EE 584
MACHINE VISION

Image Segmentation

Histogram-based Segmentation

Automatic Thresholding,K-means Clustering

Spatial Coherence

Merging and Splitting, Region Growing
Graph Theoretic Segmentation
Mean-shift Segmentation
Watersheds
Active Contours

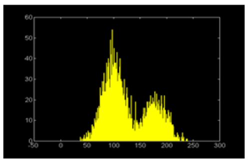
Segmentation

- Segmentation : Dividing images into (semantically meaningful) regions that appear to be images of different surfaces
- Two major approaches :
  - Histogram-based segmentation
  - Segmentation based on spatial coherence
    - Divide (split) or merge type
    - Growing regions type
    - Graph theoretic approaches

- Reliable segmentation is only possible with some a priori info, which is not mostly available

Histogram-based Segmentation

- Segmentation : Gray level → Binary mask
  by an unknown threshold
- Useful for foreground/background segmentation
- How to find an automatic threshold ?
  - considering variations in illumination & surface



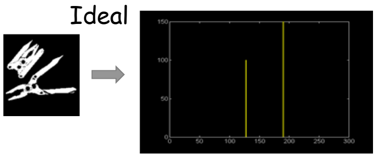
- Gray-level *histogram* gives the <u>number of picture cells</u> having a particular gray-level

METU EE 584 Lecture Notes by A.Aydin ALATAN © 2013

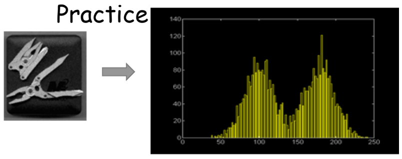
Histogram-based Segmentation

- <u>Ideally</u>, object & background have <u>constant</u> different brightness values inside their regions
  → put a threshold between peak values in histogram

Ideal



Practice



- In <u>practice</u>, due to
  - measurement noise
  - non-uniform illumination
  - non-uniform reflection from the surfaces

brightness is <u>not constant</u>; there is some *spread*

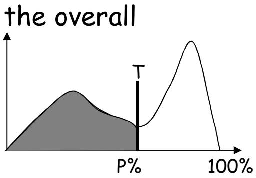
Image Thresholding Approaches

<u>Main directions</u> :

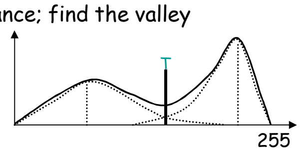
1. Histogram shape-based
   - Analyze peaks, valleys, <u>curvatures of smoothed histogram</u>
2. Clustering-based
   - <u>Iteratively, finding a threshold</u>, clustering based this threshold
3. Entropy-based
   - Choosing a threshold which <u>max the info content</u> in histogram
4. Attribute-based
   - <u>Similarities between edge</u>, etc of image & its thresholded vers.
5. Spatial thresholding (higher order stats)
   - Threshold selection on <u>higher order statistics</u> of spatial neighb.
6. Local thresholding
   - Finding threshold values at each neighborhood using <u>local stats</u>

Automatic Thresholding Methods (1/3)

1. **P-tile method**
   - Use the <u>a priori knowledge</u> about the <u>size</u> of the object : assume an object with size *p*
   - Choose the threshold such that %p of the overall histogram is determined
   - Obviously, limited use



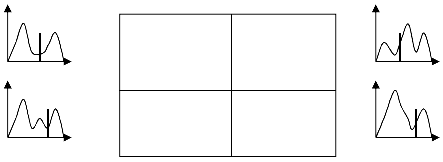
2. **Mode method**
   - Find the "peaks" and "valleys" of the histogram
   - Set threshold to the pixel value of the "valley"
   - Non-trivial to find peaks/valleys :
     - ignore local peaks; choose peaks at a distance; find the valley between those peaks



Automatic Thresholding Methods (2/3)

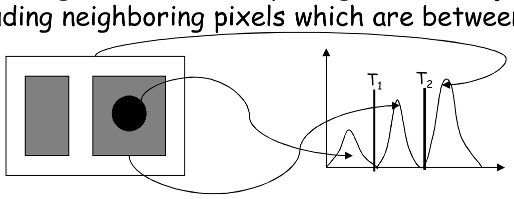
3. **Iterative threshold selection**
   - Starting with an approximate threshold, refine threshold iteratively, taking into account some goodness measure
     e.g. $T = (\mu_1 + \mu_2)/2$ where μ_i is the mean gray value of previous segmented region i

4. **Adaptive Thresholding**
   - In case of uneven illumination, global threshold has no use
   - One approach is to divide an image into *mxm* subimages and determine a threshold for each subimage



Automatic Thresholding Methods (3/3)

5. **Double Thresholding**
   - Starting from a conservative initial threshold T_1, determine the "core" parts of the object
   - Continuing from this core part, grow this object by including neighboring pixels which are between T_1 and T_2

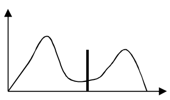


6. **Otsu Thresholding**
   - Minimize weighted sum of within-cluster variances

$$\min_t \sigma_W^2(t) = P_f(t)\sigma_f^2(t) + P_b(t)\sigma_b^2(t)$$

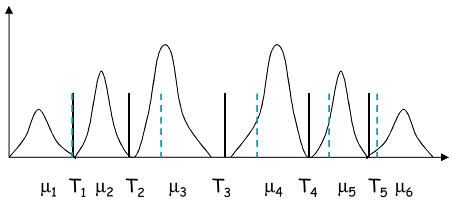
$\sigma_{f,b}^2(t)$: within − cluster variance

$P_{f,b}(t)$: cluster probability



K-means Clustering

- It is usually desired to segment an image into more than two regions



- **K-means (or ISODATA) algorithm** :

1) Choose initial mean values for k (or c) region

2) Classify n pixels by assigning to "closest" mean

3) Recompute the means as the average of samples in their (new) classes

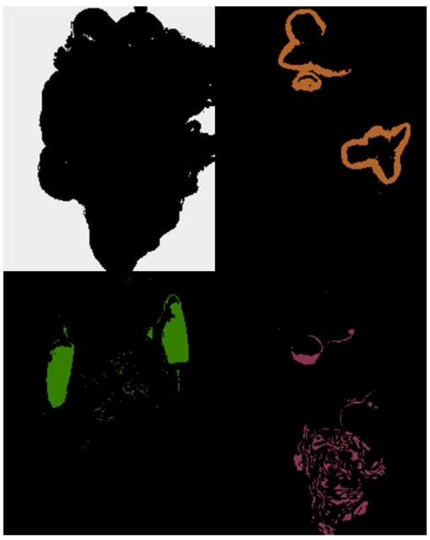
4) Continue till there is no change in mean values

K-means Clustering

K-means by using color (11 segments)

Original Image

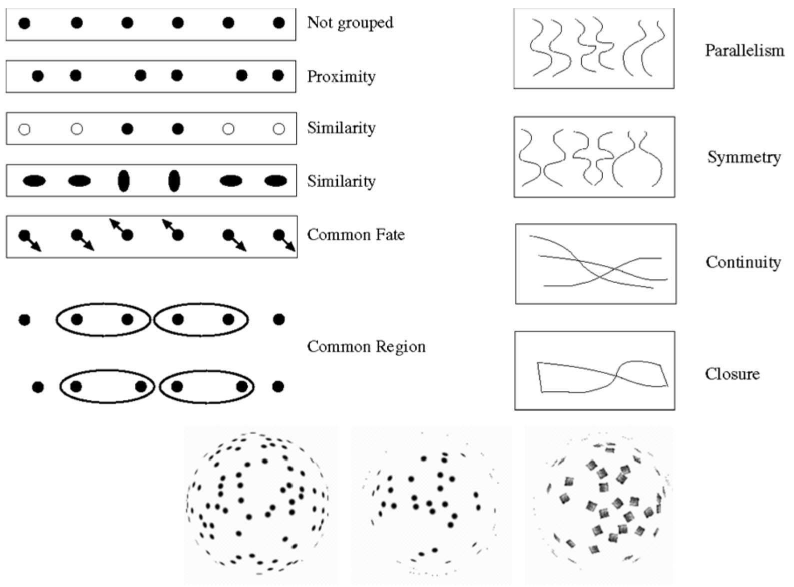
Clusters on color



Spatial Coherence

- Histogram-based methods totally neglect the dependency between neighboring pixels
- Human visual system and its perception also supports dependency between pixels within a neighborhood for grouping
  - Gestalt psychology
- Gestalt psychology identifies several properties result in grouping/segmentation:

Grouping based on Gestalt psychology :



Spatial Coherence : Merging & Splitting

- The output of any segmentation method can be improved by simply merging *similar* neighboring regions together
- Similarity can be measured by
  - A simple threshold
  - A geometrical attribute, such as "common boundary length"
  - More sophisticated methods based on statistics
- Similarly, rather than merging, splitting can be required due to geometrical attributes

