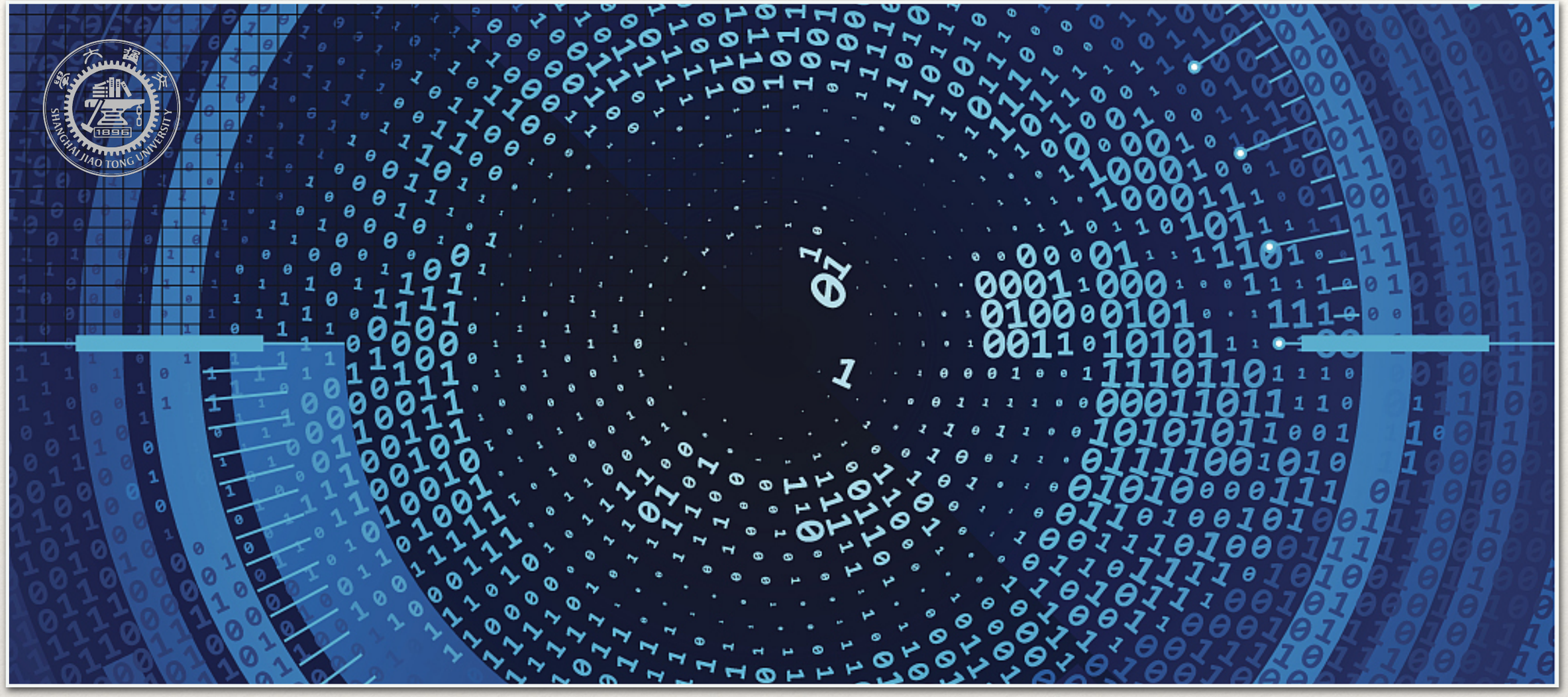


Shanghai Jiao Tong University

Computer Vision

Instructor: Xu Zhao

Class No.: C032703 F032528

Spring 2020

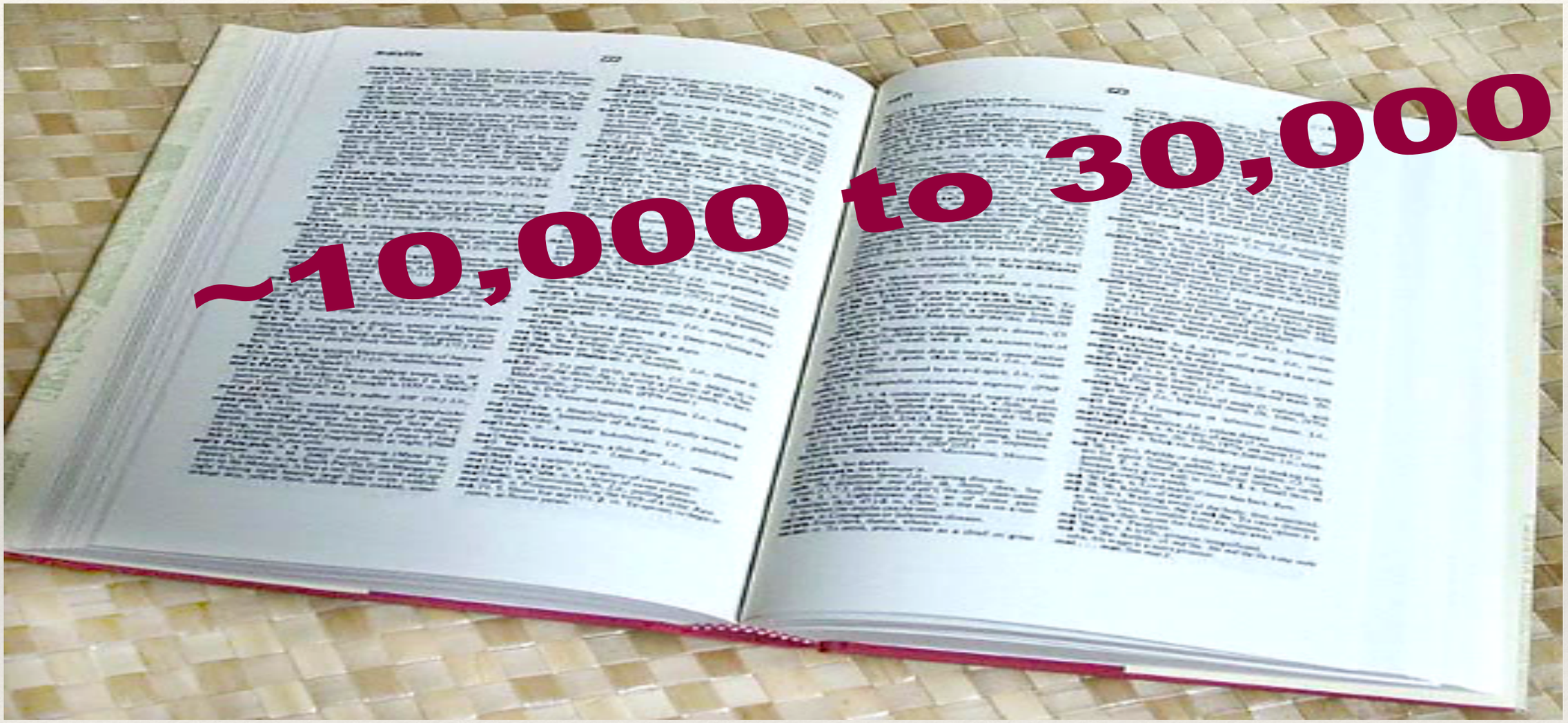
Xu Zhao @ Shanghai Jiao Tong university

Lecture 10: Recognition

Contents

- Recognition: Overview and History
- Machine Learning for Recognition
- Bag of Words Model

How many visual object categories are there?



1500-3000 basic-level nouns, ~10 types per basic-level category

Alternative explanation (Perona): ~1000 names per domain (broad scene category), 20-30 domains

~10,000 to 30,000

OBJECTS

ANIMALS

PLANTS

INANIMATE

…..

VERTEBRATE

NATURAL

MAN-MADE

MAMMALS

BIRDS

TAPIR

BOAR

GROUSE

CAMERA

Recognition tasks

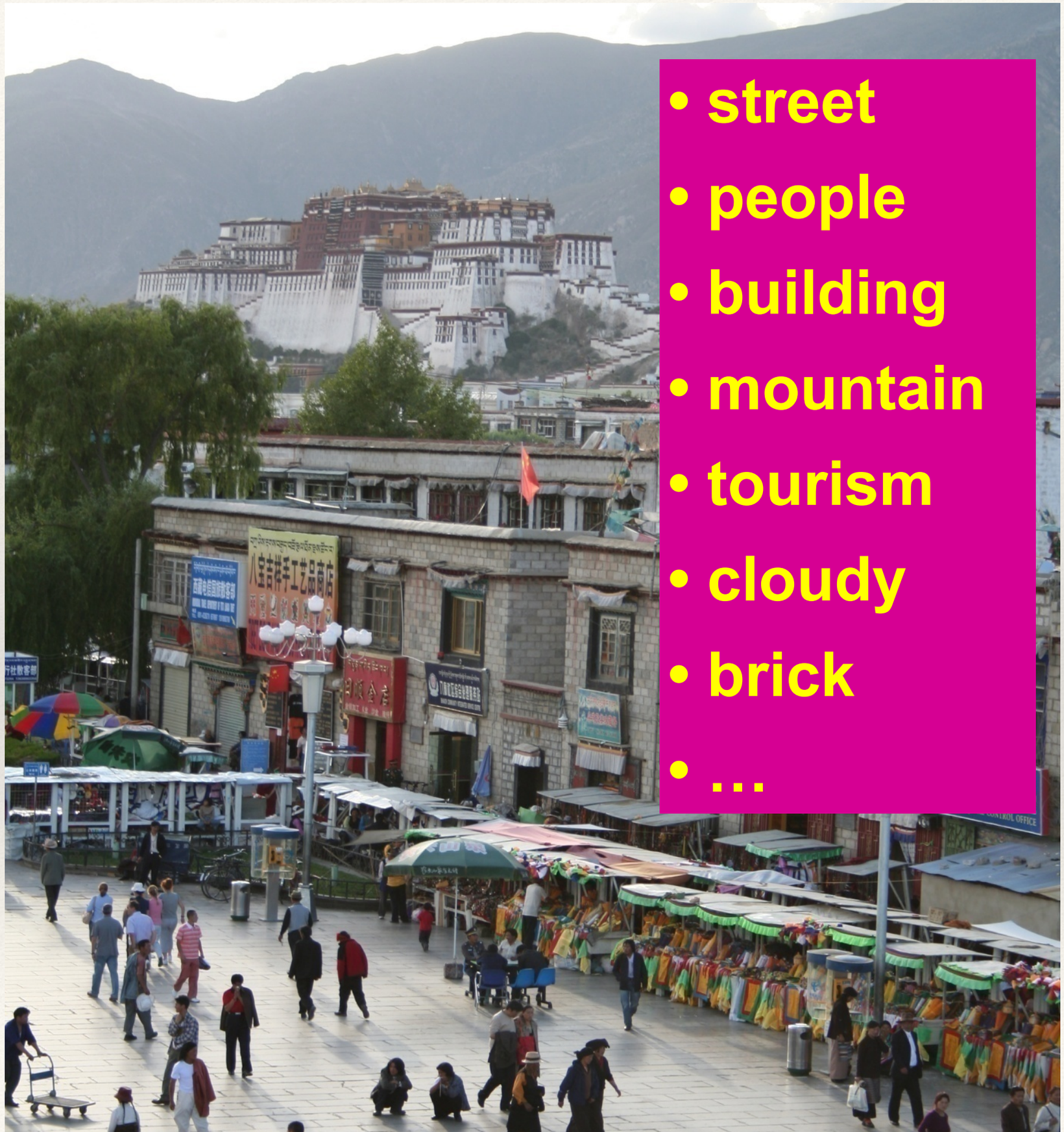
Svetlana Lazebnik

Scene categorization or classification

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

Svetlana Lazebnik

Image annotation / tagging / attributes

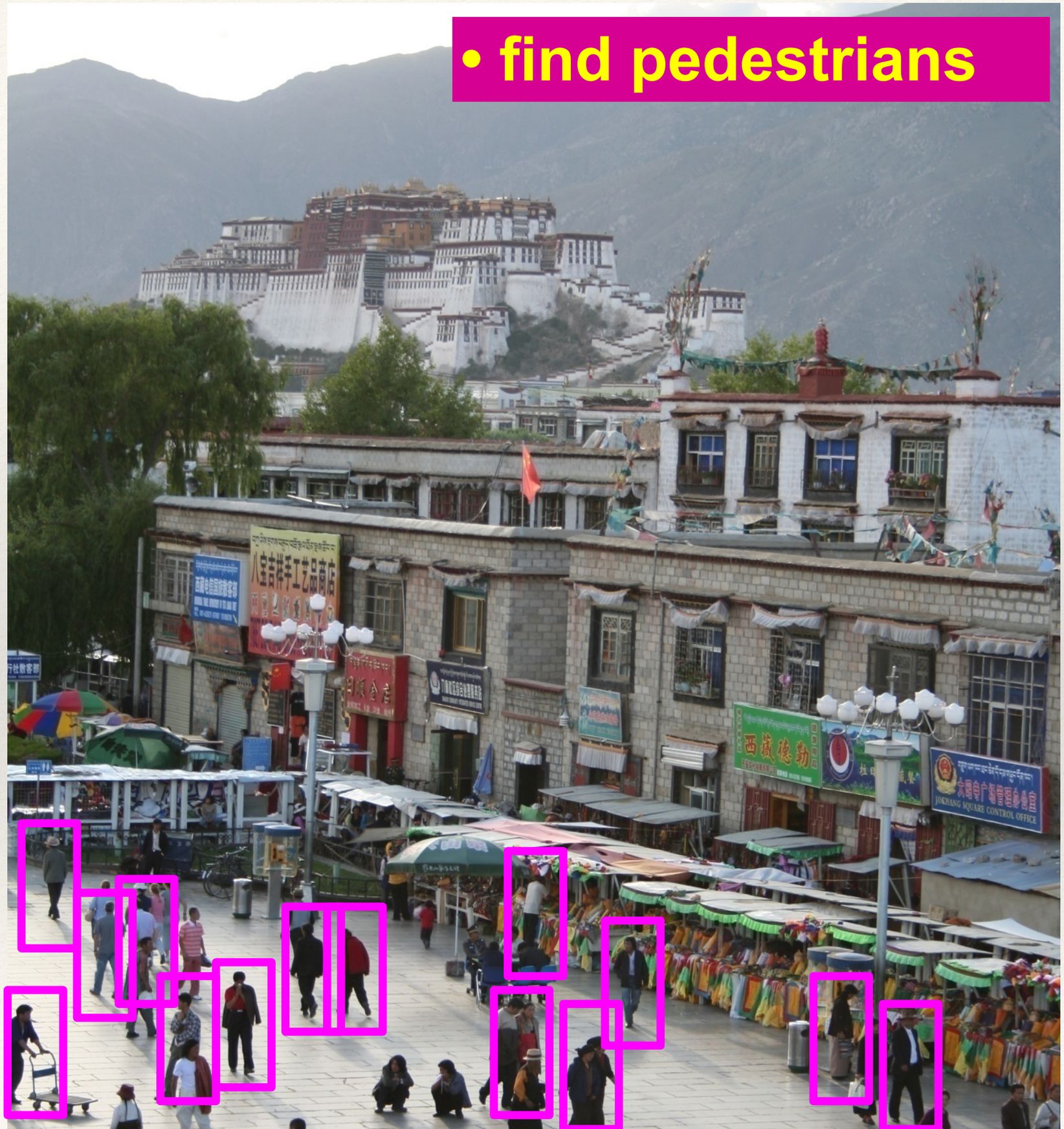


Svetlana Lazebnik

Image parsing / semantic segmentation

Svetlana Lazebnik

Object detection



Svetlana Lazebnik

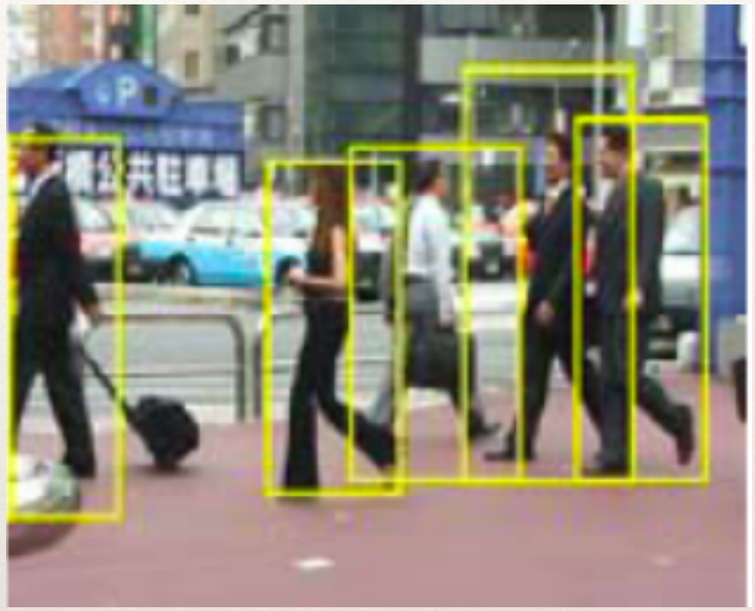
Scene understanding?

Svetlana Lazebnik

Category vs. instance recognition

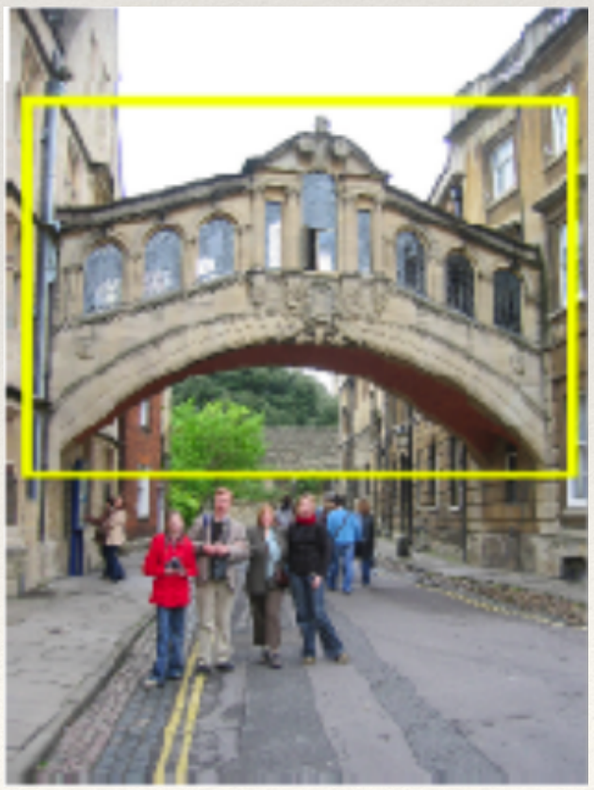
Category:

- Find all the people
- Find all the buildings
- Often within a single image
- Often ‘sliding window’

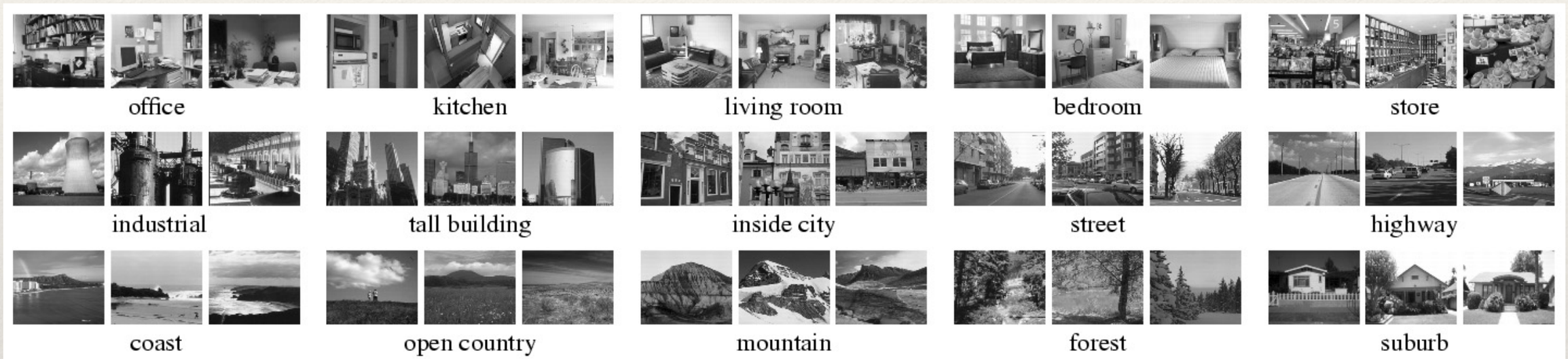


Instance:

- Is this face Trump?
- Find this specific famous building
- Often within a database of images

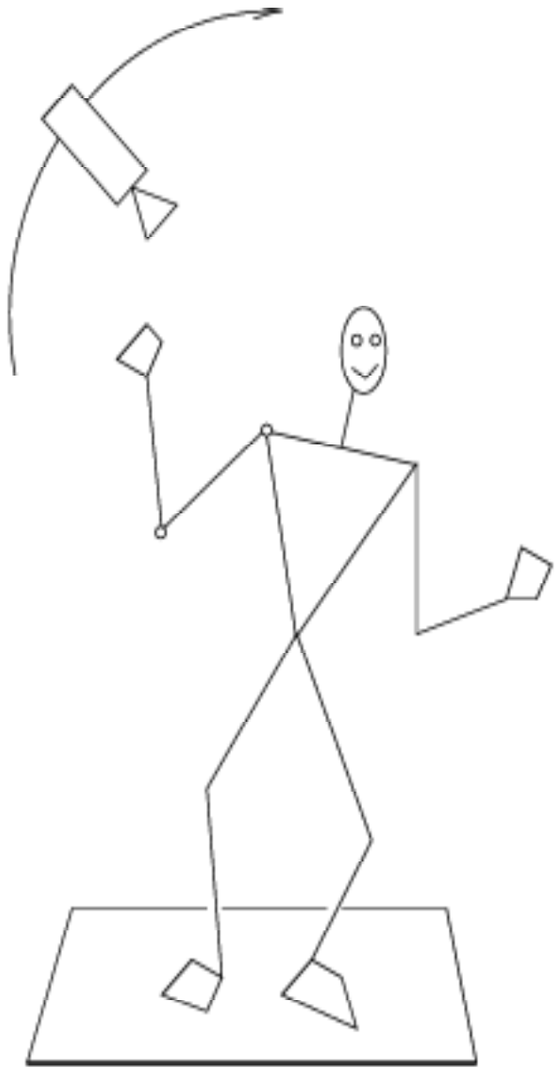


Scene recognition dataset



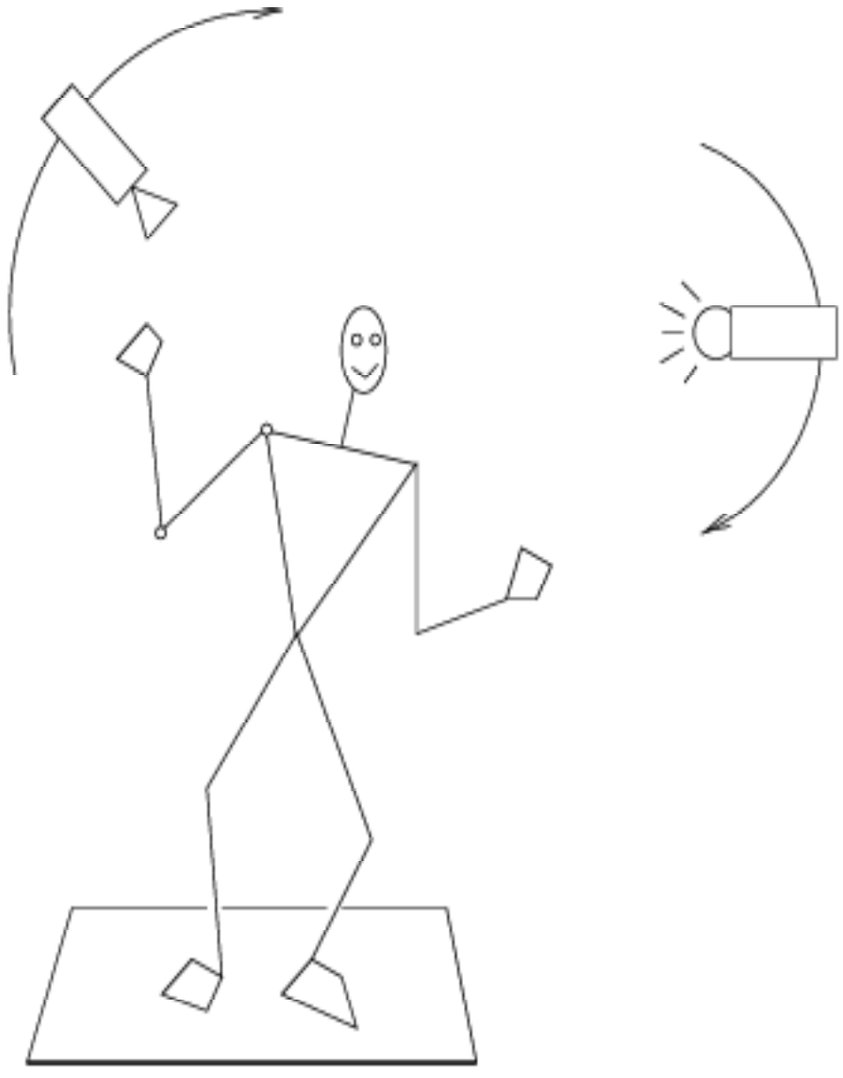
Instance or category?

Recognition is all about modeling variability

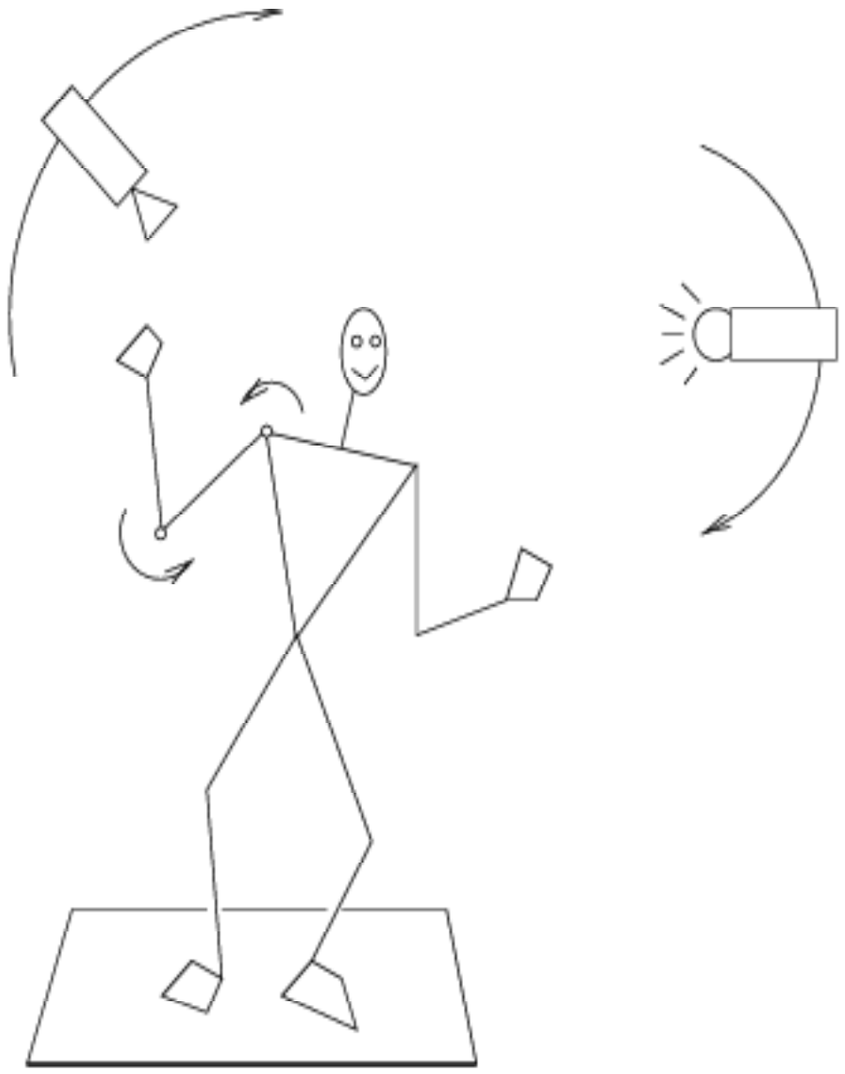


Variability: Camera position

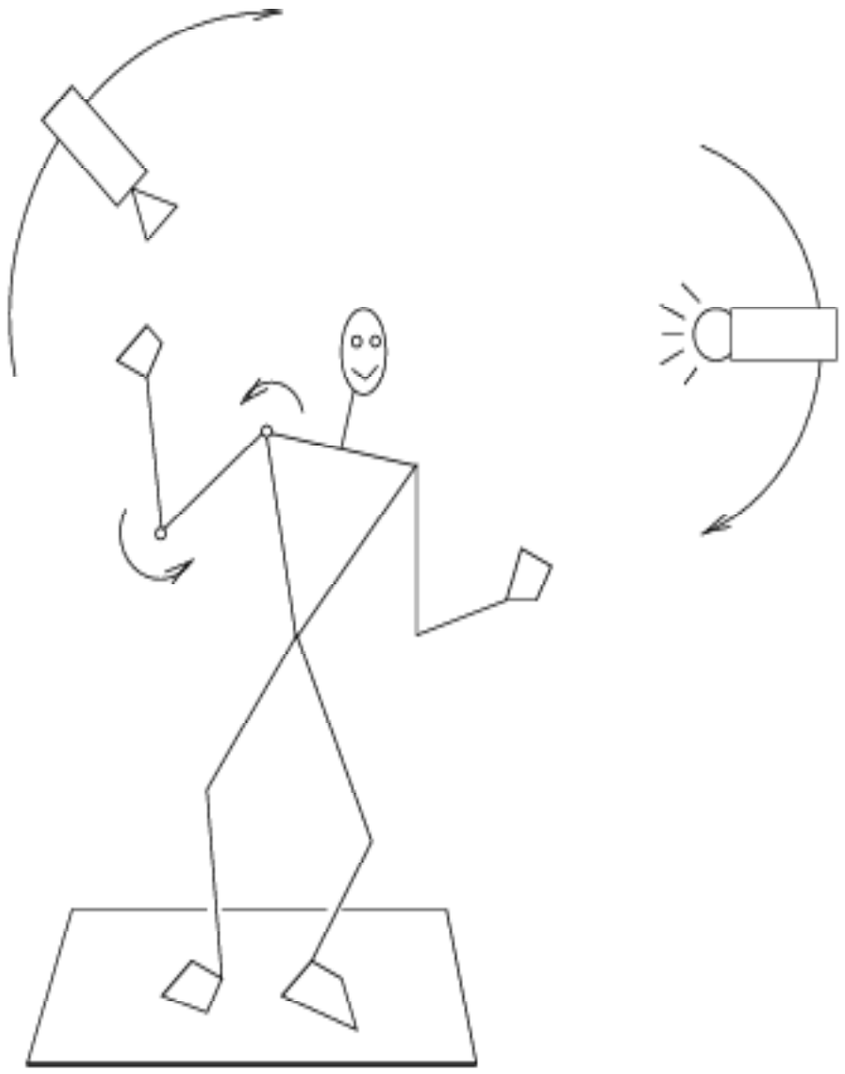
Recognition is all about modeling variability



Variability: Camera position
Illumination



Variability: Camera position
Illumination
Pose/shape parameters



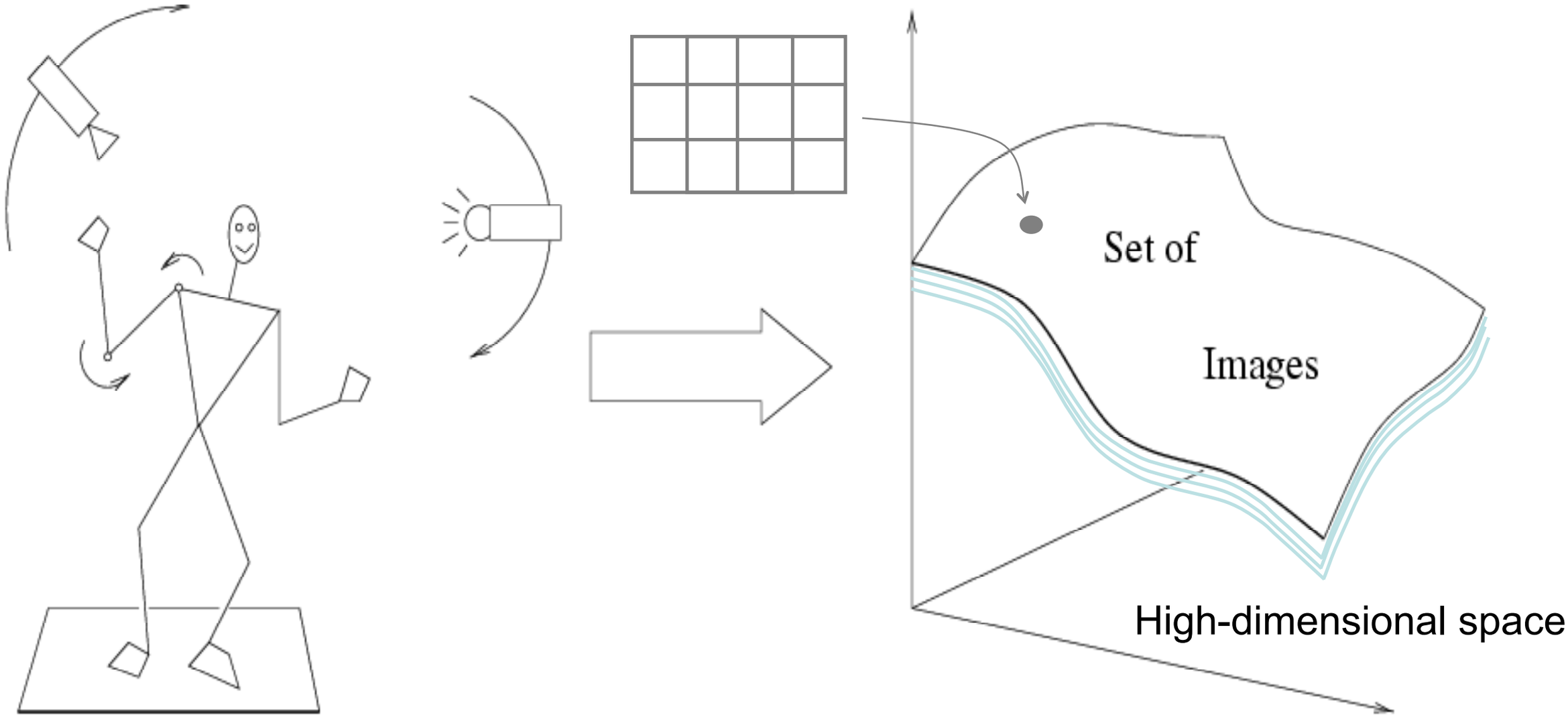
Variability: Camera position

Illumination

Pose/shape parameters

Within-class variations?

Within-class variations



Variability: Camera position

Illumination

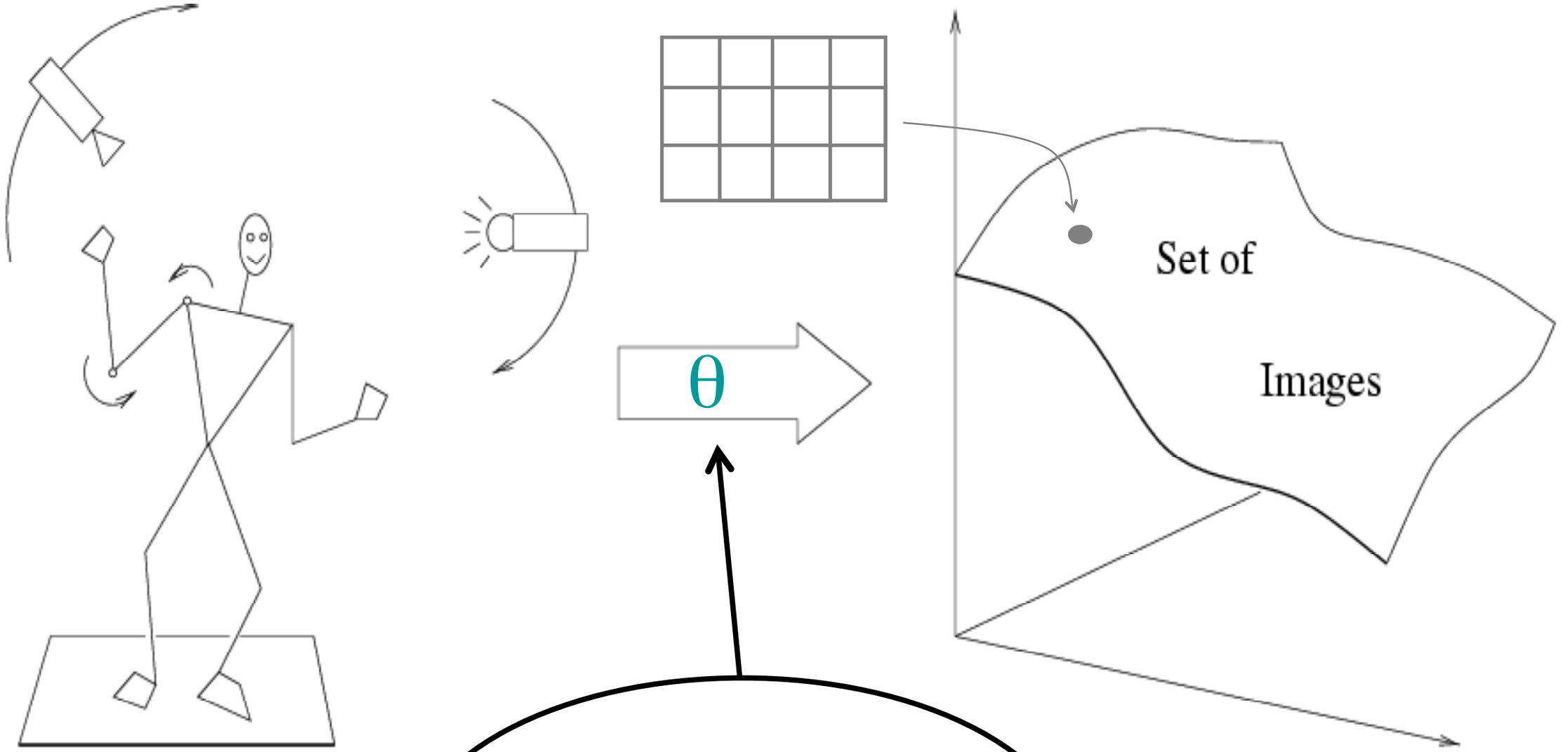
Pose/shape parameters

Within-class variation

History of ideas in recognition

- 1960s – early 1990s: the geometric era

No digital cameras!
Slow compute!



