Intelligent Multimedia Systems

Master AI, 2012, Lecture 5

Lecturers: Theo Gevers

Lab: Intelligent Systems Lab Amsterdam (ISLA)

Email: th.gevers@uva.nl

http://staff.science.uva.nl/~gevers

Lectures

- 29-10-2012, Monday, 15:00-17:00, Science Park A1.04 - Introduction
- 05-11-2011, Monday, 15:00-17:00, Science Park A1.04 - Image and Video Formation
- 12-11-2011, Monday, 15:00-17:00, Science Park A1.04 - Color Invariance and Image Processing
- 19-11-2011, Monday, 15:00-17:00, Science Park A1.04 - Feature Extraction and Tracking
- 26-11-2011, Monday, 15:00-17:00, Science Park A1.04 - Learning and Object Recognition
- 03-12-2011, Monday, 15:00-17:00, Science Park A1.04 - Visual Attention and Affective Computing
- 10-12-2011, Monday, 15:00-17:00, Science Park A1.04 - Human Behavior Analysis
- 18-12-2011, Tuesday, 15:00-18:00, Science Park, C1.10 - Examination

Today's class

Mosaics

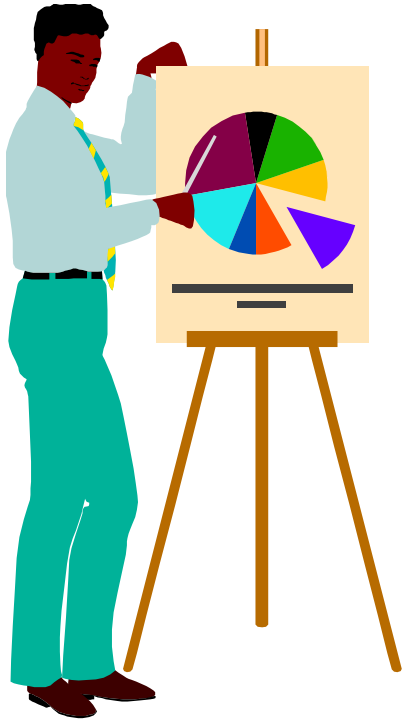
Object Recognition in Perspective

Image Descriptors

Bag-of-Models

Classifiers

Object Recognition Benchmarks



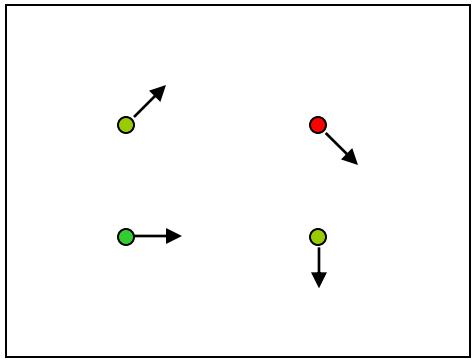
Mosaicing

Feike Winkelman

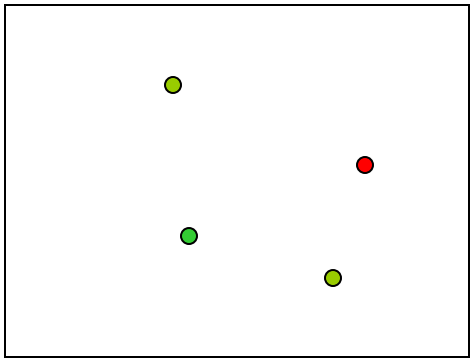
input movie:

mosaic creation:

Feature Tracking



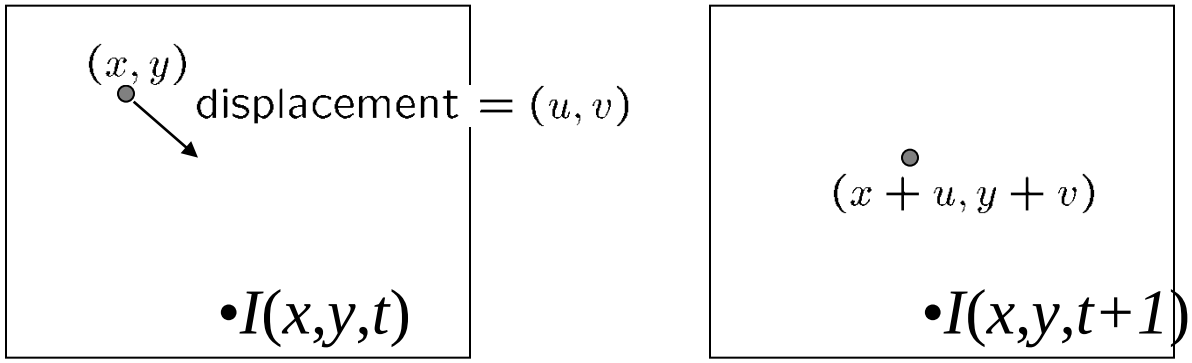
$\bullet I(x,y,t)$



$\bullet I(x,y,t+1)$

- Given two subsequent frames, estimate the point translation
- Key assumptions of Lucas-Kanade Tracker
  - **Brightness constancy:** projection of the same point looks the same in every frame
  - **Small motion:** points do not move very far
  - **Spatial coherence:** points move like their neighbors

The Brightness Constancy Constraint



- Brightness Constancy Equation:

$$I(x,y,t) = I(x+u, y+v, t+1)$$

- Take Taylor expansion of $I(x+u, y+v, t+1)$ at (x,y,t) to linearize the right side:

•Image derivative along x •Difference over frames

$$I(x+u,y+v,t+1) \approx I(x,y,t) + I_x \cdot u + I_y \cdot v + I_t$$

$$I(x+u,y+v,t+1) - I(x,y,t) = +I_x \cdot u + I_y \cdot v + I_t$$

- Hence, $I_x \cdot u + I_y \cdot v + I_t \approx 0 \quad \rightarrow \nabla \mathrm{I} \cdot \left[\mathrm{u}\ \mathrm{v}\right]^{\mathrm{T}} + \mathrm{I_t} = 0$

Solving the Ambiguity…

•B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 674–679, 1981.

- How to get more equations for a pixel?
- Spatial coherence constraint
- Assume the pixel's neighbors have the same (u,v)
  - If we use a 5x5 window, that gives us 25 equations per pixel

$$0 = I_t(\mathbf{p_i}) + \nabla I(\mathbf{p_i}) \cdot [u\ v]$$

$$\begin{bmatrix} I_x(\mathbf{p_1}) & I_y(\mathbf{p_1}) \\ I_x(\mathbf{p_2}) & I_y(\mathbf{p_2}) \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}}) & I_y(\mathbf{p_{25}}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p_1}) \\ I_t(\mathbf{p_2}) \\ \vdots \\ I_t(\mathbf{p_{25}}) \end{bmatrix}$$

Matching Patches across Images

- Overconstrained linear system

$$\begin{bmatrix} I_x(\mathbf{p_1}) & I_y(\mathbf{p_1}) \\ I_x(\mathbf{p_2}) & I_y(\mathbf{p_2}) \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}}) & I_y(\mathbf{p_{25}}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p_1}) \\ I_t(\mathbf{p_2}) \\ \vdots \\ I_t(\mathbf{p_{25}}) \end{bmatrix}$$

$$\underset{25\text{x}2}{A} \quad \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b}$$

- Least squares solution for *d* given by $(A^T A)\ d = A^T b$

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

$A^T A$ $\qquad\qquad$ $A^T b$

- The summations are over all pixels in the K x K window

- *M* = $A^T A$ is the *second moment matrix* !
- (Harris corner detector…)

$$A^T A = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \; I_y] = \sum \nabla I (\nabla I)^T$$

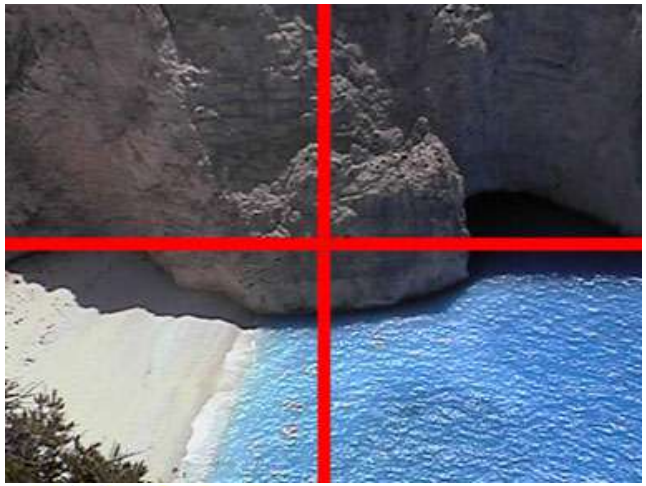
- Eigenvectors and eigenvalues of $A^T A$ relate to edge direction and magnitude
  - The eigenvector associated with the larger eigenvalue points in the direction of fastest intensity change
  - The other eigenvector is orthogonal to it

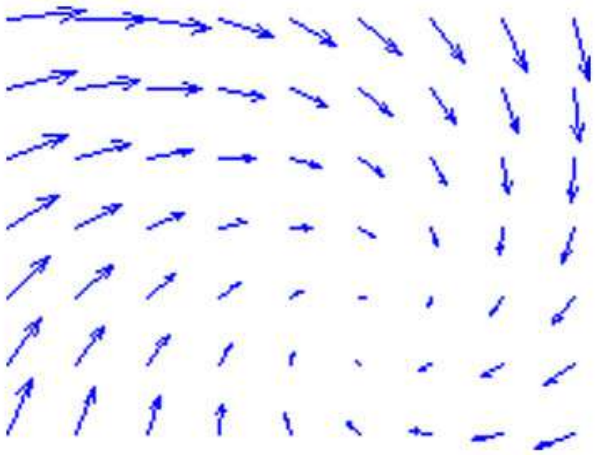
How Does is Work

- Estimate motion between successive frames I_t and I_{t+1}. The motion model used is the affine transformation.

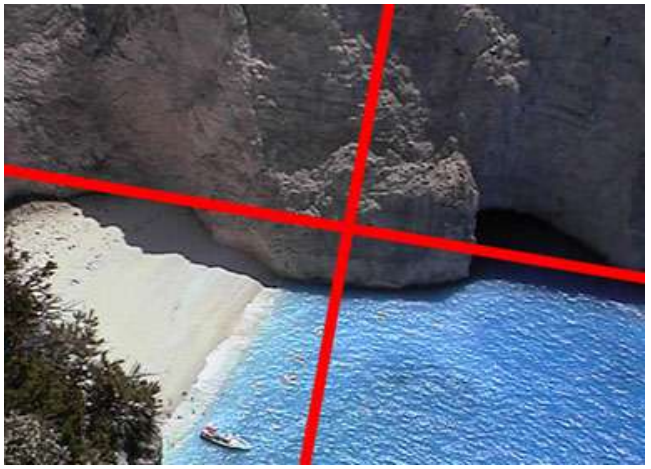
- Use this estimation to warp I_{t+1} into the coordinate system of I_t.

I_{t0}:





I_{t1}:



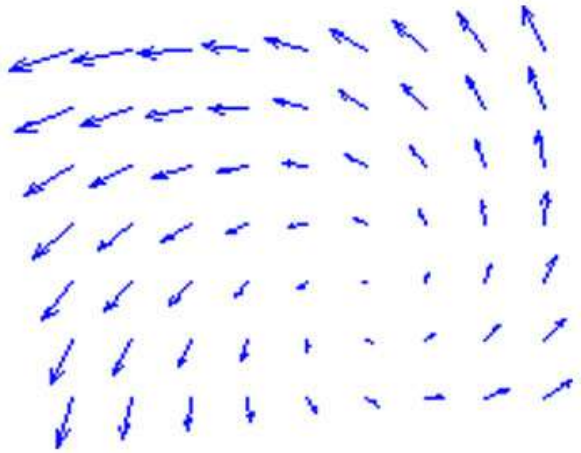
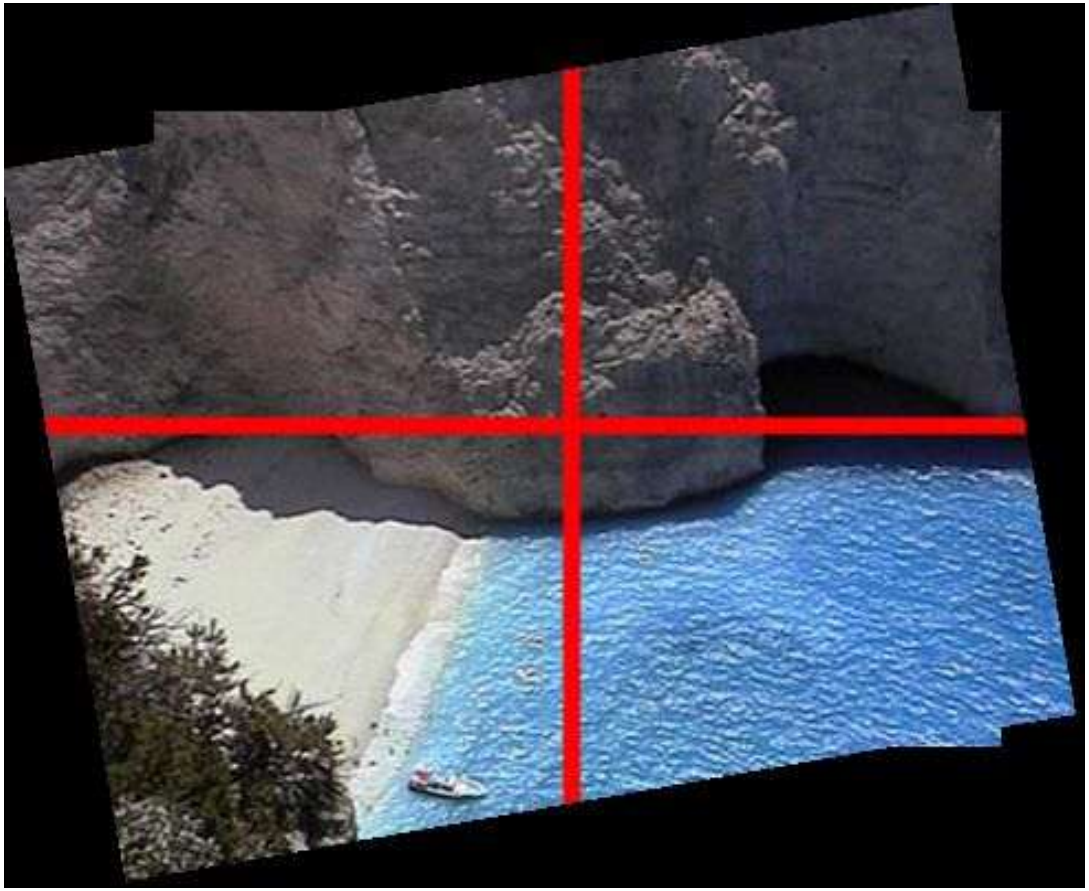


image I_{t1} warped into I_{t0} coordinates (using $A_{inverse}$):



More than 2 Images

If:

A_{t1} : affine motion from image I_{t1} to I_{t0},
A_{t2} : affine motion from image I_{t2} to I_{t1},
$A_t(I_t)$: transforms coordinates I_t to I_{t-1} with A

Then:

$A_{t1}(A_{t2}(I_{t2}))$ puts image I_{t2} into the coordinate system of image I_{t0}.

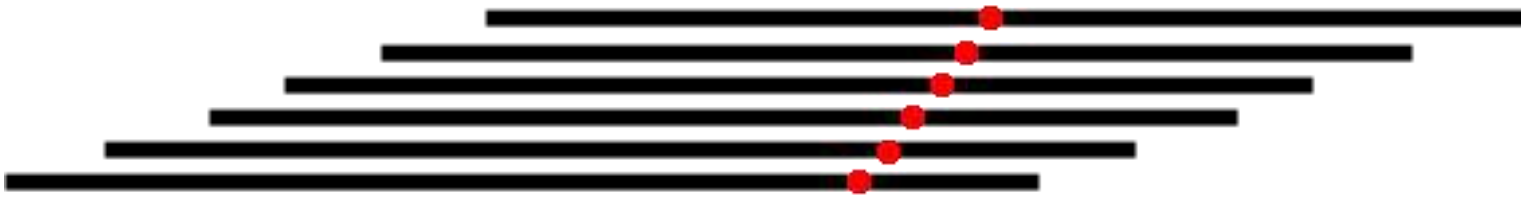
(In matrix form affine transformations can be multiplied to concatenate their effect.)

More than 2 Images

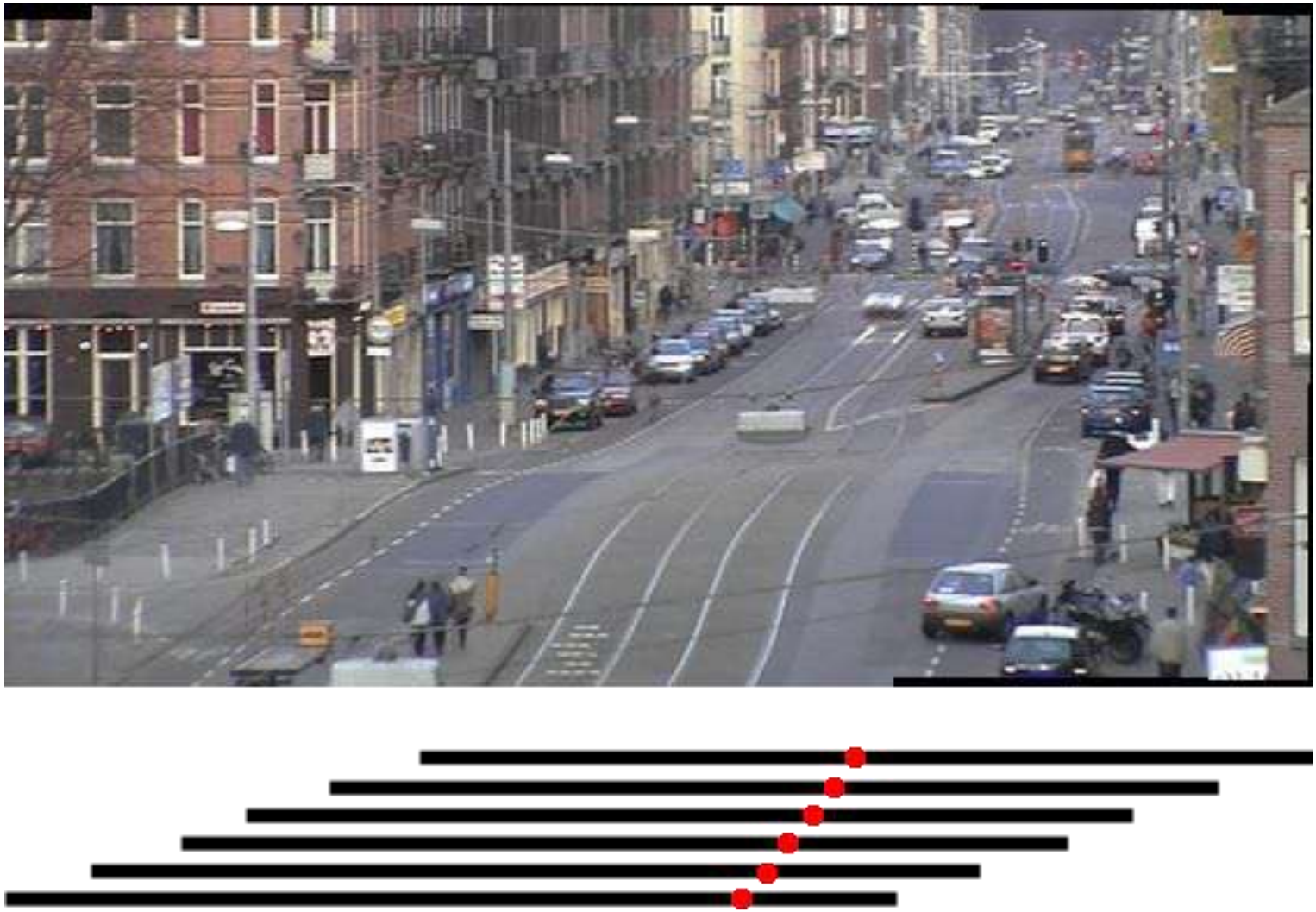
t_1 t_2 t_3 t_4 t_5 t_6

Only pixels of ‘last frame’ are used when updating the mosaic…

More than 2 Images



More than 2 Images: Median

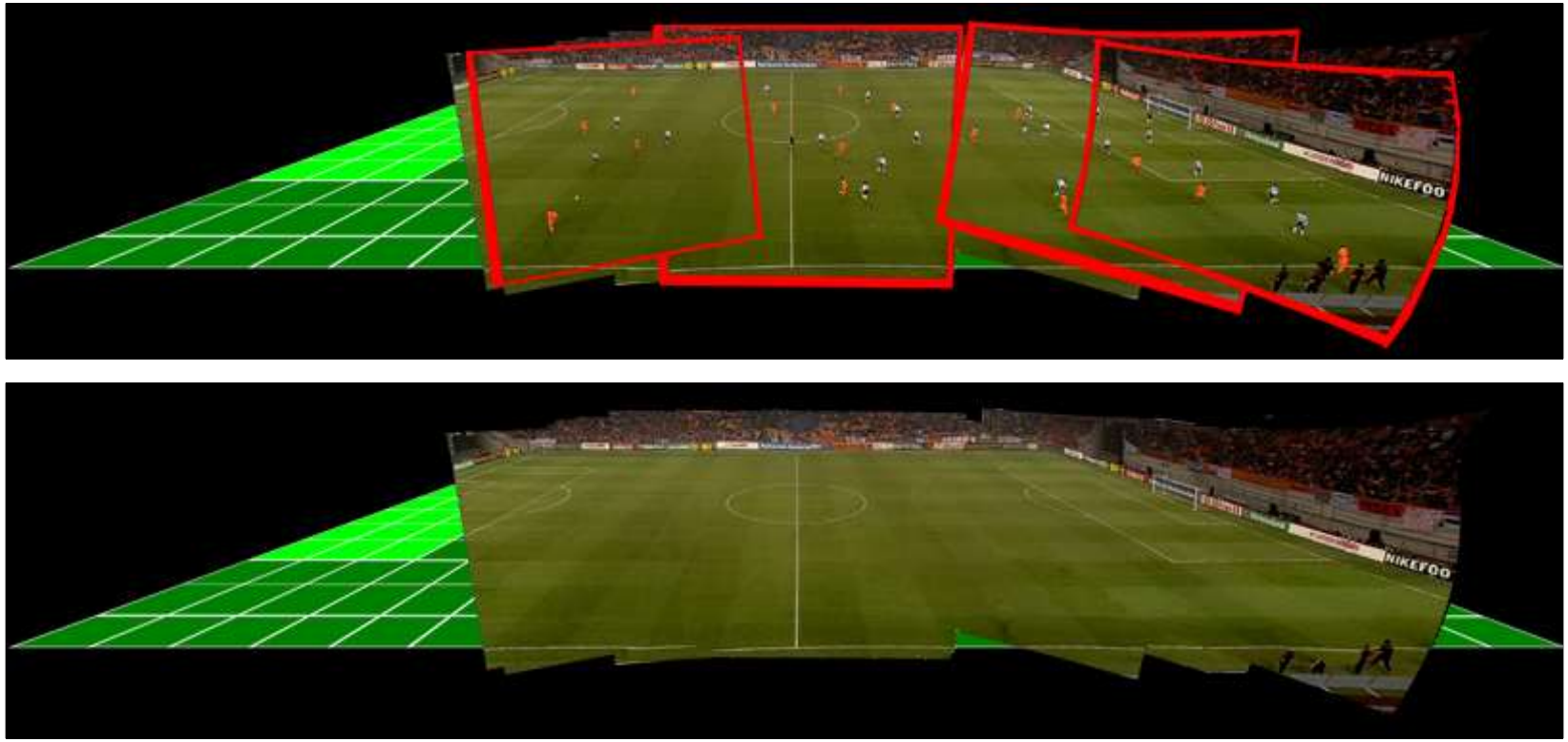


CONSTRUCTING PLAYER TRAJECTORIES FROM MOSAICS FOR SPORT STATISTICS

zoom (2x) <= pan/zoom <= pan


Adecco
Shell
Staatsloterij

CONSTRUCTING PLAYER TRAJECTORIES FROM MOSAICS FOR SPORT STATISTICS



- Composite *wide-angle* panorama consisting of a collection of seamlessly *stitched* overlapping images.

CONSTRUCTING PLAYER TRAJECTORIES FROM MOSAICS FOR SPORT STATISTICS

- Mosaic is based on *rigid-transformations (rotation, scale and translation).* Preserving the shape of individual frames.
- Mosaic is manually pre-constructed (*bootstrap* problem). Automatic construction relies on frame registration.

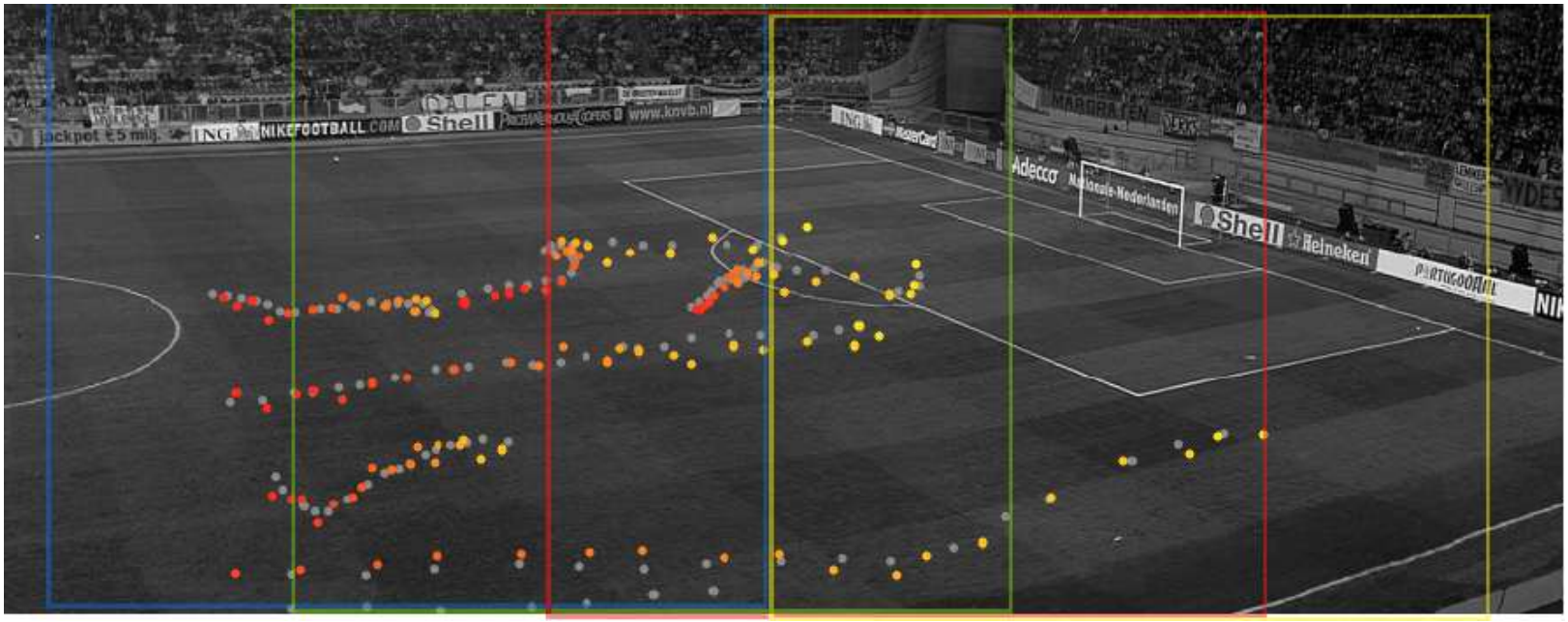
CONSTRUCTING PLAYER TRAJECTORIES FROM MOSAICS FOR SPORT STATISTICS

- Mosaic is based on *rigid-transformations (rotation, scale and translation).* Preserving the shape of individual frames.
- Mosaic is manually pre-constructed (*bootstrap* problem). Automatic construction relies on frame registration.

CONSTRUCTING PLAYER TRAJECTORIES FROM MOSAICS FOR SPORT STATISTICS

Projected on the mosaic according to their registration parameters produced by Fourier-Mellin Registration.

CONSTRUCTING PLAYER TRAJECTORIES FROM MOSAICS FOR SPORT STATISTICS



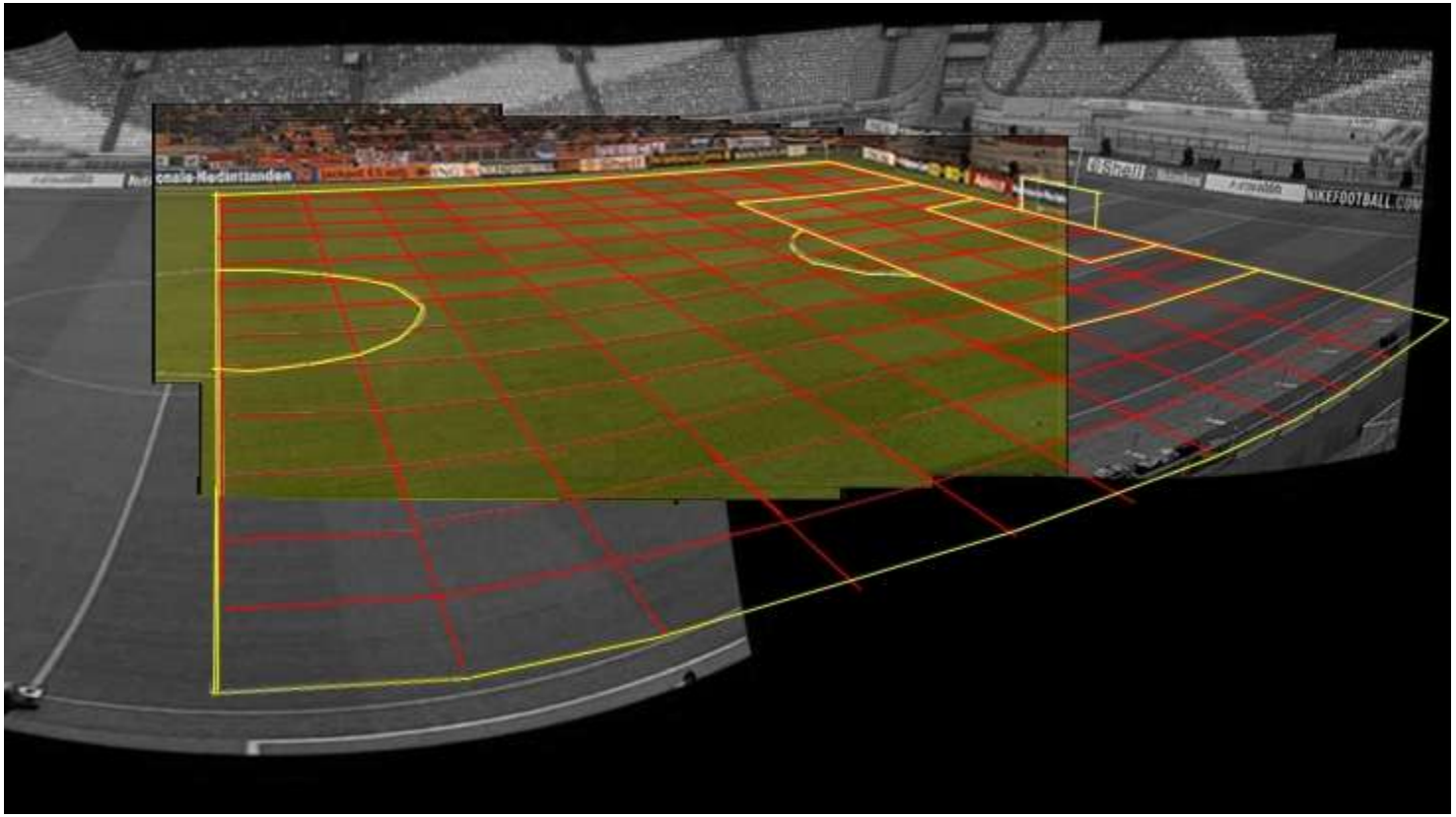
•Trajectories of Dutch players, colour-coded in orange. The ground- truth is also shown in grey dots.

•The sub-mosaics for the decomposition are represented by the coloured rectangles.

