Object Recognition

CS 554 – Computer Vision

Pinar Duygulu

Bilkent University

(Source:Kristen Grauman)

Supervised classification

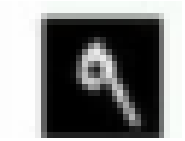
- Given a collection of *labeled* examples, come up with a function that will predict the labels of new examples.

Training examples

"four"

"nine"

Novel input



?

- How good is some function we come up with to do the classification?
- Depends on
  - Mistakes made
  - Cost associated with the mistakes

Supervised classification

- Given a collection of *labeled* examples, come up with a function that will predict the labels of new examples.

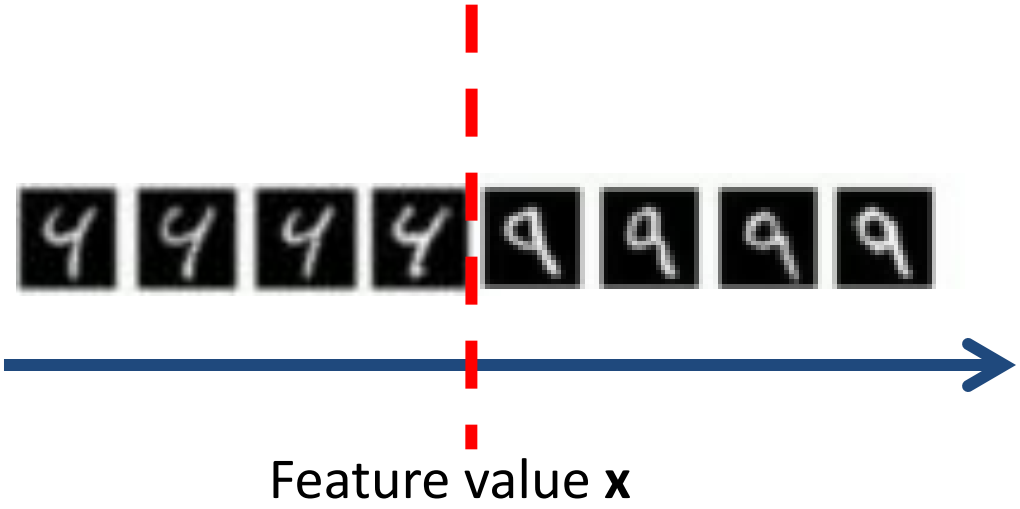
- Consider the two-class (binary) decision problem
  - L(4→9): Loss of classifying a 4 as a 9
  - L(9→4): Loss of classifying a 9 as a 4

- **Risk** of a classifier *s* is expected loss:

$$R(s) = \Pr(4 \to 9 \mid \text{using } s)L(4 \to 9) + \Pr(9 \to 4 \mid \text{using } s)L(9 \to 4)$$

- We want to choose a classifier so as to minimize this total risk

Supervised classification



Optimal classifier will minimize total risk.

At decision boundary, either choice of label yields same expected loss.

So, best decision boundary is at point **x** where

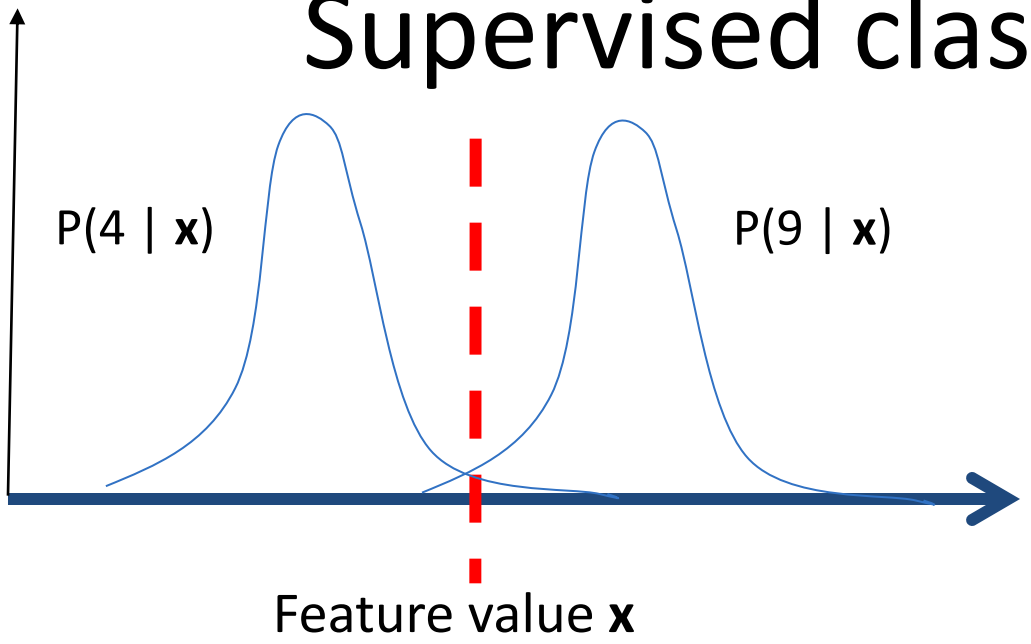
$$P(\text{class is } 9 \mid \mathbf{x})\, L(9 \to 4) = P(\text{class is } 4 \mid \mathbf{x}) L(4 \to 9)$$

To classify a new point, choose class with lowest expected loss; i.e., choose "four" if

$$P(4 \mid \mathbf{x}) L(4 \to 9) > P(9 \mid \mathbf{x}) L(9 \to 4)$$

Kristen Grauman

Supervised classification



Optimal classifier will minimize total risk.

At decision boundary, either choice of label yields same expected loss.

So, best decision boundary is at point **x** where

$$P(\text{class is } 9 \mid \mathbf{x})\, L(9 \to 4) = P(\text{class is } 4 \mid \mathbf{x}) L(4 \to 9)$$

To classify a new point, choose class with lowest expected loss; i.e., choose "four" if

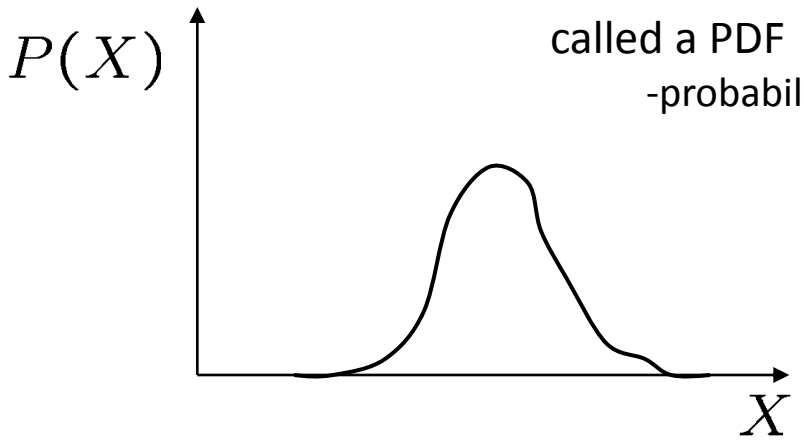
$$P(4 \mid \mathbf{x}) L(4 \to 9) > P(9 \mid \mathbf{x}) L(9 \to 4)$$

How to evaluate these probabilities?

Kristen Grauman

Probability

- Basic probability
  - X is a random variable
  - P(X) is the probability that X achieves a certain value



called a PDF

-probability distribution/density function

  - $0 \leq P(X) \leq 1$
  - $\int_{-\infty}^{\infty} P(X)dX = 1$ or $\sum P(X) = 1$

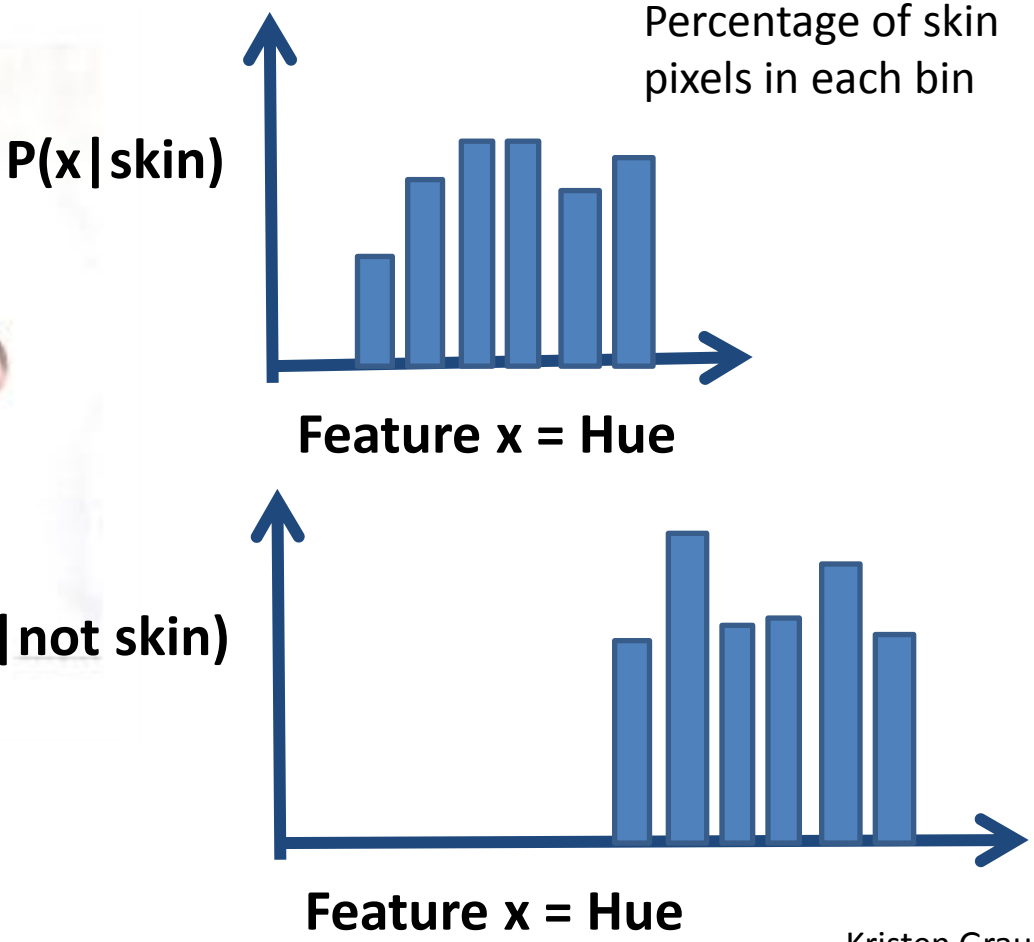
    continuous X discrete X

  - Conditional probability: P(X | Y)
    - probability of X given that we already know Y

Source: Steve Seitz

Example: learning skin colors

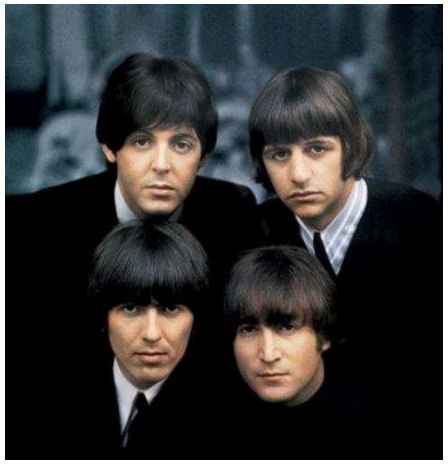
- We can represent a class-conditional density using a histogram (a "non-parametric" distribution)



Kristen Grauman

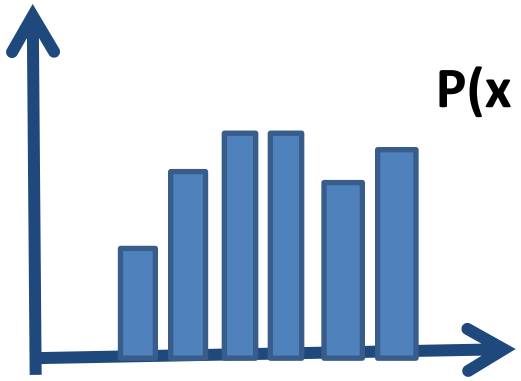
Example: learning skin colors

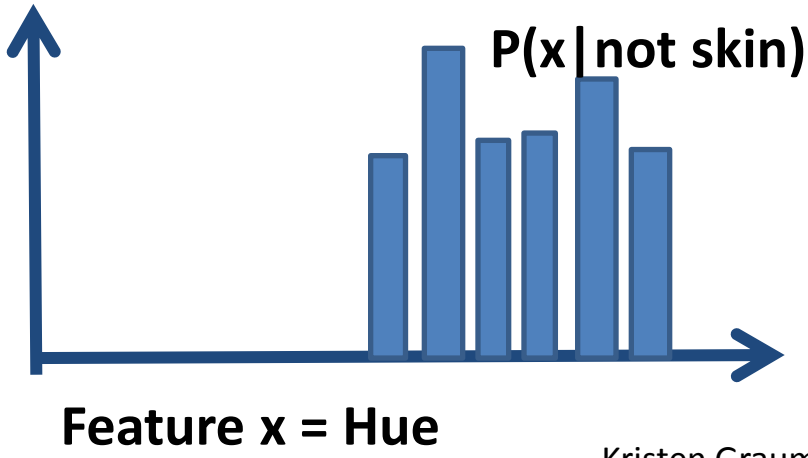
- We can represent a class-conditional density using a histogram (a "non-parametric" distribution)



Now we get a new image, and want to label each pixel as skin or non-skin.

What's the probability we care about to do skin detection?





Bayes rule

posterior

likelihood

prior

$$P(skin|x) = \frac{P(x|skin)P(skin)}{P(x)}$$

$$P(skin|x) \propto P(x|skin)P(skin)$$

Example: classifying skin pixels

Now for every pixel in a new image, we can estimate probability that it is generated by skin.

Brighter pixels → higher probability of being skin

Classify pixels based on these probabilities

- if $p(\text{skin}|\boldsymbol{x}) > \theta$, classify as skin
- if $p(\text{skin}|\boldsymbol{x}) < \theta$, classify as not skin

Example: classifying skin pixels

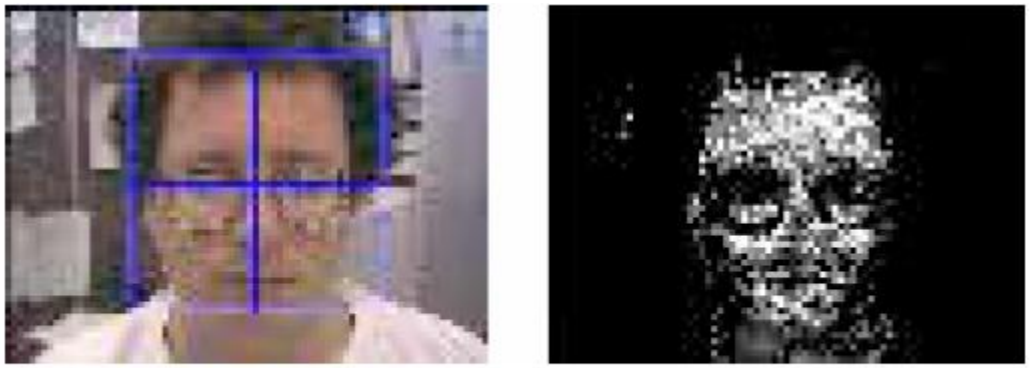


Figure 6: A video image and its flesh probability image

Figure 7: Orientation of the flesh probability distribution marked on the source video image

Gary Bradski, 1998

Example: classifying skin pixels

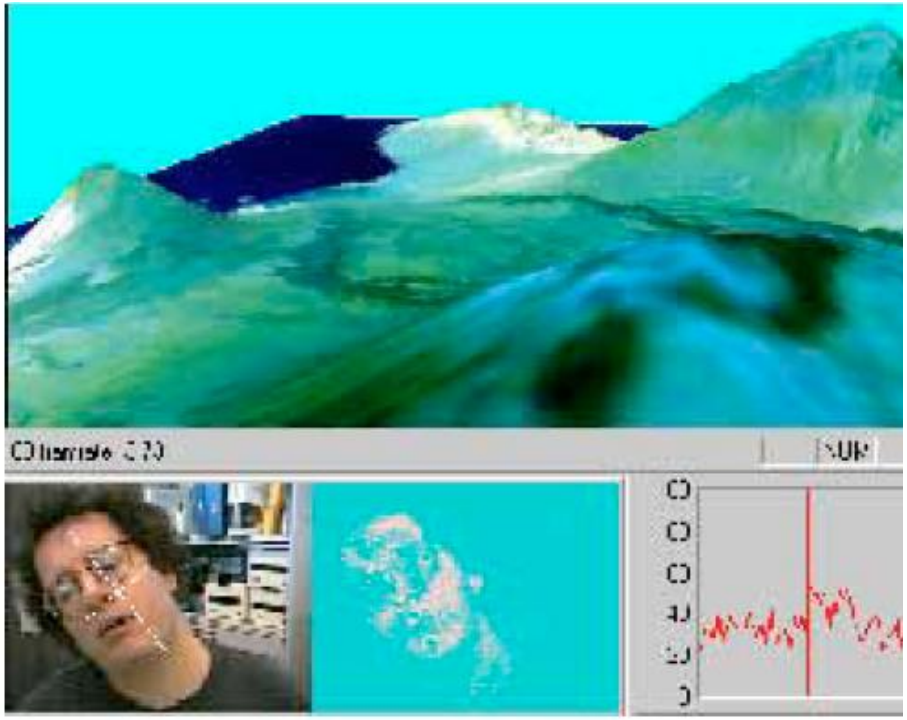


Figure 13: CAMSHIFT-based face tracker used to over a 3D graphic's model of Hawaii

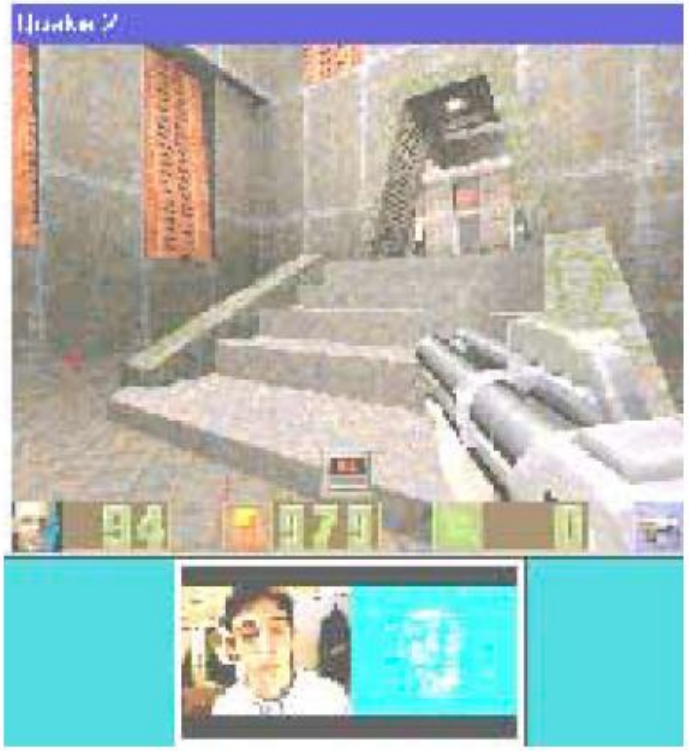


Figure 12: CAMSHIFT-based face tracker used to play Quake 2 hands free by inserting control variables into the mouse queue

Using skin color-based face detection and pose estimation as a video-based interface

Gary Bradski, 1998

Supervised classification

- Want to minimize the expected misclassification
- Two general strategies
  - Use the training data to build representative probability model; separately model class-conditional densities and priors (*generative*)
  - Directly construct a good decision boundary, model the posterior (*discriminative*)

Discriminative classifiers for image recognition

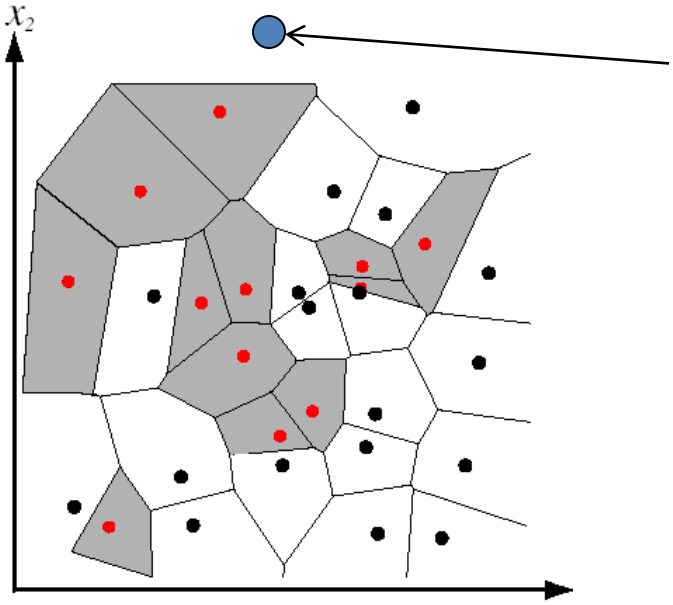
- nearest neighbors (+ scene match app)
- support vector machines (+ gender, person app)

Nearest Neighbor classification

- Assign label of nearest training data point to each test data point

Black = negative

Red = positive



from Duda *et al.*

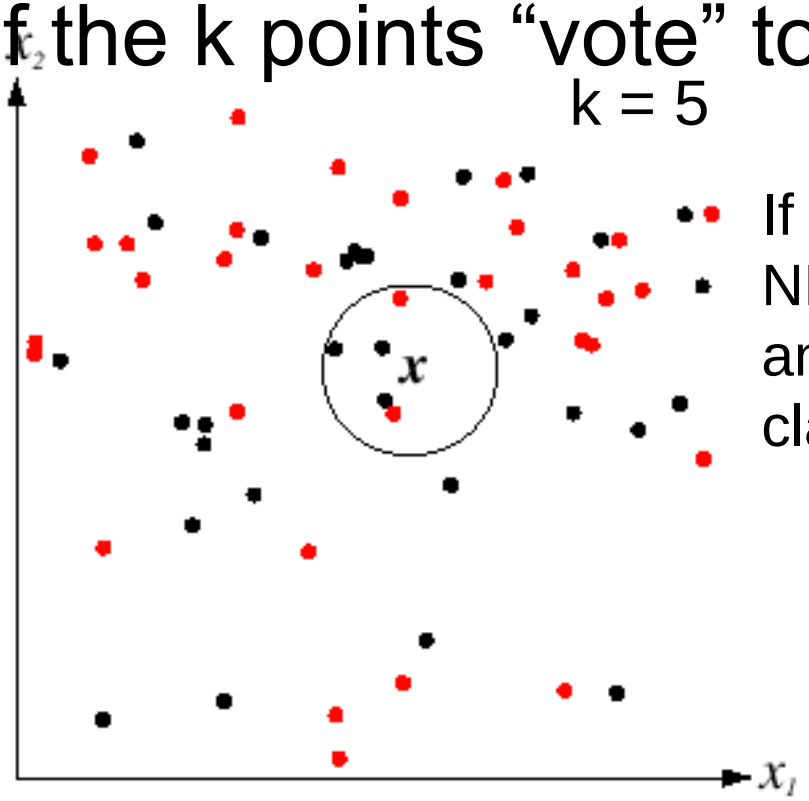
Voronoi partitioning of feature space for 2-category 2D data

K-Nearest Neighbors classification

- For a new point, find the k closest points from training data
- Labels of the k points "vote" to classify

Black = negative

Red = positive



If query lands here, the 5 NN consist of 3 negatives and 2 positives, so we classify it as negative.

Source: D. Lowe

A nearest neighbor recognition example

Where in the World?

[Hays and Efros. **im2gps**: Estimating Geographic Information from a Single Image. CVPR 2008.]

Slides: James Hays

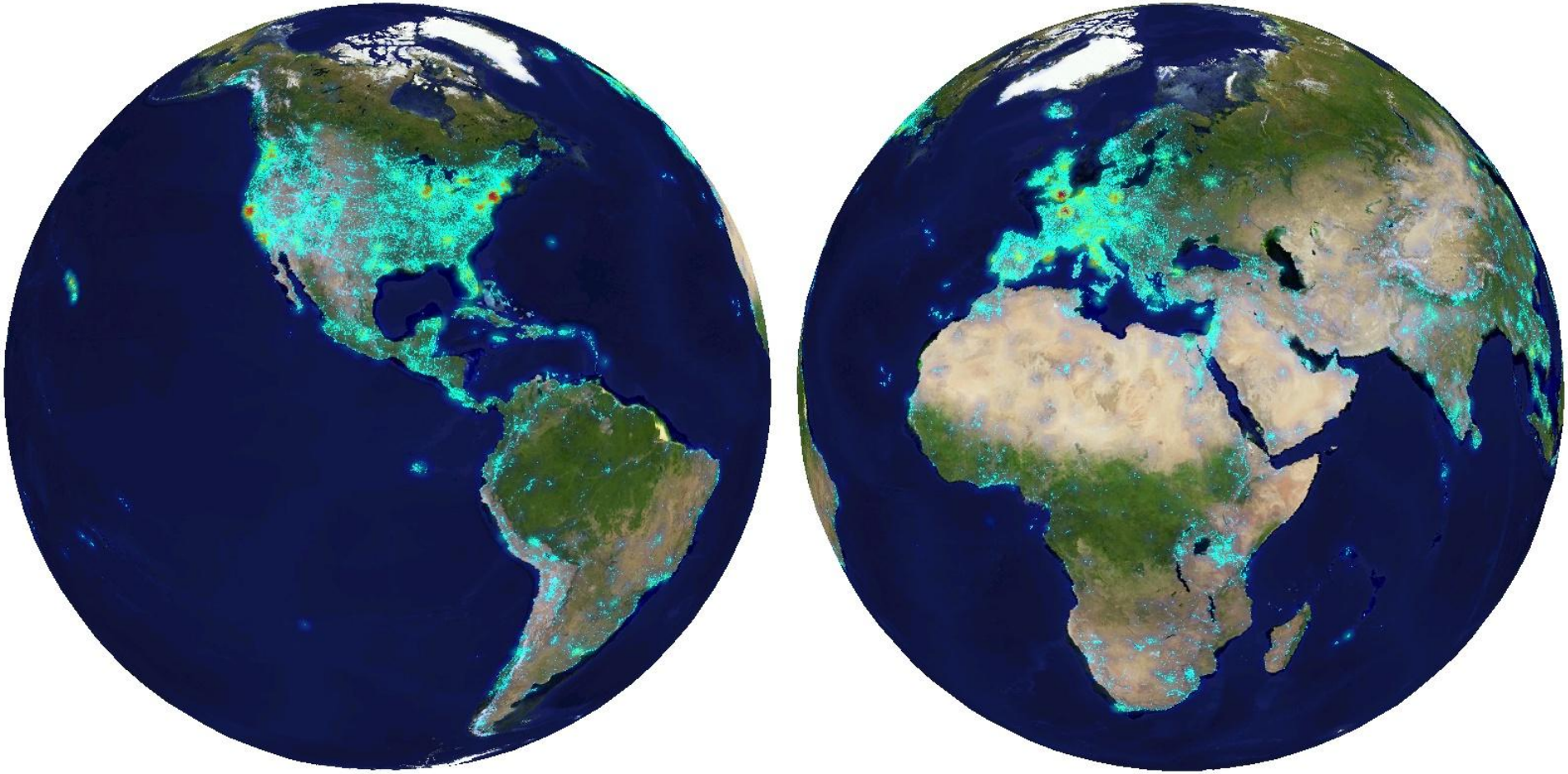
Where in the World?

