

CSE 152: Computer Vision

Hao Su

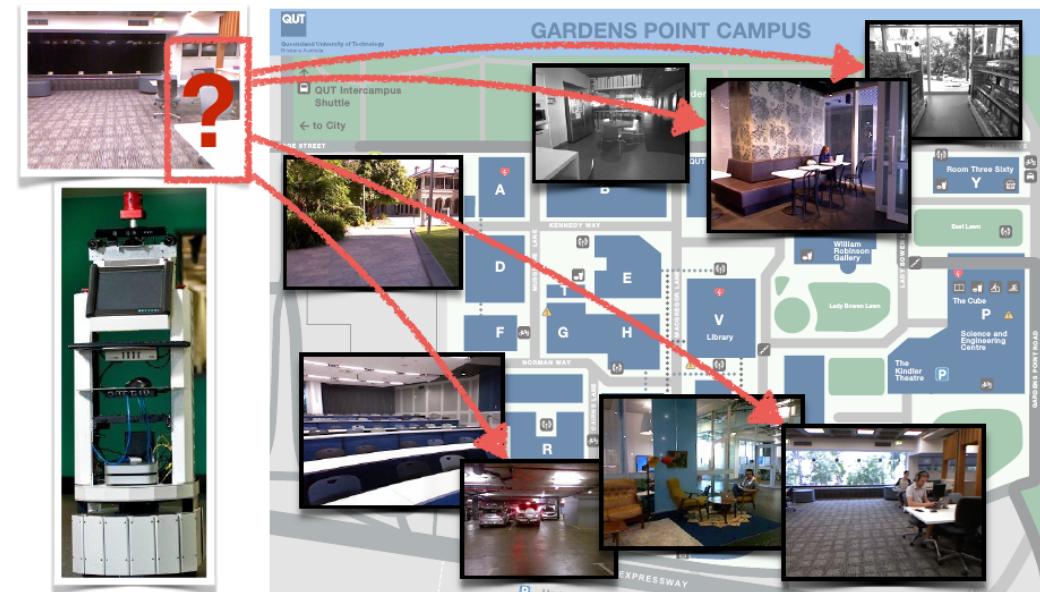
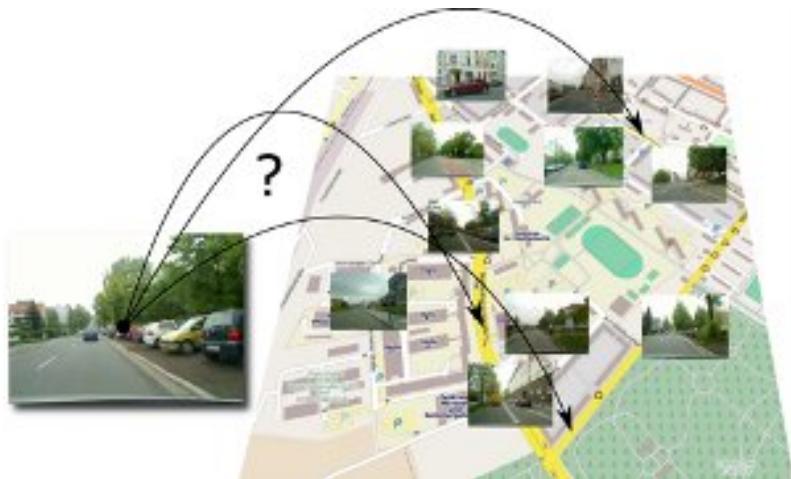
Lecture 2: Bag of Words



Credit: Manmohan Chandraker

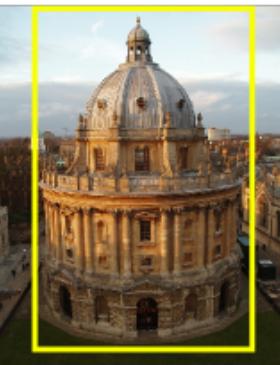
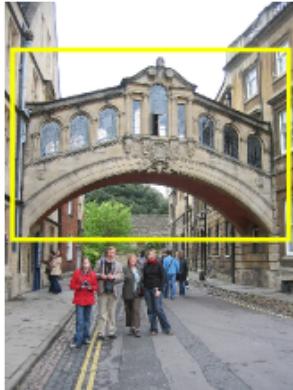
Recognizing or retrieving specific objects

Example 1: Place recognition



Recognizing or retrieving specific objects

Example 2: Search photos for particular places



Find these landmarks

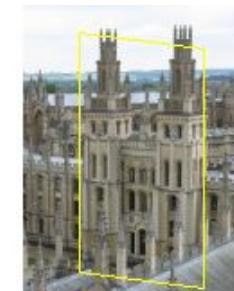
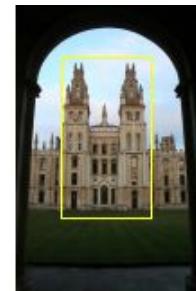
...in these images and 1M more

Why is it difficult?

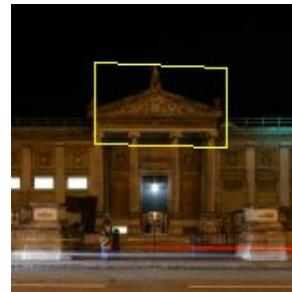
Want to find the object despite possibly large changes in scale, viewpoint, lighting and partial occlusion



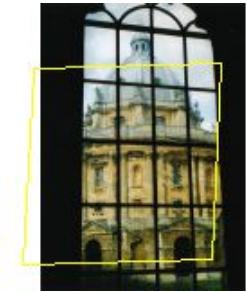
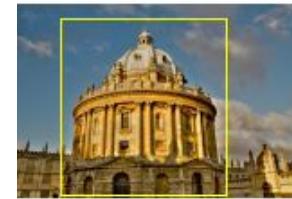
Scale



Viewpoint



Lighting



Occlusion

Object

Bag of ‘words’



Origin: Bag-of-words models

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



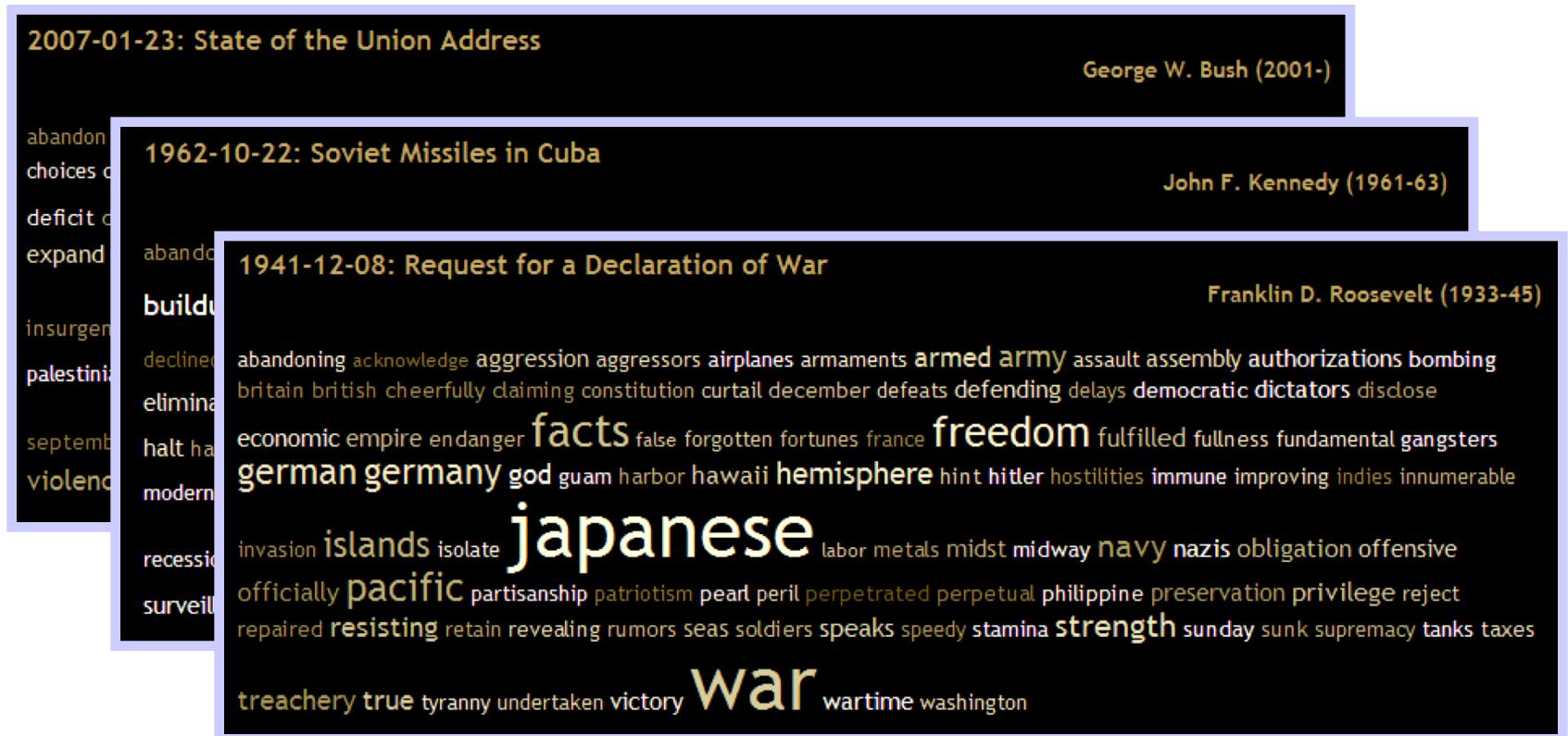
Origin: Bag-of-words models

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

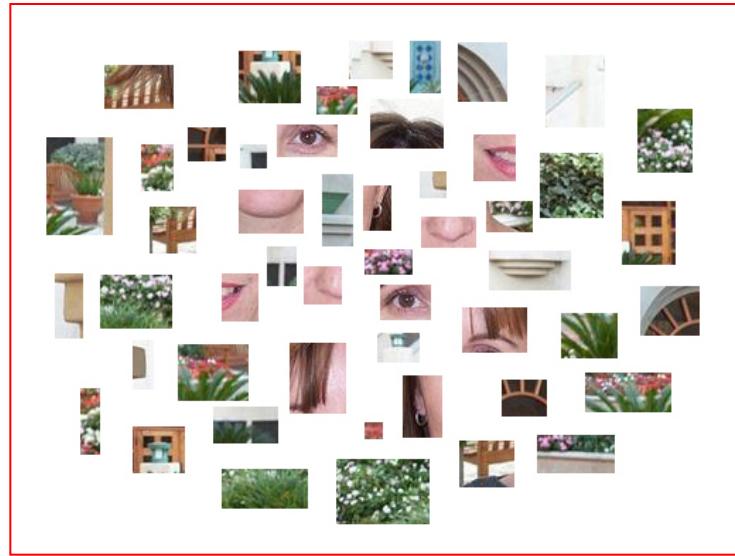
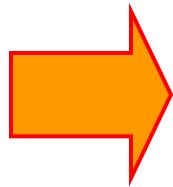


Origin: Bag-of-words models

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



Origin: Bag-of-words models



face, flowers, building

- Works quite well for image-level classification and for recognizing object *instances*

Origin: Bag-of-words models

Caltech6 dataset



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

Recall: matching local features



Image 1

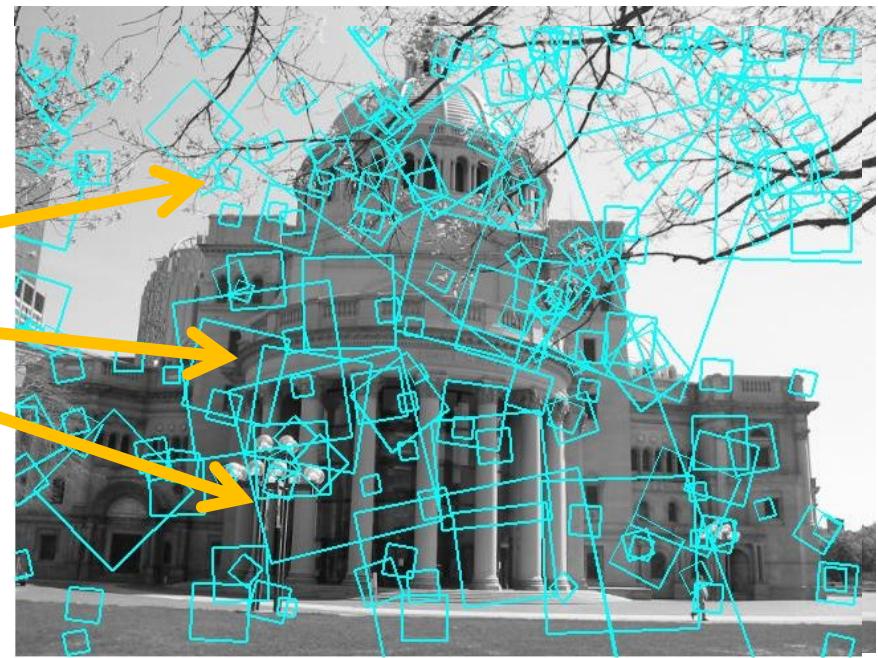


Image 2

To generate **candidate matches**, find patches that have the most similar appearance (e.g., lowest SSD)

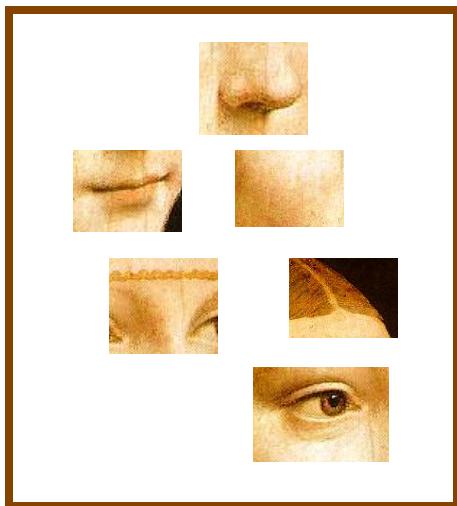
Simplest approach: compare them all, take the closest (or closest k , or within a thresholded distance)