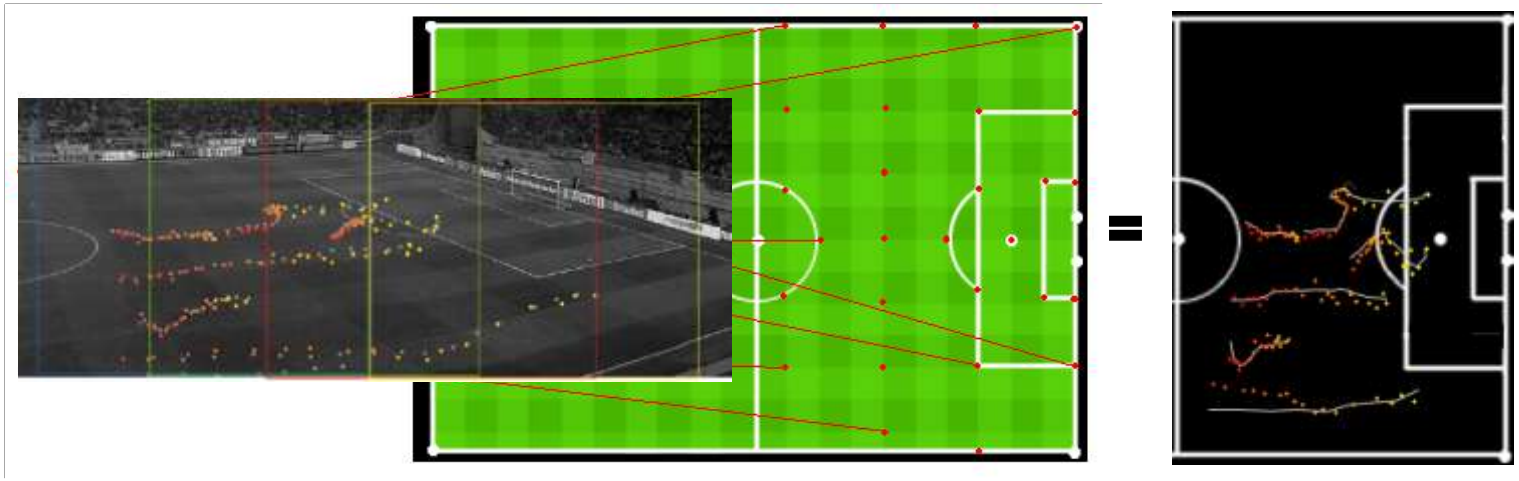
CONSTRUCTING PLAYER TRAJECTORIES FROM MOSAICS FOR SPORT STATISTICS

- Several frames projected on the mosaic, according to their recovered registration parameters.
- Showing ‘ghosts’ of players is very illustrative

CONSTRUCTING PLAYER TRAJECTORIES FROM MOSAICS FOR SPORT STATISTICS



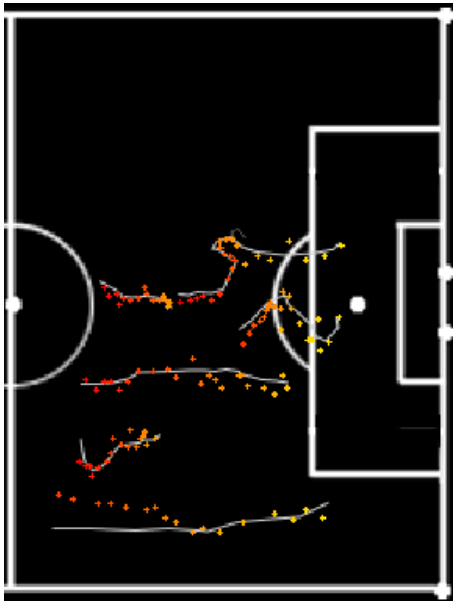
Homography Transform Phase



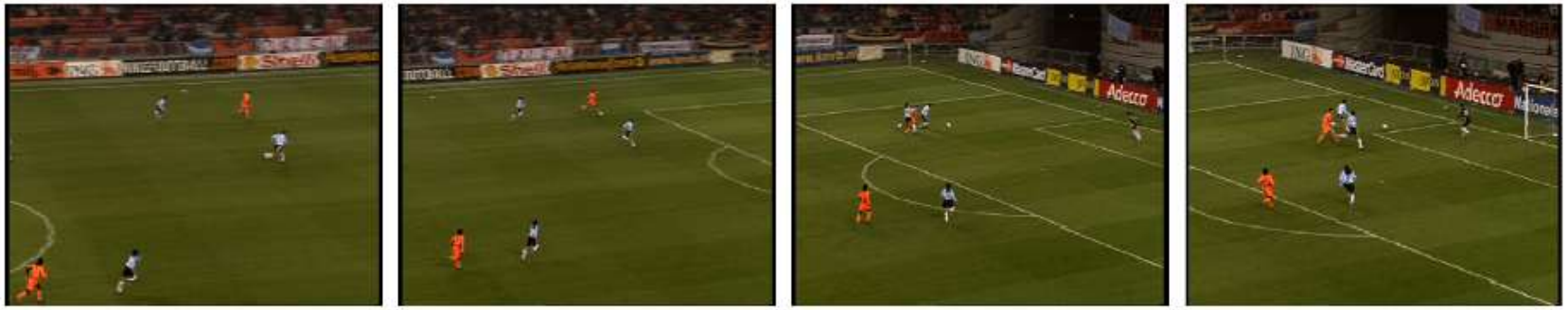
- After iteratively plotting the foot-positions of each frame a trajectory plot is constructed. Distinctive or salient features are selected and mapped to the geometrically correct line-model. Finally, conversion to an orthogonal perspective using a homography is performed.

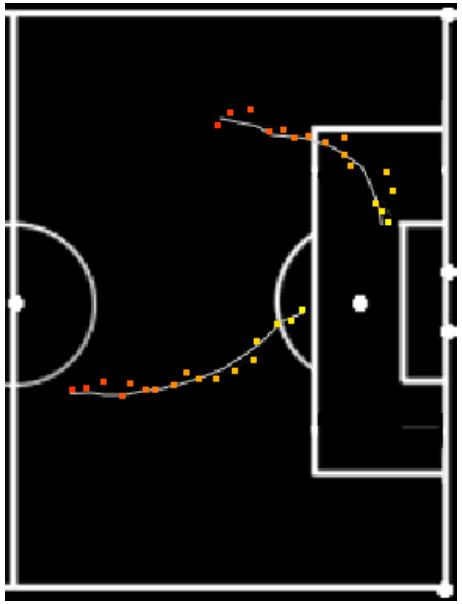
CONSTRUCTING PLAYER TRAJECTORIES FROM MOSAICS FOR SPORT STATISTICS



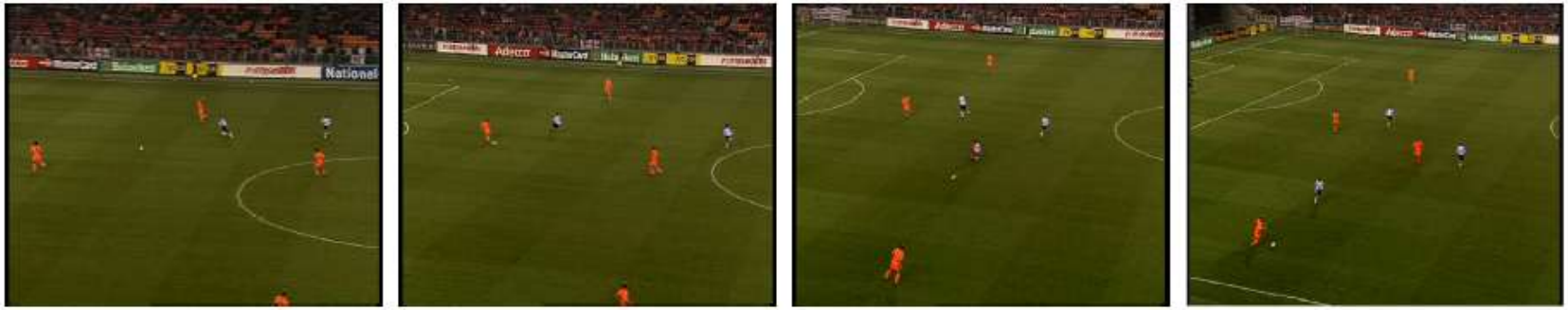



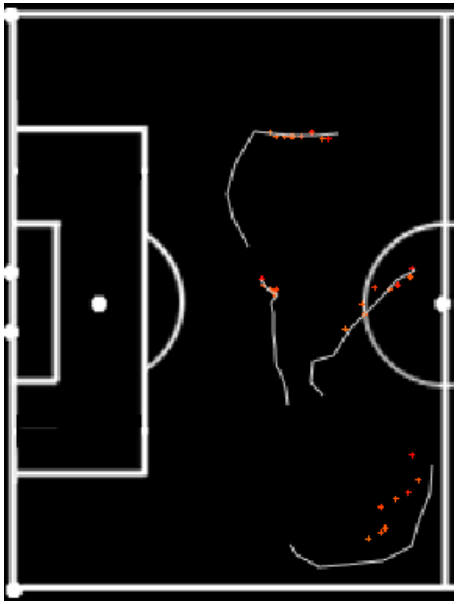
CONSTRUCTING PLAYER TRAJECTORIES FROM MOSAICS FOR SPORT STATISTICS





CONSTRUCTING PLAYER TRAJECTORIES FROM MOSAICS FOR SPORT STATISTICS





CONSTRUCTING PLAYER TRAJECTORIES FROM MOSAICS FOR SPORT STATISTICS

Overview

Mosaicking

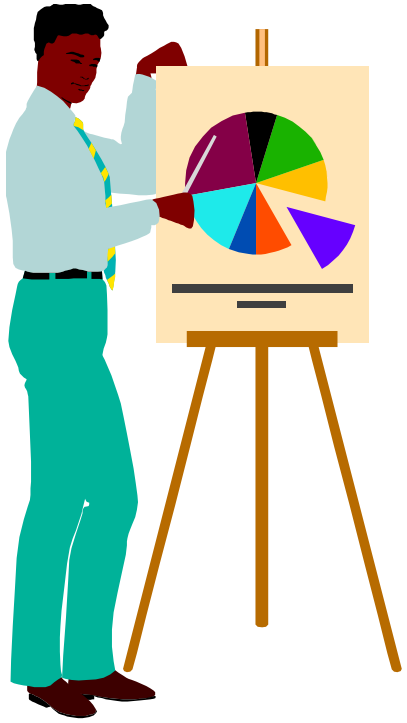
Object Recognition in Perspective

Image Descriptors

Bag-of-Models

Classifiers

Object Recognition Benchmarks

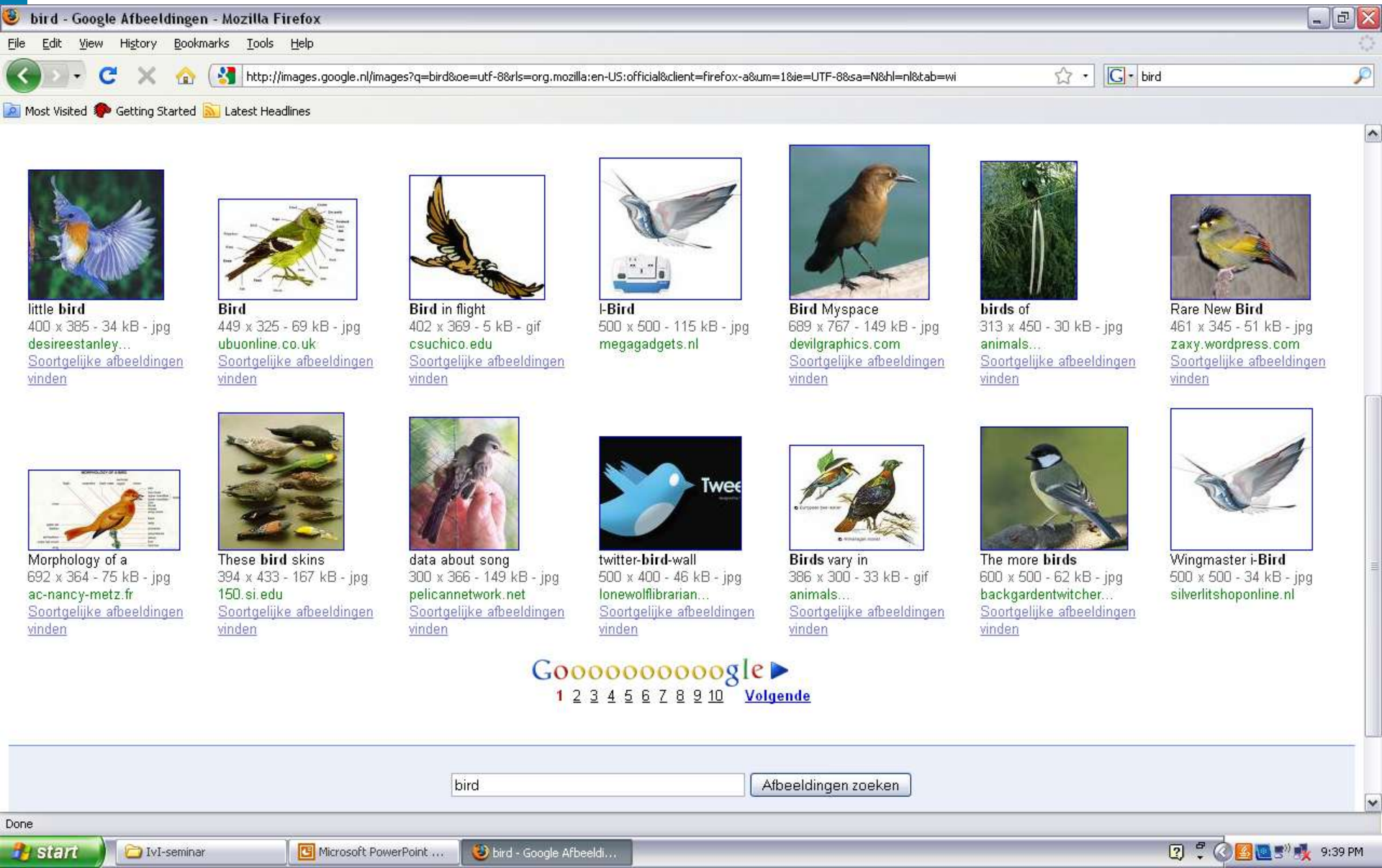


The Motivation

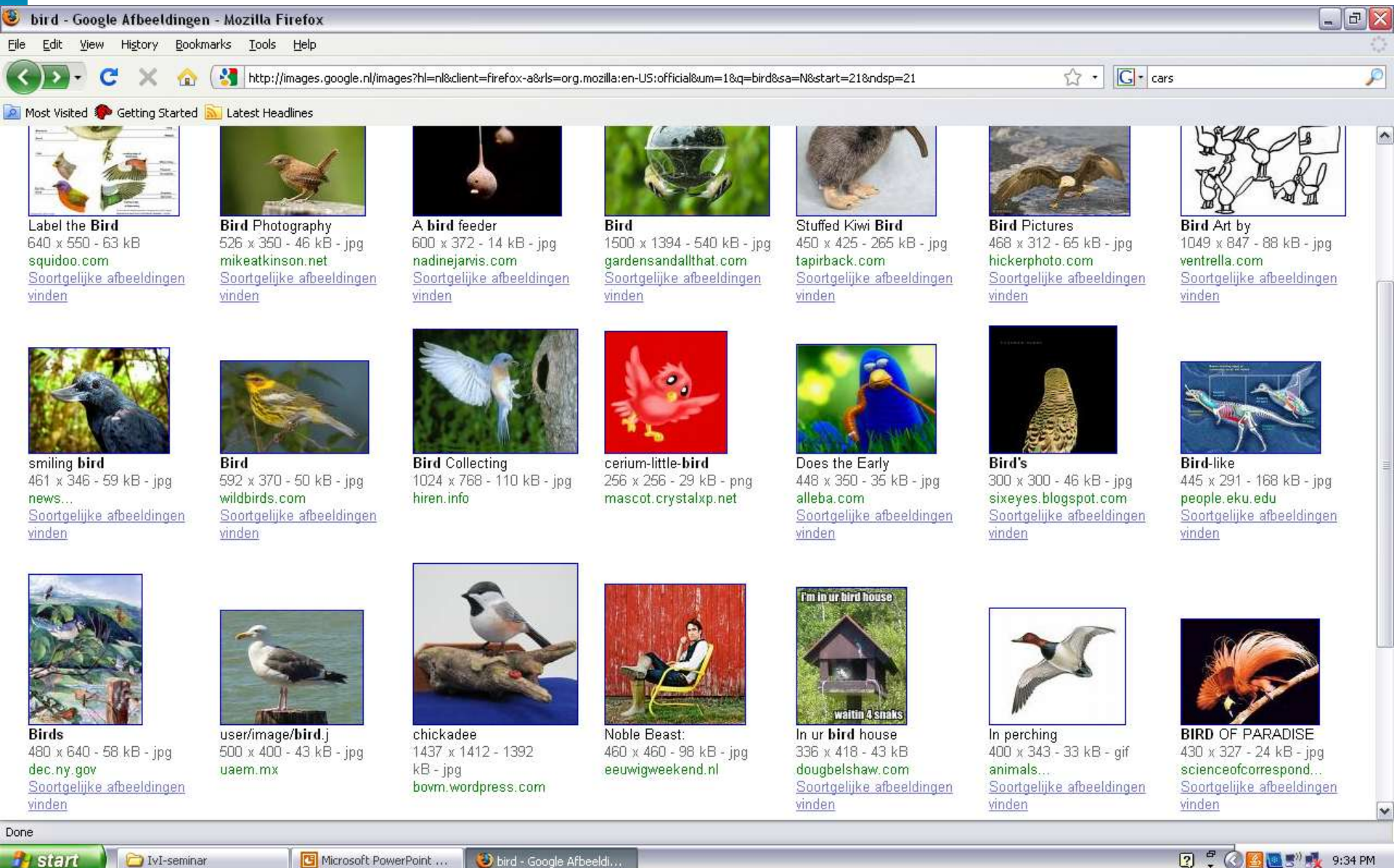
- Today, there are billions of images on the Internet and in collections such as FaceBook and Flickr.

- Suppose I want to find pictures of birds, humans, cars, boats or videos of explosion, violence etc

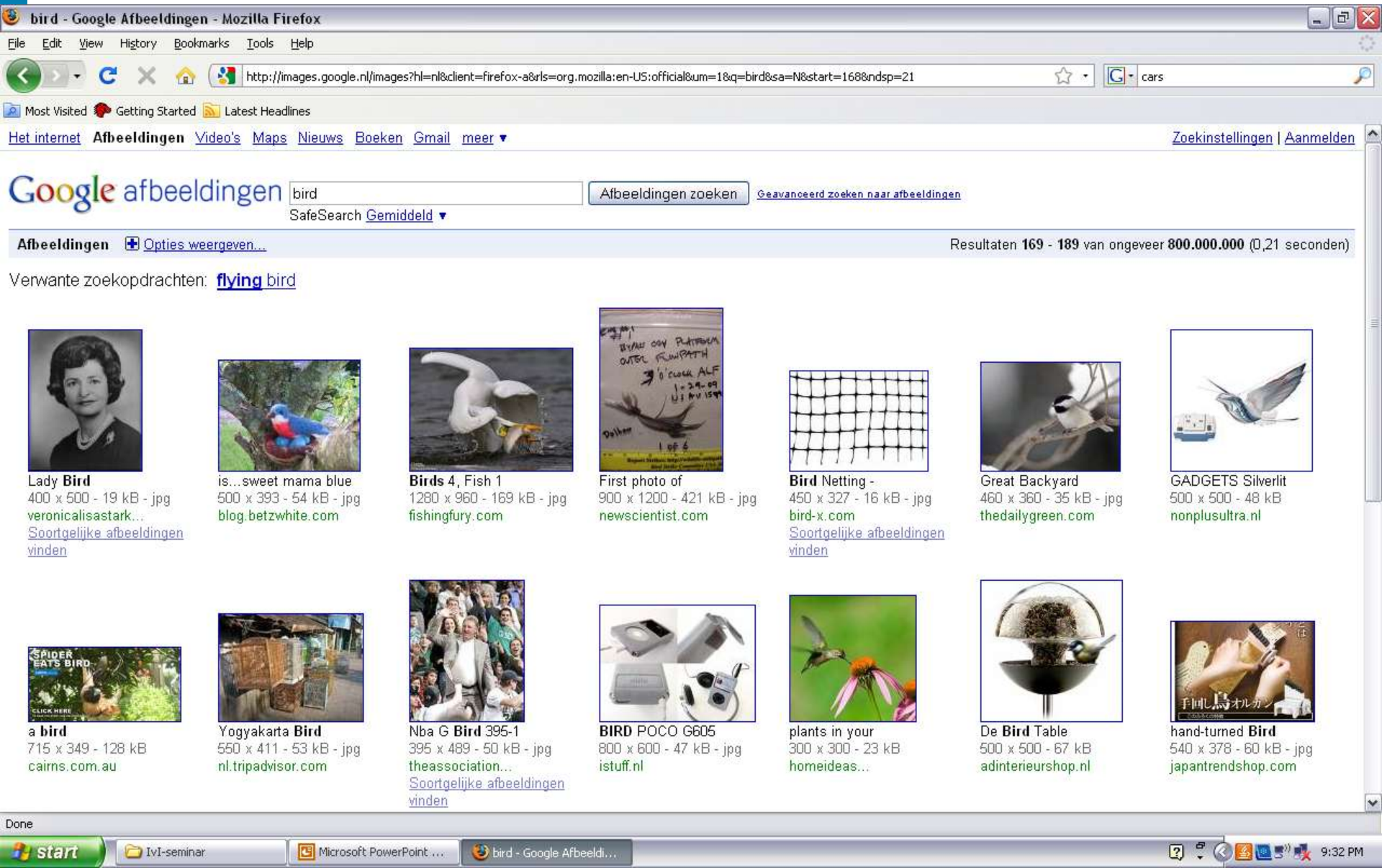
Google Image Search – Bird(1)



Google Image Search – Bird(2)



Google Image Search – Bird(3)



Video Retrieval

Given a shot from a video…
… is some semantic *concept* present in that shot?

Example concepts:

- Airplane
- Building
- **Car**
- Crowd
- **Desert**
- **Explosion**
- **Outdoor**
- People
- **Vehicle**
- **Violence**

Object/Scene Categories


Aircraft


Animal


Boat


Building


Bus


Car


Chart


Corp. leader


Court


Crowd

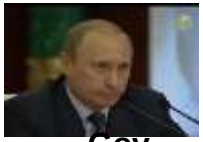

Desert


Entertainment


Explosion


Face


Flag USA


Gov. leader


Map


Meeting


Military


Mountain


Natural disaster


Office


Outdoor


People


People marching


Police / security


Prisoner


Screen


Sky


Sports


Studio


Truck

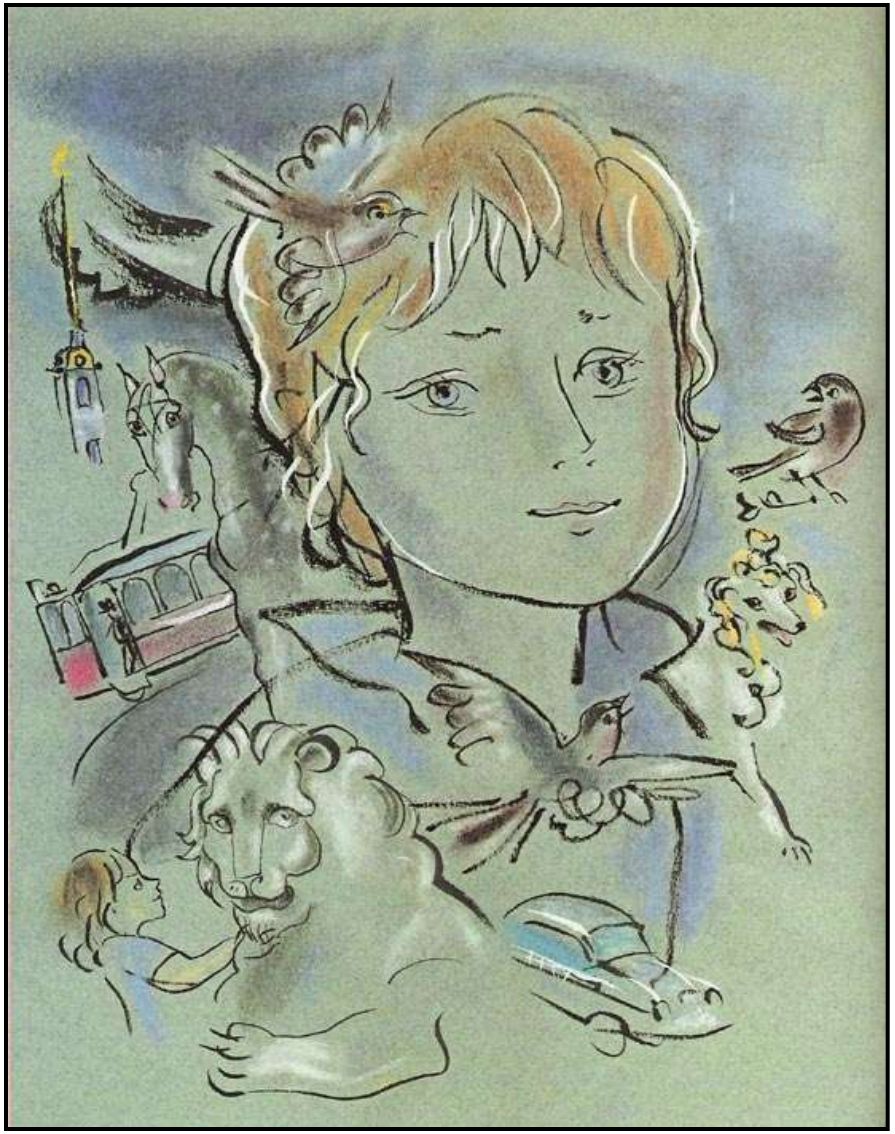

Urban


Vegetation


Vehicle

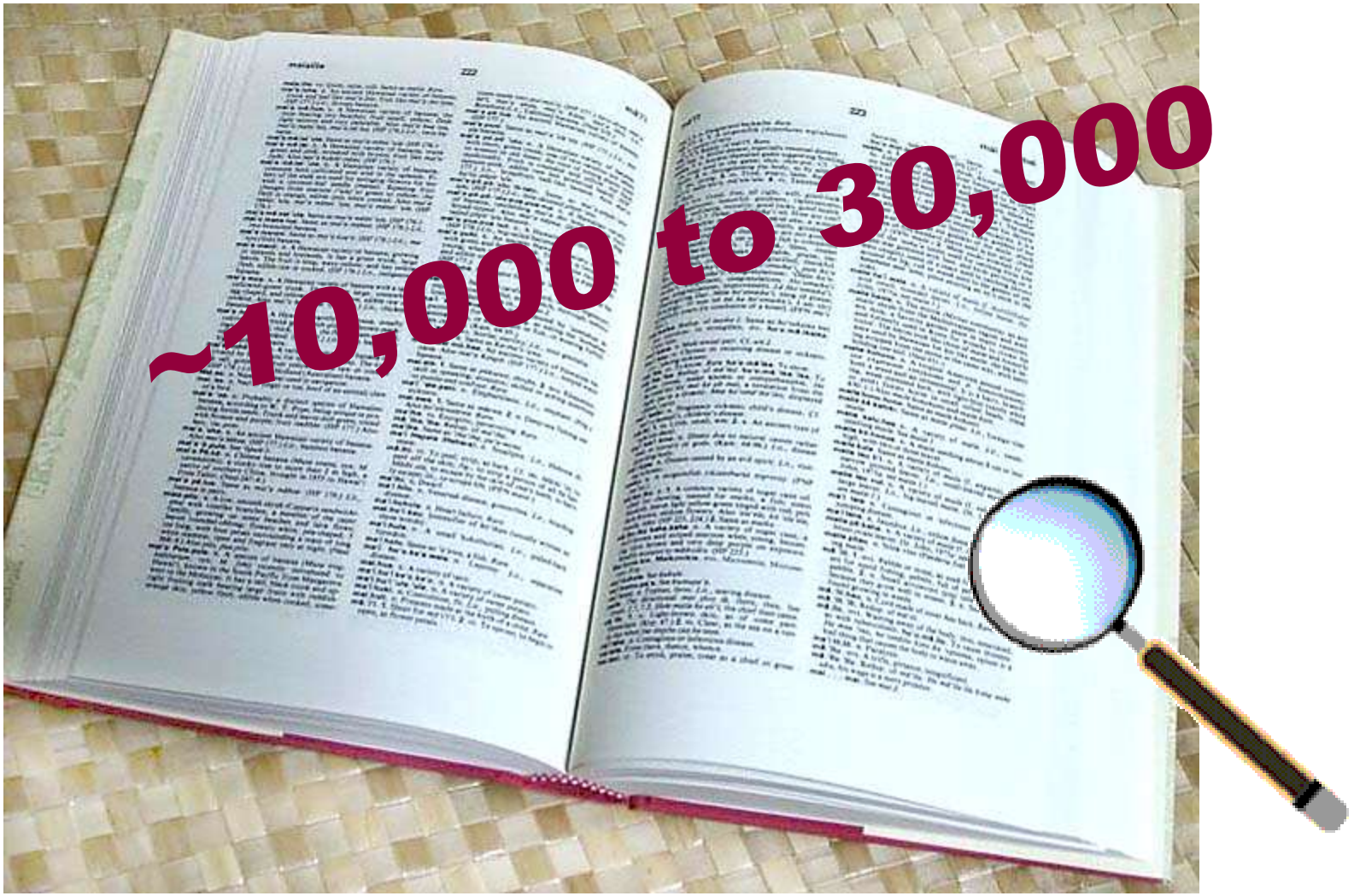

Violence

Recognition: Overview and History



Slides from Lana Lazebnik, Fei-Fei Li, Rob Fergus, Antonio Torralba, and Jean Ponce

How many visual object categories are there?



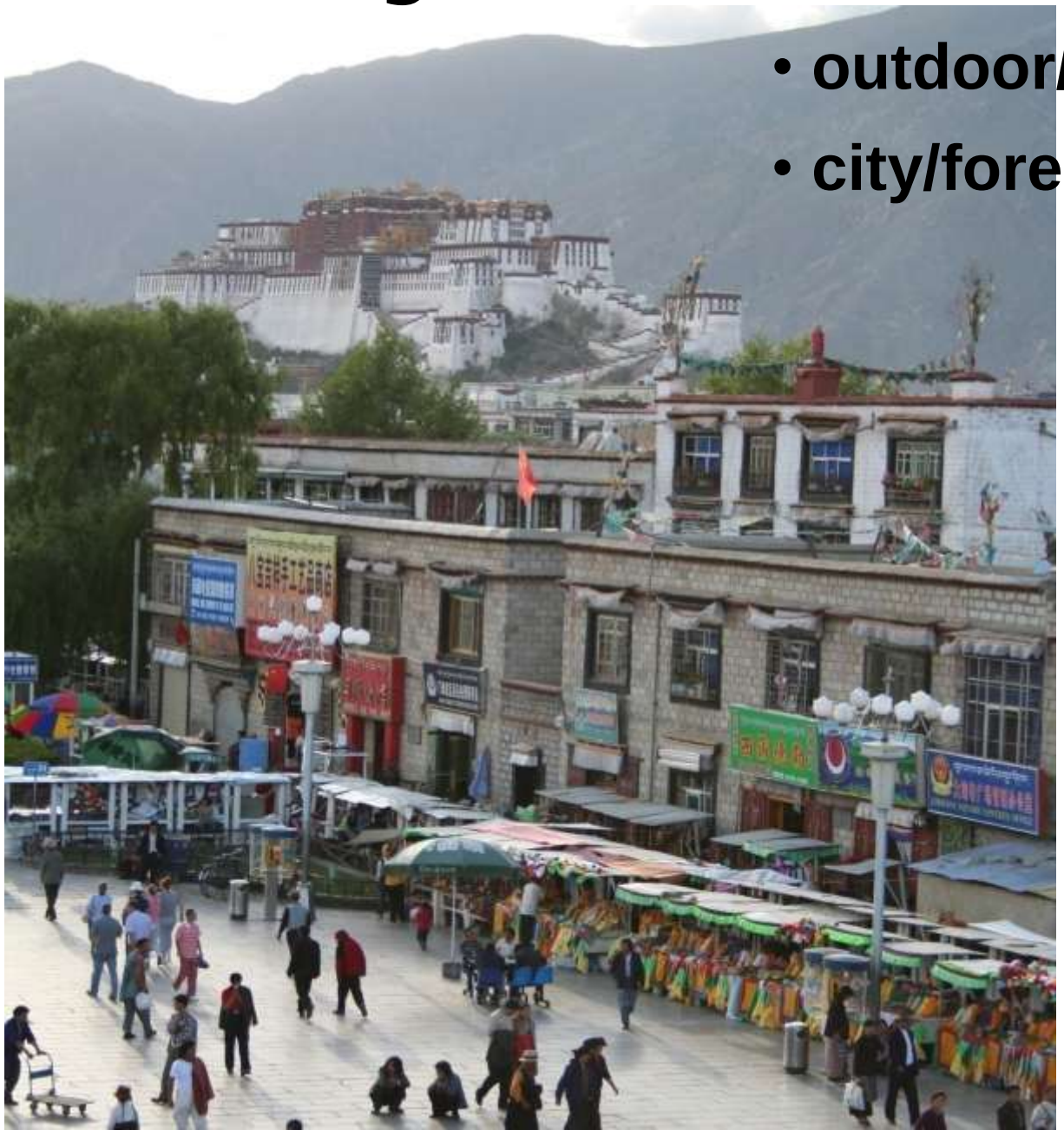
Biederman 1987


~10,000 to 30,000

OBJECTS

- ANIMALS
  - .....
  - VERTEBRATE
    - MAMMALS
      - TAPIR
      - BOAR
    - BIRDS
      - GROUSE
- PLANTS
- INANIMATE
  - NATURAL
  - MAN-MADE
    - CAMERA

