Grouping
What is grouping?
Why grouping?

• Pixels property of sensor, not world
• Reasoning at object level (might) make things easy:
  • objects at consistent depth
  • objects can be recognized
  • objects move as one
The gradient points in the direction of most rapid increase in intensity

\[ \nabla f = \left[ \frac{\partial f}{\partial x}, 0 \right] \]

The edge strength is given by the gradient magnitude:

\[ ||\nabla f|| = \sqrt{\left( \frac{\partial f}{\partial x} \right)^2 + \left( \frac{\partial f}{\partial y} \right)^2} \]

The gradient direction is given by:

\[ \theta = \tan^{-1} \left( \frac{\partial f/\partial y}{\partial f/\partial x} \right) \]

• how does this relate to the direction of the edge?
Gradient magnitude and orientation

• Orientation is undefined at pixels with 0 gradient

\[
\theta = \text{numpy.arctan2}(g_y, g_x)
\]
Non-maximum suppression for each orientation

At q, we have a maximum if the value is larger than those at both p and at r. Interpolate to get these values.

Source: D. Forsyth
Before Non-max Suppression
After Non-max Suppression
Image gradients are not enough

Hard-to-detect low contrast boundaries

Strong internal image gradients
Image gradients are not enough
What is texture?

Same thing repeated over and over
What is texture?
Julesz’s texton theory

• What is texture?
• Distributions of some elements
  • Elongated blobs of specific orientations, widths, lengths
  • Terminators (ends of line segments)
  • Crossings of line segments
Bringing textons to computer vision

- Define a “vocabulary” of textons
- Describe texture by a distribution of different textons
Bringing textons to computer vision

- Define a “vocabulary” of textons
- Describe texture by a distribution of different textons

Convolve with filter bank

Filter responses
Bringing textons to computer vision

• Define a “vocabulary” of textons
• Describe texture by a distribution of different textons
Bringing textons to computer vision

• Define a “vocabulary” of textons
• Describe texture by a distribution of different textons
Bringing textons to computer vision

• Define a “vocabulary” of textons

• **Describe texture by a distribution of different textons**

Textons in computer vision

Image ➔ Convolve with filter bank ➔ Assign to k-means centers ➔ Compute histogram

Image ➔ Convolution ➔ Point-wise non-linearity ➔ Avg pooling
Detecting texture boundaries

• Problem: gradient captures change from pixel to pixel

• But texture property of region

• Take region around pixel and divide into two halves based on hypothetical orientation

Cue combination

Brighten, Color, Texture

Average

Local computation not enough
Local computation is not enough

• Key constraints:
  • Boundaries are continuous
  • They enclose a region

• How do we go from local, patchy contours to boundaries?
Grouping by clustering

- Idea: embed pixels into high-dimensional space (e.g. 3-dimensions)
- Each pixel is a point
- Group together points
K-means

• Assumption: each group is a Gaussian with different means and same standard deviation

\[ P(x_i | \mu_j) \propto e^{-\frac{1}{2\sigma^2} \| x_i - \mu_j \|^2} \]

• Suppose we know all \( \mu_j \). Which group should a point \( x_i \) belong to?
  • The \( j \) with highest \( P(x_i | \mu_j) \)
  • = The \( j \) with smallest \( \| x_i - \mu_j \|^2 \)
K-means

- Problem: means are not known
- What if we know a set of points from each cluster?
  - $x_{k_1}, x_{k_2}, \ldots, x_{k_n}$ belong to cluster $k$
- What should be $\mu_k$?

$$
\mu_k = \frac{(x_{k_1} + x_{k_2} + \ldots + x_{k_n})}{n}
$$
K-means

• Given means, can assign points to clusters
• Given assignments, can compute means
• Idea: iterate!
K-means

- Step-1: randomly pick k centers
K-means

• Step 2: Assign each point to nearest center
K-means

- Step 3: re-estimate centers
K-means

• Step 4: Repeat
K-means

• Step 4: Repeat
K-means

• Step 4: Repeat
K-means on image pixels
K-means on image pixels

Picture courtesy David Forsyth

One of the clusters from k-means
K-means on image pixels+position

- Groups pixels together, but does not produce compact regions
Segmentation is graph partitioning
Segmentation is graph partitioning

- Every partition “cuts” some edges
- Idea: minimize total weight of edges cut!
Criterion: Min-cut?