Indexing local features



Bag of features: outline

1. Extract features



Bag of features: outline

1. Extract features
2. Learn “visual vocabulary”

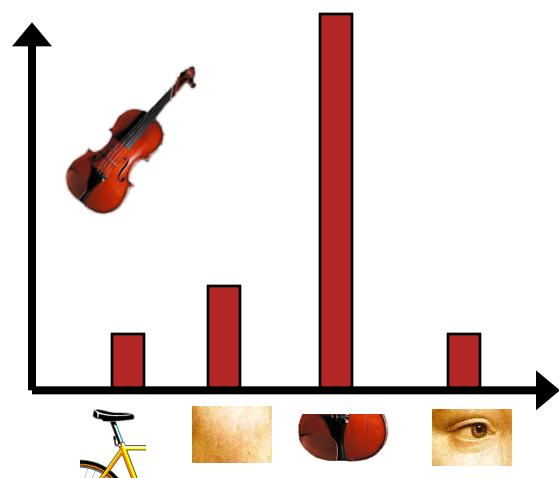
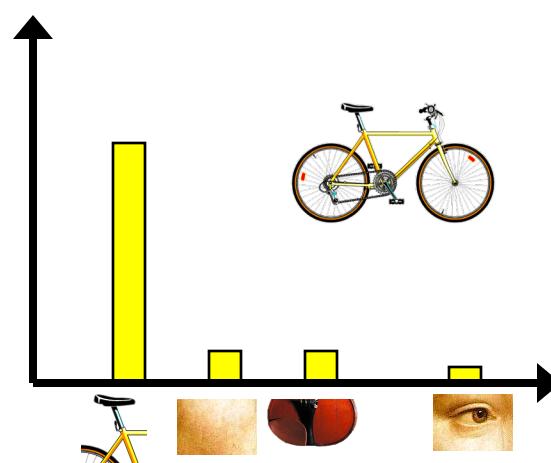
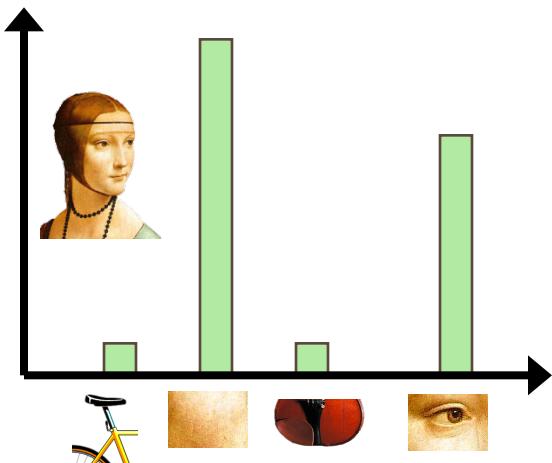


Bag of features: outline

1. Extract features
2. Learn “visual vocabulary”
3. Quantize features using visual vocabulary

Bag of features: outline

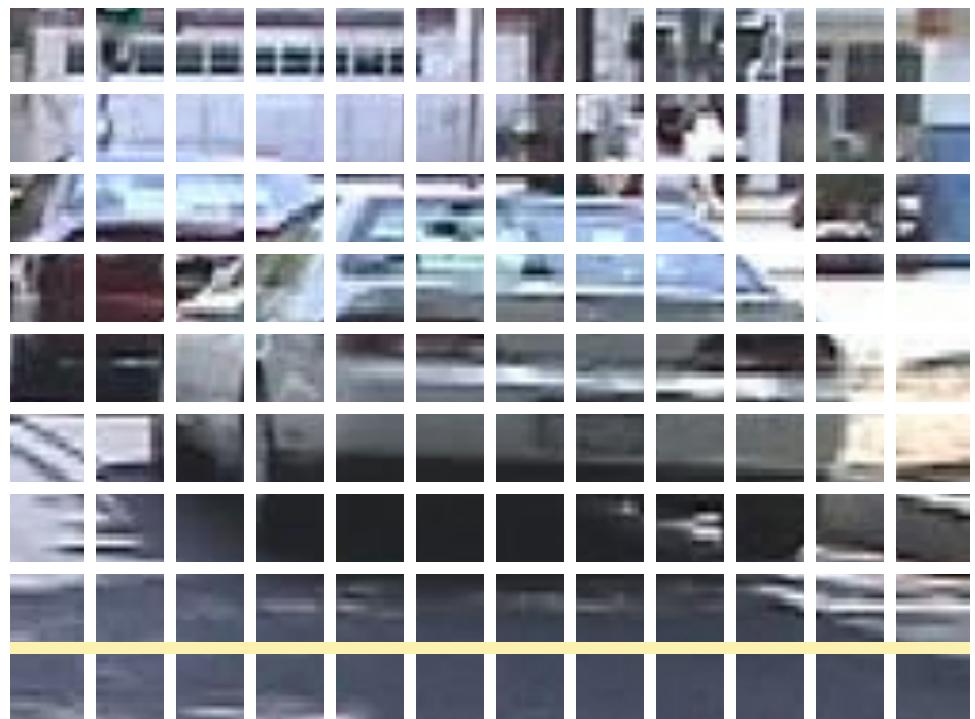
1. Extract features
2. Learn “visual vocabulary”
3. Quantize features using visual vocabulary
4. Represent images by frequencies of “visual words”



1. Feature extraction

Regular grid

- Vogel & Schiele, 2003
- Fei-Fei & Perona, 2005



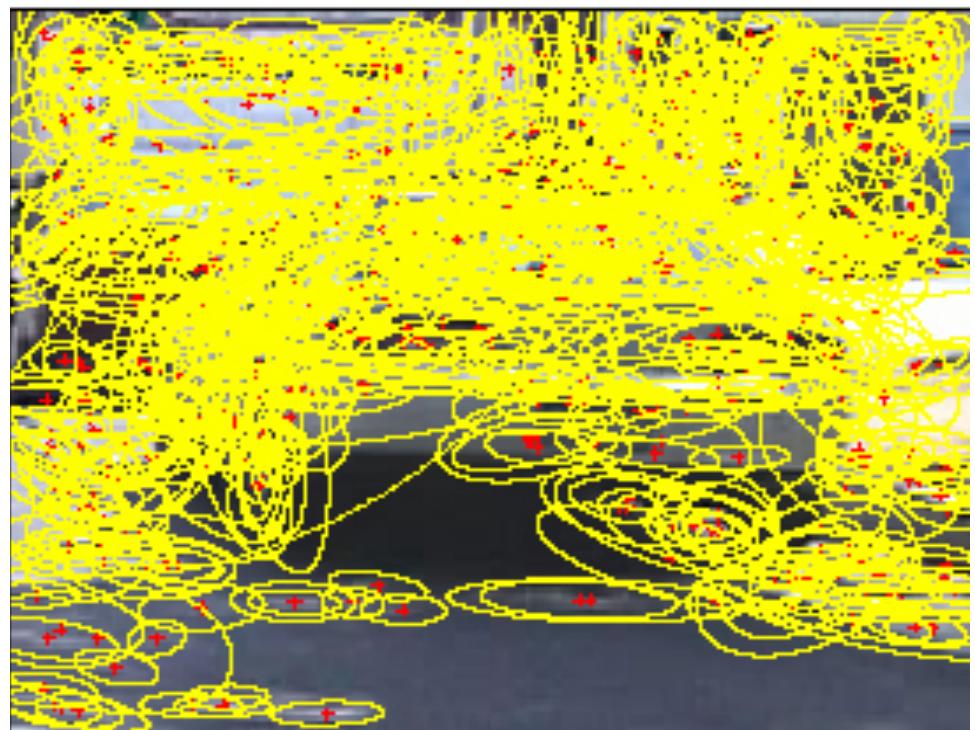
1. Feature extraction

Regular grid

- Vogel & Schiele, 2003
- Fei-Fei & Perona, 2005

Interest point detector

- Csurka et al. 2004
- Fei-Fei & Perona, 2005
- Sivic et al. 2005



1. Feature extraction

Regular grid

- Vogel & Schiele, 2003
- Fei-Fei & Perona, 2005

Interest point detector

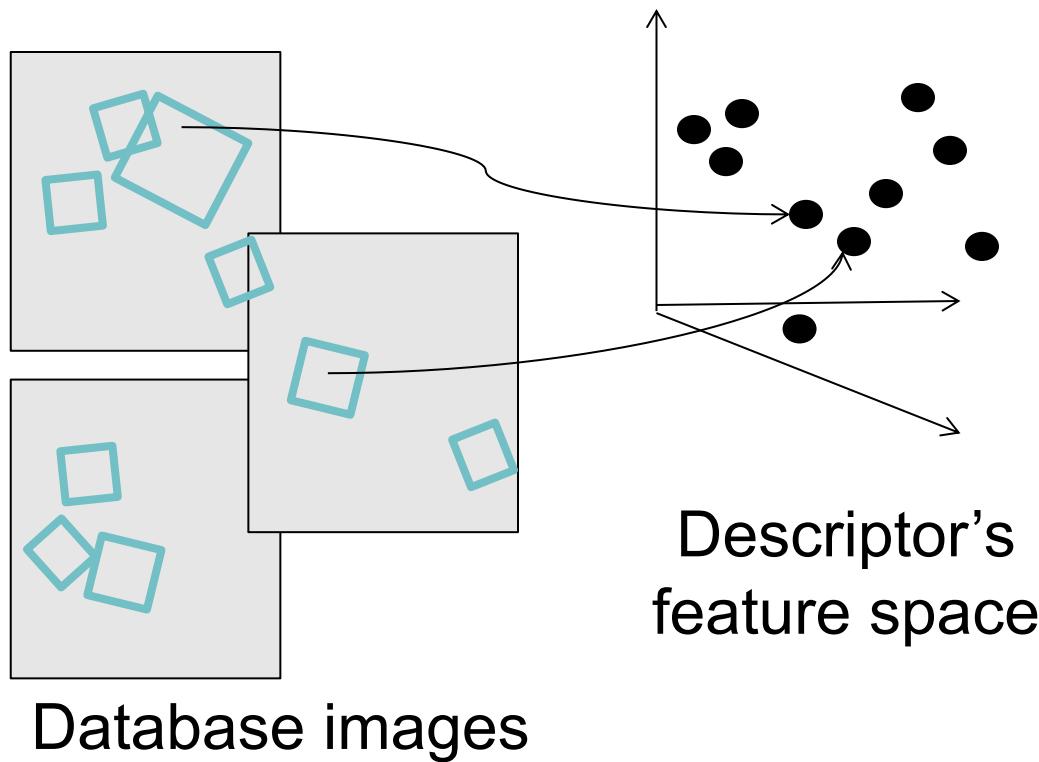
- Csurka et al. 2004
- Fei-Fei & Perona, 2005
- Sivic et al. 2005

Other methods

- Random sampling (Vidal-Naquet & Ullman, 2002)
- Segmentation-based patches (Barnard et al. 2003)

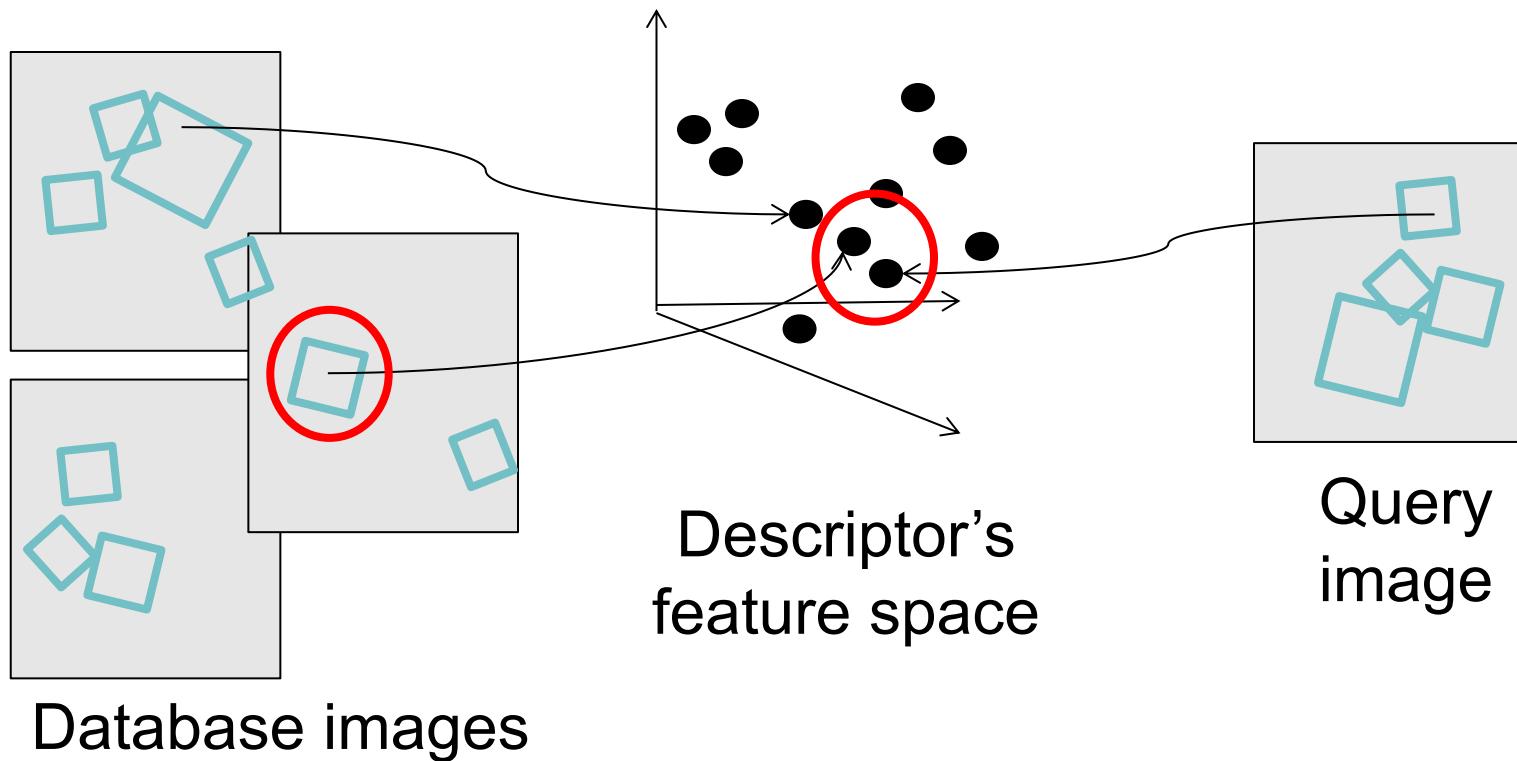
Indexing local features

- Each patch or region has a descriptor, which is a point in some high-dimensional feature space (for example, SIFT)



