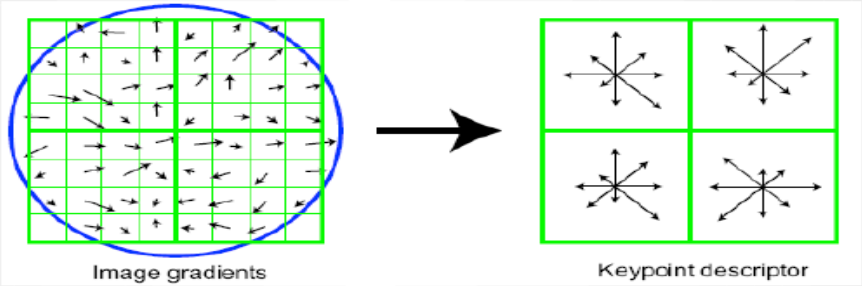
Visual Bag of Words

SIFT descriptor

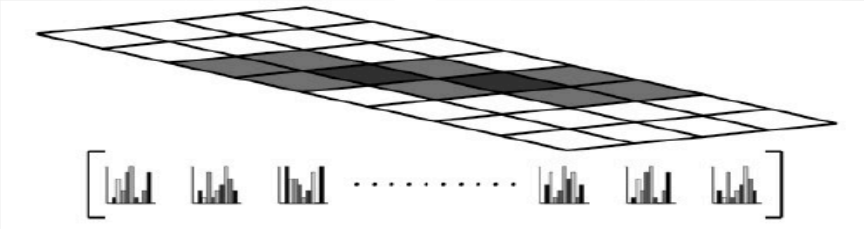
- The most popular gradient-based descriptor
- Typically used in combination with an interest point detector



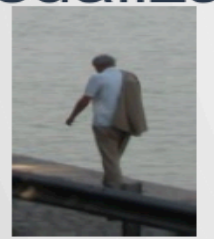
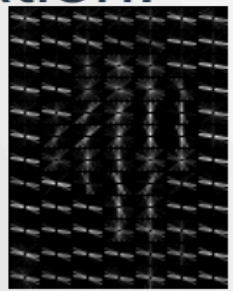
  - Region rescaled to a grid of 16x16 pixels
  - 4x4 regions = 16 histograms (concatenated)
  - Histograms: 8 orientation bins, gradients weighted by gradient magnitude
  - Final descriptor has 128 dimensions and is normalized to compensate for illumination differences

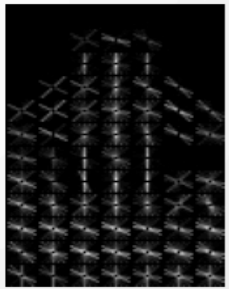
HOG Descriptor

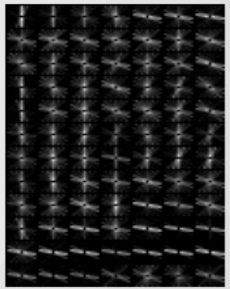
- Concatenation of Blocks



- Visualization:





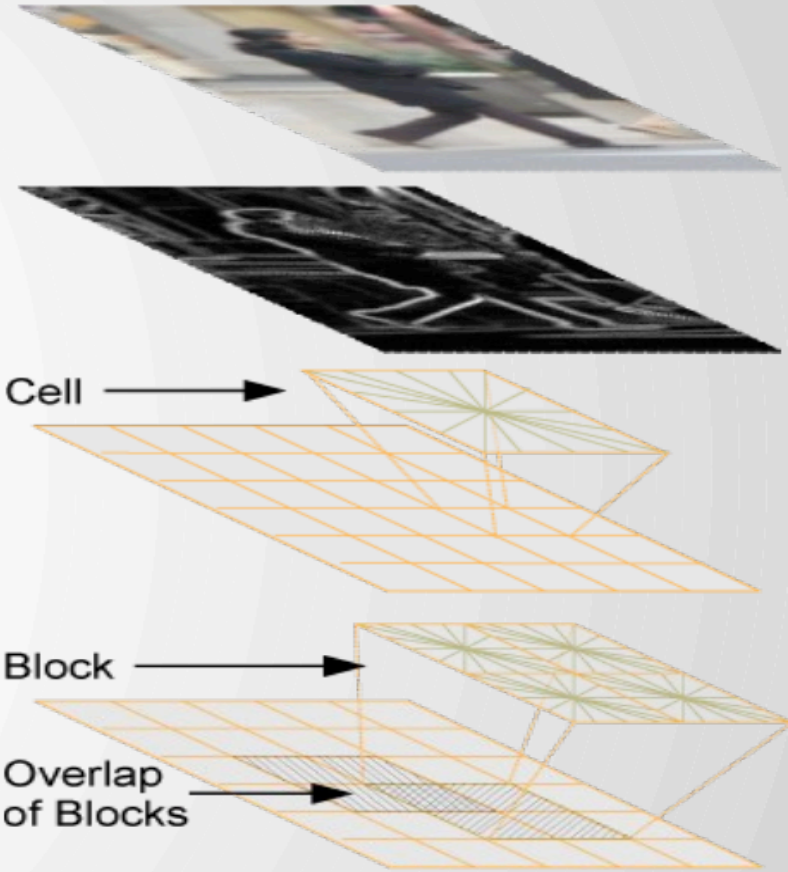
Descriptor

1. Compute gradients on an image region of 64x128 pixels

2. Compute histograms on ‘cells’ of typically 8x8 pixels (i.e. 8x16 cells)

3. Normalize histograms within overlapping blocks of cells (typically 2x2 cells, i.e. 7x15 blocks)

4. Concatenate histograms



Overview

- Origins and motivation
- Image representation
- Popular problems
  - Classification
  - Discriminative
    - Nearest Neighbor Classification.
  - Recognition (Topic discovery)
  - TinEye
- Video Google

Slide Thanks: Rob Fergus, Andrew Zisserman, Josef Sivic, Svetlana Lazebnik, misc.

trie data structure

Sign in

Web Videos Images News Maps More Search tools

About 2,24,000 results (0.28 seconds)

In computer science, a trie, also called digital **tree** and sometimes radix **tree** or prefix **tree** (as they can be searched by prefixes), is an ordered **tree data structure** that is used to store a dynamic set or associative array where the keys are usually strings.

Trie - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/**Trie**

Feedback

Trie - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/**Trie**

Jump to As a replacement for other **data structures** - In computer science, a trie, also called digital **tree** and sometimes radix **tree** or prefix **tree** (as they can be searched by prefixes), is an ordered **tree data structure** that is used to store a dynamic set or

en.wikipedia.org/wiki/**Trie**

Jump to As a replacement for other **data structures** - In computer science, a trie, also called digital **tree** and sometimes radix **tree** or prefix **tree** (as they can be searched by prefixes), is an ordered **tree data structure** that is used to store a dynamic set or associative array where the keys are usually strings.

Radix tree - Suffix tree - Directed acyclic word graph - Burstsort

Trie | (Insert and Search) - GeeksforGeeks

www.geeksforgeeks.org/**trie**-insert-and-search/

Trie is an efficient information retrieval **data structure**. Using **trie**, search complexities can be brought to optimal limit (key length). If we store keys in binary search ...

Using Tries – topcoder

https://community.topcoder.com/tc?module=Static&d1...using**Tries**

The word trie is an infix of the word "retrieval" because the trie can find a single word in a dictionary with only a prefix of the word. The main idea of the trie data structure consists of the following parts: The trie is a **tree** where each vertex represents a single word or a prefix.

The Trie Data Structure: Examples in Java | Toptal

www.toptal.com/java/the-**trie**-a-neglected-**data-structure**

See how an oft-neglected **data structure**, the **trie**, shines in application domains with specific features, like word games.

Trie

https://www.cs.bu.edu/teaching/c/tree/**trie**/

We use a **trie** to store pieces of data that have a key (used to identify the .... put the **data structure** in its own module by producing the source files **trie**.h and **trie**.c .

Tries - Department of Computer and Information Science

is much faster than set , but is it a bit faster than a hash table.

- The **set <string>** and the hash tables can only find in a dictionary words that match exactly with the single word that we are finding; the trie allow us to find words that have a single character different, a prefix in common, a character missing, etc.

The tries can be useful in TopCoder problems, but also have a great amount of applications in software engineering. For example, consider a web browser. Do you know how the web browser can auto complete your text or show you many possibilities of the text that you could be writing? Yes, with the trie you can do it very fast. Do you know how an orthographic corrector can check that every word that you type is in a dictionary? Again a trie. You can also use a trie for suggested ***corrections*** of the words that are present in the text but not in the dictionary.

What is a Tree?

You may read about how wonderful the tries are, but maybe you don't know yet what the tries are and why the tries have this name. The word trie is an infix of the word "re**trie**val" because the trie can find a single word in a dictionary with only a prefix of the word. The main idea of the trie data structure consists of the following parts:

- The trie is a tree where each vertex represents a single word or a prefix.
- The root represents an empty string (""), the vertexes that are direct sons of the root represent prefixes of length 1, the vertexes that are 2 edges of distance from the root represent prefixes of length 2, the vertexes that are 3 edges of distance from the root represent prefixes of length 3 and

Motivation

- How to “summarize” a document like Google ?

Motivation

- How to “summarize” a document like Google ?
- How do we use “words” in a document to summarize, based on the search word, and assign importance to sentences ?

Motivation

- How to “summarize” a document like Google ?
- How do we use “words” in a document to summarize, based on the search word, and assign importance to sentences ?
- Word frequency, unimportant words, document as histogram of words ?

Motivation

- What information is lost ?

Motivation

- What information is lost ?
  - Spatial.

Motivation

- What information is lost ?
  - Spatial.
  - Sentence construction (syntax/grammar).

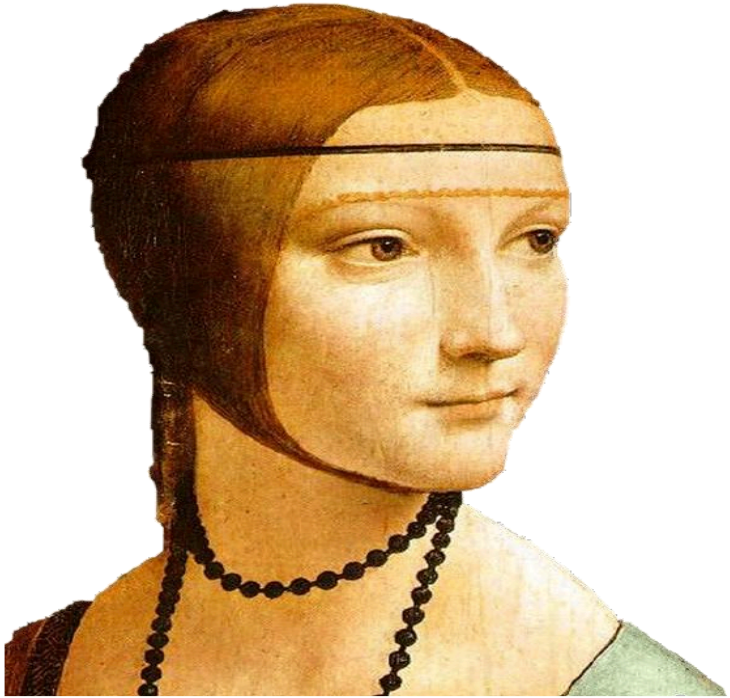
Motivation

- What information is lost ?
  - Spatial.
  - Sentence construction (syntax/grammar).
  - ...

