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!
Mapping and Overlays

Connectivity matters!
Connectomics

—> Topology needed

Courtesy of C. Petersen
Analogy

- Uses Deep Nets to find the most promising locations to focus on.
- Performs tree-based search when possible.
- Relies on reinforcement learning and other ML techniques to train.

Low level processing

High level processing
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 $\sigma$!
Interactive Delineation

- The user provides the start and end points (red x).
- The algorithm does the rest (yellow line).
Live Wire in Action
1D Dynamic Programming

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

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

where \( \phi \) denotes the gradient orientation.
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
\[ h(x_1, x_2, \ldots, x_n) = r(s, x_1) + \sum_{i=1}^{n-1} r(x_i, x_{i+1}) + r(x_n, g) \]
define
\[ f_1(x_2) = \min_{x_1} (r(s, x_1) + r(x_1, x_2)) \]
\[ f_2(x_3) = \min_{x_2} (r(x_2, x_3) + f_1(x_2)) \]
\[ \vdots \]
\[ f_{n-1}(x_n) = \min_{x_{n-1}} (r(x_{n-1}, x_n) + f_{n-2}(x_{n-1})) \]
\[ \Rightarrow \min h(x_1, x_2, \ldots, x_n) = \min_{x_n} (r(x_n, g) + f_{n-1}(x_n)) \]
2D Dynamic Programming

Notations:

- \( s \): Start point
- \( L \): List of active nodes
- \( c(u,v) \): Local costs for link \( u \to v \)
- \( d(v) \): Total cost from \( s \) to \( v \)
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:
\[ d(s) \leftarrow 0 \text{ and } d(u) \leftarrow \infty \text{ for } u \neq s \]
\[ T = \emptyset \]
\[ v = s \]

Loop until goal is reached:
\[ T \leftarrow T \cup \{v\} \]
for all \( u \rightarrow v \) edges such that \( u \notin T \)
\[ \text{if } d(v) + c(v,u) < d(u) \]
\[ d(u) \leftarrow d(v) + c(v,u) \]
end
end
\[ v = \text{argmin}_{w \notin T} d(w) \]

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:

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

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

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

Average gradient \quad \text{Average sum of squared 2nd derivatives}

\approx \quad \text{Average sum of square curvature}
Matrix Notation

\[ E = E_G + \frac{1}{2}X^t K X + \frac{1}{2}Y^t K Y \]

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

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

\[ K = \begin{bmatrix}
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
\ldots & 1 & -4 & 6 & -4 & 1 & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
\ldots & \ldots & 1 & -4 & 6 & -4 & 1 & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
\ldots & \ldots & \ldots & 1 & -4 & 6 & -4 & 1 & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
\ldots & \ldots & \ldots & \ldots & 1 & -4 & 6 & -4 & 1 & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
\ldots & \ldots & \ldots & \ldots & \ldots & 1 & -4 & 6 & -4 & 1 & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots & 1 & -4 & 6 & -4 & 1 & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots
\end{bmatrix} \]
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:

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

\[
0 = \frac{\partial E}{\partial Y} + \alpha \frac{dY}{dt} = \frac{\partial E_G}{\partial Y} + KY + \alpha \frac{dY}{dt}
\]
Iterating

At every step:

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

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

→ Solve two linear equations at each iteration.
Derivatives of the Image Gradient

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

\[
\frac{\partial E_G}{\partial X} = \begin{bmatrix}
\frac{\partial E_G}{\partial x_1} & \ldots & \frac{\partial E_G}{\partial x_N}
\end{bmatrix},
\frac{\partial E_G}{\partial Y} = \begin{bmatrix}
\frac{\partial E_G}{\partial y_1} & \ldots & \frac{\partial E_G}{\partial y_N}
\end{bmatrix}
\]

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

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

\[\rightarrow\] We need to interpolate.
Bilinear Interpolation

\[ G(x, y+1) \quad G(x+1, y+1) \]

\[ y_i \]

\[ G(x, y) \quad x_i \quad G(x+1, y) \]

\[ p = x_i - x \]

\[ q = y_i - y \]

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

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

\[ \frac{\partial G}{\partial y_i} = (1 - p)(G(x, y + 1) - G(x, y)) + p(G(x + 1, y + 1) - G(x + 1, y)) \]
Open and Closed Snakes
Cysts Tumors in Ultrasound Images

Drawn by the physician.
Refined by the Computer.

