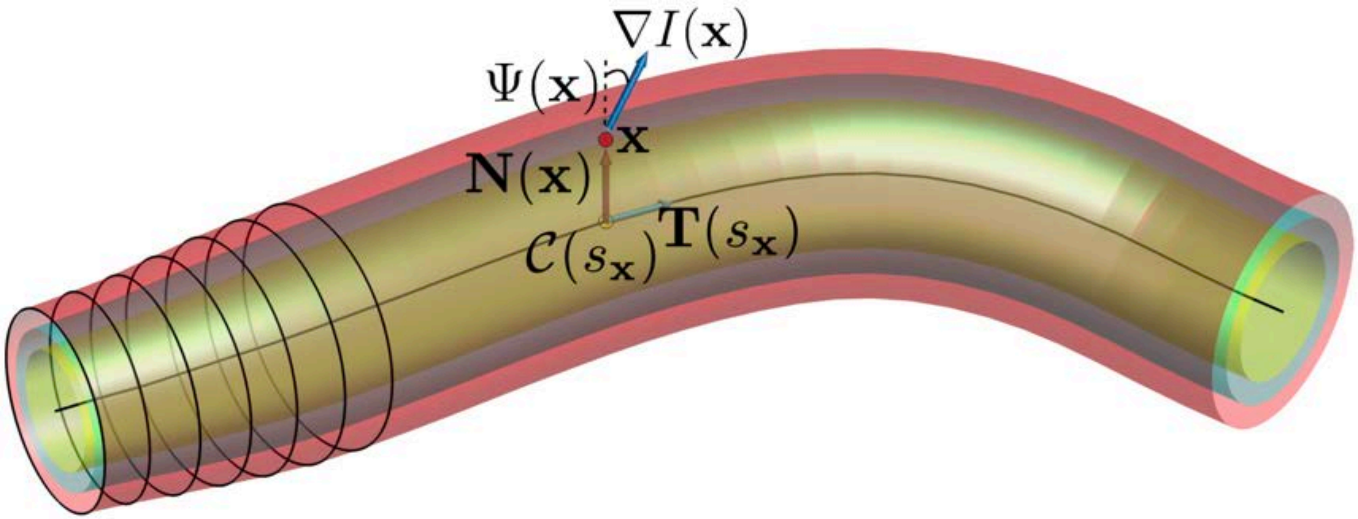


Image | Filtered image | Graph | Weighted graph | Subtree

—> Machine plays a crucial role to ensure that the **same algorithm works in different situations.**

Turetken et al, CVPR'12.

Histogram of Gradient Deviations

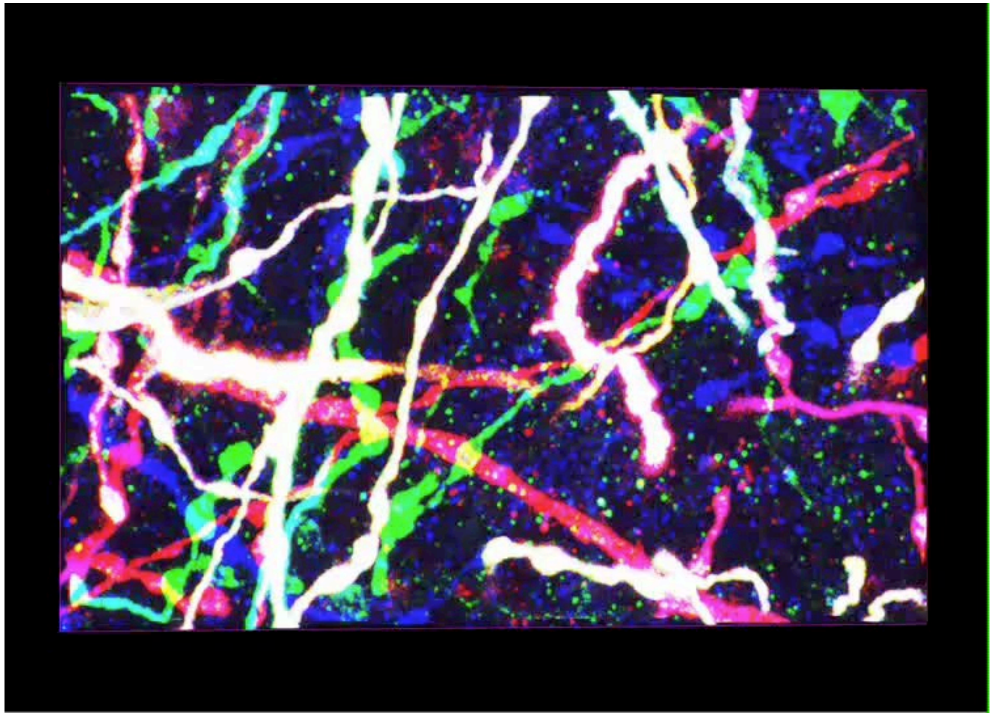


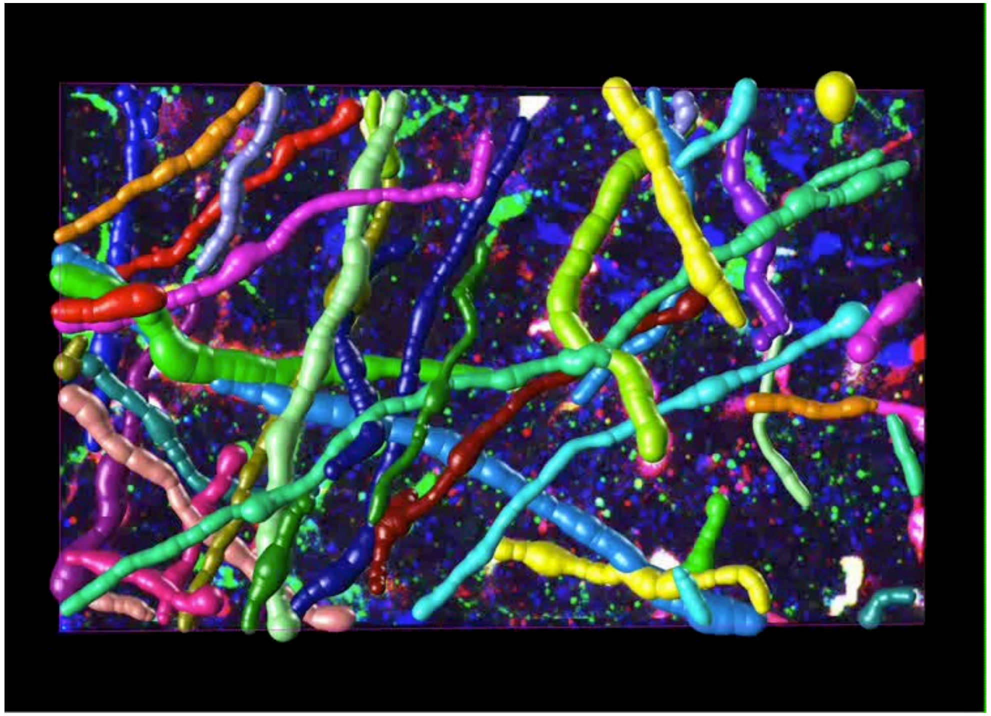
$$\Psi(\mathbf{x}) = \begin{cases} \text{angle}(\nabla I(\mathbf{x}), \mathbf{N}(\mathbf{x}))\,, \text{if } \|\mathbf{x} - \mathcal{C}(s_{\mathbf{x}})\| > \varepsilon \\ \text{angle}(\nabla I(\mathbf{x}), \mathbf{\Pi}(\mathbf{x}))\,, \text{otherwise}, \end{cases}$$

- One histogram per radius interval plus four geometric features (curvature, tortuosity, ...).
- Classifier trained on these histograms.

