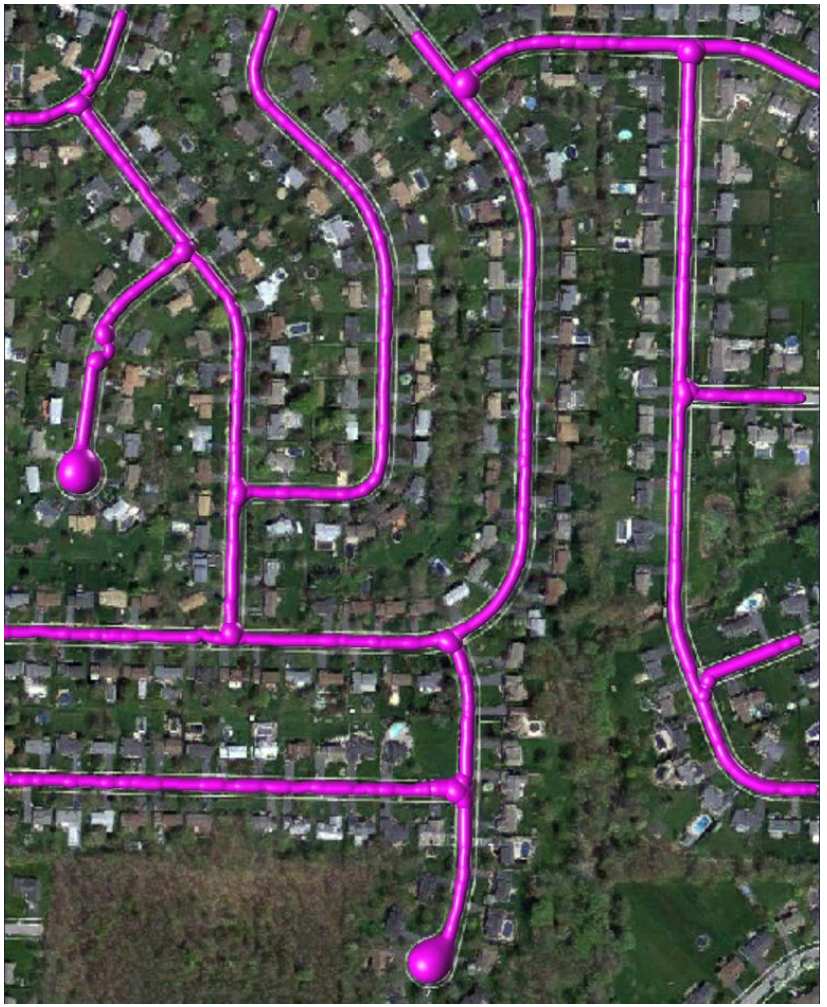
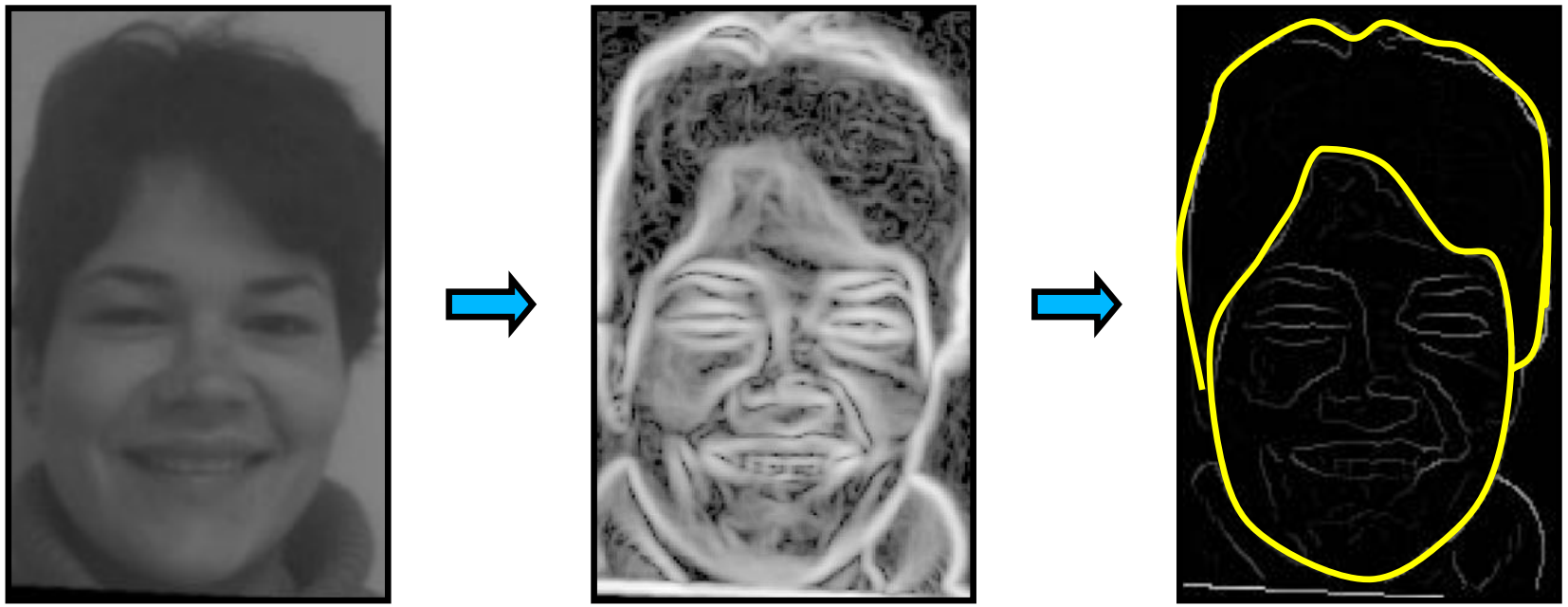
DELINEATION

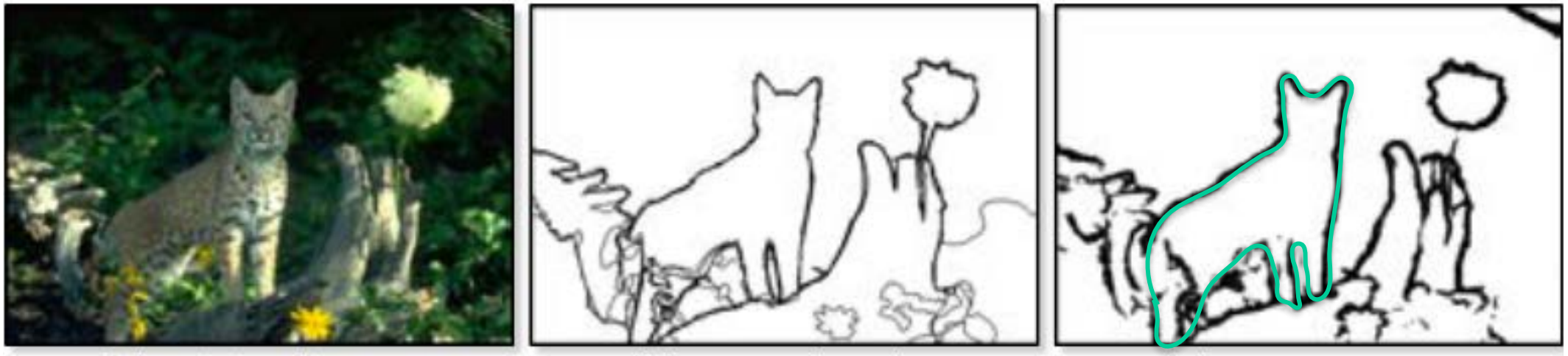
- Dynamic Programming
- Deformable Models
- Hough Transform
- Graph Based Approaches



FROM GRADIENTS TO OUTLINES



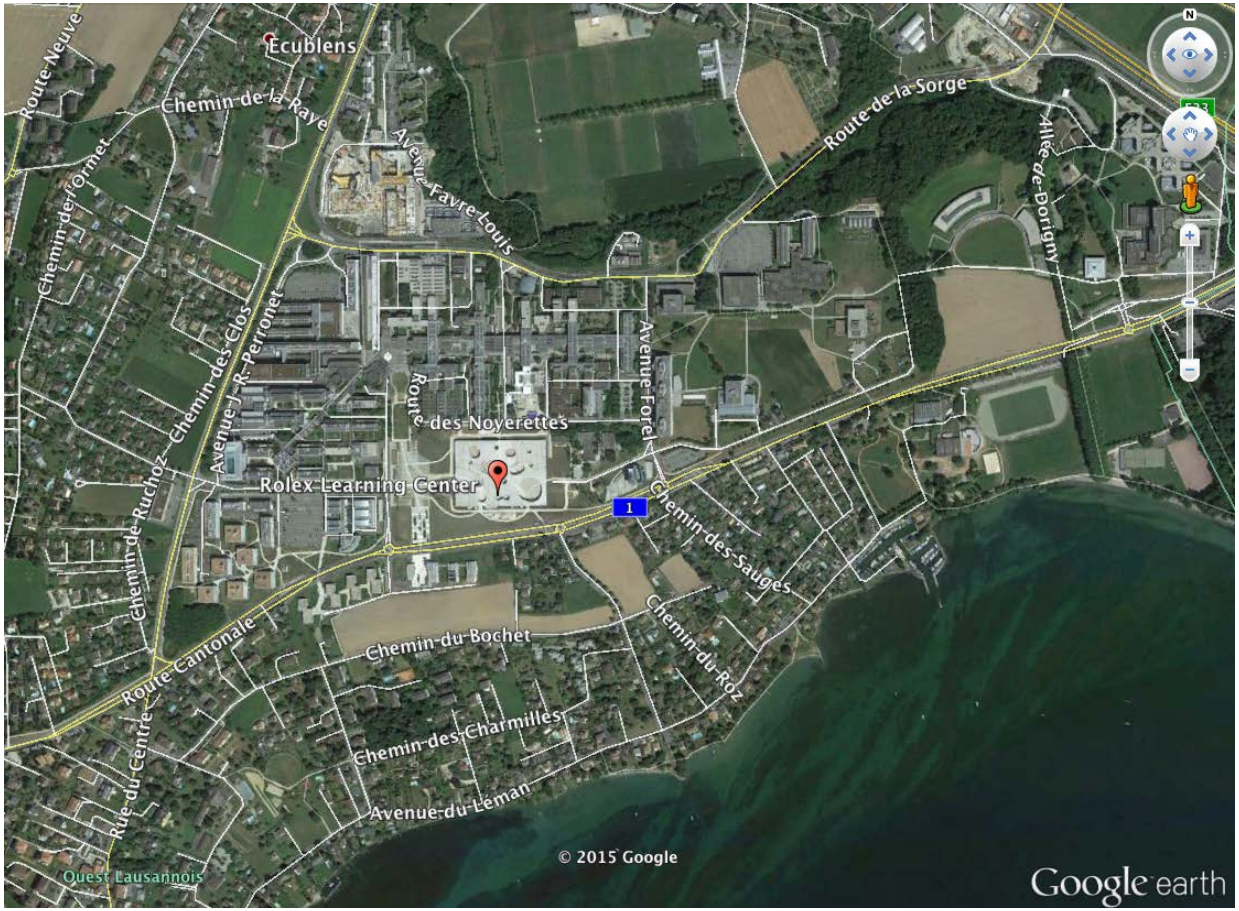
FROM DEEP-NETS TO OUTLINES



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

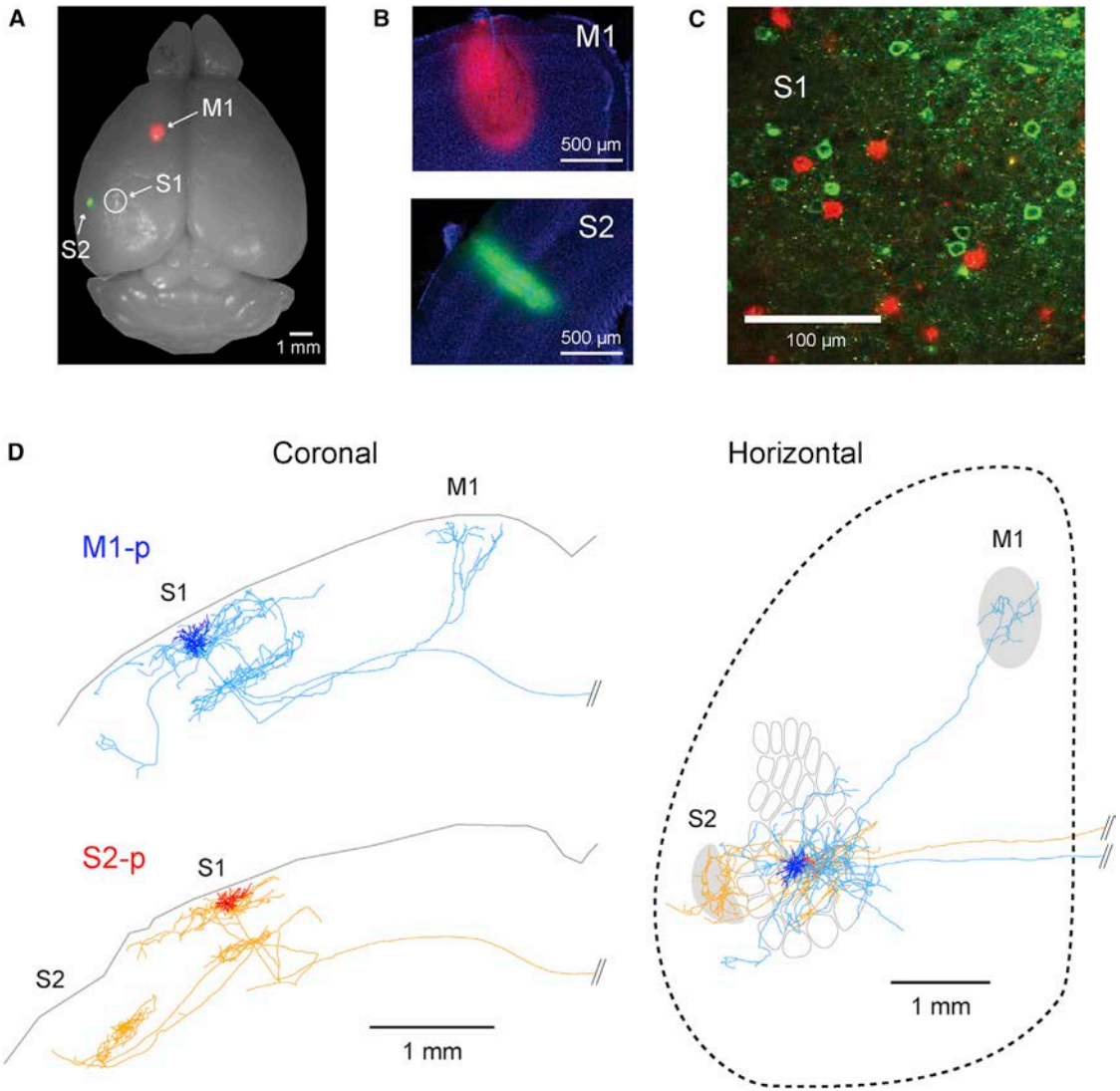
—> Still work to do!

MAPS AND OVERLAYS



—> Overlays needed

CONNECTOMICS



—> Topology needed

Courtesy of C. Petersen

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.

TECHNIQUES

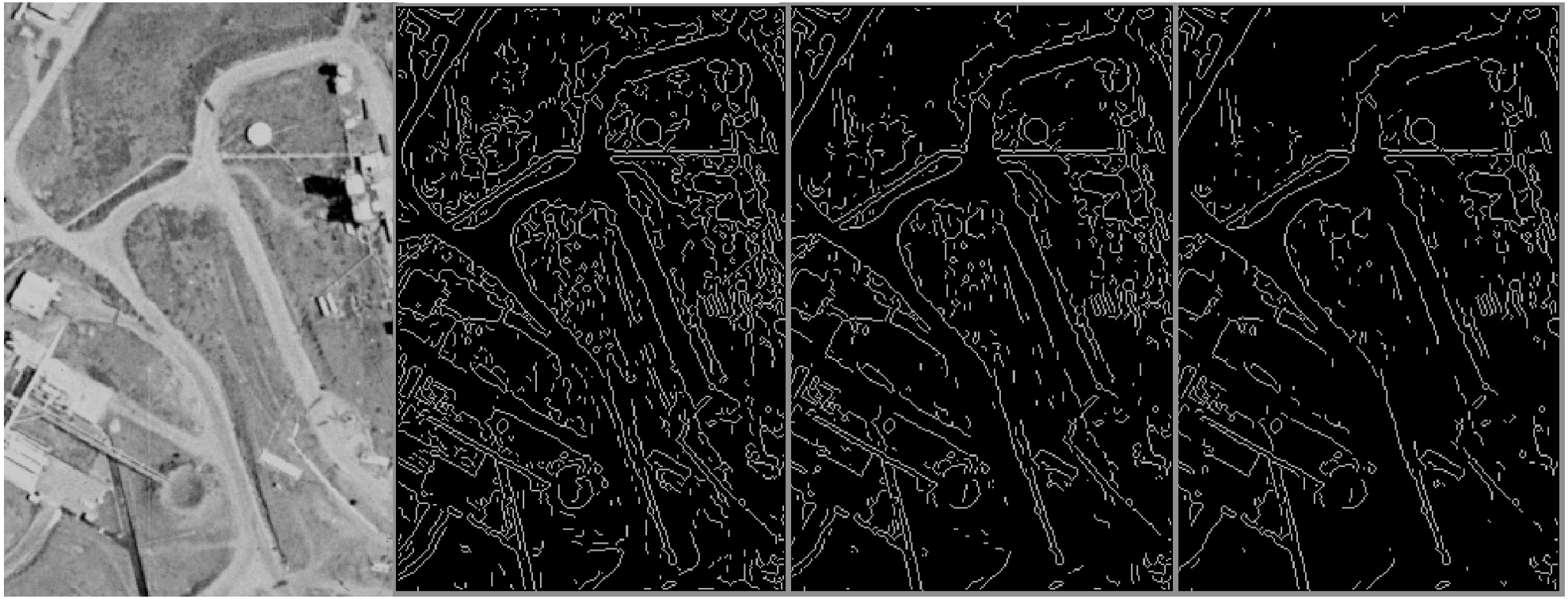
Semi-Automated Techniques:

- Dynamic Programming
- Deformable Models

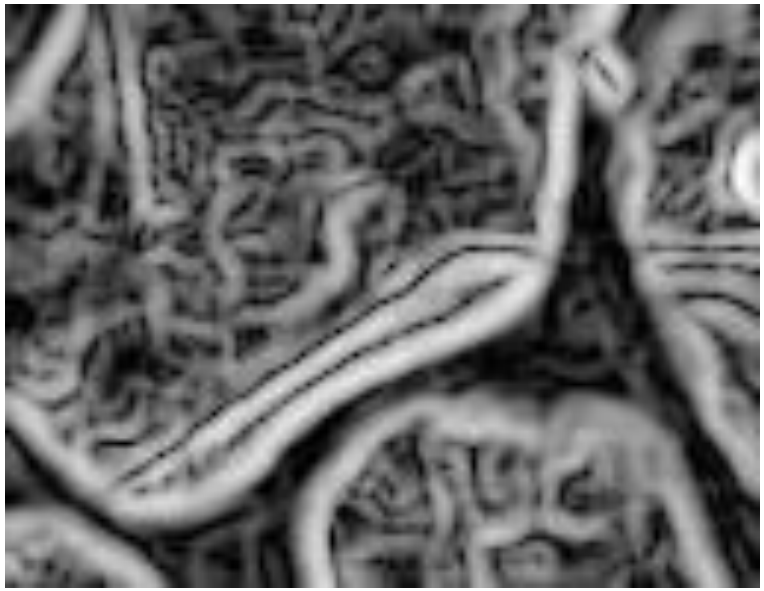
Fully Automated Techniques:

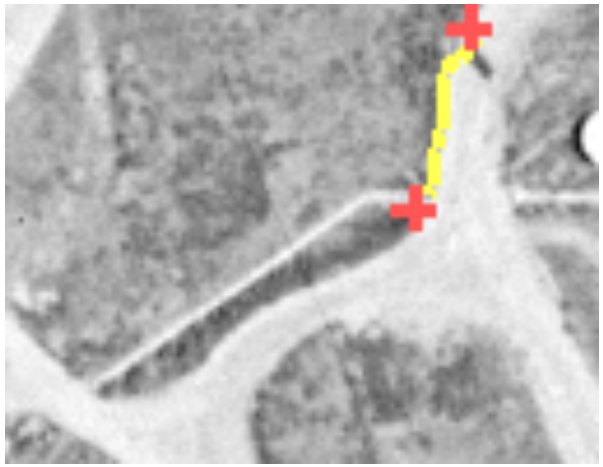
- Hough Transform
- Graph Based Approaches

