Image Segmentation

Some slides: courtesy of O. Capms, Penn State, J.Ponce and D. Fortsyth, Computer Vision Book

Regions and Edges

- **Edges** are found based on **DIFFERENCES** between values of adjacent pixels.
- **Regions** are found based on **SIMILARITIES** between values of adjacent pixels.
- Goal associate some higher level – more meaningful units with the regions of the image

J. Kosecka

Regions and Edges

- Ideally, regions are bounded by closed contours
  - We could "fill" closed contours to obtain regions
  - We could "trace" regions to obtain edges
- Unfortunately, these procedures rarely produce satisfactory results.


J. Kosecka

Segmentation

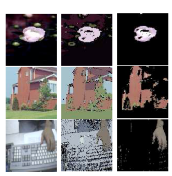
- Useful mid-level representation of an image - can facilitate better further tasks
- Partitioning image into regions should be homogeneous with respect to some characteristic
- (gray level, texture, color, motion)

- The type of desired segmentation depends on the task
- Broad theory is absent at present
- Variety of approaches/algorithms
- Applications finding people, summarizing video, annotation figures, background subtraction, finding buildings/rivers in satellite images

J. Kosecka

Segmentation and Grouping

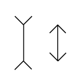
- Grouping (or clustering)
  - collect together tokens that "belong together"
- Fitting
  - associate a model with tokens
  - issues
    - which model?
    - which token goes to which element?
    - how many elements in the model?



J. Kosecka

Grouping in humans

- Figure-ground discrimination
  - grouping can be seen in terms of allocating some elements to a figure, some to ground
  - impoverished theory



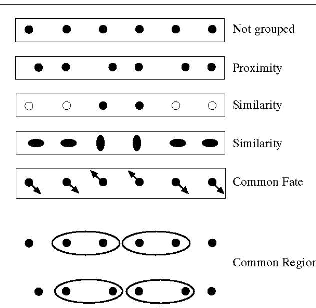
- Gestalt properties
  - elements in a collection of elements can have properties that result from relationships (Muller-Lyer effect)
    - Gestalt-qualitat
  - A series of factors affect whether elements should be grouped together
    - Gestalt factors

J. Kosecka

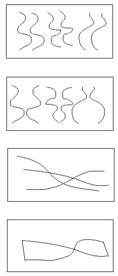
Examples of grouping

- Group video to shots
- Object –level grouping (find cars, bikes)
- Determine image regions belonging to objects
- Group foreground/background pixels

J. Kosecka

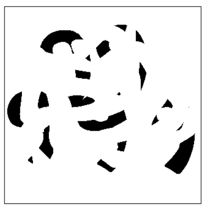


J. Kosecka



J. Kosecka

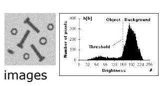
J. Kosecka



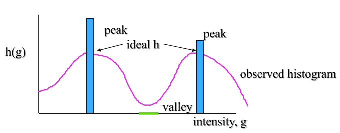
J. Kosecka

J. Kosecka

Binary segmentation

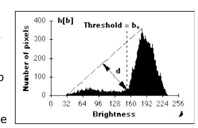


- Segmentation for simple binary images
- How do we choose the threshold t for segmentation?
- Histogram (h) - gray level frequency distribution of the gray level image F.
  - $h_F(g)$ = number of pixels in F whose gray level is g
  - $H_F(g)$ = number of pixels in F whose gray level is <=g



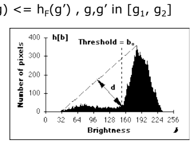
Triangle algorithm

- A line is constructed between the maximum of the histogram at brightness b_{max} and the lowest value b_{min} = (p=0)% in the image.
- The distance d between the line and the histogram h[b] is computed for all values of b from b = b_{min} to b = b_{max}.
- The brightness value b_o where the distance between $h[b_o]$ and the line is maximal is the threshold value.
- This technique is particularly effective when the object pixels produce a weak peak in the histogram.