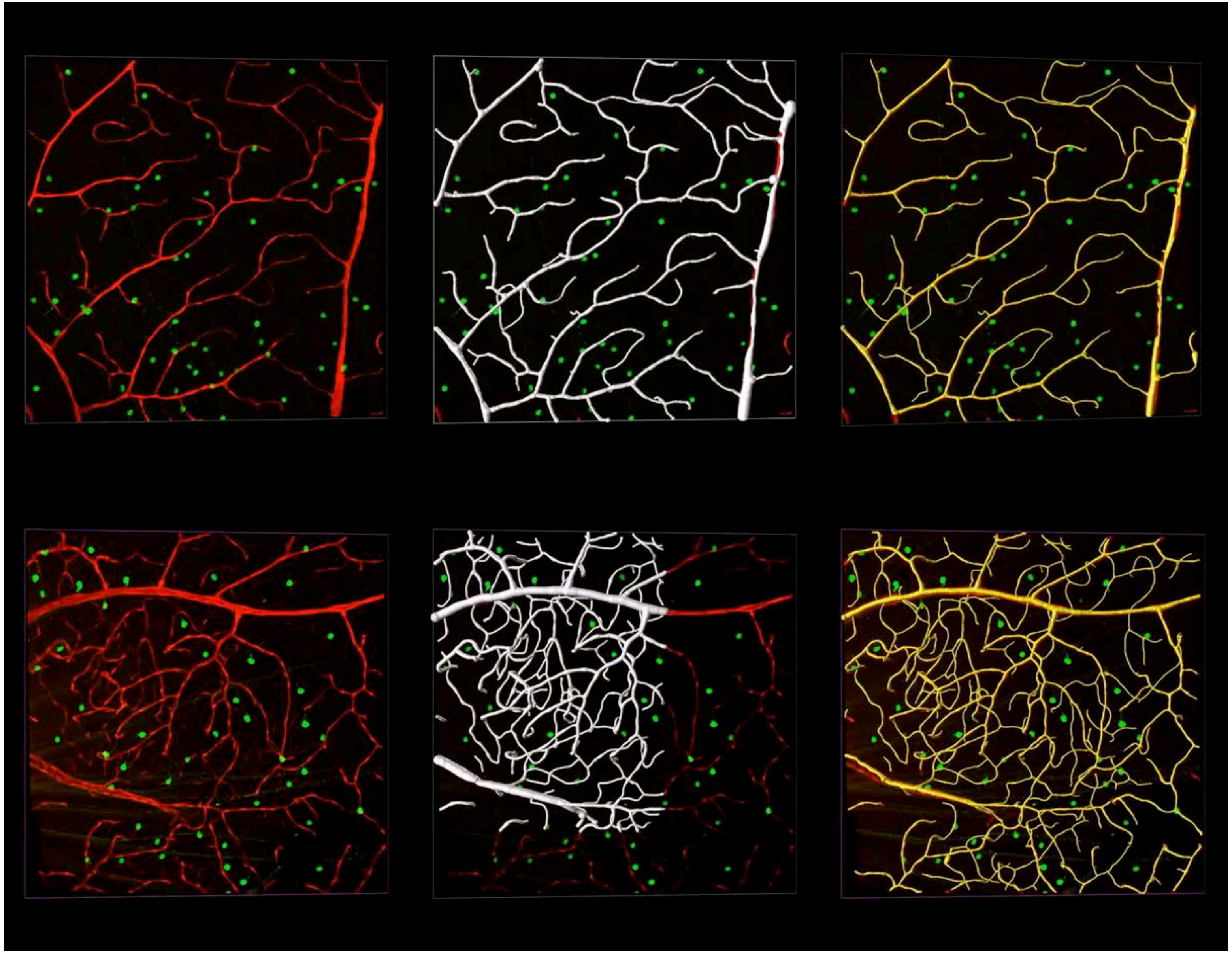
Roads

Brainbow Images





Blood Vessels



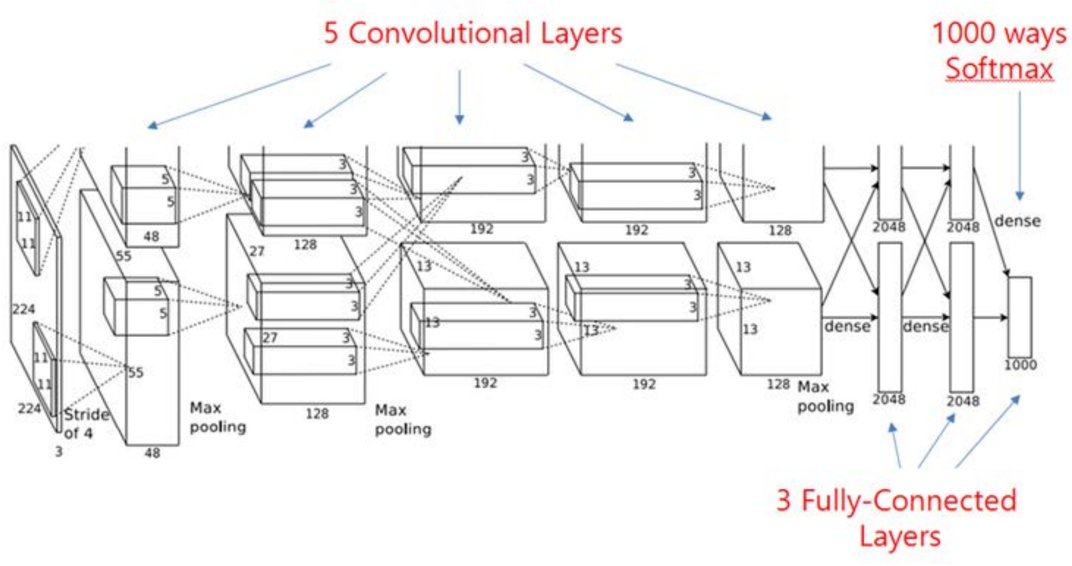
Deep Tsunami

The New York Times

Turing Award Won by Three Pioneers in Artificial Intelligence

From left, Yann LeCun, Geoffrey Hinton and Yoshua Bengio. The researchers worked on key developments for neural networks, which are reshaping how computer systems are built.

Reminder: AlexNet (2012)



Task: Image classification

Training images: Large Scale Visual Recognition Challenge 2010

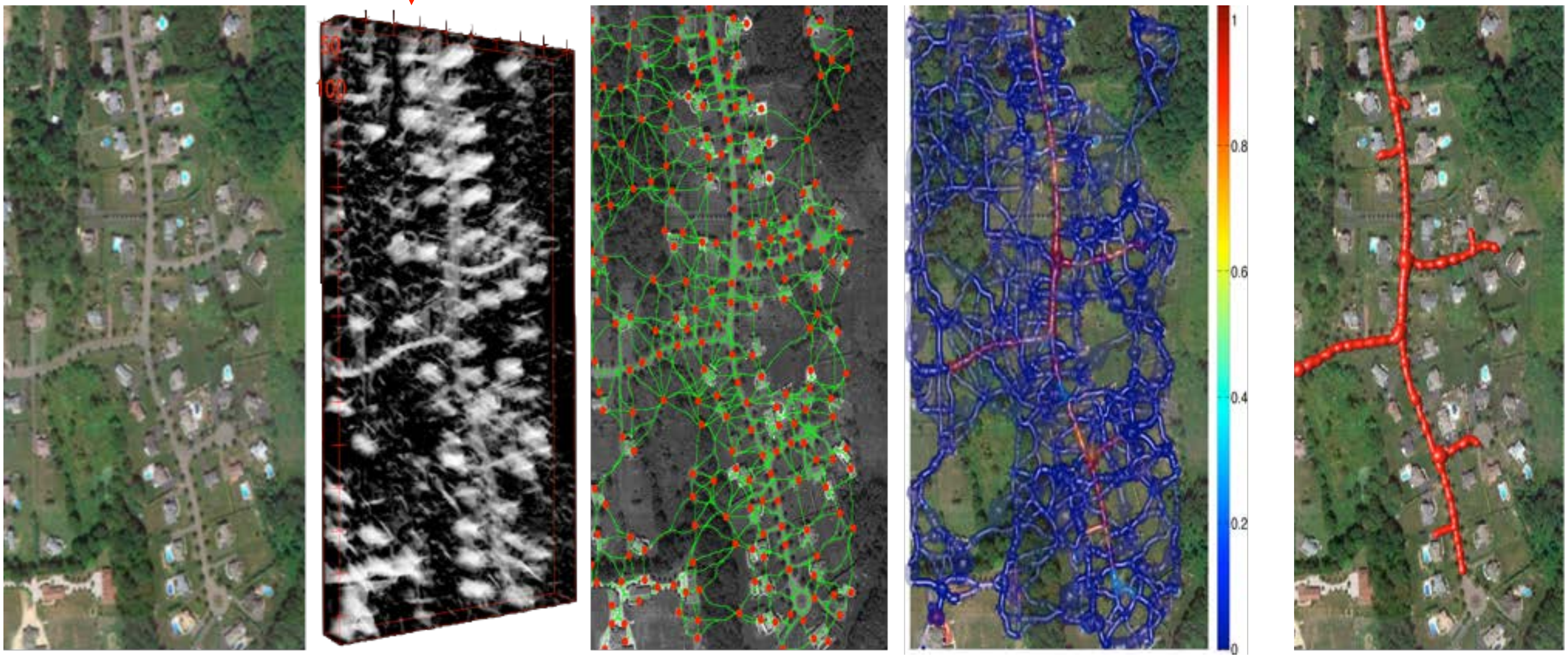
Training time: 2 weeks on 2 GPUs

Major Breakthrough: Training large networks has now been shown to be practical!!