Region Merge:

- A general region merge algorithm
  - Beginning from an initial segmentation,
  - For each region check, whether its neighboring regions are "similar", if so, merge these regions
- How to measure "region similarity" ?
  - Compare their mean intensities
    check with a predetermined threshold
  - Compare their statistical distributions
    check whether such a merge represents "observed" values better
  - Check "weakness" of the common boundary
    weak boundary: intensities on two sides differ less than a threshold
    Merge two regions if $W/S > \tau$ where W=length of weak boundary
    1) $S = \min\{S_1,S_2\}$: minimum of two boundaries
    2) S : common boundary

Region Split :

- If some property of a region is NOT constant → split
  - variance of the intensities,
  - error between the intensities and a fitted surface
- If decide on spitting, how to split, so that new regions will have constant values with this property?
  - Try equal size splitting → modified quad-tree representation

- Split and Merge
  - Starting from a presegmentation,
  - Find a region that can be splitted ➔ split into four regions
  - If any two or more neighboring subregions are "similar" ➔ merge all these regions into a single region

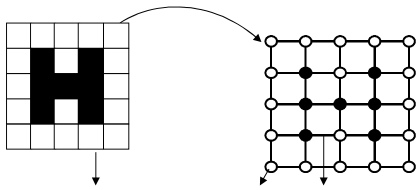
Region Growing :

- Starting from "seed regions" (small regions with some homogeneity which is based on surface fitting)
- Find "compatible" neighboring points which fit to a model (surface) of the seed region
  ➔ grow the region
- Refit the surface taking into account these new points
- Check the difference between the new and the old goodness fit for the surface; if no improvement
  ➔ stop growing

Graph Theory In Image Segmentation

For the analysis of images using graph theory

- An image is mapped onto a graph
- Each pixel (sub region) in the image corresponds to a vertex
- Typical connection between the nodes is 4 (or 8) connections.
- Vertices and edges have weights
  - Pixel grey level value is usually assigned to vertex weight
  - Weights associated with each link are based on some property of the pixels that it connects, such as their intensity differences.



Four-way Connected Graph

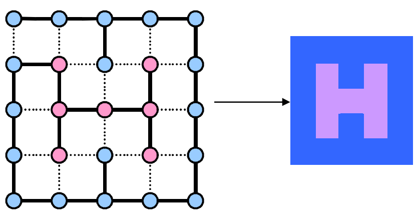
Slides modified from Neslihan Yalçın Bayramoğlu Thesis Presentaton

METU EE 584 Lecture Notes by A.Aydin ALATAN © 2013

Graph Theory In Image Segmentation

- Insertion of links between "similar" intensities creates subtrees for these objects
- Every sub tree represents a region of the image

Region 1



Region 2

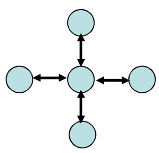
Slides modified from Neslihan Yalçın Bayramoğlu Thesis Presentaton

Graph Theory In Image Segmentation

- There could be <u>two main approaches</u> to exploit graph theoretical formulation in image segmentation
  - Bottom-up :
    - Starting from the whole image, recursively <u>merging neighboring pixels</u> (regions) to reach desired image segmentation
    - *<u>Example</u>* : Recursive Shortest Spanning Tree (RSST)
  - Top-down :
    - Starting from the whole image, recursively <u>dividing the image into subregions</u> until desired image segmentation is reached.
    - *<u>Example</u>* : Normalized Cut & Graph Cut

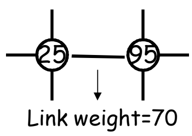
Recursive Shortest Spanning Tree (RSST)

- A <u>bottom-up</u> graph-theoretic segmentation approach
  - Start from similarity of neighboring intensities
  - Check similarity of neighboring regions



<u>Algorithm</u>:

1. Image is mapped onto a graph
2. Four-way connected graph is used
3. Absolute differences of gray levels between vertices are assigned to link weights



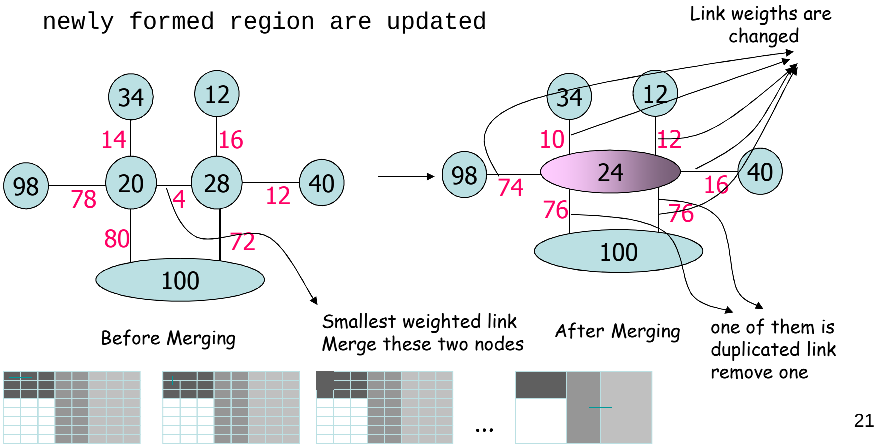
Slides modified from Neslihan Yalçın Bayramoğlu Thesis Presentaton

Recursive Shortest Spanning Tree (RSST)

Algorithm (cont'd) :

4. Nodes with the smallest link value are merged
5. Weights of the links neighboring to the newly formed region are updated

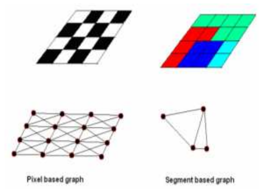
$$LW(i,j) = \left|V_i - V_j\right| \frac{N_i N_j}{N_i + N_j}$$

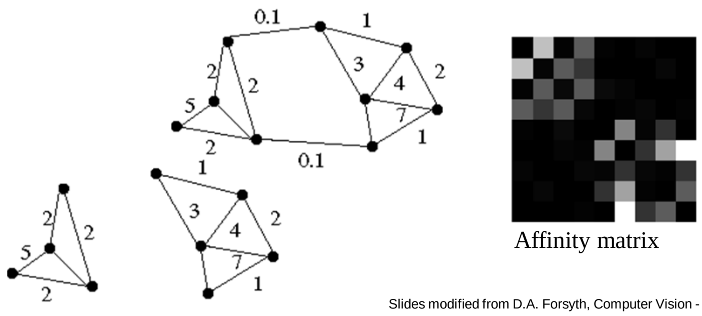


Slides modified from Neslihan Yalçın Bayramoğlu Thesis Presentaton

Minimal Cut Method

- A top-down approach
- Nodes are represented by using a weighted graph.
  - Affinity matrix, *A* (similar nodes have higher valued entries)
- Cut up this graph to get subgraphs with strong links





Affinity matrix

Slides modified from D.A. Forsyth, Computer Vision - A Modern Approach

Minimal Cut: Measuring Affinity

Intensity

$$aff(x,y) = \exp\left\{-\left(\frac{1}{2\sigma_i^2}\right)\left(\|I(x) - I(y)\|^2\right)\right\}$$

Distance

$$aff(x,y) = \exp\left\{-\left(\frac{1}{2\sigma_d^2}\right)\left(\|x - y\|^2\right)\right\}$$

Texture

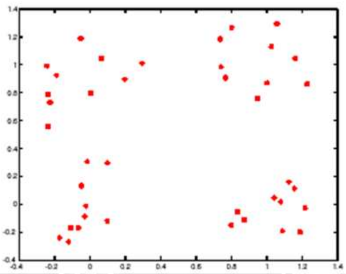
$$aff(x,y) = \exp\left\{-\left(\frac{1}{2\sigma_t^2}\right)\left(\|c(x) - c(y)\|^2\right)\right\}$$

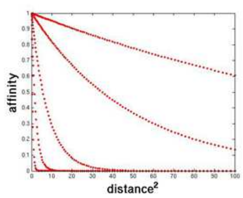
Effect of scale (σ) on the affinity

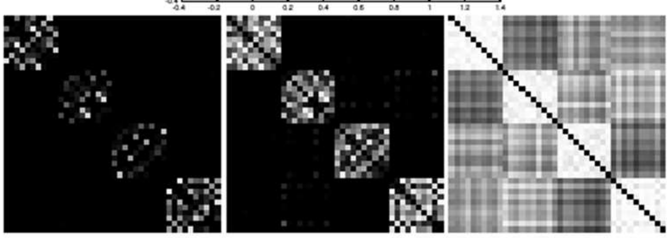
$$aff(x,y) = \exp\left\{-\left(\frac{1}{2\sigma_i^2}\right)\left(\|I(x) - I(y)\|^2\right)\right\}$$

$$aff(x,y) = \exp\left\{-\left(\frac{1}{2\sigma_d^2}\right)\left(\|x - y\|^2\right)\right\}$$

$$aff(x,y) = \exp\left\{-\left(\frac{1}{2\sigma_t^2}\right)\left(\|c(x) - c(y)\|^2\right)\right\}$$