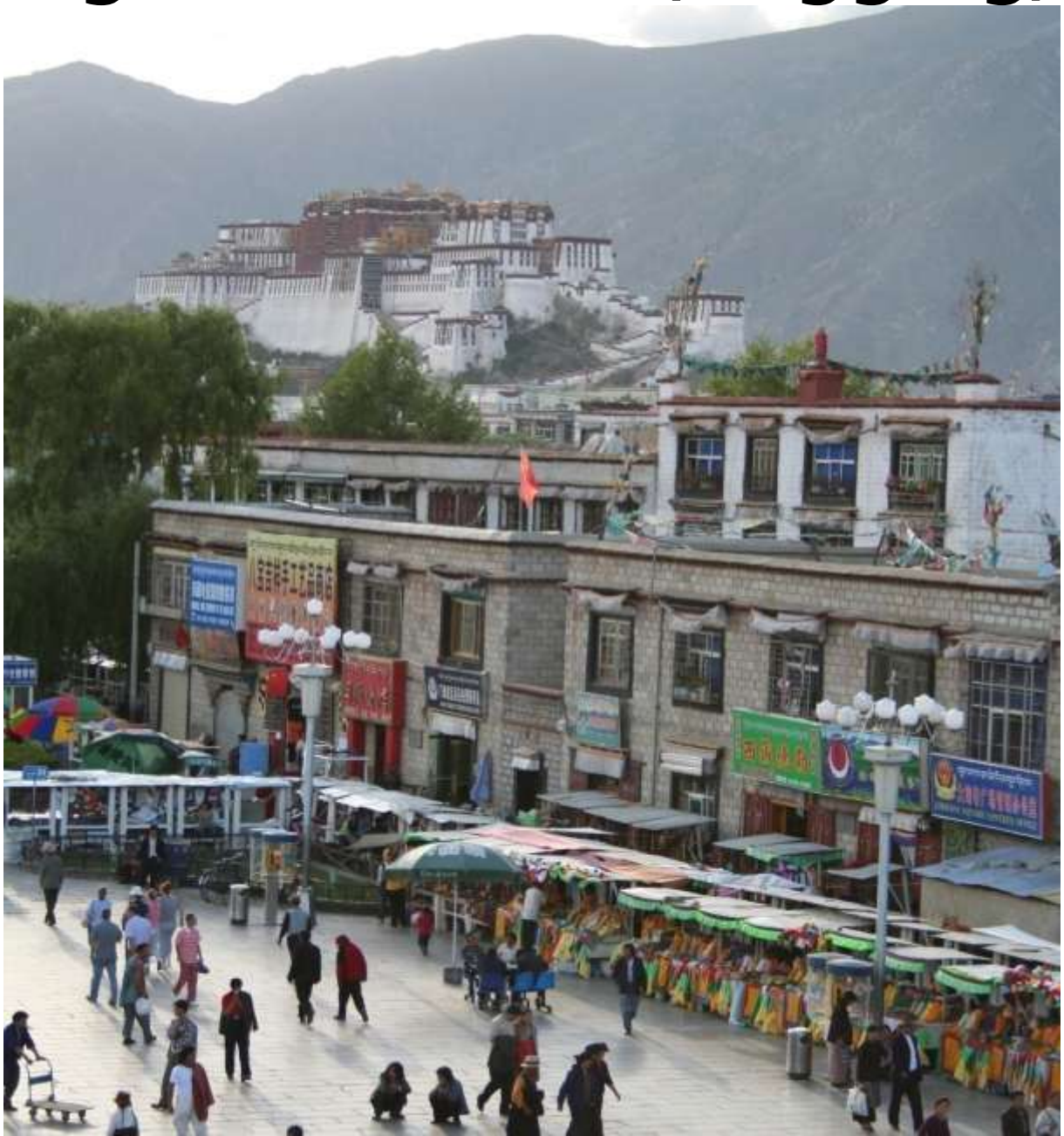
Scene categorization or classification



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

Svetlana Lazebnik

Image annotation/tagging/attributes



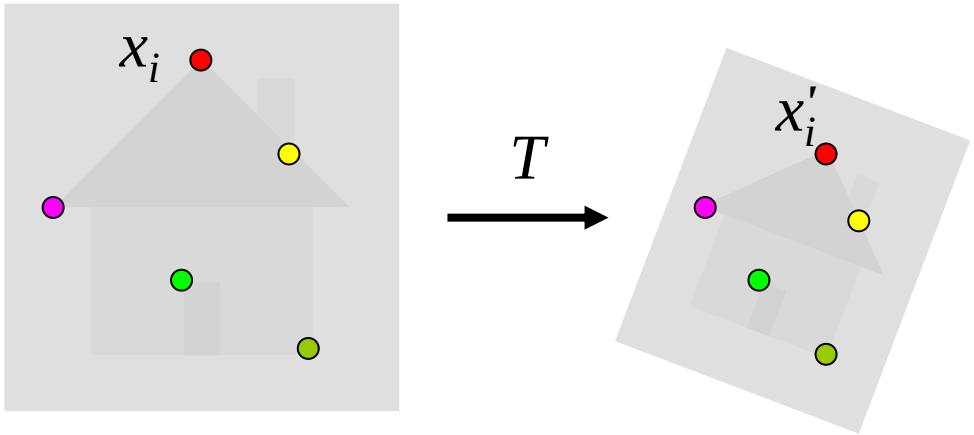
- **street**
- **people**
- **building**
- **mountain**
- **tourism**
- **cloudy**
- **brick**
- **…**

History of Ideas in Object Recognition

- 1960s – early 1990s: the geometric era

Alignment

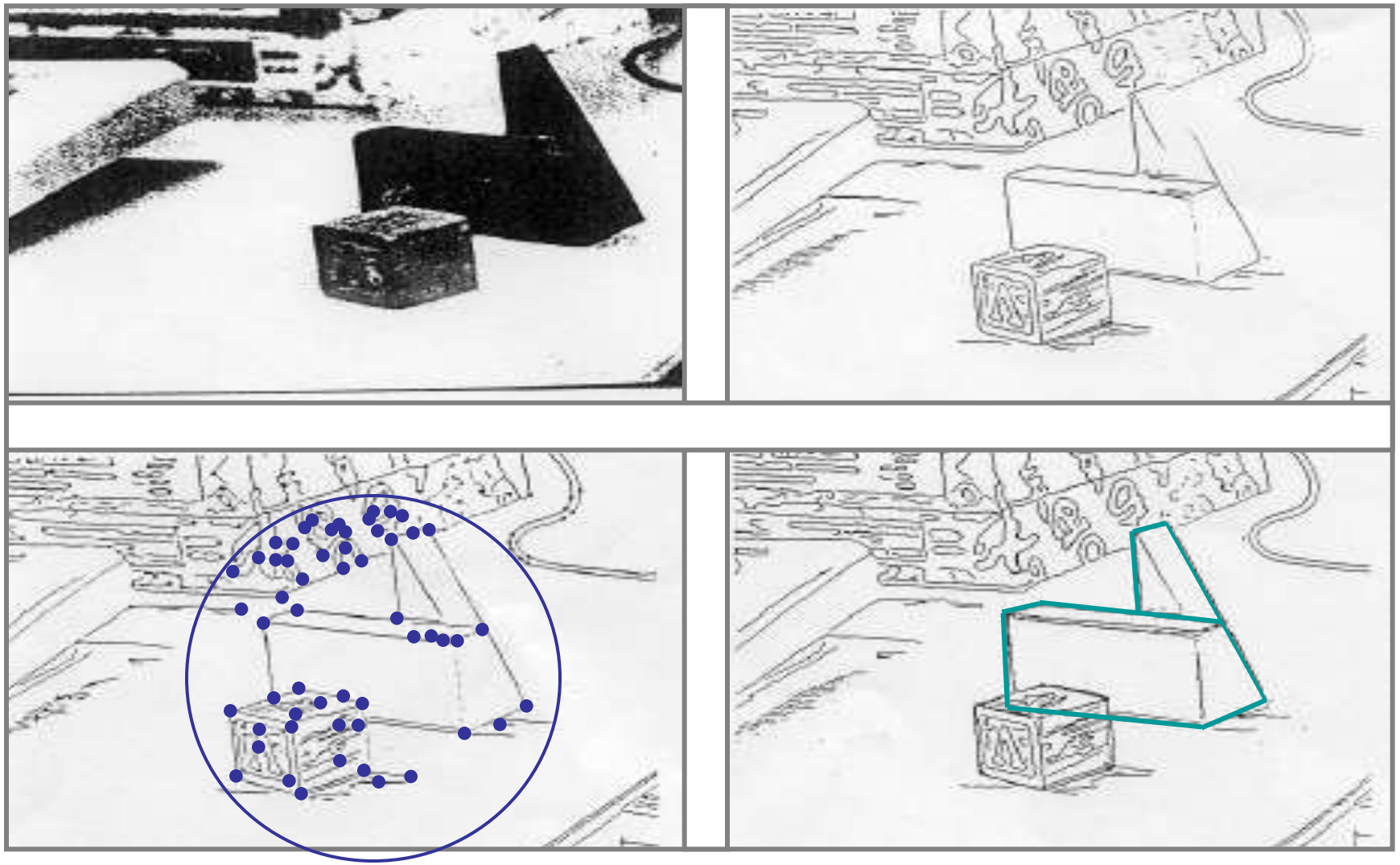
- Alignment: fitting a model to a transformation between pairs of features (*matches*) in two images



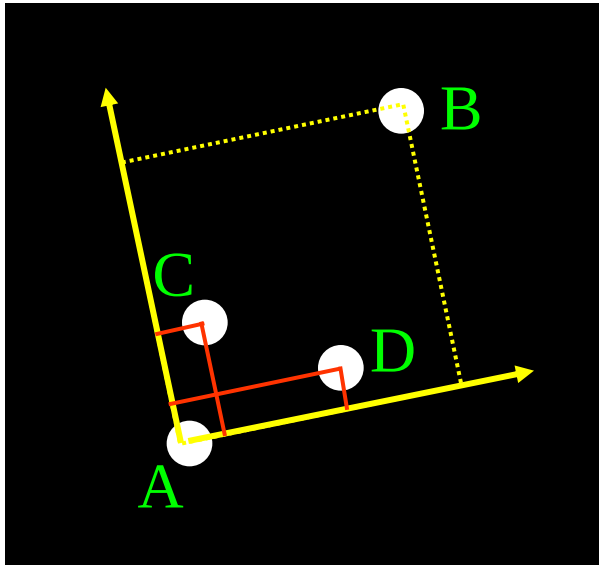
Find transformation T that minimizes

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

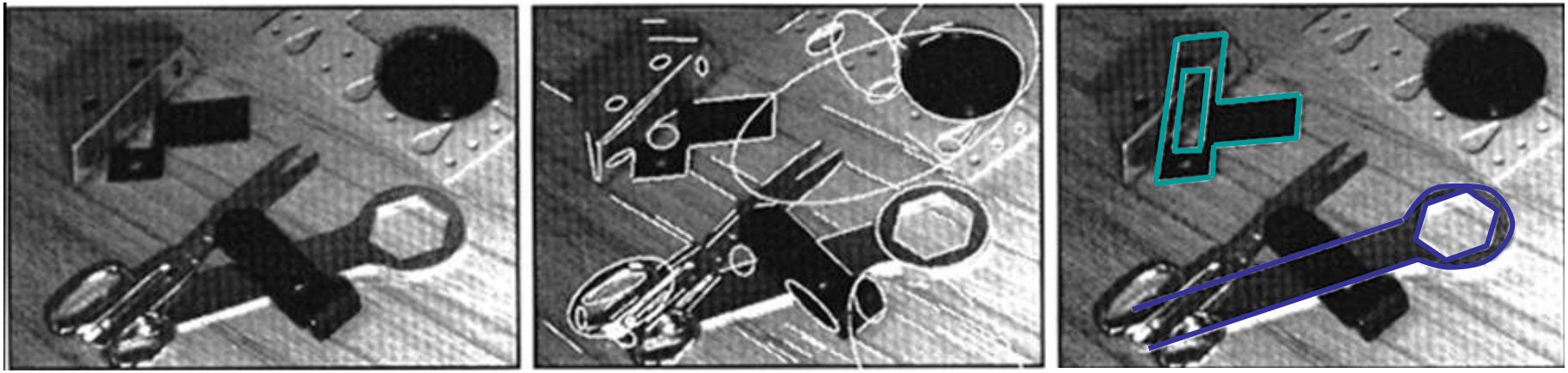
Alignment: Huttenlocher & Ullman (1987)



Example: invariant to similarity transformations computed from four points



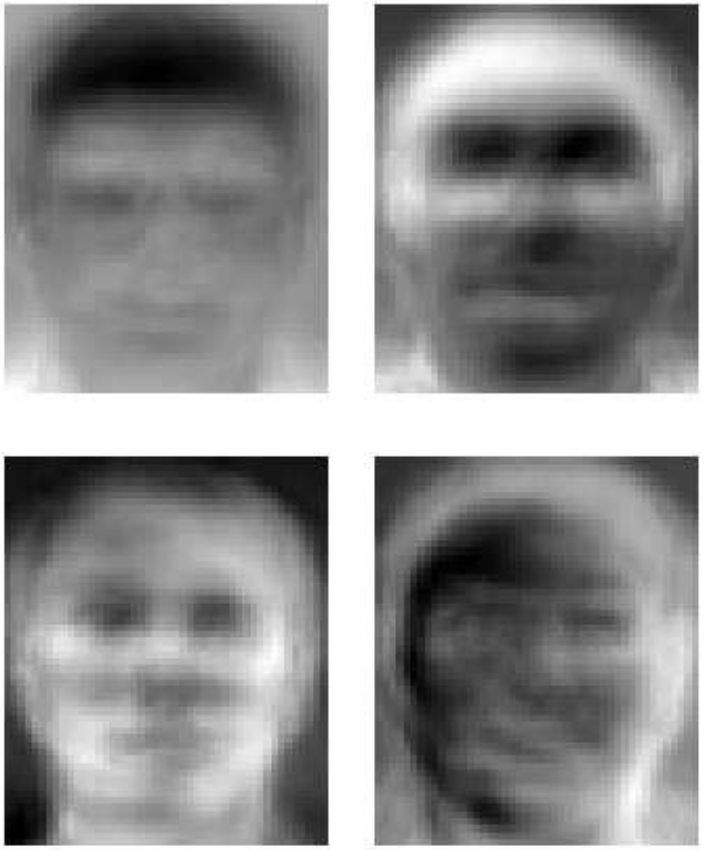
Projective invariants (Rothwell et al., 1992):



History of Ideas in Object Recognition

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

Eigenfaces (Turk & Pentland, 1991)

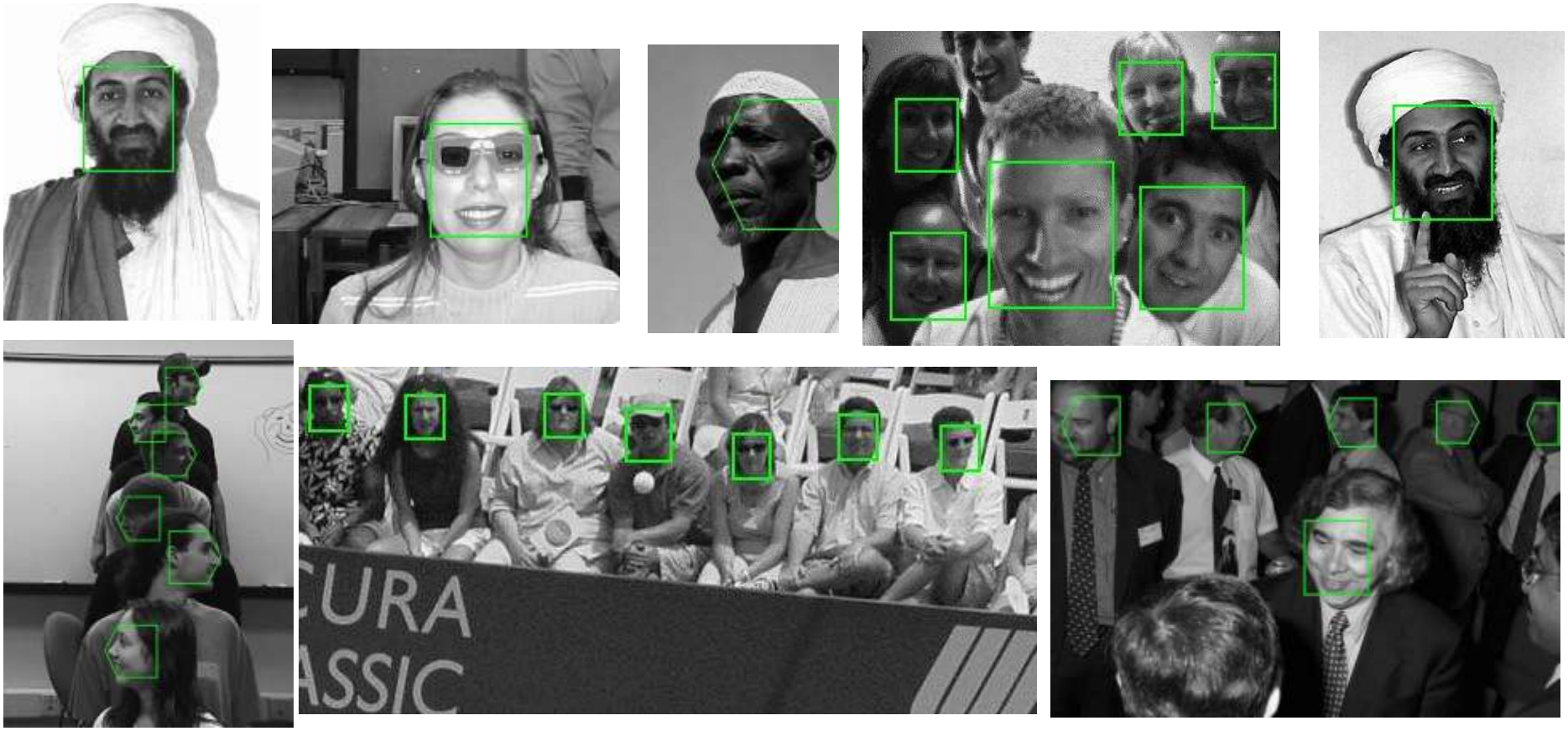


Experimental	Correct/Unknown Recognition Percentage		
Condition	Lighting	Orientation	Scale
Forced classification	96/0	85/0	64/0
Forced 100% accuracy	100/19	100/39	100/60
Forced 20% unknown rate	100/20	94/20	74/20

Svetlana Lazebnik

Face Detection

Schneiderman & Kanade (CMU), 2000…



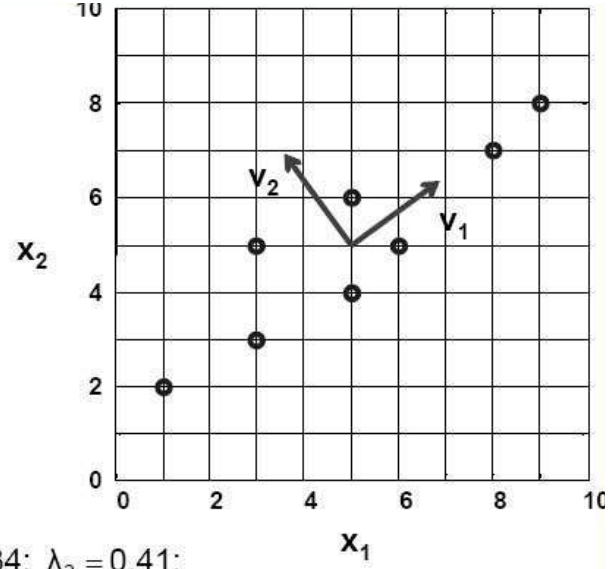
Results on various images submitted to the CMU on-line face detector

PCA Algorithm

- PCA algorithm:
  - 1. X ← Create N x d data matrix, with one row vector x_n per data point
  - 2. X subtract mean *x* from each row vector x_n in X
  - 3. Σ ← covariance matrix of X
  - Find eigenvectors and eigenvalues of Σ
  - PC's ← the M eigenvectors with largest eigenvalues

Principal Component Analysis (PCA) Example

- **Compute the principal components for the following two-dimensional dataset**
  - $X=(x_1,x_2)=\{(1,2),(3,3),(3,5),(5,4),(5,6),(6,5),(8,7),(9,8)\}$
    - Let's first plot the data to get an idea of which solution we should expect



- **SOLUTION (by hand)**
  - The (biased) covariance estimate of the data is:

$$\Sigma_x = \begin{bmatrix} 6.25 & 4.25 \\ 4.25 & 3.5 \end{bmatrix}$$

  - The eigenvalues are the zeros of the characteristic equation

$$\Sigma_x v = \lambda v \Rightarrow |\Sigma_x - \lambda I| = 0 \Rightarrow \begin{vmatrix} 6.25-\lambda & 4.25 \\ 4.25 & 3.5-\lambda \end{vmatrix} = 0 \Rightarrow \lambda_1 = 9.34;\ \lambda_2 = 0.41;$$

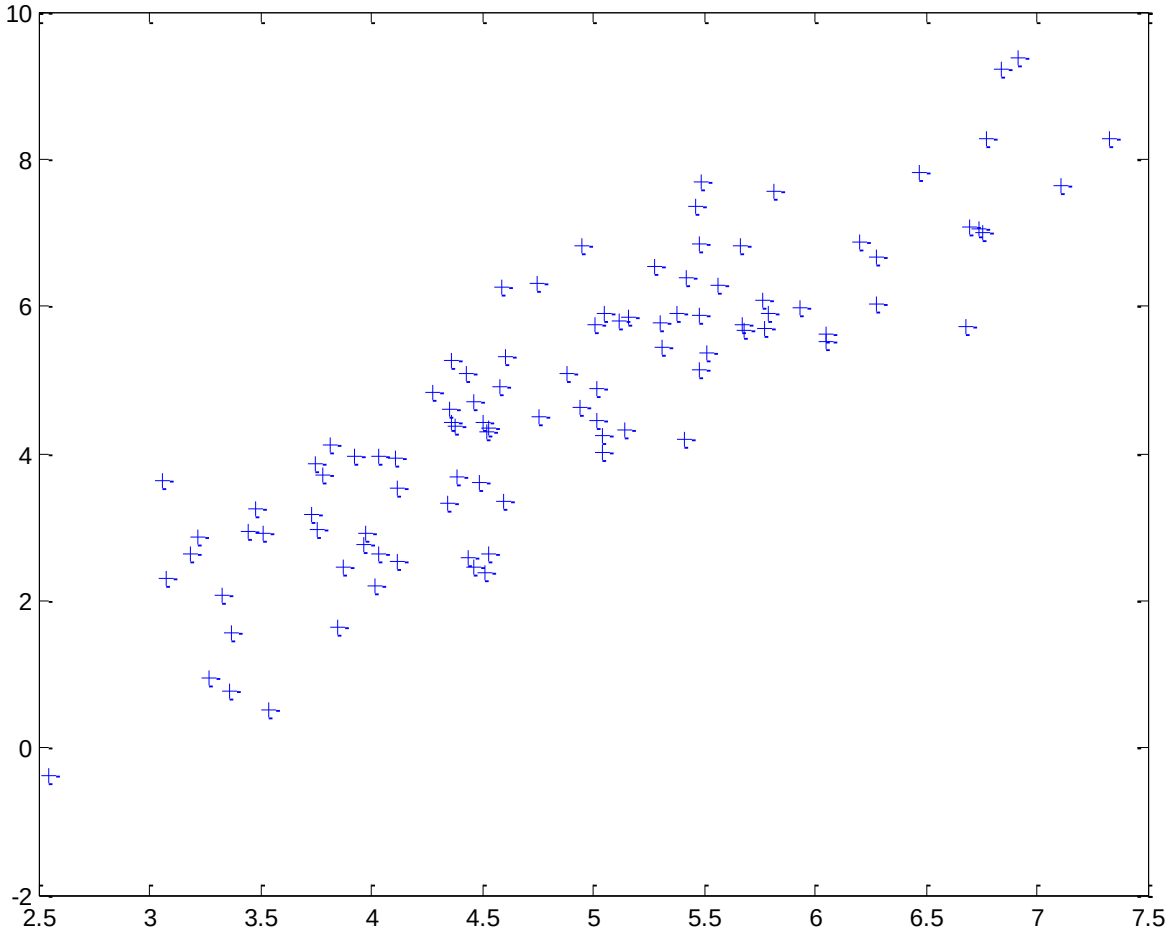
  - The eigenvectors are the solutions of the system

$$\begin{bmatrix} 6.25 & 4.25 \\ 4.25 & 3.5 \end{bmatrix}\begin{bmatrix} v_{11} \\ v_{12} \end{bmatrix} = \begin{bmatrix} \lambda_1 v_{11} \\ \lambda_1 v_{12} \end{bmatrix} \Rightarrow \begin{bmatrix} v_{11} \\ v_{12} \end{bmatrix} = \begin{bmatrix} 0.81 \\ 0.59 \end{bmatrix}$$

$$\begin{bmatrix} 6.25 & 4.25 \\ 4.25 & 3.5 \end{bmatrix}\begin{bmatrix} v_{21} \\ v_{22} \end{bmatrix} = \begin{bmatrix} \lambda_2 v_{21} \\ \lambda_2 v_{22} \end{bmatrix} \Rightarrow \begin{bmatrix} v_{21} \\ v_{22} \end{bmatrix} = \begin{bmatrix} -0.59 \\ 0.81 \end{bmatrix}$$

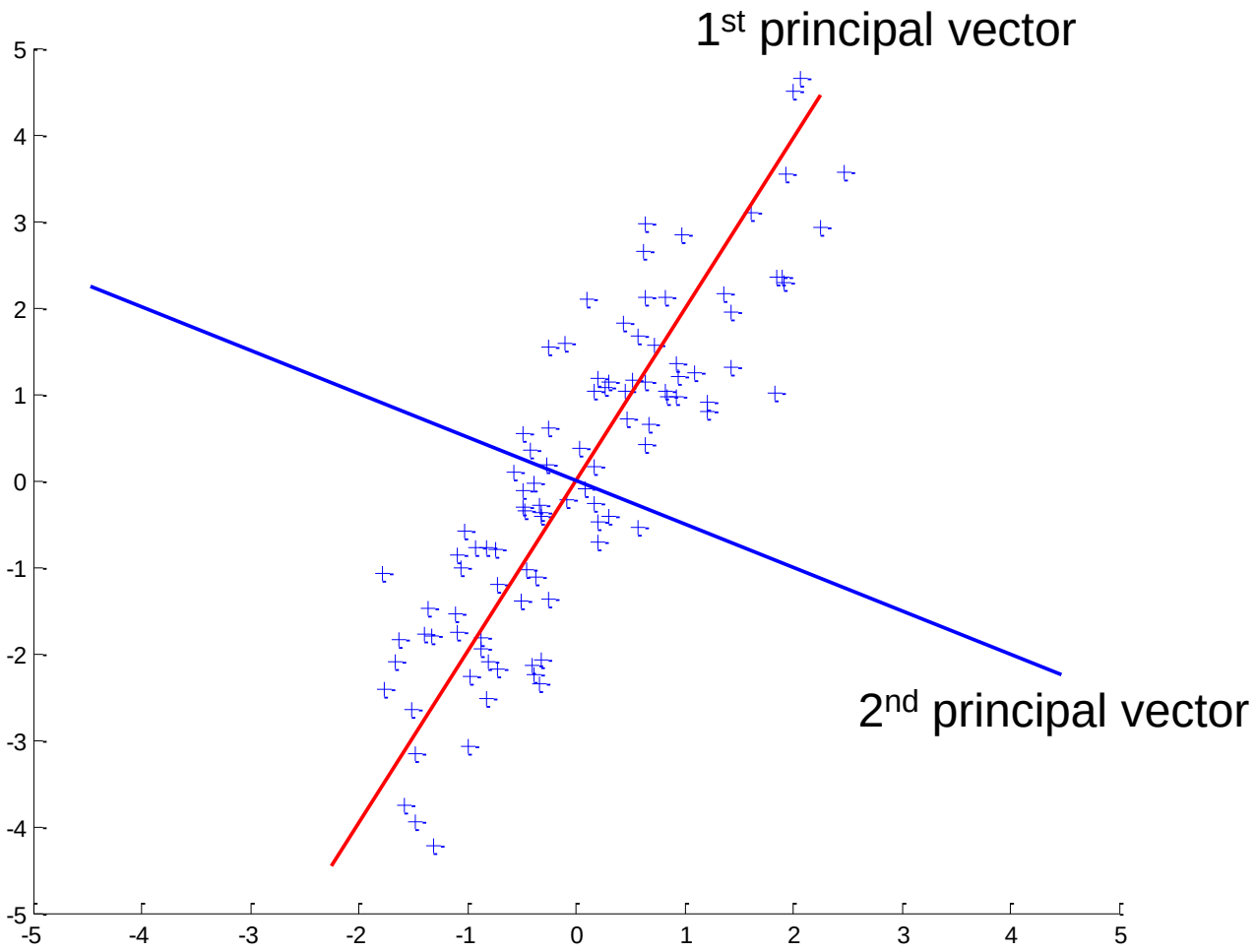
    - HINT: To solve each system manually, first assume that one of the variables is equal to one (i.e. $v_{i1}=1$), then find the other one and finally normalize the vector to make it unit-length

2d Data



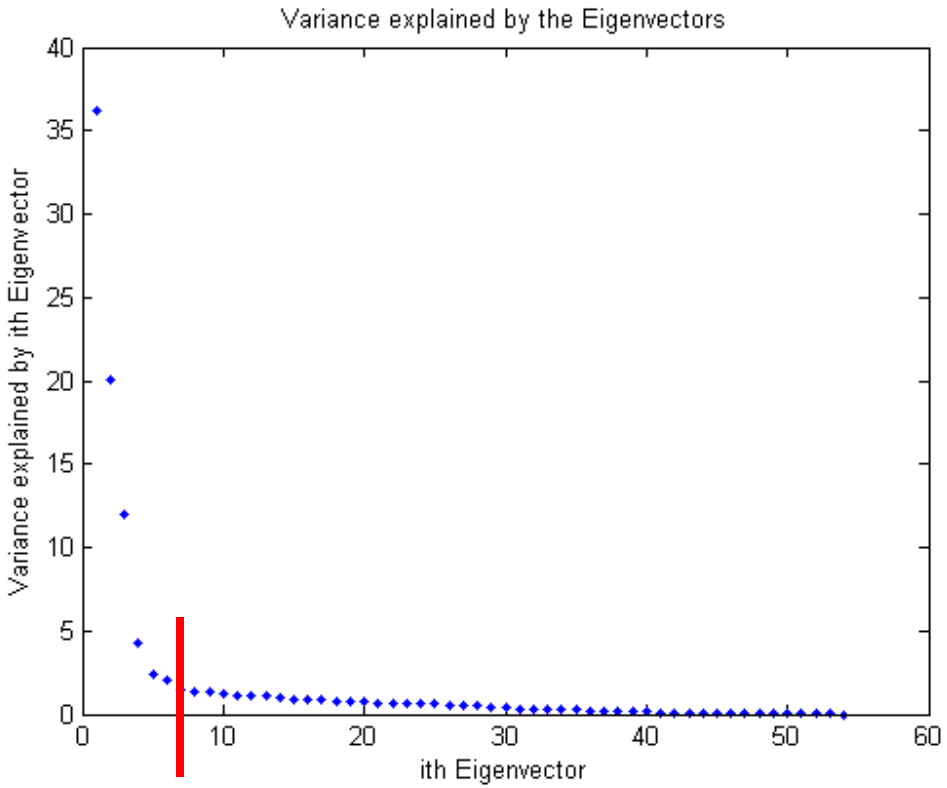
Principal Components

- Gives best axis to project
- Minimum RMS error
- Principal vectors are orthogonal



How many components?

- Check the distribution of eigen-values
- Take enough many eigen-vectors to cover 80-90% of the variance



Reconstruction from PCs

q=1

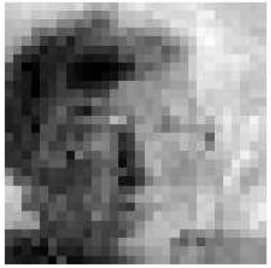
q=2

q=4

q=8

q=16

q=32

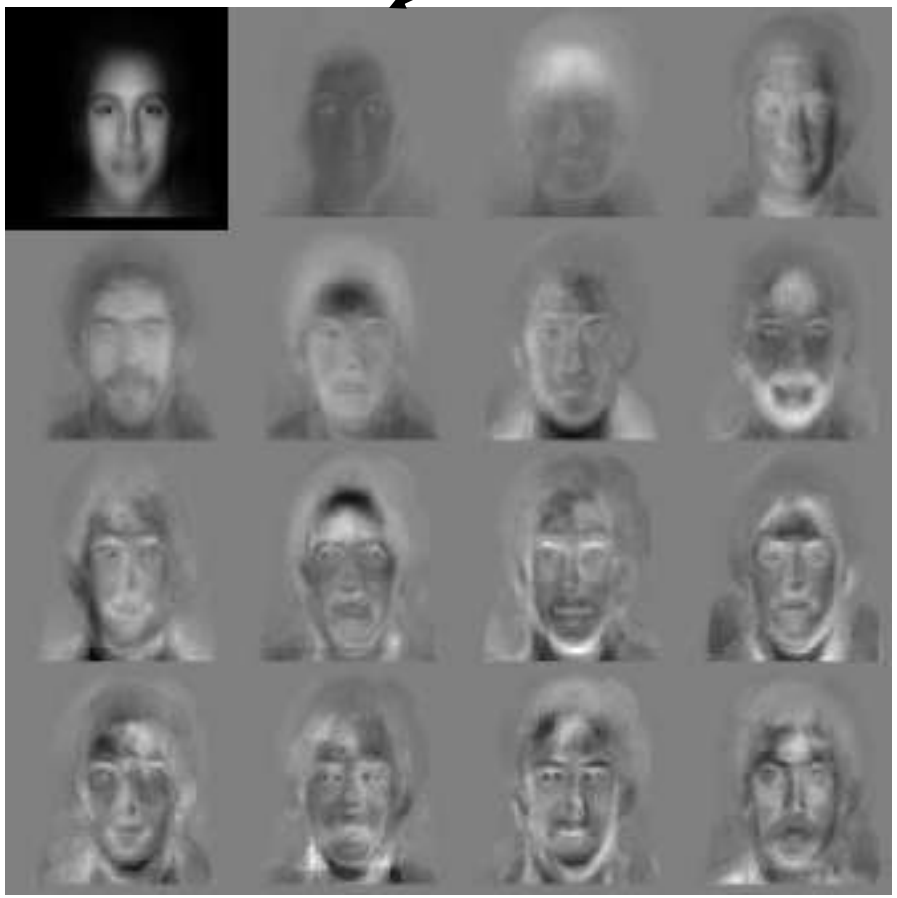


q=64

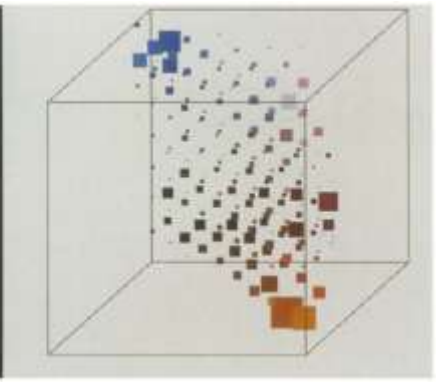
q=100…

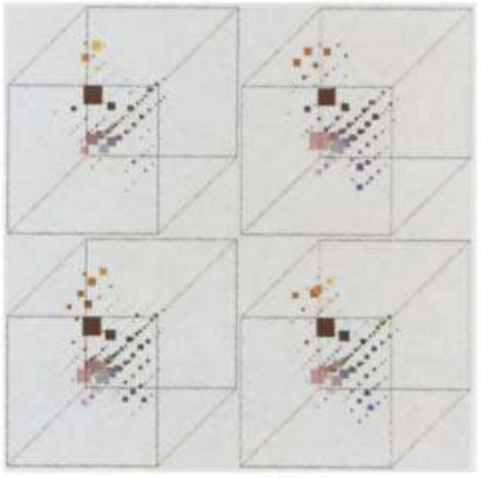
Original Image

PCA on Faces: "Eigenfaces"



Color Histograms





Swain and Ballard, Color Indexing, IJCV 1991.