Slides: James Hays

Where in the World?

Slides: James Hays

6+ million geotagged photos by 109,788 photographers



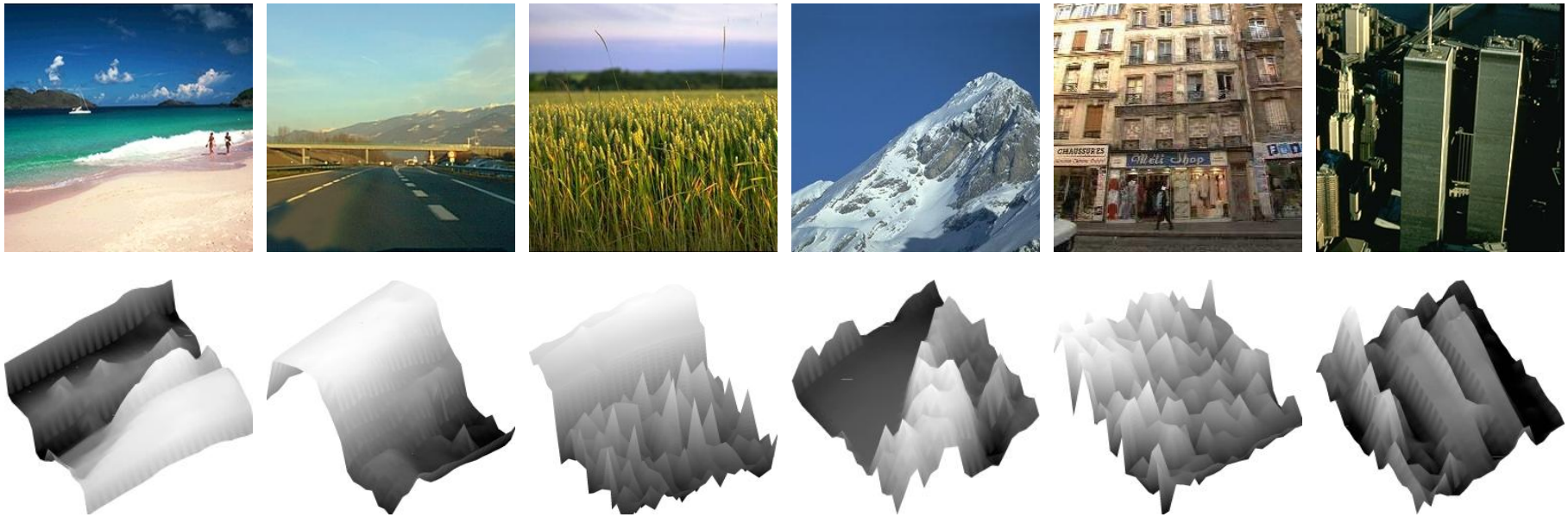
Annotated by Flickr users

Slides: James Hays

Which scene properties are relevant?

Spatial Envelope Theory of Scene Representation

Oliva & Torralba (2001)

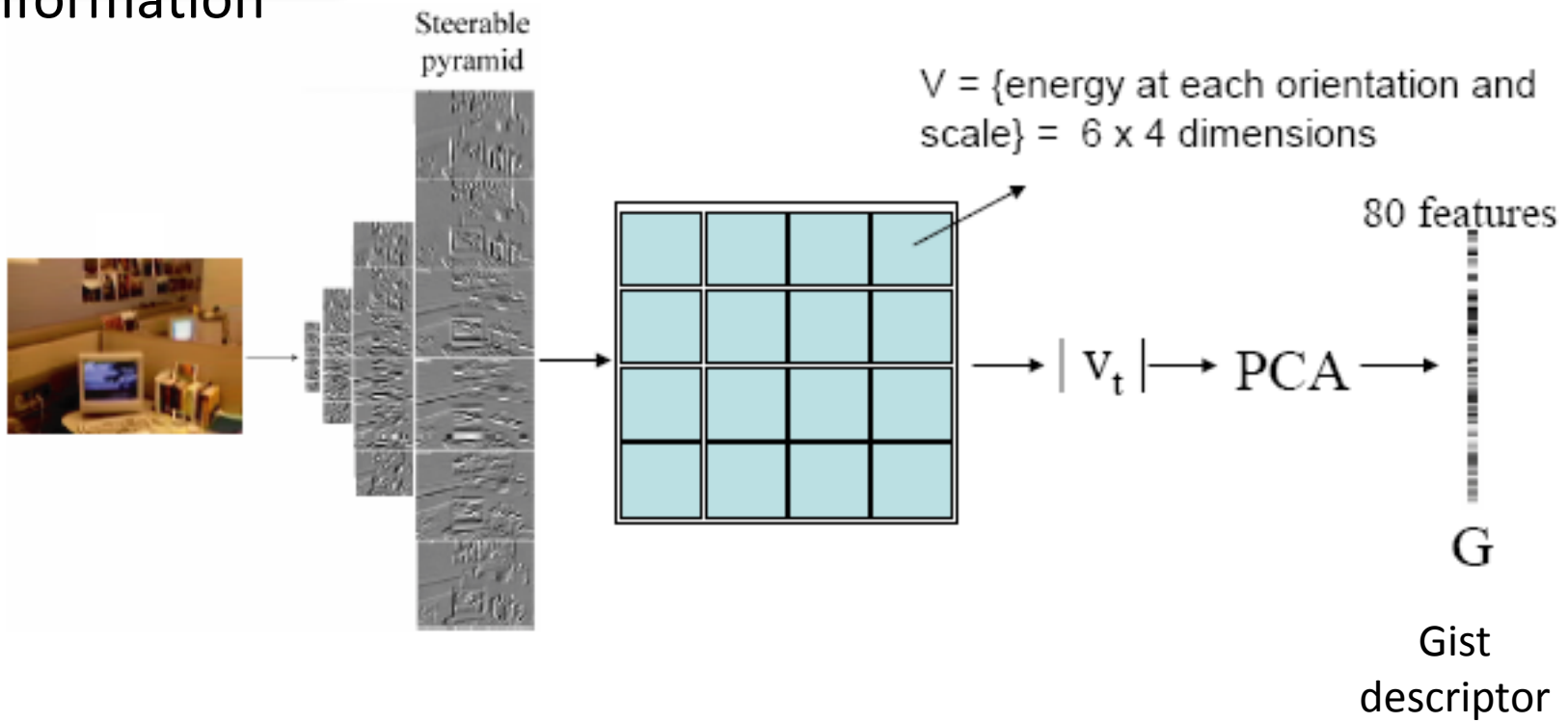


A scene is a single surface that can be represented by global (statistical) descriptors

Slide Credit: Aude Olivia

Global texture: capturing the “Gist” of the scene

Capture global image properties while keeping some spatial information



Oliva & Torralba IJCV 2001, Torralba et al. CVPR 2003

Which scene properties are relevant?

- **Gist scene descriptor**
- **Color Histograms** - L*A*B* 4x14x14 histograms
- **Texton Histograms** – 512 entry, filter bank based
- **Line Features** – Histograms of straight line stats

Scene Matches


Madrid


england


France


Paris


Croatia


heidelberg


Macau


Malta


Cairo


Italy


Italy


Italy


Latvia

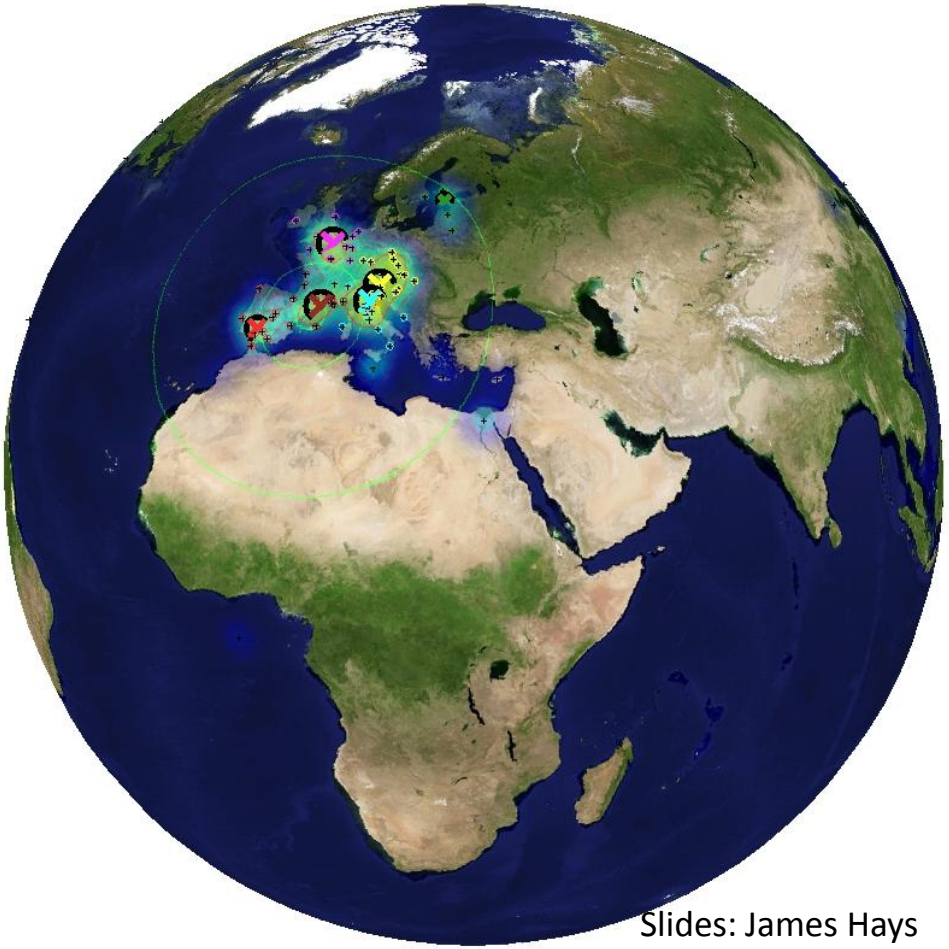

europe


Barcelona


Austria

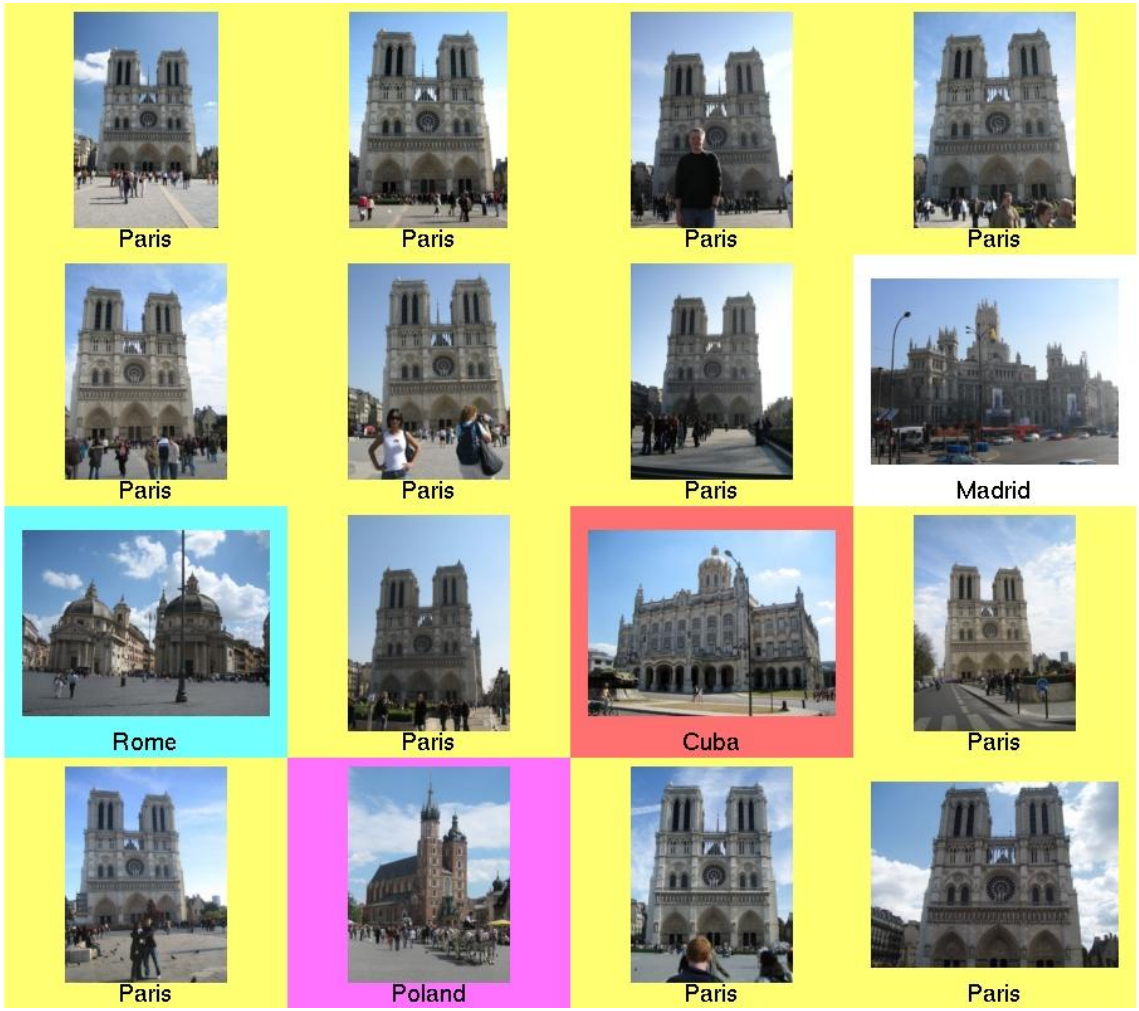
[Hays and Efros. **im2gps**: Estimating Geographic Information from a Single Image. CVPR 2008.]

Slides: James Hays



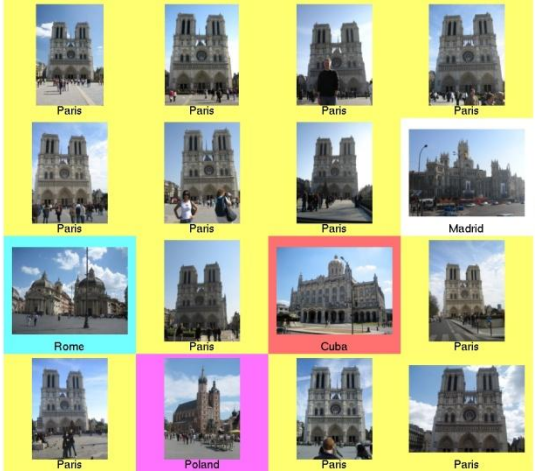
Slides: James Hays

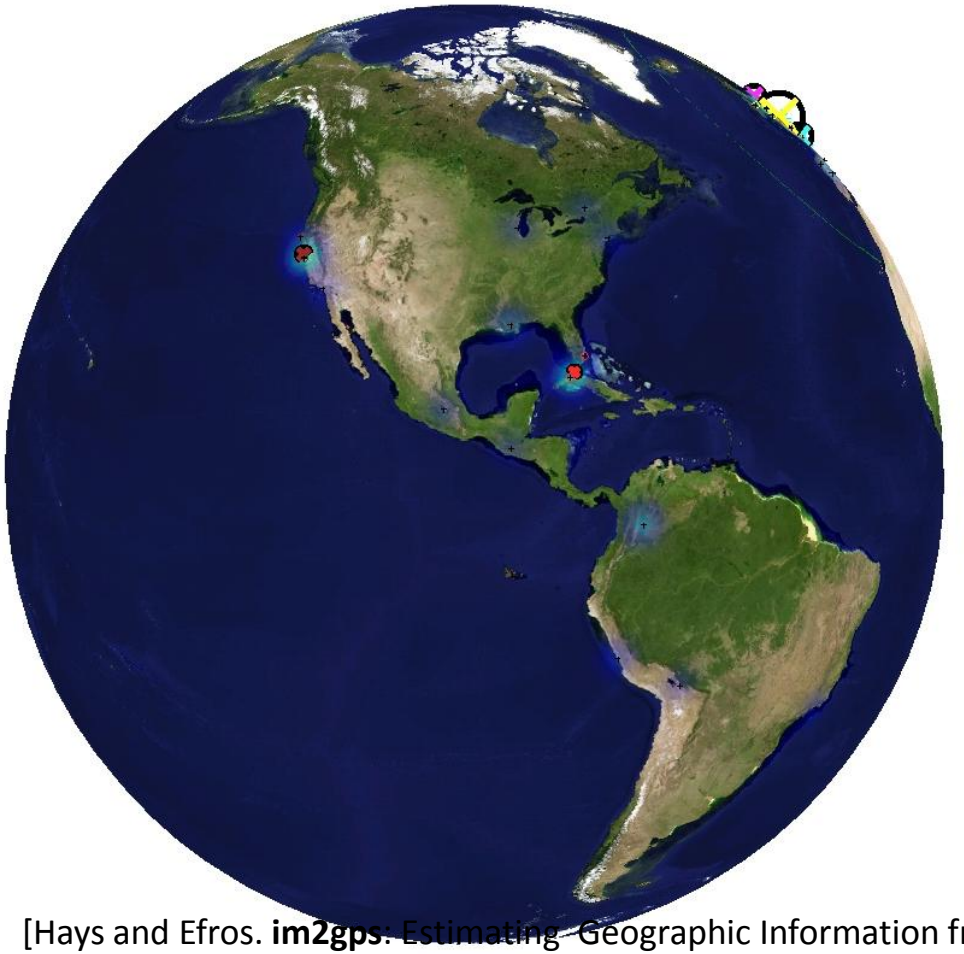
Scene Matches

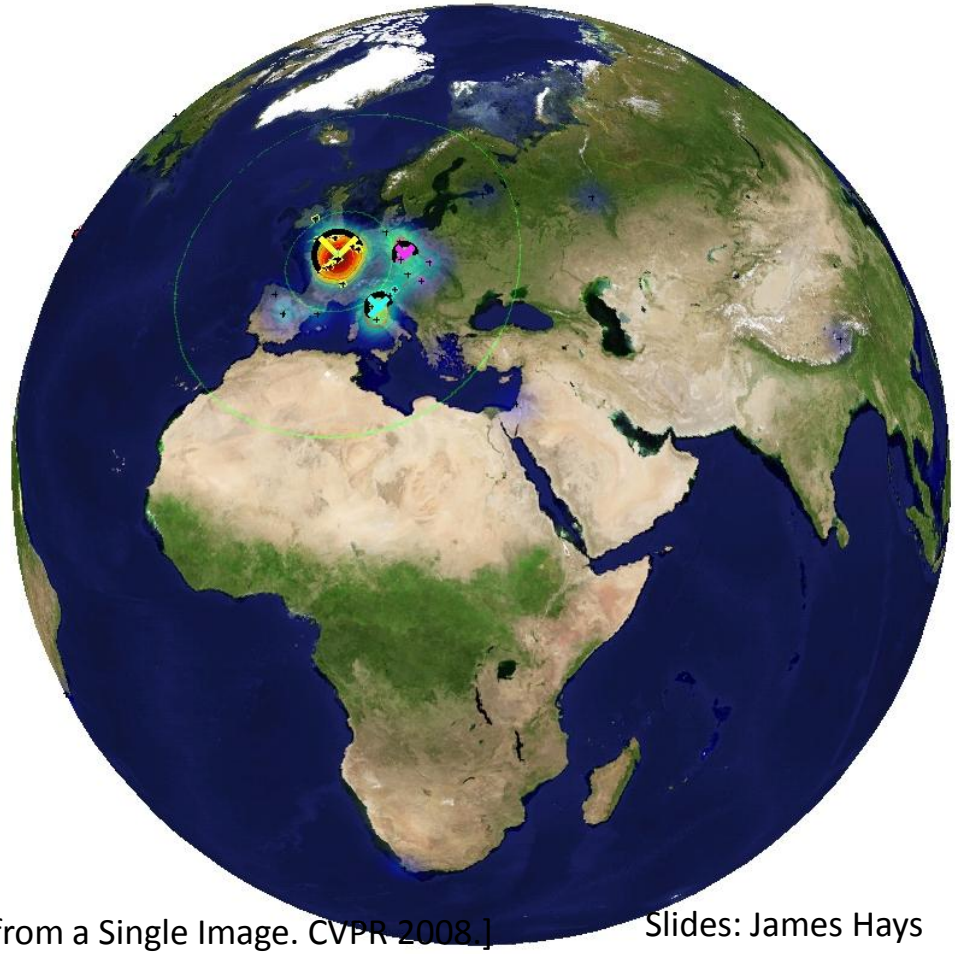


[Hays and Efros. **im2gps**: Estimating Geographic Information from a Single Image. CVPR 2008.]

Slides: James Hays







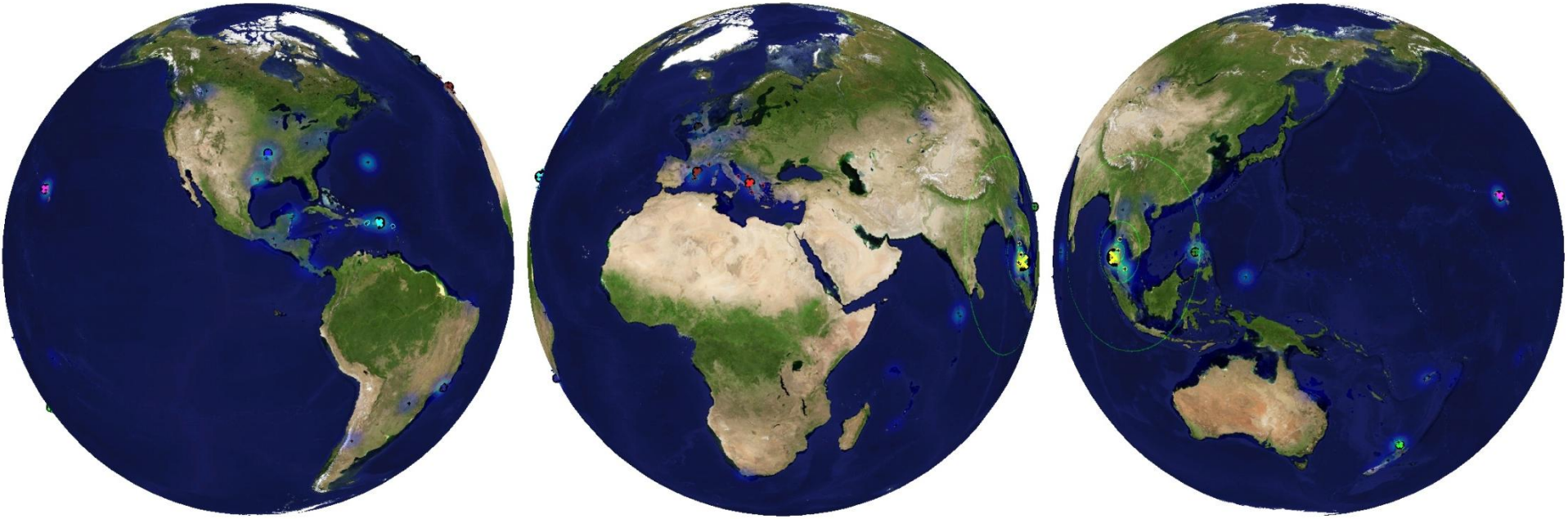
[Hays and Efros. **im2gps**: Estimating Geographic Information from a Single Image. CVPR 2008.]

Slides: James Hays

Scene Matches

[Hays and Efros. **im2gps**: Estimating Geographic Information from a Single Image. CVPR 2008.]