Minimal Cut : Solution via Eigenvectors

- Idea: Find vector, w, giving the association between each node and a cluster
  - Elements within a cluster should have strong affinity with each other
  - Maximize the following relation: $w^T A w$ (A : *affinity matrix*)

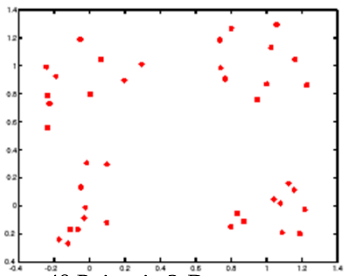
$$w^T A w = \sum_{i,j} w_i a_{i,j} w_j$$

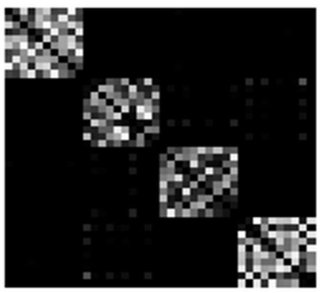
$$= \sum \Big(\underbrace{\text{assoc. of node}_i \text{ to cluster}}_{w_i} \Big) \Big(\underbrace{\text{similarity node}_i \text{ \& node}_j}_{a_{i,j}} \Big) \Big(\underbrace{\text{assoc. of node}_j \text{to cluster}}_{w_j} \Big)$$

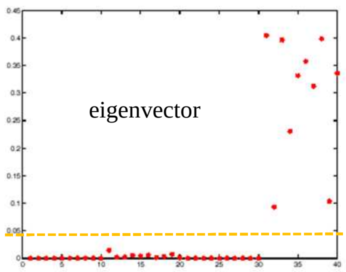
- This relation is maximized, if all 3 terms are non-zero (not very small)
- There should be an extra constraint, as $w^T w = 1$
- Optimize by method of *Lagrange* multiplier : $\max\{w^T A w + \lambda (w^T w - 1)\}$
- Solution is an eigenvalue problem

  ➔ Choose the eigenvector of A with the largest eigenvalue

Minimal Cut : Solution via Eigenvectors


40 Points in 2-D space

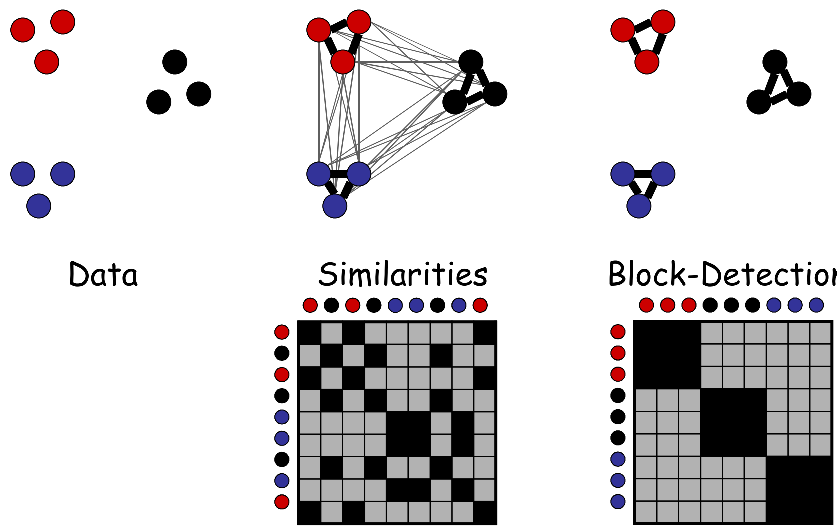

Matrix (40x40)



Minimal Cut Algorithm:

1. Form A-matrix & apply SVD
2. Determine the eigenvector corresponding to the largest eigenvalue
3. Threshold the components of this eigenvector to find node indices of a cluster

Finding the Minimal Cuts:



* Slides from Dan Klein, Sep Kamvar, Chris Manning, Natural Language Group Stanford University

METU EE 584 Lecture Notes by A.Aydin ALATAN © 2013

Eigenvectors and Blocks

- Block matrices have block eigenvectors:

1	1	0	0
1	1	0	0
0	0	1	1
0	0	1	1

eigensolver

$\lambda_1 = 2$	$\lambda_2 = 2$	$\lambda_3 = 0$	$\lambda_4 = 0$
.71	0		
.71	0		
0	.71		
0	.71		

- Near-block matrices have near-block eigenvectors

1	1	.2	0
1	1	0	-.2
.2	0	1	1
0	-.2	1	1

eigensolver

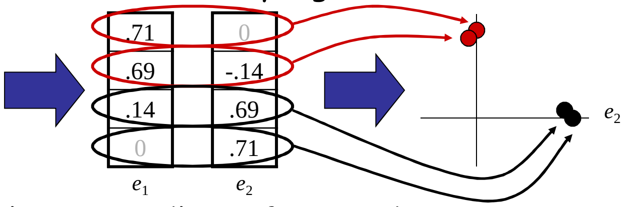
$\lambda_1 = 2.02$	$\lambda_2 = 2.02$	$\lambda_3 = -0.02$	$\lambda_4 = -0.02$
.71	0		
.69	-.14		
.14	.69		
0	.71		

* Slides from Dan Klein, Sep Kamvar, Chris Manning, Natural Language Group Stanford University

Eigenvectors and Blocks

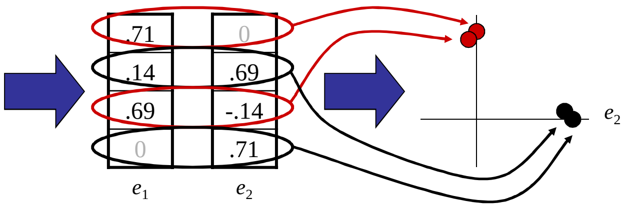
- Can put items into blocks by eigenvectors:

1	1	.2	0
1	1	0	-.2
.2	0	1	1
0	-.2	1	1



- Clusters clear regardless of row ordering:

1	.2	1	0
.2	1	0	1
1	0	1	-.2
0	1	-.2	1



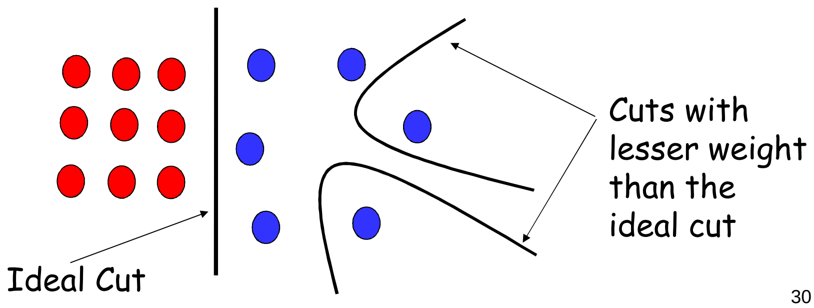
- Note that only with row ordering we can $\max\{e^T A e\}$

* Slides from Dan Klein, Sep Kamvar, Chris Manning, Natural Language Group Stanford University

Drawbacks of Minimal Cut

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

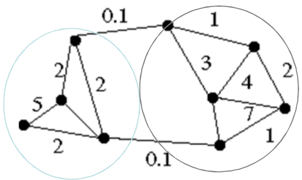
- Weight of cut is directly proportional to the number of edges in the cut.
- It tends to produce small, isolated components



* Slide from Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Normalized cuts

- Previous criterion only evaluates within cluster similarity, but not across cluster difference
- Instead, one would like to maximize within cluster similarity compared to the across cluster difference
- Write graph V, one cluster as A and the other as B



- Minimize *Normalized Cut*

$$\left(\frac{cut(A,B)}{assoc(A,V)}\right)+\left(\frac{cut(A,B)}{assoc(B,V)}\right)$$

$cut(A,B)$: sum of edges between A&B

$assoc(A,V)$: sum of edges only in A

- Construct A, B such that their within cluster similarity is high,
  - compared to their association with the rest of the graph

METU EE 584 Lecture Notes by A.Aydin ALATAN © 2013

Normalized cuts

$$NCut(A,B)=\left(\frac{cut(A,B)}{assoc(A,V)}\right)+\left(\frac{cut(A,B)}{assoc(B,V)}\right)$$

