Segmentation and Grouping
April 21st, 2020
Yong Jae Lee
UC Davis

Features and filters
Transforming and describing images; textures, edges

Grouping and fitting
Clustering, segmentation, fitting; what parts belong together?
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

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

Determine image regions

Group video frames into shots

Figure-ground

Object-level grouping
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 a set of elements to be grouped (by human visual system)

Slide credit: Kristen Grauman

Figure 14.4 from Forsyth and Ponce
Gestalt

Similarity

Symmetry
Common fate

Proximity

Illusory contours

Interesting tendency to explain by occlusion

Image credit: Arthus-Bertrand (via F. Durand)

Slide credit: Kristen Grauman

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

Slide credit: Kristen Grauman

In Vision, D. Marr, 1982
Grouping phenomena in real life

Forsyth & Ponce, Figure 14.7

Slide credit: Kristen Grauman

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.

Slide credit: 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

The goals of segmentation

Separate image into coherent “objects”

Source: Lana Lazebnik

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.
**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:
  $$\sum_{i} \sum_{p \in \text{cluster } i} ||p - c_i||^2$$
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)
K-means clustering

- Demo

[Link to Demo](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

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)
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)

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.

Slide credit: Kristen Grauman

Segmentation as clustering

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

Slide credit: 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 **texture** similarity

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

Slide credit: Kristen Grauman
Recall: texture representation example

Original image

Derivative filter responses, squared

<table>
<thead>
<tr>
<th>Win. #1</th>
<th>Win. #2</th>
<th>Win. #9</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td>18</td>
<td>7</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Statistics to summarize patterns in small windows

--

Recall: texture representation example

Windows with primarily horizontal edges

Windows with small gradient in both directions

Windows with primarily vertical edges

--

Dimension 1 (mean d/dx value)

Dimension 2 (mean d/dy value)

Both

Segmentation with texture features

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

**Image segmentation example**

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

**Color vs. texture**

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.

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

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

Mean shift

![](image)

Mean shift vector

Search window

Center of mass

Mean shift

Search window

Center of mass

Mean shift vector
Mean shift

- Search window
- Center of mass
- Mean Shift vector

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

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
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…)

Measuring affinity

• One possibility:

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

Small sigma: group only nearby points
Large sigma: group distant points

Slide credit: Kristen Grauman
Measuring affinity

Data points

Affinity matrices

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

Cuts in a graph: Min cut

Link Cut
- set of links whose removal makes a graph disconnected
- cost of a cut:
  \[ \text{cut}(A, B) = \sum_{p \in A, q \in B} w_{pq} \]

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

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

![Minimum cut diagram](image)

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) = sum of weights of all edges in A to all nodes V

- 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.

Example results
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

Motion segmentation

Summary
- Segmentation to find object boundaries or mid-level regions, tokens.
- Bottom-up segmentation via clustering
  - General choices -- features, affinity functions, and clustering algorithms
- Grouping also useful for quantization, can create new feature summaries
  - Texton histograms for texture within local region
- Example clustering methods
  - K-means
  - Mean shift
  - Graph cut, normalized cuts

Questions?

See you Thursday!