Grouping and Segmentation

CS 554 – Computer Vision
Pinar Duygulu
Bilkent University

(Source: Kristen Grauman)
Grouping in vision

• Goals:
  – Gather features that belong together
  – Obtain an intermediate representation that compactly describes key image or video parts
Examples of grouping in vision

Group video frames into shots

Determine image regions

Object-level grouping

Figure-ground

[Figure by Wang & Suter]

[Figure by Grauman & Darrell]

[Figure by J. Shi]
Grouping in vision

• Goals:
  – Gather features that belong together
  – Obtain an intermediate representation that compactly describes key image (video) parts

• Top down vs. bottom up segmentation
  – Top down: pixels belong together because they are from the same object
  – Bottom up: pixels belong together because they look similar

• Hard to measure success
  – What is interesting depends on the app.
Muller-Lyer illusion
What things should be grouped?
What cues indicate groups?
Gestalt

• 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)
Similarity
Symmetry

Common fate

Image credit: Arthus-Bertrand (via F. Durand)
Proximity
A “simple” segmentation problem
It can get a lot harder

Continuity, explanation by occlusion
Magritte, 1957
Groupings by Invisible Completions

* Images from Steve Lehar’s Gestalt papers
1970s: R. C. James
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”

Gestalt

• 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)

• Inspiring observations/explanations; challenge remains how to best map to algorithms.
Familiar configuration
Familiarity
Familiarity
Influences of grouping

Grouping influences other perceptual mechanisms such as lightness perception

Figure-ground
Grouping phenomena in real life

Forsyth & Ponce, Figure 14.7
Grouping phenomena in real life

Forsyth & Ponce, Figure 14.7
The goals of segmentation

- Separate image into coherent “objects”

Source: Lana Lazebnik
The goals of segmentation

• Separate image into coherent “objects”

• Group together similar-looking pixels for efficiency of further processing

“superpixels”


Source: Lana Lazebnik
These intensities define the three groups.
We could label every pixel in the image according to which of these primary intensities it is.
    i.e., segment the image based on the intensity feature.
What if the image isn’t quite so simple?
Now how to determine the three main intensities that define our groups?

We need to cluster.
Goal: choose three “centers” as the representative intensities, and label every pixel according to which of these centers it is nearest to.

Best cluster centers are those that minimize SSD between all points and their nearest cluster center $c_i$:

$$\sum_{\text{clusters } i} \sum_{\text{points } p \text{ in cluster } i} \|p - c_i\|^2$$
Clustering

• With this objective, it is a “chicken and egg” problem:
  – If we knew the **cluster centers**, we could allocate points to groups by assigning each to its closest center.
  – If we knew the **group memberships**, we could get the centers by computing the mean per group.
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:

\[
\sum_{\text{clusters } i} \sum_{\text{points } p \text{ in cluster } i} ||p - c_i||^2
\]
K-means

1. Ask user how many clusters they’d like. *(e.g. k=5)*
K-means

1. Ask user how many clusters they’d like. (*e.g.* $k=5$)

2. Randomly guess $k$ cluster Center locations
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)
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.

4. Each Center finds the centroid of the points it owns
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.
4. Each Center finds the centroid of the points it owns...
5. ...and jumps there
6. ...Repeat until terminated!

Andrew Moore
K-means clustering

• Java demo:
  http://kovan.ceng.metu.edu.tr/~maya/kmeans/index.html
  http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/AppletKM.html
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:

  - 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)
quantization of the feature space; segmentation label map
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)

Kristen Grauman
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

Clusters based on intensity similarity don’t have to be spatially coherent.
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)
Recall: texture representation example

Windows with primarily horizontal edges

Windows with primarily vertical edges

Windows with small gradient in both directions

Both

Dimension 1 (mean d/dx value)

Dimension 2 (mean d/dy value)

statistics to summarize patterns in small windows
Segmentation with texture features

• Find “textons” by **clustering** vectors of filter bank outputs

• Describe texture in a window based on **texton histogram**


Adapted from Lana Lazebnik
Image segmentation example
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.
Material classification example

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

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

Manik Varma
http://www.robots.ox.ac.uk/~vgg/research/texclass/with.html
Outline

• What are grouping problems in vision?

