Fitting:

Voting and the Hough Transform

Monday, Feb 14

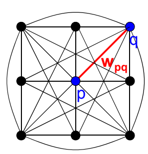
Prof. Kristen Grauman

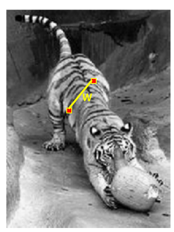
UT-Austin

Last time: Grouping

- Bottom-up segmentation via clustering
  - To find mid-level regions, tokens
  - General choices -- features, affinity functions, and clustering algorithms
  - Example clustering algorithms
    - Mean shift and mode finding: K-means, Mean shift
    - Graph theoretic: Graph cut, normalized cuts
- Grouping also useful for quantization
  - Texton histograms for texture within local region

Recall: Images as graphs

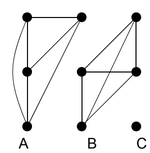




Fully-connected graph
- node for every pixel
- link between *every* pair of pixels, **p**,**q**
- similarity w_{pq} for each link
  - » similarity is *inversely proportional* to difference in color and position

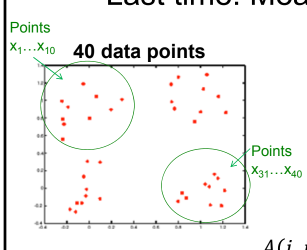
Slide by Steve Seitz

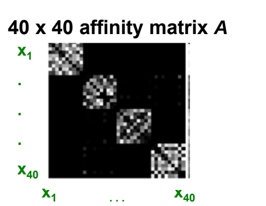
Goal: Segmentation by Graph Cuts



Break graph into segments
- Delete links that cross between segments
- Easiest to break links that have low similarity
  - similar pixels should be in the same segments
  - dissimilar pixels should be in different segments

Last time: Measuring affinity

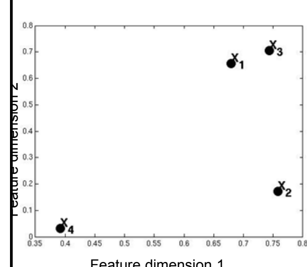




$$A(i,j) = \exp\{-(^1/_{2\sigma^2})\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2\}$$

1. What do the **blocks** signify?
2. What does the **symmetry** of the matrix signify?
3. How would the matrix change with **larger value of σ**?

Example: weighted graphs



- Suppose we have a 4-pixel image (i.e., a 2 x 2 matrix)
- Each pixel described by 2 features

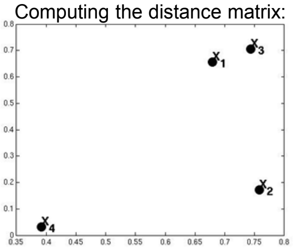
Dimension of data points : d = 2

Number of data points : N = 4

Kristen Grauman

Example: weighted graphs

Computing the distance matrix:



D(:,1)=

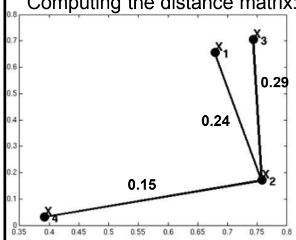
(1,:)=	(0)	0.24	0.01	0.47

```
for i=1:N
        for j=1:N
                D(i,j) = ||xi- xj||^2
        end
end
```

Kristen Grauman

Example: weighted graphs

Computing the distance matrix:



D(:,1)=

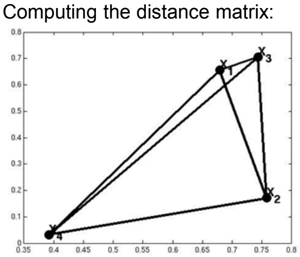
D(1,:)=	(0)	0.24	0.01	0.47

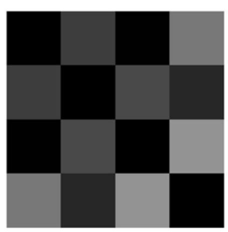
```
for i=1:N
        for j=1:N
                D(i,j) = ||xi- xj||^2
        end
end
```

