A Primer on Image Segmentation

It’s all PDE’s anyways

Jonas Actor

Rice University

21 February 2018
Table of Contents

1 Motivation
2 Simple Methods
3 Edge Methods
4 PDE Energy Methods
5 Other Methods
6 Parting Thoughts
Table of Contents

1. Motivation
2. Simple Methods
3. Edge Methods
4. PDE Energy Methods
5. Other Methods
6. Parting Thoughts
Develop new treatments for tuberculosis (*Mycobacterium tuberculosis*)

- $\approx 1/3$ of global population currently infected
- most common infectious disease worldwide
- 2 phases: innate immune response, then dormancy
- Factors in the transition between these phases only partially known

**Goal:** Track Mtb within infected cells during innate immune response
Real-World Problem

MTB Movie
Other Motivations

- Computer vision
- Self-driving cars
- Medical imaging (MRI, CT, ultrasound)
- Fingerprint recognition
- Art history and restoration
Some Notation

- $\Omega$ image domain, normally $[0, 1]^2$ or $[0, 1]^3$
- $u_0 : \Omega \to [0, 1]$ greyscale image
- $\Gamma \subset \Omega$ desired segment

Implemented in OpenCV + FEniCS in Python
Table of Contents

1 Motivation

2 Simple Methods

3 Edge Methods

4 PDE Energy Methods

5 Other Methods

6 Parting Thoughts
Simple Methods

Why?
- fast
- intuitive

When?
- no noise, blurring, obstructions, or known already
- know some properties already
Sweatshop Method

- robust to noise, blurring, variable contrast, halo effects
- often treated as "best possible" or ground truth
- time-consuming
- reliable?
For a given threshold $\theta$, 

$$\Gamma = 1_{\{u_0(x) \geq \theta : x \in \Omega\}}$$

Works best when

- clear foreground + background
- items to segment have similar intensities
- no variable contrast, obstructions
Thresholding

A macrophage

http://medcell.med.yale.edu/histology/connective_tissue_lab/macrophage_em.php
Thresholding

A macrophage, thresholded
Local Thresholding

For a given threshold map \( \theta : \Omega \rightarrow [0, 1] \),

\[
\Gamma = 1_{\{ u_0(x) \geq \theta(x) : x \in \Omega \}}
\]

For example, \( \theta(x) = \text{average}\{ u_0(y) : y \in B_{\epsilon}(x) \} \).
Local Thresholding

(a) Thresholding

(b) Local Thresholding
Simple Methods can Fail

Problems when... 

- noise
- blurred or obstructed objects
- halo / variable contrast or lighting
- no prior assumptions on object shape
Simple Methods can Fail

Cow!
Simple Methods can Fail

Cow: Thresholding
Simple Methods can Fail

Cow: Adaptive Thresholding
Motivation: Define edges as areas across high gradients

Interior within edges gives us segmented region $\Gamma$

$\rightarrow$ can recover with e.g. watershed algorithm
Canny Edge Detection

- Given image $u_0$
- Apply smoother (e.g. Gaussian kernel, Laplacian) $u_0 \mapsto u$
- Approximate $\nabla u$ by Finite Differences

$$\nabla u(x, y) = \frac{1}{h} \left[ u(x^+, y) - u(x, y) \right]$$

- Threshold $\|\nabla u\|$
Edge Detection

Not a Duck
Edge Detection

Canny Edges of a non-duck
Eikonal Equation

First order fully nonlinear PDE:

\[ \|\nabla u\| = 1 \quad \text{on } \Omega \]
\[ u = 0 \quad \text{on } \partial \Omega \]

\( u \): minimal distance to boundary over a uniform cost field

\( \longrightarrow \) continuous version of Shortest Path
Let $F = \text{CannyEdge}(u_0)$; solve

$$
\|\nabla u\| = \frac{1}{F} \quad \text{on } \Omega
$$

$$
u = 0 \quad \text{on } \partial \Omega
$$

cost field $\sim$ penalize crossing edges
Numerical solution: Fast Marching Method

- Keep track of distance values for ‘far’, ‘considered’, ‘accepted’ nodes
- Update far nodes if accepted/considered nodes nearby → considered
- Considered with smallest value → accepted
- Repeat until no more considered nodes
Eikonal Equation: Fast Marching Method

Generalization of Dijkstra’s algorithm

- more than just one previous node to calculate distance
- \( O(\text{pixels}) = O(|V|) = O(|E|) \) runtime
  
  Dijkstra’s algorithm with integer weights on undirected graph
Eikonal Equation
Eikonal Equation
Image Approximation

3-D Scene

image $u$ → blur $K$ → noise $n$

observed image $u_0$
Approximate $u_0 : \Omega \rightarrow [0, 1]$ by $u \in V$ vector space (often $H^1(\Omega)$)

Assumptions:

- well-defined solution space
- curvature of edges well-defined
- no fractals
**Goal:** Define an energy functional on an image, then minimize functional

\[
\hat{u} = \min_{u \in V, \Gamma \subseteq \Omega} E[u, \Gamma | u_0]
\]

- Mumford-Shah
- Chan-Vese
- Ambrosio-Tortorelli
Mumford-Shah Functional

\[ E[u, \Gamma|u_0] = \alpha \nu(\Gamma) + \frac{\beta}{2} \int_{\Omega \setminus \Gamma} |\nabla u|^2 \, dx + \frac{\lambda}{2} \int_{\Omega} (K[u] - u_0)^2 \, dx \]

