Computer Vision - Lecture 7

Graph-Theoretic Segmentation

15.11.2012

Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de

leibe@vision.rwth-aachen.de
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: Hough Transform

- How can we use this to find the most likely parameters $(m, b)$ for the most prominent line in the image space?
  - Let each edge point in image space vote for a set of possible parameters in Hough space
  - Accumulate votes in discrete set of bins; parameters with the most votes indicate line in image space.
Recap: Hough Transform

- Issues with usual \((m,b)\) parameter space: can take on infinite values, undefined for vertical lines.

\[
x \cos \theta - y \sin \theta = d
\]

\(d\) : perpendicular distance from line to origin

\(\theta\) : angle the perpendicular makes with the x-axis

- Point in image space \(\Rightarrow\) sinusoid segment in Hough space

Slide credit: Steve Seitz
Recap: Hough Transform for Circles

- **Circle:** center \((a, b)\) and radius \(r\)

\[
(x_i - a)^2 + (y_i - b)^2 = r^2
\]

- **For an unknown radius** \(r\), unknown gradient direction

![Diagram of image space and Hough space with a circle and corresponding Hough space representation.](image-url)
Recap: Generalized Hough Transform

- What if want to detect arbitrary shapes defined by boundary points and a reference point?

At each boundary point, compute displacement vector: \( r = a - p_i \).

For a given model shape: store these vectors in a table indexed by gradient orientation \( \theta \).

[Dana H. Ballard, Generalizing the Hough Transform to Detect Arbitrary Shapes, 1980]

Slide credit: Kristen Grauman
Recap: Image Segmentation

- Goal: identify groups of pixels that go together
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:
      $$\sum_{\text{clusters } i} \sum_{\text{points } p \text{ in cluster } i} ||p - c_i||^2$$

Slide credit: Steve Seitz
Topics of This Lecture

• Fitting as parametric search
  - Line detection
  - Hough transform
  - Extension to circles
  - Generalized Hough transform

• Segmentation as clustering
  - k-Means
  - Feature spaces

• Probabilistic clustering
  - Mixture of Gaussians, EM

• Model-free clustering
  - Mean-Shift clustering
Probabilistic Clustering

• Basic questions
  - What’s the probability that a point $x$ is in cluster $m$?
  - What’s the shape of each cluster?

• K-means doesn’t answer these questions.

• Basic idea
  - Instead of treating the data as a bunch of points, assume that they are all generated by sampling a continuous function.
  - This function is called a generative model.
  - Defined by a vector of parameters $\theta$
Mixture of Gaussians

- One generative model is a mixture of Gaussians (MoG)
  - K Gaussian blobs with means $\mu_b$ covariance matrices $V_b$, dimension $d$
    - Blob $b$ defined by:
      $$P(x|\mu_b, V_b) = \frac{1}{\sqrt{(2\pi)^d|V_b|}}e^{-\frac{1}{2}(x-\mu_b)^T V_b^{-1}(x-\mu_b)}$$
  - Blob $b$ is selected with probability $\alpha_b$
  - The likelihood of observing $x$ is a weighted mixture of Gaussians
    $$P(x|\theta) = \sum_{b=1}^{K} \alpha_b P(x|\theta_b), \quad \theta = [\mu_1, \ldots, \mu_n, V_1, \ldots, V_n]$$

Slide credit: Steve Seitz
**Expectation Maximization (EM)**

- **Goal**
  - Find blob parameters $\theta$ that maximize the likelihood function:
  $${P(data|\theta) = \prod_{x} P(x|\theta)}$$

- **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

Slide credit: Steve Seitz
EM Details

- **E-step**
  - Compute probability that point \( x \) is in blob \( b \), given current guess of \( \theta \)
  \[
P(b|x, \mu_b, V_b) = \frac{\alpha_b P(x|\mu_b, V_b)}{\sum_{i=1}^{K} \alpha_i P(x|\mu_i, V_i)}
\]