Motivation

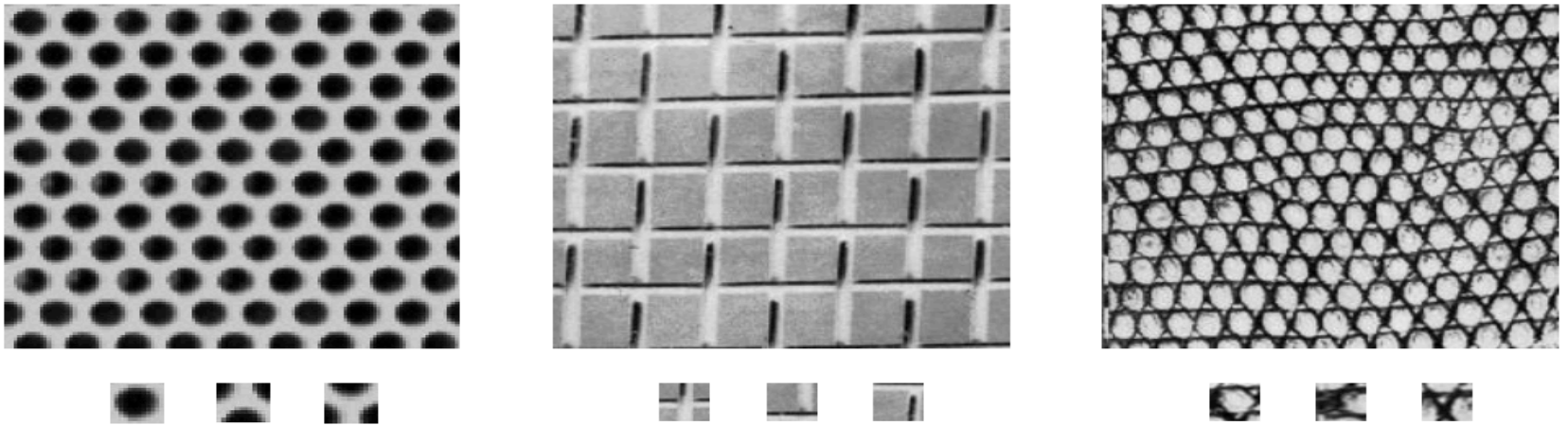
- Related problems ?
  - Comparing two document images.
    - Are these two documents on the same topic ?
    - Are these two documents on “similar” topics ?
      - What do you mean by “similarity” ?
  - Searching for a piece of text or paragraph.
    - Giving collection of words in “quotes”.
  - How can we do this with images ?

Bag-of-features models



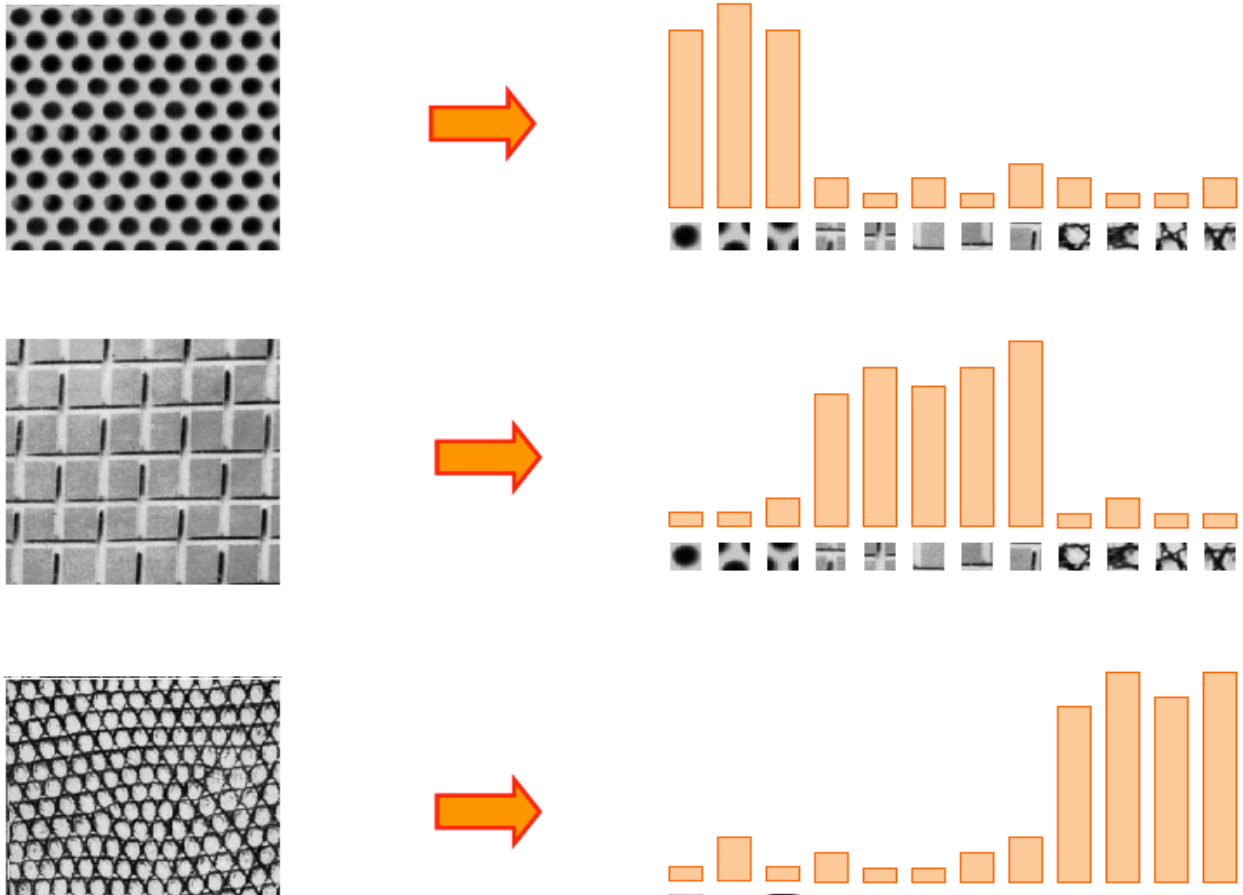
Origin 1: Texture recognition

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



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

Origin 1: Texture recognition



Functions for comparing histograms

- L1 distance

$$D(h_1, h_2) = \sum_{i=1}^{N} | h_1(i) - h_2(i) |$$

- χ^2 distance

$$D(h_1, h_2) = \sum_{i=1}^{N} \frac{(h_1(i) - h_2(i))^2}{h_1(i) + h_2(i)}$$

- Quadratic distance (*cross-bin*)

$$D(h_1, h_2) = \sum_{i,j} A_{ij} (h_1(i) - h_2(j))^2$$

Jan Puzicha, Yossi Rubner, Carlo Tomasi, Joachim M. Buhmann: Empirical Evaluation of Dissimilarity Measures for Color and Texture. ICCV 1999

Origin 2: Bag-of-words models

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

Origin 2: Bag-of-words models

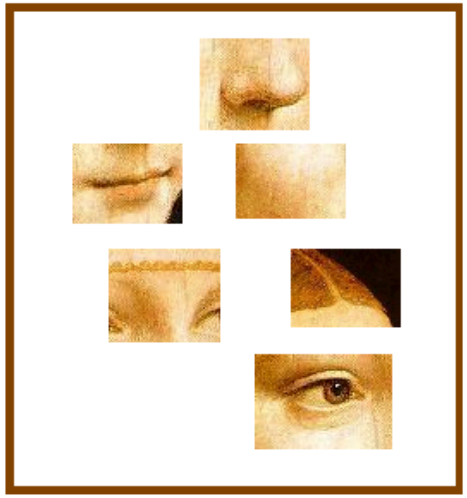
- Orderless document representation: frequencies of words from a dictionary Salton & McGill (1983)

Tag Clouds


slide
Slide
comments
just
text
Person
see
colleague
within
check
visible
View
friend
new
design
PDF
time
makes
linked
problem
click
change
size
layout
comment
presentation
Master
PowerPoint
slides
font
logo
box
add
one
need
every
e-book
master
choose
sends
messages
smaller
answer
contest
top
go
use
apply
made
Just
theme
business
MS
increase
screen
people
colour
know
second
feature
reflect
PPT
going
open
two
New
Review
create
carry
want
might
tab
book
easy
now
adds
using
Comment
particular
Now
make
also
pixels
audience
well
designer
delete
Automatically
India
wonderful
right
first
view
title
blue
key
Click
feedback
last
point
presentations
applied
layouts
option
element
left
Next
best
win
Go
Manvendra
back
small
type
list

Bags of features for image classification

1. Extract features



Bags of features for image classification

1. Extract features
2. Learn “visual vocabulary”

Difference between features & words

Words

- Dictionary/Vocab.
- Meaning.
- Finite/Precise.
- Language known.

Features

- Dictionary/Vocab ?
- Meaning ?
- Finite/Precise ?
- What Language ?

Difference between features & words

Words

- Dictionary/Vocab.
- Meaning.
- Finite/Precise.
- Language known.

Features

- Dictionary/Vocab ?
- Meaning ?
- Finite/Precise ?
- What Language ?

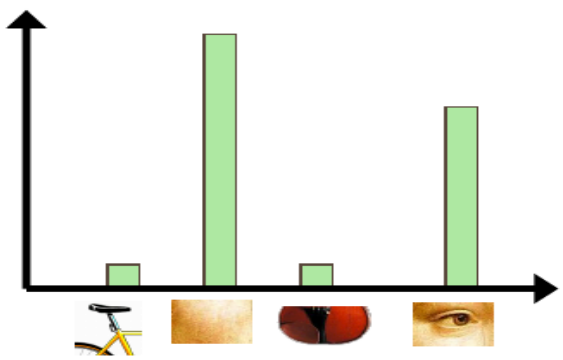
How do we get “visual” words ?