EPFL

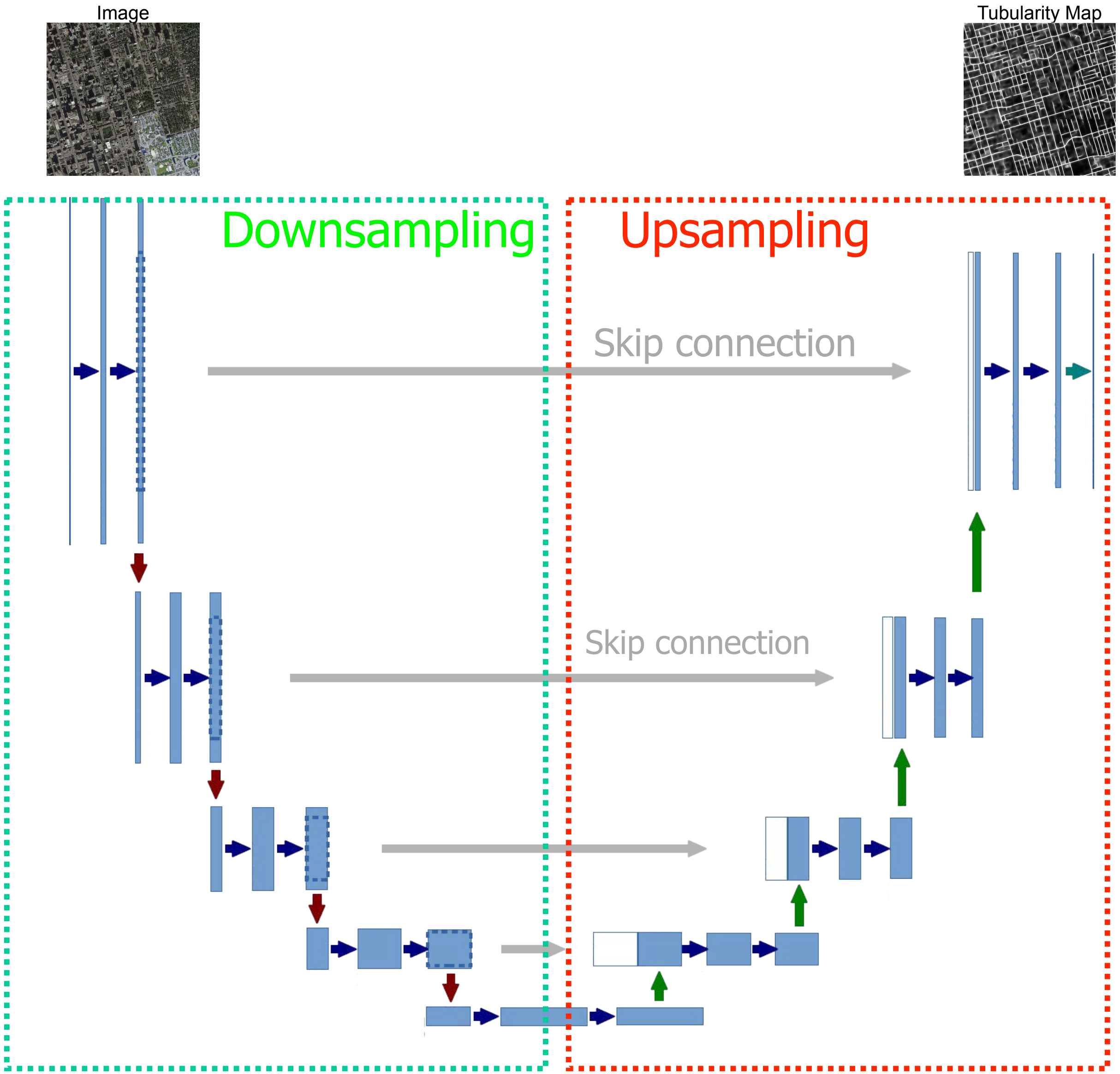
Delineation 2012

Can deep learning improve this?



Turetken et al, CVPR'12.

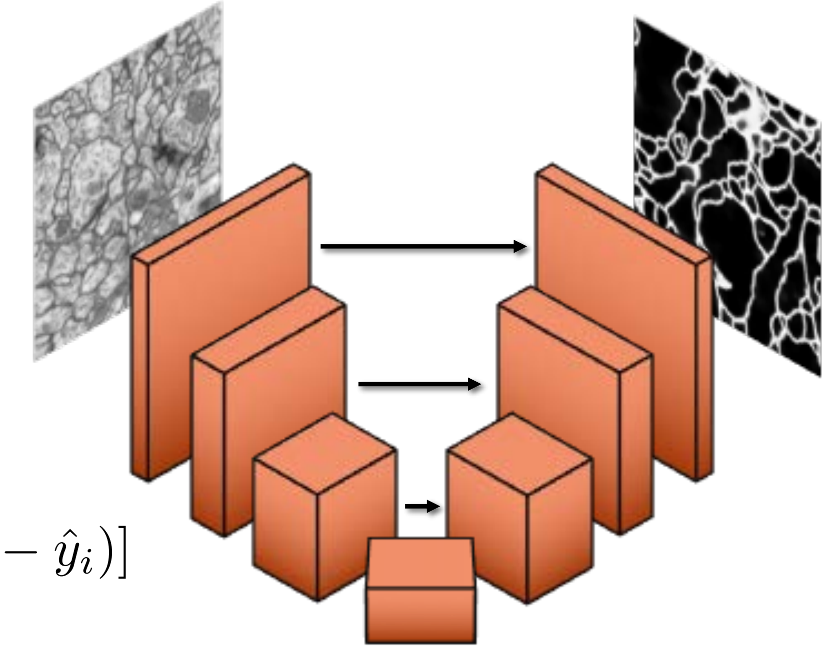
Reminder: U-Net



—> Train a U-Net to output a tubularity map.

Training a U-Net

Train Encoder-decoder U-Net architecture using binary cross-entropy



Minimize

$$L_{bce}(\mathbf{x}, \mathbf{y}; \mathbf{w}) = -\frac{1}{i}\sum_{1}^{P}[y_{\,\mathrm{i}} \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

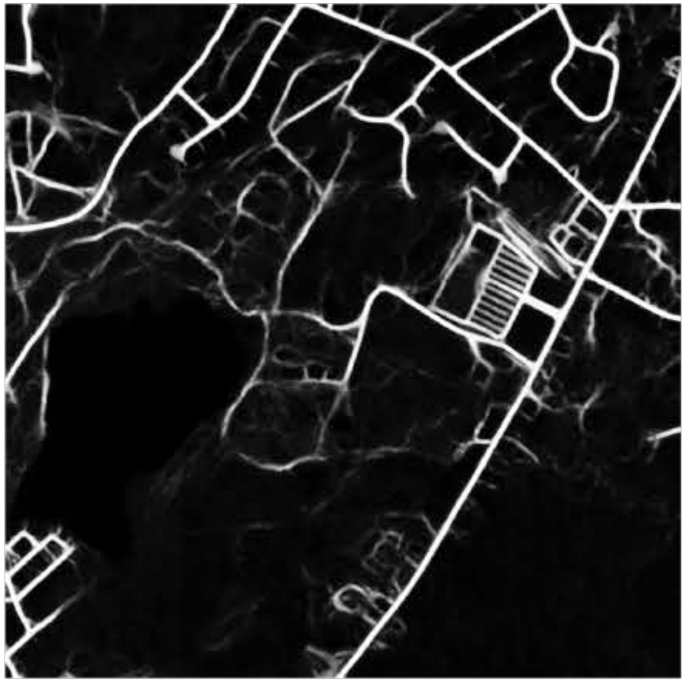
where

- $\hat{\mathbf{y}} = f_{\mathbf{w}}(\mathbf{x})$,
- $\mathbf{x}$ in an input image,
- $\mathbf{y}$ the corresponding ground truth.

Mosinska et al, CVPR'18.