Limitations of Global Appearance Models

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

History of Ideas in Object Recognition

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

Sliding Window Approaches

Sliding Window Approaches

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

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

History of Ideas in Object Recognition

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

Local Features for Object Instance Recognition

D. Lowe (1999, 2004)

Large-scale Image Search

Combining local features, indexing, and spatial constraints

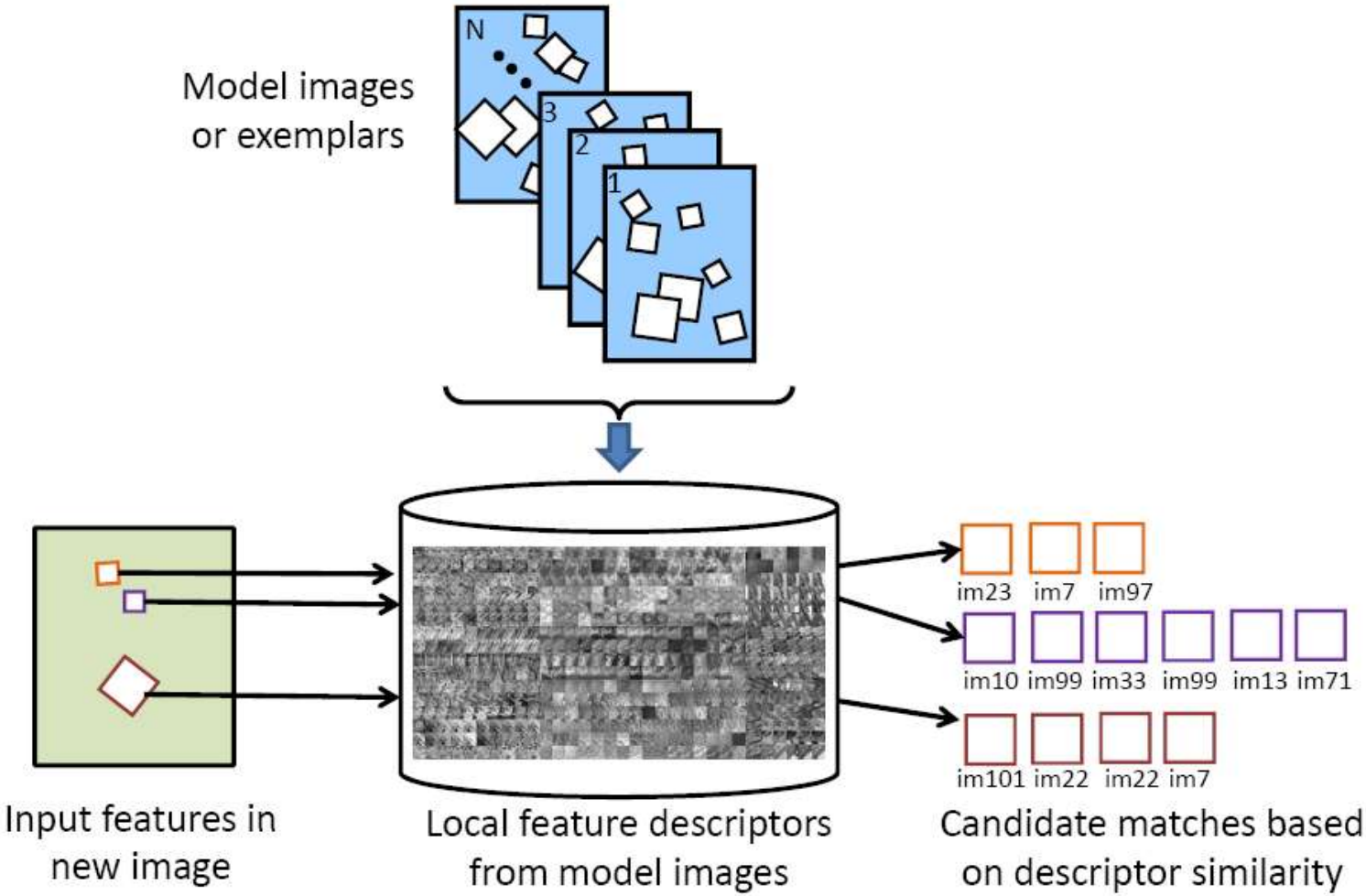
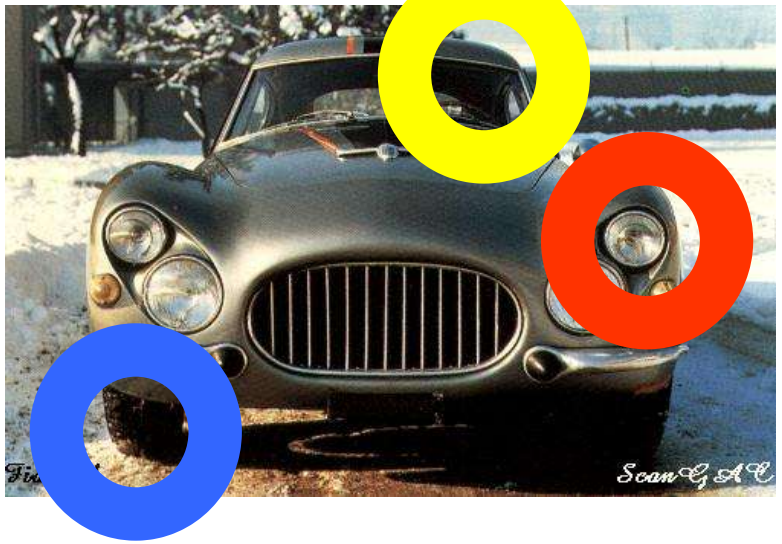


Image credit: K. Grauman and B. Leibe

History of Ideas in Object Recognition

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

Constellation Models



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

History of Ideas in Object Recognition

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

Bag-of-features Models

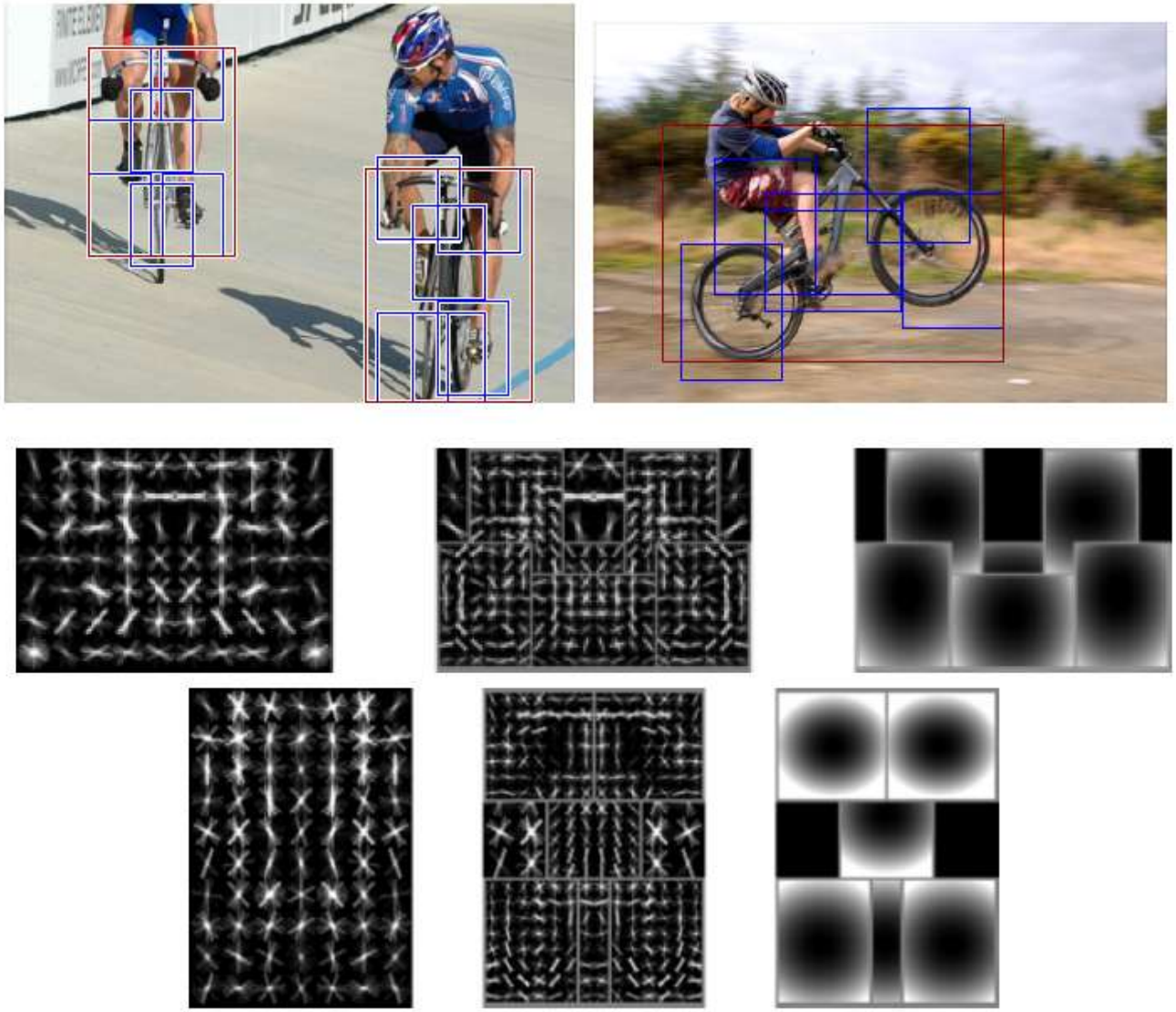
Object → **Bag of ‘words’**

Svetlana Lazebnik

History of Ideas in Object Recognition

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

Discriminatively trained part-based models



P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan, **"Object Detection with Discriminatively Trained Part-Based Models,"** PAMI 2009

Overview

Mosaicking

Object Recognition in Perspective

Image Descriptors

Bag-of-Models

Classifiers

Object Recognition Benchmarks

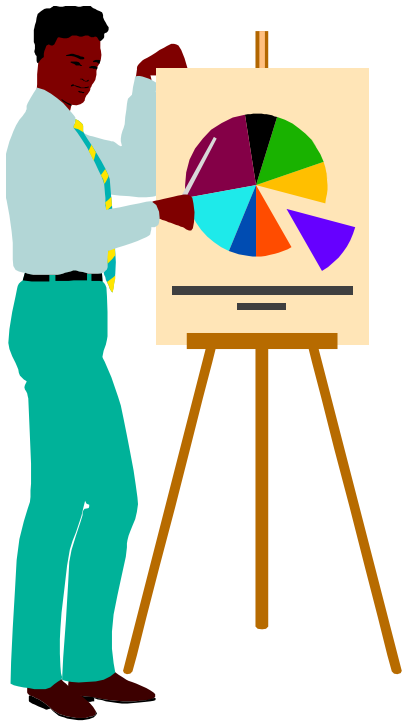
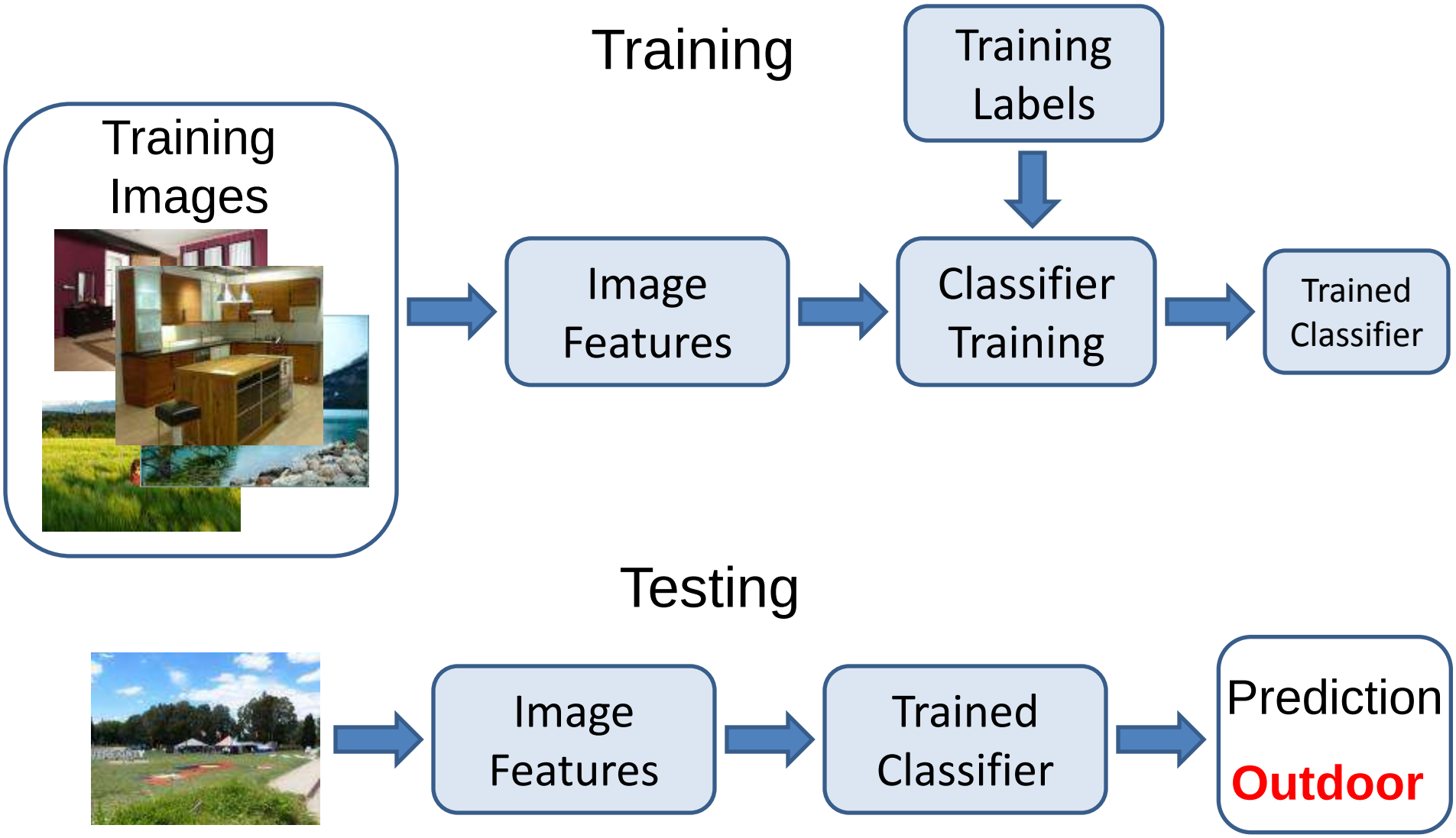


Image Categorization



Derek Hoiem

Part 1: Image features

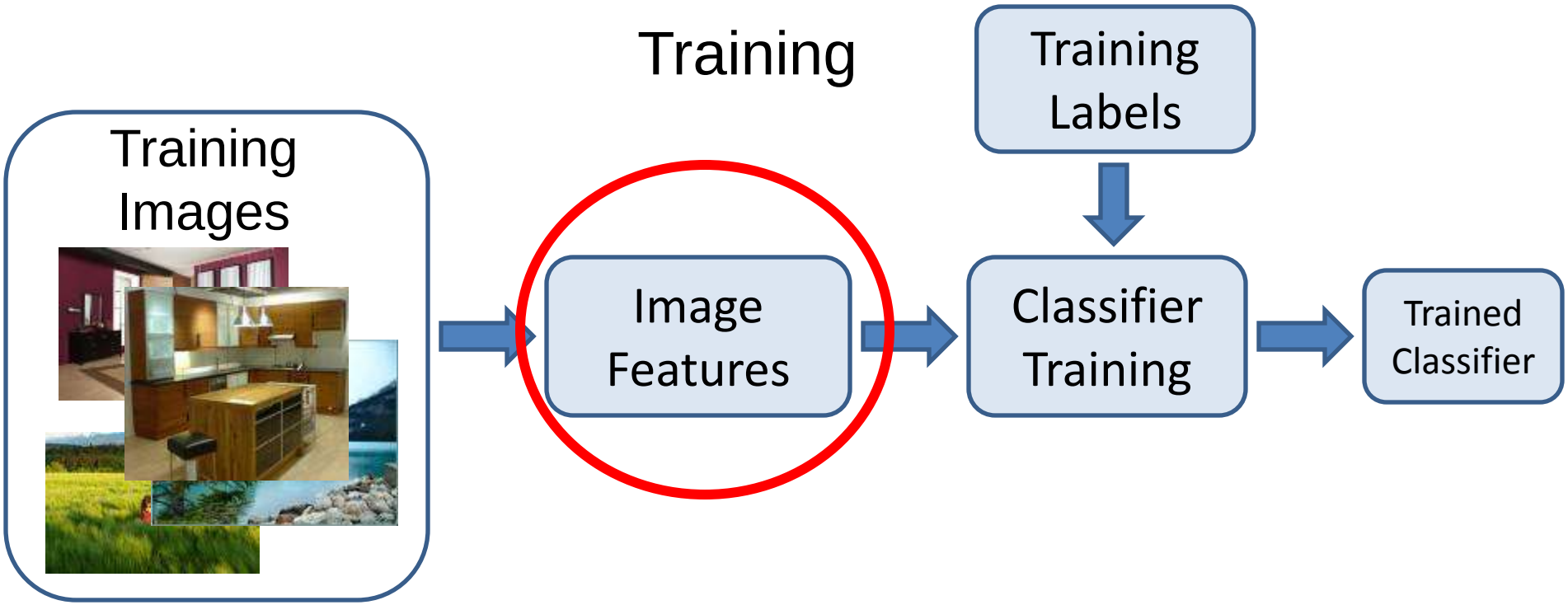
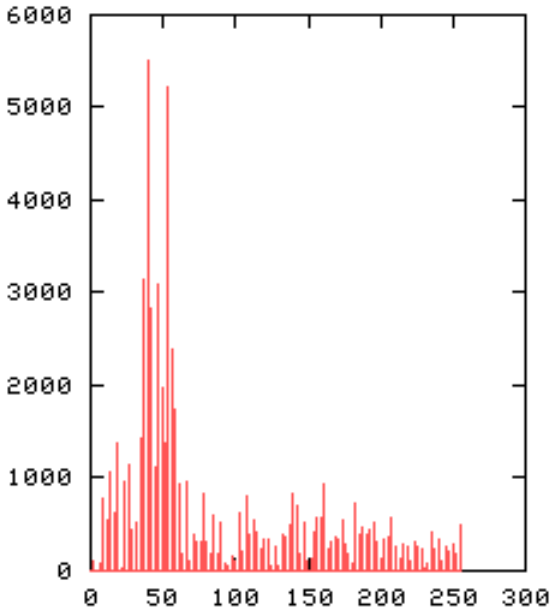
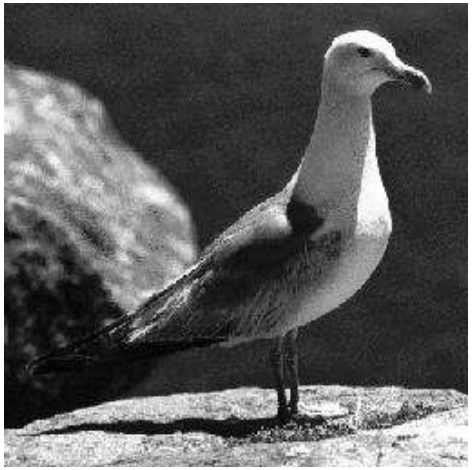


Image Representations: Histograms





Global histogram

- Represent distribution of features
  - Color, texture, depth, …

Images from Dave Kauchak

Computing Histogram Distance

$$\text{histint}(h_i, h_j) = 1 - \sum_{m=1}^{K} \min\left(h_i(m), h_j(m)\right)$$

Histogram intersection (assuming normalized histograms)

$$\chi^2(h_i, h_j) = \frac{1}{2} \sum_{m=1}^{K} \frac{[h_i(m) - h_j(m)]^2}{h_i(m) + h_j(m)}$$

Chi-squared Histogram matching distance

Cars found by color histogram matching using chi-squared

Histograms: Implementation Issues

- Quantization
  - Grids: fast but applicable only with few dimensions
  - Clustering: slower but can quantize data in higher dimensions

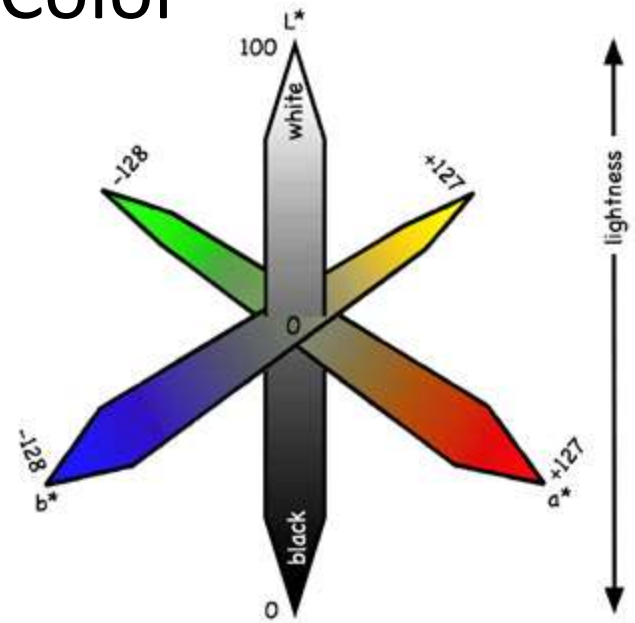
Few Bins
Need less data
Coarser representation

Many Bins
Need more data
Finer representation

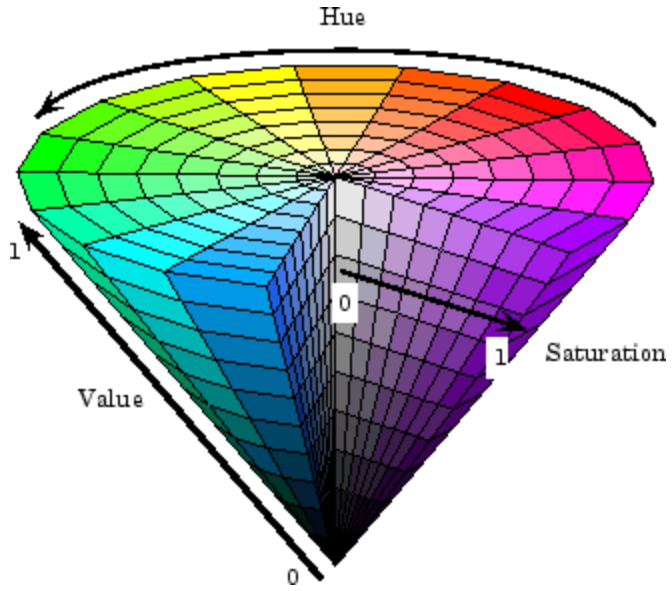
- Matching
  - Histogram intersection or Euclidean may be faster
  - Chi-squared often works better
  - Earth mover's distance is good for when nearby bins represent similar values

What kind of things do we compute histograms of?

- Color



L*a*b* color space

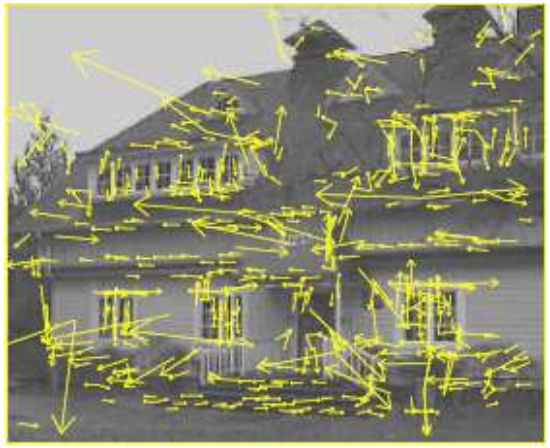


HSV color space

- Texture (filter banks or HOG over regions)

What kind of things do we compute histograms of?

- Histograms of oriented gradients



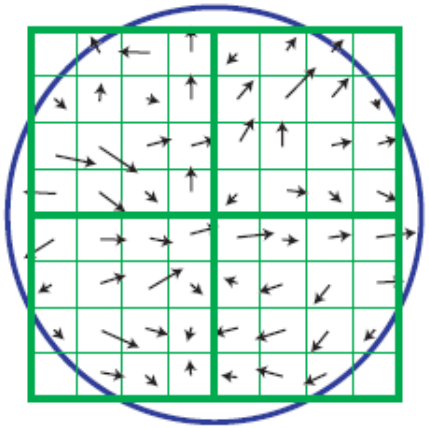
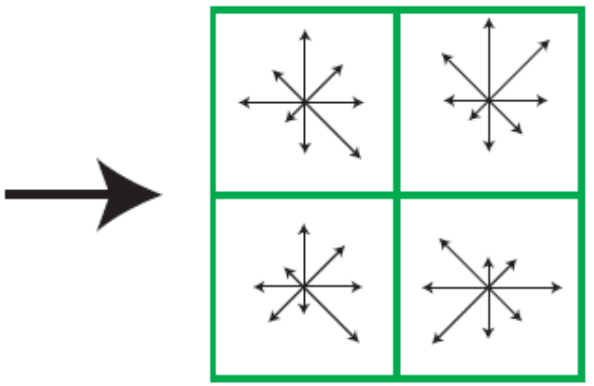


Image gradients

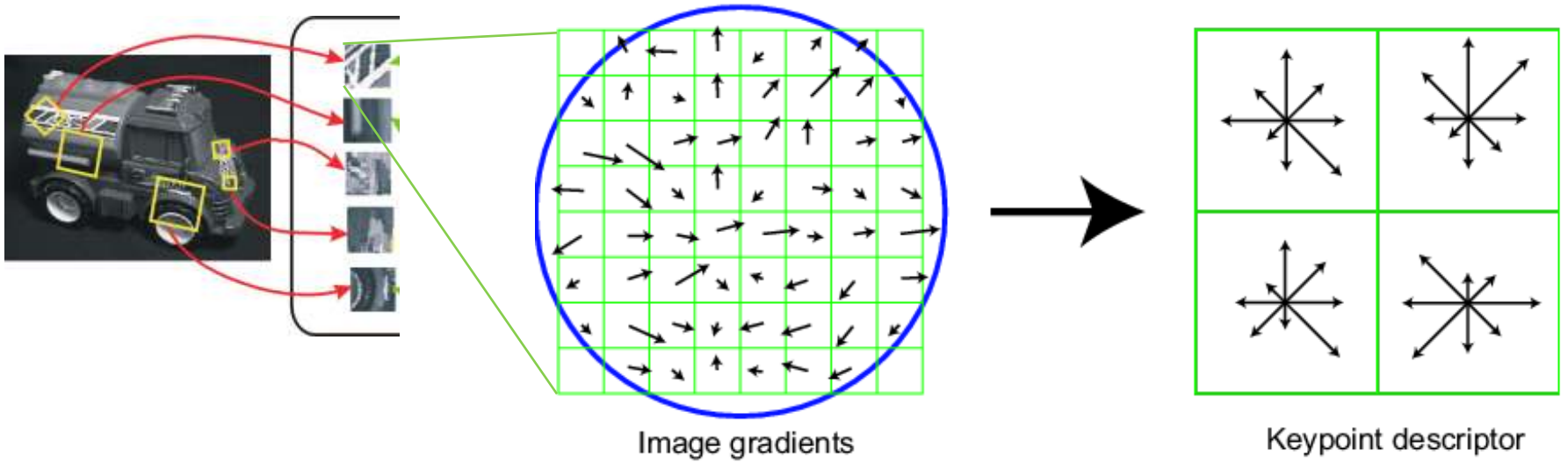


Keypoint descriptor

SIFT – Lowe IJCV 2004

SIFT Vector Formation

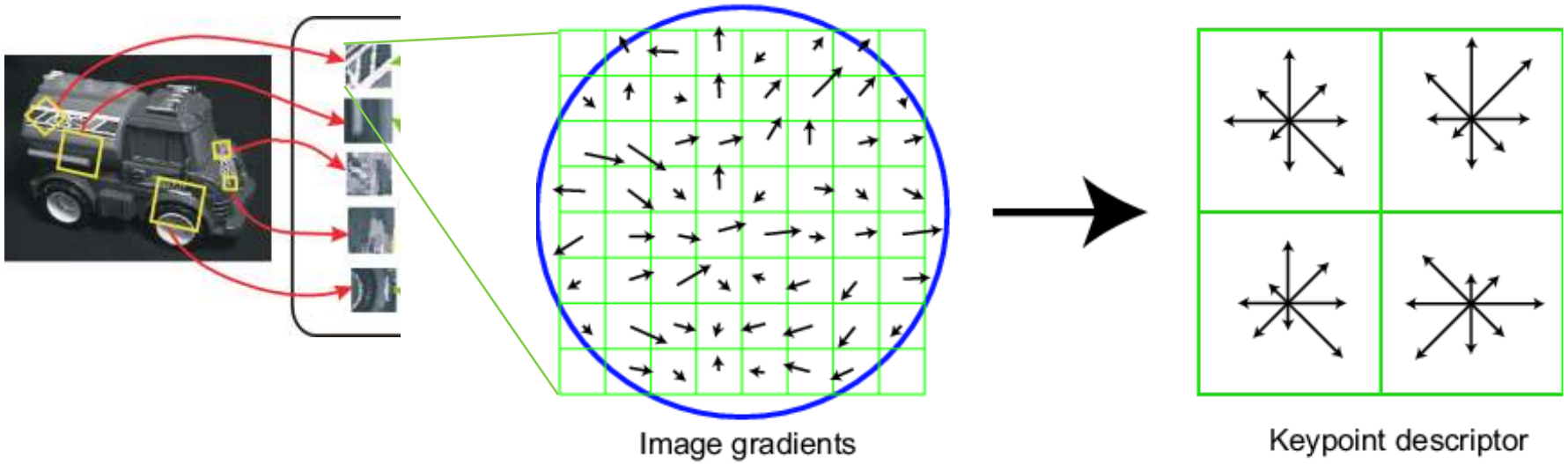
- 4x4 array of gradient orientation histograms
  - not really histogram, weighted by magnitude
- 8 orientations 4x4 array = 128 dimensions



showing only 2x2 here but is 4x4

Reduce Effect of Illumination

- 128-dim vector normalized to 1
- Threshold gradient magnitudes to avoid excessive influence of high gradients
  - after normalization, clamp gradients >0.2
  - renormalize



Diagonal-Offset Model

- Reflectance model:

$$\mathbf{f}(\mathbf{x}) = \int_{\omega} e(\lambda)\rho_k(\lambda)s(\mathbf{x},\lambda)d\lambda + \int_{\omega} a(\lambda)\rho_k(\lambda)$$

- Corresponds to diagonal-offset model of illumination change:

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix} + \begin{pmatrix} o_1 \\ o_2 \\ o_3 \end{pmatrix}$$

Von Kries 1970
Finlayson ICIP 2005

Invariance properties: Diagonal model

Lambertian reflectance model

$$\mathbf{f}(\mathbf{x}) = \int_{\omega} e(\lambda)\rho_k(\lambda)s(\mathbf{x},\lambda)d\lambda + \int_{\omega} a(\lambda)\rho_k(\lambda)$$

- Corresponds to diagonal-offset model of illumination change

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix} + \begin{pmatrix} o_1 \\ o_2 \\ o_3 \end{pmatrix}$$

Illuminant parameters

Canonical illuminant

Unknown illuminant

Unified framework for modeling:

- Shadows
- Shading
- Light color changes
- Highlights
- Scattering

Photometric Analysis

1. Light intensity change (a = b = c)

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix}$$

Examples: shadows, shading

➘ *scale-invariant* w.r.t. light intensity

$I^c = a\, I^u$

Photometric Analysis

Photometric Analysis

Light intensity shift (a = b = c = 1; $o_1 = o_2 = o_3$)

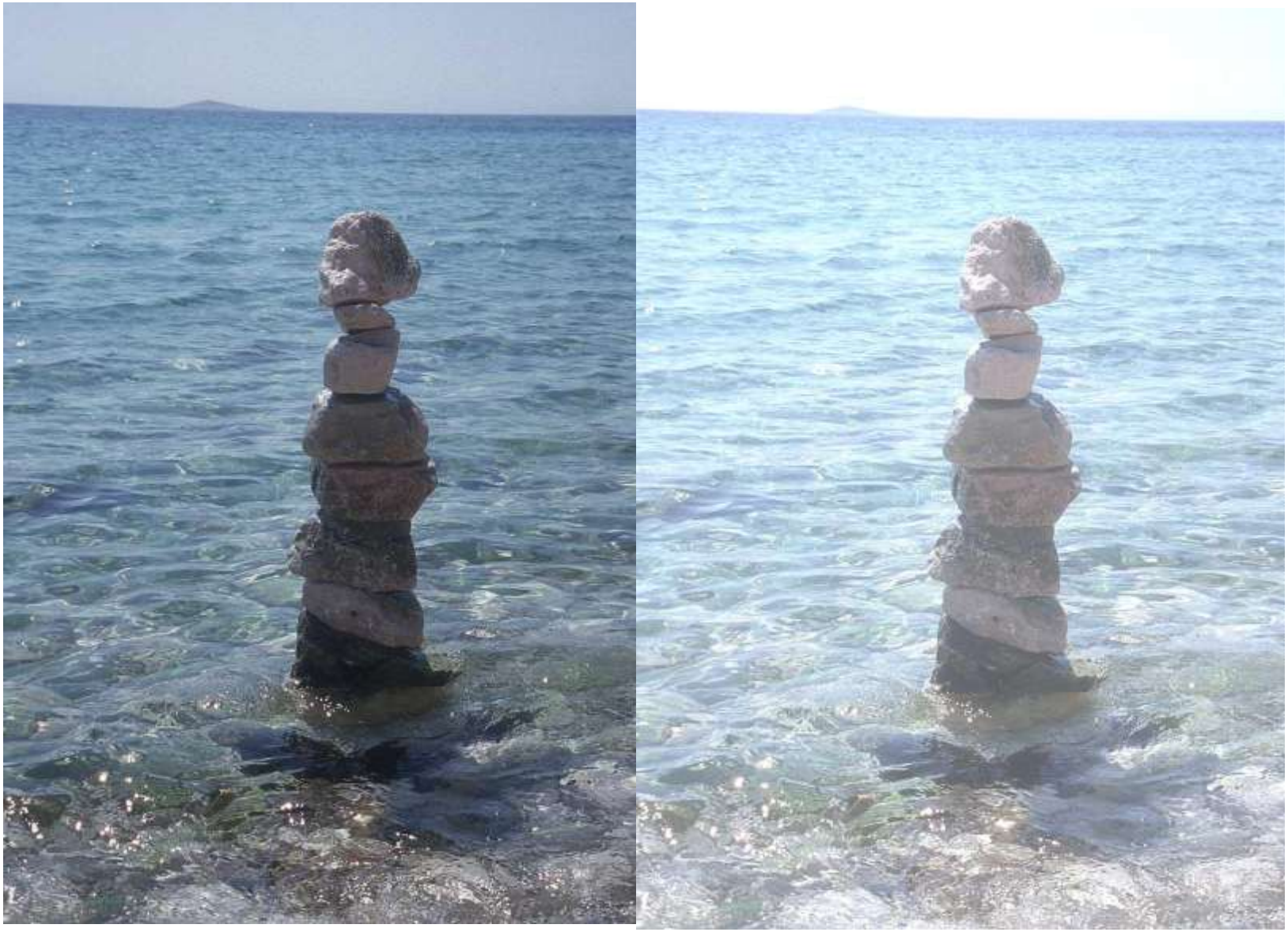
$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$$

