

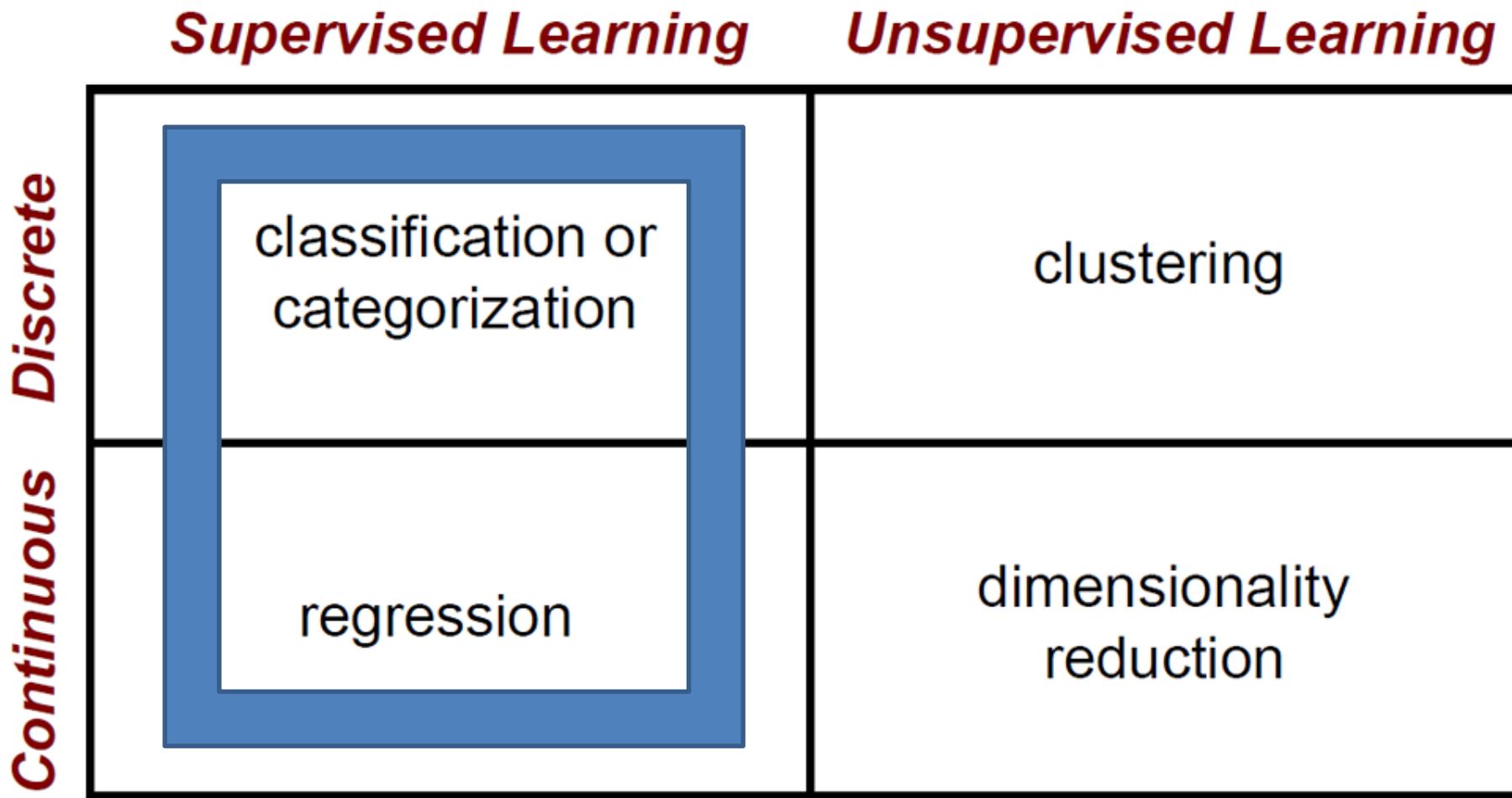
Machine Learning Crash Course



Computer Vision
James Hays

Slides: Isabelle Guyon,
Erik Sudderth,
Mark Johnson,
Derek Hoiem

Machine Learning Problems



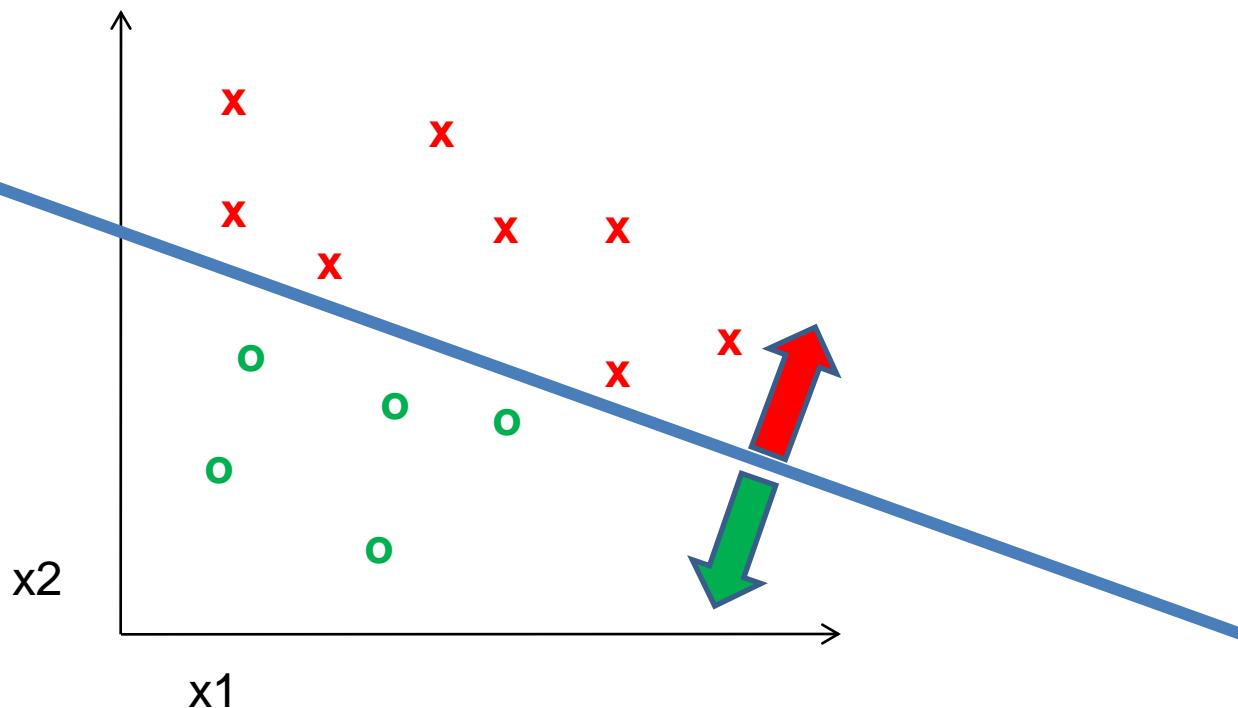
The machine learning framework

- Apply a prediction function to a feature representation of the image to get the desired output:

$$f(\text{apple}) = \text{"apple"}$$
$$f(\text{tomato}) = \text{"tomato"}$$
$$f(\text{cow}) = \text{"cow"}$$

Learning a classifier

Given some set of features with corresponding labels, learn a function to predict the labels from the features



Generalization



Training set (labels known)



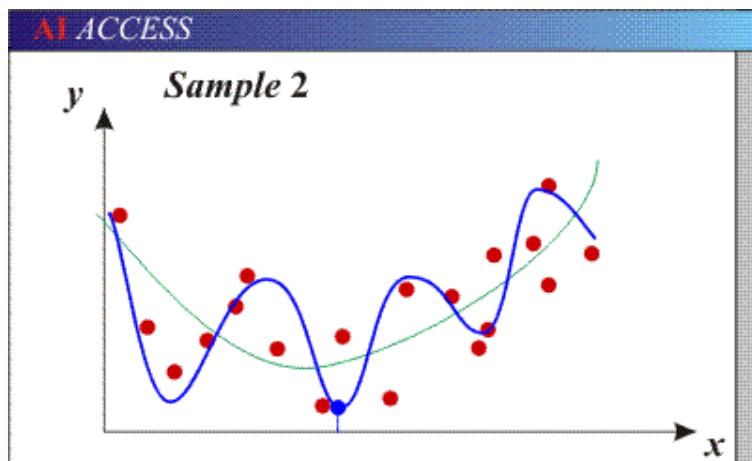
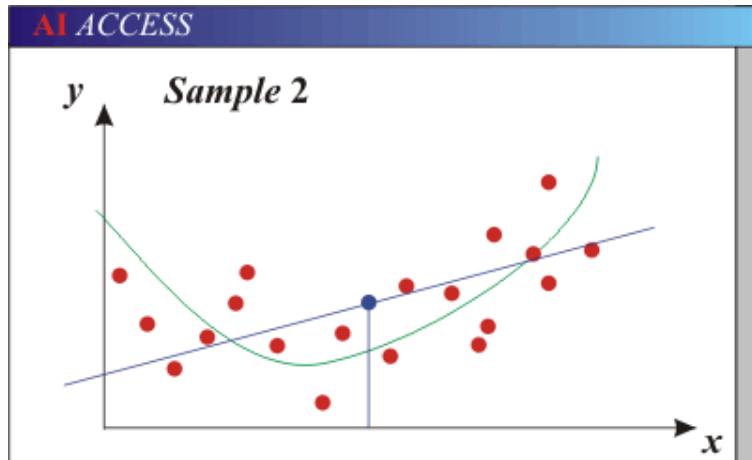
Test set (labels unknown)

- How well does a learned model generalize from the data it was trained on to a new test set?

Generalization

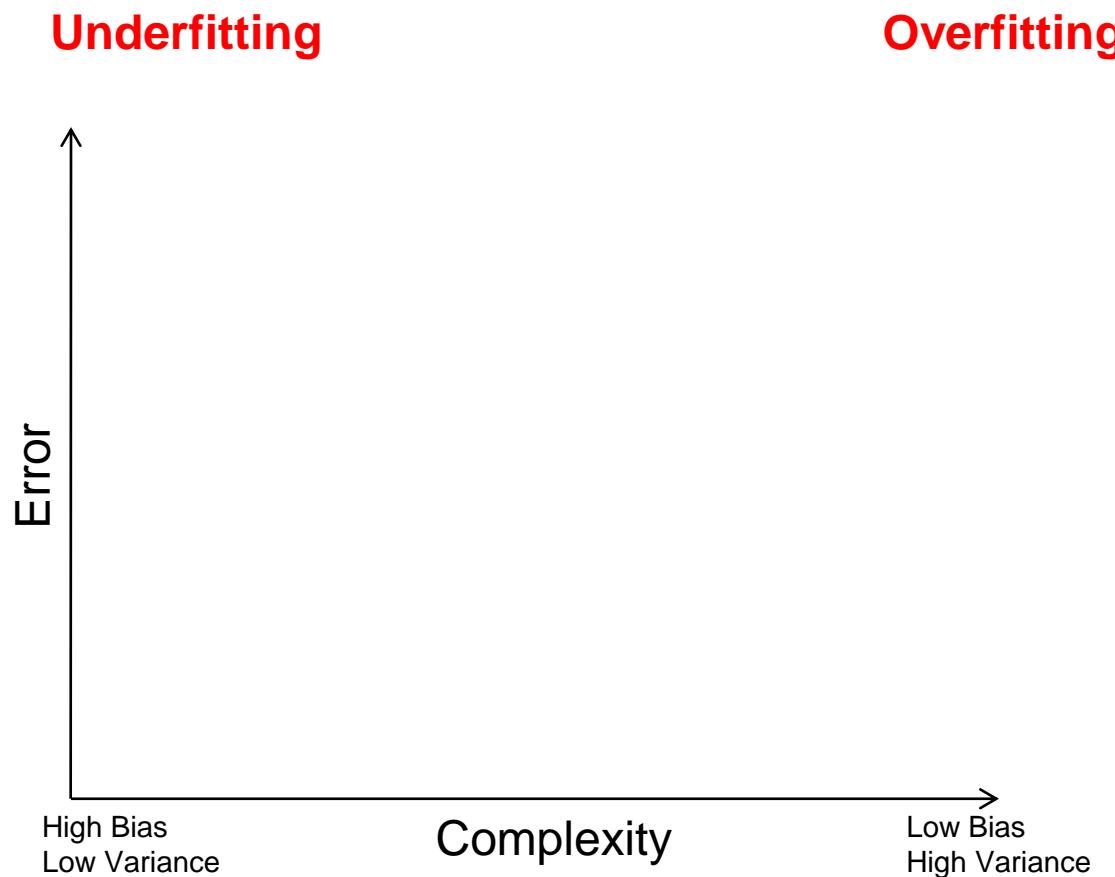
- Components of generalization error
 - **Bias:** how much the average model over all training sets differ from the true model?
 - Error due to inaccurate assumptions/simplifications made by the model.
 - **Variance:** how much models estimated from different training sets differ from each other.
- **Underfitting:** model is too “simple” to represent all the relevant class characteristics
 - High bias (few degrees of freedom) and low variance
 - High training error and high test error
- **Overfitting:** model is too “complex” and fits irrelevant characteristics (noise) in the data
 - Low bias (many degrees of freedom) and high variance
 - Low training error and high test error

Bias-Variance Trade-off

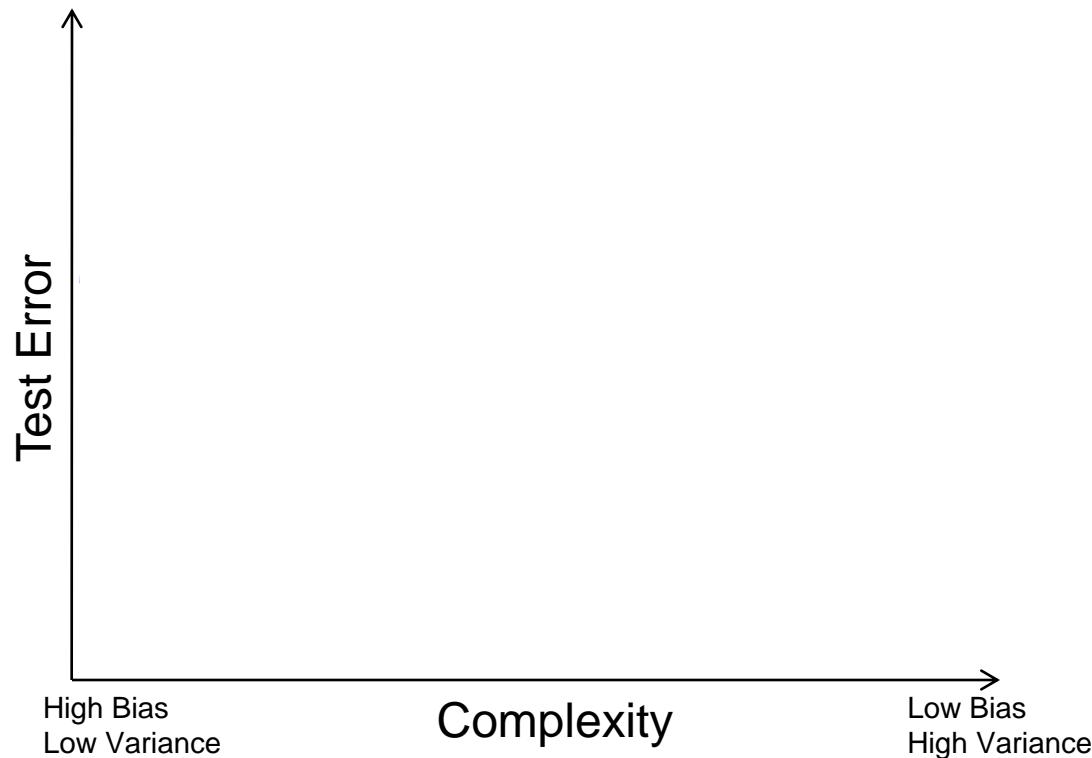


- Models with too few parameters are inaccurate because of a large bias (not enough flexibility).
- Models with too many parameters are inaccurate because of a large variance (too much sensitivity to the sample).

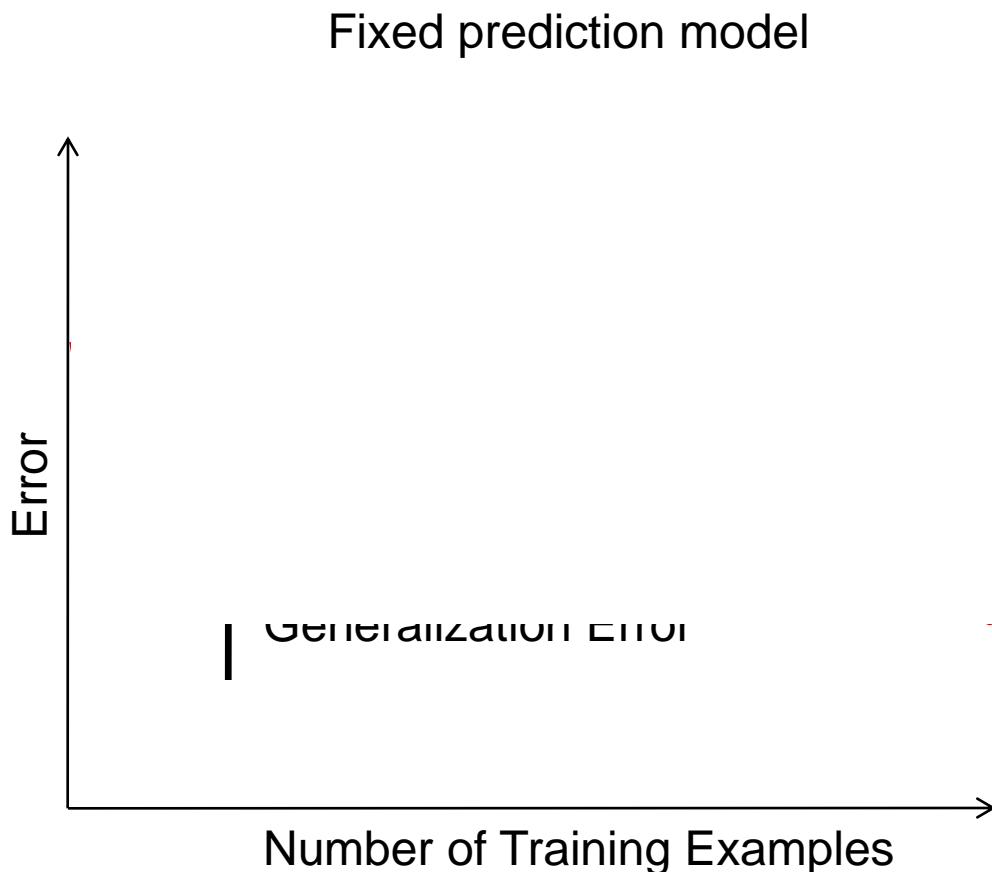
Bias-variance tradeoff



Bias-variance tradeoff



Effect of Training Size



Remember...

- No classifier is inherently better than any other: you need to make assumptions to generalize
- Three kinds of error
 - Inherent: unavoidable
 - Bias: due to over-simplifications
 - Variance: due to inability to perfectly estimate parameters from limited data



How to reduce variance?

- Choose a simpler classifier
- Regularize the parameters
- Get more training data

Very brief tour of some classifiers

- K-nearest neighbor
- SVM
- Boosted Decision Trees
- Neural networks
- Naïve Bayes
- Bayesian network
- Logistic regression
- Randomized Forests
- RBMs
- Etc.

