Graphcut Techniques

Final Project

Goal: Develop new research idea

Can work in groups of up to 3 people
  Tell us groups by this Thursday (10/27)
  Will assume you are working alone unless told otherwise

Project proposals due 10/31
Proposal presentations 10/31 and 11/2
Final presentations 11/28 and 11/30
Final paper 12/7
Last Lecture: Bayesian Matting

Separation of foreground & background
- Partial coverage with fractional alpha
- User provides a trimap
- Bayesian approach
  - Model color distribution in F & B
  - Alternate solving for $\alpha$, then F&B

Solve for each pixel independently
- Using a “data term”

More Foreground/Background

Today: Exploit both data and smoothness

Smoothness
- The alpha value of a pixel likely to be similar to that of its neighbors
- Unless the neighbors have a very different color

Data
- Color distribution of foreground and background (e.g. Bayesian matting)
Multiple Options

Use continuous optimization
See e.g. Chuang’s dissertation, Levin et al. 2006
Pros: Good treatment of partial coverage
Cons: requires energy/probabilities to be well behaved to be solvable

Quantize values of alpha & use discrete optimization
Pros: allows for flexible energy term, efficient solution
Cons: harder to handle fractional alpha

Overview

Interactive image segmentation using graph cut
Binary label: foreground vs. background
User labels some pixels
Similar to trimap, usually sparser
Exploit
Statistics of known Fg & Bg
Smoothness of label
Turn into discrete graph optimization
Graph cut (min cut / max flow)

Images from European Conference on Computer Vision 2006: "Graph Cuts vs. Level Sets", Y. Boykov (UWO), D. Cremers (U. of Bonn), V. Kolmogorov (UCL)
References

Combination of

Yuri Boykov, Marie-Pierre Jolly
Interactive Graph Cuts for Optimal Boundary & Region Segmentation of Objects in N-D Images

C. Rother, V. Kolmogorov, A. Blake. GrabCut: Interactive Foreground Extraction using Iterated Graph Cuts. ACM Transactions on Graphics (SIGGRAPH'04), 2004

Motivation

The rectangle is the only user input
[Rother et al.’s grabcut 2004]
Graph Cut is Very General Tool

Stereo depth reconstruction
Texture synthesis
Video synthesis
Image denoising

Energy Function

Unknown: labeling: one value per pixel, F or B
Energy(labeling) = data + smoothness

Very general situation
Will be minimized

Data: for each pixel
Probability that this color belongs to F (resp. B)
Similar in spirit to Bayesian matting

Smoothness (aka regularization):
per neighboring pixel pair
Penalty for having different label
Penalty is downweighted if the two pixel colors are very different
Similar in spirit to bilateral filter
Data Term

A.k.a regional term  
(because integrated over full region)

\[ D(L) = \sum_i -\log h[L_i](C_i) \]

Where \( i \) is a pixel

\( L_i \) is the label at \( i \) (F or B),

\( C_i \) is the pixel value

\( h[L_i] \) is the histogram of the marked Fg (resp Bg)

Note the minus sign

Color Histograms

Discretize R, G, B (e.g. 20x20x20)

Store 20x20x20 counters

For each pixel in the image

Add one to the appropriate counter

Cool Video

http://senchalabs.github.com/philogl/PhiloGL/examples/histogram/

---

Data Term

\[ D(L) = \sum_{i} -\log h[L_i](C_i) \]

Where \( i \) is a pixel, 
\( L_i \) is the label at \( i \) if \( L \) or \( B \), 
\( C_i \) is the pixel value. 

\( h[L_i] \) is the histogram of the marked \( F \) 
(\( \text{resp} B \)).

Here we use the histogram.

Bayesian matting uses Gaussian model.