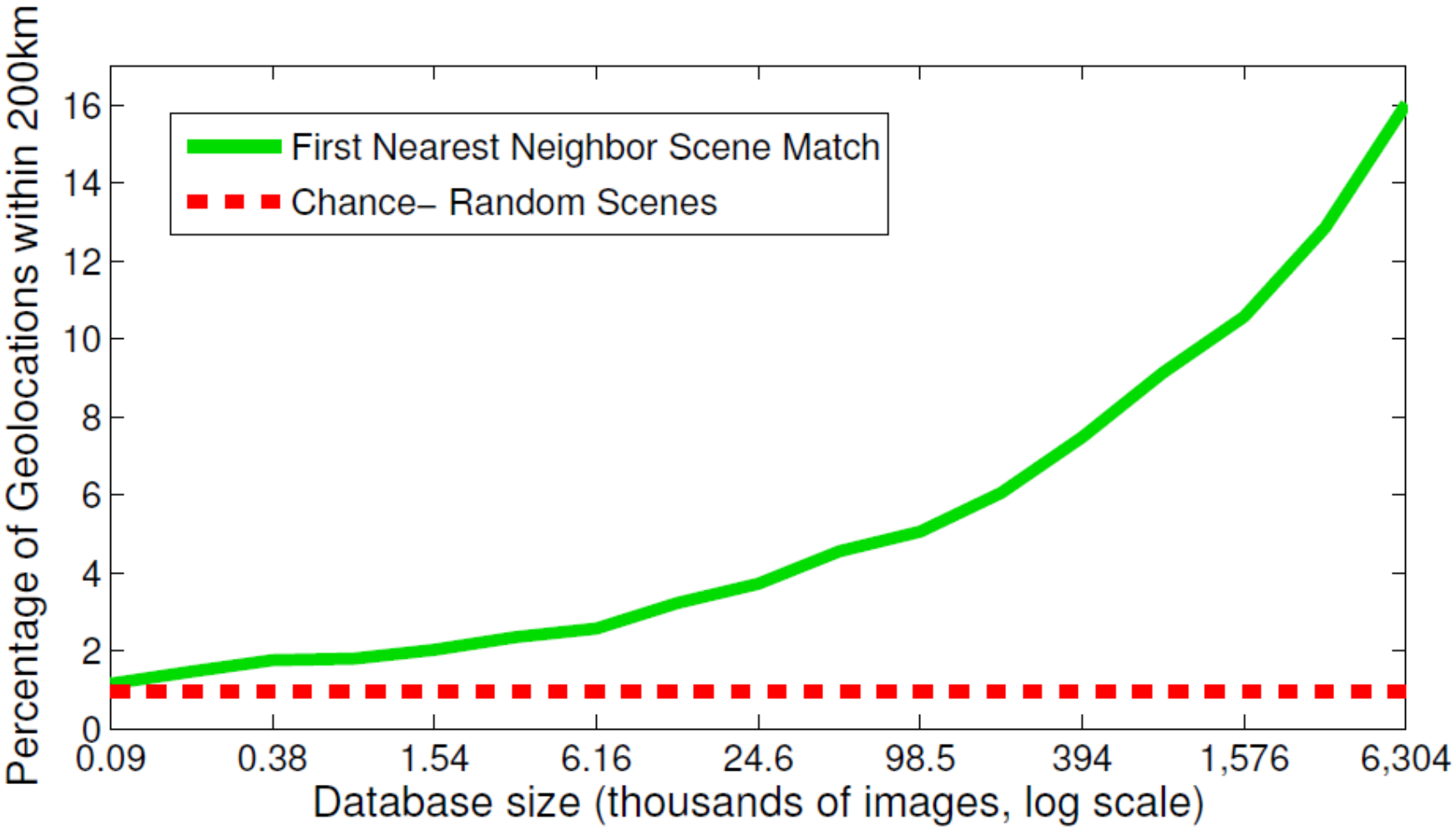
Slides: James Hays



[Hays and Efros. **im2gps**: Estimating Geographic Information from a Single Image. CVPR 2008.]

Slides: James Hays

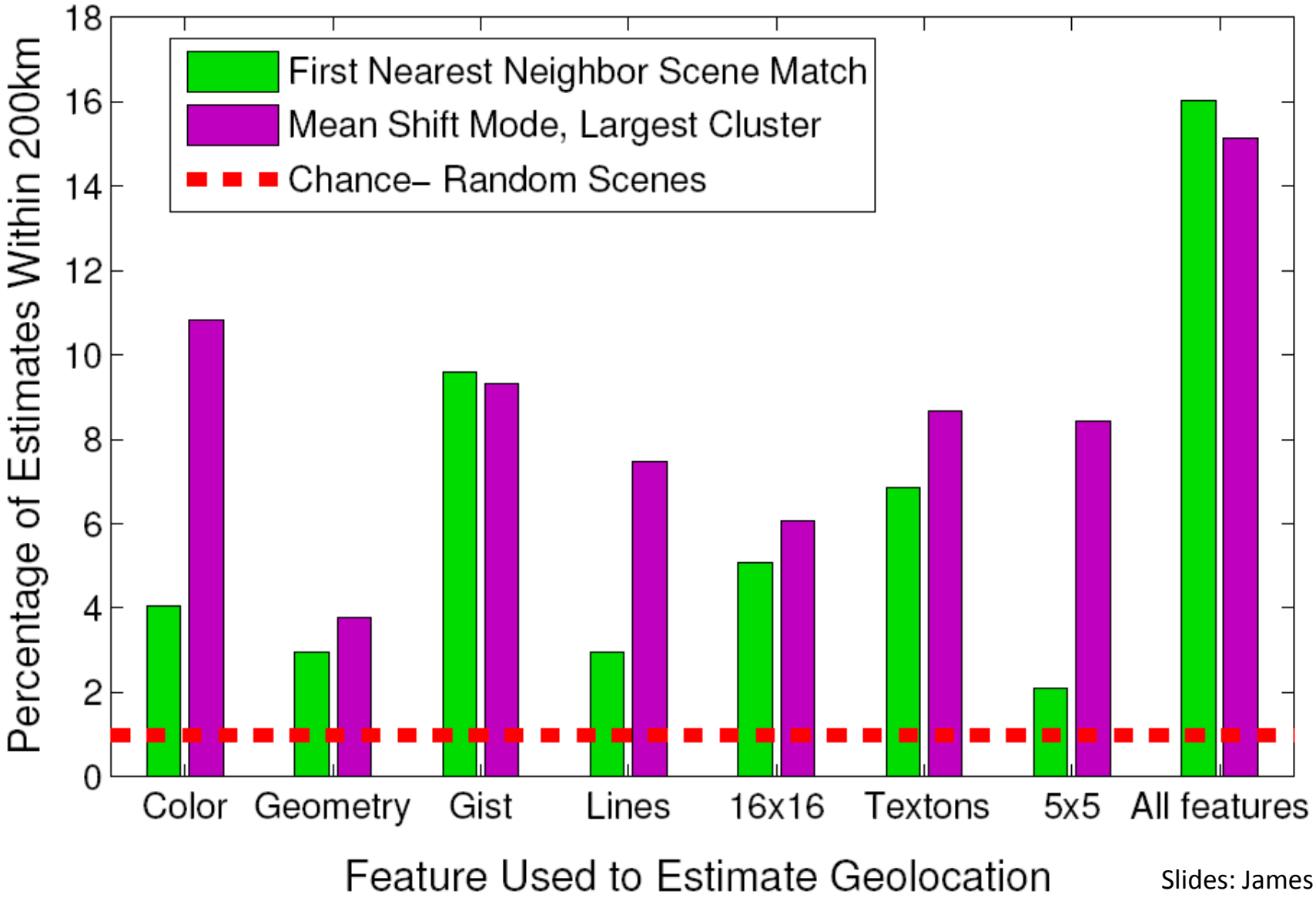
The Importance of Data



[Hays and Efros. **im2gps**: Estimating Geographic Information from a Single Image. CVPR 2008.]

Slides: James Hays

Feature Performance

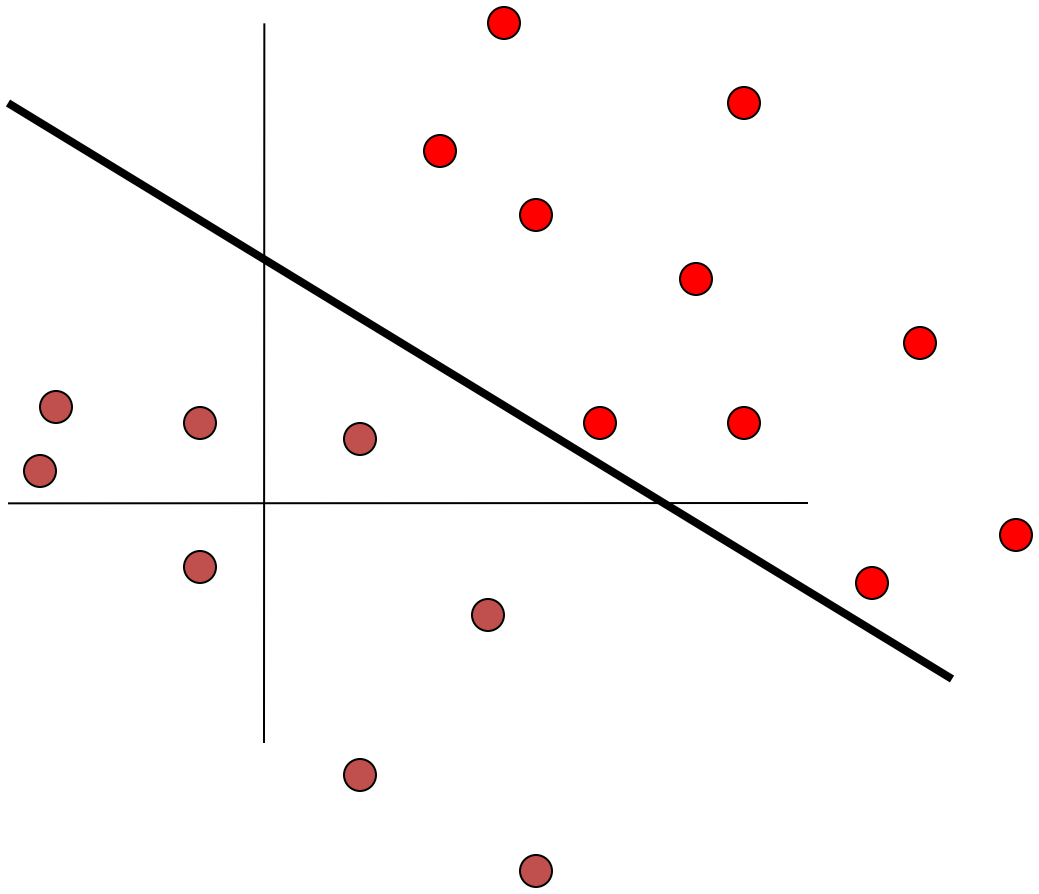


Slides: James Hays

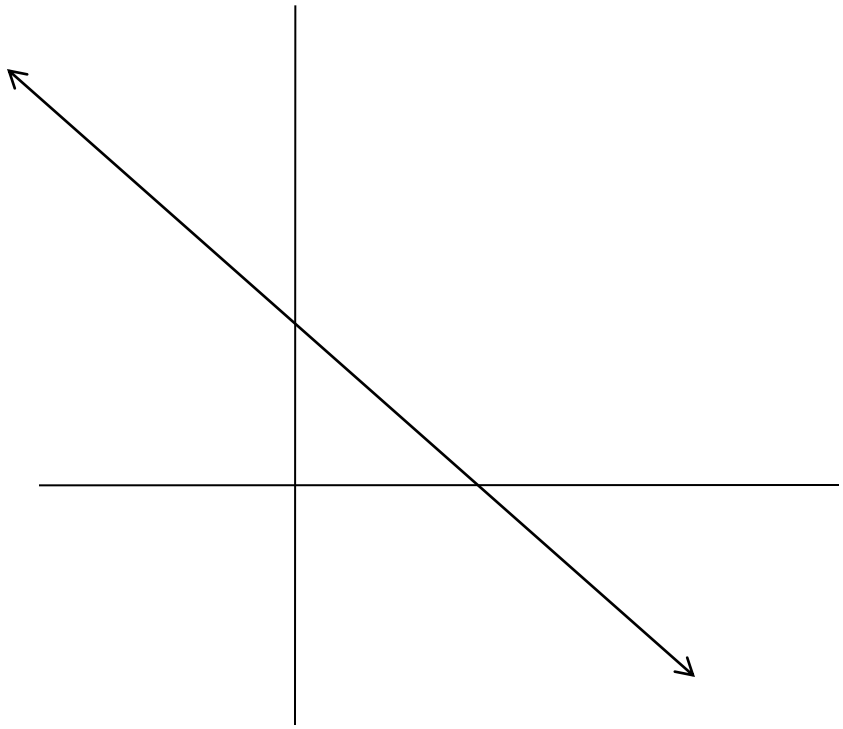
Nearest neighbors: pros and cons

- **Pros**:
  - Simple to implement
  - Flexible to feature / distance choices
  - Naturally handles multi-class cases
  - Can do well in practice with enough representative data
- **Cons:**
  - Large search problem to find nearest neighbors
  - Storage of data
  - Must know we have a meaningful distance function

Linear classifiers



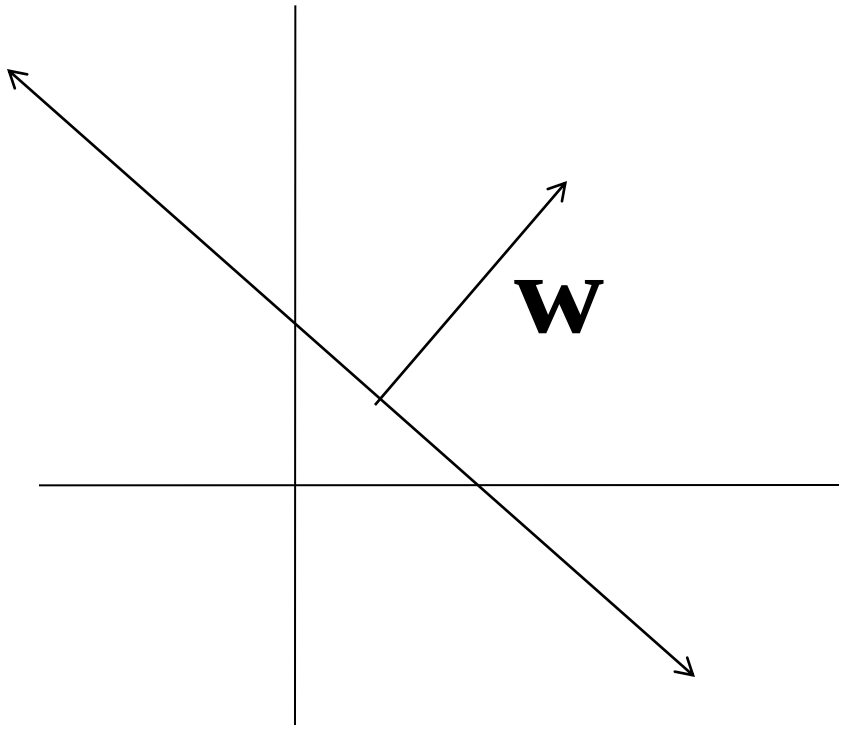
Lines in R^2



Let $\mathbf{w} = \begin{bmatrix} a \\ c \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + cy + b = 0$$

Lines in R^2



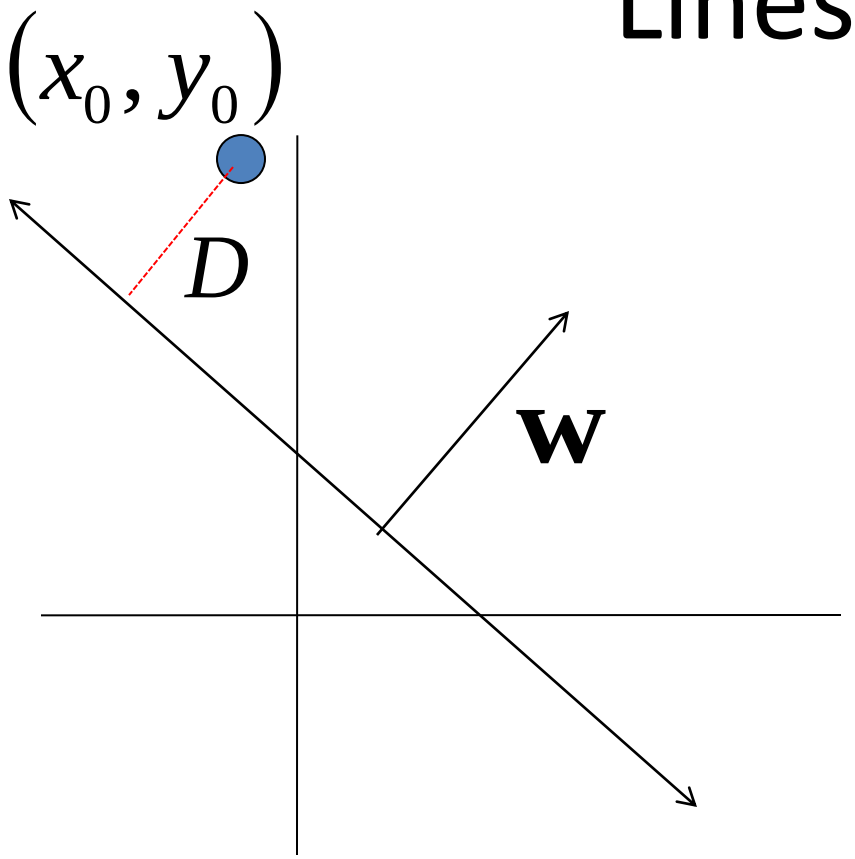
Let $\mathbf{w} = \begin{bmatrix} a \\ c \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + cy + b = 0$$

$$\updownarrow$$

$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

Lines in R^2



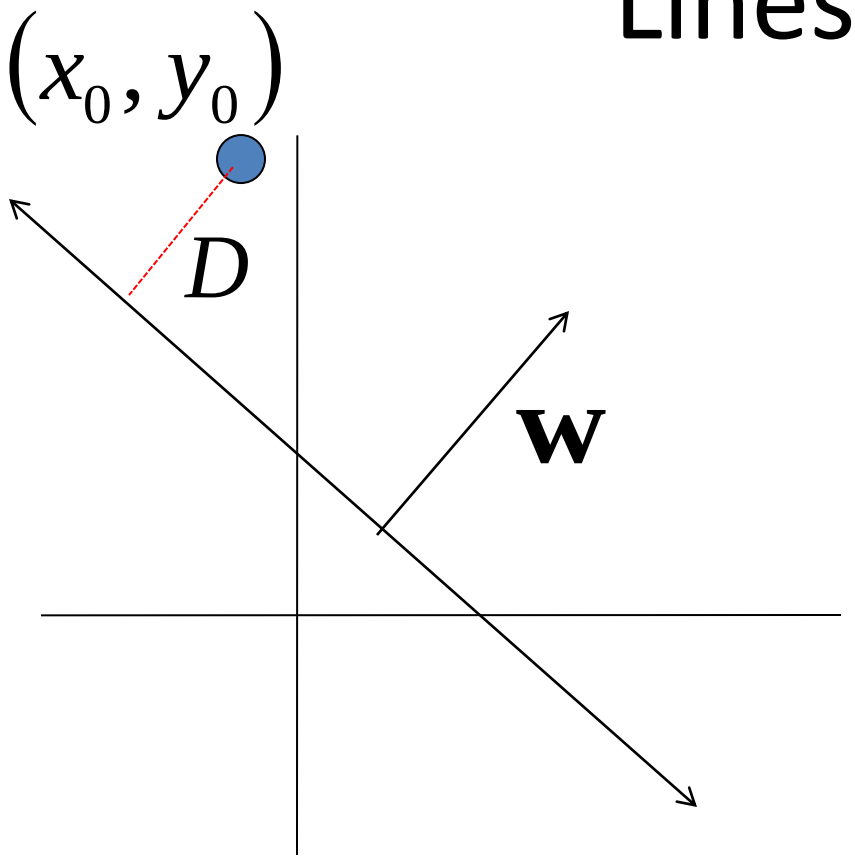
Let $\mathbf{w} = \begin{bmatrix} a \\ c \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + cy + b = 0$$

$$\updownarrow$$

$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

Lines in R^2



Let $\mathbf{w} = \begin{bmatrix} a \\ c \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + cy + b = 0$$

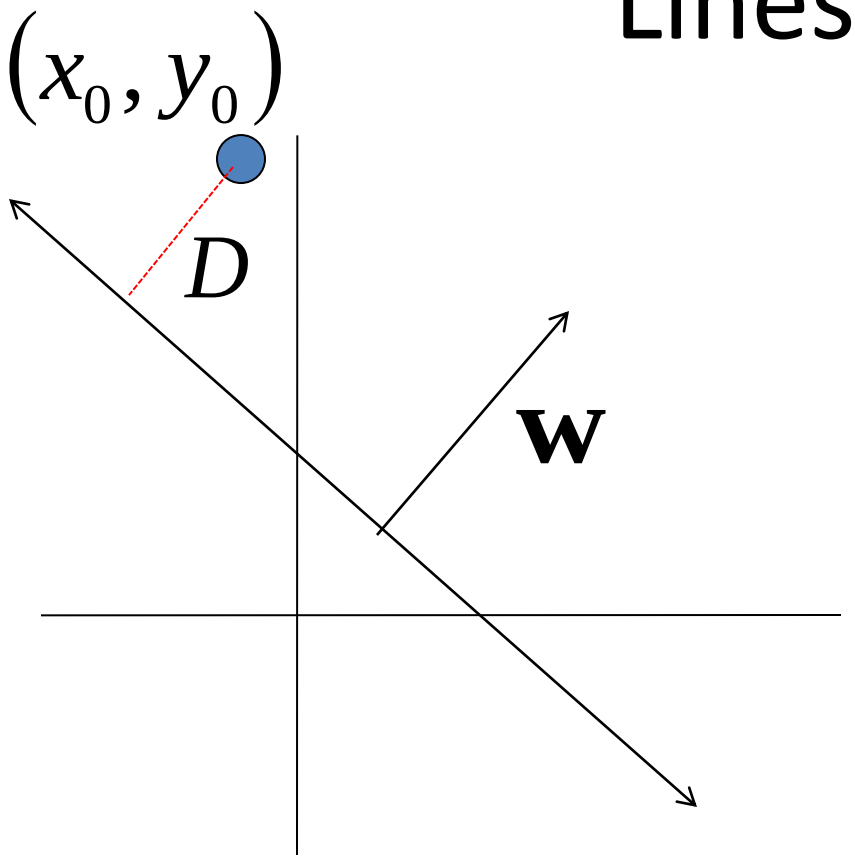
$$\updownarrow$$

$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

$$D = \frac{|ax_0 + cy_0 + b|}{\sqrt{a^2 + c^2}}$$

distance from point to line

Lines in R^2



Let $\mathbf{w} = \begin{bmatrix} a \\ c \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + cy + b = 0$$

$$\updownarrow$$

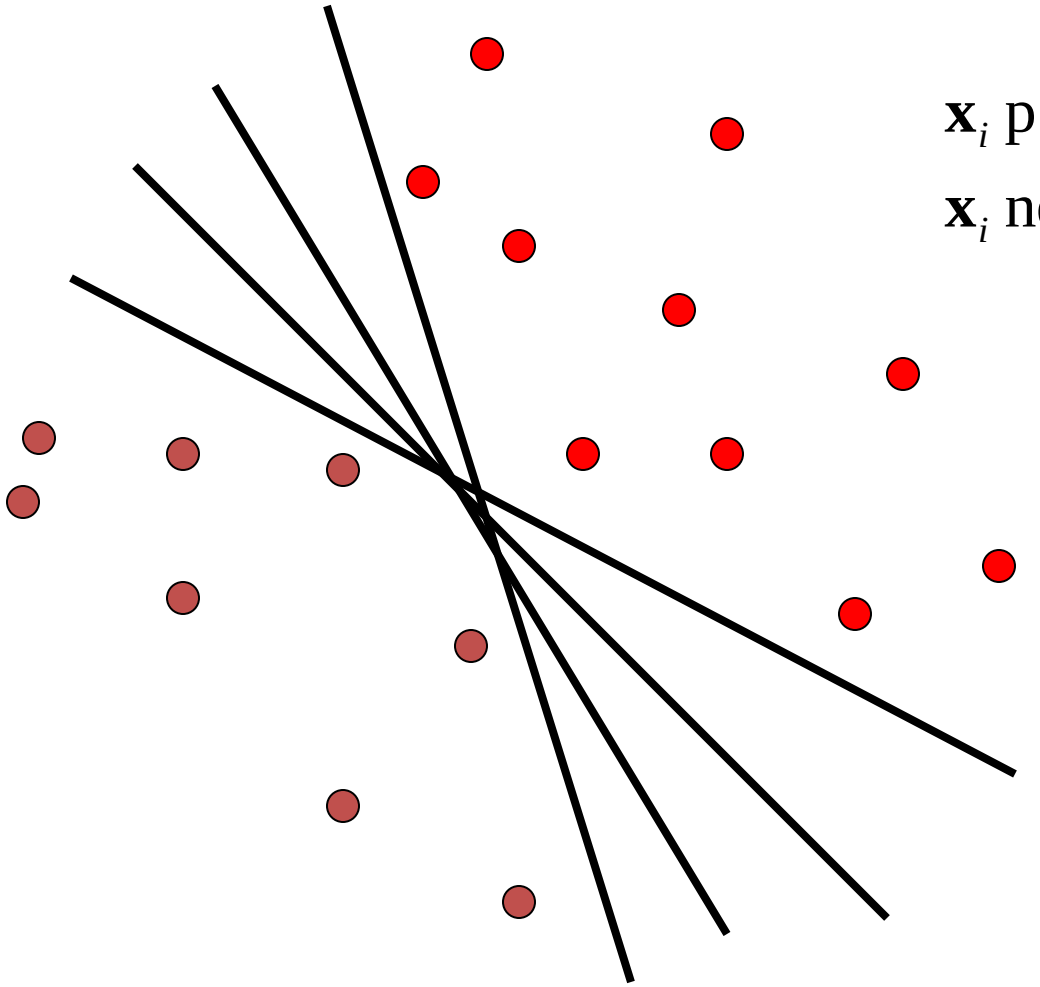
$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

$$D = \frac{|ax_0 + cy_0 + b|}{\sqrt{a^2 + c^2}} = \frac{\mathbf{w}^T\mathbf{x} + b}{\|\mathbf{w}\|}$$

distance from point to line

Linear classifiers

- Find linear function to separate positive and negative examples

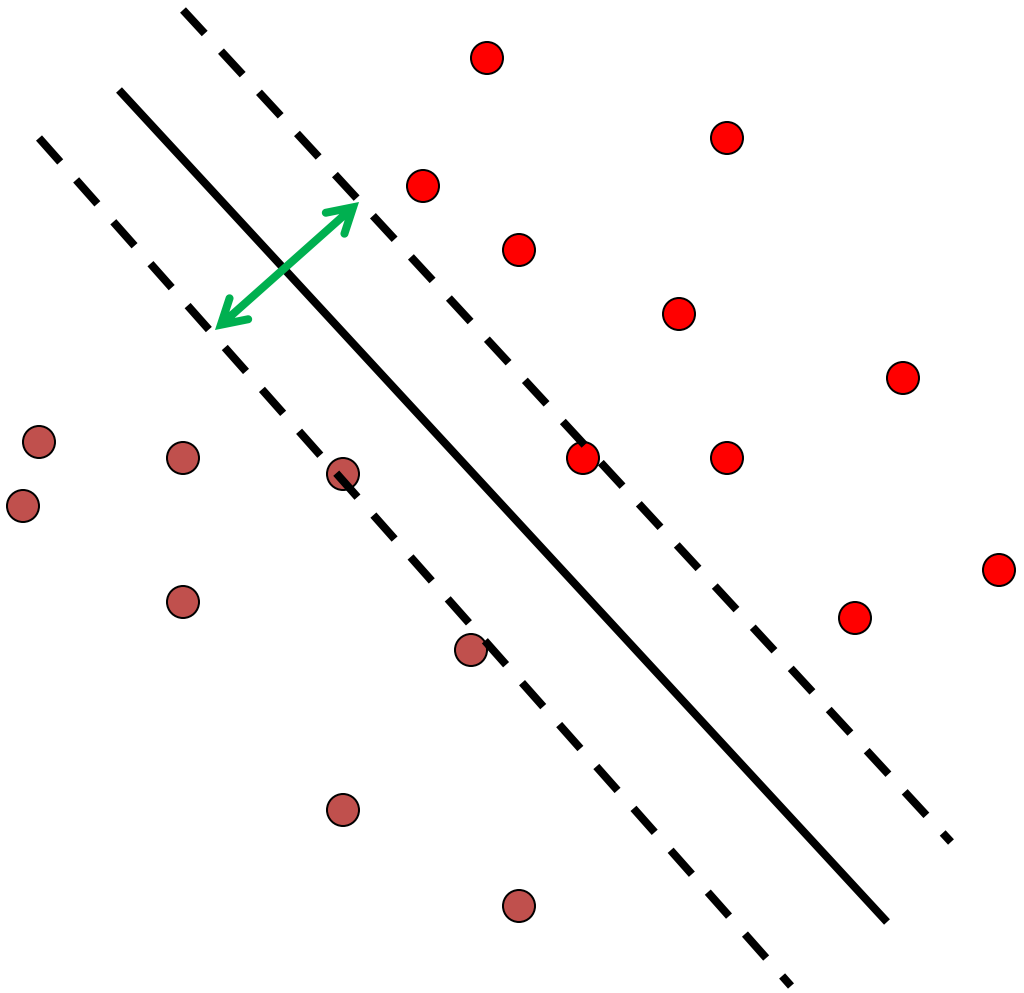


$$\mathbf{x}_i \text{ positive:} \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 0$$

$$\mathbf{x}_i \text{ negative:} \quad \mathbf{x}_i \cdot \mathbf{w} + b < 0$$

Which line is best?