Network Output

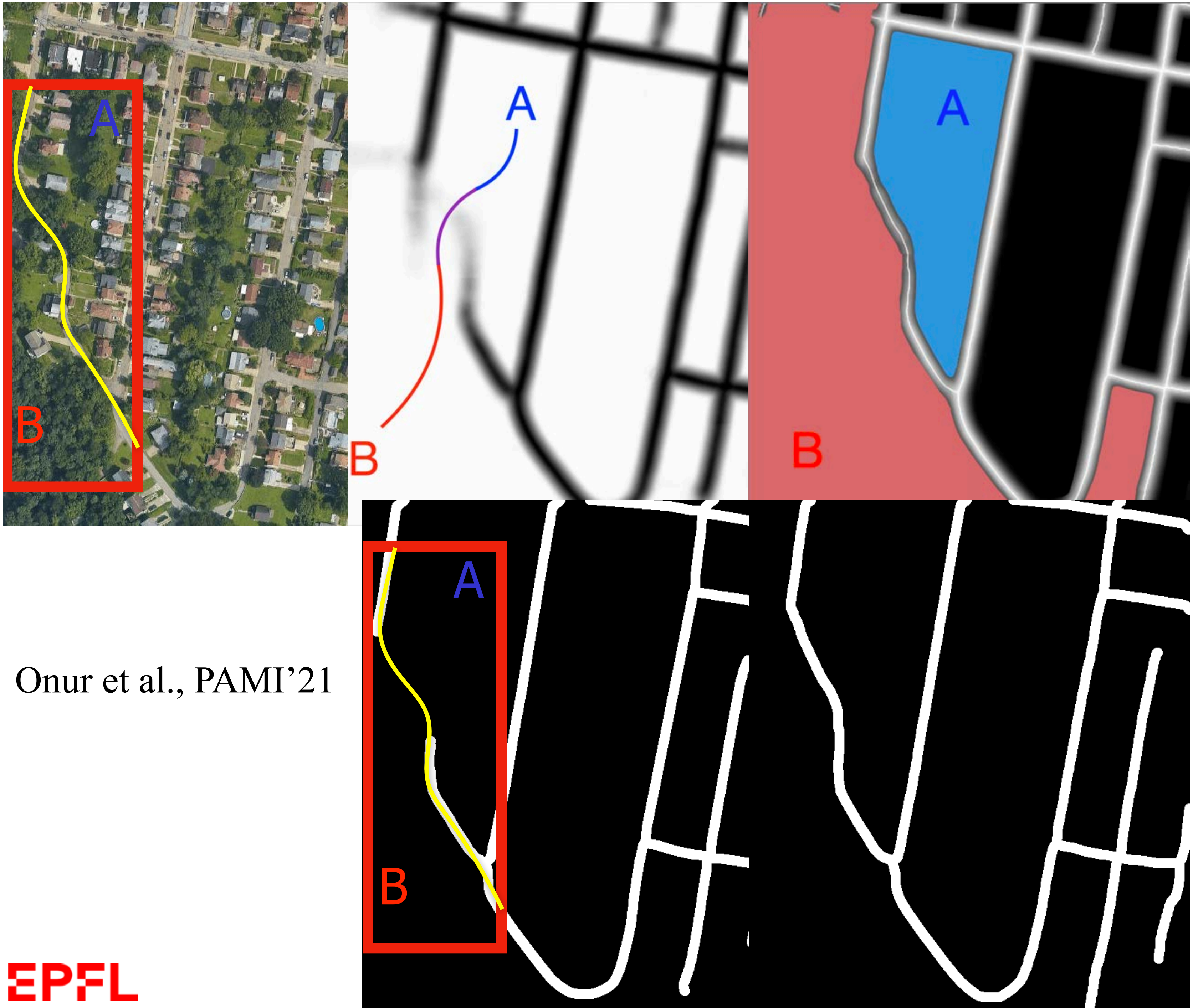
Image



BCE Loss

Ground truth

Accounting for Topology



- The yellow road is partially hidden by trees.
- A standard U-Net misses the hidden portion.
- We add to the loss function used to train the network a term that encourages points such as A and B to be separated.
- The re-trained U-Net now finds the complete road.

Onur et al., PAMI'21

Iterative Refinement

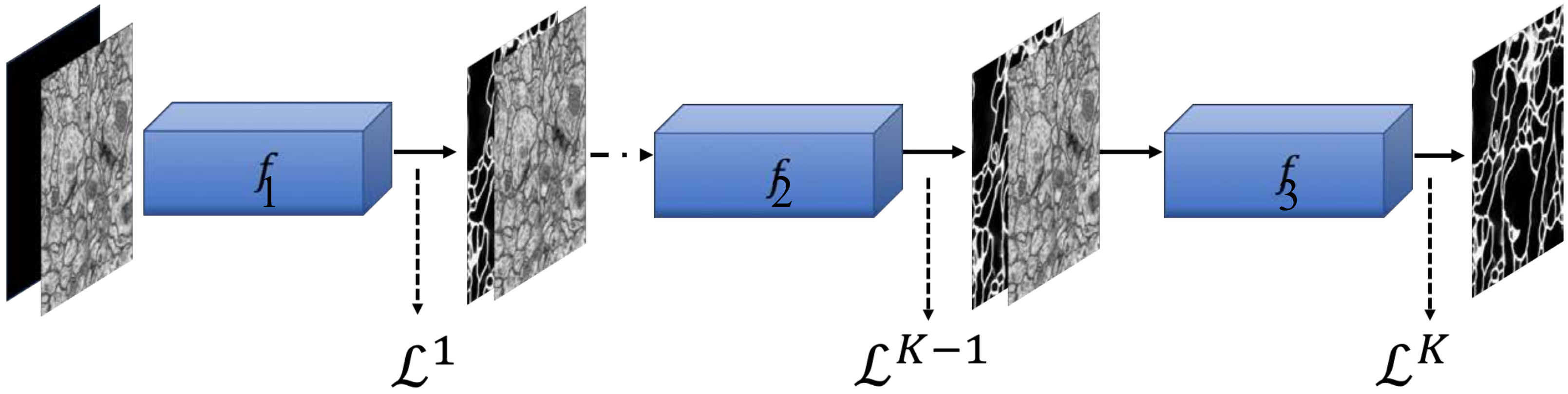
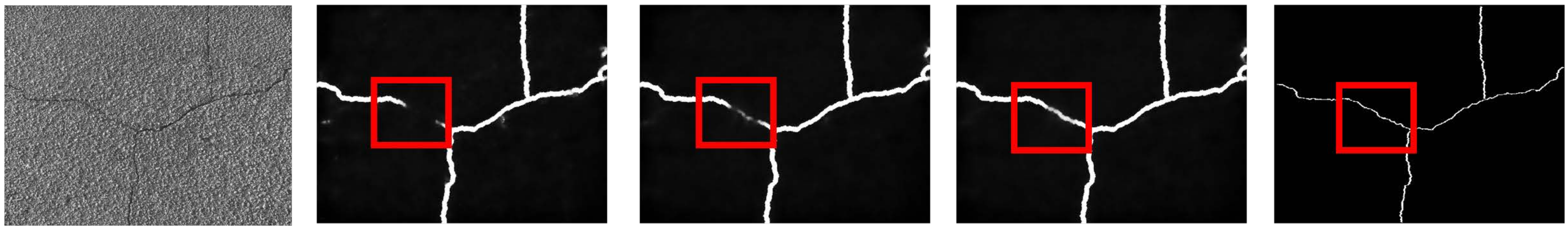
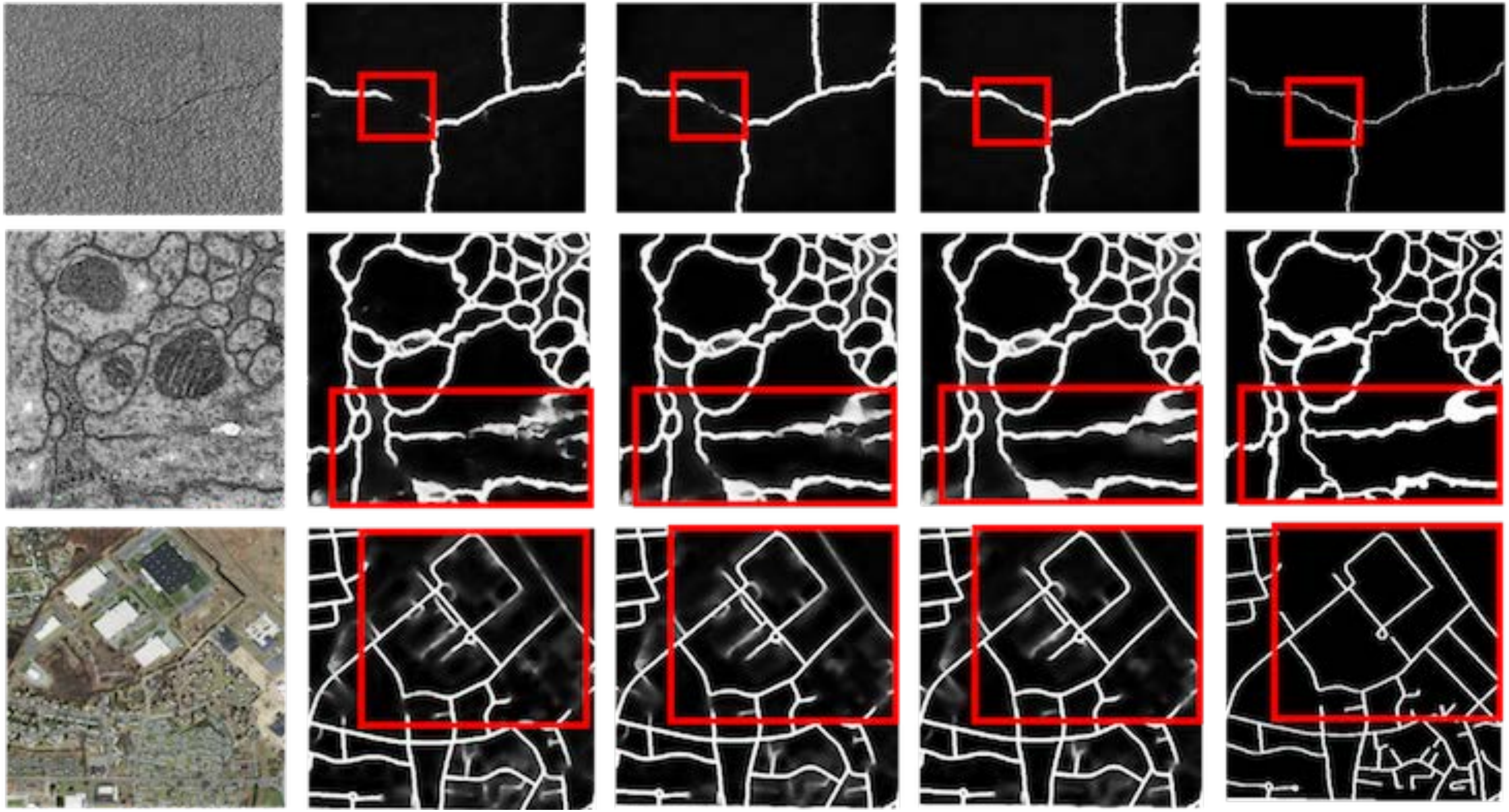