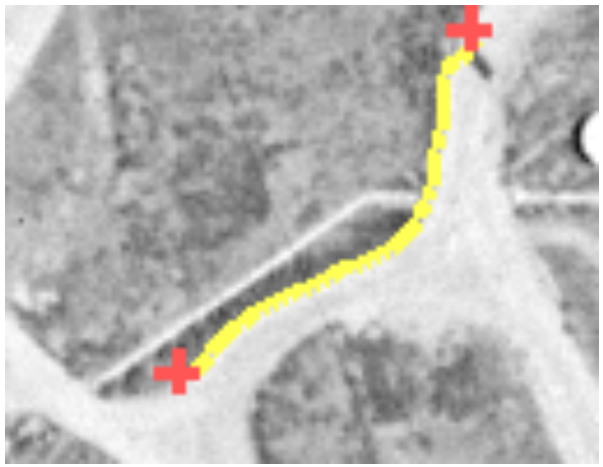
ROAD IMAGE

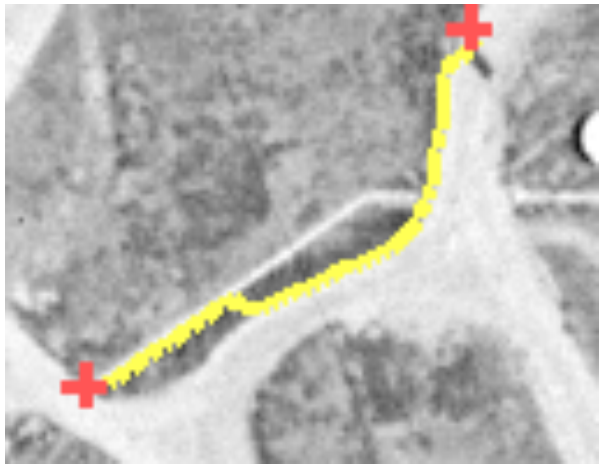


INTERACTIVE DELINEATION





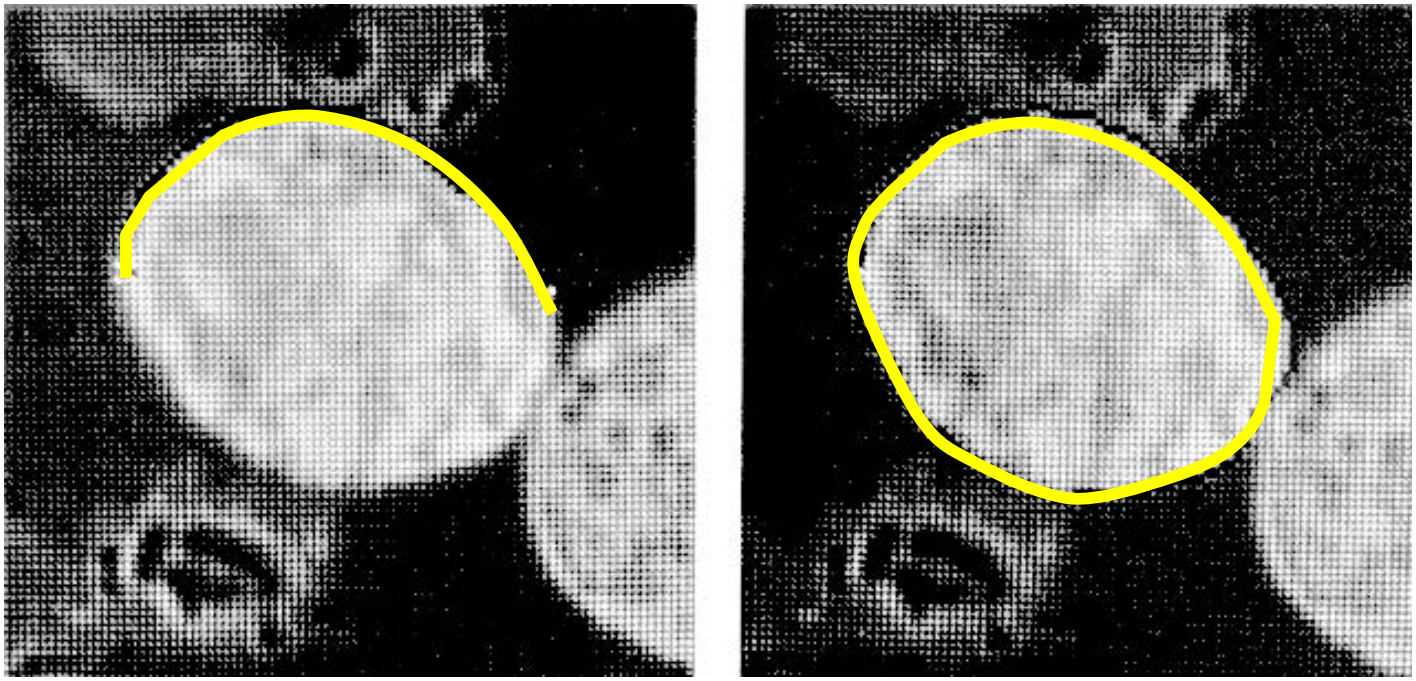




LIVE WIRE

Mortensen and Barrett,
SIGGRAPH, 1995

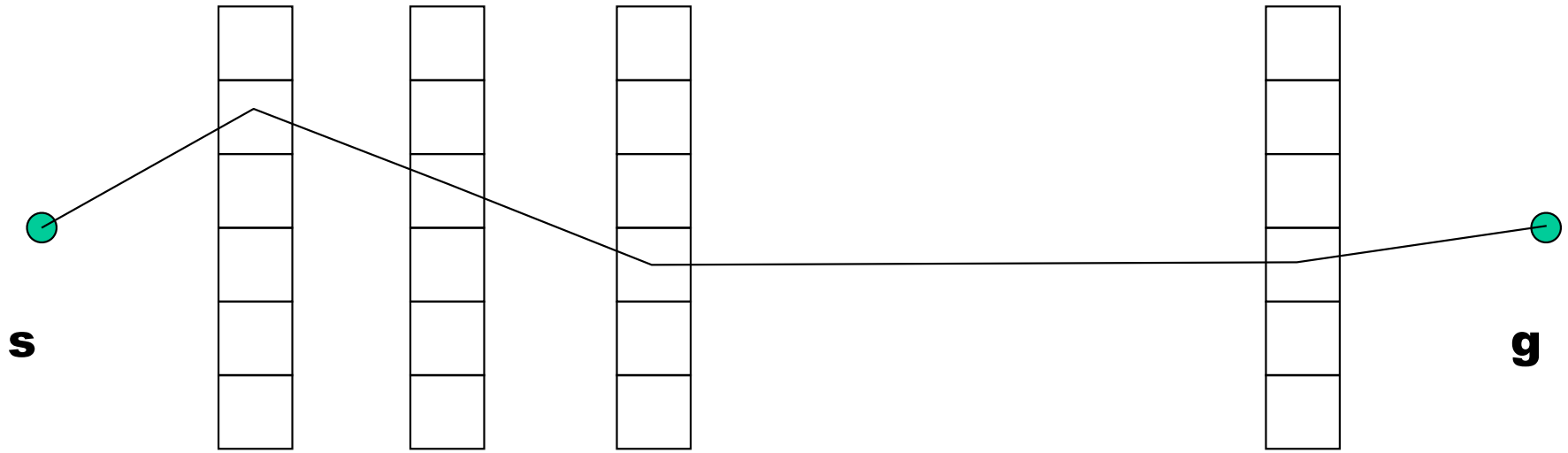
DYNAMIC PROGRAMMING



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

$$q(x_k, x_{k+1}) = diff[\phi(x_k), \phi(x_{k+1})]$$

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

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

define

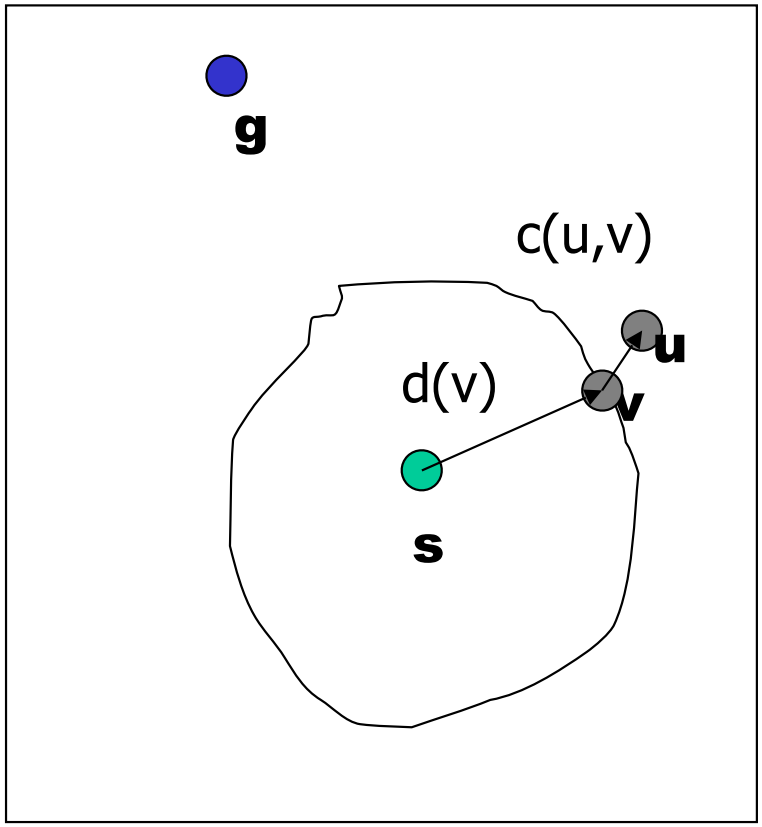
$$\begin{aligned}
f_1(x_2) &= \min_{x_1} (h(s, x_1) + h(x_1, x_2)) \\
f_2(x_3) &= \min_{x_2} (h(x_2, x_3) + f_1(x_2)) \\
&\vdots \quad \vdots \\
f_{n-1}(x_n) &= \min_{x_{n-1}} (h(x_{n-1}, x_n) + f_{n-2}(x_{n-1}))
\end{aligned}$$

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

2D DYNAMIC PROGRAMMING

Notations:

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



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

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:

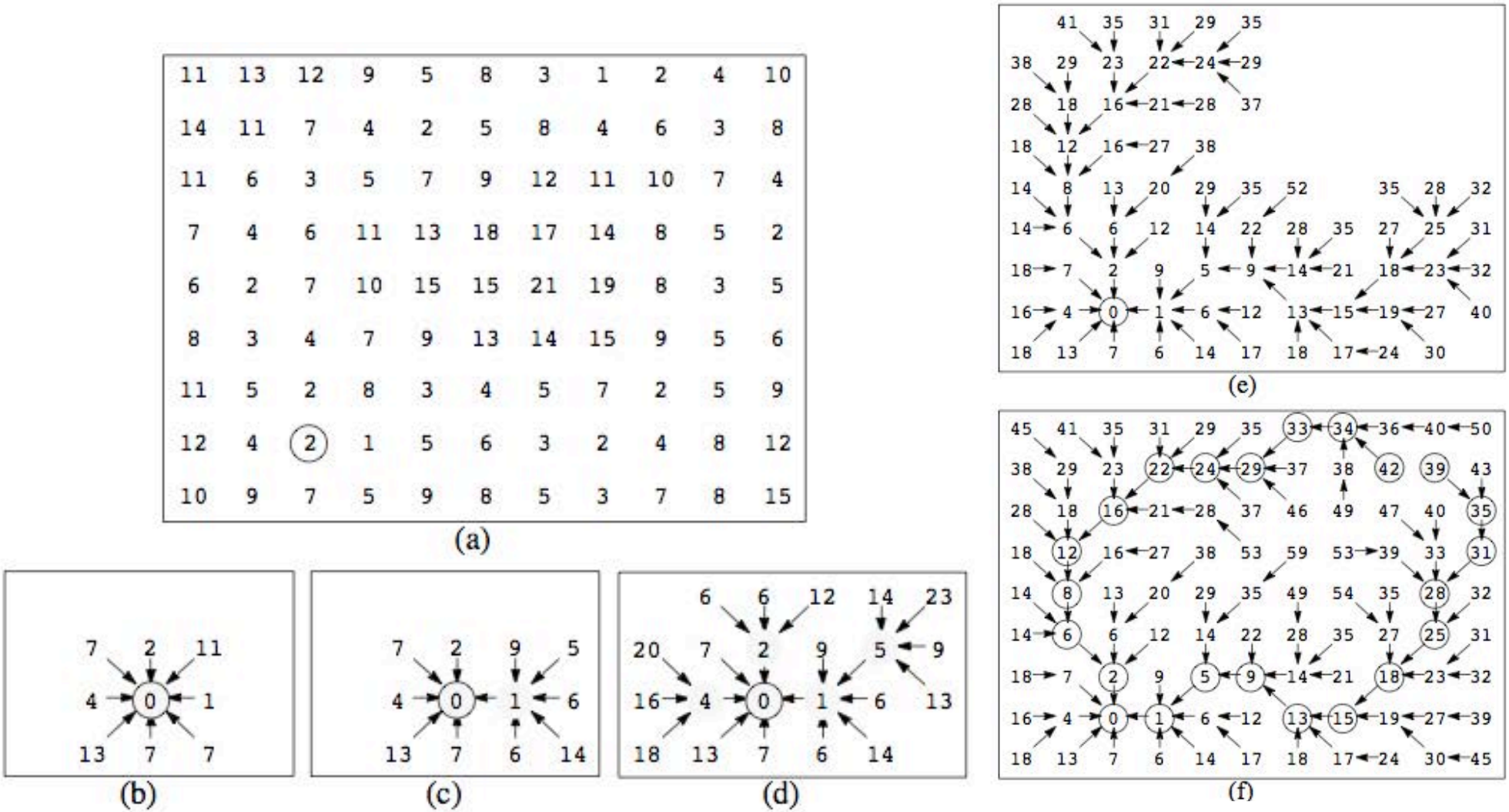
$$c(u,v) = 255 - \tfrac{1}{2}\,(g(u) + g(v))$$

- 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

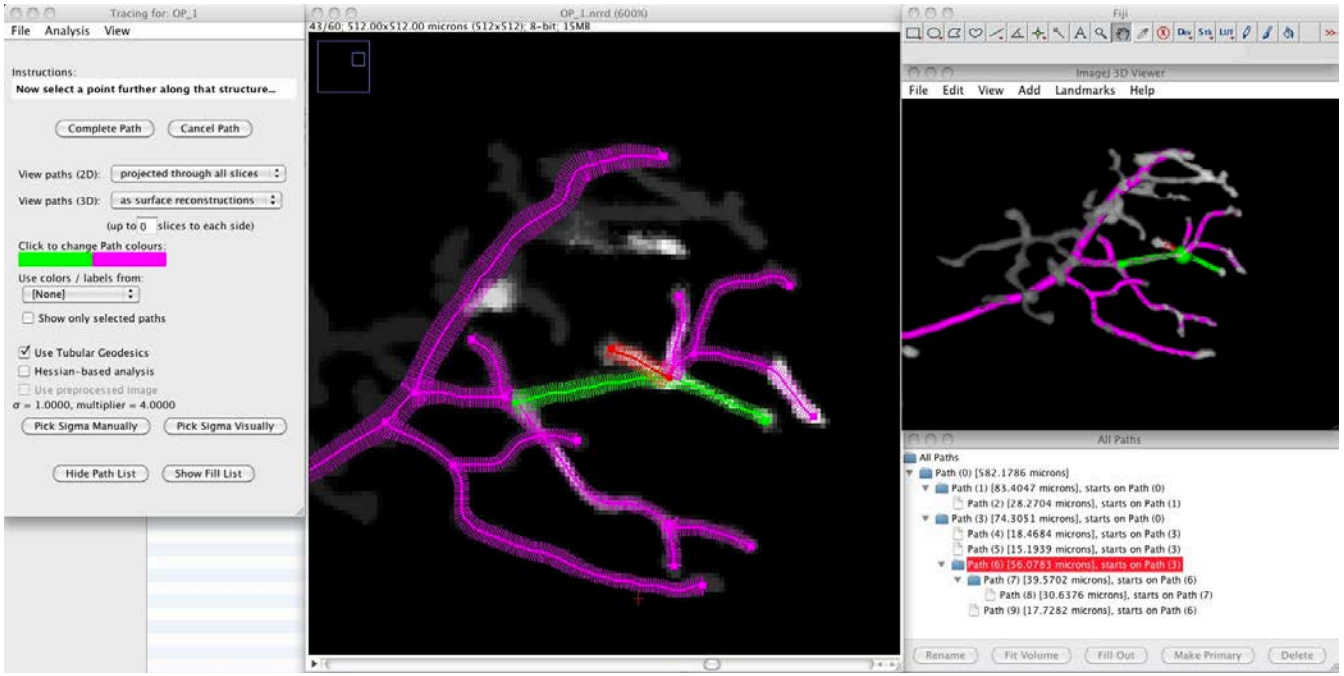


(a) Local cost map. (b) Seed point expanded. (c) 2 points expanded. (d) 5 points expanded. (e) 47 points expanded. (f) Completed cost path-pointer map with optimal paths shown from nodes with total costs 42 and 39.

FACE IMAGE

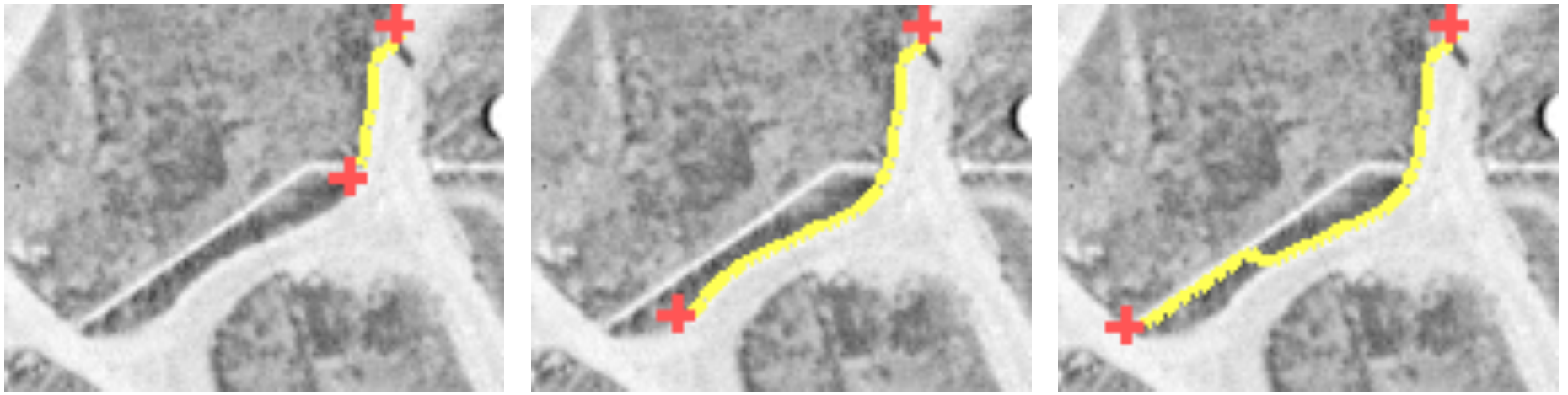
LIVE WIRE

TRACING NEURONS



- Link user provided points and extracts centerline points.
- Fiji Plug-In downloadable from http://cvlab.epfl.ch/software/delin

OPEN ISSUES



- The "optimal" path is not always the "best" one.
- Difficult to impose global constraints.
- The cost grows exponentially with the dimension of the problem.

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

TECHNIQUES

Semi-Automated Techniques:

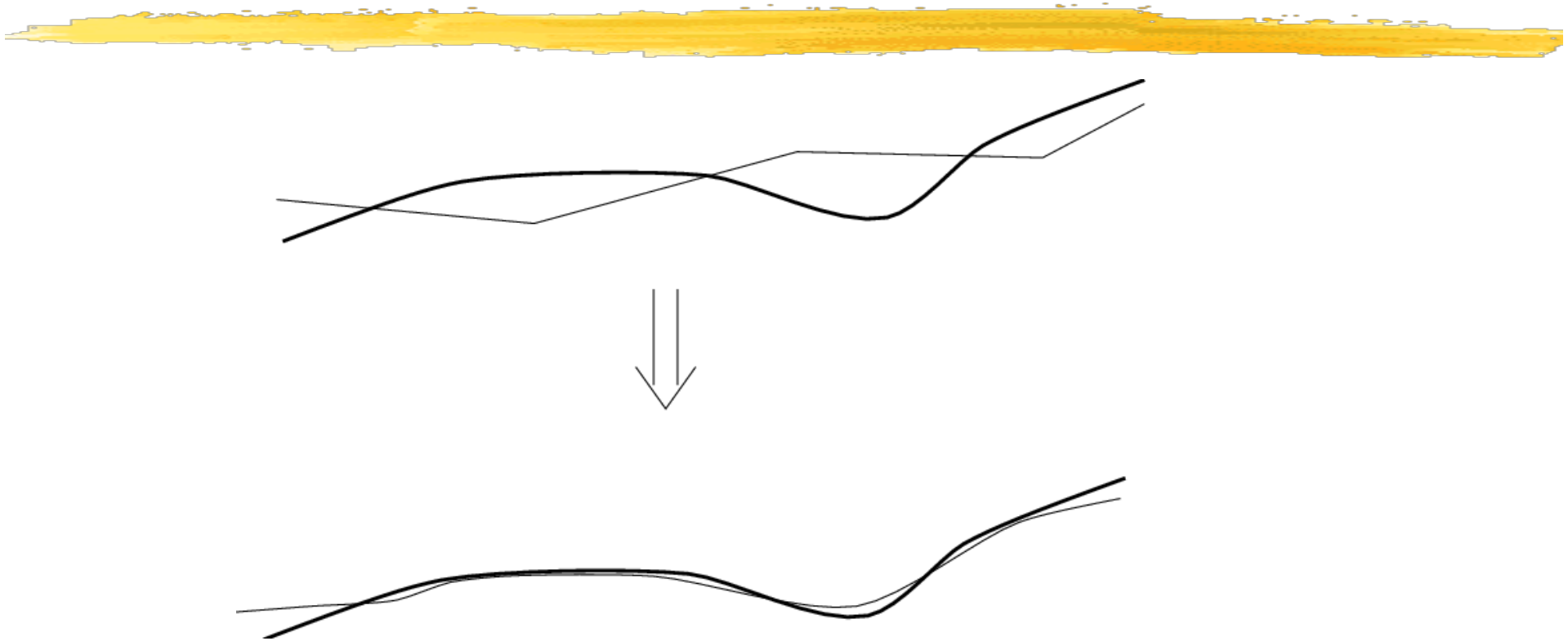
- Dynamic programming
- Deformable Models

Fully Automated Techniques:

- Hough transform
- Graph Based Approaches

SNAKES

2—D SNAKE

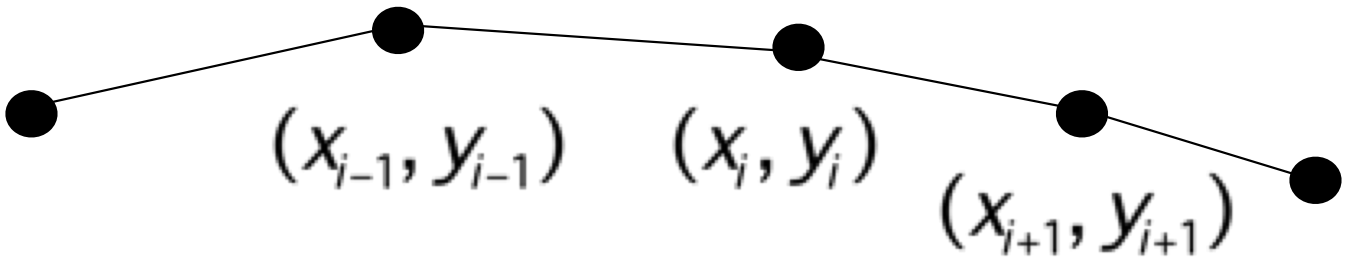


Deformable contours that

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

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

ENERGY LANSCAPE



$$
\begin{aligned}
E &= 1/L \int_{s=0}^{1} (-\lambda G(x(s), y(s)) + \gamma^2(s)) ds \\
&= 1/L \int_{s=0}^{1} (-\lambda G(x(s), y(s)) + \frac{\delta^2 x}{\delta s^2}^2 + \frac{\delta^2 y}{\delta s^2}^2) ds \\
&= 1/L(-\lambda \sum_{i=0}^{N} G(x_i, y_i) + \sum_{i=1}^{N-1} ((2x_i - x_{i-1} - x_{i+1})^2 + (2y_i - y_{i-1} - y_{i+1})^2)
\end{aligned}
$$

MATRIX NOTATION

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

$$X = [x_1, \ldots, x_N]^t$$

$$Y = [y_1, \ldots, y_N]^t$$

$$K = \begin{bmatrix}
. & . & . & . & . & . & . & . & . & . & . & . & . \\
. & . & 1 & -4 & 6 & -4 & 1 & . & . & . & . & . & . \\
. & . & . & 1 & -4 & 6 & -4 & 1 & . & . & . & . & . \\
. & . & . & . & 1 & -4 & 6 & -4 & 1 & . & . & . & . \\
. & . & . & . & . & 1 & -4 & 6 & -4 & 1 & . & . & . \\
. & . & . & . & . & . & 1 & -4 & 6 & -4 & 1 & . & . \\
. & . & . & . & . & . & . & . & . & . & . & . & .
\end{bmatrix}$$

LOCAL OPTIMUM

$$\frac{\delta E}{\delta X} = \frac{\delta E_G}{\delta X} + KX = 0$$

$$\frac{\delta E}{\delta Y} = \frac{\delta E_G}{\delta Y} + KY = 0$$

But K is not invertible!