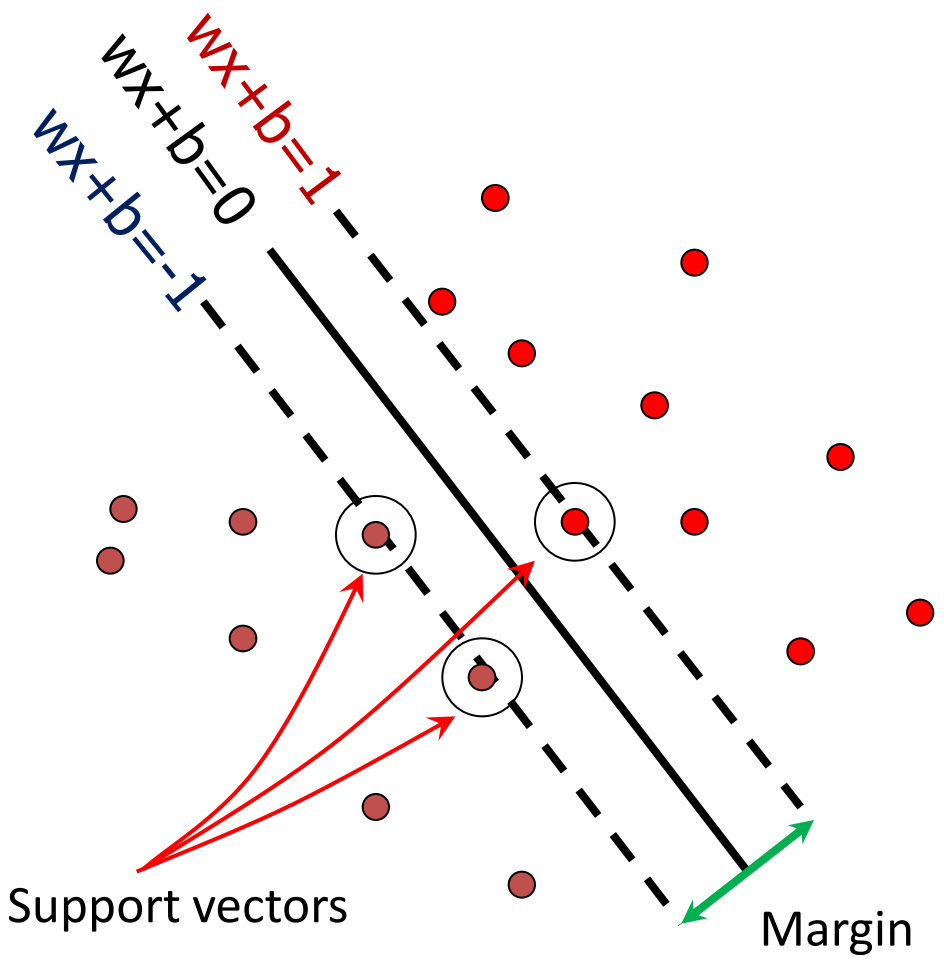
Support Vector Machines (SVMs)



- Discriminative classifier based on *optimal separating line (for 2d case)*
- Maximize the *margin* between the positive and negative training examples

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

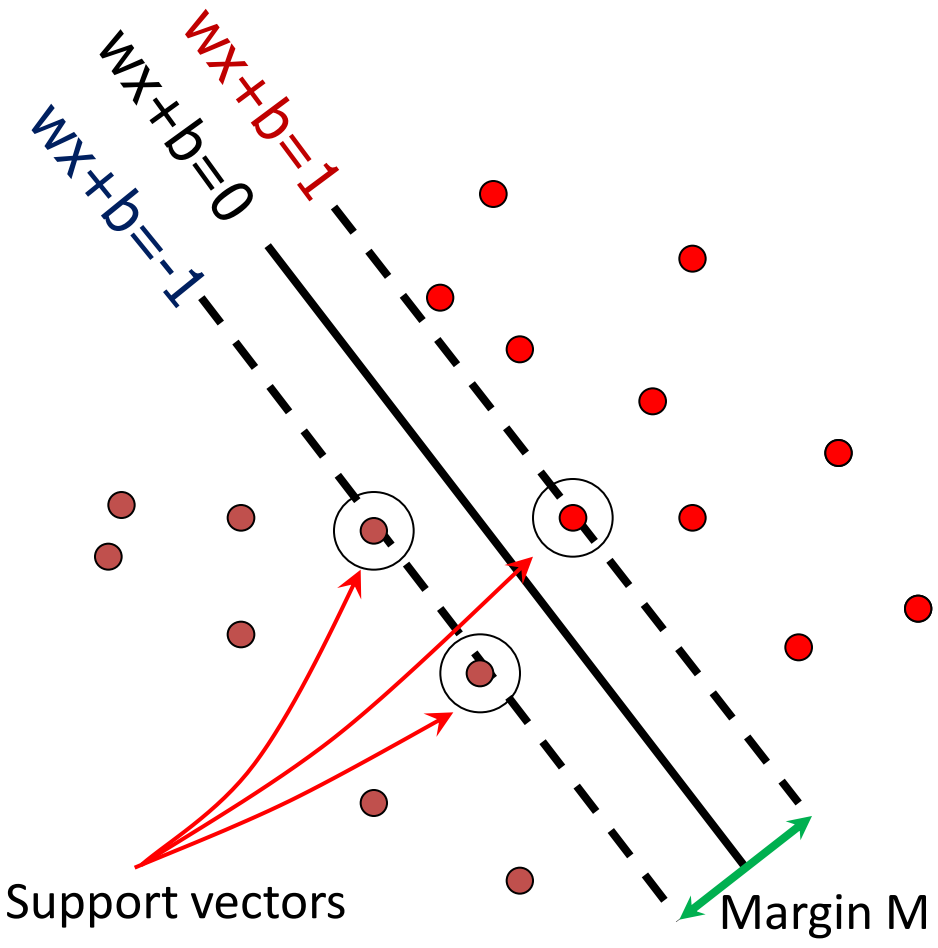
$\mathbf{x}_i$ negative $(y_i = -1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

For support, vectors, $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

$\mathbf{x}_i$ negative $(y_i = -1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

For support, vectors, $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

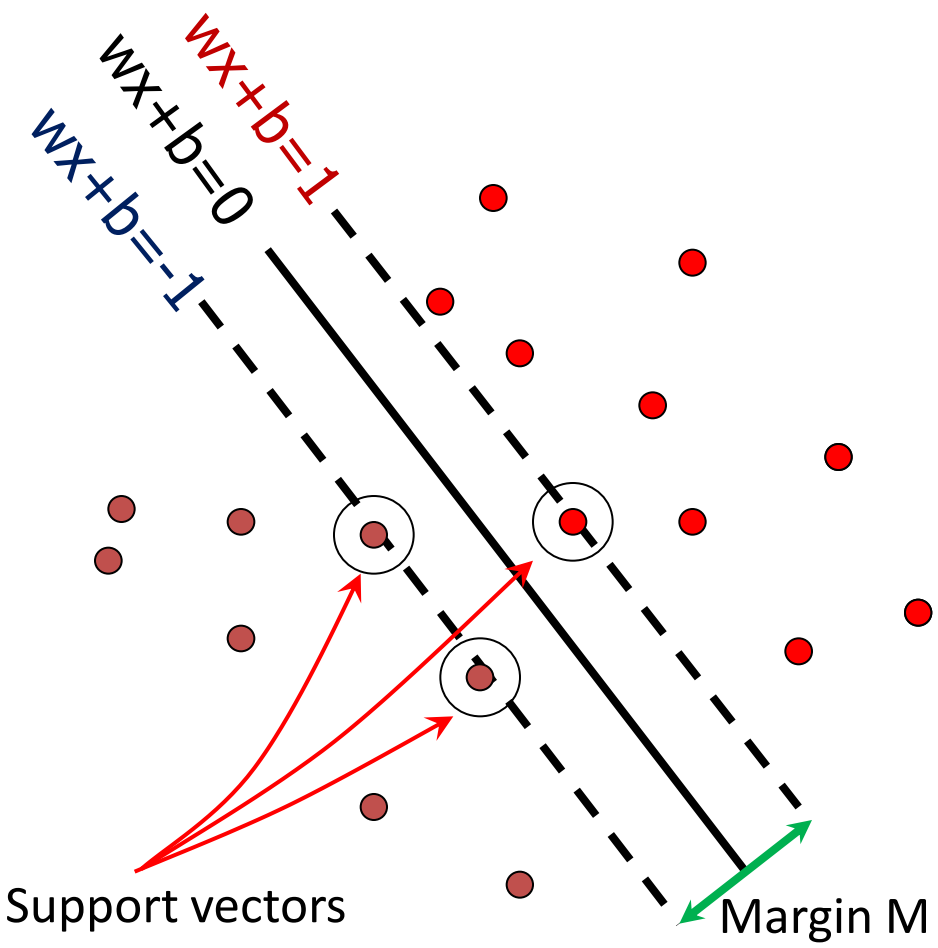
Distance between point and line: $\frac{|\mathbf{x}_i \cdot \mathbf{w} + b|}{\|\mathbf{w}\|}$

For support vectors:

$$\frac{\mathbf{w}^T \mathbf{x} + b}{\|\mathbf{w}\|} = \frac{\pm 1}{\|\mathbf{w}\|} \qquad M = \left| \frac{1}{\|\mathbf{w}\|} - \frac{-1}{\|\mathbf{w}\|} \right| = \frac{2}{\|\mathbf{w}\|}$$

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

$\mathbf{x}_i$ negative $(y_i = -1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

For support, vectors, $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

Distance between point and line: $\frac{|\mathbf{x}_i \cdot \mathbf{w} + b|}{\|\mathbf{w}\|}$

Therefore, the margin is $2 / \|\mathbf{w}\|$

Finding the maximum margin line

1. Maximize margin $2/\|\mathbf{w}\|$
2. Correctly classify all training data points:

$$\mathbf{x}_i \text{ positive}(y_i = 1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$$

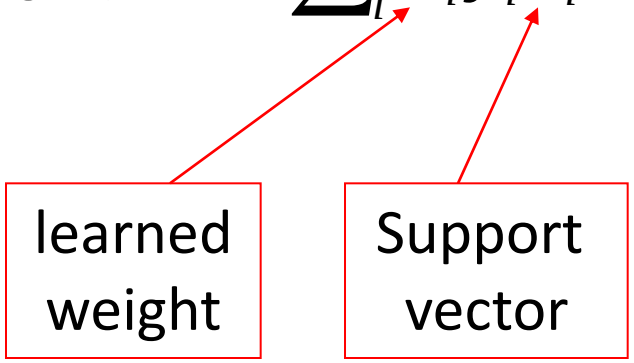
$$\mathbf{x}_i \text{ negative}\,(y_i = -1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$$

- *Quadratic optimization problem*:
- Minimize $\frac{1}{2}\mathbf{w}^T\mathbf{w}$

  Subject to $y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1$

Finding the maximum margin line

- Solution: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$



Finding the maximum margin line

- Solution: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$

  $b = y_i - \mathbf{w} \cdot \mathbf{x}_i$ (for any support vector)

  $\mathbf{w} \cdot \mathbf{x} + b = \sum_i \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b$

- Classification function:

If f(x) < 0, classify as negative,
if f(x) > 0, classify as positive

$$f(x) = \text{sign}\ (\mathbf{w} \cdot \mathbf{x} + \mathrm{b})$$
$$= \text{sign}\left(\sum_{\mathrm{i}} \alpha_{\mathrm{i}} \mathbf{x}_i \cdot \mathbf{x} + b\right)$$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

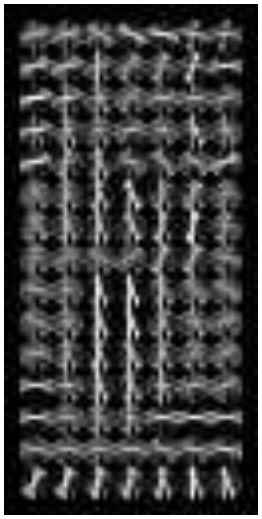
Questions

- **What if the features are not 2d?**
- What if the data is not linearly separable?
- What if we have more than just two categories?

Questions

- What if the features are not 2d?
  - Generalizes to d-dimensions – replace line with "hyperplane"
- What if the data is not linearly separable?
- What if we have more than just two categories?

Person detection with HoG's & linear SVM's



- Map each grid cell in the input window to a histogram counting the gradients per orientation.
- Train a linear SVM using training set of pedestrian vs. non-pedestrian windows.

Code available:
http://pascal.inrialpes.fr/soft/olt/

Dalal & Triggs, CVPR 2005

Person detection with HoG's & linear SVM's

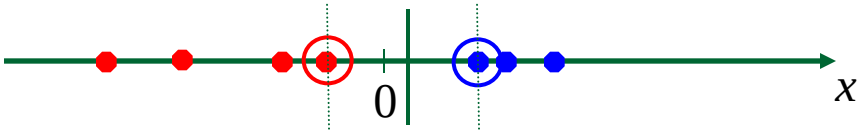
- Histograms of Oriented Gradients for Human Detection, Navneet Dalal, Bill Triggs, International Conference on Computer Vision & Pattern Recognition - June 2005
- http://lear.inrialpes.fr/pubs/2005/DT05/

Questions

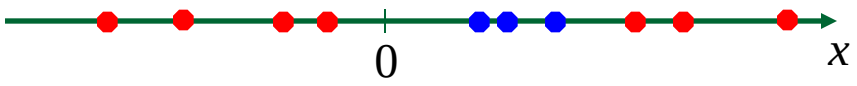
- What if the features are not 2d?
- **What if the data is not linearly separable?**
- What if we have more than just two categories?

Non-linear SVMs

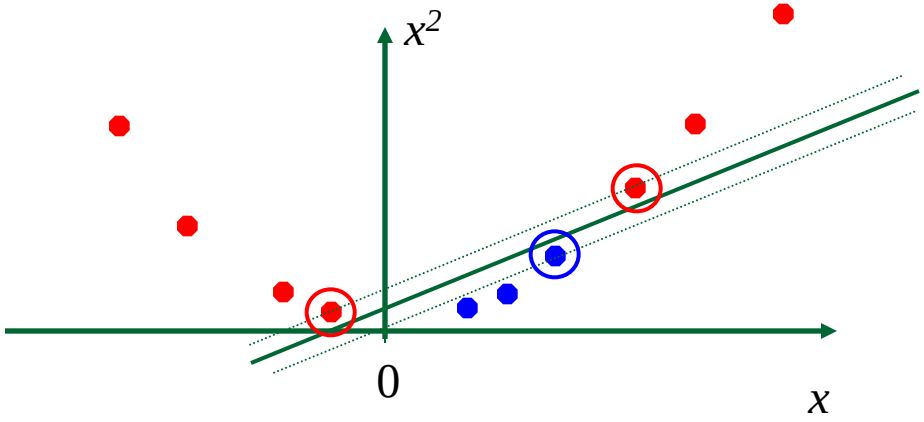
- Datasets that are linearly separable with some noise work out great:



- But what are we going to do if the dataset is just too hard?

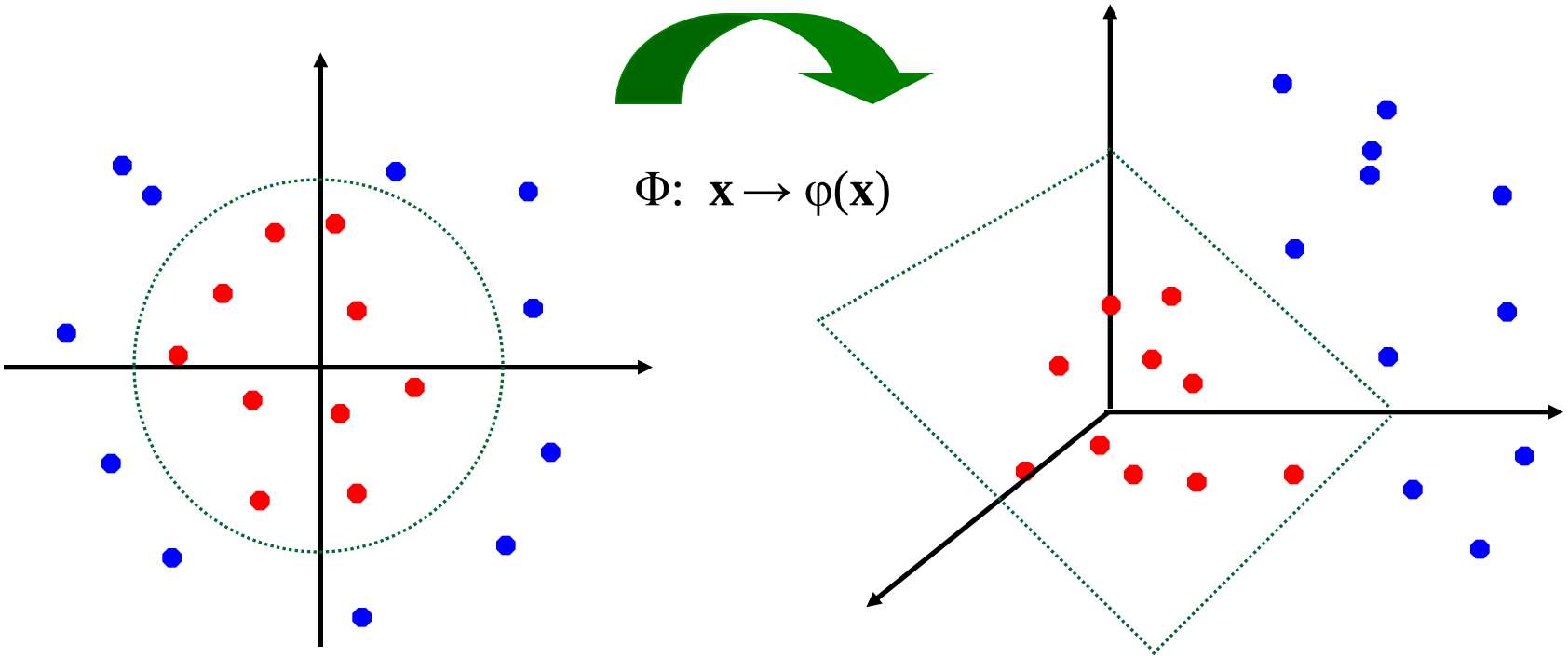


- How about… mapping data to a higher-dimensional space:



Non-linear SVMs: feature spaces

- General idea: the original input space can be mapped to some higher-dimensional feature space where the training set is separable:



Slide from Andrew Moore's tutorial: http://www.autonlab.org/tutorials/svm.html

The "Kernel Trick"

- The linear classifier relies on dot product between vectors $K(x_i,x_j)=x_i^T x_j$

- If every data point is mapped into high-dimensional space via some transformation Φ: $x \rightarrow \phi(x)$, the dot product becomes:

$$K(x_i,x_j)= \phi(x_i)^T \phi(x_j)$$

- A *kernel function* is similarity function that corresponds to an inner product in some expanded feature space.

