Looking for seams

http://www.cs.cmu.edu/~16385/
Course announcements

• Homework 7 is posted and is due on May 5th.
  - It is shorter so that it can fit in the 1.5 week you will have for it.
  - You can use all of your remaining late days for it.
Overview of today’s lecture

- Segmentation.
- Image as a graph.
- Shortest graph paths and Intelligent scissors.
- Graph-cuts and GrabCut.
- Normalized cuts.
- Boundaries.
- Clustering for segmentation.
Slide credits

Most of these slides were adapted from:

• Kris Kitani (15-463, Fall 2016).

Some slides were inspired or taken from:

• Fredo Durand (MIT).
• James Hays (Georgia Tech).
Segmentation
We perceive objects in their entirety before their individual parts.
Similar objects are grouped together

<table>
<thead>
<tr>
<th>Closer objects are grouped together</th>
<th>Similar objects are grouped together</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Closer objects" /></td>
<td><img src="image2" alt="Similar objects" /></td>
</tr>
</tbody>
</table>
Objects with similar motion or change in appearance are grouped together.
Common Region/Connectivity

Connected objects are grouped together
Continuity Principle

Features on a continuous curve are grouped together
Symmetry Principle
Illusory or subjective contours are perceived
Segmentation/Clustering
What is a “good” segmentation??
First idea: Compare to human segmentation or to “ground truth”

No objective definition of segmentation!

- [http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/resources.html](http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/resources.html)
No objective definition of segmentation!

Evaluation: Boundary agreement

Correct if $D < T$

Precision = % of detected boundary pixels that are correct
Recall = % of boundary pixels that are detected
Evaluation: Region overlap with ground truth

\[ OS(S, G) = \frac{|S \cap G|}{|S \cup G|} \]

- Segment #1: 0.825
- Segment #2: 0.892

Ground Truth
Evaluation: Region overlap with ground truth

Ground truth

Mean shift

Graph-based

Spectral

.659

.567

.841
Second idea: Superpixels

- Let’s not even try to compute a “correct” segmentation
- Let’s be content with an * oversegmentation * in which each region is very likely (formal guarantees are hard) to be uniform
Second idea: Superpixels

- Example from: How Do Superpixels Affect Image Segmentation?
Third idea:
Multiple segmentations

- Generate many segmentations of the same image
- Even though many regions are “wrong”, some consensus should emerge

Example: Improving Spatial Support for Objects via Multiple Segmentations
Multiple segmentations: Example

- Task: Regions $\rightarrow$ Features $\rightarrow$ Labels (horizontal, vertical, sky, etc.)
• Chicken and egg problem:
  – If we knew the regions, we could compute the features and label the right regions
  – But to know the right regions we need to know the labels!
• Solution:
  – Generate lots of segmentations
  – Combine the classifications to get consensus

Example from D. Hoiem

Generalities: Summary

- Match ground truth (no objective definition)
- Superpixels = oversegmentation
- Using multiple segmentations
Main approaches

- Spectral techniques
- Segmentation as boundary detection
- Graph-based techniques
- Clustering (K-means and probabilistic)
- Mean shift
Cut and paste procedure

1. Extract Sprites

2. Blend them into the composite
Cut and paste procedure

1. Extract Sprites

2. Blend them into the composite

How do we do this?
Cut and paste procedure

1. Extract Sprites

How do we do this?

Two different ways to think about the same thing:

• Finding seams (i.e., finding the pixels where to cut an image)

• Segmentation (i.e., splitting the image into “foreground” and “background”)

I will be using the two terms interchangeable
Finding seams is also useful for:

- image stitching
- segmentation
- retargeting

Applications
Image as a graph
Fundamental theme of today’s lecture

Images can be viewed as graphs

Nodes: pixels

Edges: Constraints between neighboring pixels
Graph-view of segmentation problem

Segmentation is node-labeling

Nodes: pixels

Edges: Constraints between neighboring pixels

Given pixel values and neighborhoods, decide:

• which nodes to label as foreground/background or
• which nodes to label as seams using graph algorithms
## Graph-view of segmentation problem

Today we will cover:

<table>
<thead>
<tr>
<th>Method</th>
<th>Labeling problem</th>
<th>Algorithm</th>
<th>Intuition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intelligent scissors</td>
<td>label pixels as seams</td>
<td>Dijkstra’s shortest path (dynamic programming)</td>
<td>short path is a good <strong>boundary</strong></td>
</tr>
<tr>
<td>GrabCut</td>
<td>label pixels as foreground/background</td>
<td>max-flow/min-cut (graph cutting)</td>
<td>good <strong>region</strong> has low cutting cost</td>
</tr>
</tbody>
</table>
Shortest graph paths and intelligent scissors
Intelligent scissors

Problem statement:
Given two seed points, find a good boundary connecting them

Challenges:
• Make this real-time for interaction
• Define what makes a good boundary

Mortenson and Barrett (SIGGRAPH 1995)
(you can tell it’s old from the paper’s low quality teaser figure)
Graph-view of this problem

