Announcements

• Please don’t forget to register for the exam!
  - On the Campus system

Course Outline

• Image Processing Basics
• Segmentation
  - Segmentation and Grouping
  - Graph-Theoretic Segmentation
• Recognition
  - Global Representations
  - Subspace representations
• Local Features & Matching
• Object Categorization
• 3D Reconstruction
• Motion and Tracking

Recap: Gestalt Theory

• 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)

Recap: Image Segmentation

• Goal: identify groups of pixels that go together

Recap: Gestalt Factors

- These factors make intuitive sense, but are very difficult to translate into algorithms.

Recap: Gestalt Theory

- 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)
Recap: 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:
    \[
    \sum_{\text{clusters } i} \sum_{\text{points } p \text{ in cluster } i} ||p - c_i||^2
    \]

Recap: Expectation Maximization (EM)

- Goal
  - Find blob parameters \( \theta \) that maximize the likelihood function:
    \[
    P(\text{data}|\theta) = \prod_i P(x|\theta)
    \]
- Approach:
  1. E-step: given current guess of blobs, compute ownership of each point
  2. M-step: given ownership probabilities, update blobs to maximize likelihood function
  3. Repeat until convergence

Recap: Mean-Shift Algorithm

- Iterative Mode Search
  1. Initialize random seed, and window \( W \)
  2. Calculate center of gravity (the “mean”) of \( W \):
    \[
    \sum_{x \in W} \frac{x}{H(x)}
    \]
  3. Shift the search window to the mean
  4. Repeat Step 2 until convergence

Recap: 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

Recap: Mean-Shift Segmentation

- Find features (color, gradients, texture, etc)
- Initialize windows at individual pixel locations
- Perform mean shift for each window until convergence
- Merge windows that end up near the same “peak” or mode

Back to the Image Segmentation Problem...

- Goal: identify groups of pixels that go together
- Up to now, we have focused on ways to group pixels into image segments based on their appearance...
  - Segmentation as clustering.
  - We also want to enforce region constraints.
    - Spatial consistency
    - Smooth borders
Topics of This Lecture

- Graph theoretic segmentation
  - Normalized Cuts
  - Using texture features
  - Extension: Multi-level segmentation
- Segmentation as Energy Minimization
  - Markov Random Fields
  - Graph cuts for image segmentation
  - Applications

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 (e.g., in color and position...)

Segmentation by Graph Cuts

- Break Graph into Segments
  - Delete links that cross between segments
  - Easiest to break links that have low similarity (low weight)
  - Dissimilar pixels should be in different segments

Measuring Affinity

- Distance \(\text{aff}(x,y) = \exp\left(-\frac{d(x-y)^2}{\sigma^2}\right)\)
- Intensity \(\text{aff}(x,y) = \exp\left(-\frac{d(I(x)-I(y))^2}{\sigma^2}\right)\)
- Color \(\text{aff}(x,y) = \exp\left(-\frac{d \text{dist}(c(x),c(y))^2}{\sigma^2}\right)\)
- Texture \(\text{aff}(x,y) = \exp\left(-\frac{d f(x)-f(y))^2}{\sigma^2}\right)\)

Scale Affects Affinity

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

Graph Cut

- Set of edges whose removal makes a graph disconnected
- Cost of a cut
  - Sum of weights of cut edges: \(\text{cut}(A,B) = \sum_{p \in A, q \in B} w_{pq}\)
- A graph cut gives us a segmentation
  - What is a “good” graph cut and how do we find one?
Here, the cut is nicely defined by the block-diagonal structure of the affinity matrix. ⇒ How can this be generalized?