Slide from Andrew Moore's tutorial: http://www.autonlab.org/tutorials/svm.html

Example

2-dimensional vectors $x=[x_1 \;\; x_2]$;

let $K(x_i,x_j)=(1 + x_i^T x_j)^2$

Need to show that $K(x_i,x_j)= \varphi(x_i)^T \varphi(x_j)$:

$K(x_i,x_j)=(1 + x_i^T x_j)^2$,

$= 1+ x_{i1}^2 x_{j1}^2 + 2\, x_{i1}x_{j1}\, x_{i2}x_{j2} + x_{i2}^2 x_{j2}^2 + 2x_{i1}x_{j1} + 2x_{i2}x_{j2}$

$= [1 \;\; x_{i1}^2 \;\sqrt{2}\, x_{i1}x_{i2} \;\; x_{i2}^2 \;\sqrt{2}x_{i1} \;\sqrt{2}x_{i2}]^T$

$[1 \;\; x_{j1}^2 \;\sqrt{2}\, x_{j1}x_{j2} \;\; x_{j2}^2 \;\sqrt{2}x_{j1} \;\sqrt{2}x_{j2}]$

$= \varphi(x_i)^T \varphi(x_j)$,

where $\varphi(x) = [1 \;\; x_1^2 \;\sqrt{2}\, x_1x_2 \;\; x_2^2 \;\; \sqrt{2}x_1 \;\sqrt{2}x_2]$

from Andrew Moore's tutorial: http://www.autonlab.org/tutorials/svm.html

Nonlinear SVMs

- *The kernel trick*: instead of explicitly computing the lifting transformation $\boldsymbol{\varphi}(\mathbf{x})$, define a kernel function K such that

$$K(\mathbf{x}_i, \mathbf{x}_j) = \boldsymbol{\varphi}(\mathbf{x}_i) \cdot \boldsymbol{\varphi}(\mathbf{x}_j)$$

- This gives a nonlinear decision boundary in the original feature space:

$$\sum_i \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$$

Examples of kernel functions

$$K(x_i, x_j) = x_i^T x_j$$

- Linear:

$$K(x_i, x_j) = \exp\left(-\frac{\left\|x_i - x_j\right\|^2}{2\sigma^2}\right)$$

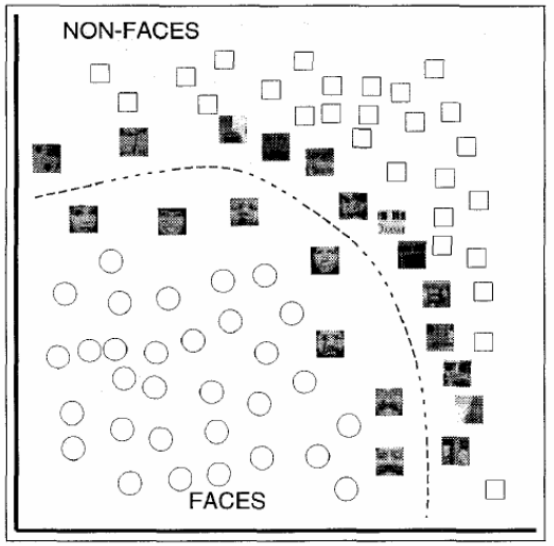
- Gaussian RBF:

$$K(x_i, x_j) = \sum_k \min(x_i(k), x_j(k))$$

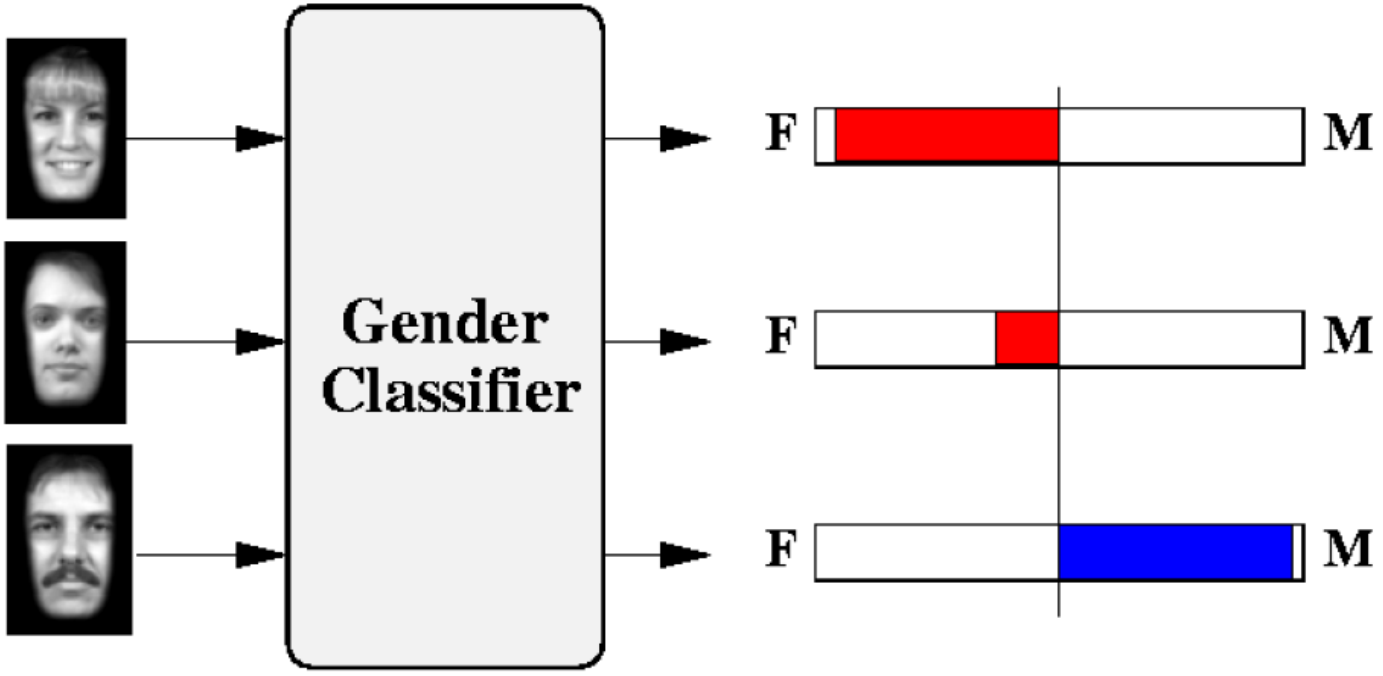
- Histogram intersection:

SVMs for recognition

1. Define your representation for each example.
2. Select a kernel function.
3. Compute pairwise kernel values between labeled examples
4. Use this “kernel matrix” to solve for SVM support vectors & weights.
5. To classify a new example: compute kernel values between new input and support vectors, apply weights, check sign of output.



Example: learning gender with SVMs



Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.

Moghaddam and Yang, Face & Gesture 2000.

Face alignment processing

Processed faces

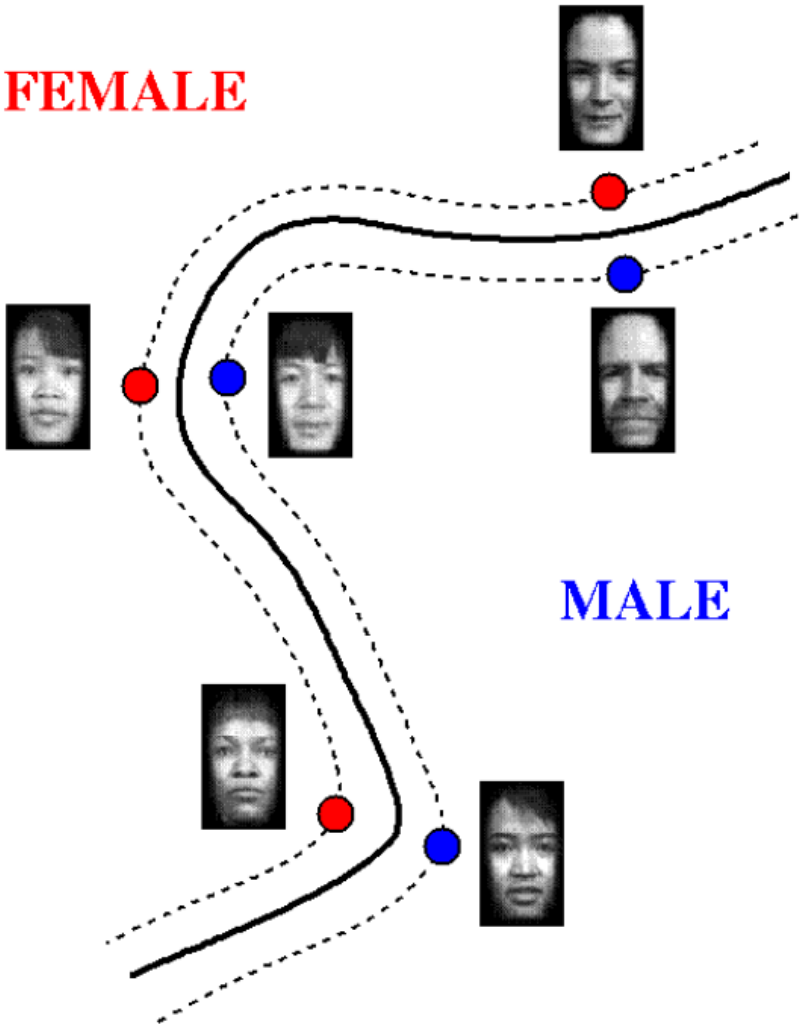
Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.

Learning gender with SVMs

- Training examples:
  - 1044 males
  - 713 females
- Experiment with various kernels, select Gaussian RBF

$$K(\mathbf{x_i}, \mathbf{x_j}) = \exp\left(-\frac{\left\|\mathbf{x_i} - \mathbf{x_j}\right\|^2}{2\sigma^2}\right)$$

Support Faces



Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.

Classifier Performance

Classifier	Error Rate		
	Overall	Male	Female
SVM with RBF kernel	**3.38%**	**2.05%**	**4.79%**
SVM with cubic polynomial kernel	**4.88%**	**4.21%**	**5.59%**
Large Ensemble of RBF	5.54%	4.59%	6.55%
Classical RBF	7.79%	6.89%	8.75%
Quadratic classifier	10.63%	9.44%	11.88%
Fisher linear discriminant	13.03%	12.31%	13.78%
Nearest neighbor	27.16%	26.53%	28.04%
Linear classifier	58.95%	58.47%	59.45%

Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.

Gender perception experiment: How well can humans do?

- Subjects:
  - 30 people (22 male, 8 female)
  - Ages mid-20's to mid-40's
- Test data:
  - 254 face images (6 males, 4 females)
  - Low res and high res versions
- Task:
  - Classify as male or female, forced choice
  - No time limit

Moghaddam and Yang, Face & Gesture 2000.

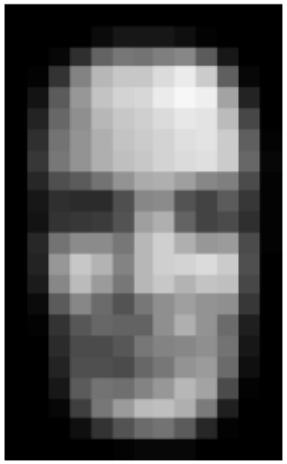
Gender perception experiment: How well can humans do?

Stimuli →

84 x 48

N = 4032

21 x 12



N = 252

Results →

High-Res	Low-Res
6.54%	30.7%
Error	Error

$\sigma = 3.7\%$

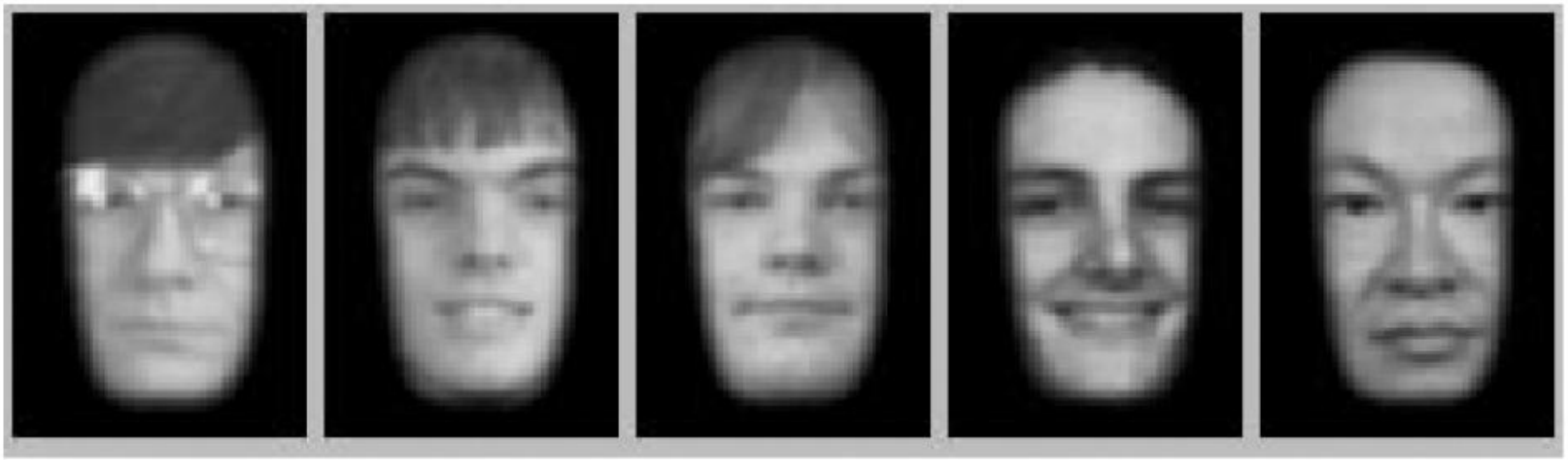
Moghaddam and Yang, Face & Gesture 2000.

Human vs. Machine

Figure 6. SVM vs. Human performance

- SVMs performed better than any single human test subject, at either resolution

Hardest examples for humans



Top five human misclassifications

Moghaddam and Yang, Face & Gesture 2000.

Questions

- What if the features are not 2d?
- What if the data is not linearly separable?
- **What if we have more than just two categories?**

Multi-class SVMs

- Achieve multi-class classifier by combining a number of binary classifiers

- **One vs. all**
  - Training: learn an SVM for each class vs. the rest
  - Testing: apply each SVM to test example and assign to it the class of the SVM that returns the highest decision value

- **One vs. one**
  - Training: learn an SVM for each pair of classes
  - Testing: each learned SVM “votes” for a class to assign to the test example

SVMs: Pros and cons

- Pros
  - Many publicly available SVM packages: http://www.kernel-machines.org/software
  - http://www.csie.ntu.edu.tw/~cjlin/libsvm/
  - Kernel-based framework is very powerful, flexible
  - Often a sparse set of support vectors – compact at test time
  - Work very well in practice, even with very small training sample sizes
- Cons
  - No “direct” multi-class SVM, must combine two-class SVMs
  - Can be tricky to select best kernel function for a problem
  - Computation, memory
    - During training time, must compute matrix of kernel values for every pair of examples
    - Learning can take a very long time for large-scale problems

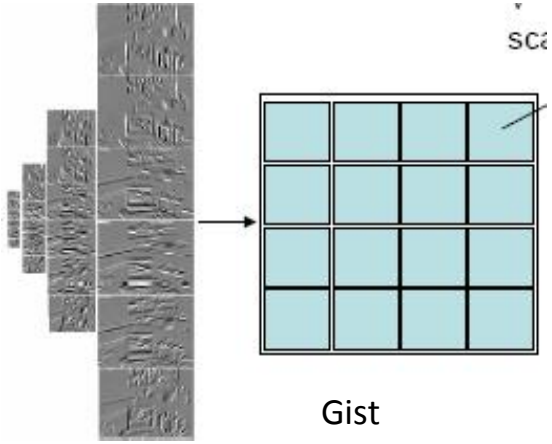
Adapted from Lana Lazebnik

Summary

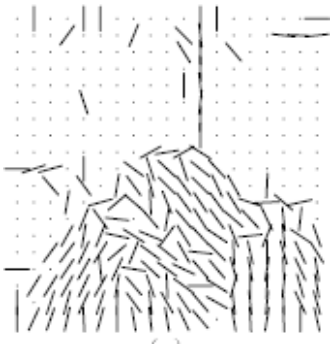
- Discriminative classifiers
  - Boosting
  - Nearest neighbors
  - Support vector machines

- Useful for object recognition when combined with “window-based” or holistic appearance descriptors

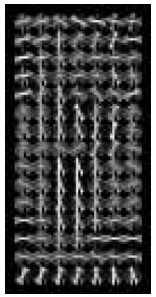
Global window-based appearance representations



Gist



Map of local orientations



Histograms of oriented gradients

- These examples are truly global; each pixel in the window contributes to the representation.
- Classifier can account for relative relevance…
- *When might this not be ideal?*

- Part-based and local feature models for generic object recognition

Part-based and local feature models for recognition

Main idea:

Rather than a representation based on holistic appearance, decompose the image into:

- local parts or patches, and
- their relative spatial relationships

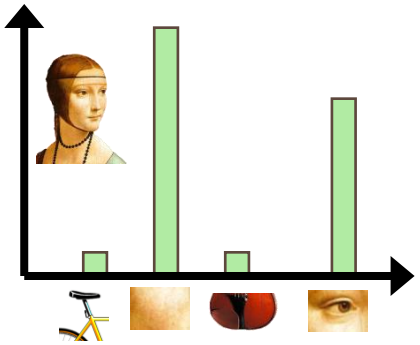
Part-based and local feature models for recognition

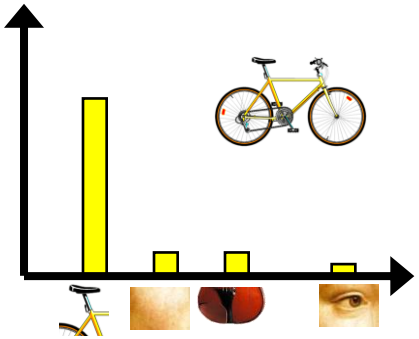
We'll look at three forms:

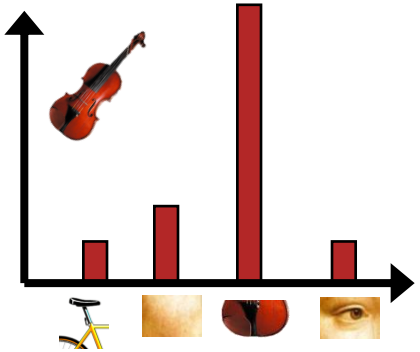
1. **Bag of words** (no geometry)
2. **Implicit shape model** (star graph for spatial model)
3. **Constellation model** (fully connected graph for spatial model)

Bag-of-words model

- Summarize entire image based on its distribution (histogram) of word occurrences.
  - Total freedom on spatial positions, relative geometry.
  - Vector representation easily usable by most classifiers.

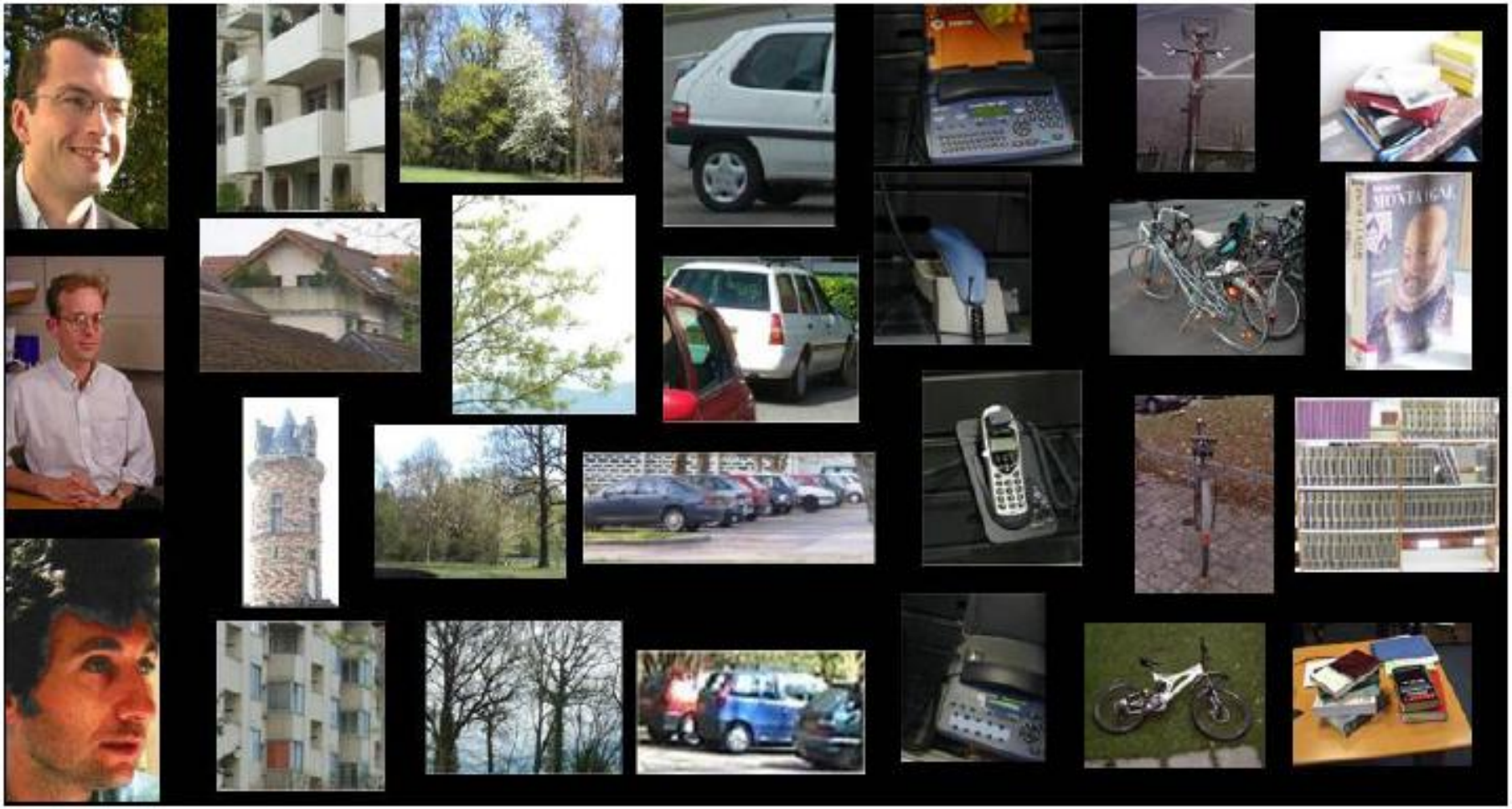






Kristen Grauman

Bag-of-words model



Our in-house database contains 1776 images in seven classes[1]: faces, buildings, trees, cars, phones, bikes and books. Fig. 2 shows some examples from this dataset.

Csurka et al. Visual Categorization with Bags of Keypoints, 2004

Words as parts

All local features

Local features from two selected clusters occurring in this image

Csurka et al. 2004

Naïve Bayes model for classification

$$c^* = \arg\max_c \; p(c \mid w) \propto p(c)\, p(w \mid c) = p(c) \prod_{n=1}^{N} p(w_n \mid c)$$

Object class decision

Prior prob. of the object classes

Image likelihood given the class

N patches

What assumptions does the model make, and what are their significance?

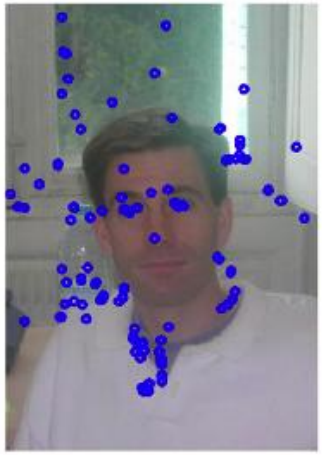
Confusion matrix

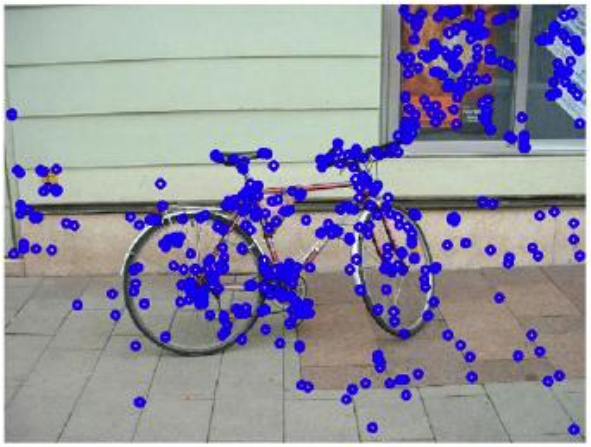
True classes →	*faces*	*buildings*	*trees*	*cars*	*phones*	*bikes*	*books*
faces	**76**	4	2	3	4	4	13
buildings	2	**44**	5	0	5	1	3
trees	3	2	**80**	0	0	5	0
cars	4	1	0	**75**	3	1	4
phones	9	15	1	16	**70**	14	11
bikes	2	15	12	0	8	**73**	0
books	4	19	0	6	7	2	**69**

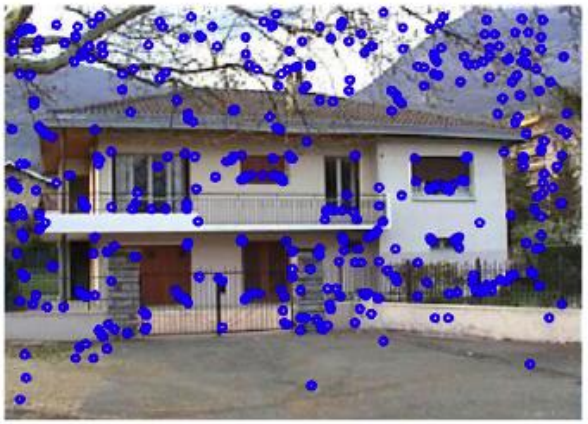
Example bag of words + Naïve Bayes classification results for generic categorization of objects

Csurka et al. 2004

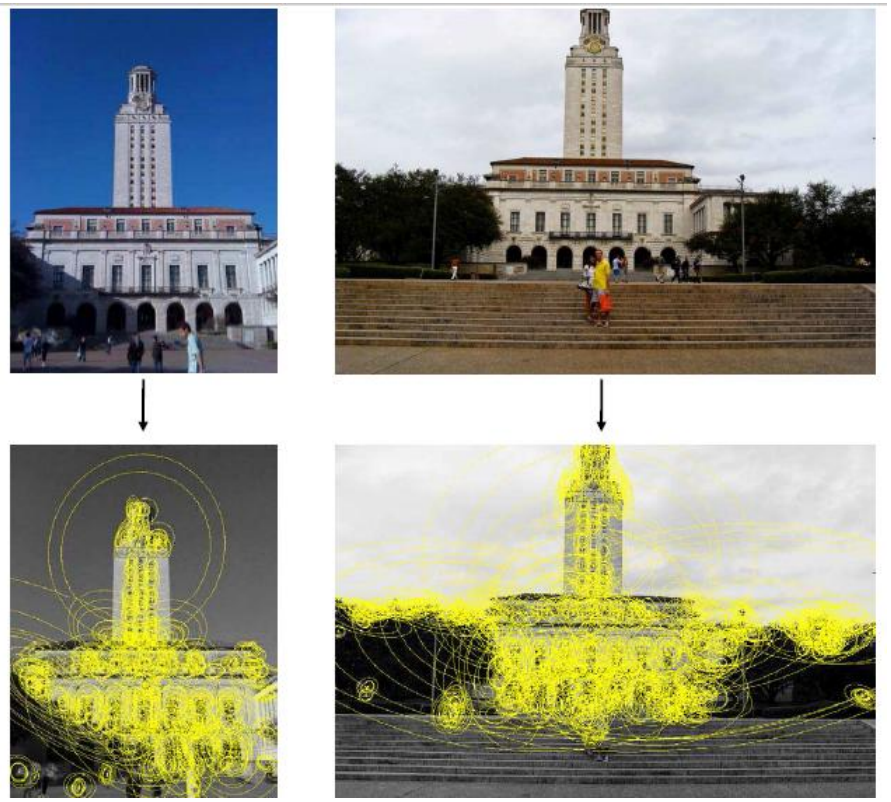
Clutter…or context?