DYNAMICS

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

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

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

ITERATING

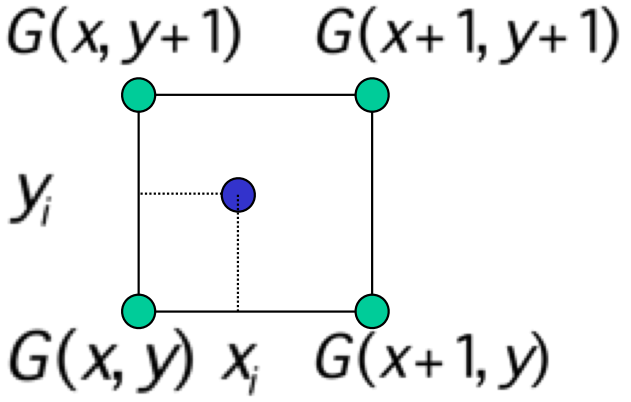
At every step:

$$0 = \frac{\delta E_G}{\delta X} + KX_t + \alpha(X_t - X_{t-1}) \quad \Rightarrow \quad (K + \alpha I)X_t = \alpha X_{t-1} - \frac{\delta E_G}{\delta X}$$

$$0 = \frac{\delta E_G}{\delta Y} + KY_t + \alpha(Y_t - Y_{t-1}) \quad \Rightarrow \quad (K + \alpha I)Y_t = \alpha Y_{t-1} - \frac{\delta E_G}{\delta Y}$$

→ Solve two linear equations at each iteration.

DERIVATIVES OF THE GRADIENT

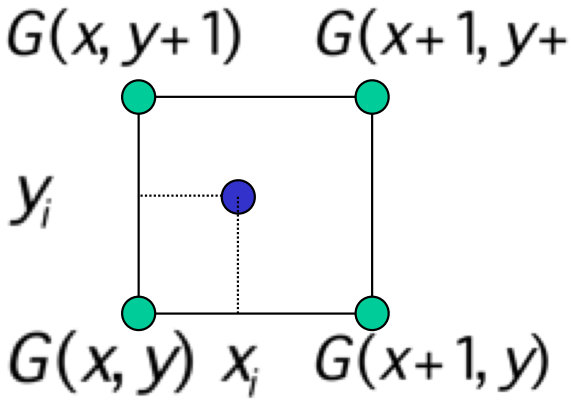


$$E_G = -\frac{1}{N}\sum_{i=1}^{N} G(x_i, y_i)$$

$$\frac{\partial E_G}{\partial X} = \left[\frac{\partial E_G}{\partial x_1} \quad \cdots \quad \frac{\partial E_G}{\partial x_N}\right] \quad , \quad \frac{\partial E_G}{\partial Y} = \left[\frac{\partial E_G}{\partial y_1} \quad \cdots \quad \frac{\partial E_G}{\partial y_N}\right]$$

$$\frac{\partial E_G}{\partial x_i} = -\frac{1}{N}\frac{\partial G}{\partial x_i}(x_i, y_i) \quad , \quad \frac{\partial E_G}{\partial y_i} = -\frac{1}{N}\frac{\partial G}{\partial y_i}(x_i, y_i)$$

BILINEAR INTERPOLATION



$$p = x_i - x$$

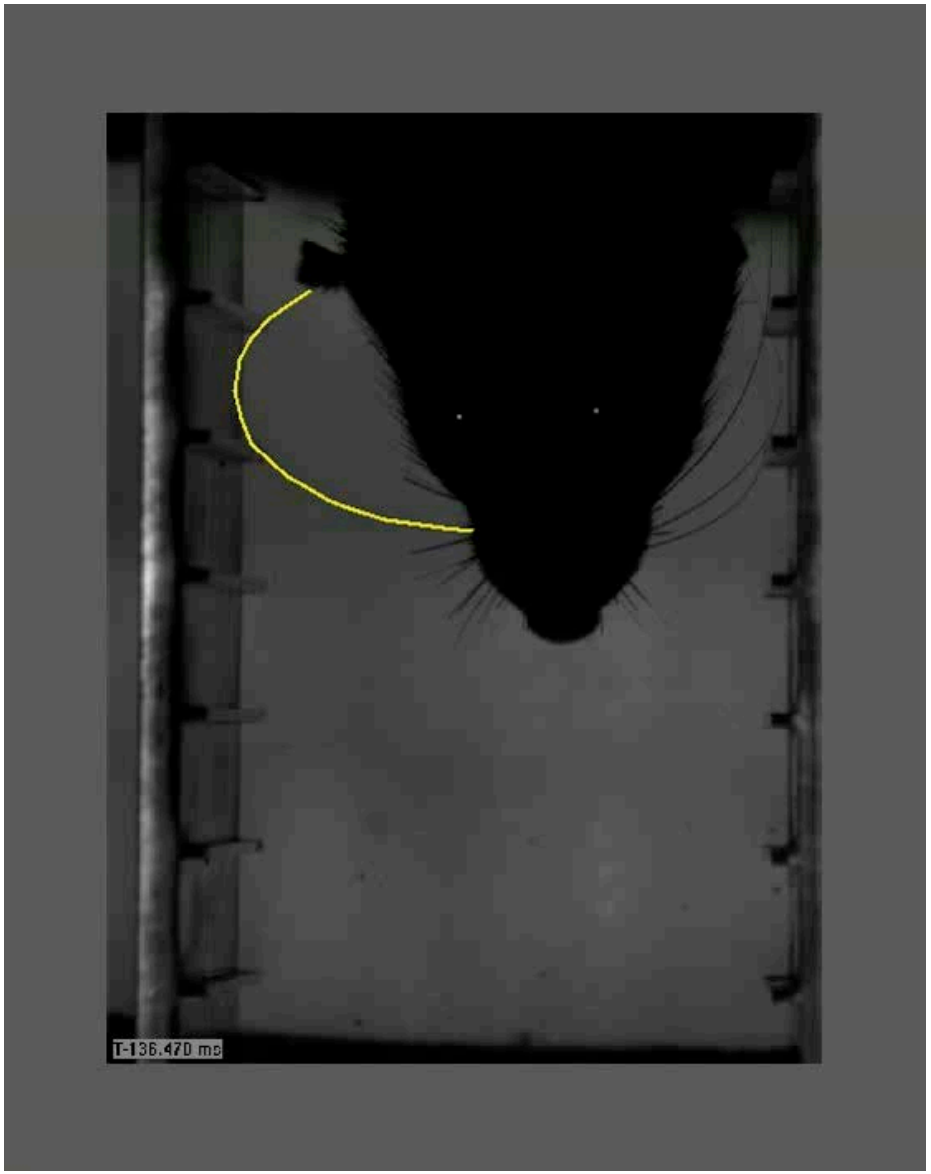
$$q = y_i - y$$

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

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

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

OPEN AND CLOSED SNAKES

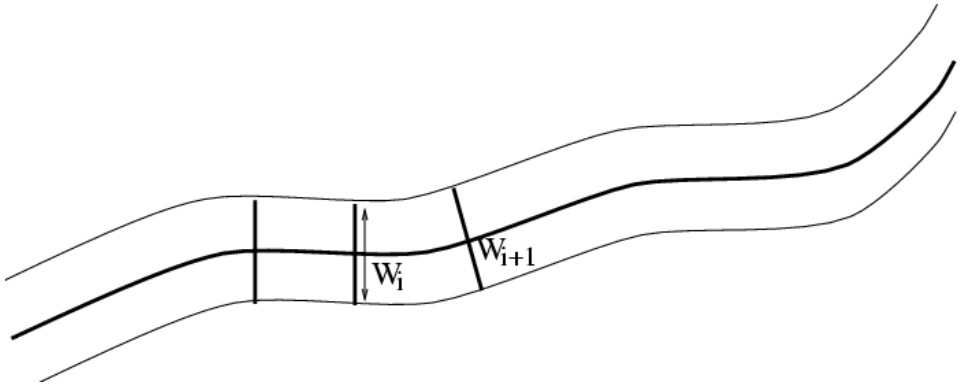


NETWORK SNAKES

--> Updated field boundaries.

Butenuth, Phd 2008

RIBBON SNAKES



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

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

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

DYNAMICS EQUATIONS

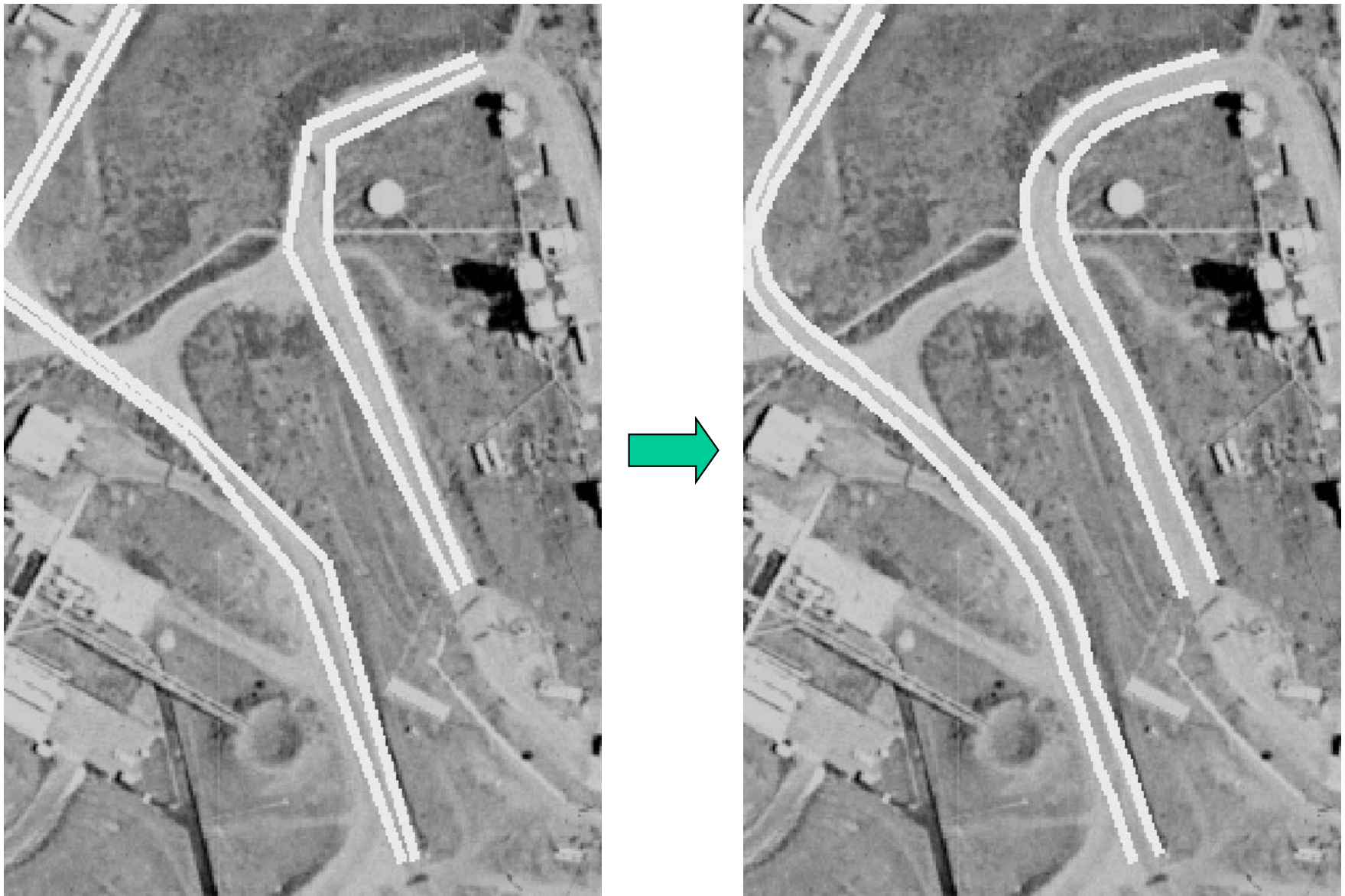
$$(K + \alpha I)X_t = \alpha X_{t-1} - \frac{\delta E_G}{\delta X}$$

$$(K + \alpha I)Y_t = \alpha Y_{t-1} - \frac{\delta E_G}{\delta Y}$$

$$(K + \alpha I)W_t = \alpha W_{t-1} - \frac{\delta E_G}{\delta W}$$

→ Solve three linear equations at each iteration.

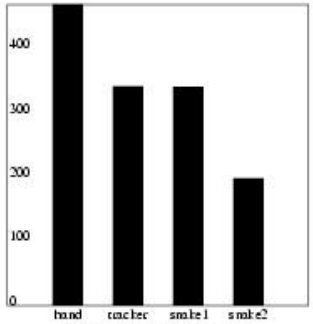
DELINEATING ROADS



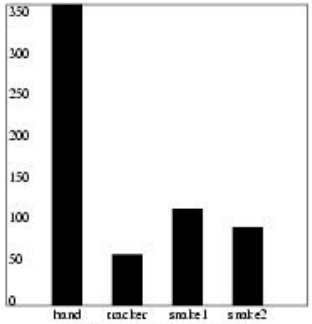
DELINEATING ROADS

DELINEATING ROADS

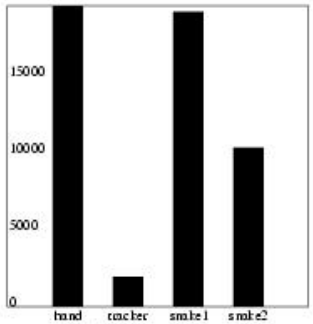
EVALUATION



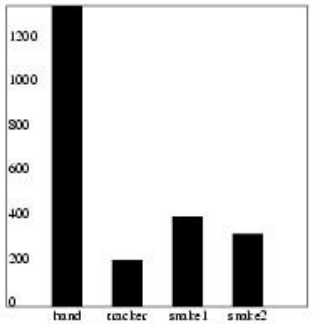
Number of object clicks



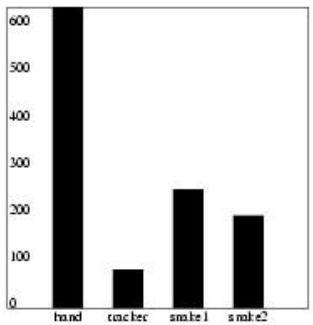
Number of mouse moves



Total mouse distance

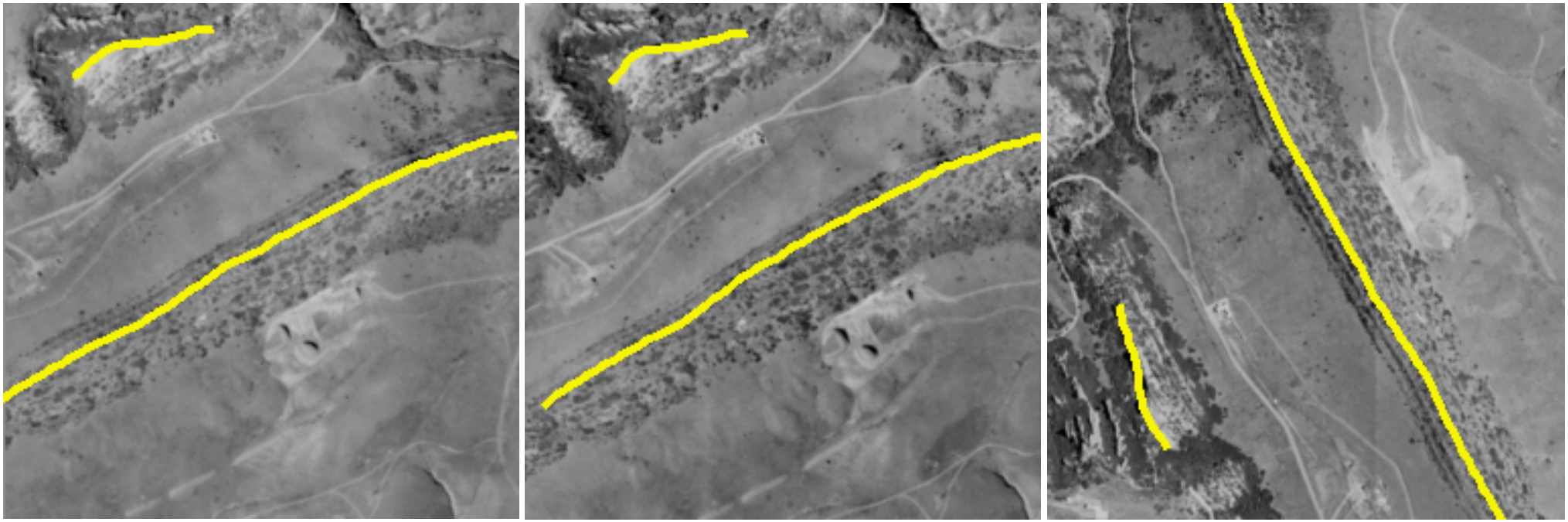


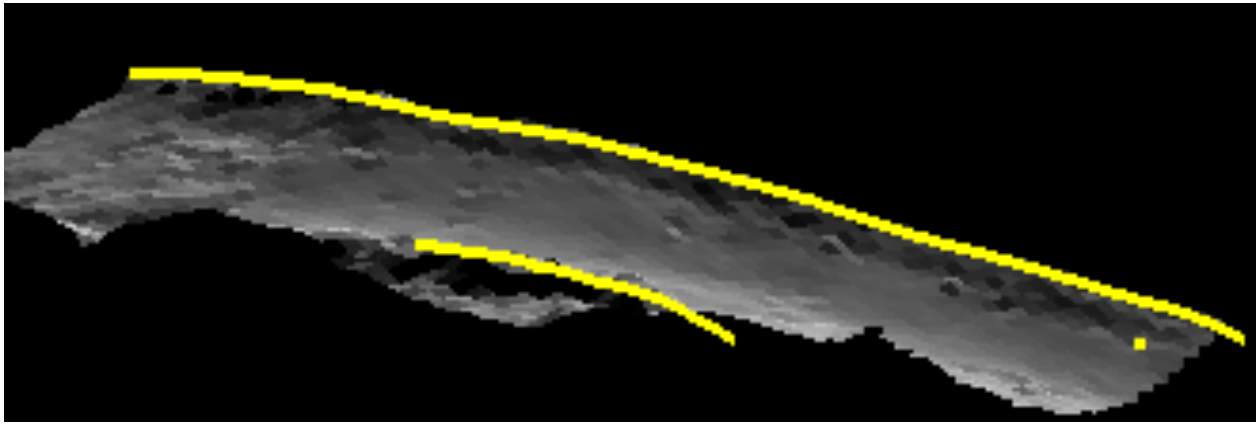
Total mouse move time



Total near mouse move time

RIDGE LINES

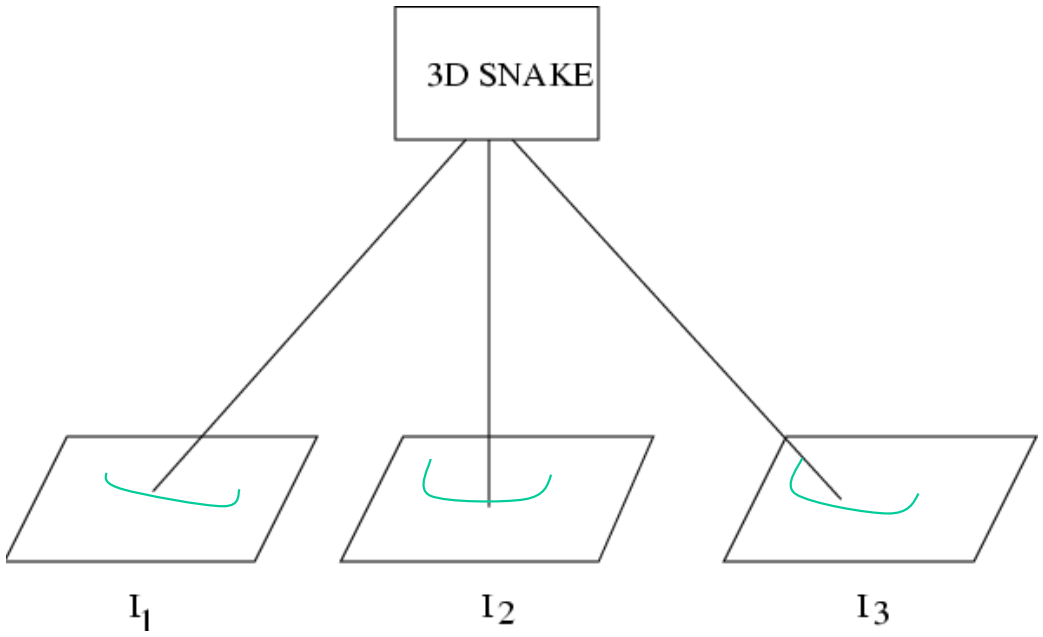


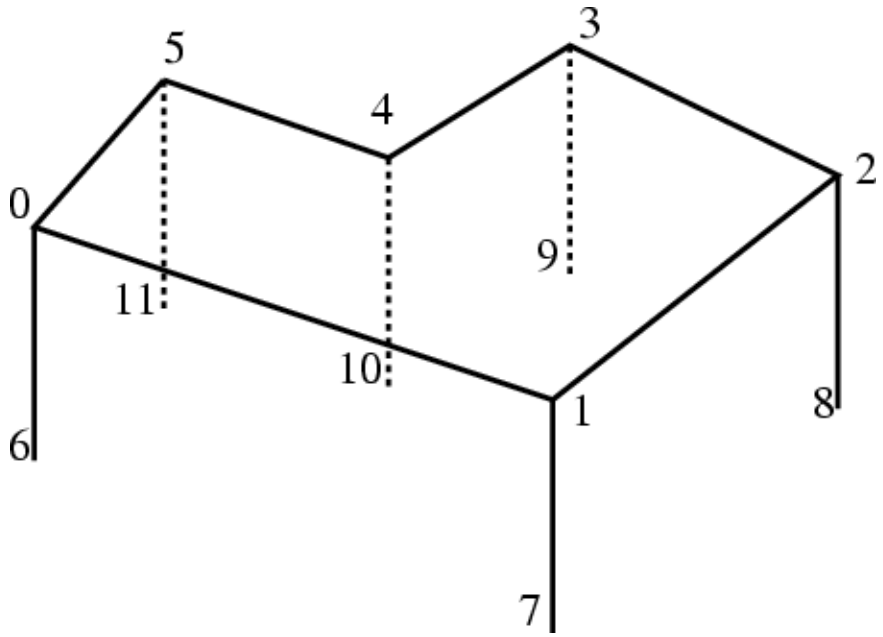


BUILDING


Ft Hood 2x3
Fort Hood Image-4
Fort Hood Image-3
Fort Hood Image-4
Fort Hood Image-3
Fort Hood Image-4
Fort Hood Image-3