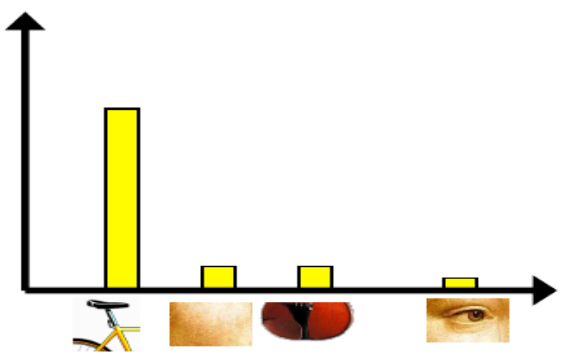
Bags of features for image classification

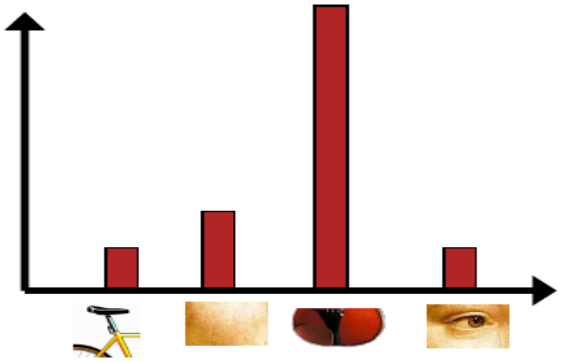
1. Extract features
2. Learn “visual vocabulary”
3. Quantize features using visual vocabulary

Bags of features for image classification

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

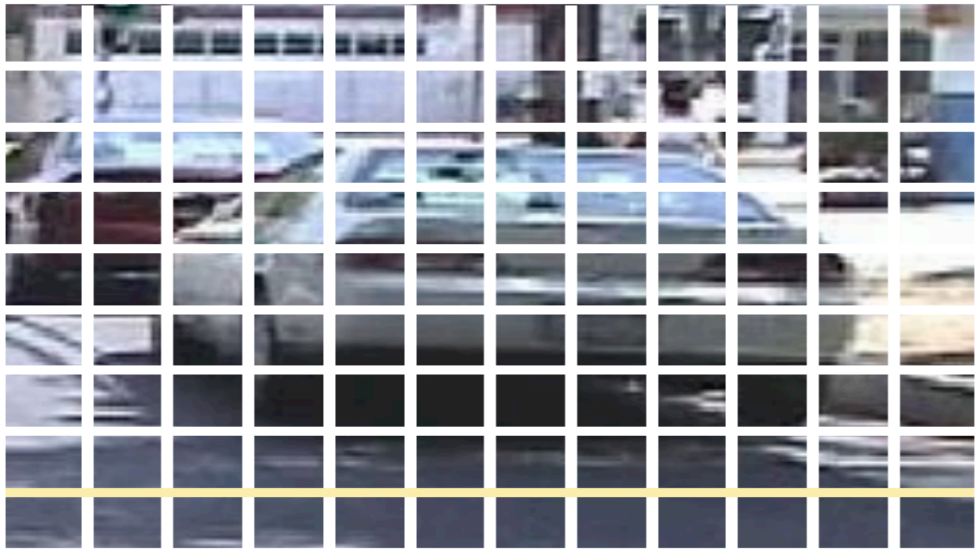






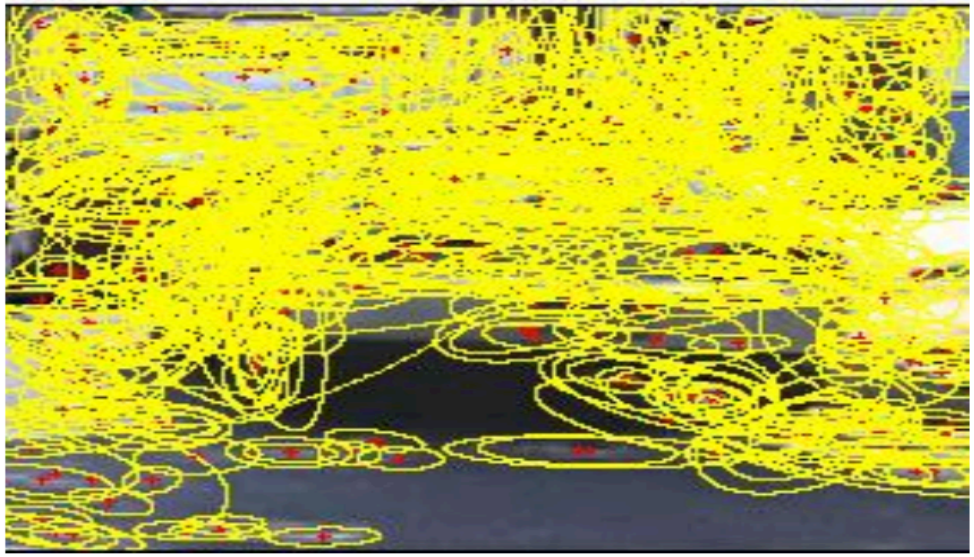
1. Feature extraction

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



1. Feature extraction

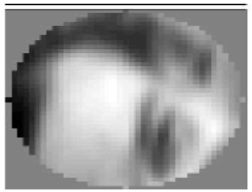
- Regular grid
  - Vogel & Schiele, 2003
  - Fei-Fei & Perona, 2005
- Interest point detector
  - Csurka et al. 2004
  - Fei-Fei & Perona, 2005
  - Sivic et al. 2005



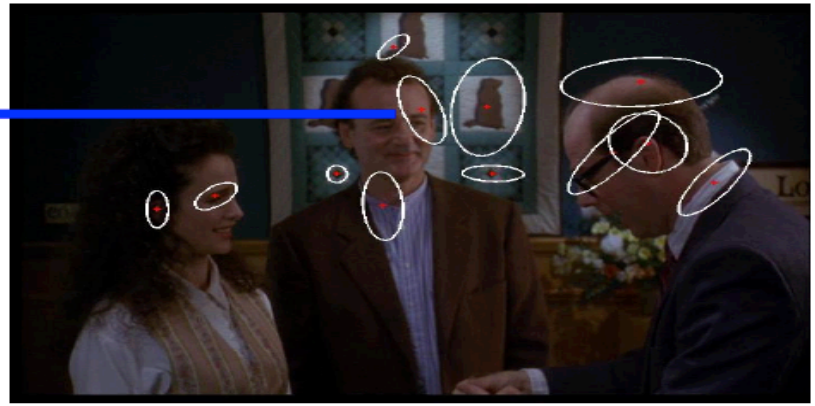
1. Feature extraction

Compute SIFT descriptor

[Lowe'99]



Normalize patch



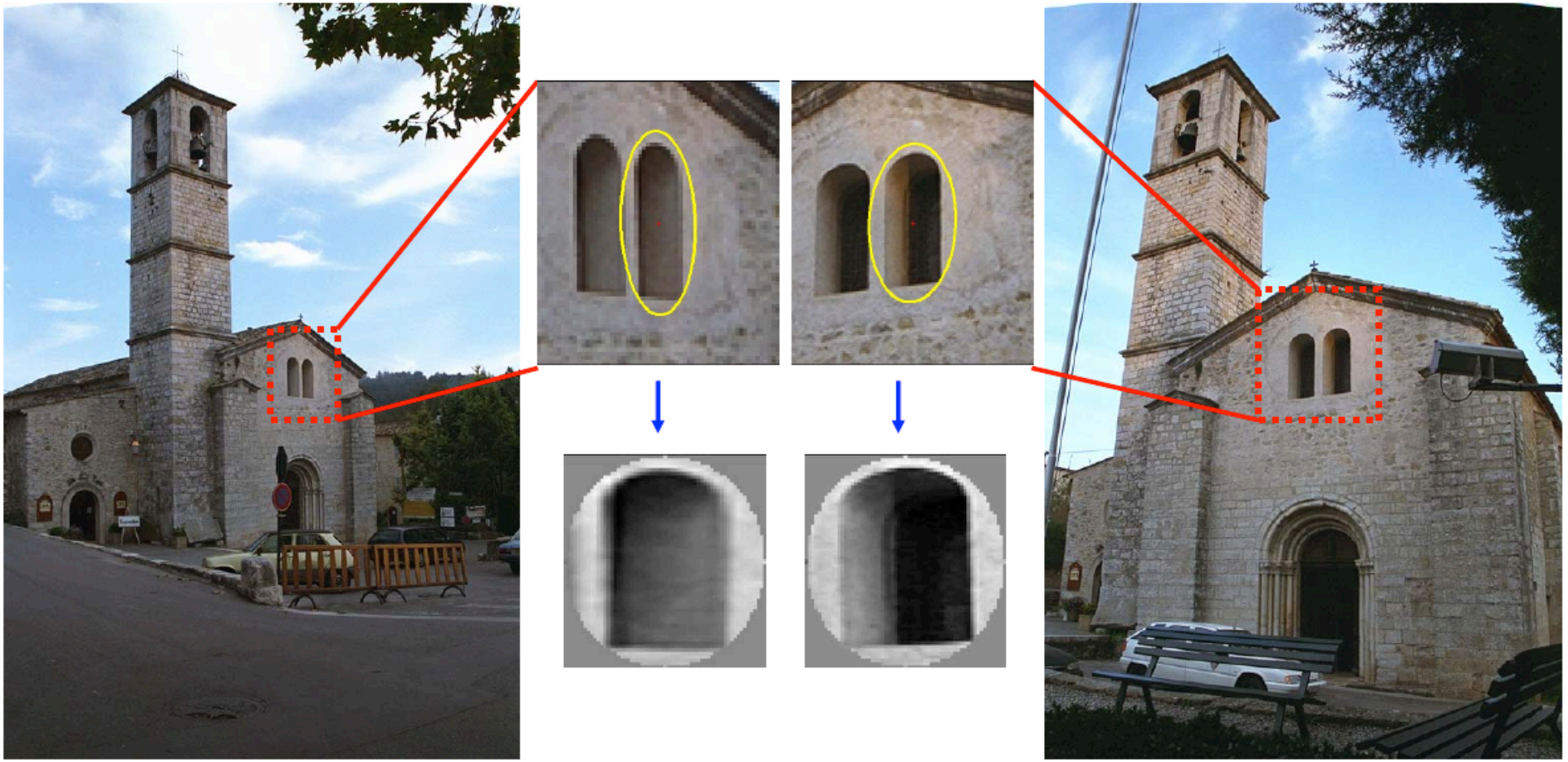
Detect patches

[Mikojaczyk and Schmid '02]

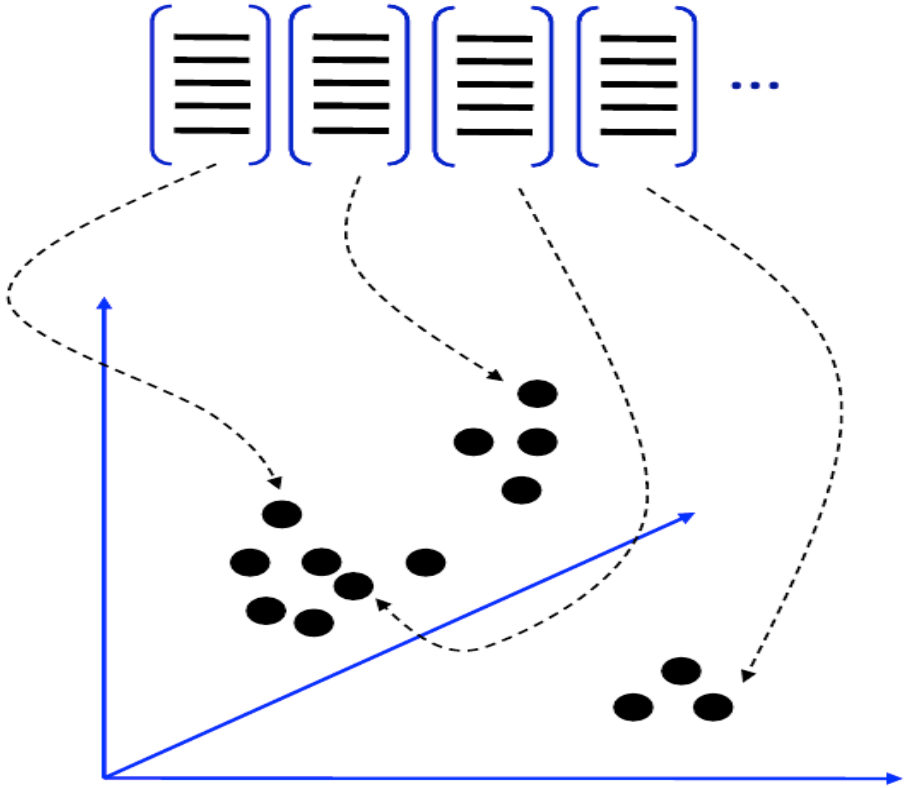
[Mata, Chum, Urban & Pajdla, '02]

