Computer Vision

António J. R. Neves / Paulo Dias

Departamento de Electrónica, Telecomunicações e Informática

Universidade de Aveiro

an@ua.pt / paulo.dias@ua.pt

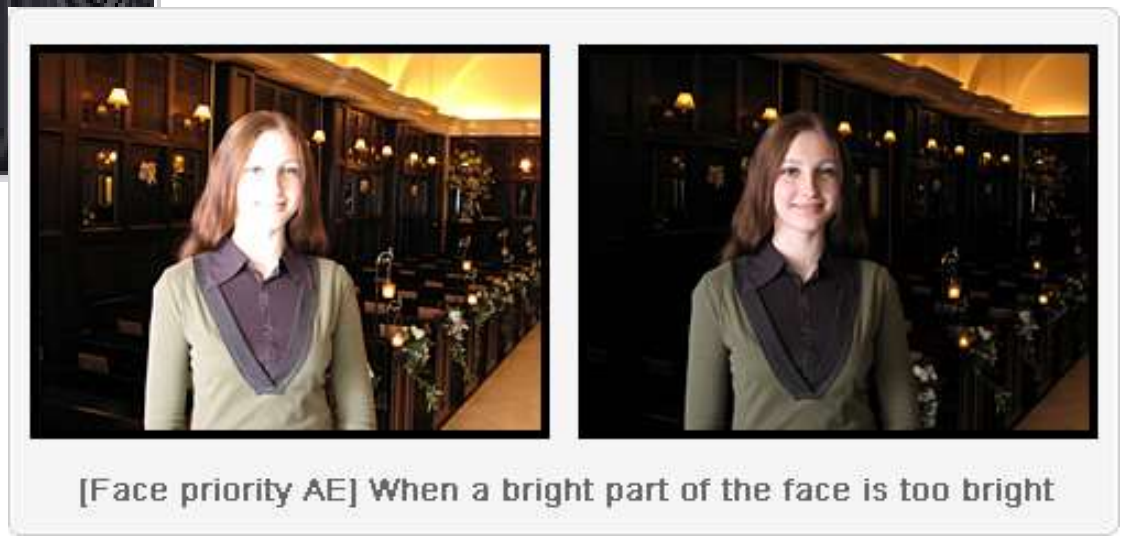
http://elearning.ua.pt/

Object detection: where are we?

Credit: Flickr user neilalderney123

- Incredible progress in the last ten years
- Better features, better models, better learning methods, better datasets
- Combination of science and hacks

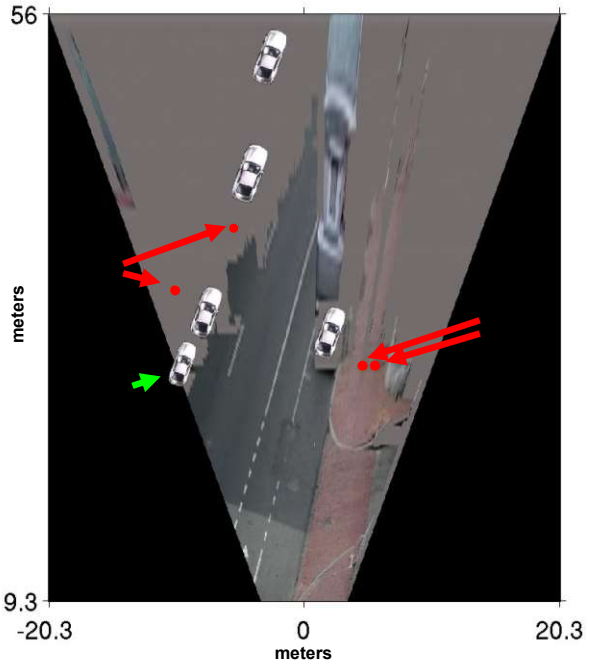
Applications: Computational photography

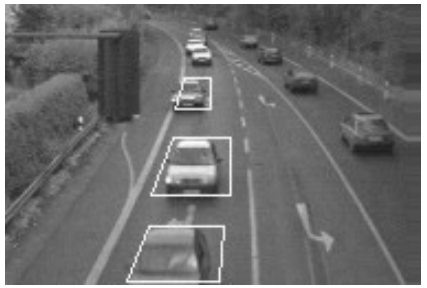


[Face priority AE] When a bright part of the face is too bright

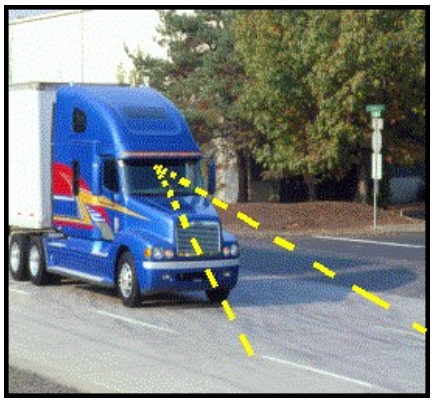
Applications: Assisted driving

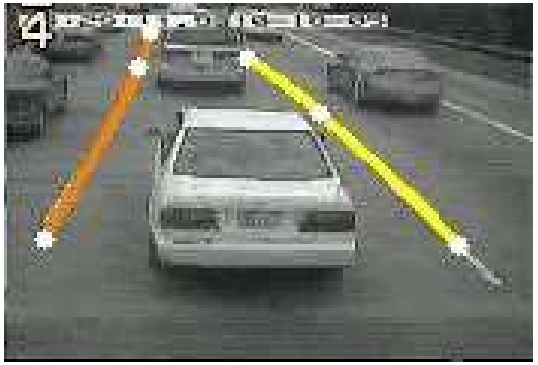
Pedestrian and car detection





Lane detection





- **Collision warning systems with adaptive cruise control,**
- **Lane departure warning systems,**
- **Rear object detection systems,**

Applications: image search

Google Similar Images LABS

Search images

Places
London
New York
Egypt
Forbidden City

Celebrities
Michael Jordan
Angelina Jolie
Halle Berry
Seth Rogan
Rihanna

Art
impressionism
Keith Haring
cubism
Salvador Dalí
pointillism

Shopping
evening gown
necklace
shoes

Refine your image search with visual similarity

Similar Images allows you to search for images using pictures rather than words. Click the "Similar images" link under an image to find other images that look like it. Try a search of your own or click on an example below.

paris


Similar images


Similar images


Similar images


Similar images

temple


Similar images

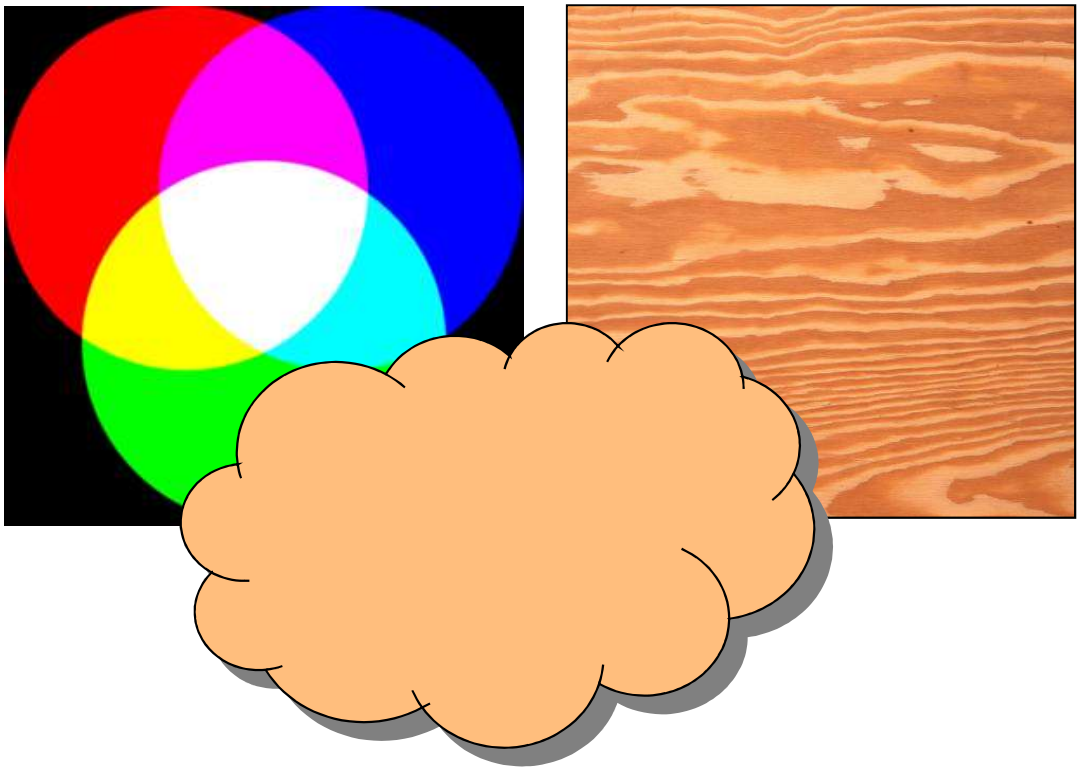

Similar images


Similar images


Similar images

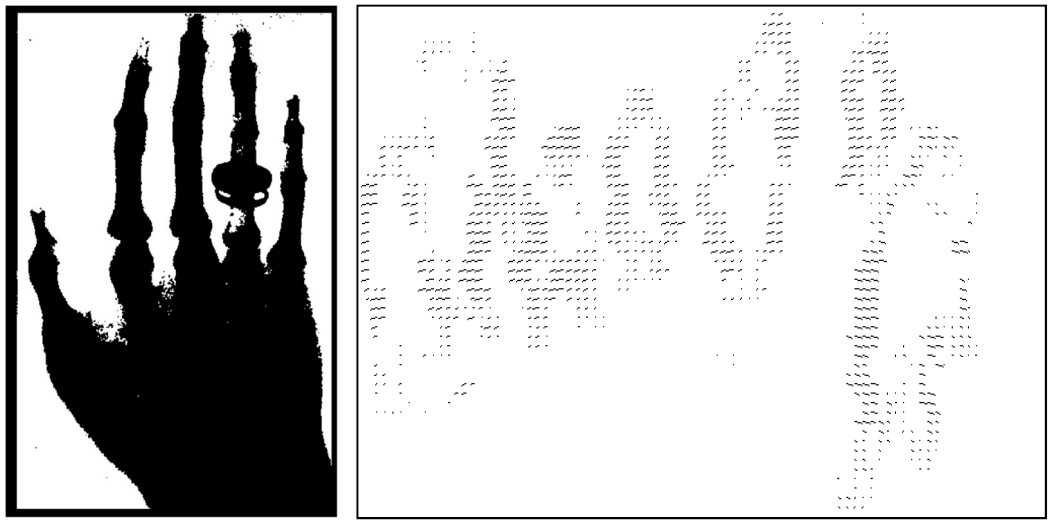
Low-level features

- Objective
- Directly reflect specific image and video features.
  - Colour
  - Texture
  - Shape
  - Motion
  - Etc.



Middle-level features

- Some degree of subjectivity
- They are typically one solution of a problem with multiple solutions.
- Examples:
  - Segmentation
  - Optical Flow
  - Identification
  - Etc.



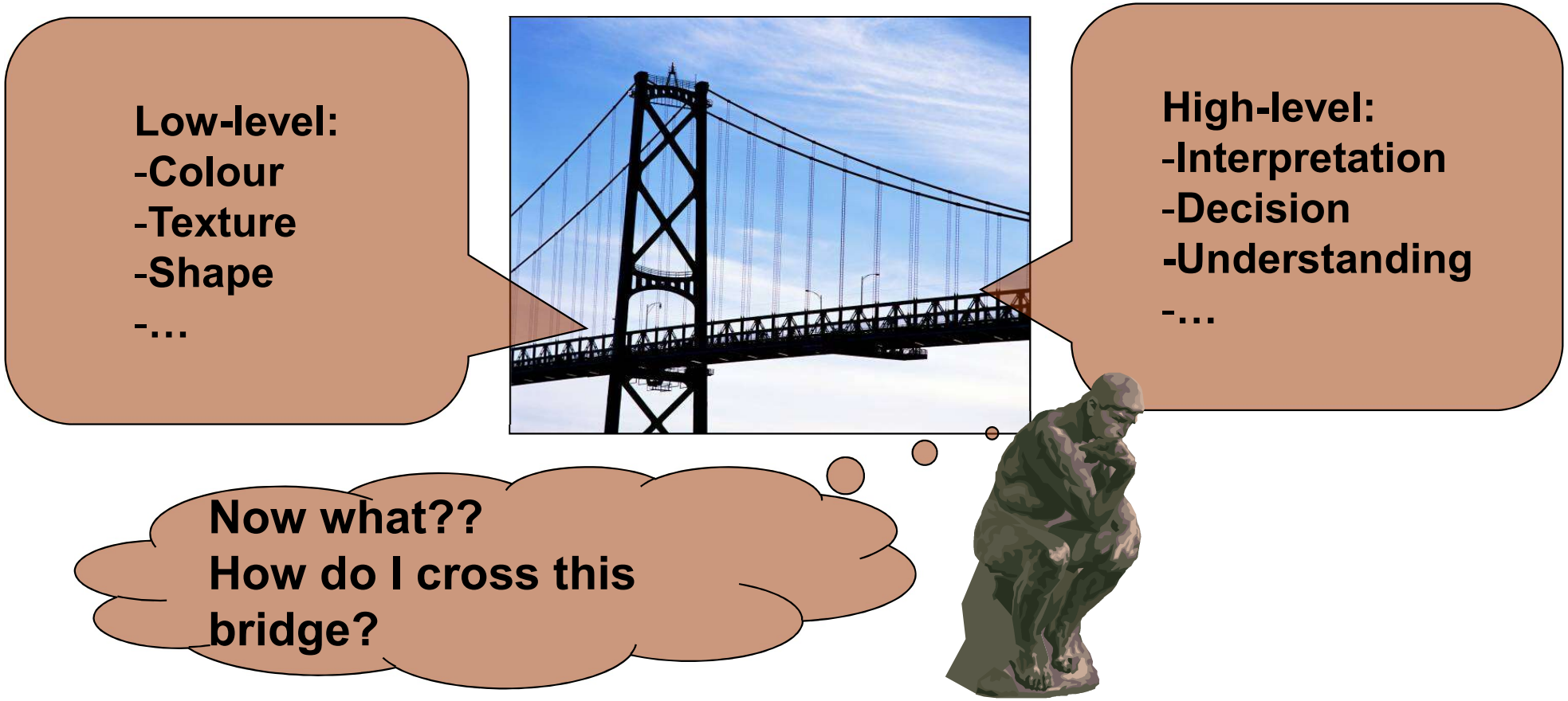
High-level features

- Semantic Interpretation
- Knowledge
- Context
- Examples:
  - This person suffers from epilepsy.
  - The virus attacks the cell with some degree of intelligence.
  - This person is running from that one.

How do humans do this so well?

The semantic gap

- Fundamental problem of current research!



Overview

Several approaches for classification/recognition. Choose the same class of objects with:

- **Shape** - similar shape descriptors
- **Appearance** - similar pixel values
- **Geometric** - similar structures in similar places with similar parameters
- **Graph** - similar part relationships
- **Bag of Words** - enough similar local feature descriptors

Shape based recognition

1. Extract object from image (segmentation)
2. Compute properties (features)
3. Use properties to compute class
4. Learning model properties for the classes

An example*

- **Problem**: sorting incoming fish on a conveyor belt according to species

- Assume that we have only two kinds of fish:
  - Salmon
  - Sea bass

Picture taken with a camera

Adapted from Duda, Hart and Stork, Pattern Classification, 2nd Ed.

An example: the problem

What ***humans*** see

0	3	2	5	4	7	6	9	8
3	0	1	2	3	4	5	6	7
2	1	0	3	2	5	4	7	6
5	2	3	0	1	2	3	4	5
4	3	2	1	0	3	2	5	4
7	4	5	2	3	0	1	2	3
6	5	4	3	2	1	0	3	2
9	6	7	4	5	2	3	0	1
8	7	6	5	4	3	2	1	0

What ***computers*** see

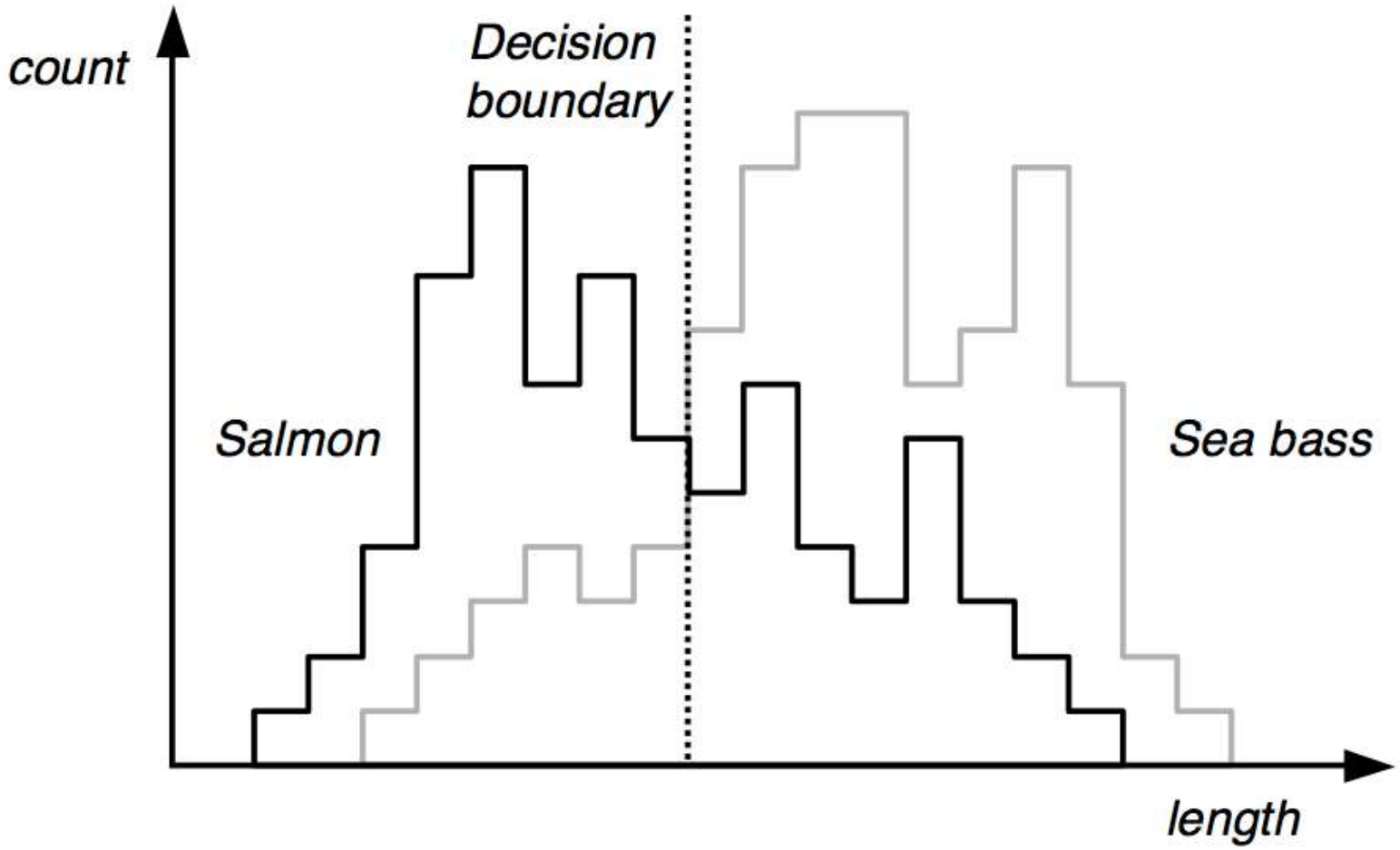
An example: decision process

- What kind of information can distinguish one species from the other?
  - Length, width, weight, number and shape of fins, tail shape, etc.
- What can cause problems during sensing?
  - Lighting conditions, position of fish on the conveyor belt, camera noise, etc.
- What are the steps in the process?
  - Capture image -> isolate fish -> take measurements -> make decision