- **M-step**
  - Compute probability that blob \( b \) is selected
  \[
  \alpha_b^{new} = \frac{1}{N} \sum_{i=1}^{N} P(b|x_i, \mu_b, V_b) \quad (N \text{ data points})
  \]
  - Mean of blob \( b \)
  \[
  \mu_b^{new} = \frac{\sum_{i=1}^{N} x_i P(b|x_i, \mu_b, V_b)}{\sum_{i=1}^{N} P(b|x_i, \mu_b, V_b)}
  \]
  - Covariance of blob \( b \)
  \[
  V_b^{new} = \frac{\sum_{i=1}^{N} (x_i - \mu_b^{new})(x_i - \mu_b^{new})^T P(b|x_i, \mu_b, V_b)}{\sum_{i=1}^{N} P(b|x_i, \mu_b, V_b)}
  \]

Slide credit: Steve Seitz
Applications of EM

- Turns out this is useful for all sorts of problems
  - Any clustering problem
  - Any model estimation problem
  - Missing data problems
  - Finding outliers
  - Segmentation problems
    - Segmentation based on color
    - Segmentation based on motion
    - Foreground/background separation
  - ...

- EM demo
Segmentation with EM

Original image

EM segmentation results

k=2  k=3  k=4  k=5

Image source: Serge Belongie
User assisted image segmentation

- User marks two regions for foreground and background.
- Learn a MoG model for the color values in each region.
- Use those models to classify all other pixels.

⇒ Simple segmentation procedure
   (building block for more complex applications)
Summary: Mixtures of Gaussians, EM

**Pros**
- Probabilistic interpretation
- Soft assignments between data points and clusters
- Generative model, can predict novel data points
- Relatively compact storage

**Cons**
- Local minima
  - k-means is NP-hard even with k=2
- Initialization
  - Often a good idea to start with some k-means iterations.
- Need to know number of components
  - Solutions: model selection (AIC, BIC), Dirichlet process mixture
- Need to choose generative model
- Numerical problems are often a nuisance
Topics of This Lecture

• Fitting as parametric search
  ➢ Line detection
  ➢ Hough transform
  ➢ Extension to circles
  ➢ Generalized Hough transform

• Segmentation as clustering
  ➢ k-Means
  ➢ Feature spaces

• Probabilistic clustering
  ➢ Mixture of Gaussians, EM

• Model-free clustering
  ➢ Mean-Shift clustering
Finding Modes in a Histogram

• How many modes are there?
  - *Mode* = local maximum of the density of a given distribution
  - Easy to see, hard to compute

Slide credit: Steve Seitz
Mean-Shift Segmentation

- An advanced and versatile technique for clustering-based segmentation

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

Mean-Shift Algorithm

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

Slide credit: Steve Seitz
Mean-Shift

Region of interest
Center of mass
Mean Shift vector

Slide by Y. Ukrainitz & B. Sarel
Mean-Shift
Mean-Shift

Region of interest
Center of mass

Mean Shift vector
Mean-Shift

Region of interest
Center of mass

Mean Shift vector

Slide by Y. Ukrainitz & B. Sarel
Mean-Shift
Mean-Shift
Mean-Shift

Region of interest

Center of mass
Real Modality Analysis

- Tessellate the space with windows
- Run the procedure in parallel

Slide by Y. Ukrainitz & B. Sarel
Real Modality Analysis

The blue data points were traversed by the windows towards the mode.

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 pixel locations
- Perform mean shift for each window until convergence
- Merge windows that end up near the same “peak” or mode

Slide credit: Svetlana Lazebnik
Mean-Shift Segmentation Results

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

Slide credit: Svetlana Lazebnik
More Results

Slide credit: Svetlana Lazebnik
More Results

Slide credit: Svetlana Lazebnik  B. Leibe
Problem: Computational Complexity

- Need to shift many windows...
- Many computations will be redundant.

