**Image Segmentation**

- Goal: identify groups of pixels that go together

---

**The goals of segmentation**

- Separate image into coherent “objects”

---

**The goals of segmentation**

- Separate image into coherent “objects”
- Group together similar-looking pixels for efficiency of further processing

---

The goals of segmentation

- Separate image into coherent “objects”
- Group together similar-looking pixels for efficiency of further processing
  “superpixels”

Segmentation

- Compact representation for image data in terms of a set of components
- Components share “common” visual properties
- Properties can be defined at different level of abstractions

What is segmentation?

- Clustering image elements that “belong together”
  - Partitioning
    - Divide into regions/sequences with coherent internal properties
  - Grouping
    - Identify sets of coherent tokens in image

Segmentation is a global process

What are the occluded numbers?
Segmentation is a global process

What are the occluded numbers?

Occlusion is an important cue in grouping.

... but not too global

Groupings by Invisible Completions

Magritte, 1957

* Images from Steve Lehar’s Gestalt papers
Groupings by Invisible Completions

Perceptual organization

“...the processes by which the bits and pieces of visual information that are available in the retinal image are structured into the larger units of perceived objects and their interrelations”

Stephen E. Palmer, Vision Science, 1999

Gestalt Psychology

- German: Gestalt - "form" or "whole"
- Berlin School, early 20th century
  - Kurt Koffka, Max Wertheimer, and Wolfgang Köhler
- Gestalt: whole or group
  - Whole is greater than sum of its parts
  - Relationships among parts can yield new properties/features
- Psychologists identified series of factors that predispose set of elements to be grouped (by human visual system)

“I stand at the window and see a house, trees, sky. Theoretically I might say there were 327 brightnesses and nuances of colour. Do I have “327”? No. I have sky, house, and trees.”

Max Wertheimer (1880-1943)
**Proximity**

Slide credit: K. Grauman

---

**Familiarity**

Slide credit: B. Freeman and A. Torralba

---

**Familiarity**

Slide credit: B. Freeman and A. Torralba

---

**Emergence**

Slide credit: S. Lazebnik

---

http://www.capitol.edu/Resources/Images/outside6_035.jpg

Slide credit: K. Grauman

---

http://en.wikipedia.org/wiki/Gestalt_psychology

Slide credit: S. Lazebnik
Gestalt cues

- Good intuition and basic principles for grouping
- Basis for many ideas in segmentation and occlusion reasoning
- Some (e.g., symmetry) are difficult to implement in practice

Segmentation methods

- Segmentation as clustering
  - K-means clustering
  - Mean-shift segmentation
- Graph-theoretic segmentation
  - Min cut
  - Normalized cuts
- Boundary detection

• Now how to determine the three main intensities that define our groups?
• We need to cluster.
Segmentation as clustering

• Cluster together (pixels, tokens, etc.) that belong together...
• Agglomerative clustering
  – attach closest to cluster it is closest to – repeat
• Divisive clustering
  – split cluster along best boundary – repeat
• Dendrograms
  – yield a picture of output as clustering process continues

Greedy Clustering Algorithms

Algorithm 15.3: Agglomerative clustering, or clustering by merging

[Algorithm details]

Algorithm 15.4: Divisive clustering, or clustering by splitting

[Algorithm details]
Agglomerative clustering

1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat
**Agglomerative clustering**

1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat

**Common similarity/distance measures**

- \( P \)-norms
  - City Block (L1)
  - Euclidean (L2)
  - L-infinity

- Mahalanobis
  - Scaled Euclidean

- Cosine distance

Here \( x \) is the distance btw. two points

**Dendograms**

**Agglomerative clustering**

How to define cluster similarity?
- Average distance between points, maximum distance, minimum distance
- Distance between means or medoids

How many clusters?
- Clustering creates a dendrogram (a tree)
- Threshold based on max number of clusters or based on distance between merges
**Agglomerative clustering**

**Good**
- Simple to implement, widespread application
- Clusters have adaptive shapes
- Provides a hierarchy of clusters

**Bad**
- May have imbalanced clusters
- Still have to choose number of clusters or threshold
- Need to use an “ultrametric” to get a meaningful hierarchy

**Segmentation methods**
- Segmentation as clustering
  - K-means clustering
  - Mean-shift segmentation
- Graph-Theoretic Segmentation
  - Min cut
  - Normalized cuts
- Interactive segmentation
- Boundary detection