An example: our system

- **Sensor**
  - The camera captures an image as a new fish enters the sorting area
- **Preprocessing**
  - Adjustments for average intensity levels
  - Segmentation to separate fish from background
- **Feature Extraction**
  - Assume a fisherman told us that a sea bass is generally longer than a salmon. We can use **length** as a feature and decide between sea bass and salmon according to a threshold on length.
- **Classification**
  - Collect a set of examples from both species
    - Plot a distribution of lengths for both classes
  - Determine a decision boundary (threshold) that minimizes the classification error

An example: features

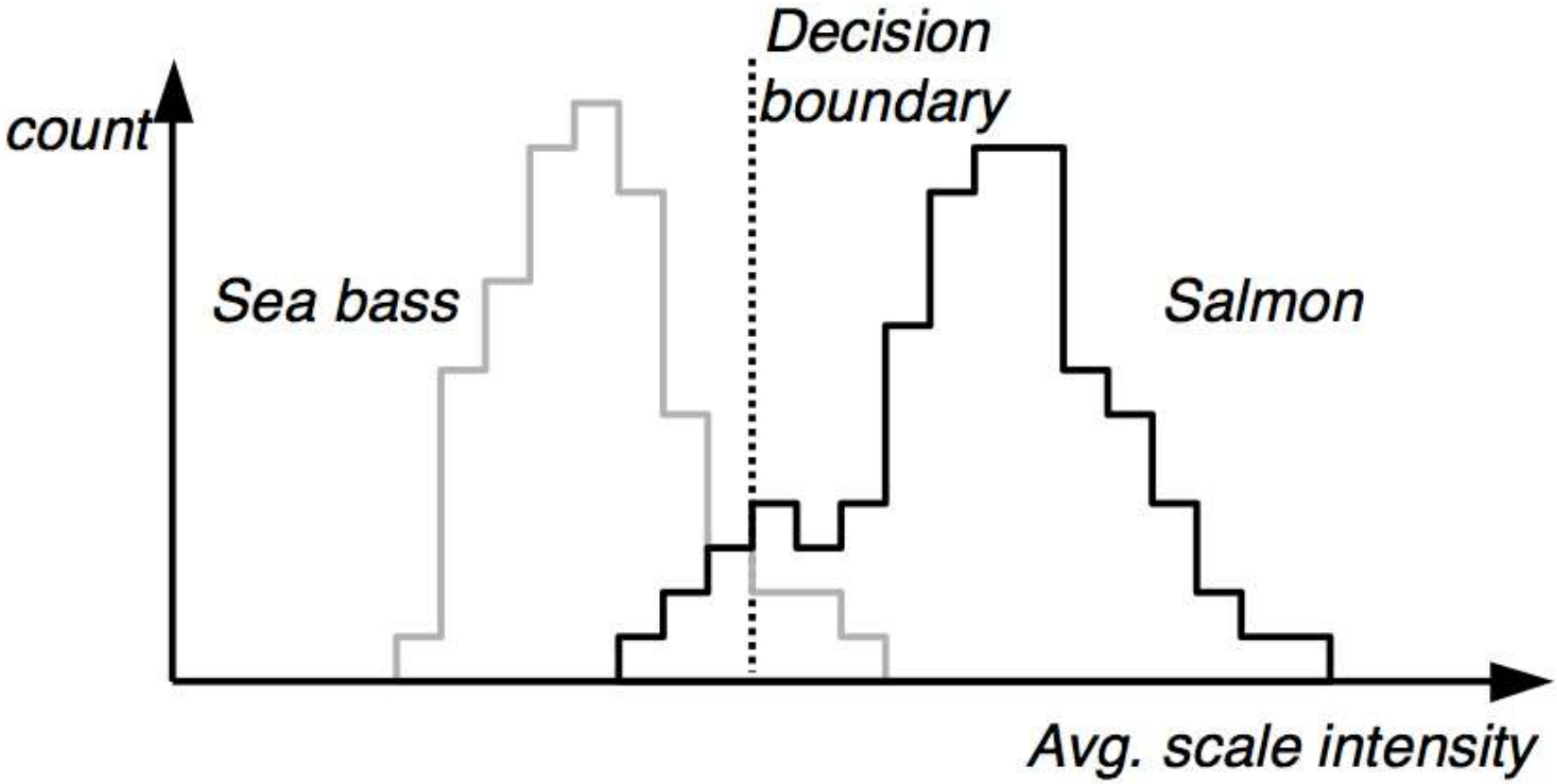


We estimate the system's probability of error and obtain a discouraging result of 40%. Can we improve this result?

An example: features

- Even though sea bass is longer than salmon on the average, there are many examples of fish where this observation does not hold
- Committed to achieve a higher recognition rate, we try a number of features
  - Width, Area, Position of the eyes w.r.t. mouth...
  - only to find out that these features contain no discriminatory information
- Finally we find a “good” feature: **average intensity of the scales**

An example: features



Histogram of the lightness feature for two types of fish in **training samples**. It looks easier to choose the threshold but we still can not make a perfect decision.

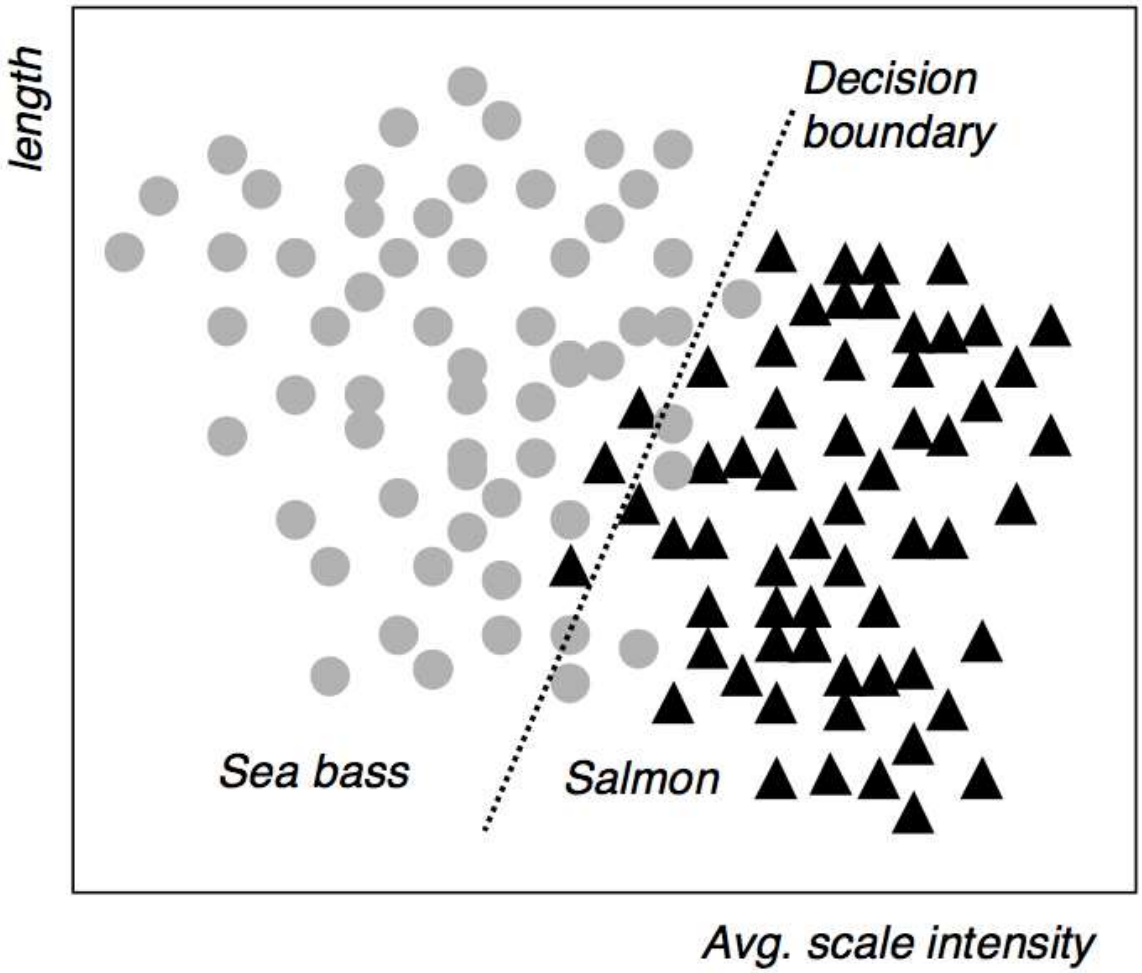
An example: multiple features

- We can use two features in our decision:
  - lightness: $\boldsymbol{x}_1$
  - length: $\boldsymbol{x}_2$
- Each fish image is now represented as a point (feature vector)

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

in a two-dimensional **feature space**.

An example: multiple features

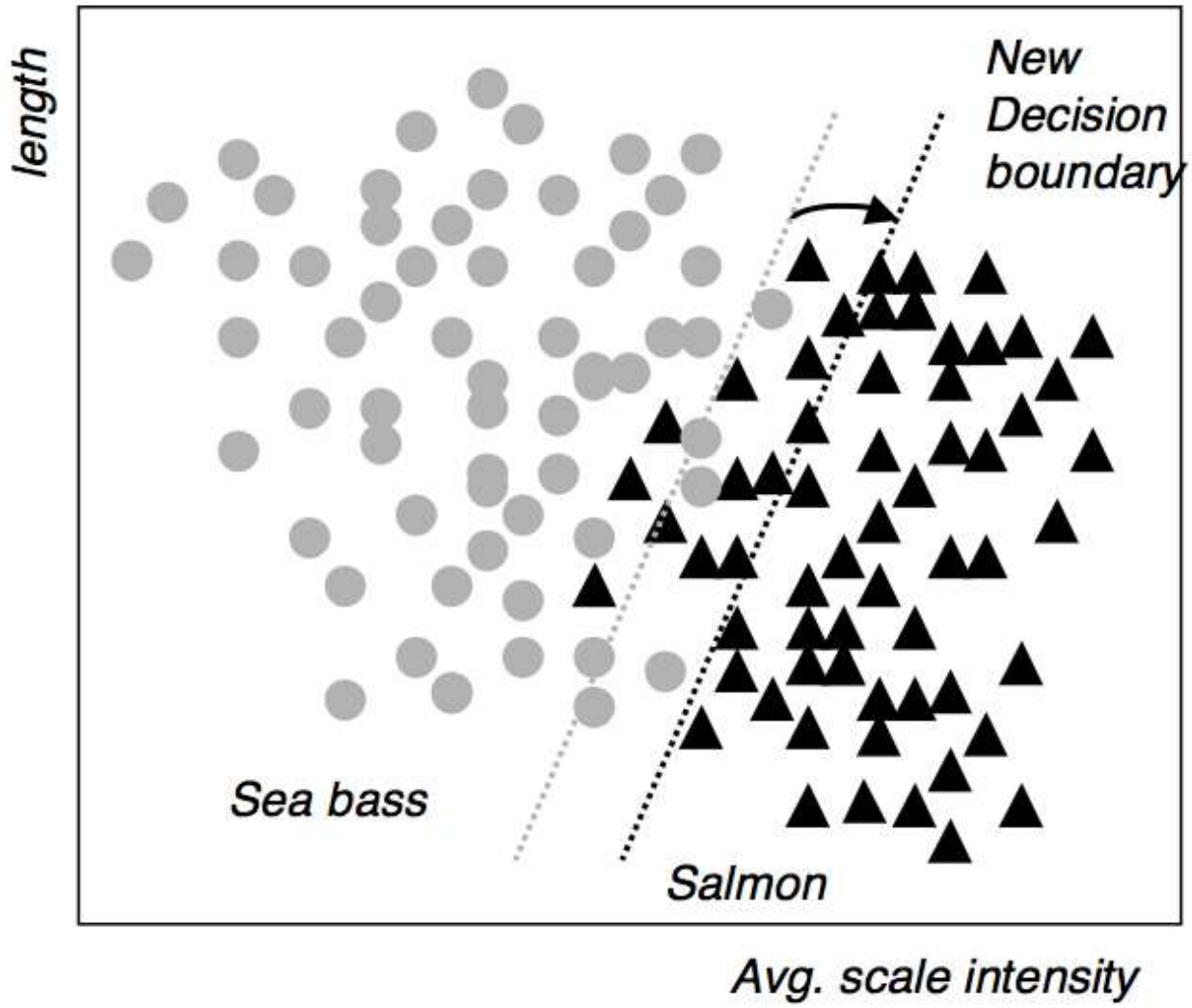


Scatter plot of lightness and length features for training samples. We can compute a **decision boundary** to divide the feature space into two regions with a classification rate of 95.7%.

An example: cost of error

- We should also consider **costs of different errors** we make in our decisions.
- For example, if the fish packing company knows that:
  - Customers who buy salmon will object vigorously if they see sea bass in their cans.
  - Customers who buy sea bass will not be unhappy if they occasionally see some expensive salmon in their cans.
- How does this knowledge affect our decision?

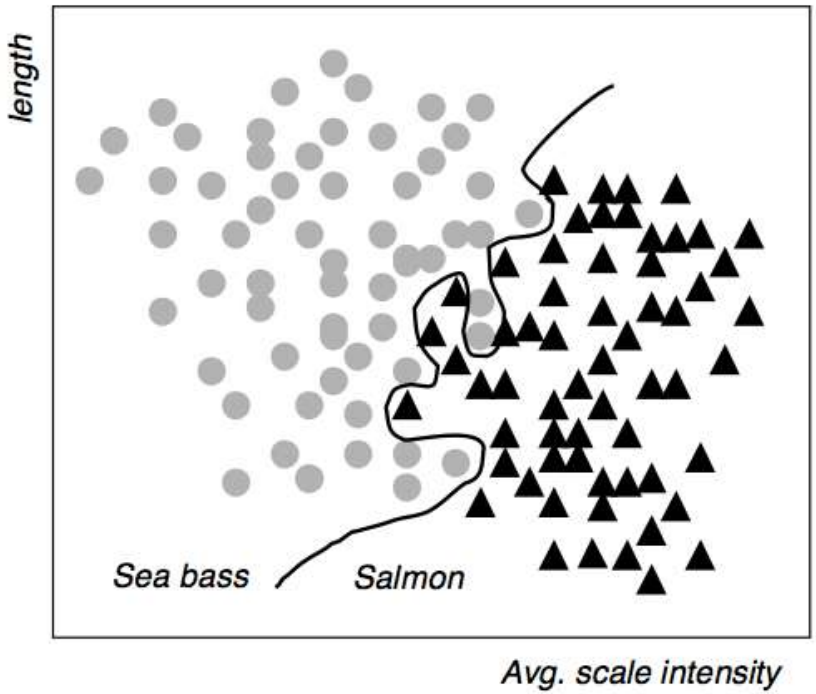
An example: cost of error



We could intuitively shift the decision boundary to minimize an alternative cost function

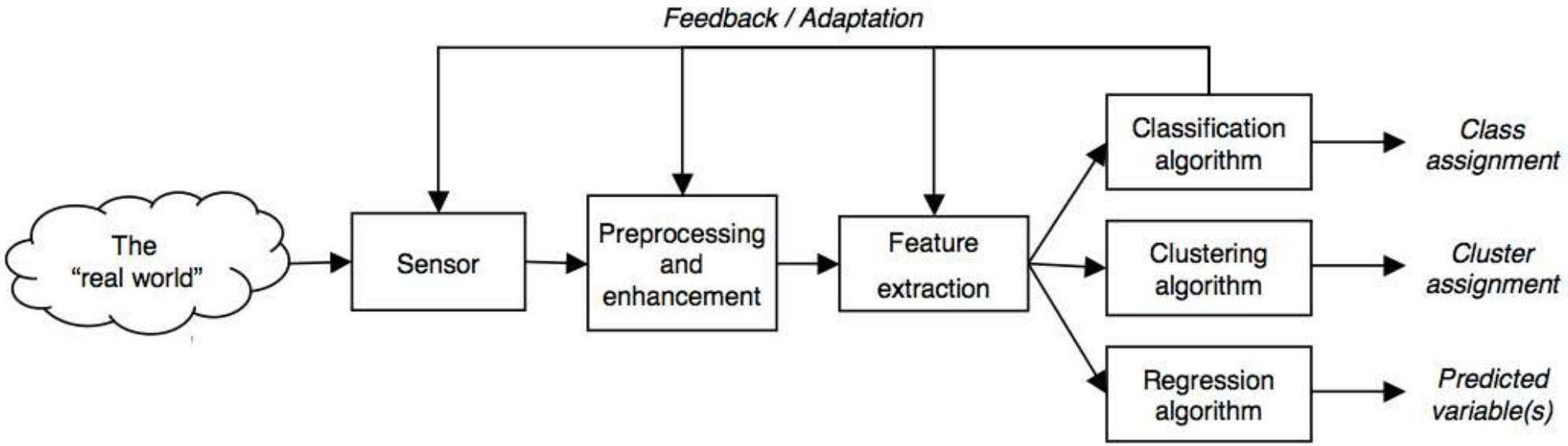
An example: generalization

- **The issue of generalization**
  - The recognition rate of our linear classifier (95.7%) met the design specifications, but we still think we can improve the performance of the system
  - We then design a über-classifier that obtains an impressive classification rate of 99.9975% with the following decision boundary



Pattern Recognition System

- **A typical pattern recognition system contains**
  - A sensor
  - A preprocessing mechanism
  - A feature extraction mechanism (manual or automated)
  - A classification or description algorithm
  - A set of examples (training set) already classified or described



Algorithms

- Classification
  - **Supervised**, **categorical** labels
  - Bayesian classifier, KNN, SVM, Decision Tree, Neural Network, etc.
- Clustering
  - **Unsupervised**, **categorical** labels
  - Mixture models, K-means clustering, Hierarchical clustering, etc.
- Regression
  - **Supervised or Unsupervised**, **real**-valued labels

Curse of dimensionality

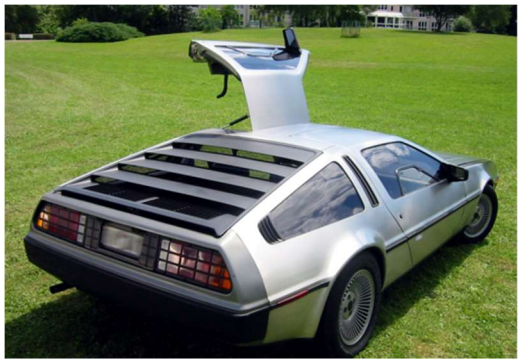
- KNN is easily misled in a high-dimension space
- Why?
  - Easy problems in low-dim are hard in hi-dim
  - Low-dim intuitions do not apply in hi-dim
- Examples
  - Normal distribution
  - Uniform distribution on hypercube
  - Points on hypergrid
  - Approximation of hypersphere by a hypercube
  - Volume of hypersphere

Feature selection

- **Filter approach**
  - Pre-select features individually (e.g. by information gain)
  - Find best transformation that reduces dimensionality (e.g. PCA – Principal Component Analysis)
- **Wrapper approach**
  - Run learner with different combinations of features
    - Forward selection
    - Backward selection
    - Etc.