[Sivic & Zisserman, '03]

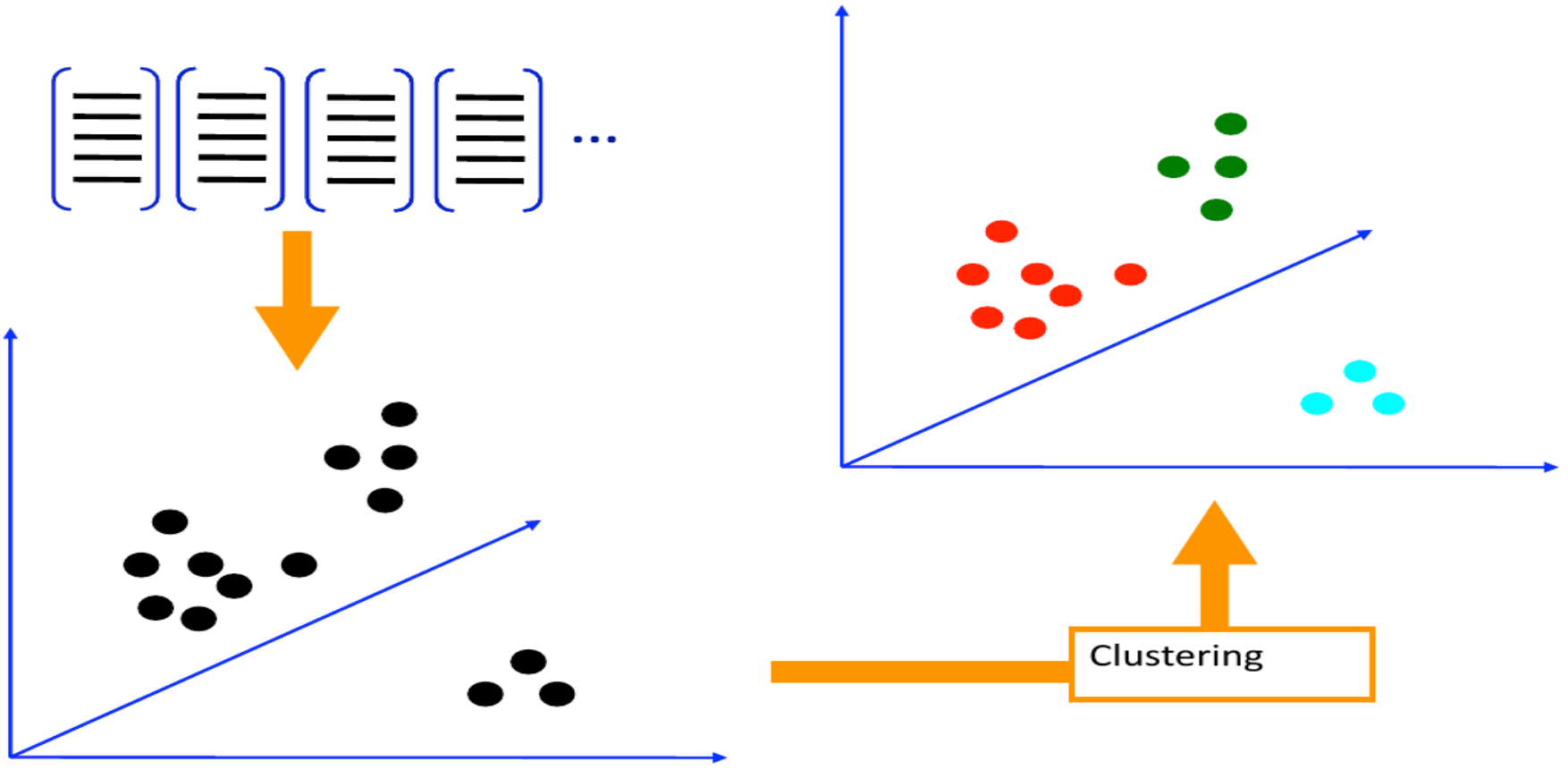
Slide credit: Josef Sivic



2. Learning the visual vocabulary



2. Learning the visual vocabulary



Slide credit: Josef Sivic

K-means clustering

- Want to minimize sum of squared Euclidean distances between points x_i and their nearest cluster centers m_k

$$D(X, M) = \sum_{\text{cluster } k} \sum_{\substack{\text{point } i \text{ in} \\ \text{cluster } k}} (x_i - m_k)^2$$

- 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

Difference between features & words

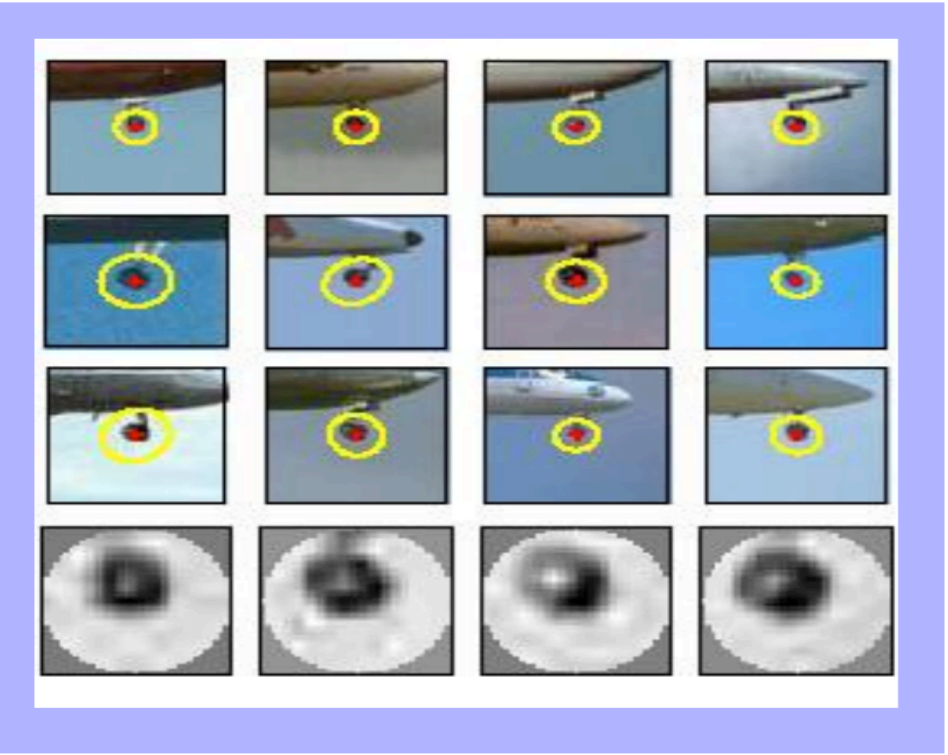
Words

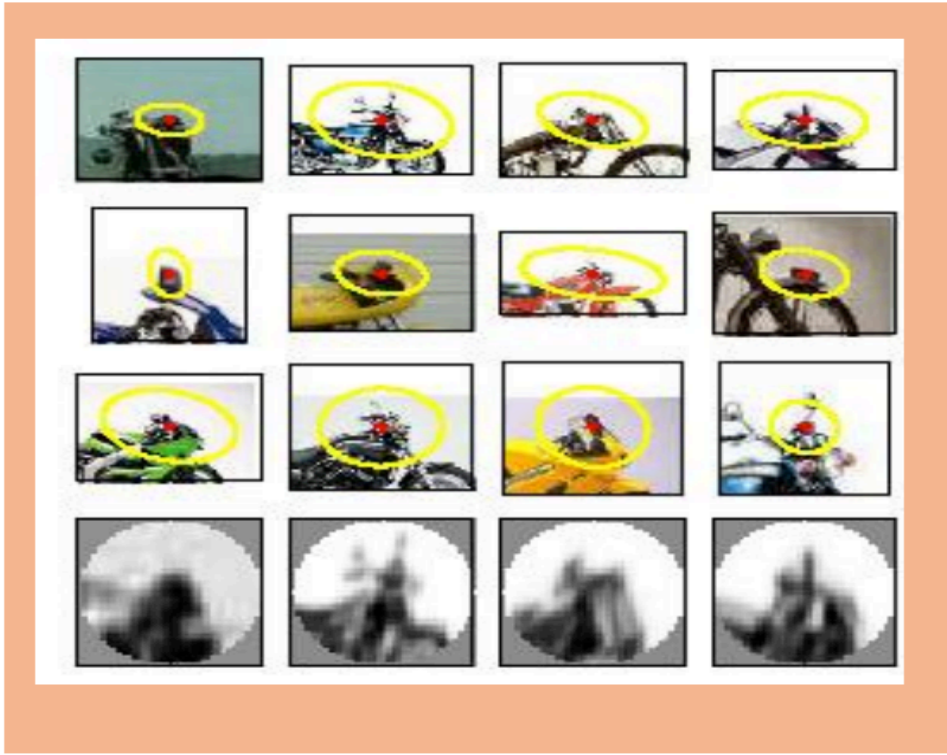
- Dictionary/Vocab.
- Meaning.
- Finite/Precise.
- Language known.
- Simple.
- Language-objective

Features

- Dictionary/Vocab ?
- Meaning ?
- Finite/Precise ?
- What language ?
- Complicated!
- Training subjective!