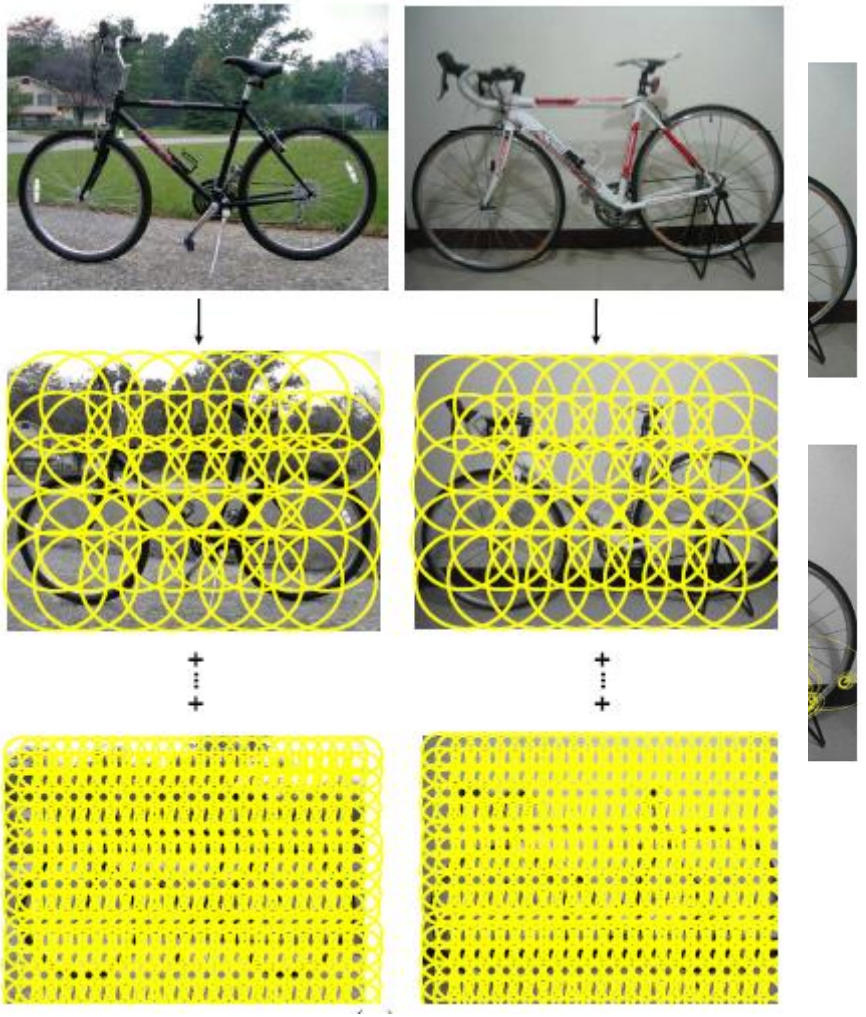




Sampling strategies



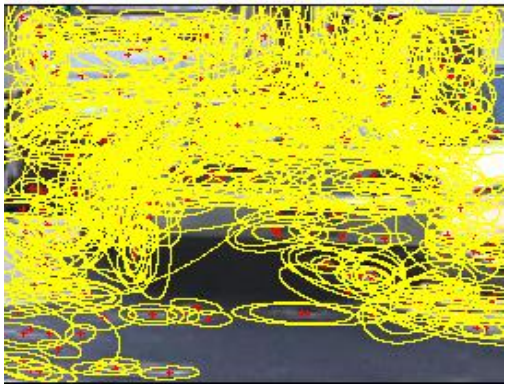
Specific object



Category

Kristen Grauman

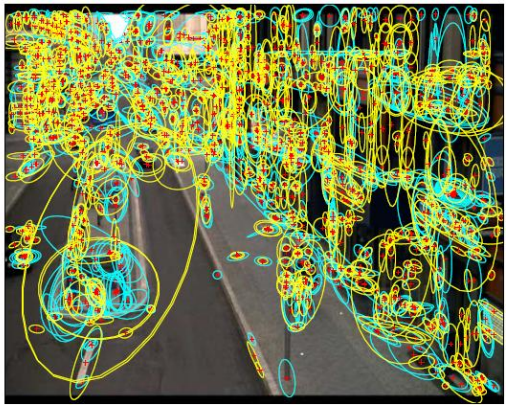
Sampling strategies



Sparse, at interest points

Dense, uniformly

Randomly



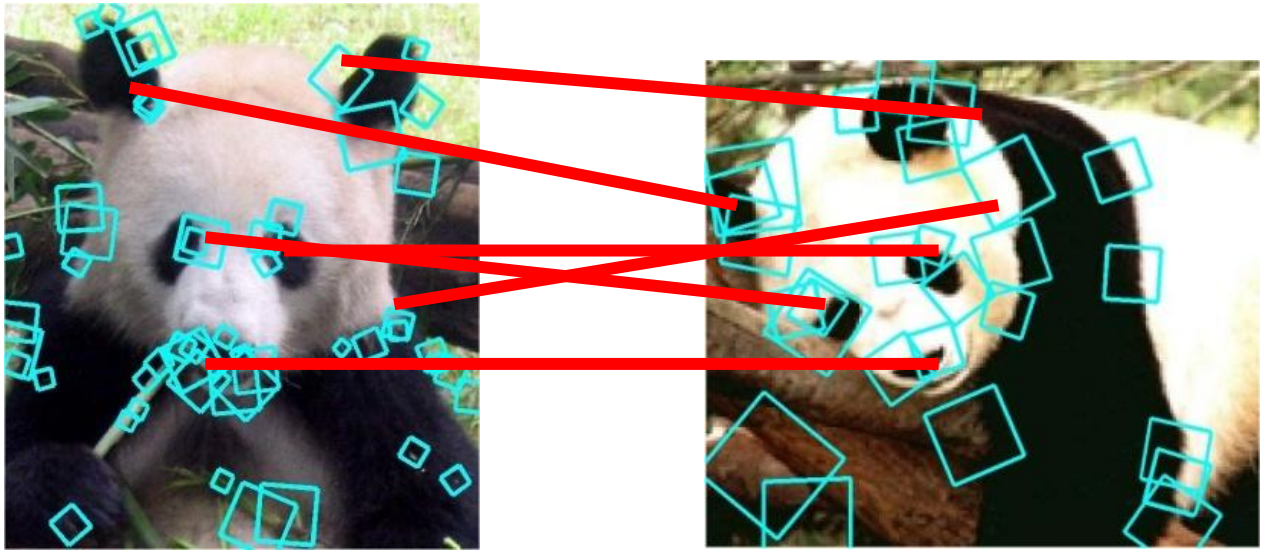
Multiple interest operators

- To find specific, textured objects, sparse sampling from interest points more reliable.
- Multiple complementary interest operators offer more image coverage.
- For object categorization, dense sampling offers better coverage.

Image credits: F-F. Li, E. Nowak, J. Sivic

[See Nowak, Jurie & Triggs, ECCV 2006]

Local feature correspondence for generic object categories

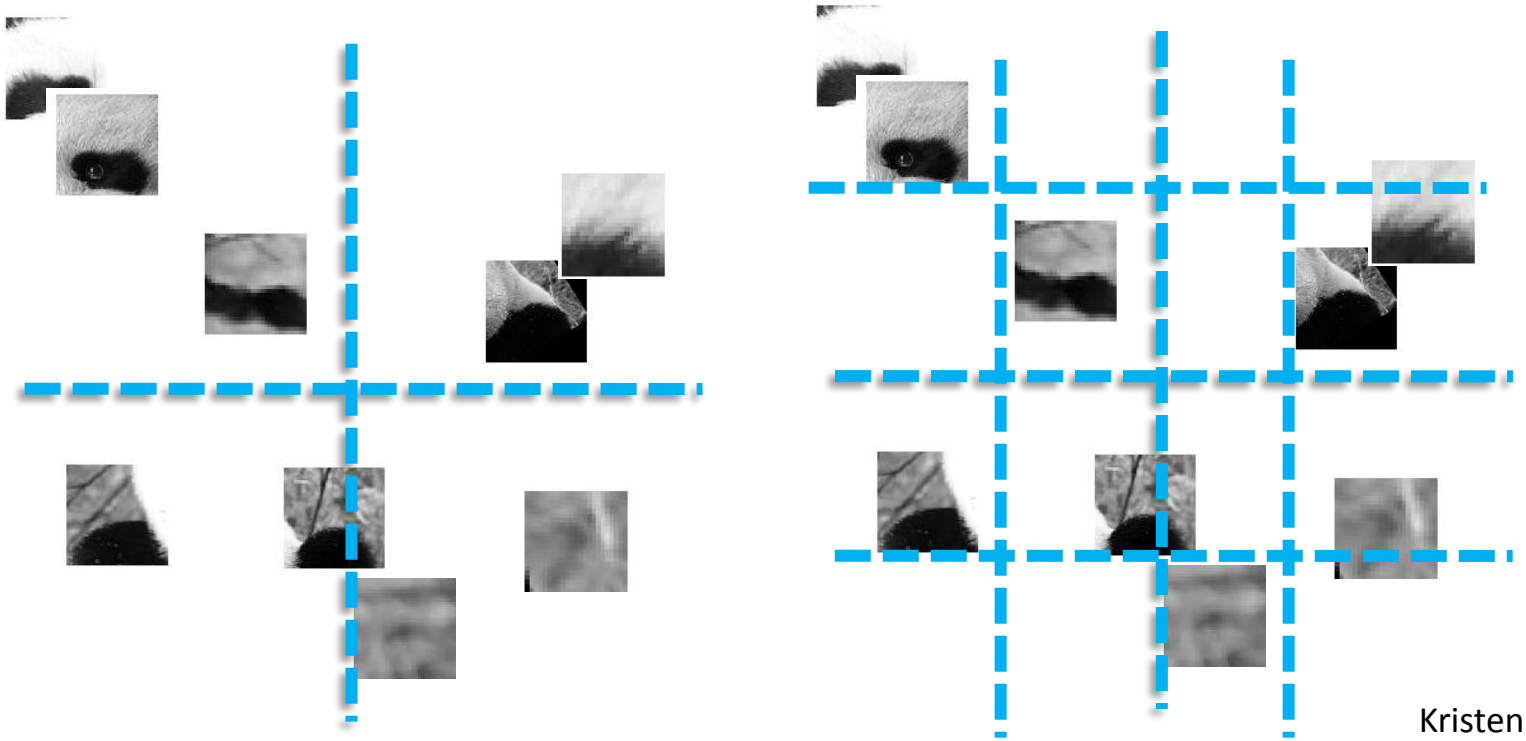


$\mathbf{X} = \{\vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_m\}$ $\mathbf{Y} = \{\vec{\mathbf{y}}_1, \ldots, \vec{\mathbf{y}}_n\}$

Kristen Grauman

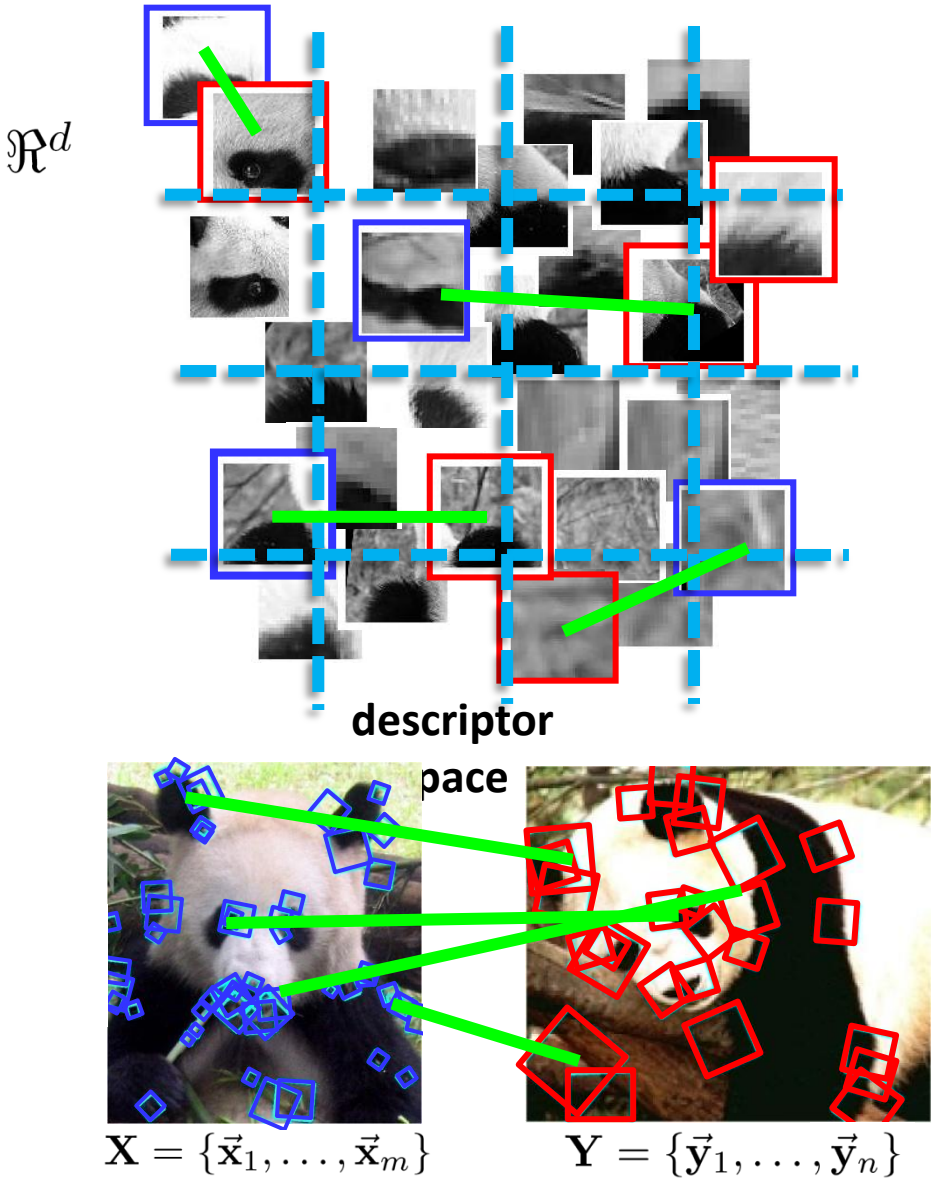
Local feature correspondence for generic object categories

- Comparing bags of words histograms coarsely reflects agreement between local “parts” (patches, words).
- *But* choice of quantization directly determines what we consider to be similar…



Kristen Grauman

Pyramid match: main idea

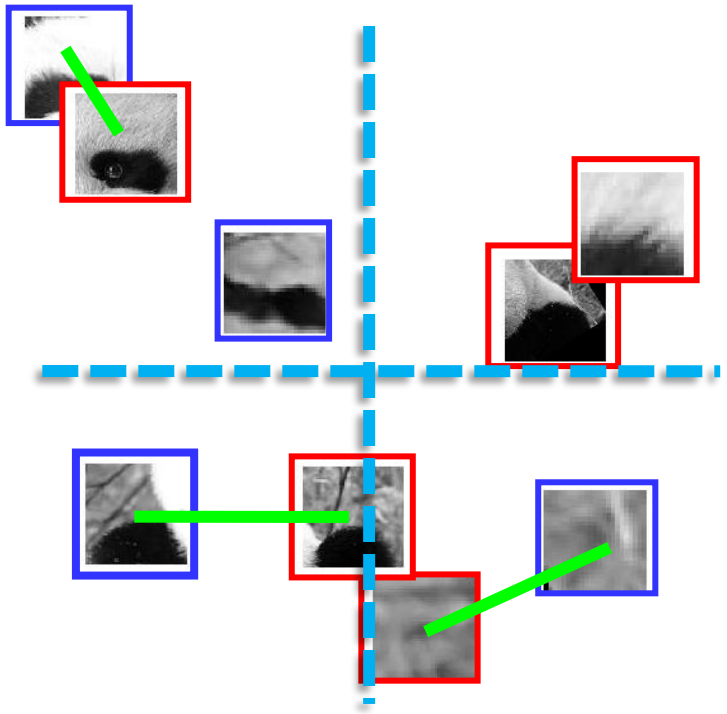


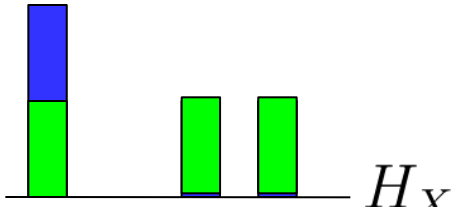
Feature space partitions serve to "match" the local descriptors within successively wider regions.

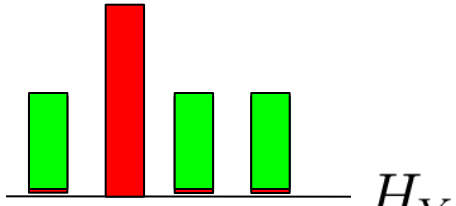
[Grauman & Darrell, ICCV 2005]

Pyramid match: main idea

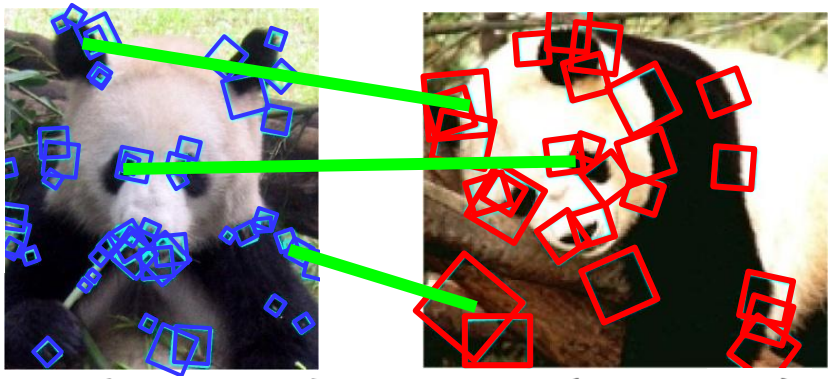
$\Re^d$







$$\mathcal{I}(H_X, H_Y) = \sum_j \min\left(H_X(j), H_Y(j)\right)$$
$$= 3$$

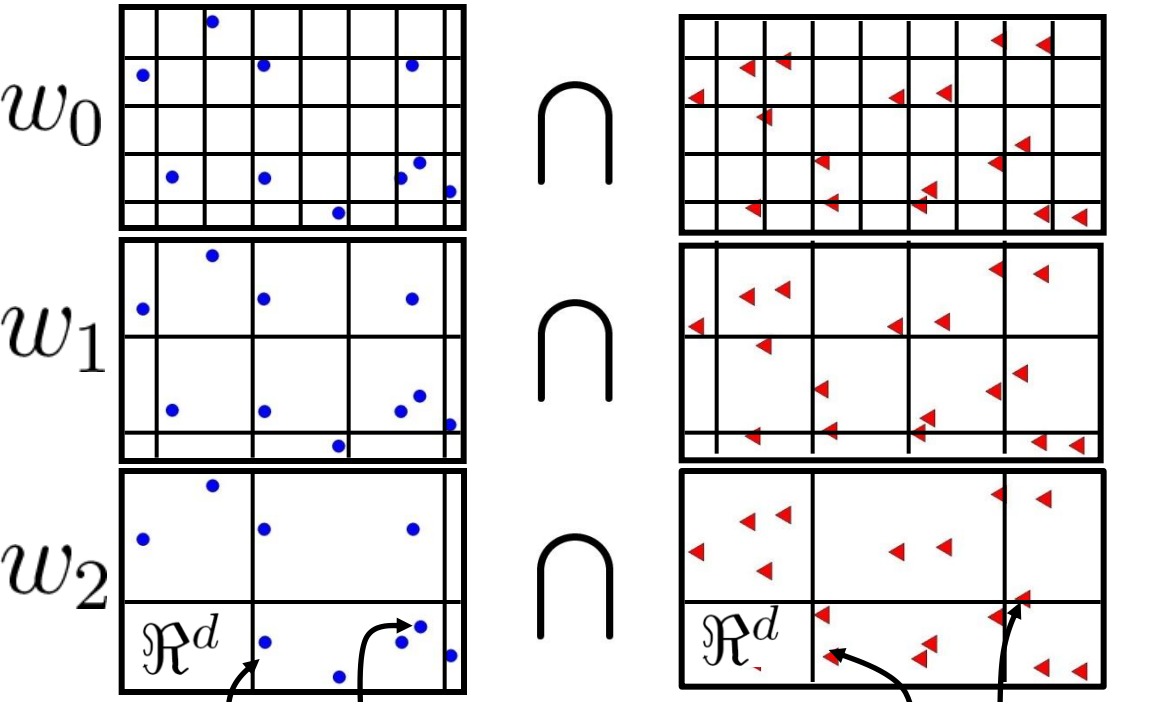


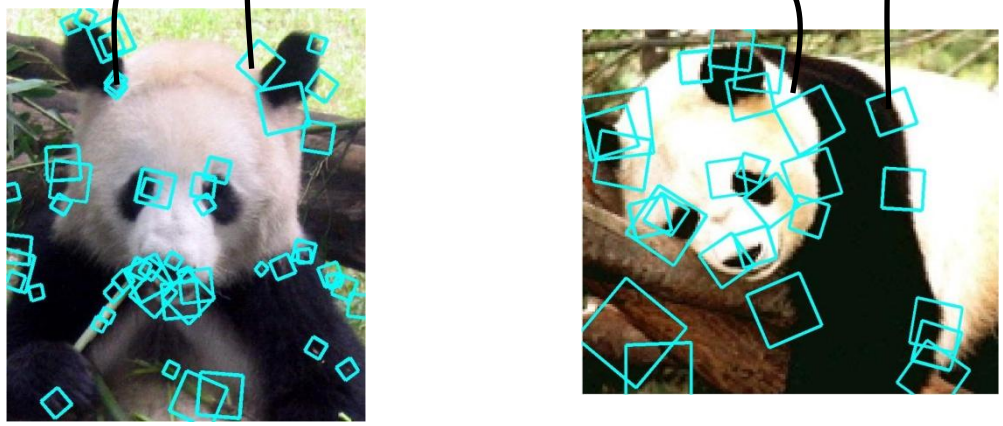
$\mathbf{X} = \{\vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_m\}$ $\mathbf{Y} = \{\vec{\mathbf{y}}_1, \ldots, \vec{\mathbf{y}}_n\}$

Histogram intersection counts number of possible matches at a given partitioning.

[Grauman & Darrell, ICCV 2005]

Pyramid match kernel

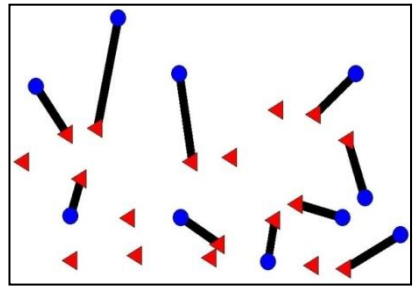




$\mathbf{X} = \{\vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_m\}$ $\mathbf{Y} = \{\vec{\mathbf{y}}_1, \ldots, \vec{\mathbf{y}}_n\}$

Optimal match: $O(m^3)$

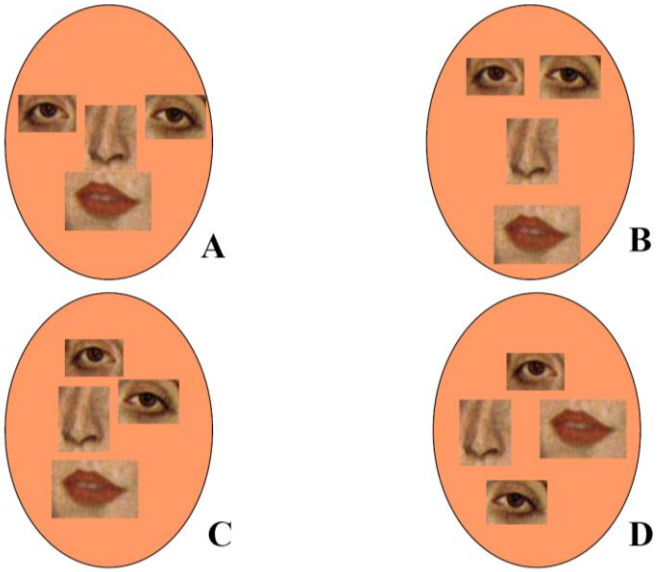
Pyramid match: $O(mL)$



optimal partial matching

[Grauman & Darrell, ICCV 2005]

Unordered sets of local features: **No** spatial layout preserved!

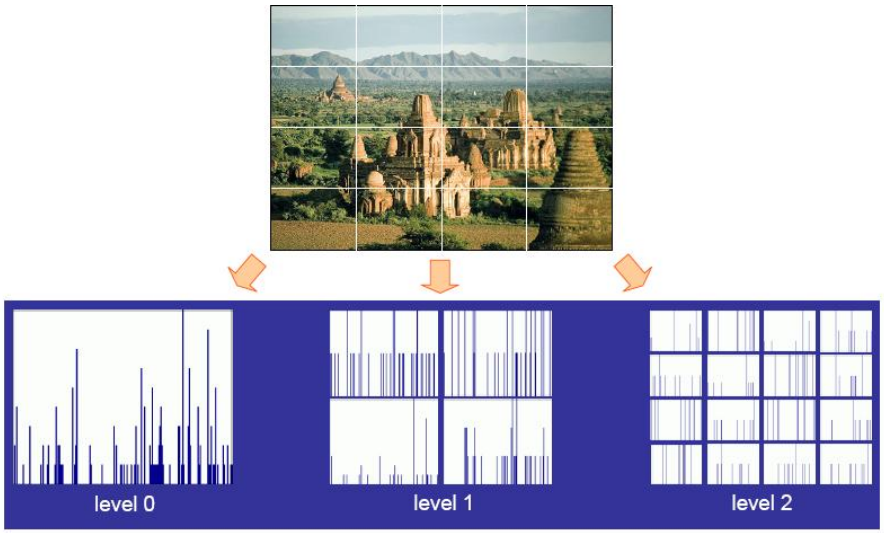


Too much?

Too little?