Image | Iter 1 | Iter 2 | Iter 3 | Ground truth



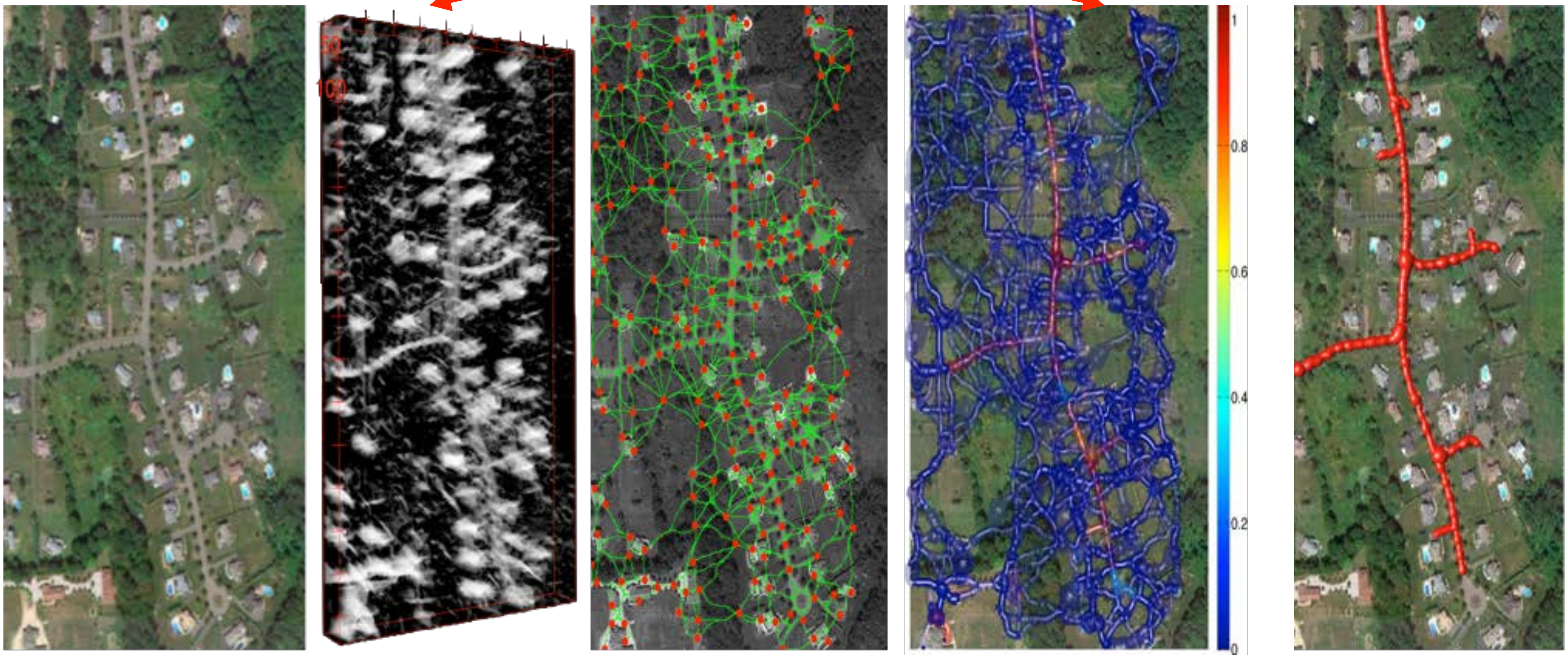
Use <u>the same network</u> to progressively refine the results keeping the number of parameters constant

Iterative Refinement



Delineation 2019

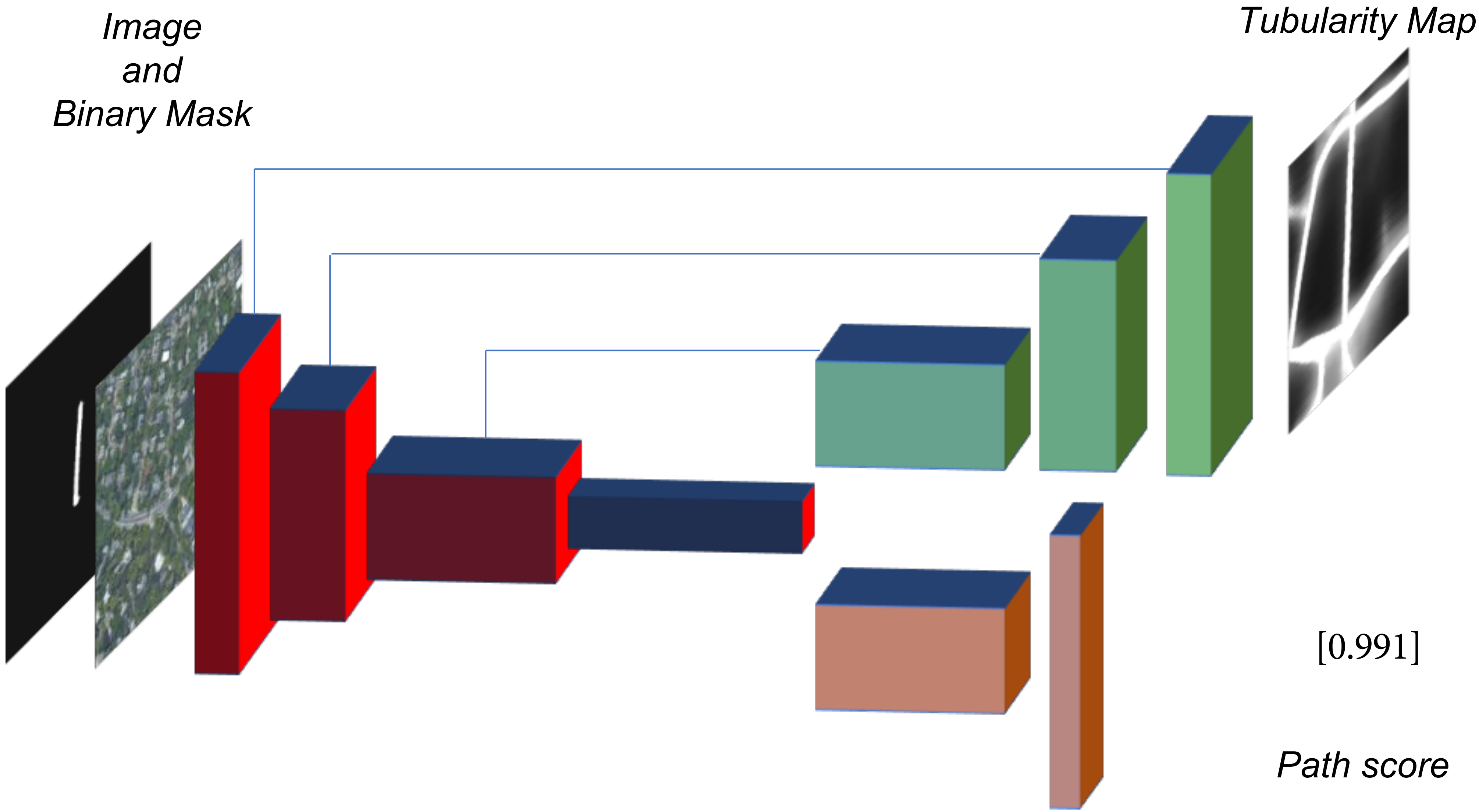
These two steps are closely related!