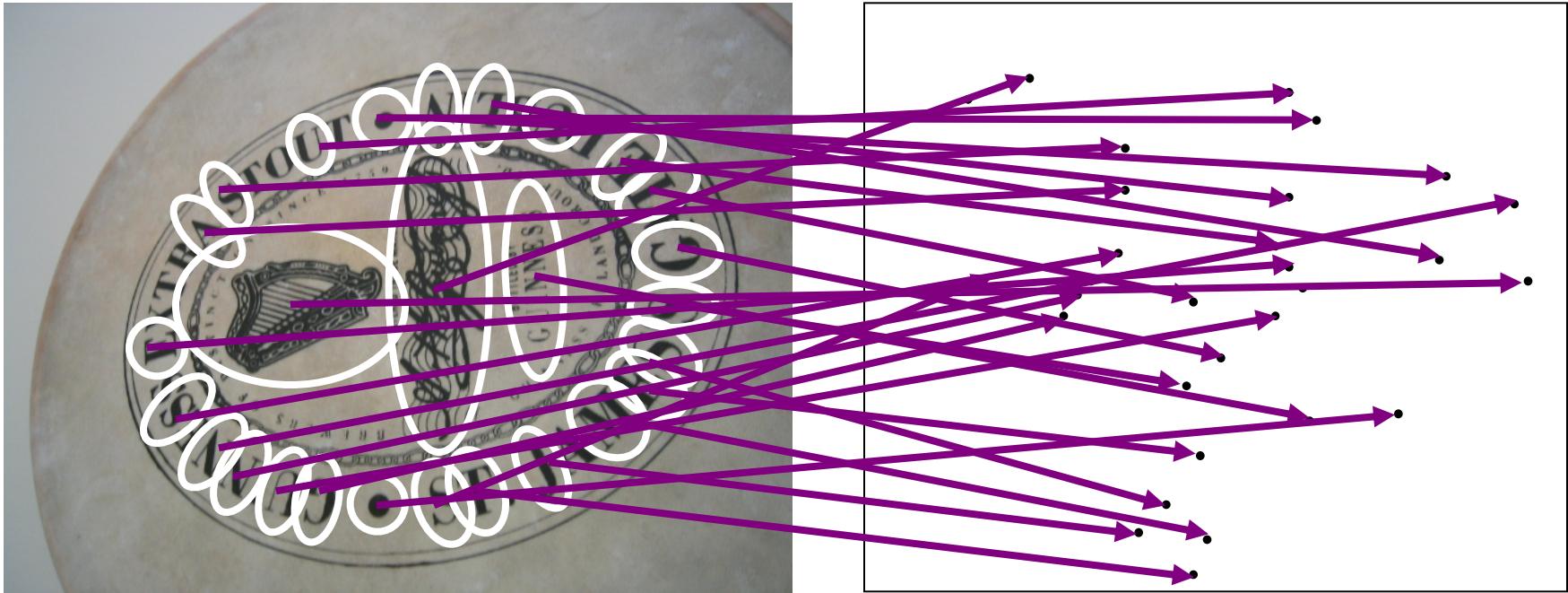
Indexing local features

- When we see close points in feature space, we have similar descriptors, which indicates similar local content.



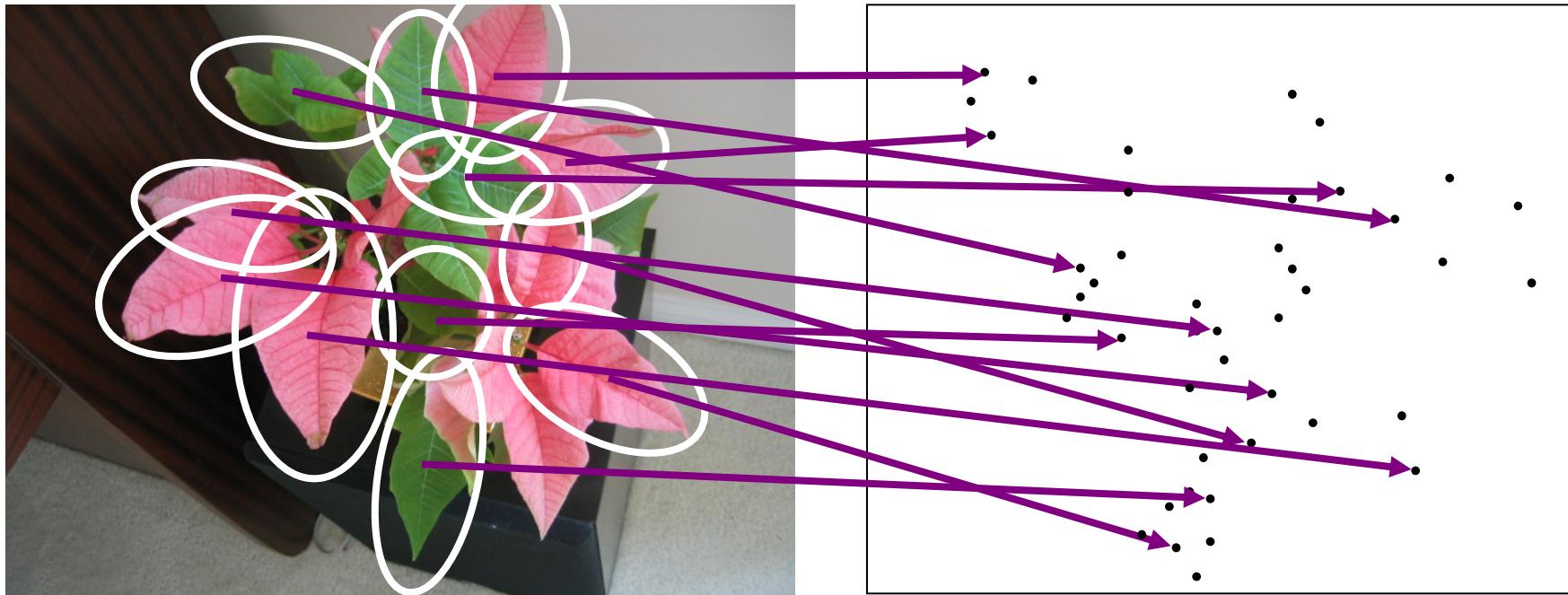
Visual words: main idea

- Extract some local features from a number of images ...

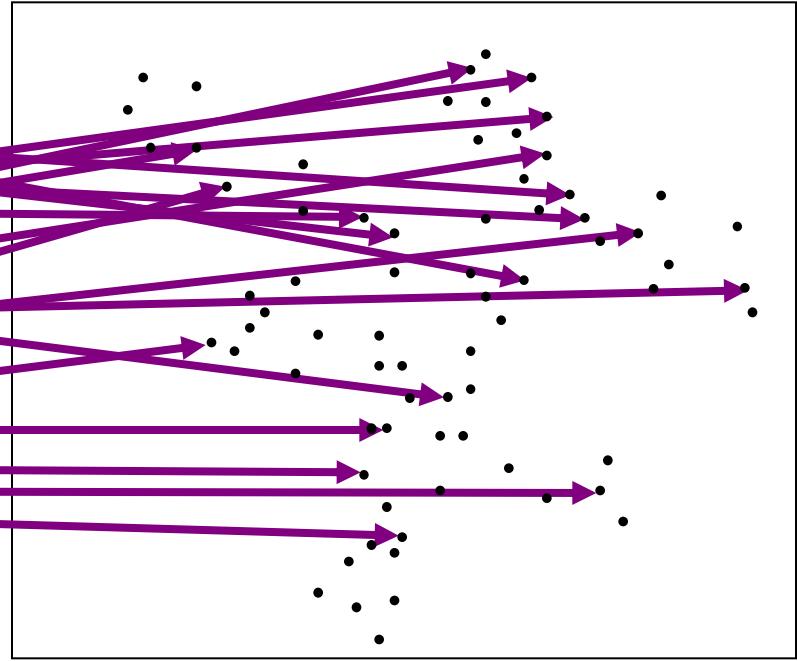
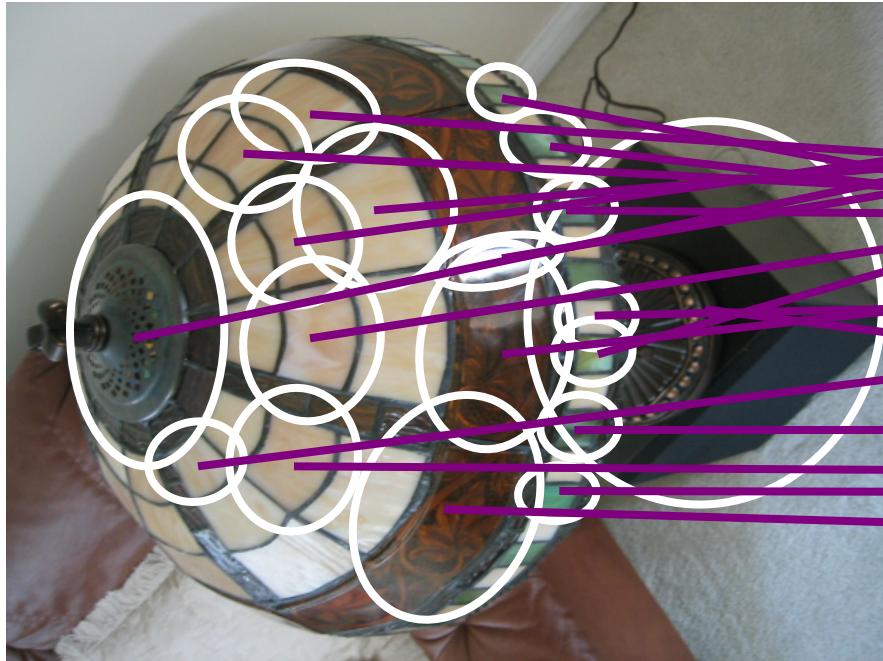


Example: For SIFT descriptor space,
each point is 128-dimensional

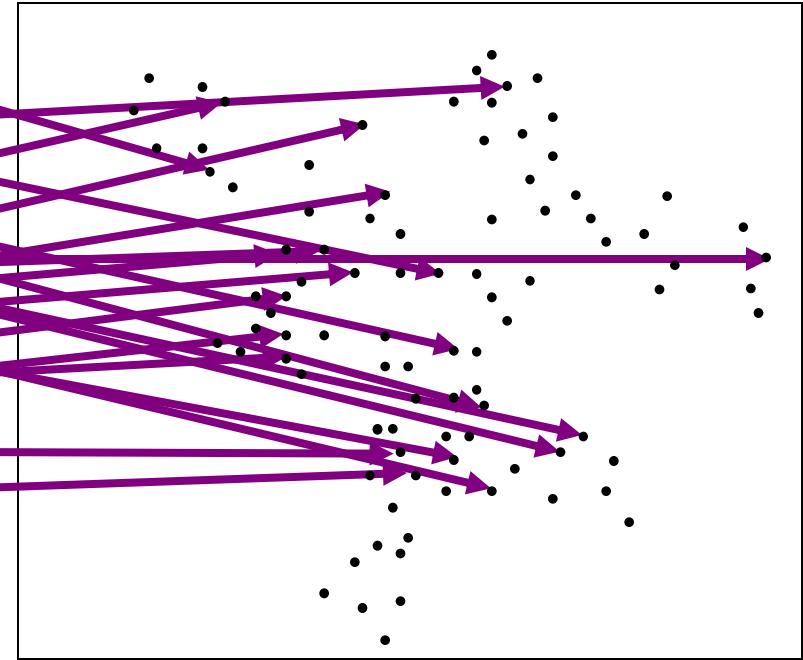
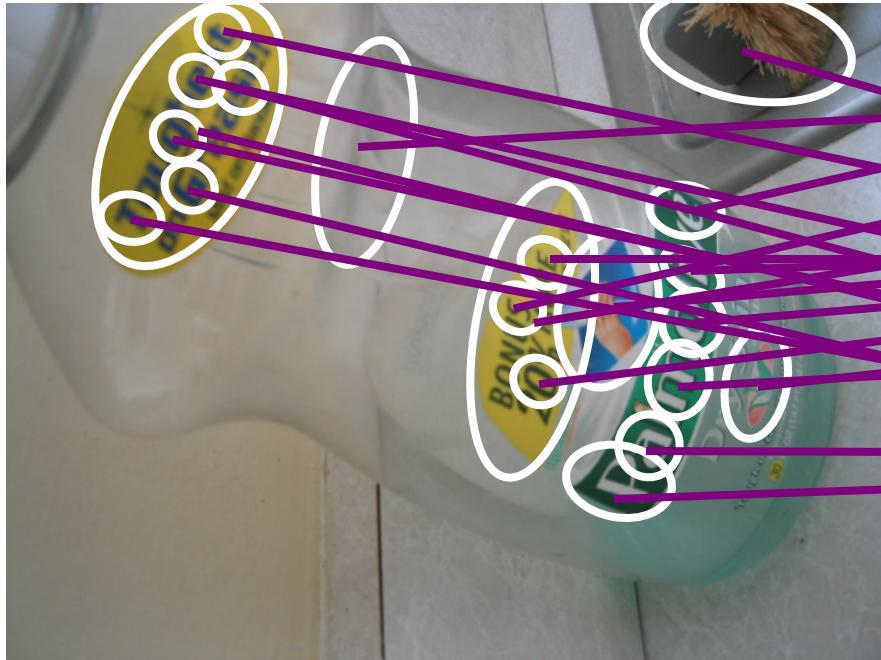
Visual words: main idea