Examples: object highlights under white light source, scattering of a white source

➘ *shift-invariant* w.r.t. light intensity

$I^c = I^u + o_1$

Photometric Analysis



Photometric Analysis

Light intensity change *and* shift (a = b = c; $o_1 = o_2 = o_3$)

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$$

➘ *scale-invariant* and *shift-invariant*

$I^c = a\, I^u + o_1$

Photometric Analysis

Light color change

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix}$$

Light color change and shift

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix} + \begin{pmatrix} o_1 \\ o_2 \\ o_3 \end{pmatrix}$$

(1,1,1)

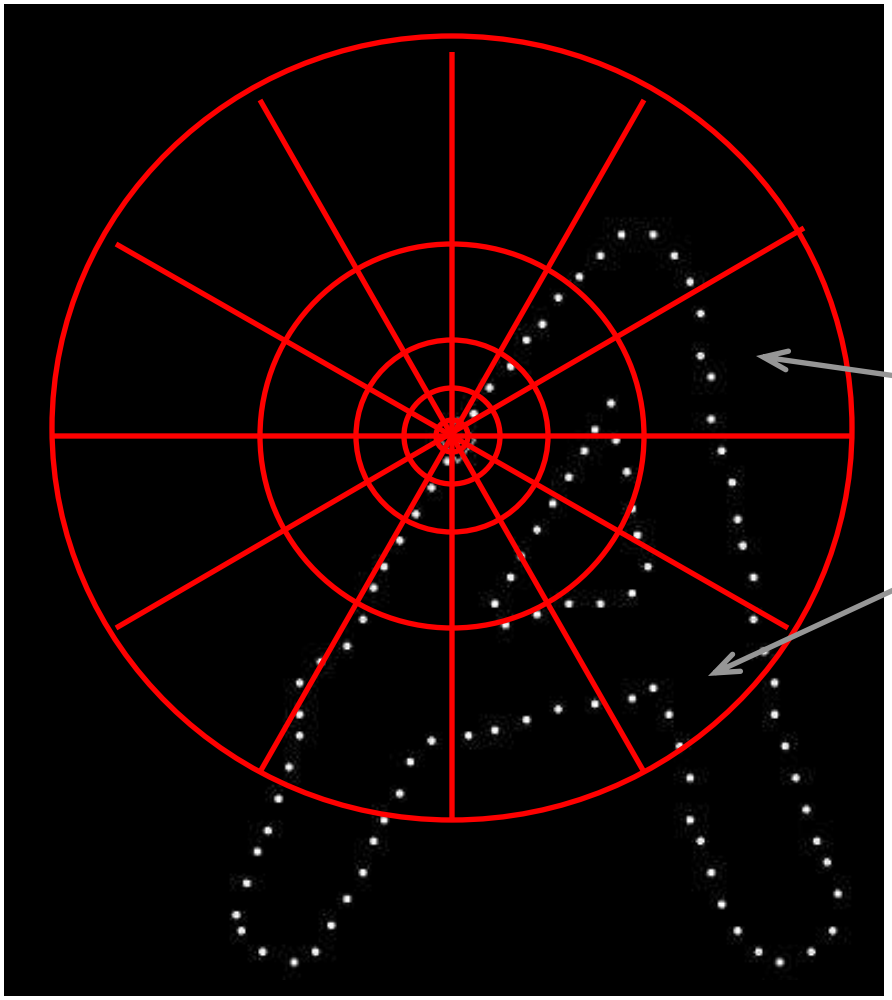
(1.8,1.2,1.2)

Color Descriptor Taxonomy

[van de Sande, IEEE PAMI, 09]

	Light intensity change $\begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$	Light intensity shift $\begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$	Light intensity change and shift $\begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$	Light color change $\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$	Light color change and shift $\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_2 \\ o_3 \end{pmatrix}$
RGB Histogram	-	-	-	-	-
O_1, O_2	-	+	-	-	-
O_3, Intensity	-	-	-	-	-
Hue	+	+	+	-	-
Saturation	+	+	+	-	-
r, g	+	-	-	-	-
Transformed color	+	+	+	+	+
Color moments	-	+	-	-	-
Moment invariants	+	+	+	+	+
SIFT (∇I)	+	+	+	+	+
HSV-SIFT	+	+	+	+/-	+/-
HueSIFT	+	+	+	+/-	+/-
OpponentSIFT	+/-	+	+/-	+/-	+/-
W-SIFT	+	+	+	+/-	+/-
rgSIFT	+	+	+	+/-	+/-
Transf. color SIFT	+	+	+	+	+

Local Descriptors: Shape Context



Count the number of points inside each bin, e.g.:

Count = 4

⋮

Count = 10

Log-polar binning: more precision for nearby points, more flexibility for farther points.

Belongie & Malik, ICCV 2001

Self-similarity Descriptor

Figure 1. *These images of the same object (a heart) do NOT share common image properties (colors, textures, edges), but DO share a similar geometric layout of local internal self-similarities.*

Matching Local Self-Similarities across Images and Videos, Shechtman and Irani, 2007

Summary

- Descriptors: histograms of pixel values, texture or oriented gradients.

- Think about the right features for the problem.

Overview

Mosaicking

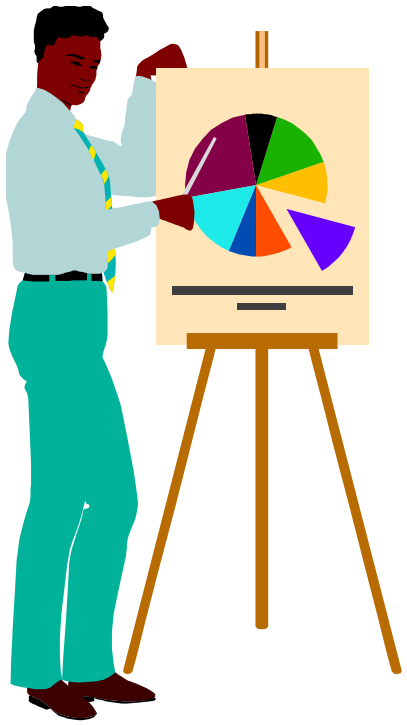
Object Recognition in Perspective

Image Descriptors

Bag-of-Models

Classifiers

Object Recognition Benchmarks



Bag-of-features Models

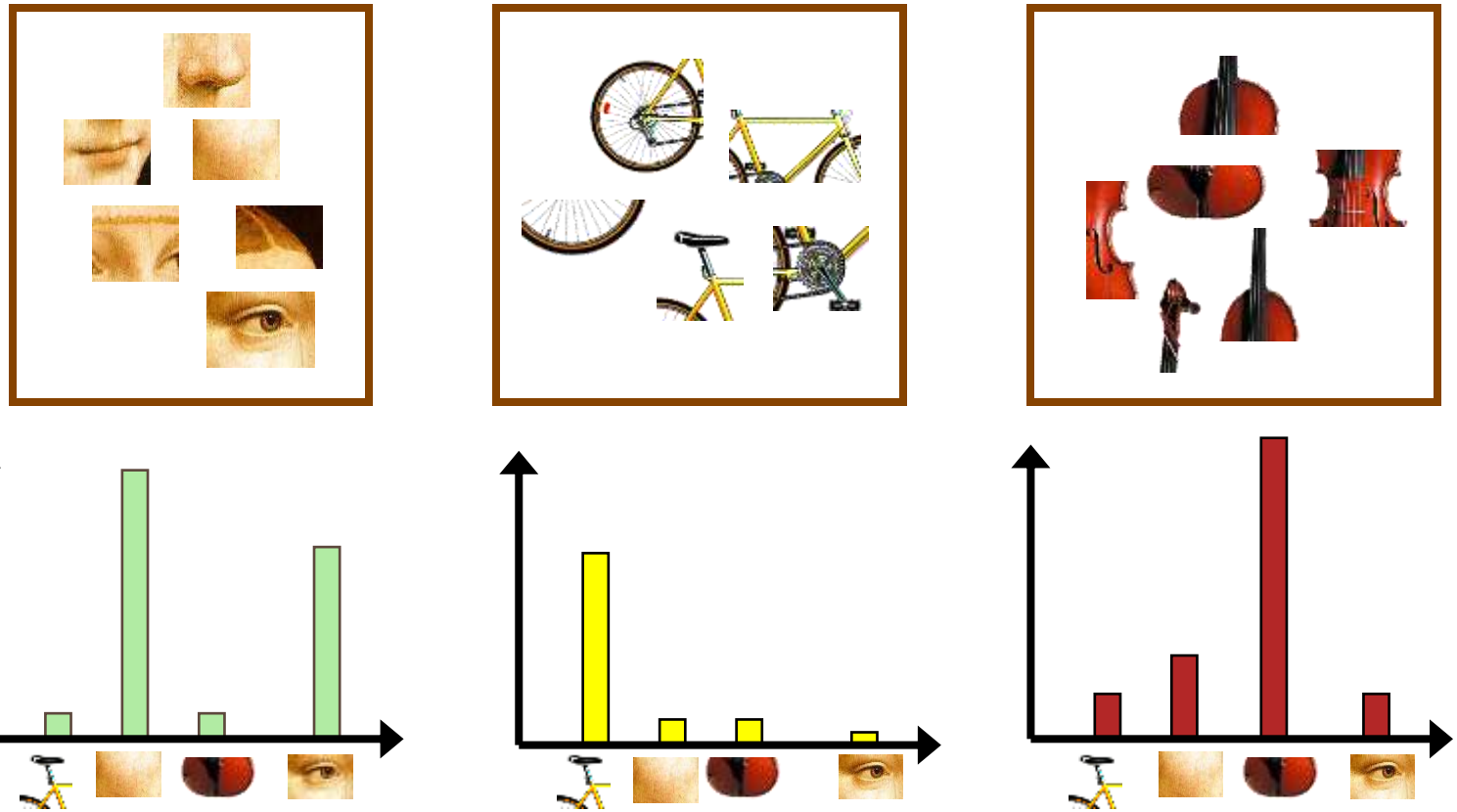
Origin: Bag-of-words models

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

US Presidential Speeches Tag Cloud
http://chir.ag/phernalia/preztags/

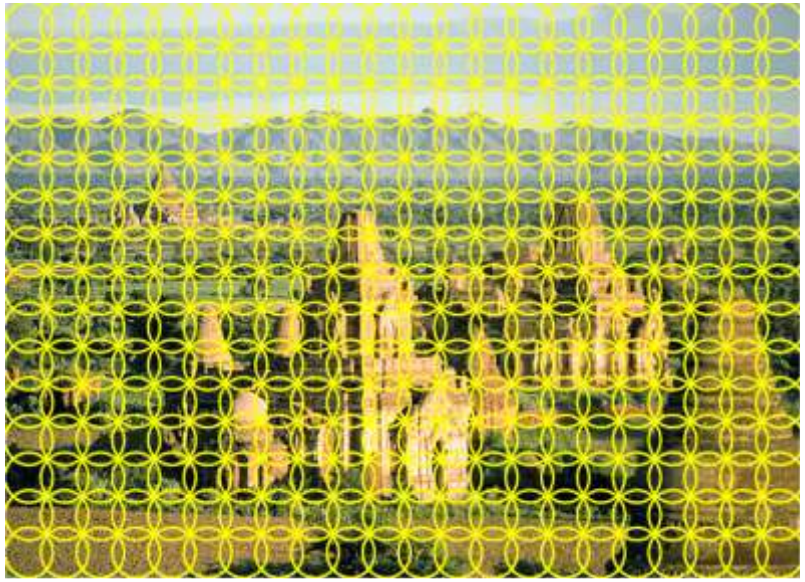
Bag-of-features Steps

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

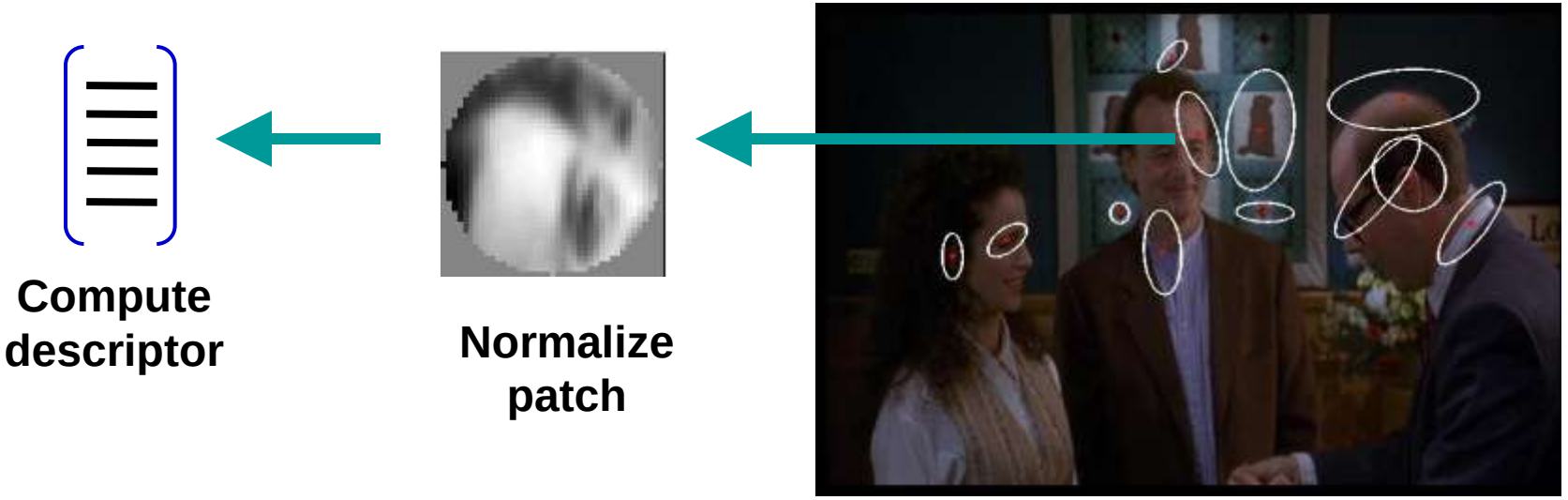


1. Feature extraction

- Regular grid or interest regions

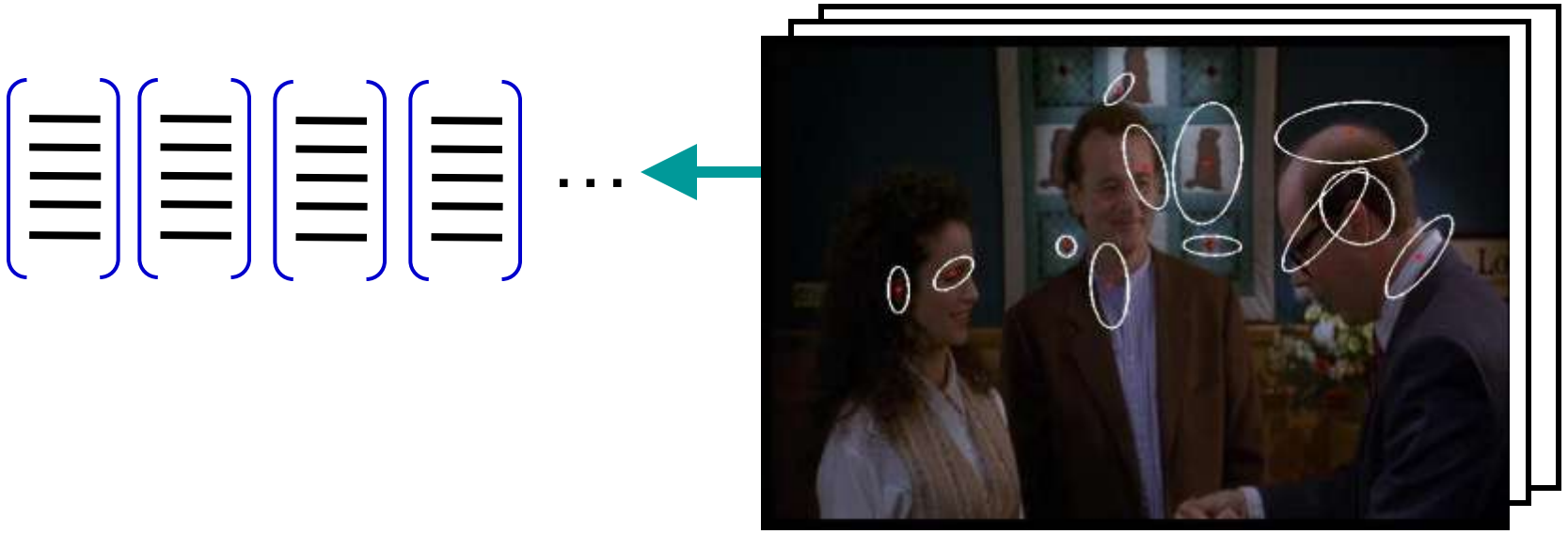


1. Feature extraction



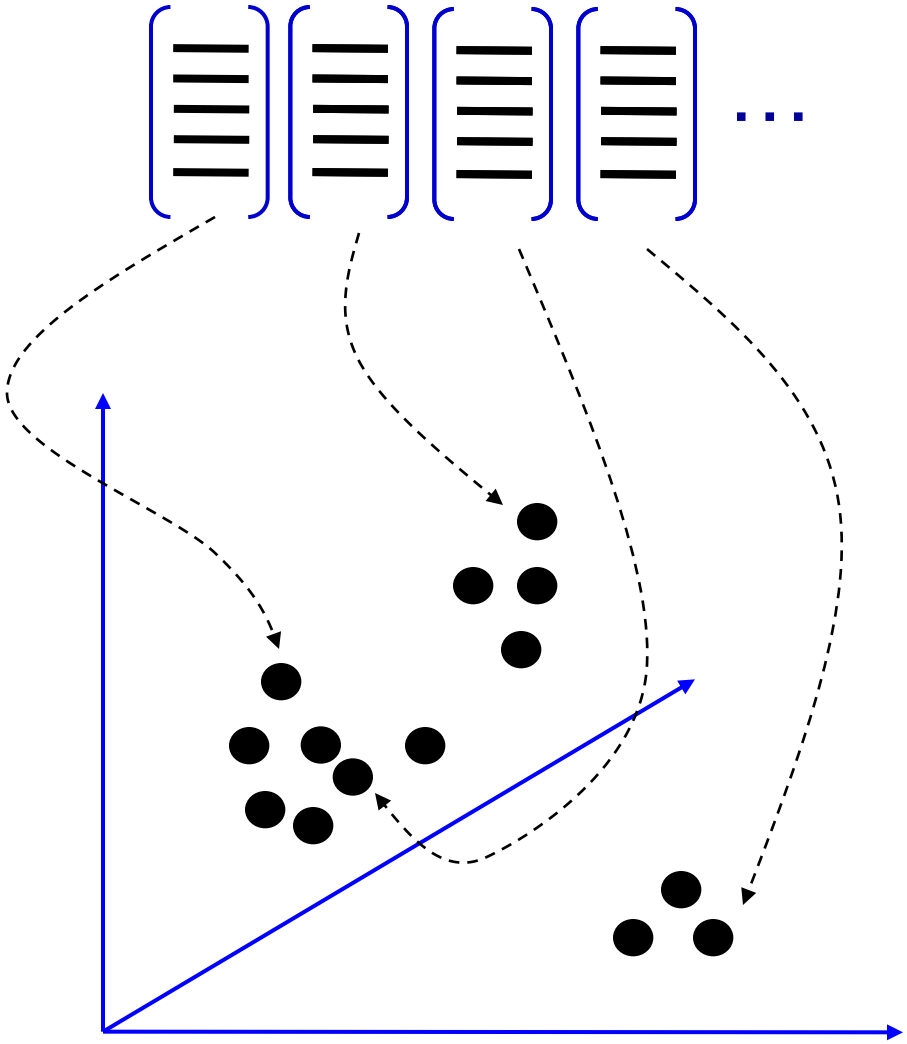
Slide credit: Josef Sivic

1. Feature extraction



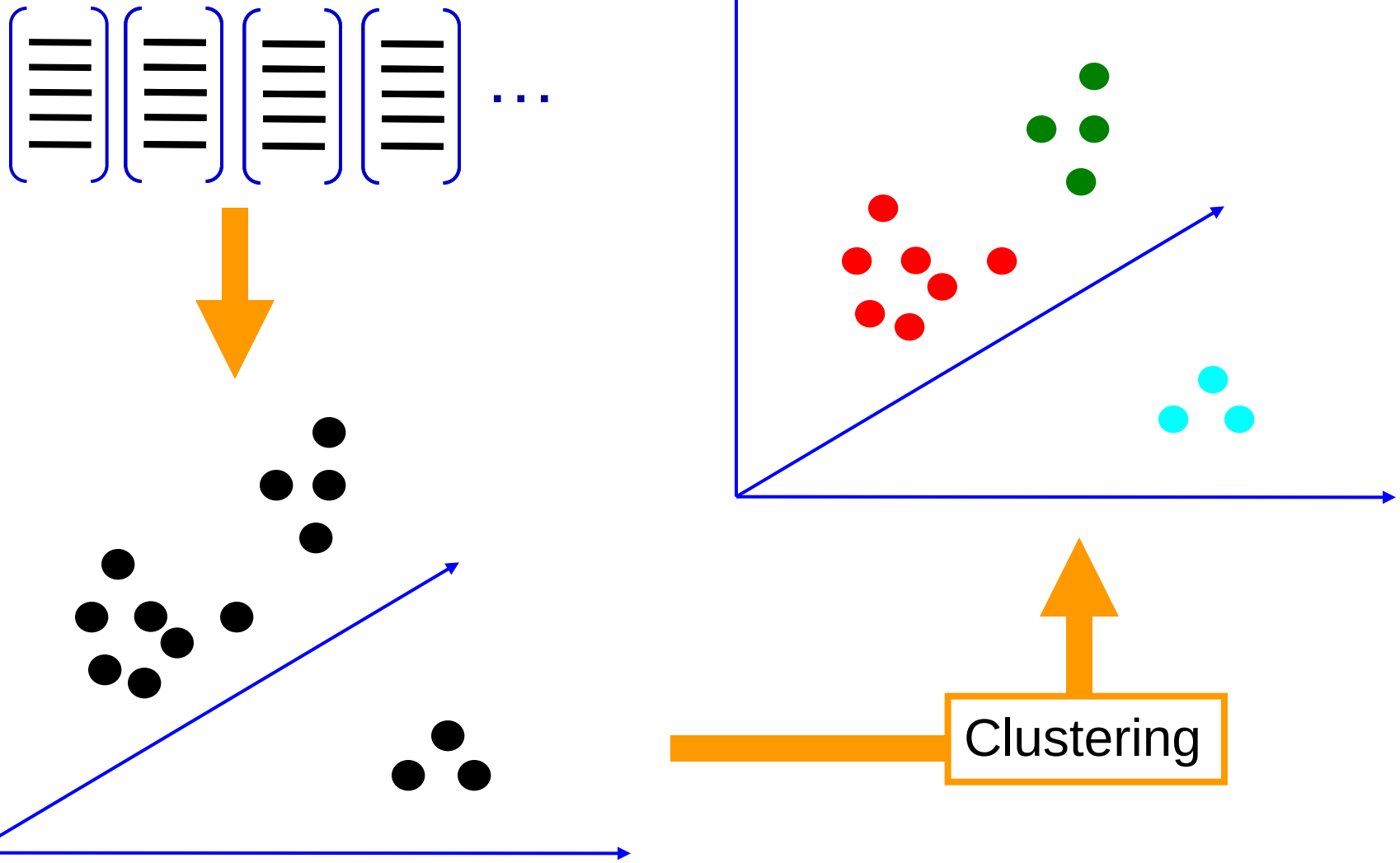
Slide credit: Josef Sivic

2. Learning the visual vocabulary



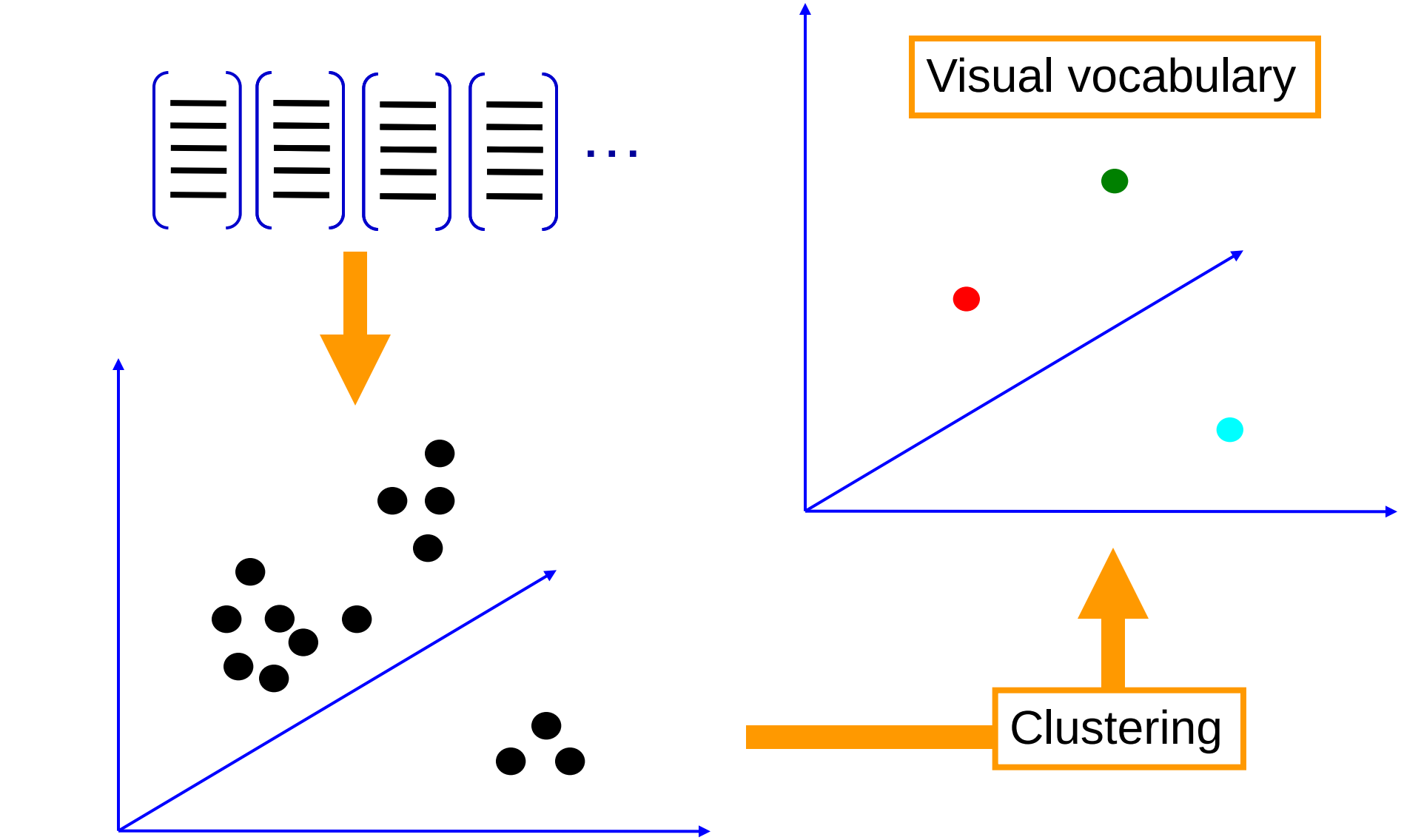
Slide credit: Josef Sivic

2. Learning the visual vocabulary



Slide credit: Josef Sivic

2. Learning the visual vocabulary



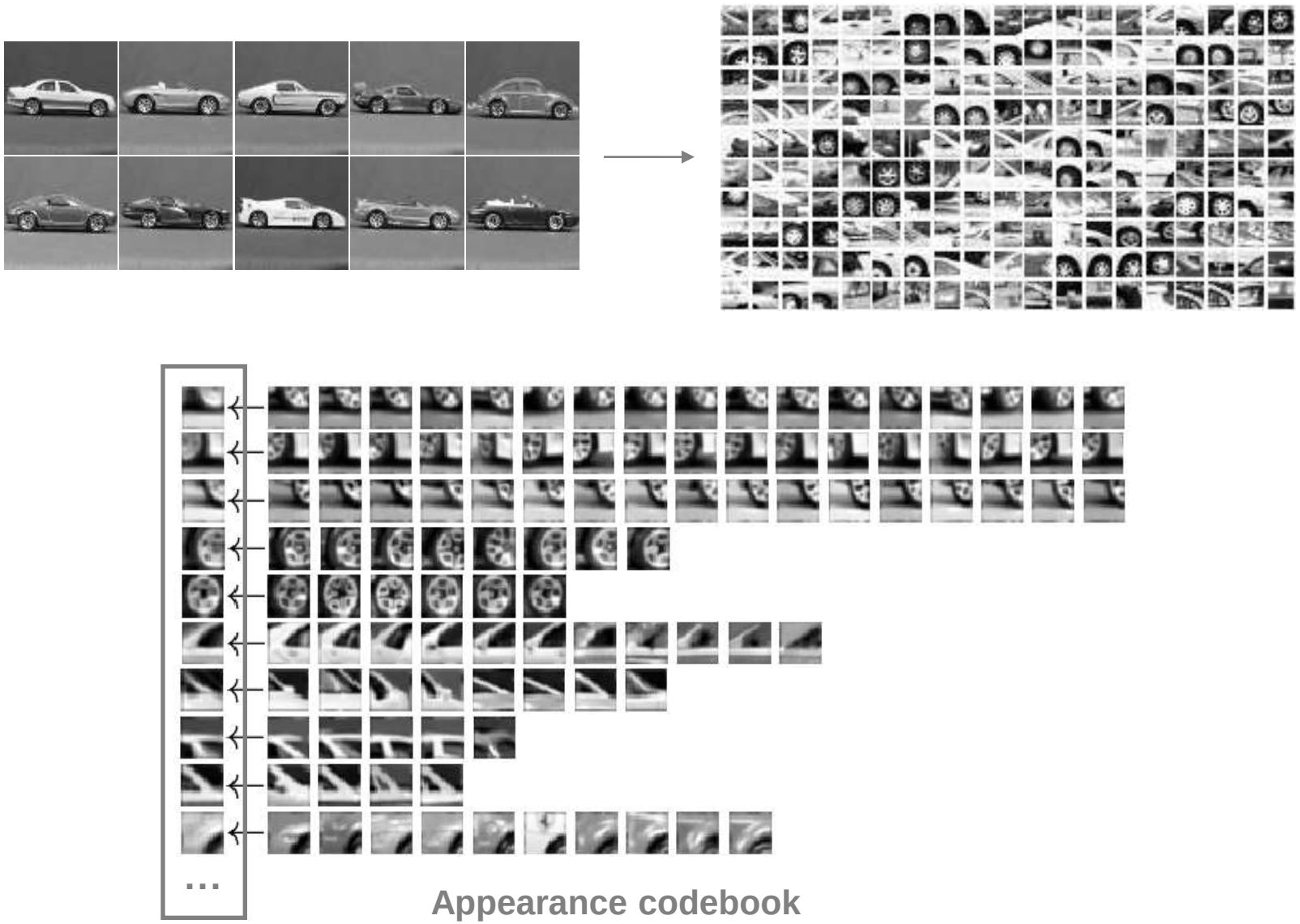
Slide credit: Josef Sivic

Clustering and Vector Quantization

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

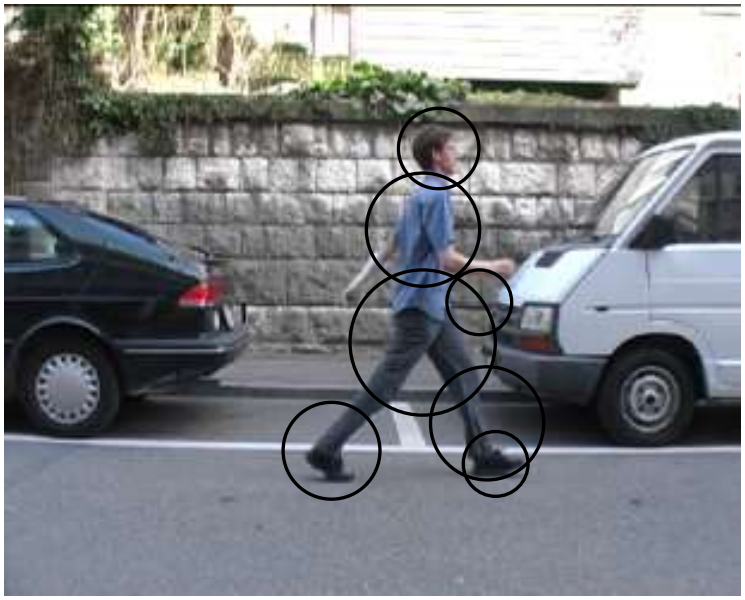
- The codebook is used for quantizing features
  - A *vector quantizer* takes a feature vector and maps it to the index of the nearest codevector in a codebook
  - Codebook = visual vocabulary
  - Codevector = visual word