---

**K-means clustering**

**Basic idea:** randomly initialize the k cluster centers, and iterate between the two steps we just saw.

1. Randomly initialize the cluster centers, \( c_1, \ldots, c_k \)
2. Given cluster centers, determine points in each cluster
   - For each point \( p \), find the closest \( c_i \). Put \( p \) into cluster \( i \)
3. Given points in each cluster, solve for \( c_i \)
   - Set \( c_i \) to be the mean of points in cluster \( i \)
4. If \( c_i \) have changed, repeat Step 2

**Properties**
- Will always converge to some solution
- Can be a “local minimum”
  - does not always find the global minimum of objective function:

\[
\sum_{i} \sum_{p \text{ in cluster } i} ||p - c_i||^2
\]
K-means
1. Ask user how many clusters they’d like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it’s closest to. (Thus each Center “owns” a set of datapoints)
4. Each Center finds the centroid of the points it owns
5. ...and jumps there
6. ...Repeat until terminated!
**K-means: pros and cons**

**Pros**
- Simple, fast to compute
- Converges to local minimum of within-cluster squared error

**Cons/issues**
- Setting k?
- Sensitive to initial centers
- Sensitive to outliers
- Detects spherical clusters
- Assuming means can be computed

---

**An aside: Smoothing out cluster assignments**

- Assigning a cluster label per pixel may yield outliers:

  ![Original image](image1.png)  ![Labeled image](image2.png)

- How to ensure they are spatially smooth?

---

**Segmentation as clustering**

Depending on what we choose as the feature space, we can group pixels in different ways.

Grouping pixels based on intensity similarity

Feature space: intensity value (1-d)
Segmentation as clustering

Depending on what we choose as the feature space, we can group pixels in different ways.

Grouping pixels based on color similarity

Feature space: color value (3-d)

K-means clustering using intensity alone and color alone

Segmentation as clustering

Grouping pixels based on intensity similarity

Clusters based on intensity similarity don’t have to be spatially coherent.

K-means using color alone, 11 segments
Segmentation as clustering

K-means using color alone, 11 segments.

Color alone often will not yield salient segments!

Segmentation as clustering

Depending on what we choose as the feature space, we can group pixels in different ways.

Grouping pixels based on intensity + position similarity

Both regions are black, but if we also include position \((x,y)\), then we could group the two into distinct segments; way to encode both similarity & proximity.

Segmentation as clustering

• Color, brightness, position alone are not enough to distinguish all regions…

Segmentation as clustering

Depending on what we choose as the feature space, we can group pixels in different ways.

Grouping pixels based on texture similarity

Feature space: filter bank responses (e.g., 24-d)
Texture representation example

- Windows with primarily horizontal edges
- Windows with small gradient in both directions
- Windows with primarily vertical edges
- Both

Statistics to summarize patterns in small windows

<table>
<thead>
<tr>
<th>Window</th>
<th>Dimension 1 (mean d/dx value)</th>
<th>Dimension 2 (mean d/dy value)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Win.#1</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>Win.#2</td>
<td>18</td>
<td>7</td>
</tr>
<tr>
<td>Win.#9</td>
<td>20</td>
<td>20</td>
</tr>
</tbody>
</table>

Dimension 1 (mean d/dx value) Dimension 2 (mean d/dy value)

Windows with small gradient in both directions
Windows with primarily vertical edges

Segmentation with texture features

- Find “textons” by clustering vectors of filter bank outputs
- Describe texture in a window based on texton histogram


Slide credit: K Grauman, L Lazebnik

Image segmentation example

Pixel properties vs. neighborhood properties

These look very similar in terms of their color distributions (histograms).

How would their texture distributions compare?
Material classification example

For an image of a single texture, we can classify it according to its global (image-wide) texton histogram.