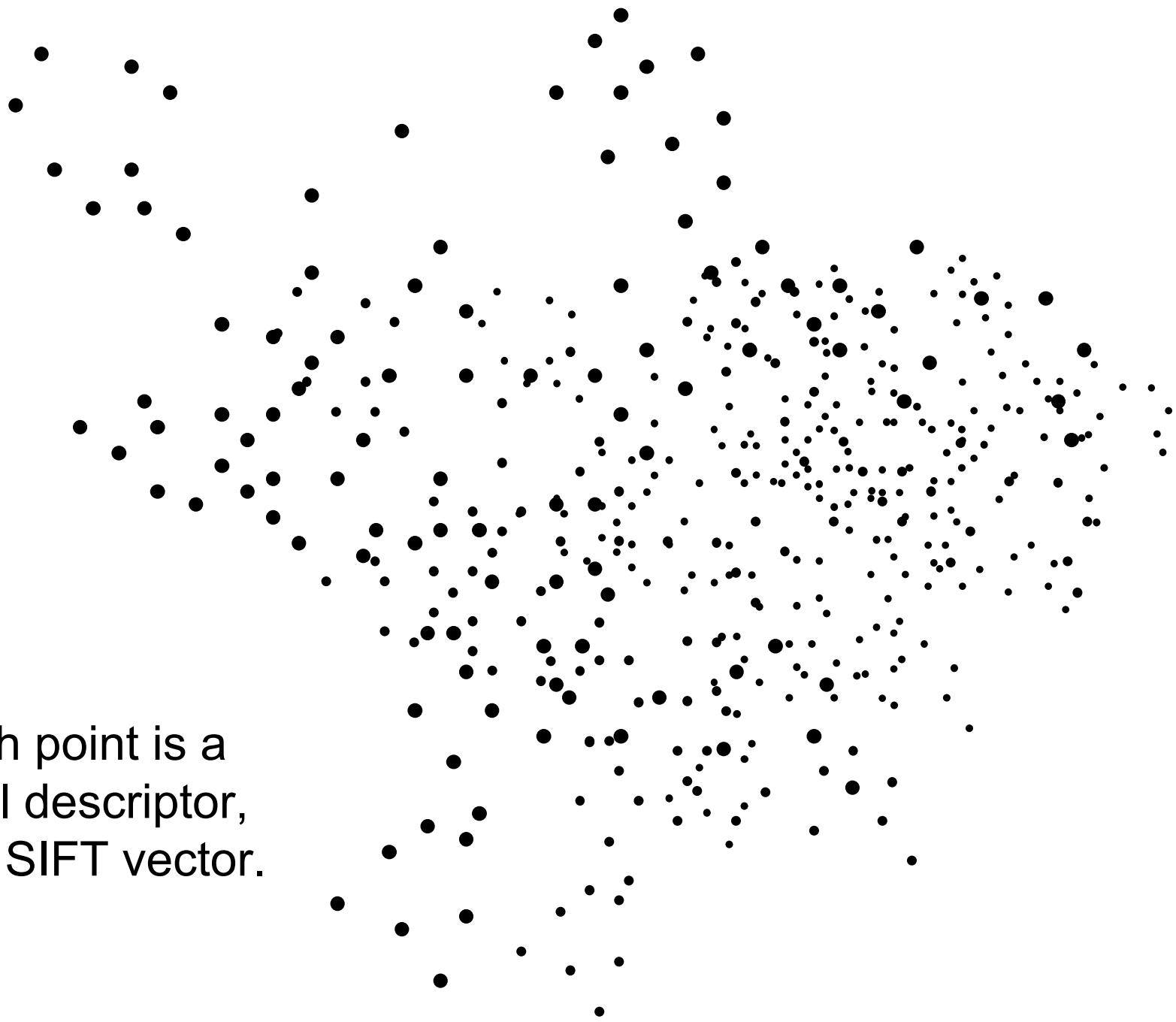
Visual words: main idea



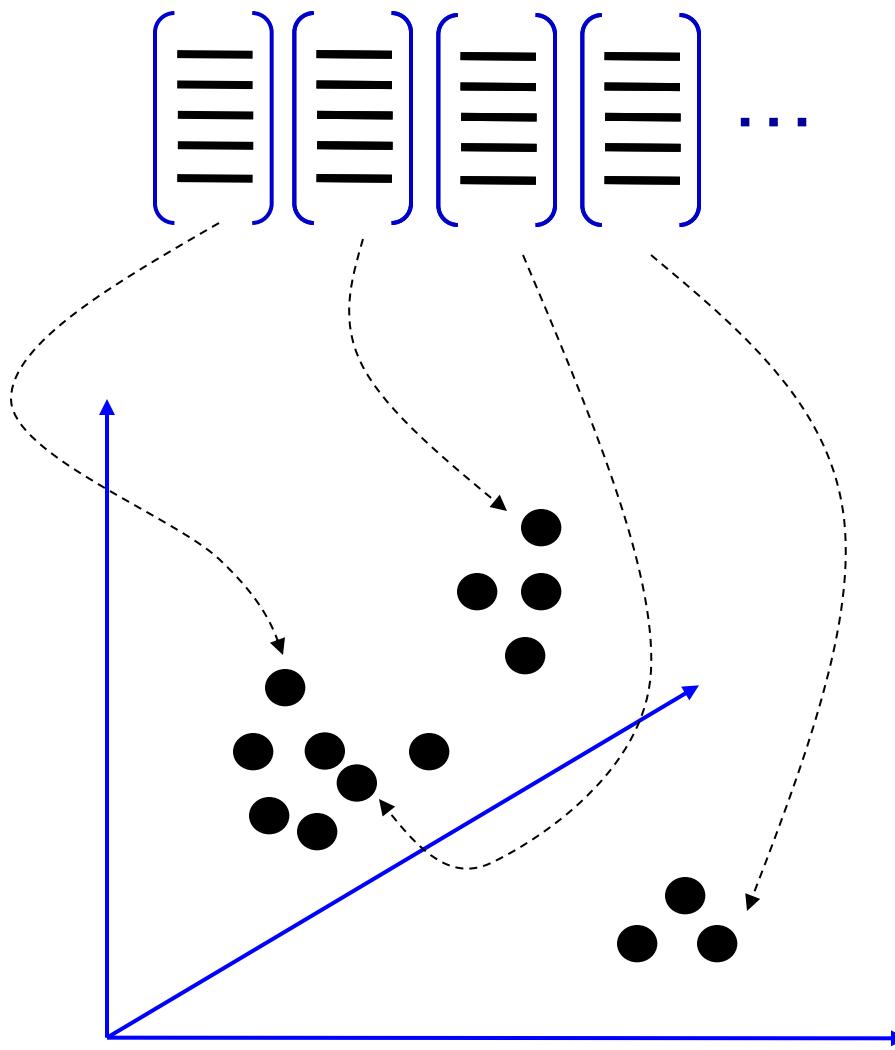
Visual words: main idea



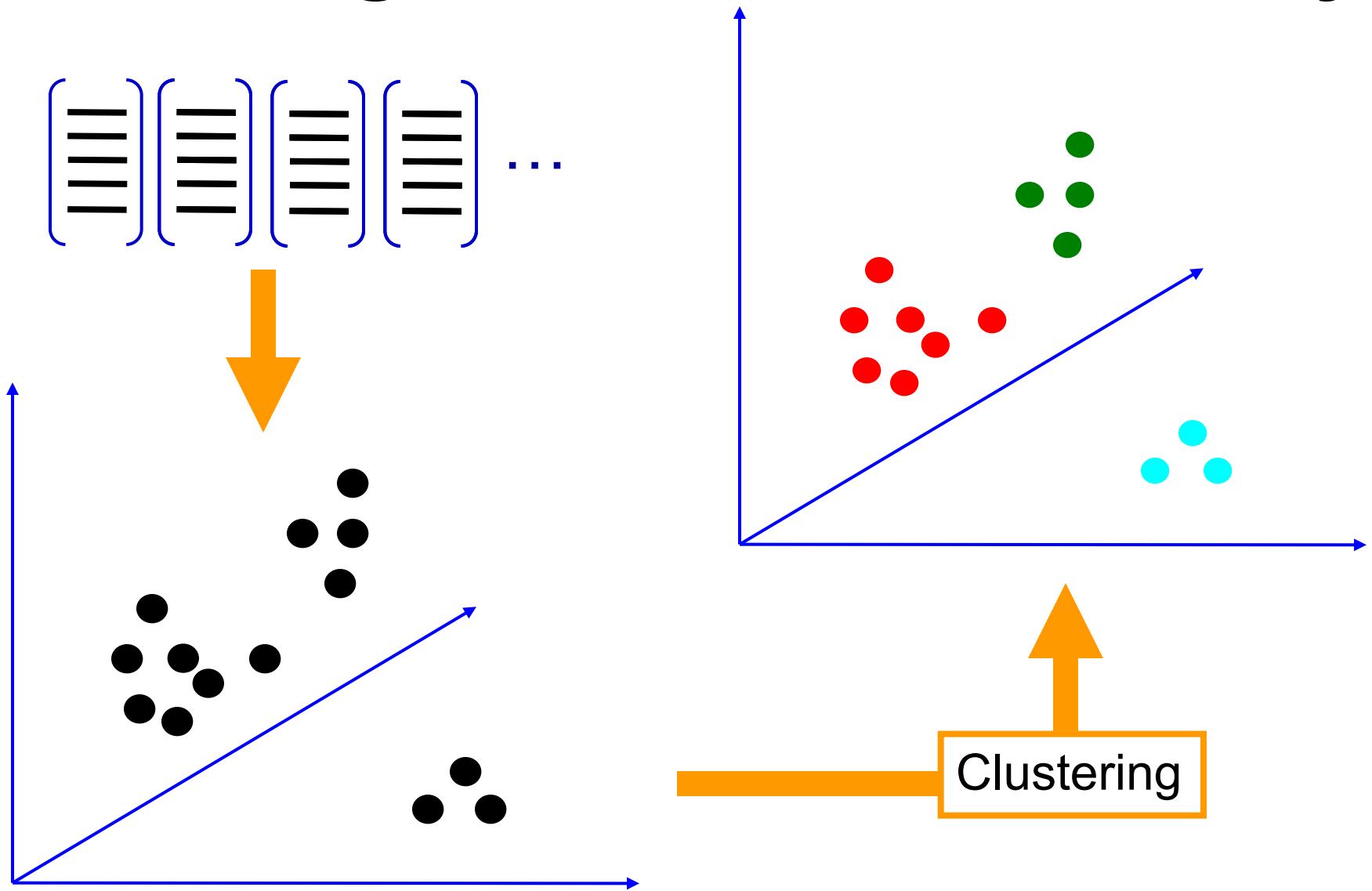
Each point is a
local descriptor,
e.g. SIFT vector.



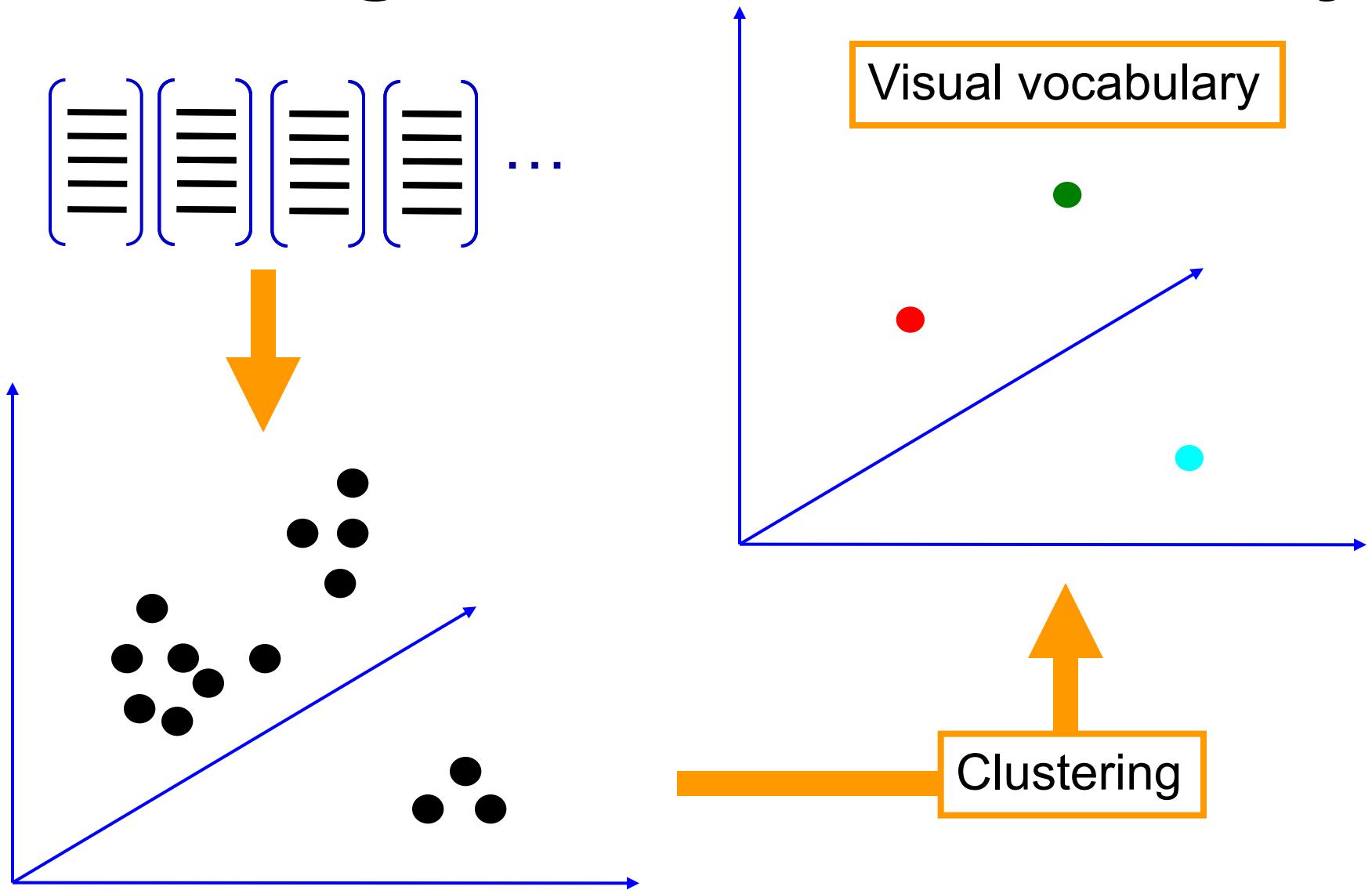
2. Learning the visual vocabulary



2. Learning the visual vocabulary



2. Learning the visual vocabulary



K-means clustering

- Want to minimize sum of squared Euclidean distances between points x_i and their nearest cluster centers \bar{c}_k

$$D(X, C) = \sum_{\text{cluster } C_k} \sum_{\text{point } i \in C_k} (x_i - \bar{c}_k)^2$$

Algorithm:

- Randomly initialize K cluster centers
- Iterate until convergence:
 - Assign each data point to the nearest center
 - Recompute each cluster center as the mean of all points assigned to it