Variability:

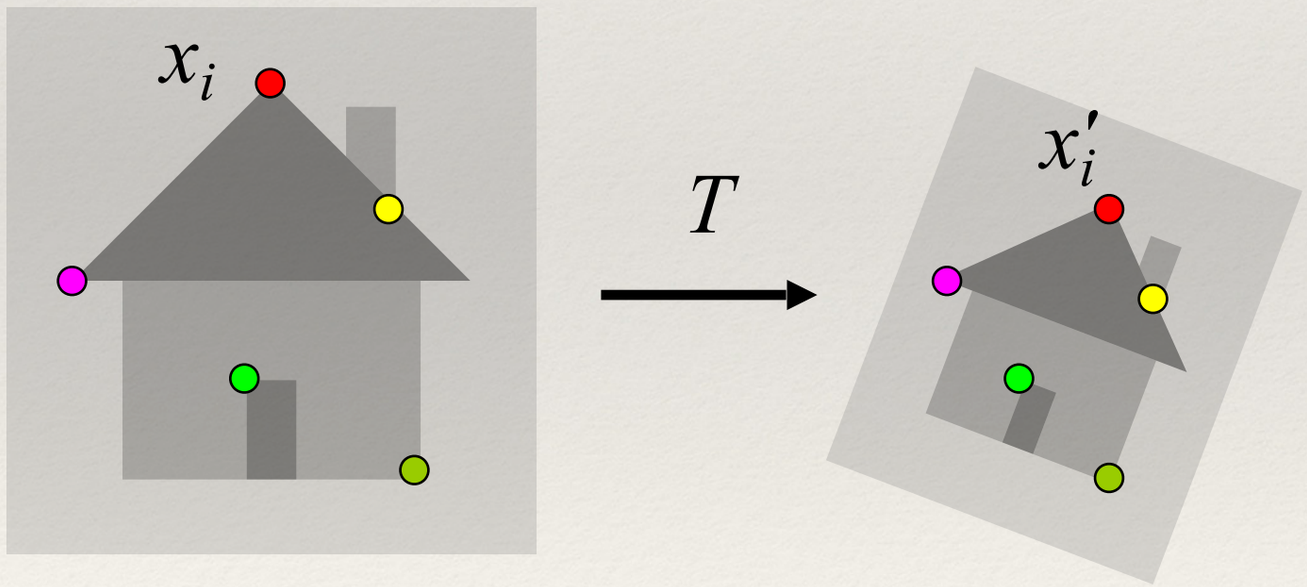
Camera position
Illumination

Shape is known

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

Alignment

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

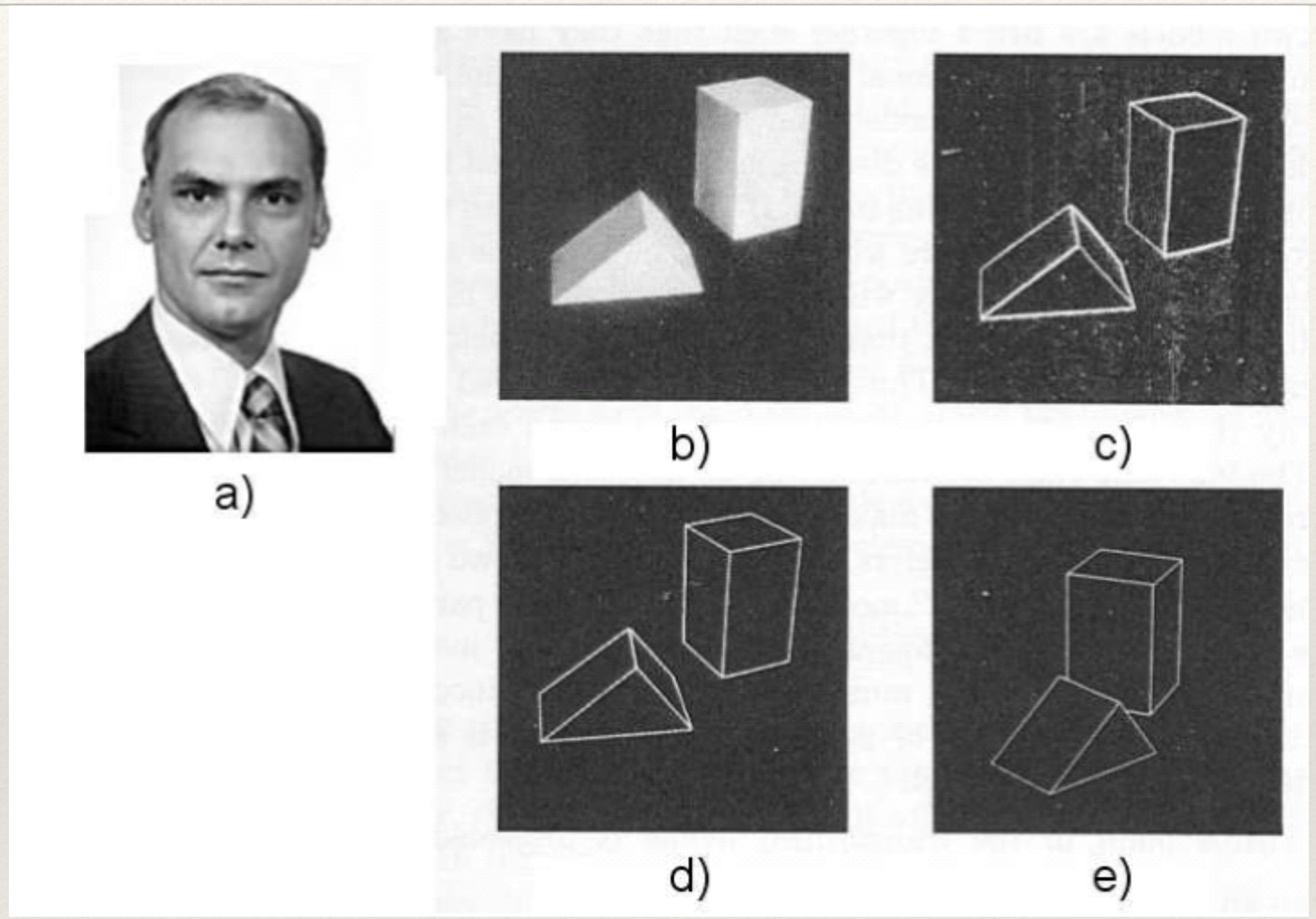


Find transformation T that minimizes

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

Svetlana Lazebnik

Recognition as an alignment problem: Block world

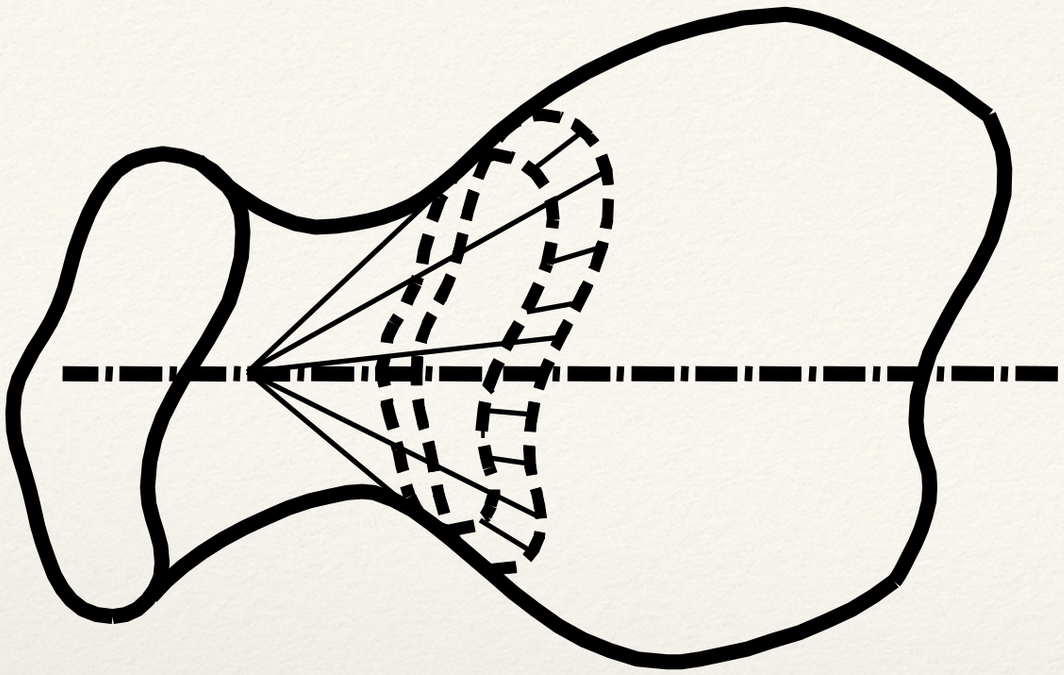


L. G. Roberts
Machine Perception of Three Dimensional Solids,
Ph.D. thesis, MIT Department of Electrical Engineering, 1963.

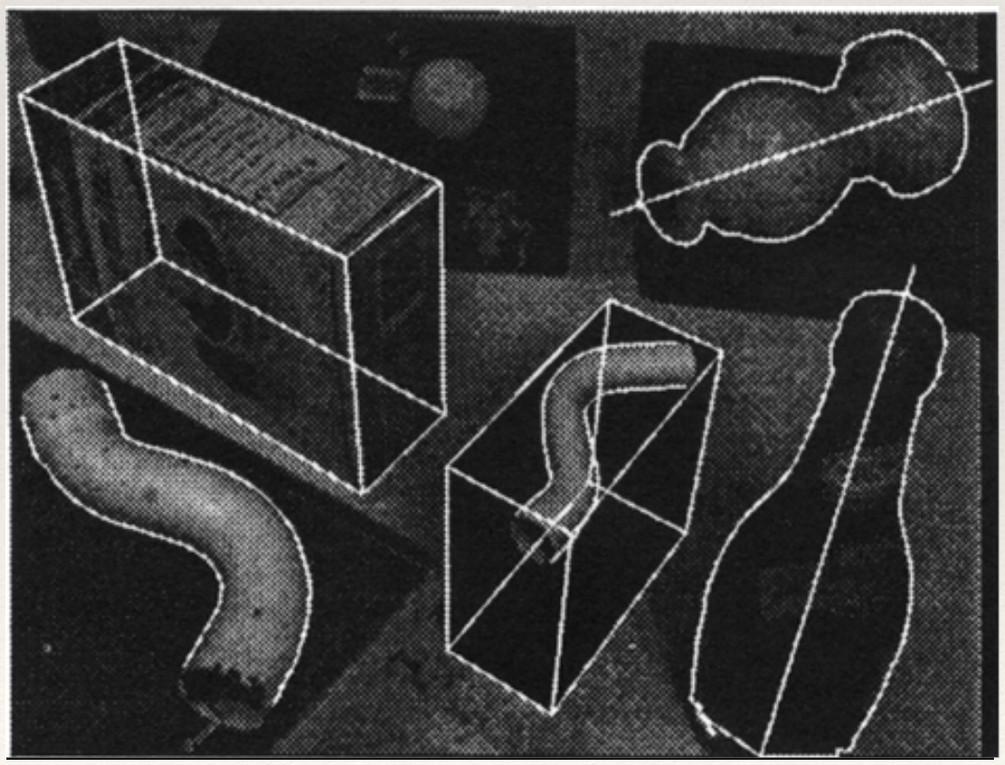
Fig. 1. A system for recognizing 3-d polyhedral scenes. a) L.G. Roberts. b)A blocks world scene. c)Detected edges using a 2x2 gradient operator. d) A 3-d polyhedral description of the scene, formed automatically from the single image. e) The 3-d scene displayed with a viewpoint different from the original image to demonstrate its accuracy and completeness. (b) - e) are taken from [64] with permission MIT Press.)

J. Mundy, Object Recognition in the Geometric Era: a Retrospective, 2006

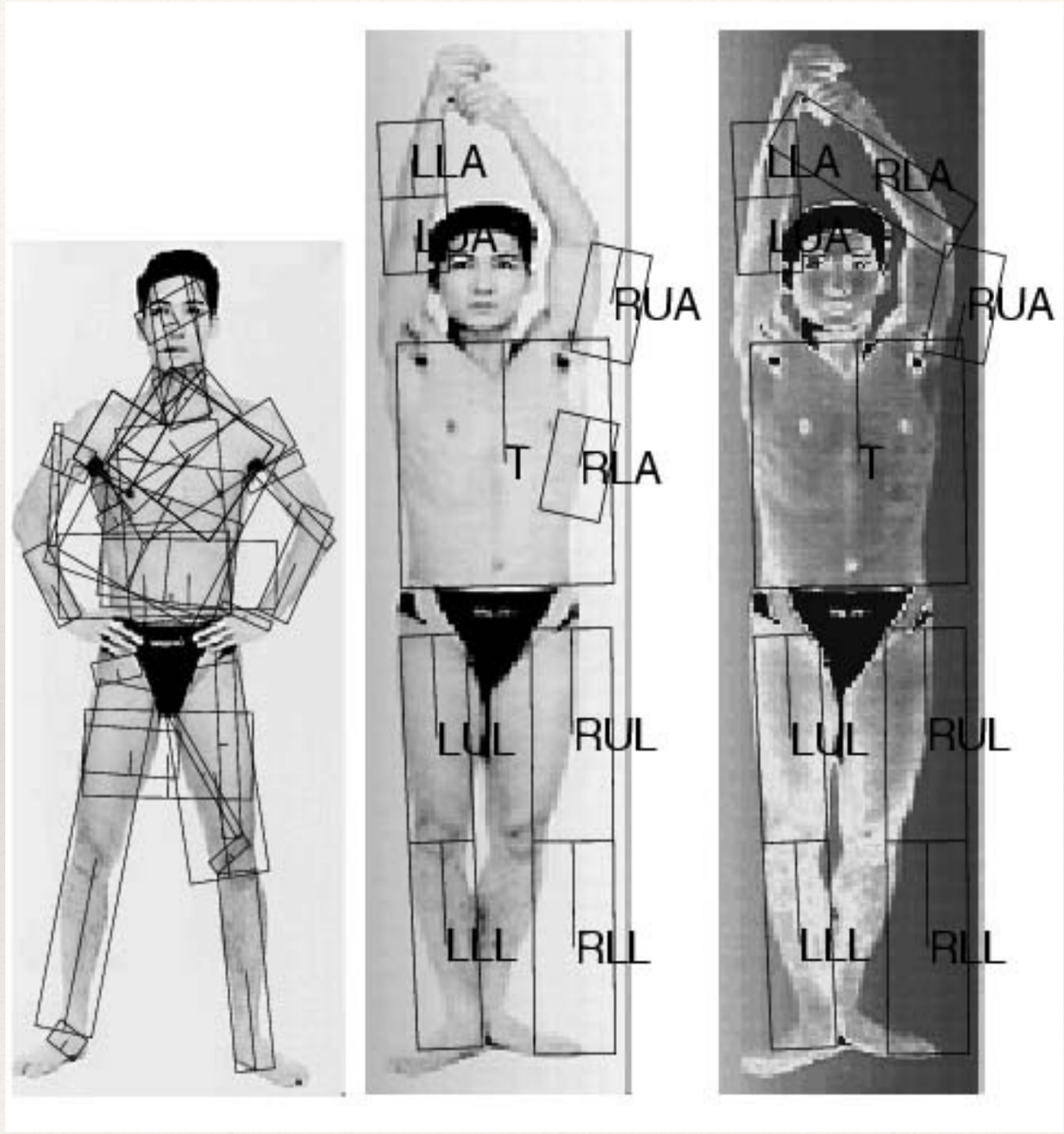
General shape primitives?



Generalized cylinders
Ponce et al. (1989)



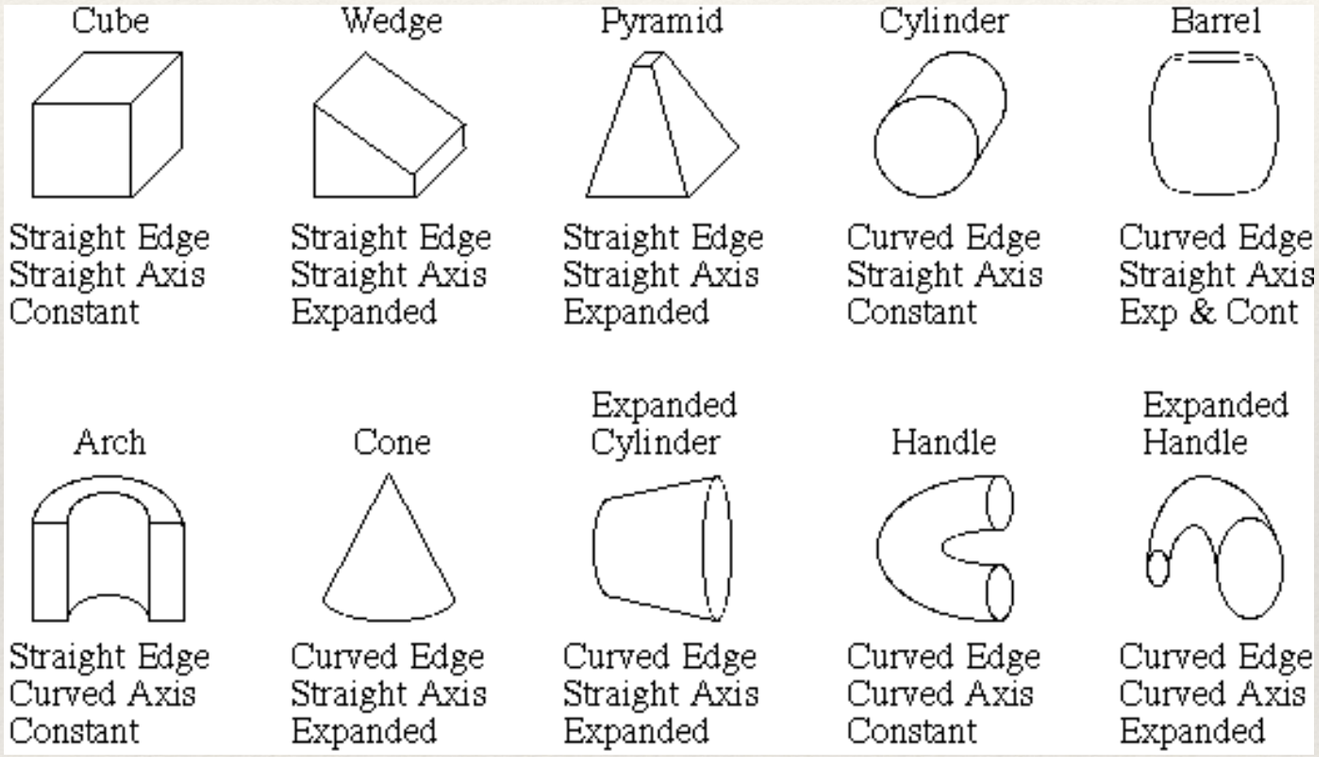
Zisserman et al. (1995)



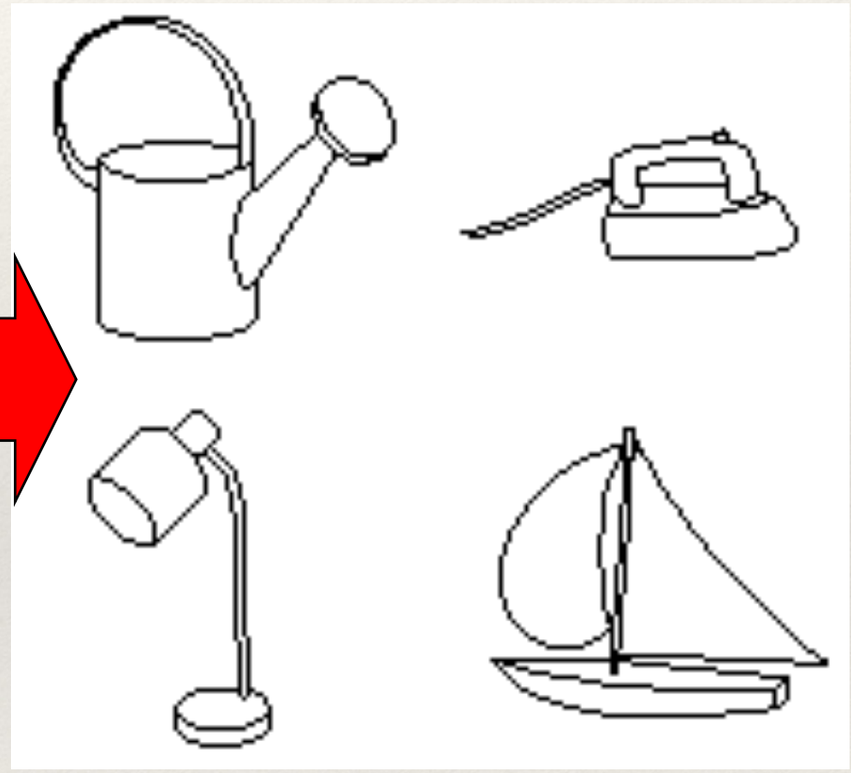
Forsyth (2000)

Recognition by components
Biederman (1987)

Primitives (geons)



Objects



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

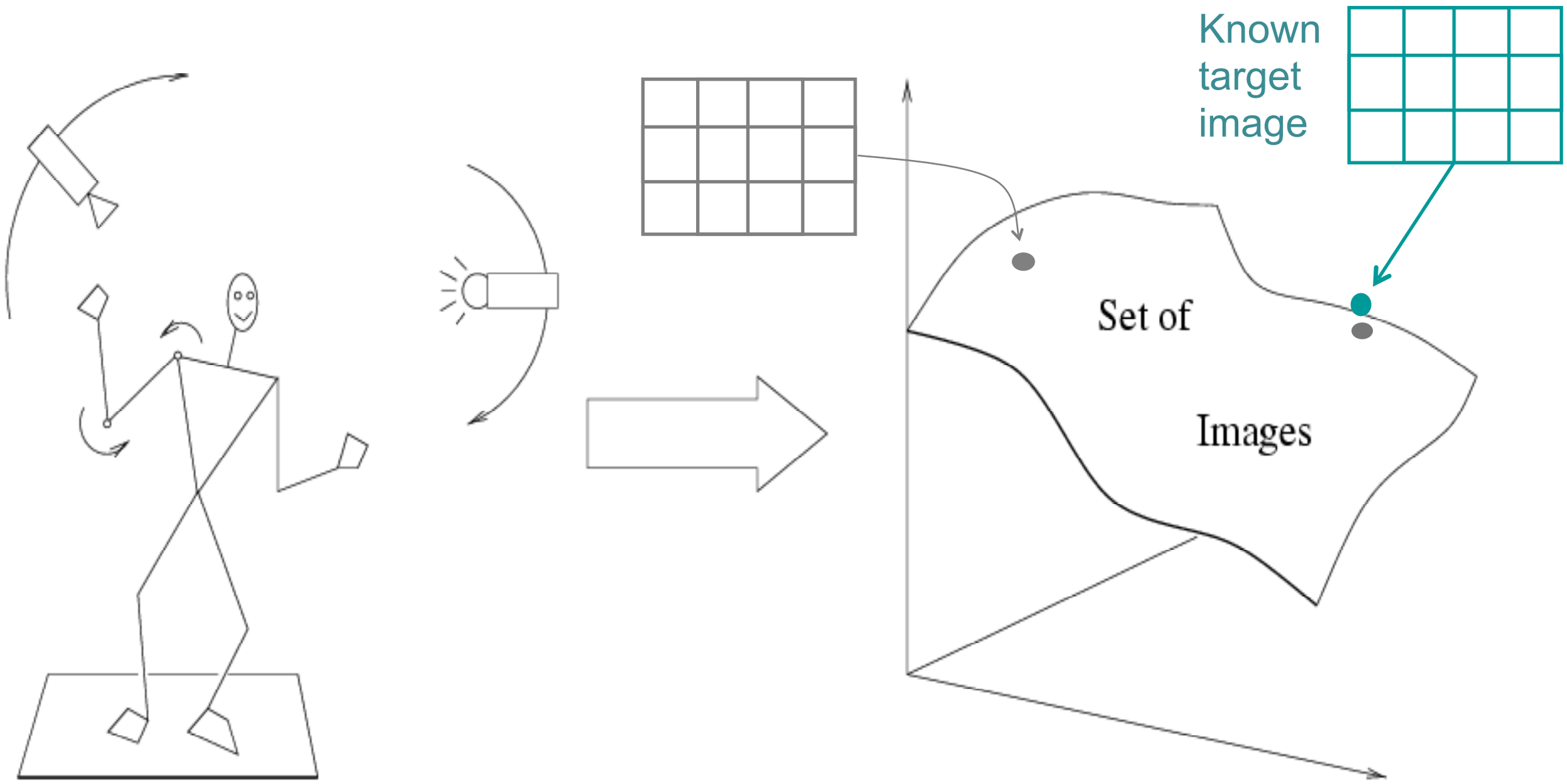
History of ideas in recognition

- 1960s – early 1990s: the geometric era

  No digital cameras!
  Slow compute!

- 1990s: appearance-based models

  Slow compute!

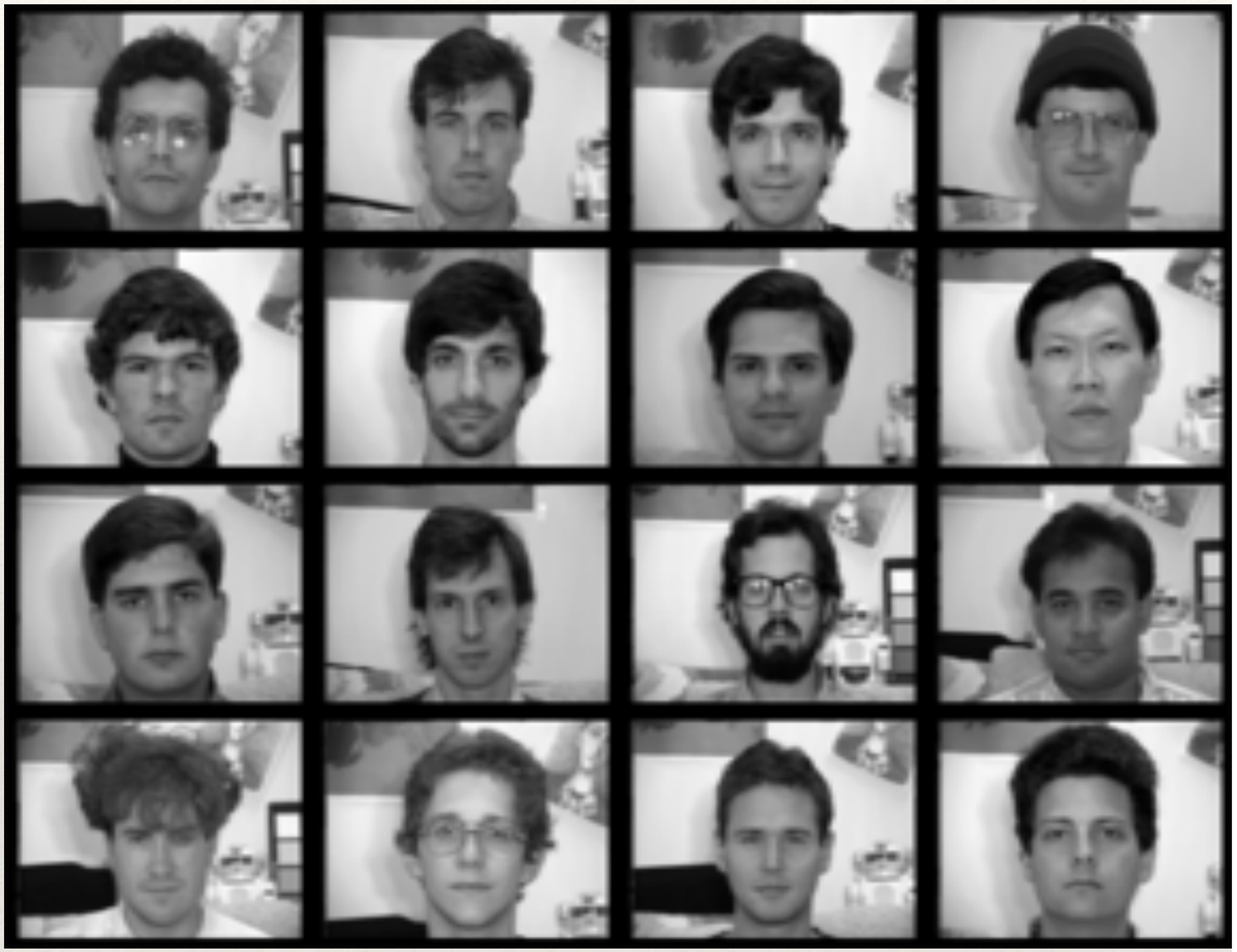


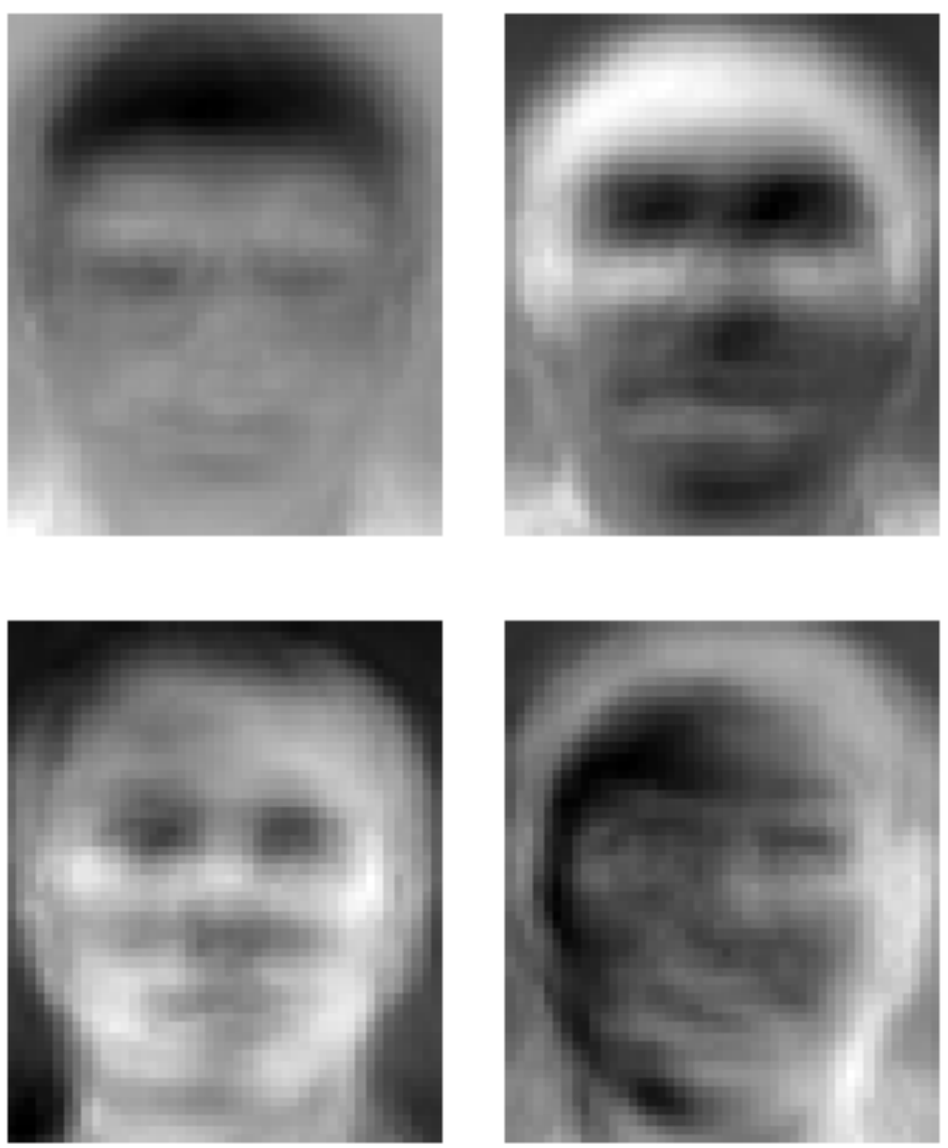
Empirical models of image variability

Appearance-based techniques

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

Eigenfaces (Turk & Pentland, 1991)

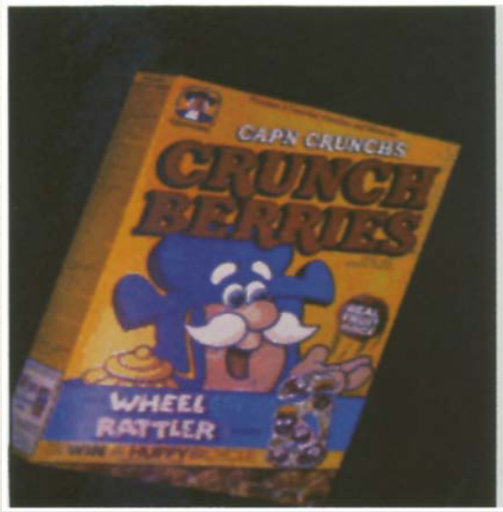


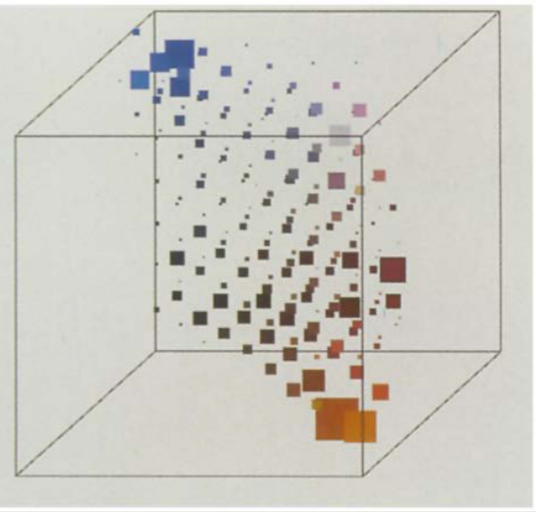


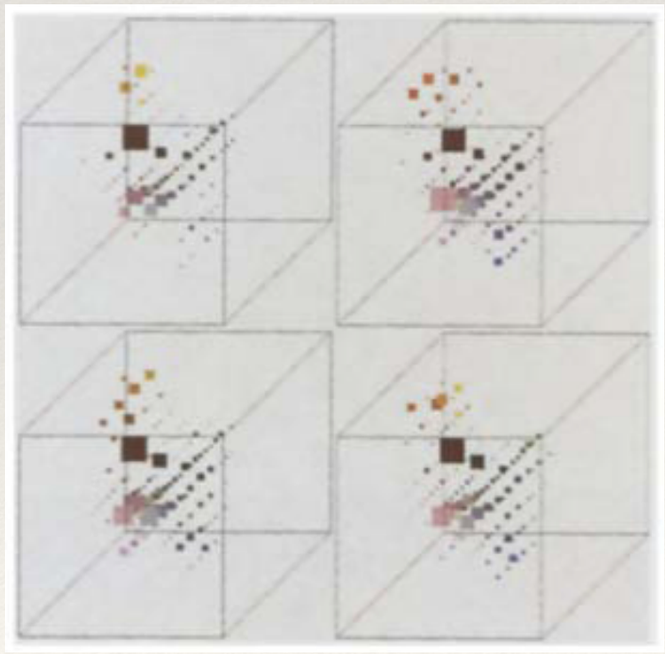
Experimental	Correct/Unknown Recognition Percentage		
Condition	Lighting	Orientation	Scale
Forced classification	96/0	85/0	64/0
Forced 100% accuracy	100/19	100/39	100/60
Forced 20% unknown rate	100/20	94/20	74/20

Svetlana Lazebnik

Color Histograms

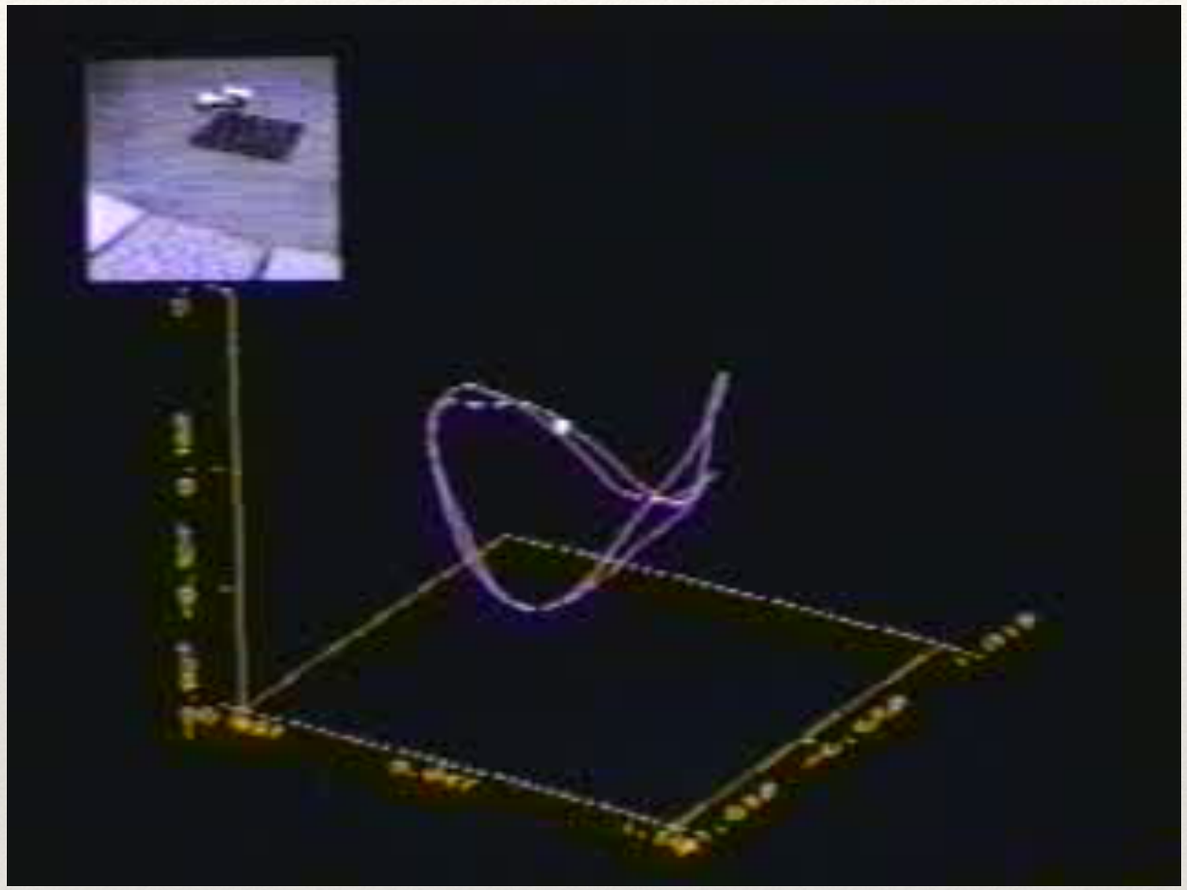






Swain and Ballard, Color Indexing, IJCV 1991.

Appearance manifolds



H. Murase and S. Nayar, Visual learning and recognition of 3-d objects from appearance, IJCV 1995

Limitations of global appearance models

- Requires global registration of patterns
- Not robust to clutter, occlusion, geometric transformations

Svetlana Lazebnik

History of ideas in recognition

- 1960s – early 1990s: the geometric era

  No digital cameras!
  Slow compute!

- 1990s: appearance-based models

  Slow compute!

- 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

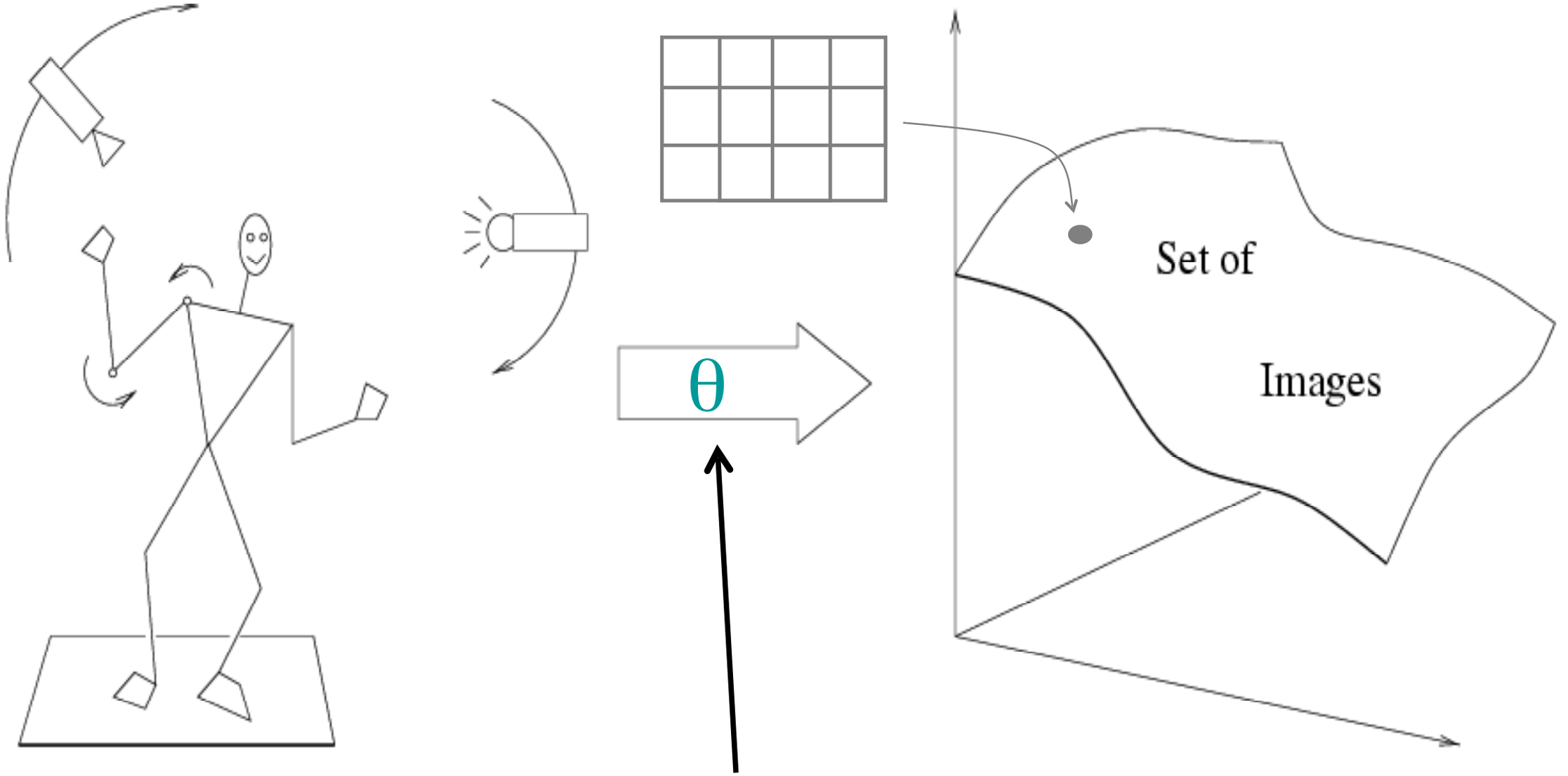
No digital cameras!
Slow compute!

- 1990s: appearance-based models

Slow compute!