Kristen Grauman

Example: weighted graphs

Computing the distance matrix:



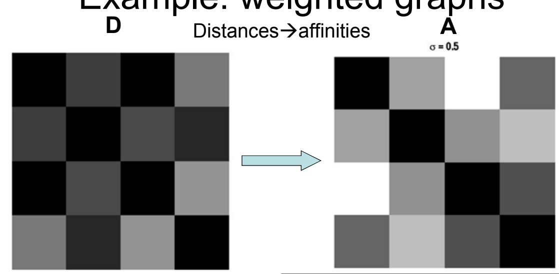


N x N matrix

```
for i=1:N
        for j=1:N
                D(i,j) = ||xi- xj||^2
        end
end
```

Kristen Grauman

Example: weighted graphs

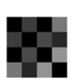



```
for i=1:N
  for j=1:N
    D(i,j) = ||xi- xj||^2
  end
end
```

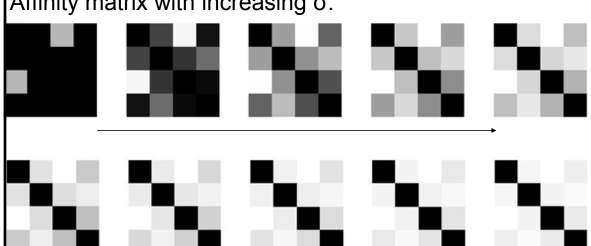
```
for i=1:N
  for j=i+1:N
      A(i,j) = exp(-1/(2*σ^2)*||xi- xj||^2);
      A(j,i) = A(i,j);
  end
end
```

Kristen Grauman

Scale parameter σ affects affinity

Distance matrix D= 

Affinity matrix with increasing σ:



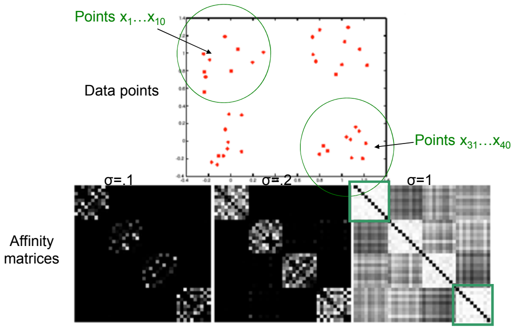
Kristen Grauman

Visualizing a shuffled affinity matrix

If we permute the order of the vertices as they are referred to in the affinity matrix, we see different patterns:

Kristen Grauman

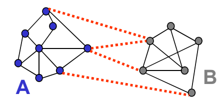
Putting these two aspects together

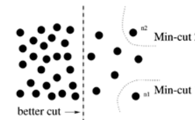


$$A(i,j) = \exp\{-(1/_{2\sigma^2})\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2\}$$

Kristen Grauman

Cuts in a graph: Min cut





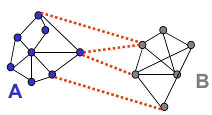
Weakness of Min cut

Link Cut
- set of links whose removal makes a graph disconnected
- cost of a cut: $cut(A,B) = \sum_{p \in A, q \in B} w_{p,q}$

Find **minimum cut**
- gives you a segmentation
- fast algorithms exist

Source: Steve Seitz

Cuts in a graph: Normalized cut



- Fix bias of Min Cut by **normalizing** for size of segments:

$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)}$$

assoc(A,V) = sum of weights of all edges that touch A

- Ncut value is small when we get two clusters with many edges with high weights, and few edges of low weight between them.
- Approximate solution: generalized eigenvalue problem.