From clustering to 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

- Example: each group of patches belongs to the same visual word

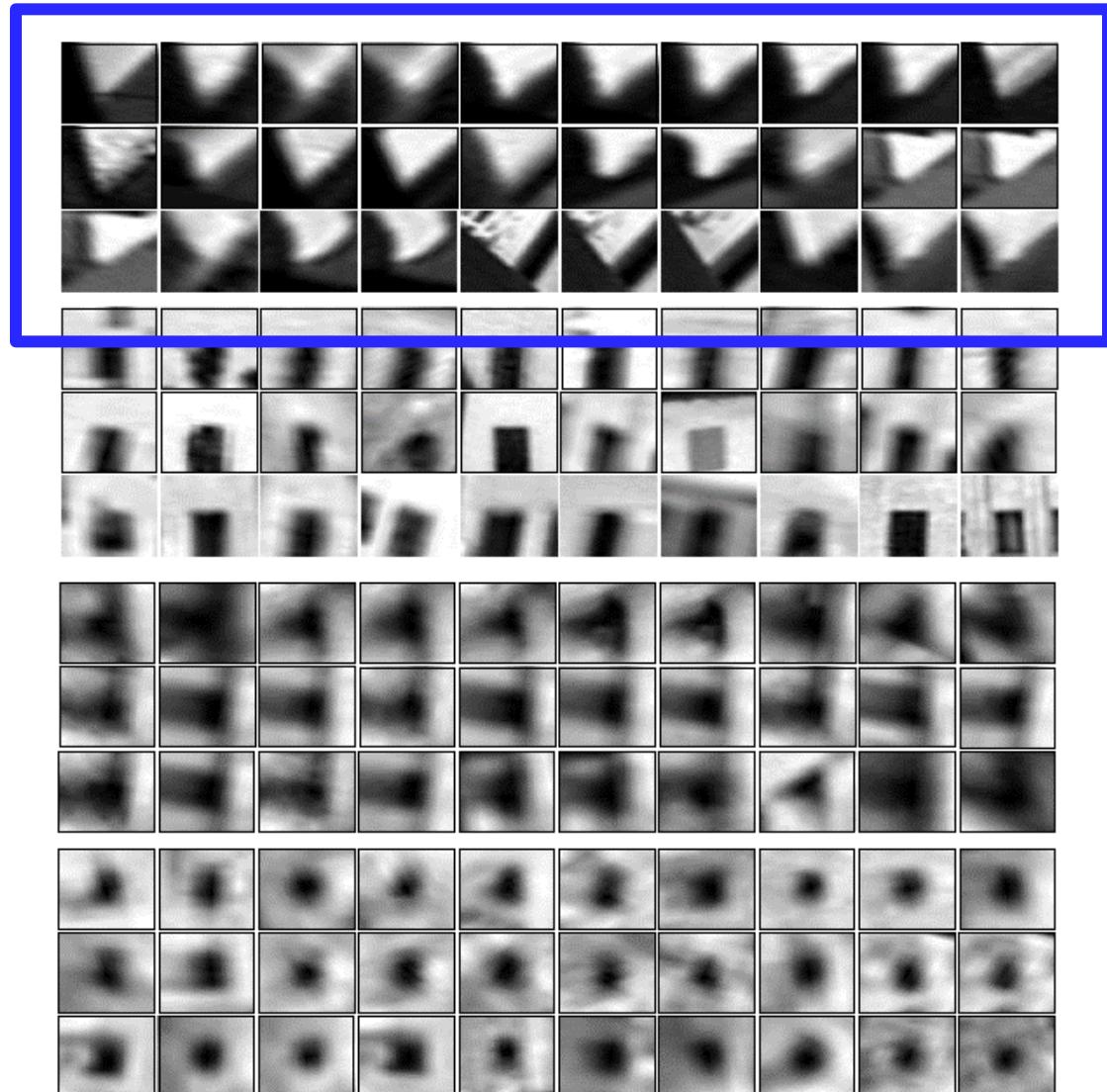
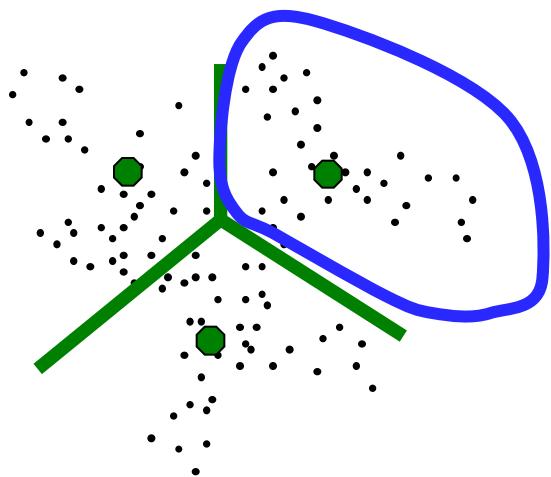
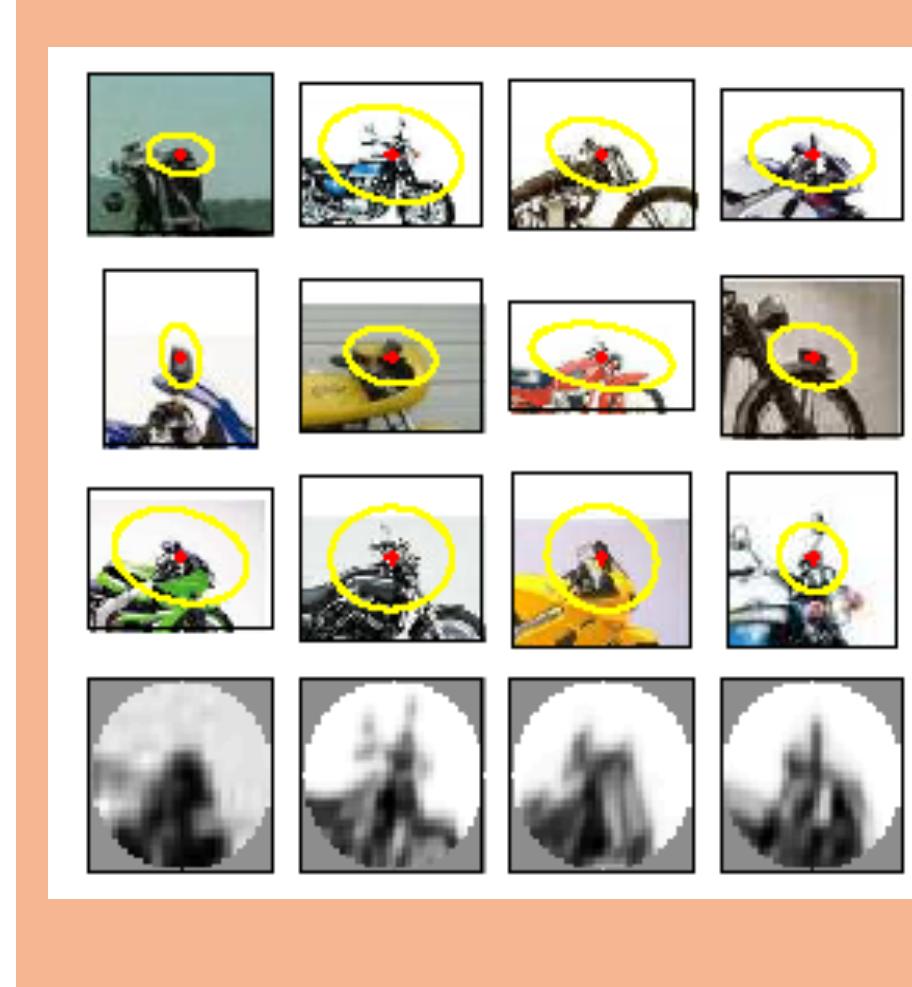
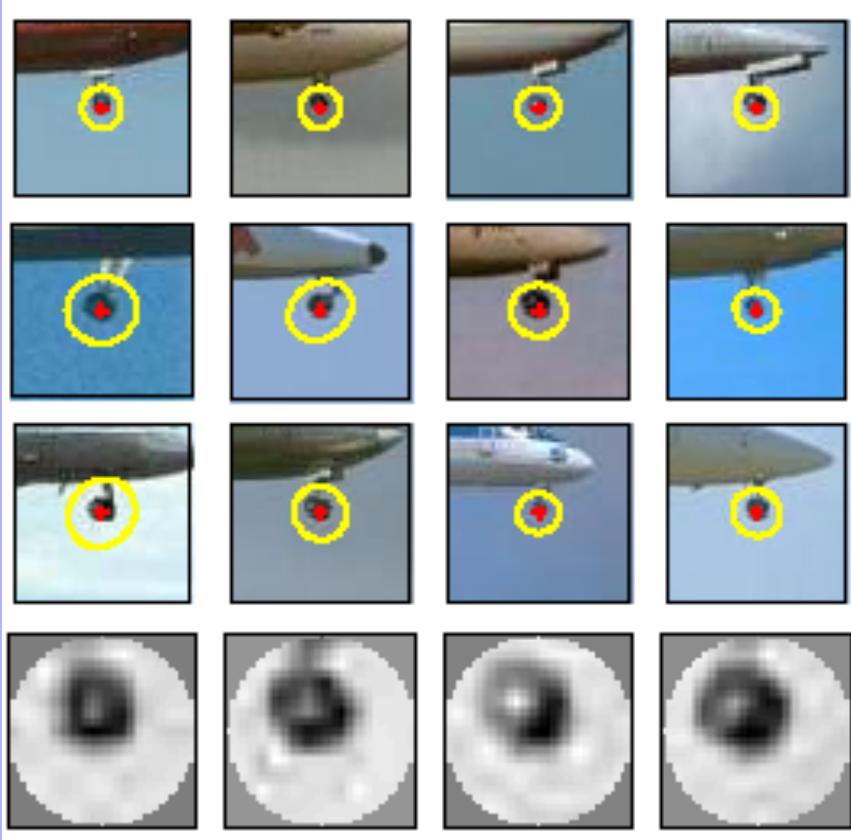
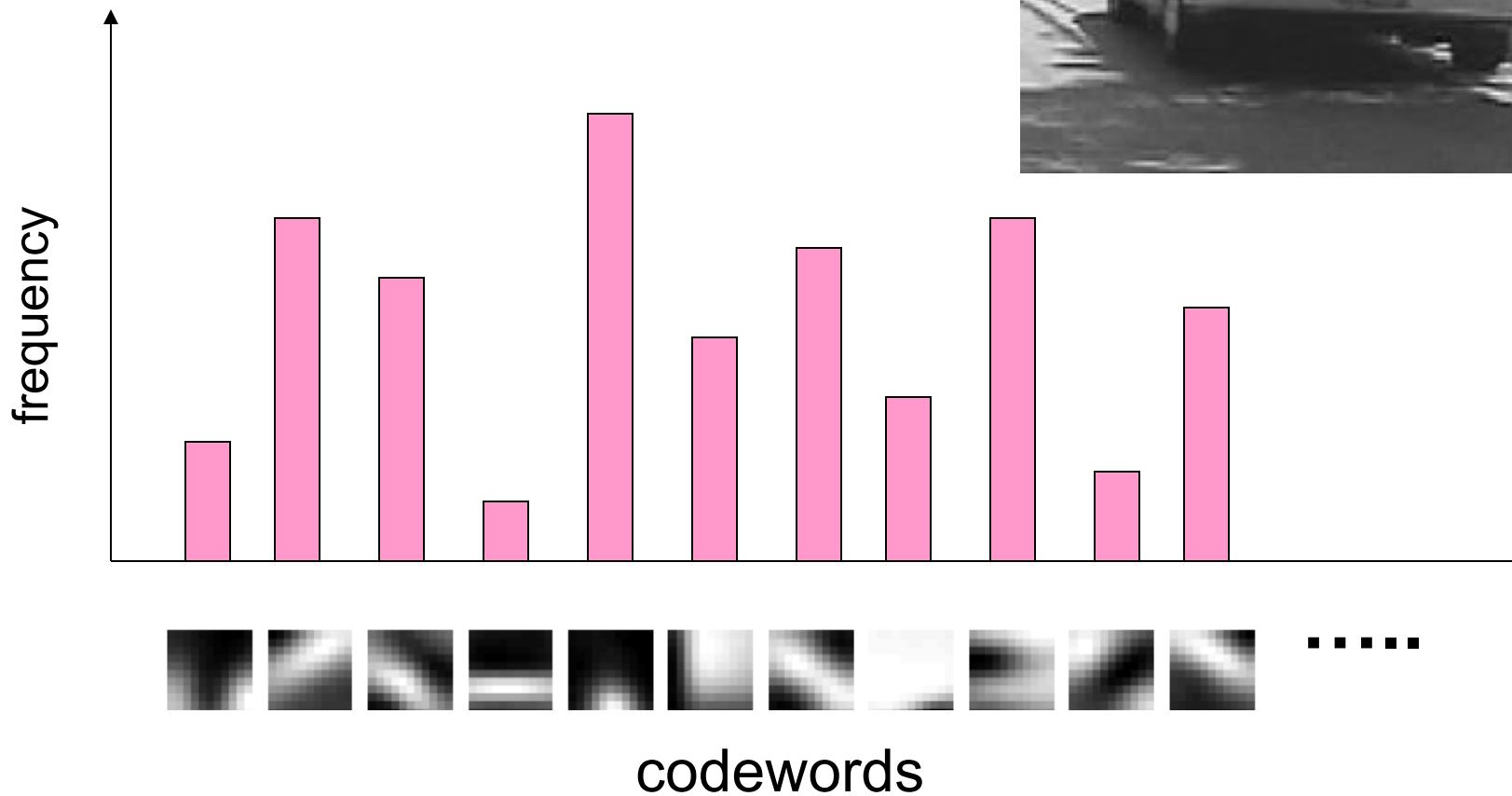