$cut(A,B)$: sum of edges between A&B
$assoc(A,V)$: sum of edges only in A

$$=\frac{(\bar{1}+x)^T(D-W)(\bar{1}+x)}{k\,\bar{1}^T D\bar{1}}+\frac{(\bar{1}-x)^T(D-W)(\bar{1}-x)}{k\,\bar{1}^T D\bar{1}}$$

$$x:\text{vector of enteries}\pm 1,\ x(i)=1\Leftrightarrow i\in A;\ \ x(i)=-1\Leftrightarrow i\in B$$

$$W:\text{affinity matrix};\ D(i,i)=\sum_j W(i,j);\ \ k=\frac{\sum_{x_i>0} D(i,i)}{\sum_i D(i,i)}$$

$$Let\ \ y\equiv(\bar{1}+x)-b(\bar{1}-x),\quad \text{where}\ b=k/(1-k)$$

Normalized cuts

- Defined vector y, has elements as
  - 1, if item is in A,
  - $-b$, if item is in B
- After derivations $\min NCut(A,B) = \min\left(\frac{cut(A,B)}{assoc(A,V)}\right) + \left(\frac{cut(A,B)}{assoc(B,V)}\right)$

  is shown to be equivalent to $\min_y \left(\frac{y^T (D-W) y}{y^T D y}\right)$
- with the constraint $y^T D 1 = 0$

  (Read proof in the distributed notes)
- This is so called *Rayleigh Quotient*

Normalized cuts

- Its solutions is the generalized eigenvalue problem

$$\min \left(y^T (D-W) y\right) \text{ subject to } \left(y^T D y = 1\right)$$

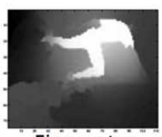
which gives

$$(D-W)y = \lambda D y$$

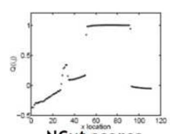
$$\Rightarrow D^{-1/2}(D-W)D^{-1/2} y = \lambda y$$

- Optimal solution is the eigenvector due to second smallest eigenvalue
- Now, look for a quantization threshold that maximizes the criterion --- i.e. all components of y above that threshold go to one, all below go to $-b$

Image



Eigenvector



NCut scores

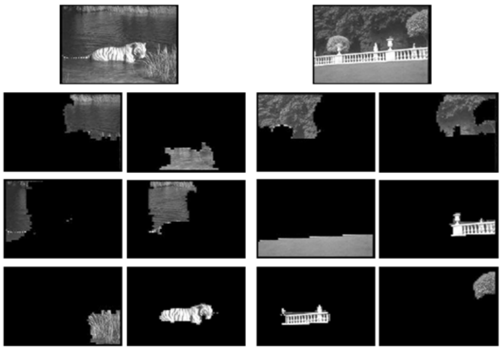
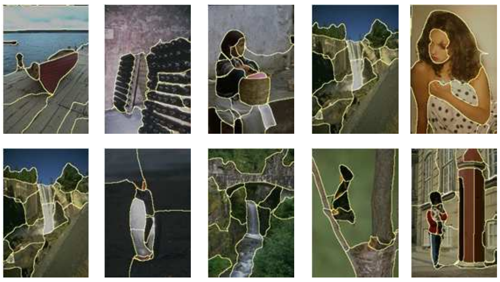


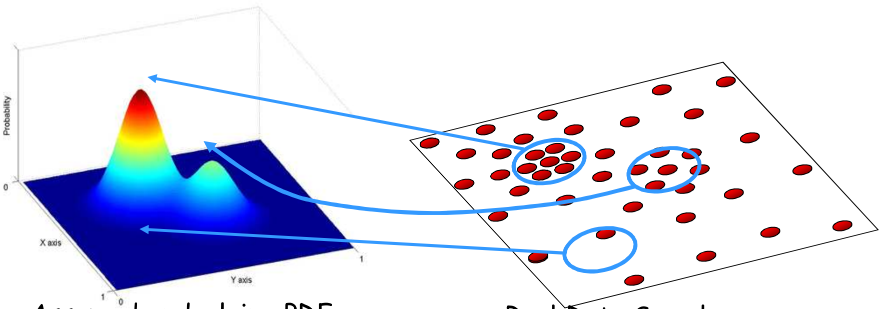
Figure from "Image and video segmentation: the normalized cut framework", by Shi and Malik, copyright IEEE, 1998



Results: Berkeley Segmentation Engine
http://www.cs.berkeley.edu/~fowlkes/BSE/

Mean-shift Clustering

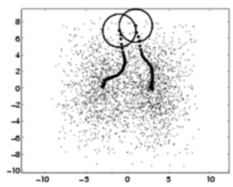
Observed data points are sampled from an underlying PDF



Assumed underlying PDF

Real Data Samples

- Mean-shift procedure is an elegant way to <u>determine the modes of a PDF</u> without explicitly calculating the density itself.
- Mean-shift vector is an estimate of the gradient of the PDF



Mean-shift Clustering

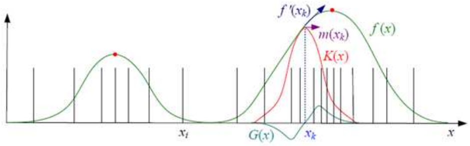
From sparse samples, a kernel density estimate of a pdf:

$$f(\boldsymbol{x}) = \sum_i K(\boldsymbol{x} - \boldsymbol{x}_i) = \sum_i k\left(\frac{\|\boldsymbol{x} - \boldsymbol{x}_i\|^2}{h^2}\right)$$

Gradient of the density, $k(\ )$, is denoted by $-g(\ )$,

$$\nabla f(\boldsymbol{x}) = \sum_i (\boldsymbol{x}_i - \boldsymbol{x}) G(\boldsymbol{x} - \boldsymbol{x}_i) = \sum_i (\boldsymbol{x}_i - \boldsymbol{x}) g\left(\frac{\|\boldsymbol{x} - \boldsymbol{x}_i\|^2}{h^2}\right)$$

$$\nabla f(\boldsymbol{x}) = \left[\sum_i G(\boldsymbol{x} - \boldsymbol{x}_i)\right] \boldsymbol{m}(\boldsymbol{x})$$



$$\boldsymbol{m}(\boldsymbol{x}) = \frac{\sum_i \boldsymbol{x}_i G(\boldsymbol{x} - \boldsymbol{x}_i)}{\sum_i G(\boldsymbol{x} - \boldsymbol{x}_i)} - \boldsymbol{x}$$

$m(x)$ vector is called (weighted) "mean shift" and mode of f, y is obtained iteratively using mean shift vector

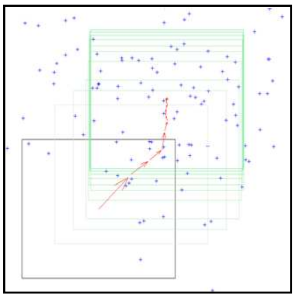
$$\boldsymbol{y}_{k+1} = \boldsymbol{y}_k + \boldsymbol{m}(\boldsymbol{y}_k) = \frac{\sum_i \boldsymbol{x}_i G(\boldsymbol{y}_k - \boldsymbol{x}_i)}{\sum_i G(\boldsymbol{y}_k - \boldsymbol{x}_i)}$$

Mean-shift Clustering

Mean Shift (General Mode Finding) Algorithm:

1. Choose a search window size.
2. Choose the initial location of the search window.
3. Compute the mean location (centroid of the data) within the search window.
4. Center/move the search window to the mean location computed in Step 3.
5. Repeat Steps 3 and 4 until convergence.

The mean shift algorithm seeks the "mode" (or point of highest density) of a data distribution



Mean Shift Segmentation

Mean Shift Segmentation Algorithm

1. Convert the image into features(via color, gradients, texture measures etc).
2. Choose initial search window locations uniformly in the data.
3. Compute the mean shift window location for each initial position.
4. Merge windows that end up on the same "peak" or mode.
5. The data these merged windows traversed are clustered together.

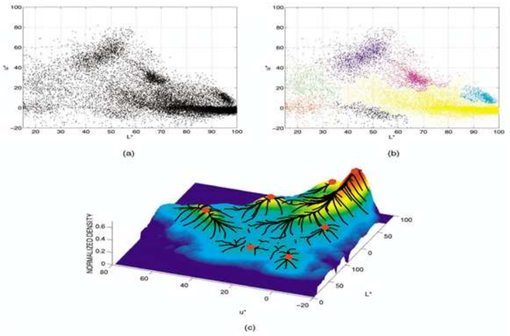


*Image From: Dorin Comaniciu and Peter Meer, Distribution Free Decomposition of Multivariate Data, Pattern Analysis & Applications (1999)2:22–30

Mean Shift Segmentation Results:

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

Watershed Image Segmentation

- Watershed is an image segmentation method.
- 2D image to 3D topological map.
  - 3rd dimension is intensity of grey scale images
- Watershed segments an image into several *catchment basins*
  - regions of an image (interpreted as a height field or landscape) where rain would flow into the same lake.
- An efficient way to compute such regions
  - start flooding the landscape at all of the local minima
  - label ridges wherever differently evolving components meet.