Object Recognition (2)

- How to recognise these and similar objects



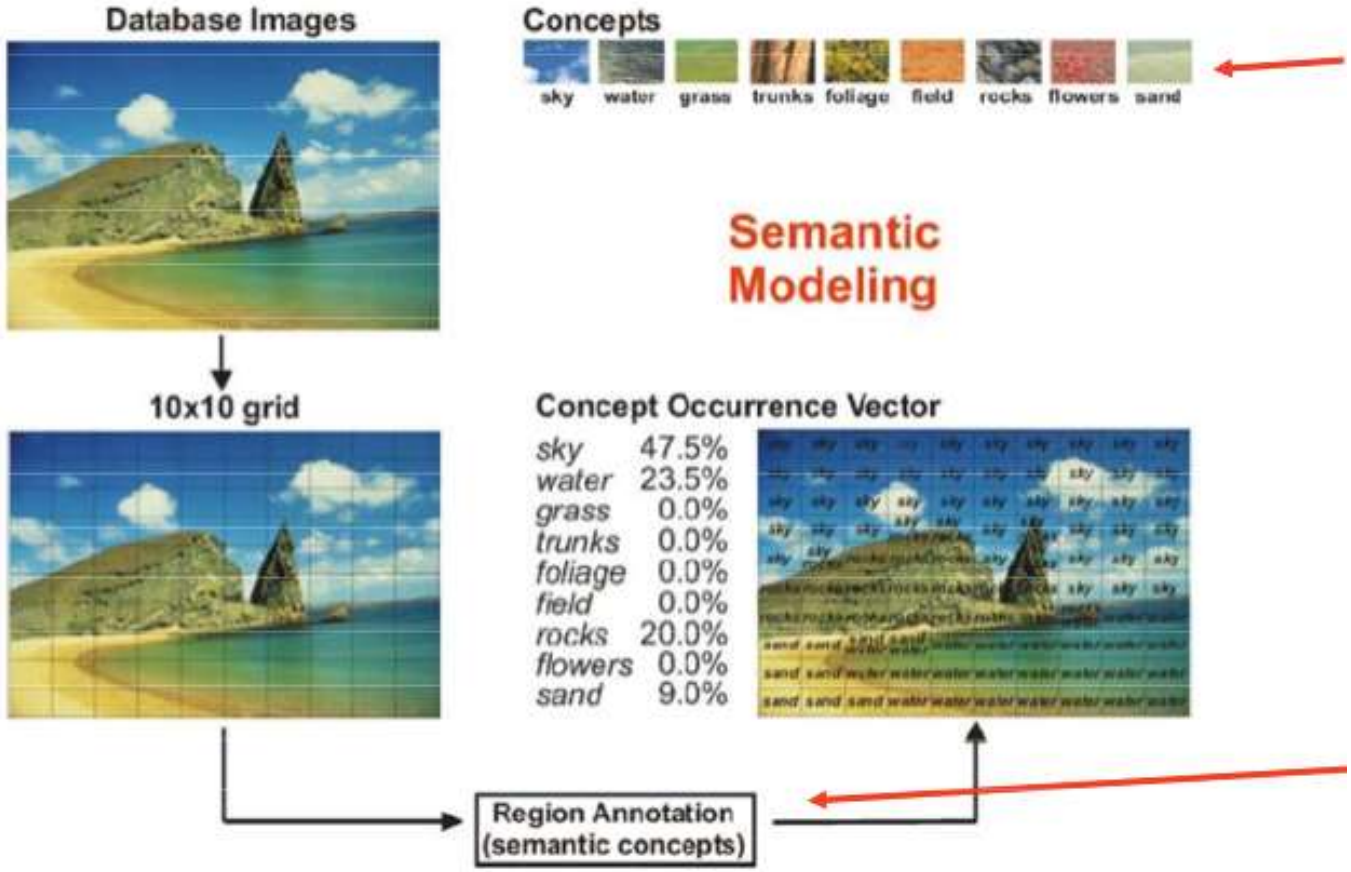
How?

- How can we subdivide an image?
  - Object segmentation, not always an easy task
    - When the background can be modeled, we can perform background subtraction

  - Grid division
  - Exhaustive search
  - Local features

Grid division

- **Exhaustive grid division**: the whole image is divided into blocks with **no overlap**



Concepts modeled by color and texture features as in the global case. However, we can extract more information per pixel as the area under analysis is smaller

Classification for each image subdivision obtained by SVM classifier

Semantic Scene Modeling and Retrieval for Content-Based Image Retrieval. Julia Vogel and Bernt Schiele. *International Journal of Computer Vision*. Vol. 72, No. 2, pp. 133-157, April 2007.

Grid division

- We may miss some objects if these are split over several image blocks
- **Over-sampling grid division**: the whole image is divided into blocks **with overlap**
- Redundant, but less prone to miss objects.

Scene Classification via PLSA. A.Bosch, A.Zisserman, X.Muñoz European Conference on Computer Vision, vol. IV, pp. 517-530. Graz, Austria. May 2006.

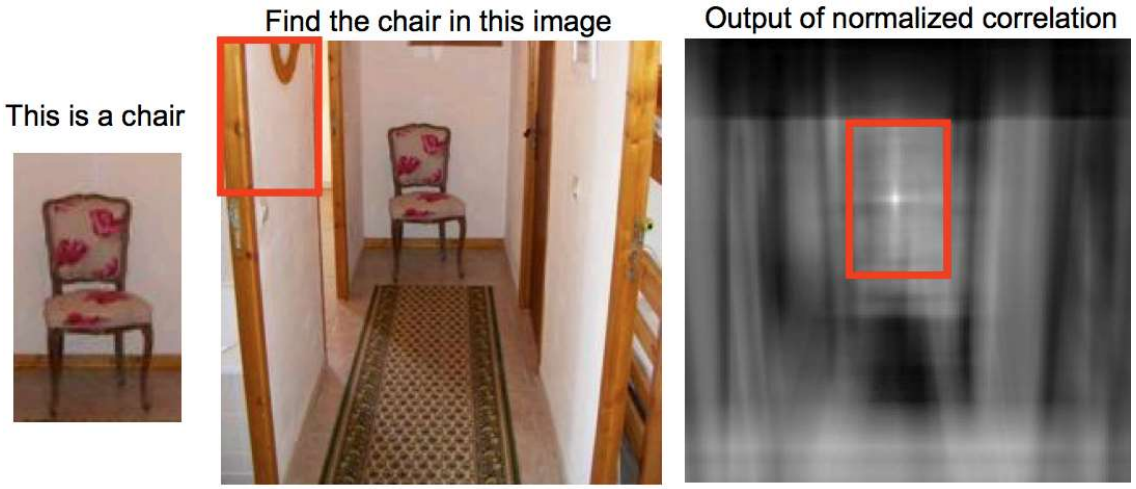
Exhaustive search

- **Scanning image division**: the image is scanned with a fine regular sampling into block (very redundant).
  - Similar to a grid division. However, it is more exhaustive.
  - To detect object at several scales several passes have to be made with variable window size (same applies to rotation).

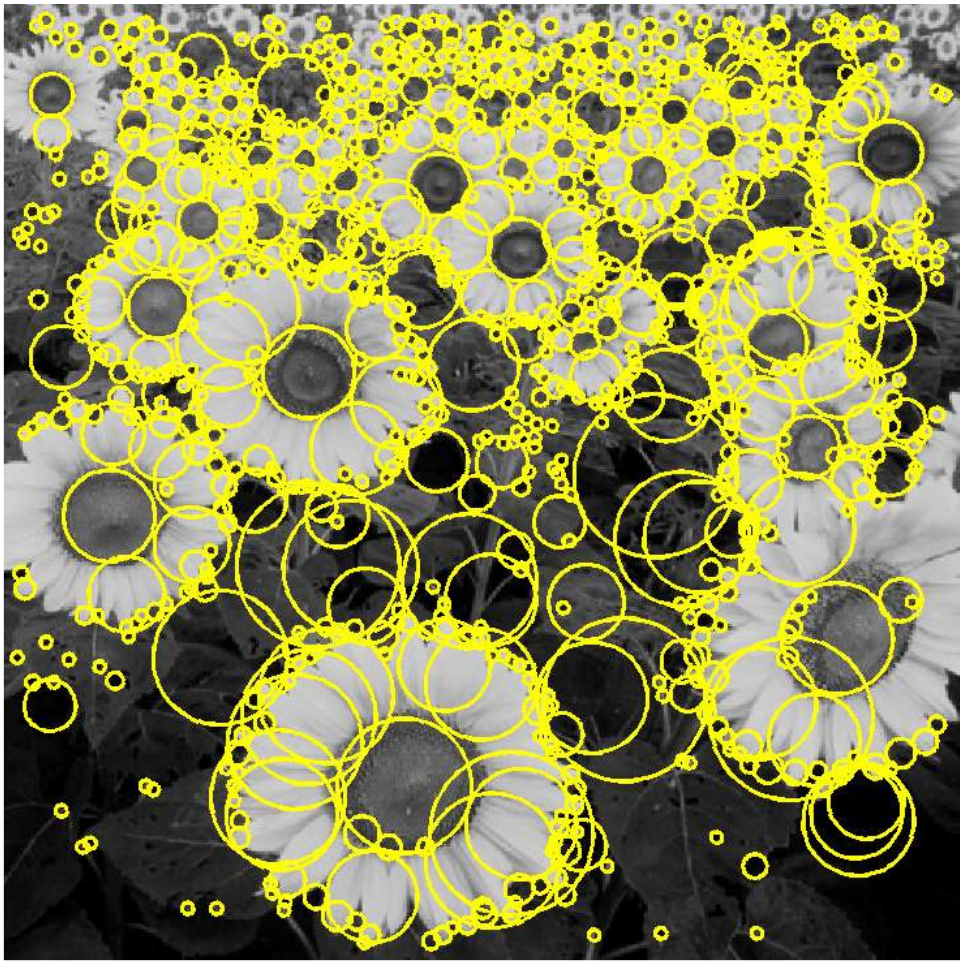
  - We can do this using **template matching**: cross-correlate the pixels in each area with a model template

Exhaustive search

- **Template matching -** sensitive to noise and **computationally expensive**, i.e. requires presented image to be correlated with every image in the database (no generalization power)

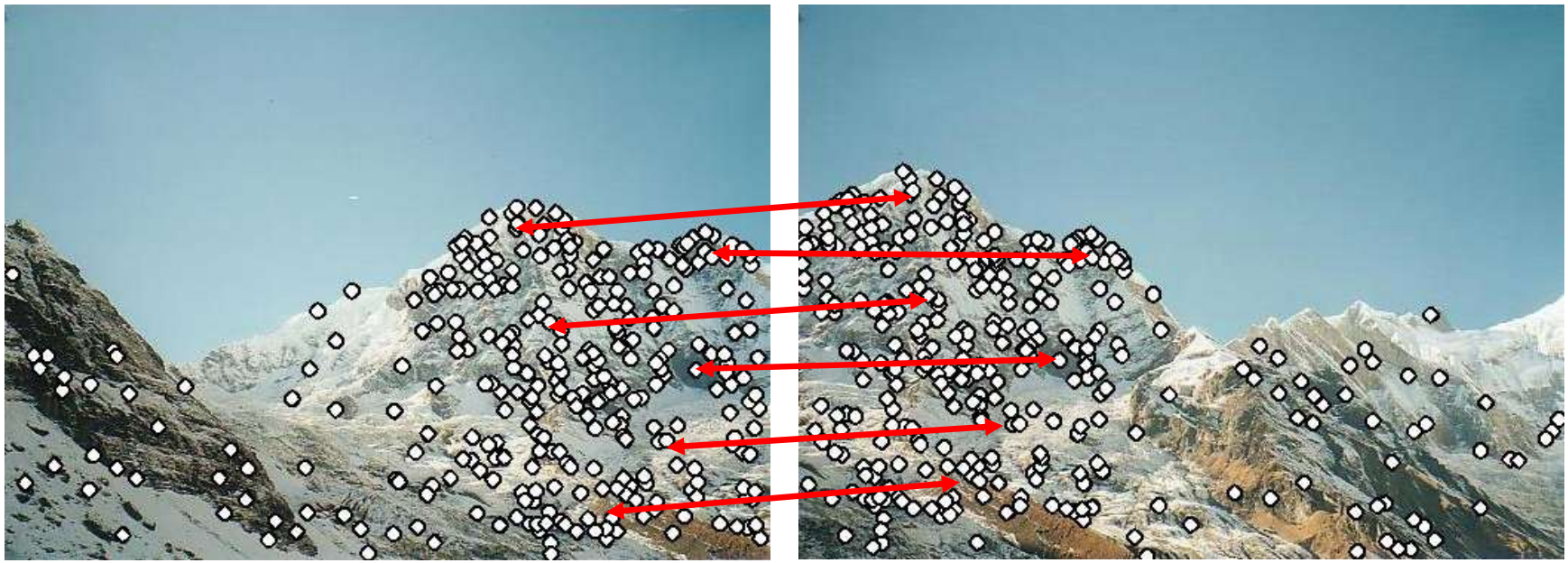


Feature extraction: Corners and blobs



Why extract features?

- Motivation: panorama stitching
  - We have two images – how do we combine them?



Step 1: extract features
Step 2: match features

Why extract features?

- Motivation: panorama stitching
  - We have two images – how do we combine them?

Step 1: extract features
Step 2: match features
Step 3: align images

Image matching

by Diva Sian

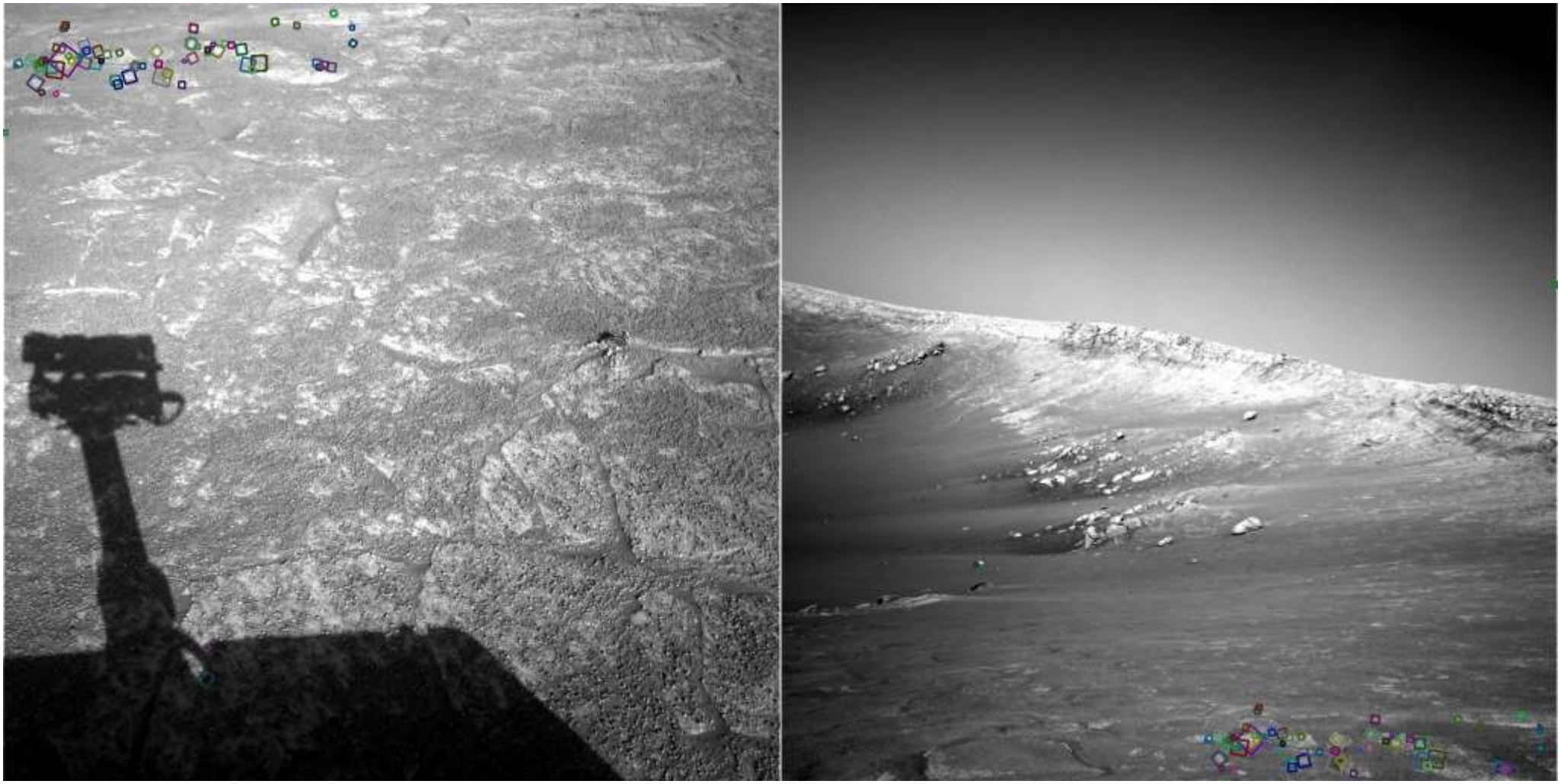
by swashford

Harder case

by Diva Sian

by scgbt

Harder still? (look for tiny colored squares…)



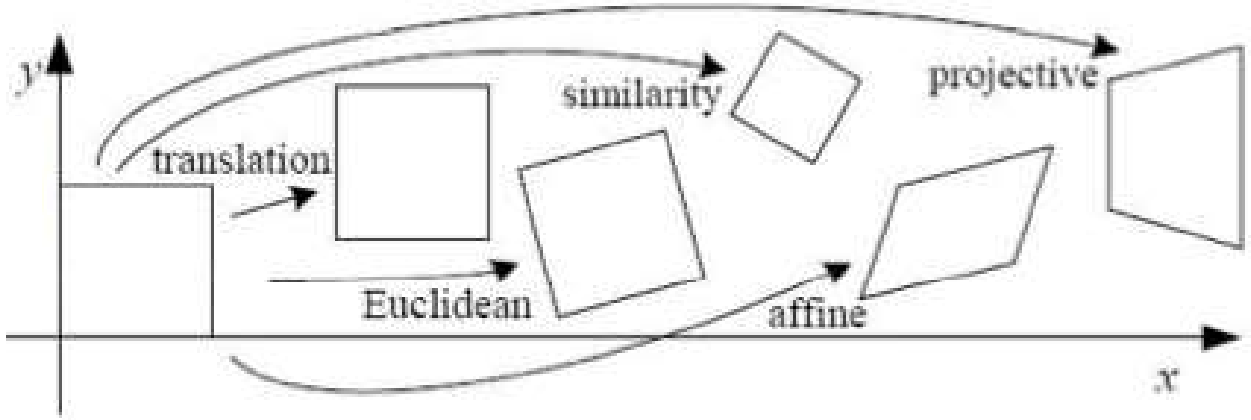
NASA Mars Rover images
with SIFT feature matches

Invariant local features

Geometric transformations

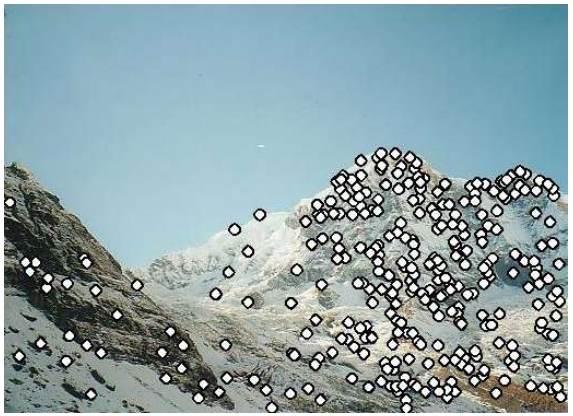
- Translation
- Euclidean (translation + rotation)
- Similarity (translation + rotation + scale)
- Affine transformations
- Projective transformations

Only holds for planar patches



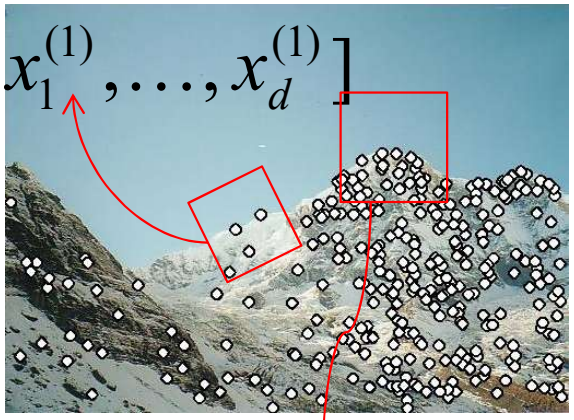
Local features: main components

1) **Detection**: Identify the interest points



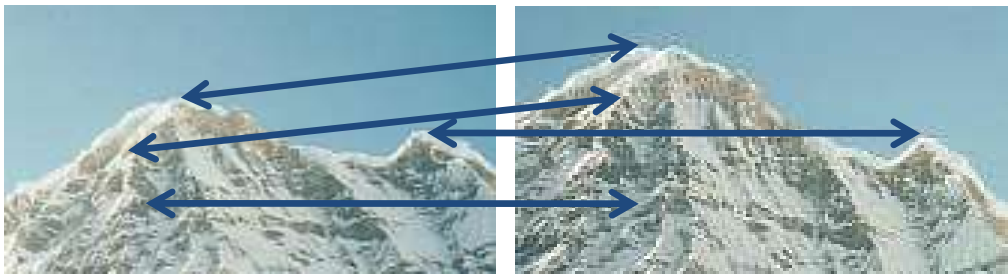
2) **Description**: Extract vector feature descriptor surrounding each interest point.