Spatial pyramid match

- Make a pyramid of bag-of-words histograms.
- Provides some loose (global) spatial layout information



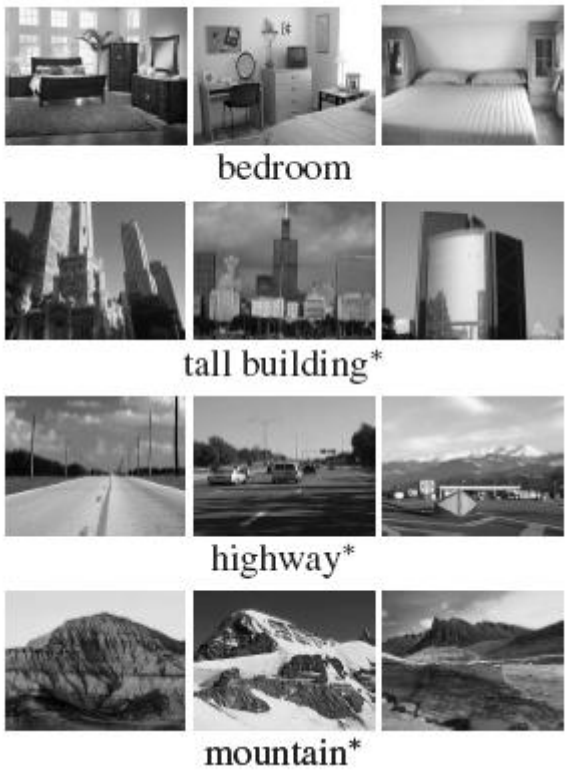
$$K^L(X,Y) = \sum_{m=1}^{M} \kappa^L(X_m, Y_m)$$

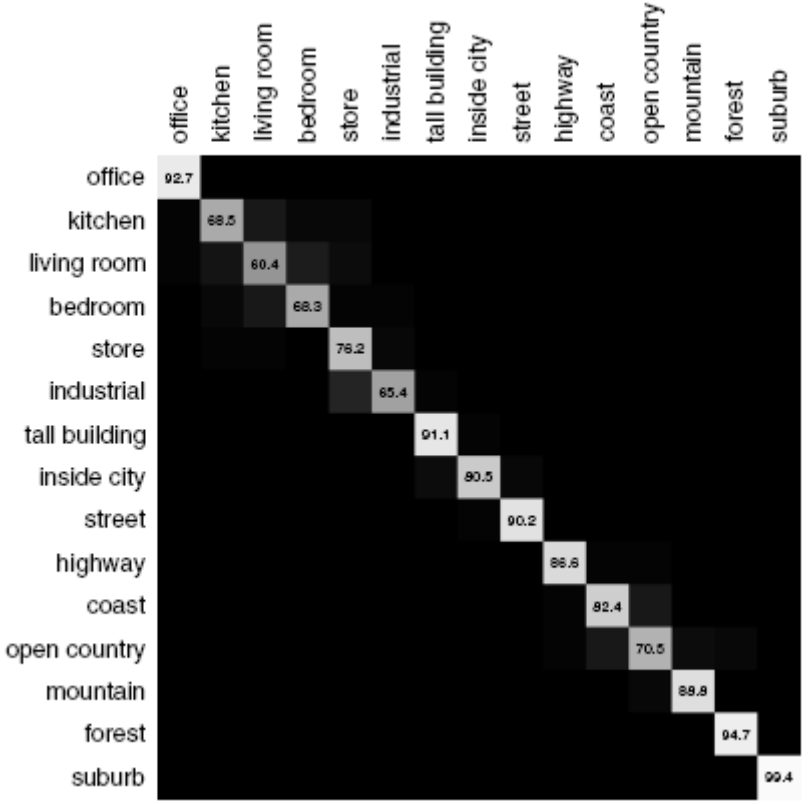
Sum over PMKs computed in *image coordinate* space, one per word.

[Lazebnik, Schmid & Ponce, CVPR 2006]

Spatial pyramid match

Captures scene categories well---texture-like patterns but with some variability in the positions of all the local pieces.

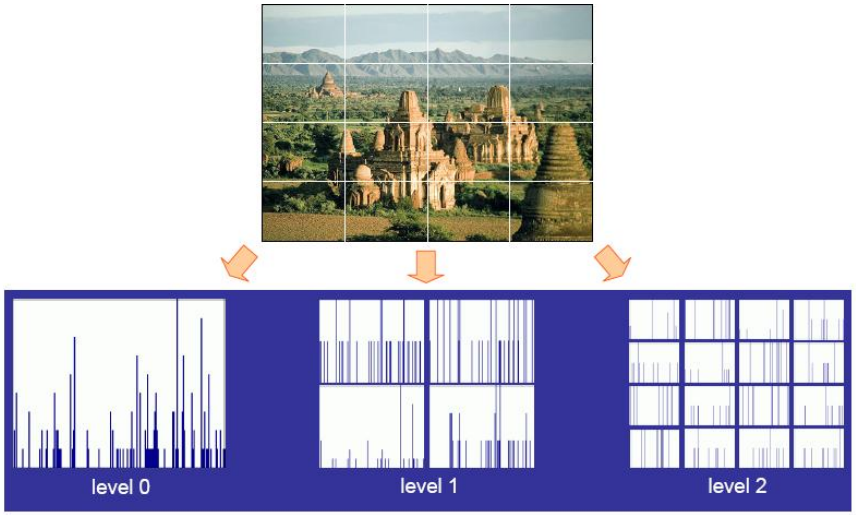




Confusion matrix

Spatial pyramid match

Captures scene categories well---texture-like patterns but with some variability in the positions of all the local pieces.



	Strong features (vocabulary size: 200)	
Level	Single-level	Pyramid
0 (1×1)	72.2 ±0.6	
1 (2×2)	77.9 ±0.6	79.0 ±0.5
2 (4×4)	79.4 ±0.3	**81.1** ±0.3
3 (8×8)	77.2 ±0.4	80.7 ±0.3

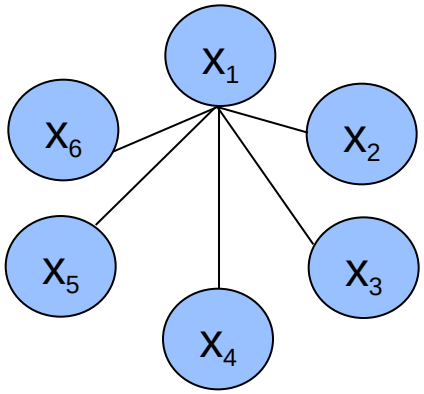
Part-based and local feature models for recognition

We'll look at three forms:

1. **Bag of words** (no geometry)
2. **Implicit shape model** (star graph for spatial model)
3. **Constellation model** (fully connected graph for spatial model)

Shape representation in part-based models

“Star” shape model

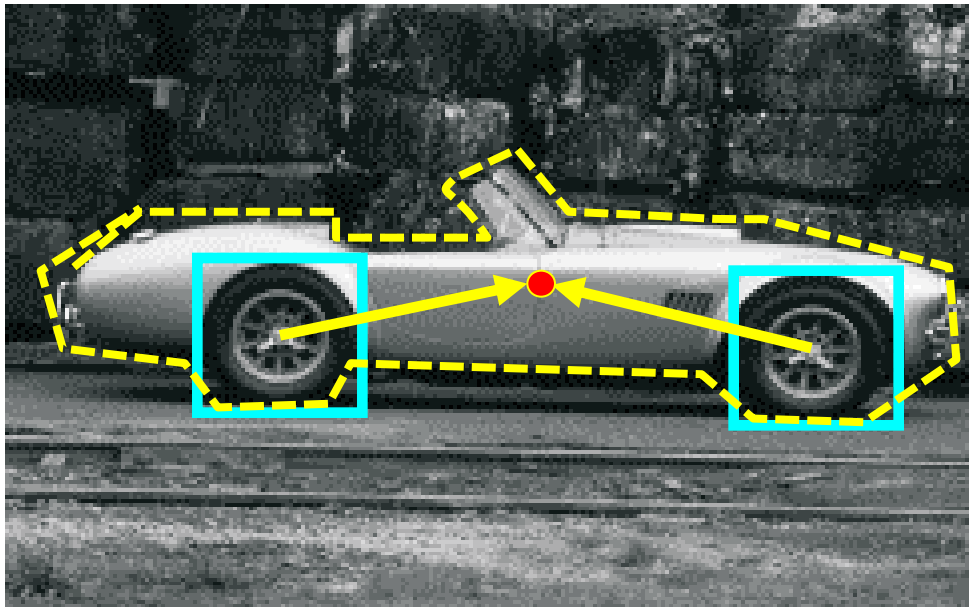


- **e.g. implicit shape model**
- **Parts mutually independent**

N image features, P parts in the model

Implicit shape models

- Visual vocabulary is used to index votes for object position [a visual word = “part”]



training image annotated with object localization info

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

Implicit shape models

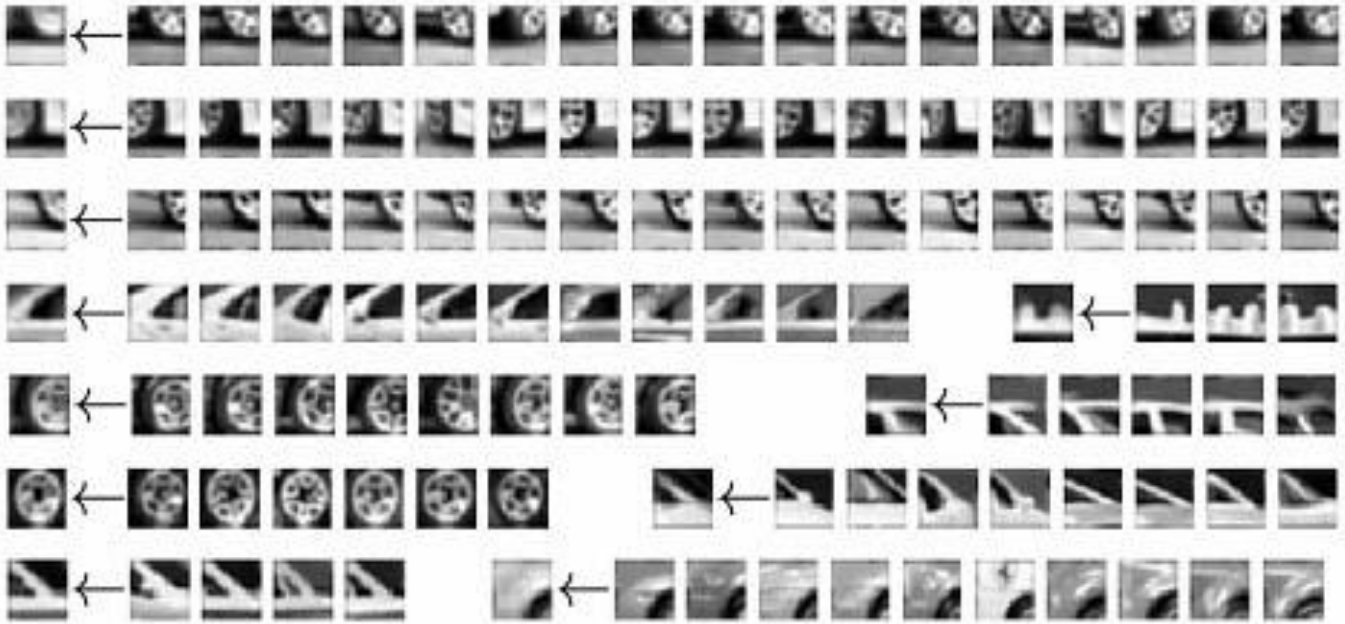
- Visual vocabulary is used to index votes for object position [a visual word = “part”]

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

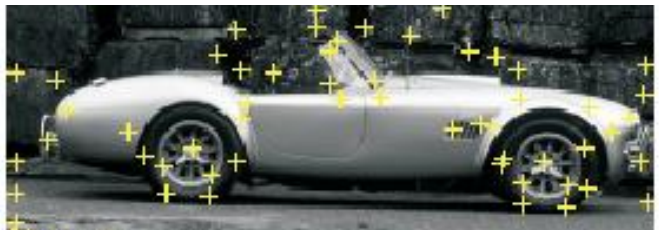
Implicit shape models: Training

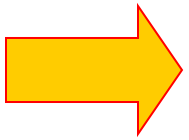
1. Build vocabulary of patches around extracted interest points using clustering

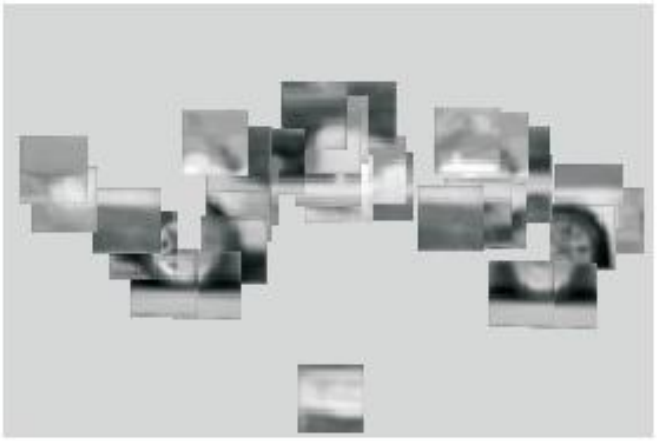


Implicit shape models: Training

1. Build vocabulary of patches around extracted interest points using clustering
2. Map the patch around each interest point to closest word

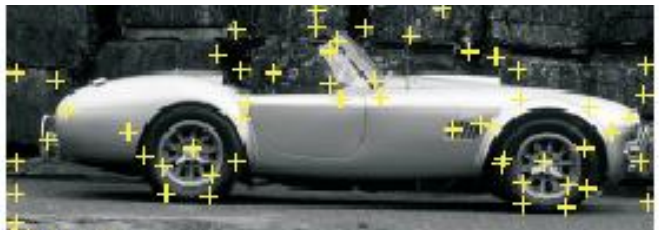


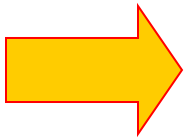


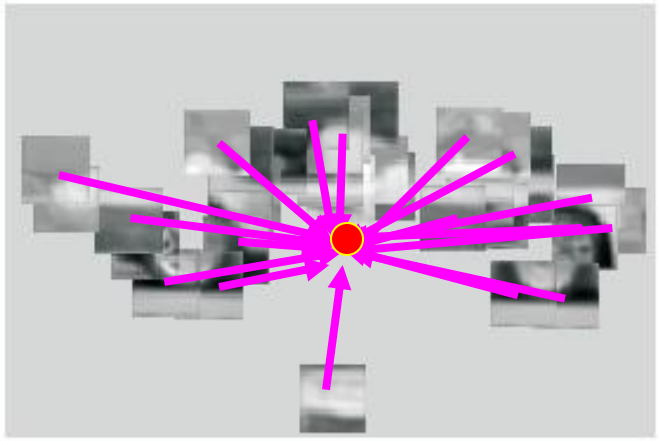


Implicit shape models: Training

1. Build vocabulary of patches around extracted interest points using clustering
2. Map the patch around each interest point to closest word
3. For each word, store all positions it was found, relative to object center







Implicit shape models: Testing

1. Given new test image, extract patches, match to vocabulary words
2. Cast votes for possible positions of object center
3. Search for maxima in voting space
4. (Extract weighted segmentation mask based on stored masks for the codebook occurrences)

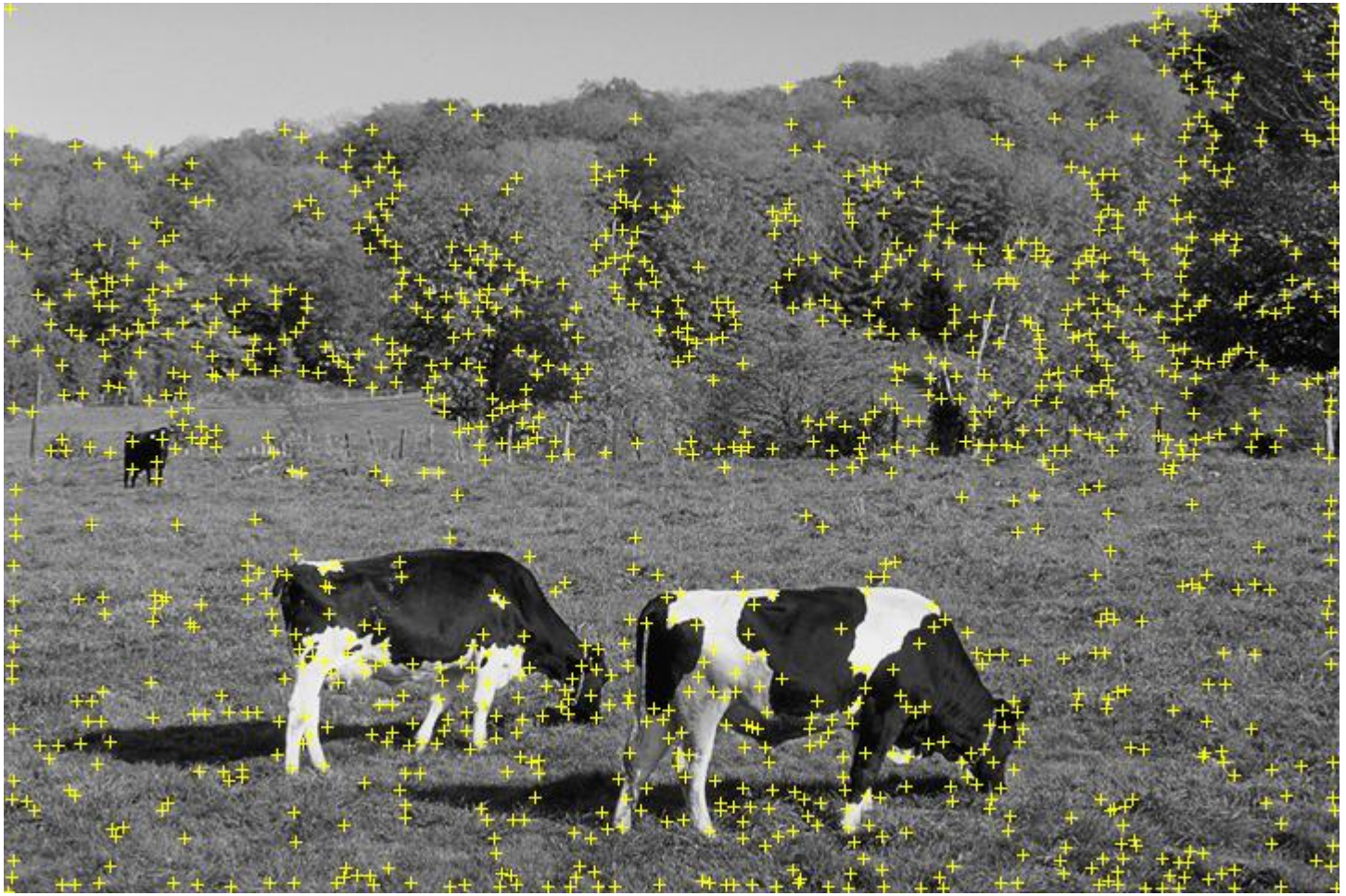
Implicit shape models: Testing

Original Image

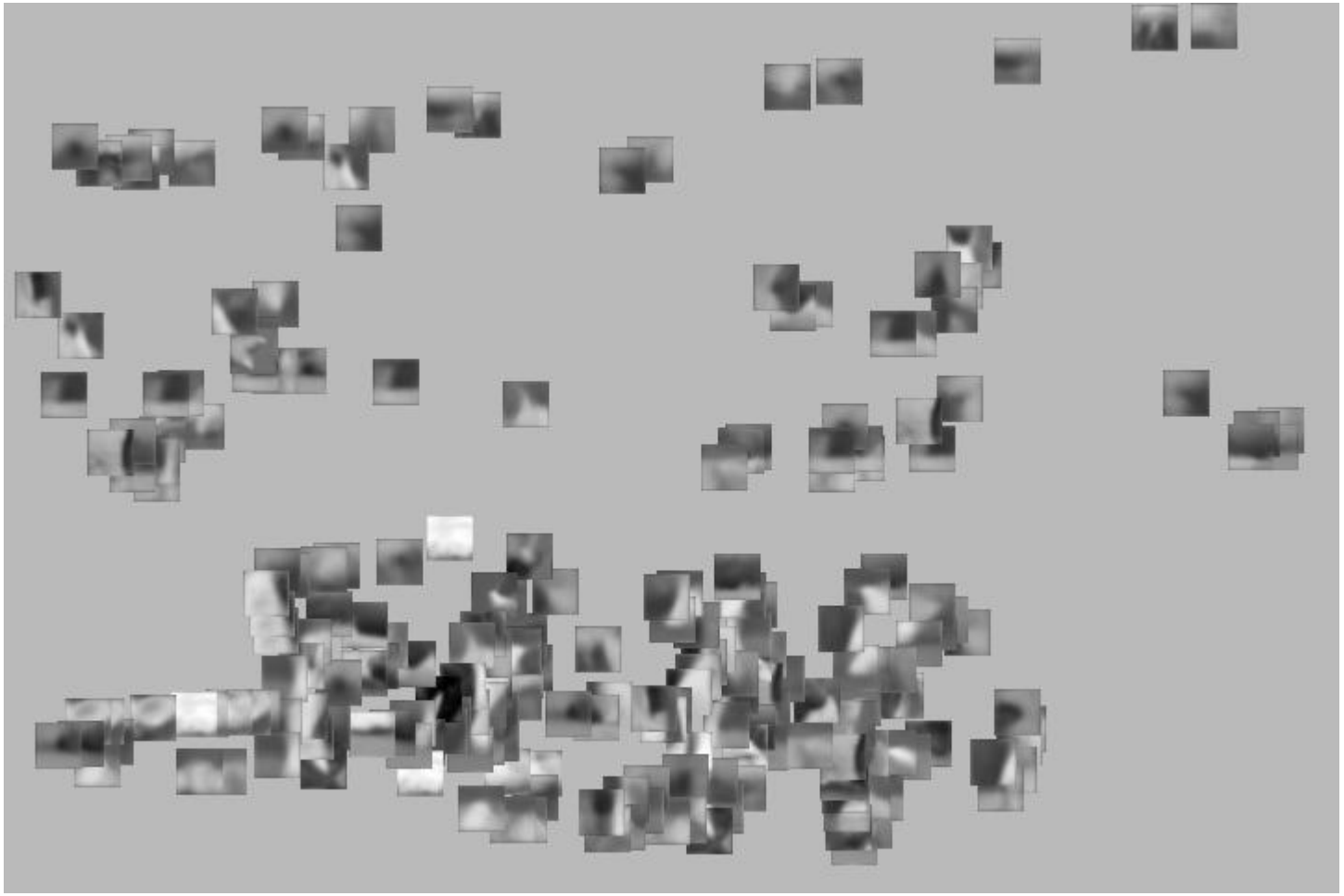
Interest Points

Example: Results on Cows

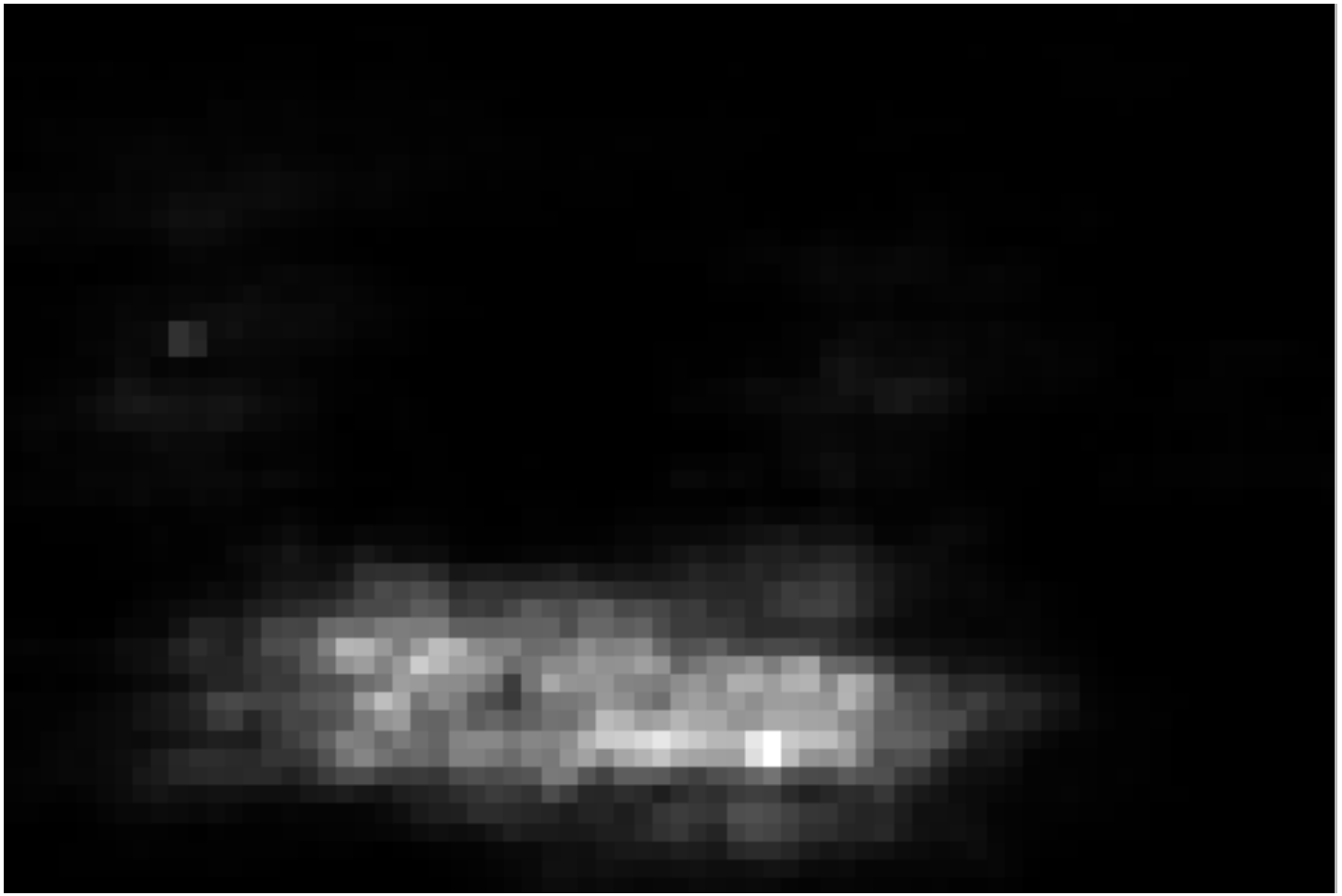
Example: Results on Cows



Example: Results on Cows



Example: Results on Cows



Example: Results on Cows

Example: Results on Cows

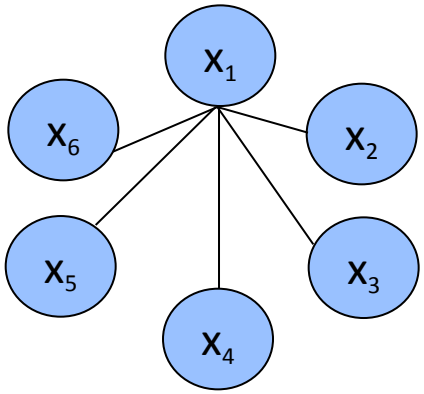
Example: Results on Cows

Detection Results

- Qualitative Performance
  - Recognizes different kinds of objects
  - Robust to clutter, occlusion, noise, low contrast