J. Shi and J. Malik, Normalized Cuts and Image Segmentation, CVPR, 1997

Steve Seitz

Example results: segments from Ncuts

Normalized cuts: pros and cons

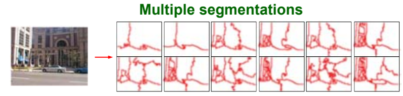
Pros:
- Generic framework, flexible to choice of function that computes weights ("affinities") between nodes
- Does not require model of the data distribution

Cons:
- Time complexity can be high
  - Dense, highly connected graphs → many affinity computations
  - Solving eigenvalue problem
- Preference for balanced partitions

Kristen Grauman

Segments as primitives for recognition

Multiple segmentations



B. Russell et al., "Using Multiple Segmentations to Discover Objects and their Extent in Image Collections," CVPR 2006

Slide credit: Lana Lazebnik

Motion segmentation

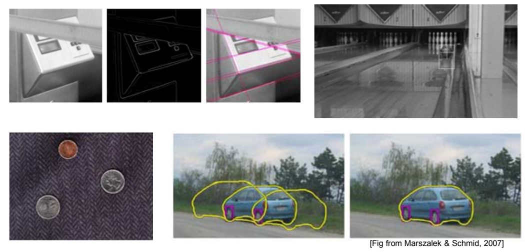
Input sequence | Image Segmentation | Motion Segmentation

Input sequence | Image Segmentation | Motion Segmentation

A. Barbu, S.C. Zhu. Generalizing Swendsen-Wang to sampling arbitrary posterior probabilities, *IEEE Trans. PAMI,* August 2005.

Now: Fitting

- Want to associate a model with observed features



[Fig from Marszalek & Schmid, 2007]

For example, the model could be a line, a circle, or an arbitrary shape.

Kristen Grauman

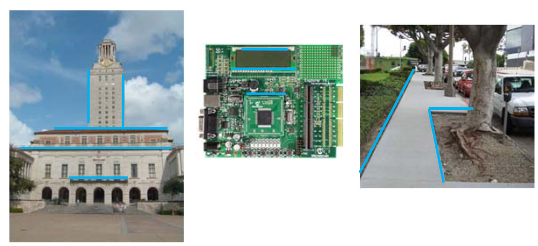
Fitting: Main idea

- Choose a parametric model to represent a set of features
- Membership criterion is not local
  - Can't tell whether a point belongs to a given model just by looking at that point
- Three main questions:
  - What model represents this set of features best?
  - Which of several model instances gets which feature?
  - How many model instances are there?
- Computational complexity is important
  - It is infeasible to examine every possible set of parameters and every possible combination of features

Slide credit: L. Lazebnik

Example: Line fitting

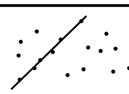
- Why fit lines?
  Many objects characterized by presence of straight lines

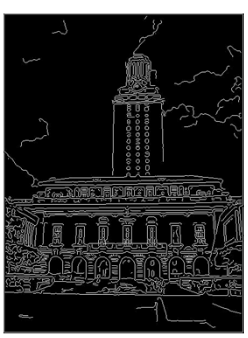


- Wait, why aren't we done just by running edge detection?

Kristen Grauman

Difficulty of line fitting





- **Extra** edge points (clutter), multiple models:
  – which points go with which line, if any?
- Only some parts of each line detected, and some parts are **missing:**
  – how to find a line that bridges missing evidence?
- **Noise** in measured edge points, orientations:
  – how to detect true underlying parameters?

Kristen Grauman

Voting

- It's not feasible to check all combinations of features by fitting a model to each possible subset.
- **Voting** is a general technique where we let the features *vote for all models that are compatible with it*.
  – Cycle through features, cast votes for model parameters.
  – Look for model parameters that receive a lot of votes.
- Noise & clutter features will cast votes too, *but* typically their votes should be inconsistent with the majority of "good" features.

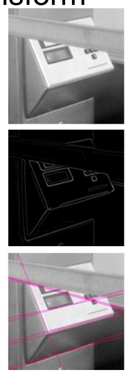
Kristen Grauman

Fitting lines: Hough transform

- Given points that belong to a line, what is the line?
- How many lines are there?
- Which points belong to which lines?

- **Hough Transform** is a voting technique that can be used to answer all of these questions.

  Main idea:
  1. Record vote for each possible line on which each edge point lies.
  2. Look for lines that get many votes.