Generative vs. Discriminative Classifiers

Generative Models

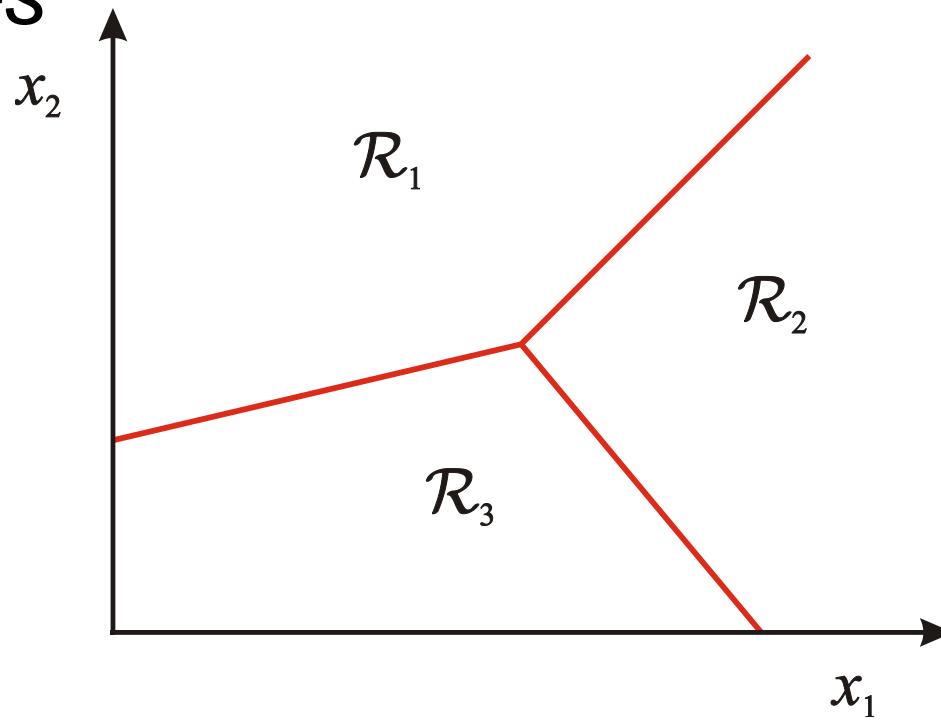
- Represent both the data and the labels
- Often, makes use of conditional independence and priors
- Examples
 - Naïve Bayes classifier
 - Bayesian network
- Models of data may apply to future prediction problems

Discriminative Models

- Learn to directly predict the labels from the data
- Often, assume a simple boundary (e.g., linear)
- Examples
 - Logistic regression
 - SVM
 - Boosted decision trees
- Often easier to predict a label from the data than to model the data

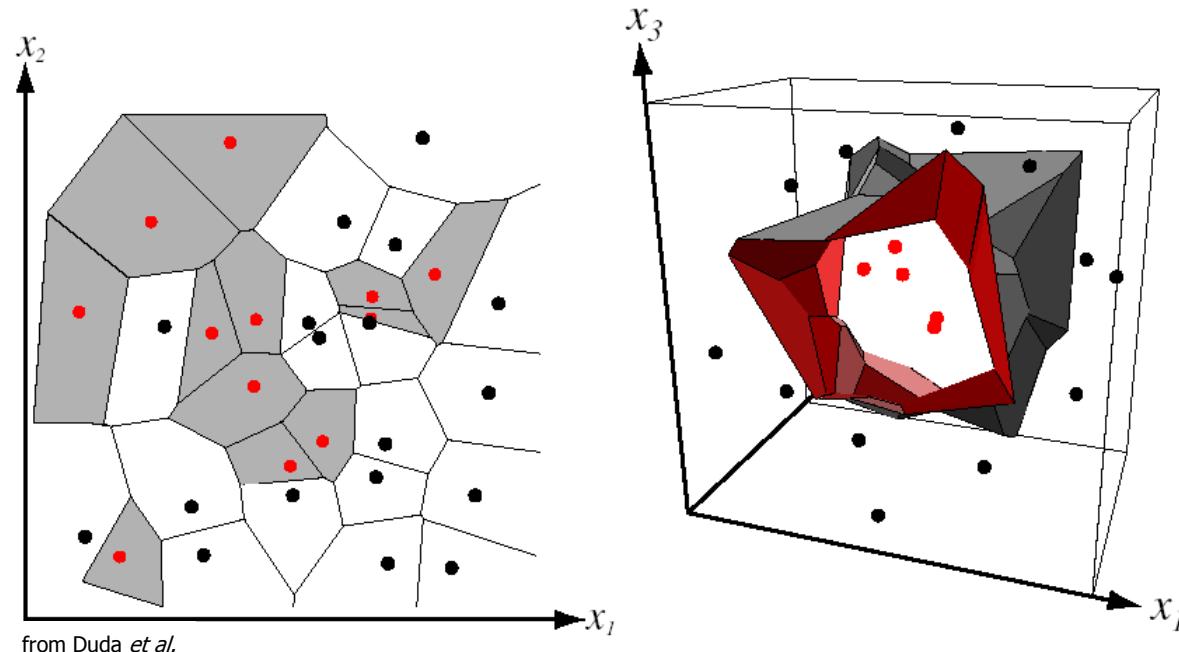
Classification

- Assign input vector to one of two or more classes
- Any decision rule divides input space into *decision regions* separated by *decision boundaries*



Nearest Neighbor Classifier

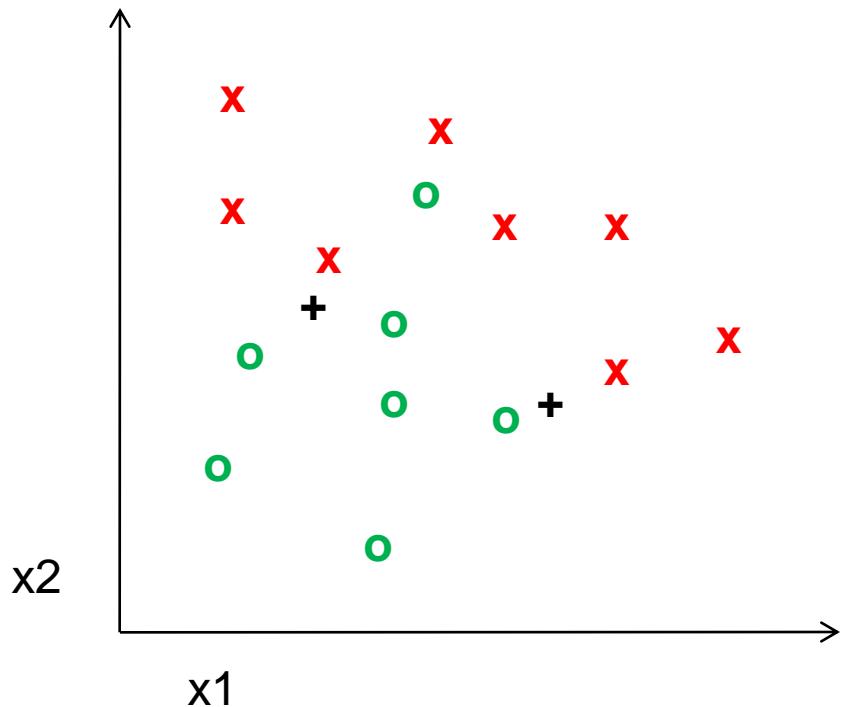
- Assign label of nearest training data point to each test data point



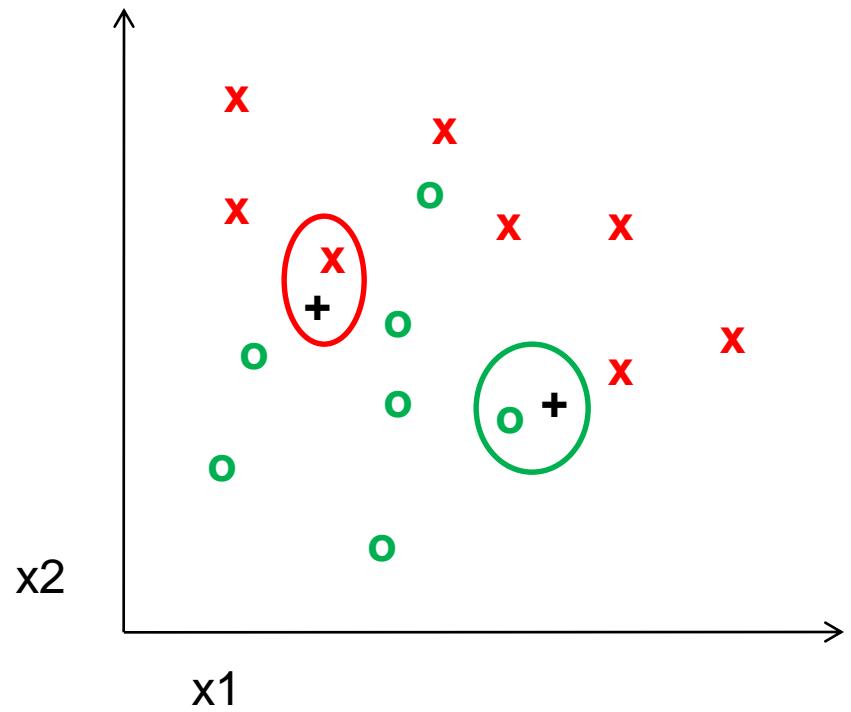
Voronoi partitioning of feature space
for two-category 2D and 3D data

Source: D. Lowe

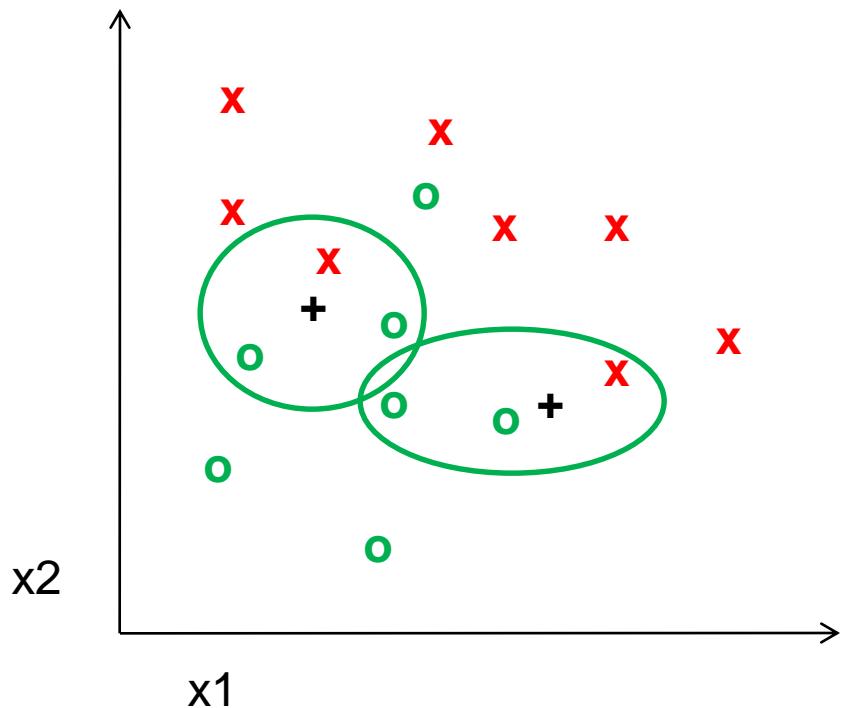
K-nearest neighbor



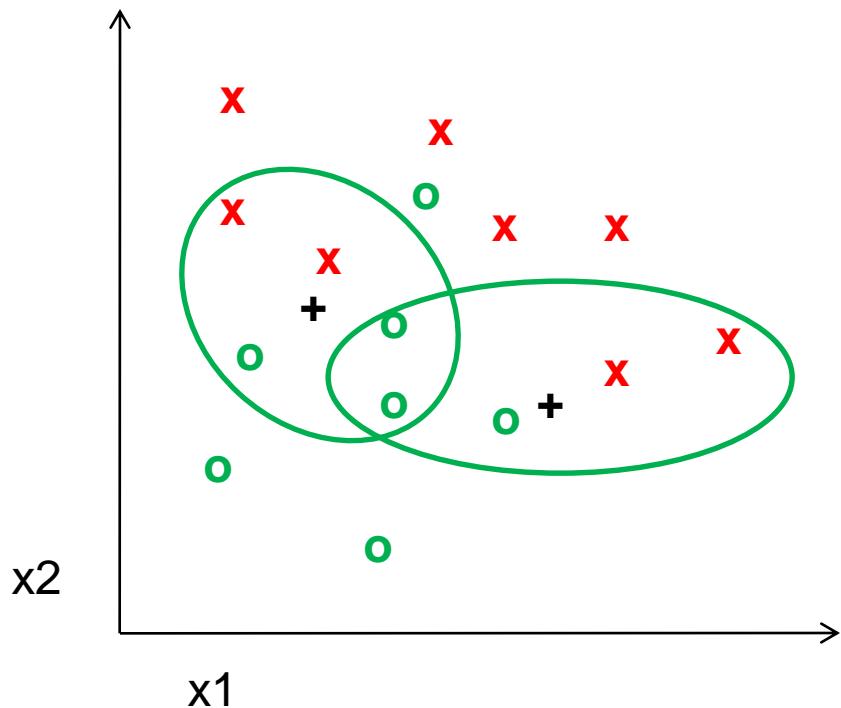
1-nearest neighbor



3-nearest neighbor



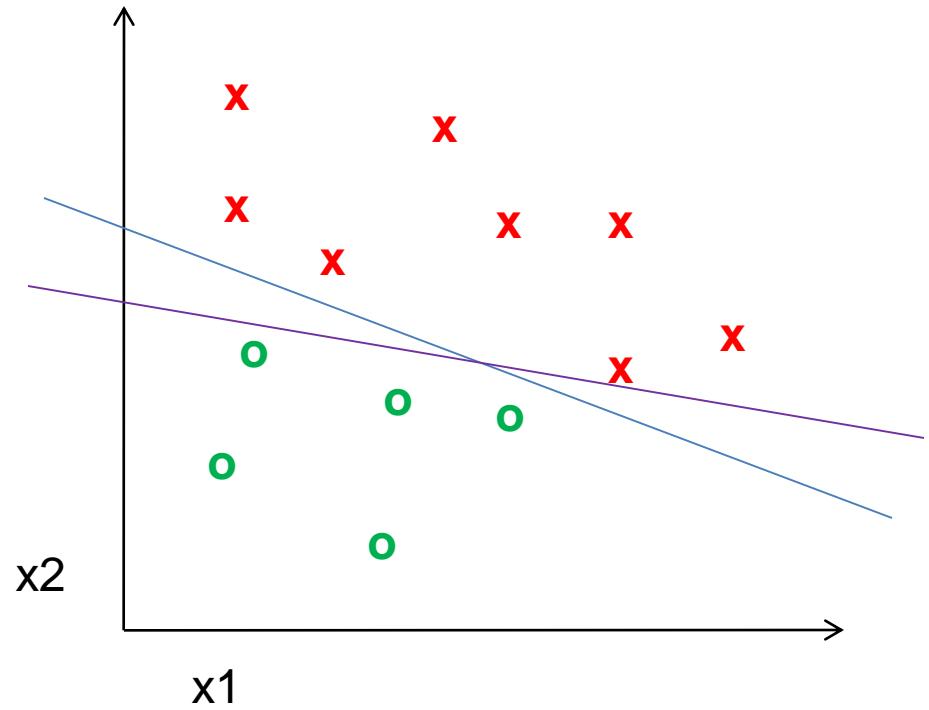
5-nearest neighbor



Using K-NN

- Simple, a good one to try first
- With infinite examples, 1-NN provably has error that is at most twice Bayes optimal error

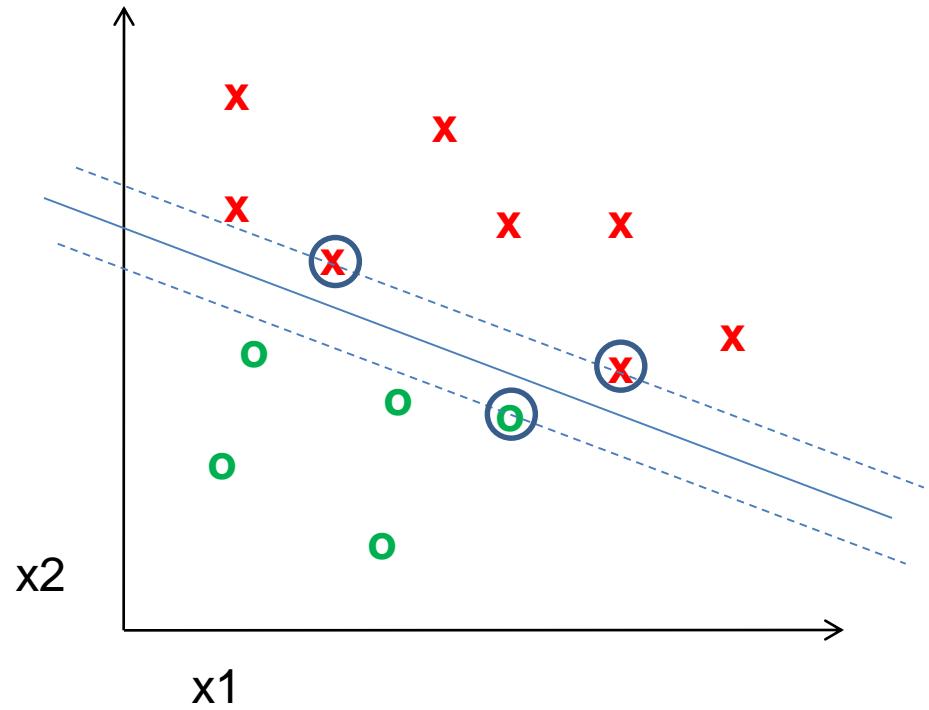
Classifiers: Linear SVM



- Find a *linear function* to separate the classes:

$$f(\mathbf{x}) = \text{sgn}(\mathbf{w} \cdot \mathbf{x} + b)$$

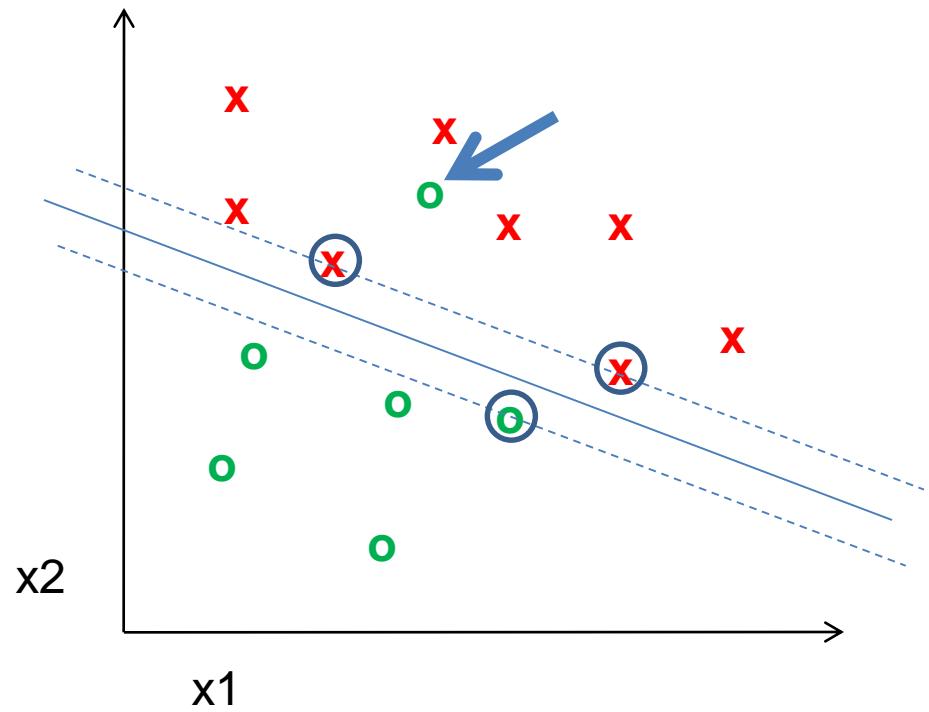
Classifiers: Linear SVM



- Find a *linear function* to separate the classes:

$$f(\mathbf{x}) = \text{sgn}(\mathbf{w} \cdot \mathbf{x} + b)$$

Classifiers: Linear SVM



- Find a *linear function* to separate the classes:

$$f(\mathbf{x}) = \text{sgn}(\mathbf{w} \cdot \mathbf{x} + b)$$

What about multi-class SVMs?