They should be performed by the same network!

Mosinska et al, PAMI'19.

Dual Use U-Net

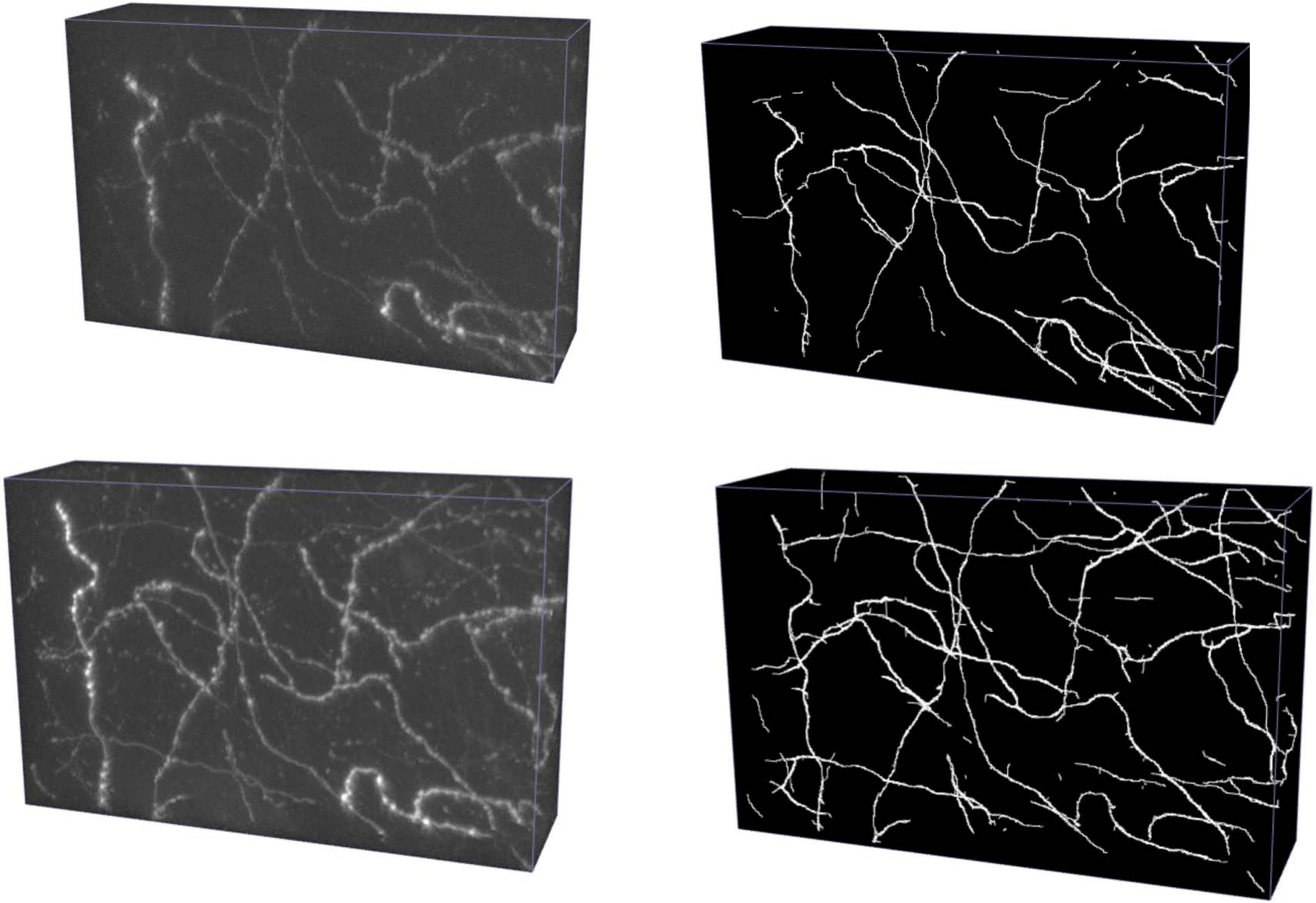


Delineation Steps

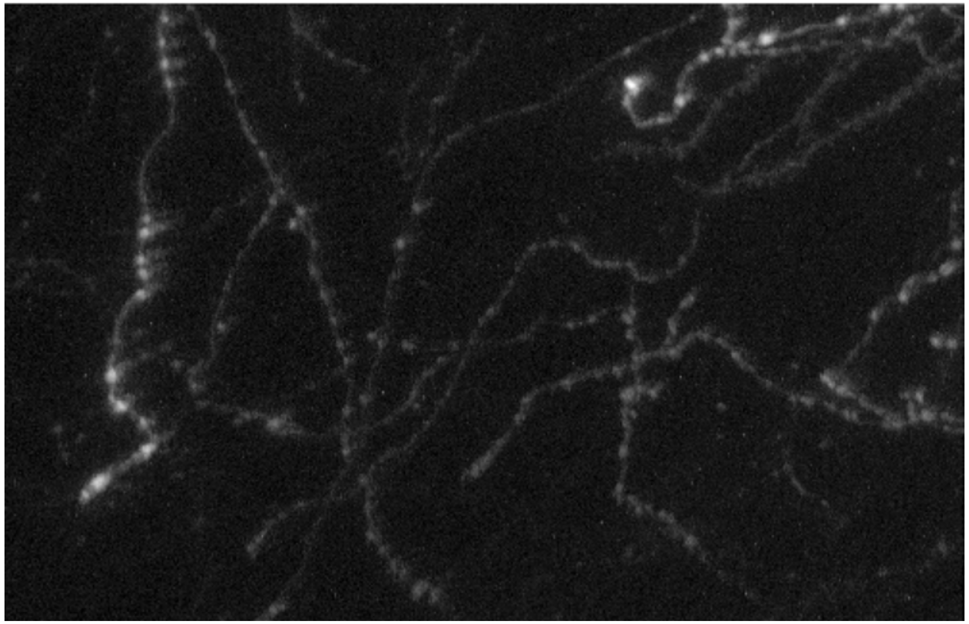
1. Compute a probability map.
2. Sample and connect the samples.
3. Assign a weight to the paths.
4. Retain the best paths.