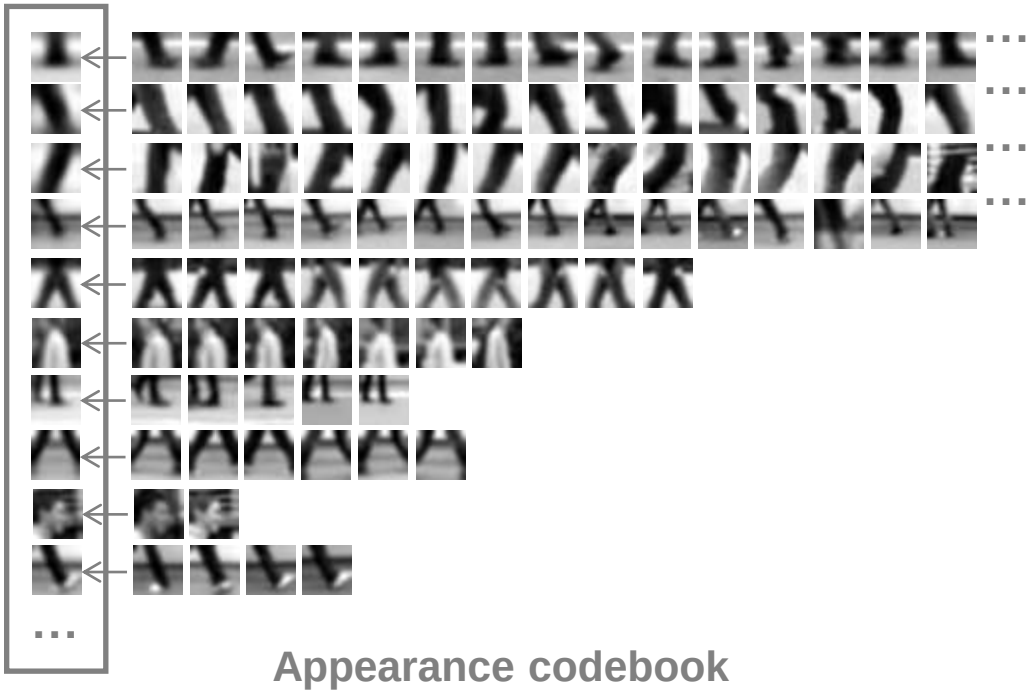
Example codebook



Source: B. Leibe

Another codebook

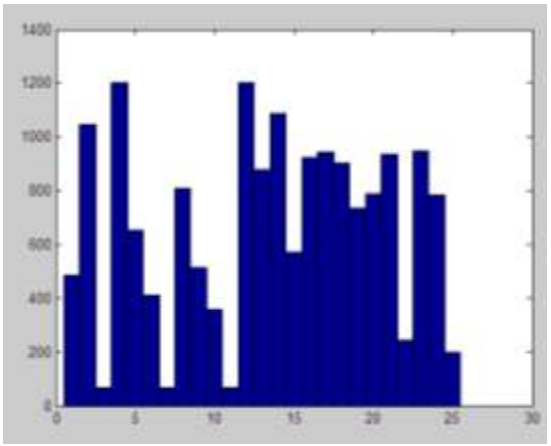




Appearance codebook

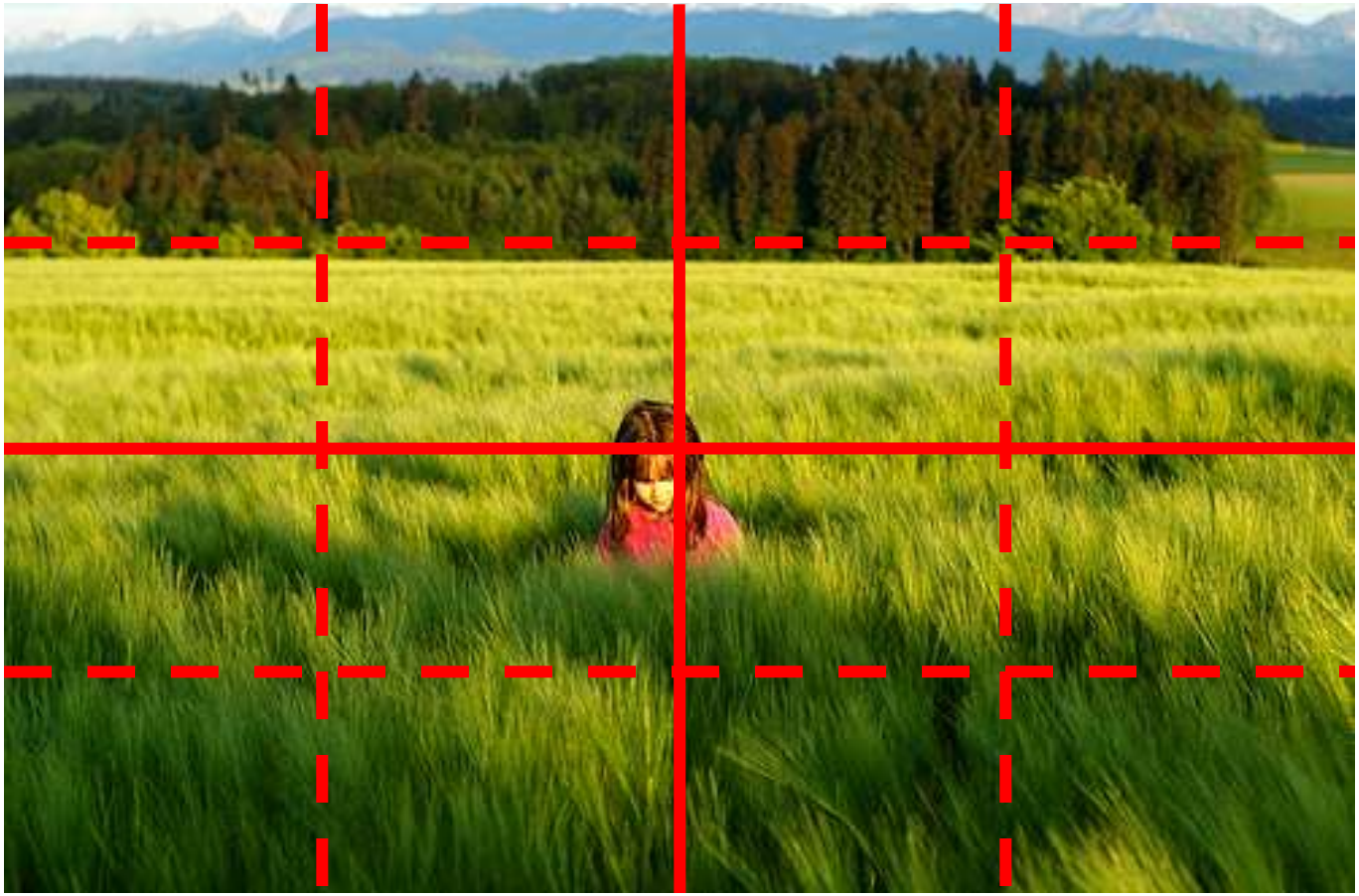
Source: B. Leibe

But what about Layout?



All of these images have the same color histogram

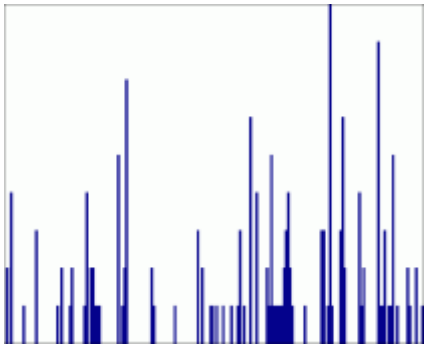
Spatial Pyramid



Compute histogram in each spatial bin

Spatial Pyramid Representation

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

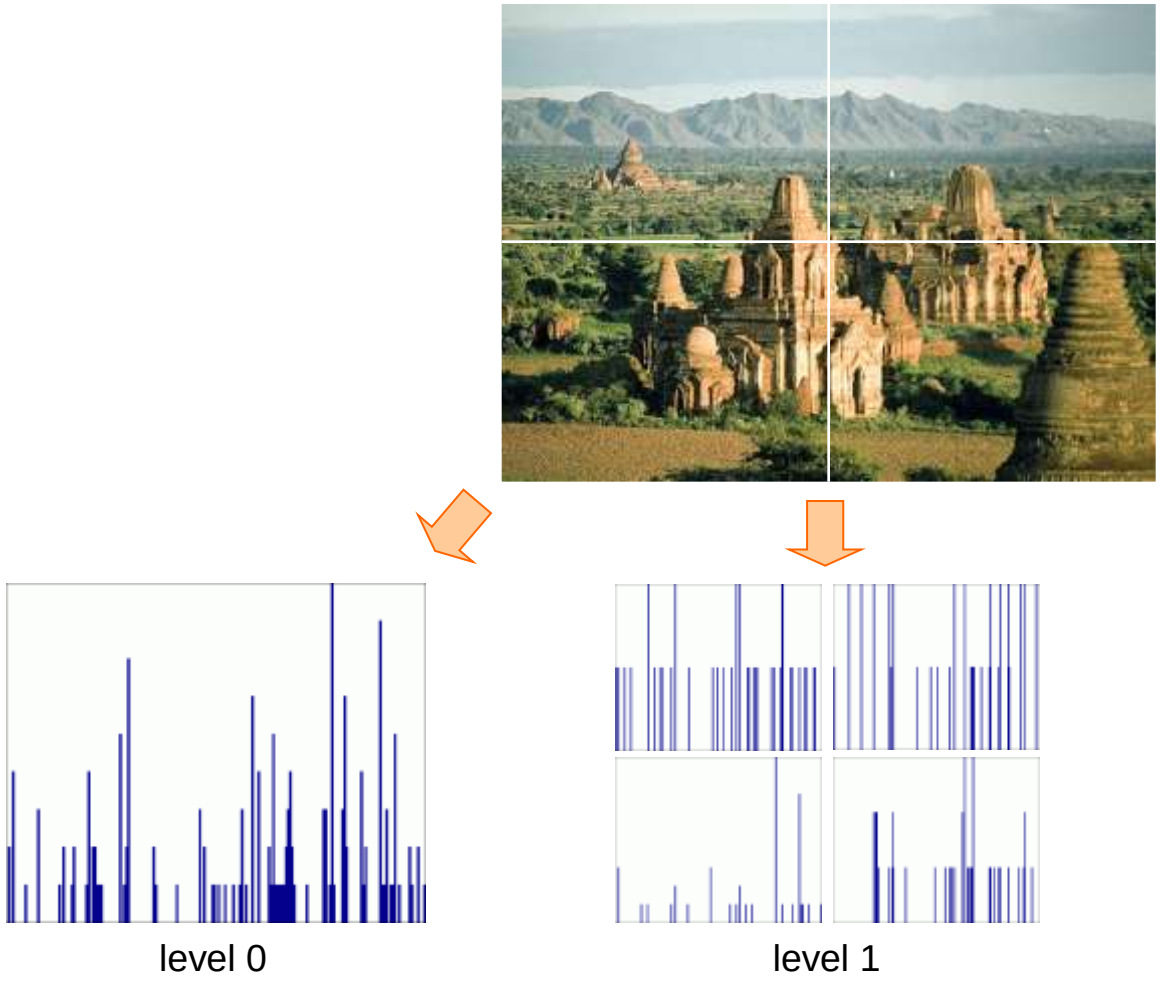


level 0

Lazebnik, Schmid & Ponce (CVPR 2006)

Spatial Pyramid Representation

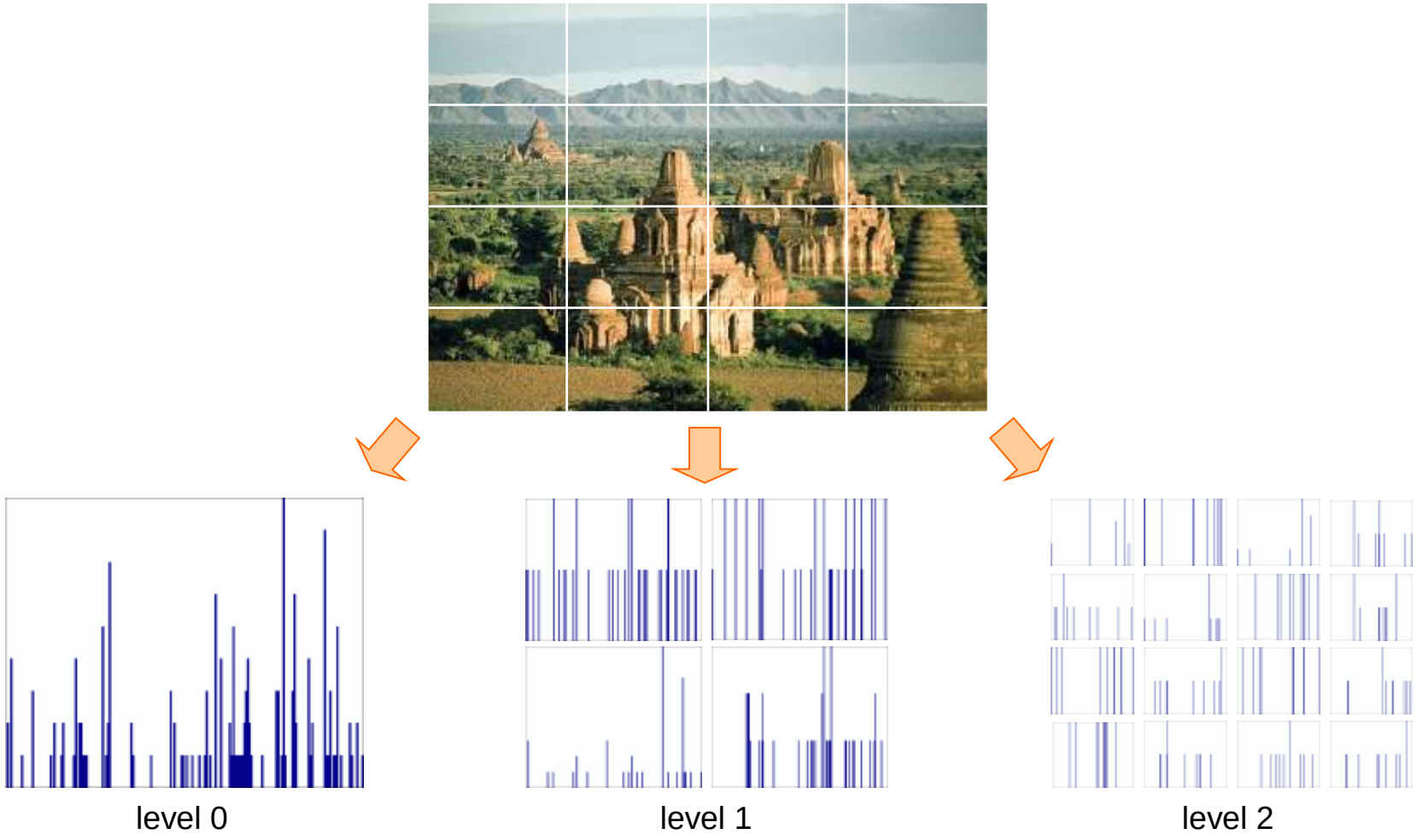
- Extension of a bag of features
- Locally orderless representation at several levels of resolution



Lazebnik, Schmid & Ponce (CVPR 2006)

Spatial Pyramid Representation

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



Lazebnik, Schmid & Ponce (CVPR 2006)

Overview

Mosaicking

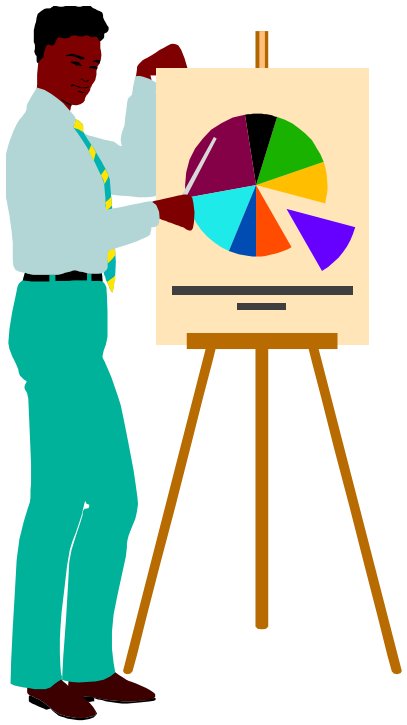
Object Recognition in Perspective

Image Descriptors

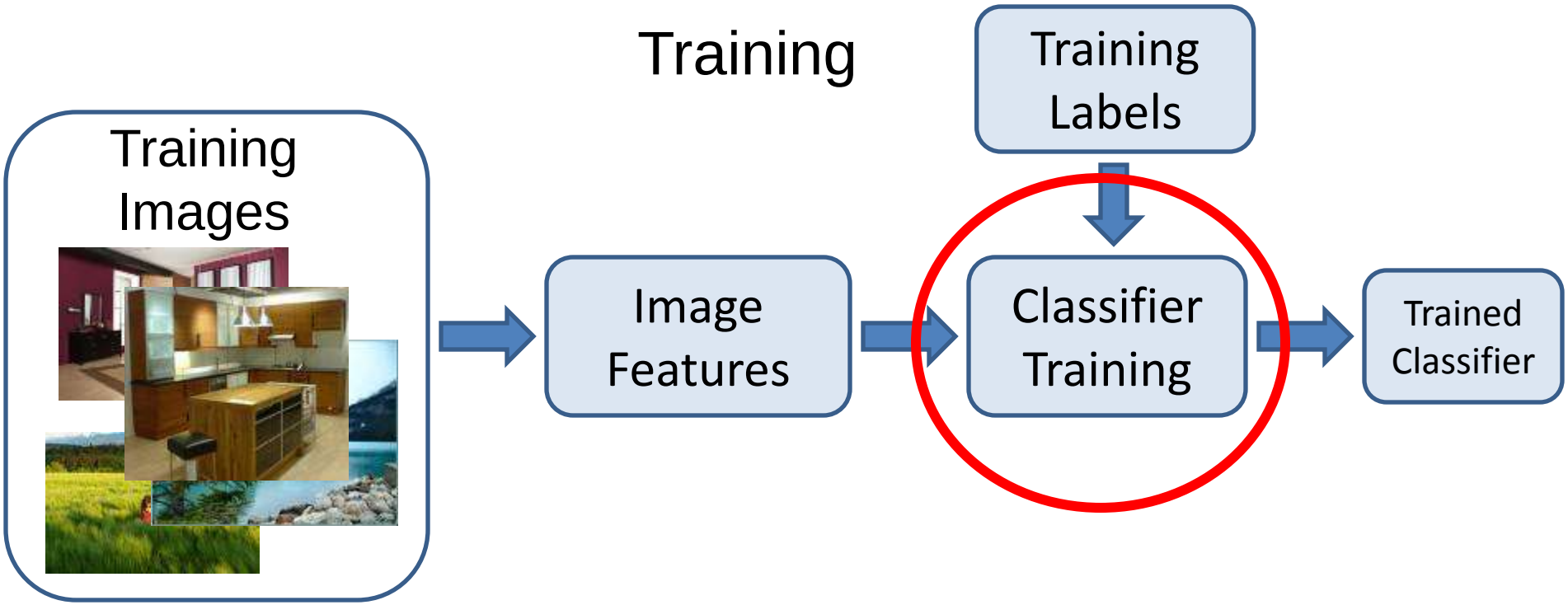
Bag-of-Models

Classifiers

Object Recognition Benchmarks



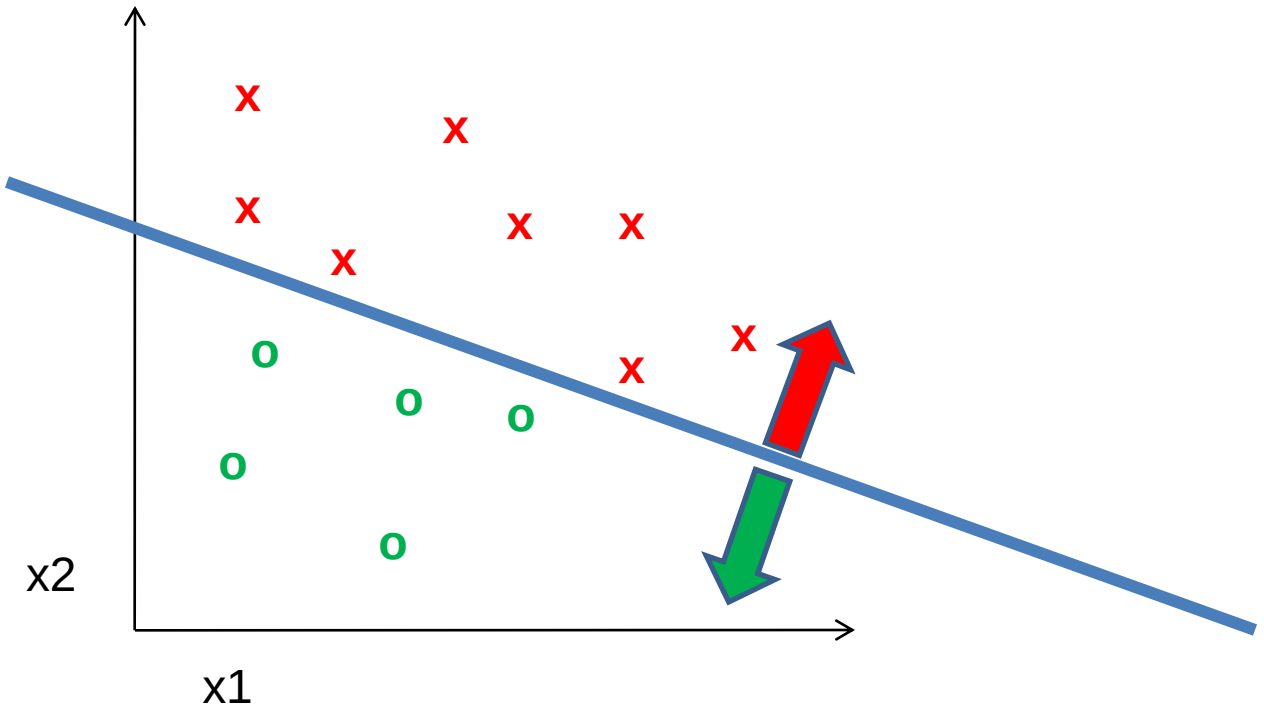
Classifiers



Slide: Derek Hoiem

Learning a classifier

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



Slide: Derek Hoiem

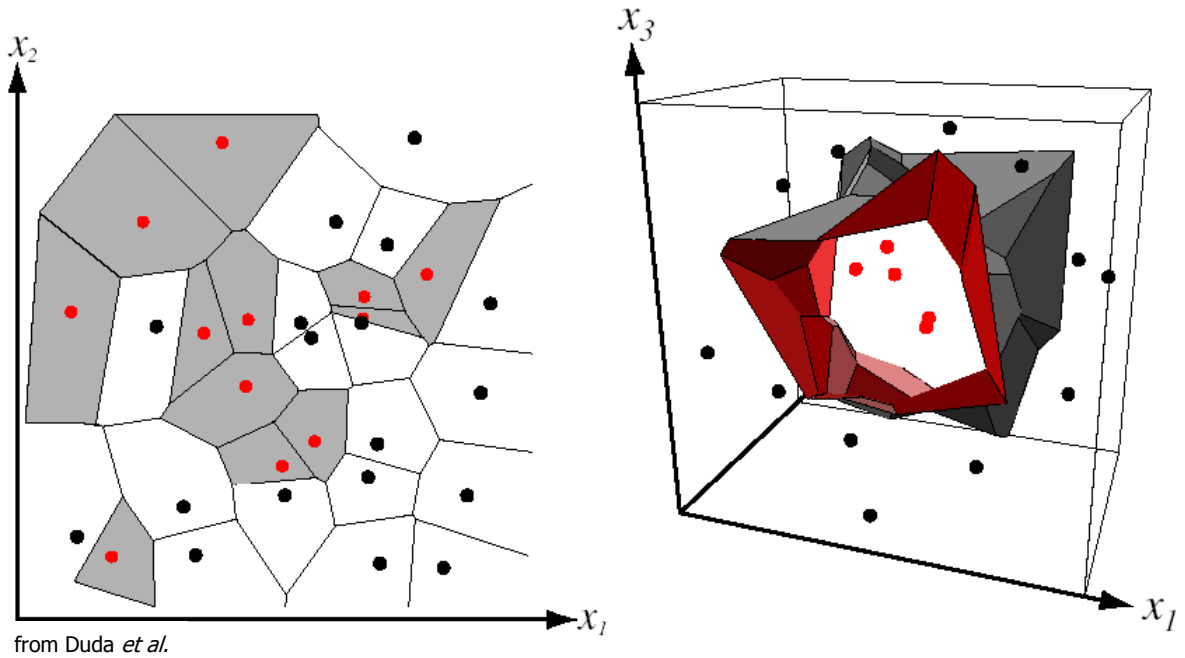
Many classifiers to choose from

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

Slide: Derek Hoiem

Nearest Neighbor Classifier

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

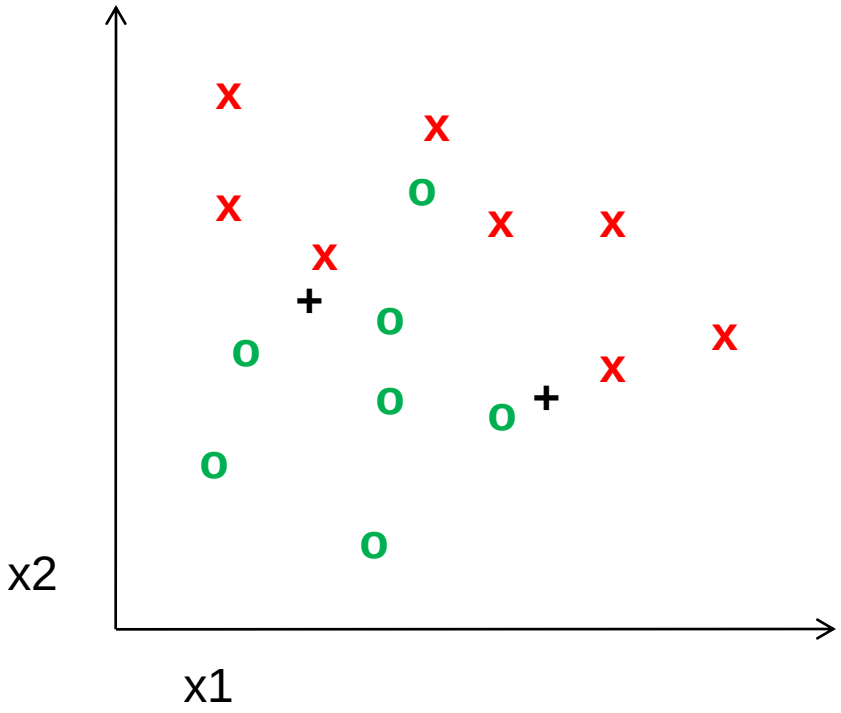


from Duda *et al.*

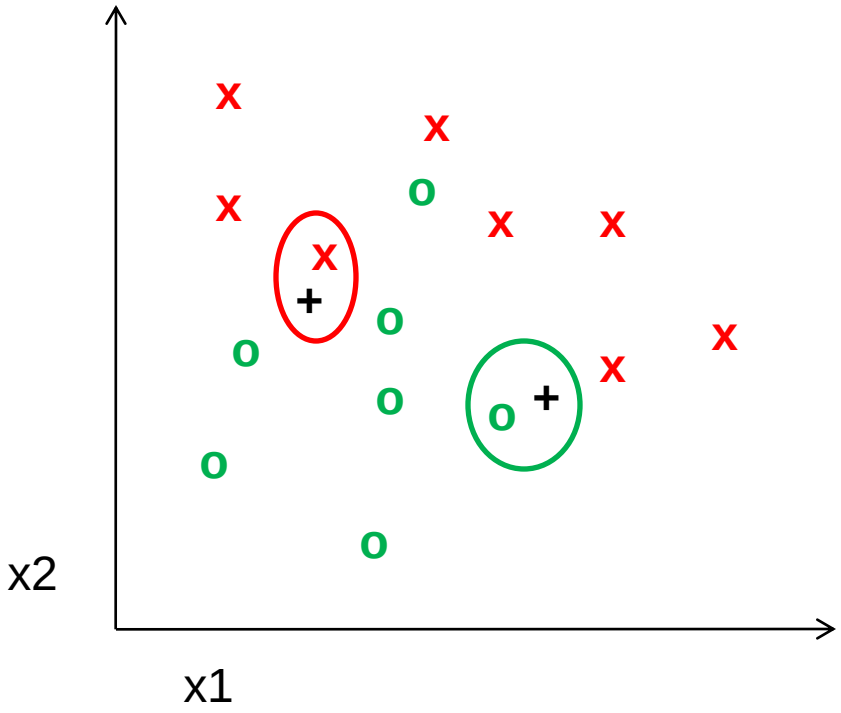
Voronoi partitioning of feature space
for two-category 2D and 3D data

Source: D. Lowe

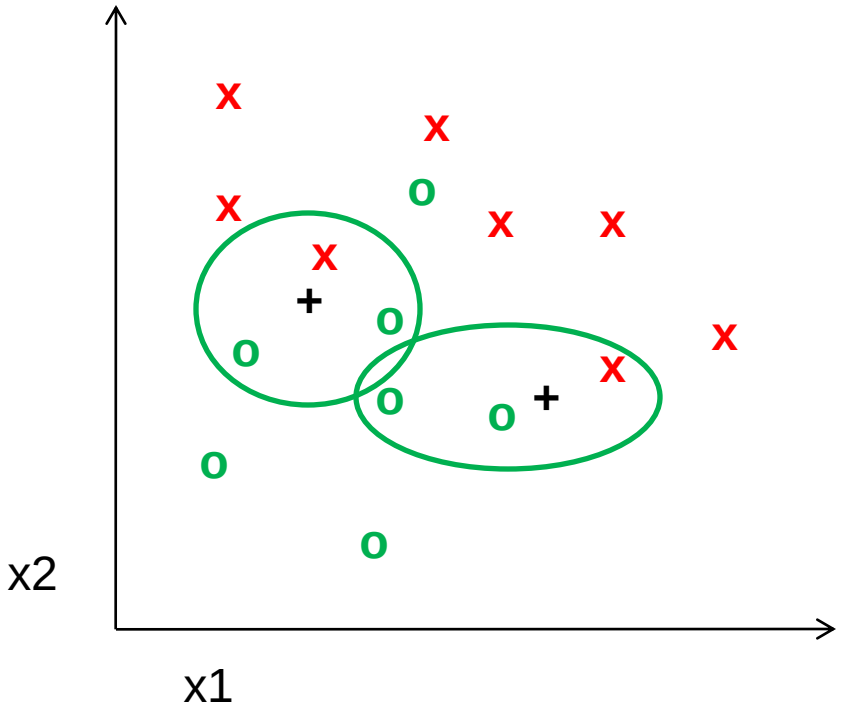
K-nearest neighbor



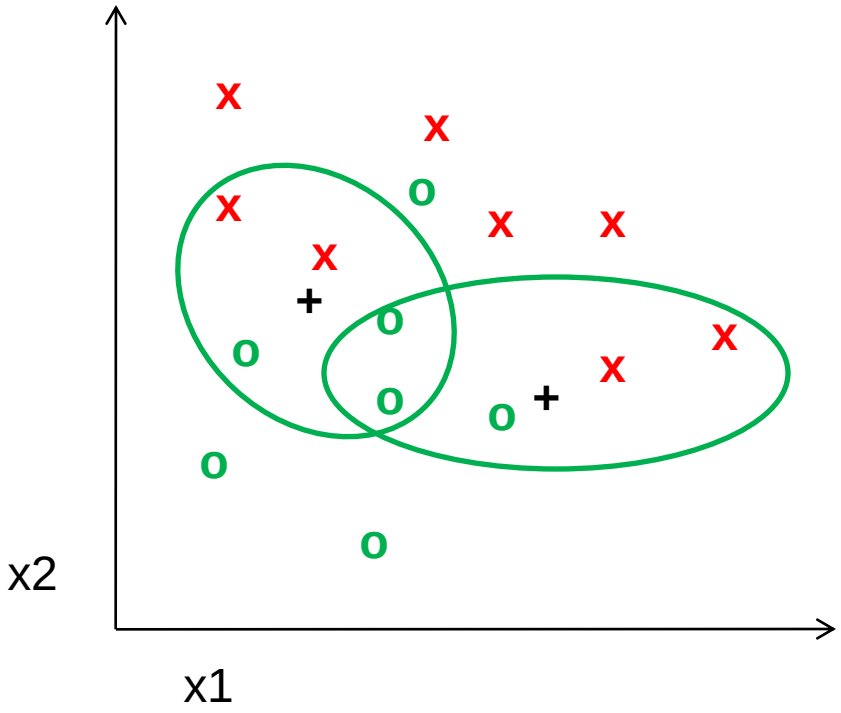
1-nearest neighbor



3-nearest neighbor



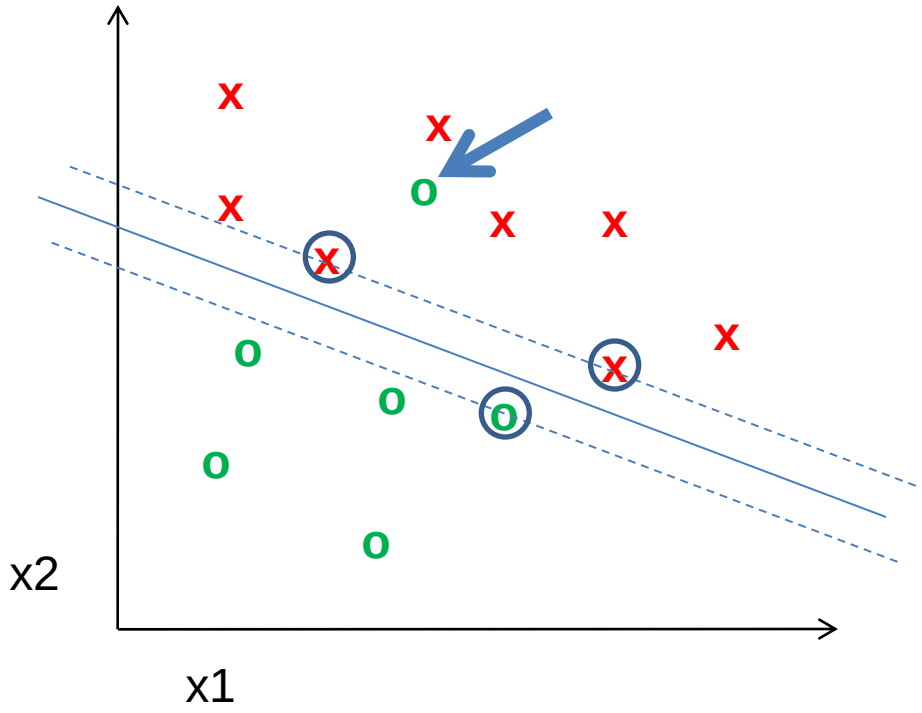
5-nearest neighbor



Using K-NN

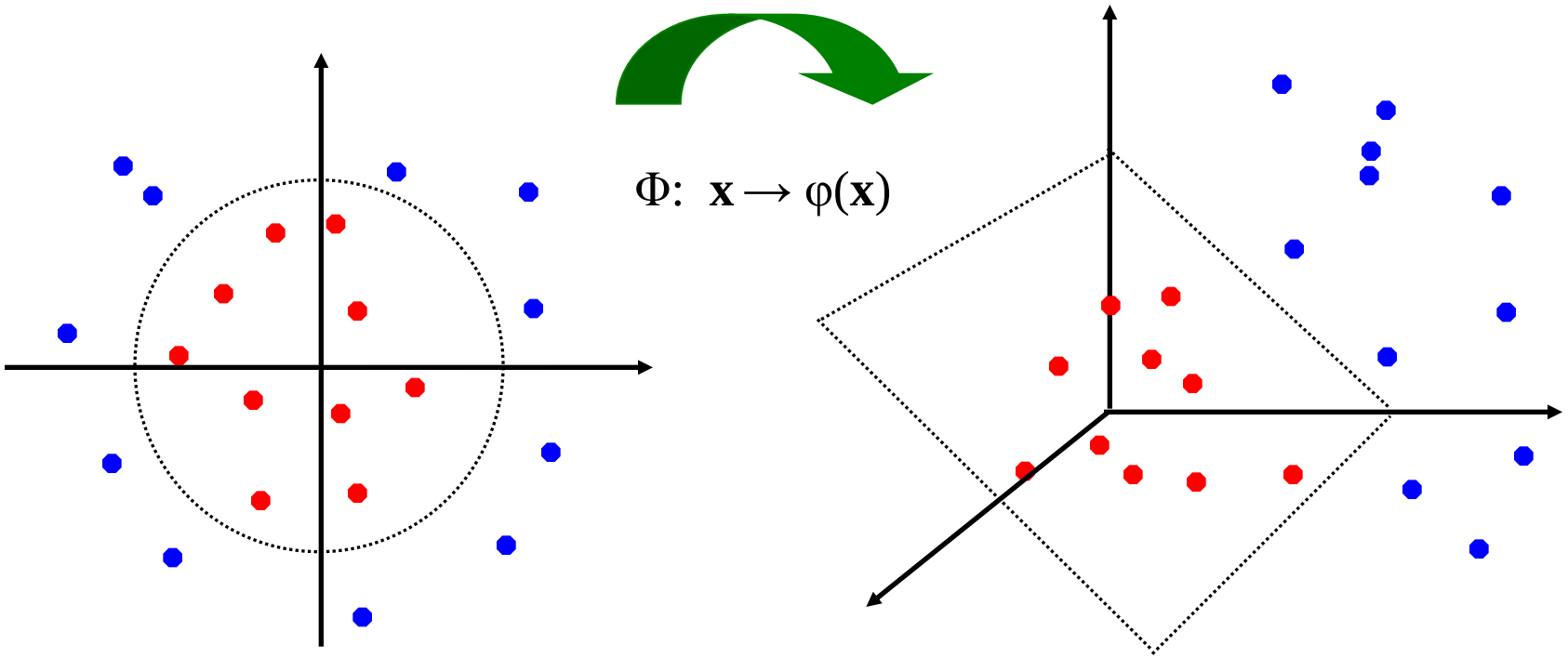
- Simple, a good one to try first
- With infinite examples, 1-NN provably has error that is at most twice Bayes optimal error

Classifiers: Linear SVM



Nonlinear SVMs

- General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:



Slide credit: Andrew Moore

Kernels for bags of Features

- Histogram intersection kernel:

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

- Generalized Gaussian kernel:

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

- *D* can be L1 distance, Euclidean distance, χ^2 distance, etc.