Images can be viewed as graphs

Nodes: pixels

Edges: Constraints between neighboring pixels
Graph-view of this problem

Graph-view of intelligent scissors:

1. Assign weights (costs) to edges
Graph-view of this problem

Graph-view of intelligent scissors:

1. Assign weights (costs) to edges
2. Select the seed nodes
Graph-view of this problem

Graph-view of intelligent scissors:

1. Assign weights (costs) to edges
2. Select the seed nodes
3. Find shortest path between them
Graph-view of this problem

Graph-view of intelligent scissors:

1. Assign weights (costs) to edges
2. Select the seed nodes
3. Find shortest path between them

What algorithm can we use to find the shortest path?
1. Assign weights (costs) to edges
2. Select the seed nodes
3. Find shortest path between them

What algorithm can we use to find the shortest path?
- Dijkstra’s algorithm (dynamic programming)
Dijkstra’s shortest path algorithm

Initialize, given seed s (pixel ID):
• \( \text{cost}(s) = 0 \) % total cost from seed to this point
• \( \text{cost}(\neg s) = \text{big} \)
• \( A = \{\text{all pixels}\} \) % set to be expanded
• \( \text{prev}(s) = \text{undefined} \) % pointer to pixel that leads to \( q=s \)

Precompute \( \text{cost}_2(q, r) \) % cost between \( q \) to neighboring pixel \( r \)

Loop while \( A \) is not empty

1. \( q = \text{pixel in } A \) with lowest cost

2. Remove \( q \) from \( A \)

3. For each pixel \( r \) in neighborhood of \( q \) that is in \( A \)
   a) \( \text{cost}_{\text{tmp}} = \text{cost}(q) + \text{cost}_2(q,r) \) %this updates the costs
   b) if \( \text{cost}_{\text{tmp}} < \text{cost}(r) \)
      i. \( \text{cost}(r) = \text{cost}_{\text{tmp}} \)
      ii. \( \text{prev}(r) = q \)
Graph-view of this problem

Graph-view of intelligent scissors:

1. Assign weights (costs) to edges
2. Select the seed nodes
3. Find shortest path between them

What algorithm can we use to find the shortest path?
• Dijkstra’s algorithm (dynamic programming)

How should we select the edge weights to get good boundaries?
Selecting edge weights

Define boundary cost between neighboring pixels:

1. Lower if an image edge is present (e.g., as found by Sobel filtering).
2. Lower if the gradient magnitude at that point is strong.
3. Lower if gradient is similar in boundary direction.
Selecting edge weights

Gradient magnitude

Edge image

Pixel-wise cost
Making it more interactive

1. Use cursor as the “end” seed, and always connect start seed to that.

2. Every time the user clicks, use that point as a new starting seed and repeat.
Examples
Seam collaging

Another use for image seam selection

Kwatra et al., Graphcut Textures: Image and Video Synthesis using Graph Cuts, SIGGRAPH 2003
Graph-view of this problem

Graph-view of image collaging:

1. Assign weights (costs) to edges
2. Select the seed nodes
3. Find shortest path between them

What edge weights would you use for collaging?
Selecting edge weights for seam collaging

Good places to cut:
• similar color in both images
• high gradient in both images
Seam carving

Another use for image seam selection

Avidan and Shamir, Seam Carving for Content-Aware Image Resizing, SIGGRAPH 2007
Graph-view of this problem

Graph-view of seam carving:

1. Assign weights (costs) to edges
2. Select the seed nodes
3. Find shortest path between them

What edge weights would you use for seam carving?
Shai Avidan
Mitsubishi Electric Research Lab
Ariel Shamir
The interdisciplinary Center & MERL
Examples

Where will intelligent scissors work well, or have problems?
Graph-cuts and GrabCut
GrabCut

Only user input is the box!

Rother et al., “Interactive Foreground Extraction with Iterated Graph Cuts,” SIGGRAPH 2004
Combining region and boundary information

user input

Magic Wand (198?)
Intelligent scissors
GrabCut

result

regions
boundary
regions & boundary
GrabCut is a mixture of two components

1. Segmentation using graph cuts
2. Foreground-background modeling using unsupervised clustering
GrabCut is a mixture of two components

1. Segmentation using graph cuts

2. Foreground-background modeling using unsupervised clustering
Segmentation using graph cuts

Remember: Graph-based view of images

Nodes: pixels

Edges: Constraints between neighboring pixels
Markov Random Field (MRF)

Assign foreground/background labels based on:

\[
\text{Energy } (y; \theta, \text{data}) = \sum_i \psi_1(y_i; \theta, \text{data}) \sum_{i,j \in \text{edges}} \psi_2(y_i, y_j; \theta, \text{data})
\]