Streets Of Toronto

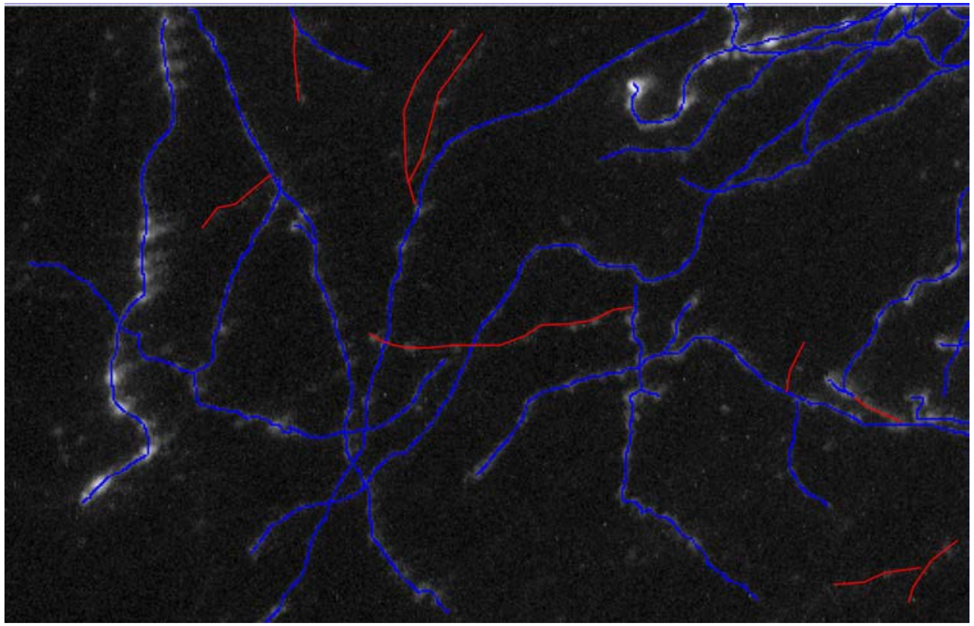
Dendrites And Axons



Typical Annotations



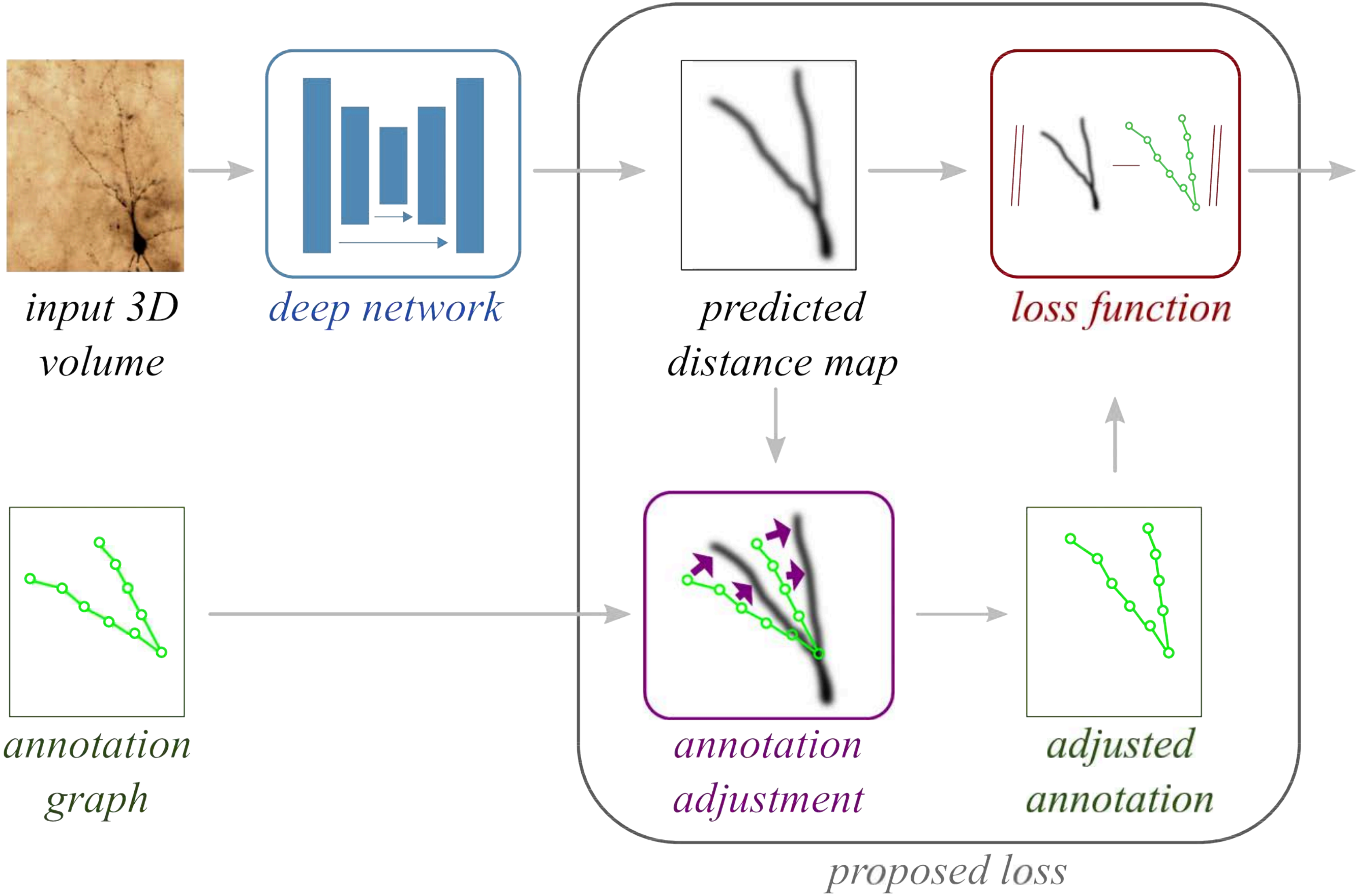
Original Image



"Ground truth" + Mistakes

—> Human annotations are often imprecise.

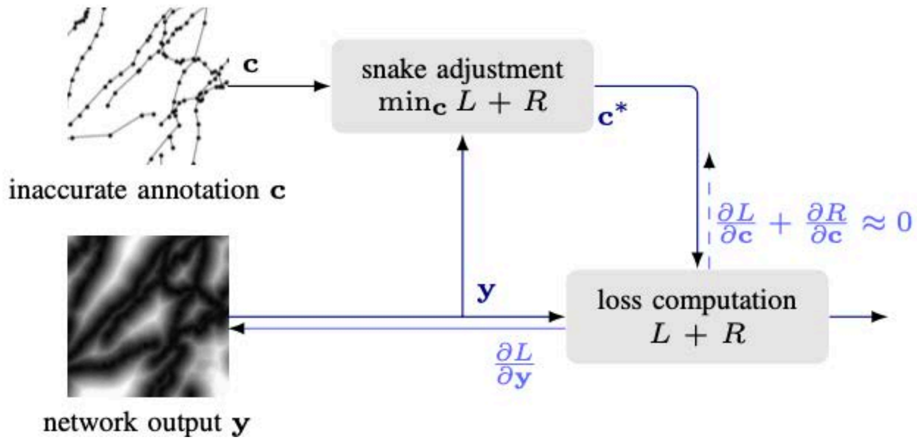
Correcting the Annotations



To account for annotation inaccuracies during training, we jointly train the network and adjust the annotations while preserving their topology.