Figure from Sivic & Zisserman, ICCV 2003

Visual words

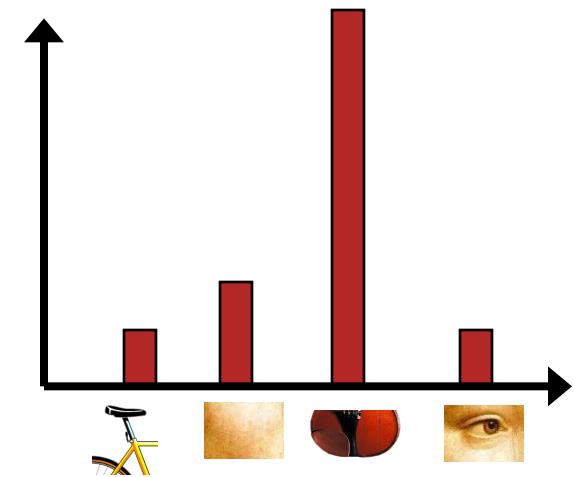
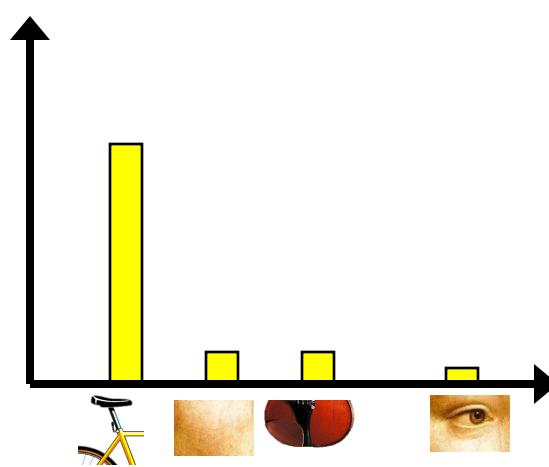
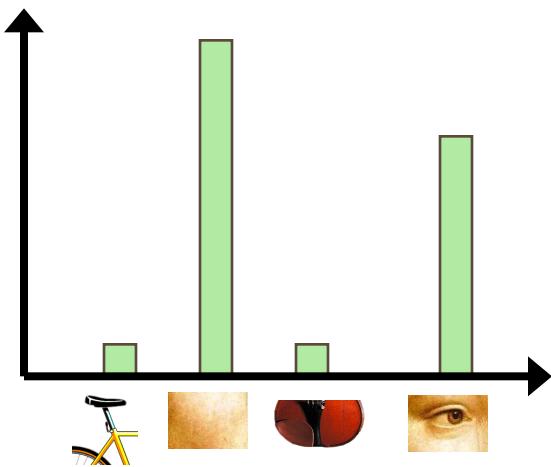
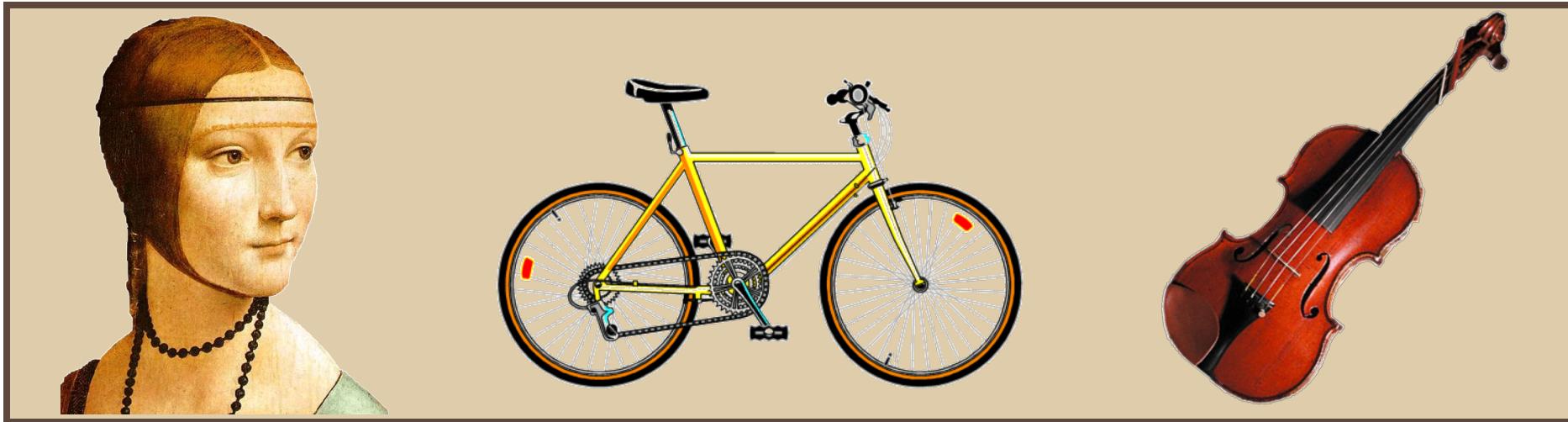


3. Image representation



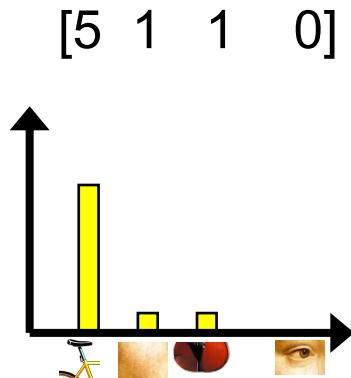
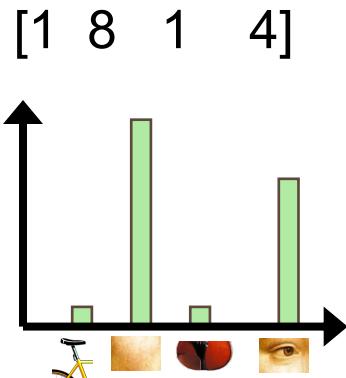
4. Image classification

- Given the bag-of-features representations of images from different classes, how do we learn a model for distinguishing them?



Comparing bags of words

- Rank frames by normalized scalar product between their (possibly weighted) occurrence counts---*nearest neighbor* search for similar images.



$$\vec{d}_j$$

$$sim(d_j, q) = \frac{\langle d_j, q \rangle}{\|d_j\| \|q\|}$$
$$= \frac{\sum_{i=1}^V d_j(i) * q(i)}{\sqrt{\sum_{i=1}^V d_j(i)^2} * \sqrt{\sum_{i=1}^V q(i)^2}}$$

for vocabulary of V words

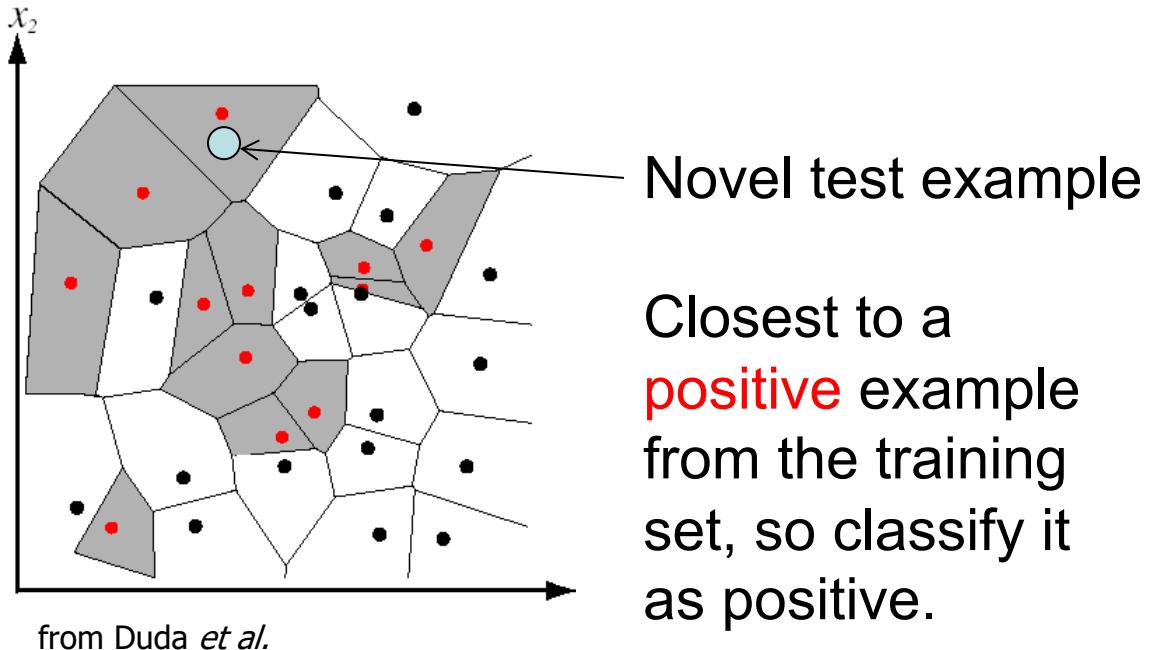
Nearest Neighbors

Nearest Neighbor classification

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

Black = negative

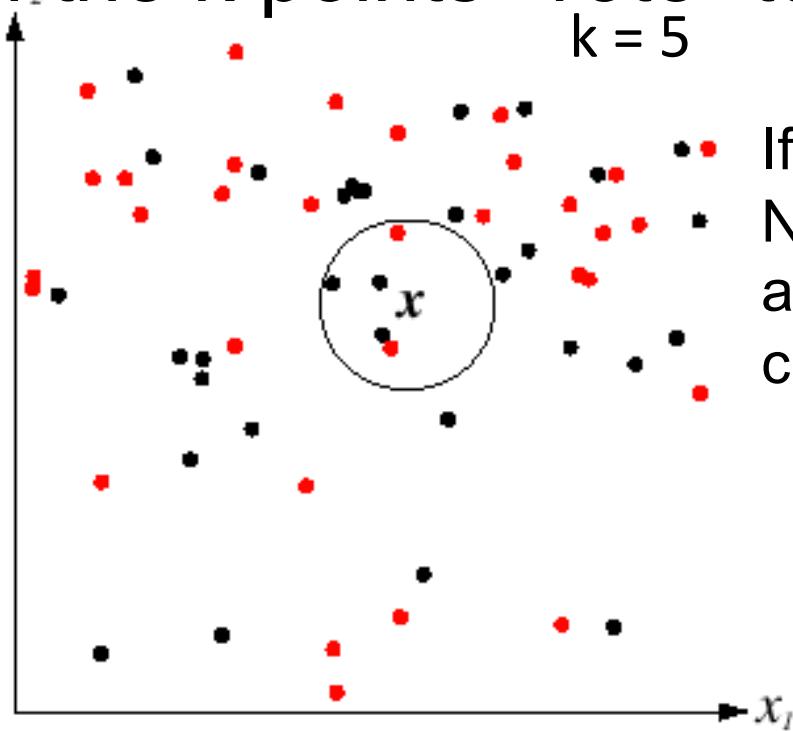
Red = positive



K-Nearest Neighbors classification

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

Black = negative
Red = positive

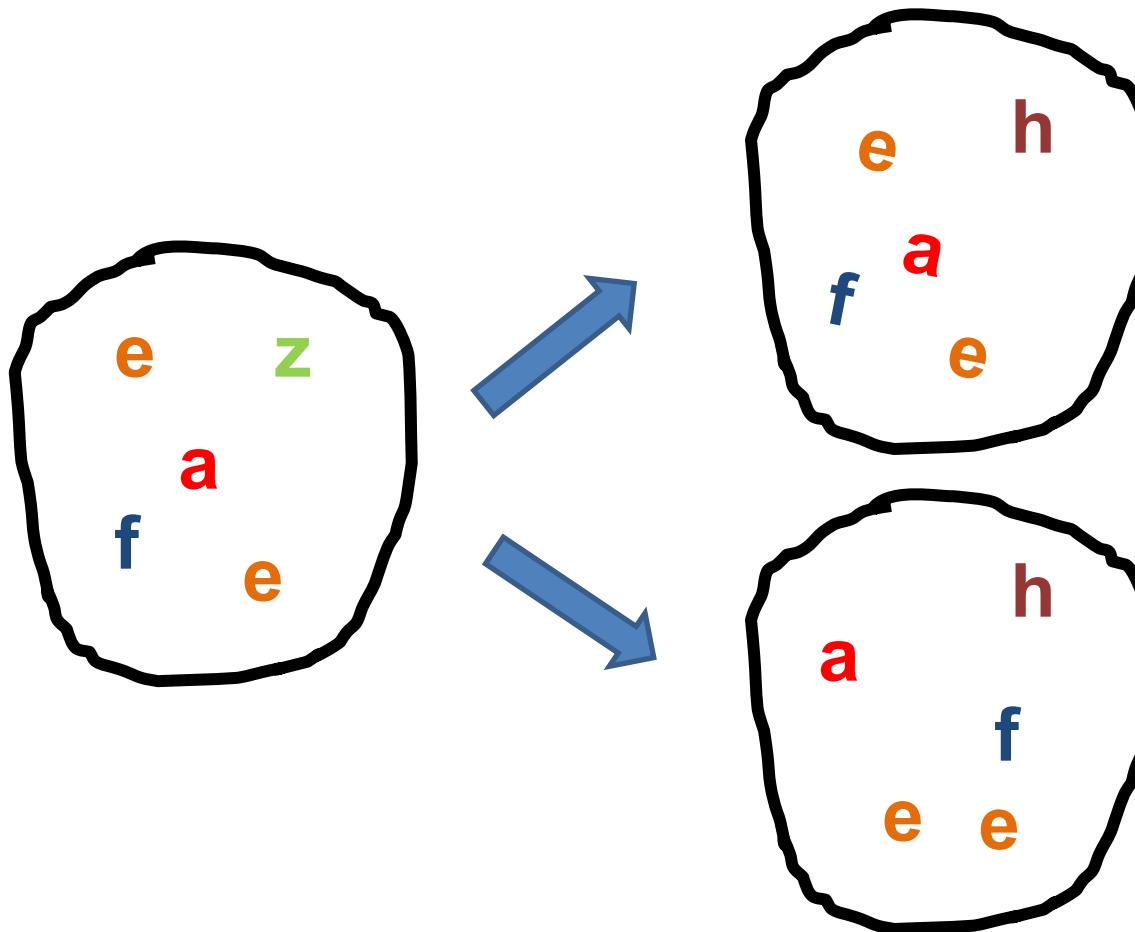


If query lands here, the 5 NN consist of 3 negatives and 2 positives, so we classify it as negative.

Nearest neighbors

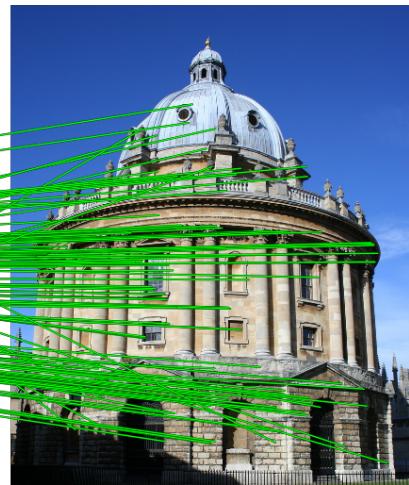
- **Advantages:**
 - Simple to implement
 - Flexible to feature or distance choices
 - Can do well in practice with enough representative data
- **Limitations:**
 - Large search problem to find nearest neighbors
 - Storage of data
 - Must know we have a meaningful distance function

Which matches better?