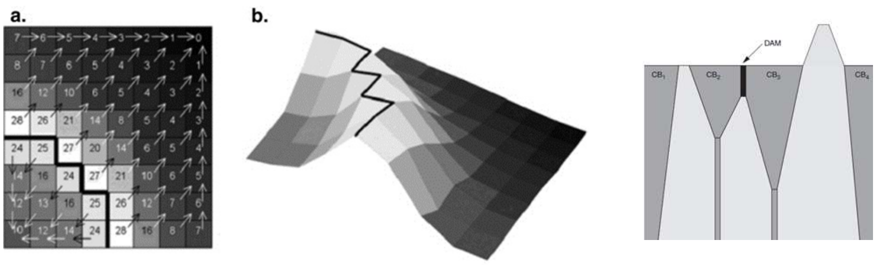
Courtesy of Emin Zerman and Mehmet Mutlu

METU EE 584 Lecture Notes by A.Aydin ALATAN © 2013

Watershed Algorithm

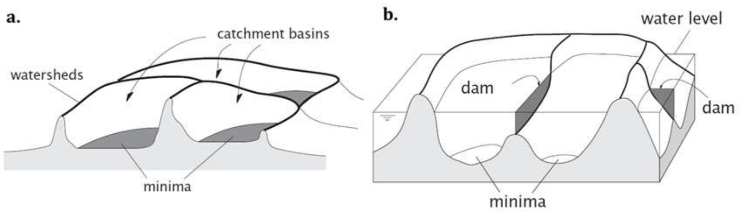
Method:

- Convert 2D image into a 3D topological map
  - Dark regions holes, bright regions peaks
- Bore holes at all local minimas
- Immerse the surface into water at constant speed
- Different label for all minimas.
- drop water on an unclassified pixel and tag the pixel as the final destination of the water drop.



Courtesy of Emin Zerman and Mehmet Mutlu

Watershed Algorithm

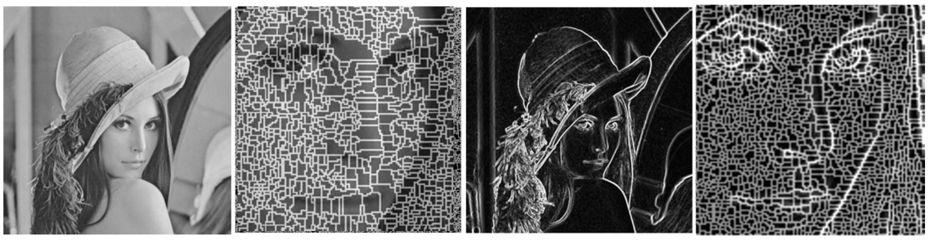


a *Minima, catchment basins, and watersheds on the topographic representation of a gray-scale image.* ***b*** *Building dams at the places where the water coming from two different minima would merge.*

Courtesy of Emin Zerman and Mehmet Mutlu

Watershed Algorithm

- Watershed segmentation on gradient of image usually gives segments mostly separated from edges.
  - Derivative operator is susceptible to noise → Filter image
- Watershed segmentation suffers from over segmentation.
  - All local minima represent a segmented region.

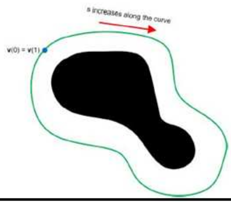


- Watershed applied on original image.
- Watershed applied on gradient image.

Courtesy of Emin Zerman and Mehmet Mutlu

Active Contours

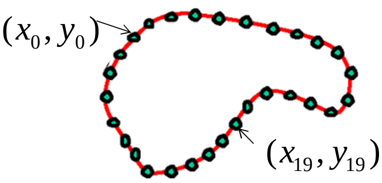
- Active Contours (snakes)
  - Energy minimizing closed curves placed on an image
  - Able to obtain and move salient image contours
    - Locating them by shape shifting
- Snakes are modeled as parametric curves:
  - $C(s) = [x(s), y(s)]^T \; s \in [0,1)$ and $C(0) = C(1)$



Courtesy of Yeti Ziya Gürbüz

Active Contours

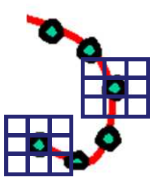
- Consider a discrete representation of the contour, consisting of a list of 2D point positions ("vertices").



$$v_i = (x_i, y_i),$$

for $i = 0, 1, \ldots, n-1$

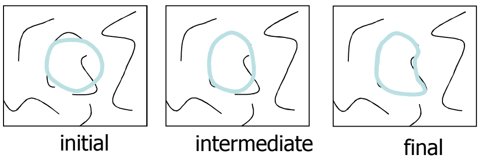
- For an active contour, at each iteration, move each vertex to another nearby location ("state") based on a "goodness" measure



Adopted from Kristen Grauman

Active Contours

- Salient image contours cannot be well-localized in low level processing
- Using some initialization localization of the edges can be improved
- Notice that such initializations are available:
  - Motion information, depth, prior knowledge...
- Aim: Induce a snake (active contour) on an image around the object and optimize its energy to locate the object of interest



Courtesy of Yeti Ziya Gürbüz

How Do Snakes Work ?

- Snakes has energies associated with them.
  - As a natural behaviour snakes deform and change shape to minimize energy.
- These energies are as follows:
  - $E_{snake} = \oint_0^1 (E_{int}(C(s)) + E_{ext}(C(s)))\, ds$

Internal — External

- How to define these energies?
  - So that snake will locate the object boundaries while minimizing its energy?

Courtesy of Yeti Ziya Gürbüz

Energies for Active Contours

- Define the internal energies such that
  - No external force ➔ Snake will be in a homogeneous, smooth shape; shrink or grow
  - With external force ➔ Snake will keep its shape continuous and smooth
- Define the external energies such that
  - Enforce snake to move to the boundaries of interest and take the shape of the boundaries
- By appropriate definitions of the energies:
  - Snake will converge to object boundaries

Courtesy of Yeti Ziya Gürbüz

Energies for Active Contours

- Define internal and external energies as:
  - $E_{int}(C(s)) = \frac{1}{2}(\alpha(s)\|C_s(s)\|^2 + \beta(s)\|C_{ss}(s)\|^2)$ (Kass[1])

Elasticity — Curvature Energy

$$E_{\text{int}} = \sum_i \alpha(i)\|f(i+1) - f(i)\|^2/h^2 + \beta(i)\|f(i+1) - 2f(i) + f(i-1)\|^2/h^4.$$

  - $E_{ext}(C(s)) = E_{img}(C(s)) + E_{con}(C(s))$
    - $E_{img} = -\|\nabla I(x,y)\|^2$ also $E_{img} = -\|\nabla[G_\sigma(x,y) * I(x,y)\|^2$ possible
    - $E_{con} \Rightarrow external\ constraint\ forces$

Courtesy of Yeti Ziya Gürbüz

Energies for Active Contours

- This energy should be minimized
  - Any gradient descent type algorithm should work
- Taking the negative gradient of the energy
  - ➔ Movement of snake & its convergence to a contour
- Note that gradient of the energy is force!

- Problems
  - Short capture range
  - Jumping local minimas
  - Cannot go into concave regions
  - Curvature estimation

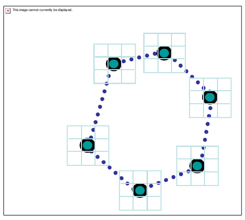
Courtesy of Yeti Ziya Gürbüz

Minimization of Energy

- Several algorithms have been proposed to fit deformable contours.
  - Greedy search
  - Dynamic programming

- For each point, search window around it and move to where energy function is minimal
- Stop when predefined number of points have not changed in last iteration, or after max number of iterations

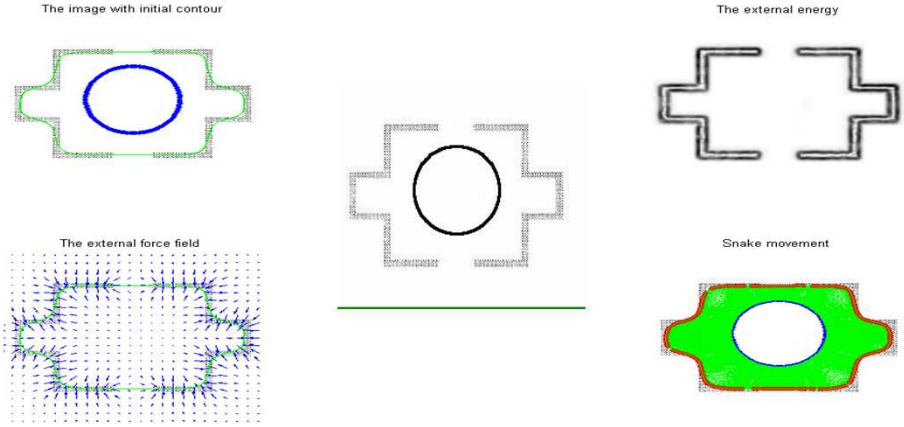
- Note:
  - Convergence not guaranteed
  - Need decent initialization