B. Leibe
Speedups: Basin of Attraction

1. Assign all points within radius \( r \) of end point to the mode.

B. Leibe
2. Assign all points within radius $r/c$ of the search path to the mode.
Summary Mean-Shift

• **Pros**
  - General, application-independent tool
  - Model-free, does not assume any prior shape (spherical, elliptical, etc.) on data clusters
  - Just a single parameter (window size h)
    - h has a physical meaning (unlike k-means)
  - Finds variable number of modes
  - Robust to outliers

• **Cons**
  - Output depends on window size
  - Window size (bandwidth) selection is not trivial
  - Computationally (relatively) expensive (~2s/image)
  - Does not scale well with dimension of feature space
Generic Clustering

- We have focused on ways to group pixels into image segments based on their appearance
  - Find groups; “quantize” feature space

- In general, we can use clustering techniques to find groups of similar “tokens”, provided we know how to compare the tokens.
  - *E.g.*, segment an image into the types of motions present
  - *E.g.*, segment a video into the types of scenes (shots) present
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...)

Slide credit: Steve Seitz
Segmentation by Graph Cuts

- Break Graph into Segments
  - 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

Slide credit: Steve Seitz
Measuring Affinity

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

- **Intensity**
  \[
  \text{aff}(x, y) = \exp \left\{ -\frac{1}{2\sigma_d^2} \|I(x) - I(y)\|^2 \right\}
  \]

- **Color**
  \[
  \text{aff}(x, y) = \exp \left\{ -\frac{1}{2\sigma_d^2} \text{dist}(c(x), c(y))^2 \right\}
  \]
  (some suitable color space distance)

- **Texture**
  \[
  \text{aff}(x, y) = \exp \left\{ -\frac{1}{2\sigma_d^2} \|f(x) - f(y)\|^2 \right\}
  \]
  (vectors of filter outputs)

Source: Forsyth & Ponce
Scale Affects Affinity

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

![Graph showing the relationship between distance and affinity for different scales of $\sigma$.]

![Image showing examples of point grouping for small, medium, and large $\sigma$ values.](Image Source: Forsyth & Ponce)
Graph Cut

- Set of edges whose removal makes a graph disconnected
- Cost of a cut
  - Sum of weights of cut edges: $$\text{cut}(A, B) = \sum_{p \in A, q \in B} w_{p,q}$$
- A graph cut gives us a segmentation
  - What is a “good” graph cut and how do we find one?
Graph Cut

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
Normalized Cut (NCut)

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

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

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

\[
= \text{cut}(A, B) \left[ \frac{1}{\sum_{p \in A} w_{p,q}} + \frac{1}{\sum_{q \in B} w_{p,q}} \right]
\]

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

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
NCuts as a Generalized Eigenvector Problem

• Definitions

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

• Rewriting Normalized Cut in matrix form:

\[ NCut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)} \]
\[ = (1 + x)^T (D - W)(1 + x) + (1 - x)^T (D - W)(1 - x) \]
\[ = \frac{(1 + x)^T (D - W)(1 + x)}{k1^T D1} + \frac{(1 - x)^T (D - W)(1 - x)}{(1 - k)1^T D1}; \quad k = \frac{\sum_{x_i > 0} D(i,i)}{\sum_i D(i,i)} \]
\[ = \ldots \]
Some More Math...

We see again this is an inflated measure, which reflects how tightly on average nodes within the group are connected to each other.

Another important property of this definition of association and disassociation of a partition is that they are naturally related:

\[
N ass(A, B) = \frac{\text{cut}(A, B)}{\text{cut}(A, V)} + \frac{\text{cut}(B, V)}{\text{cut}(B, V)} = \frac{\sum_{i,j \in A} d(i, j)}{\sum_{i,j \in A} d(i, j)} + \frac{\sum_{i,j \in B} d(i, j)}{\sum_{i,j \in B} d(i, j)} = 2 - N ass(A, B)
\]

Hence the two partition criteria that we seek in our grouping algorithm, minimizing the disassociation between the groups and maximizing the association within the group, are in fact identical, and can be satisfied simultaneously. In our algorithm, we will use this normalized cut as the partition criterion.

Having defined the graph partition criterion that we want to optimize, we will show how such an optimal partition can be computed efficiently.