Minimum Cut

- We can do segmentation by finding the minimum cut in a graph
- Efficient algorithms exist for doing this
- Drawback:
  - Weight of cut proportional to number of edges in the cut
  - Minimum cut tends to cut off very small, isolated components

Cuts with lesser weight than the ideal cut

Ideal Cut

Ideal Cut

Normalized Cut (NCut)

- A minimum cut penalizes large segments
- This can be fixed by normalizing for size of segments
- The normalized cut cost is:

\[
\text{NCut}(A, B) = \frac{\text{cut}(A, B)}{\text{assoc}(A, V)} \left( \frac{\text{cut}(A, B)}{\text{assoc}(B, V)} \right)
\]

\[\text{assoc}(A, V) = \sum_{j \in V} W_{ij}, \quad \text{assoc}(B, V) = \sum_{j \in V} W_{kj} \]

- The exact solution is NP-hard but an approximation can be computed by solving a generalized eigenvector problem.

\[\text{J. Shi and J. Malik. Normalized cuts and image segmentation. PAMI 2000} \]

NCuts as a Generalized Eigenvector Problem

- Definitions
  - \( W \): the affinity matrix, \( W(i, j) = w_{ij} \)
  - \( D \): the diag. matrix, \( D(i, i) = \sum_j W(i, j) \)
  - \( x \): a vector in \([1, -1]^N\), \( x(i) = 1 \iff i \in A \)

- Rewriting Normalized Cut in matrix form:

\[
\text{NCut}(A, B) = \frac{\text{cut}(A, B)}{\text{assoc}(A, V)} \left( \frac{\text{cut}(A, B)}{\text{assoc}(B, V)} \right)
\]

\[= \frac{(1+x)^T(D-W)(1+x)}{1+x} \left( \frac{(1-x)^T(D-W)(1-x)}{1-x} \right) \]

\[= \ldots \]

Some More Math…

- Interpretation as a Dynamical System
  - Treat the links as springs and shake the system
    - Elasticity proportional to cost
    - Vibration “modes” correspond to segments
    - Can compute these by solving a generalized eigenvector problem

Slide credit: Steve Seitz

Slide credit: Jitendra Malik

Slide credit: Svetlana Lazebnik

Slide credit: Khurram Hassan - Shafique

Slide credit: Forsyth & Ponce

Slide credit: Steve Seitz

Slide credit: Jitendra Malik

Slide credit: Svetlana Lazebnik

Slide credit: Jitendra Malik.
NCuts as a Generalized Eigenvalue Problem

- After simplification, we get
  \[ NC(i, j) = \frac{y^T(D-W)y}{y^TDy} \]
  with \( y \in [L-B], y^TDy = 0 \).
  
- This is a so-called Rayleigh Quotient
  
  Solution given by the "generalized" eigenvalue problem
  \( (D-W)y = \lambda Dy \)
  
  Solved by converting to standard eigenvalue problem
  \( D^{-1}(D-W)D^2z = \lambda z \), where \( z = D^2y \)
  
- Subtleties
  - Optimal solution is second smallest eigenvector
  - Gives continuous result—must convert into discrete values of \( y \)

Discretization

- Problem: eigenvectors take on continuous values
  
  How to choose the splitting point to binarize the image?

- Possible procedures
  a) Pick a constant value (0, or 0.5).
  b) Pick the median value as splitting point.
  c) Look for the splitting point that has the minimum NCut value:
     1. Choose a possible splitting points.
     2. Compute NCut value.
     3. Pick minimum.

NCuts Example

- Smallest eigenvectors

NCuts: Overall Procedure

1. Construct a weighted graph \( G=(V,E) \) from an image.
2. Connect each pair of pixels, and assign graph edge weights
   \( W(i,j) = \text{Prob. that } i \text{ and } j \text{ belong to the same region.} \)
3. Solve \( (D-W)y = \lambda Dy \) for the smallest few eigenvectors. This yields a continuous solution.
4. Threshold eigenvectors to get a discrete cut
   
   This is where the approximation is made (we’re not solving NP).
5. Recursively subdivide if NCut value is below a pre-specified value.

NCuts Matlab code available at
  http://www.cis.upenn.edu/~jshi/software/

Color Image Segmentation with NCuts

- Texture descriptor is vector of filter bank outputs

Using Texture Features for Segmentation

- Texture descriptor is vector of filter bank outputs
Using Texture Features for Segmentation

- Texture descriptor is vector of filter bank outputs.
- Textons are found by clustering.