3—D SNAKES





Smooth 3—D snake

Rectilinear 3—D snake

DYNAMICS EQUATIONS

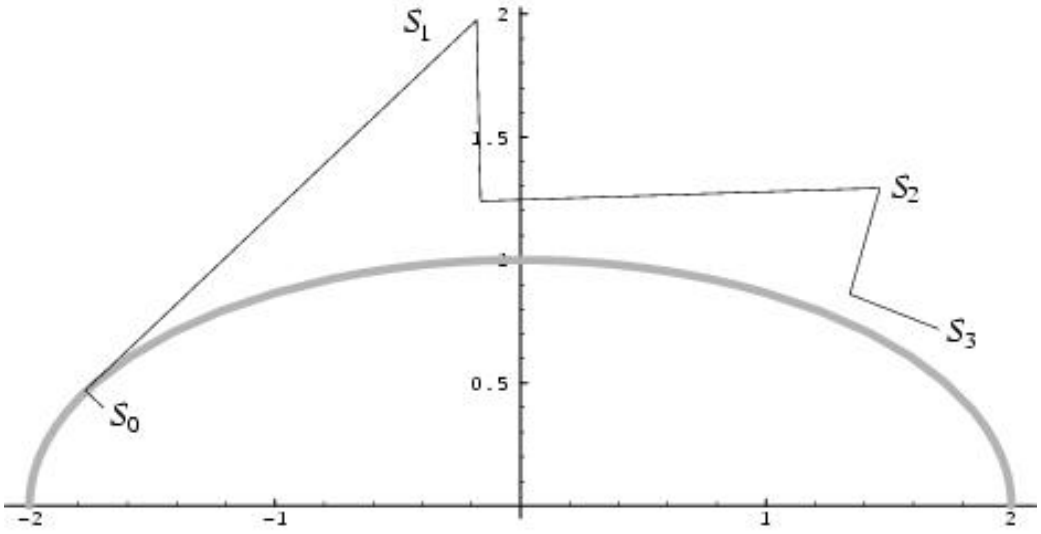
$$(K + \alpha I)X_t = \alpha X_{t-1} - \frac{\delta E_G}{\delta X}$$

$$(K + \alpha I)Y_t = \alpha Y_{t-1} - \frac{\delta E_G}{\delta Y}$$

$$(K + \alpha I)Z_t = \alpha Z_{t-1} - \frac{\delta E_G}{\delta Z}$$

→ Solve three linear equations at each iteration.

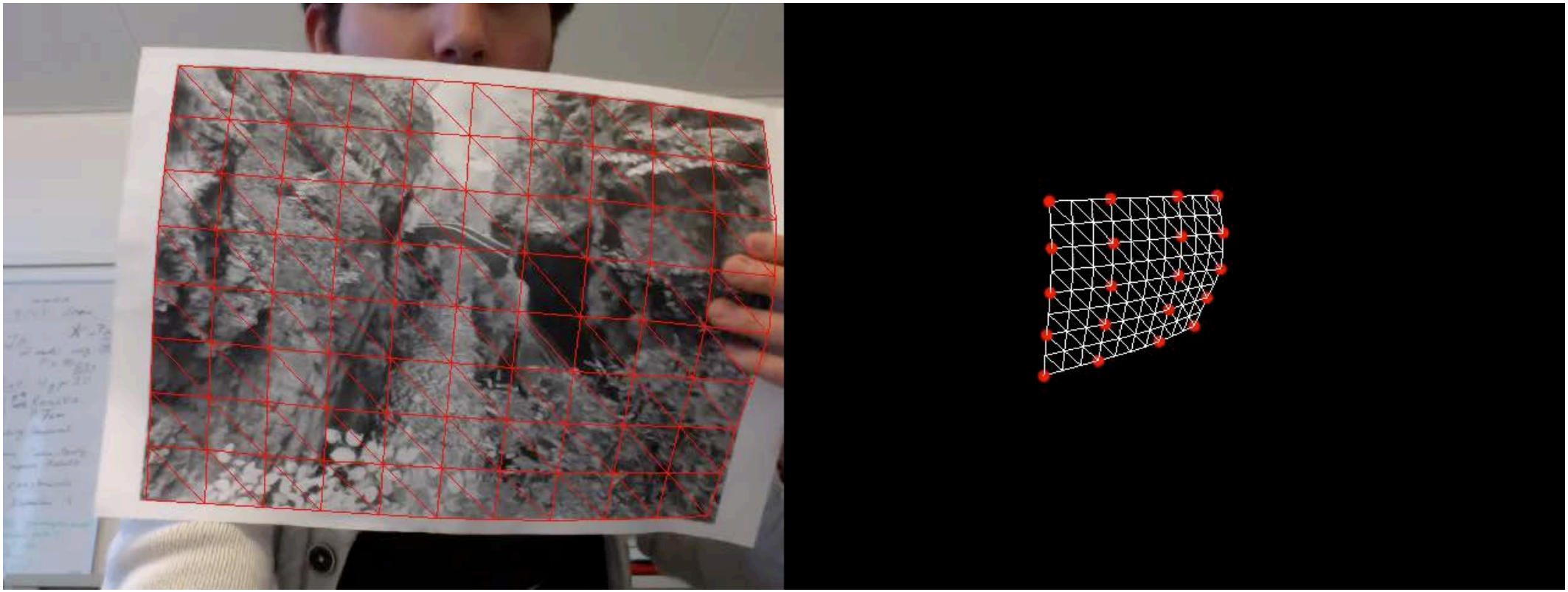
CONSTRAINED OPTIMIZATION



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

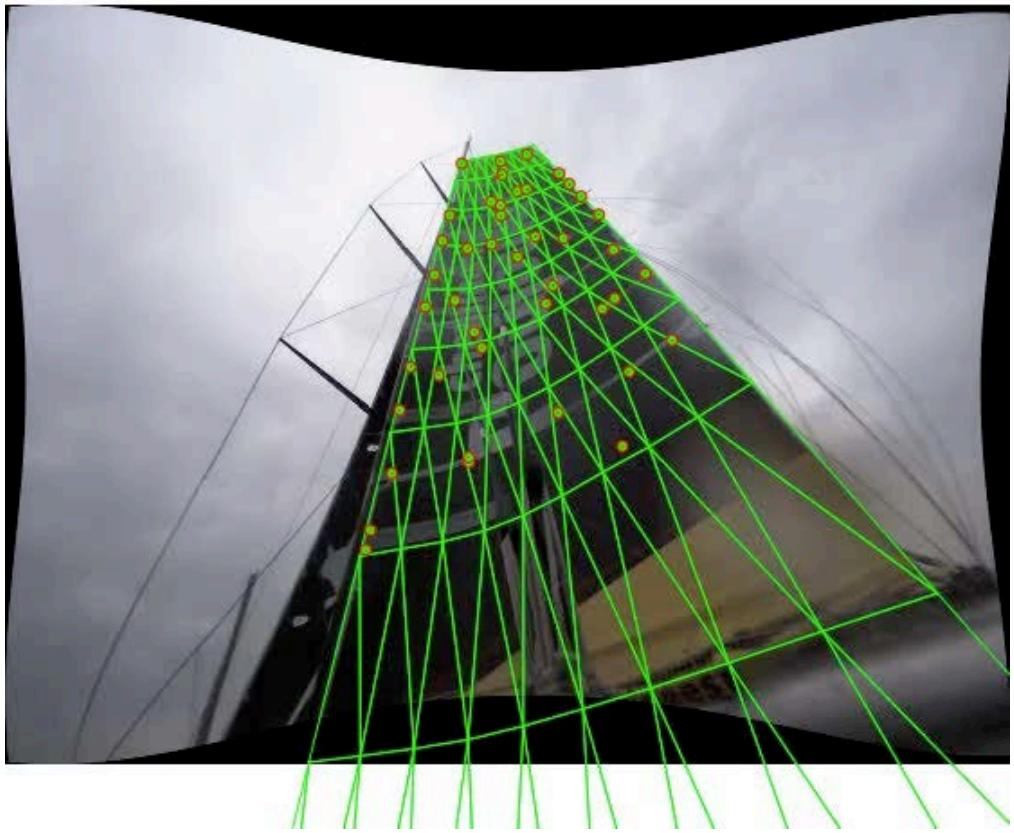
SITE MODELING

REAL-TIME 3D SHAPE



Ostlund et al., ECCV'12.

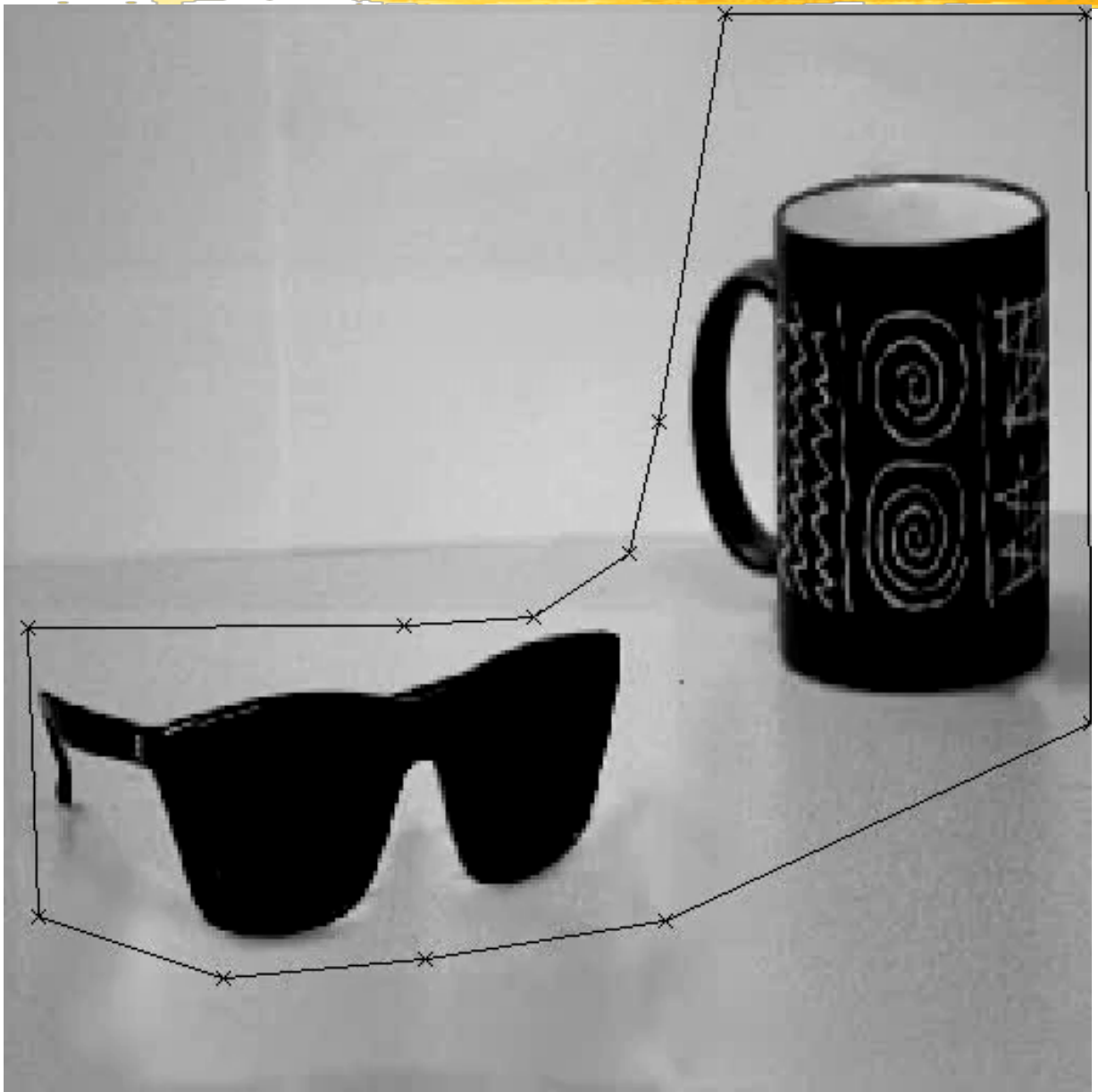
SAILS



→ Real-time performance evaluation.

WINGS

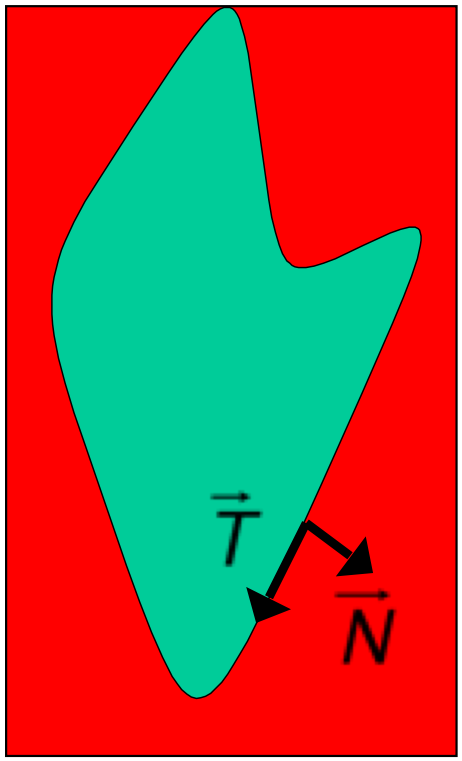
LEVEL SETS



CURVE EVOLUTION

Generic formulation: $\dfrac{\partial C}{\partial t} = \alpha(s,t)\vec{T} + \beta(s,t)\vec{N}$

Reparameterization: $\dfrac{\partial C}{\partial t} = \beta(s,t)\vec{N}$



Special cases:

Prairie fire $\beta = \pm 1$

Diffusion $\beta = \beta_0 - \beta_1 \kappa$

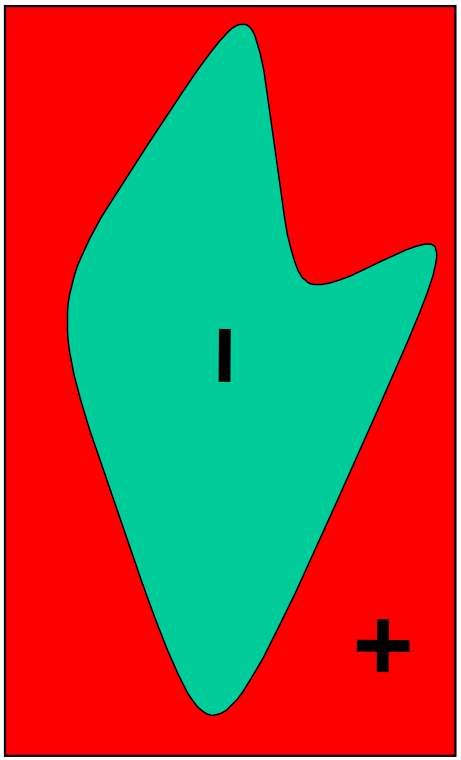
Problems:

1. Involves curve sampling and resampling.
2. Curvature estimates are numerically unstable.

LEVEL SETS

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

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

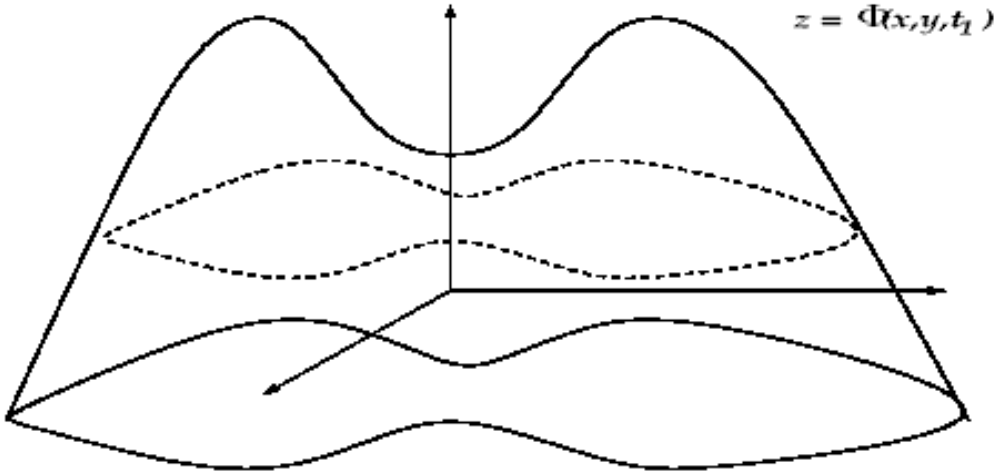


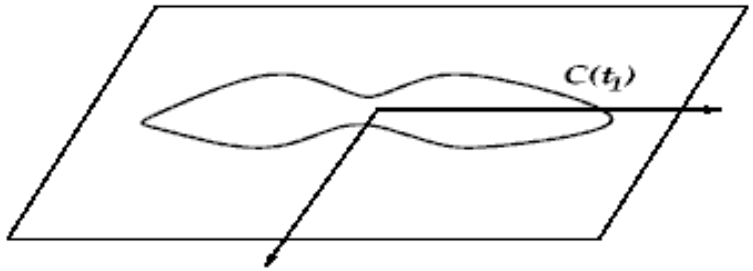
Corresponding evolution equation.

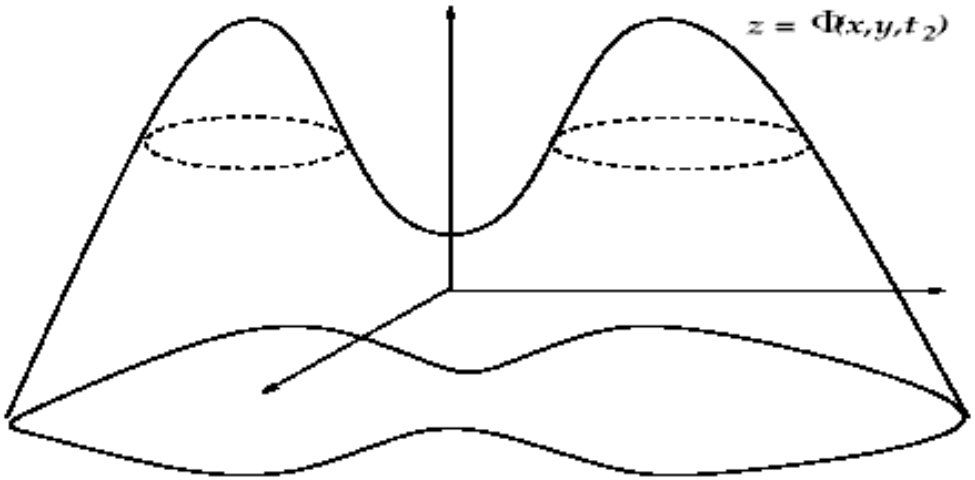
$$\begin{aligned} 0 &= \Phi_t + \beta(\kappa)\,|\nabla\Phi| \\ \text{where } \kappa &= \frac{\Phi_{xx}\Phi_y^2 - 2\Phi_{xy}\Phi_x\Phi_y + \Phi_{yy}\Phi_x^2}{\Phi_x^2 + \Phi_y^2} \end{aligned}$$

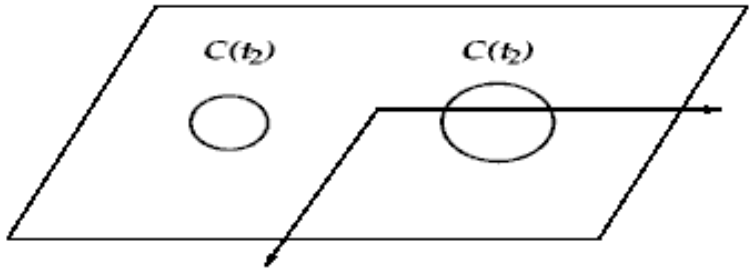
→ Much better numerical stability.