- Unfortunately, there is no “definitive” multi-class SVM formulation
- In practice, we have to obtain a multi-class SVM by combining multiple two-class SVMs
- One vs. others
 - Training: learn an SVM for each class vs. the others
 - 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

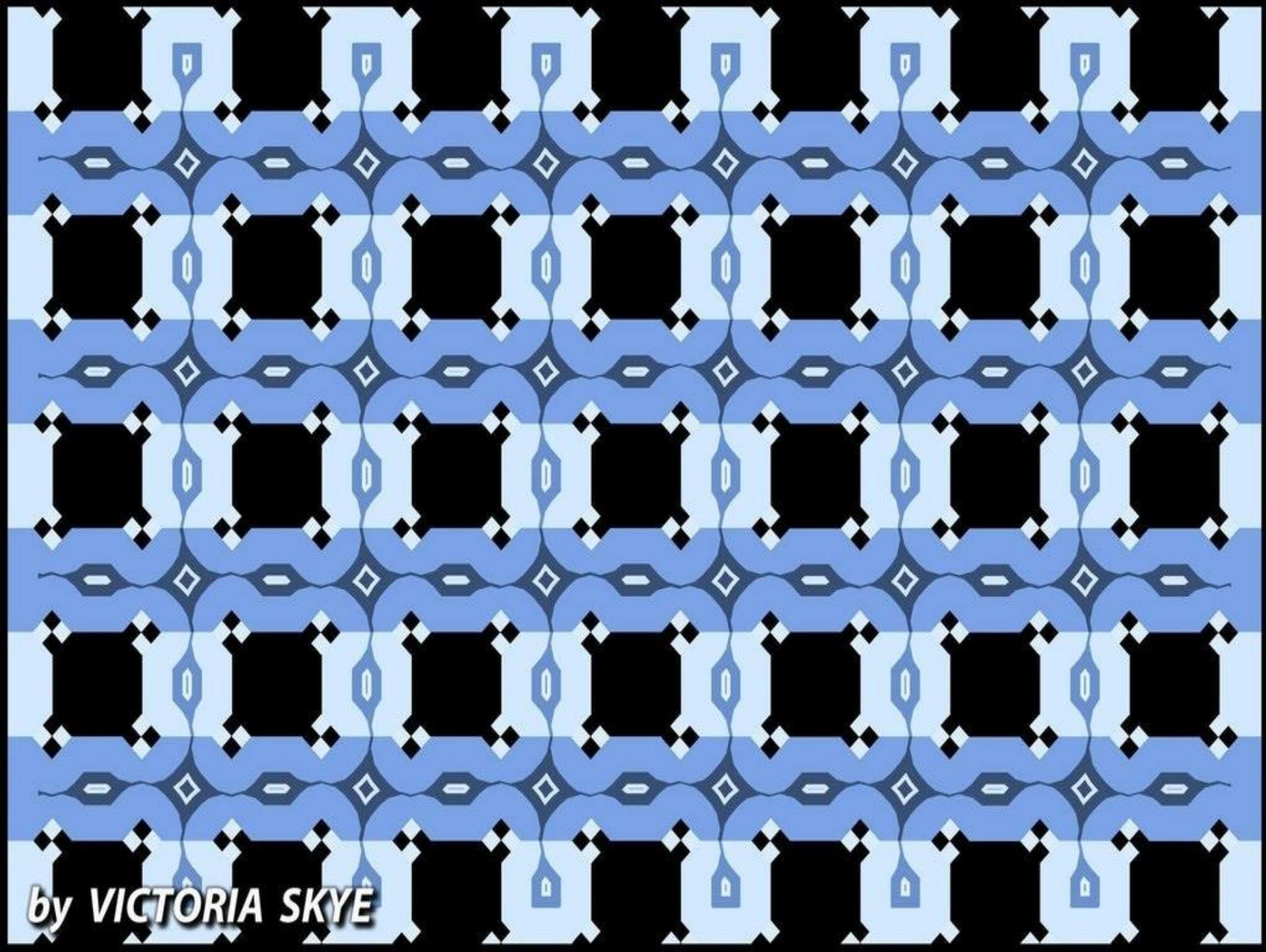
- Pros
 - Many publicly available SVM packages:
<http://www.kernel-machines.org/software>
 - Kernel-based framework is very powerful, flexible
 - SVMs work very well in practice, even with very small training sample sizes
- Cons
 - No “direct” multi-class SVM, must combine two-class SVMs
 - Computation, 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

What to remember about classifiers

- No free lunch: machine learning algorithms are tools, not dogmas
- Try simple classifiers first
- Better to have smart features and simple classifiers than simple features and smart classifiers
- Use increasingly powerful classifiers with more training data (bias-variance tradeoff)

Machine Learning Considerations

- 3 important design decisions:
 - 1) What data do I use?
 - 2) How do I represent my data (what feature)?
 - 3) What classifier / regressor / machine learning tool do I use?
- These are in decreasing order of importance
- Deep learning addresses 2 and 3 simultaneously (and blurs the boundary between them).
- You can take the representation from deep learning and use it with any classifier.

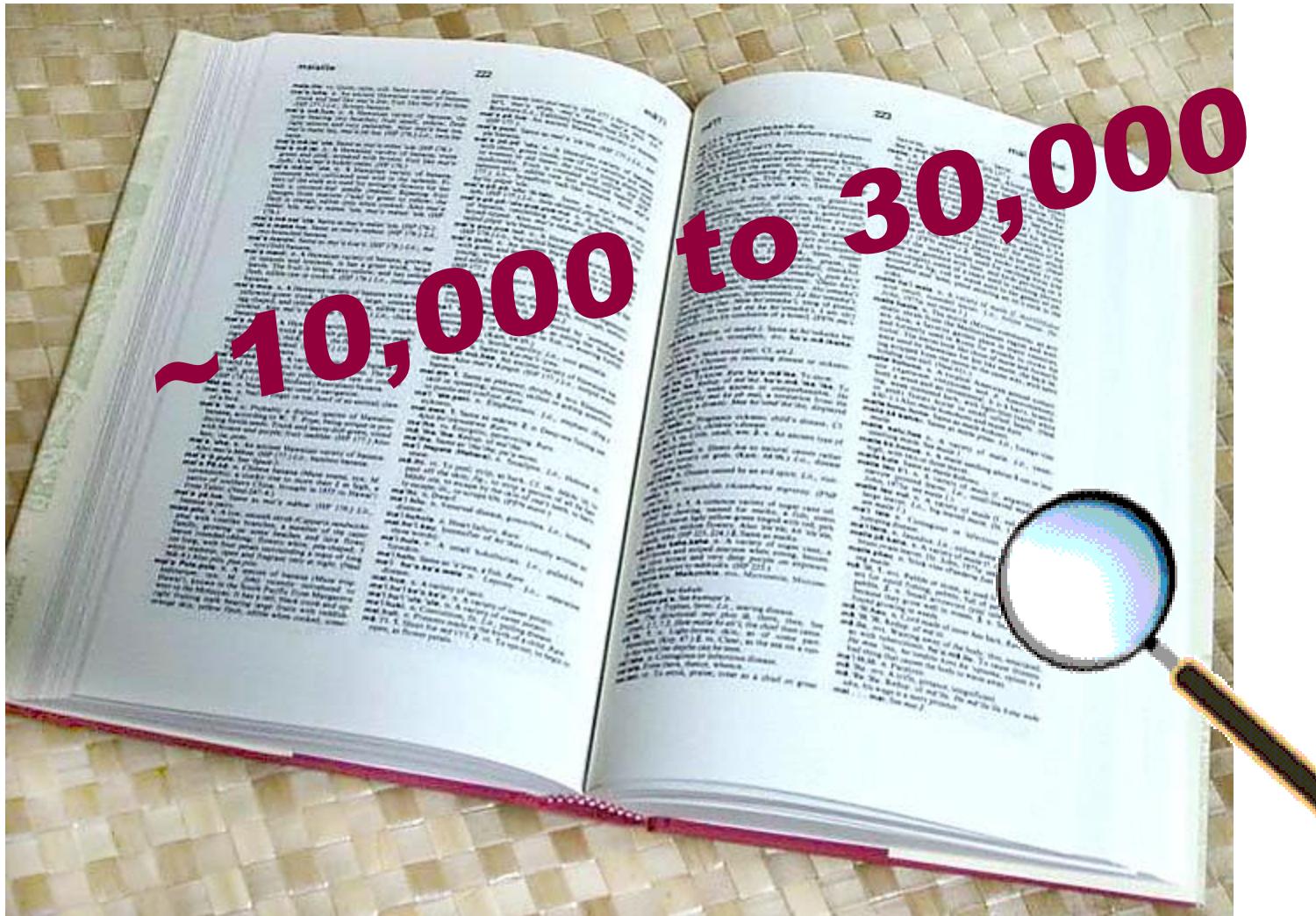


by VICTORIA SKYE

Recognition: Overview and History



How many visual object categories are there?



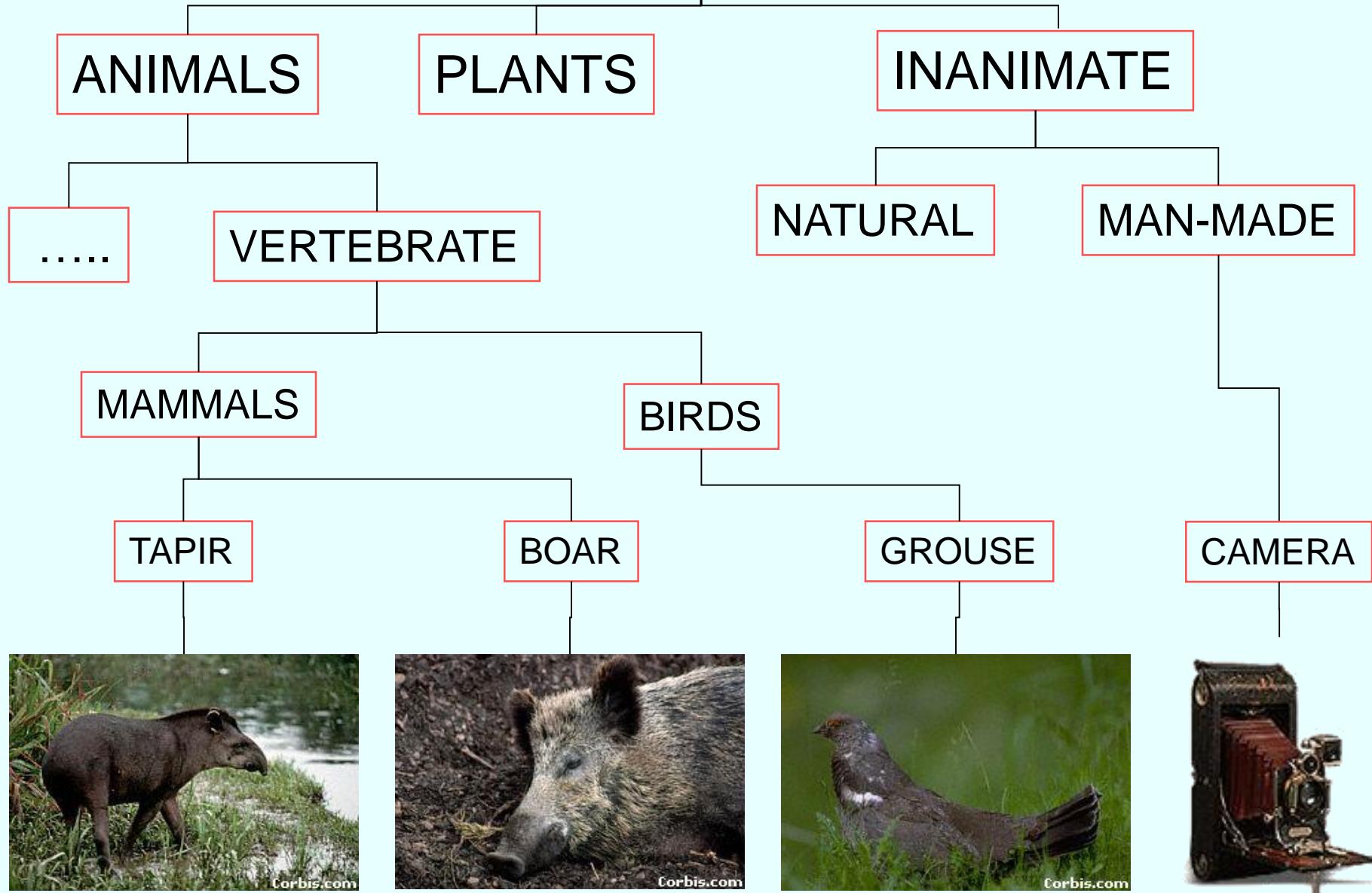
Biederman 1987



~10,000 to 30,000



OBJECTS



Specific recognition tasks



Scene categorization or classification



- outdoor/indoor
- city/forest/factory/etc.

Image annotation / tagging / attributes



- street
- people
- building
- mountain
- tourism
- cloudy
- brick
- ...

Object detection

- find pedestrians

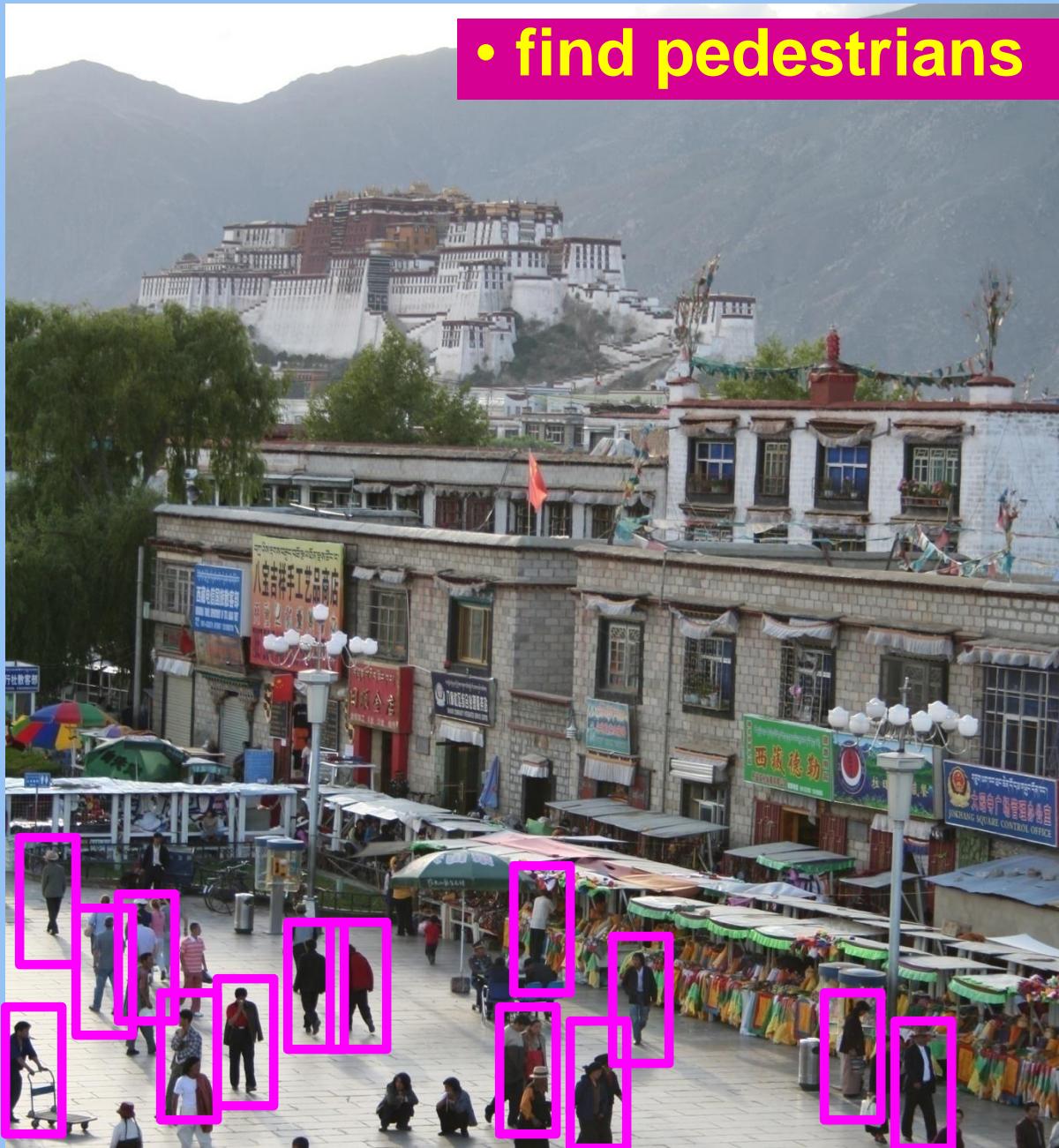


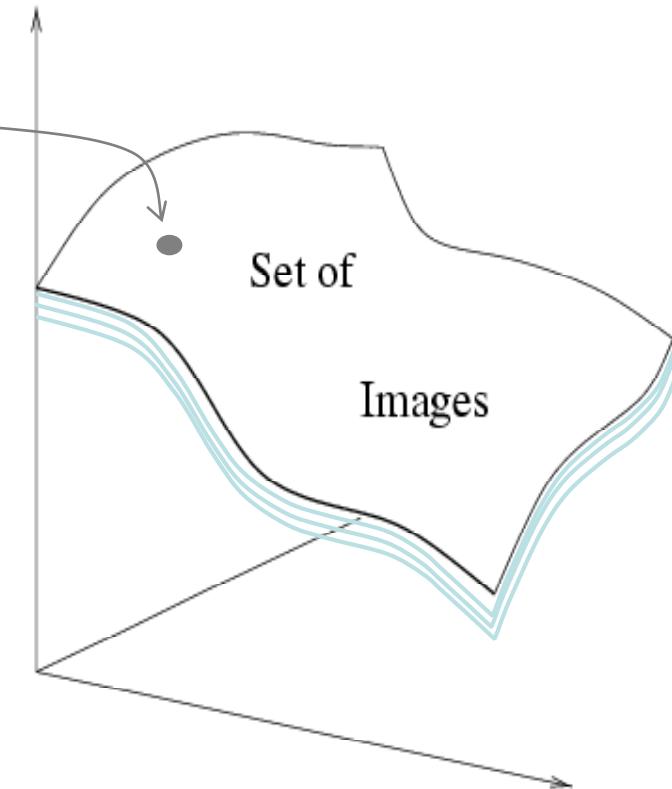
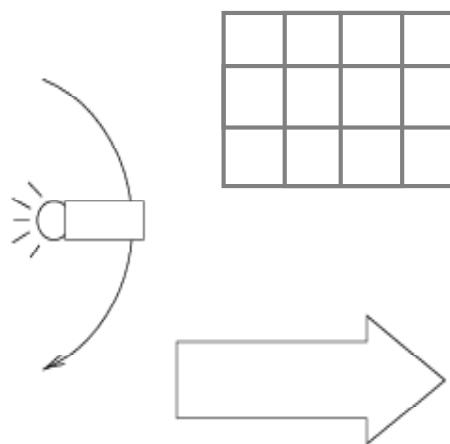
Image parsing / semantic segmentation



Scene understanding?



Recognition is all about modeling variability



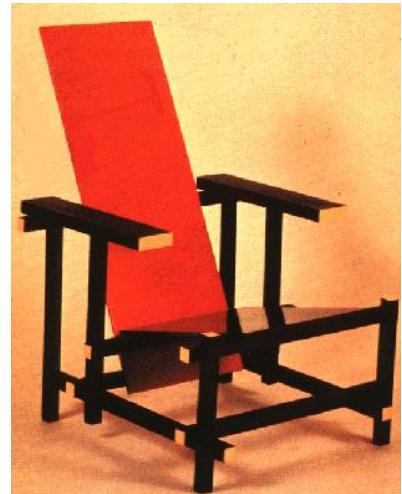
Variability:

- Camera position
- Illumination
- Shape parameters



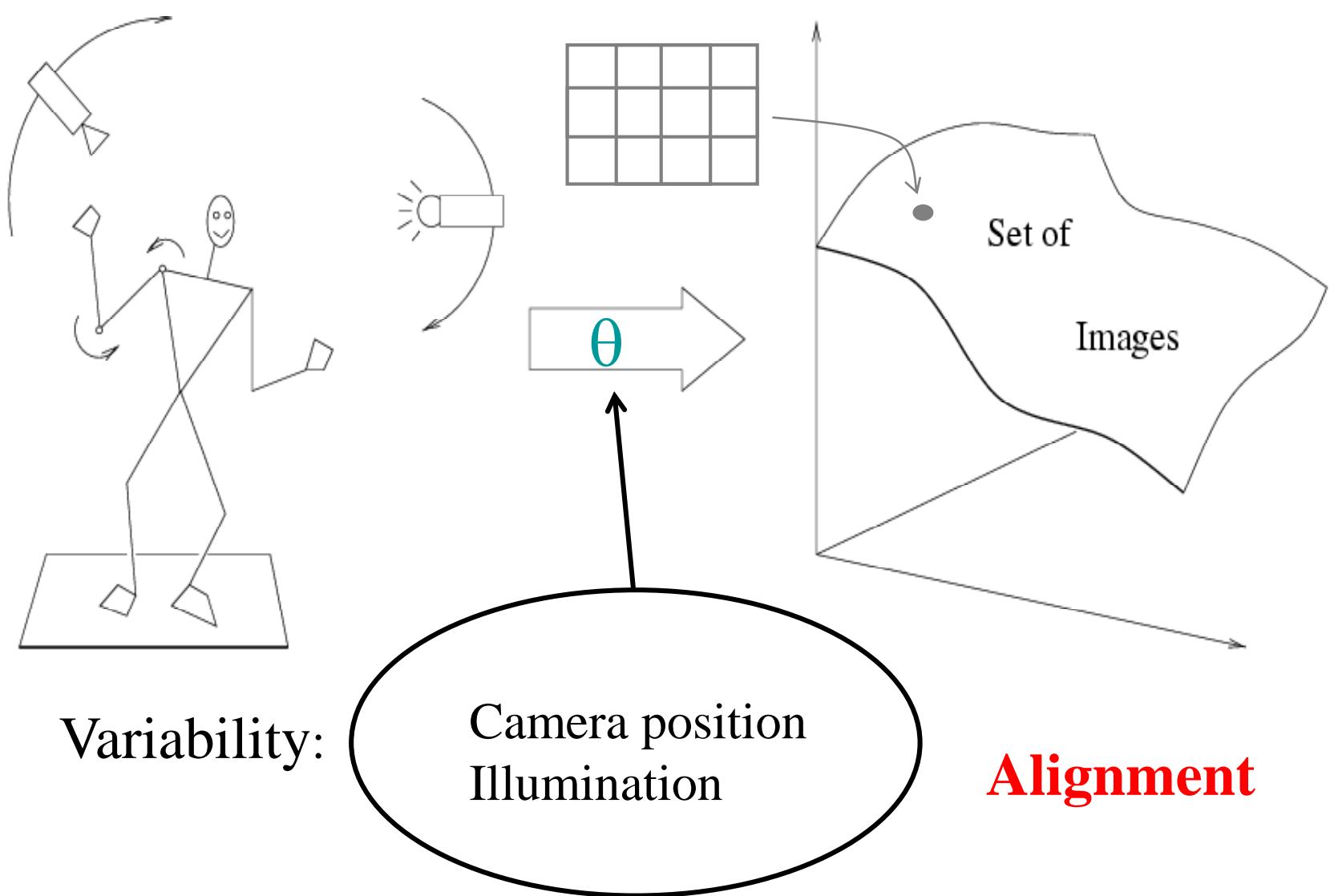
Within-class variations?