• Inspiration from human perception
  – Gestalt properties

• Bottom-up segmentation via clustering
  – Algorithms:
    • Mode finding and mean shift: k-means, mean-shift
    • Graph-based: normalized cuts
  – Features: color, texture, ...
    • Quantization for texture summaries
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
Mean shift algorithm

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

Image

Feature space
(L* u* v* color values)
Mean shift
Mean shift

Search window
Center of mass
Mean Shift vector

Slide by Y. Ukrainitz & B. Sarel
Mean shift

Search window
Center of mass
Mean Shift vector

Slide by Y. Ukrainitz & B. Sarel
Mean shift

Search window
Center of mass
Mean Shift vector

Slide by Y. Ukrainitz & B. Sarel
Mean shift

Search window

Center of mass

Mean Shift vector

Slide by Y. Ukrainitz & B. Sarel
Mean shift

Search window

Center of mass

Mean Shift vector

Slide by Y. Ukrainitz & B. Sarel
Mean shift

Search window

Center of mass

Slide by Y. Ukrainitz & B. Sarel
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
Mean shift segmentation results

http://www.caip.rutgers.edu/~comanici/MSPAMI/msPamiResults.html
Mean shift segmentation results
Mean shift

• **Pros:**
  – Does not assume shape on clusters
  – One parameter choice (window size)
  – Generic technique
  – Find multiple modes

• **Cons:**
  – Selection of window size
  – Does not scale well with dimension of feature space

Kristen Grauman
Outline

• What are grouping problems in vision?

• Inspiration from human perception
  – Gestalt properties

• Bottom-up segmentation via clustering
  – Algorithms:
    • Mode finding and mean shift: k-means, mean-shift
    • Graph-based: normalized cuts
  – Features: color, texture, ...
    • Quantization for texture summaries
Images as graphs

• **Fully-connected** graph
  – node (vertex) for every pixel
  – link between *every* pair of pixels, p,q
  – affinity weight $w_{pq}$ for each link (edge)
    • $w_{pq}$ measures *similarity*
      – similarity is *inversely proportional* to difference (in color and position...)

Source: Steve Seitz
Measuring affinity

• One possibility:

\[ \text{aff}(x, y) = \exp\left\{ -\left(\frac{1}{2\sigma^2_d}\right)(\|x - y\|^2) \right\} \]

Small sigma: group only nearby points

Large sigma: group distant points
Measuring affinity

Data points

Affinity matrices

σ=.1

σ=.2

σ=1
Segmentation by Graph Cuts

- Break Graph into Segments
  - Want to delete links that cross *between* segments
  - Easiest to break links that have low similarity (low weight)
    - similar pixels should be in the same segments
    - dissimilar pixels should be in different segments

Source: Steve Seitz
Cuts in a graph: Min cut

• Link Cut
  – set of links whose removal makes a graph disconnected
  – cost of a cut:

\[
cut(A, B) = \sum_{p \in A, q \in B} w_{p,q}
\]

Find minimum cut
  • gives you a segmentation
  • fast algorithms exist for doing this

Source: Steve Seitz
Minimum cut

- Problem with minimum cut:
  Weight of cut proportional to number of edges in the cut; tends to produce small, isolated components.

Fig. 1. A case where minimum cut gives a bad partition.

[Shi & Malik, 2000 PAMI]
Cuts in a graph: Normalized cut

Normalized Cut

- fix bias of Min Cut by normalizing for size of segments:

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

\(assoc(A, V) = \text{sum of weights of all edges that touch A}\)

- Ncut value small when we get two clusters with many edges with high weights, and few edges of low weight between them
- Approximate solution for minimizing the Ncut value: generalized eigenvalue problem.

Source: Steve Seitz

J. Shi and J. Malik, Normalized Cuts and Image Segmentation, CVPR, 1997
Example results
Results: Berkeley Segmentation Engine

http://www.cs.berkeley.edu/~fowlkes/BSE/
Normalized cuts: pros and cons

**Pros:**
- Generic framework, flexible to choice of function that computes weights ("affinities") between nodes
- Does not require model of the data distribution

**Cons:**
- Time complexity can be high
  - Dense, highly connected graphs \(\rightarrow\) many affinity computations
  - Solving eigenvalue problem
- Preference for balanced partitions
Segments as primitives for recognition

B. Russell et al., “Using Multiple Segmentations to Discover Objects and their Extent in Image Collections,” CVPR 2006

Slide credit: Lana Lazebnik
Top-down segmentation


Slide credit: Lana Lazebnik
Top-down segmentation

Normalized cuts

Top-down segmentation


Kristen Grauman
Image grouping

Kristen Grauman