Thresholding

- Peak and valley method
  - Find the two most prominent peaks of h
    - g is a peak if $h_F(g) > h_F(g \pm \Delta g)$, $\Delta g = 1, \ldots, k$
  - Let g_1 and g_2 be the two highest peaks, with $g_1 < g_2$
  - Find the deepest valley, g, between g_1 and g_2
    - g is the valley if $h_F(g) <= h_F(g')$, g,g' in $[g_1, g_2]$
  - Use g as the threshold



Thresholding

- Hand selection
  - select a threshold by hand at the beginning of the day
  - use that threshold all day long!
- Many threshold selection methods in the literature
  - Probabilistic methods
    - make parametric assumptions about object and background intensity distributions and then derive "optimal" thresholds
  - Structural methods
    - Evaluate a range of thresholds wrt properties of resulting binary images
      - one with straightest edges, most easily recognized objects, etc.
  - Local thresholding
    - apply thresholding methods to image windows

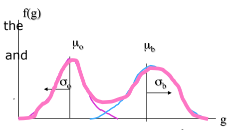
An advanced probabilistic threshold selection method - minimizing Kullback information distance

- Suppose the observed histogram, f, is a mixture of the gray levels of the pixels from the object(s) and the pixels from the background
  - in an ideal world the histogram would contain just two spikes (this depends of the class of images/ objects)
  - but
    - measurement noise
    - model noise (e.g., variations in ink density within a character)
    - edge blur (misalignment of object boundaries with pixel boundaries and optical imperfections of camera)

  spread these spikes out into hills

Kullback information distance

- Make a parametric model of the shapes of the component histograms of the objects(s) and background
- Parametric model - the component histograms are assumed to be Gaussian
  - p_o and p_b are the proportions of the image that comprise the objects and background
  - μ_o and μ_b are the mean gray levels of the objects and background
  - σ_o and σ_b- are their standard deviations



$$f_o(g) = \frac{p_o}{\sqrt{2\pi}\sigma_o} e^{-1/2(\frac{g-\mu_o}{\sigma_o})^2}$$

$$f_b(g) = \frac{p_b}{\sqrt{2\pi}\sigma_b} e^{-1/2(\frac{g-u_b}{\sigma_b})^2}$$

Kullback information distance

- Now, if we hypothesize a threshold, t, then all of these unknown parameters can be approximated from the image histogram.
- Let f(g) be the observed and normalized histogram
  - f(g) = percentage of pixels from image having gray level g

$$p_o(t) = \sum_{g=0}^{t} f(g) \qquad p_b(t) = 1 - p_0(t)$$

$$\mu_o(t) = \sum_{g=0}^{t} f(g)g \qquad \mu_b(t) = \sum_{g=t+1}^{max} f(g)g$$

Kullback information distance

- So, for any hypothesized t, we can "predict" what the total normalized image histogram **should** be if our model (mixture of two Gaussians) is correct.
  - $P_t(g) = p_o f_o(g) + p_b f_b(g)$
- The total normalized image histogram is **observed to be** f(g)
- So, the question reduces to:
  - determine a suitable way to measure the <u>similarity</u> of P and f
  - then search for the t that gives the highest similarity

Kullback information distance

- A suitable similarity measure is the Kullback directed divergence, defined as

  If P_t matches f exactly, then each term of the sum is 0 and K(t) takes on its minimal value of 0
- Gray levels where P_t and f disagree are penalized by the log term, weighted by the importance of that gray level (f(g))

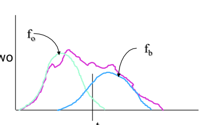
$$K(t) = \sum_{g=0}^{max} f(g) \log[\frac{f(g)}{P_t(g)}]$$

An alternative - minimize probability of error

- Using the same mixture model, we can search for the t that minimizes the predicted probability of error during thresholding
- Two types of errors
  - background points that are marked as object points. These are points from the background that are darker than the threshold
  - object points that are marked as background points. These are points from the object that are brighter than the threshold

An alternative - mimimize probability of error

- For each "reasonable" threshold
  - compute the parameters of the two Gaussians and the proportions
  - compute the two probability of errors
- Find the threshold that gives
  - minimal overall error
  - most equal errors



$$e_b(t) = p_b \sum_{g=0}^{t} f_b(g)$$

$$e_o(t) = p_o \sum_{g=t+1}^{max} f_o(g)$$

Segmentation by Clustering

- Pattern recognition
- Process of partitioning a set of 'patterns' into clusters
- Find subsets of points which are close together

- Examples
- \- Cluster pixels based on intensity values
- \- Color properties
- \- Motion/optical flow properties
- \- Texture measurements etc.