Adopted from Kristen Grauman

Active Contour Examples

- Growing of a Snake

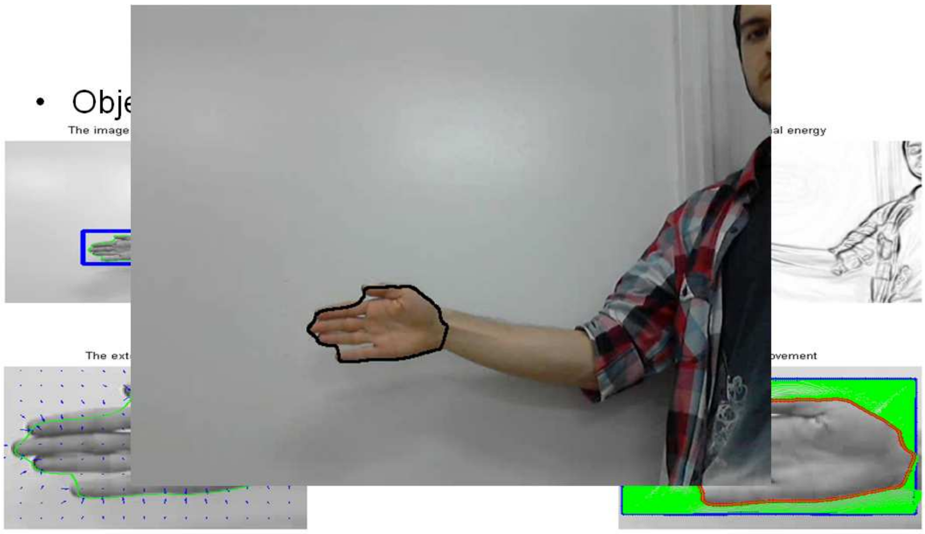


53

Courtesy of Yeti Ziya Gürbüz

Active Contour Examples

- Obje



54

Courtesy of Yeti Ziya Gürbüz

Segmentation Using Probabilistic Graphical Models

- Another graph-based approach for image segmentation (and other vision problems) are based on probabilistic models
- Unknown segmentation labels of pixels are modeled as a random field and being conditioned on the observed intensities, these labels are estimated

A toy example

Suppose we have a system of 5 interacting variables, perhaps some are observed and some are not. There's some probabilistic relationship between the 5 variables, described by their joint probability,

$$P(x1, x2, x3, x4, x5).$$

If we want to find out what the likely state of variable x1 is, what can we do?

Two reasonable choices are:

(a) find the value of $x1$ (and of all the other variables) that gives the maximum of $P(x1, x2, x3, x4, x5)$; (the MAP solution).

(b) marginalize over all the other variables and then take the mean or the maximum of the other variables. (the MMSE solution) $\sum_{x_2, x_3, x_4, x_5} P(x_1, x_2, x_3, x_4, x_5)$

slide from T. Darrel

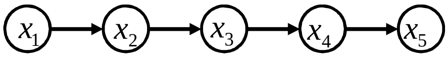
A toy example

If the system really is high dimensional, such solutions will quickly become intractable.

If there is some modularity in $P(x1, x2, x3, x4, x5)$, then things become tractable again.

Suppose the variables form a Markov chain:
"$x1$ *causes* $x2$ *which causes* $x3$, ... *etc*".

We might draw out this relationship as follows:



57

slide from T. Darrel

METU EE 584 Lecture Notes by A.Aydin ALATAN © 2013

Remember : P(a,b) = P(b|a) P(a)

By the chain rule, for any probability distribution, we have:

$$P(x_1,x_2,x_3,x_4,x_5) = P(x_1)P(x_2,x_3,x_4,x_5 \mid x_1)$$

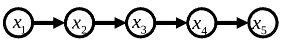
$$= P(x_1)P(x_2 \mid x_1)P(x_3,x_4,x_5 \mid x_1,x_2)$$

$$= P(x_1)P(x_2 \mid x_1)P(x_3 \mid x_1,x_2)P(x_4,x_5 \mid x_1,x_2,x_3)$$

$$= P(x_1)P(x_2 \mid x_1)P(x_3 \mid x_1,x_2)P(x_4 \mid x_1,x_2,x_3)P(x_5 \mid x_1,x_2,x_3,x_4)$$

If we exploit the assumed modularity of the probability distribution over the 5 variables (in this case, the assumed Markov chain structure), then that expression simplifies:

$$= P(x_1)P(x_2 \mid x_1)P(x_3 \mid x_2)P(x_4 \mid x_3)P(x_5 \mid x_4)$$



Our marginalization summations distribute through those terms:

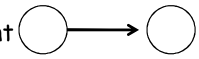
$$\sum_{x_2,x_3,x_4,x_5} P(x_1,x_2,x_3,x_4,x_5) = \quad P(x_1)\sum_{x_2} P(x_2 \mid x_1)\sum_{x_3} P(x_3 \mid x_2)\sum_{x_4} P(x_4 \mid x_3)\sum_{x_5} P(x_5 \mid x_4)$$

58

slide from T. Darrel

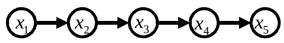
Directed graphical models

- A directed, acyclic graph.
- Nodes are random variables
  - Can be scalars or vectors, continuous or discrete.
- The direction of the edge tells the parent-child-relation:



- Every node i is associated by a conditional pdf defined by all the parent nodes π_i of node i.
  - The resulting conditional probability is denoted as $P_{x_i | x_{\pi_i}}$
- The joint distribution depicted by the graph is the product of all those conditional probabilities:

$$P_{x_1 \dots x_n} = \prod_{i=1}^{n} P_{x_i | x_{\pi_i}}$$

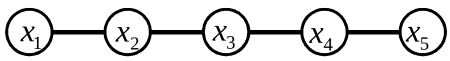


$$P(x_1, x_2, x_3, x_4, x_5) = P(x_1)P(x_2 | x_1)P(x_3 | x_2)P(x_4 | x_3)P(x_5 | x_4)$$

slide from T. Darrel

Undirected graphical models

Another modular probabilistic structure, more common in vision problems, is an undirected graph:



The joint probability for this graph is given by:

$$P(x_1, x_2, x_3, x_4, x_5) = \Phi(x_1, x_2)\Phi(x_2, x_3)\Phi(x_3, x_4)\Phi(x_4, x_5)$$

where $\Phi(x_1, x_2)$ is called a "compatibility function".

We can define compatibility functions and we result in the same joint probability as for the directed graph described in the previous slides

60

slide from T. Darrel

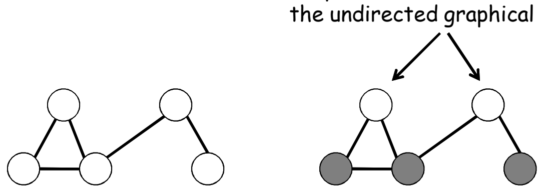
Undirected graphical models

Remember definition of conditional independency :

$$P(a,b|c) = P(a|c)\,P(b|c)$$

- A set of nodes joined by undirected edges.
- The graph makes conditional independencies explicit:
  - If two nodes are not linked, and we condition on every other node in the graph, then those two nodes are conditionally independent.

Conditionally independent, because they are not connected by a line in the undirected graphical model

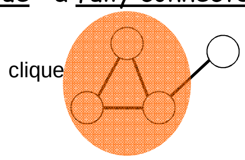


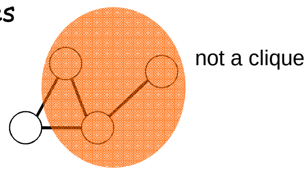
61

slide from T. Darrel

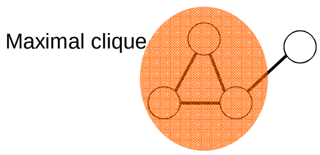
Undirected graphical models : cliques

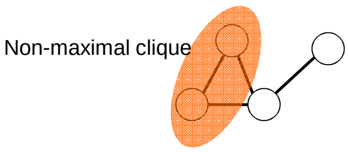
- Clique: a fully connected set of nodes





- A maximal clique is a clique that can not include more nodes of the graph w/o losing the clique property.