- Mid-1990s: sliding window approaches
- Late 1990s: local features

Svetlana Lazebnik



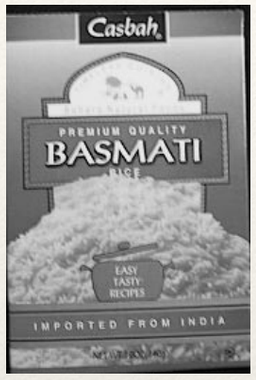
Variability:

Camera position
Illumination

Shape is partially known

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

Local features for object instance recognition



D. Lowe (1999, 2004)

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

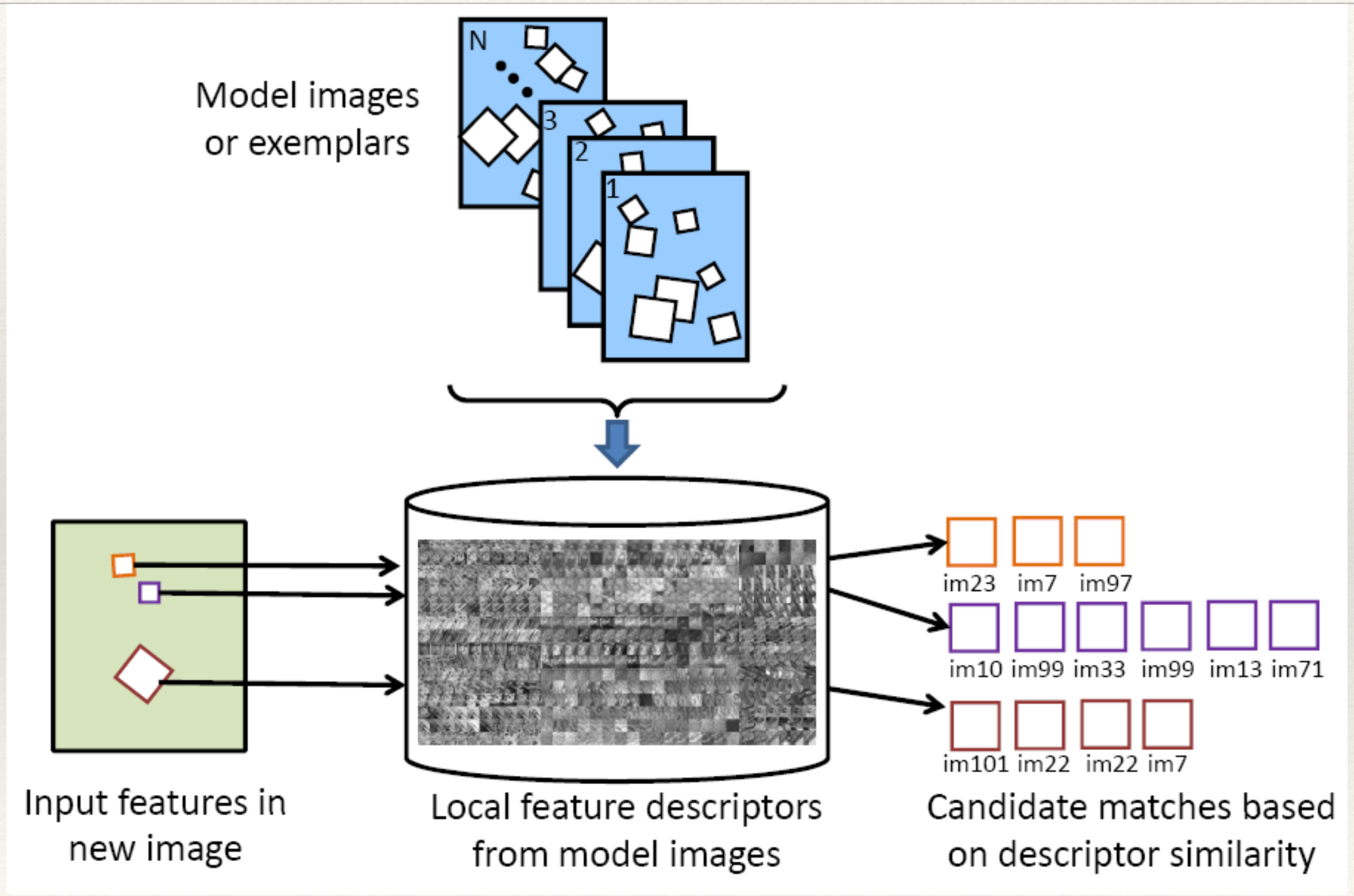


Image credit: K. Grauman and B. Leibe

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

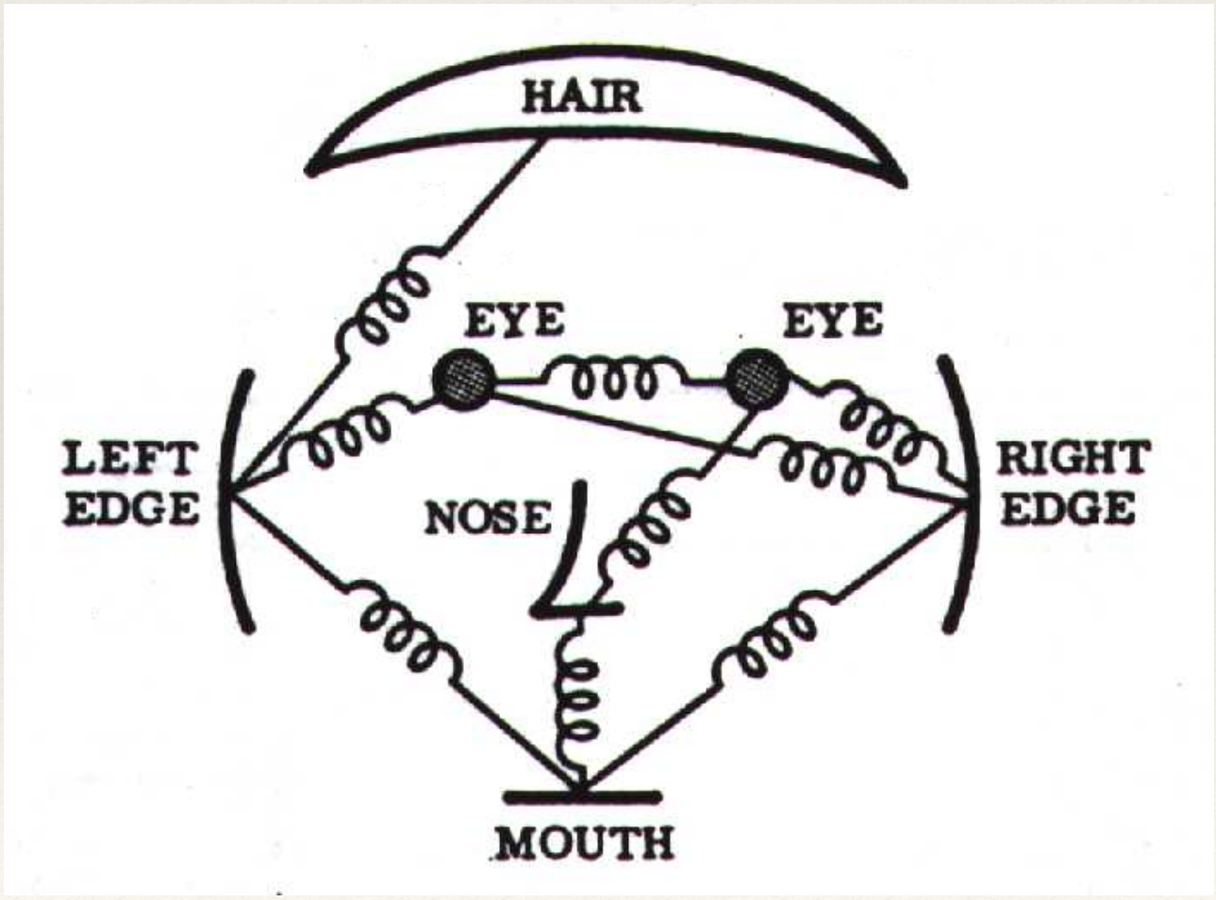
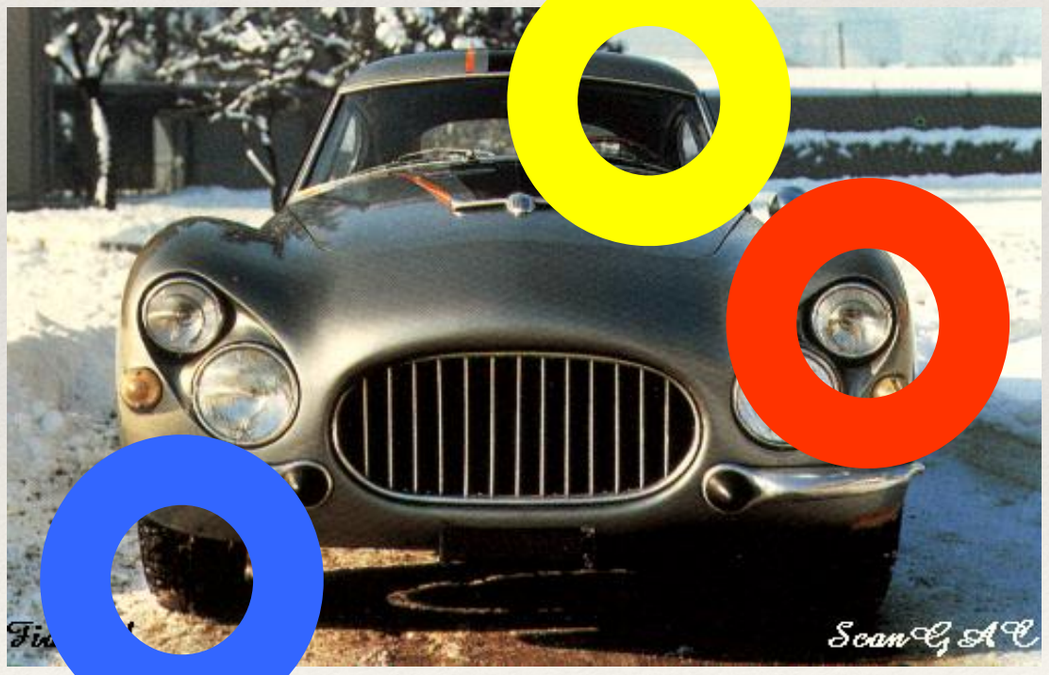


Figure from [Fischler & Elschlager 73]

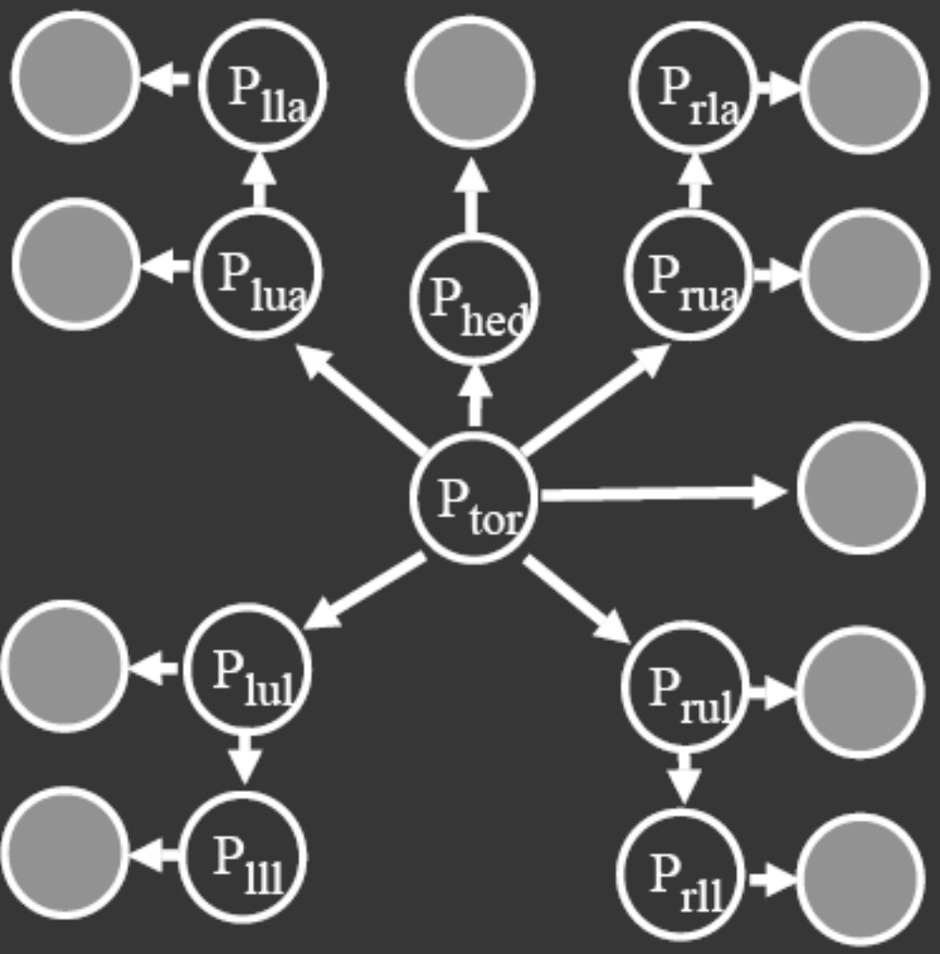
Constellation models

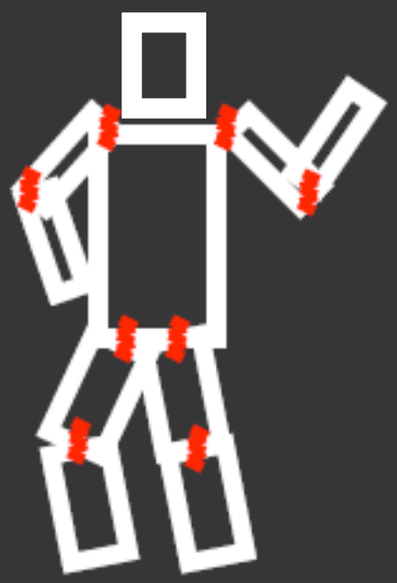


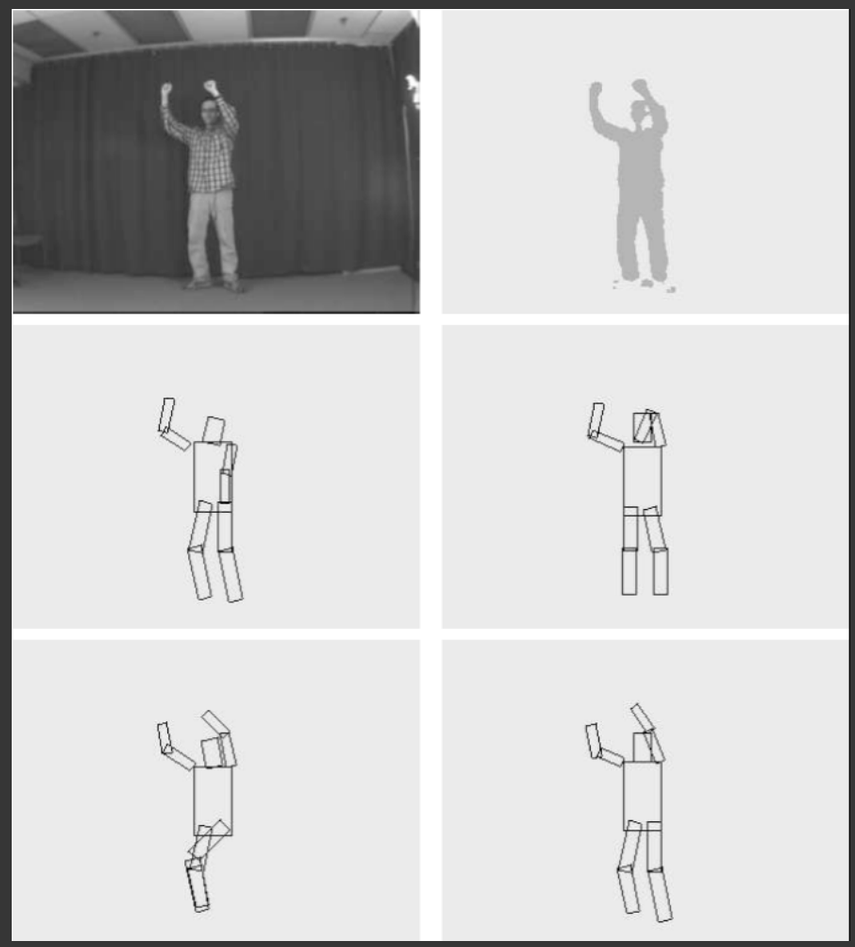
Weber, Welling & Perona (2000), Fergus, Perona & Zisserman (2003)

Pictorial structure model

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





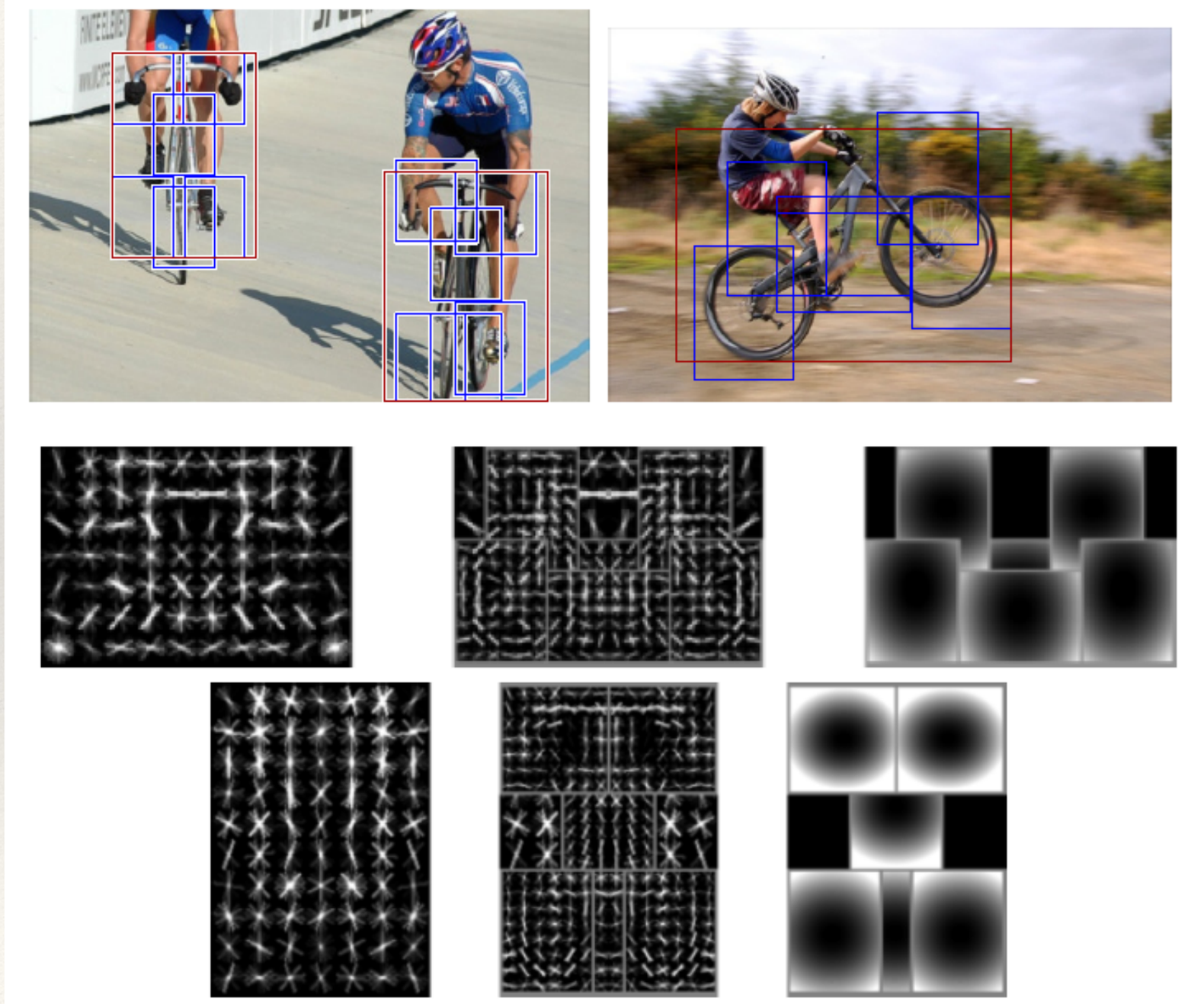


$$\Pr(P_{tor}, P_{arm}, \ldots | Im) \propto \prod_{i,j} \Pr(P_i | P_j) \prod_i \Pr(Im(P_i))$$

part geometry

part appearance

Discriminatively trained part-based models



P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan, PAMI 2009,

“Object Detection with Discriminatively Trained Part-Based Models”

History of ideas in recognition

- 1960s – early 1990s: the geometric era

No digital cameras!
Slow compute!

- 1990s: appearance-based models

Slow compute!

- Mid-1990s: sliding window approaches
- Late 1990s: local features
- Early 2000s: parts-and-shape models
- Mid-2000s: bags of features (next!)

Early GPU compute.

Svetlana Lazebnik

History of ideas in recognition

- 1960s – early 1990s: the geometric era — No digital cameras! Slow compute!
- 1990s: appearance-based models — Slow compute!
- Mid-1990s: sliding window approaches
- Late 1990s: local features
- Early 2000s: parts-and-shape models
- Mid-2000s: bags of features (next!) — Early GPU compute.
- *Present trends:*
  Combined local and global methods, context, deep learning — GPU/cloud compute.

Recognition Issues

- How to bridge the gap between feature and label?
- How to summarize the content of an entire image? How to gauge overall similarity?
- How large should the vocabulary be? How to perform quantization efficiently?
- How to score the retrieval results?
- How might we add more spatial verification?

The machine learning framework

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

f() = "apple"

f() = "tomato"

f() = "cow"

The machine learning framework

$$f(\mathbf{x}) = y$$

Prediction function (or *classifier*)

Image feature

Output (label)

- **Training:** Given a *training set* of labeled examples: $\{(\mathbf{x}_1,y_1), \ldots, (\mathbf{x}_N,y_N)\}$
  Estimate the prediction function f by minimizing the prediction error on the training set.
- **Testing:** Apply f to a unseen *test example* $\mathbf{x}_u$ and output the predicted value $y_u = f(\mathbf{x}_u)$ to *classify* $\mathbf{x}_u$.

Slide credit: L. Lazebnik

Dataset

Training Images

- Train classifier

Validation Images

- Measure error
- Tune model hyperparameters

Testing Images

- Secret labels
- Measure error

Random train/validate splits = cross validation

Training

Images

Labels

Image Features

Training

Trained classifier

Testing

Image *not in training set*

Image Features

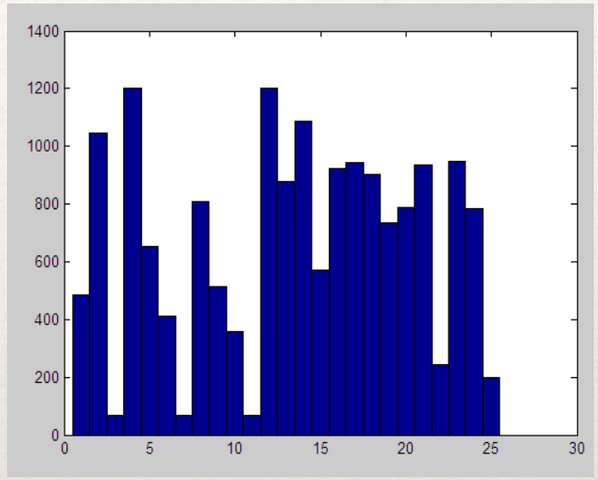
Apply classifier

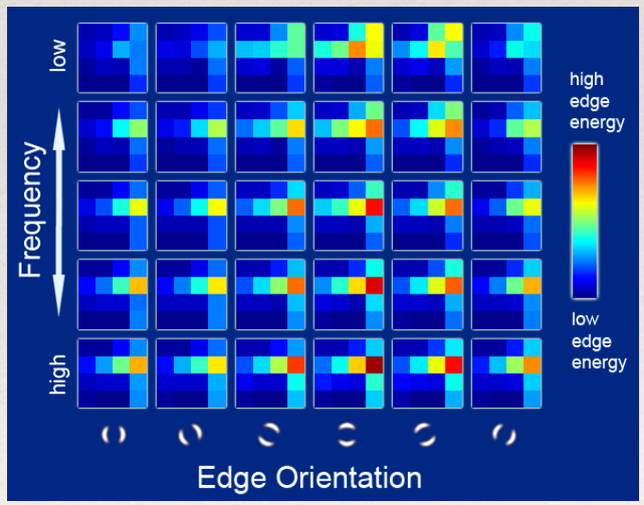
Prediction

Slide credit: D. Hoiem and L. Lazebnik

Features

- Raw pixels
- Histograms
- Templates
- SIFT descriptors
  - GIST
  - ORB
  - HOG….





L. Lazebnik

Training

Images

Labels

Image Features

Training

Trained classifier

Testing

Image *not in training set*

Image Features

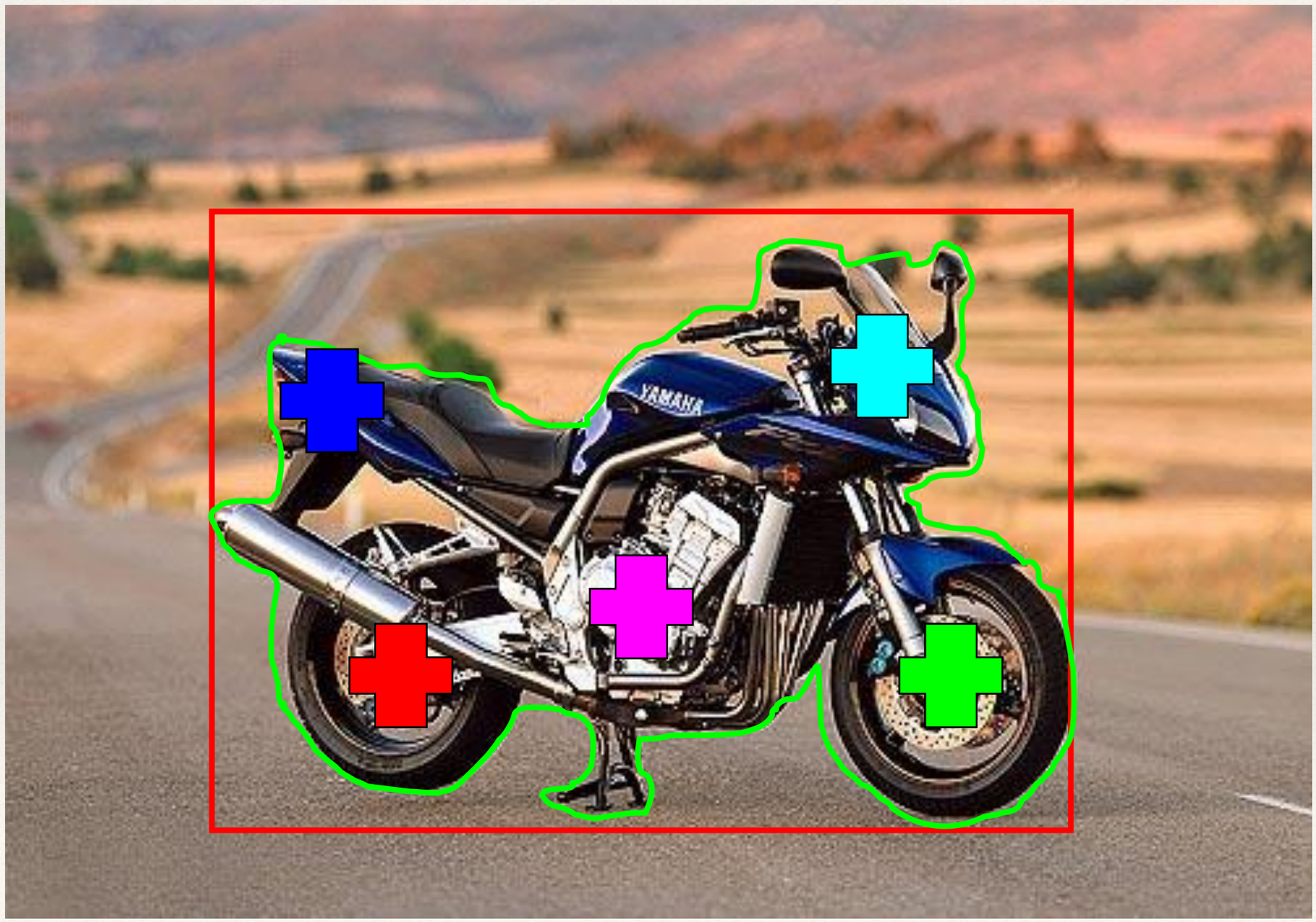
Apply classifier

Prediction

Slide credit: D. Hoiem and L. Lazebnik

Recognition task and supervision

- Images in the training set must be annotated with the "correct answer" that the model is expected to produce



L. Lazebnik

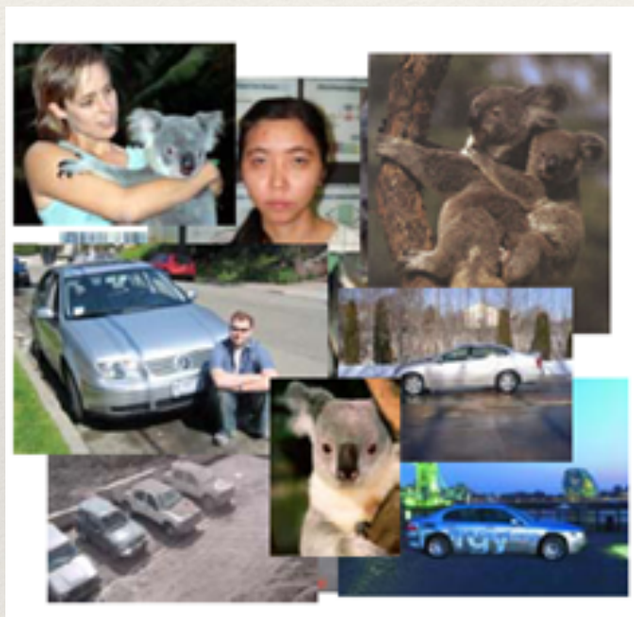
Spectrum of supervision

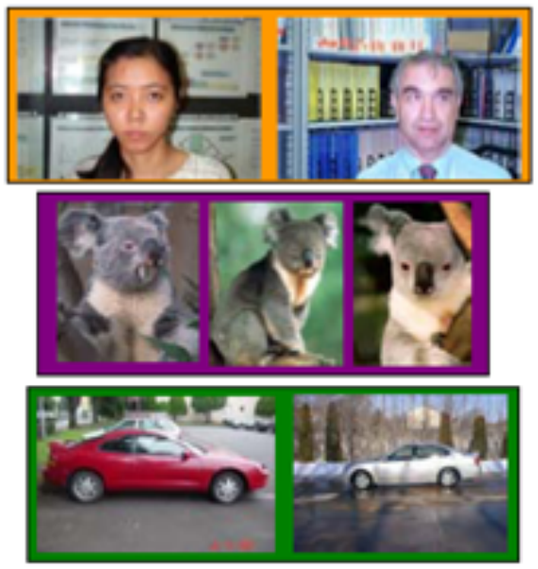
Less

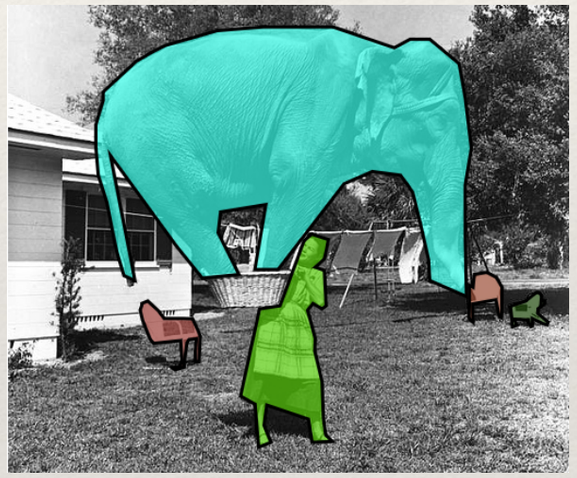
More

E.G., ImageNet

E.G., MS Coco







Unsupervised

“Weakly” supervised

Fully supervised

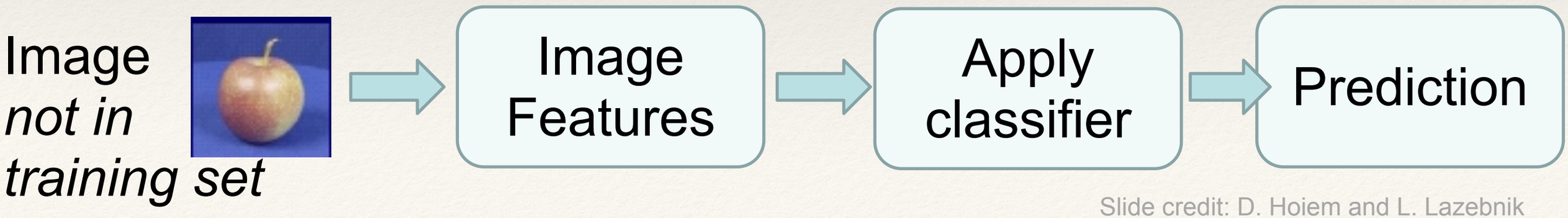
Fuzzy; definition depends on task

‘Semi-supervised’: small partial labeling

Lazebnik

Training

Testing



Slide credit: D. Hoiem and L. Lazebnik

The machine learning framework

$$f(\mathbf{x}) = y$$

Prediction function (or *classifier*)

Image feature

Output (label)

- **Training:** Given a *training set* of labeled examples: $\{(\mathbf{x}_1,y_1), \ldots, (\mathbf{x}_N,y_N)\}$
  Estimate the prediction function f by minimizing the prediction error on the training set.
- **Testing:** Apply f to a unseen *test example* $\mathbf{x}_u$ and output the predicted value $y_u = f(\mathbf{x}_u)$ to *classify* $\mathbf{x}_u$.

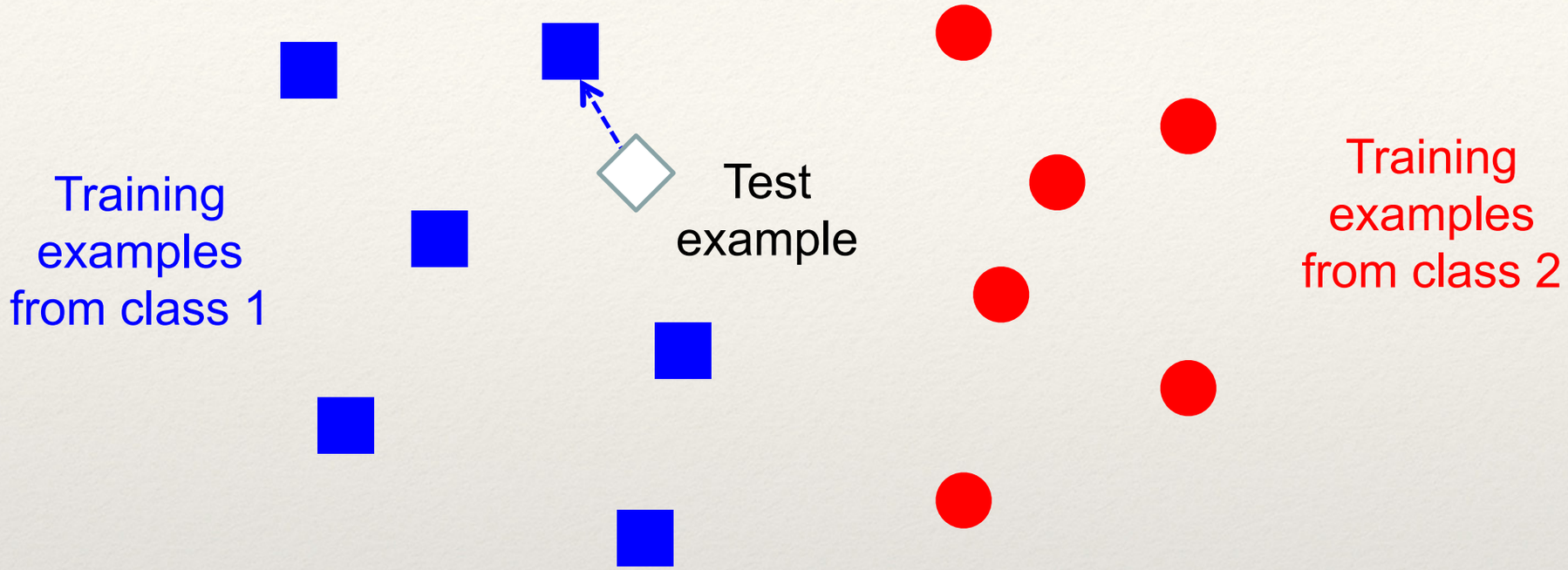
Slide credit: L. Lazebnik

Classification

Assign **x** to one of two (or more) classes.

A decision rule divides input space into *decision regions* separated by *decision boundaries* – literally boundaries in the space of the features.

Classifiers: Nearest neighbor



f(**x**) = label of the training example nearest to **x**

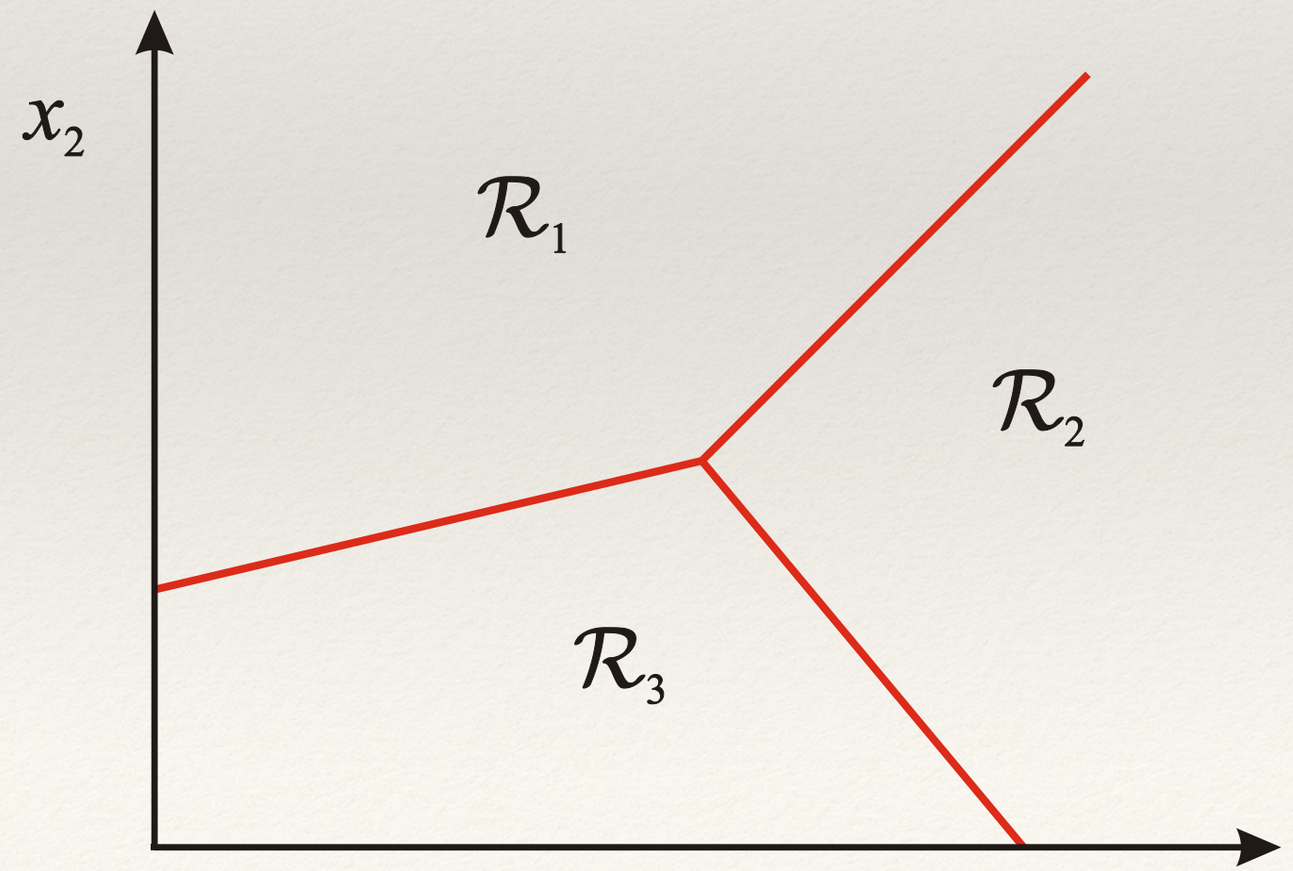
- All we need is a distance function for our inputs
- No training required!

Slide credit: L. Lazebnik

Classification

Assign **x** to one of two (or more) classes.