Within-class variations



History of ideas in recognition

- 1960s – early 1990s: the geometric era

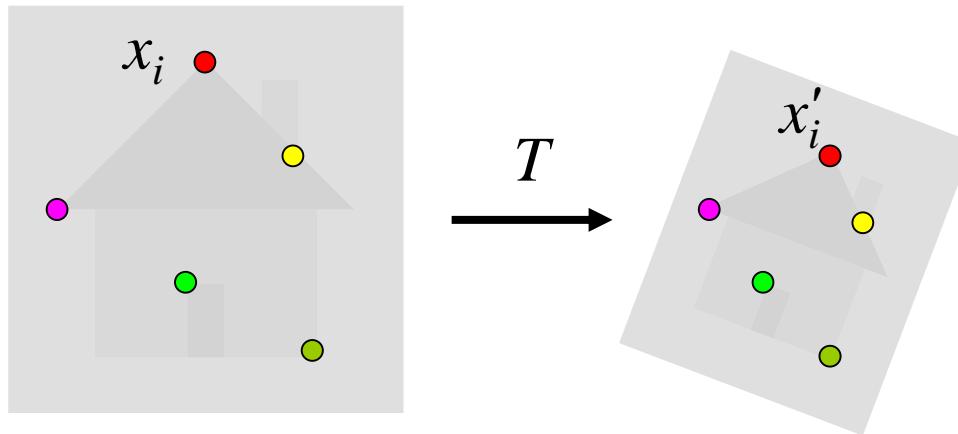


Shape: assumed known

Roberts (1965); Lowe (1987); Faugeras & Hebert (1986); Grimson & Lozano-Perez (1986);
Huttenlocher & Ullman (1987)

Recall: Alignment

- Alignment: fitting a model to a transformation between pairs of features (*matches*) in two images



Find transformation T
that minimizes

$$\sum_i \text{residual}(T(x_i), x'_i)$$

Recognition as an alignment problem: Block world

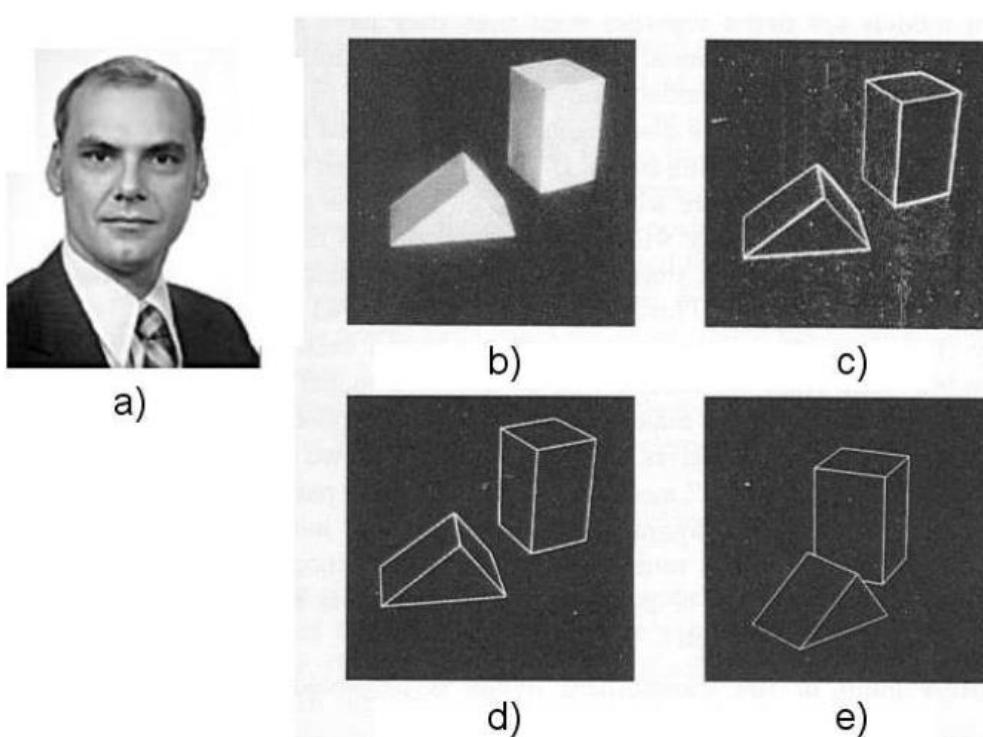
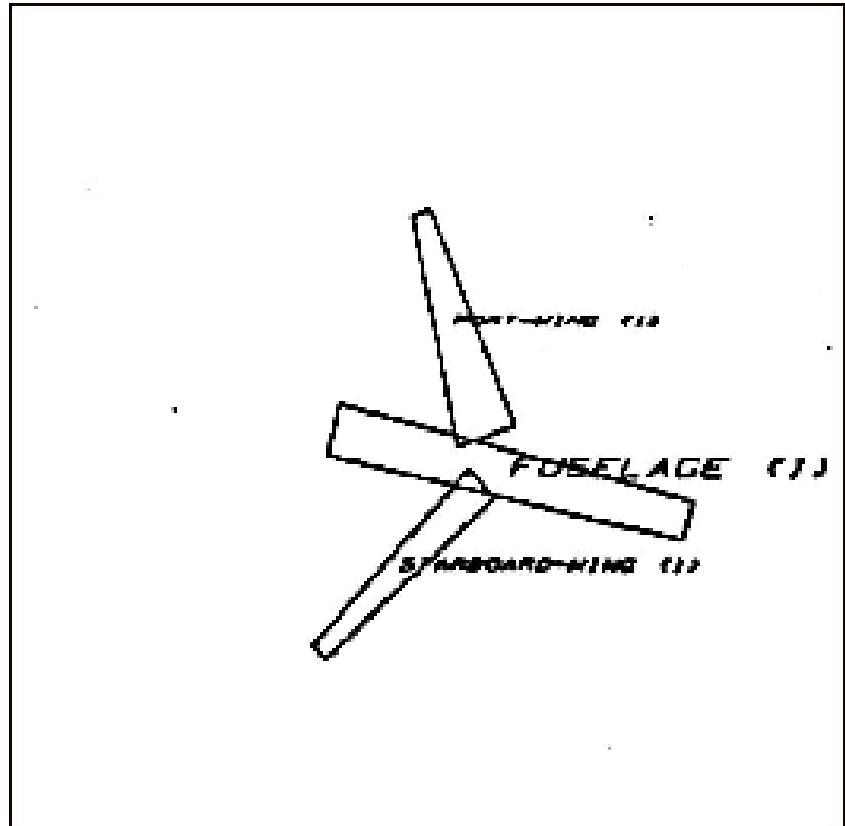
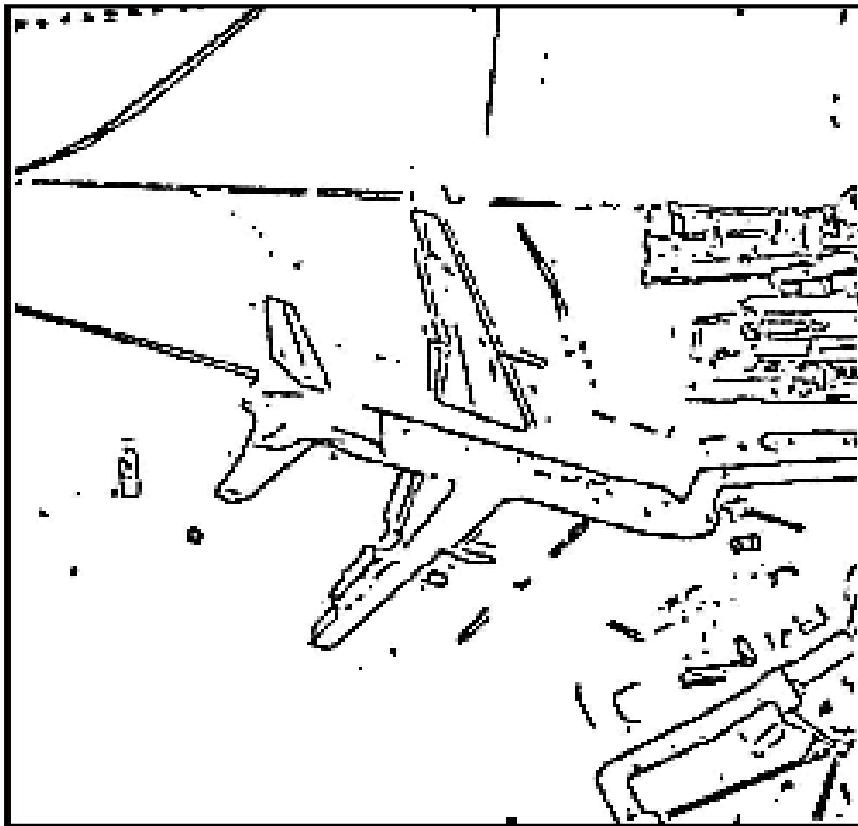


Fig. 1. A system for recognizing 3-d polyhedral scenes. a) L.G. Roberts. b) A blocks world scene. c) Detected edges using a 2×2 gradient operator. d) A 3-d polyhedral description of the scene, formed automatically from the single image. e) The 3-d scene displayed with a viewpoint different from the original image to demonstrate its accuracy and completeness. (b) - e) are taken from [64] with permission MIT Press.)

L. G. Roberts, *Machine Perception of Three Dimensional Solids*, Ph.D. thesis, MIT Department of Electrical Engineering, 1963.

Representing and recognizing object categories is harder...



ACRONYM (Brooks and Binford, 1981)

Binford (1971), Nevatia & Binford (1972), Marr & Nishihara (1978)

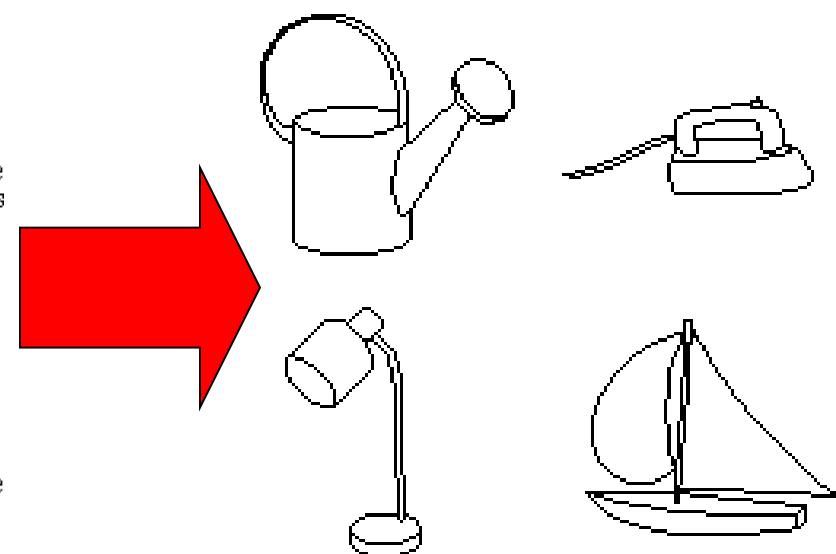
Recognition by components

Biederman (1987)

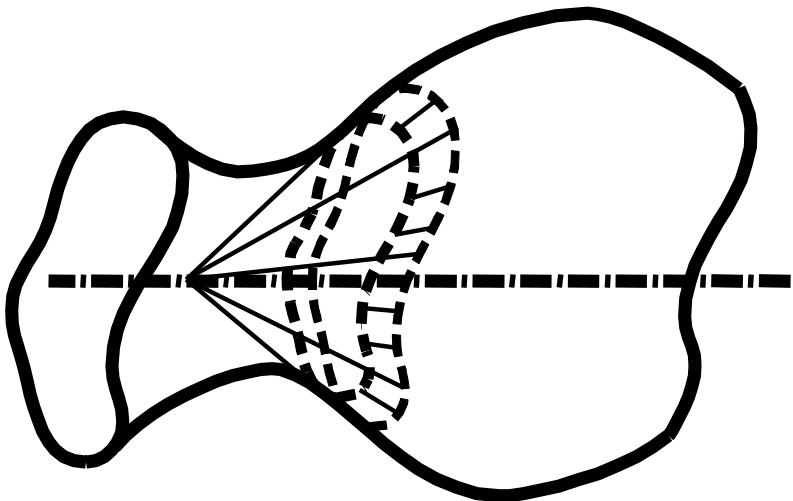
Primitives (geons)

Cube	Wedge	Pyramid	Cylinder	Barrel
Straight Edge Straight Axis Constant	Straight Edge Straight Axis Expanded	Straight Edge Straight Axis Expanded	Curved Edge Straight Axis Constant	Curved Edge Straight Axis Exp & Cont
Arch	Cone	Expanded Cylinder	Handle	Expanded Handle
Straight Edge Curved Axis Constant	Curved Edge Straight Axis Expanded	Curved Edge Straight Axis Expanded	Curved Edge Curved Axis Constant	Curved Edge Curved Axis Expanded

Objects

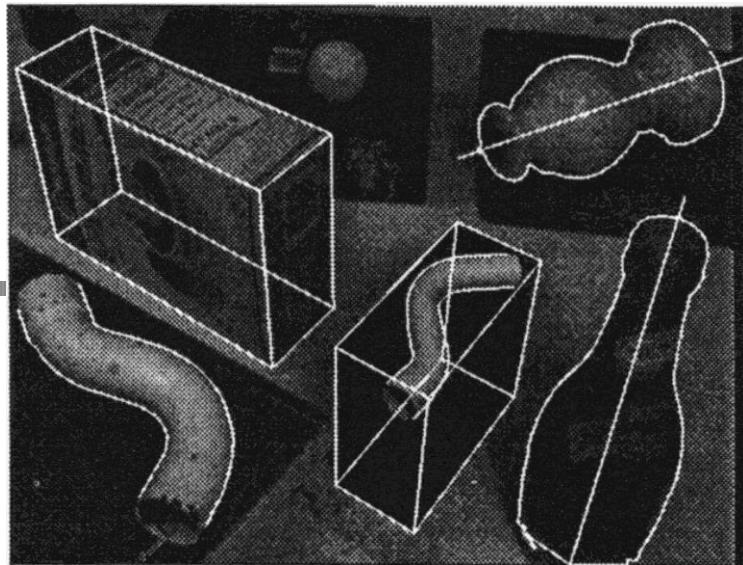
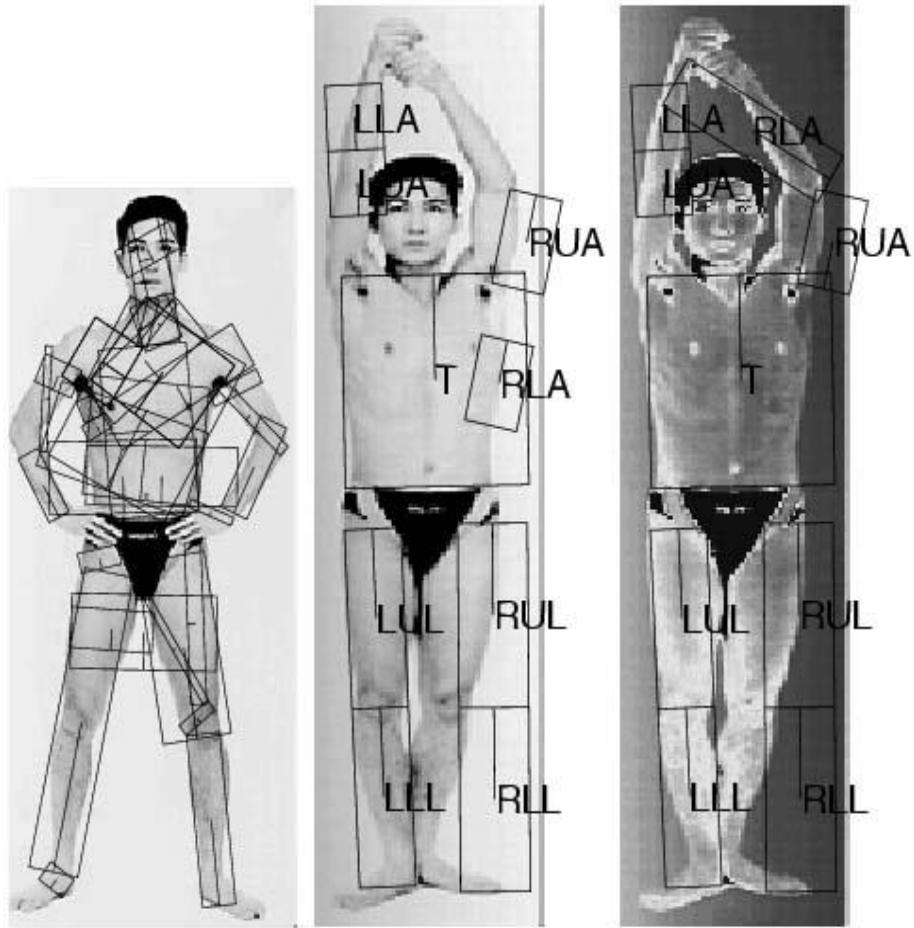


http://en.wikipedia.org/wiki/Recognition_by_Components_Theory



Generalized cylinders
Ponce et al. (1989)

General shape primitives?