Luo et al., 2017
Network Snakes

--> Updated field boundaries.
Ribbon Snakes

\[ E = E_G + \frac{1}{2}X^t K X + \frac{1}{2}Y^t K Y + \frac{1}{2}W^t K_W W \]

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

\[ K_W = \begin{bmatrix}
\vdots & \vdots & \vdots & \vdots & -1 & 2 & -1 & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & -1 & 2 & -1 & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & -1 & 2 & -1 & \vdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & -1 & 2 & -1 & \vdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & -1 & 2 & -1 & \vdots \\
\end{bmatrix} \]
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
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

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

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

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

→ Solve three linear equations at each iteration.
Constrained Optimization

- Minimize $F(S)$ subject to $C(S) = 0$
Site Modeling (1996)
Implicit vs Explicit

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

\[ z = \Phi(x, y), \]
\[ z > 0 \text{ outside,} \]
\[ z < 0 \text{ inside,} \]

\[ \rightarrow \text{Consider the curve as the zero level set of a surface.} \]
Topology Changes are Possible

\[ z = \Phi(x, y, t_1) \]

\[ z = \Phi(x, y, t_2) \]
Consider the curve as the zero level set of the surface:

\[ z = \Phi(x, y, t) \]

Evolution equation:

\[ 0 = \Phi_t + \beta(\kappa) |\nabla \Phi| \]

where \( \kappa = \frac{\Phi_{xx}\Phi_y^2 - 2\Phi_{xy}\Phi_x\Phi_y + \Phi_{yy}\Phi_x^2}{\Phi_x^2 + \Phi_y^2} \)

\( \beta(\kappa) \) 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

Evolution equation:  \[ 0 = \Phi_t + \beta(\kappa) |\nabla \Phi| \]

where:  \[ \beta(\kappa) = k_I (1 - \epsilon \kappa) \]
\[ k_I = \frac{1}{1 + \nabla I} \]

\[ \rightarrow \quad \text{Expansion stops at the boundaries.} \]
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.
\[ x \cos(\theta) + y \sin(\theta) = r \ , \ 0 \leq \theta \leq \pi \]
Once the contour points are associated to individual lines, you can perform least squares fitting.
Real Lines
Road Lines
Road Edges
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 of equation:

\[ x = x_0 + r \cos(\theta) \]
\[ y = y_0 + r \sin(\theta) \]

Therefore:

\[ x_0 = x - r \cos(\theta) \]
\[ y_0 = y - r \sin(\theta) \]
Gradient Orientation

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

Result:
Eye Image

Image and accumulator:

Best four candidates:
Ellipses
Ellipse of equation:

\[ x = x_0 + a \cos(\theta) \]
\[ y = y_0 + b \sin(\theta) \]

Therefore:

\[ x_0 = x - a \cos(\theta) \]
\[ y_0 = y - b \sin(\theta) \]
Gradient Orientation

For each ellipse point:

\[
\frac{dx}{d\theta} = -a \sin(\theta)
\]

\[
\frac{dy}{d\theta} = b \cos(\theta)
\]

\[
\phi = \text{atan}
\left(\frac{\frac{dx}{d\theta}}{\frac{dy}{d\theta}}\right)
\]

\[
= \text{atan}(a \sin(\theta), b \cos(\theta))
\]

\[
= \text{atan}\left(\frac{a}{b} \tan(\theta)\right)
\]

\[
\tan(\phi) = \frac{a}{b} \tan(\theta)
\]

\[
\Rightarrow \theta = \text{atan}(\frac{b}{a} \tan(\phi))
\]

The accumulator need only be incremented for this \(\theta\).
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 \(\phi\).

Ballard. , PR’81
### R-Table

<table>
<thead>
<tr>
<th>$\phi$</th>
<th>$R(\phi_i)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\phi_1$</td>
<td>$(r_{11}^1, \alpha_{11}^1), (r_{21}^1, \alpha_{21}^1), \ldots, (r_{n1}^1, \alpha_{n1}^1)$</td>
</tr>
<tr>
<td>$\phi_2$</td>
<td>$(r_{12}^2, \alpha_{12}^2), (r_{22}^2, \alpha_{22}^2), \ldots, (r_{n2}^2, \alpha_{n2}^2)$</td>
</tr>
<tr>
<td>..</td>
<td>..</td>
</tr>
<tr>
<td>$\phi_m$</td>
<td>$(r_{1m}^m, \alpha_{1m}^m), (r_{2m}^m, \alpha_{2m}^m), \ldots, (r_{im}^m, \alpha_{im}^m), \ldots, (r_{nm}^m, \alpha_{nm}^m)$</td>
</tr>
<tr>
<td>..</td>
<td>..</td>
</tr>
<tr>
<td>$\phi_M$</td>
<td>$(r_{1M}^M, \alpha_{1M}^M), (r_{2M}^M, \alpha_{2M}^M), \ldots, (r_{nM}^M, \alpha_{nM}^M)$</td>
</tr>
</tbody>
</table>

Set of potential displacement vectors $r, \alpha$ given the boundary orientation $\phi$.

--> Generalized template matching.

\[
x_c = x + r_i^m \cos(\alpha_i^m) \\
y_c = y + r_i^m \sin(\alpha_i^m)
\]
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
Instead of indexing displacements by gradient orientation, index by “visual codeword”. 
Instead of indexing displacements by gradient orientation, index by "visual codeword".