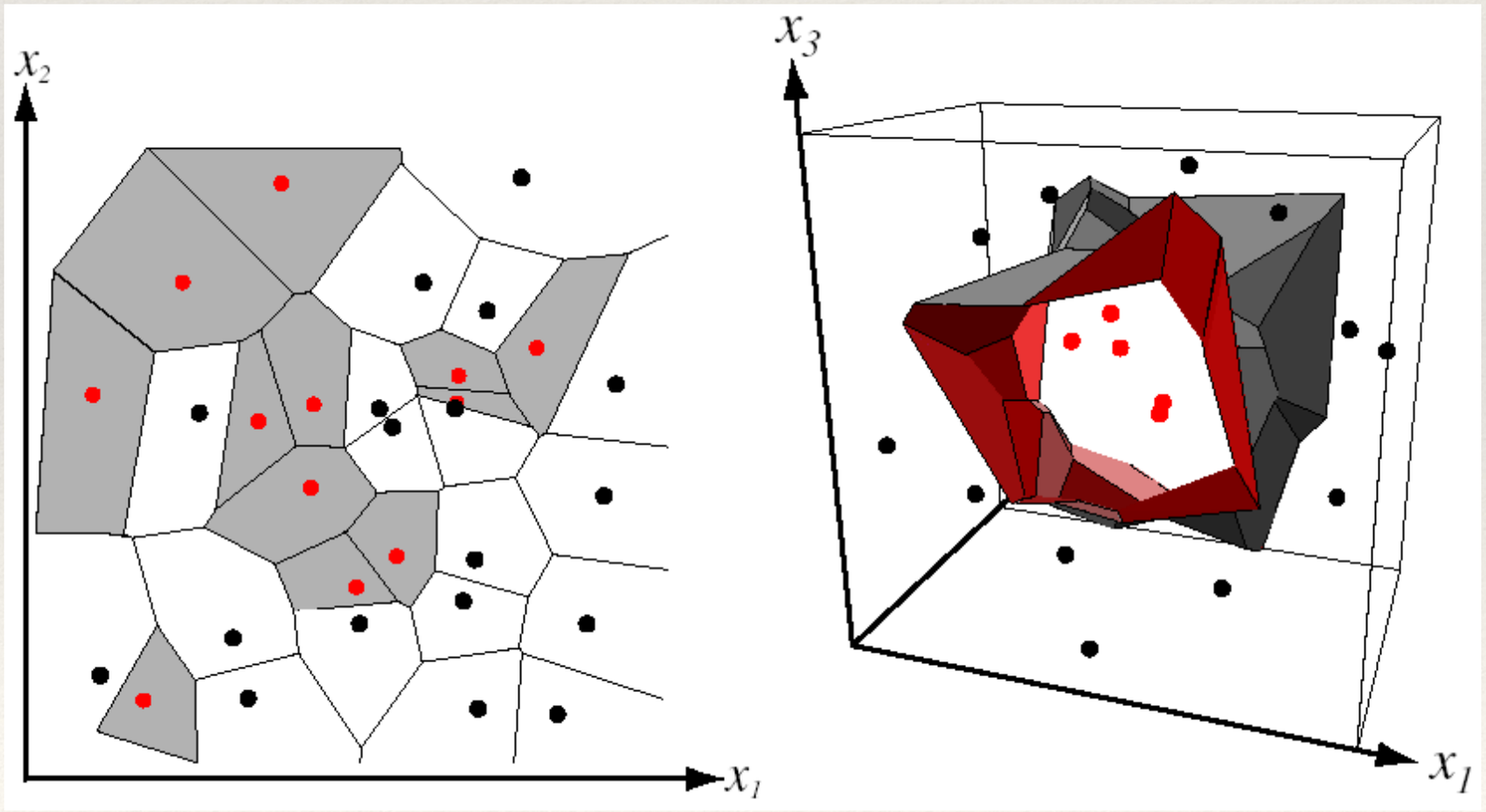
A decision rule divides input space into *decision regions* separated by *decision boundaries* – literally boundaries in the space of the features.



L. Lazebnik

Decision boundary for Nearest Neighbor Classifier

Divides input space into *decision regions* separated by *decision boundaries – Voronoi.*

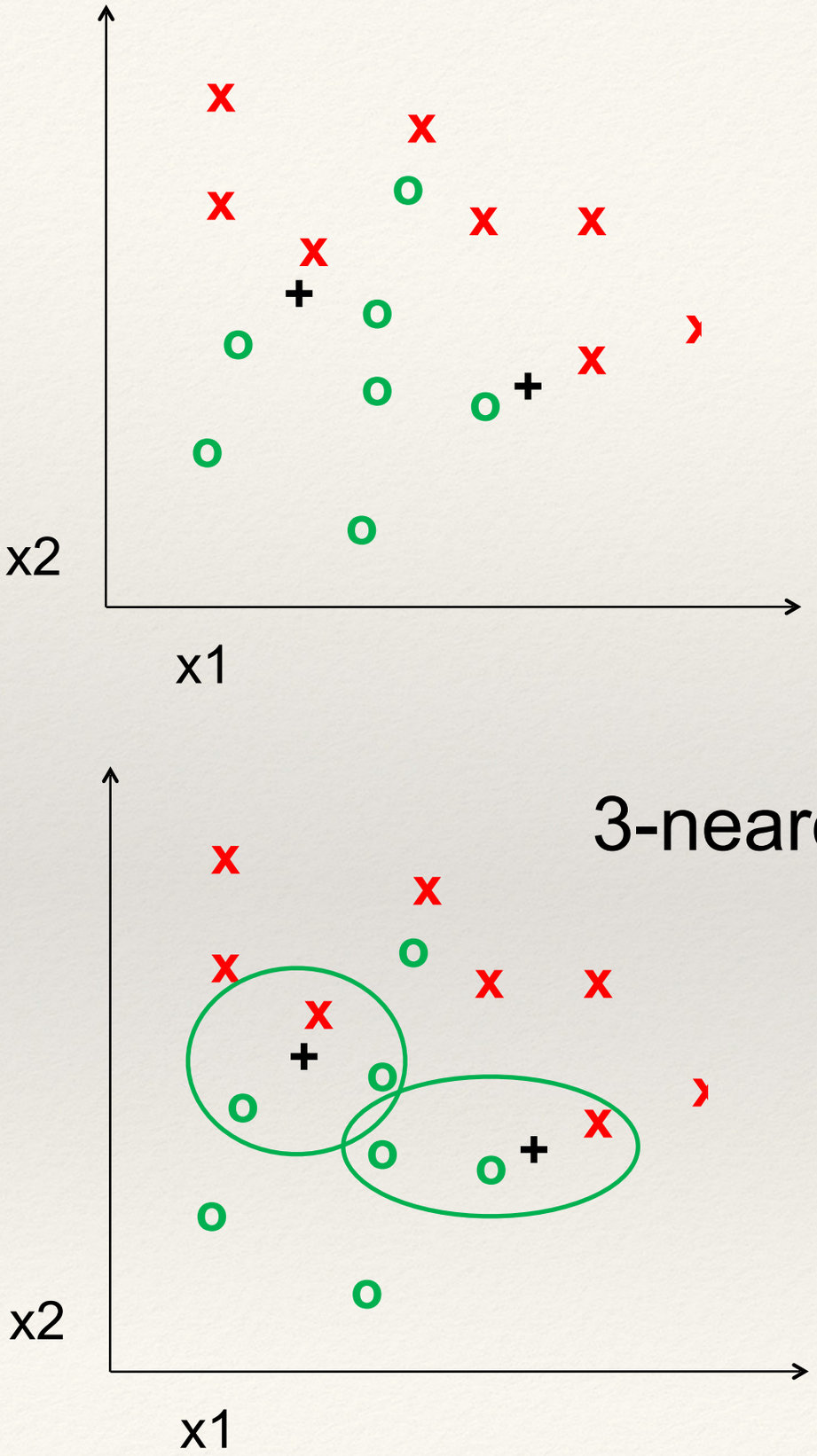


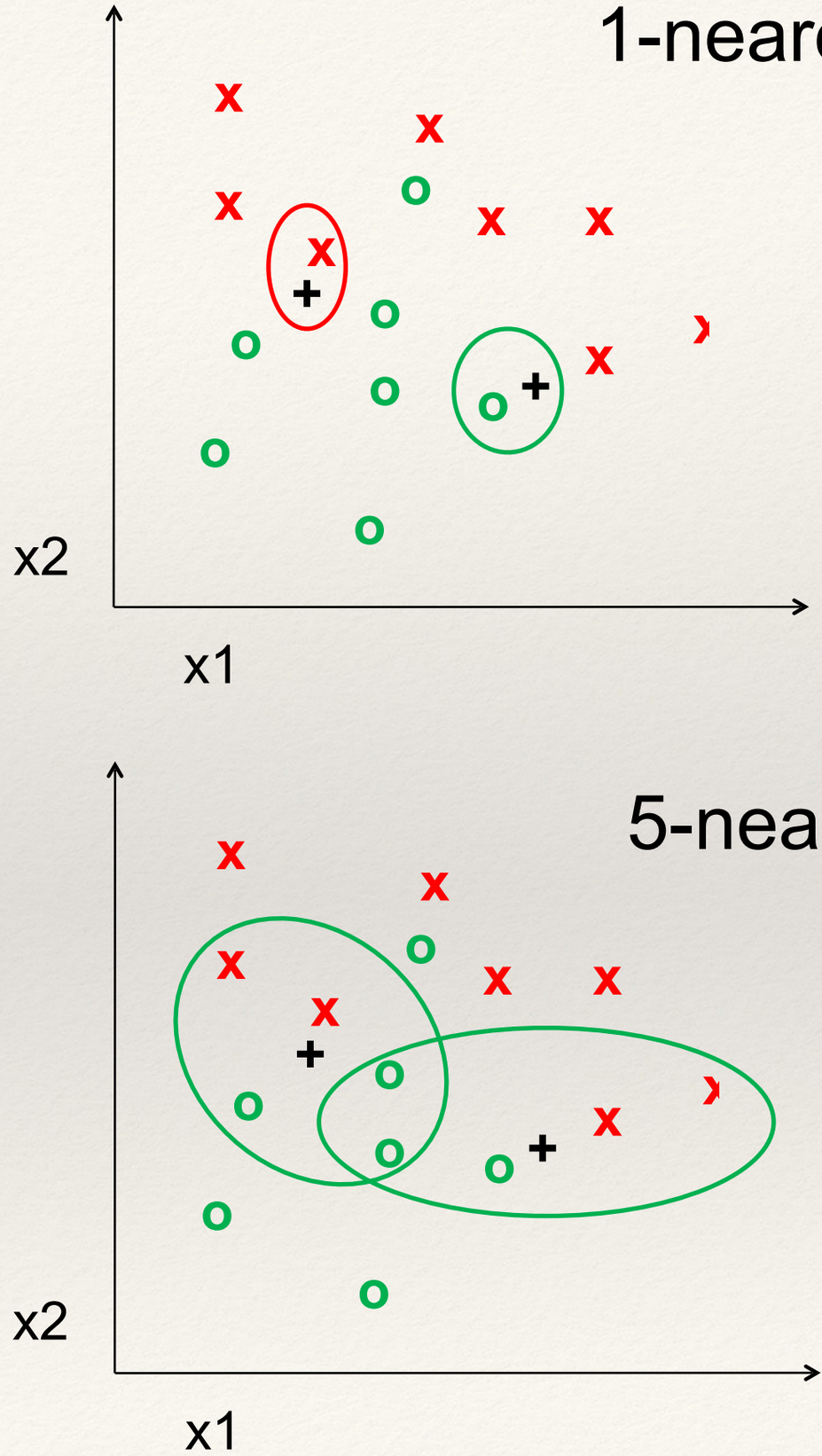
from Duda *et al.*

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

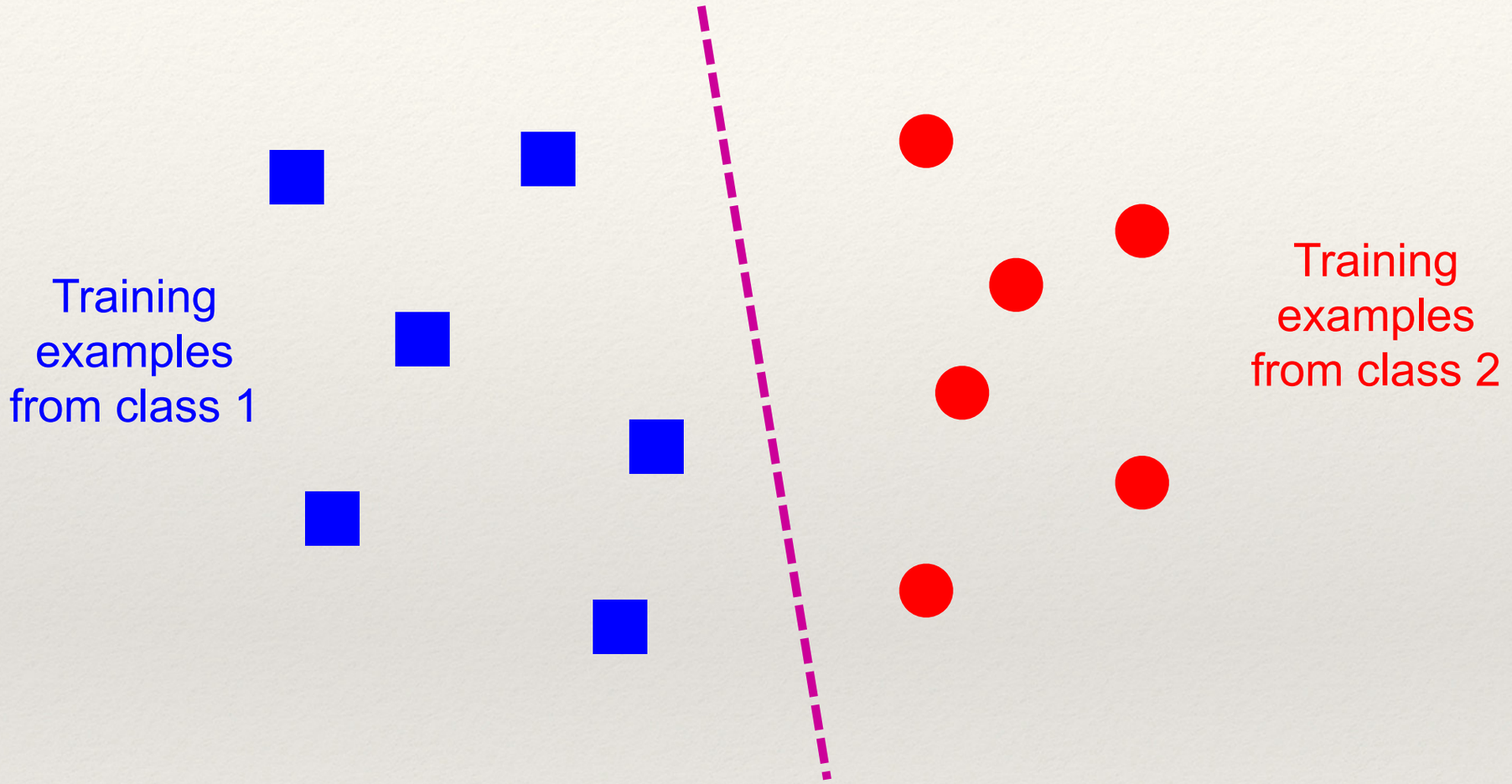
Source: D. Lowe

k-nearest neighbor





Classifiers: Linear

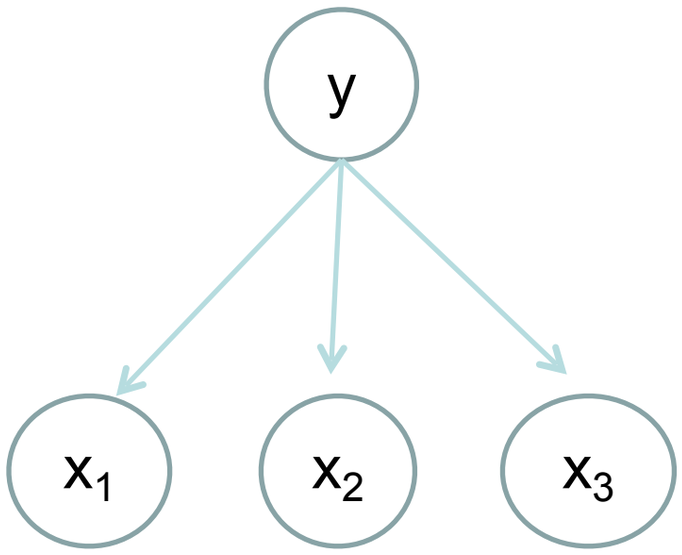


Find a *linear function* to separate the classes

Slide credit: L. Lazebnik

Classifier: Naïve Bayes

$$p(C_k \mid x_1, \ldots, x_n) = \frac{1}{Z} p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)$$



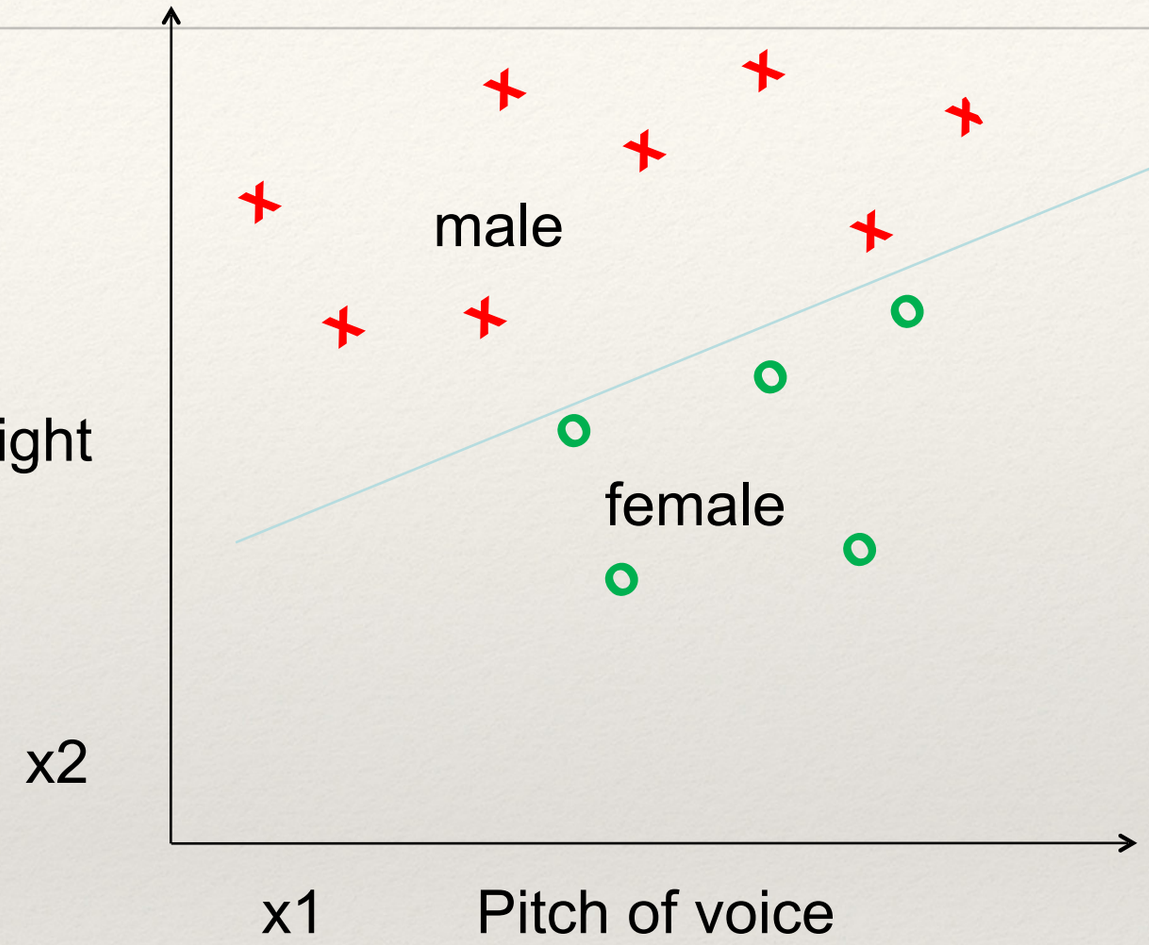
- Conditional probability model over *classes* C_k

Classifier:

$$\hat{y} = \underset{k \in \{1,\ldots,K\}}{\operatorname{argmax}} \; p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k).$$

Classifiers: Logistic Regression

Maximize likelihood of label given data, assuming a log-linear model



$$\log \frac{P(x_1, x_2 \mid y = 1)}{P(x_1, x_2 \mid y = -1)} = \mathbf{w}^T \mathbf{x}$$

$$P(y = 1 \mid x_1, x_2) = 1 / \left(1 + \exp\left(-\mathbf{w}^T \mathbf{x}\right)\right)$$

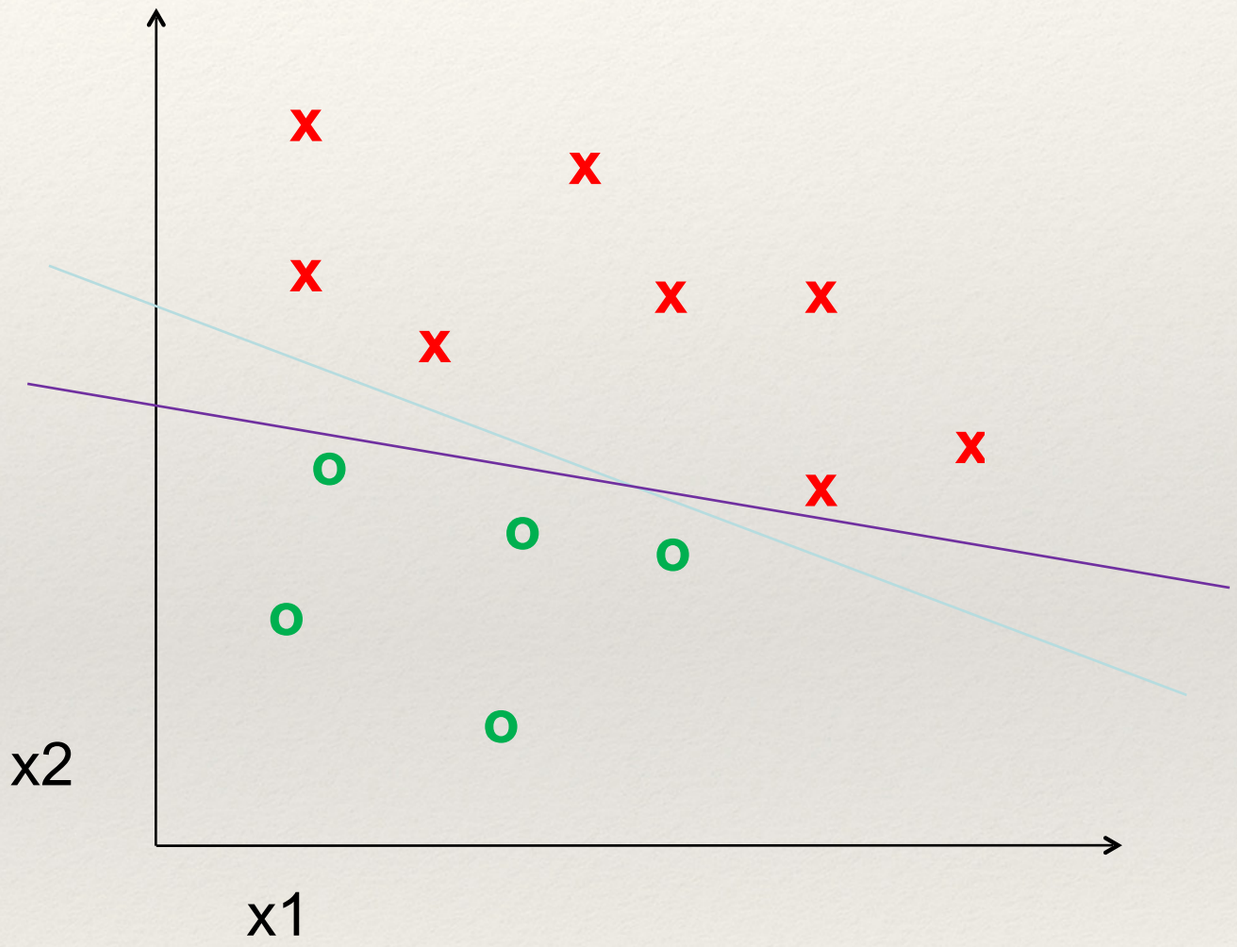
Using Logistic Regression

- Quick, simple classifier (try it first)
- Outputs a probabilistic label confidence
- Use L2 or L1 regularization
  - L1 does feature selection and is robust to irrelevant features but slower to train

Classifiers: Linear SVM

Find a *linear function* to separate the classes:

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



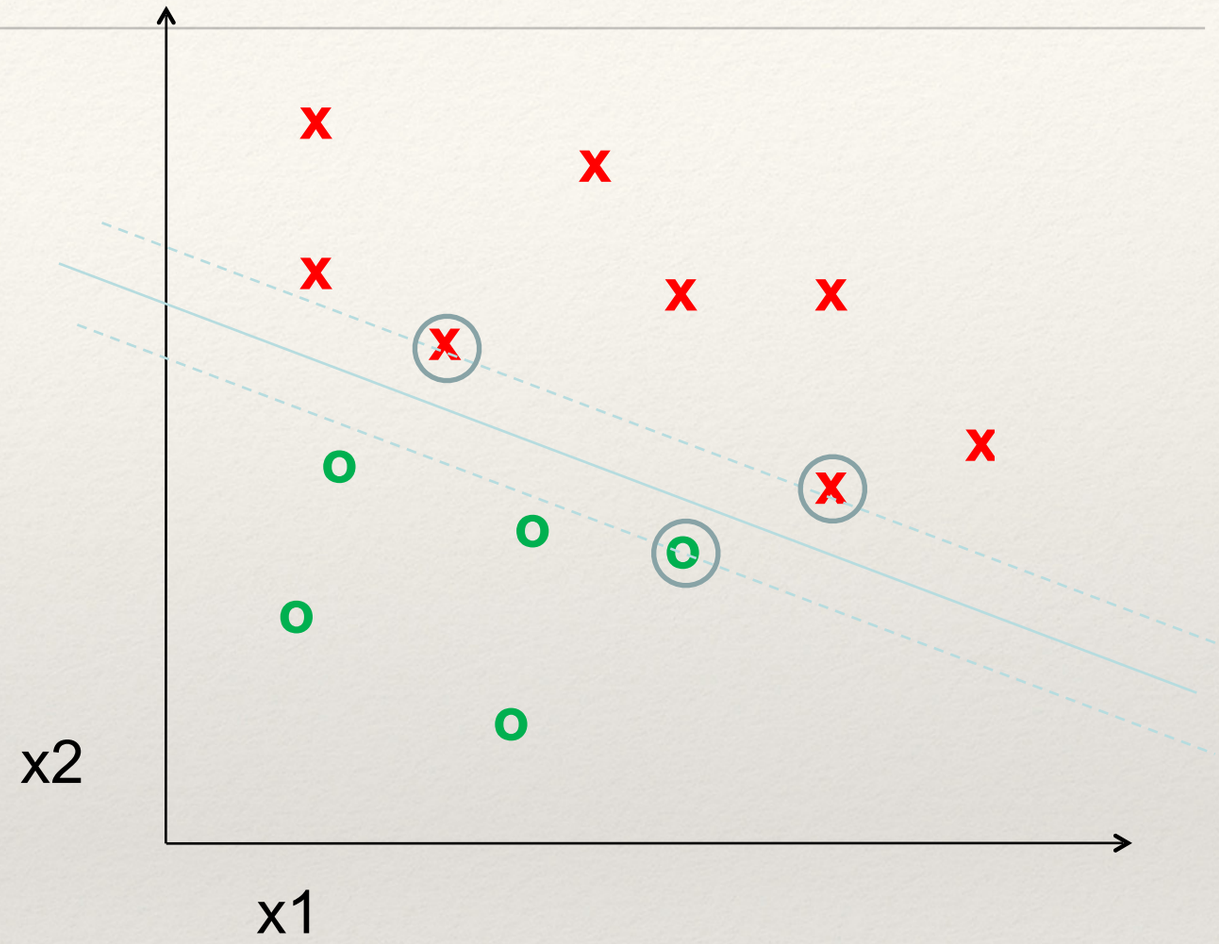
Classifiers: Linear SVM

Find a *linear function* to separate the classes:

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

How?

X = all data points



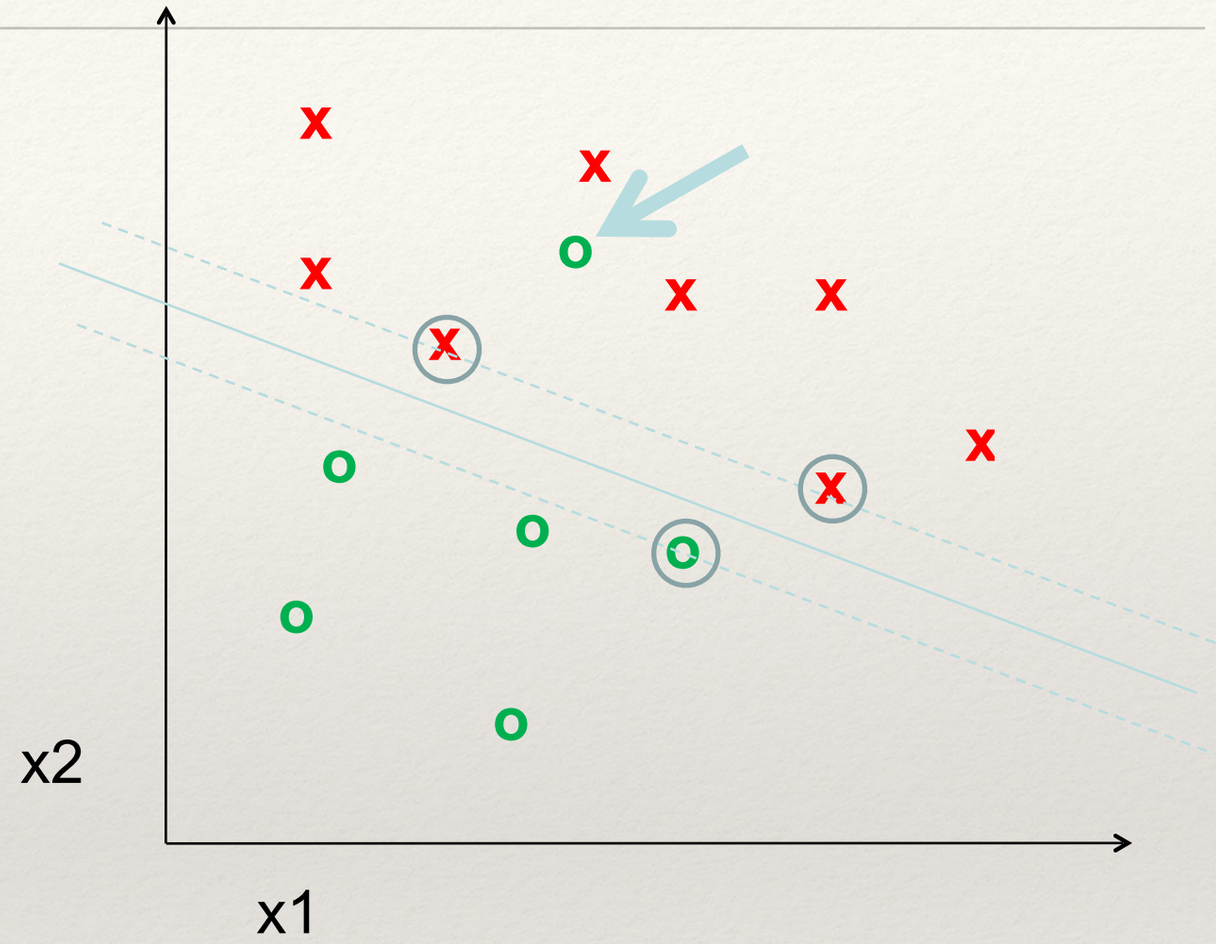
Define *hyperplane* **tX-b = 0**, where **t** is tangent to hyperplane.

Minimize **||t||** s.t. **tX-b** produces correct label for all **X**

Classifiers: Linear SVM

Find a *linear function* to separate the classes:

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

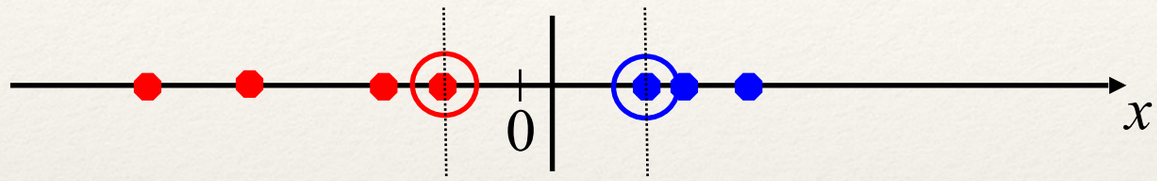


What if my data are not linearly separable?

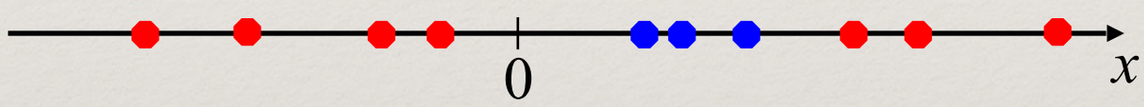
Introduce flexible ‘hinge’ loss (or ‘soft-margin’)

Nonlinear SVMs

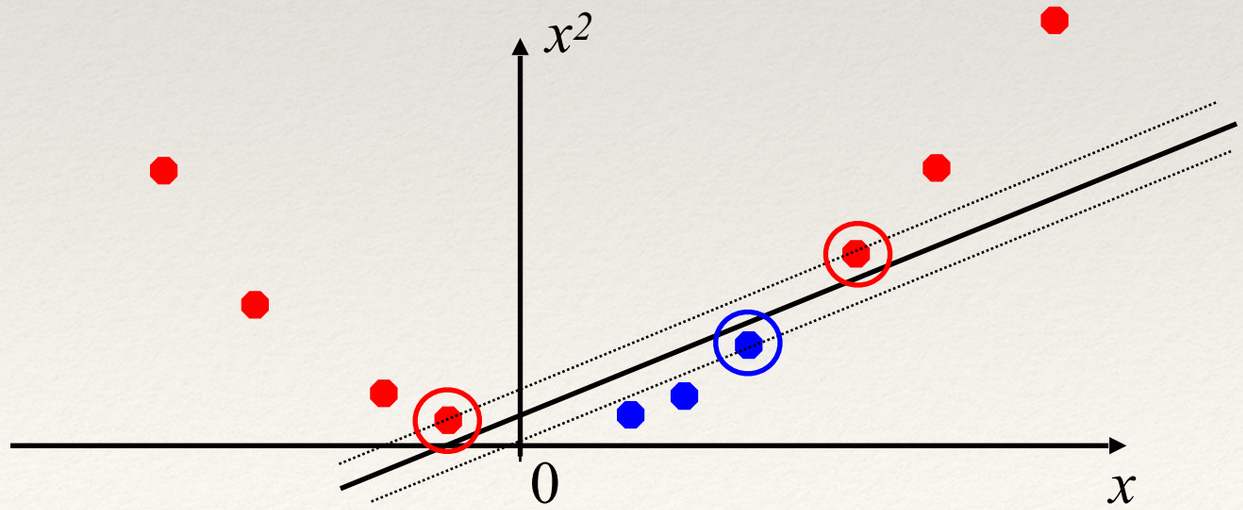
- Datasets that are linearly separable work out great:



- But what if the dataset is just too hard?



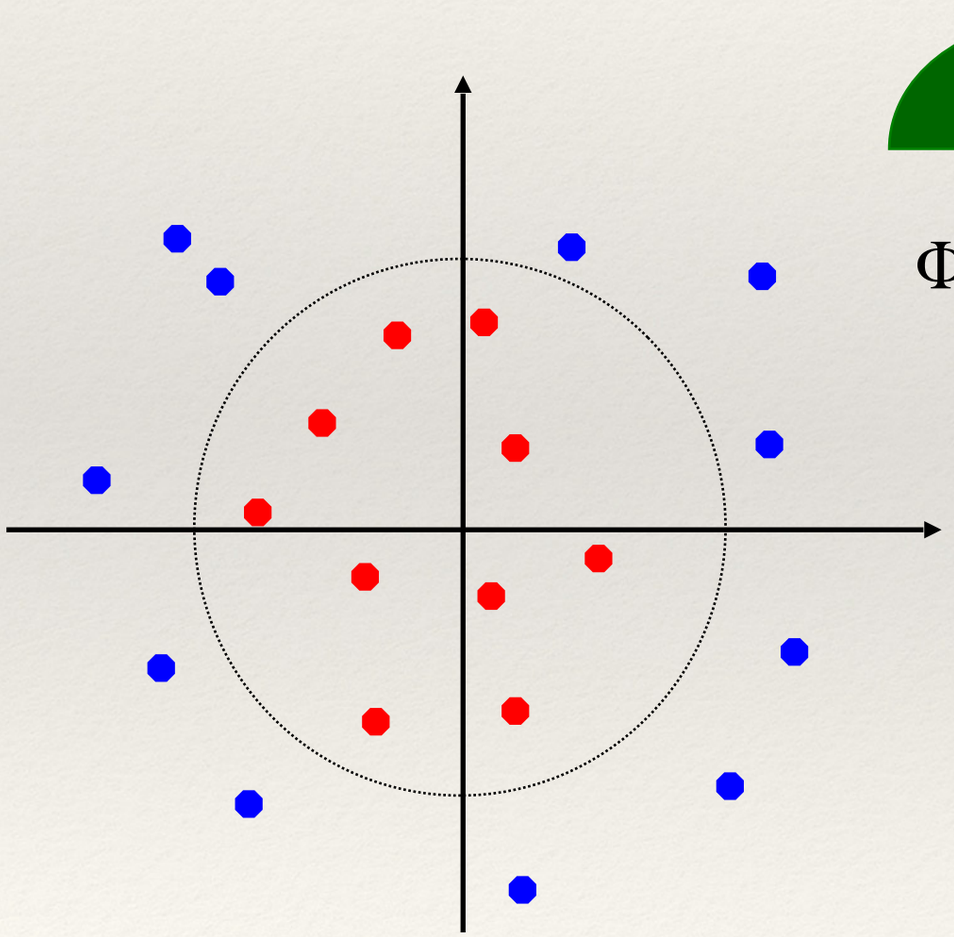
- We can map it to a higher-dimensional space:



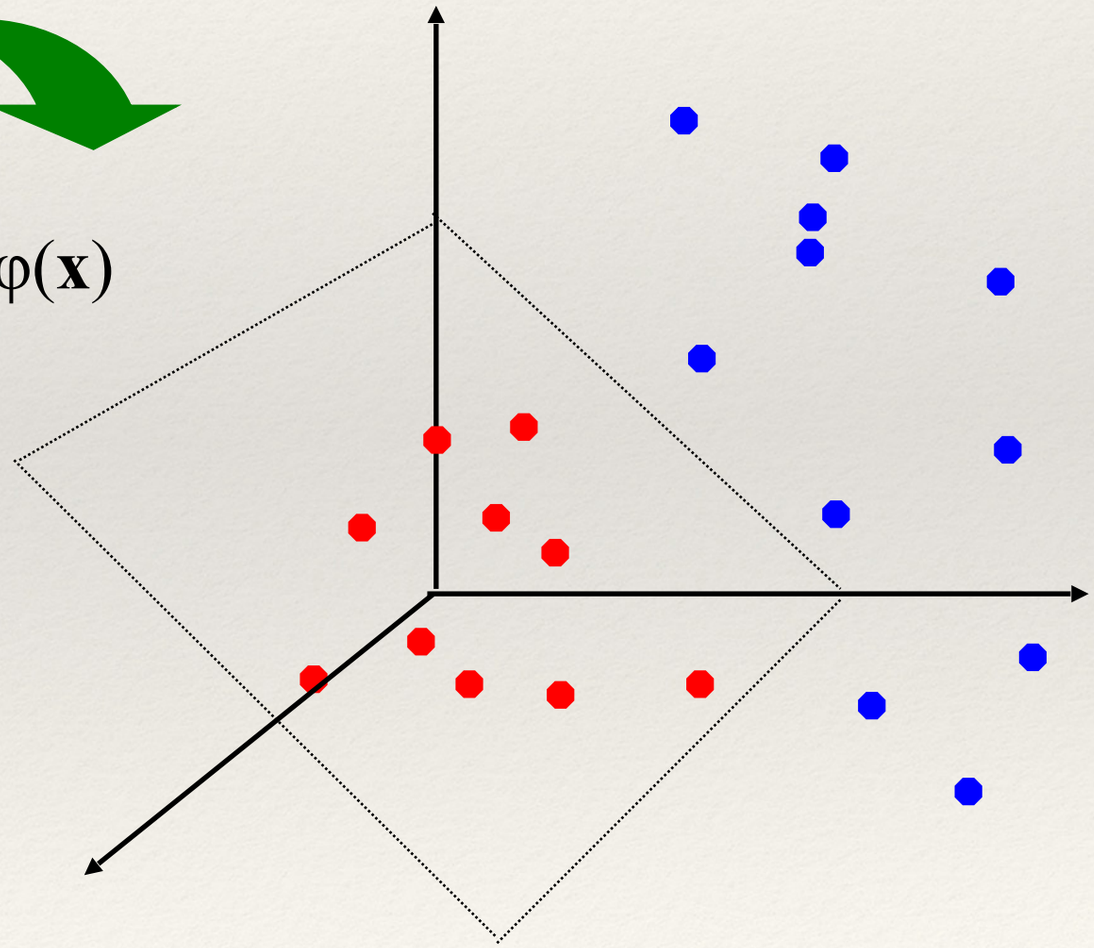
Andrew Moore

Nonlinear SVMs

Map the original input space to some higher-dimensional feature space where the training set is separable:



$\Phi: \mathbf{x} \rightarrow \varphi(\mathbf{x})$



Slide credit: Andrew Moore

Nonlinear SVMs

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

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

This gives a *non-linear* decision boundary in the original feature space:

$$\sum_i \alpha_i y_i \varphi(\boldsymbol{x}_i) \cdot \varphi(\boldsymbol{x}) + b = \sum_i \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b$$

But…we only transformed the distance function *K!*

Common kernel function: Radial basis function kernel

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

Kernels for bags of features

- Histogram intersection kernel:

$$I(h_1, h_2) = \sum_{i=1}^{N} \min(h_1(i), h_2(i))$$

- Generalized Gaussian kernel:

$$K(h_1, h_2) = \exp\left(-\frac{1}{A} D(h_1, h_2)^2 \right)$$

D can be (inverse) L1 distance, Euclidean distance, χ^2 distance, etc.