Input – set of measurements x1, x2, …, xm

Output – set of clusters and their centers

J. Kosecka

Clustering

- Find set of clusters such that the least squares

Error is minimized

$$E = \sum_{k=1}^{K} \sum_{x_i \in C_i} \|x_i - m_k\|^2$$

Iterative K-means clustering algorithm
1. set iter = 1;
2. Choose randomly K-means m1, … mk
3. For each data point xi, compute distance to each of the means and assign the point the cluster with the nearest mean
4. iter = iter + 1
5. Recompute the means based on the new assignments of points to clusters
6. Repeat 3-5 until the cluster centers do not change much

J. Kosecka

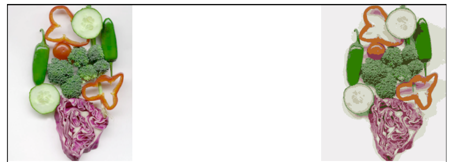


Image | Clusters on color

K-means using color alone, 11 segments

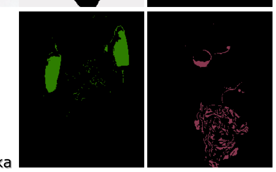
J. Kosecka

Image | Clusters on intensity | Clusters on color

K-means clustering using intensity alone and color alone

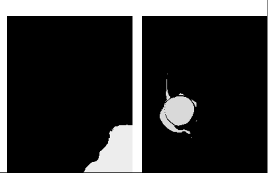
J. Kosecka

K-means using color alone, 11 segments.



J. Kosecka

K-means using colour and position, 20 segments

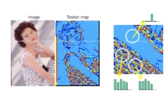


J. Kosecka

Segmentation and clustering

- Texture based clustering
- Group pixels based on texture similarity
- Texture representation
- Cluster output of the filter banks – clusters are so called textons
- Example – histograms of textons computed over window

- In the lecture on texture – we can classify the entire image based on histogram of textons
- Recognition was done by finding a images with the closest histogram
- Histogram distance was measure using chi-squared distance between histograms



$$\chi^2(h_i, h_j) = \sum_{1}^{K} \frac{(h_i(k) - h_j(k))^2}{h_i(k) + h_j(k)}$$

Clustering

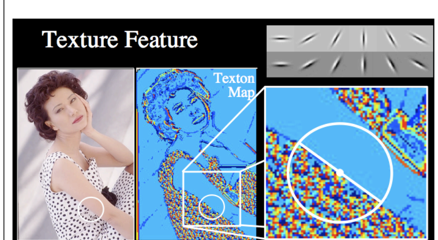
- Pros
- simple, fast to compute
- If k is large (approximate nearest neighbour methods for computing distances to clusters)
- Converges to local minimum of within cluster squared error

- Cons
- How large is K ?
- Sensitive to initial outliers
- Detection of spherical clusters
- Assumes that means can be computed

- Issues: Depending what we choose as feature space we get different clusters (color, textures, motion etc)
- Clusters often not spatially coherent

J. Kosecka

Texture Segmentation



J. Kosecka

Segmentation with EM

- There are n – pixels and g groups – compute how likely is a pixel belonging to group and also what are the parameters of the group
- Probabilistic K-means clustering
- E.g. Use of texture and color cues

J. Kosecka

Segmentation with EM

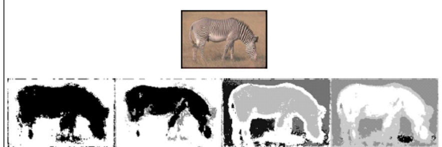


Figure from "Color and Texture Based Image Segmentation Using EM and Its Application to Content Based Image Retrieval",S.J. Belongie et al., Proc. Int. Conf. Computer Vision, 1998, c1998, IEEE

J. Kosecka

Motion Segmentation

Given a set of image points obtain:
- Number of independently moving objects
- Segmentation: object to which each point belongs
- Motion: rotation and translation of each object
- Structure: depth of each point

J. Kosecka

Motion Segmentation Problem

Three frames from the MPEG "flower garden" sequence

- Given optical flow at each point
- partition/segment the flow field into regions belonging to individual planes "layers"

Figure from "Representing Images with layers,", by J. Wang and E.H. Adelson, IEEE Transactions on Image Processing, 1994, c 1994, IEEE

J. Kosecka

Some example slides from Forsythe and Ponce. Computer Vision, A modern approach.