TOPOLOGY CHANGE









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

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

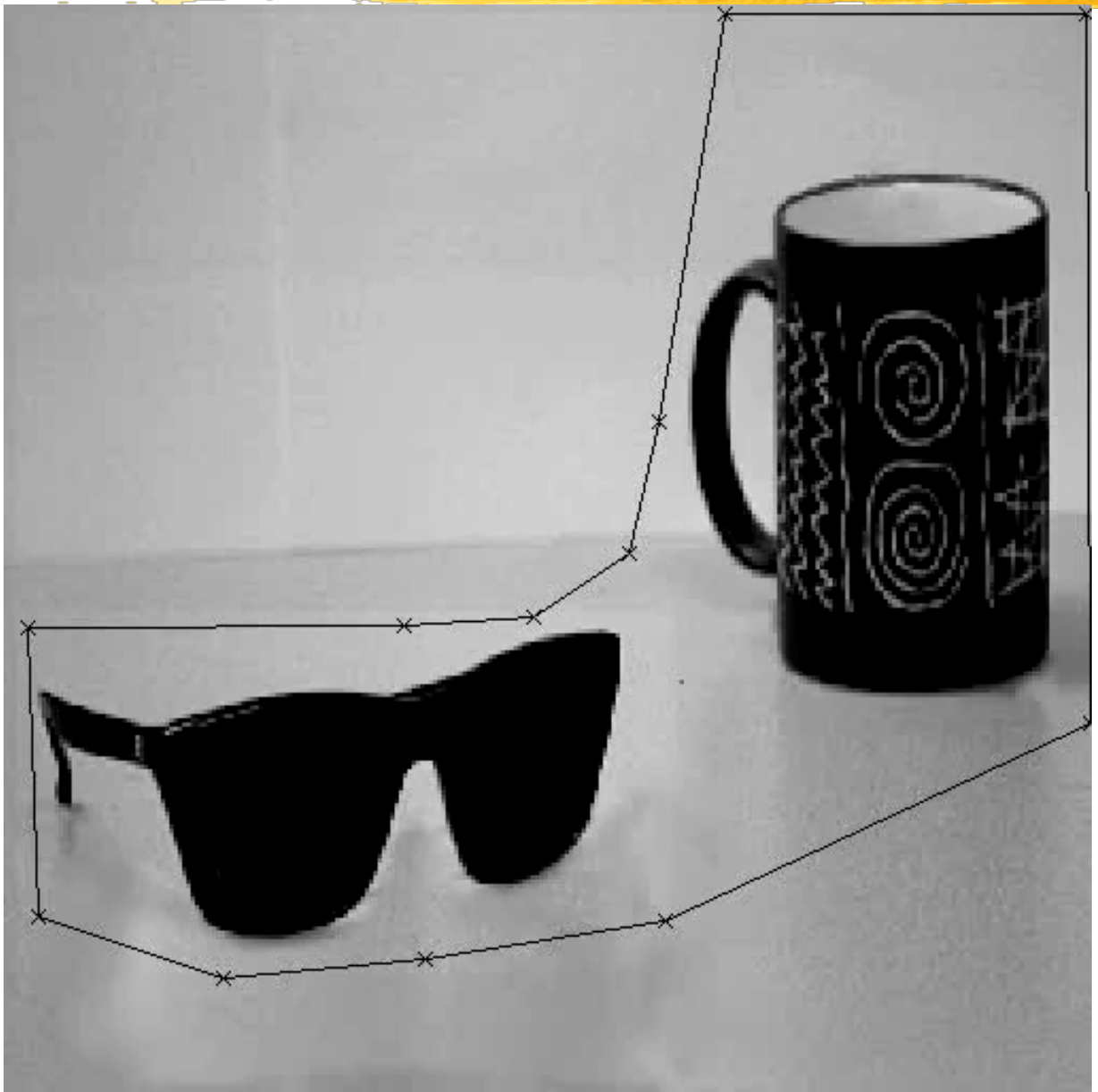
$$\text{where: } \beta(\kappa) = k_I(1 - \epsilon\kappa)$$

$$k_I = \frac{1}{1 + \nabla I}$$

→ Expansion stops at the boundaries.

LEVEL SETS

LEVEL SETS



TECHNIQUES

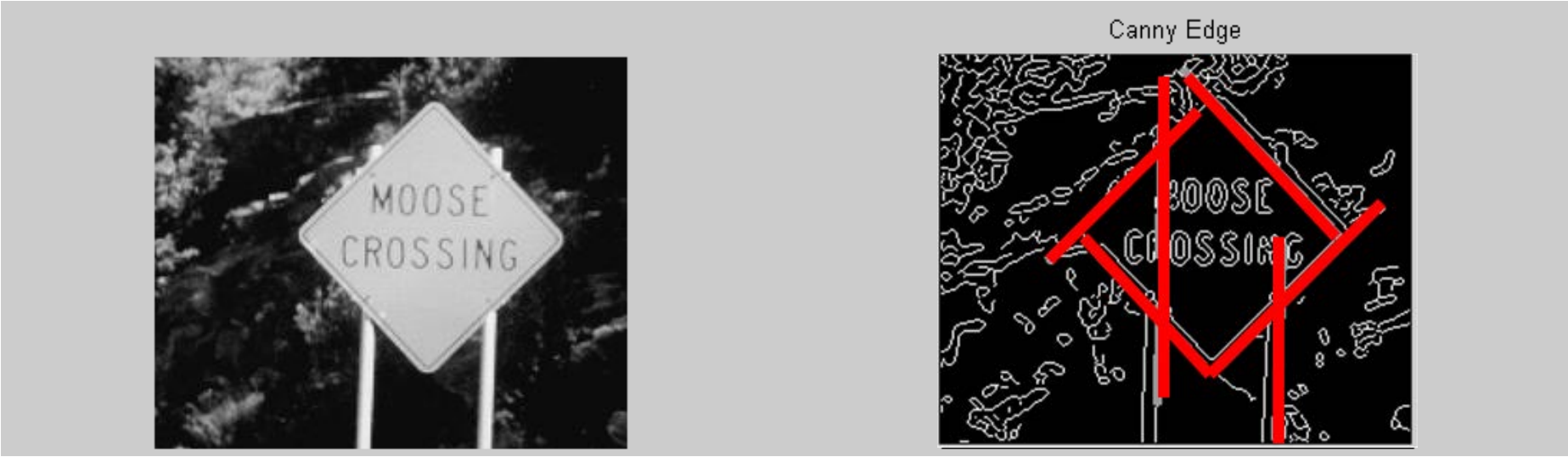
Semi-Automated Techniques:

- Dynamic programming
- Deformable Models

Fully Automated Techniques:

- Hough transform
- Graph Based Approaches

FINDING LINES



Input:

- Canny edge points.
- Gradient magnitude and orientation.

Output:

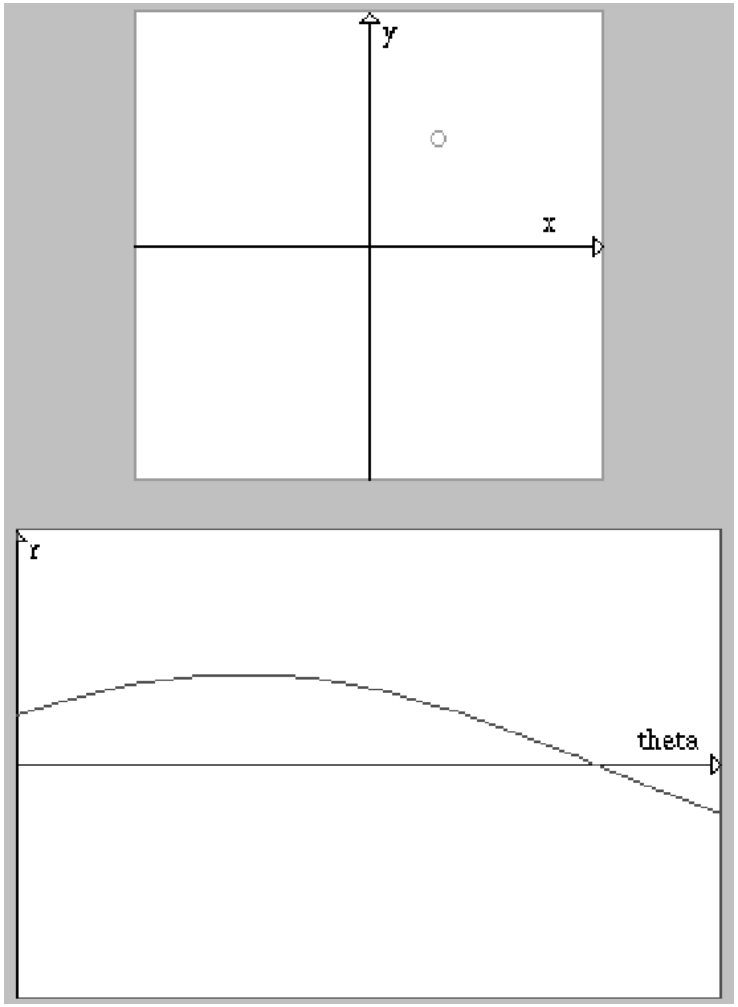
- All straight lines in image.

HOUGH TRANSFORM

Given a parametric model of a curve:

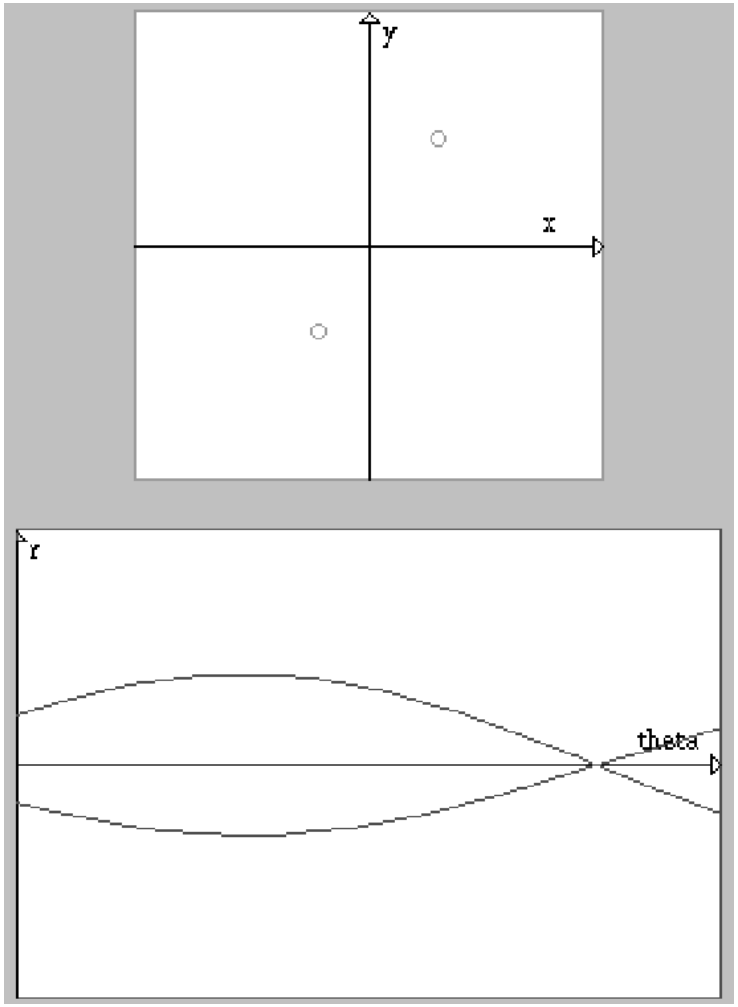
- Map each contour point onto the set of parameter values for which the curves passes through it.
- Find the intersection for all parameter sets thus mapped.