J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid, IJCV 2007
Local Features and Kernels for Classifcation of Texture and Object Categories: A Comprehensive Study

What about multi-class SVMs?

Unfortunately, there is no “definitive” multi-class SVM.

In practice, we combine 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

Slide credit: L. Lazebnik

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

Ideals for a classification algorithm

- Objective function: encodes the right loss for the problem
- Parameterization: takes advantage of the structure of the problem
- Regularization: good priors on the parameters
- Training algorithm: can find parameters that maximize objective on training set
- Inference algorithm: can solve for labels that maximize objective function for a test example

Two ways to think about classifiers

1. What is the objective?
   What are the parameters?
   How are the parameters learned?
   How is the learning regularized?
   How is inference performed?

2. How is the data modeled?
   How is similarity defined?
   What is the shape of the boundary?

Slide credit: D. Hoiem

Training

Training Images

Training Labels

Image Features

Training

Learned classifier

Testing

Test Image

Image Features

Apply classifier

Prediction

Slide credit: D. Hoiem and L. Lazebnik

Features and **distance measures**

define visual similarity.

Training labels

dictate that examples are the same or different.

Classifiers

learn weights (or parameters) of features and distance measures…

so that visual similarity predicts label similarity.

Generalization

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

Training set (labels known)

Test set (labels unknown)

Slide credit: L. Lazebnik

Generalization Error

Bias:

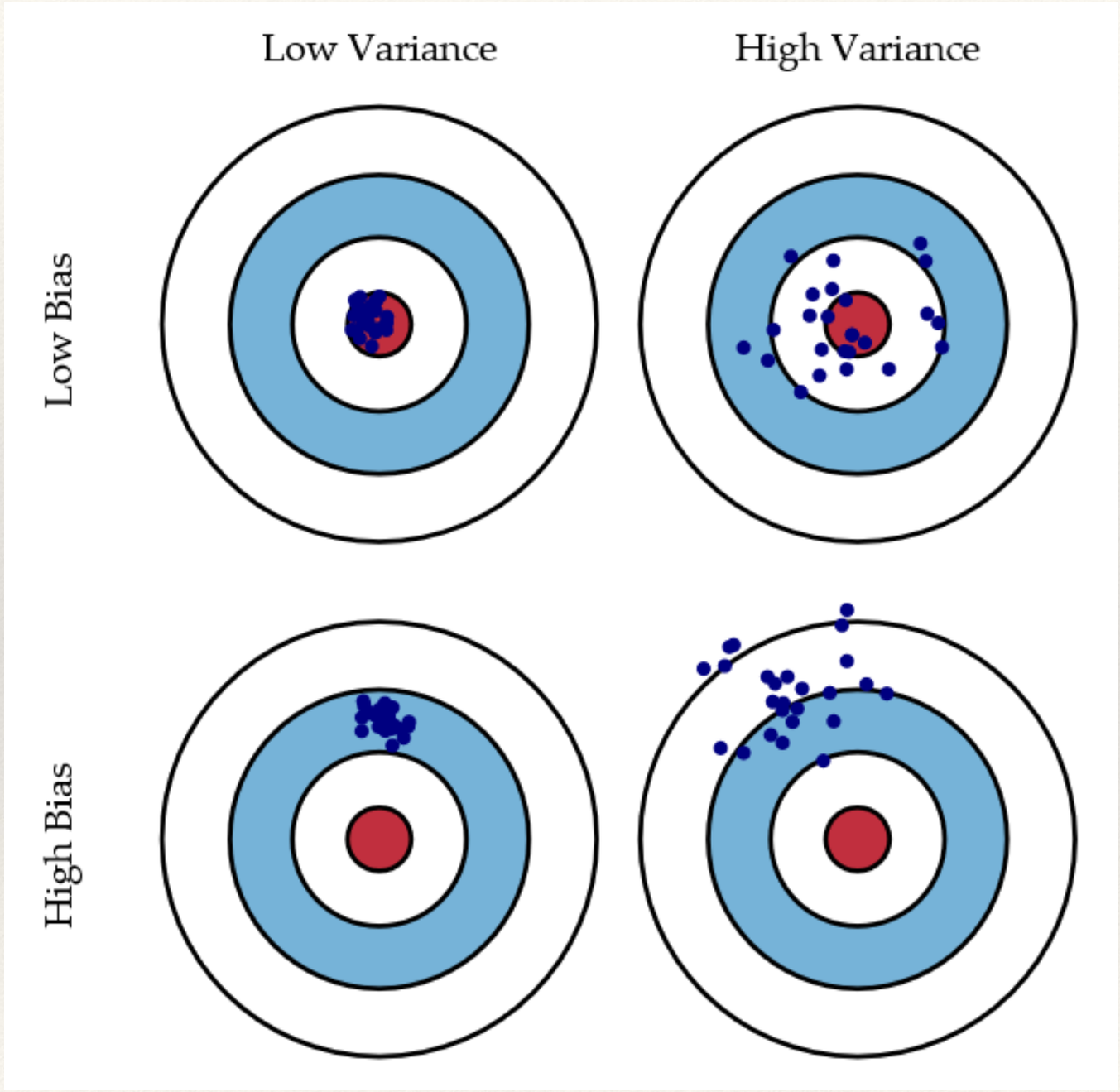
- Difference between the expected (or average) prediction of our model and the correct value.
- Error due to inaccurate assumptions/simplifications.

Variance:

- Amount that the estimate of the target function will change if different training data was used.

Slide credit: L. Lazebnik

Bias/variance trade-off



Bias = accuracy

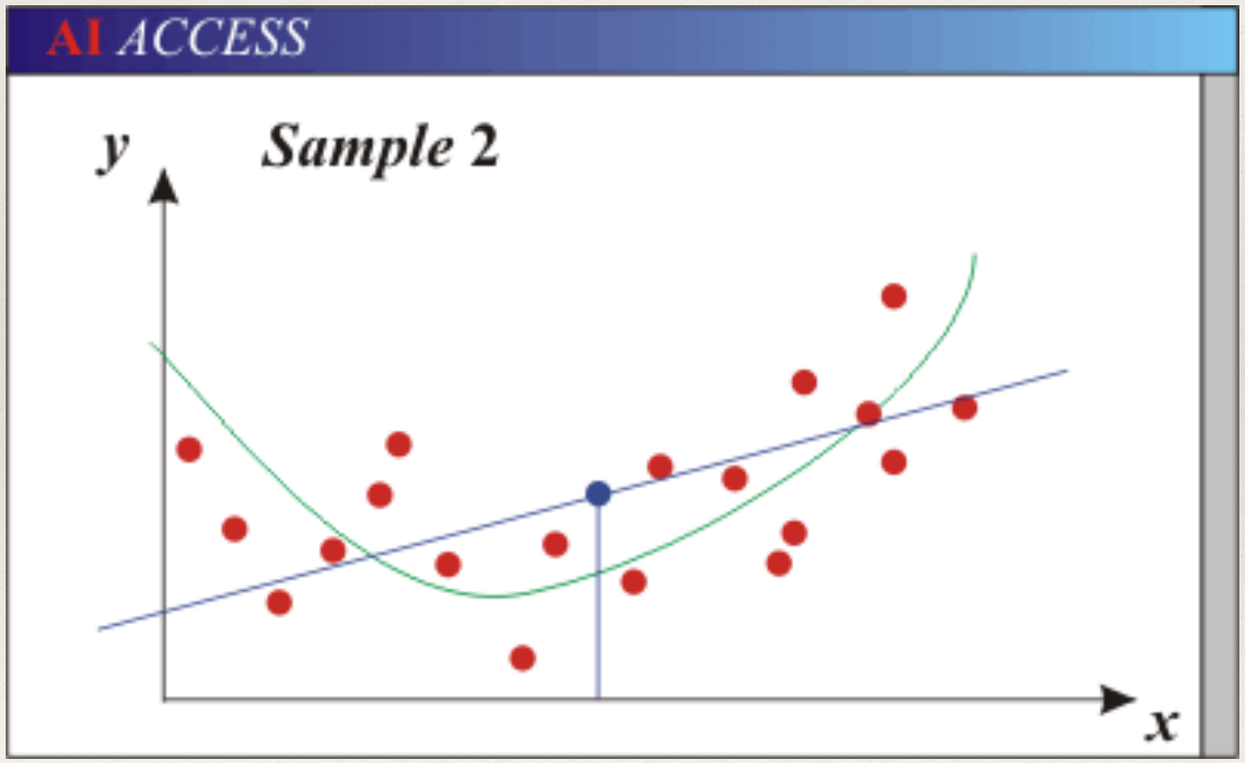
Variance = precision

[Scott Fortmann-Roe]

Generalization Error Effects

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



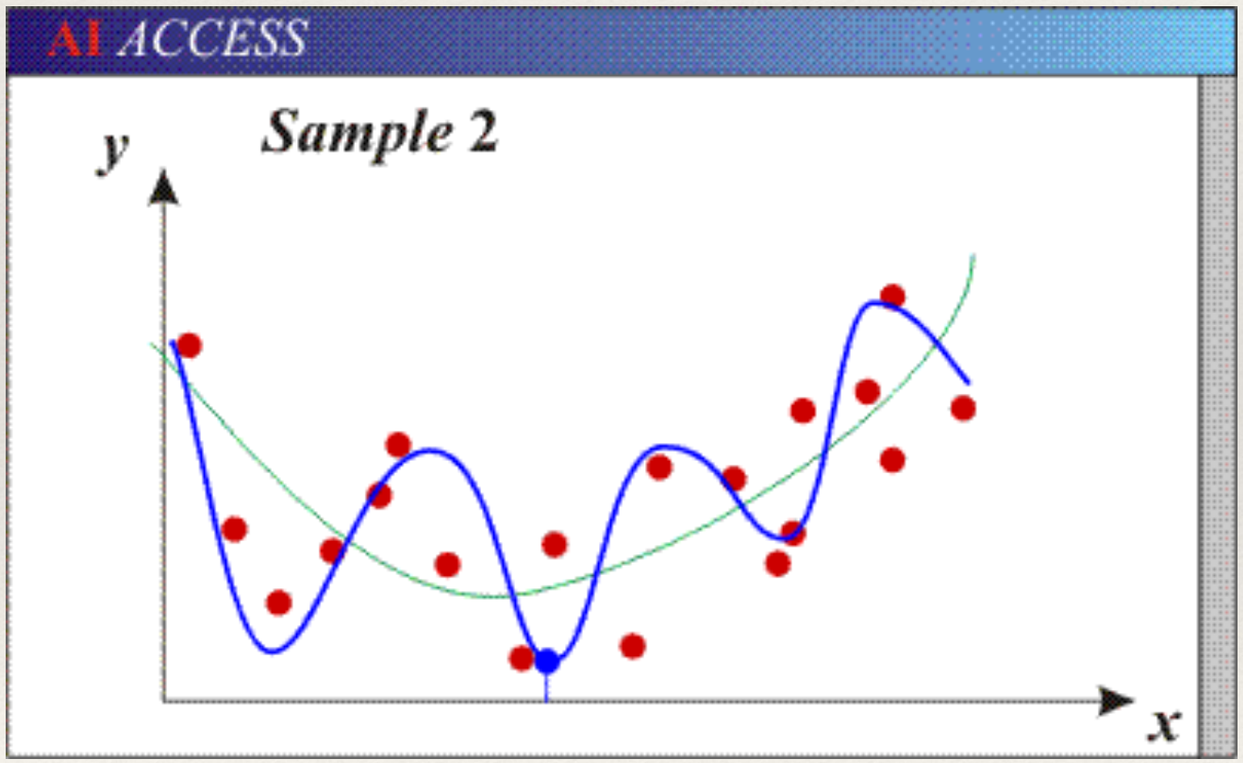
Green line = true data-generating function without noise
Blue line = data model which underfits

Slide credit: L. Lazebnik

Generalization Error Effects

Overfitting: model is too "complex" and fits irrelevant characteristics (noise) in the data

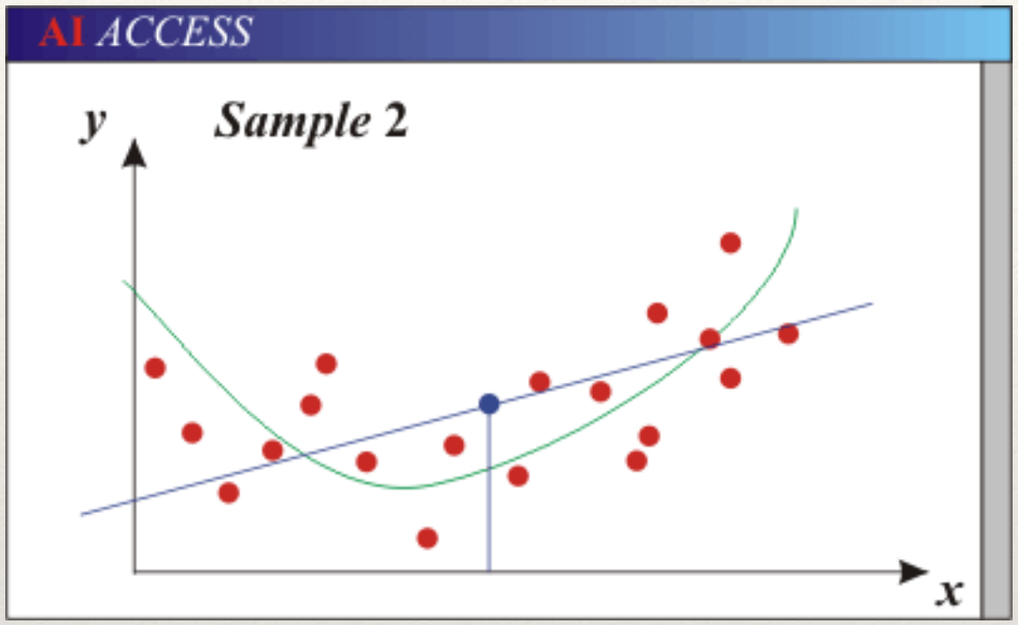
- Low bias (many degrees of freedom) and high variance
- Low training error and high test error



Green line = true data-generating function without noise
Blue line = data model which overfits

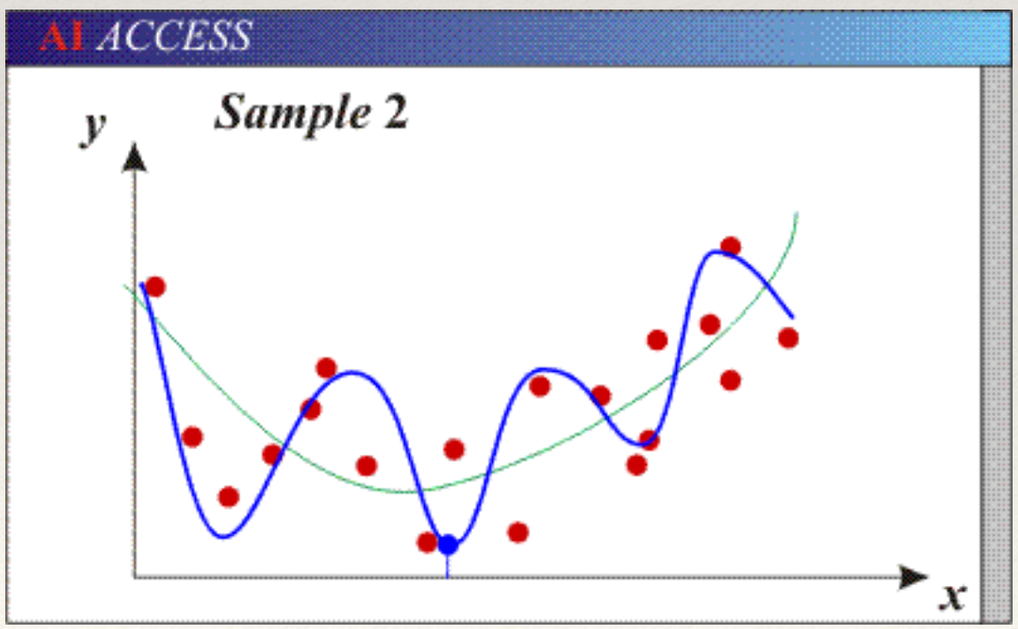
Slide credit: L. Lazebnik

Bias-Variance Trade-off



Models with too few parameters are inaccurate because of a large bias.

- Not enough flexibility!
- Too many assumptions



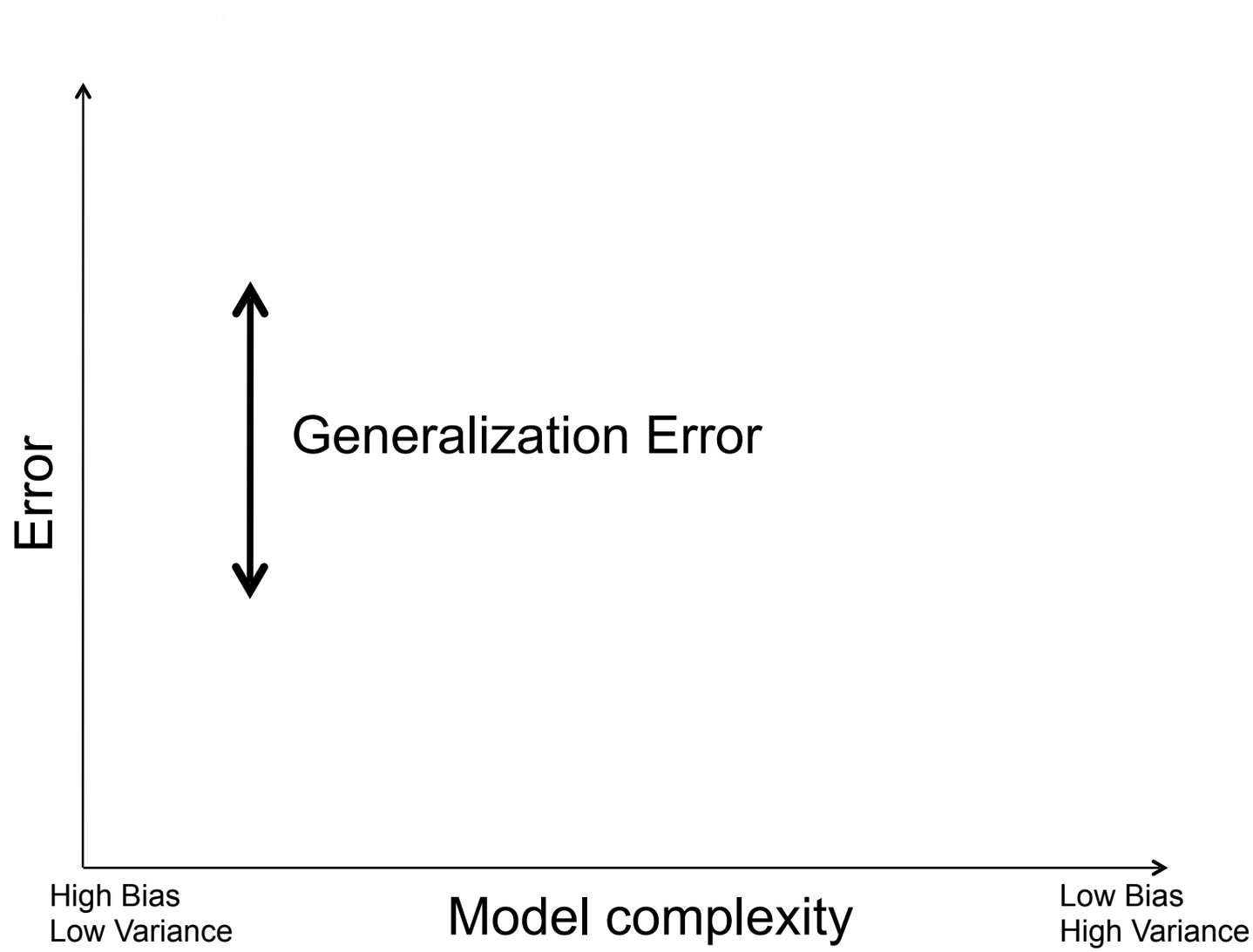
Models with too many parameters are inaccurate because of a large variance.

- Too much sensitivity to the sample.
- Slightly different data ->
  very different function.

Slide credit: D. Hoiem

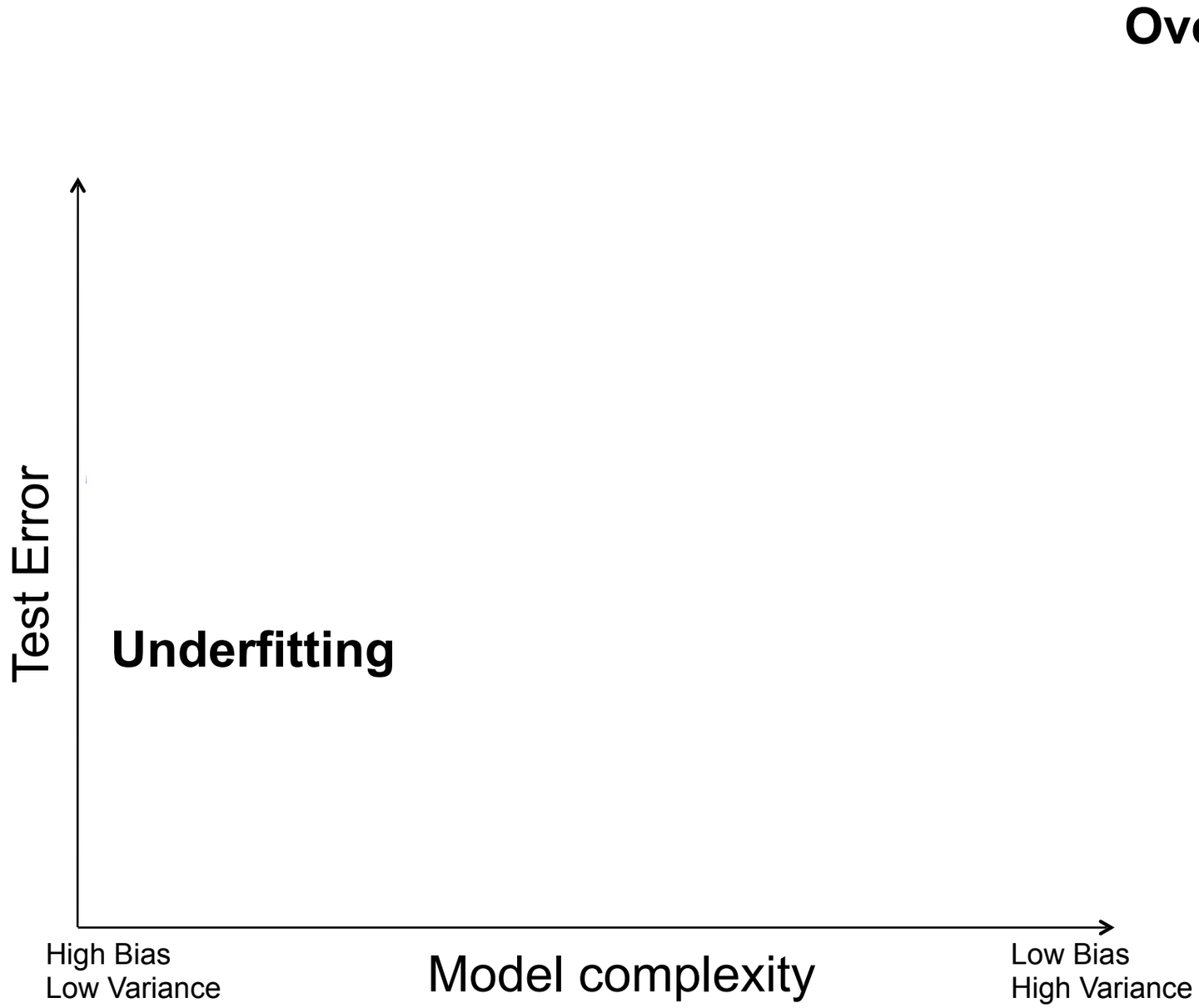
Bias-variance tradeoff

Fixed number of training examples



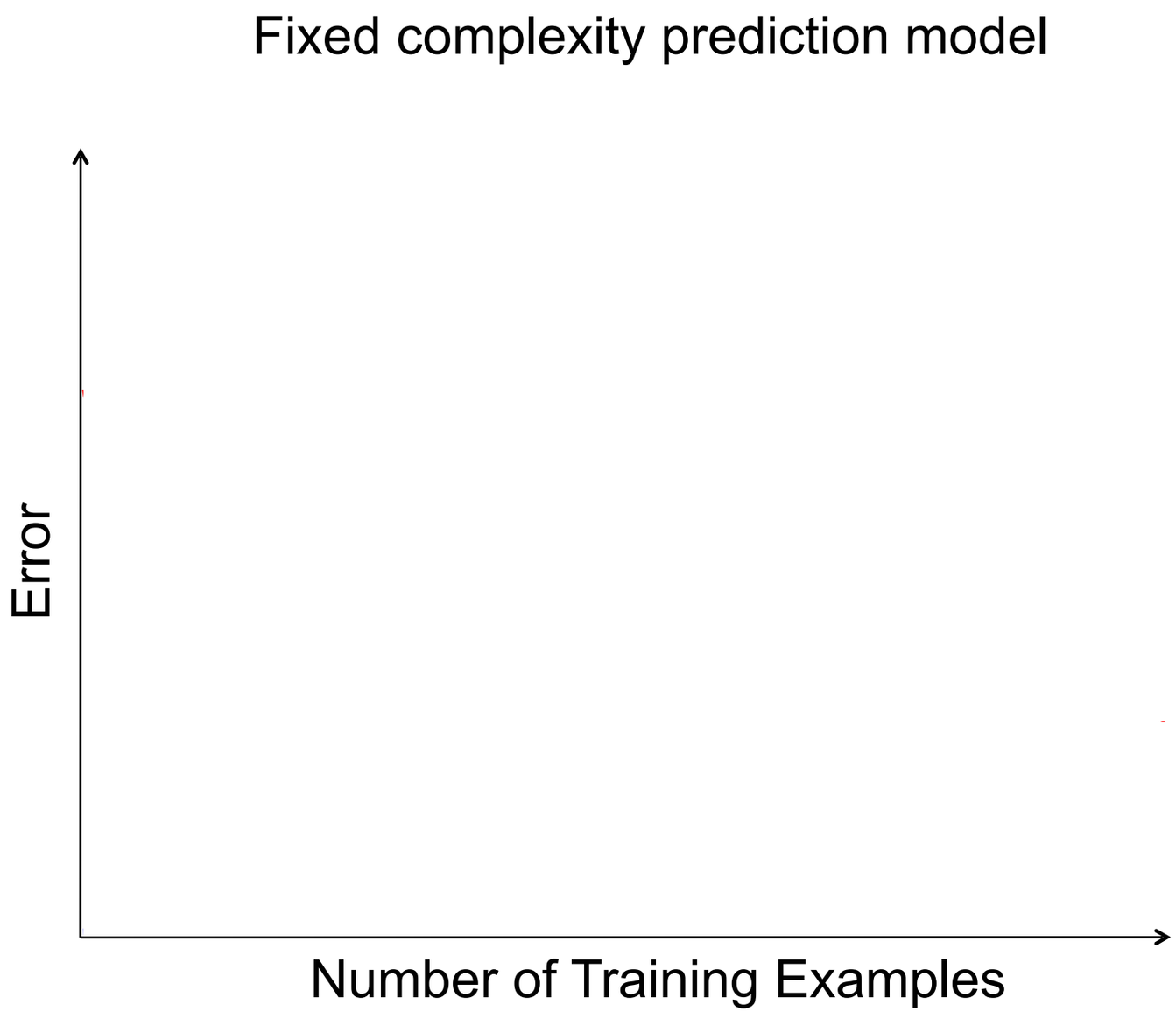
Slide credit: D. Hoiem

Bias-variance tradeoff



Slide credit: D. Hoiem

Effect of Training Size



Slide credit: D. Hoiem

How to reduce variance?

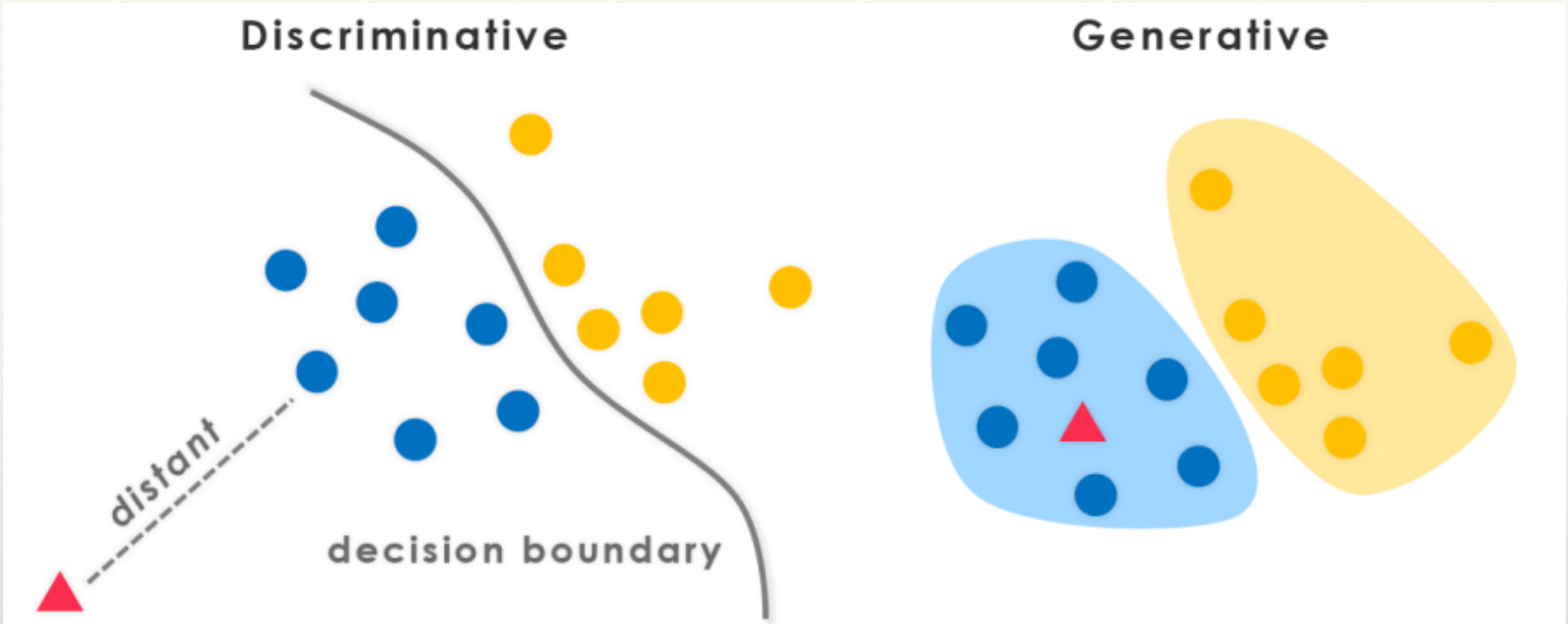
- Choose a simpler classifier

- Regularize the parameters

- Get more training data

Slide credit: D. Hoiem

[evolvingai.org]



“Learn the data boundary”

Given:
Observations X
Targets Y

Learn conditional distribution:
$P(Y \mid X = x)$

“Represent the data and then define boundary”

Given:
Observations X
Targets Y

Learn joint distribution:
$P(X, Y)$

Generative vs. Discriminative Classifiers

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

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

Slide credit: D. Hoiem

Making decisions about data

- 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.

Many classifiers to choose from…

- K-nearest neighbor
- SVM
- Naïve Bayes
- Bayesian network
- Logistic regression
- Randomized Forests
- Boosted Decision Trees
- Restricted Boltzmann Machines
- Neural networks
- Deep Convolutional Network
- …

Claim:

It is more important to have more or better labeled data than to use a different supervised learning technique.

"The Unreasonable Effectiveness of Data" - Norvig

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)

Slide credit: D. Hoiem

Recognition Issues

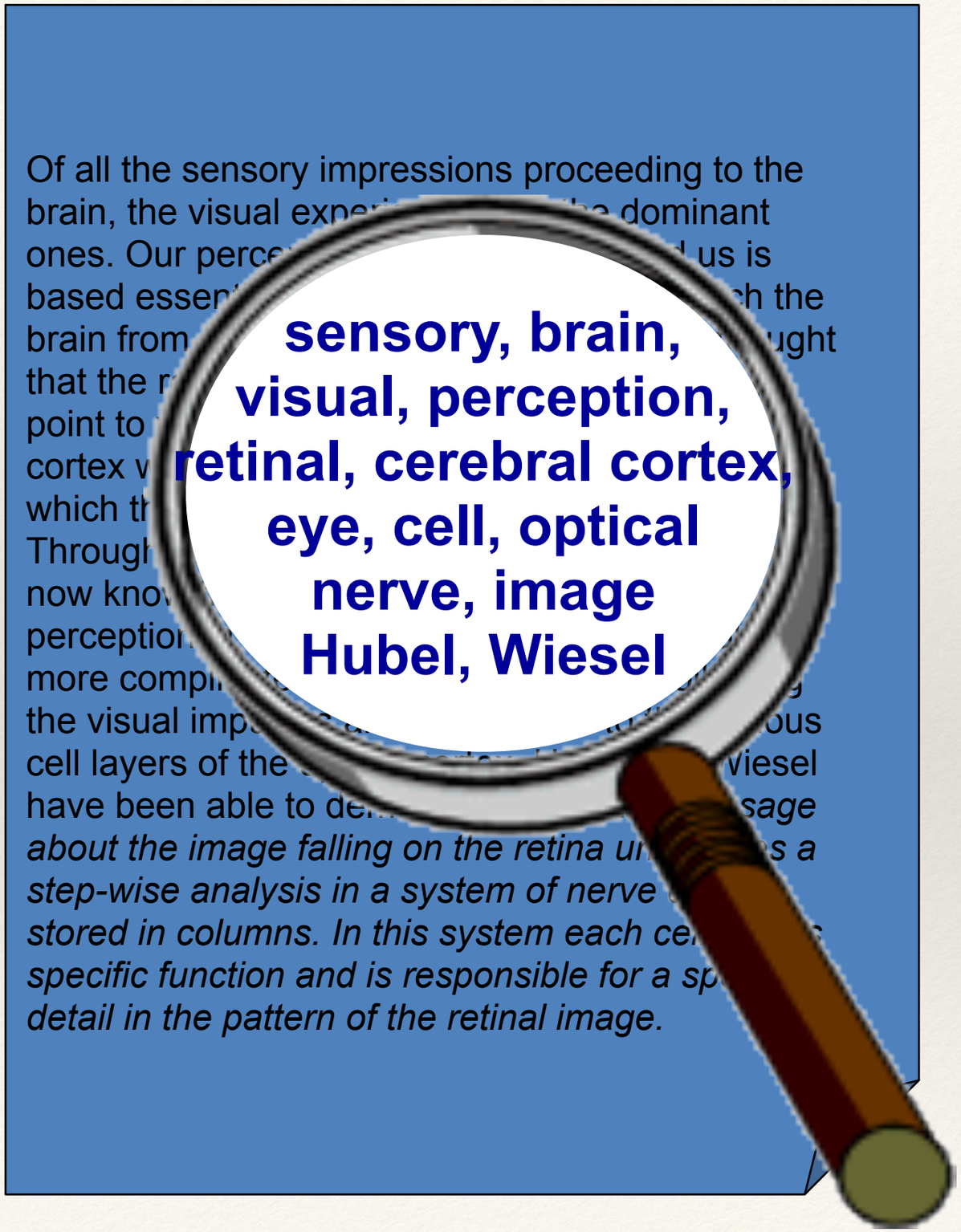
- How to bridge the gap between feature and label?
- How to summarize the content of an entire image?
  How to gauge overall similarity?
- How large should the vocabulary be?
  How to perform quantization efficiently?
- How to score the retrieval results?
- How might we add more spatial verification?

Bag-of-features models

Object → **Bag of ‘words’**

Svetlana Lazebnik

Analogy to documents



Of all the sensory impressions proceeding to the brain, the visual experience is the dominant ones. Our perception of us is based essentially on the way in which the brain from ught that the r ... point to ... cortex w ... which th ... Through ... now know ... perception ... more compl ... the visual imp ... ous cell layers of the ... Wiesel have been able to de ... sage *about the image falling on the retina u ... s a step-wise analysis in a system of nerve ... stored in columns. In this system each ce ... specific function and is responsible for a sp ... detail in the pattern of the retinal image.*

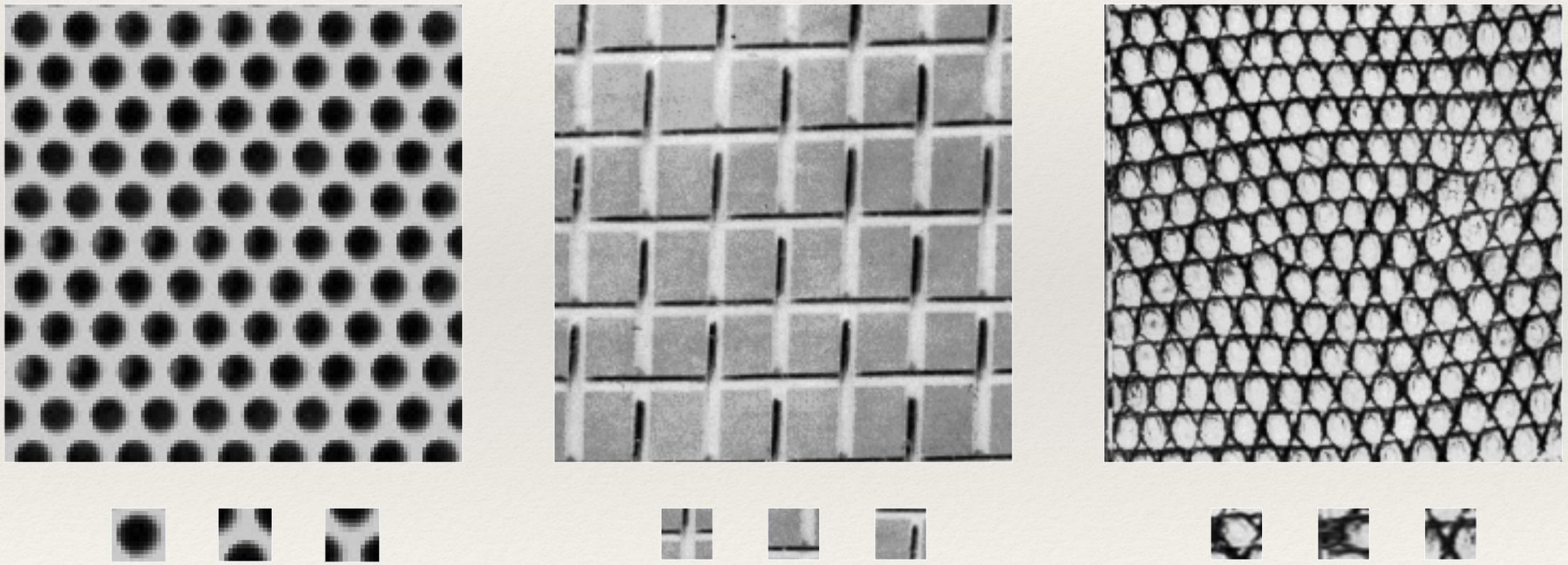
China is forecasting a trade surplus of $90bn (£51bn) to $100bn this year, a threefold increase on 2004's $32bn. T... istry said the surplus would ... 0% jump in exports t... rise in imports to ... ther annoy th... a's exports ... underv... too high ... Bank of ... country a... c demand s... y. China increa... the dollar by 2.1% ... e within a narrow ba... yuan to be allowed to trade fre... g has made it clear that it will take its time an... carefully before allowing the yuan to rise ... in value.