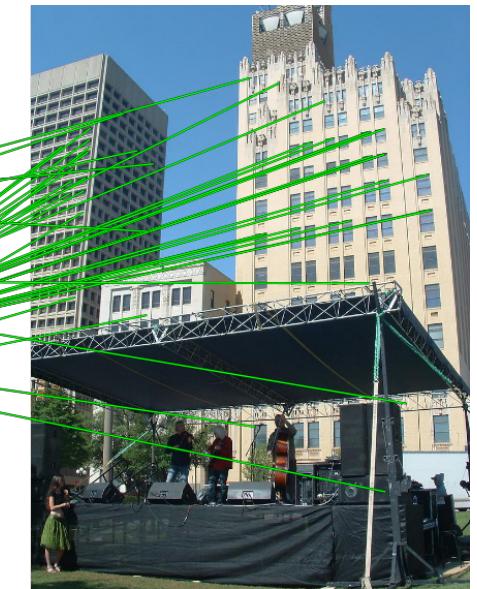
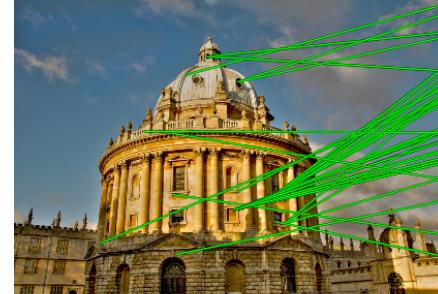
Spatial Verification

Query



DB image with high BoW
similarity

Query

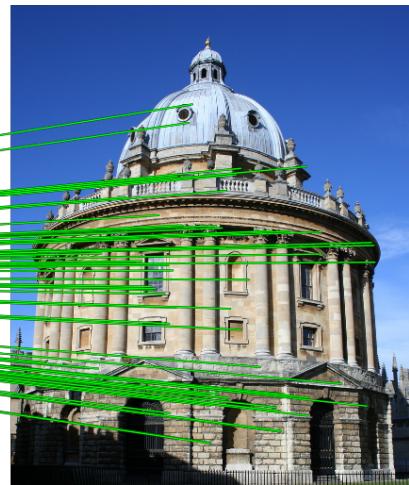
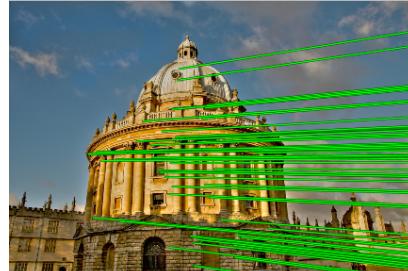


DB image with high BoW
similarity

Both image pairs have many visual words in common.

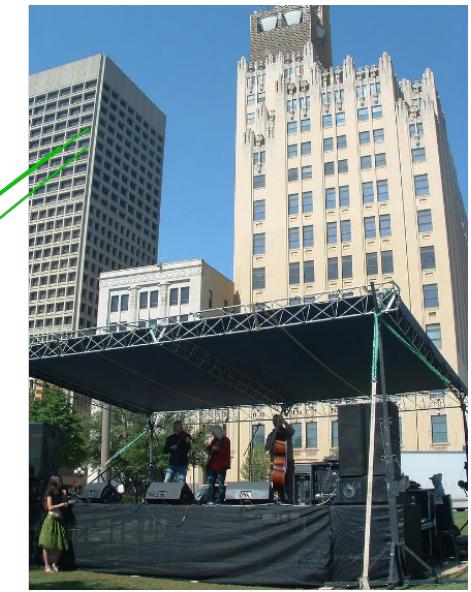
Spatial Verification

Query



DB image with high BoW
similarity

Query



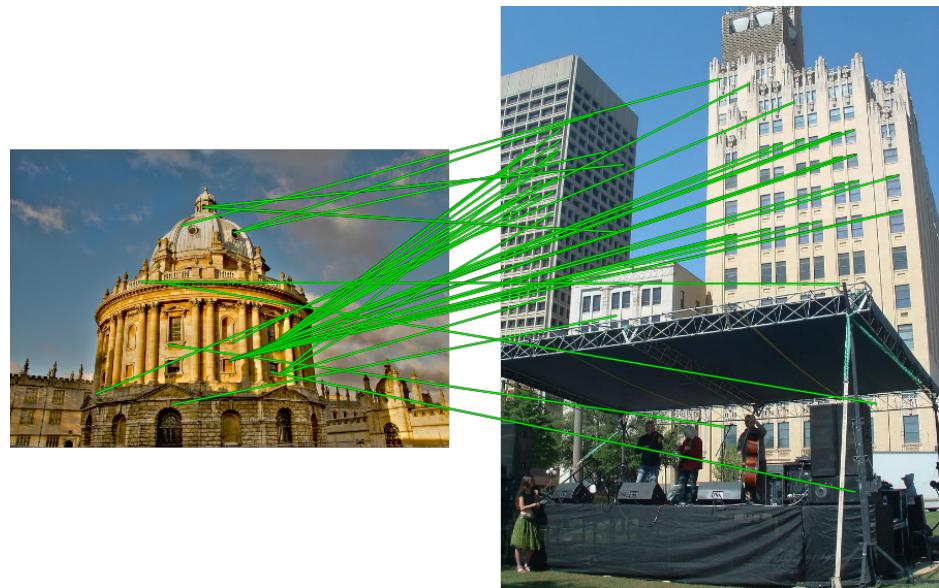
DB image with high BoW
similarity

Only some of the matches are mutually consistent

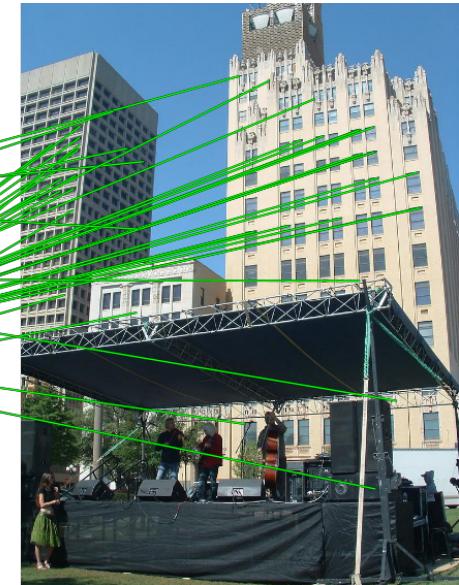
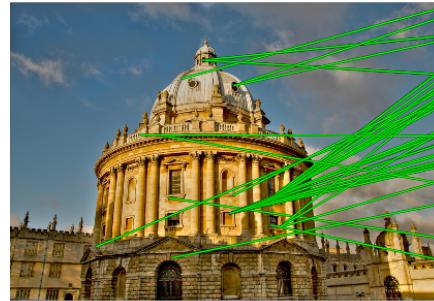
Spatial Verification

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

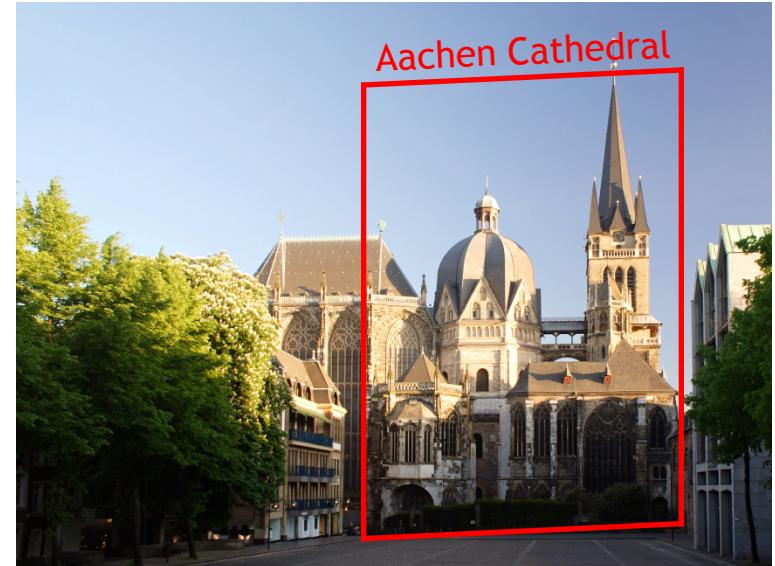
RANSAC verification



RANSAC verification



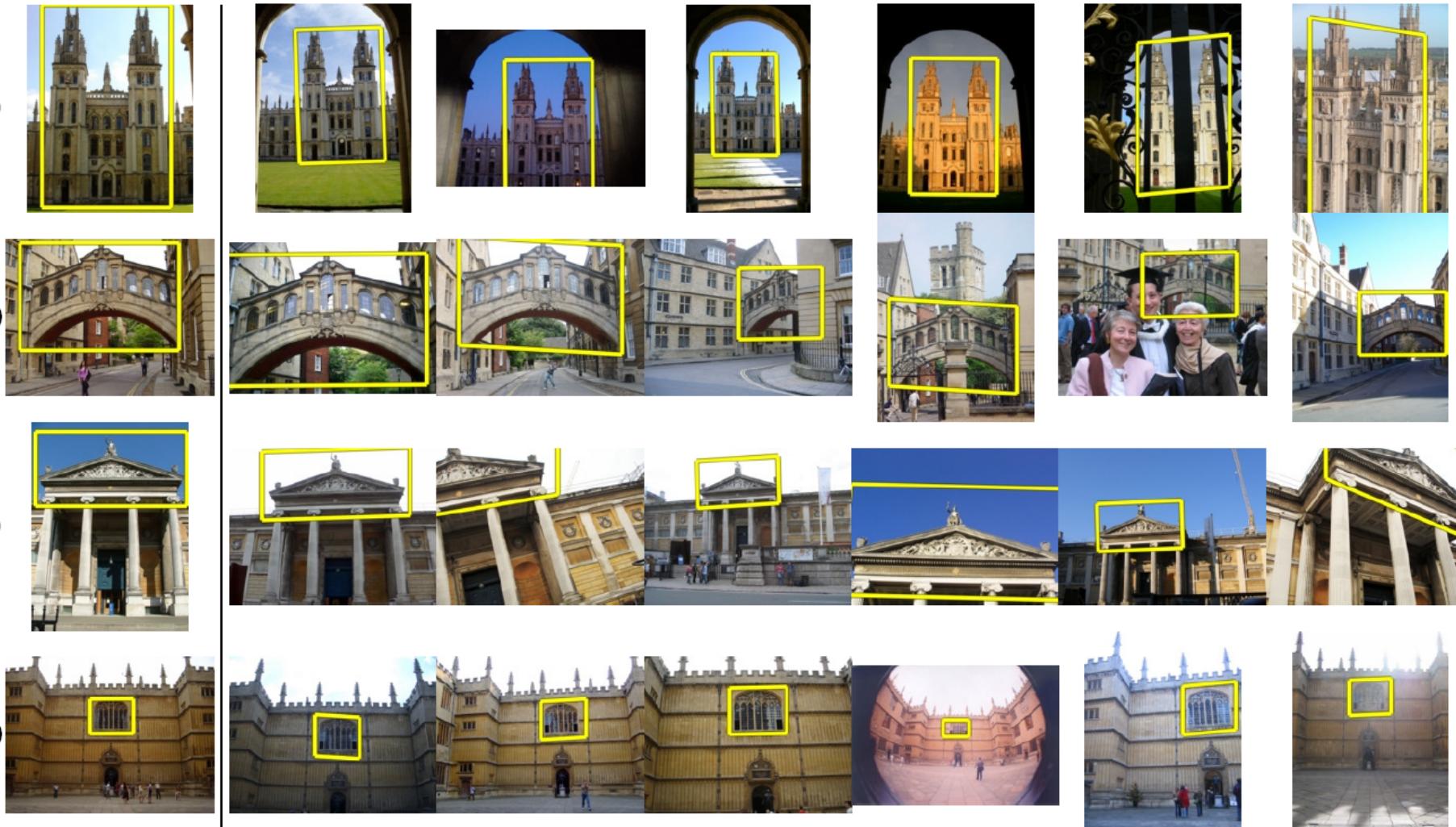
Application: landmark recognition



Mobile tourist guide

- Self-localization
- Object or building recognition
- Photo or video augmentation

Application: large-scale retrieval



Query

Results from 5k Flickr images (demo available for 100k set)

Application: movie poster recognition

50'000 movie posters indexed

Query-by-image from mobile phone

http://www.kooaba.ch - kooaba ooserver: random_movie - Mozilla Firefox

kooaba



JACKIE CHAN OWEN WILSON

SHANGHAI NOON

THE CLASSIC WESTERN GETS A KICK IN THE PANTS.

Show another poster

1. Take a picture with your mobile phone camera
2. Send it:
 - in Switzerland to **5555** (Orange Customers 079 394 5700).
 - in Germany to **84000**
 - everywhere else to **m@kooaba.ch**
3. Search result is sent straight to your phone.

Done



Scoring retrieval quality

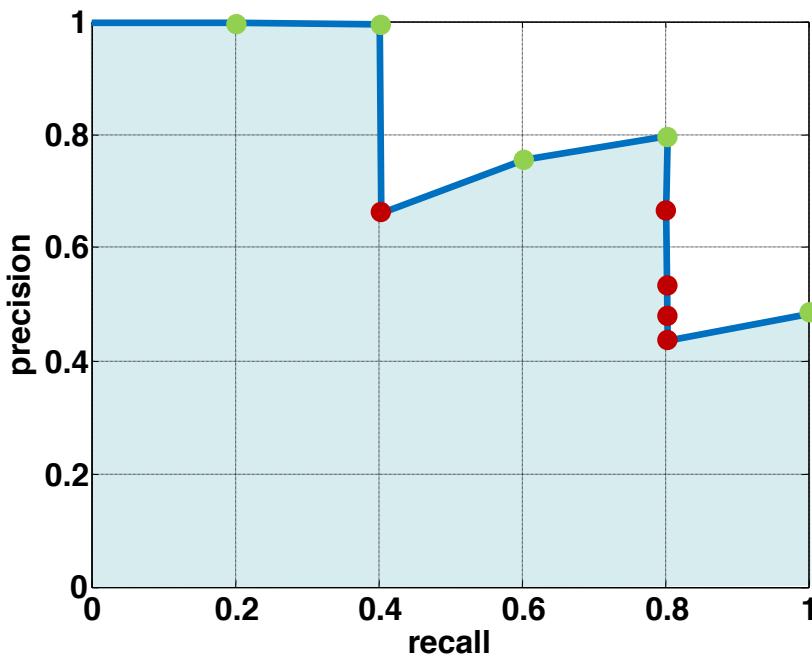


Database size: 10 images
Relevant (total): 5 images

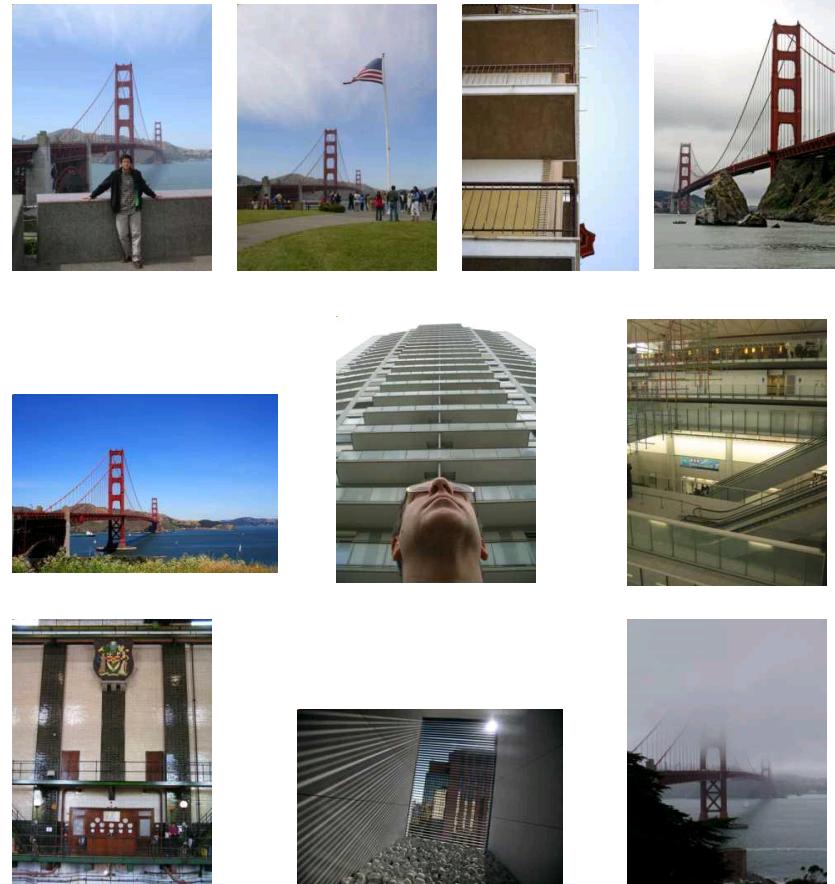
Query

$$\text{precision} = \frac{\text{Number of relevant}}{\text{Number of returned}}$$

$$\text{recall} = \frac{\text{Number of relevant}}{\text{Number of total relevant}}$$