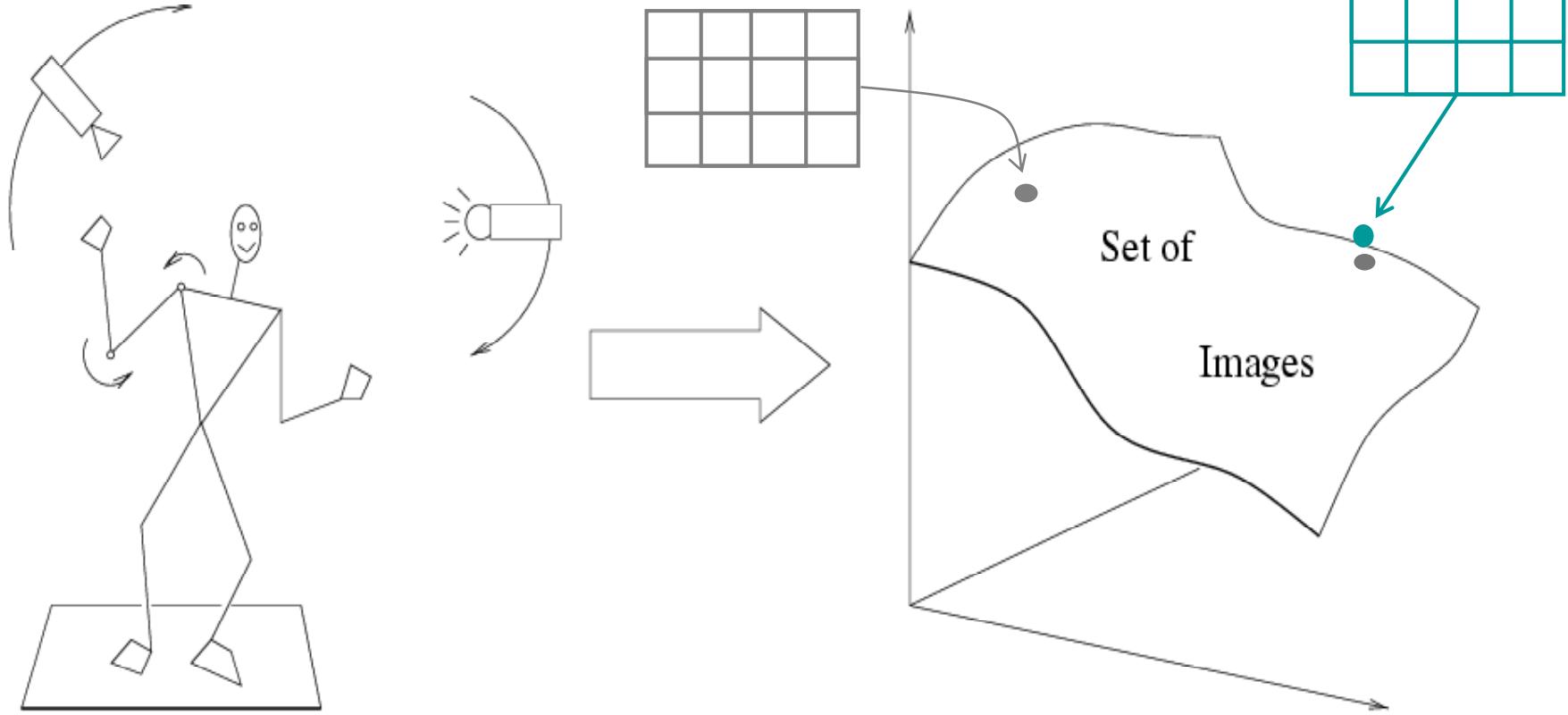


Zisserman et al. (1995)

Forsyth (2000)

History of ideas in recognition

- 1960s – early 1990s: the geometric era
- 1990s: appearance-based models

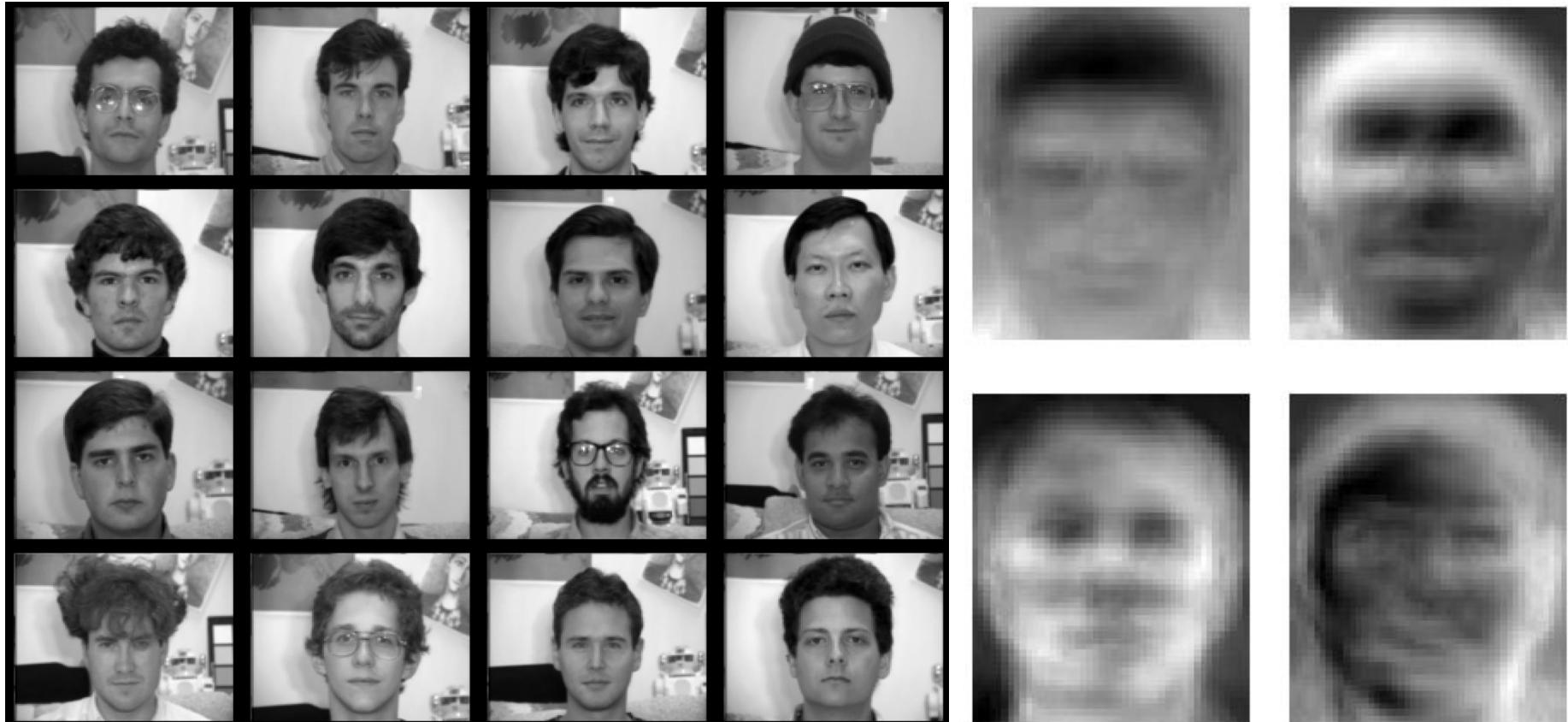


Empirical models of image variability

Appearance-based techniques

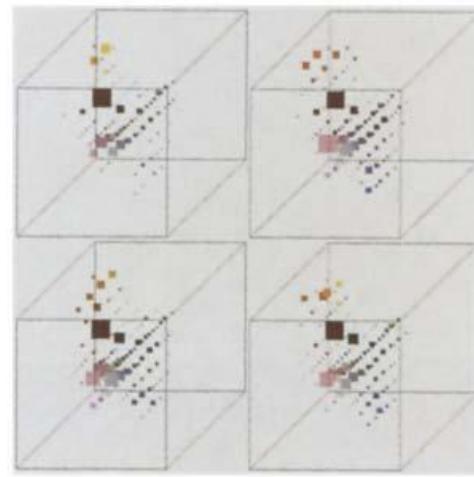
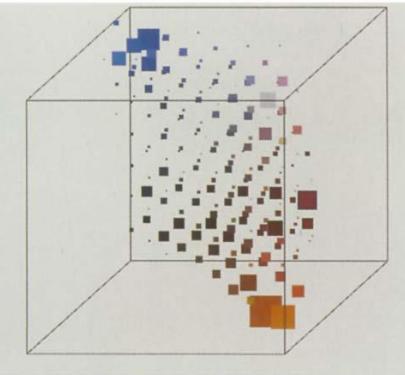
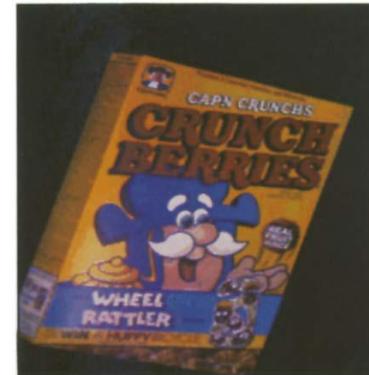
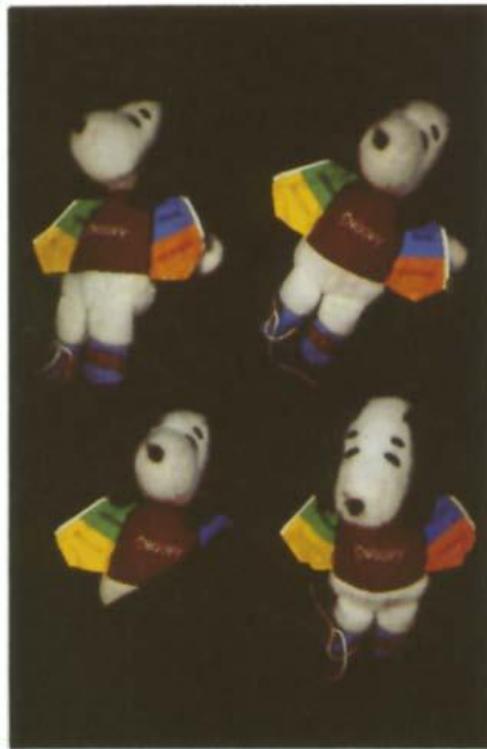
Turk & Pentland (1991); Murase & Nayar (1995); etc.

Eigenfaces (Turk & Pentland, 1991)



Experimental Condition	Correct/Unknown Recognition Percentage		
Condition	Lighting	Orientation	Scale
Forced classification	96/0	85/0	64/0
Forced 100% accuracy	100/19	100/39	100/60
Forced 20% unknown rate	100/20	94/20	74/20

Color Histograms



Swain and Ballard, [Color Indexing](#), IJCV 1991.

Svetlana Lazebnik

History of ideas in recognition

- 1960s – early 1990s: the geometric era
- 1990s: appearance-based models
- 1990s – present: sliding window approaches

Sliding window approaches



Sliding window approaches



- Turk and Pentland, 1991
- Belhumeur, Hespanha, & Kriegman, 1997
- Schneiderman & Kanade 2004
- Viola and Jones, 2000



- Schneiderman & Kanade, 2004
- Argawal and Roth, 2002
- Poggio et al. 1993

History of ideas in recognition

- 1960s – early 1990s: the geometric era
- 1990s: appearance-based models
- Mid-1990s: sliding window approaches
- Late 1990s: local features

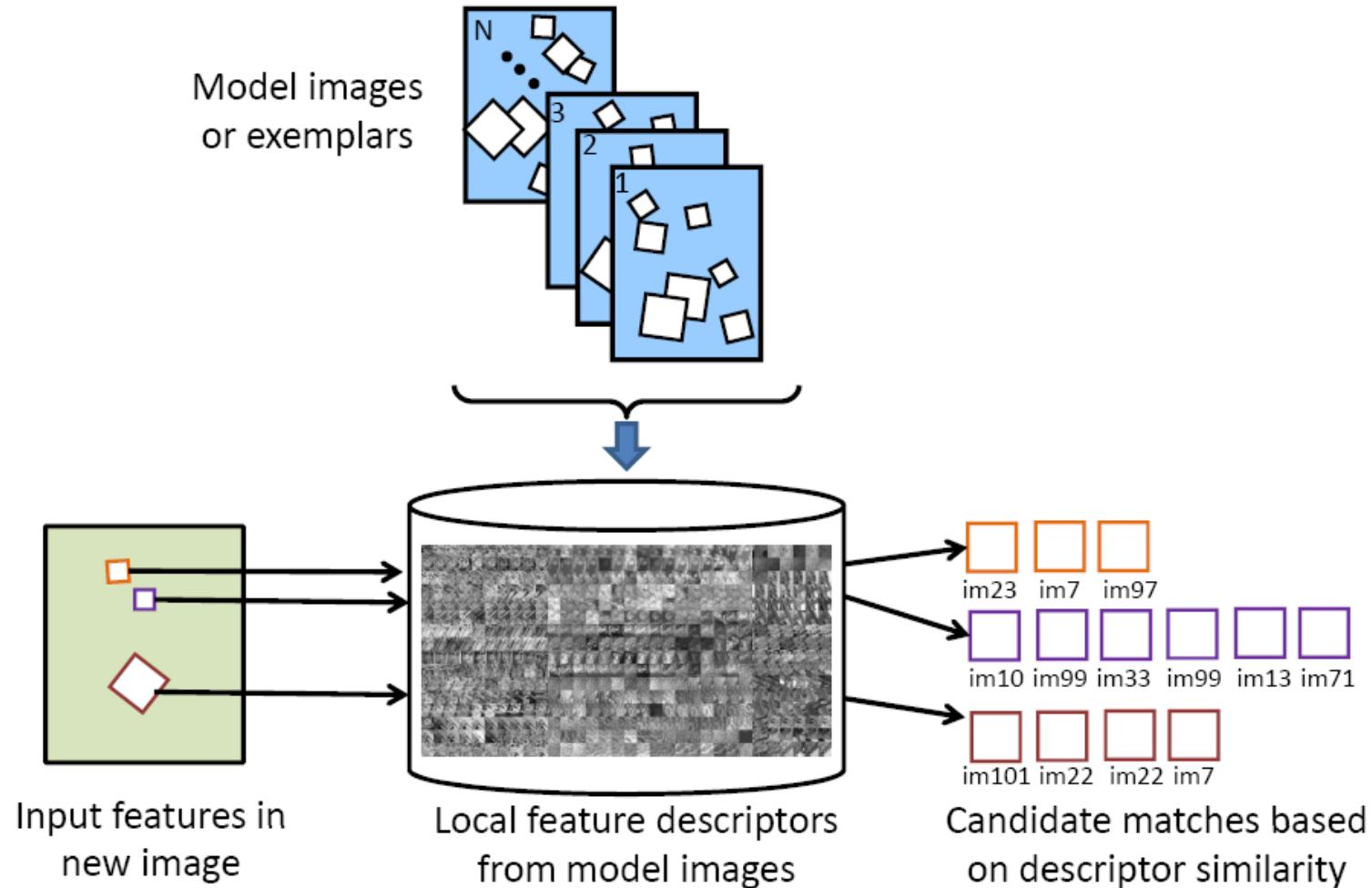
Local features for object instance recognition



D. Lowe (1999, 2004)

Large-scale image search

Combining local features, indexing, and spatial constraints



Large-scale image search

Combining local features, indexing, and spatial constraints



Philbin et al. '07

Large-scale image search

Combining local features, indexing, and spatial constraints

Google Goggles in Action

Click the icons below to see the different ways Google Goggles can be used.



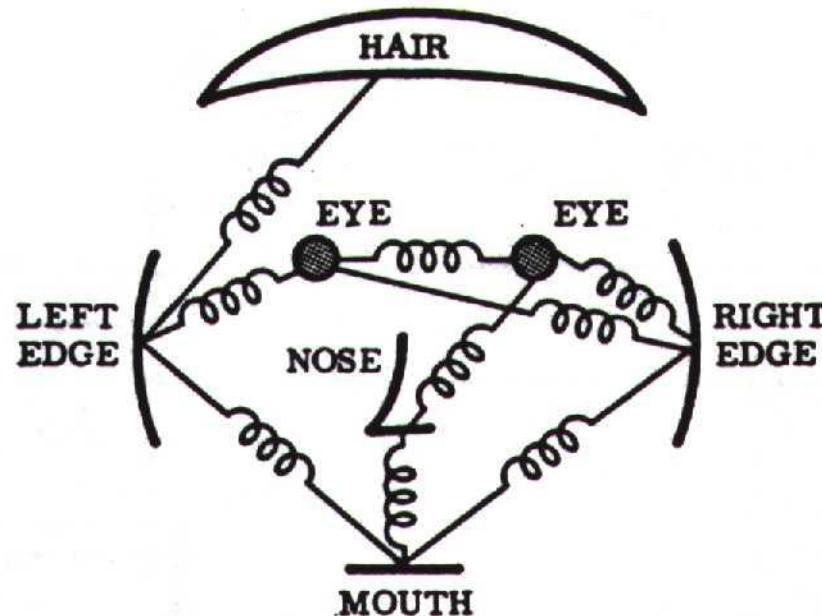
Available on phones that run Android 1.6+ (i.e. Donut or Eclair)

History of ideas in recognition

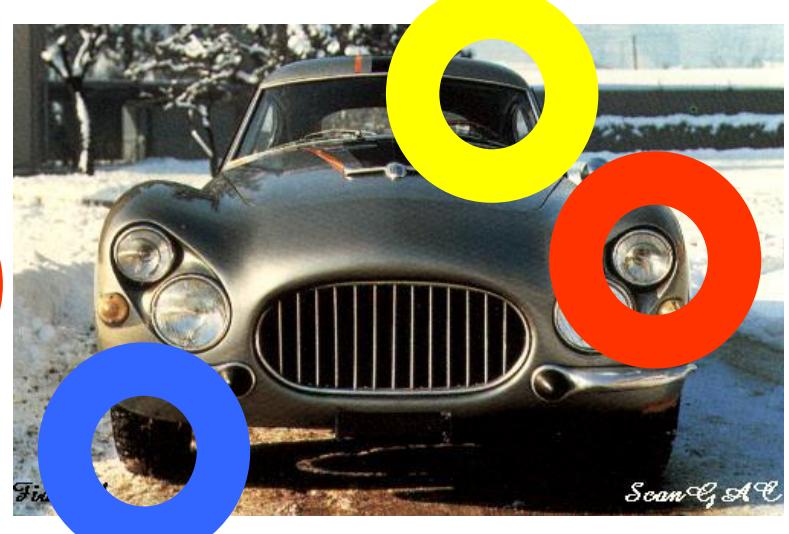
- 1960s – early 1990s: the geometric era
- 1990s: appearance-based models
- Mid-1990s: sliding window approaches
- Late 1990s: local features
- Early 2000s: parts-and-shape models

Parts-and-shape models

- Model:
 - Object as a set of parts
 - Relative locations between parts
 - Appearance of part



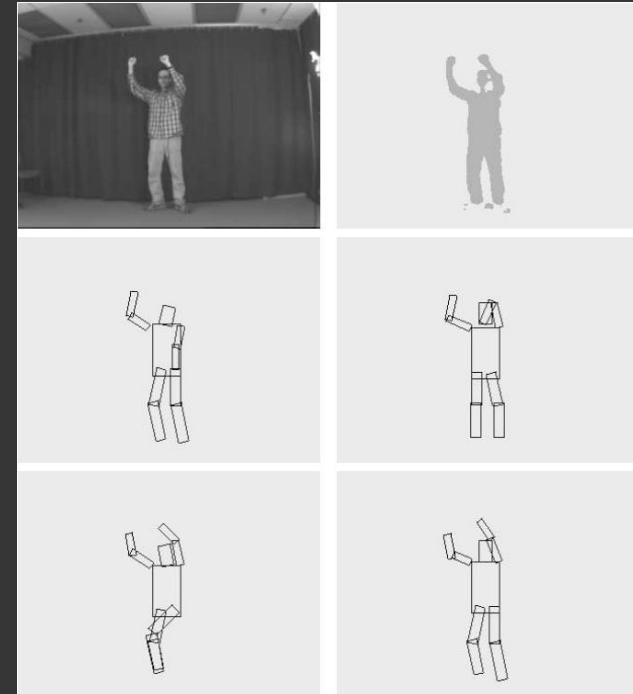
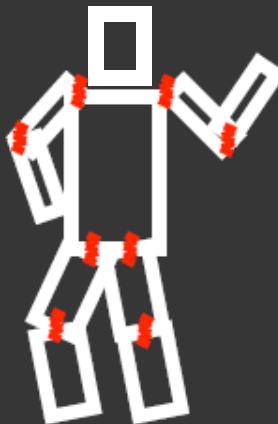
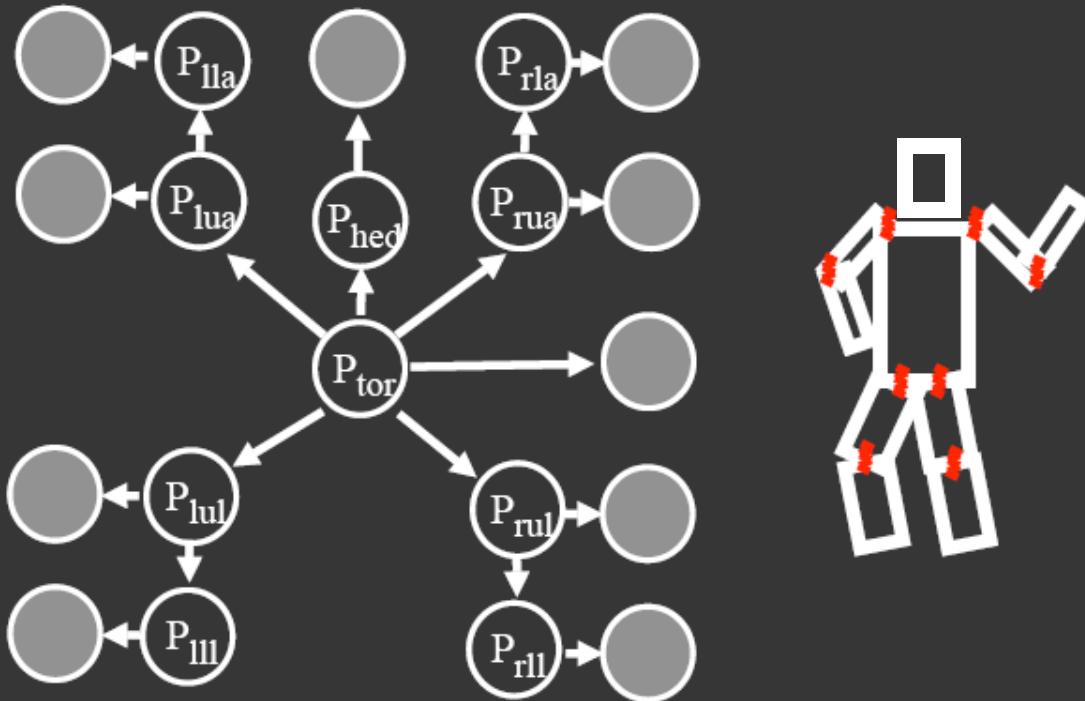
Constellation models