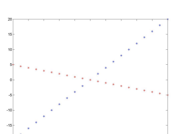
Model Estimation and Grouping

- Given a set of data points and a particular model
- The model parameters can be estimated by LSE fitting data to the model
- Previous model examples – essential/fundamental matrix, homographies, lines/planes
- In order to estimate the model parameters we need to know which data point belongs to which model
- Difficulty – we have multiple models – we do not know initially which data point belongs to which model and we do not the model parameters
- chicken and egg problem

J. Kosecka

Model Estimation and Grouping

- Line Example
- Set of points belonging to two lines
- We need to estimate
  1. parameters of the lines slope and intercept
  2. which point belongs to which line



Solution: EM algorithm
Idea: Each of the above steps Assumes the other one is solved and iterate

J. Kosecka

EM algorithm

- Basic structure of the EM algorithm
- 1. Start with random parameter values for each model
- 2. Iterate until parameter values converge
  - E step: assign points to the model that fits best
  - M step : update the parameters of the models using only points assigned to it

Simplistic explanation here –
Theoretical foundation probabilistic (model parameters are random variables) - EM (Expectation Maximization)

J. Kosecka

E- Step

- Case of two lines given by slopes and intercepts (a_1, b_1) and (a_2, b_2)
- For each data point i, estimate the residual (difference between the prediction and the model)

$$r_1(i) = a_1 x_i + b_1 - y_i$$
$$r_2(i) = a_2 x_i + b_2 - y_i$$

- Calculate the weights, which correspond to the probabilities of particular data point belonging to particular model

$$w_1(i) = \frac{e^{-r_1^2(i)/\sigma^2}}{e^{-r_1^2(i)/\sigma^2} + e^{-r_2^2(i)/\sigma^2}} \qquad w_2(i) = \frac{e^{-r_2^2(i)/\sigma^2}}{e^{-r_1^2(i)/\sigma^2} + e^{-r_2^2(i)/\sigma^2}}$$

J. Kosecka

M-step

- Given the weights recalculate the parameters of the model (a_1, b_1) and (a_2, b_2)
- Least squares estimation of line parameters

$$\begin{bmatrix} \sum_i x_i^2 & \sum_i x_i \\ \sum_i x_i & \sum 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum_i x_i y_i \\ \sum_i y_i \end{bmatrix}$$

- In our case we will have weighted least squares estimation of line parameters

$$\begin{bmatrix} \sum_i w_i x_i^2 & \sum_i w_i x_i \\ \sum_i w_i x_i & \sum w_i 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum_i w_i x_i y_i \\ \sum_i w_i y_i \end{bmatrix}$$

- Solve such estimation problem twice – once for each line

J. Kosecka

M-step

- Iterate until the parameters of the
- Lines converge

- Issues with EM
- Local maxima
  - can be a serious nuisance in some problems
  - no guarantee that we have reached the "right" maximum
- Starting if we do not know how many models we have
  - k means to cluster the points is often a good idea

J. Kosecka

Example: motion segmentation

- Consider motion model, when the flow field can be approximated by some parametric model with small number of parameters
- We can write x and y parameters of the flow field – assume that models are locally translational, i.e. we can locally approximate the model by pure translation (u, v)
- Suppose entire flow field can be explained by two translational models (u_1, v_1) (u_2, v_2)
- EM algorithm can be applied in analogous way

J. Kosecka

Example: motion segmentation

- Compute residuals

$$r_1(i,j)^2 = (u_1 - v_x(i,j))^2 + (v_1 - v_y(i,j)^2)$$

$$r_2(i,j)^2 = (u_2 - v_x(i,j))^2 + (v_2 - v_y(i,j)^2)$$

- Compute associated weights

$$w_1(i,j) = \frac{e^{-r_1^2(i,j)/\sigma^2}}{e^{-r_1^2(i,j)/\sigma^2} + e^{-r_2^2(i,j)/\sigma^2}} \quad w_2(i,j) = \frac{e^{-r_2^2(i,j)/\sigma^2}}{e^{-r_1^2(i,j)/\sigma^2} + e^{-r_2^2(i,j)/\sigma^2}}$$