Origin 1: Bag-of-words models

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

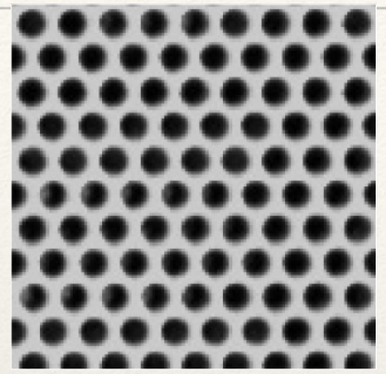
Origin 2: Texture recognition

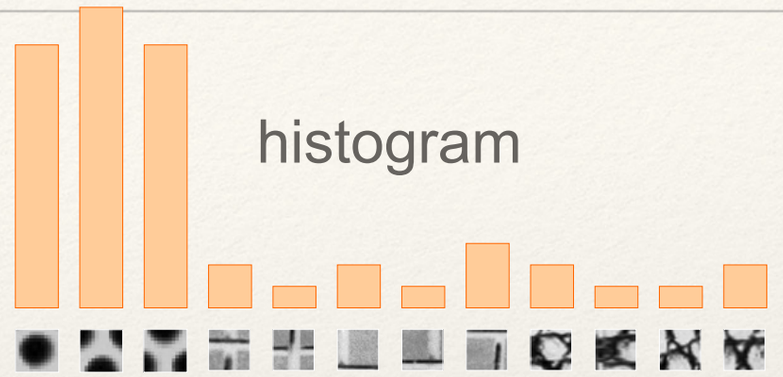
- Characterized by repetition of basic elements or *textons*
- For stochastic textures, the identity of textons matters,
  not their spatial arrangement

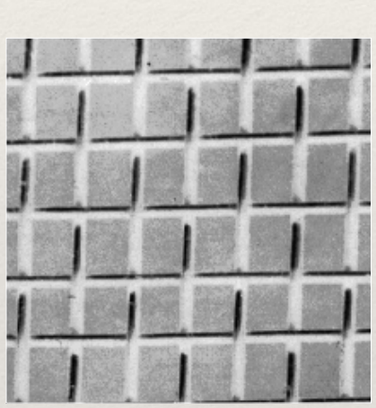


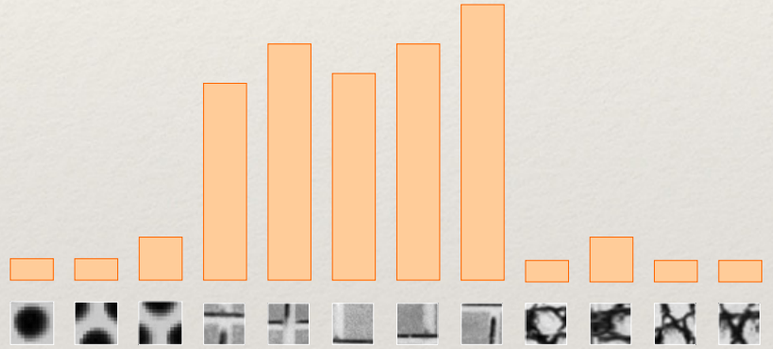
Julesz, 1981; Cula & Dana, 2001; Leung & Malik 2001; Mori, Belongie & Malik, 2001; Schmid 2001; Varma & Zisserman, 2002, 2003; Lazebnik, Schmid & Ponce, 2003

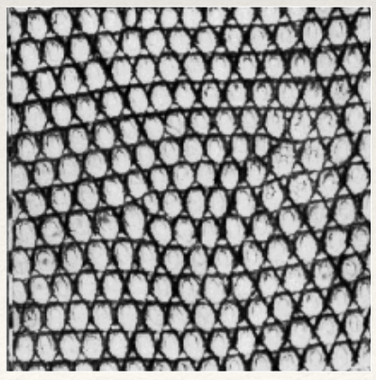
Origin 2: Texture recognition

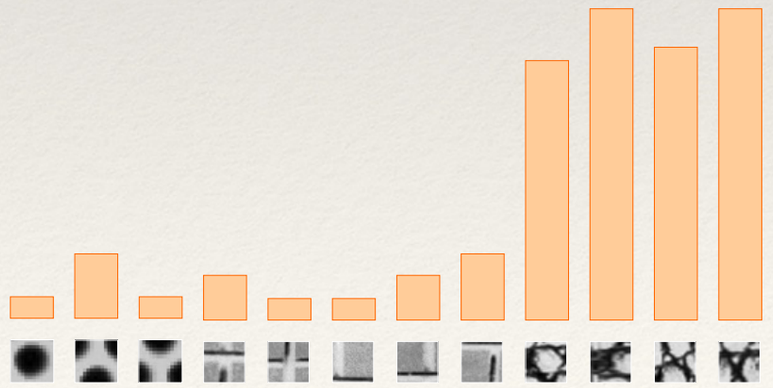










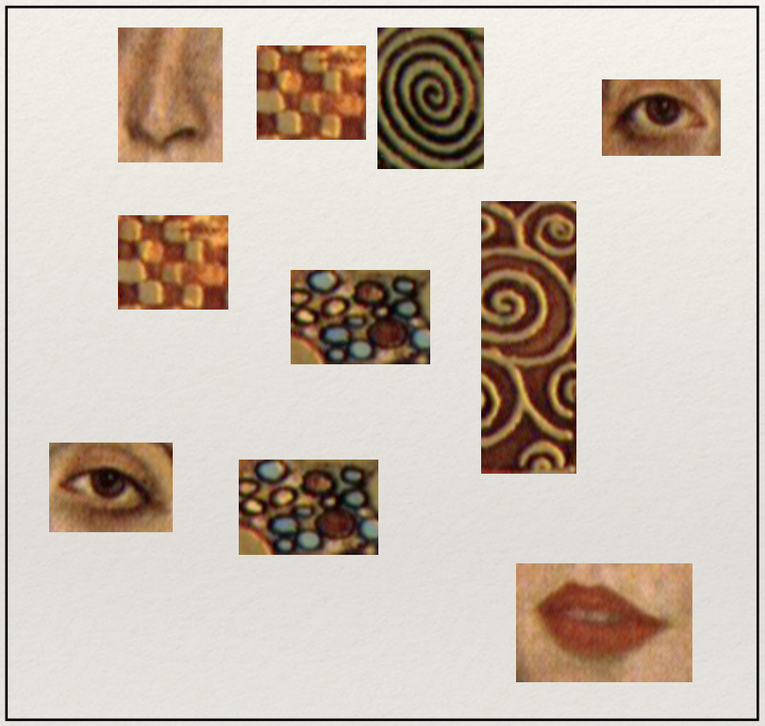


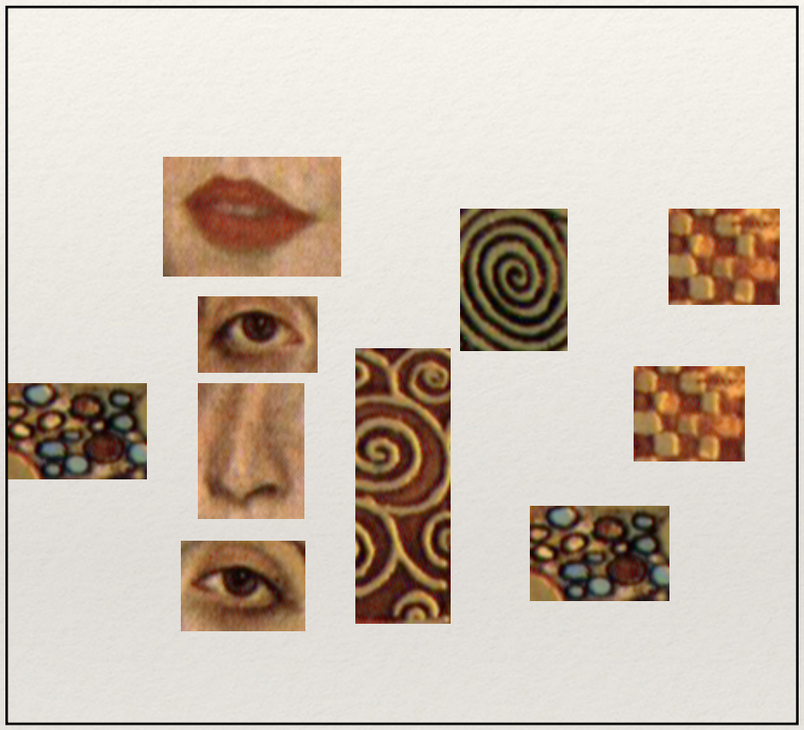
Julesz, 1981; Cula & Dana, 2001; Leung & Malik 2001; Mori, Belongie & Malik, 2001; Schmid 2001; Varma & Zisserman, 2002, 2003; Lazebnik, Schmid & Ponce, 2003

Bag-of-features models

Objects as texture

- All of these are treated as being the same

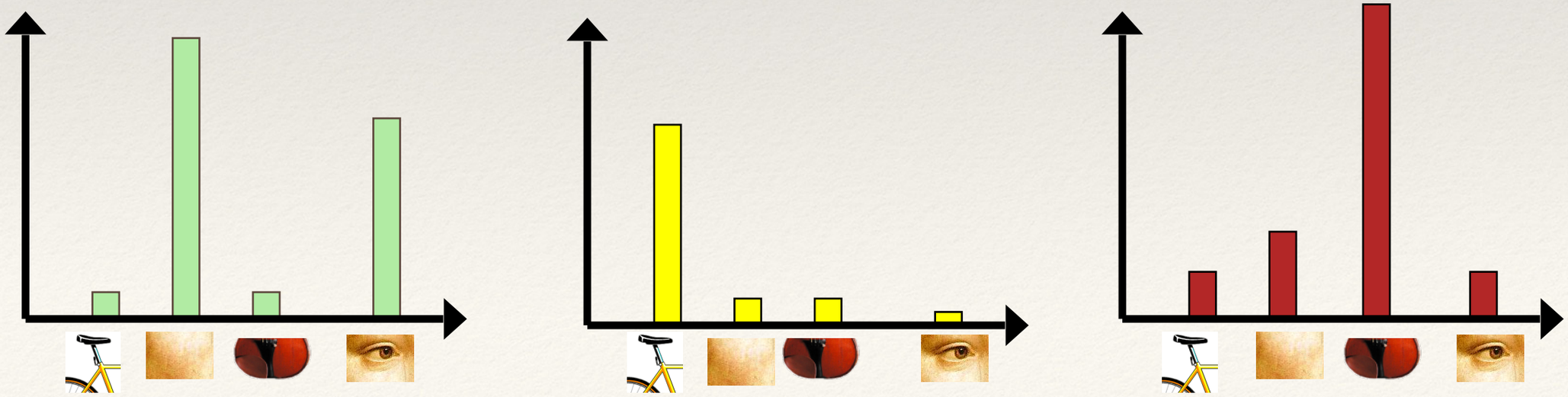




- No distinction between foreground and background: scene recognition?

Bag-of-features steps

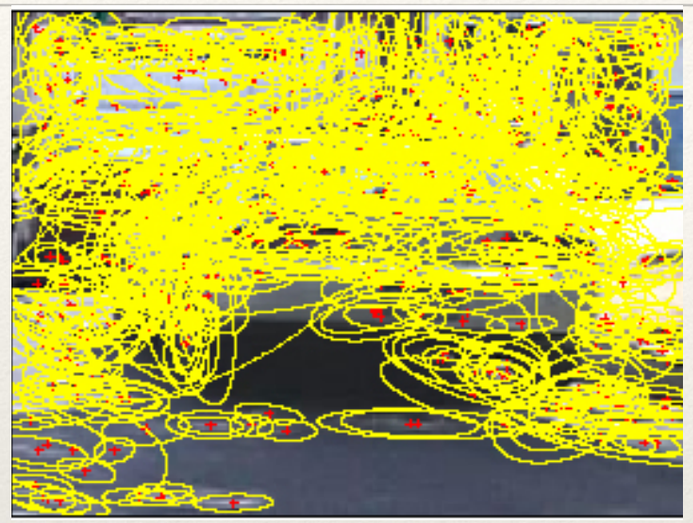
1. Feature extraction
2. Learn “visual vocabulary”
3. Quantize features using visual vocabulary
4. Represent images by frequencies of “visual words”



1. Feature extraction

- Regio

Sampling strategies



Sparse, at interest points

Dense, uniformly

Randomly

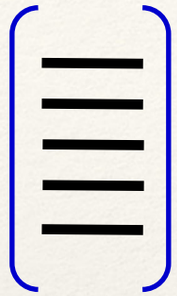
Multiple interest operators

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

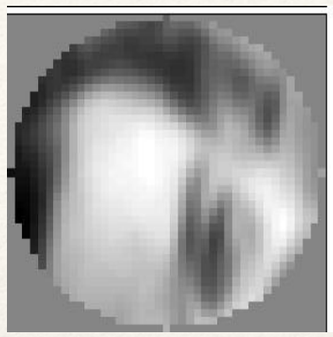
[See Nowak, Jurie & Triggs, ECCV 2006]

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

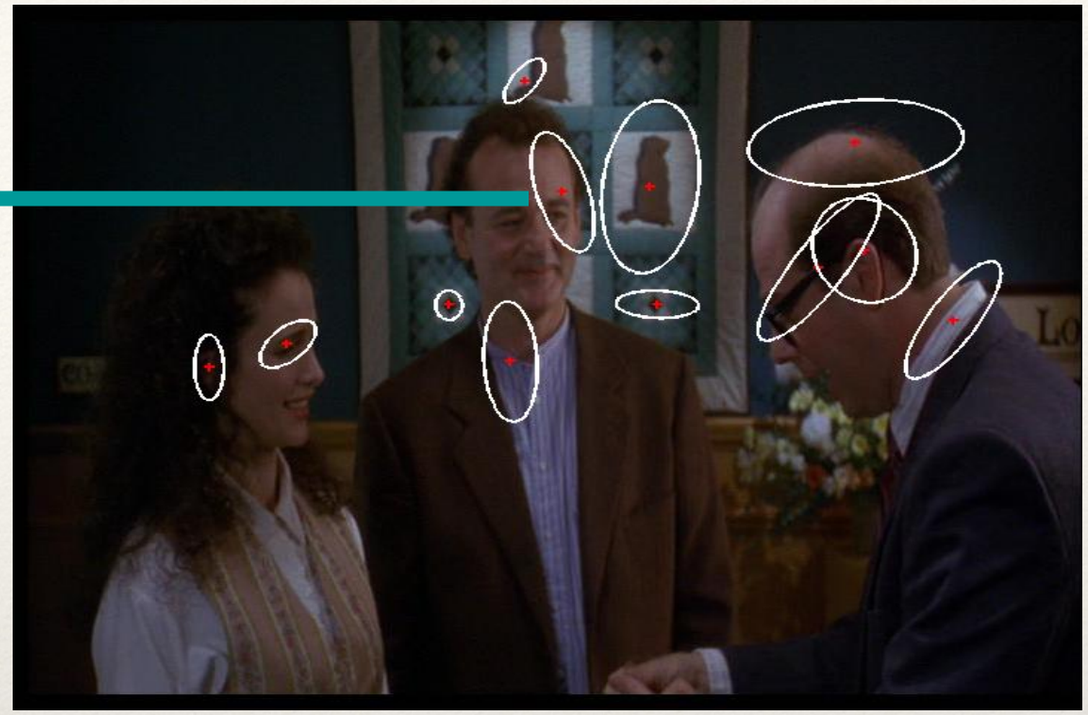
1. Feature extraction



Compute descriptor



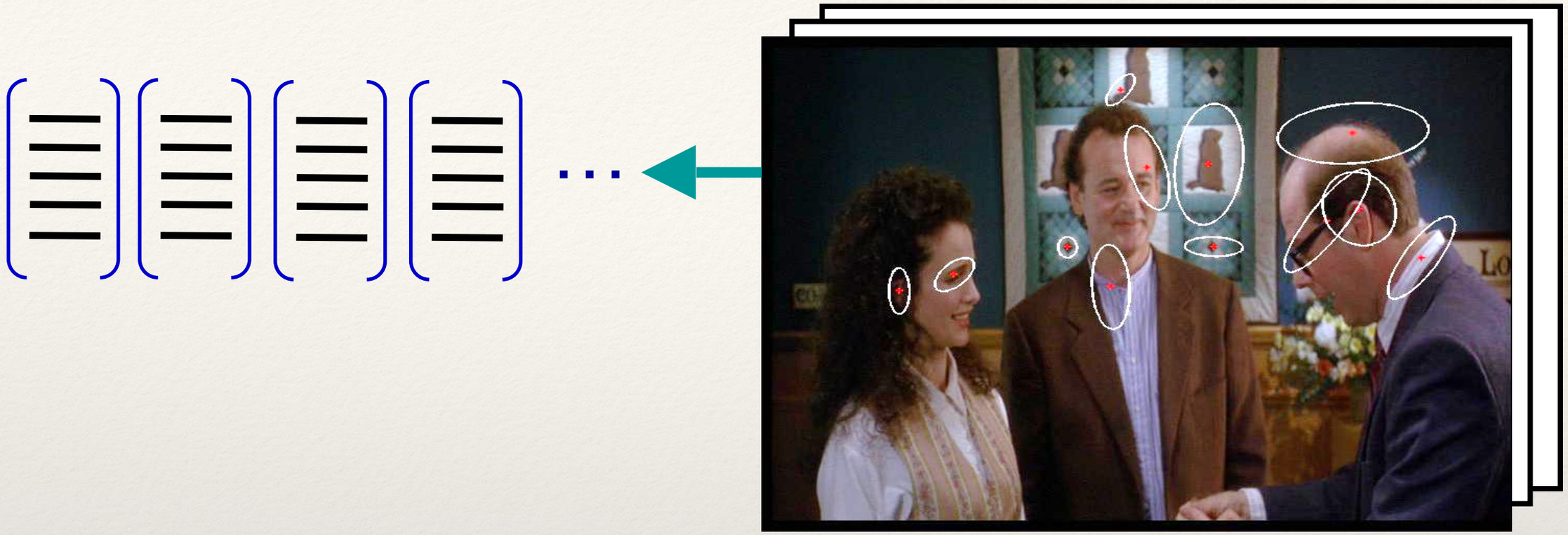
Extract patch



Detect patches

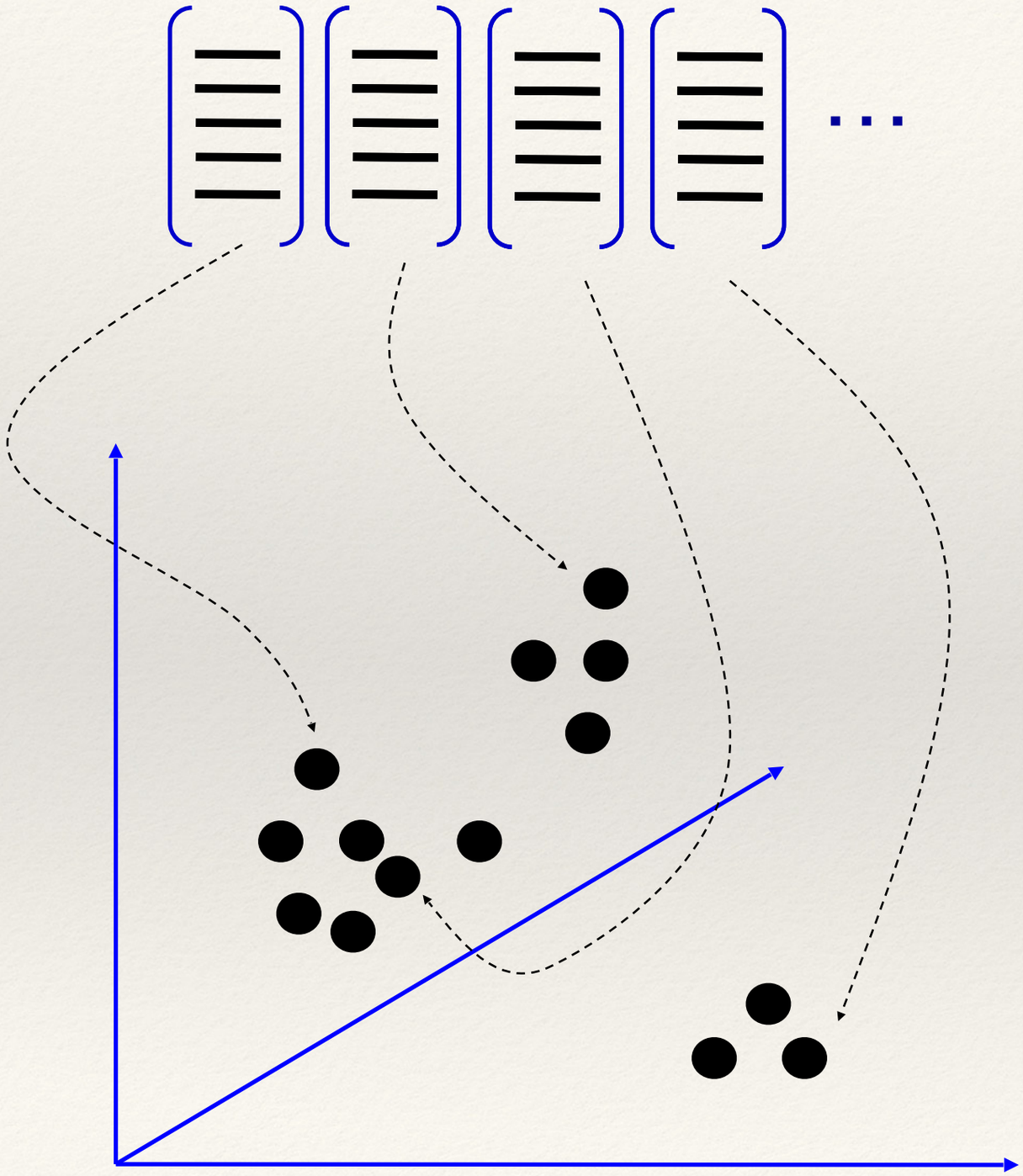
Slide credit: Josef Sivic

1. Feature extraction



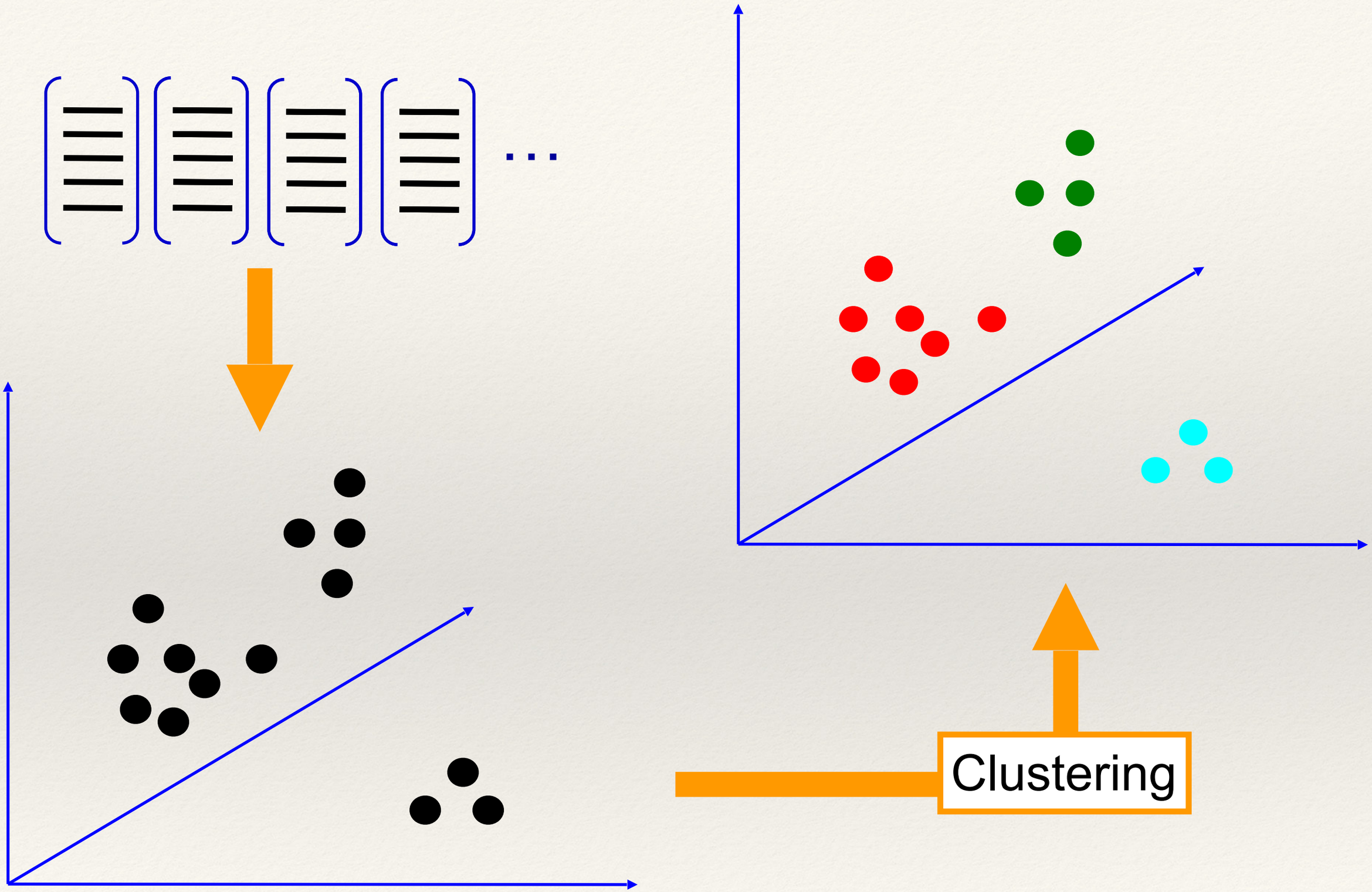
Slide credit: Josef Sivic

2. Learning the visual vocabulary



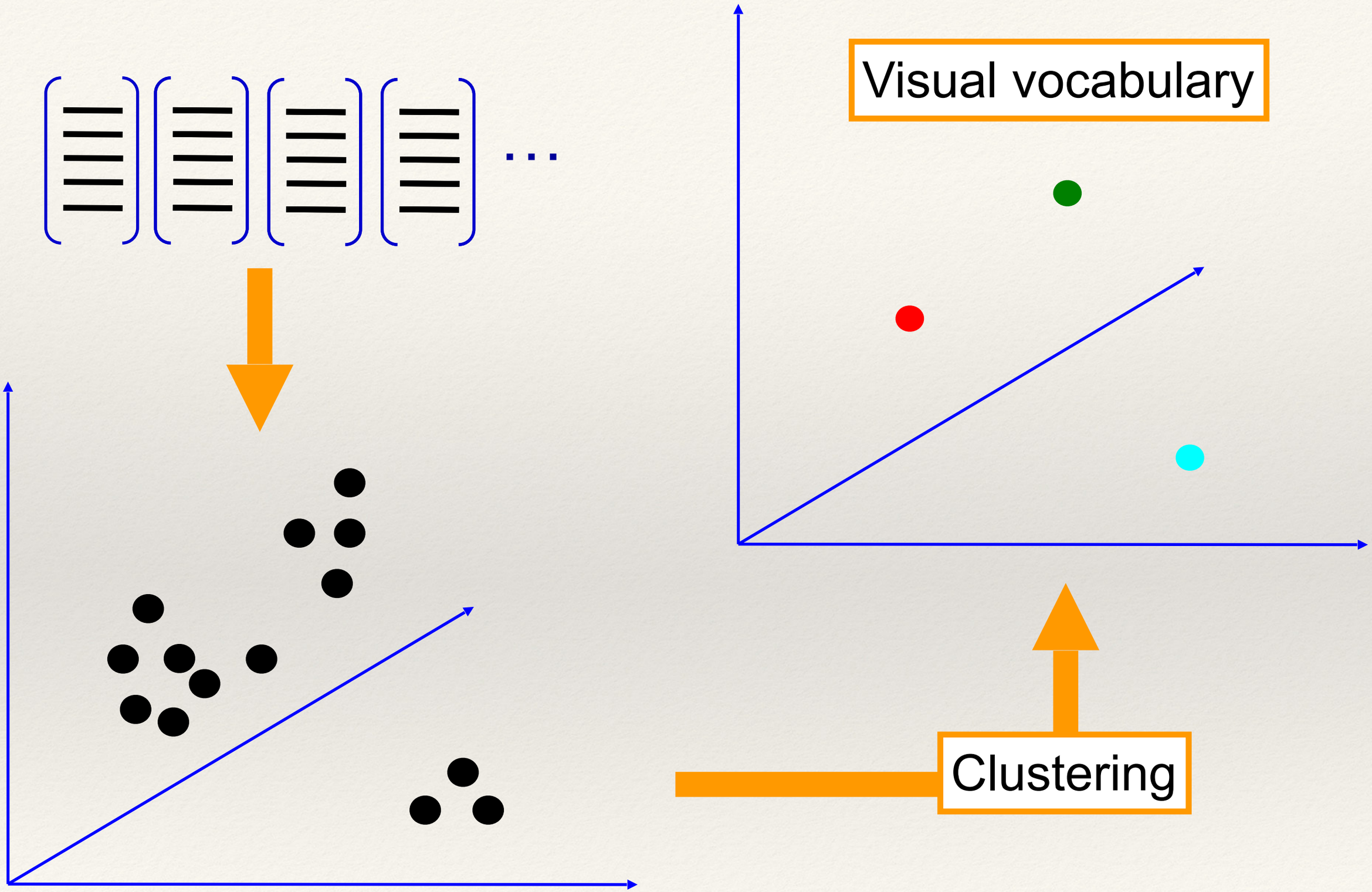
Slide credit: Josef Sivic

2. Learning the visual vocabulary



Slide credit: Josef Sivic

3. Quantize the visual vocabulary



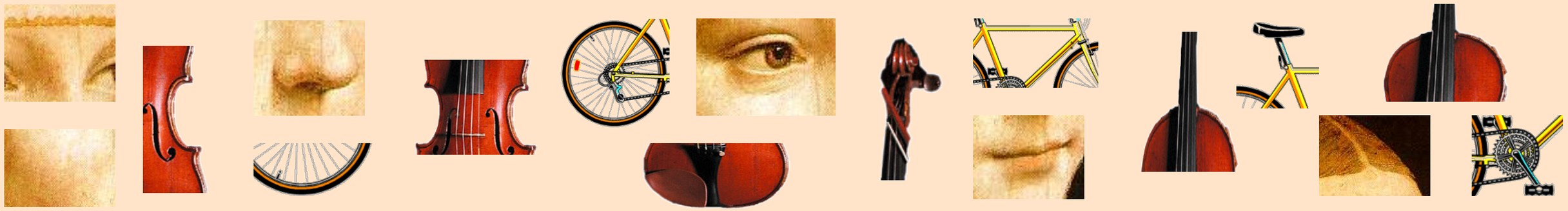
Slide credit: Josef Sivic

Clustering and vector quantization

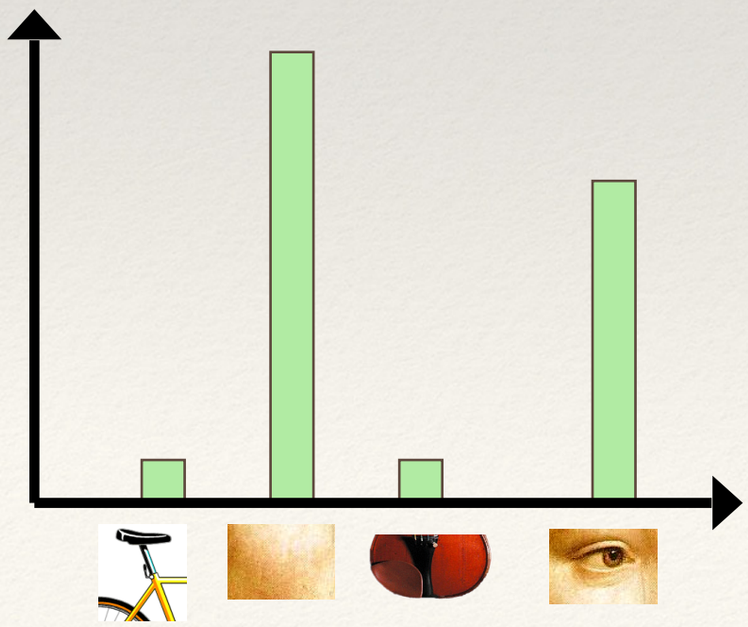
- Clustering is a common method for learning a visual vocabulary or codebook
  - Unsupervised learning process
  - Each cluster center produced by k-means becomes a codevector
  - Codebook can be learned on separate training set
  - Provided the training set is sufficiently representative, the codebook will be “universal”

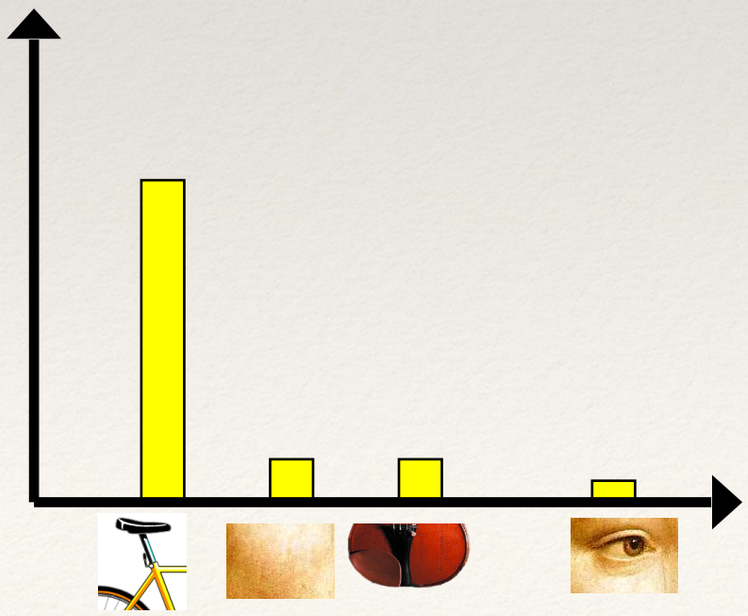
- The codebook is used for quantizing features
  - A *vector quantizer* takes a feature vector and maps it to the index of the nearest codevector in a codebook
  - Codebook = visual vocabulary
  - Codevector = visual word