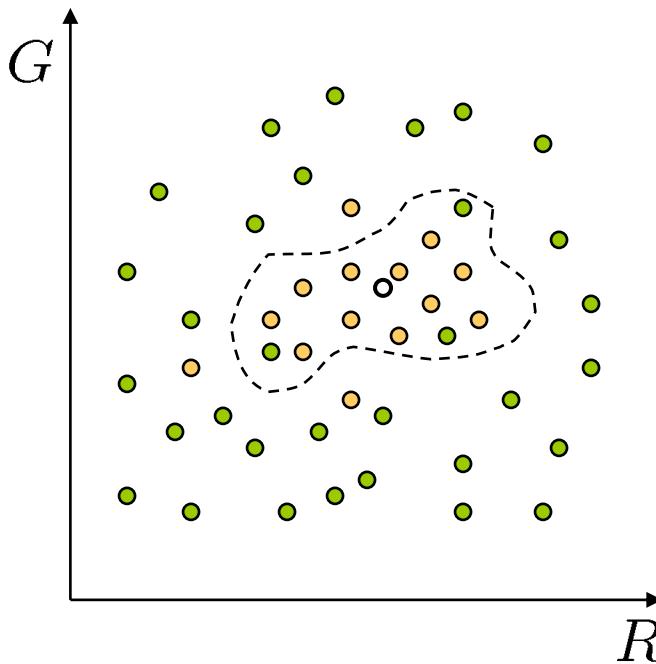


Results (ordered):



Bayesian Estimation

Skin classification techniques



Skin classifier

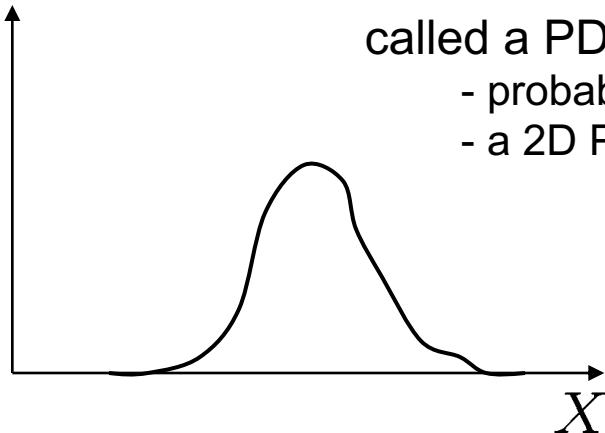
- Given $X = (R, G, B)$: how to determine if it is skin or not?
- Nearest neighbor
 - find labeled pixel closest to X
 - choose the label for that pixel
- Data modeling
 - fit a model (curve, surface, or volume) to each class
- Probabilistic data modeling
 - fit a probability model to each class

Probability

- Basic probability
 - \mathbf{X} is a random variable
 - $\mathbf{P}(\mathbf{X})$ is the probability that \mathbf{X} achieves a certain value

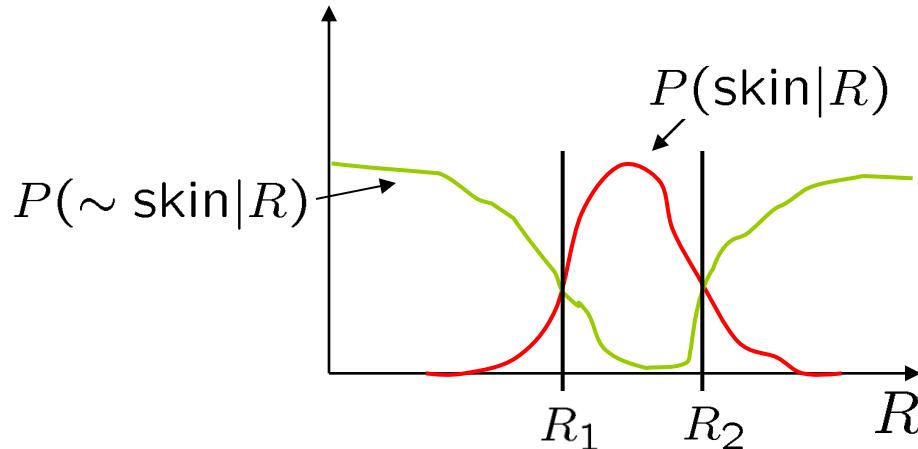
$P(X)$ called a PDF

- probability distribution/density function
- a 2D PDF is a surface, 3D PDF is a volume



- $0 \leq P(X) \leq 1$
- $\int_{-\infty}^{\infty} P(X)dX = 1$ or $\sum P(X) = 1$
 - continuous \mathbf{X}
 - discrete \mathbf{X}
- Conditional probability: $\mathbf{P}(\mathbf{X} | \mathbf{Y})$
 - probability of \mathbf{X} given that we already know \mathbf{Y}

Probabilistic skin classification



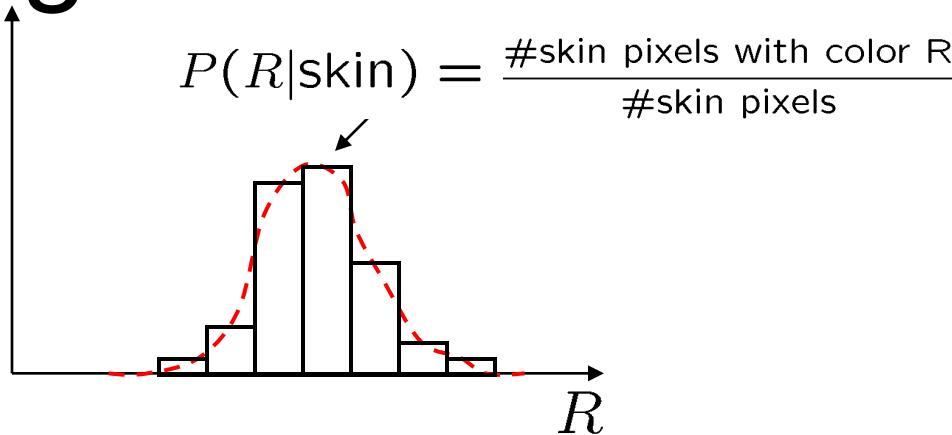
- Now we can model uncertainty
 - Each pixel has a probability of being skin or not skin
 - $P(\sim \text{skin}|R) = 1 - P(\text{skin}|R)$

Skin classifier

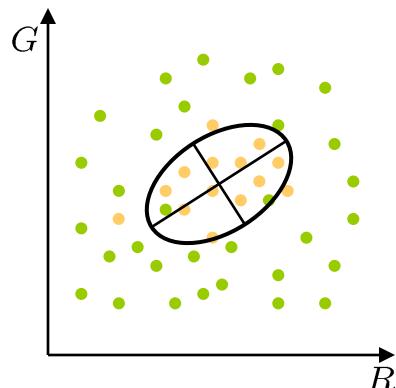
- Given $X = (R, G, B)$: how to determine if it is skin or not?
- Choose interpretation of highest probability
 - set X to be a skin pixel if and only if $R_1 < X \leq R_2$

Where do we get $P(\text{skin}|R)$ and $P(\sim \text{skin}|R)$?

Learning conditional PDFs



- We can calculate $\mathbf{P}(\mathbf{R} \mid \text{skin})$ from a set of training images
 - It is simply a histogram over the pixels in the training images
 - each bin R_i contains the proportion of skin pixels with color R_i

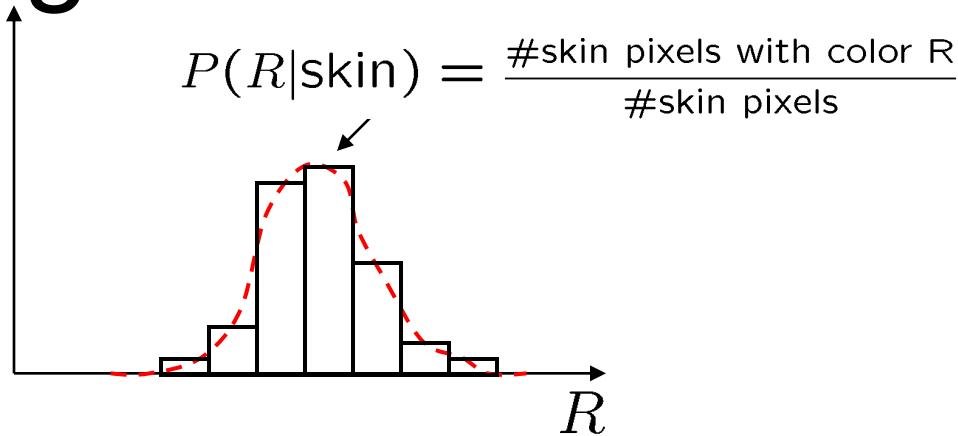


Approach: fit parametric PDF functions

- common choice is rotated Gaussian
 - center $c = \bar{X}$
 - covariance $\sum_X (X - \bar{X})(X - \bar{X})^T$

» orientation, size defined by eigenvecs, eigenvals

Learning conditional PDFs



- We can calculate $\mathbf{P}(\mathbf{R} \mid \mathbf{skin})$ from a set of training images
 - It is simply a histogram over the pixels in the training images
 - each bin R_i contains the proportion of skin pixels with color R_i

But this isn't quite what we want

- Why not? How to determine if a pixel is skin?
- We want $\mathbf{P}(\mathbf{skin} \mid \mathbf{R})$, not $\mathbf{P}(\mathbf{R} \mid \mathbf{skin})$
- How can we get it?

Bayes rule

$$P(X|Y) = \frac{P(Y|X)P(X)}{P(Y)}$$

- In terms of our problem:

$$P(\text{skin}|R) = \frac{P(R|\text{skin}) P(\text{skin})}{P(R)}$$

↑

what we measure
(likelihood)

what we want
(posterior)

domain knowledge
(prior)

normalization term

$$P(R) = P(R|\text{skin})P(\text{skin}) + P(R|\sim \text{skin})P(\sim \text{skin})$$

The prior: **P(skin)**

- Could use domain knowledge
 - **P(skin)** may be larger if we know the image contains a person
 - For a portrait, **P(skin)** may be higher for pixels in the center
- Could learn the prior from the training set. How?
 - **P(skin)** could be the proportion of skin pixels in training set

Skin detection results

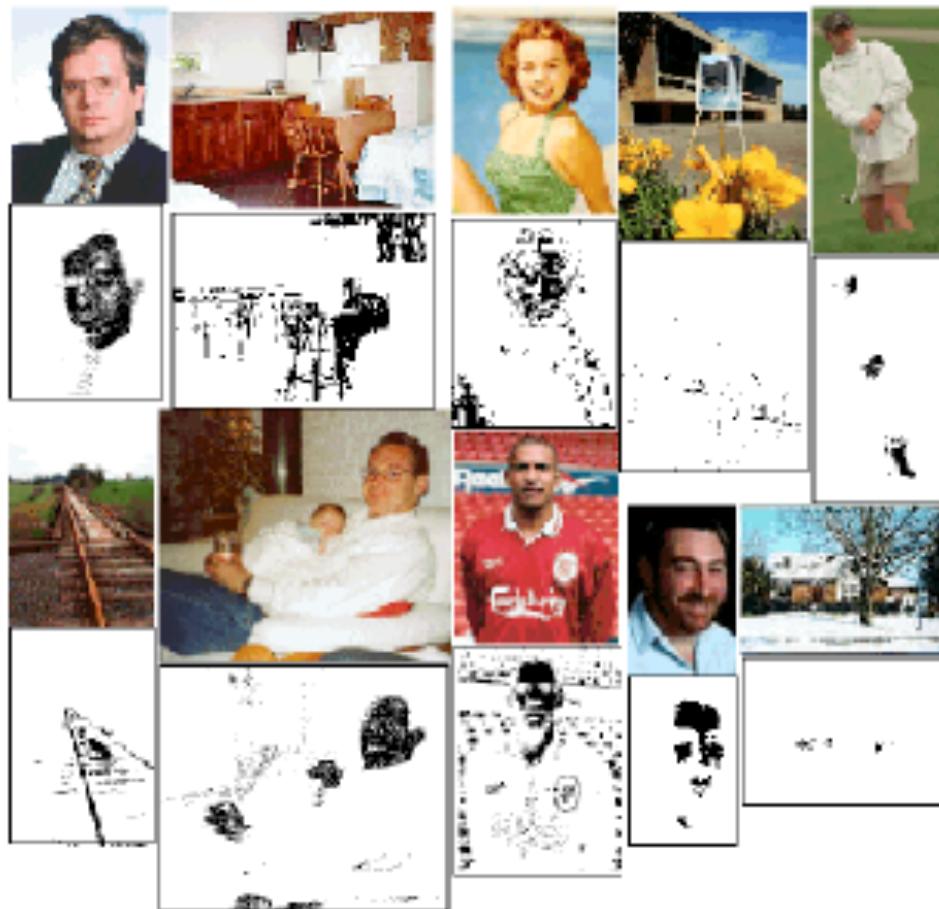


Figure 25.3. The figure shows a variety of images together with the output of the skin detector of Jones and Rehg applied to the image. Pixels marked black are skin pixels, and white are background. Notice that this process is relatively effective, and could certainly be used to focus attention on, say, faces and hands. *Figure from "Statistical color models with application to skin detection," M.J. Jones and J. Rehg, Proc. Computer Vision and Pattern Recognition, 1999 © 1999, IEEE*