$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$



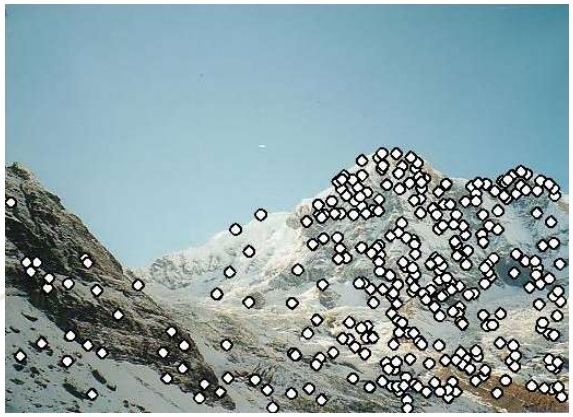
$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$

3) **Matching**: Determine correspondence between descriptors in two views



Local features: main components

1) Detection: Identify the interest points

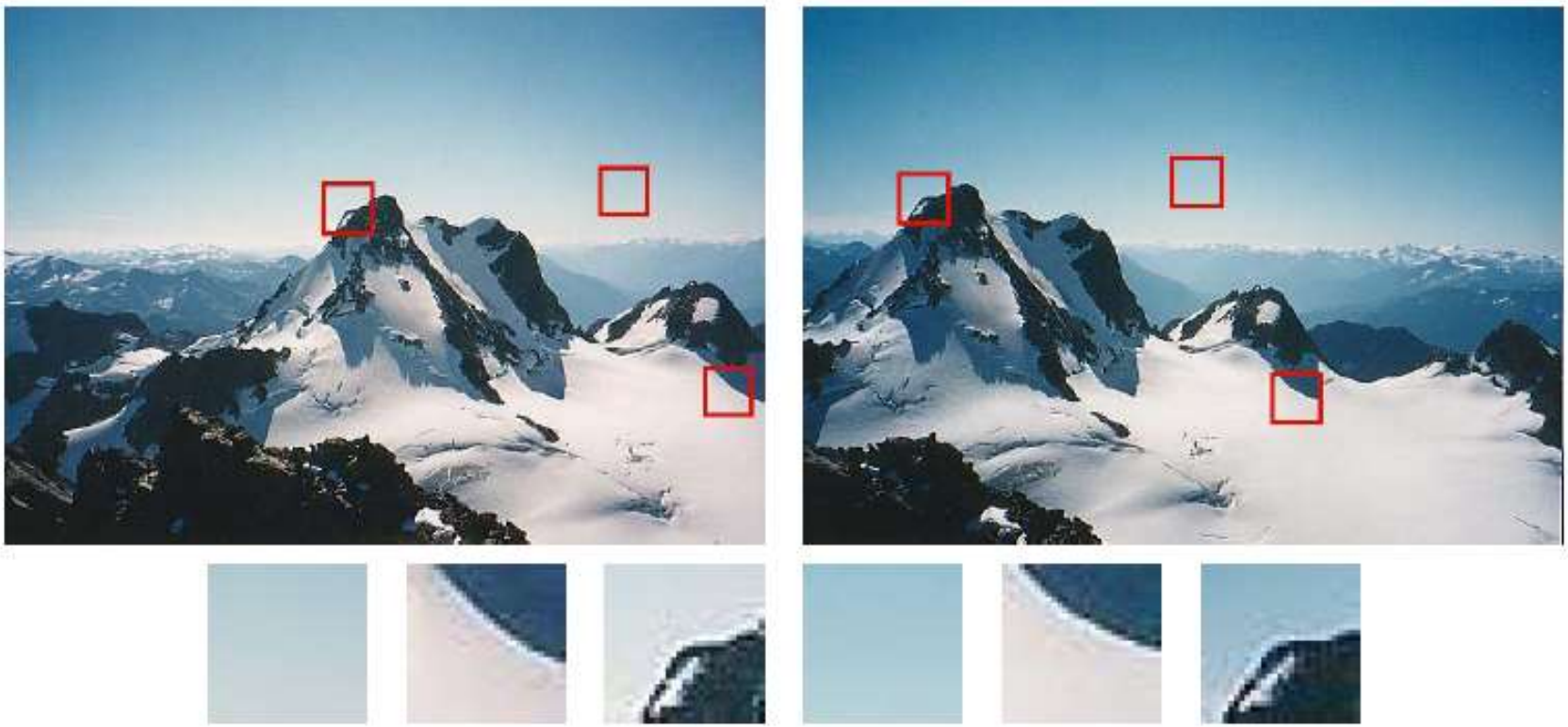


2) Description: Extract vector feature descr
each interest point.

3) Matching: Determine correspondence between descriptors in two views

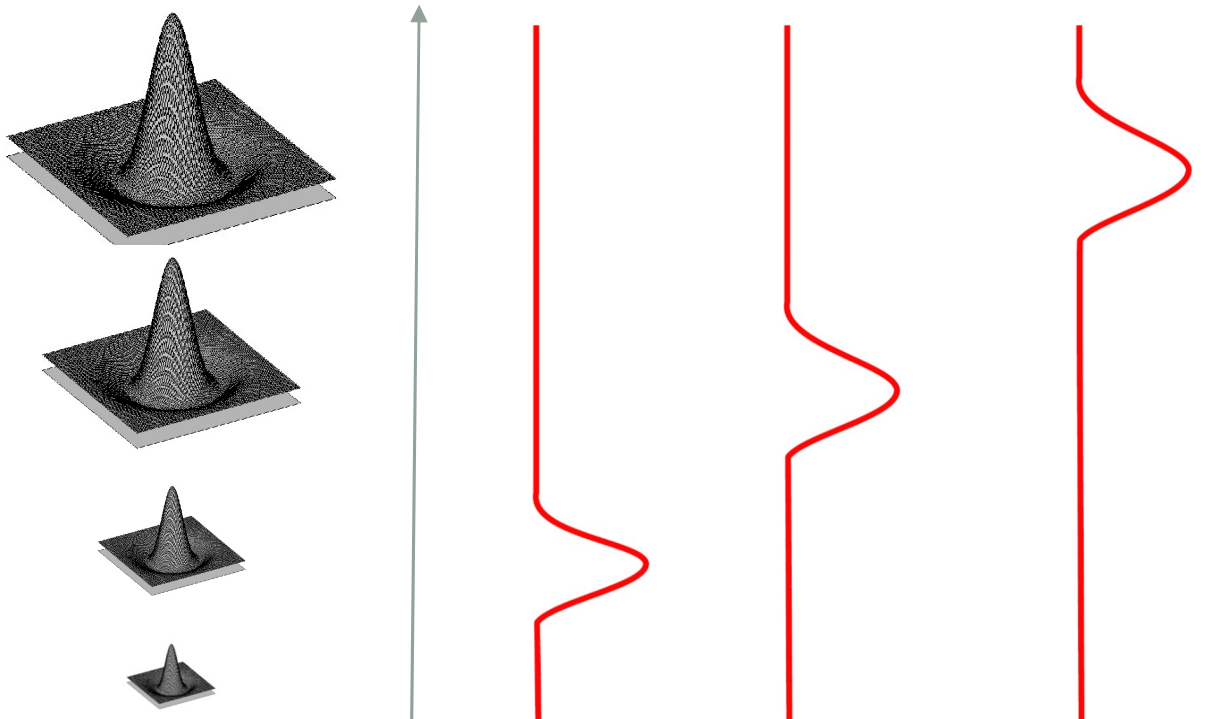
Local features detectors

- What are salient features that can be *detected* in multiple views?



Scale invariance detection

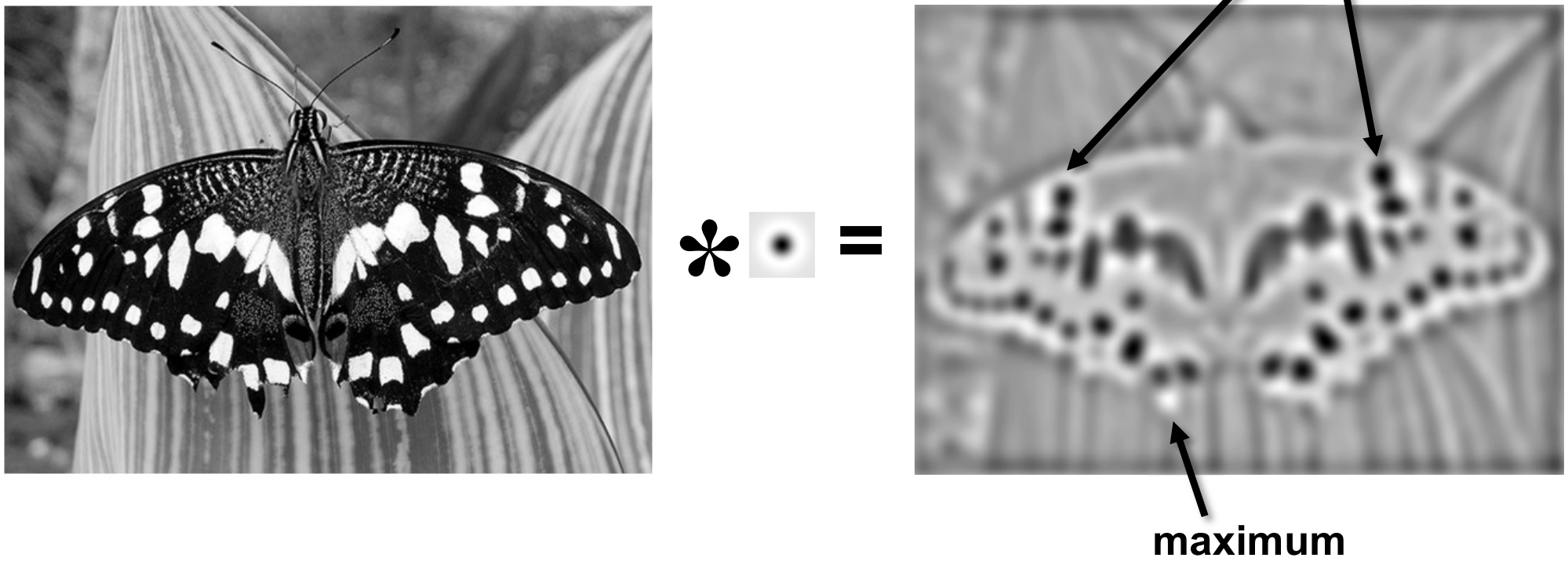
- Useful signature function
  - Laplacian-of-Gaussian = “blob” detector



$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$$

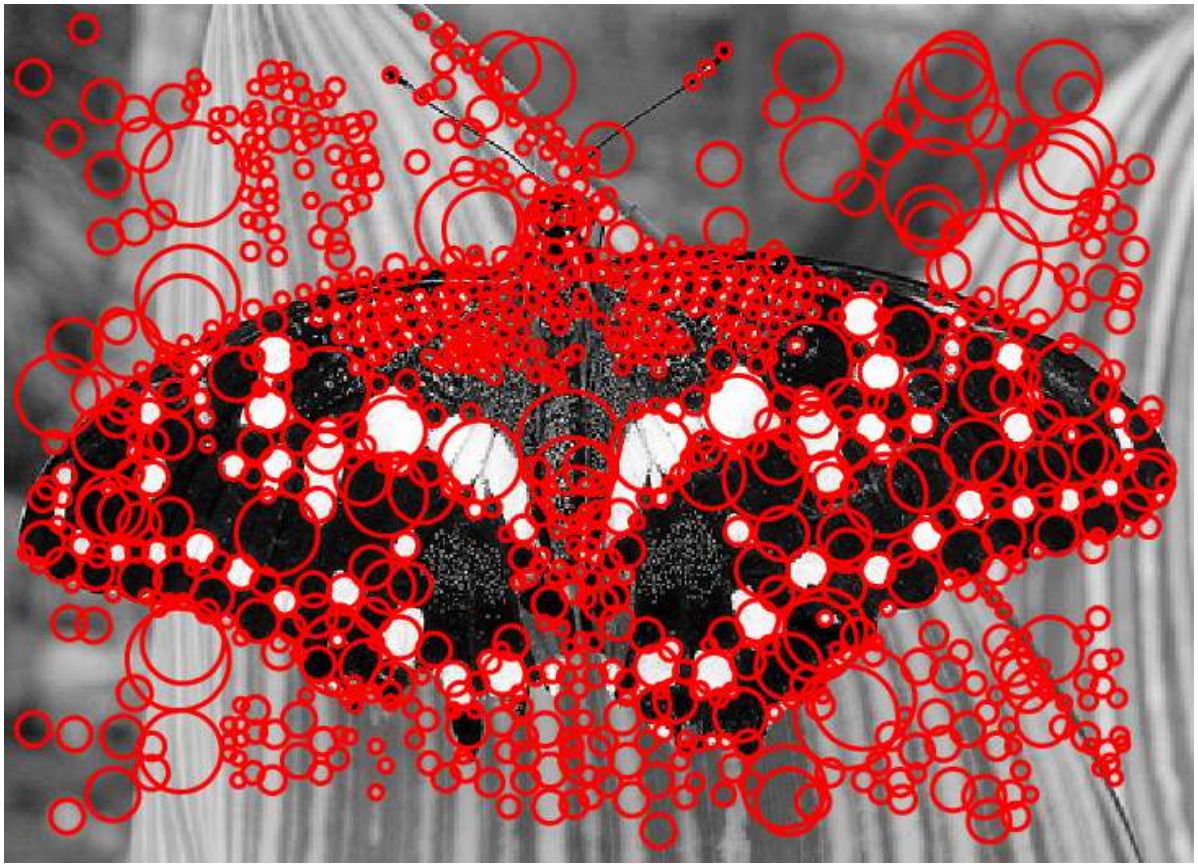
Laplacian of Gaussian

- "Blob" detector



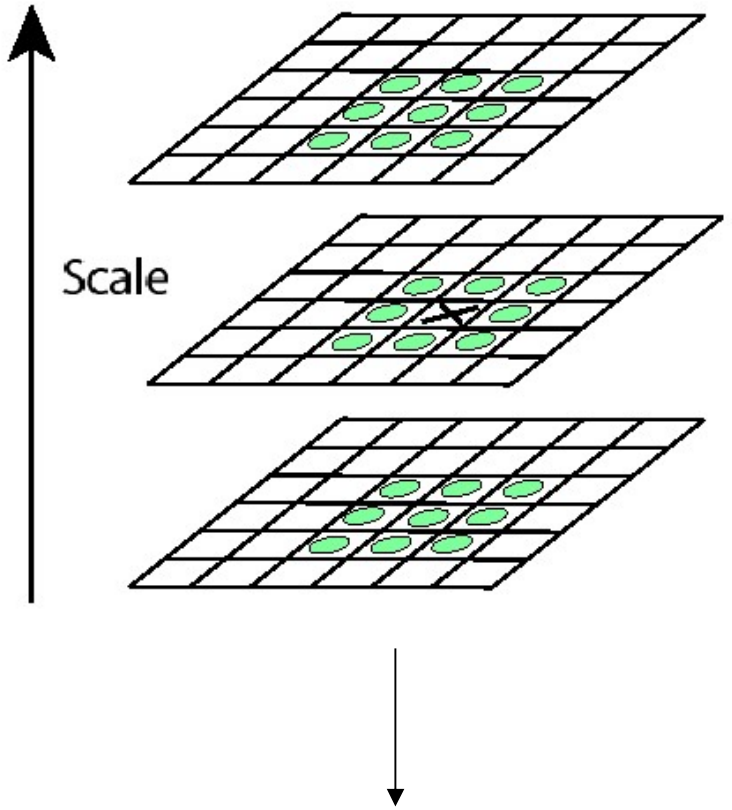
- Find maxima *and minima* of LoG operator in space and scale

Scale-space blob detector: Example



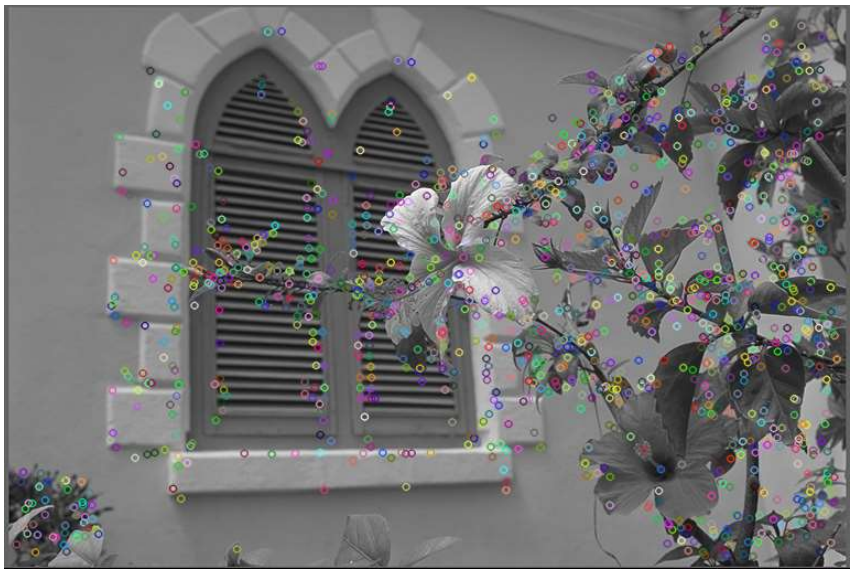
Scale invariance detection

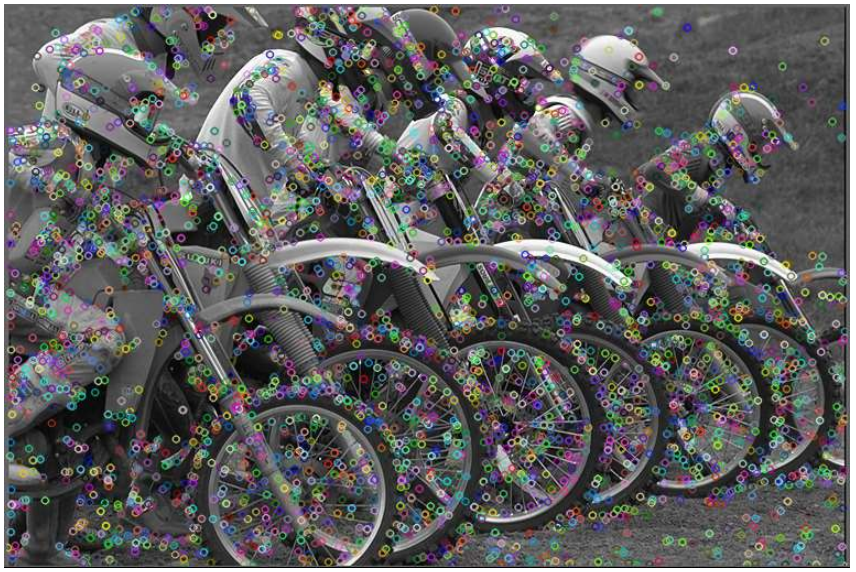
- LoG can be approximated by a Difference of two Gaussians (DoG) at different scales
- Detect maxima of DoG in the scale space volume
- Reject points with low contrast (threshold)
- Reject points that are localized along an edge



Candidate keypoints:
list of (x,y,σ)

SIFT / SURF keypoints



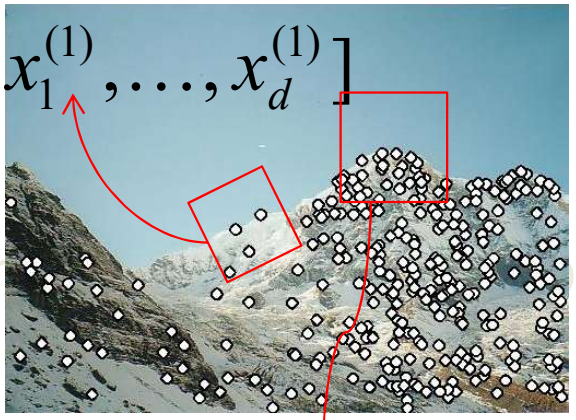


Local features: main components

1) Detection: Identify the interest points

2) Description: Extract vector feature descriptor surrounding each interest point.