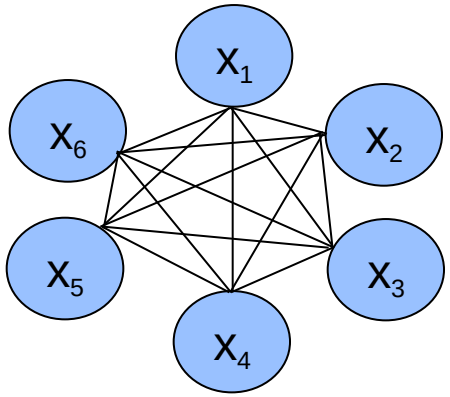
Shape representation in part-based models

"Star" shape model



- **e.g. implicit shape model**
- **Parts mutually independent**

Fully connected constellation model



- **e.g. Constellation Model**
- **Parts fully connected**

N image features, P parts in the model

Slide credit: Rob Fergus

Probabilistic constellation model

$$P(image \mid object) = P(appearance, shape \mid object)$$

Part descriptors

Part locations

Candidate parts

Source: Lana Lazebnik

Probabilistic constellation model

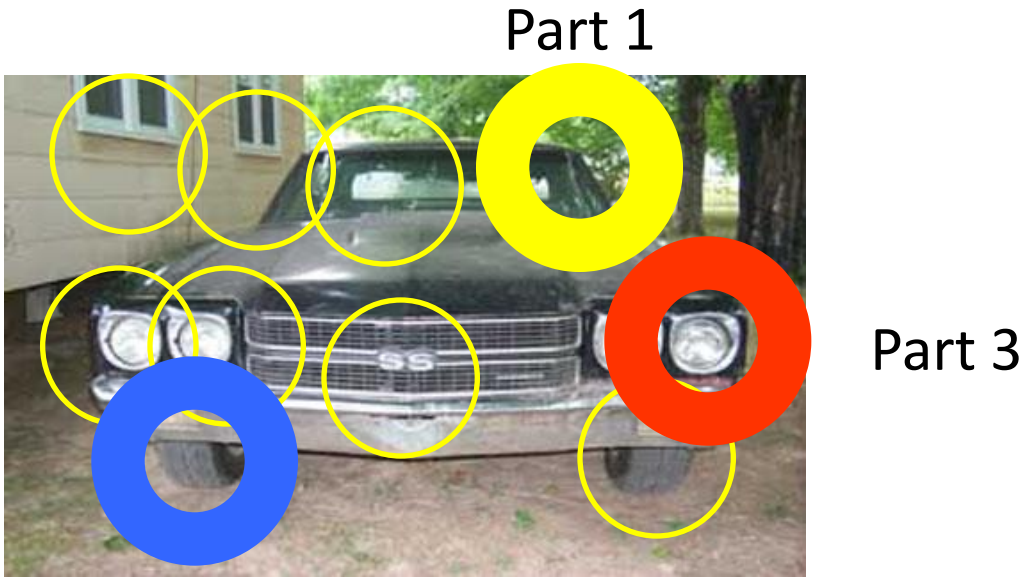
$$P(image \mid object) = P(appearance, shape \mid object)$$

Source: Lana Lazebnik

Probabilistic constellation model

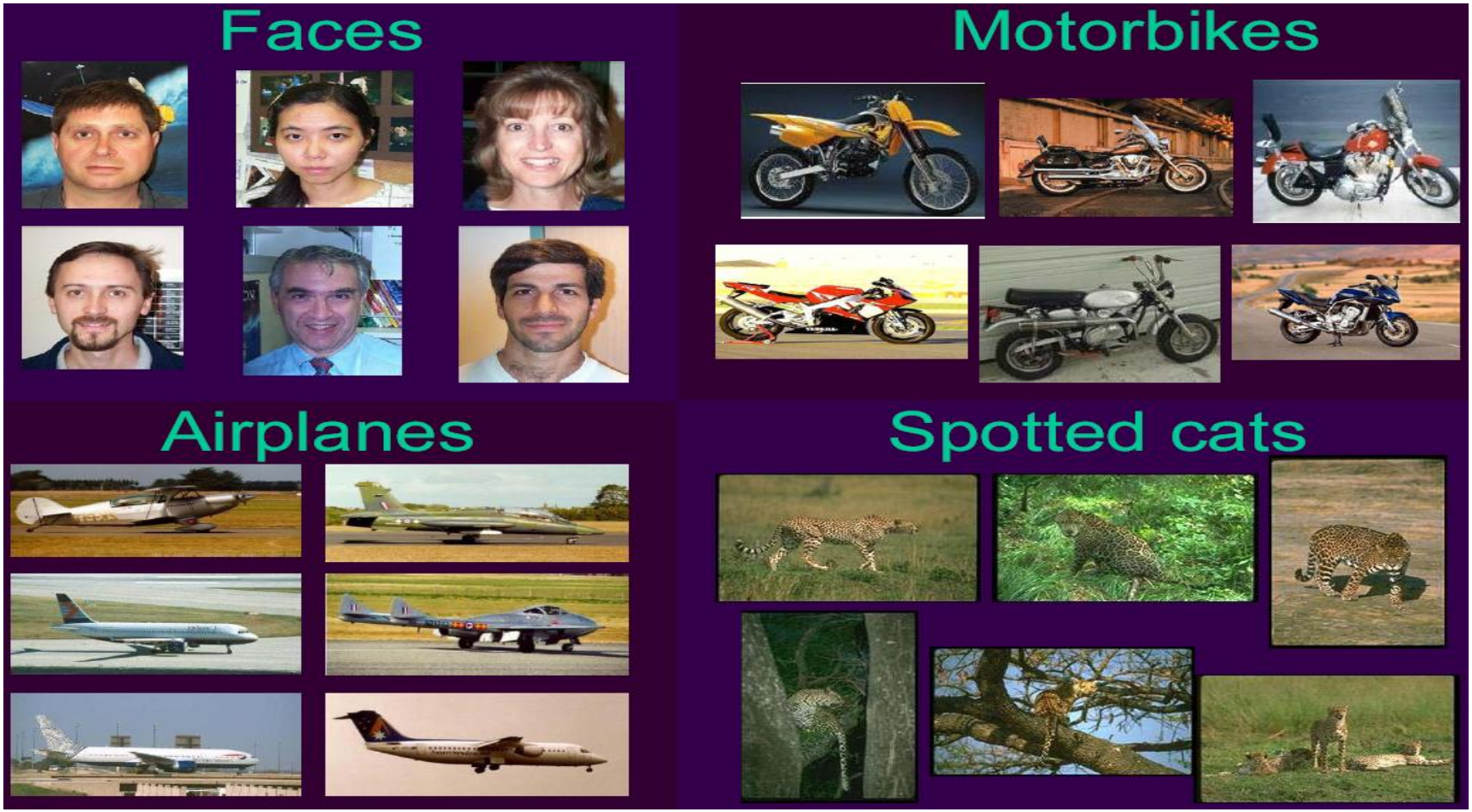
$$P(image \mid object) = P(appearance, shape \mid object)$$
$$= \max_h P(appearance \mid h, object)\, p(shape \mid h, object)\, p(h \mid object)$$

h: assignment of features to parts



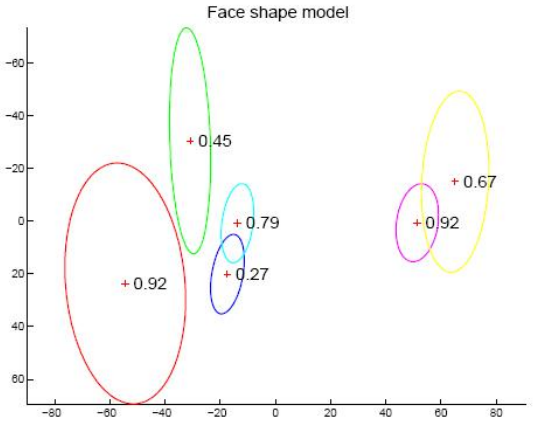
Source: Lana Lazebnik

Example results from constellation model: data from four categories



Slide from Li Fei-Fei http://www.vision.caltech.edu/feifeili/Resume.htm

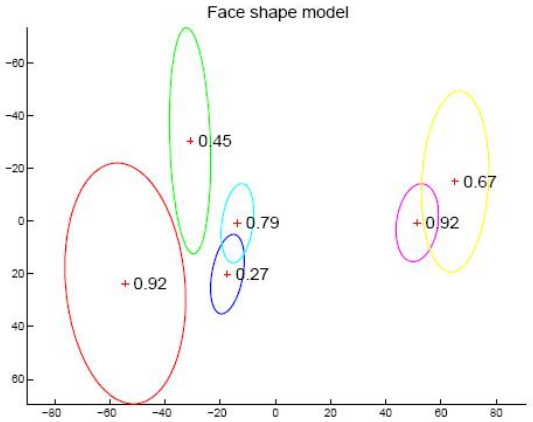
Face model

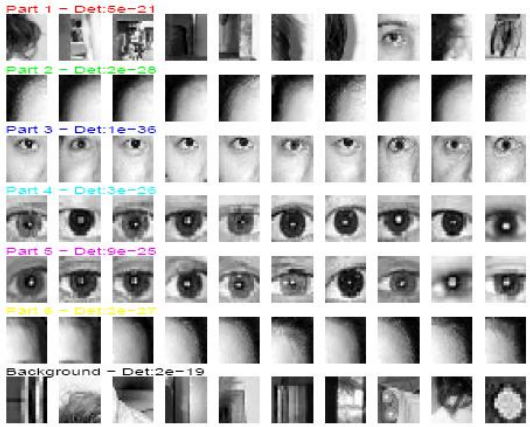


Appearance: 10 patches closest to mean for each part

Fergus et al. CVPR 2003

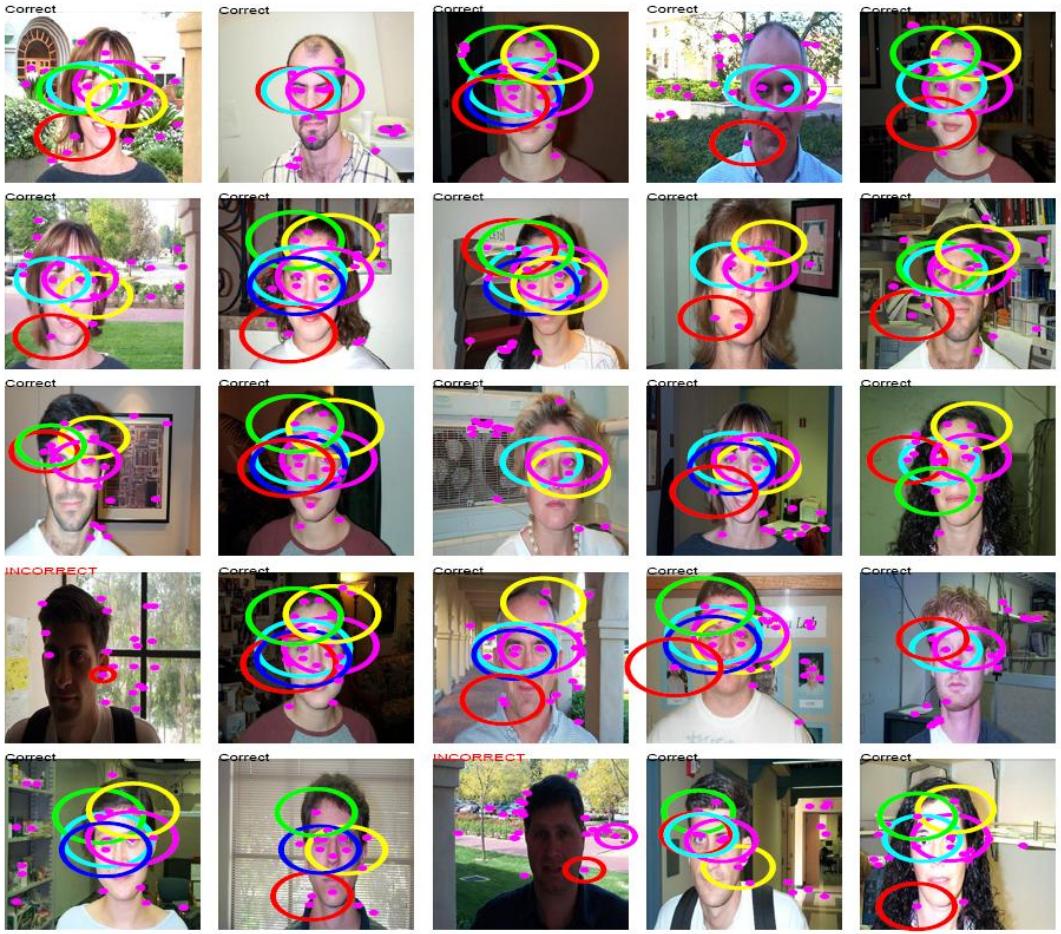
Face model





Appearance: 10 patches closest to mean for each part

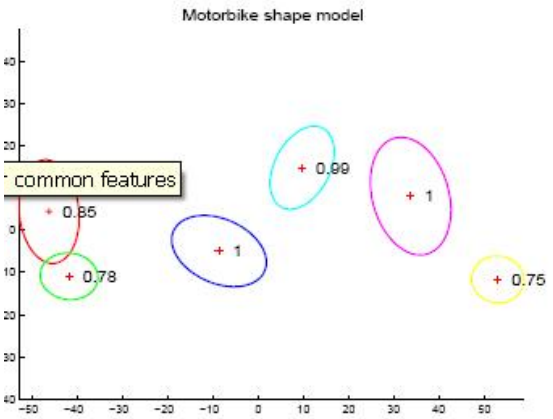
Recognition results



Test images: size of circles indicates score of hypothesis

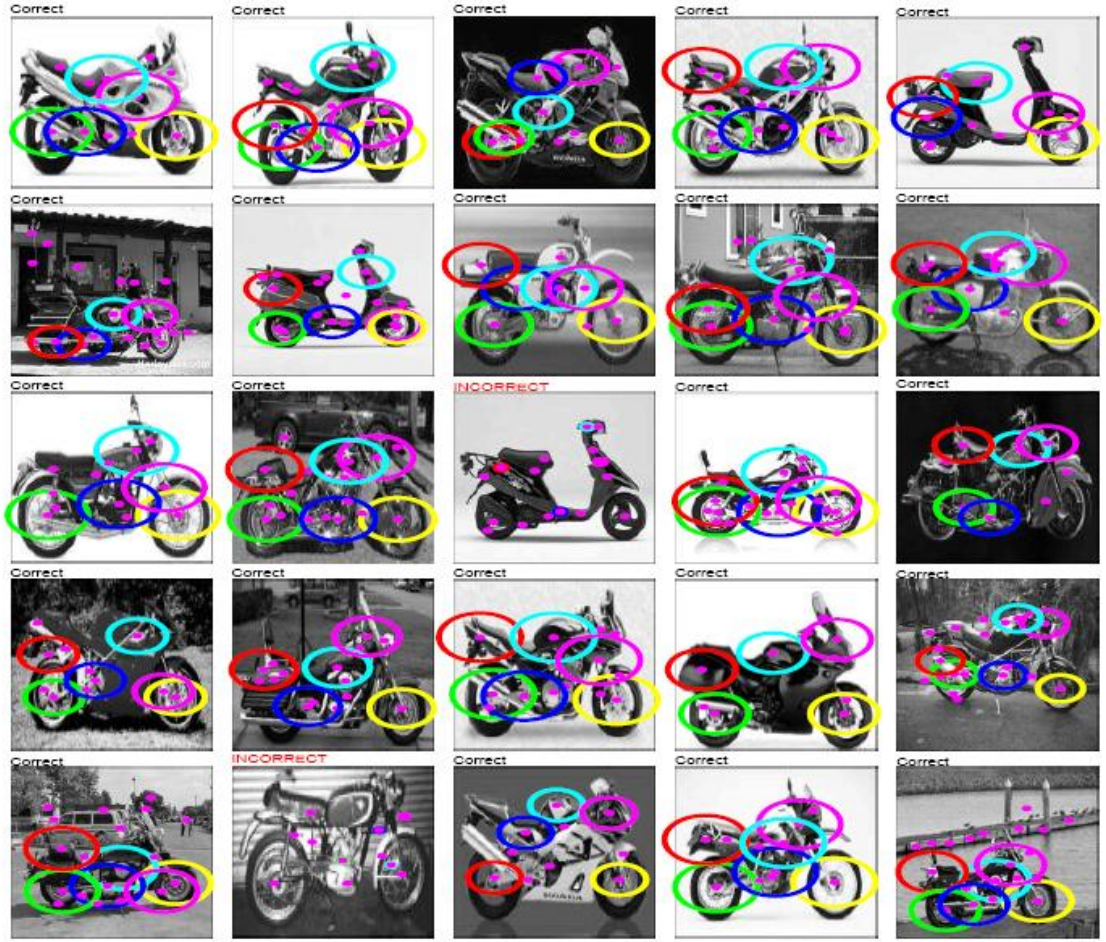
Fergus et al. CVPR 2003

Motorbike model



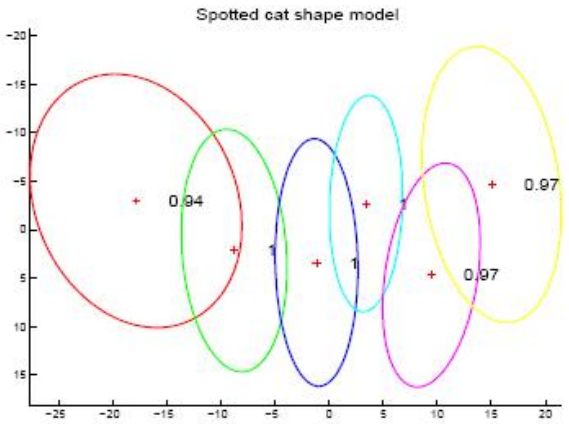
Appearance: 10 patches closest to mean for each part

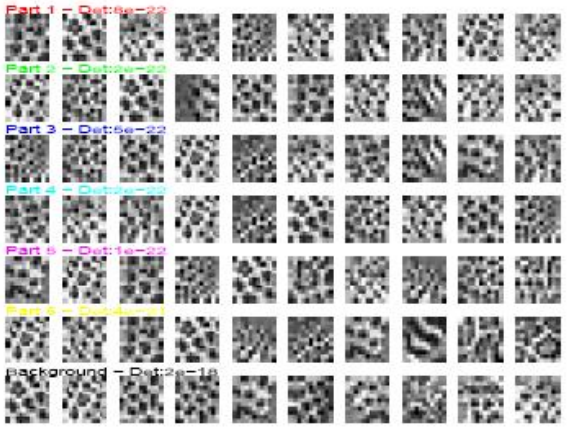
Recognition results



Fergus et al. CVPR 2003

Spotted cat model





Appearance: 10 patches closest to mean for each part

Recognition results

Kristen Grauman

Fergus et al. CVPR 2003

Comparison



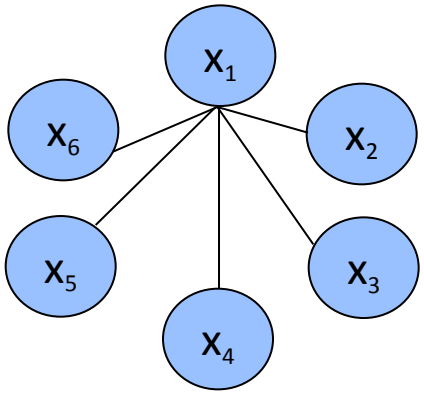



class	bag of features Zhang et al. (2005)	bag of features Willamowski et al. (2004)	Part-based model Fergus et al. (2003)
airplanes	**98.8**	97.1	90.2
cars (rear)	98.3	**98.6**	90.3
cars (side)	**95.0**	87.3	88.5
faces	**100**	99.3	96.4
motorbikes	**98.5**	98.0	92.5
spotted cats	**97.0**	—	90.0

Source: Lana Lazebnik

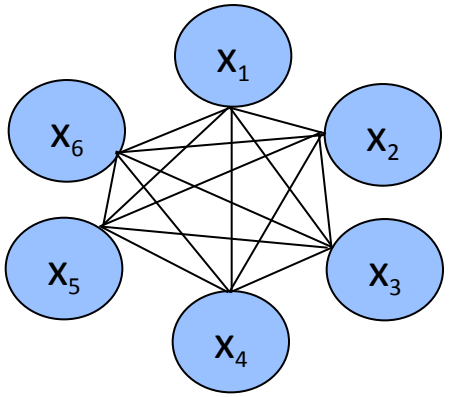
Shape representation in part-based models

"Star" shape model



- **e.g. implicit shape model**
- **Parts mutually independent**
- **Recognition complexity: O(NP)**
- **Method: Gen. Hough Transform**

Fully connected constellation model



- **e.g. Constellation Model**
- **Parts fully connected**
- **Recognition complexity: $O(N^P)$**
- **Method: Exhaustive search**

N image features, P parts in the model

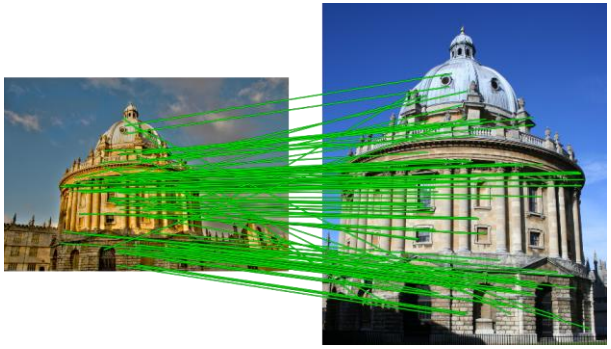
Slide credit: Rob Fergus

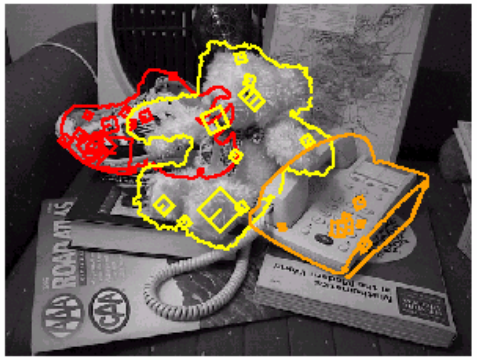
Summary:
part-based and local feature models for generic object recognition

- **Histograms of visual words** to capture global or local layout in the bag-of-words framework
  - Pyramid match kernels
  - Powerful in practice for image recognition

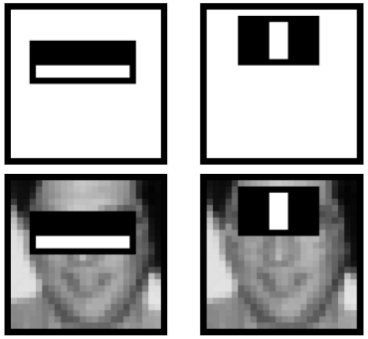
- **Part-based models** encode category's part appearance together with 2d layout and allow detection within cluttered image
  - **"implicit shape model"**: shape based on layout of all parts relative to a reference part; Generalized Hough for detection
  - **"constellation model"**: explicitly model mutual spatial layout between all pairs of parts; exhaustive search for best fit of features to parts

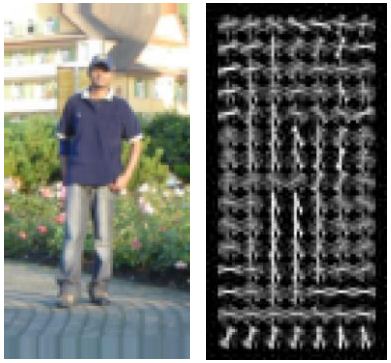
Recognition models



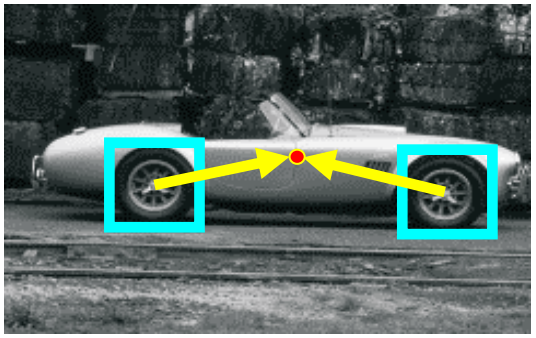


Instances: recognition by alignment





Categories: Holistic appearance models (and sliding window detection)



Categories: Local feature and part-based models