Image patch examples of visual words

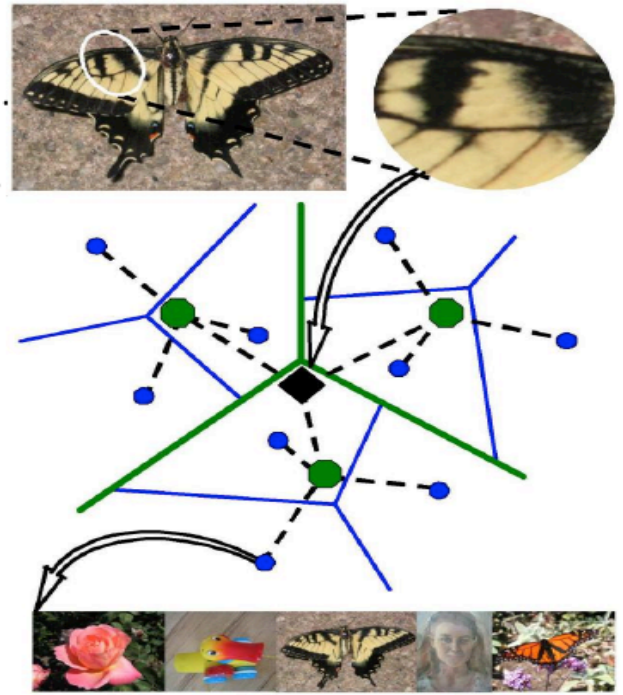




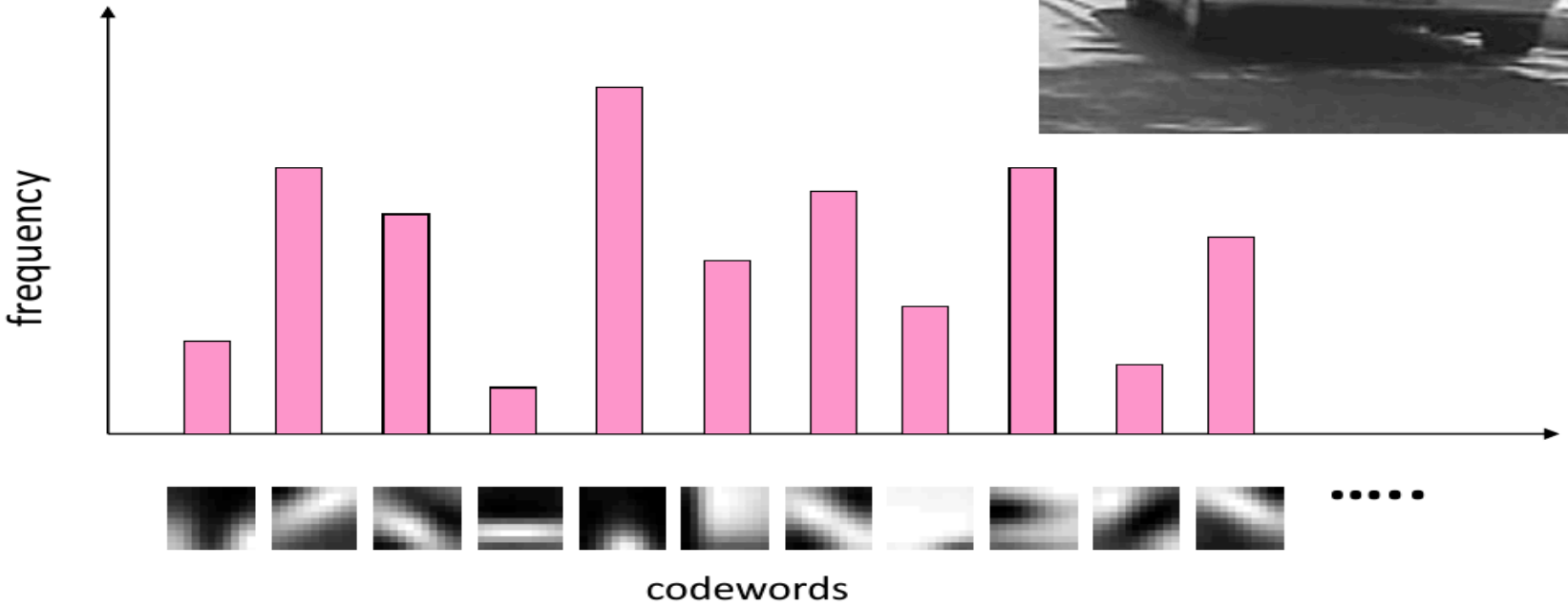
Sivic et al. 2005

Visual vocabularies: Issues

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



3. Image representation

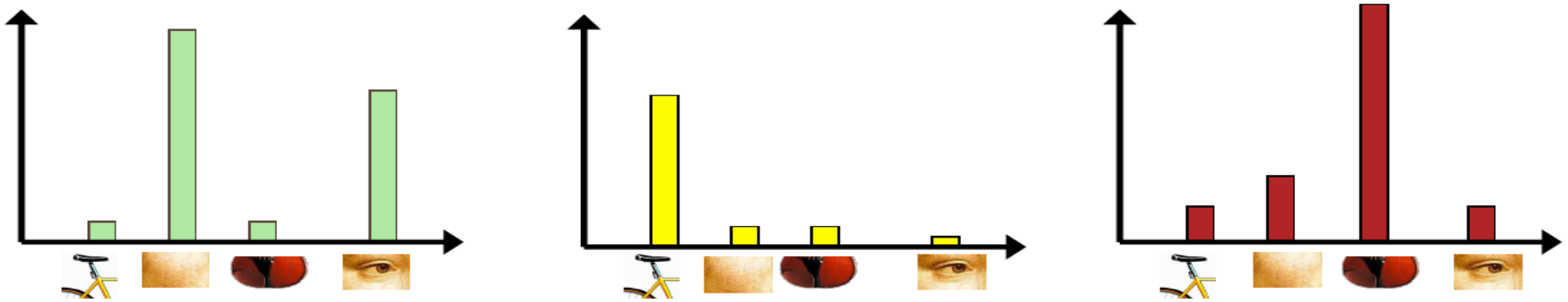


Problems we can solve (attempt) ?

- Classification: Do these two images (visual documents) belong to the same subject ?
- Recognition: Which images contain chairs ?
- If images are documents, what are videos ?
- Actions are sequence of visual words organized in time.
- How to get better (spatial) representation (to enforce “structure” in documents/images) ?
- Search!

Image classification

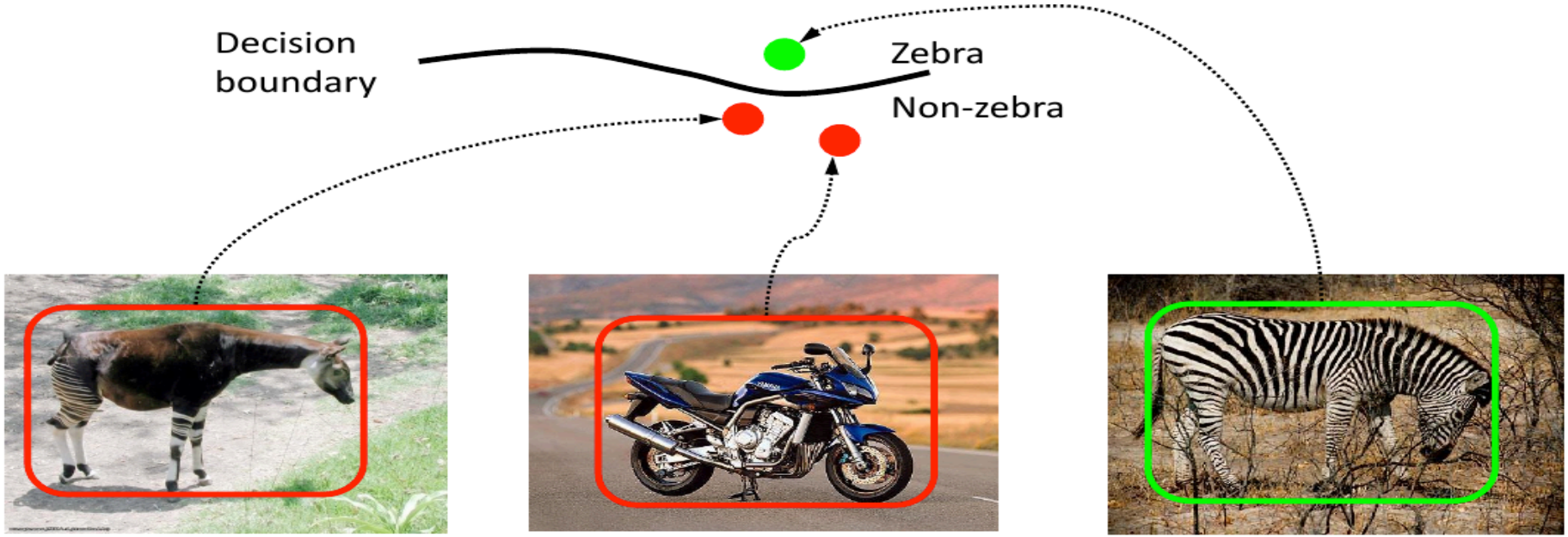
- Given the bag-of-features representations of images from different classes, how do we learn a model for distinguishing them?



Discriminative and generative methods for bags of features

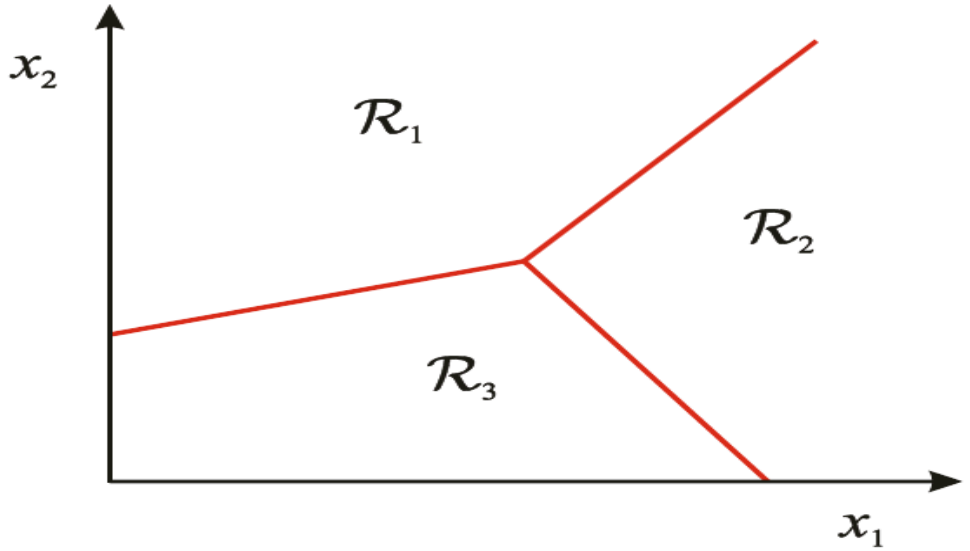
Discriminative methods

- Learn a decision rule (classifier) assigning bag-of-features representations of images to different classes