Nearest neighbor classification: label the input according to the nearest known example’s label.

\[ \chi^2(h_1, h_j) = \frac{1}{2} \sum_{k=1}^{K} \frac{(h_1(k) - h_j(k))^2}{h_1(k) + h_j(k)} \]

Segmentation methods

- Segmentation as clustering
  - K-means clustering
  - Mean-shift segmentation
- Graph-theoretic segmentation
  - Min cut
  - Normalized cuts
- Boundary detection

Mean shift clustering and segmentation

- An advanced and versatile technique for clustering-based segmentation

**Finding Modes in a Histogram**

- How Many Modes Are There?
  - Easy to see, hard to compute

**Mean shift algorithm**

- The mean shift algorithm seeks *modes* or local maxima of density in the feature space

**Mean Shift Algorithm**

1. Choose a search window size.
2. Choose the initial location of the search window.
3. Compute the mean location (centroid of the data) in the search window.
4. Center the search window at the mean location computed in Step 3.
5. Repeat Steps 3 and 4 until convergence.

The mean shift algorithm seeks the “mode” or point of highest density of a data distribution:

Two issues:
(1) Kernel to interpolate density based on sample positions.
(2) Gradient ascent to mode.
**Mean shift**

- Search window
- Center of mass
- Mean Shift vector

**Mean shift clustering**

- Cluster: all data points in the attraction basin of a mode
- Attraction basin: the region for which all trajectories lead to the same mode

**Mean shift clustering/segmentation**

- Find features (color, gradients, texture, etc)
- Initialize windows at individual feature points
- Perform mean shift for each window until convergence
- Merge windows that end up near the same “peak” or mode
Apply mean shift jointly in the image (left col.) and range (right col.) domains

Intensities of pixels within image domain window

Center of mass of pixels within both image and range domain windows

Center of mass of pixels within both image and range domain windows

Window in range domain

Window in image domain

Mean shift segmentation results

http://www.caip.rutgers.edu/~comanici/MSPAMI/msPamiResults.html

More results
Mean shift pros and cons

- **Pros**
  - Does not assume spherical clusters
  - Just a single parameter (window size)
  - Finds variable number of modes
  - Robust to outliers

- **Cons**
  - Output depends on window size
  - Computationally expensive
  - Does not scale well with dimension of feature space

Segmentation methods

- Segmentation as clustering
  - K-means clustering
  - Mean-shift segmentation
- Graph-theoretic segmentation
  - Min cut
  - Normalized cuts
- Boundary detection

Graph-Theoretic Image Segmentation

Build a weighted graph \( G=(V,E) \) from image

- \( V \): image pixels
- \( E \): connections between pairs of nearby pixels
- \( W_{ij} \): probability that \( i \& j \) belong to the same region

Segmentation = graph partition
Graphs Representations

\[ W = \begin{bmatrix} 1 & 1 & 0.3 & 0 & 0 \\ 1 & 1 & 0.4 & 0.2 & 0 \\ 0.3 & 0.4 & 1 & 0.6 & 0.7 \\ 0 & 0 & 0.6 & 1 & 1 \\ 0 & 0.2 & 0.7 & 1 & 1 \end{bmatrix} \]

Adjacent Matrix

Affinity Matrix

Affinity between pixels

Similarities among pixel descriptors
\[ W_{ij} = \exp(-||z_i - z_j||^2 / \sigma^2) \]

\( \sigma = \) Scale factor… it will hunt us later

Segmentation by graph partitioning

- Break graph into segments
  - Delete links that cross between segments
  - Easiest to break links that have low affinity
    - similar pixels should be in the same segments
    - dissimilar pixels should be in different segments
**Affinity between pixels**

Similarities among pixel descriptors

\[ W_{ij} = \exp(-||z_i - z_j||^2 / \sigma^2) \]

Interleaving edges

\[ W_{ij} = 1 - \max Pb \]

\( \sigma = \text{Scale factor... it will hunt us later} \)

\( \text{Line between } i \text{ and } j \)

With \( Pb = \text{probability of boundary} \)

---

**Scale affects affinity**

- Small \( \sigma \): group only nearby points
- Large \( \sigma \): group far-away points

---

**Feature grouping by “relocalisation” of eigenvectors of the proximity matrix**

*British Machine Vision Conference, pp. 103-108, 1990*

Guy L. Scott
Department of Engineering Science
University of Oxford

H. Christopher Longuet-Higgins
University of Sussex
Brighton

**W**

\[ W_{ij} = \exp(-||z_i - z_j||^2 / \sigma^2) \]

With an appropriate \( \sigma \)

\[
\begin{array}{ccc}
A & B & C \\
A & 1.00 & 0.63 & 0.03 \\
B & 0.63 & 1.00 & 0.0 \\
C & 0.03 & 0.0 & 1.00 \\
\end{array}
\]

The eigenvectors of **W** are:

The first 2 eigenvectors group the points as desired...