$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$



$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$

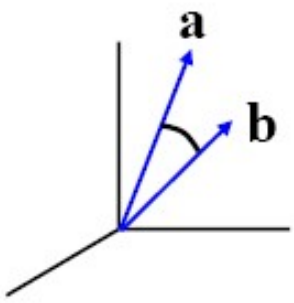
3) Matching: Determine correspondence between descriptors in two views

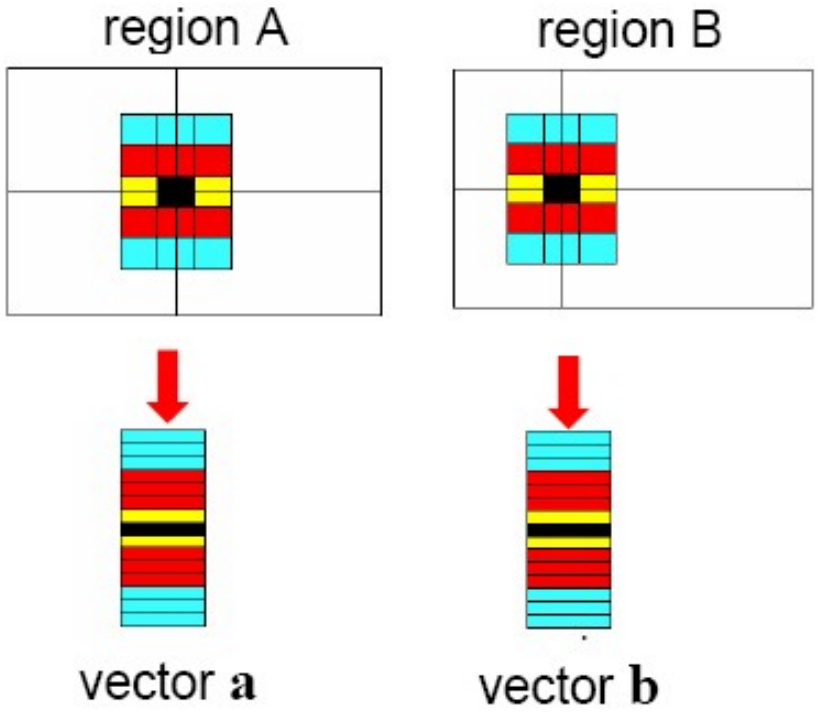
Local descriptors

- Simplest descriptor: list of intensities within a patch.

Write regions as vectors

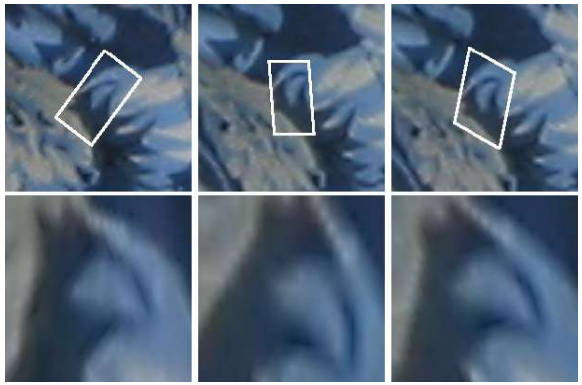
$$A \rightarrow \mathbf{a}, \quad B \rightarrow \mathbf{b}$$



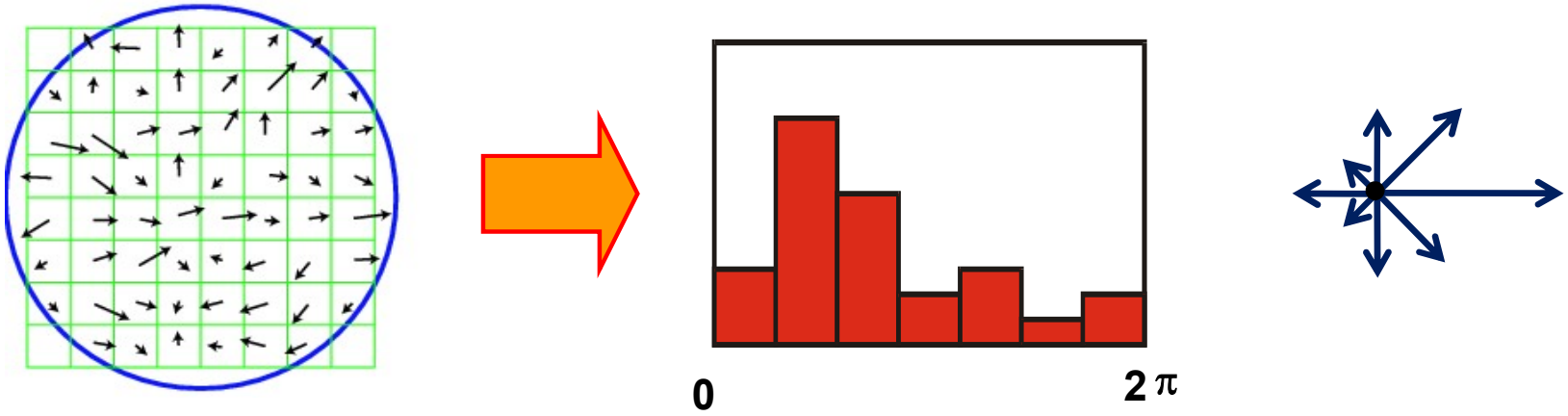


Local descriptors

- Disadvantage of patches as descriptors:
  - Small shifts can affect matching score a lot



- Solution: histograms



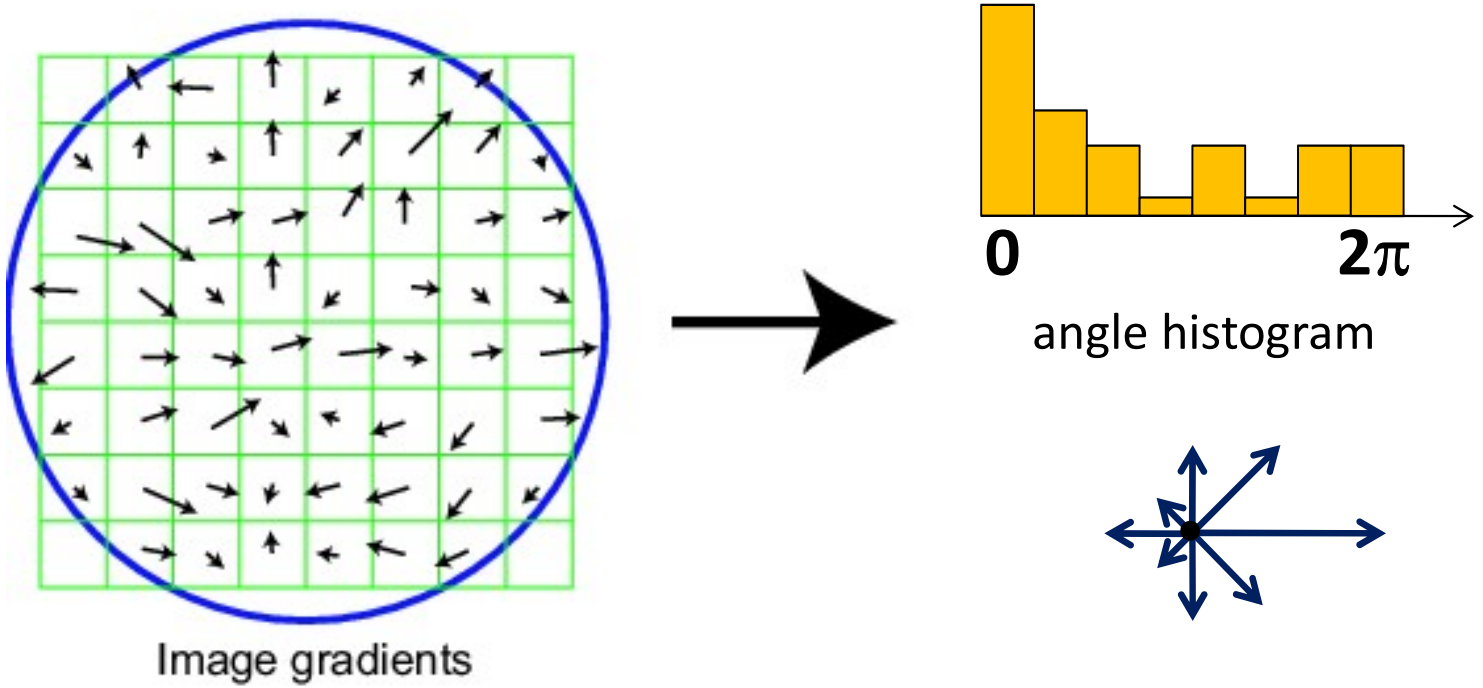
Local descriptors

- Histogram-based descriptors
  - Based on the histogram of oriented gradient
  - SIFT, SURF, GLOH and HOG
- Compact descriptors
  - Based on binary strings obtained comparing pairs of image intensities
  - BRIEF, ORB, BRISK and FREAK

SIFT descriptor

Basic idea:

- Take 16x16 square window around detected feature
- Compute edge orientation (angle of the gradient - 90°) for each pixel
- Throw out weak edges (threshold gradient magnitude)
- Create histogram of surviving edge orientations



Distinctive image features from scale-invariant keypoints. David G. Lowe. *IJCV* 60 (2), pp. 91-110, 2004.

SIFT descriptor

Full version

- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Compute an orientation histogram for each cell
- 16 cells * 8 orientations = 128 dimensional descriptor

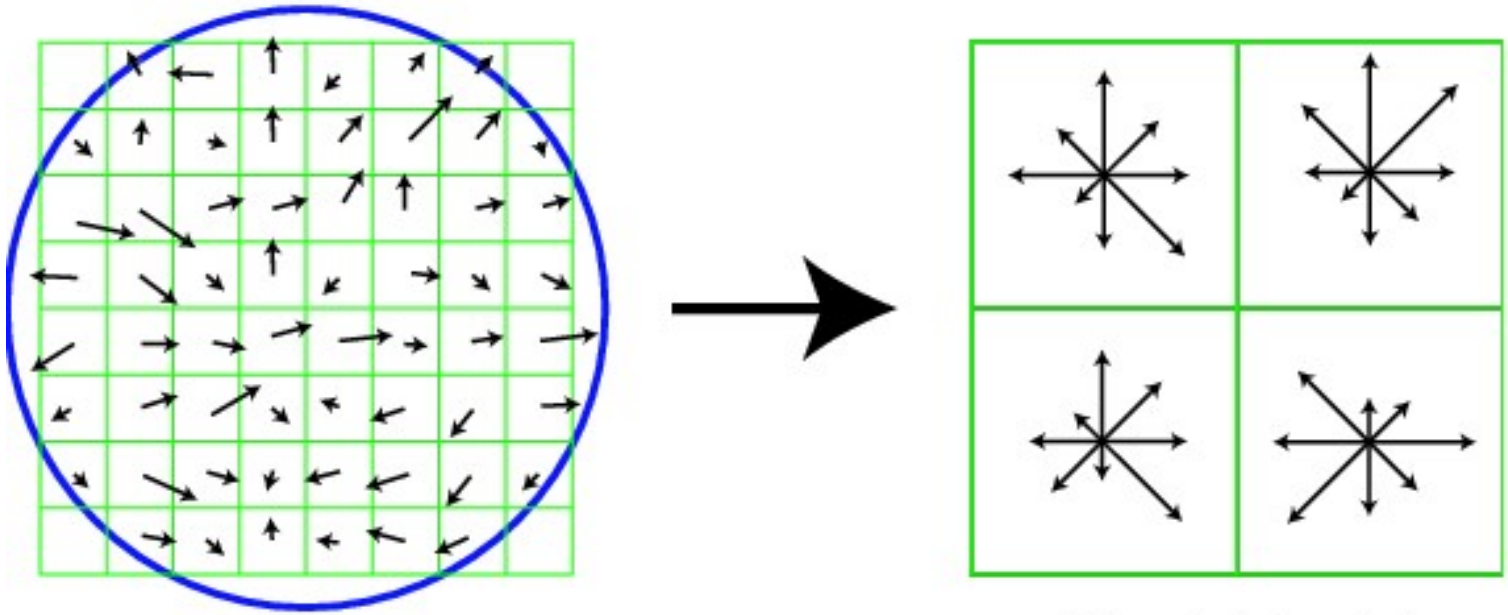


Image gradients

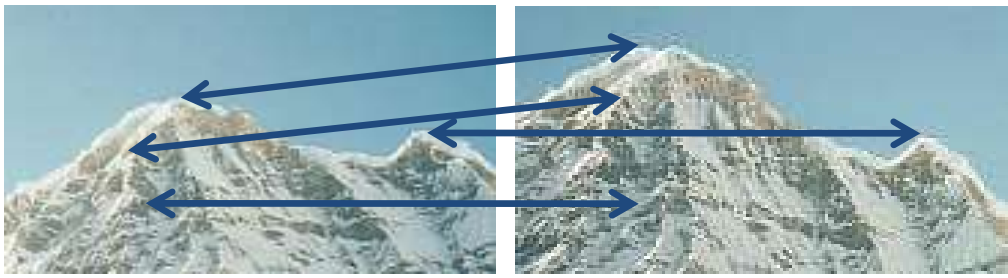
Keypoint descriptor

Local features: main components

1) Detection: Identify the interest points

2) Description: Extract vector feature descriptor surrounding each interest point.

3) Matching: Determine correspondence between descriptors in two views



Feature matching

Given a feature in I_1, how to find the best match in I_2?

1. Define distance function that compares two descriptors
2. Test all the features in I_2, find the one with min distance

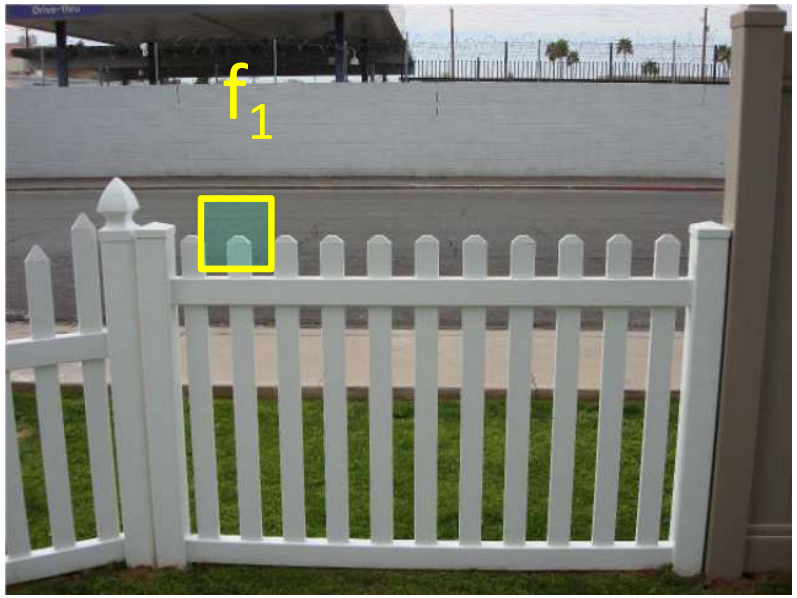
I_1

I_2

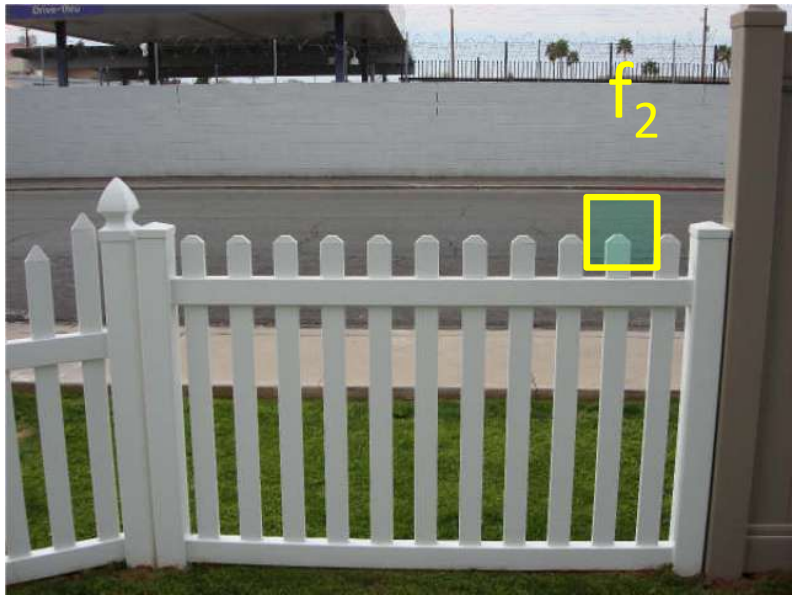
Feature matching

How to define the difference between two features f_1, f_2?