- M-step analogous to line fitting

$$\begin{bmatrix} \sum_{i,j} w_1(i,j) & 0 \\ 0 & \sum_{i,j} w_1(i,j) \end{bmatrix} \begin{bmatrix} u_1 \\ v_1 \end{bmatrix} = \begin{bmatrix} \sum_{i,j} w_1(i,j) v_x(i,j) \\ \sum_{i,j} w_1(i,j) v_y(i,j) \end{bmatrix}$$

Iterate until convergence

J. Kosecka

Example: motion segmentation

- Model image pair (or video sequence) as consisting of regions of parametric motion
  - affine motion – commonly used –
  - Approximates locally motion of the planar surface

$$\begin{bmatrix} v_x(x,y) \\ v_y(x,y) \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} d_1 \\ d_2 \end{bmatrix}$$

- Now we need to
  - determine which pixels belong to which region
  - estimate parameters

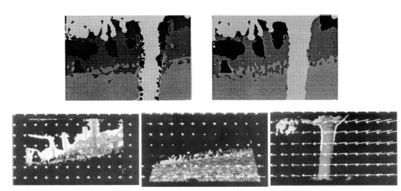
J. Kosecka

Three frames from the MPEG "flower garden" sequence

Figure from "Representing Images with layers,", by J. Wang and E.H. Adelson, IEEE Transactions on Image Processing, 1994, c 1994, IEEE

J. Kosecka

Grey level shows region no. with highest probability



Segments and motion fields associated with them

Figure from "Representing Images with layers,", by J. Wang and E.H. Adelson, IEEE Transactions on Image Processing, 1994, c 1994, IEEE

J. Kosecka

Other examples

- Segmentation
  - a segment is a gaussian that emits feature vectors (which could contain color; or color and position; or color, texture and position).
  - segment parameters are mean and (perhaps) covariance
  - if we knew which segment each point belonged to, estimating these parameters would be easy

J. Kosecka

Some generalities

- Many, but not all problems that can be attacked with EM can also be attacked with RANSAC
  - need to be able to get a parameter estimate with a manageably small number of random choices.
  - RANSAC is usually better
- We wish to choose a model to fit to data
  - e.g. is it a line or a circle?
  - e.g is this a perspective or orthographic camera?
  - e.g. is there an airoplane there or is it noise?
  - In general, models with more parameters will fit a dataset better, but are poorer at prediction

J. Kosecka

Segmentation as Graph Partitioning

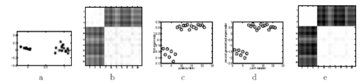
- (Shi & Malik ''97)
- Idea – each pixel in the image is a node in the graph
- Arcs represent similarities between adjacent pixels
- Graph is fully connected
- Goal – partition the graph into a sets of vertices (regions), such that the similarity within the region is high – and similarity across the regions is low.

- See textbook (Ponce and Forsythe) for detailed description the algorithm.

J. Kosecka

Segmentation

- Toy example
- Bright entries in the affinity matrix high
- Likely to belong together



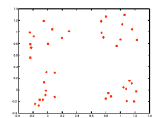
- one possible affinity based on distance

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

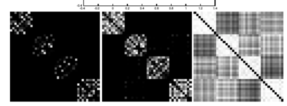
J. Kosecka

Scale affects affinity

Depending of the scale the blocks are more (middle)
Or less obvious (left and right)



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



J. Kosecka $\sigma_d = \{0.1, 0.2, 1\}$

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

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

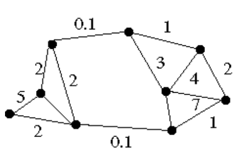
Texture

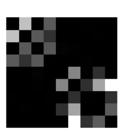
$$aff(x,y) = \exp\left\{-\left(\frac{1}{2\sigma_t^2}\right)\left(\|c(x)-c(y)\|^2\right)\right\}$$

J. Kosecka

Graph theoretic clustering

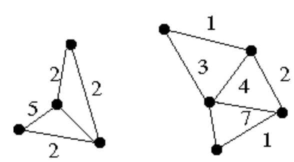
- Represent tokens using a weighted graph
- affinity matrix
- Cut up this graph to get subgraphs with strong interior links
- Delete links which have weak connections