- Given its intensity value, how likely is a pixel to be foreground or background?
- Given their intensity values, how likely two neighboring pixels to have two labels?
- What kind of cost functions would you use for GrabCut?
Solving MRFs using max-flow/min-cuts (graph cuts)

\[
\text{Energy} \ (y; \theta, \text{data}) = \sum_i \psi_1 (y_i; \theta, \text{data}) \sum_{i,j \in \text{edges}} \psi_2 (y_i, y_j; \theta, \text{data})
\]
Solving MRFs using max-flow/min-cuts (graph cuts)

\[ \text{Energy } (y; \theta, \text{data}) = \sum_i \psi_1(y_i; \theta, \text{data}) \sum_{i, j \in \text{edges}} \psi_2(y_i, y_j; \theta, \text{data}) \]
A toy visual example

Unary potential

0: $-\log P(y_i = 0 ; \text{data})$
1: $-\log P(y_i = 1 ; \text{data})$

Pairwise potential

\begin{align*}
0 & \quad 1 \\
0 & \quad 0 & K \\
1 & \quad K & 0 \\
\end{align*}

Low cost
Graph-cuts segmentation

1. Define graph
   - usually 4-connected or 8-connected
2. Set weights to foreground/background
3. Set weights for edges between pixels
   \[
   \text{unary}_\text{potential}(x) = -\log \left( \frac{P(c(x); \theta_{\text{foreground}})}{P(c(x); \theta_{\text{background}}} \right)
   \]
   \[
   \text{edge}_\text{potential}(x, y) = k_1 + k_2 \exp \left( \frac{-\|c(x) - c(y)\|^2}{2\sigma^2} \right)
   \]
4. GraphCut: Apply min-cut/max-flow algorithm

How would you determine these for GrabCut?
GrabCut is a mixture of two components

1. Segmentation using graph cuts

2. Foreground-background modeling using unsupervised clustering
Foreground-background modeling

Given foreground/background labels

build a color model for both
Learning color models

Given a set of points, fit k Gaussians.

you can think the axes as ‘red’ and ‘blue’ channels
Learning color models

Given a set of points, fit k Gaussians.

You can think the axes as ‘red’ and ‘blue’ channels.

How would you solve this problem?
Intuition: “hard” clustering using K-means

Given k:
1. Select initial centroids at random.
2. Assign each object to the cluster with the nearest centroid.
3. Compute each centroid as the mean of the objects assigned to it.
4. Repeat previous 2 steps until no change.
1. Select initial centroids at random
1. Select initial centroids at random

2. Assign each object to the cluster with the nearest centroid.
K-means visualization

1. Select initial centroids at random

2. Assign each object to the cluster with the nearest centroid.

3. Compute each centroid as the mean of the objects assigned to it (go to 2)
1. Select initial centroids at random

2. Assign each object to the cluster with the nearest centroid.

3. Compute each centroid as the mean of the objects assigned to it (go to 2)

K-means visualization

Repeat previous 2 steps until no change
1. Select initial centroids at random
2. Assign each object to the cluster with the nearest centroid.
3. Compute each centroid as the mean of the objects assigned to it (go to 2)

K-means visualization

2. Assign each object to the cluster with the nearest centroid.
Expectation-Maximization: “soft” version of K-means

Given k:
1. Select initial centroids at random.
2. Assign each object to the cluster with the nearest centroid.
3. Compute each centroid as the mean of the objects assigned to it.
4. Repeat previous 2 steps until no change.

E-step: compute the probability of each object being in a cluster and covariance
M-step: weighed by the probability of being in that cluster
Unsupervised clustering

Model: **Mixture of Gaussians**

Algorithm: **Expectation Maximization**

**E step**

\[ Q(\theta|\theta^{(t)}) = E_{Z|X,\theta^{(t)}} [\log L(\theta; X, Z)] \]

**M step**

\[ \theta^{(t+1)} = \arg \max_\theta Q(\theta|\theta^{(t)}) \]

Compute the expected log-likelihood

Update parameters based on likelihood

Important result for GrabCut:
we can compute the **likelihood** of a pixel belonging to the **foreground** or **background** as:

\[
p(c(x); \theta) = \prod_{k=1}^{K} \alpha_k \cdot \mathcal{N}(c(x); \mu_k, \Sigma_k)\]
GrabCut is a mixture of two components

1. Segmentation using graph cuts
   • Requires having foreground model

2. Foreground-background modeling using unsupervised clustering
   • Requires having segmentation

What do we do?


GrabCut: iterate between two steps

1. Segmentation using graph cuts
   • Requires having foreground model

2. Foreground-background modeling using unsupervised clustering
   • Requires having segmentation

What do we do?
Iteration can be interactive.

- User specified box
- User edit
- Iterated graph cut
- Output
Examples

Magic Wand

Magnetic Lasso

Knockout 2

Bayes Matte

BJ – Graph Cut

GrabCut
Examples

What is easy or hard about these cases for graph cut-based segmentation?
Examples
Examples

Lazy Snapping
[Li et al. SIGGRAPH 2004]
Graph-cuts are a very general, very useful tool

- denoising
- stereo
- texture synthesis
- segmentation
- classification
- recognition
- ...

3D model of scene
References

Basic reading:
• Szeliski textbook, Sections 5.1.3, 5.3.1, 9.3.2, 9.3.3, 10.4.3.