VOTING SCHEME

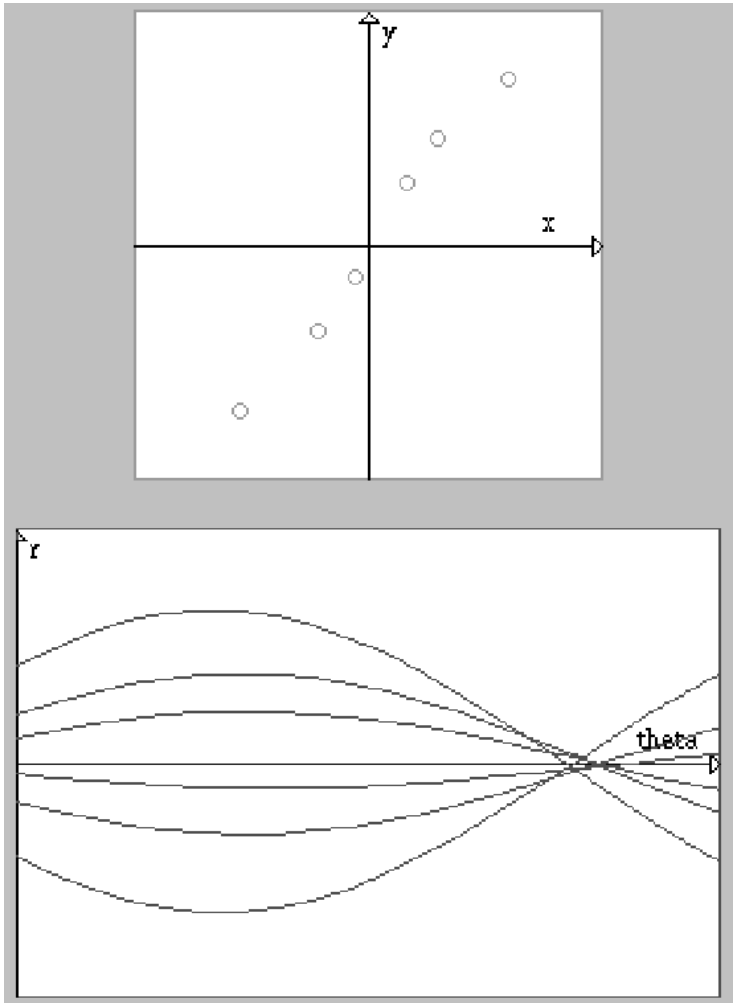


$$x\cos(\theta) + y\sin(\theta) = r, 0 \leq \theta < \pi$$

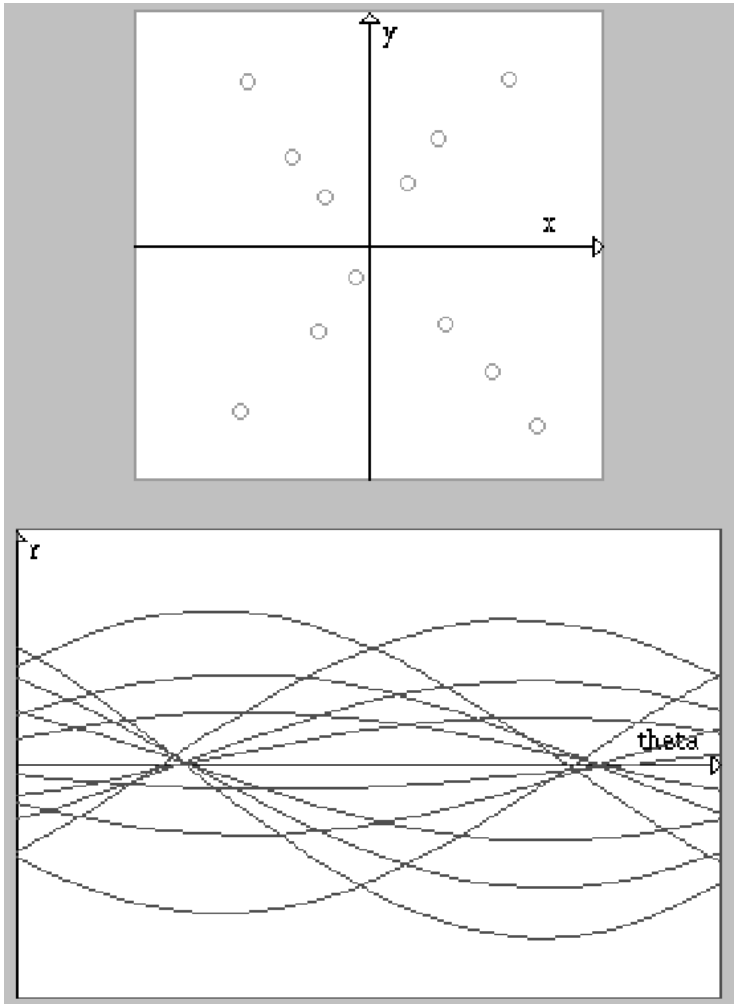
VOTING SCHEME



VOTING SCHEME

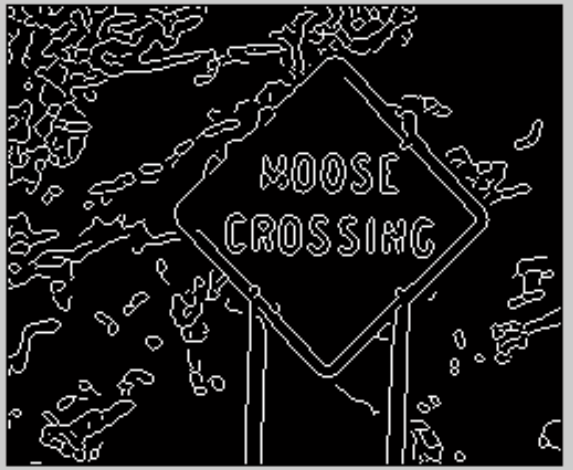


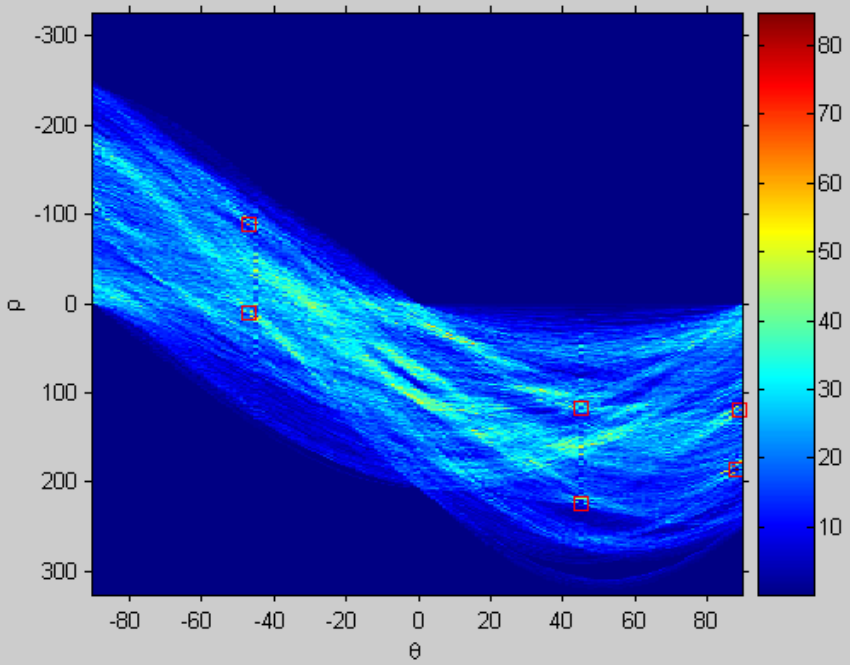
VOTING SCHEME



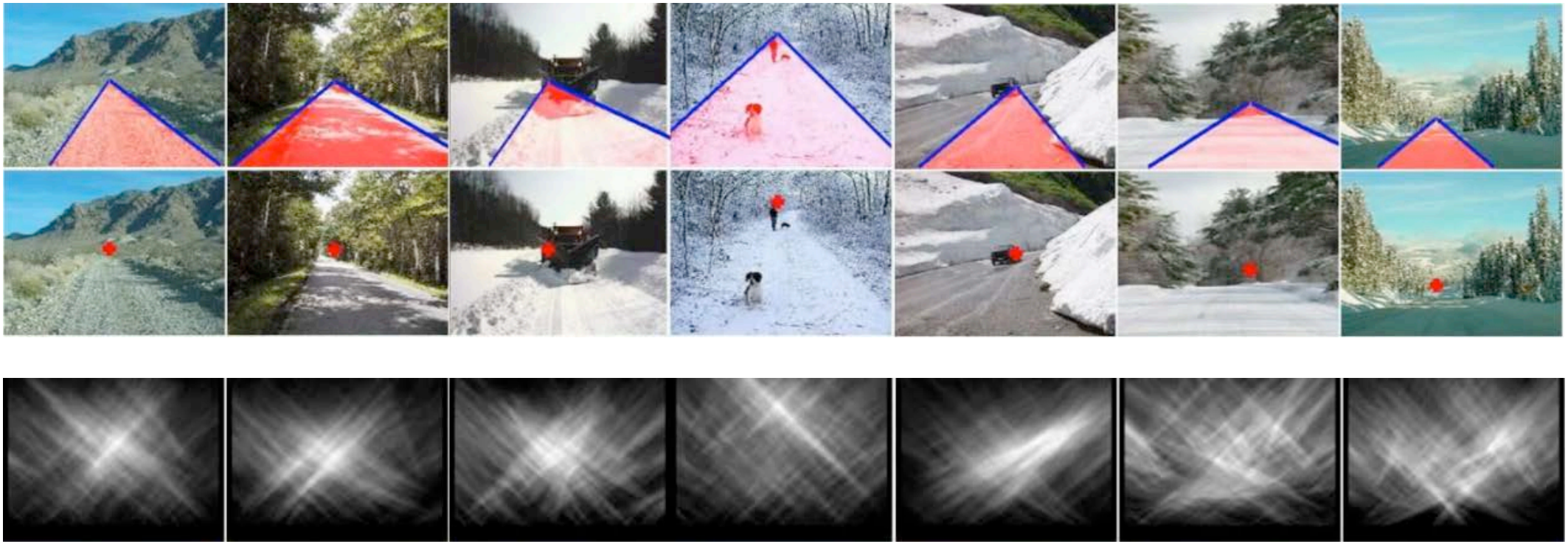
REAL WORLD LINES

Canny Edge





ROAD EDGES

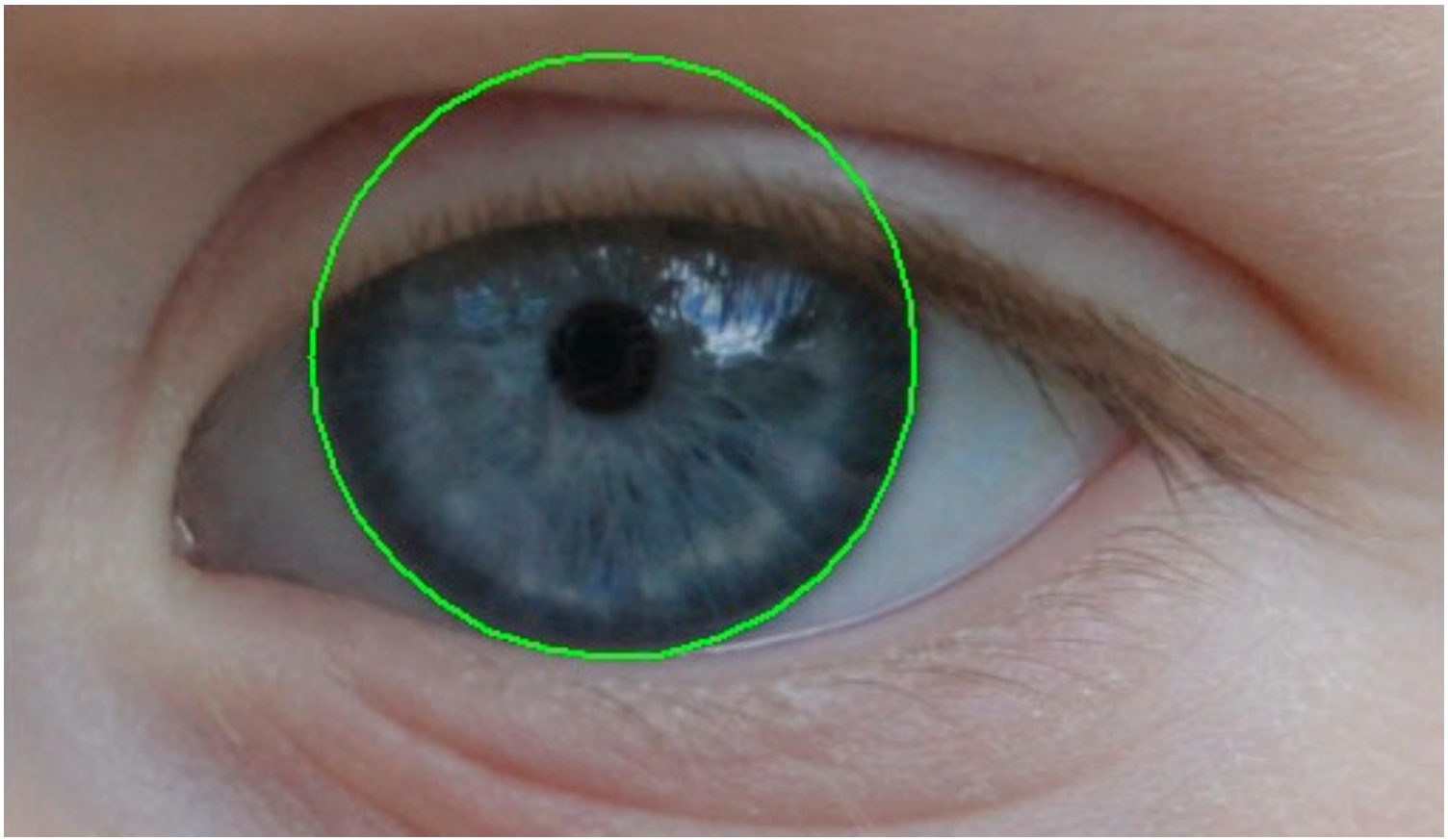


Kong et al. CVPR'09

GENERIC ALGORITHM

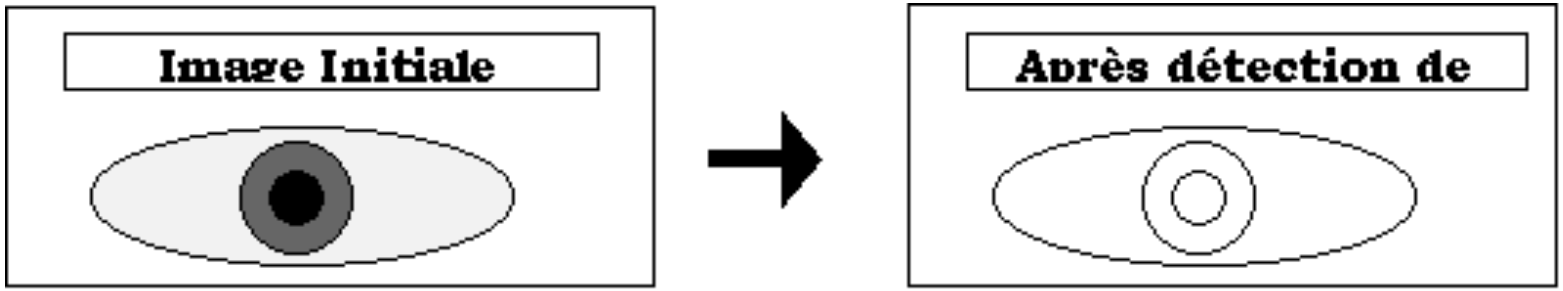
- Quantize parameter space (1 dimension per parameter)
- Form an accumulator array
- For each point in the gradient image such that the gradient strength exceeds a threshold, increment appropriate element of the accumulator
- Find local maxima in the accumulator

IRIS DETECTION



OCCLUSIONS

In theory:



In practice:

CIRCLE DETECTION

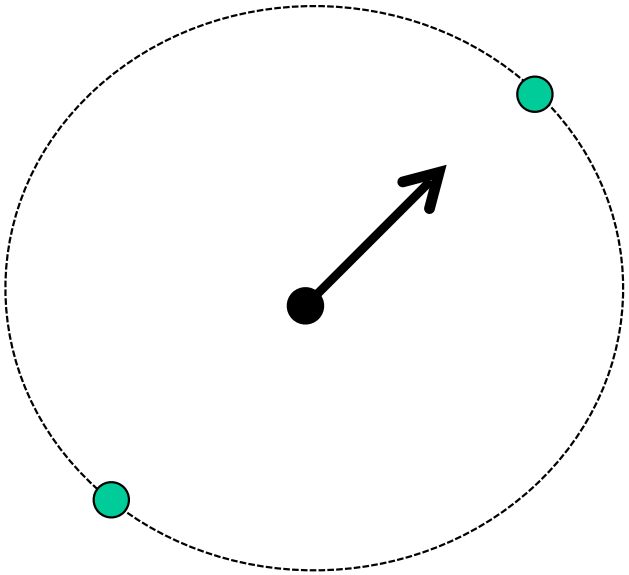
Circle of equation:

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

Therefore:

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

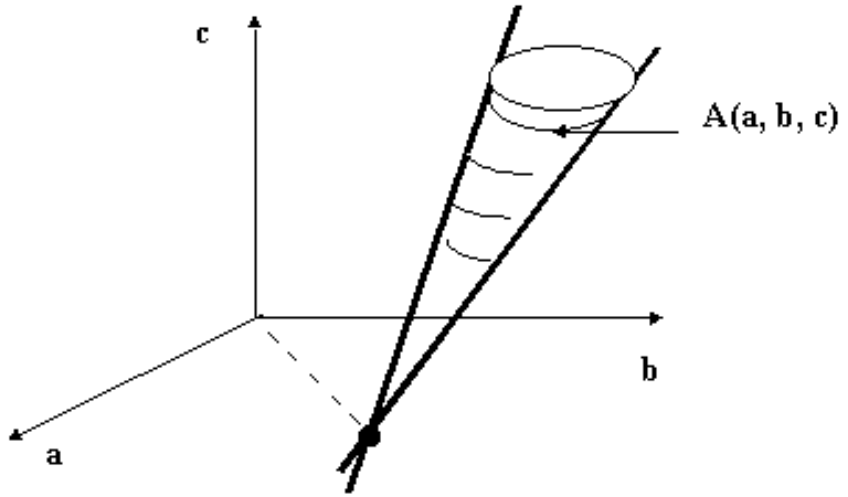
GRADIENT ORIENTATION



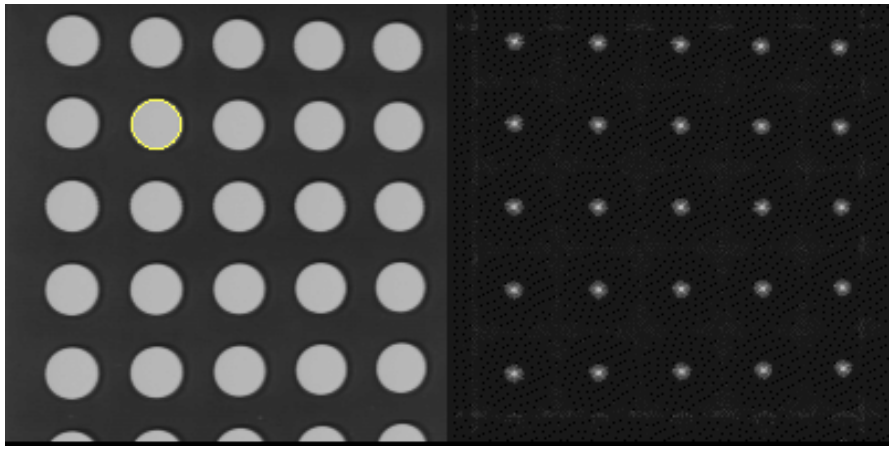
Can vote either along the entire circle or only at two points per value of the radius.

SIMPLE IMAGE

Voting scheme:

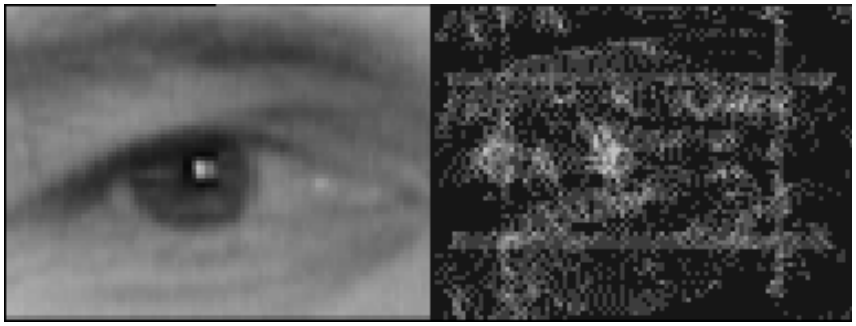


Result:

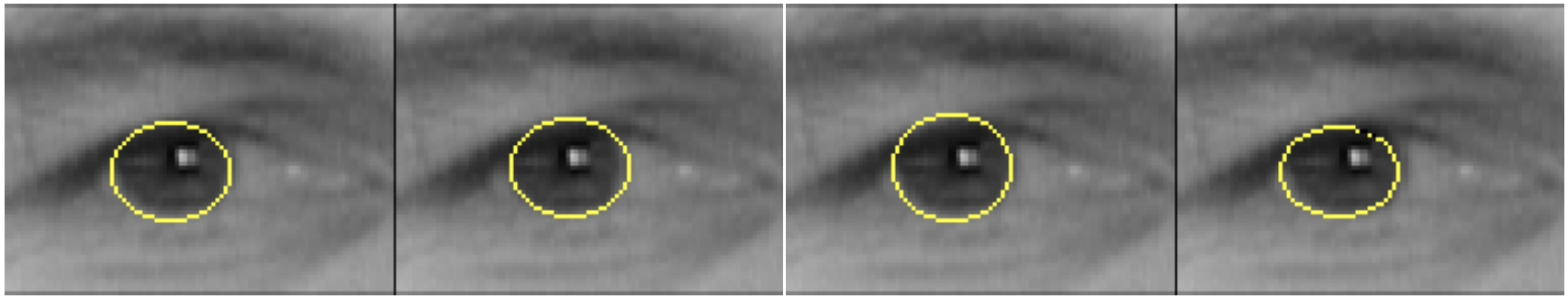


EYE IMAGES

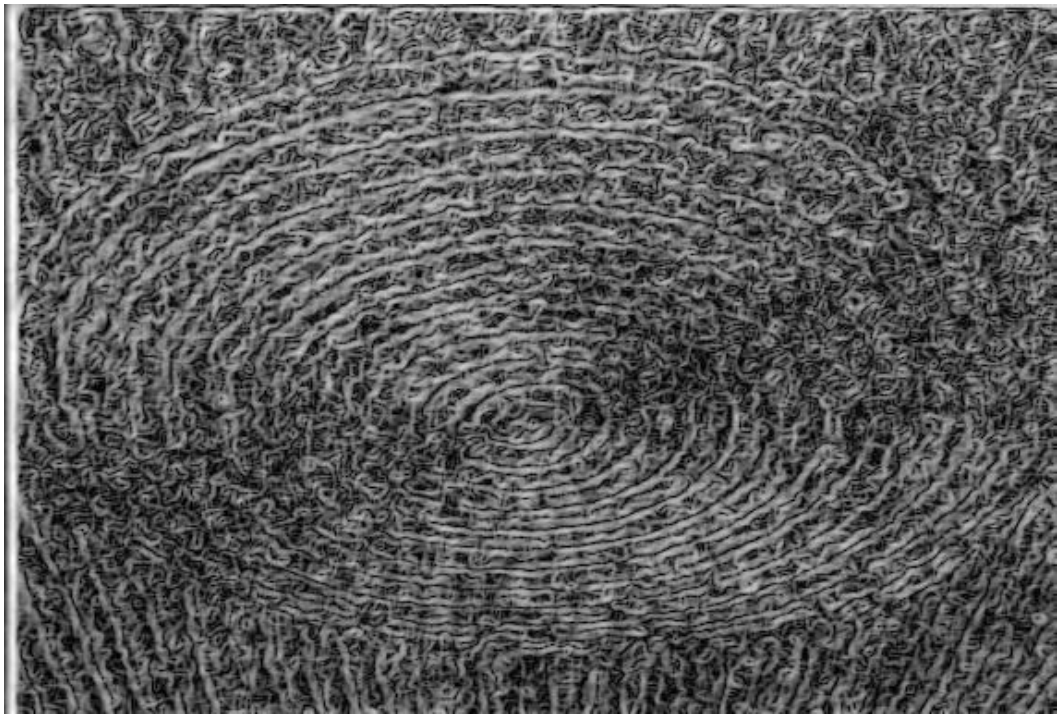
Image and accumulator:



Best four candidates:



ELLIPSES



ELLIPSE DETECTION

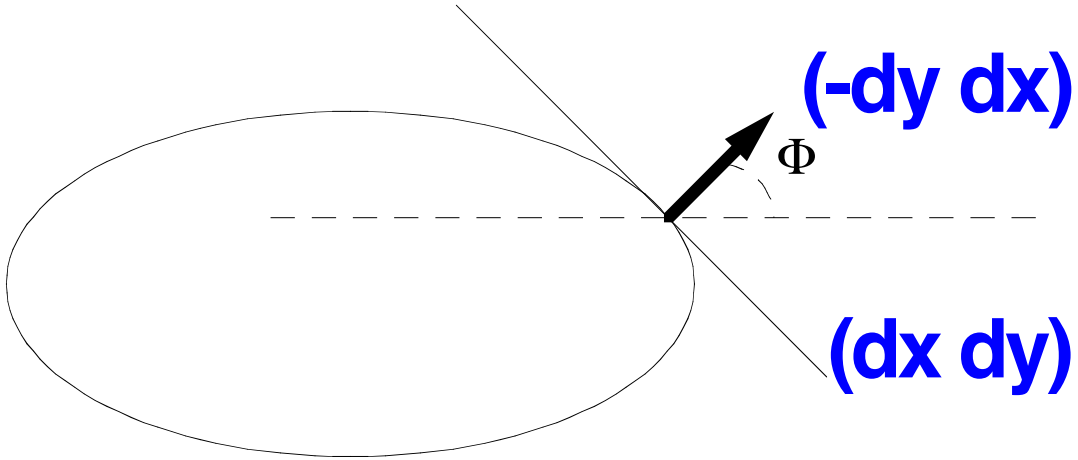
Ellipse of equation:

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

Therefore:

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

GRADIENT ORIENTATION



For each ellipse point:

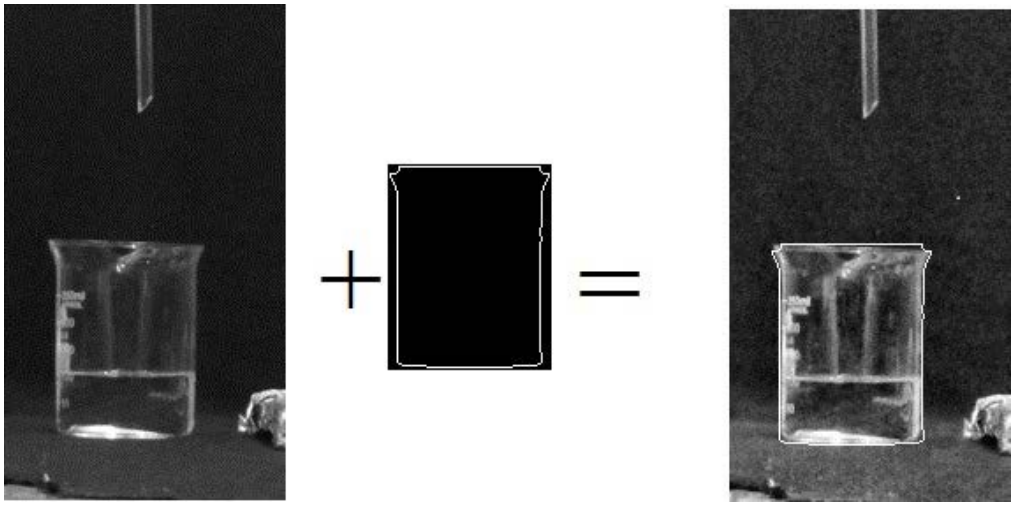
$$\frac{dy}{dx} = -\frac{b}{a} \cdot \frac{\cos(\theta)}{\sin(\theta)}$$

$$-\frac{dx}{dy} = \frac{a}{b} \cdot \tan(\theta)$$

The accumulator need only be incremented for:

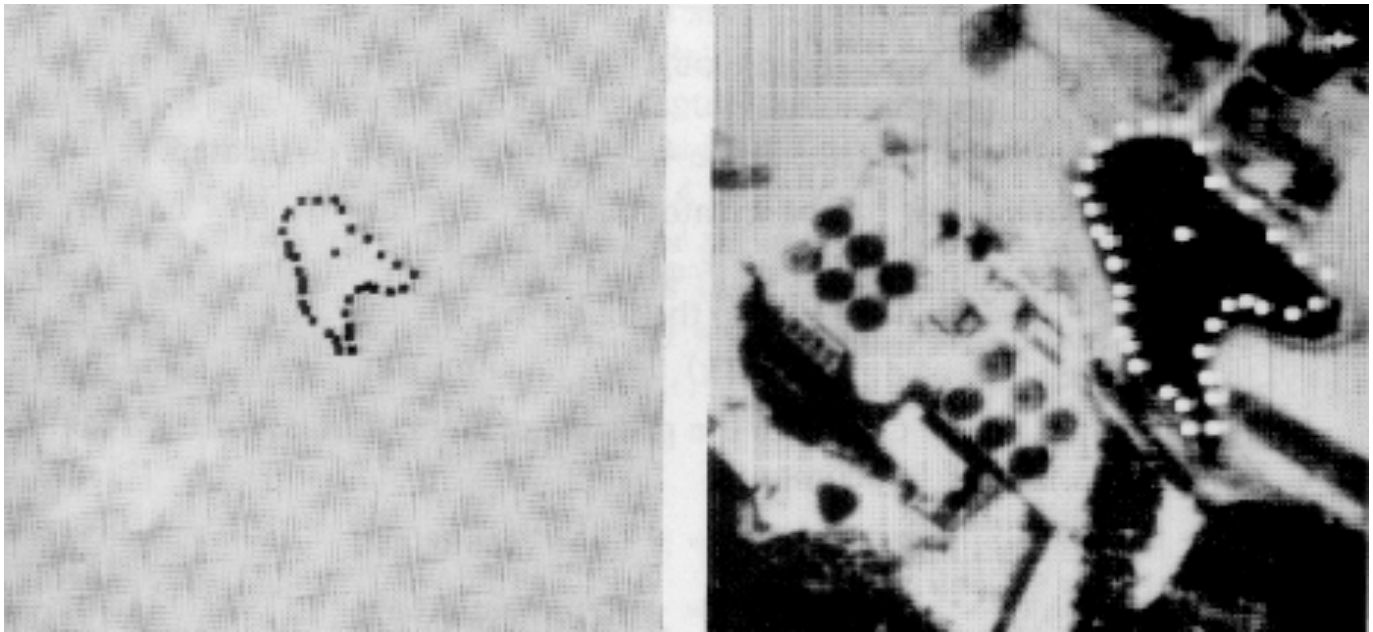
$$\theta = \operatorname{atan}\left(\frac{b}{a}\tan(\Phi)\right)$$

GENERALIZED HOUGH

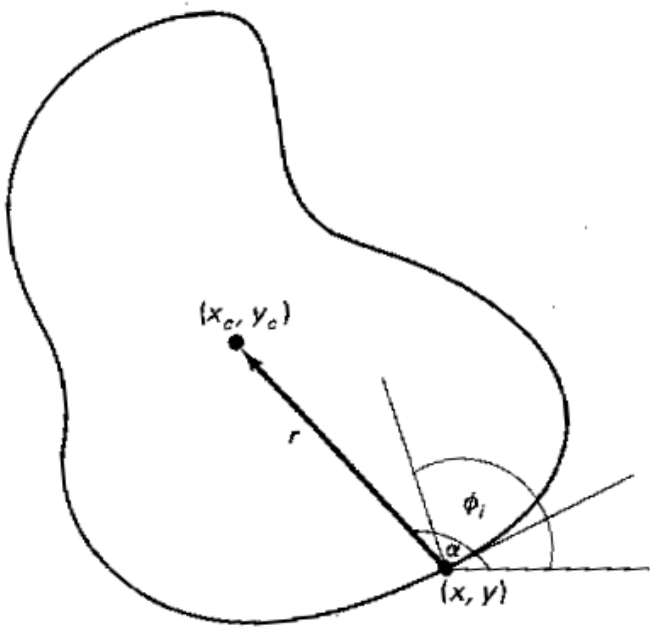


Finding a becher …

… or a lake.