Kristen Grauman

Finding lines in an image: Hough space

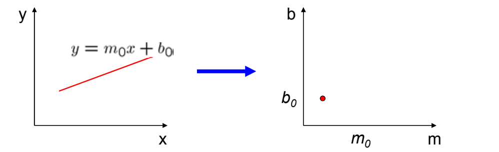


image space   Hough (parameter) space

Connection between image (x,y) and Hough (m,b) spaces
- A line in the image corresponds to a point in Hough space
- To go from image space to Hough space:
  - given a set of points (x,y), find all (m,b) such that y = mx + b

Slide credit: Steve Seitz

Finding lines in an image: Hough space

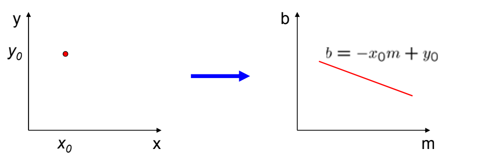


image space   Hough (parameter) space

Connection between image (x,y) and Hough (m,b) spaces
- A line in the image corresponds to a point in Hough space
- To go from image space to Hough space:
  - given a set of points (x,y), find all (m,b) such that y = mx + b
- What does a point (x_0, y_0) in the image space map to?
  - Answer: the solutions of $b = -x_0m + y_0$
  - this is a line in Hough space

Slide credit: Steve Seitz

Finding lines in an image: Hough space

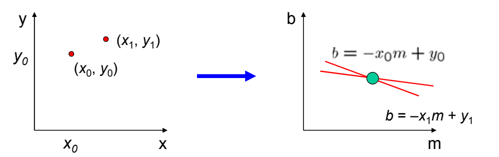


image space   Hough (parameter) space

What are the line parameters for the line that contains both (x_0, y_0) and (x_1, y_1)?
- It is the intersection of the lines $b = -x_0m + y_0$ and $b = -x_1m + y_1$

Finding lines in an image: Hough algorithm

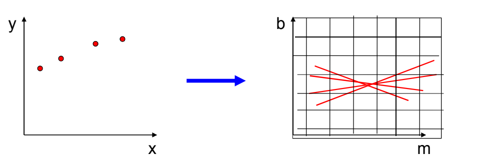
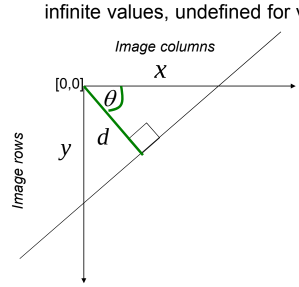


image space   Hough (parameter) space

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.

Polar representation for lines

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



d : perpendicular distance from line to origin

θ : angle the perpendicular makes with the x-axis

$$x \cos\theta - y \sin\theta = d$$

Point in image space → sinusoid segment in Hough space

Kristen Grauman

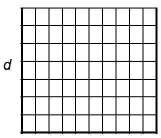
- [Hough line demo]
- http://www.dis.uniroma1.it/~iocchi/slides/icra2001/java/hough.html

Hough transform algorithm

Using the polar parameterization:

$$x\cos\theta - y\sin\theta = d$$

H: accumulator array (votes)



Basic Hough transform algorithm

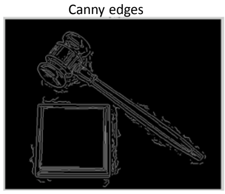
1. Initialize H[d, θ]=0
2. for each edge point I[x,y] in the image
   for θ = [θ_{min} to θ_{max}] // some quantization
   $d = x\cos\theta - y\sin\theta$
   H[d, θ] += 1
3. Find the value(s) of (d, θ) where H[d, θ] is maximum
4. The detected line in the image is given by $d = x\cos\theta - y\sin\theta$

Time complexity (in terms of number of votes per pt)?