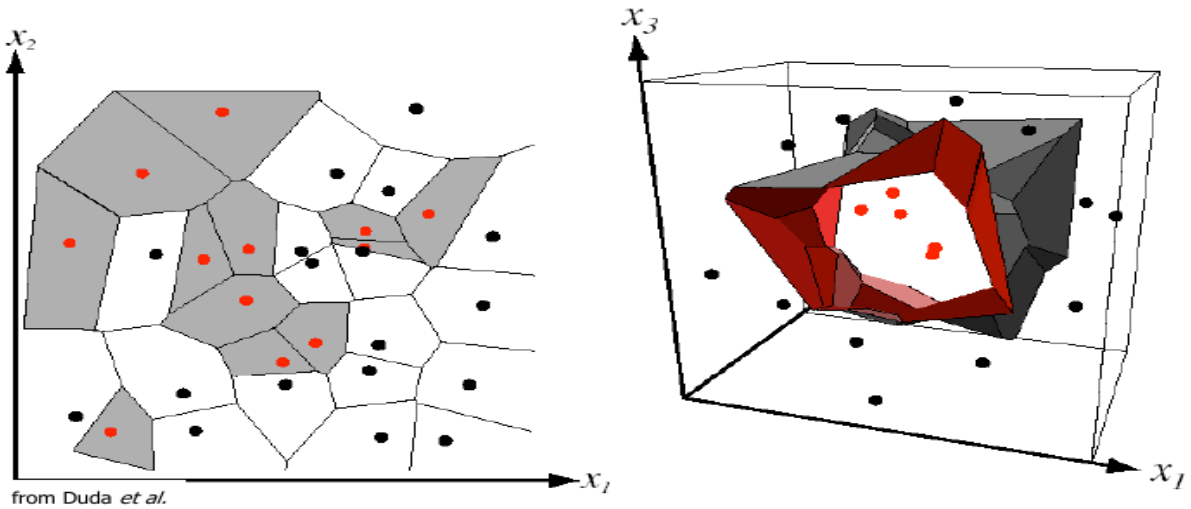
Classification

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



Nearest Neighbor Classifier

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



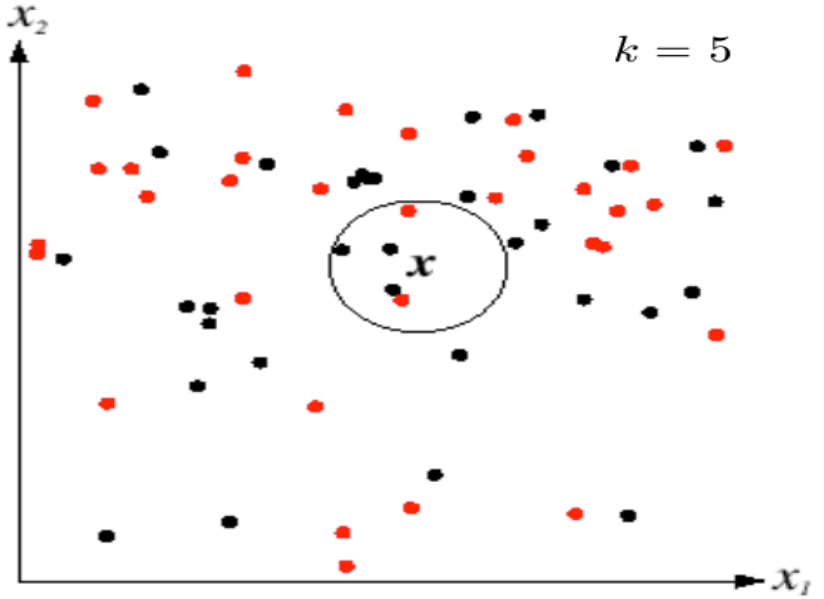
from Duda *et al.*

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

Source: D. Lowe

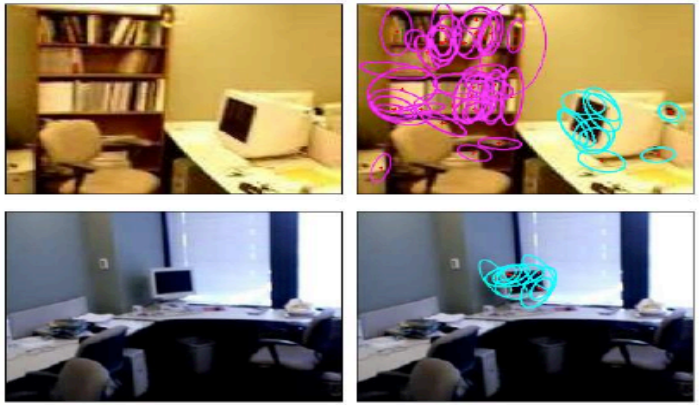
K-Nearest Neighbors

- For a new point, find the k closest points from training data
- Labels of the k points “vote” to classify
- Works well provided there is lots of data and the distance function is good



Source: D. Lowe

Topic discovery in images



J. Sivic, B. Russell, A. Efros, A. Zisserman, B. Freeman, **Discovering Objects and their Location in Images**, *ICCV 2005*

Multiple Actions

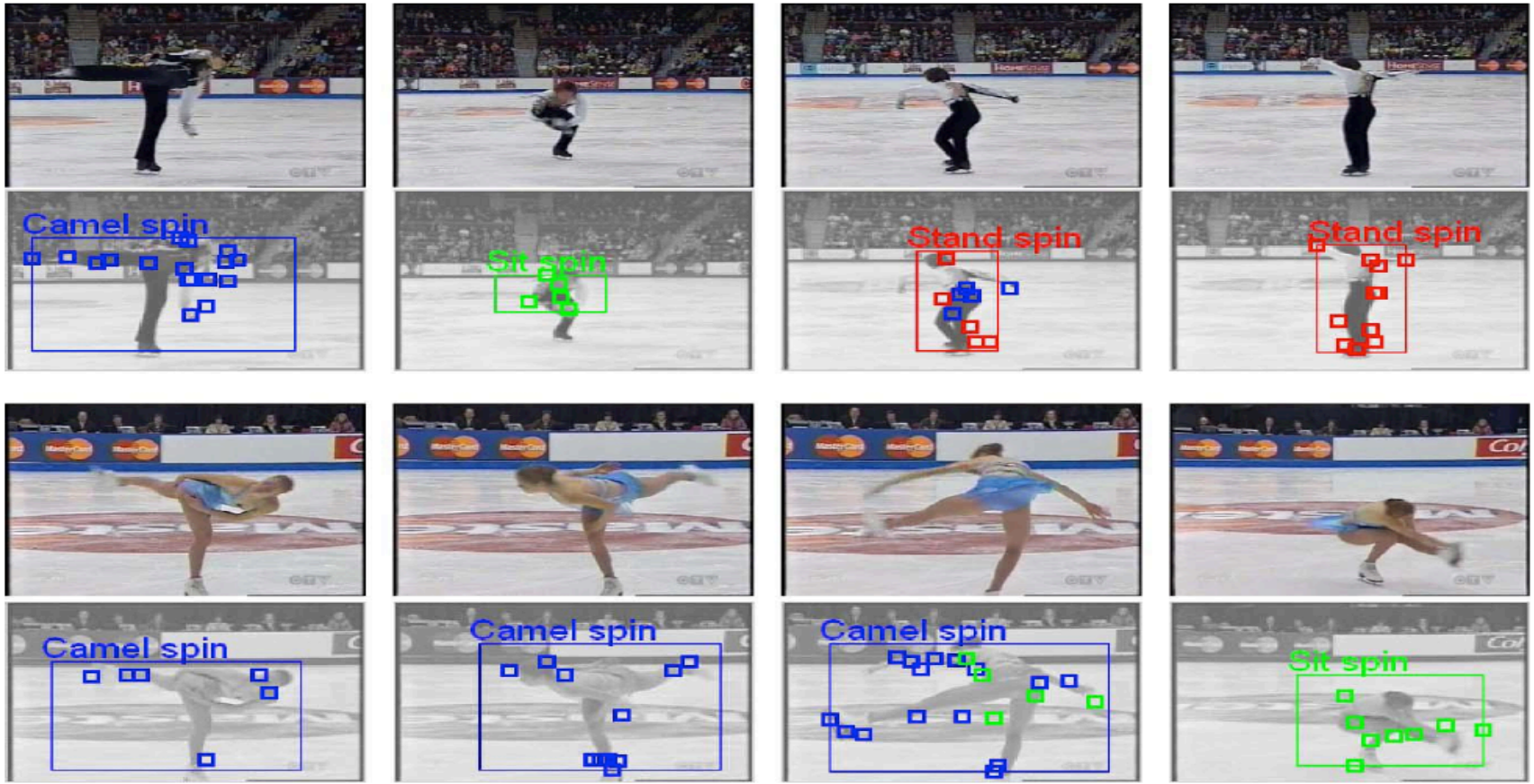


Image and Document differences

Document

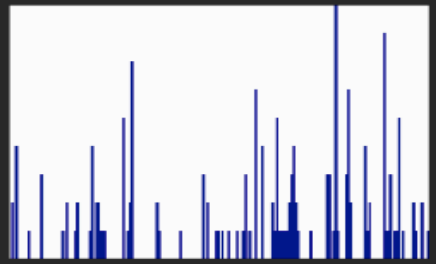
- Single scale.
- Words have order in document!
- 1D space.
- Google!

Image

- Multiple scales!
- Features are *loosely* coupled.
- 2D space.
- TinEye!

Spatial pyramid representation

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



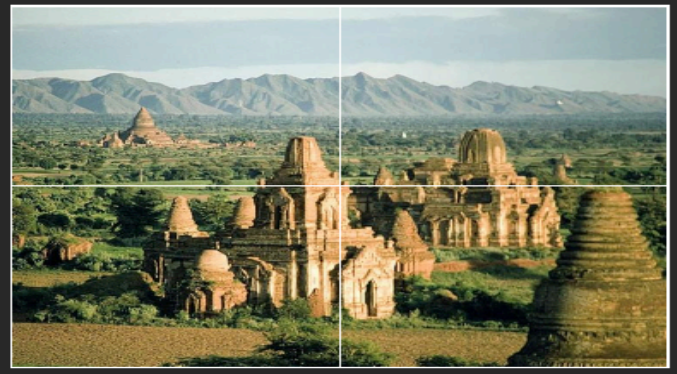
level 0

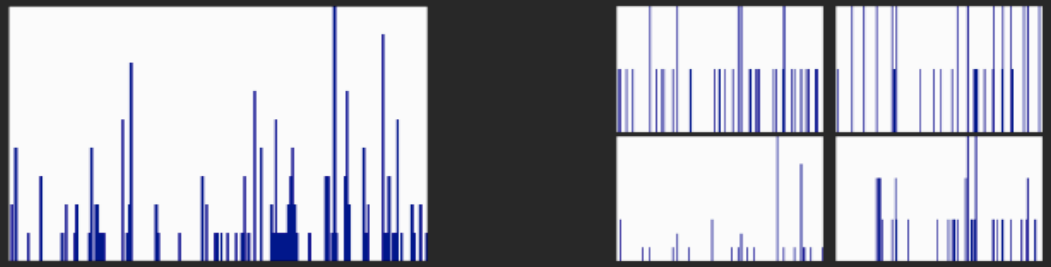
Lazebnik, Schmid & Ponce (CVPR 2006)

Slide: S. Lazebnik

Spatial pyramid representation

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



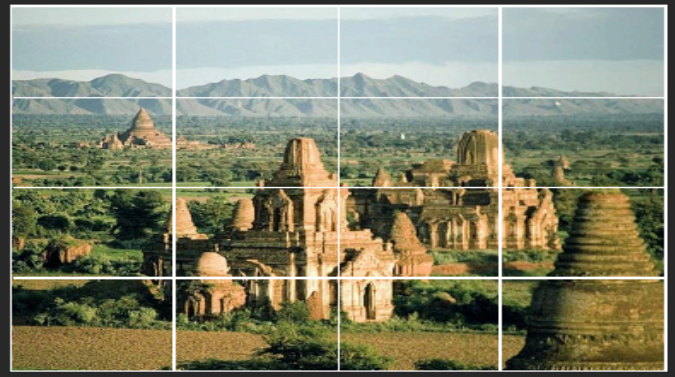


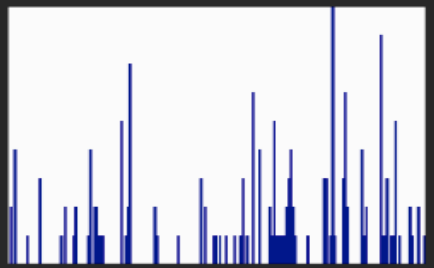
Lazebnik, Schmid & Ponce (CVPR 2006)

Slide: S. Lazebnik

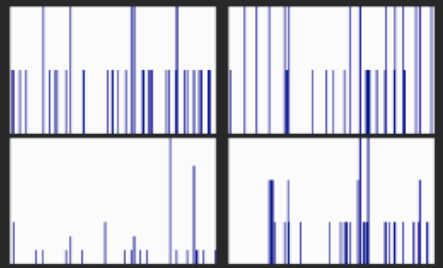
Spatial pyramid representation

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

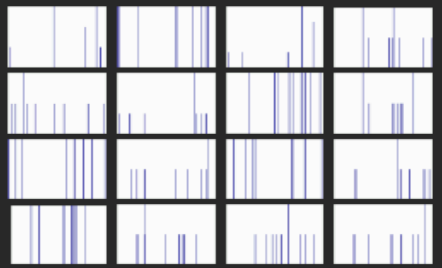




level 0



level 1



level 2

Lazebnik, Schmid & Ponce (CVPR 2006)

Slide: S. Lazebnik

Image Search!