---

**Example eigenvector**

*Slide credit: B. Freeman and A. Torralba*
Example eigenvector

- Points
- Eigenvector
- Affinity matrix

Slide credit: B. Freeman and A. Torralba

Graph cut

- Set of edges whose removal makes a graph disconnected
- Cost of a cut: sum of weights of cut edges
- A graph cut gives us a segmentation
  - What is a “good” graph cut and how do we find one?

Slide credit: S. Seitz

Minimum cut

A cut of a graph $G$ is the set of edges $S$ such that removal of $S$ from $G$ disconnects $G$.

Cut: sum of the weight of the cut edges:

$$\text{cut}(A, B) = \sum_{u \in A, v \in B} W(u, v),$$

with $A \cap B = \emptyset$

Slide credit: B. Freeman and A. Torralba

Minimum cut

- We can do segmentation by finding the minimum cut in a graph
  - Efficient algorithms exist for doing this

Minimum cut example

Slide credit: S. Lazebnik
Minimum cut

- We can do segmentation by finding the minimum cut in a graph
  - Efficient algorithms exist for doing this

**Minimum cut example**

![Graph example with minimum cut](image)

Slide credit: S. Lazebnik

Drawbacks of Minimum cut

- Weight of cut is directly proportional to the number of edges in the cut.

![Graph example with cuts](image)

Slide credit: B. Freeman and A. Torralba

Normalized cuts

Write graph as V, one cluster as A and the other as B

\[
\text{Ncut}(A, B) = \frac{\text{cut}(A, B)}{\text{assoc}(A, V)} + \frac{\text{cut}(A, B)}{\text{assoc}(B, V)}
\]

where:

- \( \text{cut}(A, B) \) is sum of weights with one end in A and one end in B
  \[
  \text{cut}(A, B) = \sum_{u \in A, v \in B} W(u, v), \quad \text{with } A \cap B = \emptyset
  \]

- \( \text{assoc}(A, V) \) is sum of all edges with one end in A.
  \[
  \text{assoc}(A, V) = \sum_{u \in A, v \in B} W(u, v)
  \]


Normalized cut

- Let \( W \) be the adjacency matrix of the graph
- Let \( D \) be the diagonal matrix with diagonal entries \( D(i, i) = \sum_j W(i, j) \)
- Then the normalized cut cost can be written as
  \[
  \frac{y^T(D - W)y}{y^T Dy}
  \]

where \( y \) is an indicator vector whose value should be 1 in the \( i \)th position if the \( i \)th feature point belongs to A and a negative constant otherwise.

**Normalized cut**

- Finding the exact minimum of the normalized cut cost is NP-complete, but if we relax $y$ to take on arbitrary values, then we can minimize the relaxed cost by solving the generalized eigenvalue problem $(D - W)y = \lambda Dy$.
- The solution $y$ is given by the generalized eigenvector corresponding to the second smallest eigenvalue.
- Intuitively, the $i$th entry of $y$ can be viewed as a “soft” indication of the component membership of the $i$th feature.
  - Can use 0 or median value of the entries as the splitting point (threshold), or find threshold that minimizes the Ncut cost.

**Global optimization**

- In this formulation, the segmentation becomes a global process.
- Decisions about what is a boundary are not local (as in Canny edge detector).

**Normalized cut algorithm**

1. Given an image or image sequence, set up a weighted graph $G = (V, E)$, and set the weight on the edge connecting two nodes being a measure of the similarity between the two nodes.
2. Solve $(D - W)x = \lambda Dx$ for eigenvectors with the smallest eigenvalues.
3. Use the eigenvector with second smallest eigenvalue to bipartition the graph.
4. Decide if the current partition should be sub-divided, and recursively repartition the segmented parts if necessary.

**Boundaries of image regions defined by a number of attributes**

- Brightness/color
- Texture
- Motion
- Stereoscopic depth
- Familiar configuration
Brightness Image Segmentation

N pixels = ncols * nrows

Figure 12: Subplot (1) plots the smallest eigenvectors of the generalized eigenvalue system (11). Subplot (2) - (9) shows the eigenvectors corresponding the 2nd smallest to the 9th smallest eigenvalues of the system. The eigenvectors are reshaped to be the size of the image.