J. Kosecka

Example: Graph theoretic clustering



J. Kosecka

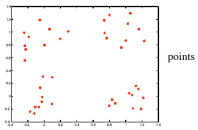
Eigenvectors and segmentation
Spectral clustering

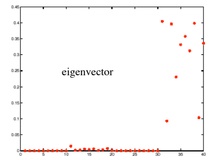
- Simplest idea: we want a vector giving the association between each element and a cluster
- We want elements within this cluster to, on the whole, have strong affinity with one another
- We could maximize
  $a^T A a$
- But need the constraint
  $a^T a = 1$

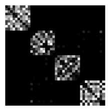
To avoid arbitrary scaling of entries of vector *a*

- This is an eigenvalue problem - choose the eigenvector of A with largest eigenvalue - single good cluster
- Vector *a* – indicator vector denoting how likely is the element to be associated with the cluster

Example eigenvector







- Entries associated with the largest cluster are high
- Searching for additional clusters – look at
- Eigenvectors associated with second largest Eigenvalue etc

J. Kosecka

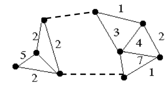
More than two segments

- Reasoning about other eigenvectors - consider that affinity matrix is block diagonal.
- Until there are sufficient clusters pick eigenvector associated with the largest eigenvalue, zero the elements which were clustered, threshold elements with large association weights - those will form a new cluster
- Keep going until there is sufficient number of clusters and all elements have been accounted for
- Spectral Clustering Techniques (A. Ng and M. Jordan)
- Problems - if the eigenvalues are similar - eigenvectors do not reveal the clusters
- Normalized cut - graph cut - alternative optimization criterion J. Shi and J. Malik

J. Kosecka

Normalized Cuts

- Find set of links whose removal will make the graph disconnected



http://www.cis.upenn.edu/~jshi/GraphTutorial/

- Min cut idea $cut(A,B) = \sum_{p \in A, q \in B} w_{p,q}$
- Tends to produce small isolated clusters
- Normalized cut
$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(A,V)}$$
- *assoc(A,V)* sum of weight of all edges with one end in A

Normalized cuts

- Goal is to minimzie Ncut values
- In general NP-complete
- Approximate solutions for minimizing the Ncut value: generalized eigenvalue problem
$$\max_y \left(y^T (D-W) y\right) \text{ subject to } \left(y^T D y = 1\right)$$
$$(D-W)y = \lambda D y$$
- Now look for a quantization threshold that maximises the criterion --- i.e all components of y above that threshold go to one, all below go to –b
- More details in the tutorial slides

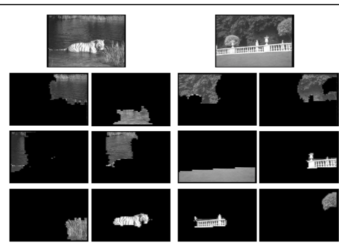


Figure from "Image and video segmentation: the normalised cut framework", by Shi and Malik, copyright IEEE, 1998

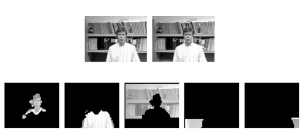
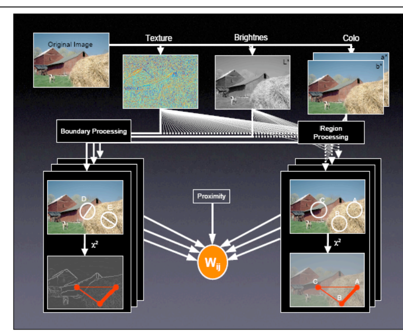
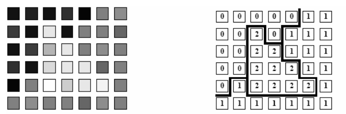


Figure from "Normalized cuts and image segmentation," Shi and Malik, copyright IEEE, 2000