2.1 Computing the optimal partition

Given a partition of nodes of a graph, \(V\), into two sets \(A\) and \(B\), let \(\mathbf{W}\) be a \(N \times N\) dimensional indicator vector, \(\mathbf{w}_i = 1\) if node \(i\) is in \(A\), and \(-1\) otherwise. Let \(d(i) = \sum_{j} w(i, j)\) be the total connection from node \(i\) to all other nodes. With the definitions \(a\) and \(d\) we can rewrite \(N ass(A, B)\) as:

\[
N ass(A, B) = \frac{\text{cut}(A, B)}{\text{cut}(A, V)} + \frac{\text{cut}(B, V)}{\text{cut}(B, V)} = \frac{\sum_{i,j \in A} d(i) d(j)}{\sum_{i,j \in A} d(i) d(j)} + \frac{\sum_{i,j \in B} d(i) d(j)}{\sum_{i,j \in B} d(i) d(j)} = 2 - N ass(A, B)
\]

Let \(\mathbf{D}\) be an \(N \times N\) diagonal matrix with \(d\) on its diagonal, \(\mathbf{W}\) be an \(N \times N\) symmetrical matrix with \(W(i,j) = w_{ij}\), and \(x\) be an \(N \times 1\) vector of all ones. Using the fact \(\frac{1}{2}x^T D x\) and \(\frac{1}{2}x^T W x\) are indicator vectors for \(x > 0\) and \(x < 0\) respectively, we can rewrite \(N ass(A, B)\) as:

\[
(\text{cut}(A, B))_+ + (\text{cut}(B, V))_+ = \frac{\|x\|^2 D x}{\|x\|^2 D x} + \frac{\|x\|^2 W x}{\|x\|^2 W x} = \frac{a^T (D - W) x}{a^T (D - W) x} + \frac{b^T (D - W) x}{b^T (D - W) x} = \frac{a^T (D - W) x + b^T (D - W) x}{a^T (D - W) x + b^T (D - W) x}
\]

Let \(a^T (D - W) x + b^T (D - W) x\), \(\alpha\), \(\gamma\) are \(\gamma^T (D - W) x\), and \(\gamma = \gamma^T (D - W) x\), and \(M = a^T D x\), we can then further expand the above equation as:

\[
\frac{\alpha + \gamma + 2(1 - \beta)\beta}{\alpha + \gamma + 2(1 - \beta)\beta} = \frac{\alpha + \gamma + 2(1 - \beta)\beta}{\alpha + \gamma + 2(1 - \beta)\beta} - \frac{2\alpha + \gamma}{\alpha + \gamma + 2(1 - \beta)\beta}
\]

dropping the last constant term, which in this case equals 0, we get:

\[
\frac{1 - 2\beta + 2\beta^2}{1 - 2\beta + 2\beta^2} = \frac{1 - 2\beta + 2\beta^2}{1 - 2\beta + 2\beta^2} - \frac{2\alpha}{\alpha + \gamma + 2(1 - \beta)\beta}
\]

Letting \(\delta = \frac{\alpha}{\gamma}\), and since \(\gamma = 0\), it becomes:

\[
\frac{1 - 2\beta + 2\beta^2}{1 - 2\beta + 2\beta^2} = \frac{1 - 2\beta + 2\beta^2}{1 - 2\beta + 2\beta^2} - \frac{2\alpha}{\alpha + \gamma + 2(1 - \beta)\beta}
\]

Note that:

\[
(1 - 2\beta + 2\beta^2)\alpha + 2(1 - \beta)\beta(\alpha + \gamma) = \alpha + \gamma + 2(1 - \beta)\beta\alpha + 2\alpha
\]

Letting \(\delta = \frac{\alpha}{\gamma}\), and since \(\gamma = 0\), it becomes:

\[
(1 - 2\beta + 2\beta^2)\alpha + 2(1 - \beta)\beta(\alpha + \gamma) = \alpha + \gamma + 2(1 - \beta)\beta\alpha + 2\alpha
\]

Note that:

\[
(1 - 2\beta + 2\beta^2)\alpha + 2(1 - \beta)\beta(\alpha + \gamma) = \alpha + \gamma + 2(1 - \beta)\beta\alpha + 2\alpha
\]

Letting \(\delta = \frac{\alpha}{\gamma}\), and since \(\gamma = 0\), it becomes:

\[
(1 - 2\beta + 2\beta^2)\alpha + 2(1 - \beta)\beta(\alpha + \gamma) = \alpha + \gamma + 2(1 - \beta)\beta\alpha + 2\alpha
\]