Weber, Welling & Perona (2000), Fergus, Perona & Zisserman (2003)

Pictorial structure model

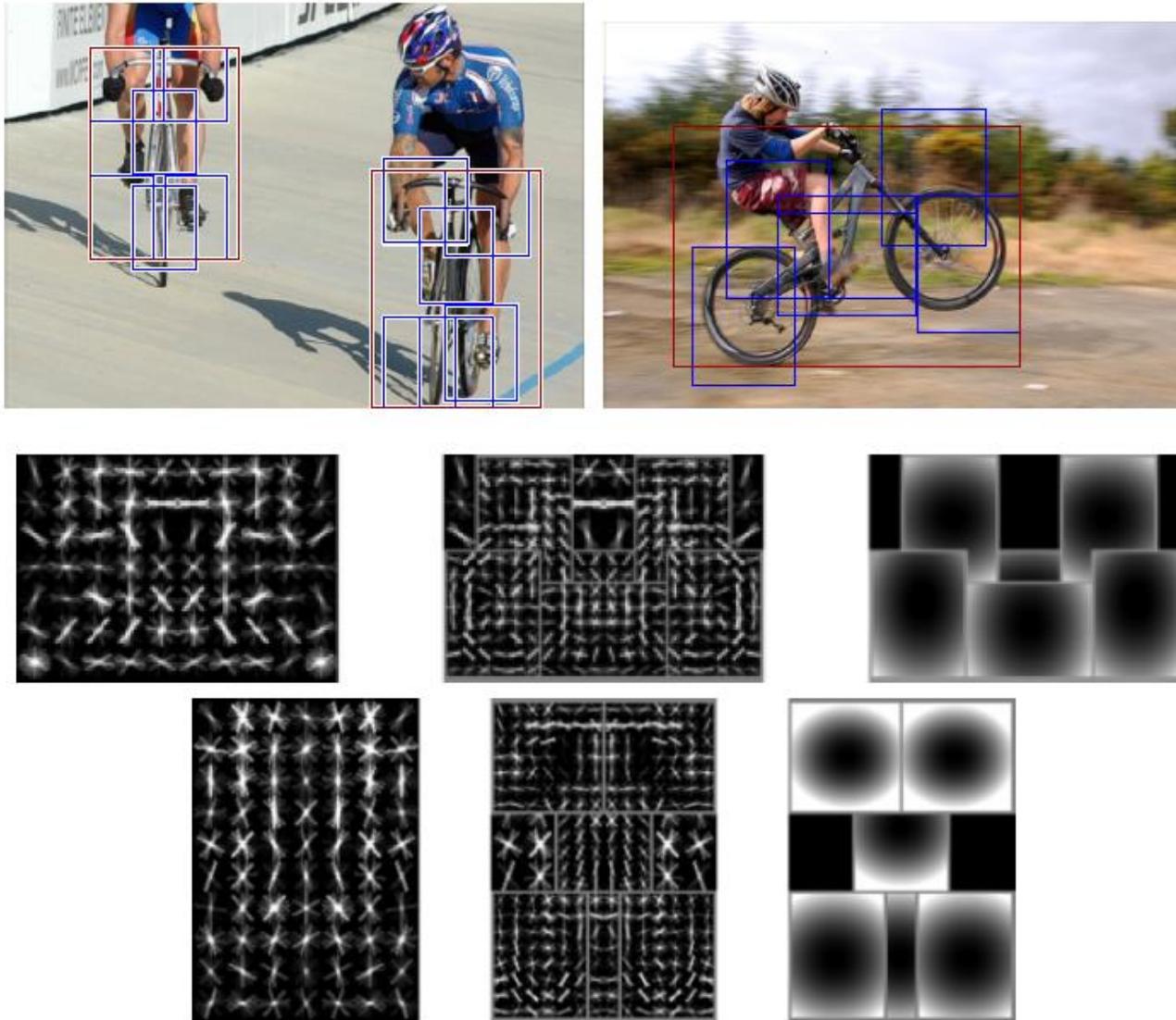
Fischler and Elschlager(73), Felzenszwalb and Huttenlocher(00)



$$\Pr(P_{tor}, P_{arm}, \dots | Im) \propto \prod_{i,j} \Pr(P_i | P_j) \prod_i \Pr(Im(P_i))$$

↑
part geometry ↙
part appearance

Discriminatively trained part-based models

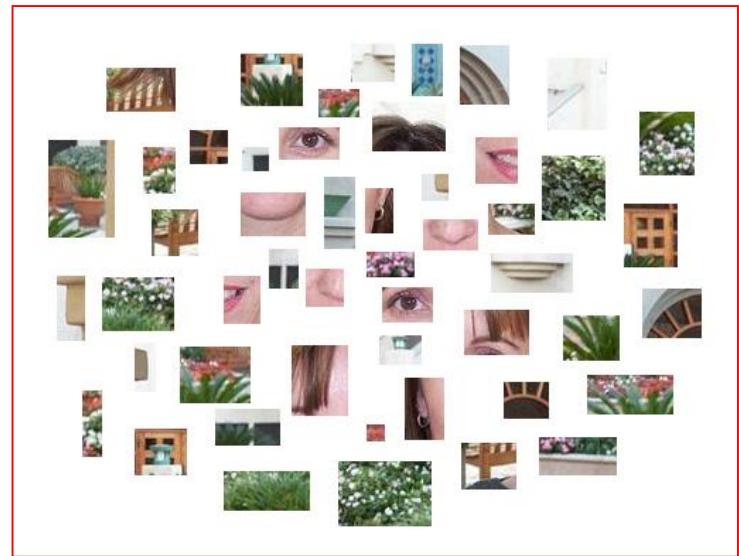
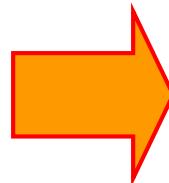


P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan, ["Object Detection with Discriminatively Trained Part-Based Models,"](#) PAMI 2009

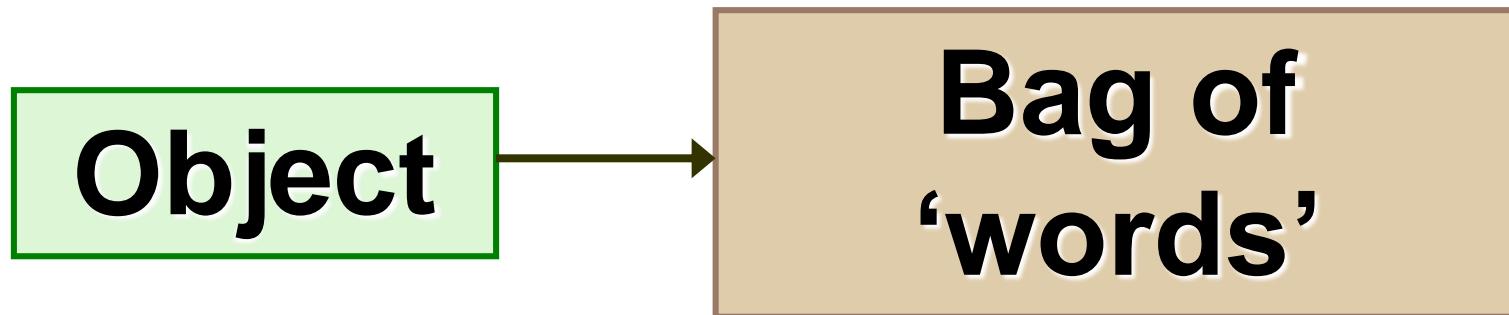
History of ideas in recognition

- 1960s – early 1990s: the geometric era
- 1990s: appearance-based models
- Mid-1990s: sliding window approaches
- Late 1990s: local features
- Early 2000s: parts-and-shape models
- Mid-2000s: bags of features

Bag-of-features models

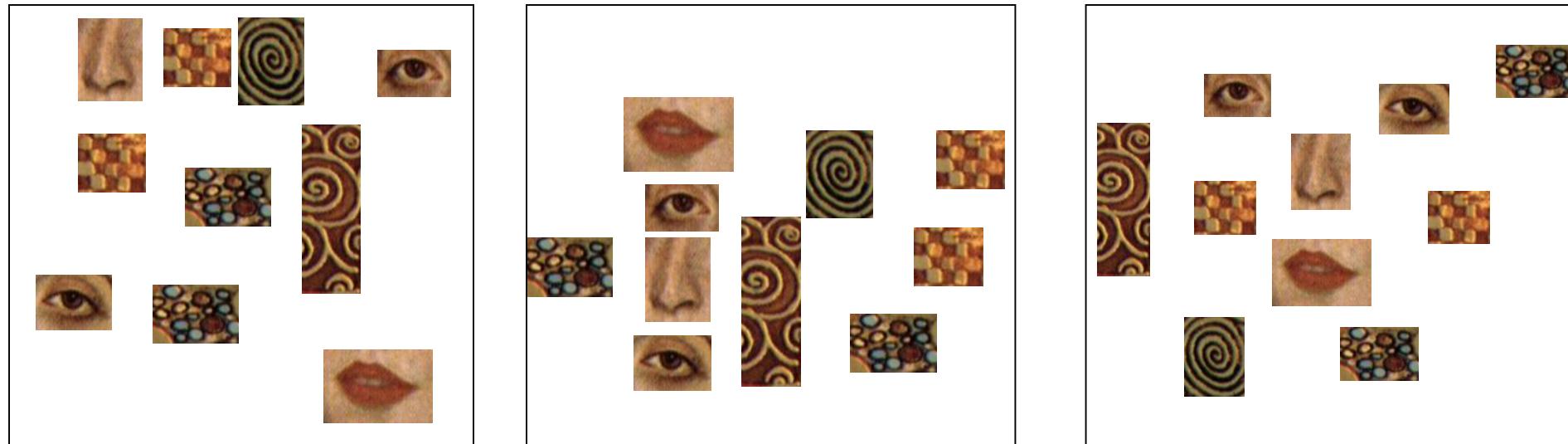


Bag-of-features models



Objects as texture

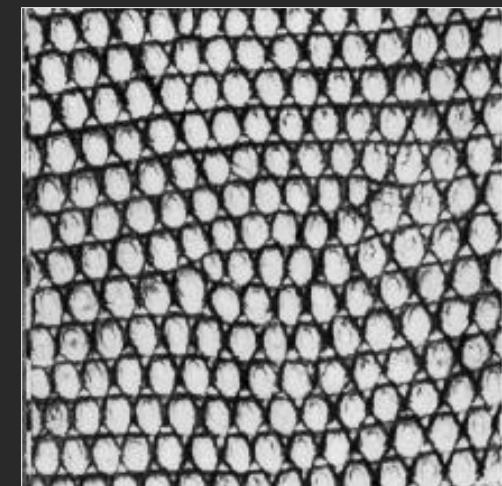
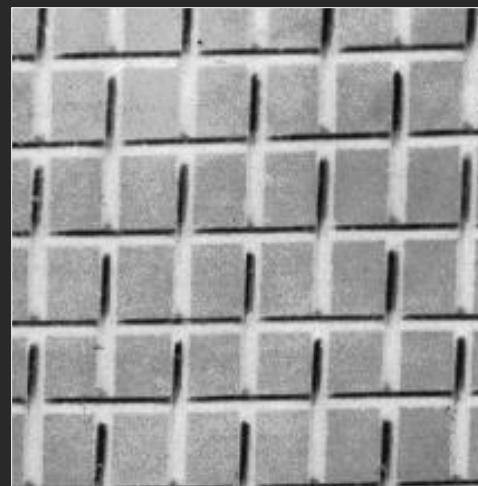
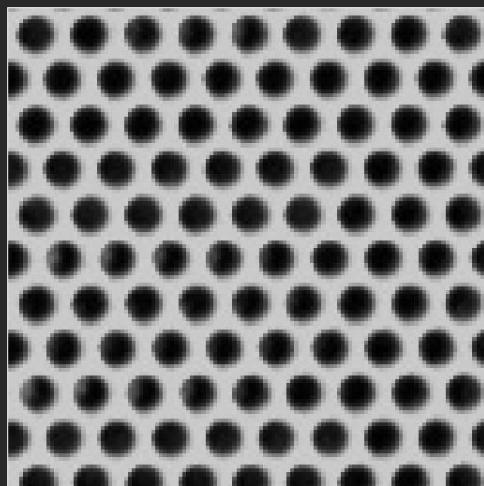
- All of these are treated as being the same



- No distinction between foreground and background: scene recognition?

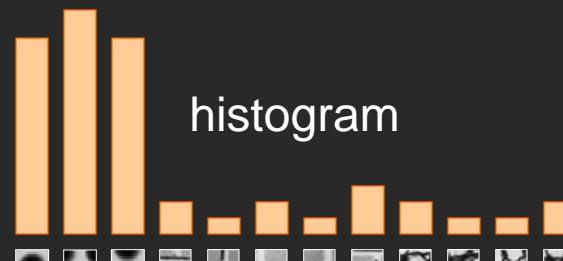
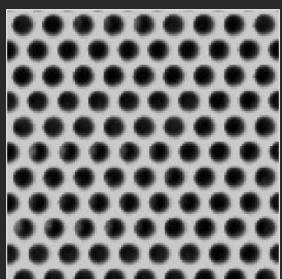
Origin 1: Texture recognition

- Texture is characterized by the repetition of basic elements or *textons*
- For stochastic textures, it is the identity of the textons, not their spatial arrangement, that matters

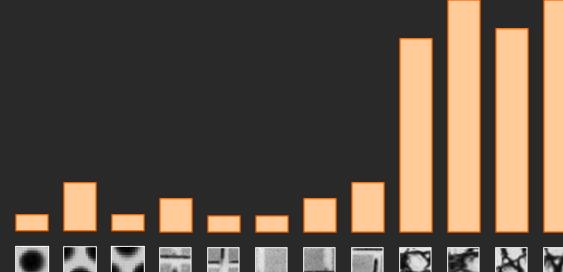
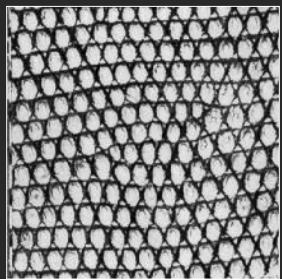
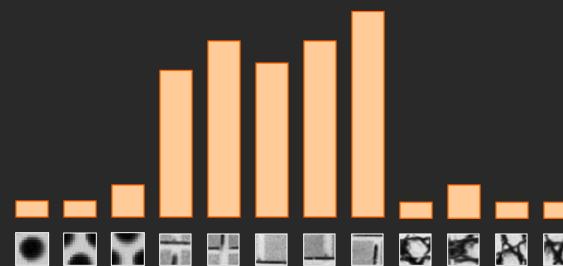
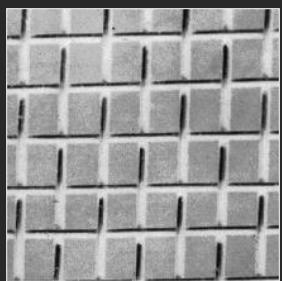


Julesz, 1981; Cula & Dana, 2001; Leung & Malik 2001; Mori, Belongie & Malik, 2001; Schmid 2001; Varma & Zisserman, 2002, 2003; Lazebnik, Schmid & Ponce, 2003

Origin 1: Texture recognition



Universal texton dictionary



Julesz, 1981; Cula & Dana, 2001; Leung & Malik 2001; Mori, Belongie & Malik, 2001; Schmid 2001; Varma & Zisserman, 2002, 2003; Lazebnik, Schmid & Ponce, 2003

Origin 2: Bag-of-words models

- Orderless document representation: frequencies of words from a dictionary Salton & McGill (1983)

Origin 2: Bag-of-words models

- Orderless document representation: frequencies of words from a dictionary Salton & McGill (1983)



Origin 2: Bag-of-words models

- Orderless document representation: frequencies of words from a dictionary Salton & McGill (1983)



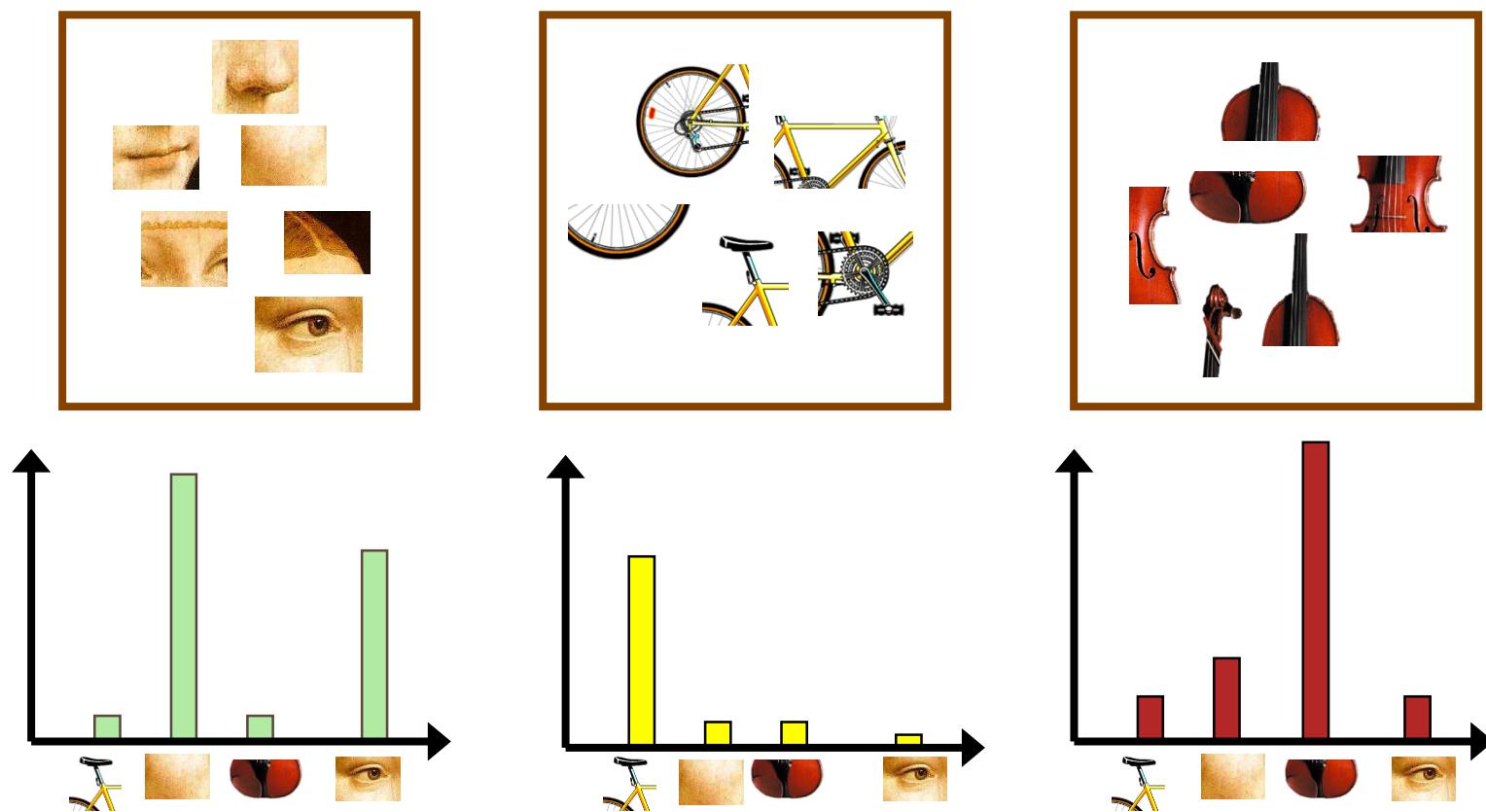
Origin 2: Bag-of-words models

- Orderless document representation: frequencies of words from a dictionary Salton & McGill (1983)