- Simple approach is SSD(f_1, f_2)
  - sum of square differences between entries of the two descriptors
  - can give good scores to very ambiguous (bad) matches



I_1

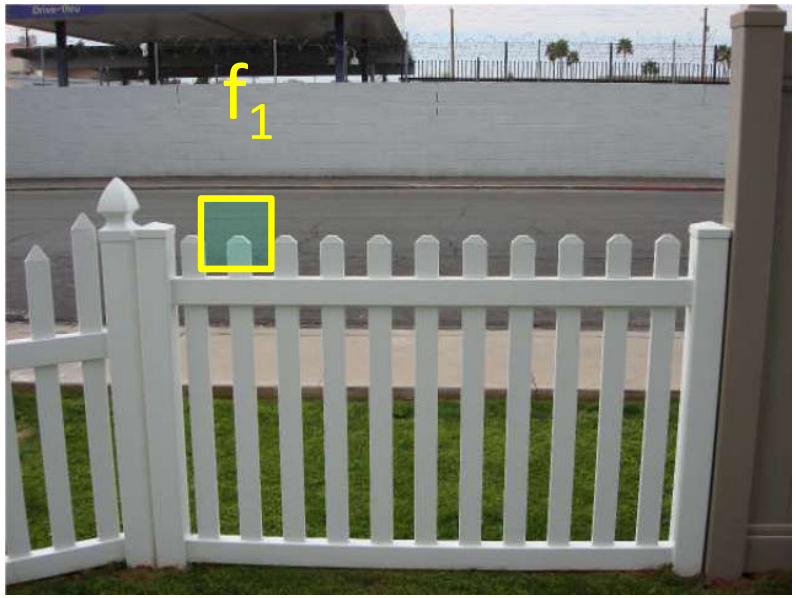


I_2

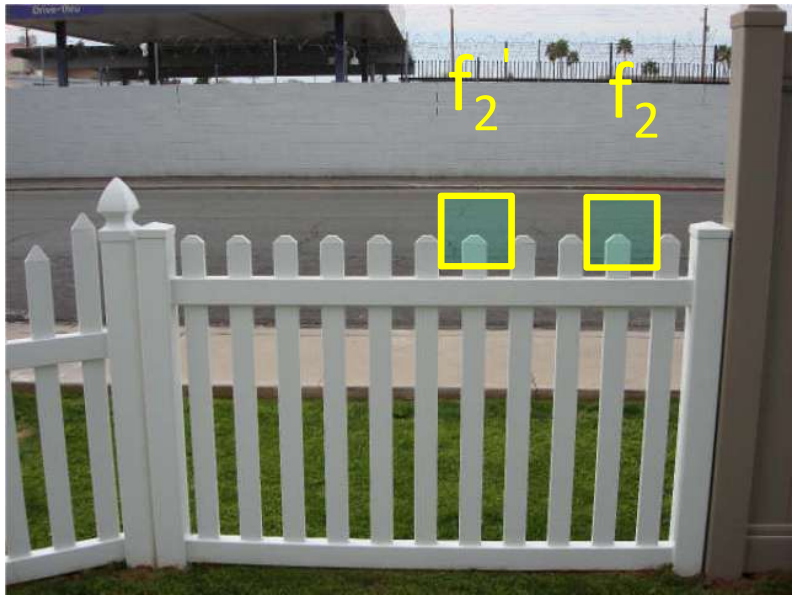
Feature matching

How to define the difference between two features f_1, f_2?

- Ratio distance = SSD(f_1, f_2) / SSD(f_1, f_2’)
  - f_2 is best SSD match to f_1 in I_2
  - f_2’ is 2nd best SSD match to f_1 in I_2
  - gives large values (~1) for ambiguous matches

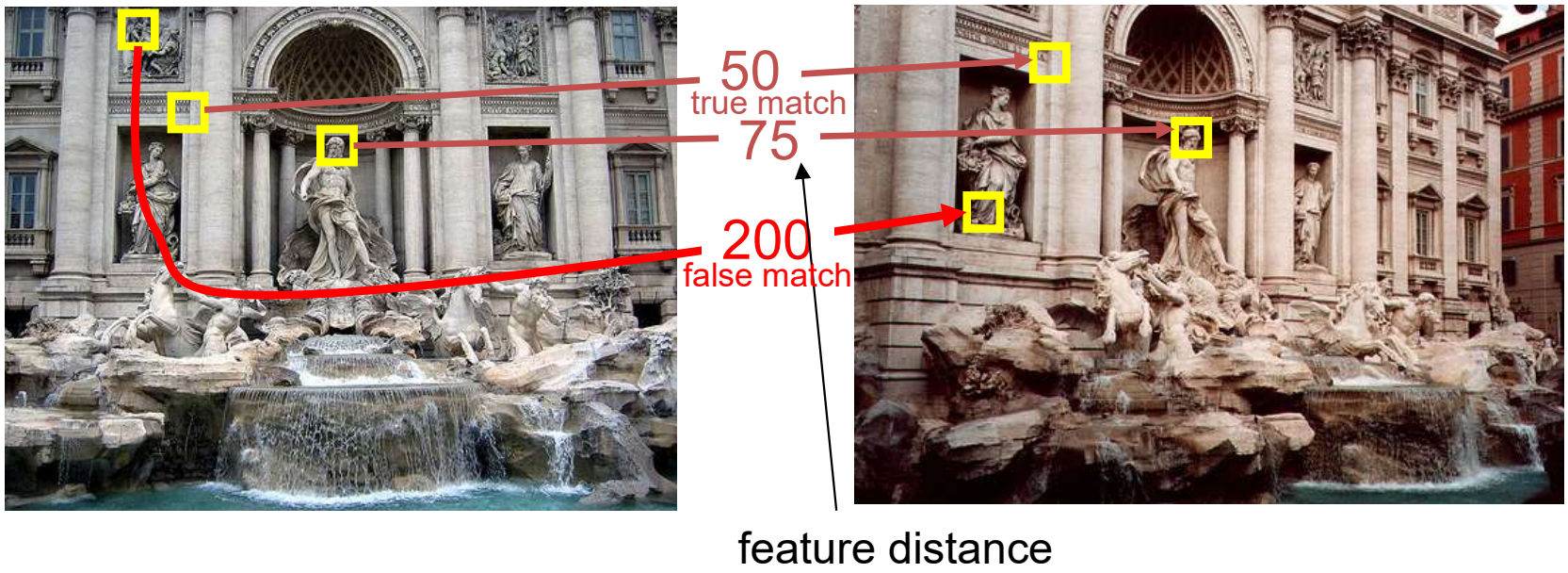


I_1



I_2

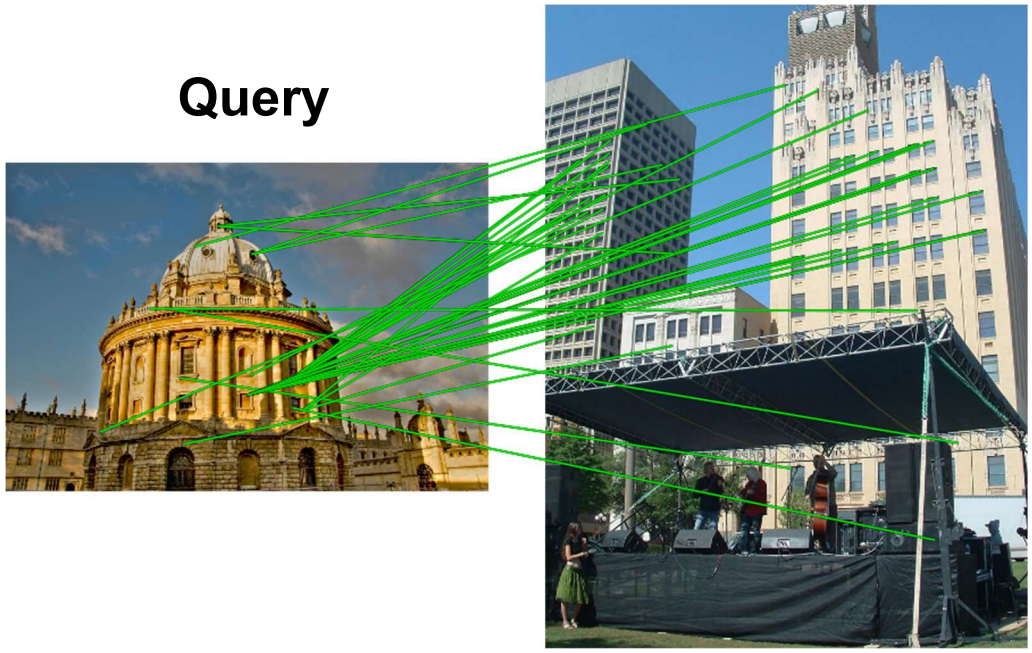
Feature matching



- Eliminate **bad matches**: throw out features with distance > threshold

Spatial Verification

DB image with high BoW similarity



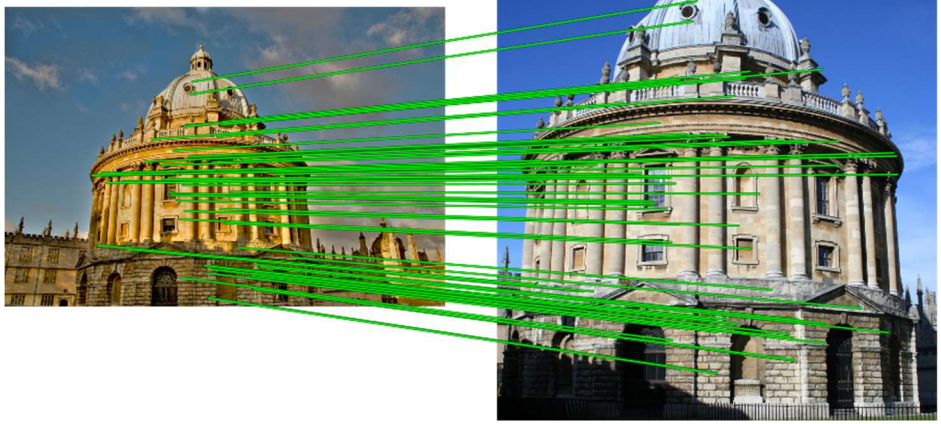
DB image with high BoW similarity

Both image pairs have many visual words in common.

Slide credit: Ondrej Chum

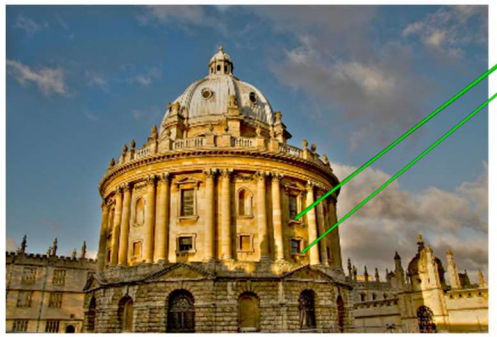
Spatial Verification

Query



DB image with high BoW similarity

Query

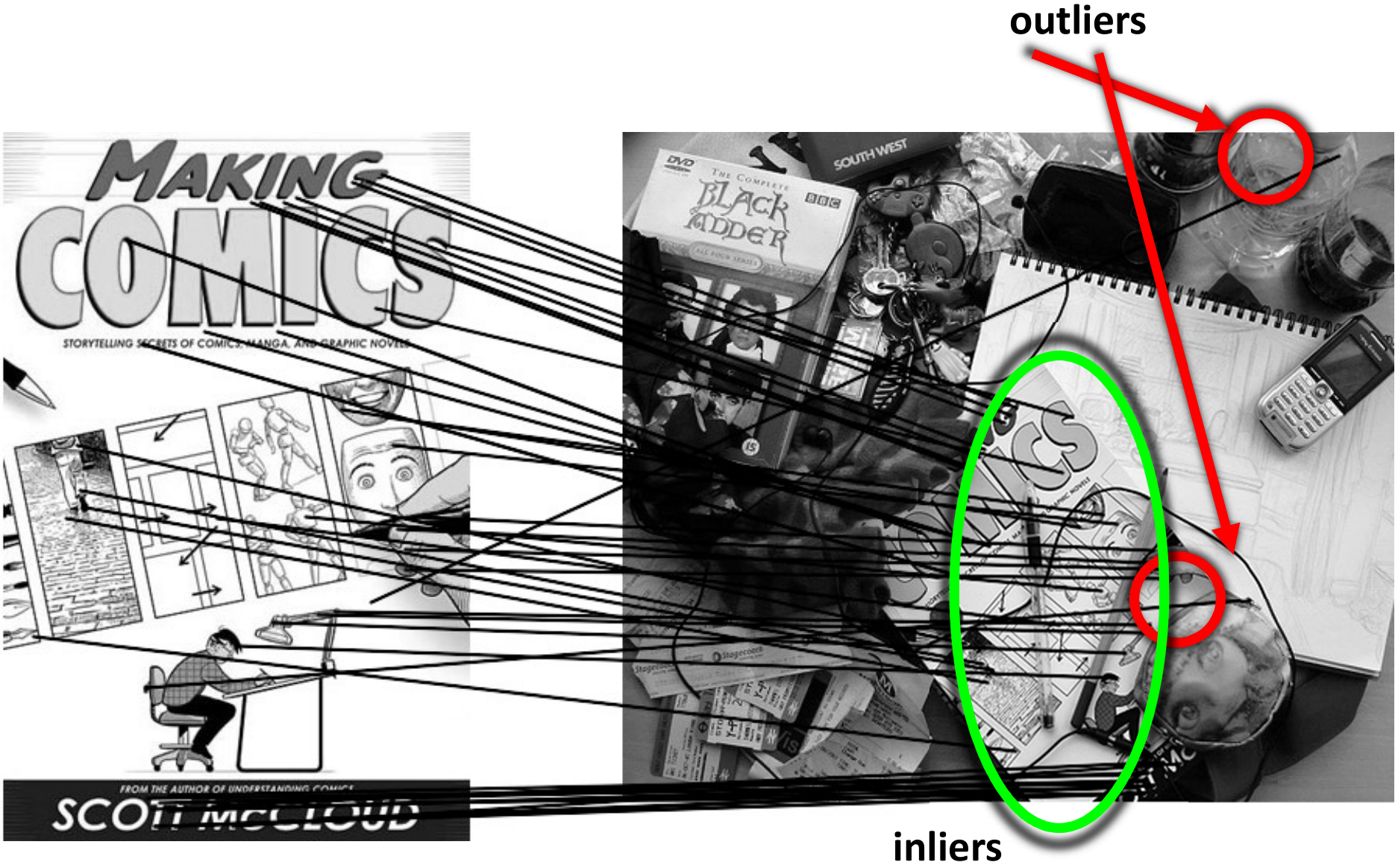


DB image with high BoW similarity

Only some of the matches are mutually consistent

Slide credit: Ondrej Chum

Outliers

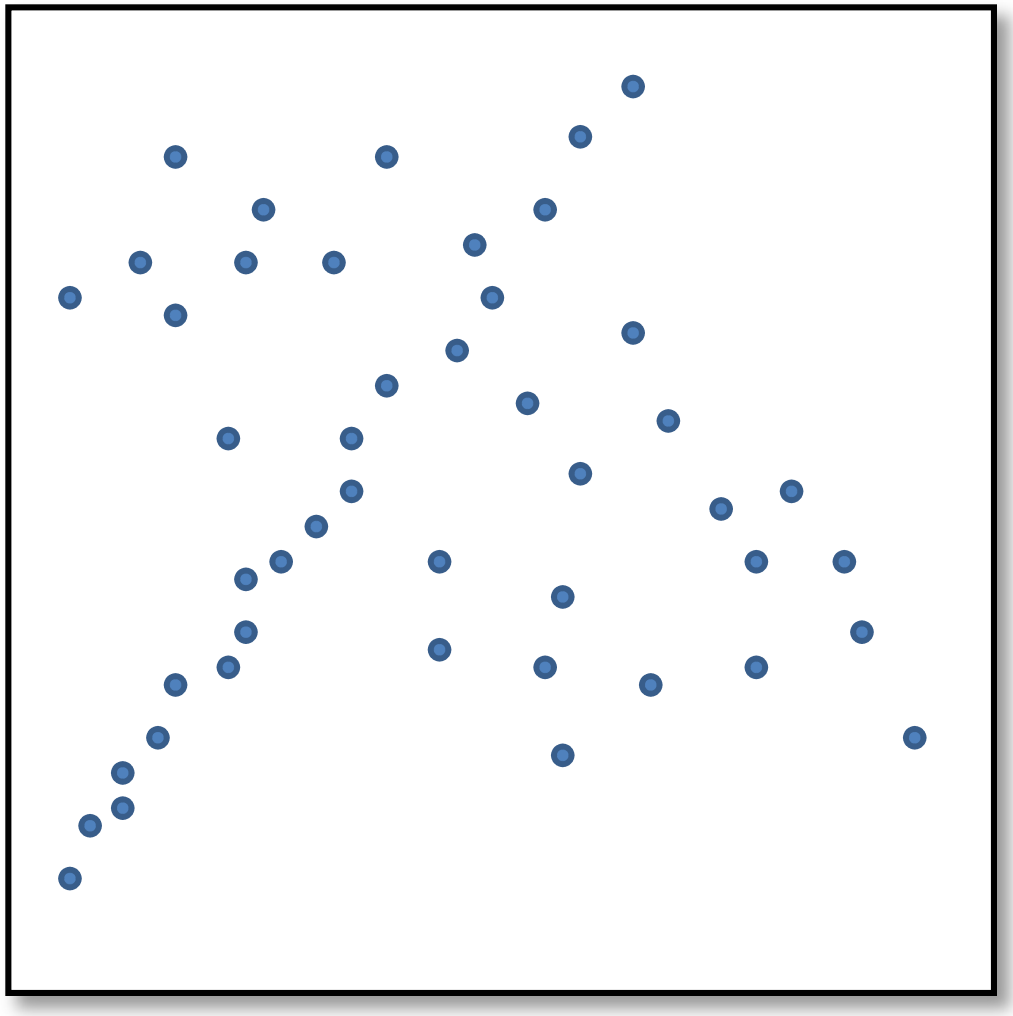

outliers
MAKING COMICS
STORYTELLING SECRETS OF COMICS, MANGA, AND GRAPHIC NOVELS
FROM THE AUTHOR OF UNDERSTANDING COMICS
SCOTT McCLOUD
SOUTH WEST
THE COMPLETE BLACK ADDER
BBC
inliers