This is partially because discrete optimization has fewer 
computational constraints. No need for linear least square.
Hard Constraints

User has labeled some pixels = hard constraints

How to include in optimization?
Replace data term by a huge penalty $K$ if not respected

$D(L_i) = 0$ if respected
$D(L_i) = K$ if not respected
• e.g. $K = -\#\text{pixels}$

Smoothness Term

a.k.a boundary term, a.k.a. regularization

$S(L) = \sum_{\langle i, j \rangle \in N} B(C_i, C_j) \delta(L_i - L_j)$

Where $i, j$ are neighbors
e.g. 8-neighborhood (but I show 4 for simplicity)

$\delta(L_i - L_j)$ is 0 if $L_i = L_j$, 1 otherwise

$B(C_i, C_j)$ is high when $C_i$ and $C_j$ are similar, low if discontinuity between those two pixels
e.g. $\exp(-||C_i - C_j||^2/2\sigma^2)$ where $\sigma$ can be a constant or the local variance

Note positive sign
Recap: Energy function

Labeling: one value $L_i$ per pixel, F or B

Energy(labeling) = Data + Smoothness

Data: for each pixel
- Probability that this color belongs to F (resp. B)
- Using histogram
  $D(L) = \sum_i -\log h[L_i](C_i)$

Smoothness (aka regularization): per neighboring pixel pair
- Penalty for having different label
- Penalty is downweighted if the two pixel colors are very different
  $S(L) = \sum_{\{i,j\} \in N} B(C_i, C_j) \delta(L_i - L_j)$

Optimization

$E(L) = D(L) + \lambda S(L)$

$\lambda$ is a black-magic constant

Find the labeling that minimizes $E$

In this case, how many possibilities?

$2^9 (512)$

We can try them all!

What about megapixel images?
Discussion of Lambda

Lambda balances the smoothness and data terms
Bad news: not invariant to image size

The smoothness (boundary) term scales linearly
Because it depends on the length
You only pay it at the boundary between $F_g$ and $B_g$

The data (regional) term scales quadratically
Because it depends on the area

Makes life difficult

Questions?

Recap:
Labeling $F$ or $B$
Energy(Labeling) = Data+Smoothness
Need efficient way to find labeling with lowest energy
Labeling As Graph Problem

Each pixel = node
Add two label nodes F & B
Labeling: link each pixel to either F or B

F  F  B
F  F  B
F  B  B

Desired result

Idea

Start with a graph with too many edges
- Represents all possible labeling
- Strength of edges depends on data and smoothness terms

Solve as min cut
**Data Term**

Put one edge between each pixel and both F & G

Weight of edge = minus data term
  - Don’t forget huge weight for hard constraints
  - Careful with sign

**Smoothness Term**

Add an edge between each neighbor pair

Weight = smoothness term
Min Cut

Energy optimization equivalent to graph min cut
Cut: Remove edges to disconnect F from B
Minimum: minimize sum of cut edge weight

Min Cut

Graph with one source & one sink node
Edge = bridge
Edge label = cost to cut bridge
What is the min-cost cut that separates source from sink
Min Cut <=> Labeling

In order to be a cut:
For each pixel, either the F or B edge has to be cut

In order to be minimal
Only one edge label per pixel can be cut (otherwise could be added)

Min Cut <=> Optimal Labeling

Energy = - $\Sigma$ weight of remaining links to F & B
+ $\Sigma$ weight cut neighbor links
**Min Cut <=> Optimal Labeling**

Energy = $- \sum \text{all weights to } F \& B$
$+ \sum \text{weight of cut links to } F \& B$
$+ \sum \text{weight cut neighbor links}$

Minimized when last 2 terms are minimized

---

**Questions?**

Recap: We have turned our pixel labeling problem into a graph min cut
- nodes = pixels + 2 labels
- edges from pixel to label = data term
- edges between pixels = smoothness

Now we need to solve the min cut problem
**Min Cut**

Graph with one source & one sink node

- Edge = bridge; Edge label = cost to cut bridge
- Find the min-cost cut that separates source from sink
  - Turns out it’s easier to see it as a flow problem
  - Hence source and sink

**Max Flow**

Directed graph with one source & one sink node

- Directed edge = pipe
- Edge label = capacity
- What is the max flow from source to sink?
Max Flow

Graph with one source & one sink node
Edge = pipe
Edge label = capacity
What is the max flow from source to sink?

Max Flow

What is the max flow from source to sink?
Look at residual graph
Remove saturated edges (green here)
Min cut is at boundary between 2 connected components
Max Flow

What is the max flow from source to sink?

Look at residual graph

Remove saturated edges (gone here)

Min cut is at boundary between 2 connected components

Equivalence of Min Cut / Max Flow

The three following statements are equivalent

The maximum flow is $f$

The minimum cut has weight $f$

The residual graph for flow $f$ contains no directed path from source to sink
Questions?