R-TABLE



Angle measured from figure boundary to reference point	*Set of radii* $\{\mathbf{r}^k\}$ *where* $\mathbf{r} = (r, \alpha)$
ϕ_1	$\mathbf{r}_1^1, \mathbf{r}_2^1, \ldots, \mathbf{r}_{n_1}^1$
ϕ_2	$\mathbf{r}_1^2, \mathbf{r}_2^2, \ldots, \mathbf{r}_{n_2}^2$
.	.
.	.
.	.
ϕ_m	$\mathbf{r}_1^m, \mathbf{r}_2^m, \ldots, \mathbf{r}_{n_m}^m$

Code the shape in the form of an R-Table: Set of possible positions of a reference point given boundary orientation.

--> 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.

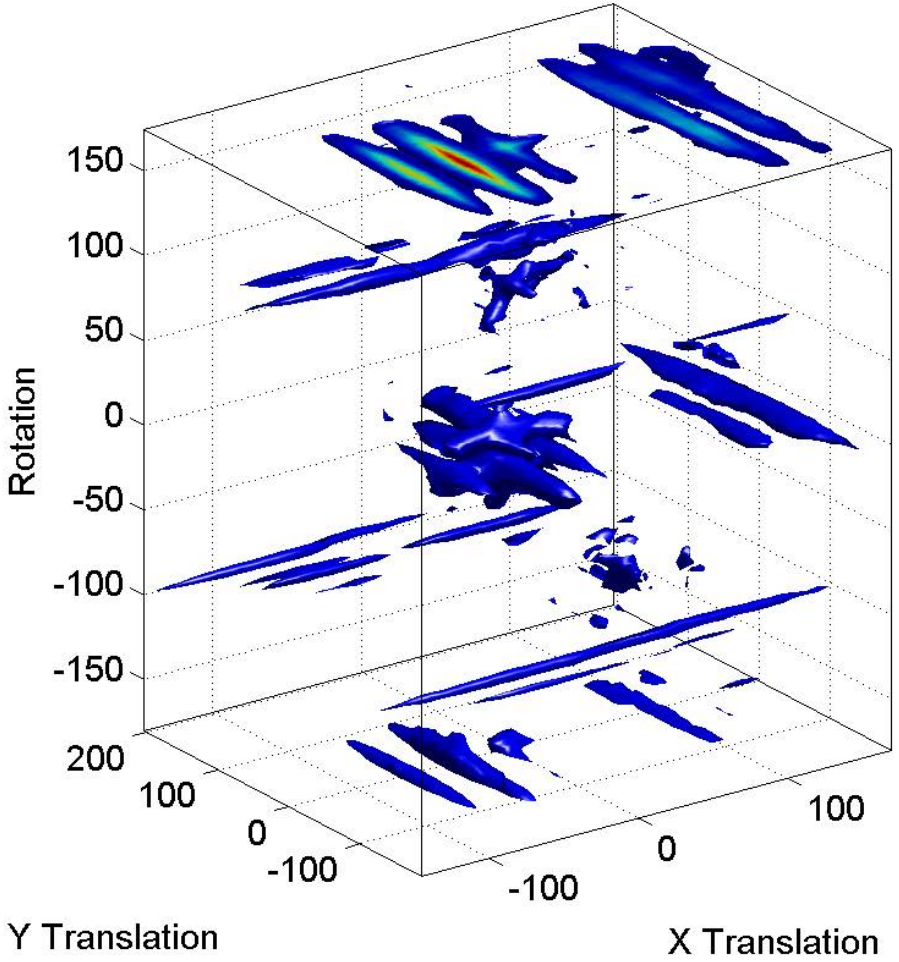
3. For each edge point,

   - Compute the possible centers, that is, for each table entry, compute
   $$x = x_e + r_\phi \cos(\theta(\phi))$$
   $$y = y_e + r_\phi \sin(\theta(\phi))$$

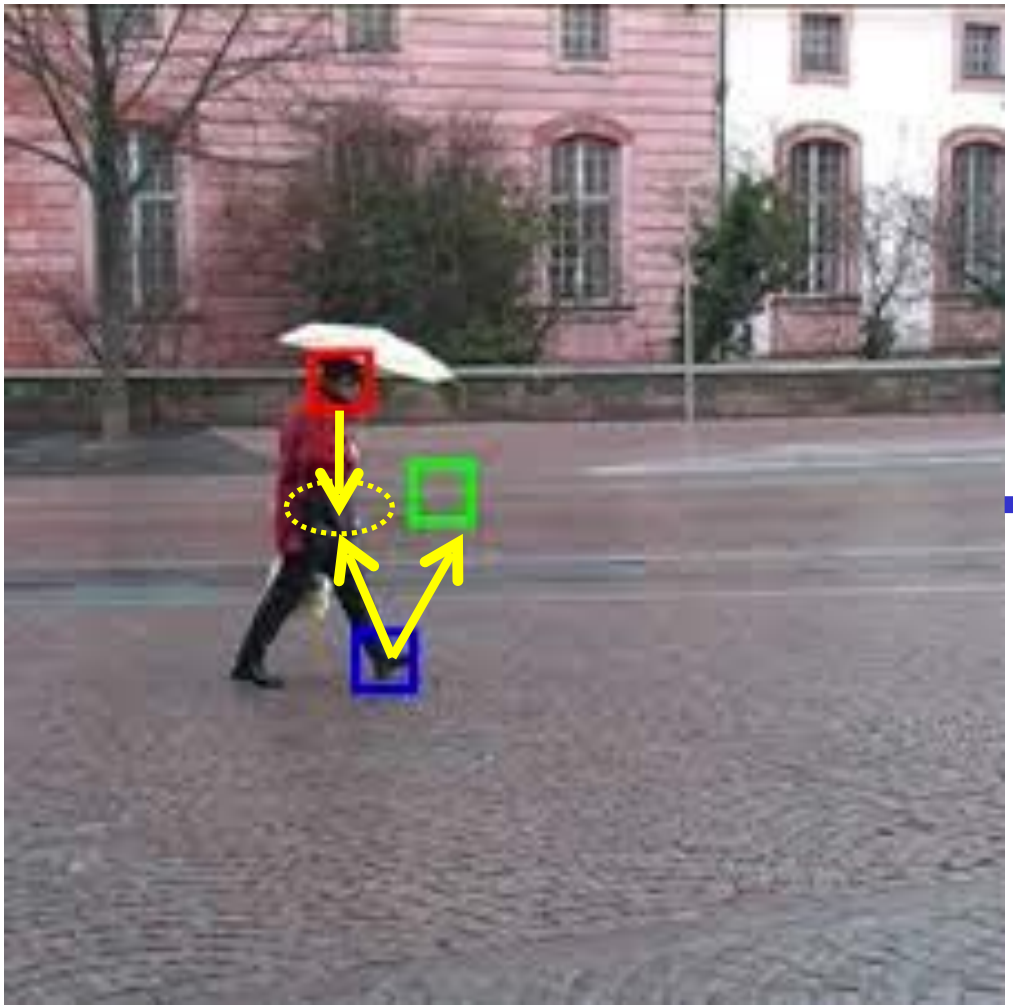
   - Increment the accumulator array

REAL-TIME HOUGH

ACCUMULATOR



HOUGH FORESTS



Gall & al. , PAMI'11

OCCLUSION HANDLING

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:

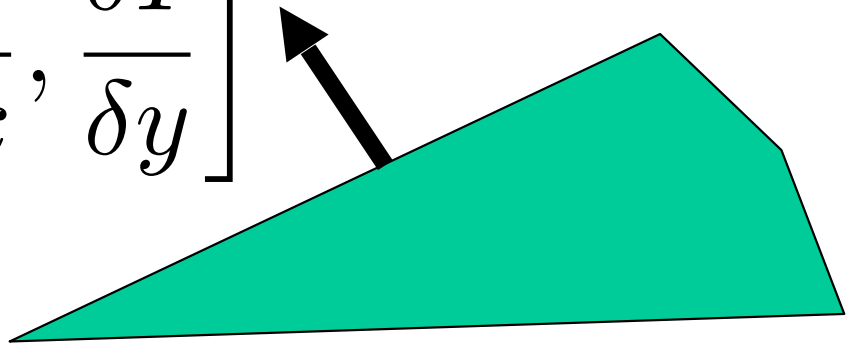
- Dynamic programming
- Deformable Models

Fully Automated Techniques:

- Hough transform
- Graph Based Approaches

MAGNITUDE AND ORIENTATION

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

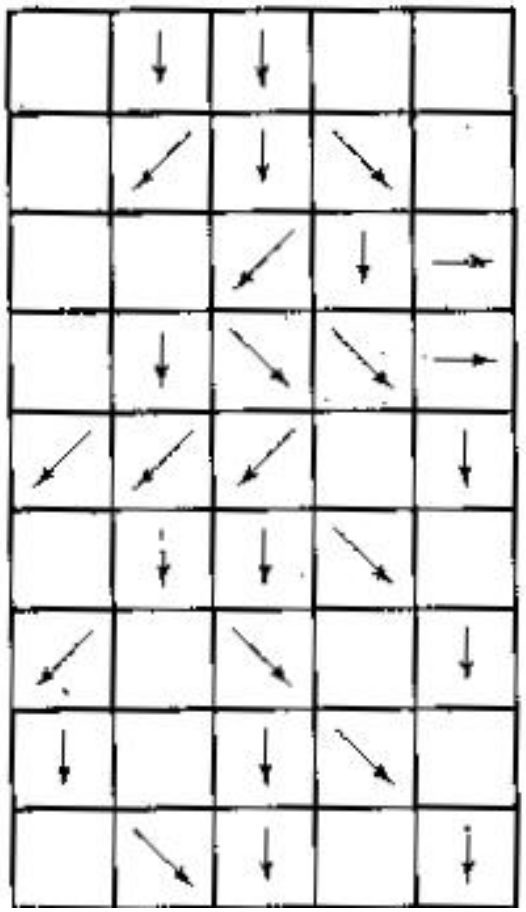


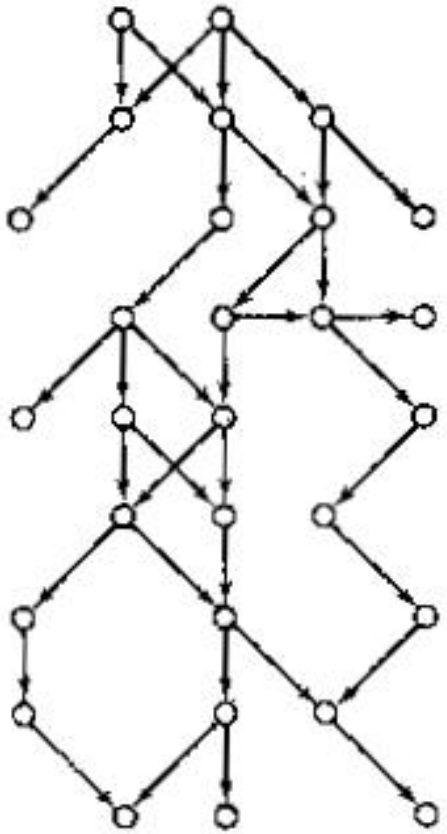
$$\text{Contrast: } G = \sqrt{\frac{\delta I}{\delta x}^2 + \frac{\delta I}{\delta y}^2}$$

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

MINIMUM SPANNING TREE

Image modeled as a graph:





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

DELINEATION 1998

Detect road points

Find generic paths

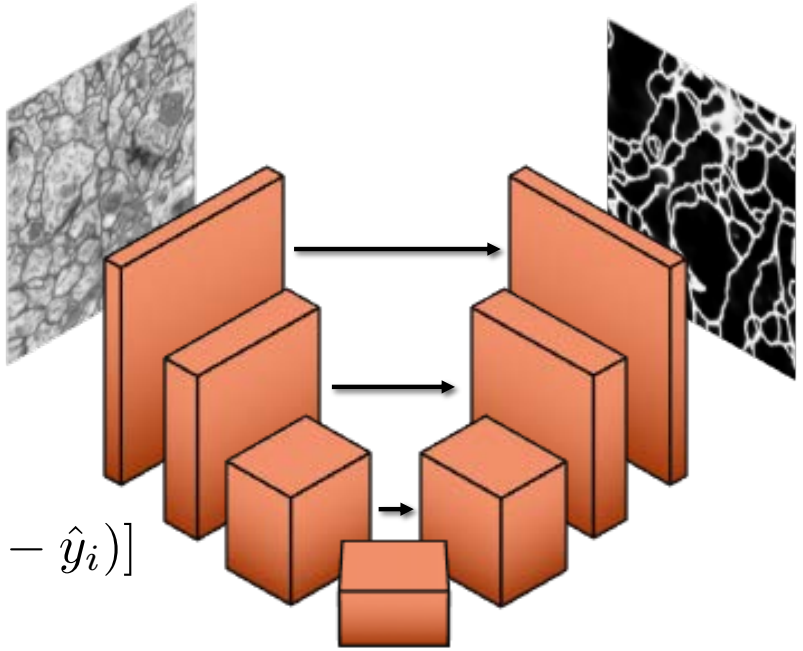
Apply semantic filter

Find road widths

Fischler & Heller, 1998.

DELINEATION 2018

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



Minimize

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

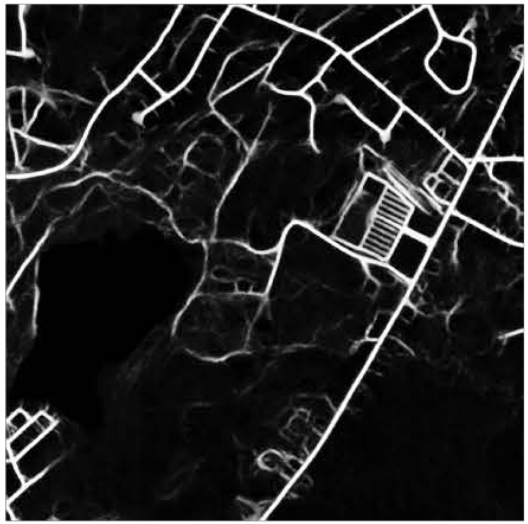
where

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

Mosinska et al, 2018.

NETWORK OUTPUT

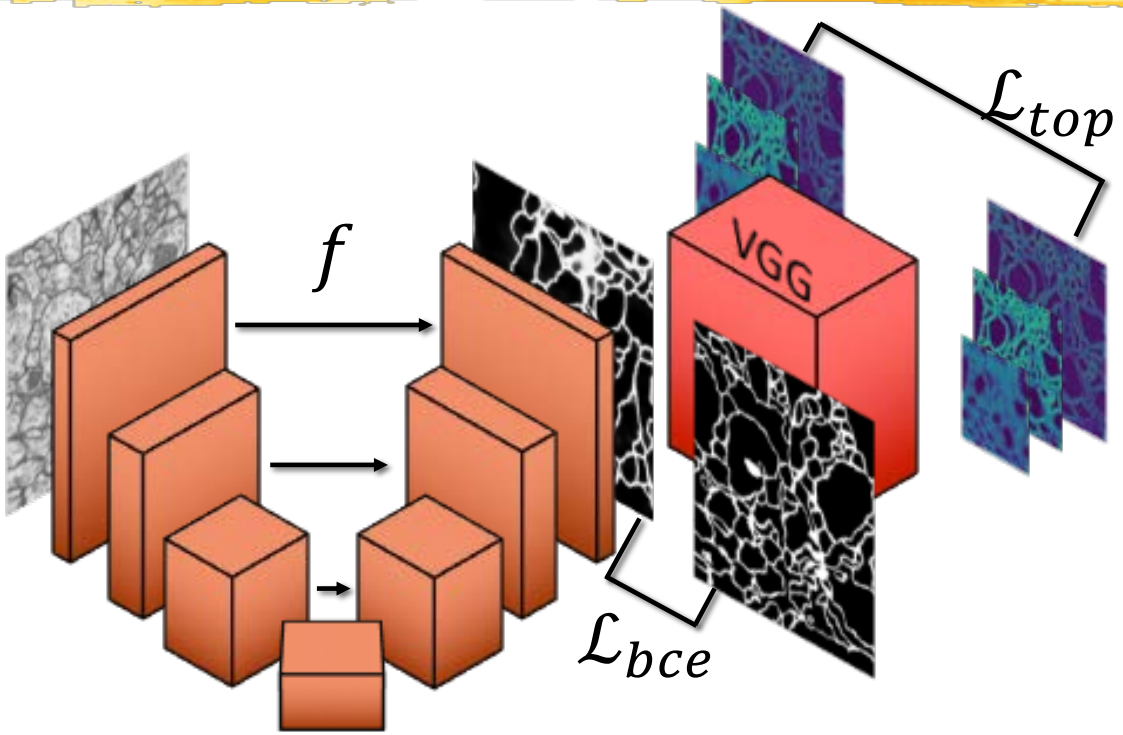
Image



BCE Loss

Ground truth

ACCOUNTING FOR TOPOLOGY



Minimize

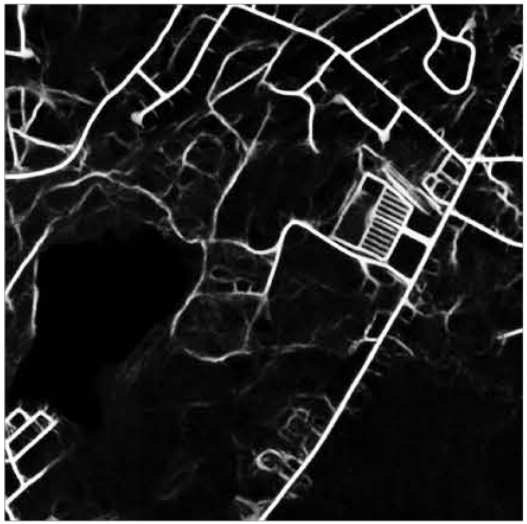
$$L(\mathbf{x}, \mathbf{y}; \mathbf{w}) = L_{bce}(\mathbf{x}, \mathbf{y}; \mathbf{w}) + L_{top}(\mathbf{x}, \mathbf{y}; \mathbf{w}) \ ,$$

$$L_{top}(\mathbf{x}, \mathbf{y}; \mathbf{w}) \propto \sum_{n=1}^{N} \sum_{m=1}^{M_n} \| l_n^m(\mathbf{y}) - l_n^m(f(\mathbf{x}, \mathbf{w})) \|_2^2 \ ,$$

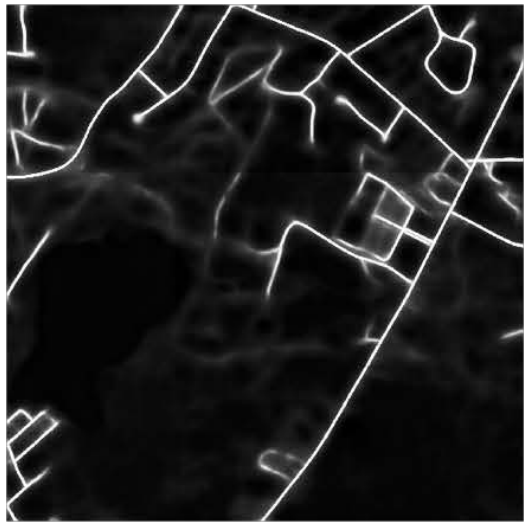
L_{top}: Sum of square differences between feature maps in a pretrained VGG.