Idea

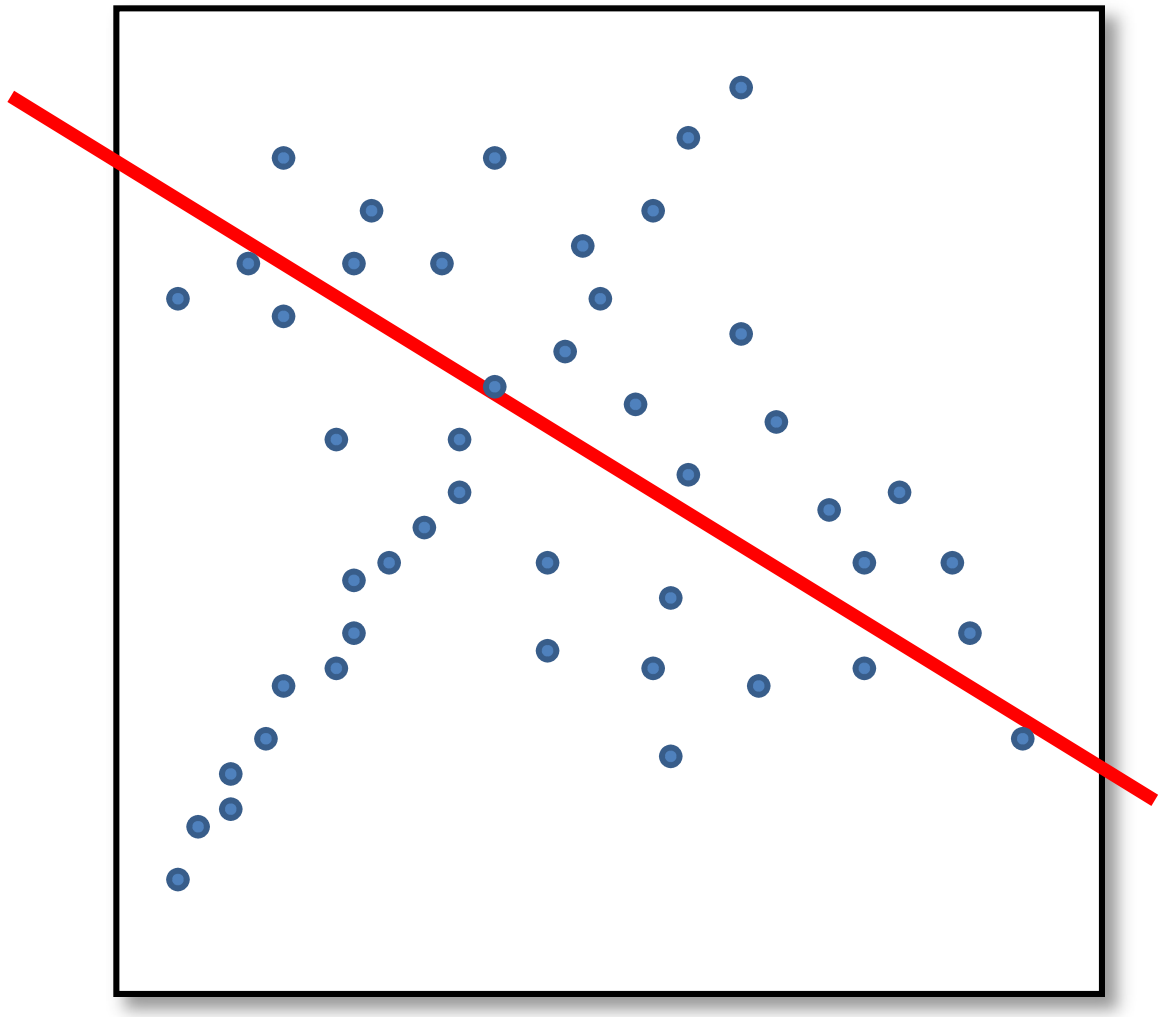
- Given a hypothesized line
- Count the number of points that "agree" with the line
  - "Agree" = within a small distance of the line
  - I.e., the **inliers** to that line

- For all possible lines, select the one with the largest number of inliers

Counting inliers

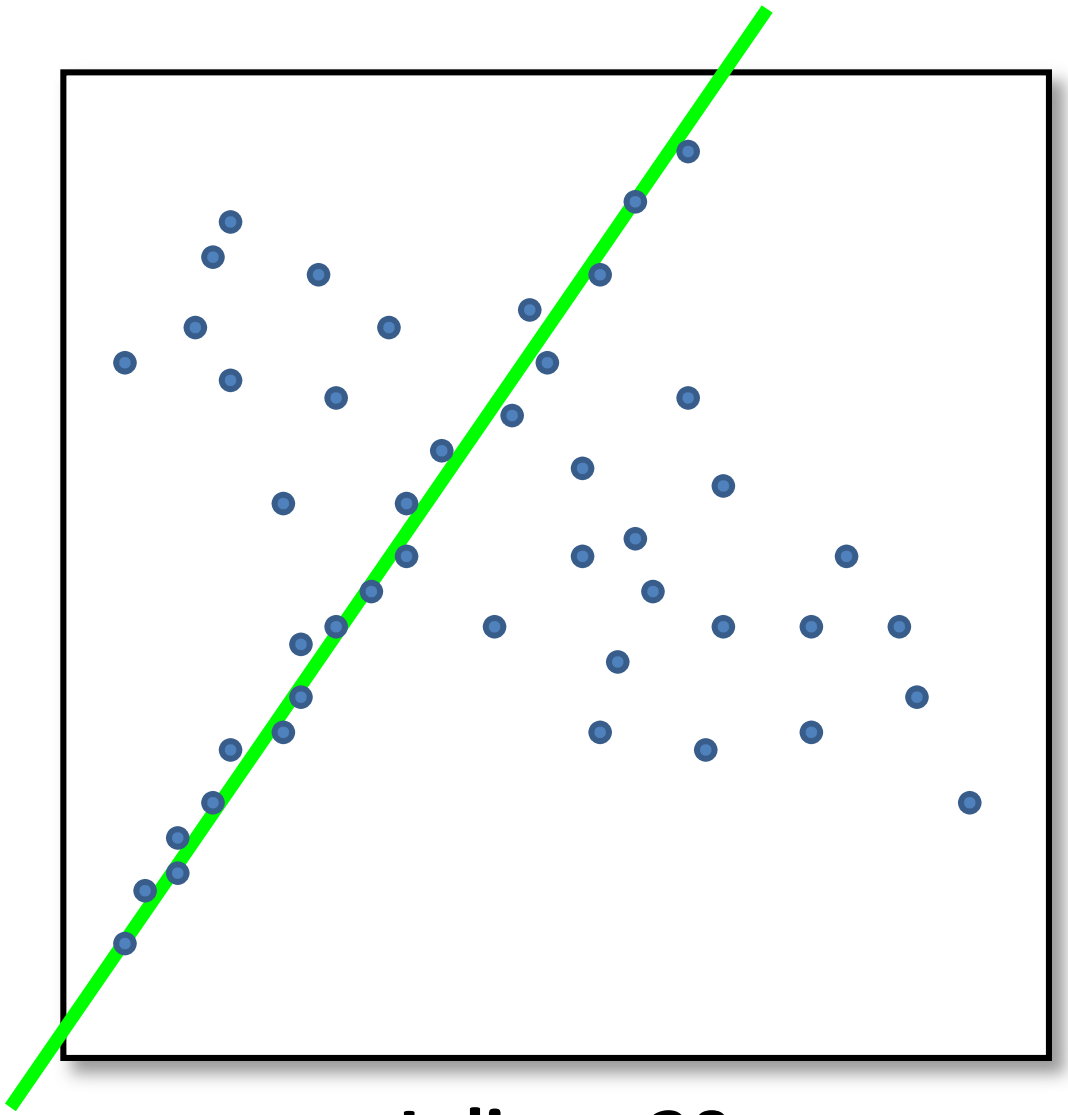


Counting inliers



Inliers: 3

Counting inliers

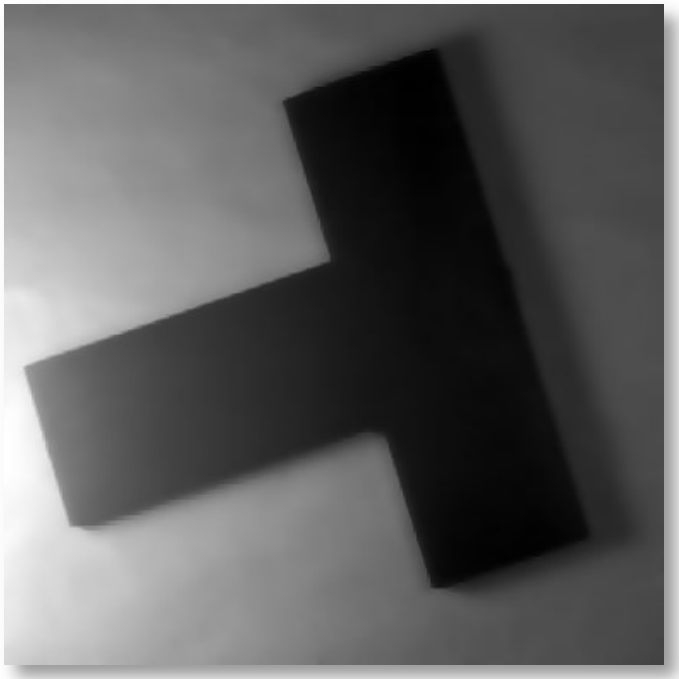


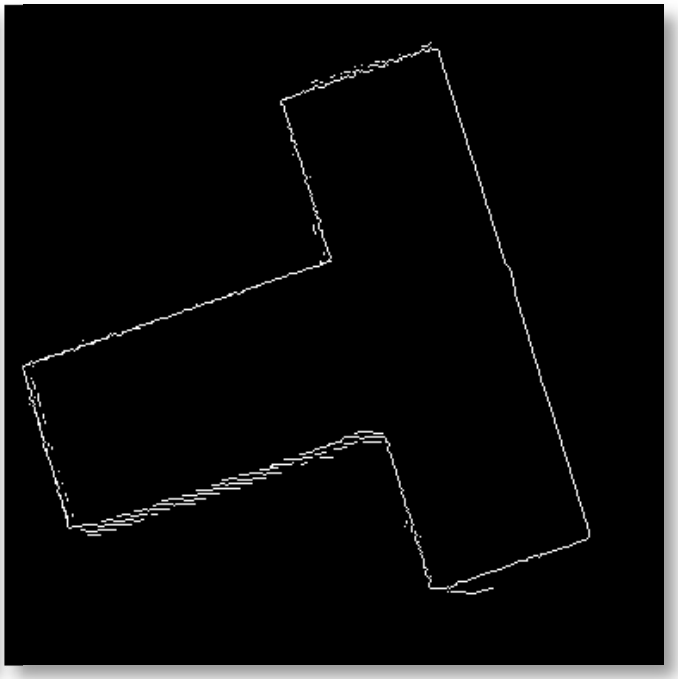
Inliers: 20

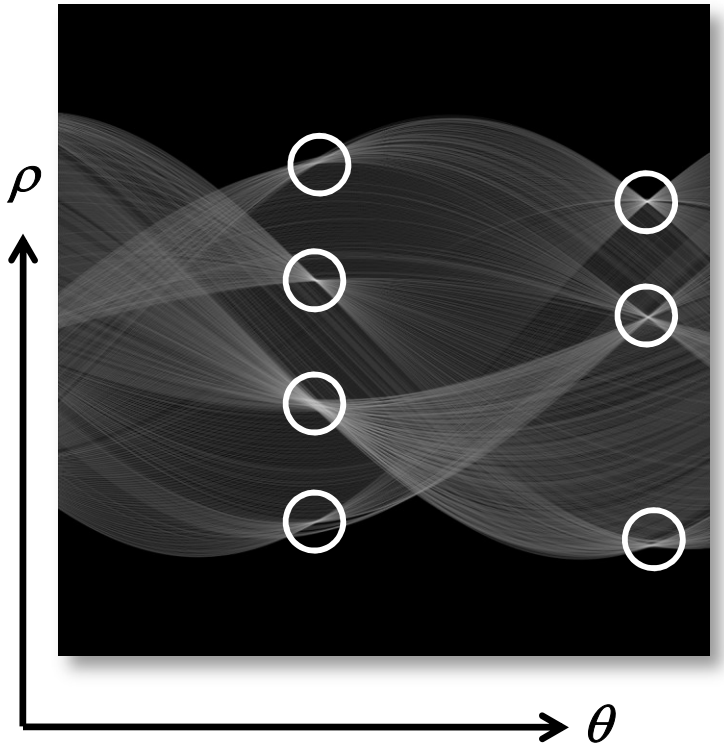
RANSAC

- RAndom SAmple Consensus
- An example of a “voting”-based fitting scheme
- Each hypothesis gets voted on by each data point, best hypothesis wins
- Idea:
  - All the inliers will agree with each other on the translation vector; the (hopefully small) number of outliers will (hopefully) disagree with each other

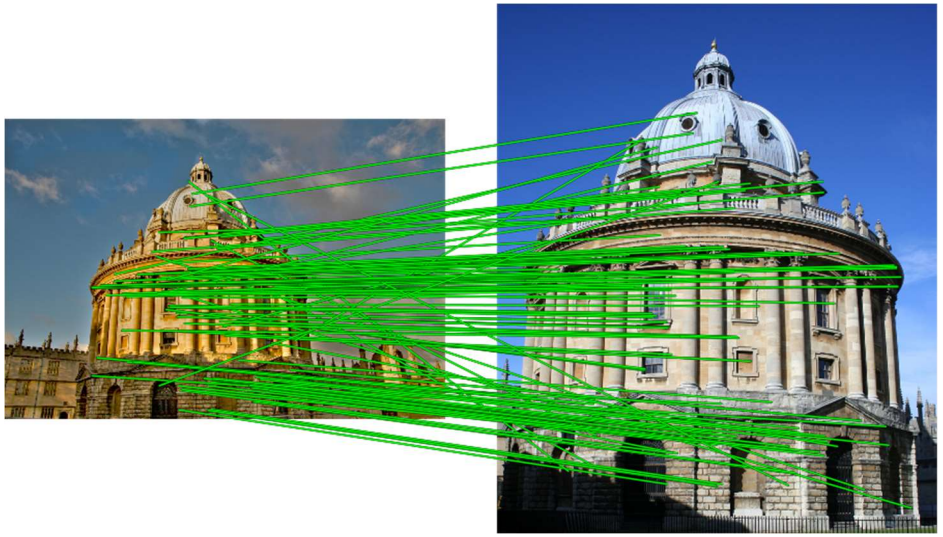
Hough transform

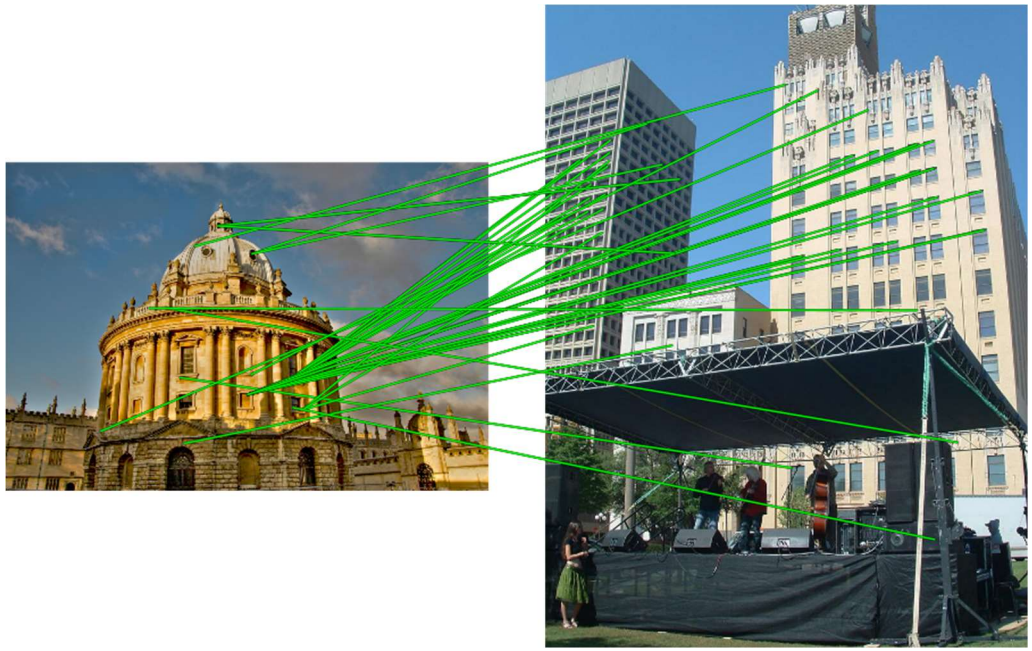


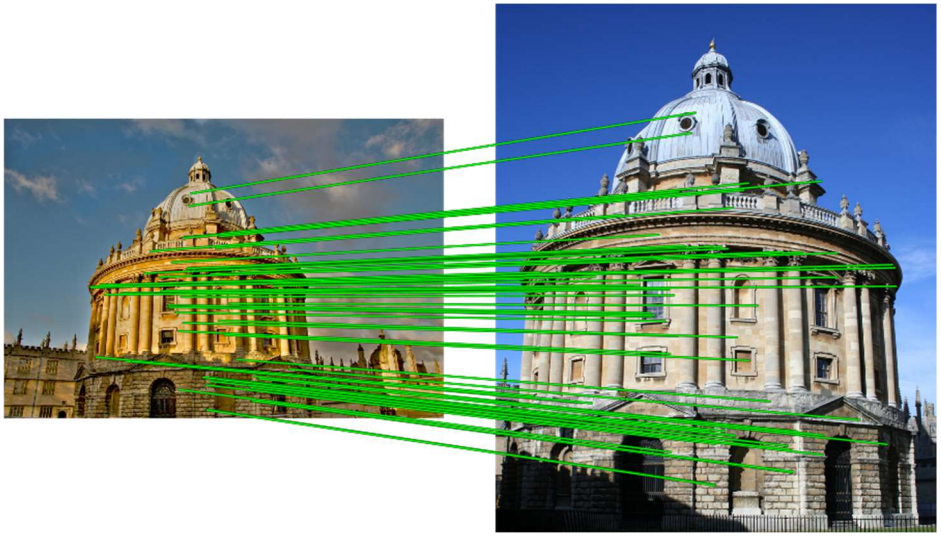




RANSAC verification







Why not use SIFT matching for everything?