Note that:

\[
(1 - 2\beta + 2\beta^2)\alpha + 2(1 - \beta)\beta(\alpha + \gamma) = \alpha + \gamma + 2(1 - \beta)\beta\alpha + 2\alpha
\]
NCuts as a Generalized Eigenvalue Problem

- After simplification, we get
  \[ NCut(A, B) = \frac{y^T(D-W)y}{y^TDy}, \quad \text{with } y_i \in \{1,-b\}, \quad y^TD1 = 0. \]

- 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
    \[ D^{-\frac{1}{2}}(D-W)D^{-\frac{1}{2}}z = \lambda z, \quad \text{where } z = D^{\frac{1}{2}}y \]

- Subtleties
  - Optimal solution is second smallest eigenvector
  - Gives continuous result—must convert into discrete values of y

This is hard, as \( y \) is discrete!

Relaxation: continuous \( y \).
NCuts Example

NCuts segments

Smallest eigenvectors

Image source: Shi & Malik
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 $n$ possible splitting points.
     2. Compute $NCut$ value.
     3. Pick minimum.
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
Using Texture Features for Segmentation

- Texture descriptor is vector of filter bank outputs


Slide credit: Svetlana Lazebnik
Using Texture Features for Segmentation

- Texture descriptor is vector of filter bank outputs.
- **Textons** are found by clustering.
Using Texture Features for Segmentation

- Texture descriptor is vector of filter bank outputs.
- **Textons** are found by clustering.
- Affinities are given by similarities of texton histograms over windows given by the “local scale” of the texture.
Results with Color & Texture
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 $\Rightarrow$ 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

Slide credit: Kristen Grauman
Extension: Multi-Level Segmentation

Example segmentations for several contrasts

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

Source: [S. Todorovic, N. Ahuja, CVPR'06]
Multiscale Segmentation Tree

- Segmentations can be arranged in a tree

Example segmentations

Source: [S. Todorovic, N. Ahuja, CVPR'06]
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

Observed evidence

Hidden “true states”

Neighborhood relations
MRF Nodes as Pixels

Original image

Degraded image

Reconstruction from MRF modeling pixel neighborhood statistics
MRF Nodes as Patches

\[ \Phi(x_i, y_i) \]
\[ \Psi(x_i, x_j) \]
Network Joint Probability

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

- Scene
- Image
- Image-scene compatibility function
- Local observations
- Scene-scene compatibility function
- Neighboring scene nodes

Slide credit: William Freeman
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**

\[- E(x, y) = \sum_i \phi(x_i, y_i) + \sum_{i,j} \psi(x_i, x_j)\]

**Single-node potentials**
- 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**
- 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

Minimum cost cut can be computed in polynomial time
(max-flow/min-cut algorithms)

\[ w_{pq} = \exp \left\{ -\frac{\Delta I_{pq}}{2\sigma^2} \right\} \]

[Boykov & Jolly, ICCV’01]

Slide credit: Yuri Boykov
Simple Example of Energy

\[ E(L) = \sum_p D_p(L_p) + \sum_{pq \in N} w_{pq} \cdot \delta(L_p \neq L_q) \]

- **Regional term**: \( D_p(L_p) \)
- **Boundary term**: \( w_{pq} \cdot \delta(L_p \neq L_q) \)

\( w_{pq} = \exp \left\{ -\frac{\Delta I_{pq}}{2\sigma^2} \right\} \)

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

Slide credit: Yuri Boykov
Adding Regional Properties

Regional bias example

Suppose $I^s$ and $I^t$ are given “expected” intensities of object and background

NOTE: hard constrains are not required, in general.