Oner et al. , TMI'22

Annotations as Network Snakes

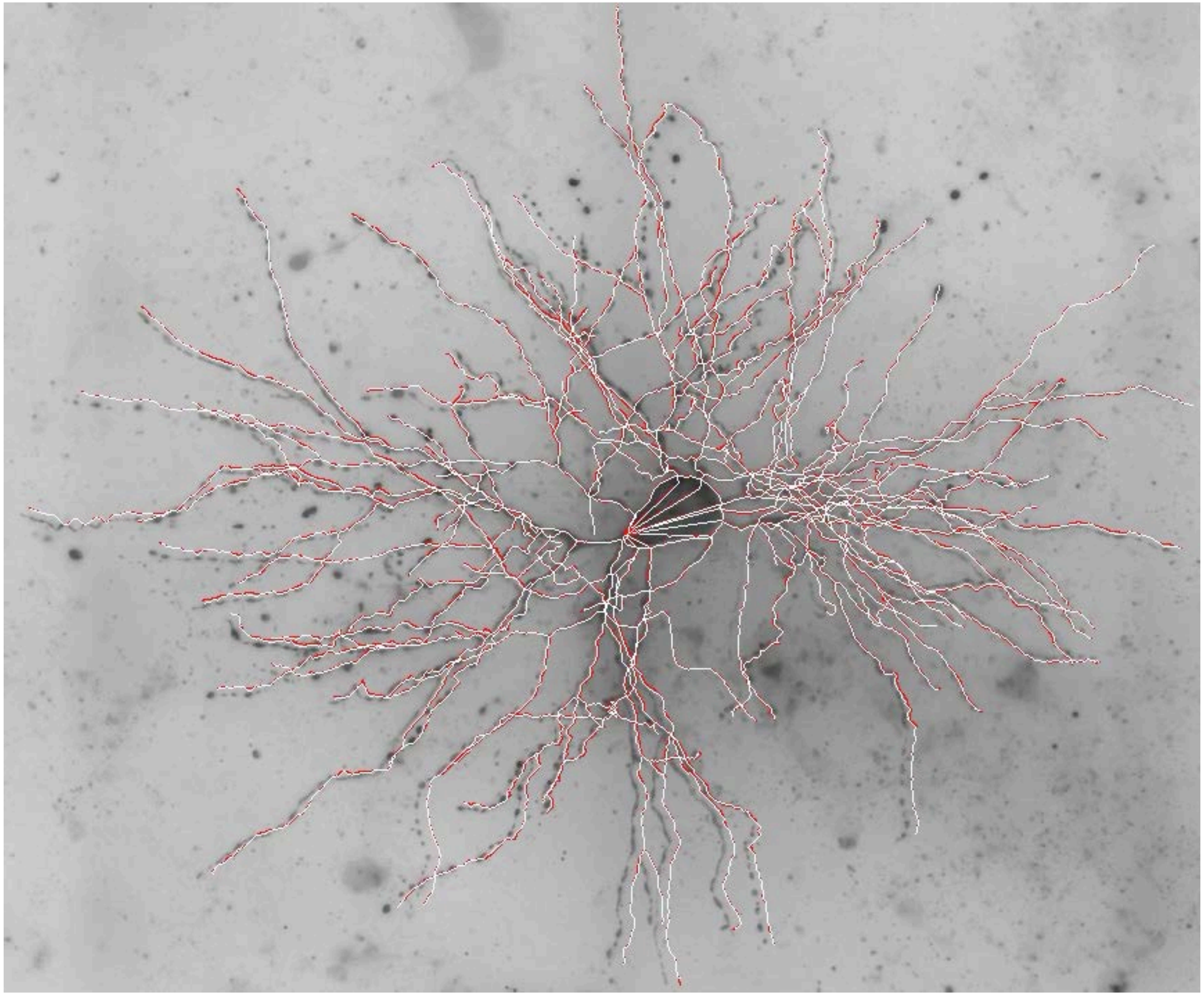


$$\Theta^*, \mathbf{C}^* = \text{argmin}_{\Theta,\mathbf{C}} \sum_{i=1}^{N} \underbrace{\mathscr{L}\left(D(\mathbf{c}_i), \mathbf{y}_i\right)}_{\text{Distance between network output and annotations.}} + R(\mathbf{c}_i)$$

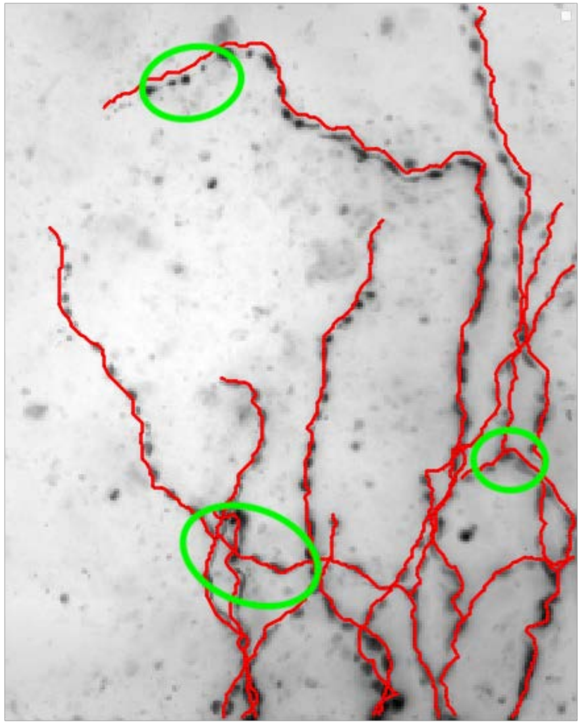
where

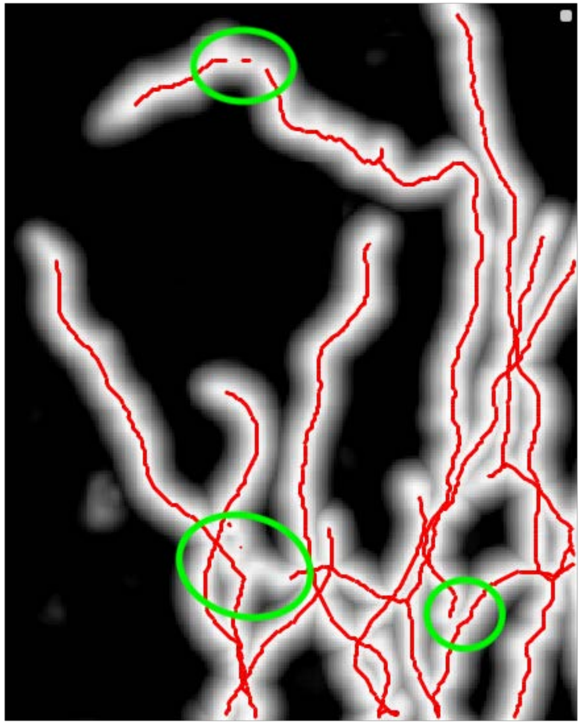
- The y_i are the network outputs;
- The $\mathbf{c}_i$ are the annotation vertices;
- $\mathbf{C}$ is the vector obtained by concatenating all the $\mathbf{c}_i$;
- D is a distance transform ;
- $\mathscr{L}$ is the MSE loss;
- R is a regularization term.

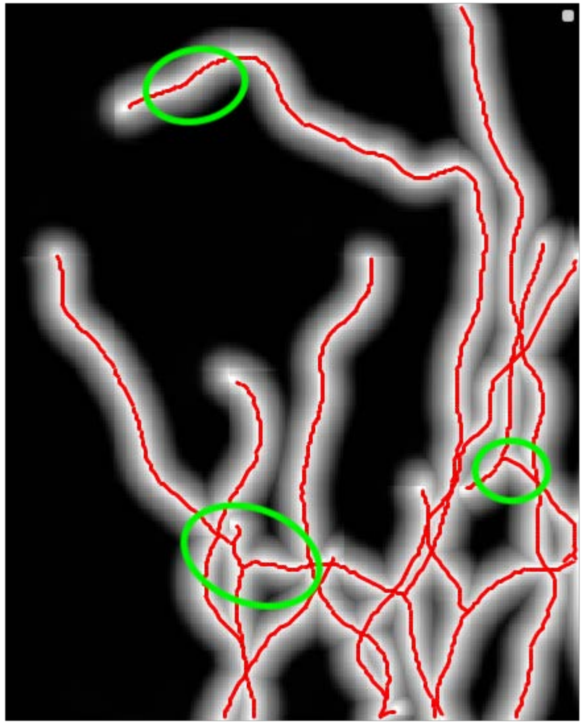
Snake Optimization



Improved Results


Annotated image


Vanilla U-Net


Network snakes

EPFL

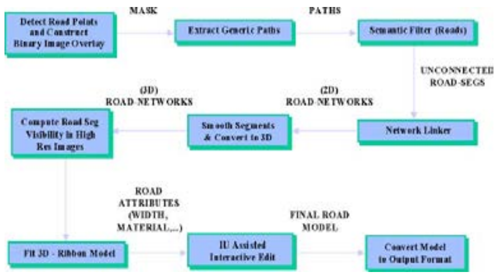
On the Job

The new job is good, we do of course a lot of Deep Learning, but **also some good old-school computer vision** e.g. registration 😉 So the material from Computer Vision class is definitely helpful and I wouldn't change it to another all-Deep Learning class (even in the light of today's Turing Award).

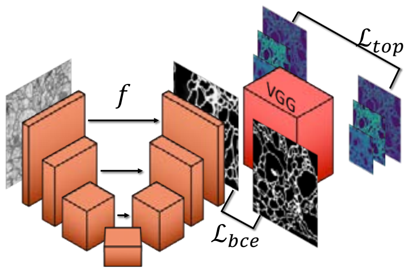
Best,

Agata

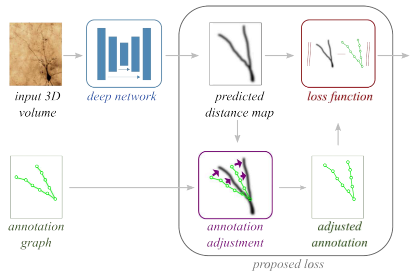
1998 - 2038



1998
Heuristics



2018
Deep Learning



2025
DL + Topology

?

2030

It is difficult to make predictions, especially about the future.
Sometimes attributed to Niels Bohr.

Analogy

Low level processing

- Uses Deep Nets to find the most promising locations to focus on.

High level processing

- Performs tree-based search when possible.

- Relies on reinforcement learning and other ML techniques to train.

In Short

- Edge and image information is noisy.

- Models are required to make sense of it.

→ An appropriate combination of graph-based techniques, machine learning, and semi-automated tools is required.