- Works well for object *instances*

- Not great for generic object *categories*\

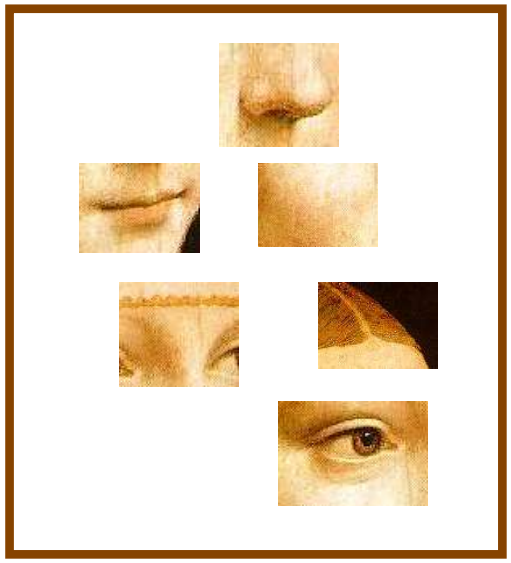
Bag of features

- First, take a bunch of images, extract features, and build up a “dictionary” or “visual vocabulary” – a list of common features

- Given a new image, extract features and build a histogram – for each feature, find the closest visual word in the dictionary

Bag of features: outline

1. Extract features



Bag of features: outline

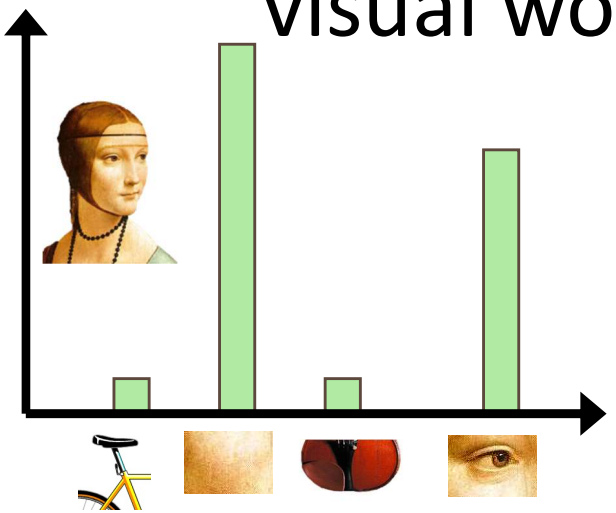
1. Extract features
2. Learn “visual vocabulary”

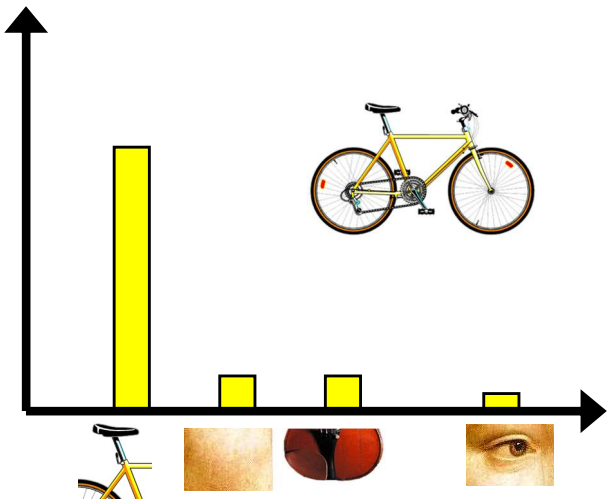
Bag of features: outline

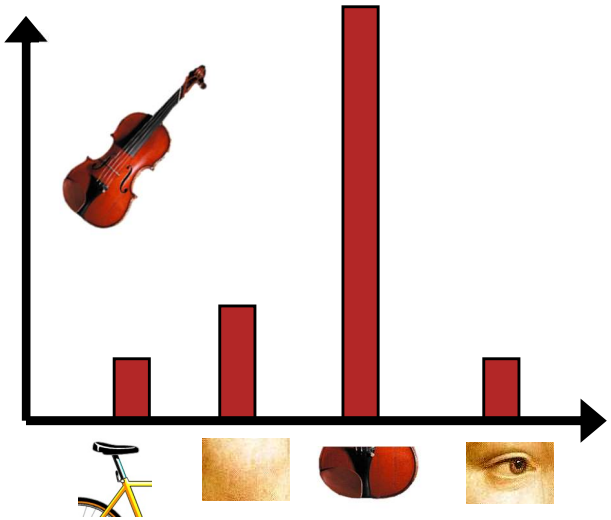
1. Extract features
2. Learn “visual vocabulary”
3. Quantize features using visual vocabulary

Bag of features: outline

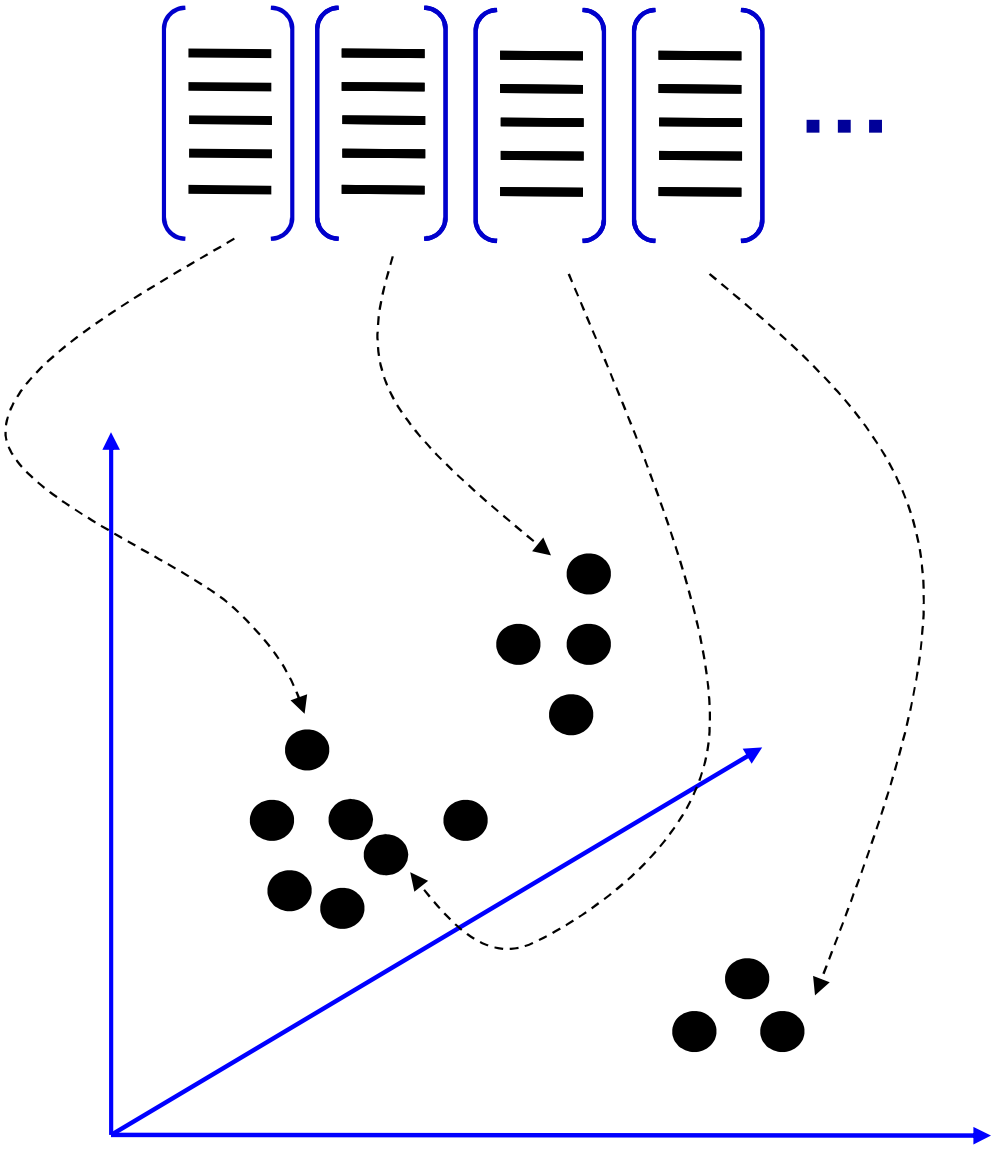
1. Extract features
2. Learn “visual vocabulary”
3. Quantize features using visual vocabulary
4. Represent images by frequencies of “visual words”



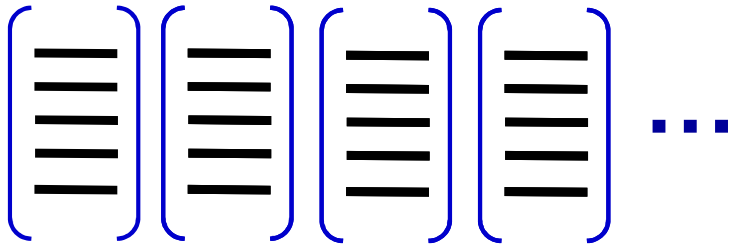


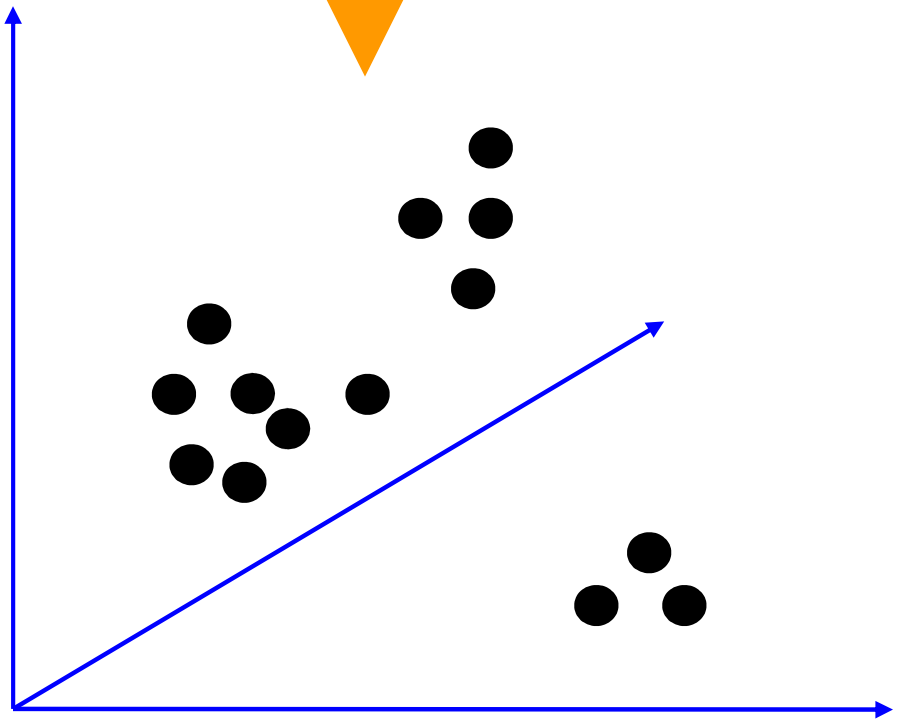


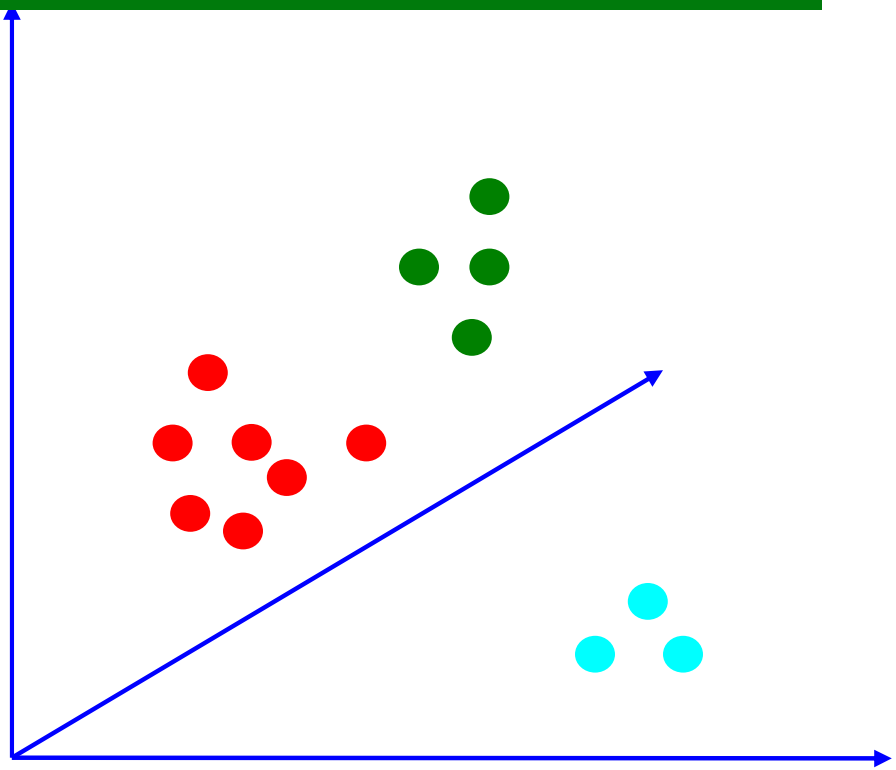
2. Learning the visual vocabulary



2. Learning the visual vocabulary



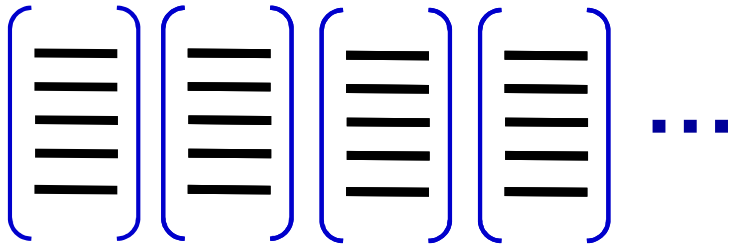


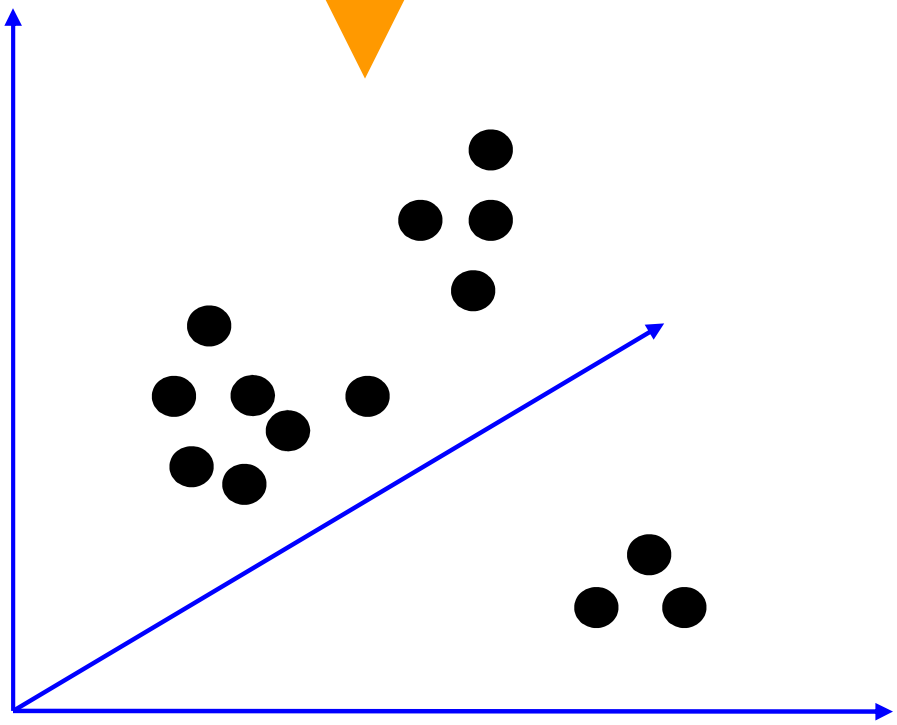


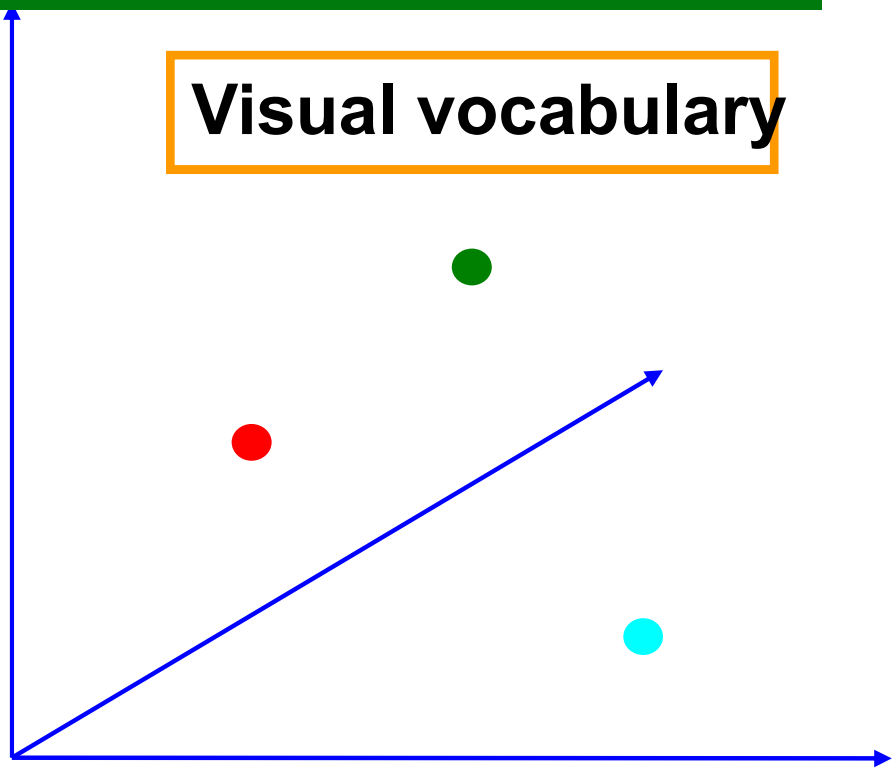
Clustering

Slide credit: Josef Sivic

2. Learning the visual vocabulary







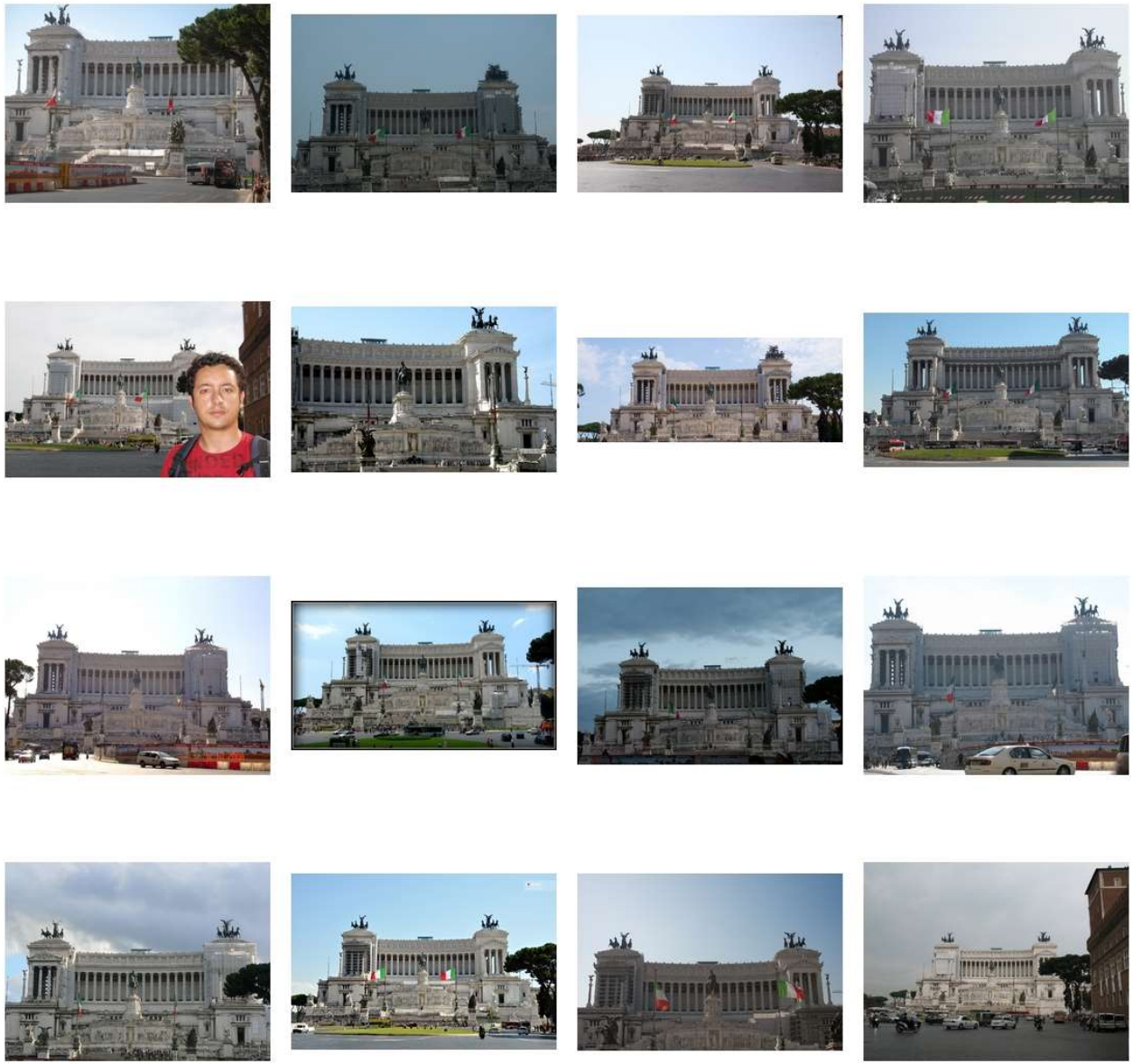
Clustering

Slide credit: Josef Sivic

Large-scale image matching

- How can we match 1,000,000 images to each other?
- Brute force approach: 500,000,000,000 pairs
  - won't scale
- Better approach: use bag-of-words technique to find *likely* matches
- For each image, find the top M scoring other images, do detailed SIFT matching with those

Example bag-of-words matches



Example bag-of-words matches