ACCOUNTING FOR TOPOLOGY

Image



BCE Loss



BCE andTopo Loss

Ground truth

ITERATIVE REFINEMENT

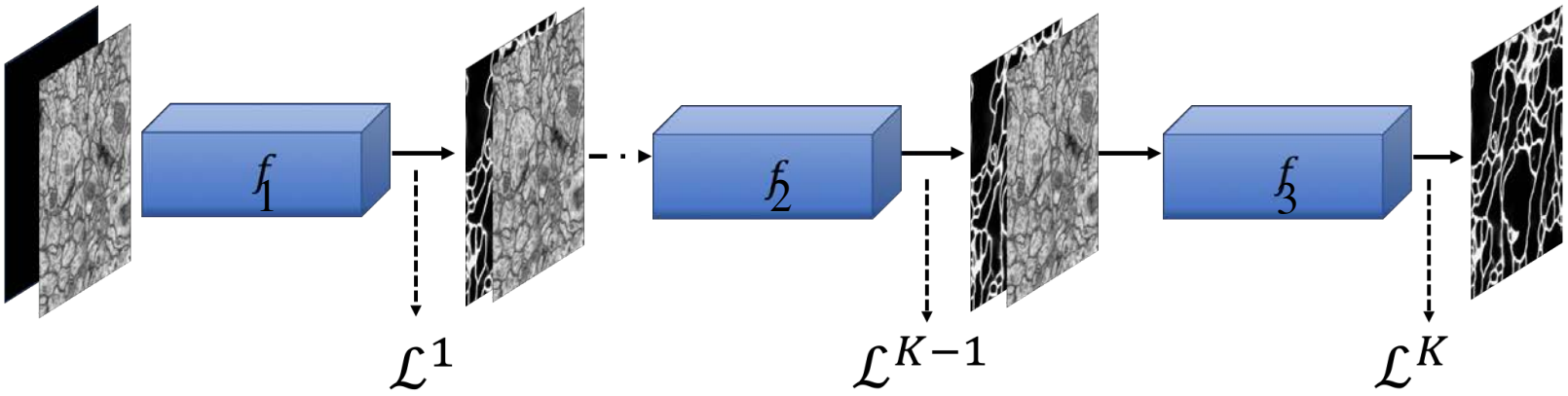
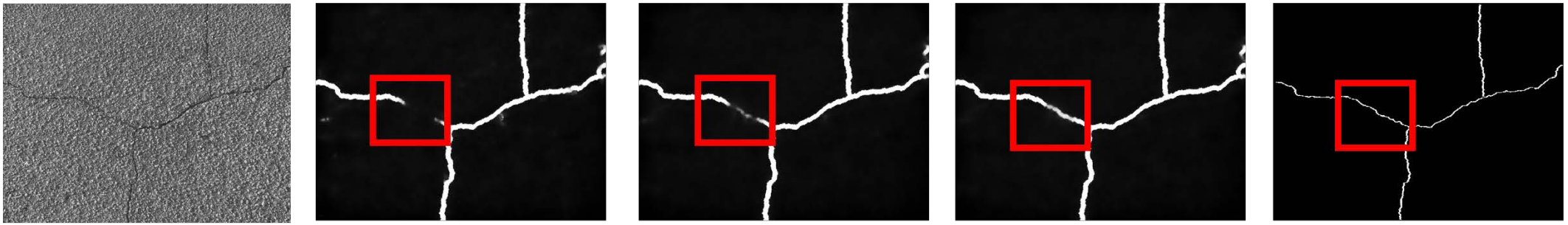


Image | Iter 1 | Iter 2 | Iter 3 | Ground truth



Use the same network to progressively refine the results keeping the number of parameters constant

DELINEATION STEPS

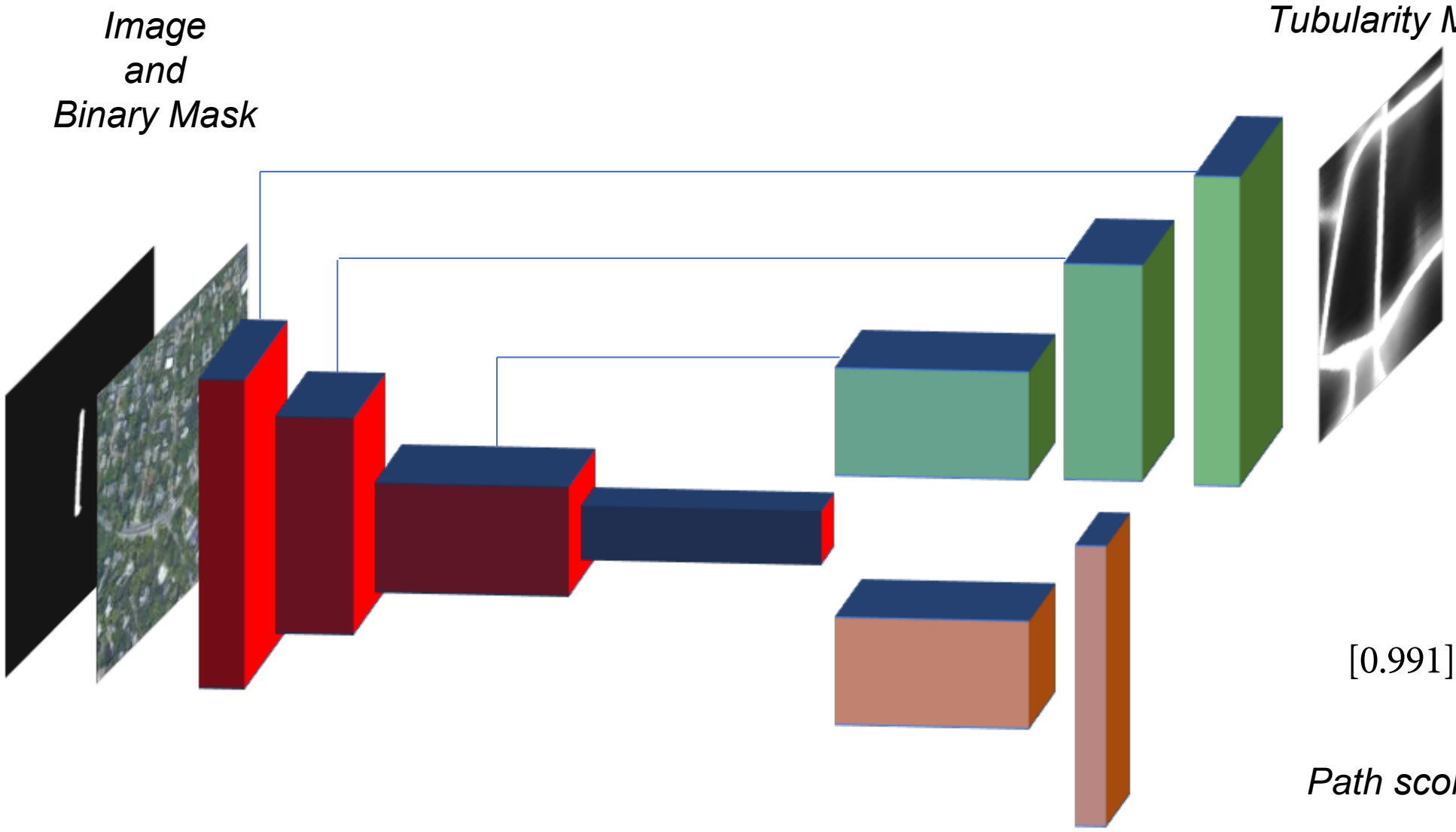
1.Compute a probability map.

2. Sample and connect the samples.

3. Assign a weight to the paths.

4. Retain the best paths.

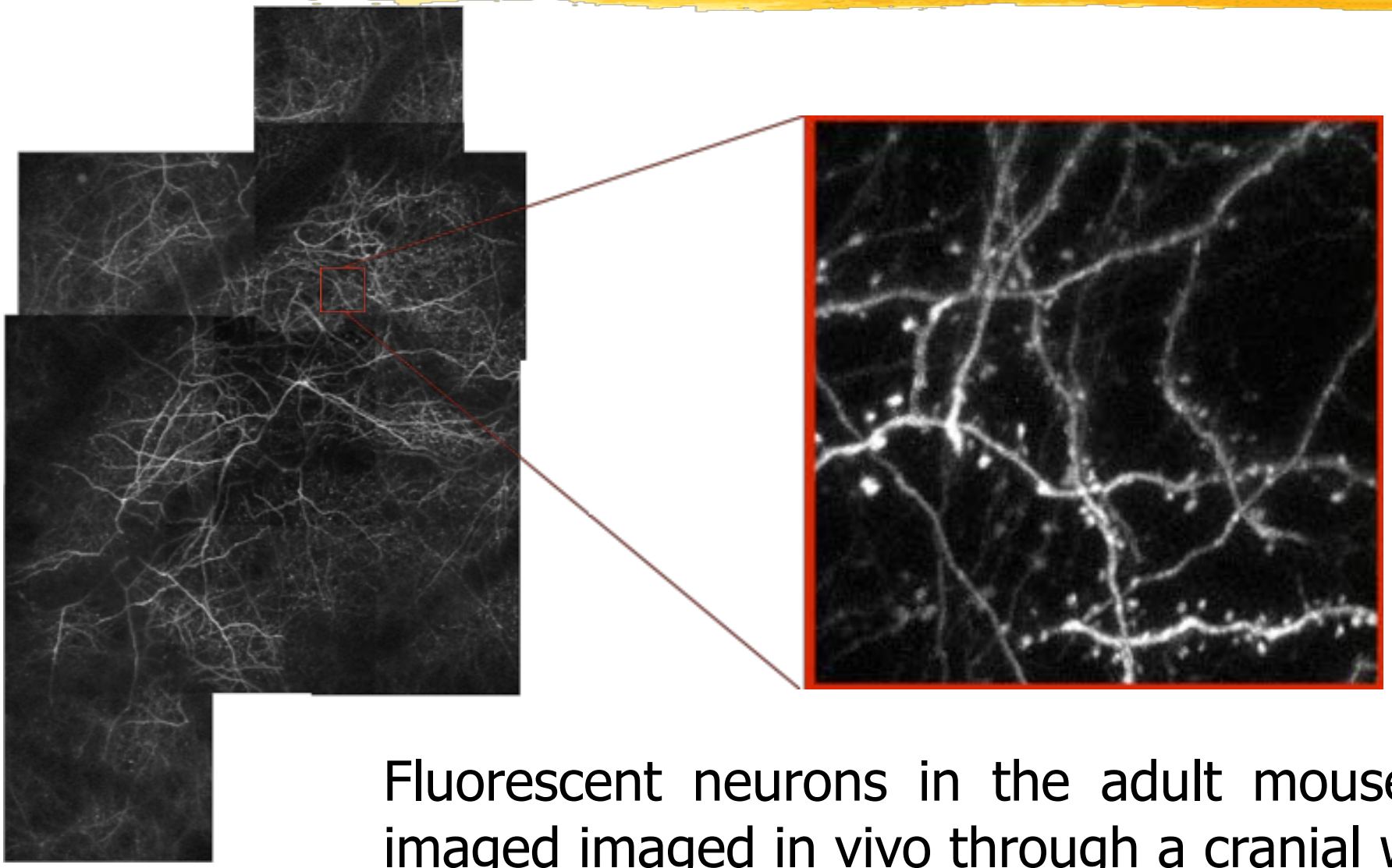
DUAL USE U-NET



STREETS OF TORONTO

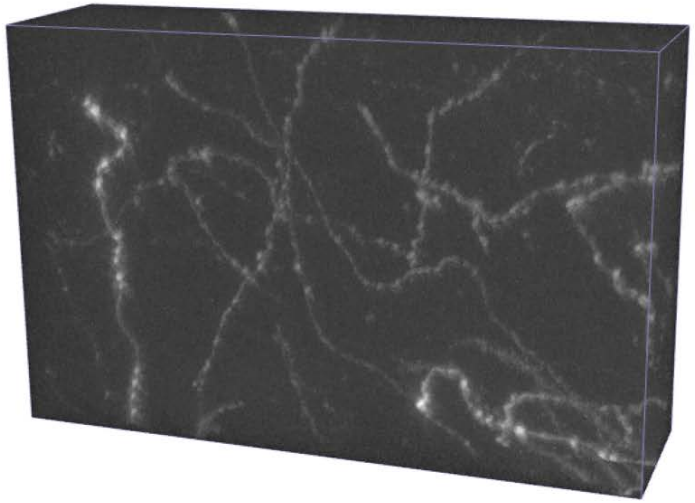
— False negatives
— False positives

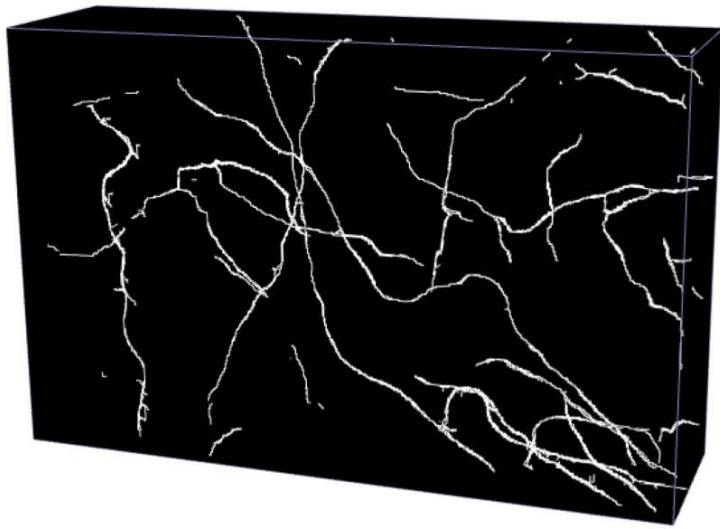
DENDRITES AND AXONS

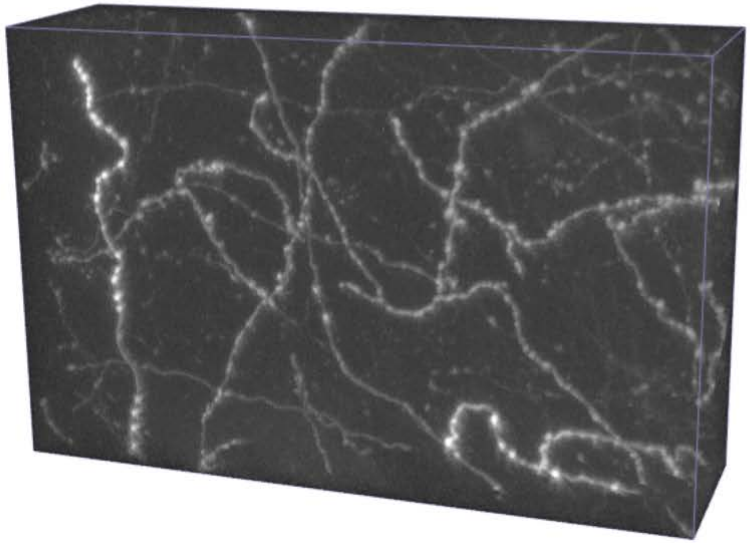


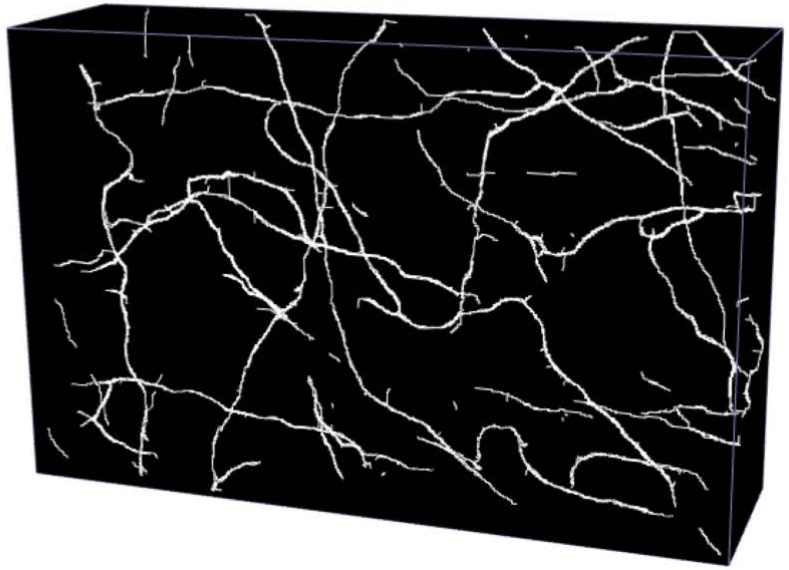
Fluorescent neurons in the adult mouse brain imaged imaged in vivo through a cranial window using a 2-photon microscope.

DENDRITES AND AXONS

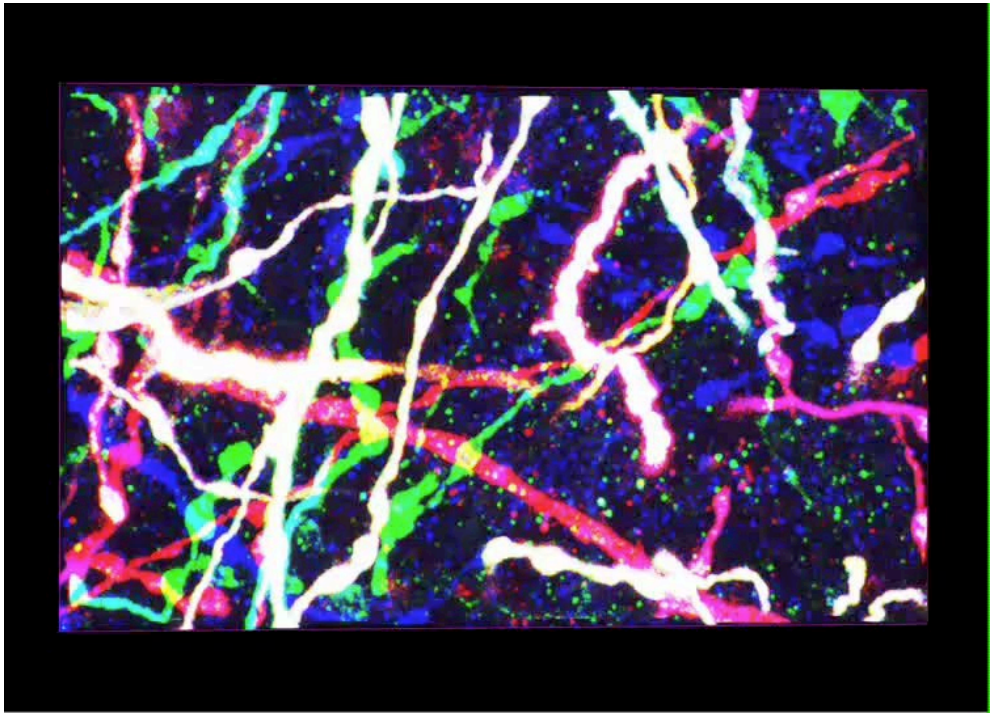


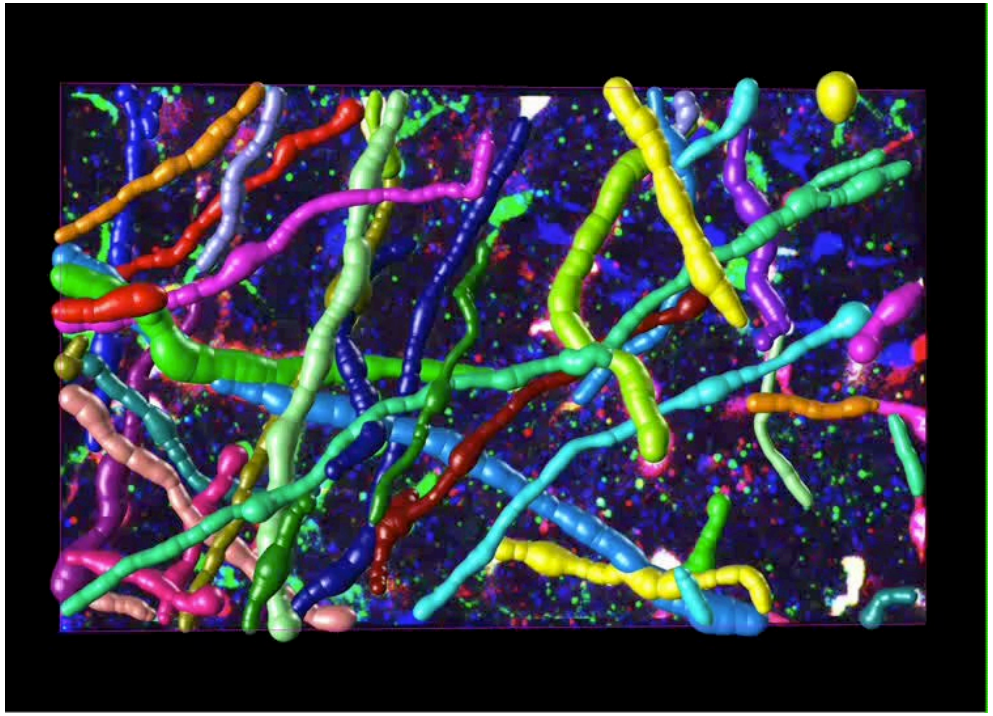




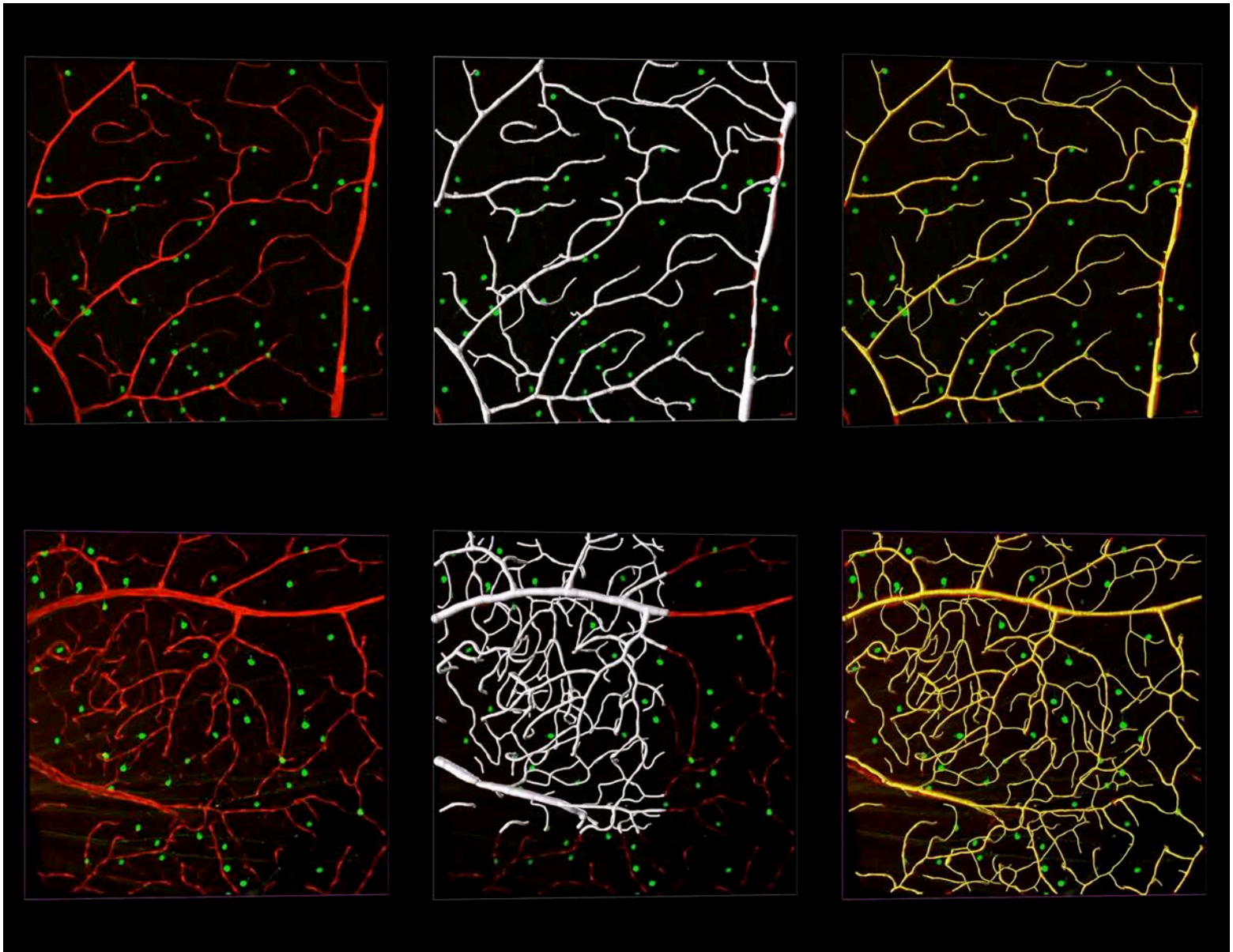


BRAINBOW IMAGES





BLOOD VESSELS



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.