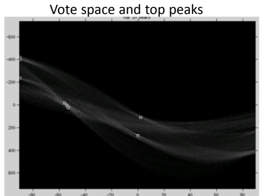
Source: Steve Seitz

Original image

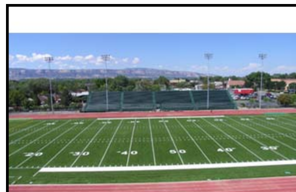
Canny edges

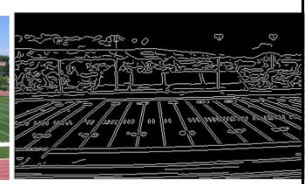


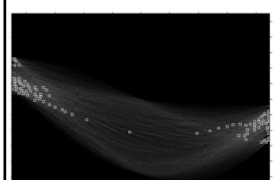
Vote space and top peaks

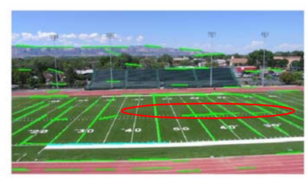


Kristen Grauman





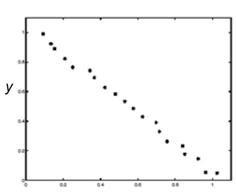




Showing longest segments found

Kristen Grauman

Impact of noise on Hough



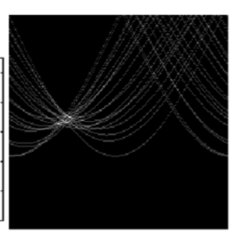
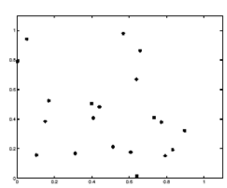


Image space edge coordinates **Votes**

What difficulty does this present for an implementation?

Impact of noise on Hough



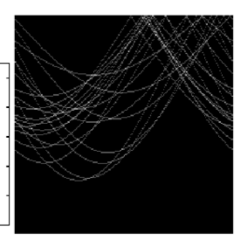


Image space edge coordinates **Votes**

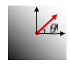
Here, everything appears to be "noise", or random edge points, but we still see peaks in the vote space.

Extensions

Extension 1: Use the image gradient

1. same
2. for each edge point I[x,y] in the image
   θ = gradient at (x,y)
   $d = x\cos\theta - y\sin\theta$
   H[d, θ] += 1
3. same
4. same

(Reduces degrees of freedom)



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

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

Extensions

Extension 1: Use the image gradient

1. same
2. for each edge point I[x,y] in the image
   compute unique (d, θ) based on image gradient at (x,y)
   H[d, θ] += 1
3. same
4. same

(Reduces degrees of freedom)

Extension 2
- give more votes for stronger edges (use magnitude of gradient)

Extension 3
- change the sampling of (d, θ) to give more/less resolution

Extension 4
- The same procedure can be used with circles, squares, or any other shape…