$D_p(t) \propto \exp \left( -\| I_p - I^s \|^2 / 2\sigma^2 \right)$

$D_p(s) \propto \exp \left( -\| I_p - I^t \|^2 / 2\sigma^2 \right)$
Adding Regional Properties

“expected” intensities of object and background $I^s$ and $I^t$ can be re-estimated

EM-style optimization

\[ D_p(t) \propto \exp \left( -\| I_p - I^s \|^2 / 2\sigma^2 \right) \]

\[ D_p(s) \propto \exp \left( -\| I_p - I^t \|^2 / 2\sigma^2 \right) \]

Slide credit: Yuri Boykov

[Boykov & Jolly, ICCV’01]
Adding Regional Properties

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

\[
D_p (s) = \log \Pr(I_p | I_p)
\]

\[
D_p (t) = \log \Pr(I_p | I_p)
\]

Given object and background intensity histograms.
How to Set the Potentials? Some Examples

- **Color potentials**
  - e.g. modeled with a Mixture of Gaussians
    \[
    \pi(x_i, y_i; \theta_\pi) = \log \sum_k \theta_\pi(x_i, k) P(k \mid x_i) N(y_i; \bar{y}_k, \Sigma_k)
    \]

- **Edge potentials**
  - e.g. a “contrast sensitive Potts model”
    \[
    \phi(x_i, x_j, g_{ij}(y); \theta_\phi) = -\theta_\phi^T g_{ij}(y) \delta(x_i \neq x_j)
    \]
    where
    \[
    g_{ij}(y) = e^{-\beta \|y_i - y_j\|^2} \quad \beta = 2 \cdot \text{avg} \left( \|y_i - y_j\|^2 \right)
    \]

- **Parameters** \(\theta_\pi, \theta_\phi\) need to be learned, too!
When Can s-t Graph Cuts Be Applied?

\[ E(L) = \sum_{p} E_p(L_p) + \sum_{pq \in N} E(L_p, L_q) \]

- s-t graph cuts can only globally minimize binary energies that are submodular. [Boros & Hummer, 2002, Kolmogorov & Zabih, 2004]

\[ E(s, s) + E(t, t) \leq E(s, t) + E(t, s) \]

Submodularity ("convexity")

- Non-submodular cases can still be addressed with some optimality guarantees.
  - Current research topic
Topics of This Lecture

• Graph theoretic segmentation
  - Normalized Cuts
  - Using color and texture features

• Hierarchical segmentation
  - Case study: segmentation-based recognition

• Segmentation as Energy Minimization
  - Markov Random Fields
  - Graph cuts for image segmentation
  - Applications
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.

User segmentation cues

Additional segmentation cues

Slide credit: Matthieu Bray
GrabCut: Data Model

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

Slide credit: Carsten Rother
GrabCut: Coherence Model

- An object is a coherent set of pixels:

\[ \psi(x, y) = \gamma \sum_{(m,n) \in C} \delta[x_n \neq x_m] e^{-\beta \| y_m - y_n \|^2} \]

How to choose \( \gamma \)?
Iterated Graph Cuts

Result

Energy after each iteration

Color model (Mixture of Gaussians)
GrabCut: Example Results

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

Image source: Carsten Rother
Applications: Interactive 3D Segmentation
Improving Efficiency of 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
Superpixels for Pre-Segmentation

Graph structure

<table>
<thead>
<tr>
<th>Image</th>
<th>Dimension</th>
<th>Nodes Ratio</th>
<th>Edges Ratio</th>
<th>Lag with Pre-segmentation</th>
<th>Lag without Pre-segmentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boy</td>
<td>(408, 600)</td>
<td>10.7</td>
<td>16.8</td>
<td>0.12s</td>
<td>0.57s</td>
</tr>
<tr>
<td>Ballet</td>
<td>(440, 800)</td>
<td>11.4</td>
<td>18.3</td>
<td>0.21s</td>
<td>1.39s</td>
</tr>
<tr>
<td>Twins</td>
<td>(1024, 768)</td>
<td>20.7</td>
<td>32.5</td>
<td>0.25s</td>
<td>1.82s</td>
</tr>
<tr>
<td>Girl</td>
<td>(768, 1147)</td>
<td>23.8</td>
<td>37.6</td>
<td>0.22s</td>
<td>2.49s</td>
</tr>
<tr>
<td>Grandpa</td>
<td>(1147, 768)</td>
<td>19.3</td>
<td>30.5</td>
<td>0.22s</td>
<td>3.56s</td>
</tr>
</tbody>
</table>

Speedup
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
References and Further Reading

- Background information on Normalized Cuts can be found in Chapter 14 of

- 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)