J. Kosecka



Graph Cuts, MRF's



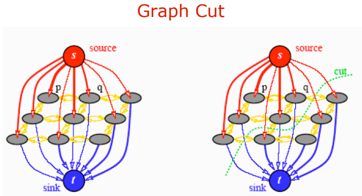
(a) An image (b) A labeling

Pixel labeling problem - find such labels that the objective function is minimized
labels (stereo, object/no-object, original intensities (denoising)

$$E(f)=\sum_{p\in P} D_p(f_p)+\sum_{p,q\in N} V_{p,q}(f_p,f_q)$$

J. Kosecka

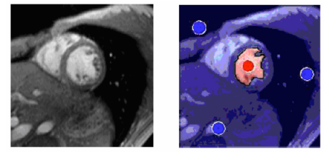
Graph Cut



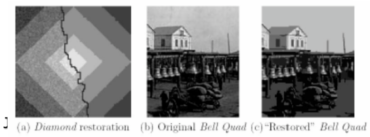
(a) A graph $\mathcal{G}$ (b) A cut on $\mathcal{G}$

Formulated as minimum cost flow -
Network flow problem from Graph Theory
Kolmogorov, Boykov (et al)
efficient solvers available

J. Kosecka



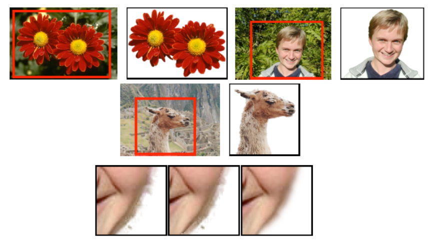
Original image A minimum cut



(a) *Diamond* restoration (b) Original *Bell Quad* (c) "Restored" *Bell Quad*

J

Interactive Foreground Segmentation



Interaction Foreground Segmentation: Grab Cuts
Rother, Kolmogorov, Blake, SIGRAPH 2005

J. Kosecka

Segmentation of Time Varying Images (and tracking)

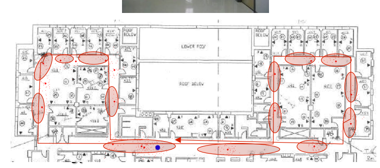
J. Kosecka

Technique: Shot Boundary Detection

- Find the shots in a sequence of video
  - shot boundaries usually result in big differences between succeeding frames
- Strategy:
  - compute interframe distances
  - declare a boundary where these are big
- Possible distances
  - frame differences
  - histogram differences
  - block comparisons
  - edge differences
- Applications:
  - representation for movies, or video sequences
    - find shot boundaries
    - obtain "most representative" frame
  - supports search

J. Kosecka

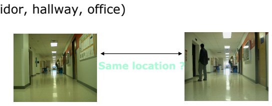
Vision Based Localization
Buidling Model of the environment from video



Kosecka

Global Topology and Local Geometry of the Environment

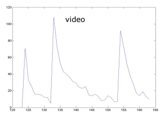
- Impose some discrete structure on the space of continuous visual observations
- Develop methods applicable to large scale environments
- Associate semantic labels with individual locations (corridor, hallway, office)

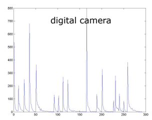


Issues for Vision Based Localization
- Representation of individual locations
- Learning neighborhood relationships between locations

Partitioning the video sequence

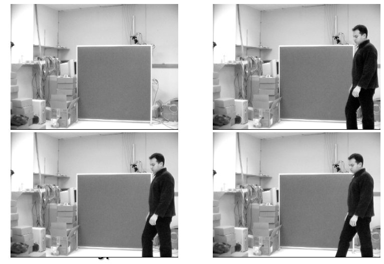
- Transitions between individual locations determined during exploration
- Location sub-sequence across which frames can be matched successfully
- (matching cri
- Location Representation - set of representative views





# of frames vs. # of matched features 1st – i-th view

Image Differencing

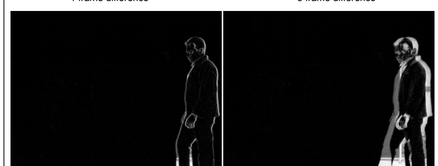


Technique: Background Subtraction

- If we know what the background looks like, it is easy to identify "interesting bits"
- Applications
  - Person in an office
  - Tracking cars on a road
  - surveillance
- Approach:
  - use a moving average to estimate background image
  - subtract from current frame
  - large absolute values are interesting pixels
    - trick: use morphological operations to clean up pixels