Bag-of-features steps

1. Extract features
2. Learn “visual vocabulary”
3. Quantize features using visual vocabulary
4. Represent images by frequencies of “visual words”

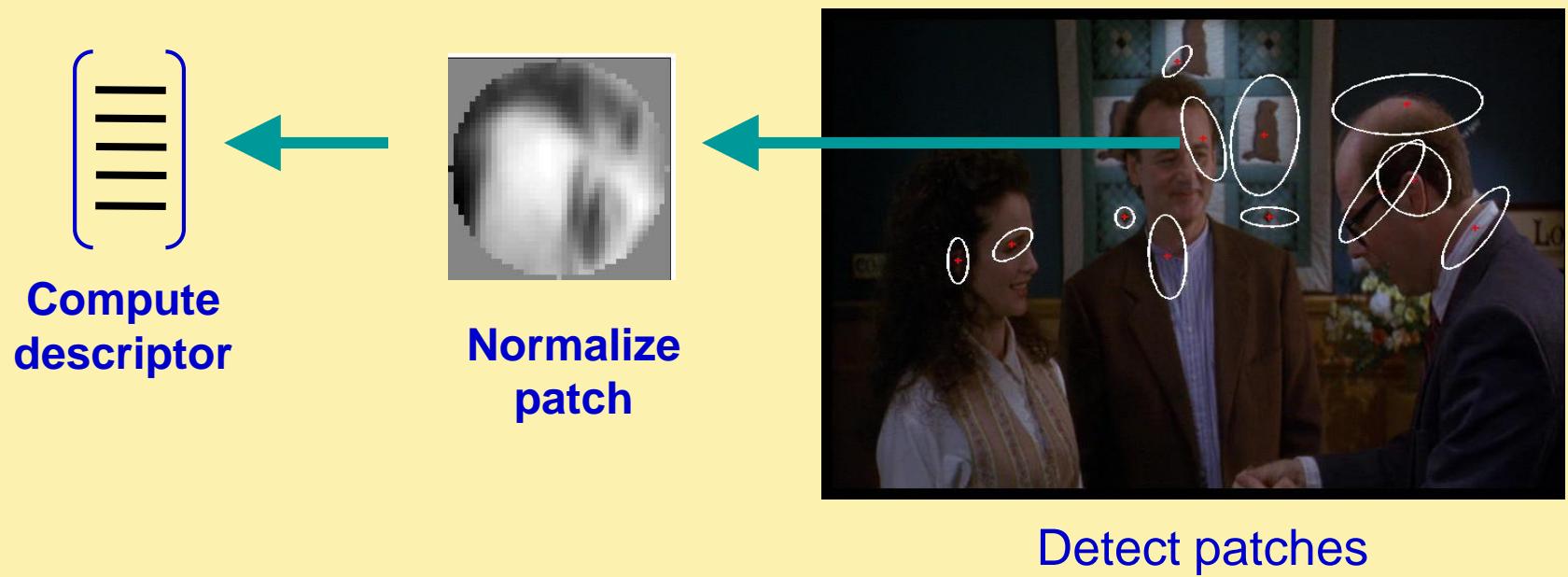


1. Feature extraction

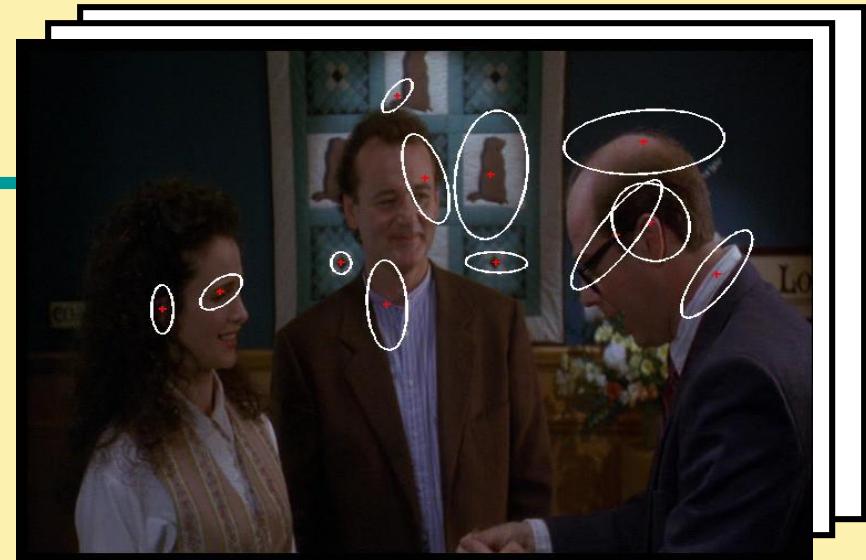
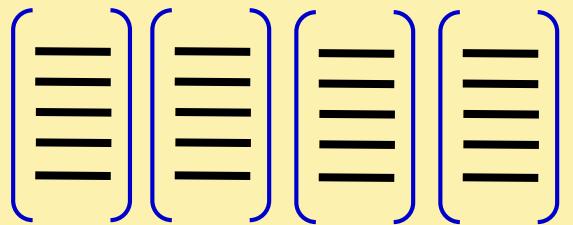
- Regular grid or interest regions



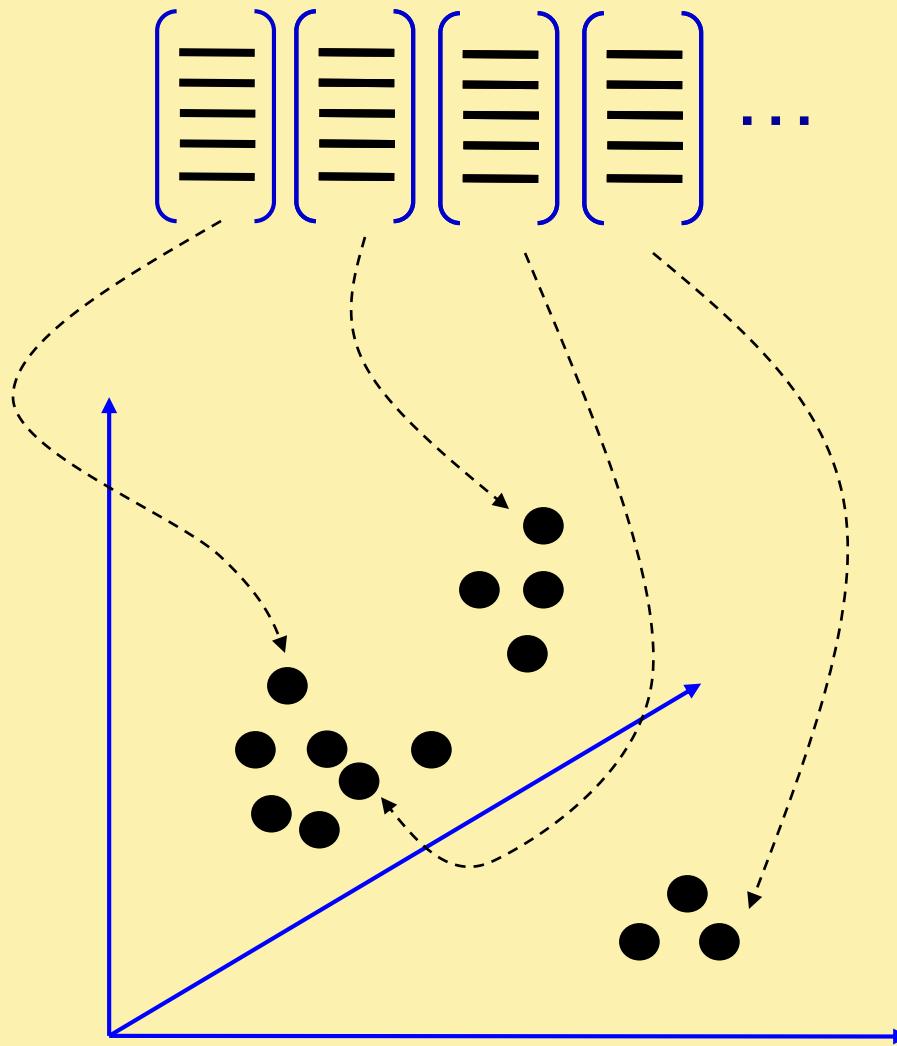
1. Feature extraction



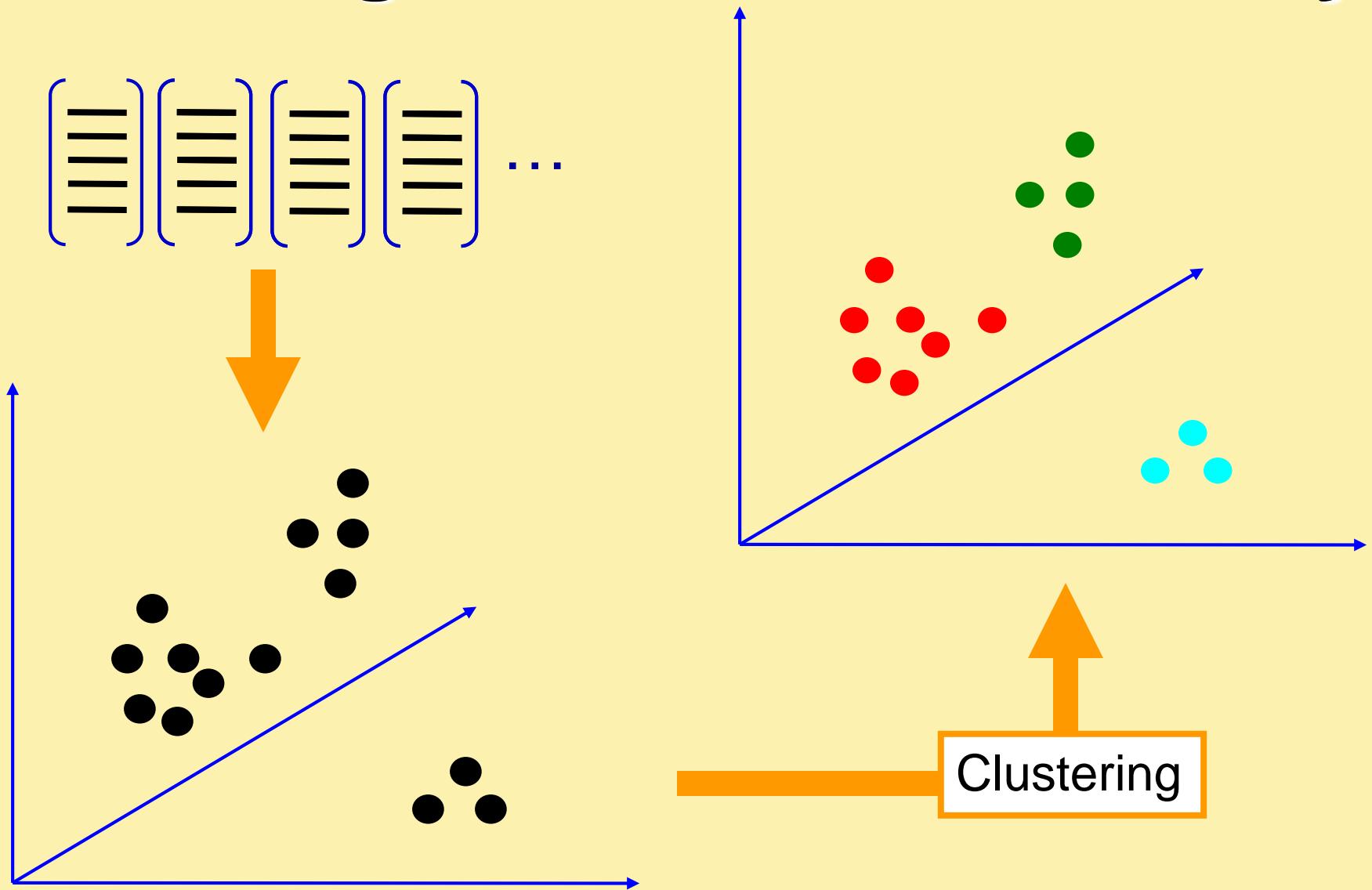
1. Feature extraction



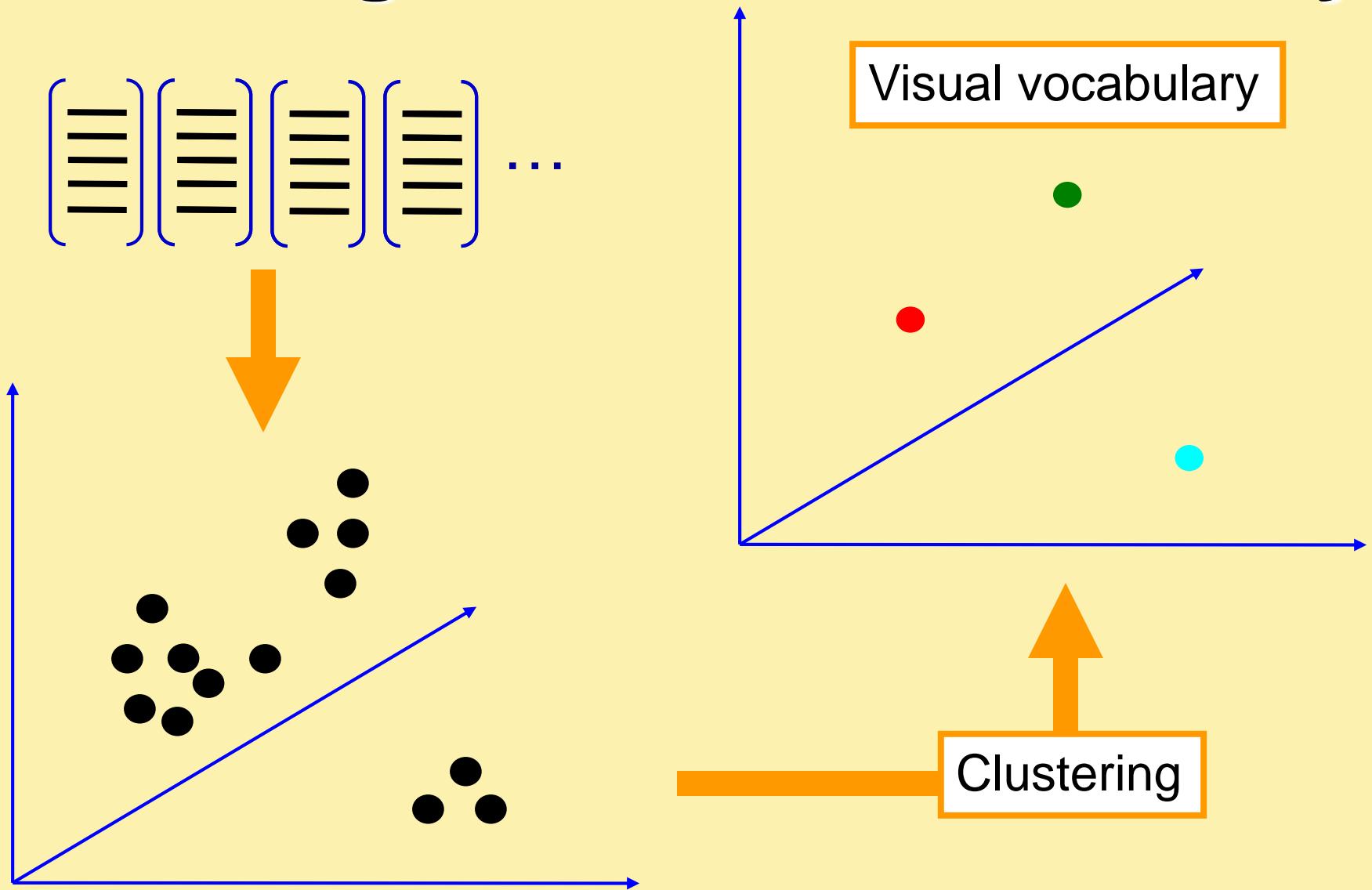
2. Learning the visual vocabulary



2. Learning the visual vocabulary



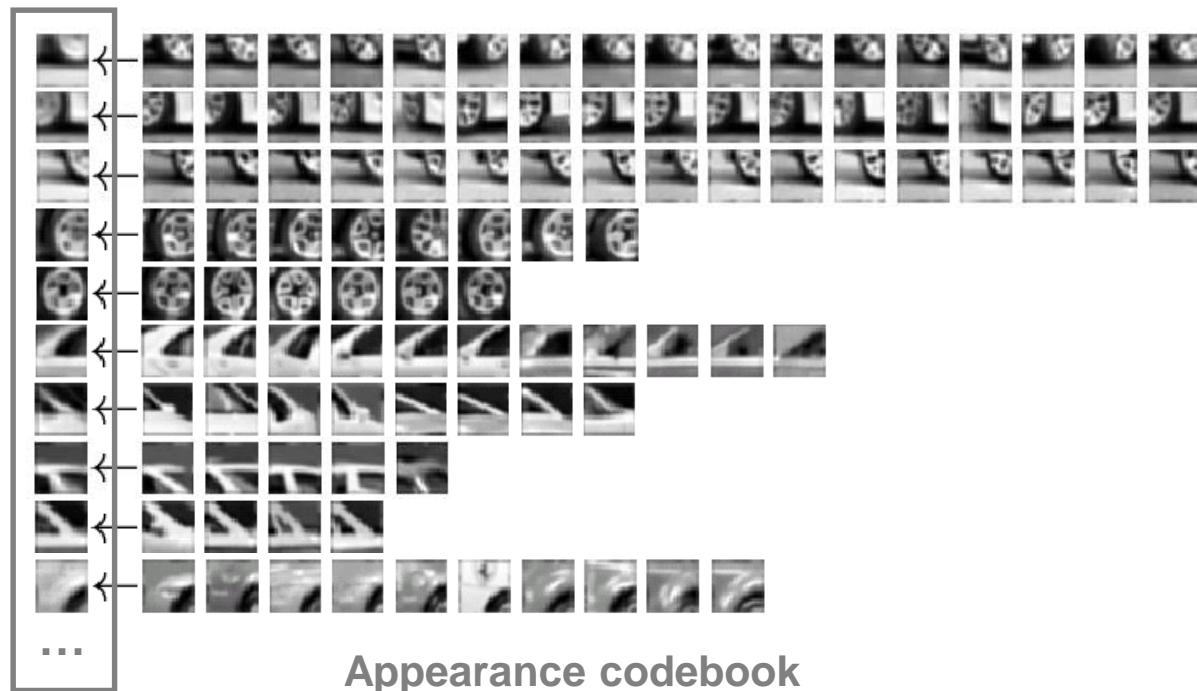
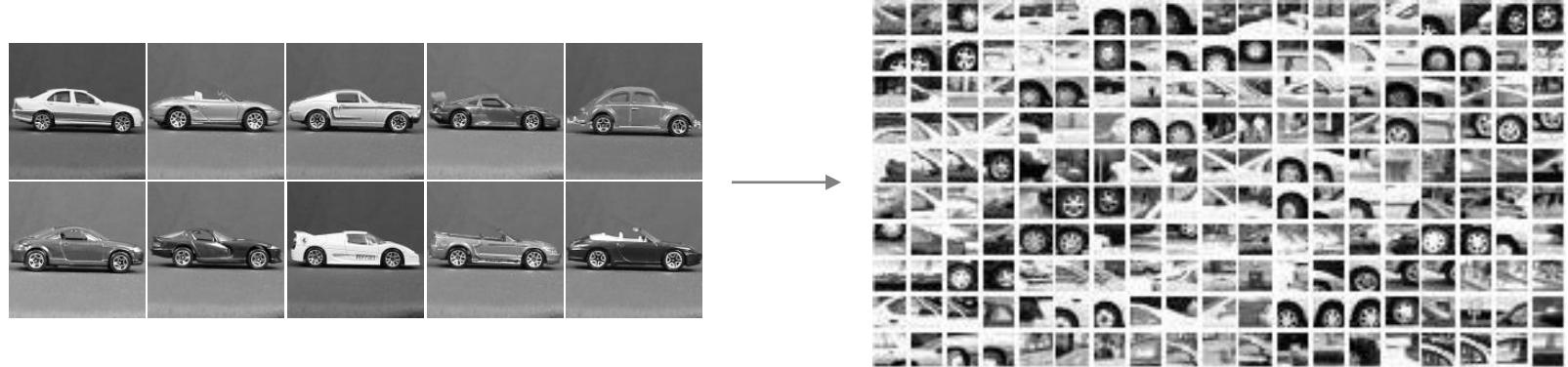
2. Learning the visual vocabulary



Clustering and vector quantization

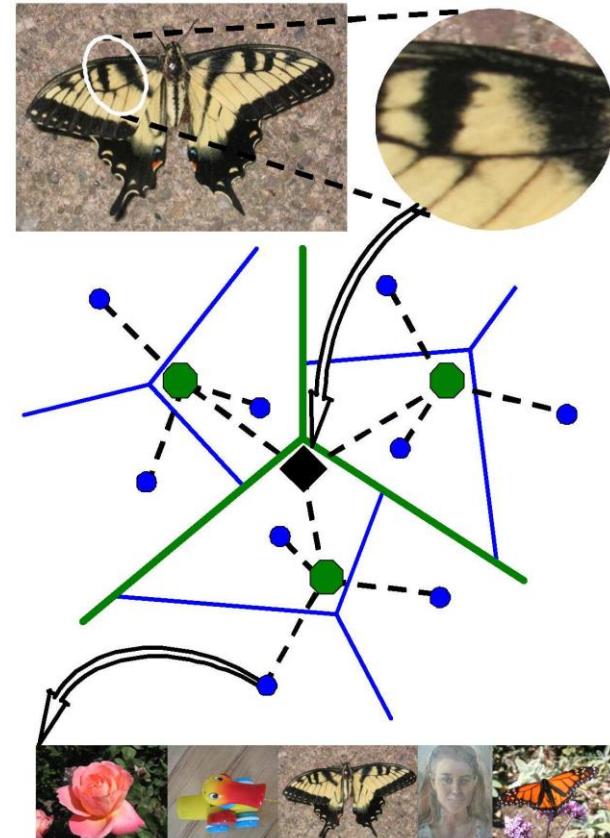
- Clustering is a common method for learning a visual vocabulary or codebook
 - Unsupervised learning process
 - Each cluster center produced by k-means becomes a codevector
 - Codebook can be learned on separate training set
 - Provided the training set is sufficiently representative, the codebook will be “universal”
- The codebook is used for quantizing features
 - A *vector quantizer* takes a feature vector and maps it to the index of the nearest codevector in a codebook
 - Codebook = visual vocabulary
 - Codevector = visual word

Example codebook

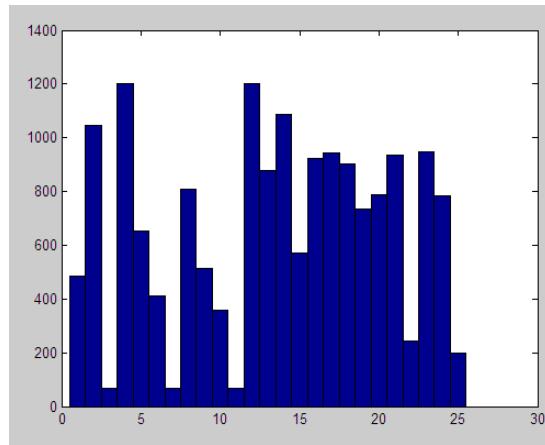


Visual vocabularies: Issues

- How to choose vocabulary size?
 - Too small: visual words not representative of all patches
 - Too large: quantization artifacts, overfitting
- Computational efficiency
 - Vocabulary trees
(Nister & Stewenius, 2006)



But what about layout?



