A Primer on Image Segmentation

It's all PDE's anyways

Jonas Actor

Rice University

21 February 2018

Table of Contents

Table of Contents

Real-World Problem

Develop new treatments for tuberculosis (*Mycobacterium tuberculosis*)

- $\approx 1/3$ of global population currently infected
- most common infectios 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

- Ω 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

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?

Thresholding

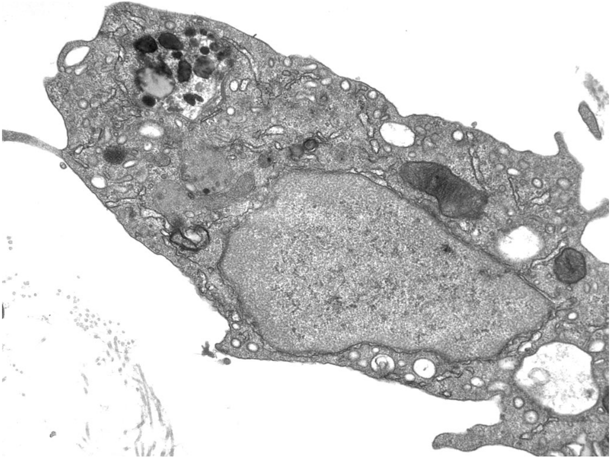
For a given threshold θ,

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

Works best when

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

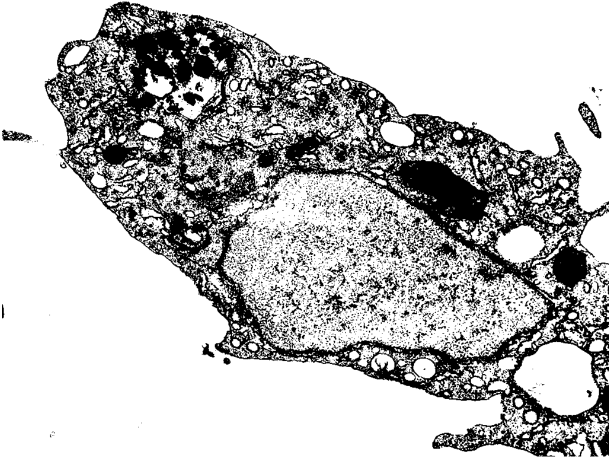
Thresholding



A macrophage

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

Thresholding



A macrophage, thresholded

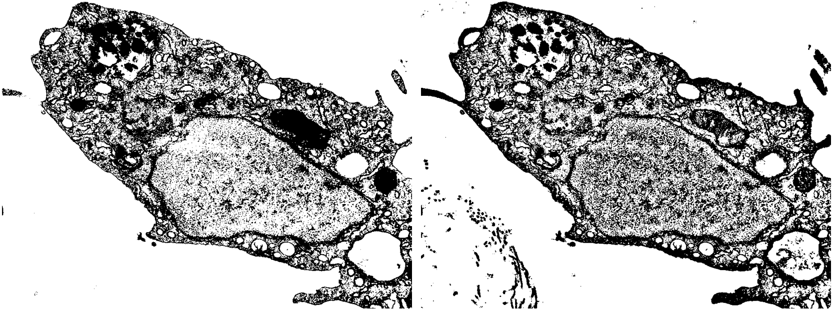
Local Thresholding

For a given threshold **map** $\theta : \Omega \to [0, 1]$,

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

For example, $\theta(x) = average\{u_0(y) \;\; : \;\; y \in B_\epsilon(x)\}$

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

Table of Contents

Edge Detection

Motivation: Define edges as areas across high gradients

Interior within edges gives us segmented region Γ
$\longrightarrow$ 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 ∇u by Finite Differences

$$\nabla u(x, y) = \frac{1}{h} \begin{bmatrix} u(x^+, y) - u(x, y) \\ u(x, y^+) - u(x, y) \end{bmatrix}$$

- Threshold $\|\nabla u\|$

Edge Detection

Not a Duck

Edge Detection

Canny Edges of a non-duck

Eikonal Equation

First order fully nonlinear PDE:

$$\begin{aligned} \|\nabla u\| &= 1 \quad \text{on } \Omega \\ u &= 0 \quad \text{on } \partial\Omega \end{aligned}$$

u: minimal distance to boundary over a uniform cost field
$\longrightarrow$ continuous version of Shortest Path

Eikonal Equation for Segmentation

Let $F = \mathit{CannyEdge}(u_0)$; solve

$$\begin{aligned} \|\nabla u\| &= \frac{1}{F} \quad \text{on } \Omega \\ u &= 0 \quad \text{on } \partial\Omega \end{aligned}$$

cost field $\sim$ penalize crossing edges

Eikonal Equation: Numerical Solution

Numerical solution: Fast Marching Method

- Keep track of distance values for ‘far’, ‘considered’, ‘accepted’ nodes
- Update far nodes if accepted/considered nodes nearby $\rightarrow$ considered
- Considered with smallest value $\rightarrow$ accepted
- Repeat until no more considered nodes