• Min-cut carves out small isolated parts of the graph
• In image segmentation: individual pixels
Normalized cuts

• “Cut” = total weight of cut edges
• Small cut means the groups don’t “like” each other
• But need to normalize w.r.t how much they like themselves
• “Volume” of a subgraph = total weight of edges within the subgraph
Normalized cut

\[ \frac{\text{cut}(A, \bar{A})}{\text{vol}(A)} + \frac{\text{cut}(A, \bar{A})}{\text{vol}(\bar{A})} \]
Min-cut vs normalized cut

• Both rely on interpreting images as graphs
• By itself, min-cut gives small isolated pixels
  • But can work if we add other constraints
• min-cut can be solved in polynomial time
  • Dual of max-flow
• N-cut is NP-hard
  • But approximations exist!
Graphs and matrices

- $w(i,j) = \text{weight between } i \text{ and } j \text{ (Affinity matrix)}$
- $d(i) = \text{degree of } i = \sum_j w(i, j)$
- $D = \text{diagonal matrix with } d(i) \text{ on diagonal}$
Graphs and matrices
Graphs and matrices

$$E_{ij} = \frac{w_{ij}}{\sum_k w_{ik}}$$

$$E = D^{-1}W$$
Graphs and matrices

- How do we represent a clustering?
- A label for N nodes
  - 1 if part of cluster A, 0 otherwise
- An N-dimensional vector!

\[
\begin{array}{c|c|c|c|c|c|c|c|c|c}
& 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\
\hline
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
2 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\
3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

\[v_1\]
Graphs and matrices

• How do we represent a clustering?
• A label for N nodes
  • 1 if part of cluster A, 0 otherwise
• An N-dimensional vector!

<table>
<thead>
<tr>
<th></th>
<th>v₁</th>
<th>v₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Graphs and matrices

• How do we represent a clustering?
• A label for $N$ nodes
  • 1 if part of cluster A, 0 otherwise
• An $N$-dimensional vector!

<table>
<thead>
<tr>
<th></th>
<th>$v_1$</th>
<th>$v_2$</th>
<th>$v_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0:</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1:</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2:</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3:</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4:</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5:</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6:</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>7:</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8:</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9:</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Graphs and matrices

\[ E = D^{-1}W \]
Graphs and matrices

\[ E = D^{-1}W \]

\[ E_{ij} = \frac{w_{ij}}{\sum_k w_{ik}} \]
Graphs and matrices

$$E = D^{-1}W$$

$$E_{ij} = \frac{w_{ij}}{\sum_k w_{ik}}$$
Graphs and matrices

\[ E = D^{-1}W \]

\[ E_{ij} = \frac{w_{ij}}{\sum_k w_{ik}} \]
Graphs and matrices

\[ E = D^{-1}W \]
Graphs and matrices

\[ E = D^{-1}W \]

\[ E_{ij} = \frac{w_{ij}}{\sum_k w_{ik}} \]
Graphs and matrices

\[ D^{-1} W y \approx y \]

Define \( z \) so that

\[ y = D^{-\frac{1}{2}} z \]

\[ D^{-1} W D^{-\frac{1}{2}} z \approx D^{-\frac{1}{2}} z \]

\[ \Rightarrow D^{-\frac{1}{2}} W D^{-\frac{1}{2}} z \approx z \]

\[ \Rightarrow (I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}})z \approx 0 \]
Graphs and matrices

\[
\Rightarrow (I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}})z \approx 0 \\
\Rightarrow \mathcal{L}z \approx 0
\]

\[
\mathcal{L} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}
\]

is called the
Normalized Graph Laplacian
Graphs and matrices

\[ \mathcal{L} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}} \]

• We want \( \mathcal{L} z \approx 0 \)
• Trivial solution: all nodes of graph in one cluster, nothing in the other
• To avoid trivial solution, look for the eigenvector with the second smallest eigenvalue

\[ \mathcal{L} z = \lambda z \]
\[ \lambda_1 < \lambda_2 < \ldots < \lambda_N \]

• Find \( z \) s.t. \( \mathcal{L} z = \lambda_2 z \)
Normalized cuts

• Approximate solution to normalized cuts
• Construct matrix $W$ and $D$
• Construct normalized graph laplacian
  \[ \mathcal{L} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}} \]
• Look for the second smallest eigenvector
  \[ \mathcal{L} z = \lambda_2 z \]
• Compute \( y = D^{-\frac{1}{2}} z \)

**Threshold $y$ to get clusters**
  • Ideally, sweep threshold to get lowest N-cut value
Eigenvectors of images

- The eigenvector has as many components as pixels in the image
Eigenvalues of images

- The eigenvector has as many components as pixels in the image.
Another example

2\textsuperscript{nd} eigenvector

3\textsuperscript{rd} eigenvector

4\textsuperscript{th} eigenvector
Eigenvectors of images
How do we group things?

- *Gestalt* principles
- Principle of *proximity*

https://courses.lumenlearning.com/wsu-sandbox/chapter/gestalt-principles-of-perception/
How do we group things?

• Gestalt principles
• Principle of *similarity*
How do we group things?

• Gestalt principles
• Principle of *continuity* and *closure*

https://courses.lumenlearning.com/wsu-sandbox/chapter/gestalt-principles-of-perception/
How do we group things?

• Gestalt principles
• Principle of *common fate*
Gestalt principles in the context of images

- Principle of proximity: nearby pixels are part of the same object
- Principle of similarity: similar pixels are part of the same object
  - Look for differences in color, intensity, or texture across the boundary
- Principle of closure and continuity: contours are likely to continue
- High-level knowledge?