Test image
Training

1. Build codebook of patches around extracted interest points using clustering.
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.
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
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

\[
\begin{bmatrix}
\frac{\delta I}{\delta x'} & \frac{\delta I}{\delta y}
\end{bmatrix}
\]

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

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

From Image To Roads

Detect Road Points and Construct Binary Image Overlay

MASK

Extract Generic Paths

PATHS

Semantic Filter (Roads)

UNCONNECTED ROAD-SEGS

(3D) ROAD-NETWORKS

Compute Road Seg Visibility in High Res Images

(2D) ROAD-NETWORKS

Smooth Segments & Convert to 3D

Network Linker

ROAD ATTRIBUTES (WIDTH, MATERIAL,...)

Fit 3D - Ribbon Model

IU Assisted Interactive Edit

FINAL ROAD MODEL

Convert Model to Output Format
Road Editing
Dendrites And Axons

Fluorescent neurons in the adult mouse brain imaged in vivo through a cranial window using a 2-photon microscope.
Delineation 2012: Neurites ...

Filtered Image

Graph

Maximum Likelihood Subtree
→ Machine plays a crucial role to ensure that the same algorithm works in different situations.
Histogram of Gradient Deviations

\[ \Psi(x) = \begin{cases} 
\text{angle}(\nabla I(x), N(x)), & \text{if } \|x - C(s_x)\| > \varepsilon \\
\text{angle}(\nabla I(x), \Pi(x)), & \text{otherwise}, 
\end{cases} \]

\( \rightarrow \) One histogram per radius interval plus four geometric features (curvature, tortuosity, ....).
Optional: Embedding

$\rightarrow$ Same length feature vectors whatever the actual length of the path.
Optional: Path Classification

Training Images
Ground-truth
Tubular Graphs

Sample Positives
Sample Negatives

Extract Features
GBDT

Learned Path Model

Tubular Graph
Edge Pairs
Extract Features
Trained GBDT
Weighed Graph
Optional: Finding the Best Tree

\[ t^* = \arg\max_{t \in \mathcal{T}(G)} P(T = t | I), \]

\[ = \arg\max_{t \in \mathcal{T}(G)} P(I | T = t) P(T = t), \]

\[ = \arg\min_{t \in \mathcal{T}(G)} \sum_{e_{ij} \in G} c_{ij}^d t_{ij} \]
Optional: QMIP Formulation

\[
\begin{align*}
\min & \quad \sum_{e_{ij} \in E, e_{jk} \in E} c_{ijk} t_{ij} t_{jk} \\
\text{s.t.} & \quad \sum_{v_j \in V \setminus \{v_r\}} y_{rj}^l \leq 1, \quad \forall v_l \in V \setminus \{v_r\}, \\
& \quad \sum_{v_j \in V \setminus \{v_k\}} y_{jk}^l \leq 1, \quad \forall v_l \in V \setminus \{v_r\}, \\
& \quad \sum_{v_j \in V \setminus \{v_i,v_r\}} y_{ij}^l - \sum_{v_j \in V \setminus \{v_i,v_l\}} y_{ji}^l = 1, \quad \forall v_k \in V \setminus \{v_r\}, \quad \forall v_i \in V \setminus \{v_r,v_k\}, \\
& \quad y_{ij}^l \leq t_{ij}, \quad \forall e_{ij} \in E, v_l \in V \setminus \{v_r,v_i,v_j\}, \\
& \quad y_{il}^l = t_{il}, \quad \forall e_{il} \in E, \\
& \quad y_{ij}^l \geq 0, \quad \forall e_{ij} \in E, v_l \in V \setminus \{v_r,v_i\}, \\
& \quad t_{ij} \in \{0, 1\}, \quad \forall e_{ij} \in E.
\end{align*}
\]

given the root note \(v_r\).
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!!
Delineation 2012

These two steps are closely related!

Turetken et al., PAMI’16
Reminder: U-Net

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

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

Minimize

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

where

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

Mosinska et al, CVPR’18.
Network Output

Image  BCE Loss  Ground truth
Accounting for Topology

Onur et al., PAMI’21

- 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.
Iterative Refinement

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.
These two steps are performed by the same network.

Mosinska et al, PAMI’19.
Dual Use U-Net

Image and Binary Mask

Tubularity Map

[0.991]

Path score
Streets Of Toronto
Dendrites And Axons
Typical Annotations

Original Image

“Ground truth” + Mistakes

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

Oner et al. , arXiv’21
\[ \Theta^*, C^* = \arg \min_{\Theta, C} \sum_{i=1}^{N} \mathcal{L}(D(c_i), y_i) + R(c_i) \]

where

- The \( y_i \) are the network outputs;
- The \( c_i \) are the annotation vertices;
- \( C \) is the vector obtained by concatenating all the \( c_i \);
- \( D \) is a distance transform;
- \( L \) is the MSE loss;
- \( R \) is a regularization term.
Snake Optimization
Improved Results

Annotated image  Vanilla U-Net  Network snakes
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
It is difficult to make predictions, especially about the future. Sometimes attributed to Niels Bohr.
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.