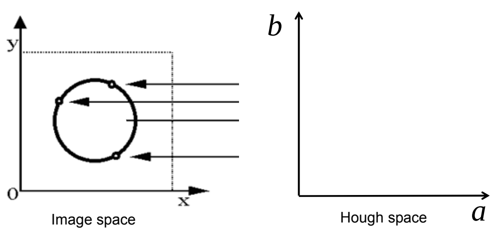
Source: Steve Seitz

Hough transform for circles

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

$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

- For a fixed radius r, unknown gradient direction



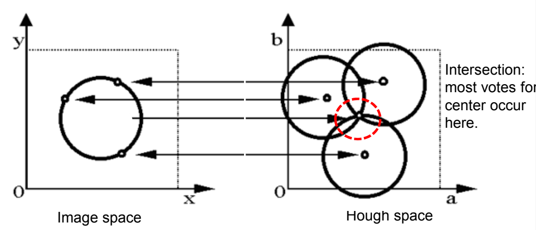
Kristen Grauman

Hough transform for circles

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

$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

- For a fixed radius r, unknown gradient direction



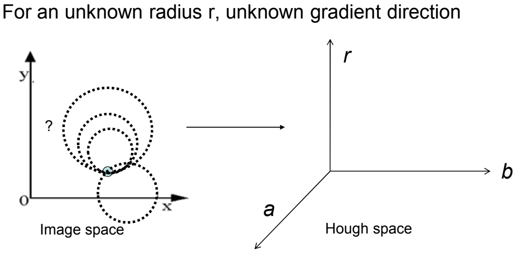
Kristen Grauman

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



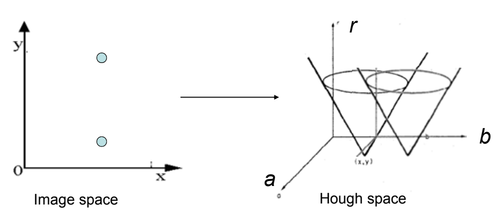
Kristen Grauman

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



Kristen Grauman

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, **known** gradient direction

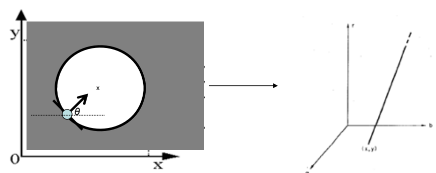


Image space

Hough space

Kristen Grauman

Hough transform for circles

For every edge pixel (*x*,*y*) :
  For each possible radius value *r*:
    For each possible gradient direction *θ*:
    // or use estimated gradient at (x,y)
      a = *x* – *r* cos(*θ*) // column
      b = *y* + *r* sin(*θ*) // row
      H[*a*,*b*,*r*] += 1
  end
end

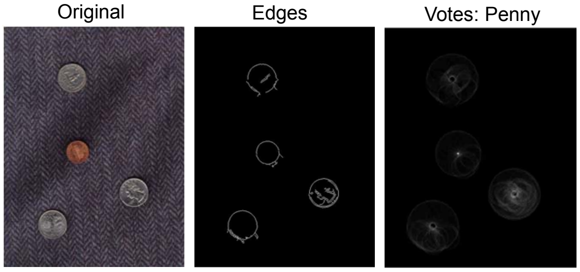
Time complexity per edgel?

- Check out online demo : http://www.markschulze.net/java/hough/

Kristen Grauman

Example: detecting circles with Hough

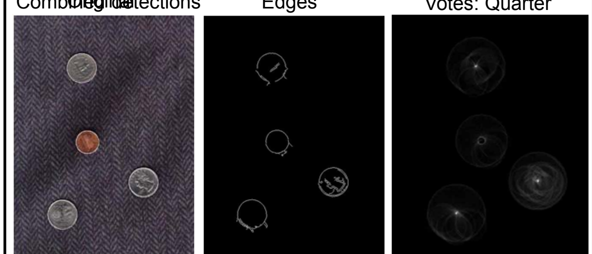
Original Edges Votes: Penny



Note: a different Hough transform (with separate accumulators) was used for each circle radius (quarters vs. penny).

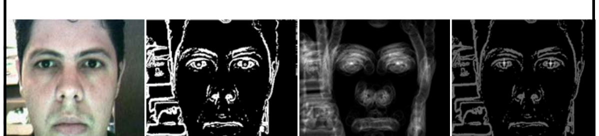
Example: detecting circles with Hough

Original / Combined detections Edges Votes: Quarter



Coin finding sample images from: Vivek Kwatra

Example: iris detection

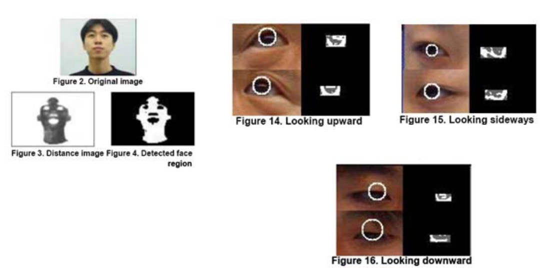


Gradient+threshold Hough space (fixed radius) Max detections

- Hemerson Pistori and Eduardo Rocha Costa http://rsbweb.nih.gov/ij/plugins/hough-circles.html

Kristen Grauman

Example: iris detection



- An Iris Detection Method Using the Hough Transform and Its Evaluation for Facial and Eye Movement, by Hideki Kashima, Hitoshi Hongo, Kunihito Kato, Kazuhiko Yamamoto, ACCV 2002.