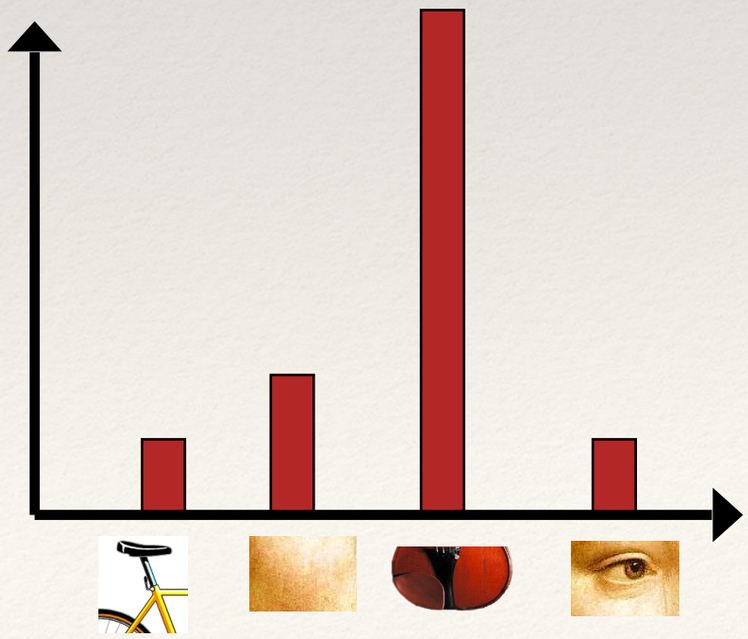
Visual words



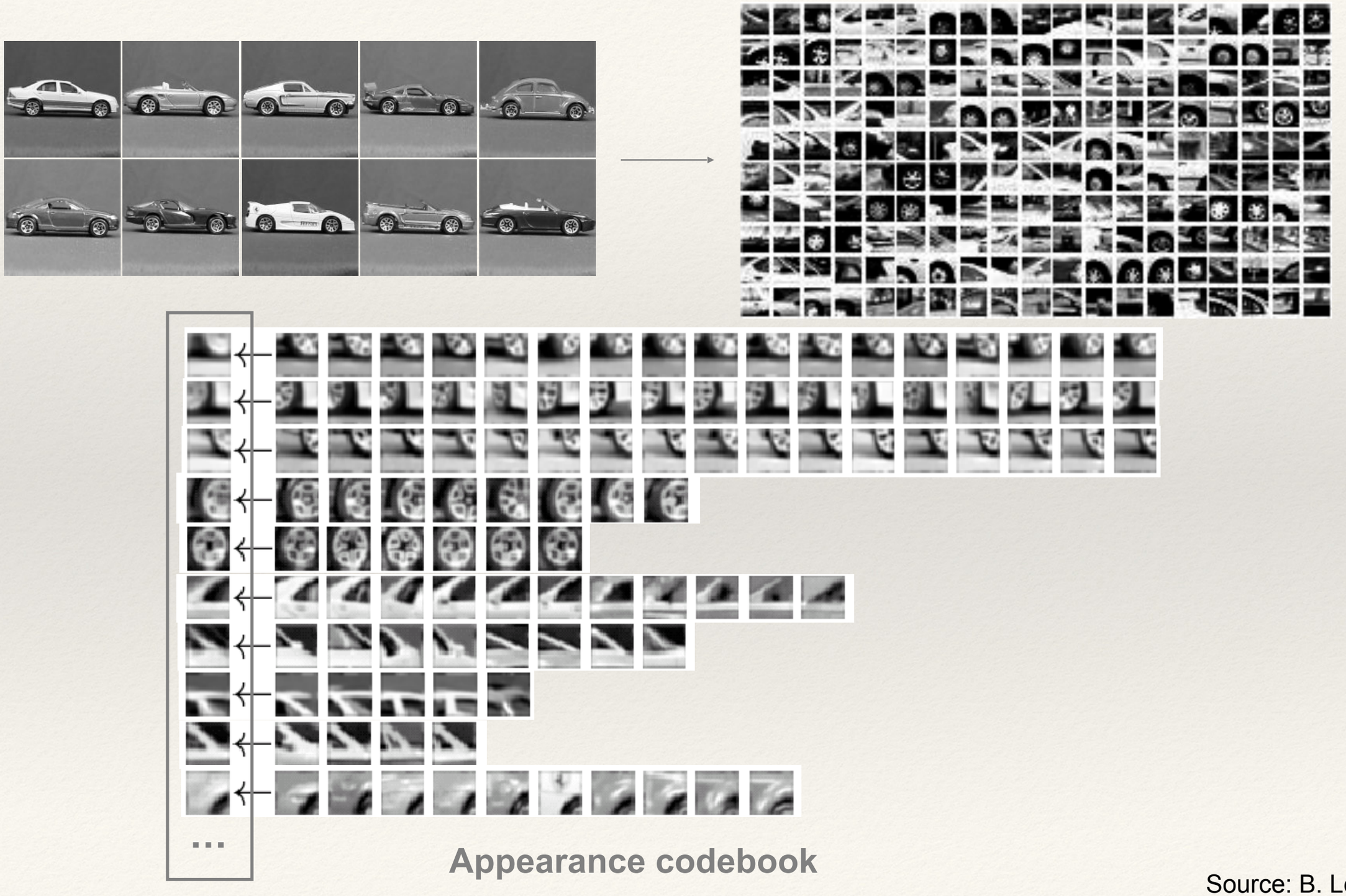
Bag of visual words histograms







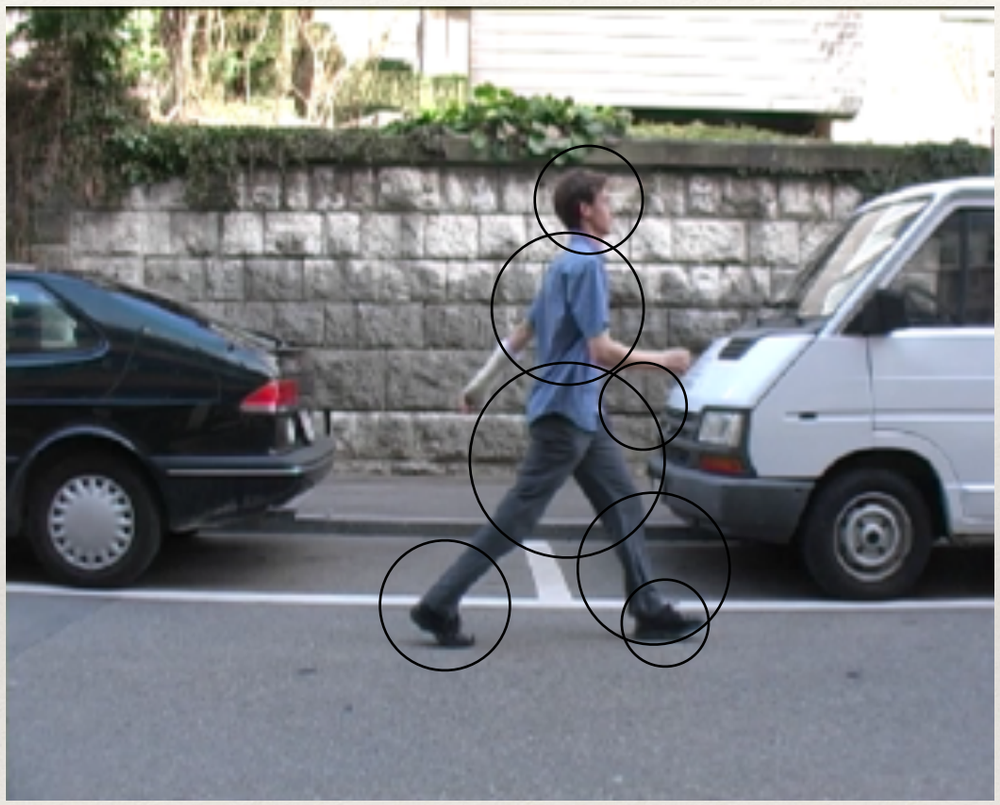
Example real codebook

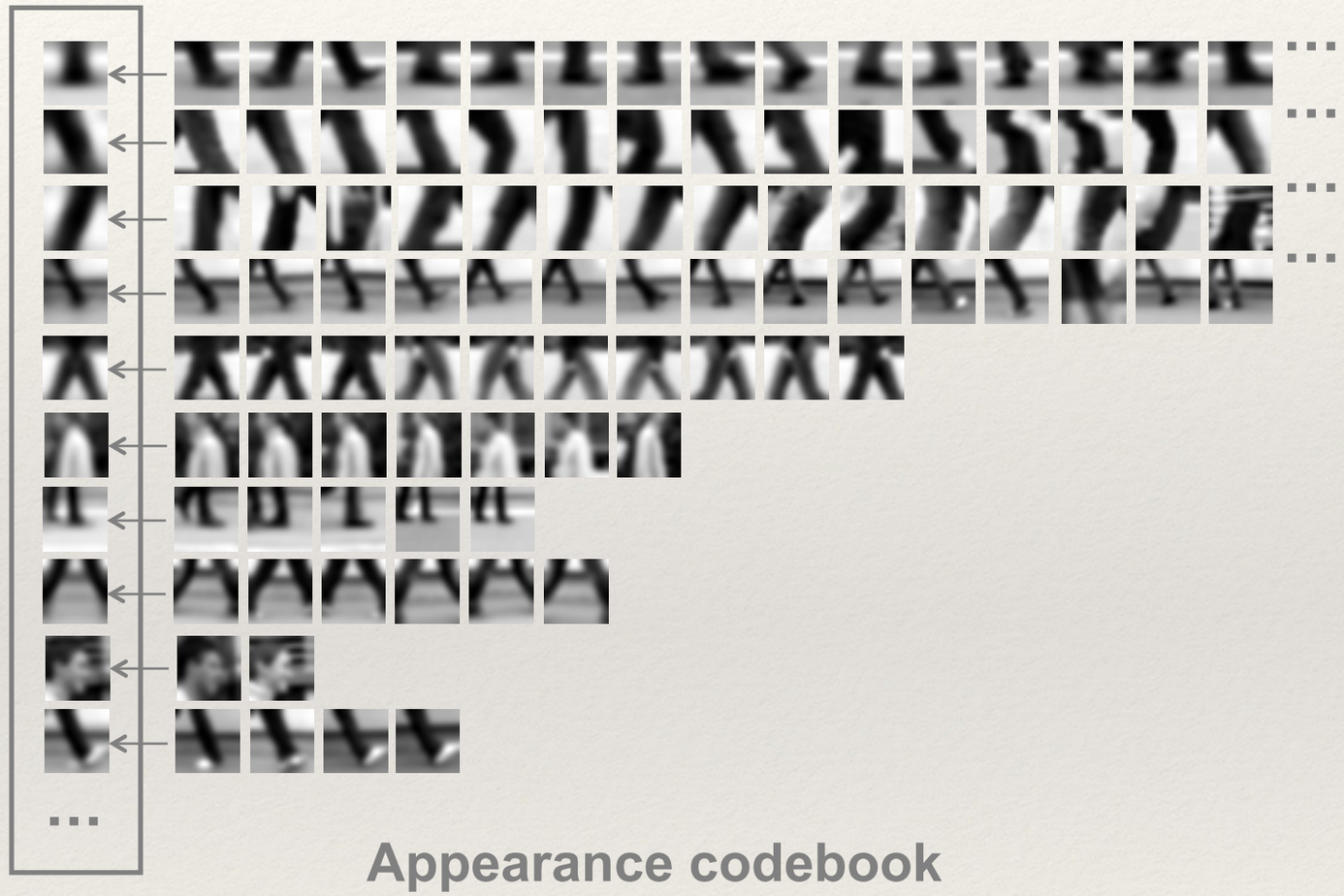


Appearance codebook

Source: B. Leibe

Another codebook



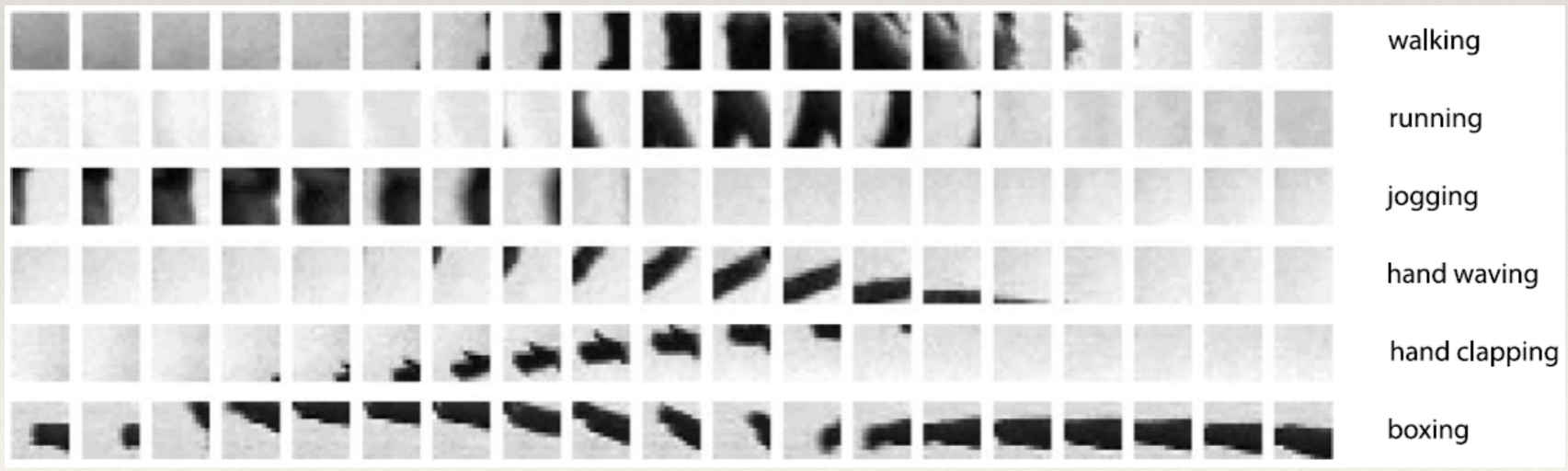


Appearance codebook

Source: B. Leibe

Bags of features for action recognition

Space-time interest points



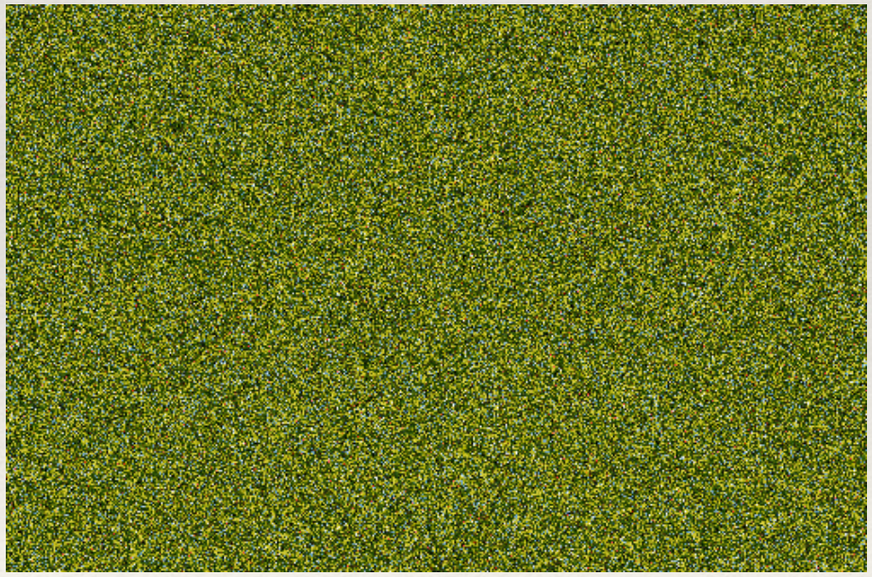
Juan Carlos Niebles, Hongcheng Wang and Li Fei-Fei, **Unsupervised Learning of Human Action Categories Using Spatial-Temporal Words**, IJCV 2008.

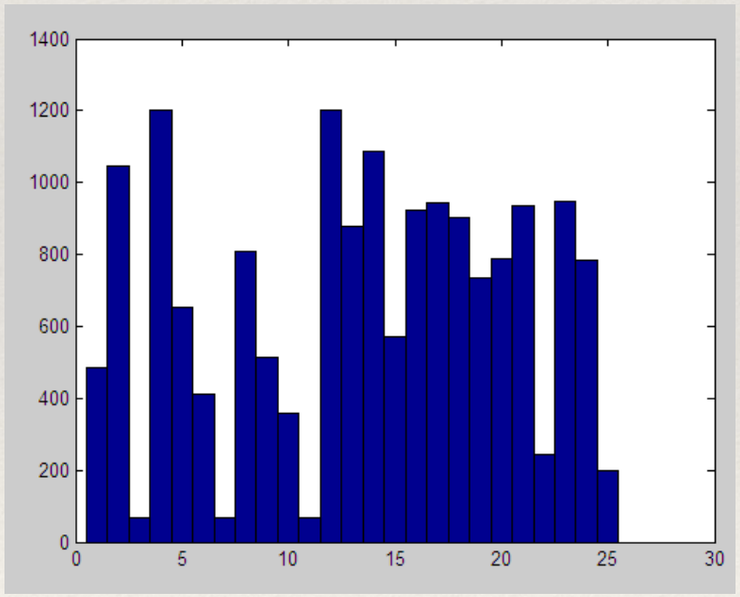
Visual words/bags of words

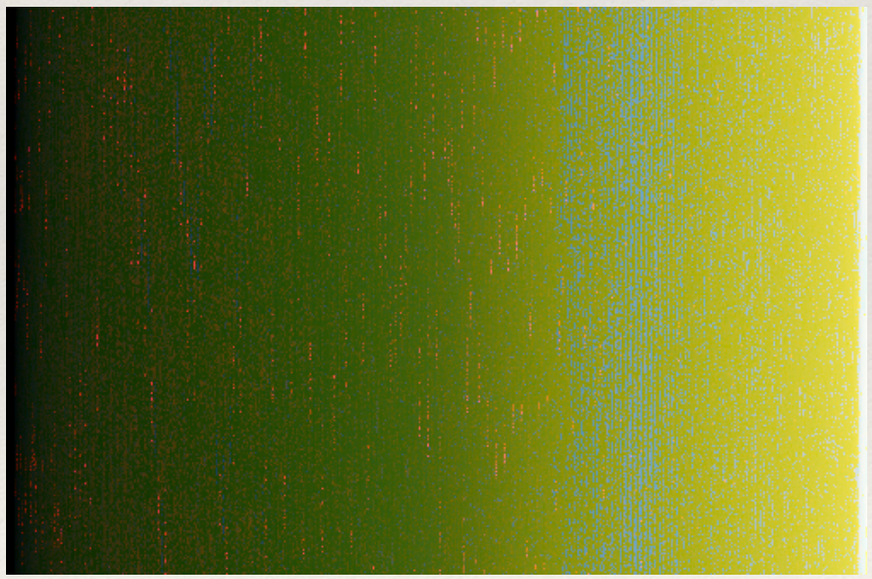
+ flexible to geometry / deformations / viewpoint
+ compact summary of image content
+ provides fixed dimensional vector representation for sets
+ very good results in practice

- background and foreground mixed when bag covers whole image *-> is it really instance recognition?*
- optimal vocabulary formation remains unclear
- basic model ignores geometry – must verify afterwards, or encode via features

But what about layout?







All of these images have the same color histogram.

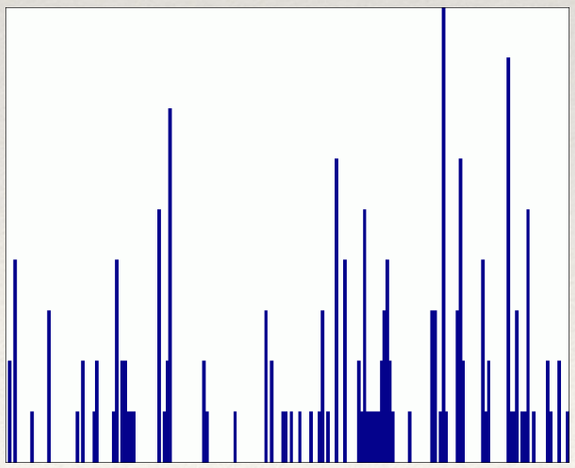
How to extend bag of words?

Spatial pyramid

Compute histogram in each spatial bin

Spatial pyramid representation

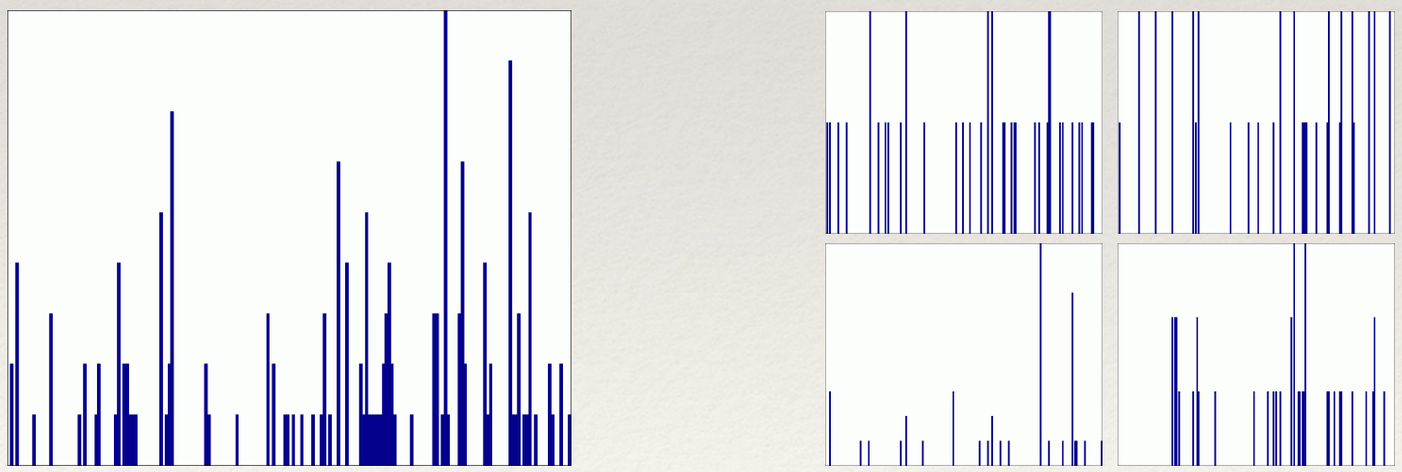
- Extension of a bag of features
- Locally orderless representation at several levels of resolution



Lazebnik, Schmid & Ponce (CVPR 2006)

Spatial pyramid representation

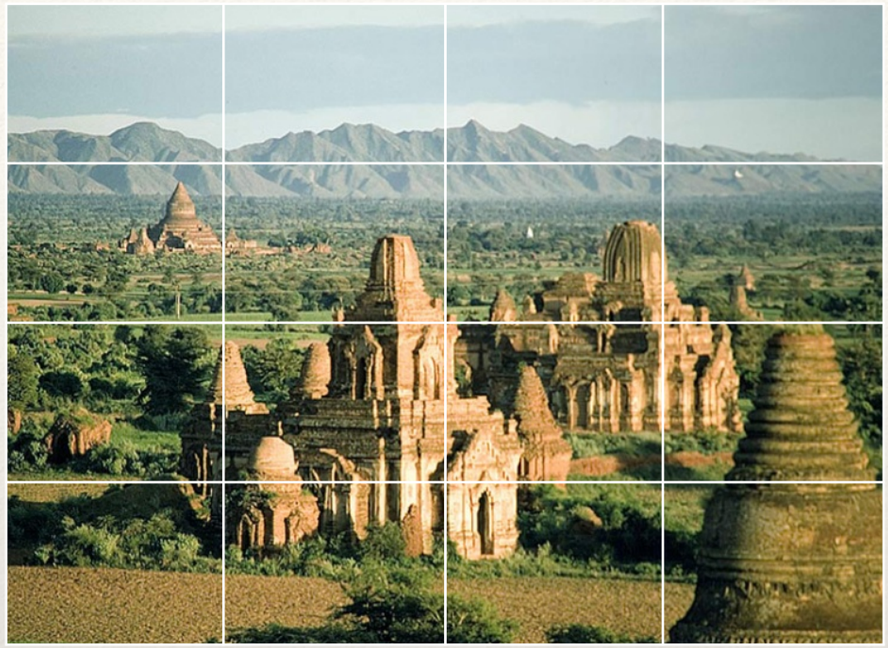
- Extension of a bag of features
- Locally orderless representation at several levels of resolution

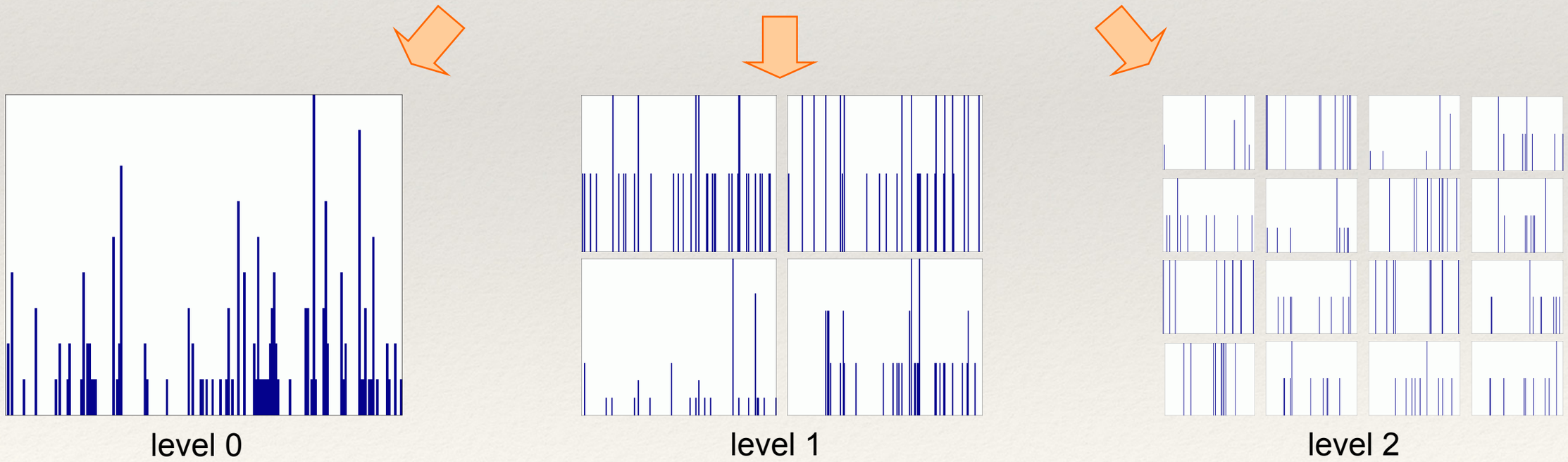


Lazebnik, Schmid & Ponce (CVPR 2006)

Spatial pyramid representation

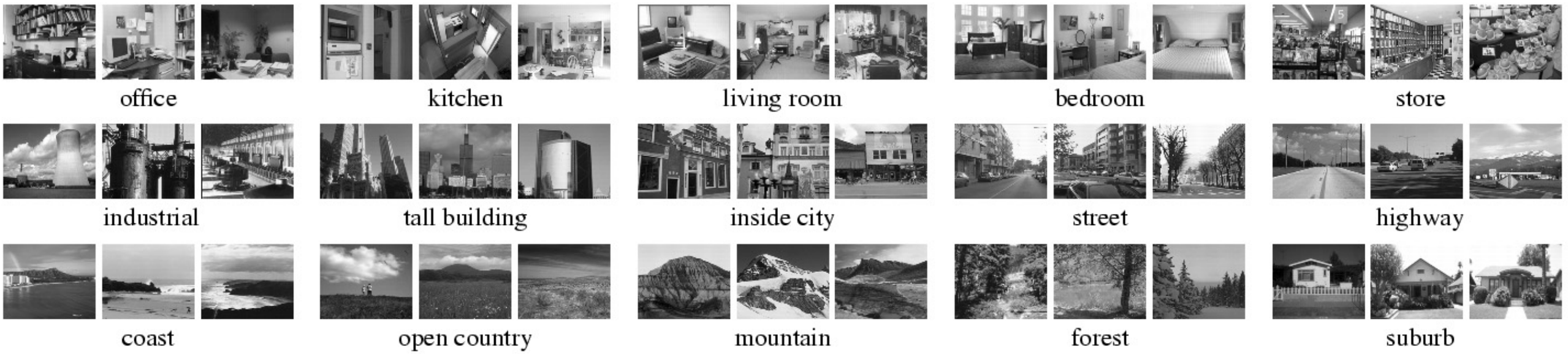
- Extension of a bag of features
- Locally orderless representation at several levels of resolution





Lazebnik, Schmid & Ponce (CVPR 2006)

Scene category dataset



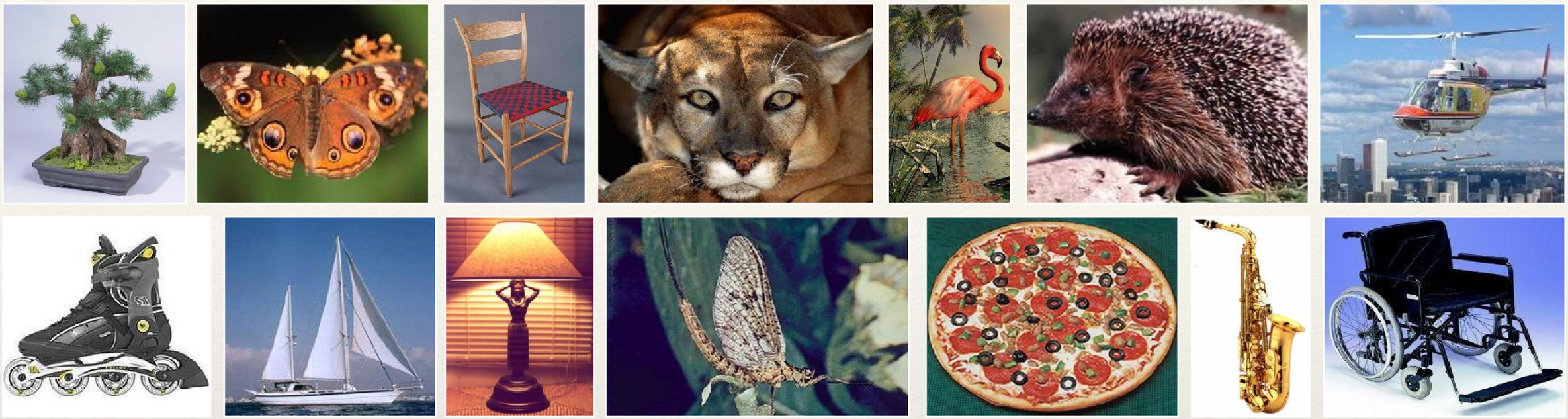
Multi-class classification results

(100 training images per class)

	Weak features (vocabulary size: 16)		Strong features (vocabulary size: 200)	
Level	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

Recognition Issues

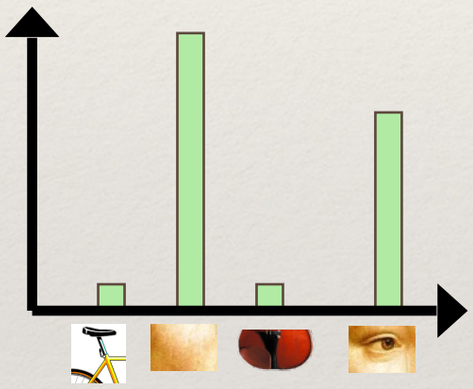
- How to bridge the gap between feature and label?
- How to summarize the content of an entire image?
  How to gauge overall similarity?
- How large should the vocabulary be?
  How to perform quantization efficiently?
- How to score the retrieval results?
- How might we add more spatial verification?

Comparing bags of words

Compute cosine similarity (normalized scalar (dot) product) between their occurrence counts, then rank and pick smallest. *Nearest neighbor* search for similar images.

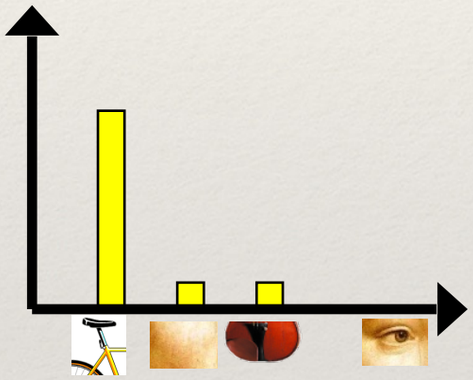
Database image

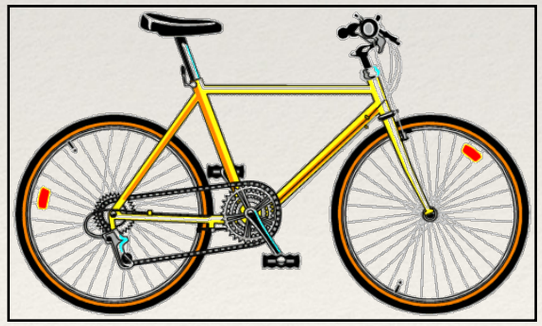
$\vec{d}_j = [1 \quad 8 \quad 1 \quad 4]$



Query

$\vec{q} = [5 \quad 1 \quad 1 \quad 0]$



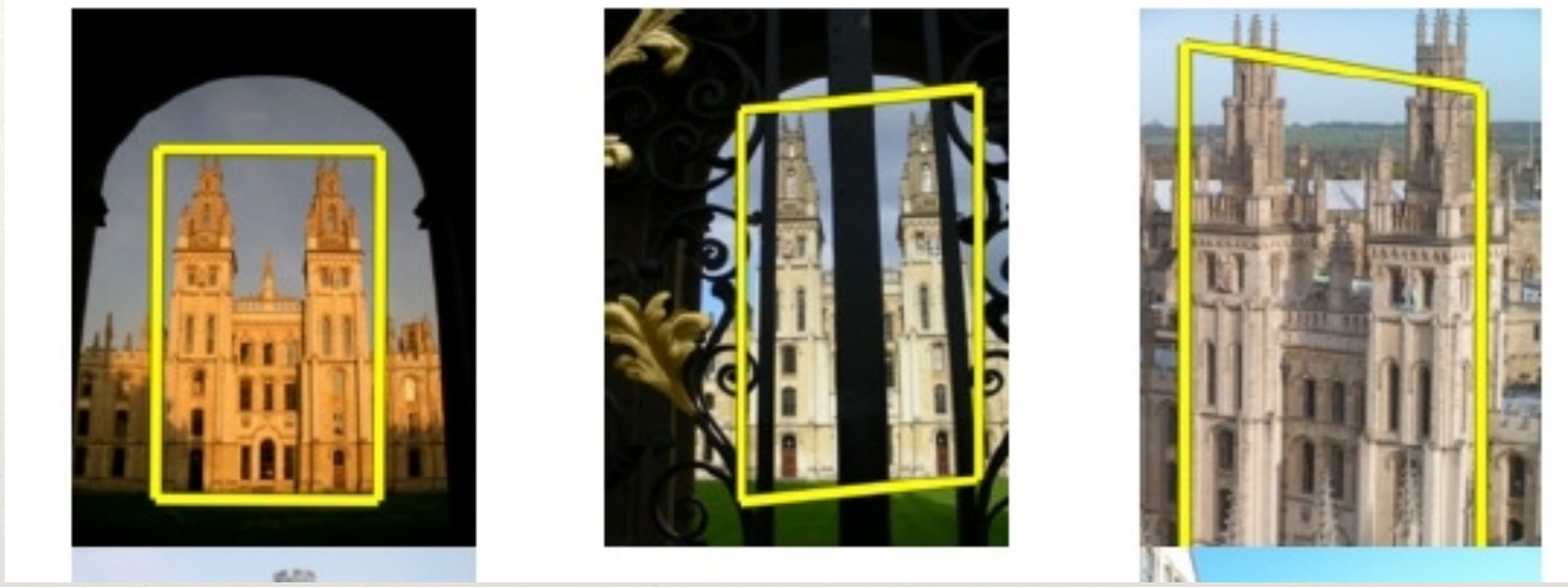


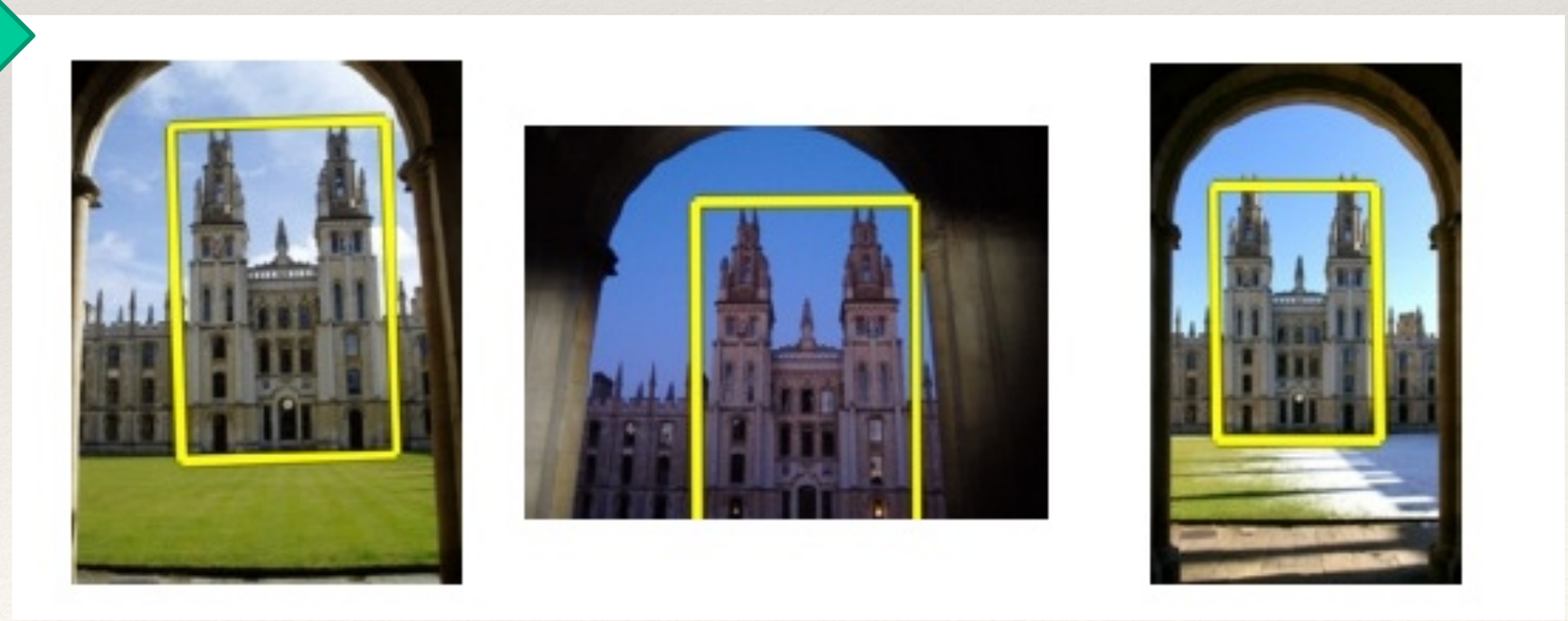
$$sim(d_j, q) = \frac{\langle d_j, q \rangle}{\|d_j\| \|q\|}$$

$$= \frac{\sum_{i=1}^{V} d_j(i) \times q(i)}{\sqrt{\sum_{i=1}^{V} d_j(i)^2} \times \sqrt{\sum_{i=1}^{V} q(i)^2}}$$

for vocabulary of *V* words

- How can we quickly find images in a large database that match a given image region?

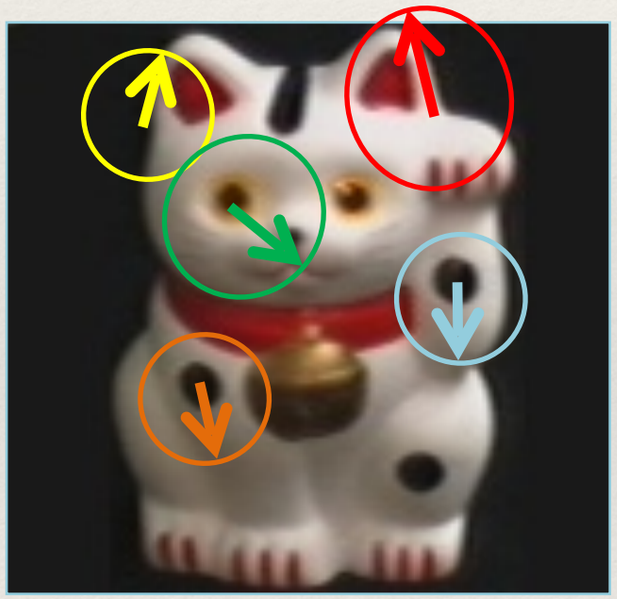




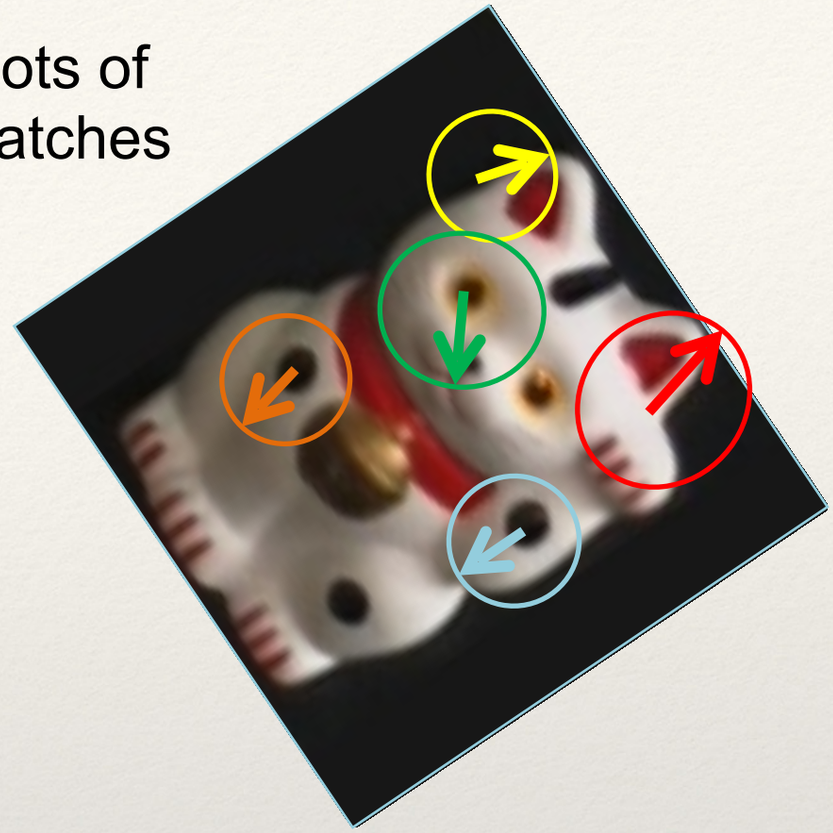
Instance recognition

Simple idea

See how many keypoints are close to keypoints in each other image



Lots of Matches



Few or No Matches

But this will be really, really slow!

Fast lookup: inverted index

Index

- For text documents, an efficient way to find all *pages* on which a *word* occurs is to use an index…
- We want to find all *images* in which a *feature* occurs.

Build Inverted Index from Database

Image #1

Image #2

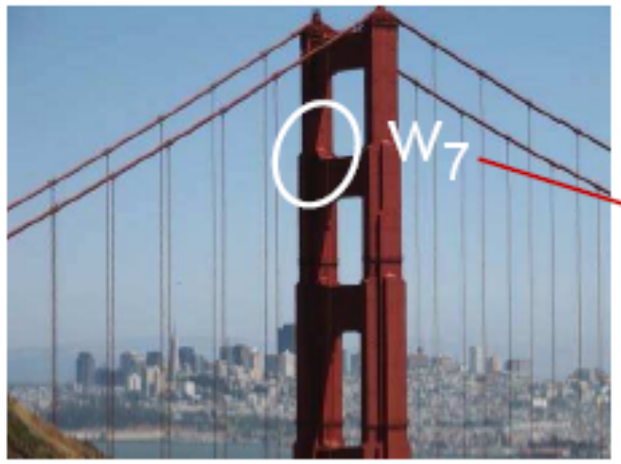
Image #3

Word #	Image #
1	3
2 …	
7	1, 2
8	3
9	
10 …	
91	2

…

Kristen Grauman

Query Inverted Index



New query image

Word #	Image #
1	3
2	
…	
7	1, 2
…	
8	3
9	
10	
…	
91	2

⋮ ⋮

Candidate matches

Image #1

Image #2

Query Inverted Index

New query image

Word #	Image #
1	3
2	
…	
7	1, 2
8	3
9	
10	
…	
91	2
⋮	⋮

Candidate matches

Image #1

Image #2

1. Extract words in query
2. Inverted file index to find relevant frames
3. Compare/sort word counts

Inverted index

Key requirement: *sparsity*.

If most images contain most words, then we're not better off than exhaustive search.

- Exhaustive search would mean comparing the visual word distribution of a query versus every page.

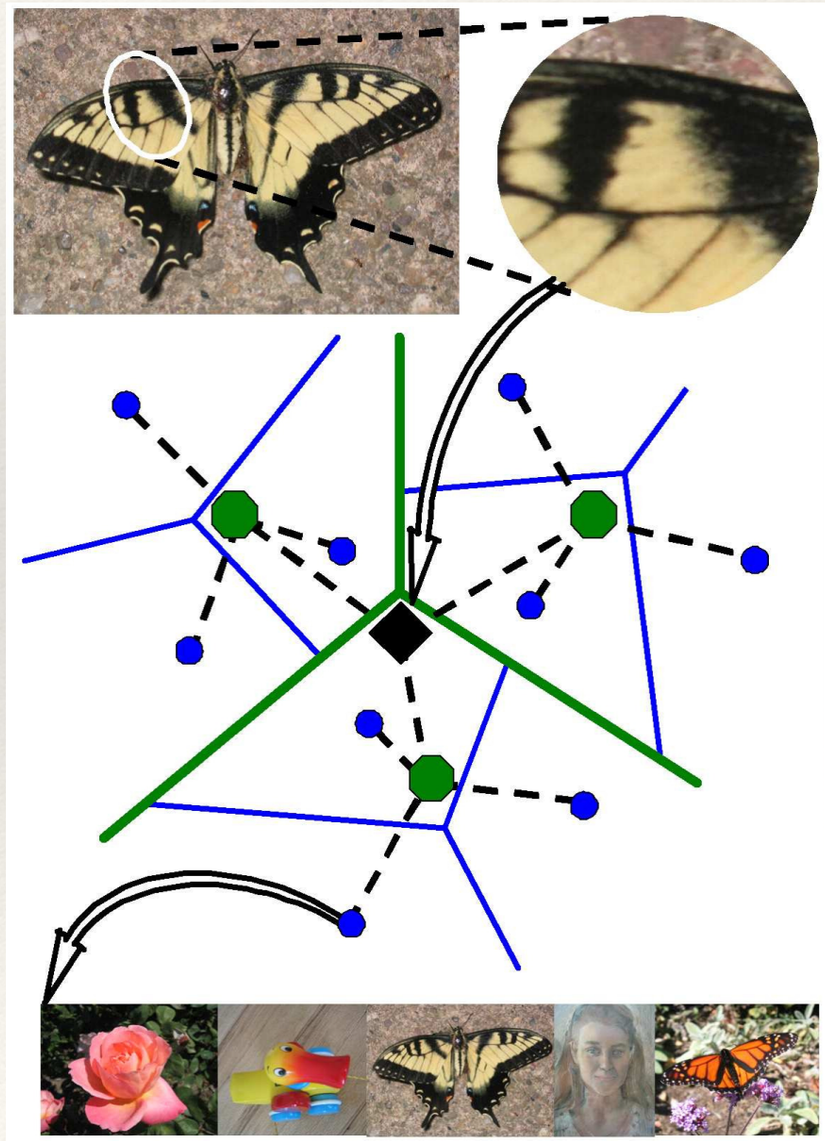
Recognition Issues

- How to bridge the gap between feature and label?
- How to summarize the content of an entire image? How to gauge overall similarity?
- How large should the vocabulary be? How to perform quantization efficiently?
- How to score the retrieval results?
- How might we add more spatial verification?

Kristen Grauman

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)

Recognition Issues

- How to bridge the gap between feature and label?
- How to summarize the content of an entire image? How to gauge overall similarity?
- How large should the vocabulary be? How to perform quantization efficiently?
- How to score the retrieval results?
- How might we add more spatial verification?