Recap:
- We have reduced labeling to a graph min cut
  - vertices for pixels and labels
  - edges to labels (data) and neighbors (smoothness)
- We have reduced min cut to max flow

- Now how do we solve max flow???

Max Flow Algorithm

We will study a strategy where we keep augmenting paths (Ford-Fulkerson, Dinic)

Keep pushing water along non-saturated paths
Use residual graph to find such paths
Max flow algorithm

Set flow to zero everywhere
Big loop
  compute residual graph
  Find path from source to sink in residual
    If path exist add corresponding flow
  Else
    Min cut = {vertices reachable from source; other vertices}
    terminate

Animation at
http://www.cse.yorku.ca/~aaw/Wang/MaxFlowStart.htm

Efficiency Concerns

The search for a shortest path becomes prohibitive for the large graphs generated by images
For practical vision/image applications, better (yet related) approaches exist

Maintain two trees from sink & source.
Augment tree until they connect
Add flow for connection
Can require more iterations because not shortest path
  But each iteration is cheaper because trees are reused
Data (Regional) Term

(a) Original B&W photo
(b) Segmentation results
(c) Details of segmentation with regional term
(d) Details of segmentation without regional term

Beyond binary

Graphcut has been beyond binary variables
e.g. discretize depth for stereo into 0, 1, ..., 15
Usually consider a pair of values at a time and solve a graph cut
iterate

http://www.cs.cornell.edu/~rdz/graphcuts.html
Grabcut

Rother et al. 2004

Less user input: only rectangle
Handle color
Extract matte as post-process

Figure 1: Three examples of GrabCut. The user drags a rectangle loosely around an object. The object is then extracted automatically.
Color Data Term

Model 3D color histogram with Gaussians

Because brute force histogram would be sparse
Although I question this. My advice: go brute force, use a volumetric grid in RGB space and blur the histogram

Gaussian Mixture Model (GMM)
Just means histogram = sum of Gaussians
They advise 5 Gaussians

Getting a GMM

Getting one Gaussian is easy: mean / covariance
To get K Gaussians, we cluster the data
And use mean/covariance of each cluster

K-means clustering can do this for us
Idea: Define clusters and their center. Points belong to the cluster with closest center
Take K random samples as seed centers
Iterate:
For each sample
Assign to closest cluster
For each cluster
Center = mean of samples in cluster
Grabcut: Iterative Approach

Initialize
- Background with rectangle boundary pixels
- Foreground with the interior of rectangle

Iterate until convergence
- Compute color probabilities (GMM) of each region
- Perform graphcut segmentation

Apply matting at boundary
Potentially, user edits to correct mistakes

Iterated Graph Cut

User Initialisation

K-means for learning colour distributions
Graph cuts to infer the segmentation
Iterated Graph Cuts

Guaranteed to converge

Result

Energy after each Iteration

Border Matting

Figure 6: Border matting. (a) Original image with trimap overlaid. (b) Notation for contour parameterisation and distance map. Contour C (yellow) is obtained from hard segmentation. Each pixel in $T_U$ is assigned values (integer) of contour parameter $t$ and distance $r_n$ from $C$. Pixels shown share the same value of $t$. (c) Soft step-function for $\alpha$-profile $g$, with centre $\Delta$ and width $\sigma$. 

slide: Rother et al. - Interactive Foreground Extraction
**Results**

Figure 5: **User editing.** After the initial user interaction and segmentation (top row), further user edits (fig. 3) are necessary. Marking roughly with a foreground brush (white) and a background brush (red) is sufficient to obtain the desired result (bottom row).

**Moderately straightforward examples**

... GrabCut completes automatically
Difficult Examples

Camouflage & Low Contrast

Fine structure

No telepathy

Initial Rectangle

Initial Result

Photomontage

Minimize: \[ C(L) = \sum_p C_i(p, L(p)) + \sum_{p,q} C_s(p, q, L(p), L(q)) \]

**Image Objective**  
**Seam Objective**

**Graph-cut optimization** [Li 04] [Rother 04] [Kwatra 03] [Kolmogorov 02] [Boykov 01]
Image Objective: max contrast

Photomontage initial composite
Computed labeling

Source gradients

Gradient composite

Integrated composite

[Source: Perez et al. 03]

Photomontage integrated composite
Photomontage composite
Input images from Debevec

Glamorous

Frightening
Photomontage composite
Image Objective: most common color
Clean plate production
Time-lapse mosaics