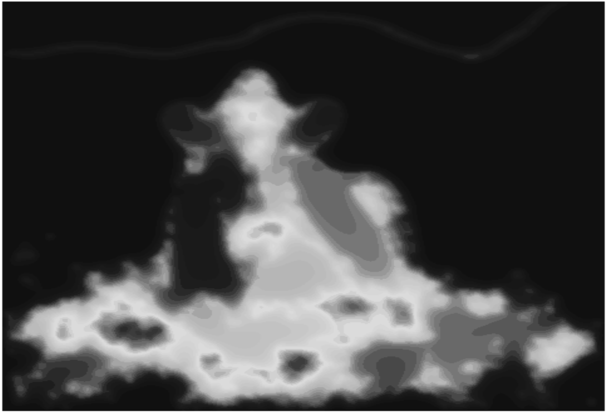
Eikonal Equation: Fast Marching Method

Generalization of Dijkstra's algorithm

- more than just one previous node to calculate distance
- $O(pixels) = O(|V|) = O(|E|)$ runtime

  Dijkstra's algorithm with integer weights on undirected graph

Eikonal Equation



Eikonal Equation

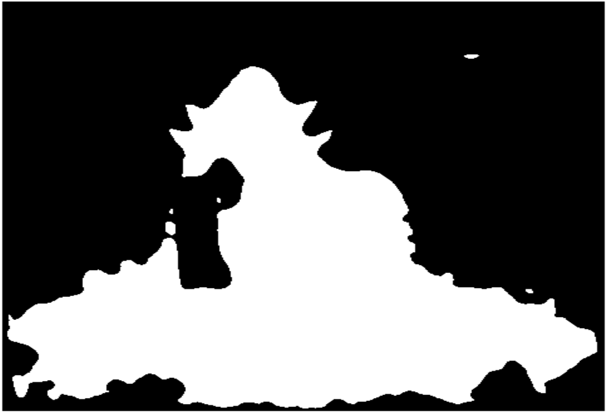


Table of Contents

Image Approximation

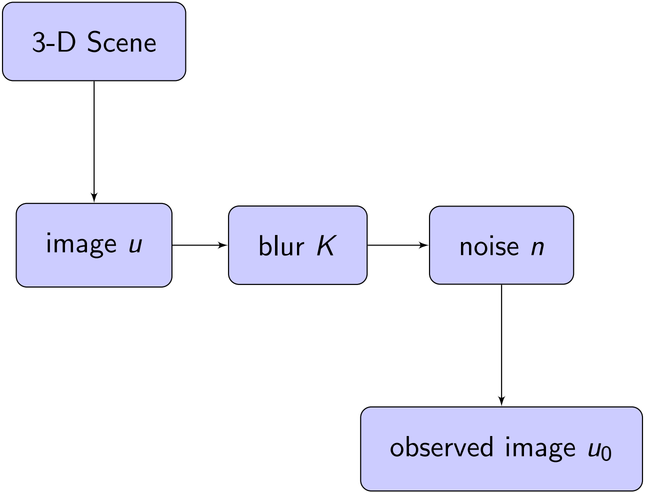


Image Approximation

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

Assumptions:

- well-defined solution space
- curvature of edges well-defined
- no fractals

Image Approximation with PDE Functionals

Goal: Define an energy functional on an image, then minimize functional

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

- 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 Γ while smoothing noise
- Use u instead of u_0 for other segmentation methods
- $V = \{$ piecewise constants $\} \rightarrow$ thresholding

Mumford-Shah Functional

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 Γ region with Φ

$$x \in \Gamma \iff \Phi(x) \text{ large}$$

- Bilevel optimization problem

$$\widehat{u} = \min_{\alpha,\beta,\lambda} \min_{u \in V} E[u,\Gamma|u_0]$$

Laplacian Models

Send $\alpha \to \infty$ and drop Γ 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

$$\begin{aligned} -\beta \triangle u + \lambda(u-u_0) &= 0 \qquad \text{on } \Omega \\ \partial_n u &= 0 \qquad \text{on } \partial\Omega \end{aligned}$$

Laplacian Models

Solve the following PDE:

$$\begin{aligned} -\beta \triangle u + \lambda(u - u_0) &= 0 \qquad \text{on } \Omega \\ \partial_n u &= 0 \qquad \text{on } \partial\Omega \end{aligned}$$

Multigrid solver $\longrightarrow$ resolution at different scales

- Hierarchical scale separation of image features
- Corner identification
- Skeleton construction (stick figures)

Table of Contents

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

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