Kristen Grauman

Precision and Recall

True positive (tp) – correct attribution
True negative (tn) – correct rejection

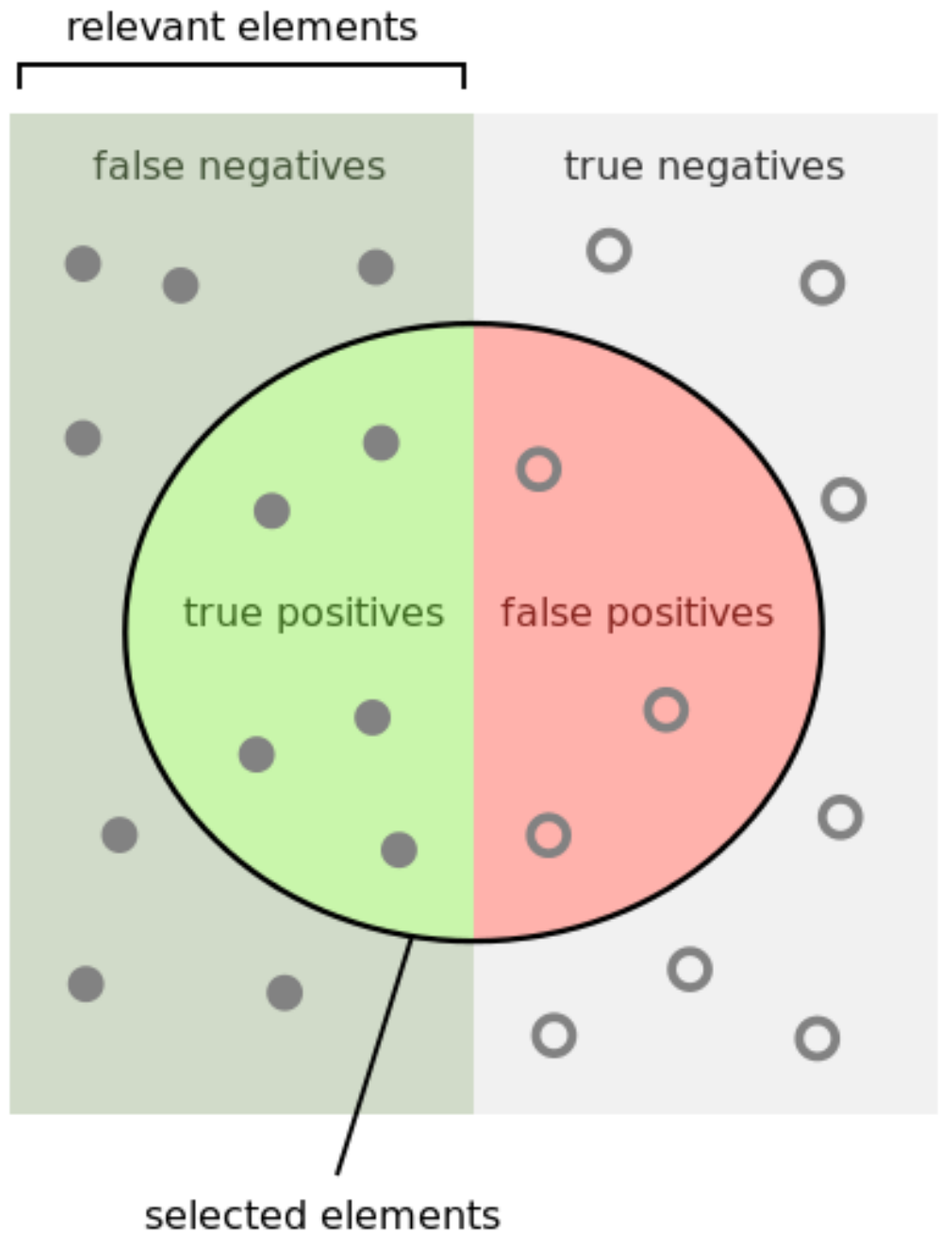
False positive (fp) – incorrect attribution
False negative (fn) – incorrect rejection

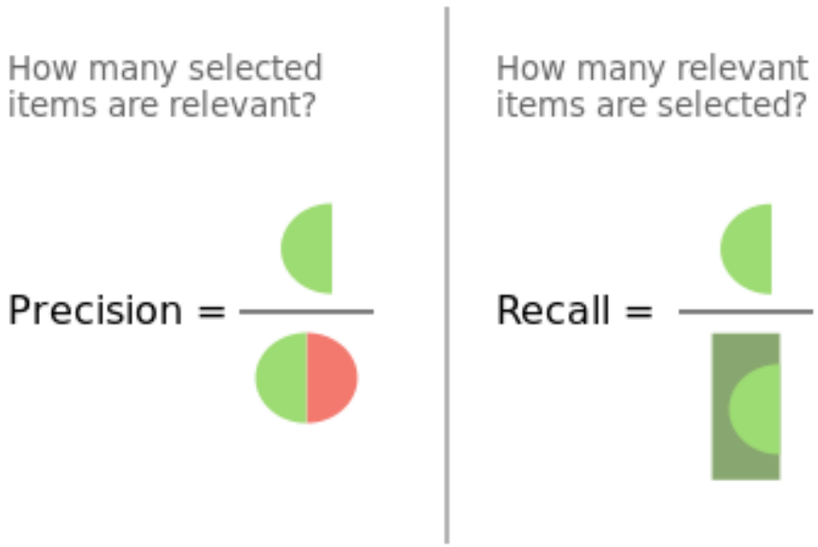
$$\text{Precision} = \frac{tp}{tp + fp}$$

Precision = #relevant / #returned

$$\text{Recall} = \frac{tp}{tp + fn}$$

Recall = #relevant / #total relevant



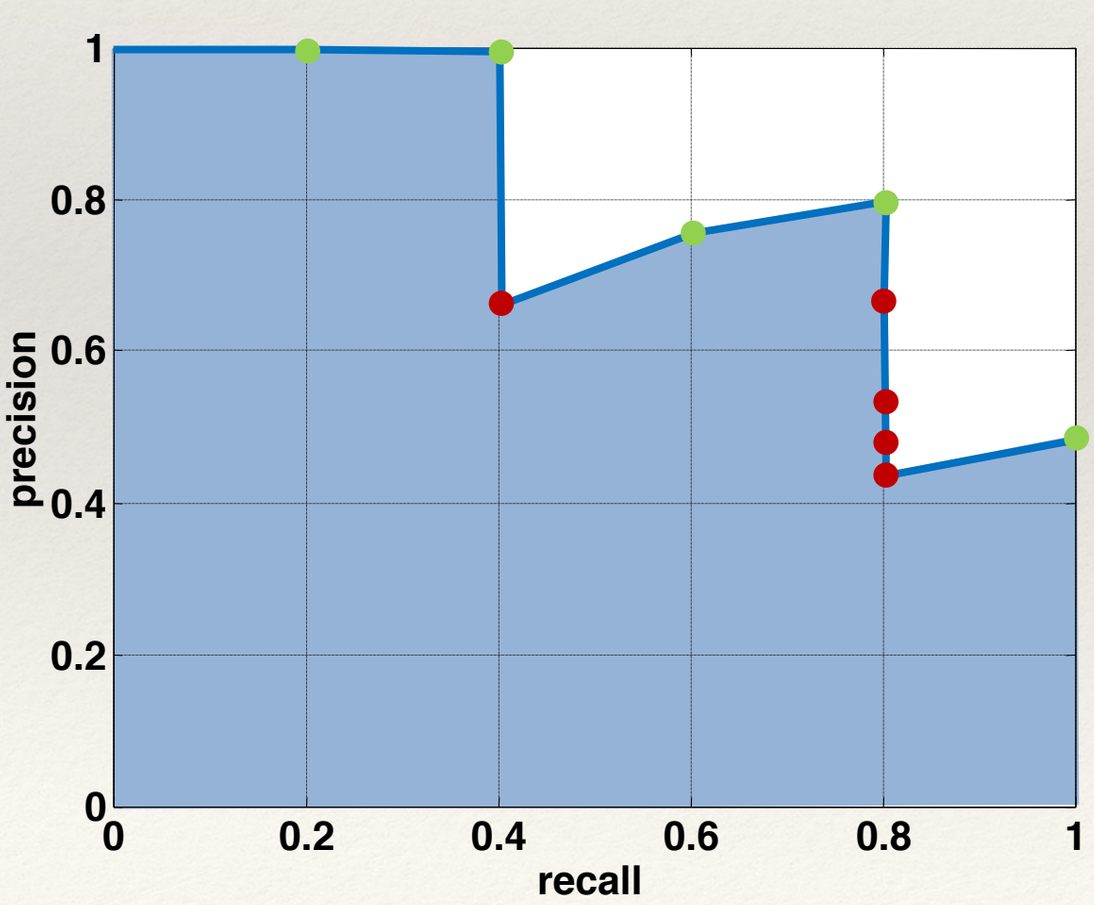


Scoring retrieval quality

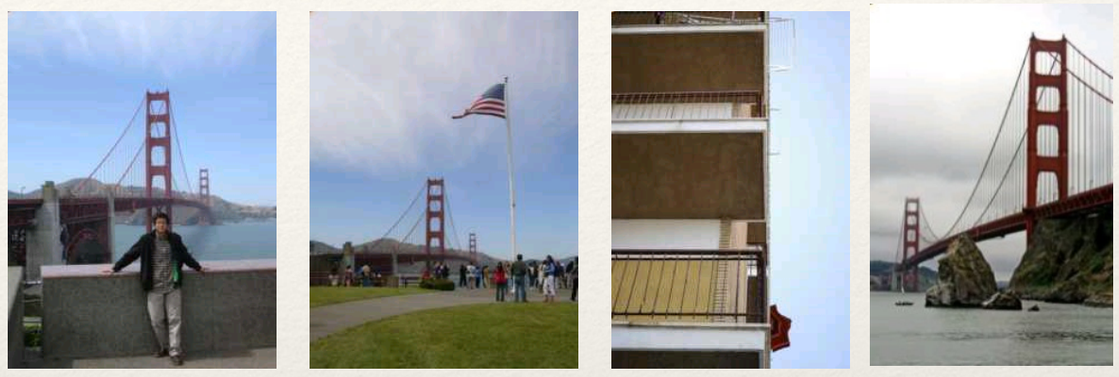
Query

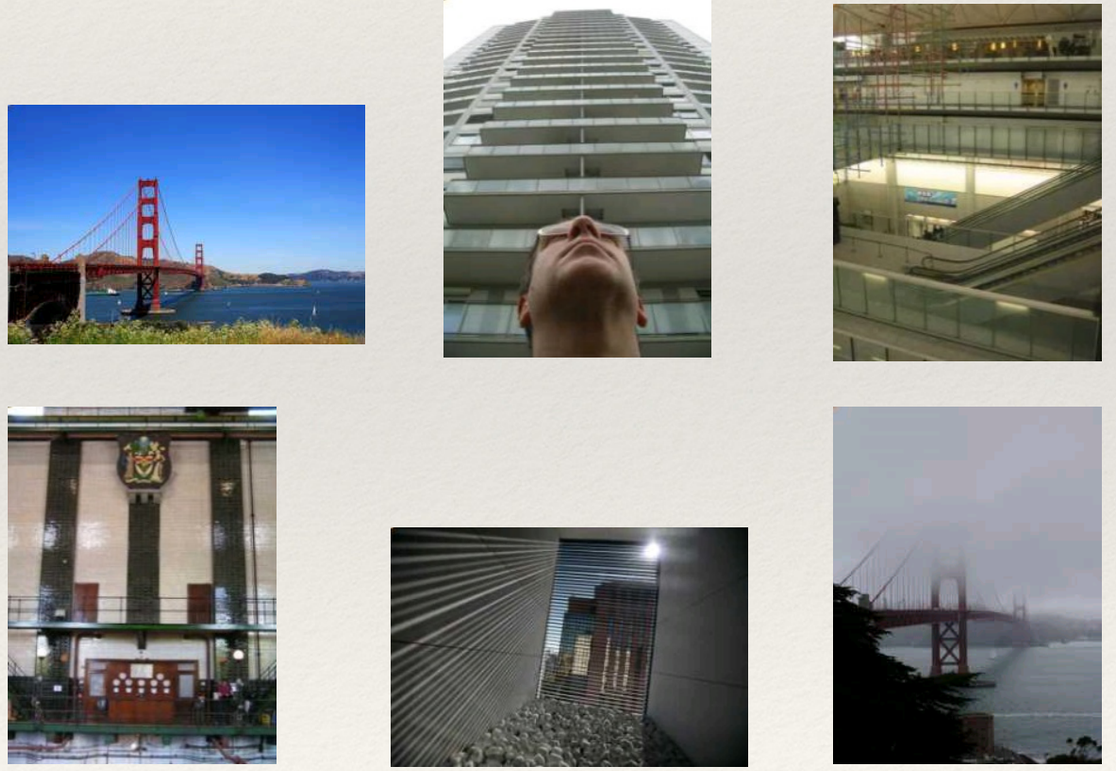
Database size: 10 images
Relevant (total): 5 images

precision = #relevant / #returned
recall = #relevant / #total relevant



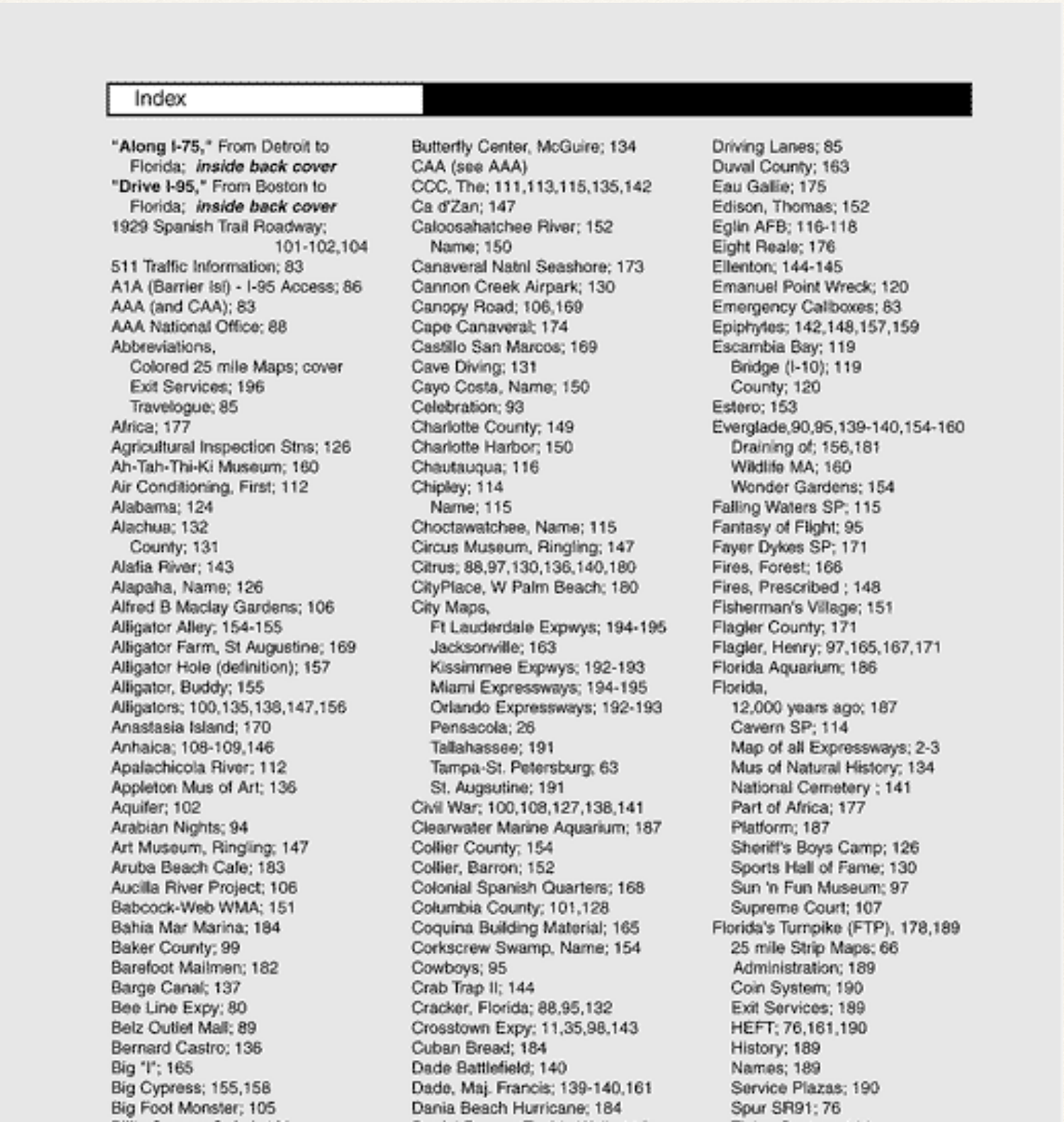
Results (ordered):





Slide credit: Ondrej Chum

What else can we borrow from text retrieval?



Index

China is forecasting a trade surplus of $90bn (£51bn) to $100bn this year, a threefold increase on 2004's $32bn. T[illegible]istry said the surplus would [illegible]0% jump in exports t[illegible]rise in imports to[illegible]ther annoy th[illegible]a's exports underv[illegible] too high Bank of country a[illegible]c demand s[illegible]y. China increa[illegible]the dollar by 2.1% [illegible]e within a narrow ba[illegible] yuan to be allowed to trade fre[illegible]g has made it clear that it will take its time an[illegible] carefully before allowing the yuan to rise[illegible] in value.

China, trade, surplus, commerce, exports, imports, US, yuan, bank, domestic, foreign, increase, trade, value

tf-idf weighting

- **T**erm **f**requency – **i**nverse **d**ocument **f**requency
- Describe image by frequency of each word within it, downweight words that appear often in the database
- (Standard weighting for text retrieval)

Number of occurrences of word i in document d

Number of words in document d

$$t_i = \frac{n_{id}}{n_d} \log \frac{N}{n_i}$$

Total number of documents in database

Number of documents word i occurs in, in whole database

Kristen Grauman

Query expansion

Query: ***golf green***

Results:

- How can the grass on the ***greens*** at a ***golf*** course be so perfect?
- For example, a skilled ***golf***er expects to reach the ***green*** on a par-four hole in **...**
- Manufactures and sells synthetic ***golf*** putting ***green****s* and mats.

Irrelevant result can cause a `topic drift':

- Volkswagen ***Golf***, 1999, ***Green***, 2000cc, petrol, manual, , hatchback, 94000miles, 2.0 GTi, 2 Registered Keepers, HPI Checked, Air-Conditioning, Front and Rear Parking Sensors, ABS, Alarm, Alloy

Slide credit: Ondrej Chum

Query expansion

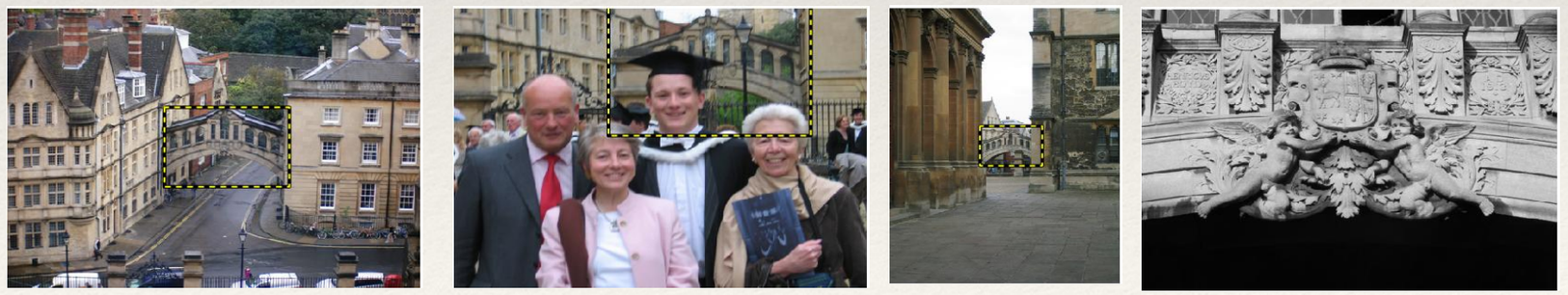
Results

Spatial verification

Query image

New query

New results



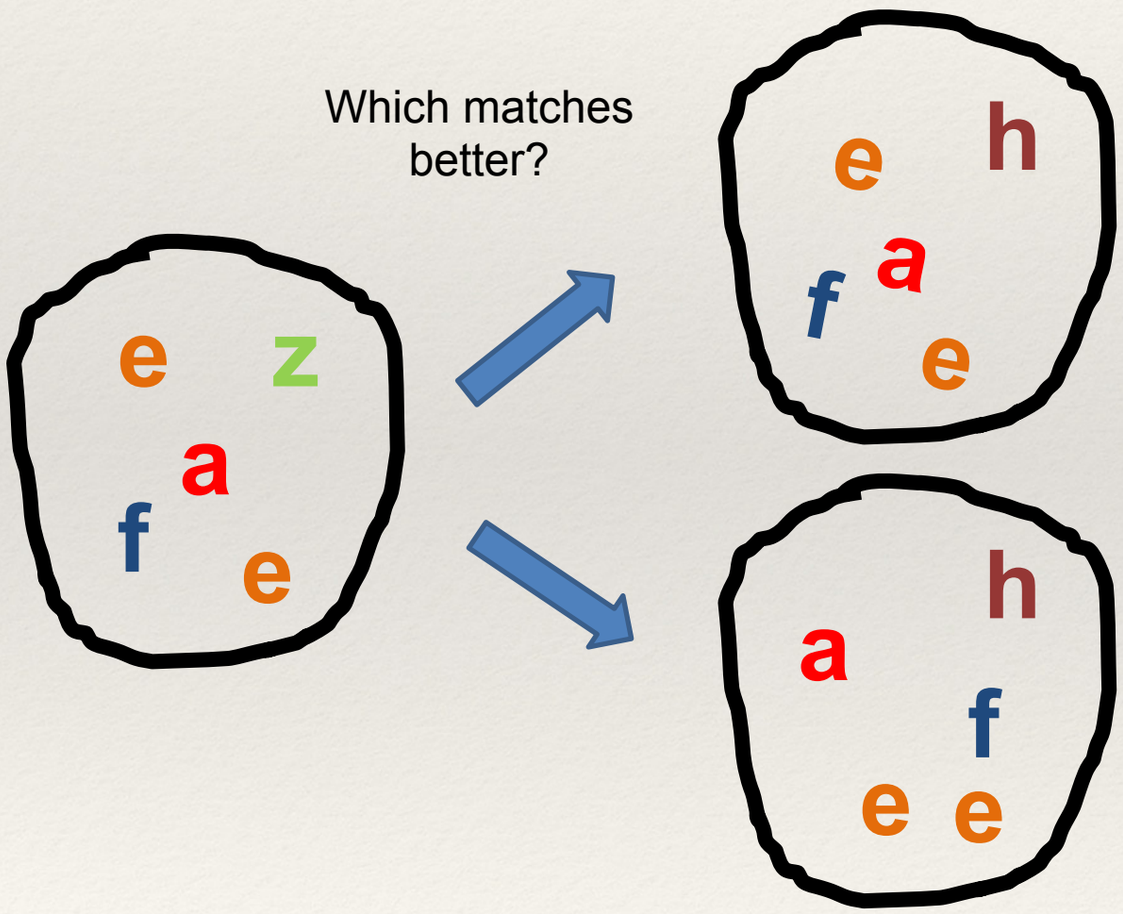
Chum, Philbin, Sivic, Isard, Zisserman: Total Recall…, ICCV 2007

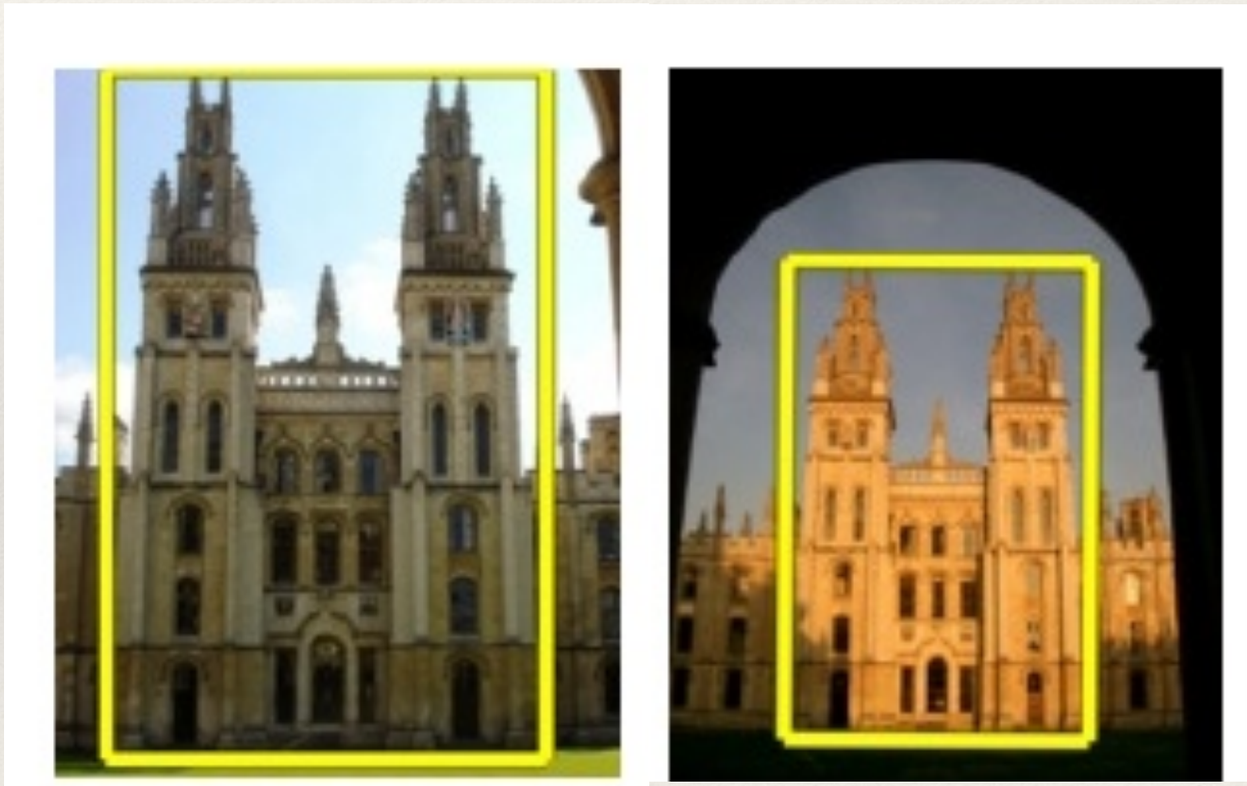
Recognition Issues

- How to bridge the gap between feature and label?
- How to summarize the content of an entire image? How to gauge overall similarity?
- How large should the vocabulary be? How to perform quantization efficiently?
- How to score the retrieval results?
- How might we add more spatial verification?

Can we be more accurate?

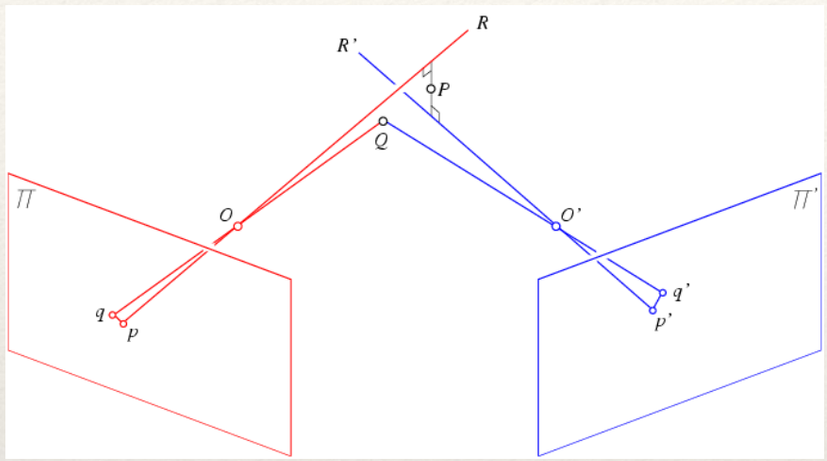
So far, we treat each image as containing a “bag of words”, with no spatial information

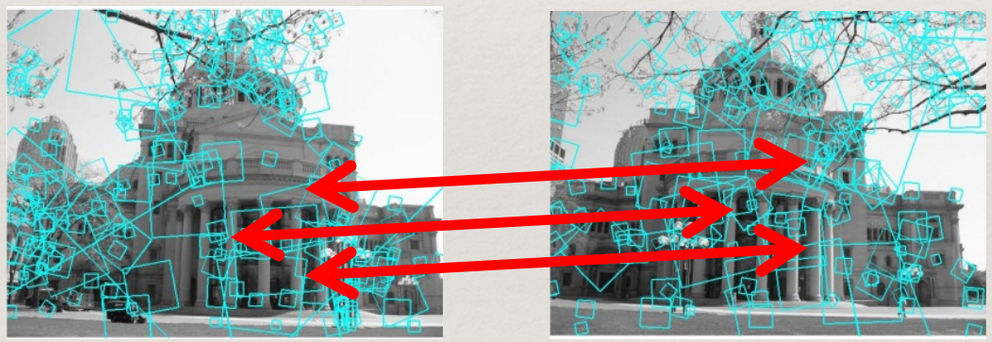




Real objects have consistent geometry

Multi-view matching

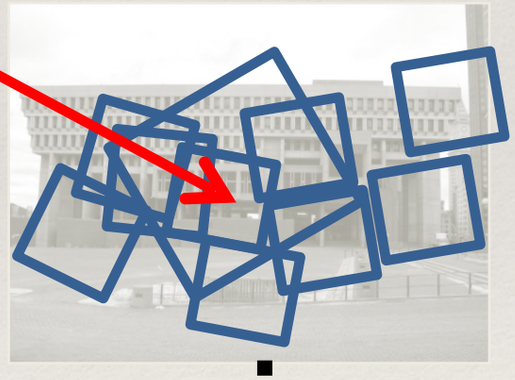




Matching two given views for depth

vs

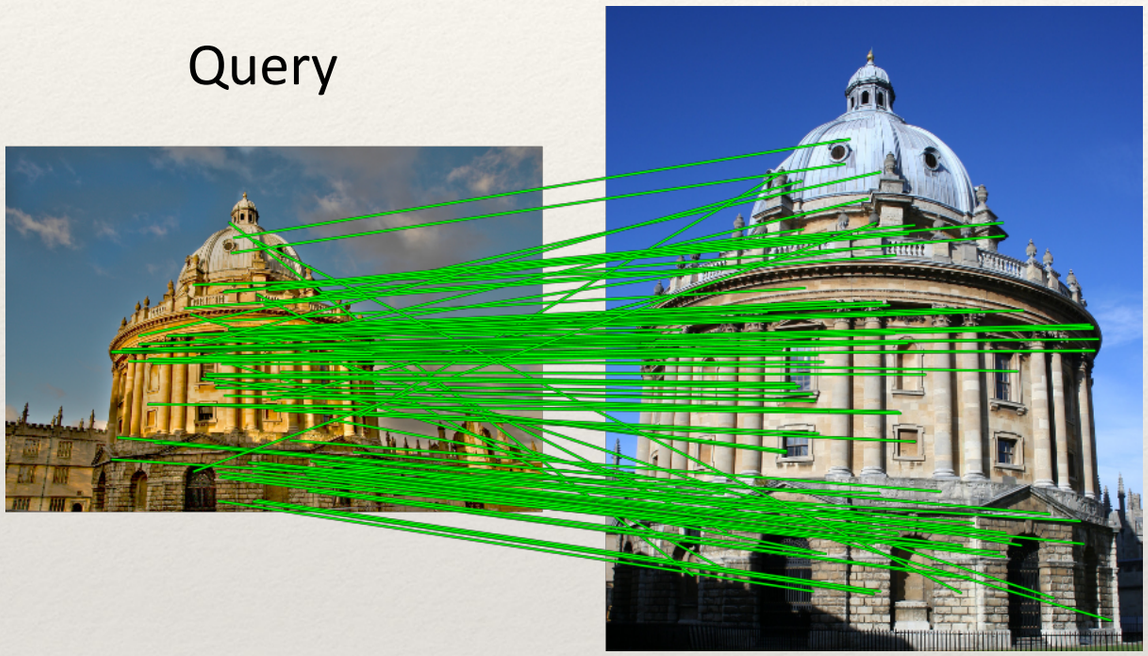
?



Search for a matching view for recognition

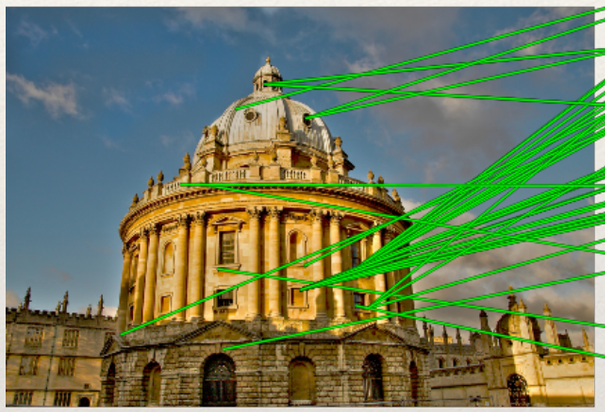
Spatial Verification

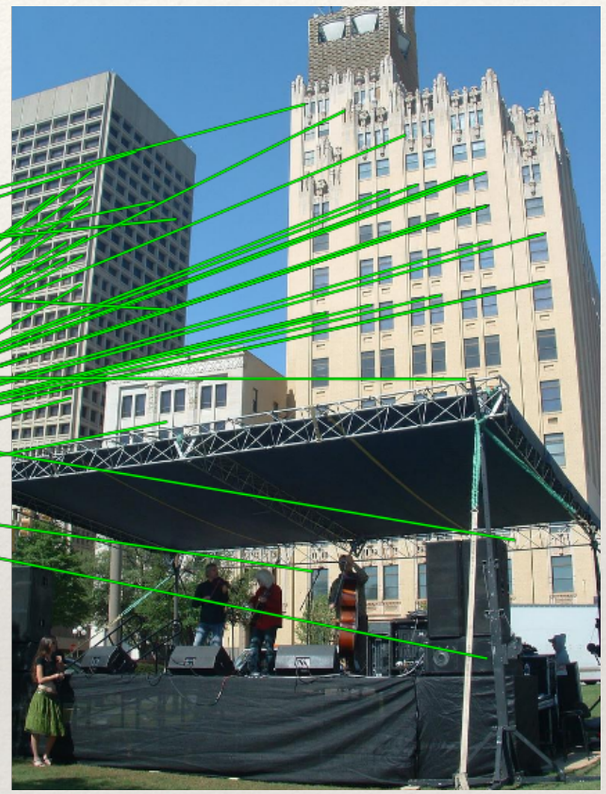
Query



DB image with high BoW similarity

Query





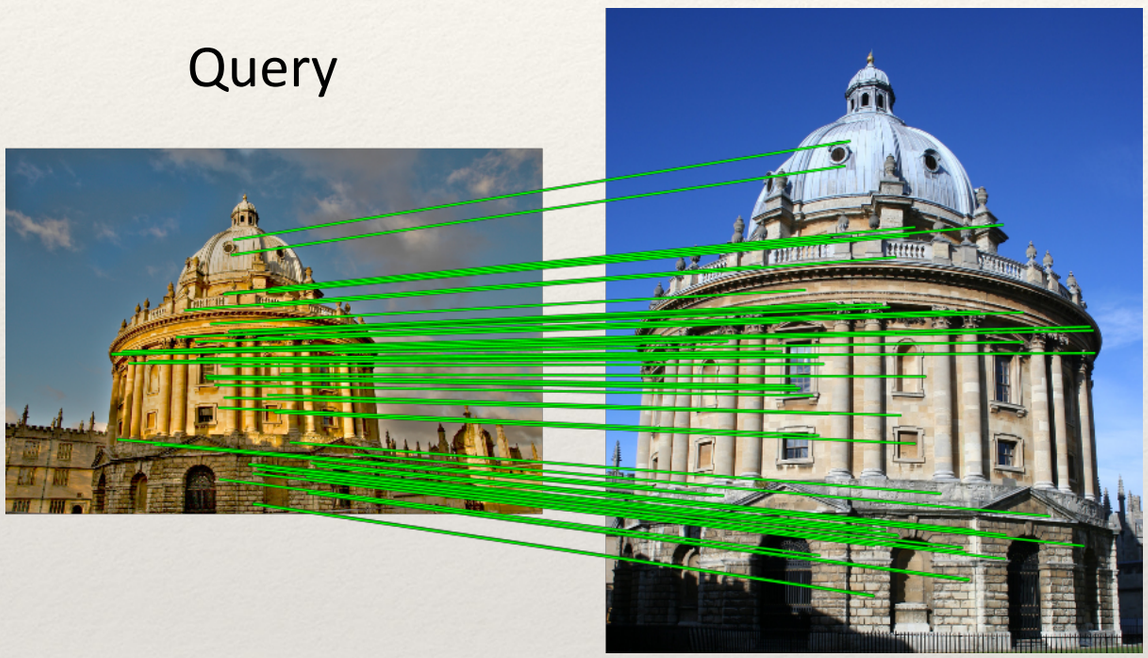
DB image with high BoW similarity

Both image pairs have many visual words in common.

Slide credit: Ondrej Chum

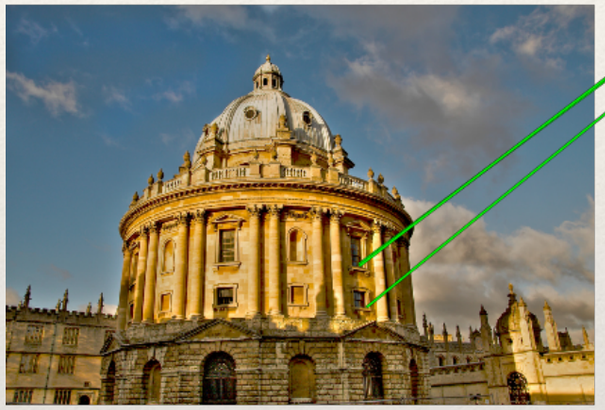
Spatial Verification

Query



DB image with high BoW similarity

Query



DB image with high BoW similarity

Only some of the matches are mutually consistent with real-world geometry imaged by a camera.

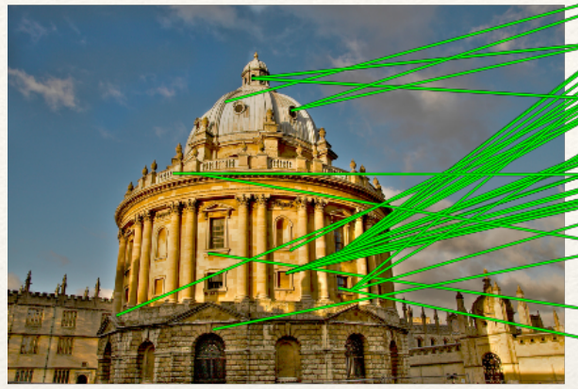
Ondrej Chum

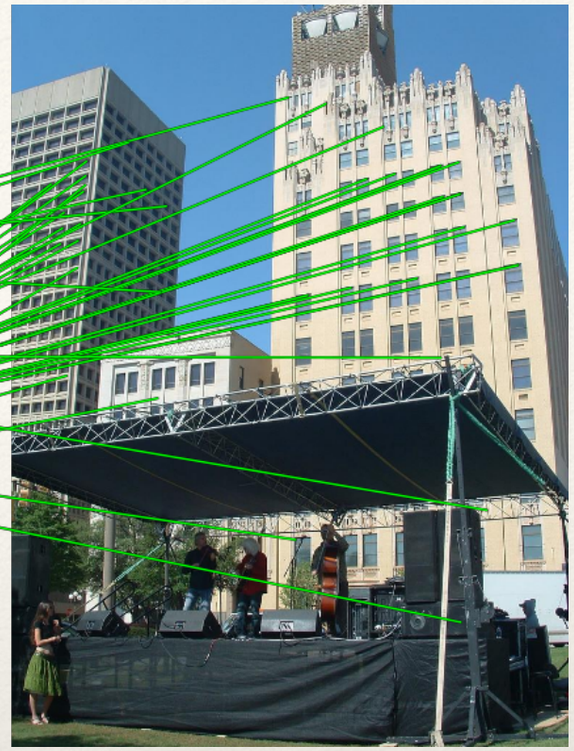
Spatial Verification: two basic strategies

- RANSAC
  - Typically sort by BoW similarity as initial filter
  - Verify by checking support (inliers) for possible transformations
    - e.g., “success” if find a transformation with > N inlier correspondences

- Generalized Hough Transform
  - Let each matched feature cast a vote on location, scale, orientation of the model object
  - Verify parameters with enough votes

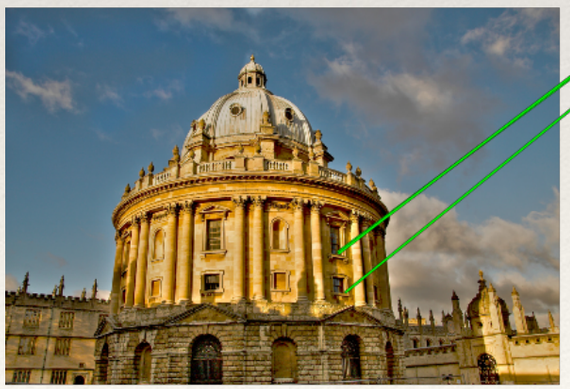
No verification





RANSAC verification

Fails to meet threshold on # inliers! Good!



Summary

- **Bag of words**: quantize feature space into discrete visual words
  - Summarize image by distribution of words
- **Inverted index**: visual word index for faster query time
- **Evaluation:**
- **Additional spatial verification alignment:**
  - Robust fitting : RANSAC, Generalized Hough Transform