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

Summary: SVMs for image classification

1. Pick an image representation (in our case, bag of features)
2. Pick a kernel function for that representation
3. Compute the matrix of kernel values between every pair of training examples
4. Feed the kernel matrix into your favorite SVM solver to obtain support vectors and weights
5. At test time: compute kernel values for your test example and each support vector, and combine them with the learned weights to get the value of the decision function

Summary: Classifiers

- Nearest-neighbor and k-nearest-neighbor classifiers
  - L1 distance, χ^2 distance, quadratic distance, histogram intersection
- Support vector machines
  - Linear classifiers
  - Margin maximization
  - The kernel trick
  - Kernel functions: histogram intersection, generalized Gaussian, pyramid match
  - Multi-class
- Of course, there are many other classifiers out there
  - Neural networks, boosting, decision trees, …

Slide credit: L. Lazebnik

Overview

Mosaicking

Object Recognition in Perspective

Image Descriptors

Bag-of-Models

Classifiers

Object Recognition Benchmarks

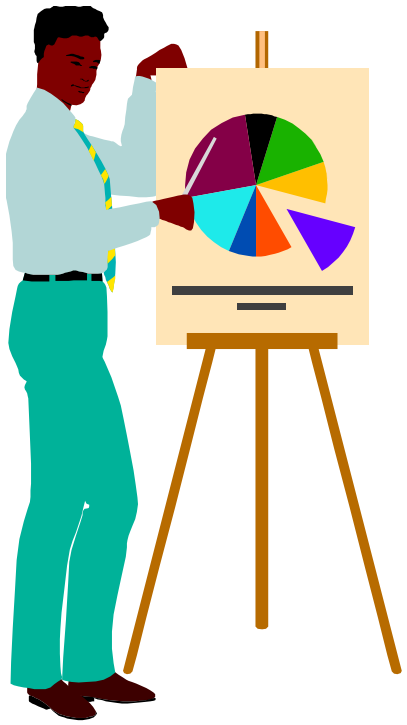
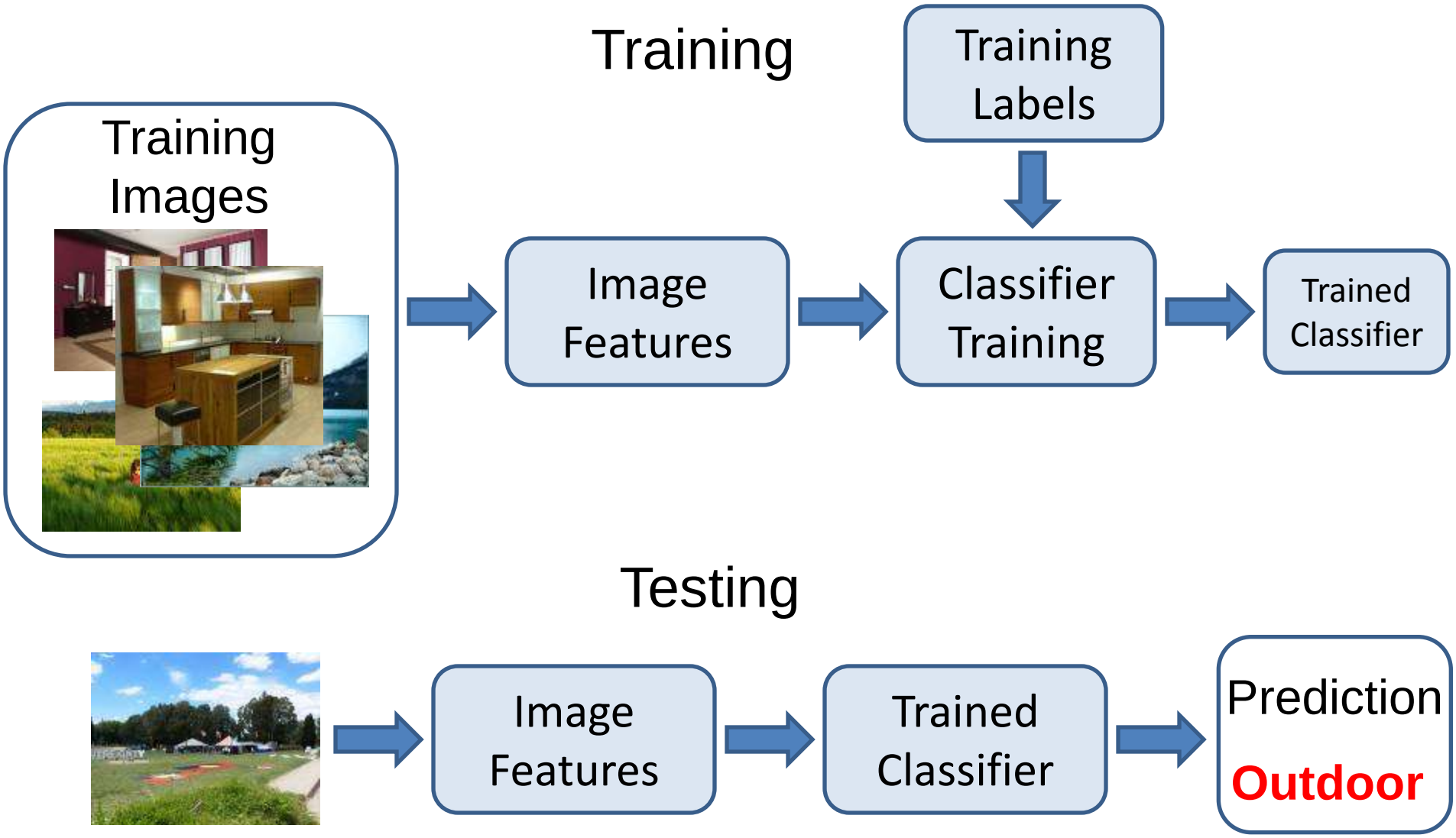
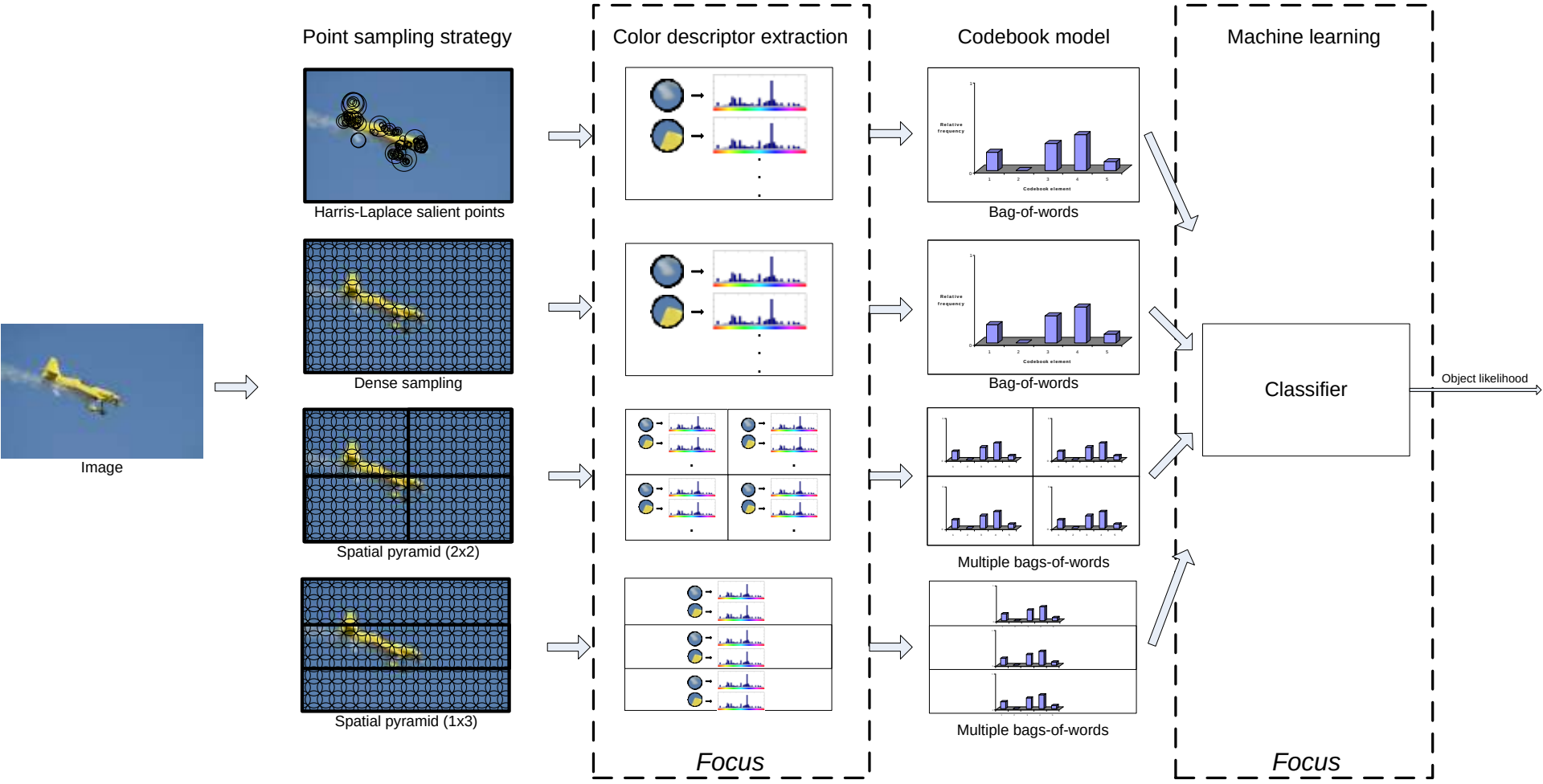


Image Categorization



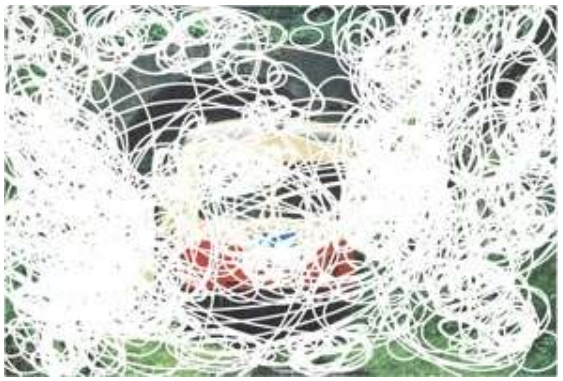
Derek Hoiem

Pipeline Overview

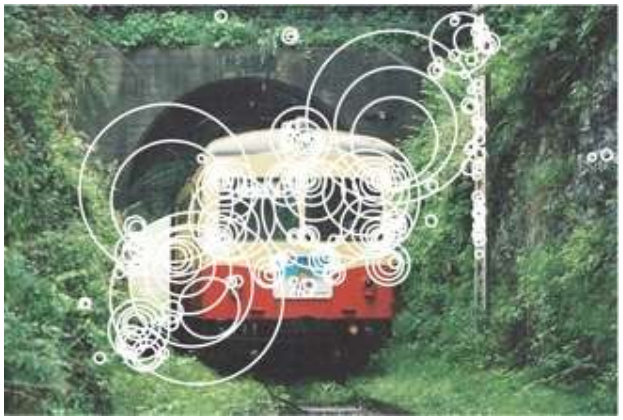


Original Image

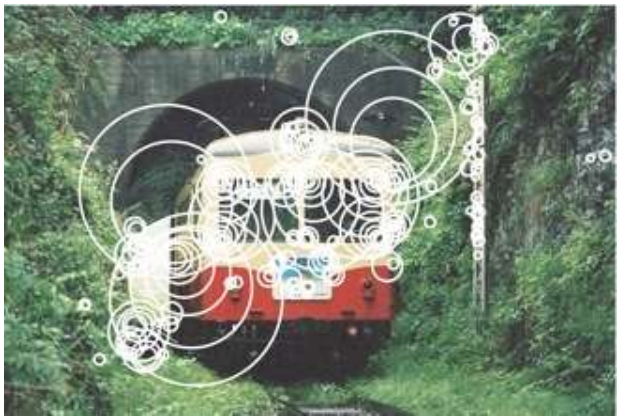
Harris Laplacian
impl. by Mikolajczyk (e.g. CVPR06)



Shape adapted Harris Laplacian
impl. by Mikolajczyk (ICCV07)



Color salient points
Quasi invariant HSI

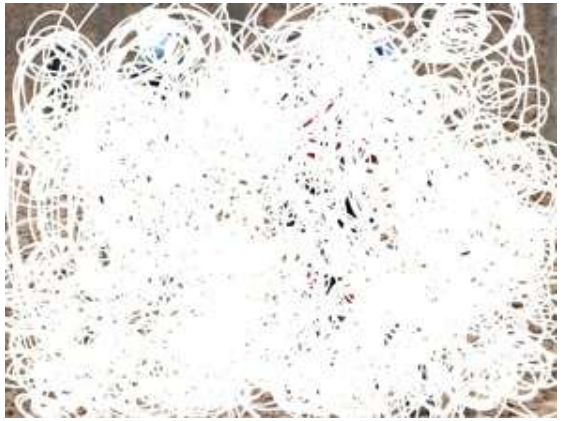


Color salient points
Color boosted OCS

Most of the time, both color approaches agree on the most salient parts of an image.

Original Image

Harris Laplacian
impl. by Mikolajczyk (e.g. CVPR06)



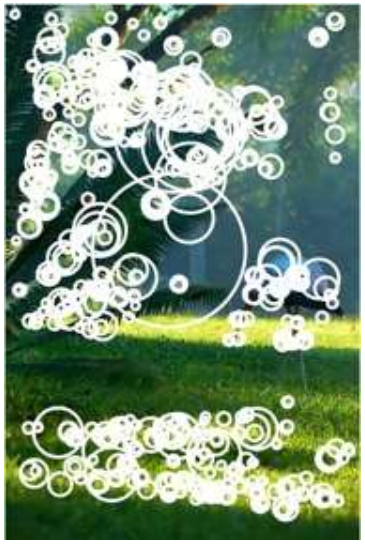
Shape adapted Harris Laplacian
impl. by Mikolajczyk (ICCV07)

Color salient points
Quasi invariant HSI

Color salient points
Color boosted OCS

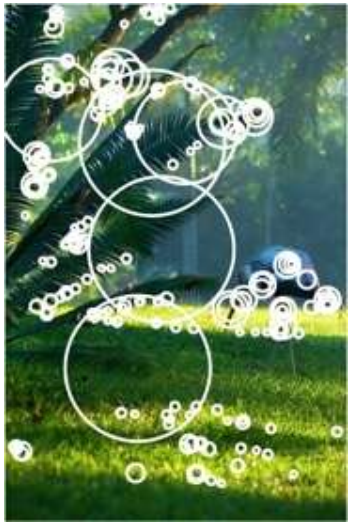
Structured backgrounds of same color tones and shadowing effects are discarded effectively.

Original Image

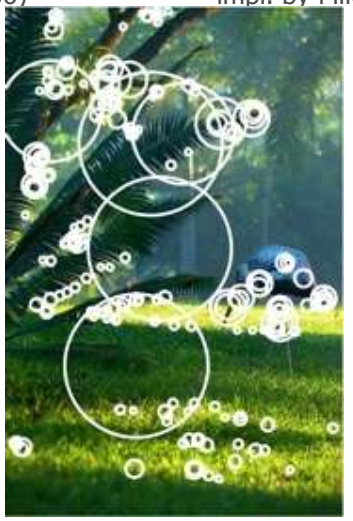


Harris Laplacian
impl. by Mikolajczyk (e.g. CVPR06)

Shape adapted Harris Laplacian
impl. by Mikolajczyk (ICCV07)



Color salient points
Quasi invariant HSI



Color salient points
Color boosted OCS

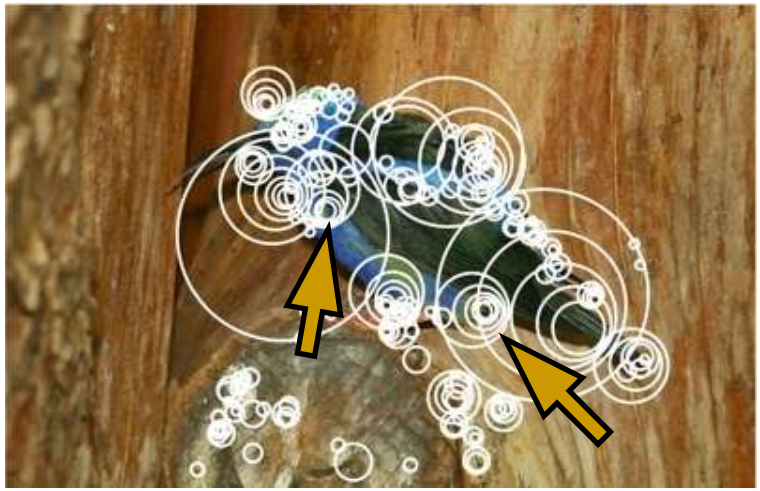
Illumination invariance shifts the features to real color differences - shadows are less salient.

Original Image

Harris Laplacian
impl. by Mikolajczyk (e.g. CVPR06)

Shape adapted Harris Laplacian
impl. by Mikolajczyk (ICCV07)

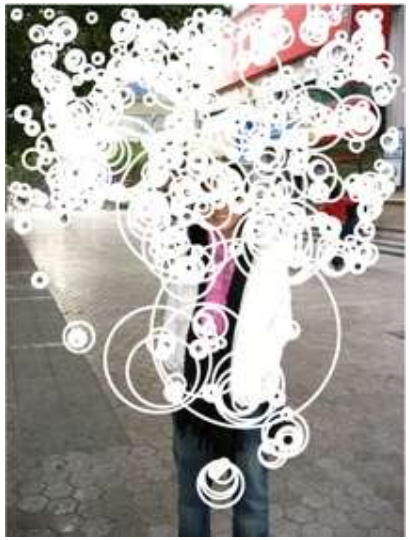
Color salient points
Quasi invariant HSI



Color salient points
Color boosted OCS

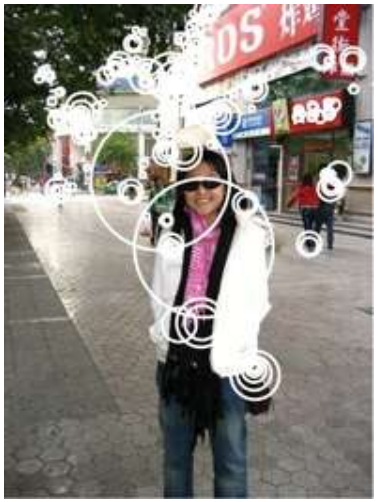
Real color edges are prioritized for HSI, rare colors are boosted in OCS.

Original Image



Harris Laplacian
impl. by Mikolajczyk (e.g. CVPR06)

Shape adapted Harris Laplacian
impl. by Mikolajczyk (ICCV07)

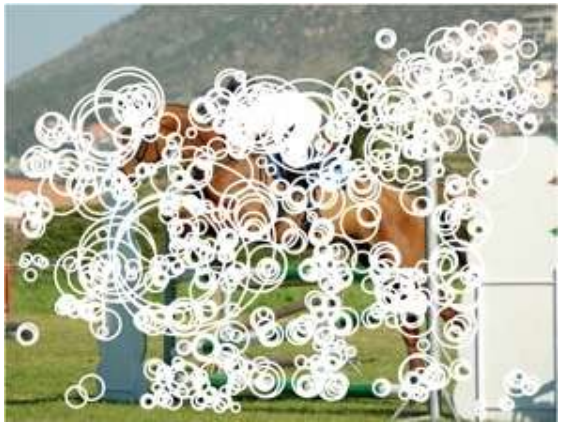


Color salient points
Quasi invariant HSI

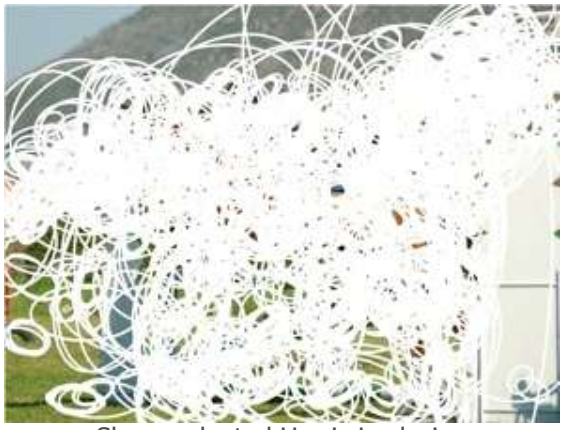
Color salient points
Color boosted OCS

Very colorful background with high contrast is challenging.

Original Image

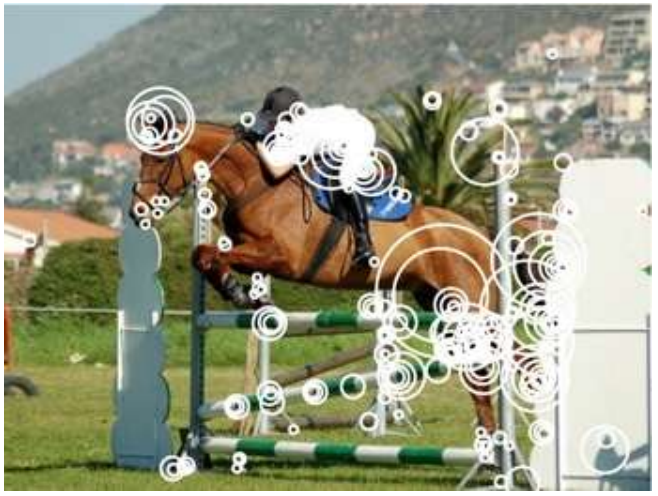


Harris Laplacian
impl. by Mikolajczyk (e.g. CVPR06)



Shape adapted Harris Laplacian
impl. by Mikolajczyk (ICCV07)

Color salient points
Quasi invariant HSI



Color salient points
Color boosted OCS

In the end, we can discard ~50% of the data maintaining the same quality of the local description.

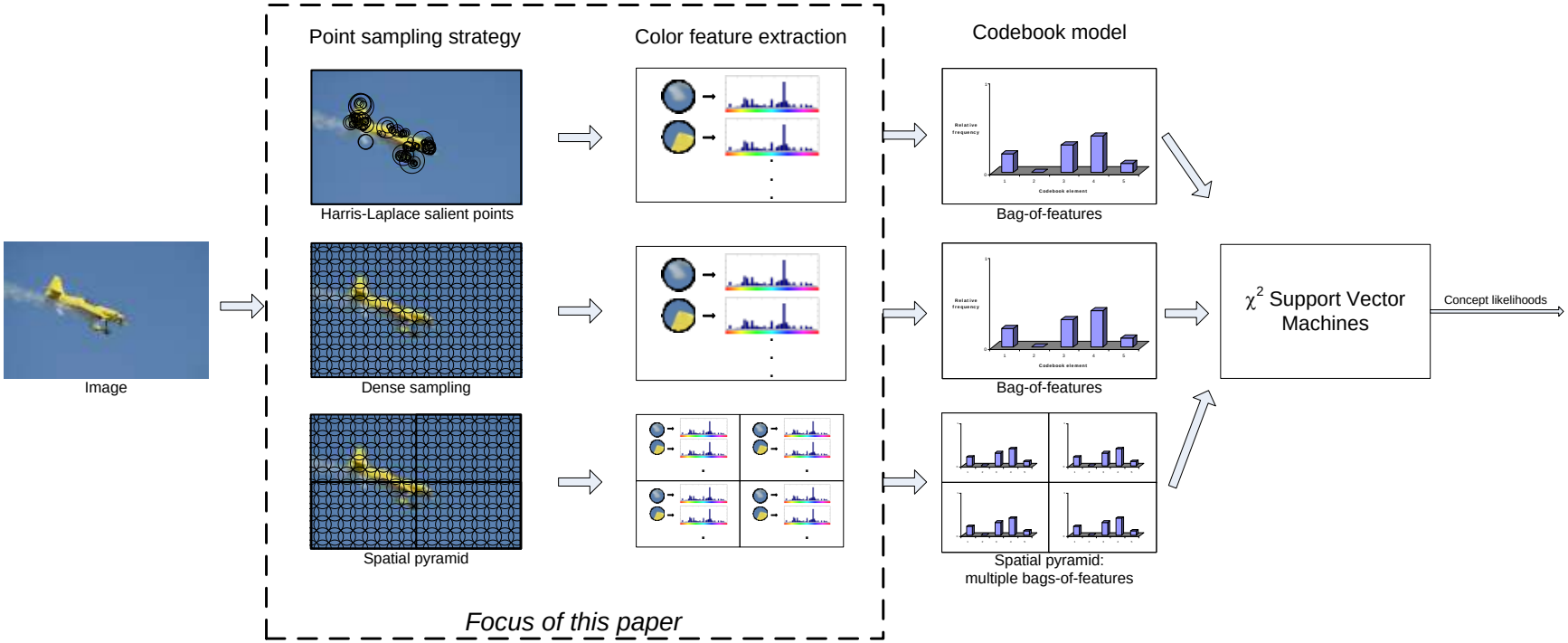
Distinctiveness

Distinctiveness studied experimentally:

- Image benchmark: PASCAL Visual Object Classes Challenge 2007
- 9963 photos from Flickr
- 20 object types
- Earth Movers Distance (EMD) between cluster sets of different images, used in EMD kernel function for SVM [ZhangIJCV2007]

PASCAL VOC 2007/2008

Codebook size=4000



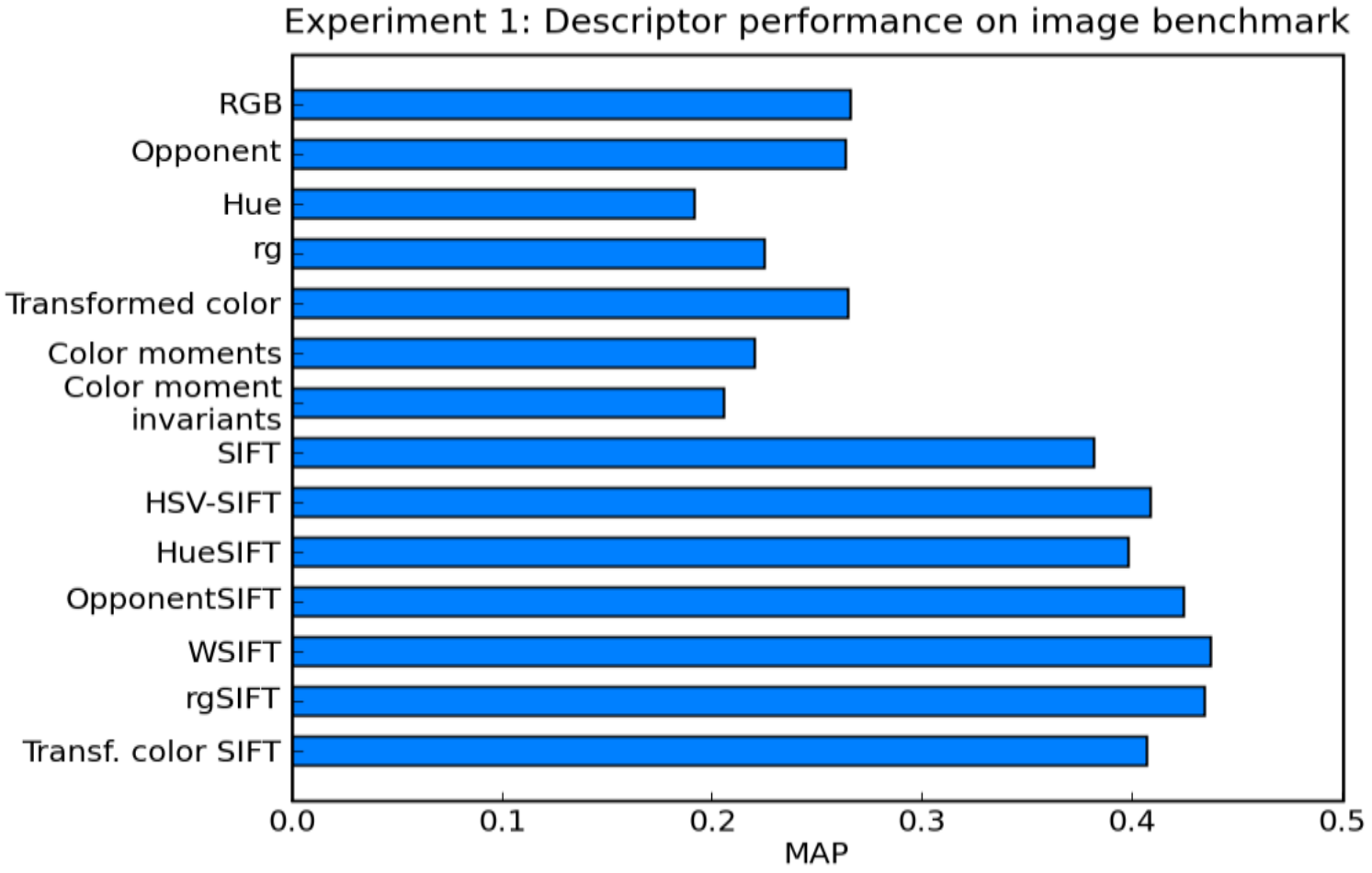
Point sampling
Harris-Laplace
Dense sampling

Spatial Pyramid
1x1
2x2
1x3

Color Descriptor
SIFT
OpponentSIFT
WSIFT
rgSIFT
Transformed color SIFT

[van de Sande, IEEE PAMI, 09]