Summary: Normalized Cuts

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

- **Cons:**
  - Time and memory complexity can be high
  - Dense, highly connected graphs ⇒ many affinity computations
  - Solving eigenvalue problem for each cut
  - Preference for balanced partitions
  - If a region is uniform, NCuts will find the modes of vibration of the image dimensions

Results with Color & Texture

Example segmentations for several contrasts

- It is often difficult to extract a single good segmentation
  > Idea: Extract a hierarchy of segmentations instead

Extension: Multi-Level Segmentation

Multiscale Segmentation Tree

- Segments can be arranged in a tree

Sample cutsets
Topics of This Lecture

- Graph theoretic segmentation
  - Normalized Cuts
  - Using color and texture features
  - Extension: Multi-level segmentation
- Segmentation as Energy Minimization
  - Markov Random Fields
  - Graph cuts for image segmentation
  - Applications

Markov Random Fields

- Allow rich probabilistic models for images
- But built in a local, modular way
  - Learn local effects, get global effects out

MRF Nodes as Pixels

- Original image
- Degraded image
- Reconstruction from MRF modeling pixel neighborhood statistics

MRF Nodes as Patches

- Image patches
- Scene patches

Network Joint Probability

\[ P(x, y) = \prod_i \Phi(x_i, y_i) \prod_{i,j} \Psi(x_i, x_j) \]

Energy Formulation

- Joint probability
  \[ P(x, y) = \prod_i \Phi(x_i, y_i) \prod_{i,j} \Psi(x_i, x_j) \]
- Maximizing the joint probability is the same as minimizing the negative log
  \[ \log P(x, y) = \sum_i \log \Phi(x_i, y_i) + \sum_{i,j} \log \Psi(x_i, x_j) \]
  \[ -E(x, y) = \sum_i \phi(x_i, y_i) + \sum_{i,j} \psi(x_i, x_j) \]
- This is similar to free-energy problems in statistical mechanics (spin glass theory). We therefore draw the analogy and call \( E \) an energy function.
- \( \phi \) and \( \psi \) are called potentials.
Energy Formulation

- **Energy function**
  \[ -E(x,y) = \sum_x \phi(x,y) + \sum_{i,j} \psi(x_i,y_i,x_j,y_j) \]

  - **Single-node potentials** \( \phi \)
    - Encode local information about the given pixel/patch
    - How likely is a pixel/patch to belong to a certain class (e.g. foreground/background)?
  - **Pairwise potentials** \( \psi \)
    - Encode neighborhood information
    - How different is a pixel/patch’s label from that of its neighbor? (e.g. based on intensity/color/texture difference, edges)

Energy Minimization

- **Goal:** Infer the optimal labeling of the MRF.
- **Many inference algorithms are available,** e.g.
  - Gibbs sampling, simulated annealing
  - Iterated conditional modes (ICM)
  - Variational methods
  - Belief propagation
  - Graph cuts
- **Recently, Graph Cuts have become a popular tool**
  - Only suitable for a certain class of energy functions
  - But the solution can be obtained very fast for typical vision problems (~1MPixel/sec).

Topics of This Lecture

- **Graph theoretic segmentation**
  - Normalized Cuts
  - Using color and texture features
  - Extension: Multi-level segmentation
- **Segmentation as Energy Minimization**
  - Markov Random Fields
  - Graph cuts for image segmentation
  - Applications

Graph Cuts for Optimal Boundary Detection

- **Idea:** convert MRF into source-sink graph

## Simple Example of Energy

- **Regional term**
  \[ E(L) = \sum_P D_p(L_p) \]
  - \( D_p \) represents the data term for the pixel/patch

- **Boundary term**
  \[ \sum_{pq} w_{pq} \delta(L_p \neq L_q) \]
  - \( w_{pq} \) is the weight for the edge between pixels/patches

**Boundary term**

\[ L_p \in \{s,t\} \] (binary object segmentation)

Adding Regional Properties

- **Regional bias example**
  - Suppose \( I_0' \) and \( I_1' \) are given “expected” intensities of object and background
  - **Regional bias**
    - For object: \( D_p(I_0') \approx \exp\left(-\frac{||L_p - I_0'||^2}{2\sigma^2}\right) \)
    - For background: \( D_p(I_1') \approx \exp\left(-\frac{||L_p - I_1'||^2}{2\sigma^2}\right) \)