- What kind of searches possible ?
  - We will see at least 2 types.

- Why is image search important ?
  - Copyright problems / attribution !
  - Words might not be enough. (Find me images of dresses / goggles worn by …. in ….. movie)
  - What is the name of the monument in this image ?

Image Search

- Concept of “visual word” slightly different in this case.
  - Definitions are more contextual!
- Problem statement allows to add more constraints to search.
  - Geometric, Spatial.

tineye.com

Search

We are hiring!

TinEye

About Products Solutions Labs

Register | Contact Us | Log in

Uses of TinEye ?

TinEye

Reverse Image Search

UPLOAD Your Image

Browse... No file selected.

or

ENTER Image Address

paste URL here (page or image) Search

Interesting searches: Dollar bill, Man on the moon, V-J day

Use a Browser Plugin

Available for Firefox, Chrome, Safari, Internet Explorer & Opera.

Drag & Drop:

Drag any image file and drop it anywhere on this page.

TinEye Inc

- About
- Clients
- Careers

TinEye APIs for Business

- TinEye Services
- MulticolorEngine
- MatchEngine

TinEye Resources

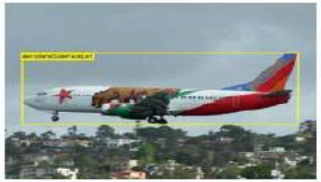
- Add Your Images
- Updates
- Plugins

Contact TinEye

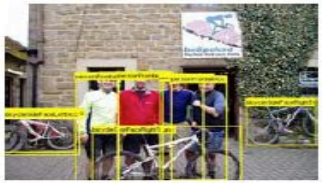
- Get in Touch
- TinEye Blog

Examples from PASCAL VOC Challenge 2010

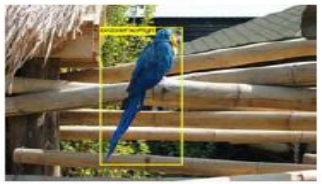
Aeroplane



Bicycle



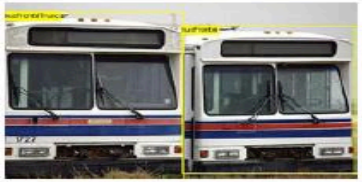
Bird



Boat

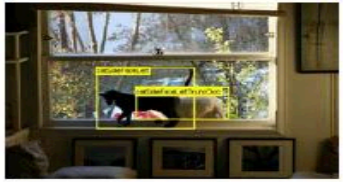
Bottle

Bus



Car

Cat



Chair

Cow

Video Google

Enable video, e.g. a feature length film, to be searched on its visual content with the same ease and success as a Google search of text documents.

"Run Lola Run" ('Lola Rennt')
[Tykwer, 1999]

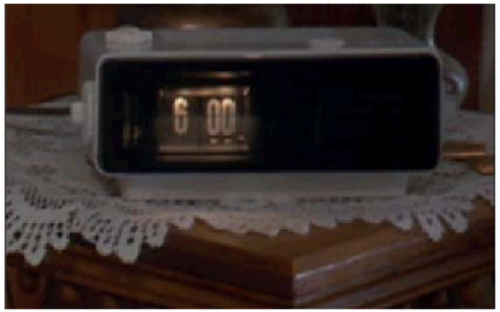
"Groundhog Day" [Rammis, 1993]

Given an object specified in one frame, retrieve all shots containing the object:

- must handle viewpoint change
- must be efficient at run time

Example : Groundhog Day

close-up



Example: Groundhog Day

73 keyframes retrieved

53 correct, first incorrect ranked 27

Rank: 12 35 50 69

Approach

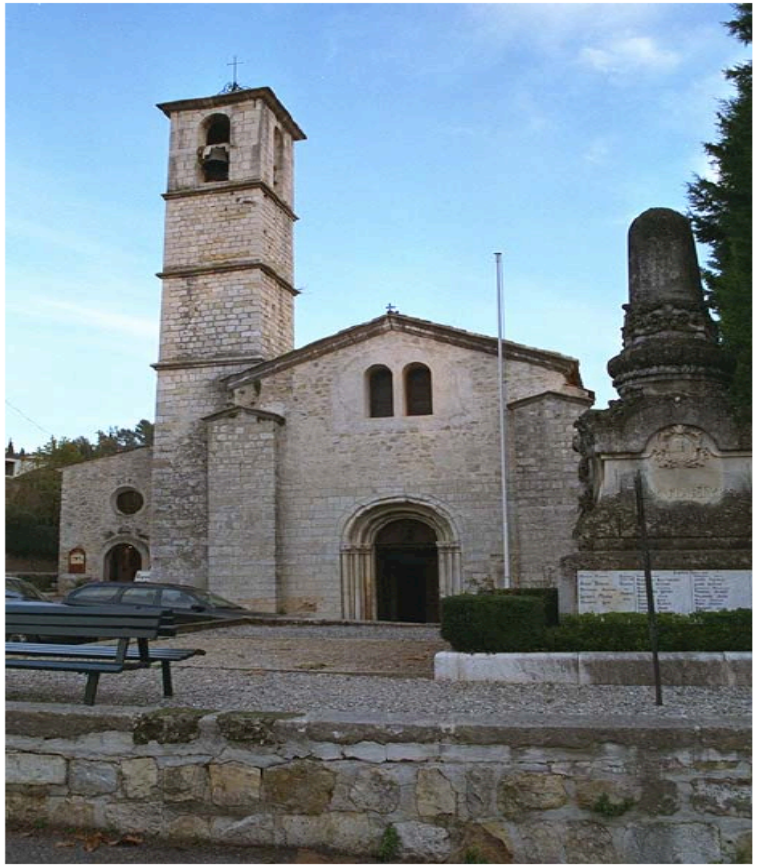
- Determine regions (segmentation) and vector descriptors in each frame which are invariant to camera viewpoint changes

- Match descriptors between frames using invariant vectors

- Verify / reject frame matches using spatial consistency and multiple view geometric relations:
  - spatial neighbours match
  - homographies
  - epipolar geometry

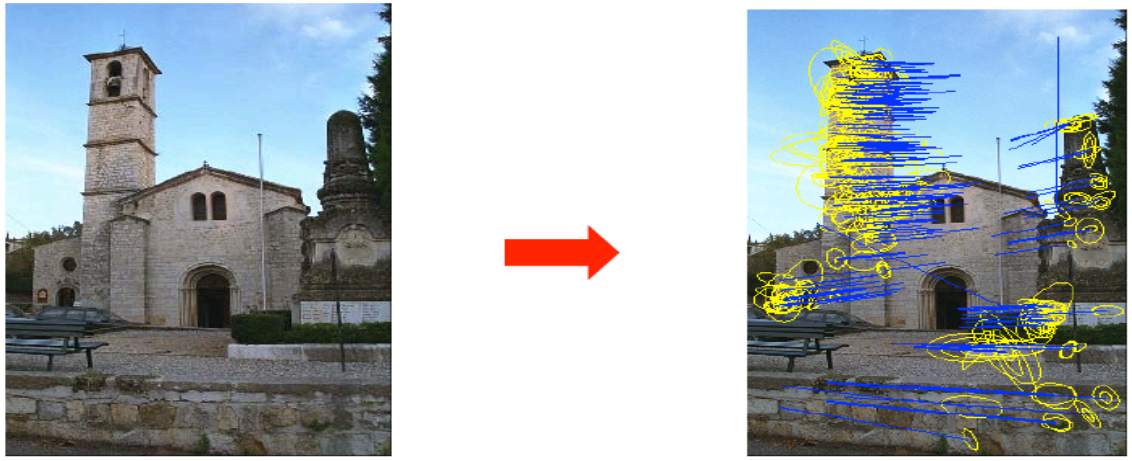
- Frames are matched if a sufficient number of regions satisfy the geometric relations between views

Example

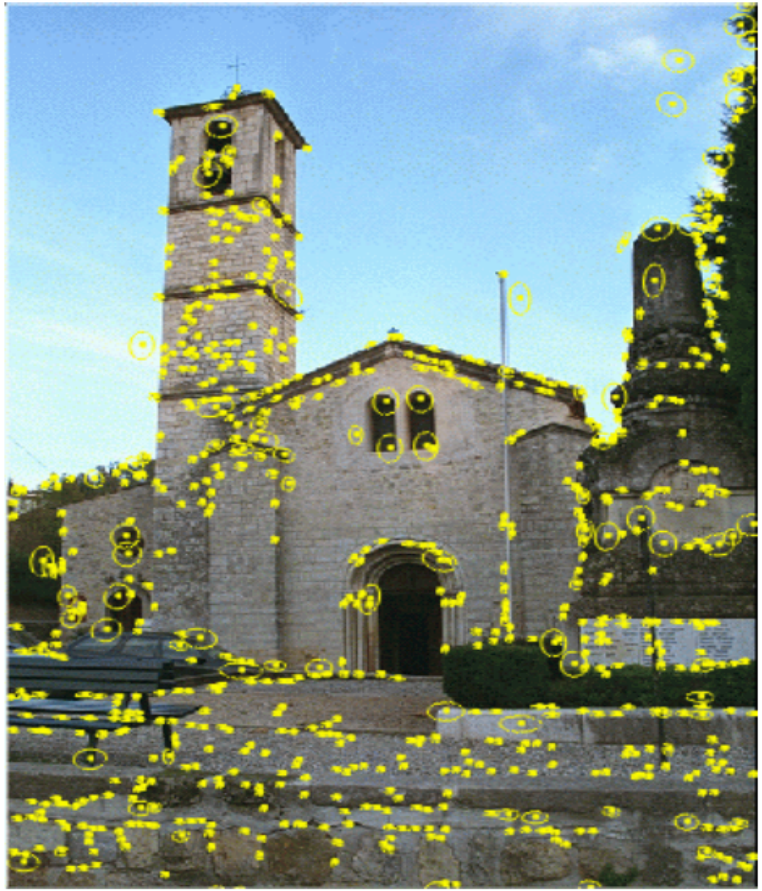


512 x 768 pixel images

Goal: establish matches



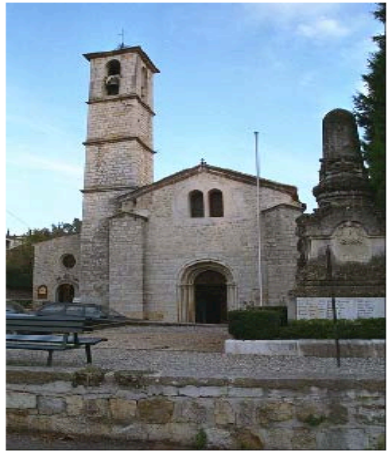
A sub-set of the shape adapted interest points