62

slide from T. Darrel

Undirected graphical models : Probability Factorization

- Hammersley-Clifford theorem addresses the pdf factorization implied by a graph:
  - The probability factorizes according to the cliques of the graph
- A distribution has the Markov structure implied by an undirected graph *iff* it can be represented in the factored form

$$P_x = \frac{1}{Z} \prod_{c \in \xi} \Psi_{x_c}$$

Normalizing constant

Potential functions of states of variables in maximal clique

set of maximal cliques

slide from T. Darrel

METU EE 584 Lecture Notes by A.Aydin ALATAN © 2013

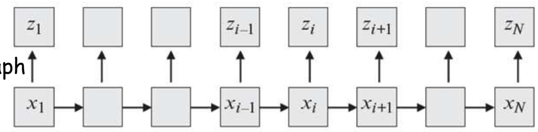
Hidden Markov Models

Let $\mathbf{x}=(x1,x2,\ldots,xN)$ be (unknown) state variables (i.e. segmentation labels)
Let $\mathbf{z}=(z1,z2,\ldots,zN)$ be (measured) observations (i.e. image pixel values)

$$P(\mathbf{X}=\mathbf{x} \mid \mathbf{Z}=\mathbf{z}) \propto P(\mathbf{Z}=\mathbf{z} \mid \mathbf{X}=\mathbf{x}) P(\mathbf{X}=\mathbf{x})$$

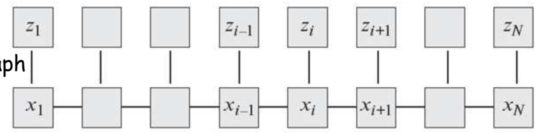
$P(\mathbf{X}=\mathbf{x} \mid \mathbf{Z}=\mathbf{z})$: posteriori distribution
$P(\mathbf{X}=\mathbf{x})$: prior probability (Markov chain)

Directed graph



$$P(\mathbf{z} \mid \mathbf{x}) = P(z_N \mid x_N) P(z_{N-1} \mid x_{N-1}) \ldots P(z_1 \mid x_1).$$

Undirected graph



$$P(\mathbf{z} \mid \mathbf{x}) = \Phi_N(x_N)\Phi_{N-1}(x_{N-1}) \ldots \Phi_1(x_1),$$

where trivially $\Phi_i(x_i) = P(z_i \mid x_i)$.

Hidden Markov Models

Three canonical problems for discrete HMMs.

1. Evaluating the observation probability $P(\mathbf{z} \mid \omega)$ ← model parameters

$$P(\mathbf{z} \mid \omega) = \sum_{\mathbf{x} \in \mathcal{L}^N} P(\mathbf{z} \mid \mathbf{x}, \omega) P(\mathbf{x} \mid \omega). \qquad \max_{\omega \in \Omega} P(\mathbf{z} \mid \omega).$$

$$P(x_i, z_1, \ldots, z_i \mid \omega) = P(z_i \mid x_i, \omega) \sum_{x_{i-1}} P(x_i \mid x_{i-1}, \omega) P(x_{i-1}, z_1, \ldots, z_{i-1} \mid \omega).$$

$$P(\mathbf{z} \mid \omega) = \sum_{x_N} P(x_N, \mathbf{z} \mid \omega)$$

2. MAP estimation $\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} P(\mathbf{x} \mid \mathbf{z}, \omega)$

$$P(\mathbf{x} \mid \mathbf{z}, \omega) \propto P(\mathbf{z} \mid \mathbf{x}, \omega) P(\mathbf{x} \mid \omega),$$

where the model parameters $\omega \in \Omega$

3. Parameter estimation : Given the observation, z, find the optimal parameters, ω.

MRFs: Markov Models on Graphs

- <u>Hammersley-Clifford theorem</u> $P(\mathbf{x}) = \prod_{c \in \mathcal{C}} F_c(\mathbf{x})$

The factorized probabilities are assumed to be Gibbs:

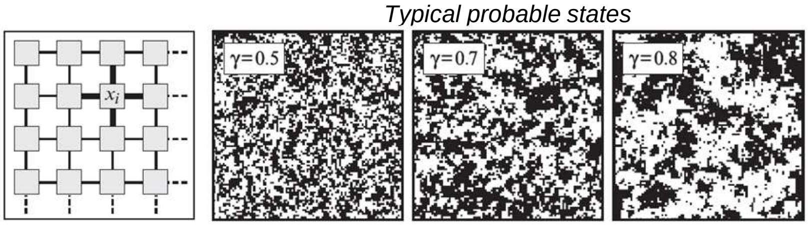
$$P(\mathbf{x}) = \frac{1}{Z(\omega)} \exp(-E(\mathbf{x}, \omega)) \text{ where } E(\mathbf{x}, \omega) = \sum_{c \in \mathcal{C}} \Psi_c(\mathbf{x}_c, \omega),$$

- <u>Example</u>: *Ising Model* with single parameter $\omega=\{\gamma\}$

$$x_i \in \mathcal{L} = \{0, 1\} \qquad \Psi_{ij}(x_i, x_j) = \gamma |x_i - x_j|.$$

whenever x_i and x_j different from each other, a penalty γ decreases $P(\mathbf{x})$ by e^{γ} .

Typical probable states



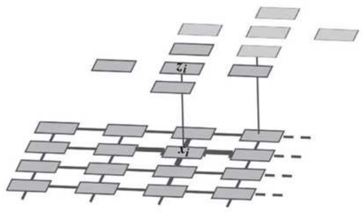
Hidden MRFs

- A Markov Random field, as before, can act as a prior model for a set of hidden random variables, $\mathbf{X}$, under a set of observations $\mathbf{z}$.

$$P(\mathbf{x} \mid \mathbf{z}, \omega) \propto P(\mathbf{z} \mid \mathbf{x}, \omega) P(\mathbf{x} \mid \omega),$$

$$P(\mathbf{x} \mid \mathbf{z}, \omega) = \frac{1}{Z(\mathbf{z}, \omega)} \exp -E(\mathbf{x}, \mathbf{z}, \omega),$$

$$\text{where } E(\mathbf{x}, \mathbf{z}, \omega) = \sum_{c \in \mathcal{C}} \Psi_c(\mathbf{x}, \omega) + \sum_i \Phi_i(x_i, z_i).$$



$$\hat{\mathbf{x}} = \arg\max P(\mathbf{x} \mid \mathbf{z}) \qquad \hat{\mathbf{x}} = \arg\min E(\mathbf{x}, \mathbf{z}, \omega).$$

$$E(\mathbf{x}, \mathbf{z}, \omega) = \sum_{i \in \mathcal{V}} \Phi_i(x_i, z_i, \omega) + \sum_{(i,j) \in \mathcal{E}} \Psi_{ij}(x_i, x_j, \omega).$$

METU EE 584 Lecture Notes by A.Aydin ALATAN © 2013

Image Segmentation using MRFs

- How to apply (hidden) MRFs to image segmentation?
  - X : the (unknown) label of the pixel indicating the segment index
  - z : the observed image
  - ω : model parameters.

$$E(\mathbf{x}, \mathbf{z}, \omega) = \sum_{i \in \mathcal{V}} \Phi_i(x_i, z_i, \omega) + \sum_{(i,j) \in \mathcal{E}} \Psi_{ij}(x_i, x_j, \omega).$$

- Some typical energy functions:

$$\Phi_i(z_i) = \log h_F(z_i) - \log h_B(z_i)$$

$h_{F,B}$: image histogram of Bg/Fg

$$\Psi_{ij}(x_i, x_j) = \gamma |x_i - x_j|.$$

Isling model

$$\sum_{n \in T_U} -\log h_B(z_i)[x_n = 0] - \log h_F(z_i)[x_n = 1] + \sum_{n \in T_F \cup T_B} H(x_n, n)$$

H(.,.) : constraints certain variables to Fg/Bg.

$$\sum_{(m,n) \in \mathcal{N}} \text{dis}(m,n)^{-1} (\lambda_1 + \lambda_2 \exp\{-\beta \|z_m - z_n\|^2\}) [x_n \neq x_m],$$

Considers distance between pixels and image contrast

How to determine MAP estimates for MRFs?

A number of methods exist for minimization of E

$$E(\mathbf{x}, \mathbf{z}, \omega) = \sum_{i \in \mathcal{V}} \Phi_i(x_i, z_i, \omega) + \sum_{(i,j) \in \mathcal{E}} \Psi_{ij}(x_i, x_j, \omega).$$

- Iterated Conditional Modes (ICM)
  - Choose an arbitrary location with the label, x_i
  - Change value of x_i so that E has maximum decrease
  - Repeat until there is no change in E.

  No guarantee for a global minimum; suboptimal.
- Simulated Annealing
- Belief Propagation
- Min-Cut / Max-Flow (binary-labeling optimal !)