Slide credit: B. Freeman and A. Torralba
Results on color segmentation

Example results

Results: Berkeley Segmentation Engine
Normalized cuts: Pro and con

- **Pros**
  - Generic framework, can be used with many different features and affinity formulations

- **Cons**
  - High storage requirement and time complexity
  - Bias towards partitioning into equal segments

Segmentation methods

- **Segmentation as clustering**
  - K-means clustering
  - Mean-shift segmentation

- **Graph-theoretic segmentation**
  - Min cut
  - Normalized cuts

- **Boundary detection**
Protocol

You will be presented a photographic image. Divide the image into some number of segments, where the segments represent “things” or “parts of things” in the scene. The number of segments is up to you, as it depends on the image. Something between 2 and 30 is likely to be appropriate. It is important that all of the segments have approximately equal importance.

- Custom segmentation tool
- Subjects obtained from work-study program (UC Berkeley undergraduates)
Segmentations are Consistent

- A, C are refinements of B
- A, C are mutual refinements
- A, B, C represent the same percept
  - Attention accounts for differences

Perceptual organization forms a tree:

- Two segmentations are consistent when they can be explained by the same segmentation tree (i.e., they could be derived from a single perceptual organization).

Dataset Summary

- 30 subjects, age 19-23
  - 17 men, 13 women
  - 9 with artistic training
- 8 months
- 1,458 person hours
- 1,020 Corel images
- 11,595 Segmentations
  - 5,555 color, 5,554 gray, 486 inverted/negated

Pb Detector

- Image → Boundary Cues → Cue Combination → Pb
  - Boundary Cues: Brightness, Color, Texture
  - Cue Combination

Challenges: texture cue, cue combination

Goal: learn the posterior probability of a boundary Pb from local information only.
Brightness and Color Features

- 1976 CIE $L^*a^*b^*$ colorspace
- Brightness Gradient (B)
  - $\chi^2$ difference in $L^*$ distribution
- Color Gradient (C)
  - $\chi^2$ difference in $a^*$ and $b^*$ distributions

$$\chi^2(g,h) = \frac{1}{2} \sum_i \frac{(g_i - h_i)^2}{g_i + h_i}$$

Texture Feature

- Texture Gradient (T)
- $\chi^2$ difference of texton histograms
  - Textons are vector-quantized filter outputs

Cue Combination Models

- Classification Trees
  - Top-down splits to maximize entropy, error bounded
- Density Estimation
  - Adaptive bins using k-means
- Logistic Regression, 3 variants
  - Linear and quadratic terms
  - Confidence-rated generalization of AdaBoost (Schapire&Singer)
- Hierarchical Mixtures of Experts (Jordan&Jacobs)
  - Up to 8 experts, initialized top-down, fit with EM
- Support Vector Machines (libsvm, Chang&Lin)

- Range over bias, complexity, parametric/non-parametric
Computing Precision/Recall

Recall = Pr(signal|truth) = fraction of ground truth found by the signal
Precision = Pr(truth|signal) = fraction of signal that is correct
- Always a trade-off between the two
- Standard measures in information retrieval (van Rijsbergen XX)
- ROC from standard signal detection the wrong approach

Strategy
- Detector output (Pb) is a soft boundary map
- Compute precision/recall curve:
  - Threshold Pb at many points t in [0,1]
  - Recall = Pr(Pb>t|seg=1)
  - Precision = Pr(seg=1|Pb>t)

Cue Calibration

- All free parameters optimized on training data
- All algorithmic alternatives evaluated by experiment

- Brightness Gradient
  - Scale, bin/kernel sizes for KDE
- Color Gradient
  - Scale, bin/kernel sizes for KDE, joint vs. marginals
- Texture Gradient
  - Filter bank: scale, multiscale?
  - Histogram comparison
  - Number of textons, Image-specific vs. universal textons
- Localization parameters for each cue
Findings

1. A simple linear model is sufficient for cue combination
   - All cues weighted approximately equally in logistic
2. Proper texture edge model is not optional for complex natural images
   - Texture suppression is not sufficient!
3. Significant improvement over state-of-the-art in boundary detection
4. Empirical approach critical for both cue calibration and cue combination