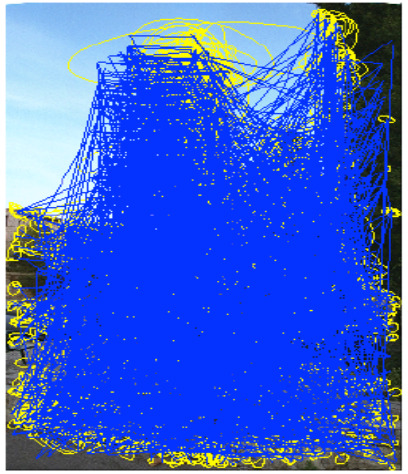


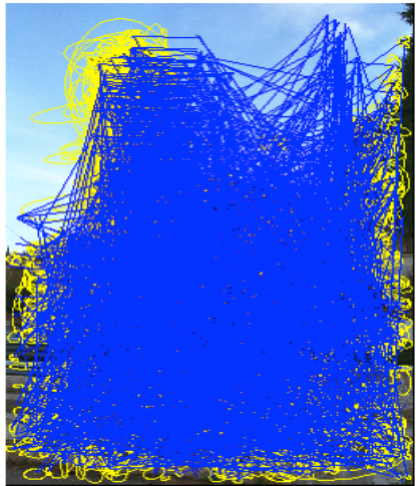
Stage (1): invariant indexing

Match closest vectors

- all vectors within a ball in invariant space





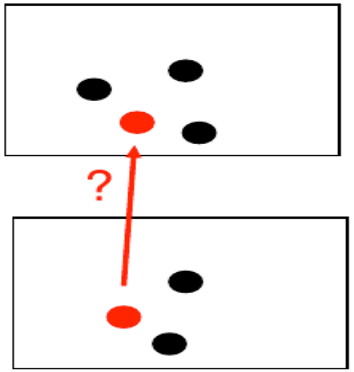


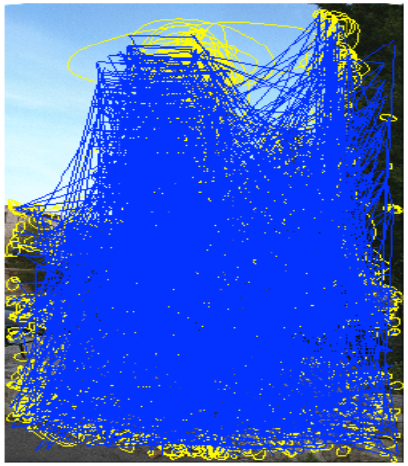
Stage (2): neighbourhood consensus

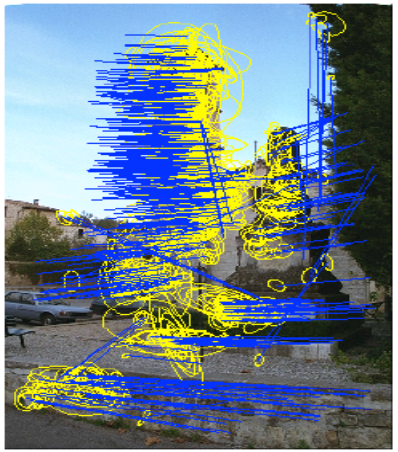
Loose constraint on spatial consistency for a putative match

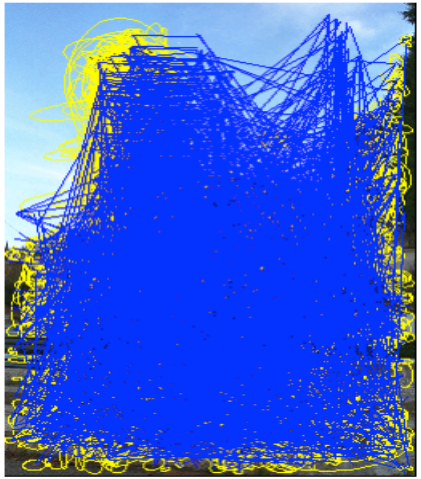
- compute the K closest matched neighbours of the point in each image
- require that at least N of these must match

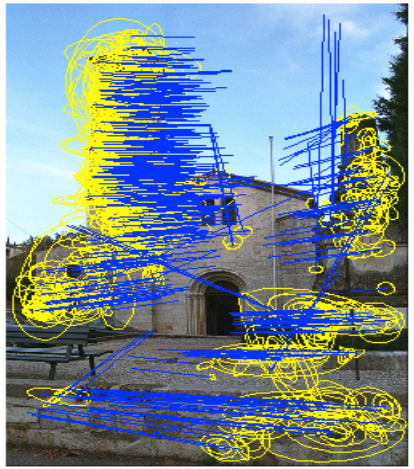
Here K =10, N = 1











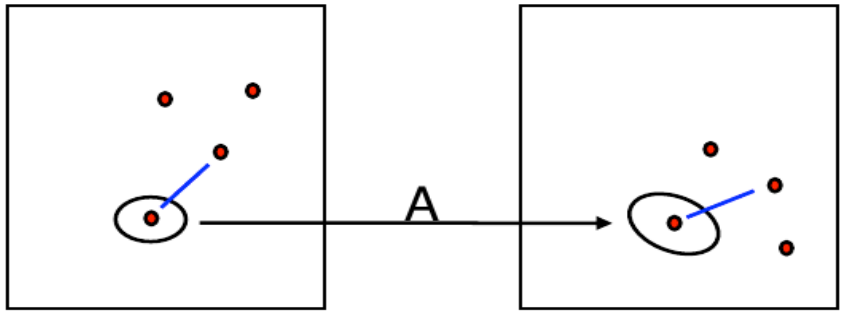
e.g. 1: local verification

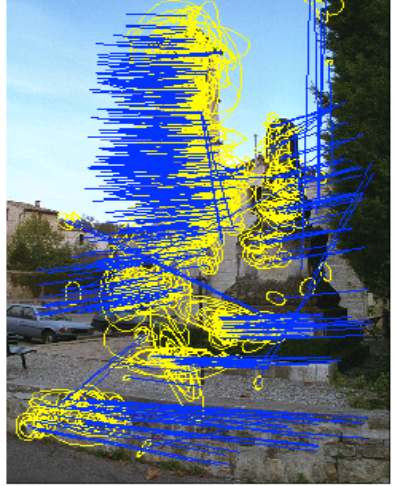
Matched invariants do not imply that the regions match

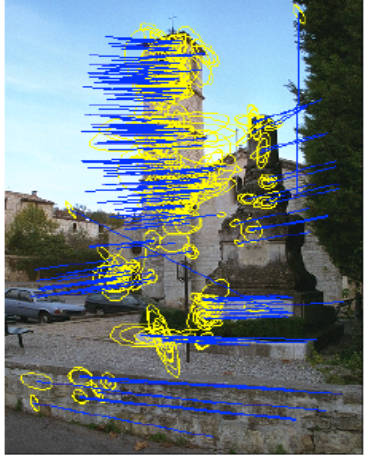
- regions match if cross-correlation after registration is below threshold

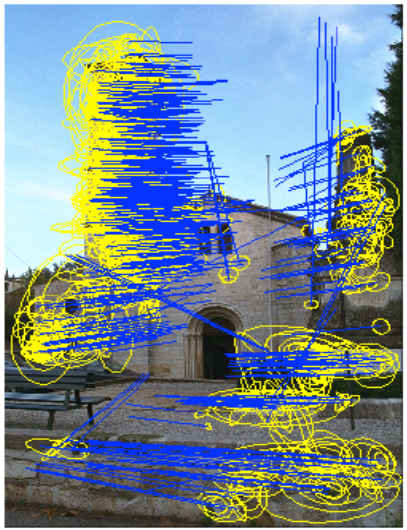
e.g. 2: local geometry

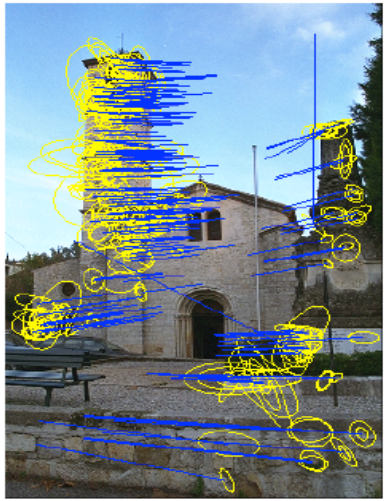
- neighbouring correspondences consistent with the affine transformation



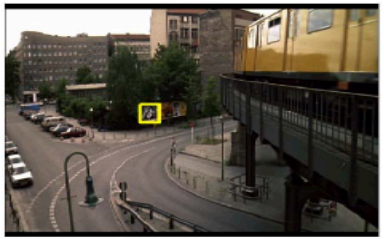


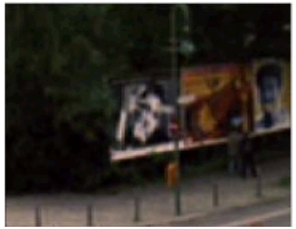






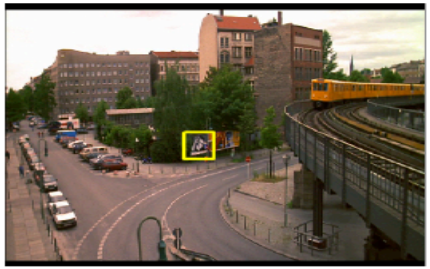
Example : Run Lola Run

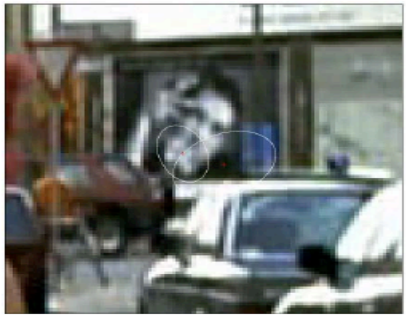




20 keyframes retrieved

all correct





Rank: 1 12 16 20

Questions!