**NOTE:** hard constrains are not required, in general.
Adding Regional Properties

- More generally, regional bias can be based on any intensity models of object and background

\[ D_s(s) = \exp \left( - \frac{||I_s - I_{s'}||^2}{2\sigma^2} \right) \]
\[ D_t(t) = \exp \left( - \frac{||I_t - I_{t'}||^2}{2\sigma^2} \right) \]

given object and background intensity histograms

EM-style optimization

Adding Regional Properties

- “expected” intensities of object and background can be re-estimated

How to Set the Potentials? Some Examples

- Color potentials
  - e.g. modeled with a Mixture of Gaussians
  \[ \pi(x, y; \theta) = \log \sum_i \theta_i(x, k) P(k | x) N(y; \mu_i, \Sigma_i) \]

- Edge potentials
  - e.g. a “contrast sensitive Potts model”
  \[ \phi(x, y; \theta) = \frac{1}{2} \sum_i g_i(y) \delta(x_i \neq x_j) \]
  \[ g_i(y) = \exp \left( - \beta \cdot \frac{||y - \bar{y}||}{\bar{r}} \right) \]

- Parameters \( \theta_x, \theta_y \) need to be learned, too!

When Can s-t Graph Cuts Be Applied?

- s-t graph cuts can only globally minimize binary energies that are submodular.

GraphCut Applications: “GrabCut”

- Interactive Image Segmentation [Boykov & Jolly, ICCV'01]
  - Rough region cues sufficient
  - Segmentation boundary can be extracted from edges

- Procedure
  - User marks foreground and background regions with a brush.
  - This is used to create an initial segmentation which can then be corrected by additional brush strokes.
Perceptual and Sensory Augmented Computing

GrabCut: Data Model

- Obtained from interactive user input
  - User marks foreground and background regions with a brush
  - Alternatively, user can specify a bounding box

GrabCut: Coherence Model

- An object is a coherent set of pixels:
  \[ \psi(x, y) = \gamma \sum_{(m, n) \in C} \delta(x_m \neq x_m) A_f(x_m) \]

Iterated Graph Cuts

- Obtained from interactive user input
  - User marks foreground and background regions with a brush
  - Alternatively, user can specify a bounding box

GrabCut: Example Results

- This is included in the newest version of MS Office!

Applications: Interactive 3D Segmentation

- Problem: Images contain many pixels
  - Even with efficient graph cuts, an MRF formulation has too many nodes for interactive results.
- Efficiency trick: Superpixels
  - Group together similar-looking pixels for efficiency of further processing.
  - Cheap, local oversegmentation
  - Important to ensure that superpixels do not cross boundaries
- Several different approaches possible
  - Superpixel code available here
Summary: Graph Cuts Segmentation

**Pros**
- Powerful technique, based on probabilistic model (MRF).
- Applicable for a wide range of problems.
- Very efficient algorithms available for vision problems.
- Becoming a de-facto standard for many segmentation tasks.

**Cons/Issues**
- Graph cuts can only solve a limited class of models
  - Submodular energy functions
- Can capture only part of the expressiveness of MRFs
- Only approximate algorithms available for multi-label case

Segmentation: Caveats

- We’ve looked at bottom-up ways to segment an image into regions, yet finding meaningful segments is intertwined with the recognition problem.
- Often want to avoid making hard decisions too soon
- Difficult to evaluate; when is a segmentation successful?
  - Often depends on the rest of the system pipeline.

References and Further Reading

- Background information on Normalized Cuts can be found in Chapter 14 of
  D. Forsyth, J. Ponce,
  *Computer Vision - A Modern Approach*.
  Prentice Hall, 2003

- Try the NCuts Matlab code at
  [http://www.cis.upenn.edu/~jshi/software/](http://www.cis.upenn.edu/~jshi/software/)

- Try the GraphCut implementation at
  [http://www.adastral.ucl.ac.uk/~vladkolm/software.html](http://www.adastral.ucl.ac.uk/~vladkolm/software.html)