Image Differencing: Results

1 frame difference | 5 frame difference



Motion detection

- Background subtraction
  - create an image of the stationary background by averaging a long sequence
    - for any pixel, most measurements will be from the background
    - computing the median measurements, for example, at each pixel, will with high probability assign that pixel the true background intensity - fixed threshold on differencing used to find "foreground" pixels
    - can also compute a distribution of background pixels by fitting a mixture of Gaussians to set of intensities and assuming large population is the background - adaptive thresholding to find foreground pixels

Motion Detection

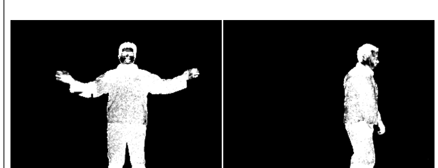
- difference a frame from the known background frame
  - even for interior points of homogeneous objects, likely to detect a difference
  - this will also detect objects that are stationary but different from the background
  - typical algorithm used in surveillance systems
- Motion detection algorithms such as these only work if the camera is stationary and objects are moving against a fixed background

A 300-Frame Sequence with a "Busy" Background

click to start movie

Background Subtraction: Results

Confidence corresponds to gray-level value.
High confidence – bright pixels, low confidence – dark pixels.



Background modeling: color-based

- At each pixel model colors *(r,g,b)* or gray-level values *g*. The following equations are used to recursively estimate the mean and the variance at each pixel:

$$\mu_{t+1} = \alpha\mu_t + (1-\alpha)z_{t+1}$$
$$\sigma^2_{t+1} = \alpha(\sigma^2_t + (\mu_{t+1}-\mu_t)^2) + (1-\alpha)(z_{t+1}-\mu_{t+1})^2$$

where z_{t+1} is the current measurement. The mean μ and the variance σ can both be time varying. The constant α is set empirically to control the rate of adaptation ($0<\alpha<1$).
- A pixel is marked as foreground if given red value *r* (or for any other measurement, say *g* or *b*) we have

$$|r-\mu_t| > 3\max(\sigma_r, \sigma_{rcam})$$

Background model

- σ_{rcam} is the variance of the camera noise, can be estimated from image differences of any two frames.
- If we compute differences for all channels, we can set a pixel as foreground if any of the differences is above the preset threshold.
- Noise can be cleaned using connected component analysis and ignoring small components.
- Similarly we can model the chromaticity values rc, gc and use them for background subtraction:
  $r_c=r/(r+g+b)$, $g_c=g/(r+g+b)$

Background model: edge-based

- Model edges in the image. This can be done two different ways:
  - Compute models for edges in a the average background image
  - Subtract the background (model) image and the new frame; compute edges in the subtraction image; mark all edges that are above a threshold.
    - The threshold can be learned from examples
    - The edges can be combined (color edges) or computed separately for all three color channels

Foreground model

- Use either color histograms (4-bit per color), texture features, edge histograms to model the foreground
- Matching the foreground objects between frames: **tracking**
- Can compare foreground regions directly: shift and subtract. SSD or correlation: *M, N* are two foreground regions.

$$SSD = \sum_{i=1}^{n}\sum_{j=1}^{n}[M(i,j) - N(i,j)]^2$$

$$C = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} M(i,j)N(i,j)}{[\sum_{i=1}^{n}\sum_{j=1}^{n} M(i,j)^2 \sum_{i=1}^{n}\sum_{j=1}^{n} N(i,j)^2]^{1/2}}$$

Histogram Matching

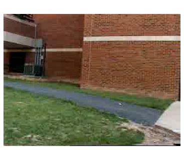
- Histogram Intersection

$$I(h_c, h_b) = \frac{\sum_i \min\{h_c(i), h_b(i)\}}{\sum_i \max\{h_c(i), h_b(i)\}}$$

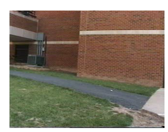
- Chi Squared Formula

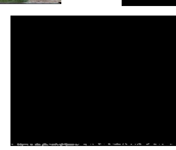
$$\chi^2(h_c, h_b) = \sum_i 2\frac{(h_c(i) - h_b(i))^2}{h_c(i) + h_b(i)}$$

Surveillance: Interacting people




Background Subtraction





Background Subtraction