- \( u_0 \): image
- \( u \): desired approximation
- \( \Gamma \subset \Omega \): segmented section
Mumford-Shah Functional

\[ E[u, \Gamma | u_0] = \alpha \nu(\Gamma) + \frac{\beta}{2} \int_{\Omega \setminus \Gamma} |\nabla u|^2 \, dx + \frac{\lambda}{2} \int_{\Omega} (K[u] - u_0)^2 \, dx \]

- \( \nu(\Gamma) \): (Hausdorff) length of curve \( \partial \Gamma \)
- Keeps from making a curve around every pixel
Mumford-Shah Functional

\[
E[u, \Gamma|u_0] = \alpha \nu(\Gamma) + \frac{\beta}{2} \int_{\Omega \setminus \Gamma} |\nabla u|^2 \, dx + \frac{\lambda}{2} \int_{\Omega} (K[u] - u_0)^2 \, dx
\]

- penalize regions of high gradient or crossing \( \partial \Gamma \)
- groups similar parts of the image into contiguous segments
Mumford-Shah Functional

\[ E[u, \Gamma | u_0] = \alpha \nu(\Gamma) + \frac{\beta}{2} \int_{\Omega \setminus \Gamma} |\nabla u|^2 \, dx + \frac{\lambda}{2} \int_{\Omega} (K[u] - u_0)^2 \, dx \]

- keep \( u_0 \) close to \( u \)
- \( K \) image (deblurring) operator; often \( K = I \)
Mumford-Shah Functional

- Maintains edges in $\Gamma$ while smoothing noise
- Use $u$ instead of $u_0$ for other segmentation methods
- $V = \{\text{piecewise constants}\} \rightarrow$ thresholding
Still not a duck...
Mumford-Shah Functional

Mumford-Shah approximation of Cow
Mumford-Shah Functional

(a) Canny Edge on $u_0$

(b) Canny Edge on $u = MS[u_0]$
Mumford-Shah Functional

\[ E[u, \Gamma|u_0] = \alpha \nu(\Gamma) + \frac{\beta}{2} \int_{\Omega \setminus \Gamma} |\nabla u|^2 \, dx + \frac{\lambda}{2} \int_{\Omega} (K[u] - u_0)^2 \, dx \]

- \( \alpha, \beta, \lambda \geq 0 \)
- Chan-Vese Active Contour model: \( \beta \to \infty \)
- Ambrosio-Tortorelli: replace \( \Gamma \) region with \( \Phi \)
  \[ x \in \Gamma \iff \Phi(x) \text{ large} \]
- Bilevel optimization problem
  \[ \hat{u} = \min_{\alpha, \beta, \lambda} \min_{u \in V} E[u, \Gamma|u_0] \]
Send $\alpha \to \infty$ and drop $\Gamma$ dependence:

$$E[u|u_0] = \frac{\beta}{2} \int_{\Omega} |\nabla u|^2 \, dx + \frac{\lambda}{2} (K[u] - u_0)^2 \, dx$$

**Theorem (Elliptic Solution)**

Suppose $u_0 \in L^\infty(\Omega)$. Then, the energy functional $E[u|u_0]$ has a unique minimizer $u_* \in H^1(\Omega)$ that satisfies the elliptic PDE

$$-\beta \Delta u + \lambda (u - u_0) = 0 \quad \text{on } \Omega$$

$$\partial_n u = 0 \quad \text{on } \partial \Omega$$
Solve the following PDE:

\[-\beta \Delta u + \lambda (u - u_0) = 0 \quad \text{on } \Omega\]
\[\partial_n u = 0 \quad \text{on } \partial \Omega\]

Multigrid solver $\rightarrow$ resolution at different scales

- Hierarchical scale separation of image features
- Corner identification
- Skeleton construction (stick figures)
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Motivation</td>
</tr>
<tr>
<td>2</td>
<td>Simple Methods</td>
</tr>
<tr>
<td>3</td>
<td>Edge Methods</td>
</tr>
<tr>
<td>4</td>
<td>PDE Energy Methods</td>
</tr>
<tr>
<td>5</td>
<td>Other Methods</td>
</tr>
<tr>
<td>6</td>
<td>Parting Thoughts</td>
</tr>
</tbody>
</table>
Other Methods: Graphs

- min-cut
- normalized min-cut
- spectral Graph Laplacian
- equivalence to Finite Differences for specific PDE methods

Often have good algorithms!
Other Methods: Neural Networks

- CNN, U-Net
- Many different architectures
- Labeled training data
- Viewed as state-of-the-art
# Table of Contents

1. Motivation
2. Simple Methods
3. Edge Methods
4. PDE Energy Methods
5. Other Methods
6. Parting Thoughts
Parting Thoughts

- Simple methods for occasional users
- PDE methods
  - PDEs provide strong analytical framework
  - Can be fast
  - Robust to noise
  - Can do more complicated tasks: inpainting, deblurring, etc.
- Other methods
  - Graph methods
  - NN methods
    - Need training data
    - Hard analysis?
  - Scattering, Sampling, many others too!
Questions Moving Forward

• Extension to 3D?
• Learning on images, not just segmentation
• Incoporation into models
• Automation? Supervised? Semisupervised?
• Future Work:
  • Using PDE methods to evaluate NN fitness
  • Unrolling methods: bilevel optimization $\mapsto$ NN