Kristen Grauman

Voting: practical tips

- Minimize irrelevant tokens first
- Choose a good grid / discretization

Too fine ? Too coarse

- Vote for neighbors, also (smoothing in accumulator array)
- Use direction of edge to reduce parameters by 1
- To read back which points voted for "winning" peaks, keep tags on the votes.

Kristen Grauman

Hough transform: pros and cons

Pros
- All points are processed independently, so can cope with occlusion, gaps
- Some robustness to noise: noise points unlikely to contribute *consistently* to any single bin
- Can detect multiple instances of a model in a single pass

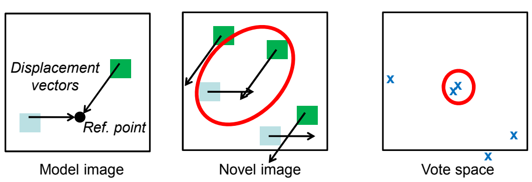
Cons
- Complexity of search time increases exponentially with the number of model parameters
- Non-target shapes can produce spurious peaks in parameter space
- Quantization: can be tricky to pick a good grid size

Kristen Grauman

Generalized Hough Transform

- What if we want to detect arbitrary shapes?

Intuition:

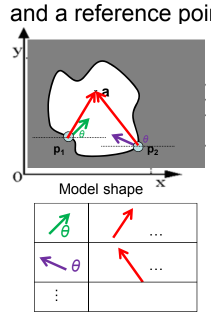


Now suppose those colors encode gradient directions…

Kristen Grauman

Generalized Hough Transform

- Define a model shape by its boundary points and a reference point.



Offline procedure:

At each boundary point, compute displacement vector: $\mathbf{r} = \mathbf{a} - \mathbf{p}_i$.

Store these vectors in a table indexed by gradient orientation θ.

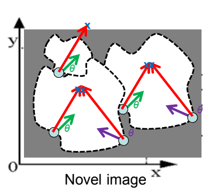
[Dana H. Ballard, Generalizing the Hough Transform to Detect Arbitrary Shapes, 1980]

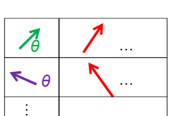
Kristen Grauman

Generalized Hough Transform

Detection procedure:

For each edge point:
- Use its gradient orientation θ to index into stored table
- Use retrieved **r** vectors to vote for reference point



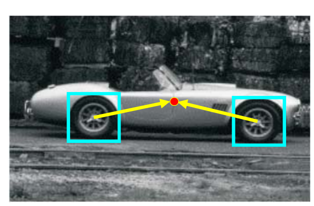


Assuming translation is the only transformation here, i.e., orientation and scale are fixed.

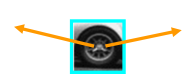
Kristen Grauman

Generalized Hough for object detection

- Instead of indexing displacements by gradient orientation, index by matched local patterns.



training image



"visual codeword" with displacement vectors

B. Leibe, A. Leonardis, and B. Schiele, Combined Object Categorization and Segmentation with an Implicit Shape Model, ECCV Workshop on Statistical Learning in Computer Vision 2004

Source: L. Lazebnik

Generalized Hough for object detection

- Instead of indexing displacements by gradient orientation, index by “visual codeword”

test image

B. Leibe, A. Leonardis, and B. Schiele, Combined Object Categorization and Segmentation with an Implicit Shape Model, ECCV Workshop on Statistical Learning in Computer Vision 2004

Source: L. Lazebnik

Summary

- **Grouping/segmentation** useful to make a compact representation and merge similar features
  - associate features based on defined similarity measure and clustering objective
- **Fitting** problems require finding any supporting evidence for a model, even within clutter and missing features.
  - associate features with an explicit model
- **Voting** approaches, such as the **Hough transform**, make it possible to find likely model parameters without searching all combinations of features.
  - Hough transform approach for lines, circles, …, arbitrary shapes defined by a set of boundary points, recognition from patches.

Kristen Grauman