All of these images have the same color histogram

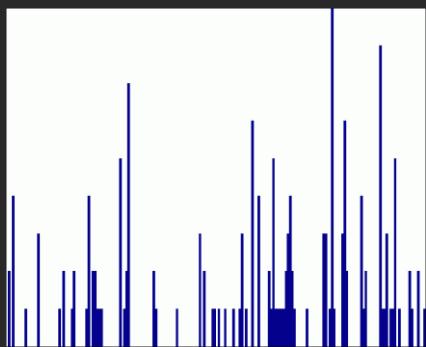
Spatial pyramid



Compute histogram in each spatial bin

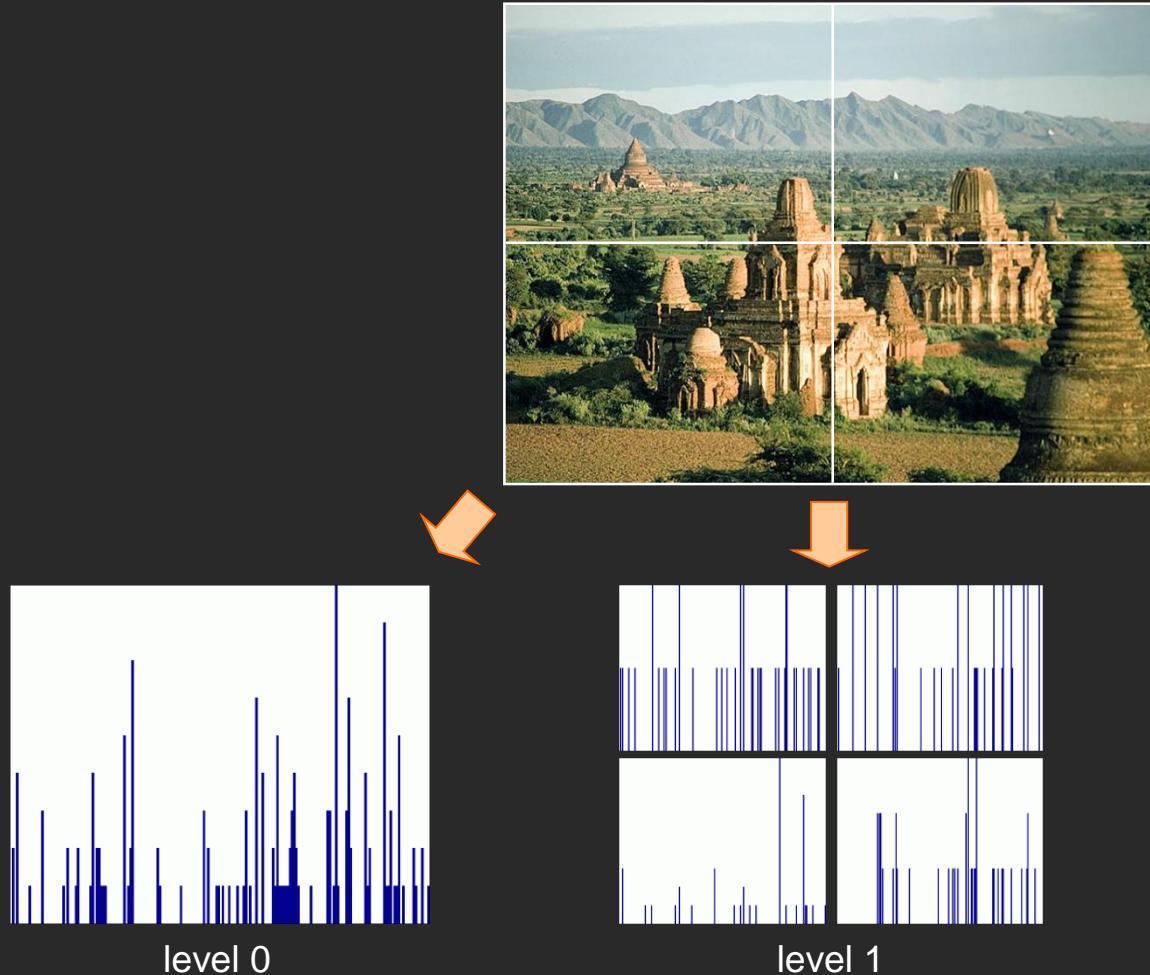
Spatial pyramid representation

- Extension of a bag of features
- Locally orderless representation at several levels of resolution



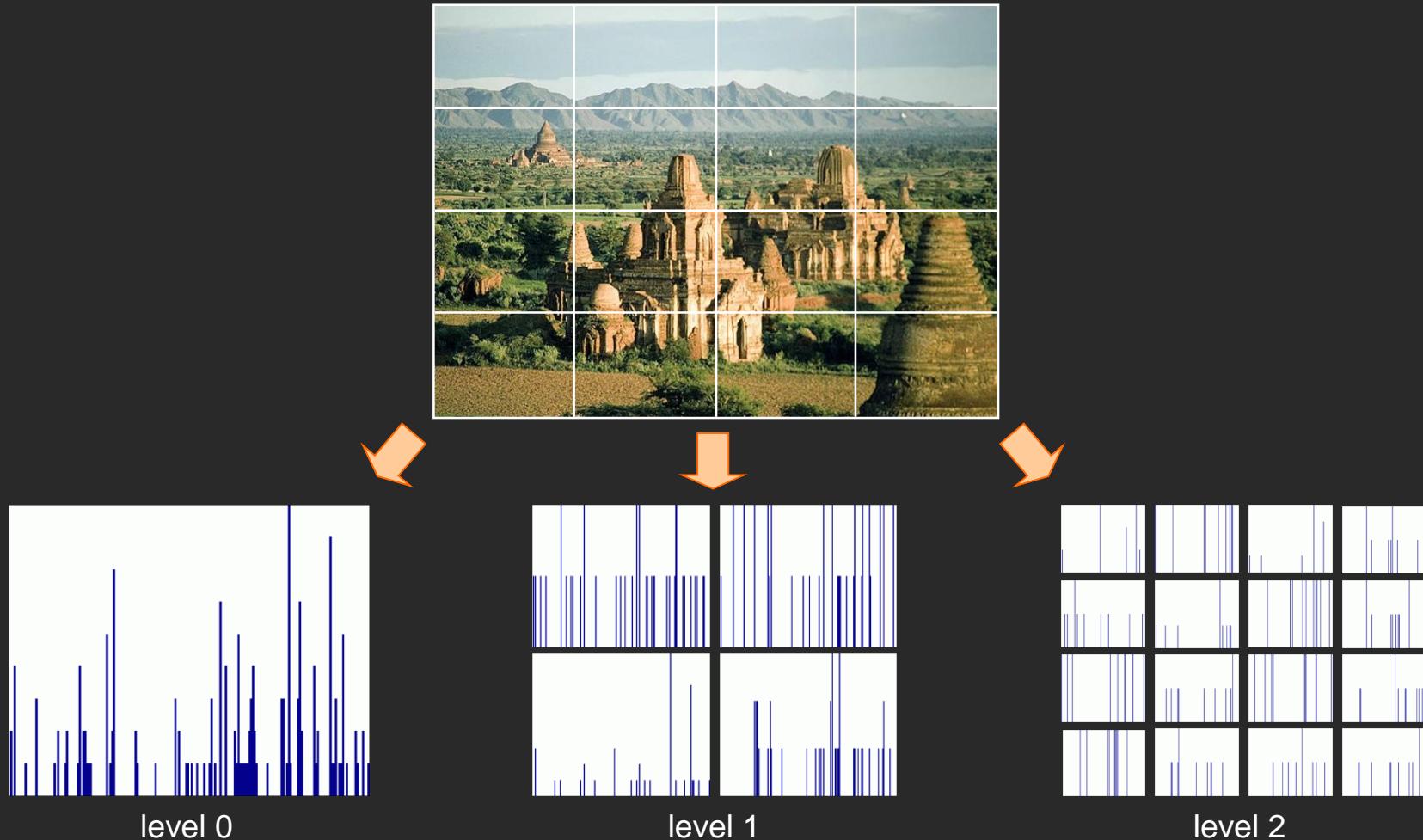
Spatial pyramid representation

- Extension of a bag of features
- Locally orderless representation at several levels of resolution

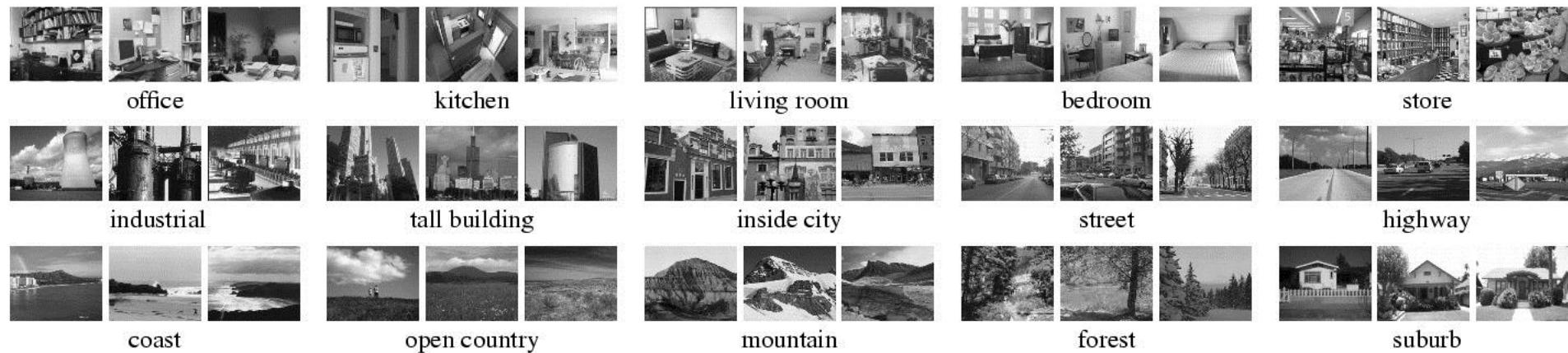


Spatial pyramid representation

- Extension of a bag of features
- Locally orderless representation at several levels of resolution



Scene category dataset

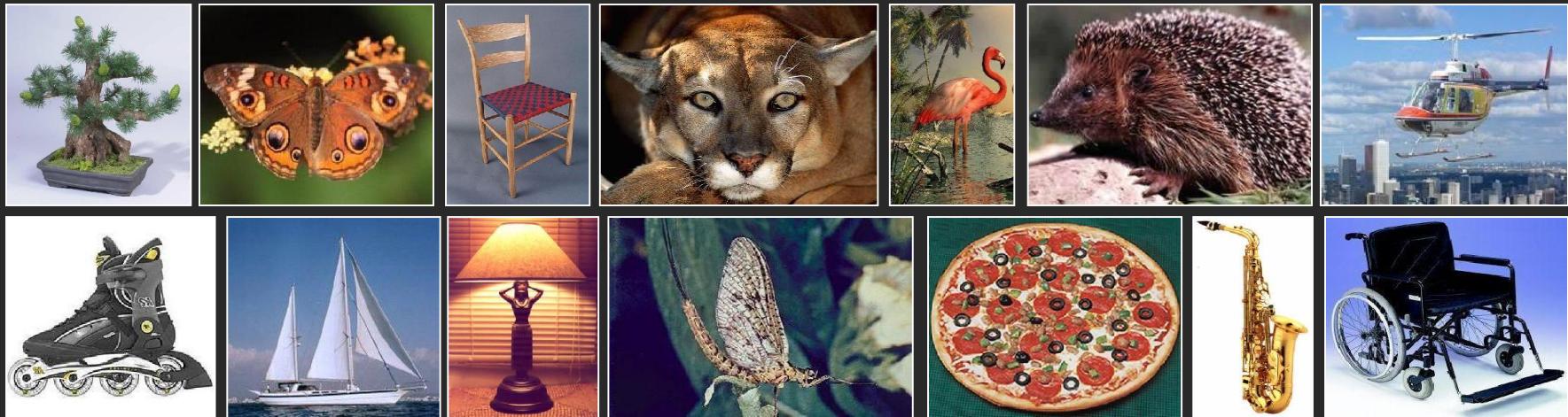


Multi-class classification results
(100 training images per class)

Level	Weak features (vocabulary size: 16)		Strong features (vocabulary size: 200)	
	Single-level	Pyramid	Single-level	Pyramid
0 (1×1)	45.3 ± 0.5		72.2 ± 0.6	
1 (2×2)	53.6 ± 0.3	56.2 ± 0.6	77.9 ± 0.6	79.0 ± 0.5
2 (4×4)	61.7 ± 0.6	64.7 ± 0.7	79.4 ± 0.3	81.1 ± 0.3
3 (8×8)	63.3 ± 0.8	66.8 ± 0.6	77.2 ± 0.4	80.7 ± 0.3

Caltech101 dataset

http://www.vision.caltech.edu/Image_Datasets/Caltech101/Caltech101.html

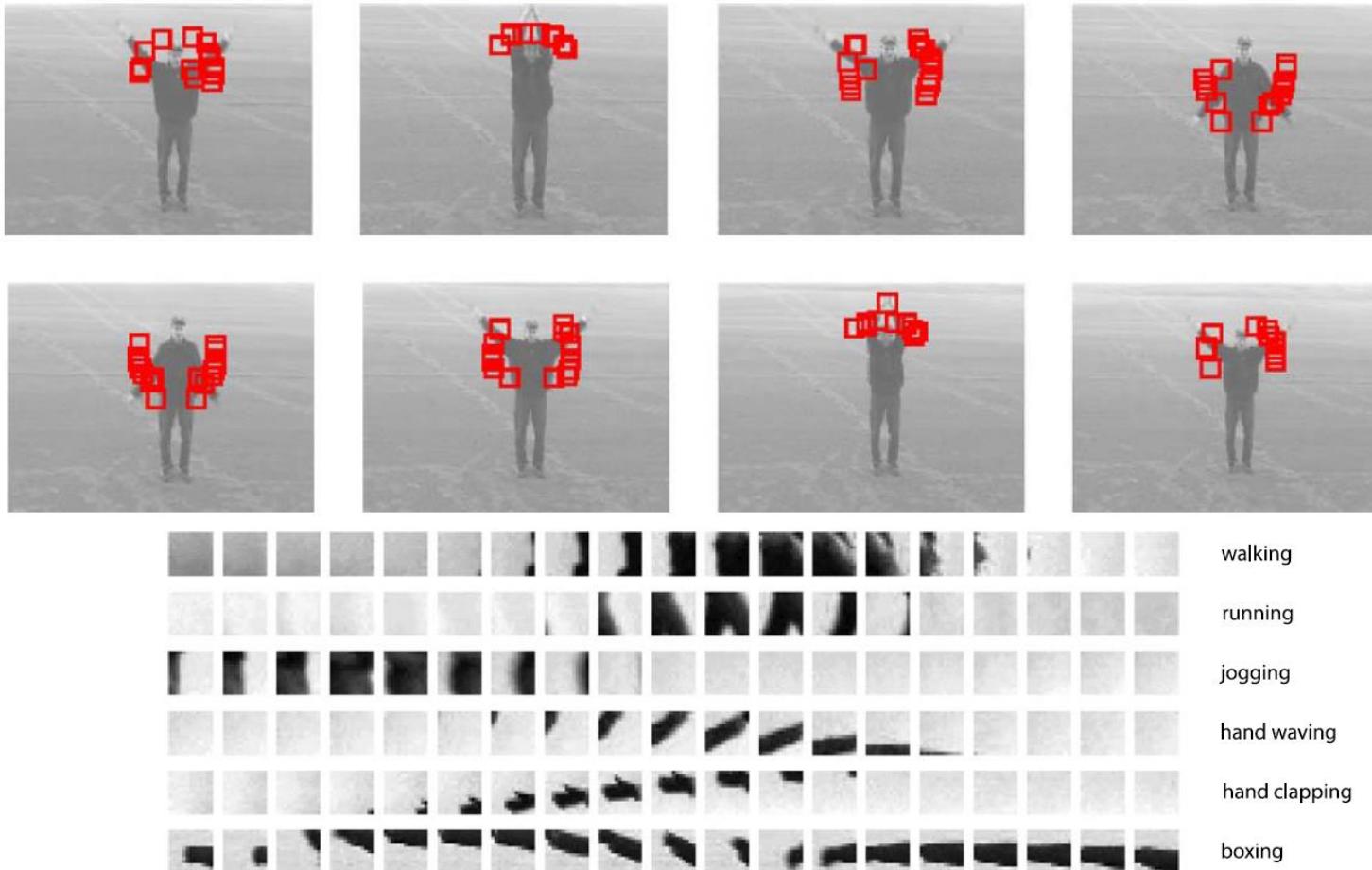


Multi-class classification results (30 training images per class)

	Weak features (16)		Strong features (200)	
Level	Single-level	Pyramid	Single-level	Pyramid
0	15.5 ± 0.9		41.2 ± 1.2	
1	31.4 ± 1.2	32.8 ± 1.3	55.9 ± 0.9	57.0 ± 0.8
2	47.2 ± 1.1	49.3 ± 1.4	63.6 ± 0.9	64.6 ± 0.8
3	52.2 ± 0.8	54.0 ± 1.1	60.3 ± 0.9	64.6 ± 0.7

Bags of features for action recognition

Space-time interest points



Juan Carlos Niebles, Hongcheng Wang and Li Fei-Fei, [Unsupervised Learning of Human Action Categories Using Spatial-Temporal Words](#), IJCV 2008.

History of ideas in recognition

- 1960s – early 1990s: the geometric era
- 1990s: appearance-based models
- Mid-1990s: sliding window approaches
- Late 1990s: local features
- Early 2000s: parts-and-shape models
- Mid-2000s: bags of features
- Present trends: combination of local and global methods, context, *deep learning*