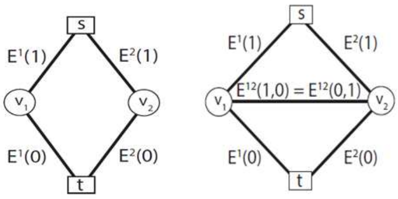
MRF Energy Minimization via Min-Cut

Assume the following (MRF) energy, $E(x_1, x_2)$, on a binary labeling (2 segments) problem with only two variables x_1 and x_2.

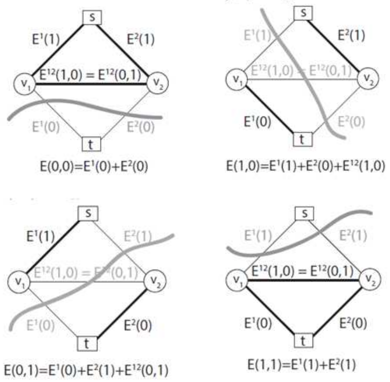
Aim is minimizing this energy (i.e. maximizing probability) wrt x_i.

Convert the problem into a graph with 2 extra nodes (s,t)

$$E(x_1, x_2) = \sum_{i \in 1,2} E^i(x_i) + \sum_{e_{ij} \in \mathcal{E}} E^{ij}(x_i, x_j)$$
$$= E^1(x_1) + E^2(x_2) + E^{12}(x_1, x_2).$$



e.g. $x_1 = 0$ ➔ v_1 belongs s

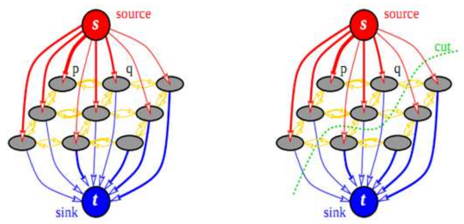


Courtesy by Ozan Sener

MRF Energy Minimization via Min-Cut

Minimization of MRF energy on binary labeling problem is equivalent to finding minimum cut.

$$E(\mathbf{x}, \mathbf{z}, \omega) = \sum_{i \in \mathcal{V}} \Phi_i(x_i, z_i, \omega) + \sum_{(i,j) \in \mathcal{E}} \Psi_{ij}(x_i, x_j, \omega).$$



(a) A graph $\mathcal{G}$ (b) A cut on $\mathcal{G}$

edge	weight (cost)	for
$\{p,q\}$	$B_{\{p,q\}}$	$\{p,q\} \in \mathcal{N}$
$\{p,S\}$	$\lambda \cdot R_p(\text{"bkg"})$	$p \in \mathcal{P},\ p \notin \mathcal{O} \cup \mathcal{B}$
	K	$p \in \mathcal{O}$
	0	$p \in \mathcal{B}$
$\{p,T\}$	$\lambda \cdot R_p(\text{"obj"})$	$p \in \mathcal{P},\ p \notin \mathcal{O} \cup \mathcal{B}$
	0	$p \in \mathcal{O}$
	K	$p \in \mathcal{B}$

where

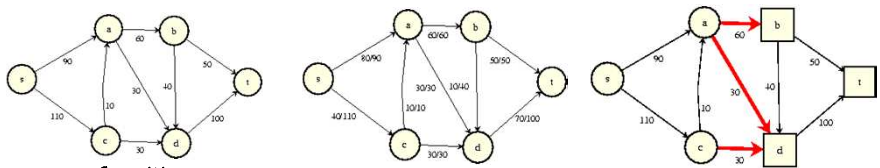
$$K = 1 + \max_{p \in \mathcal{P}} \sum_{q:\ \{p,q\} \in \mathcal{N}} B_{\{p,q\}}.$$

O: Set of nodes selected as object
B: Set of nodes selected as background

Courtesy by Ozan Sener

Finding Min-Cut

- Min-cut/max-flow equivalance theorem:
  - Finding minimum cut is equivalent to finding maximum flow from source to sink.
    - Abtraction: "*Maximum amount of water that can be sent from source to sink by interpreting edges as directed pipes with capacities as weights*"



Capacities Max-flow = 120 Min-cut = 120

- Max-flow can be obtained by augmenting paths algorithm.

Courtesy by Ozan Sener

Augmenting Paths Algorithm

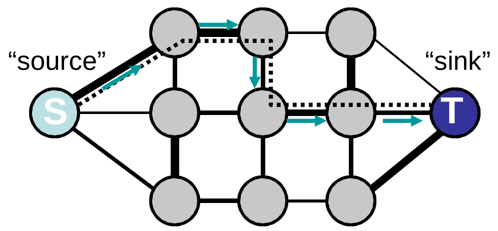
- Find shortest s-t path on the unsaturated edges of graph
- Push maximum available flow through this s-t path.
- Iterate until there exist no s-t path with at least one unsaturated edge

Courtesy by Ozan Sener

Slide by Yuri Boykov

METU EE 584 Lecture Notes by A.Aydin ALATAN © 2013

Augmenting Paths Algorithm



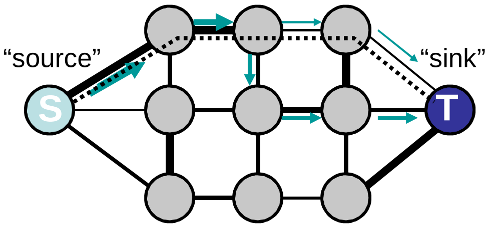
A graph with two terminals

- Find a path from S to T along non-saturated edges
- Increase flow along this path until some edge saturates

Slide by Yuri Boykov

METU EE 584 Lecture Notes by A.Aydin ALATAN © 2013

Augmenting Paths Algorithm



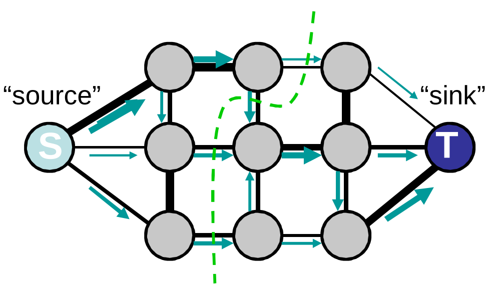
A graph with two terminals

- Find a path from S to T along non-saturated edges
- Increase flow along this path until some edge saturates
- Find next path…
- Increase flow…

Slide by Yuri Boykov

METU EE 584 Lecture Notes by A.Aydin ALATAN © 2013

Augmenting Paths Algorithm



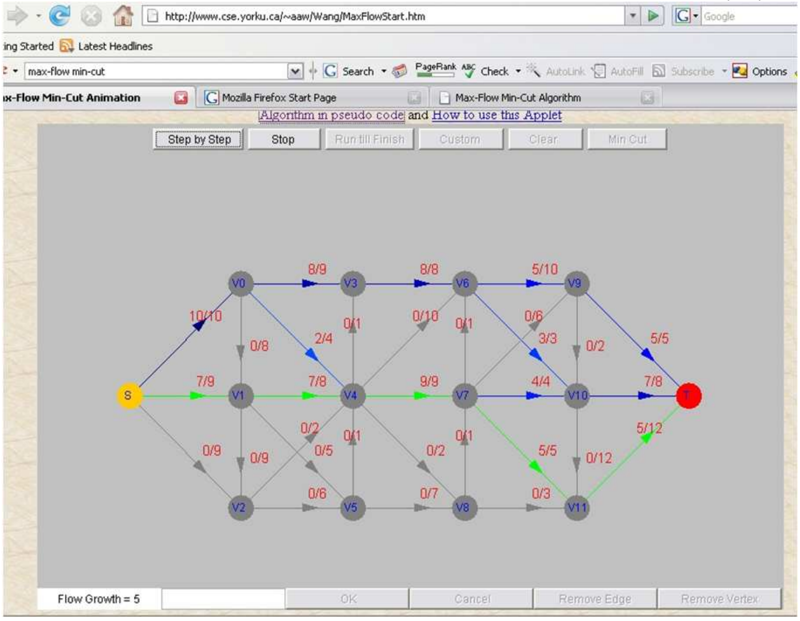
A graph with two terminals

- Find a path from S to T along non-saturated edges
- Increase flow along this path until some edge saturates

Iterate until … all paths from S to T have at least one saturated edge

MAX FLOW ⇔ MIN CUT

Break between the two sets always cutting connections that are at capacity



http://www.cse.yorku.ca/~aaw/Wang/MaxFlowStart.htm

Segmentation Using Probabilistic Graphical Models

Summary

- Observed image pixels and unknown segment labels are modeled via (hidden) MRF formulation
- A (Gibbs) energy is minimized to determine the unknown segmentation labels
- Minimization of the energy is achieved by using min-cut/max-flow solution in an optimal way for binary problems
- Unknown model parameters of segments should be provided
  - User interaction or other a priori information