Results on PASCAL VOC 2007

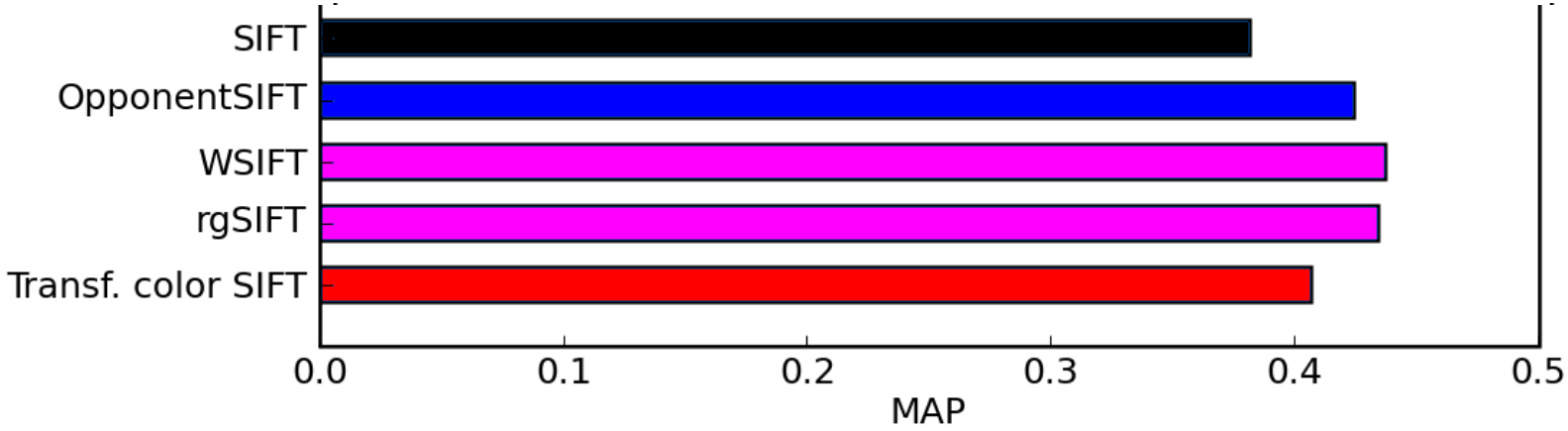


Color Descriptor Taxonomy

[van de Sande, IEEE PAMI, 09]

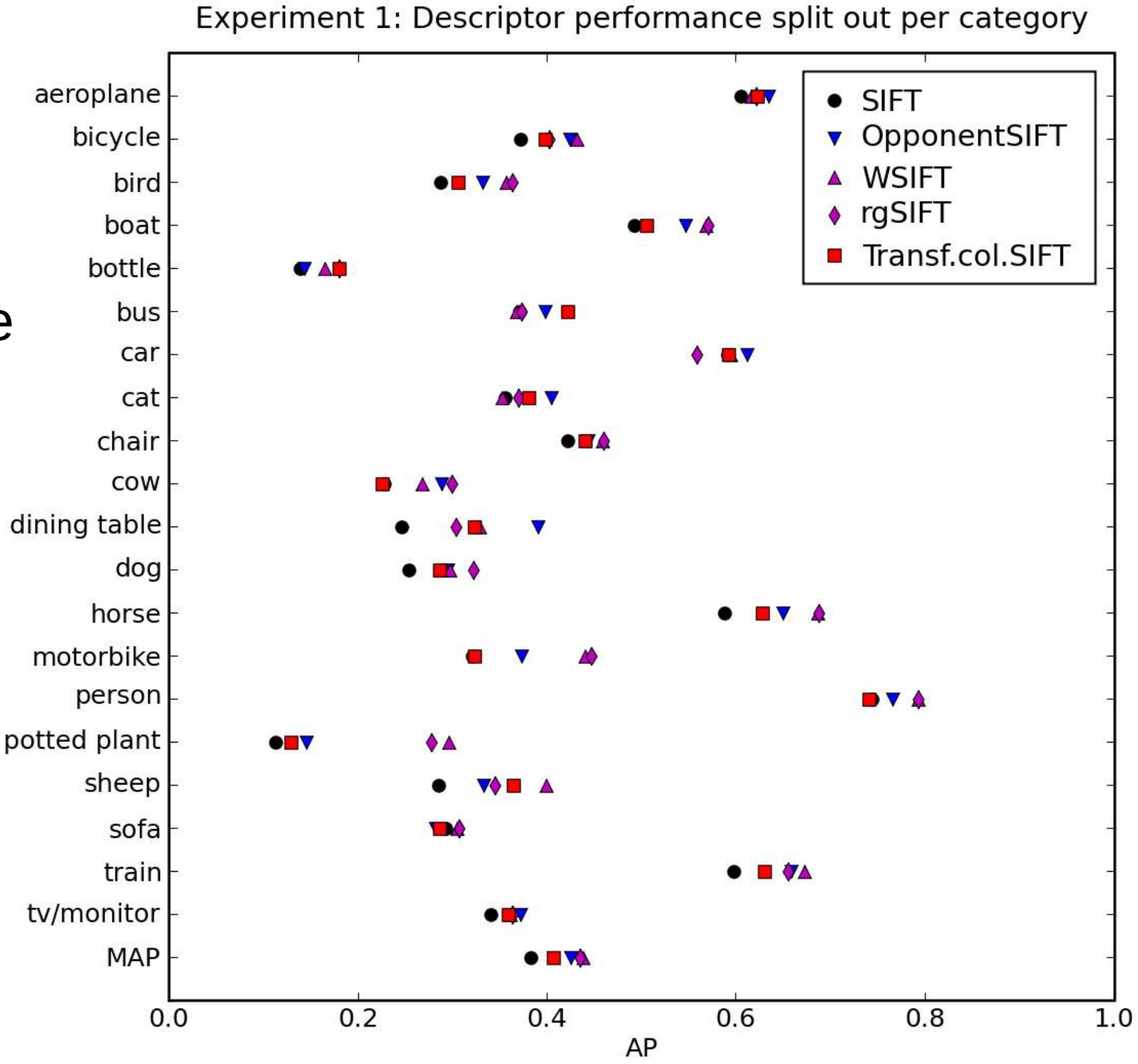
	Light intensity change $\begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$	Light intensity shift $\begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$	Light intensity change and shift $\begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$	Light color change $\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$	Light color change and shift $\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_2 \\ o_3 \end{pmatrix}$
RGB Histogram	-	-	-	-	-
O_1, O_2	-	+	-	-	-
O_3, Intensity	-	-	-	-	-
Hue	+	+	+	-	-
Saturation	+	+	+	-	-
r, g	+	-	-	-	-
Transformed color	+	+	+	+	+
Color moments	-	+	-	-	-
Moment invariants	+	+	+	+	+
SIFT (∇I)	+	+	+	+	+
HSV-SIFT	+	+	+	+/-	+/-
HueSIFT	+	+	+	+/-	+/-
OpponentSIFT	+/-	+	+/-	+/-	+/-
W-SIFT	+	+	+	+/-	+/-
rgSIFT	+	+	+	+/-	+/-
Transf. color SIFT	+	+	+	+	+

Results on PASCAL VOC 2007



	Light intensity change $\begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$	Light intensity shift $\begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$	Light intensity change and shift $\begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$	Light color change $\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$	Light color change and shift $\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_2 \\ o_3 \end{pmatrix}$
SIFT	+	+	+	+	-
OpponentSIFT	+	+	+	-	-
WSIFT	+	+	+	-	-
rgSIFT	+	+	+	-	-
RGB SIFT	+	+	+	+	+

Scale-invariance w.r.t. light intensity important



Conclusion

- #1 model for the VOC:

Light intensity change and shift

$$\begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$$

- You need scale-invariance and shift-invariance w.r.t. light intensity
- Invariance to light color is not needed and decreases the discriminative power

VOC2007 results –airplanes (1)

VOC2007 results – airplanes (2)

VOC2007 results –persons (1)

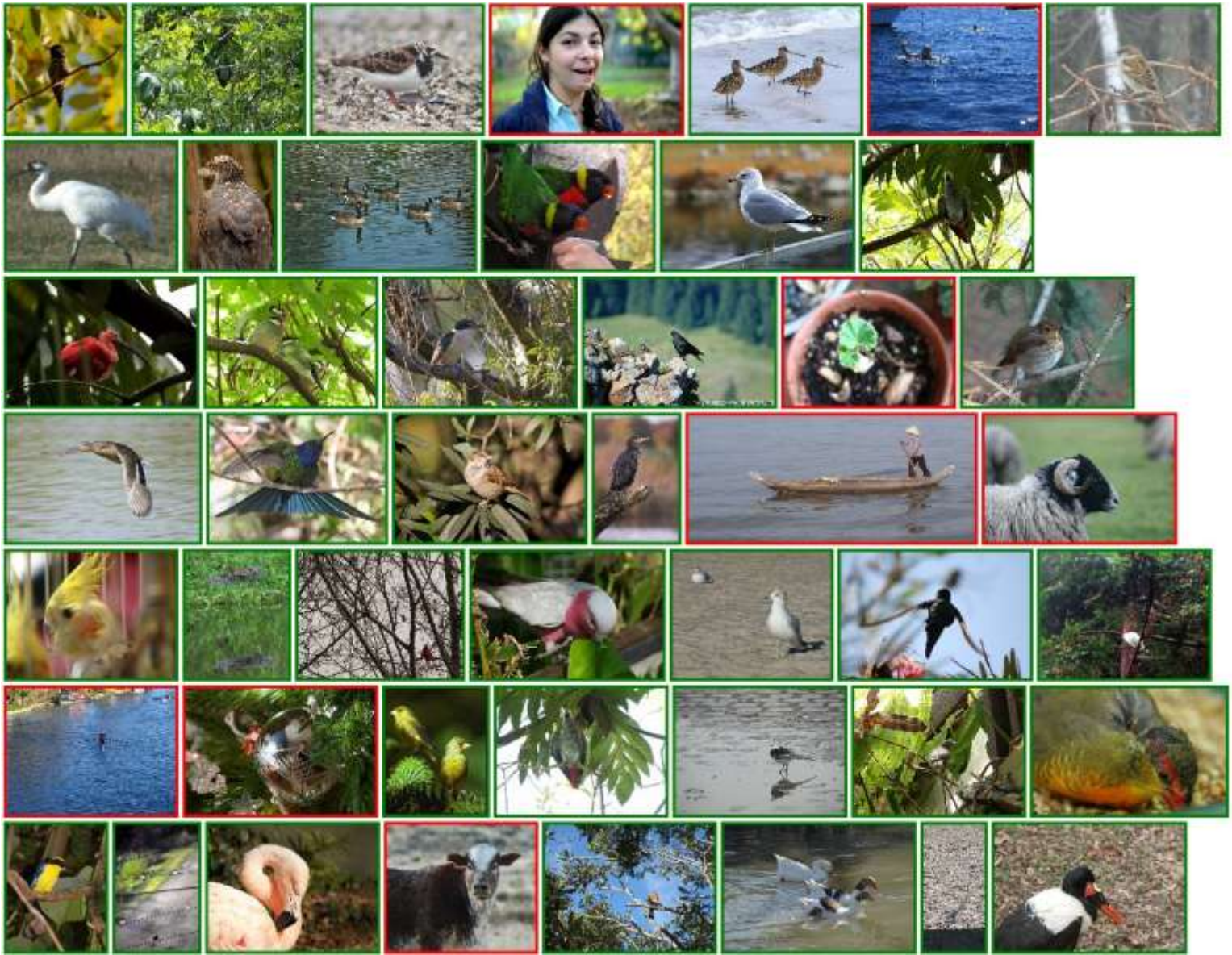
VOC2007 results – persons (2)

VOC2007 results –horses (1)

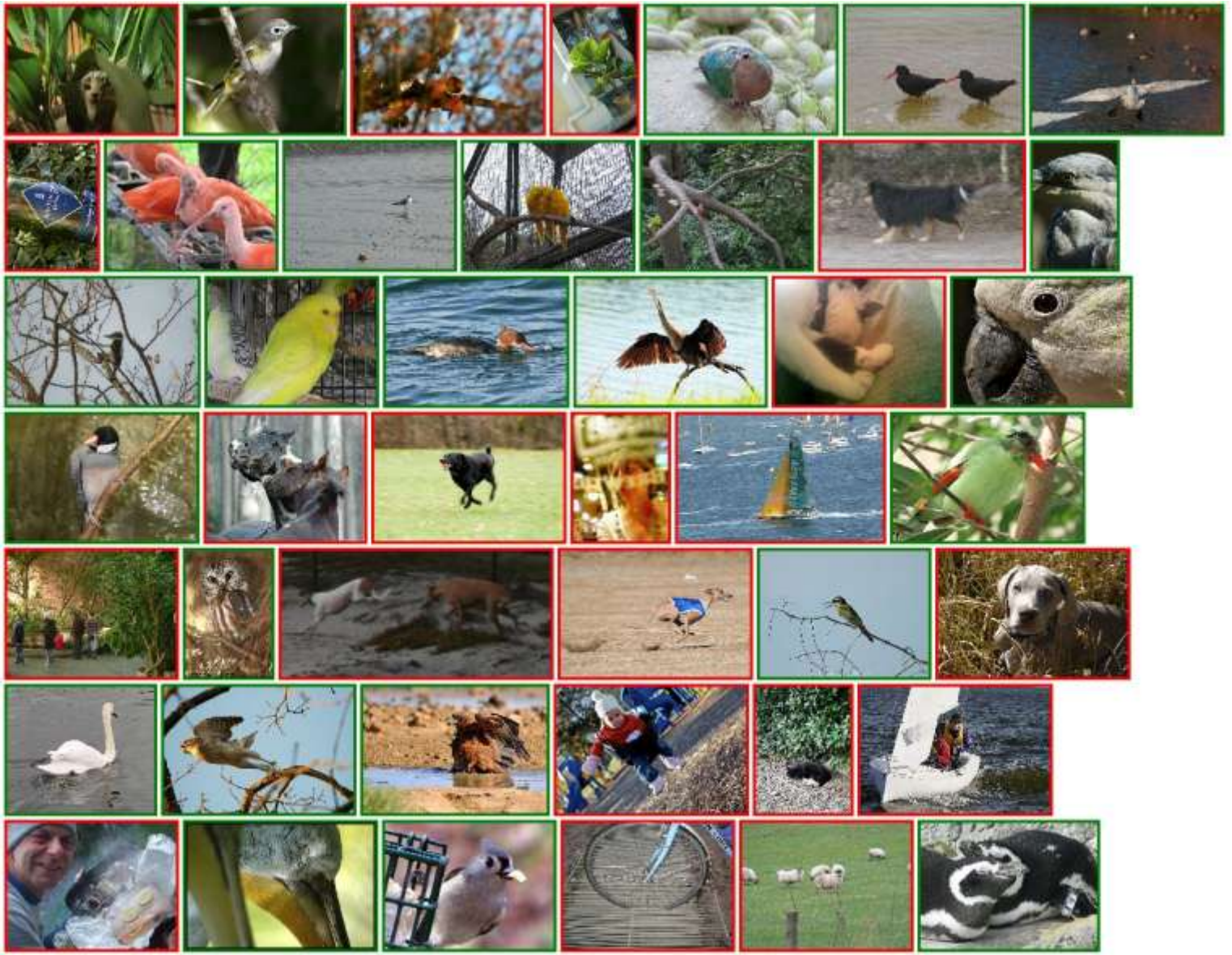
VOC2007 results – horses (2)

VOC2007 results –dogs (1)

VOC2007 results – birds(1)



VOC2007 results – birds (2)



Color Descriptors on VOC08

- Invariance properties of the descriptors used

	Light intensity change $\begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$	Light intensity shift $\begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$	Light intensity change and shift $\begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$	Light color change $\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$	Light color change and shift $\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_2 \\ o_3 \end{pmatrix}$
SIFT	+	+	+	+	+
OpponentSIFT	+	+	+	-	-
WSIFT	+	+	+	-	-
*rg*SIFT	+	+	+	-	-
RGBSIFT	+	+	+	+	+

Descriptors	MAP on VOC2008val
Intensity SIFT	42,3
All five (=Soft5ColorSIFT)	45,5

By adding color: **+8%**

TRECVid

Koen van de Sande

Cees Snoek

Jan van Gemert

Jasper Uijlings

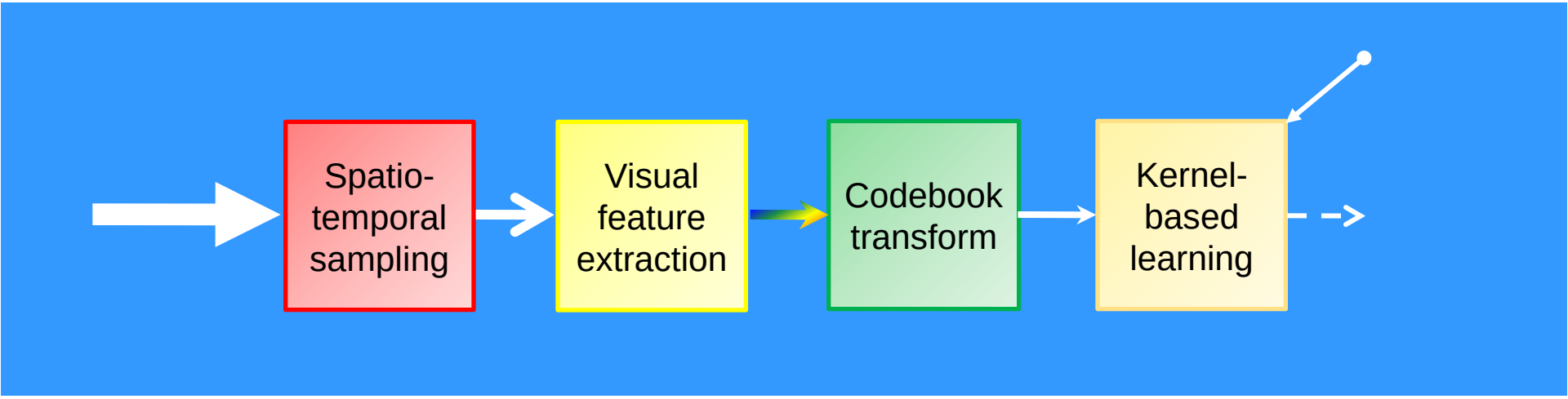
Jan-Mark Geusebroek

Theo Gevers

Arnold Smeulders

University of Amsterdam

Concept Detection Stages

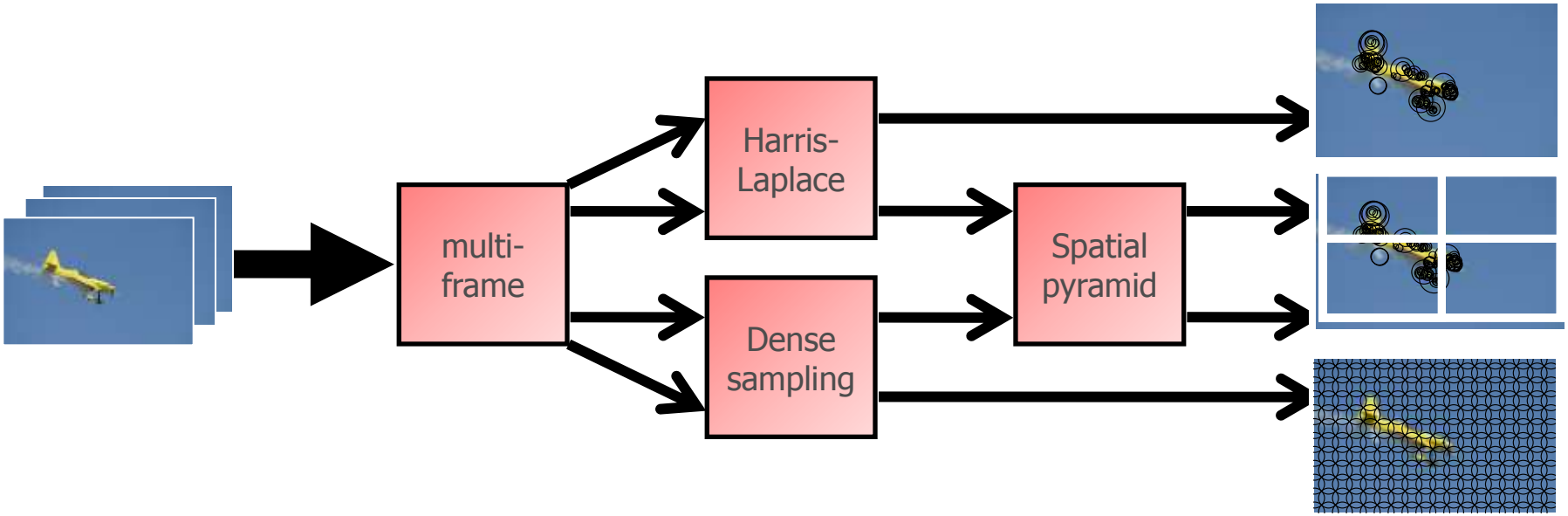


[Lazebnik, CVPR06]
[Marszalek, VOC07]

Spatio-Temporal Sampling

- Spatial pyramid
  - 1x1 whole image
  - 2x2 image quarters
  - 1x3 horizontal bars

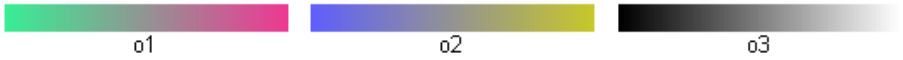
- Temporal analysis of up to 5 frames per shot



Invariant Visual Descriptors

Color SIFT:

- Intensity-based SIFT
- OpponentSIFT
- C-SIFT
- *rg*SIFT
- Transformed color SIFT



- Add color, but also keep intensity information

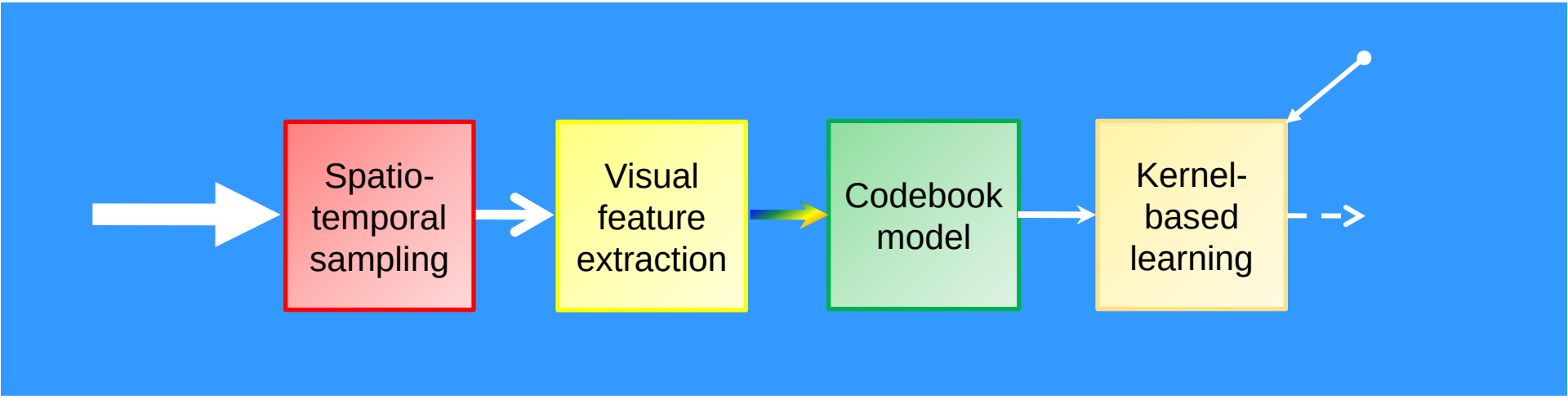
Visual Descriptors	MAP on TV2007test
Intensity SIFT	0,144
5x Color SIFT	0,155

relative **+8%**

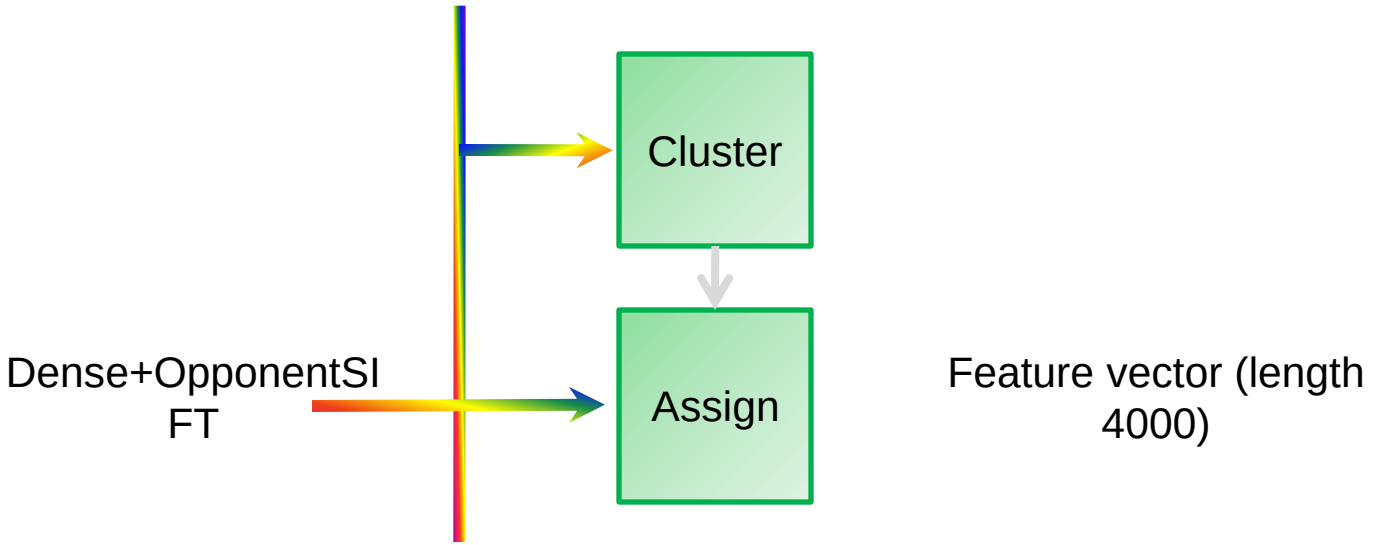
TV2007test results:

- Trained on TRECVID2007 development set
- Evaluated on TRECVID2007 test set
- TRECVID2007 development + test = 2008 development

Concept Detection Stages



Visual Codebook Model



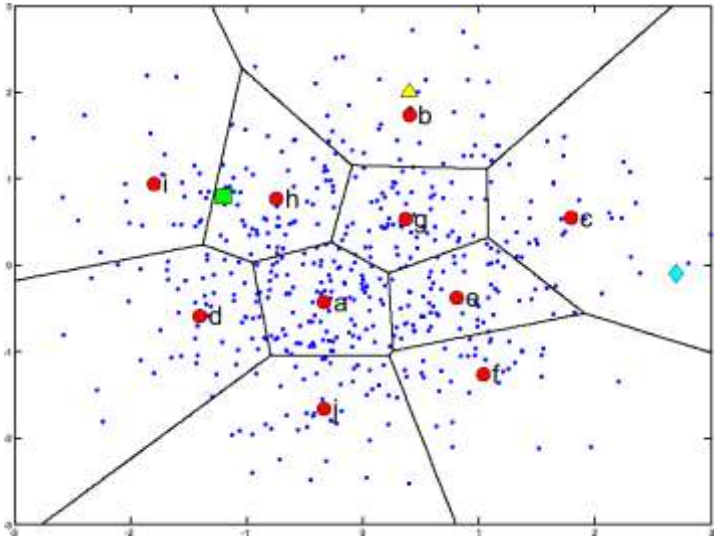
- Codebook consists of codewords
- Constructed with k-means clustering on descriptors
- We use 4,000 codewords per codebook

[van Gemert, ECCV08]

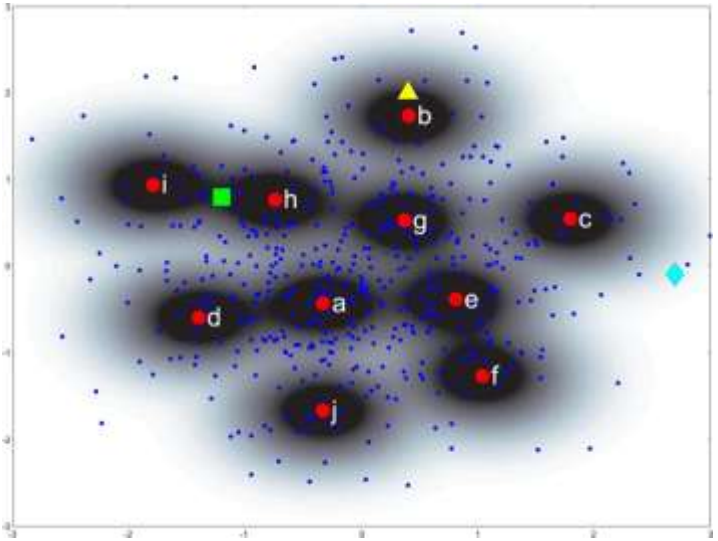
Codebook Assignment

- Codeword

Soft assignment using Gaussian kernel



Hard assignment

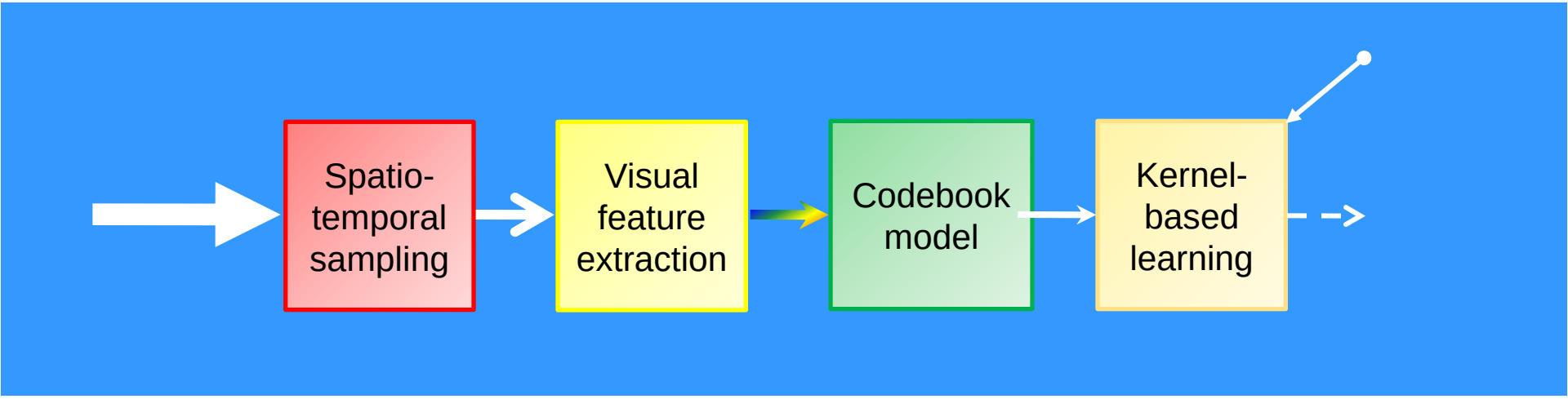


Soft assignment

Assignment	MAP on TV2007test
Hard	0,155
Soft	0,166

relative +7%

Concept Detection Stages



[Snoek, ICME05]

Robust Temporal Approach

- No cloud computing yet: need to be efficient ↑↑
- Process 5 frames per shot in test set
- Linear increase in computation: x5

Codebook library	Frames/shot	MiAP on TV2008test
3x Color SIFT	1	0,152
3x Color SIFT	5	0,184

relative **+20%**

- In 2005 paper 7.5% to 38% improvement noted for multi-frame (worst-case vs. best-case using oracle)
- **Robust color SIFT *with* temporal = ~20% improvement**

The Good

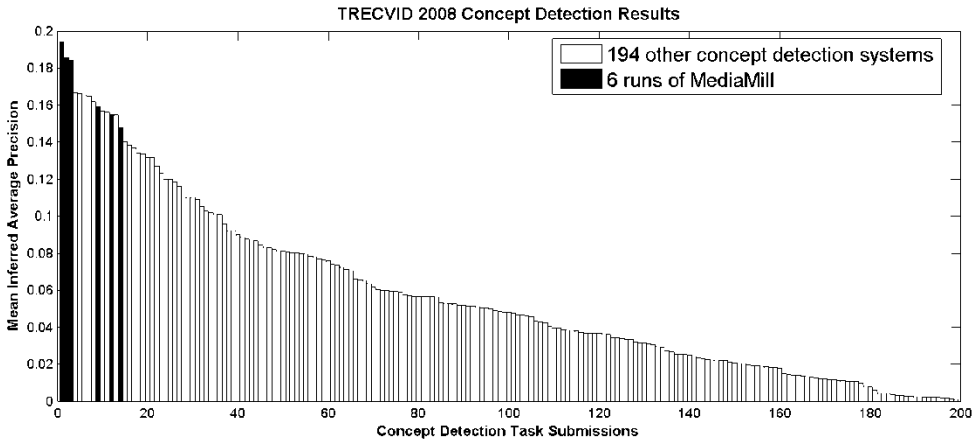
- Close-up of hands

- 

- 

Conclusions

- Illumination conditions affect concept detection
- SIFT+colorSIFT improves ~8%
- Soft codebook assignment improves ~7%
- Robust colorSIFT with simple multi-frame improves ~20%:
  - Room for more advanced methods in TRECVID 2009
- Precomputed kernel matrix reduces SVM computation time
- Near-duplicates from trailers hamper progress:
  - We suggest to exclude them, or count only once



Discussion

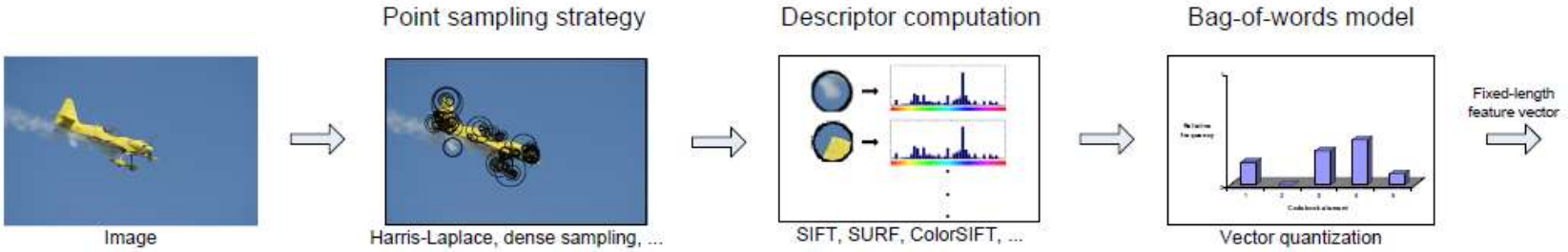
- Usefulness of scale-invariance w.r.t. light intensity depends on dataset and concept
  (MM: indoor/outdoor distinction)
- Almost all descriptors are shift-invariant
  ➘ no adverse effects, at least
- Invariance to light color changes and shifts is domain-specific.

Conclusion

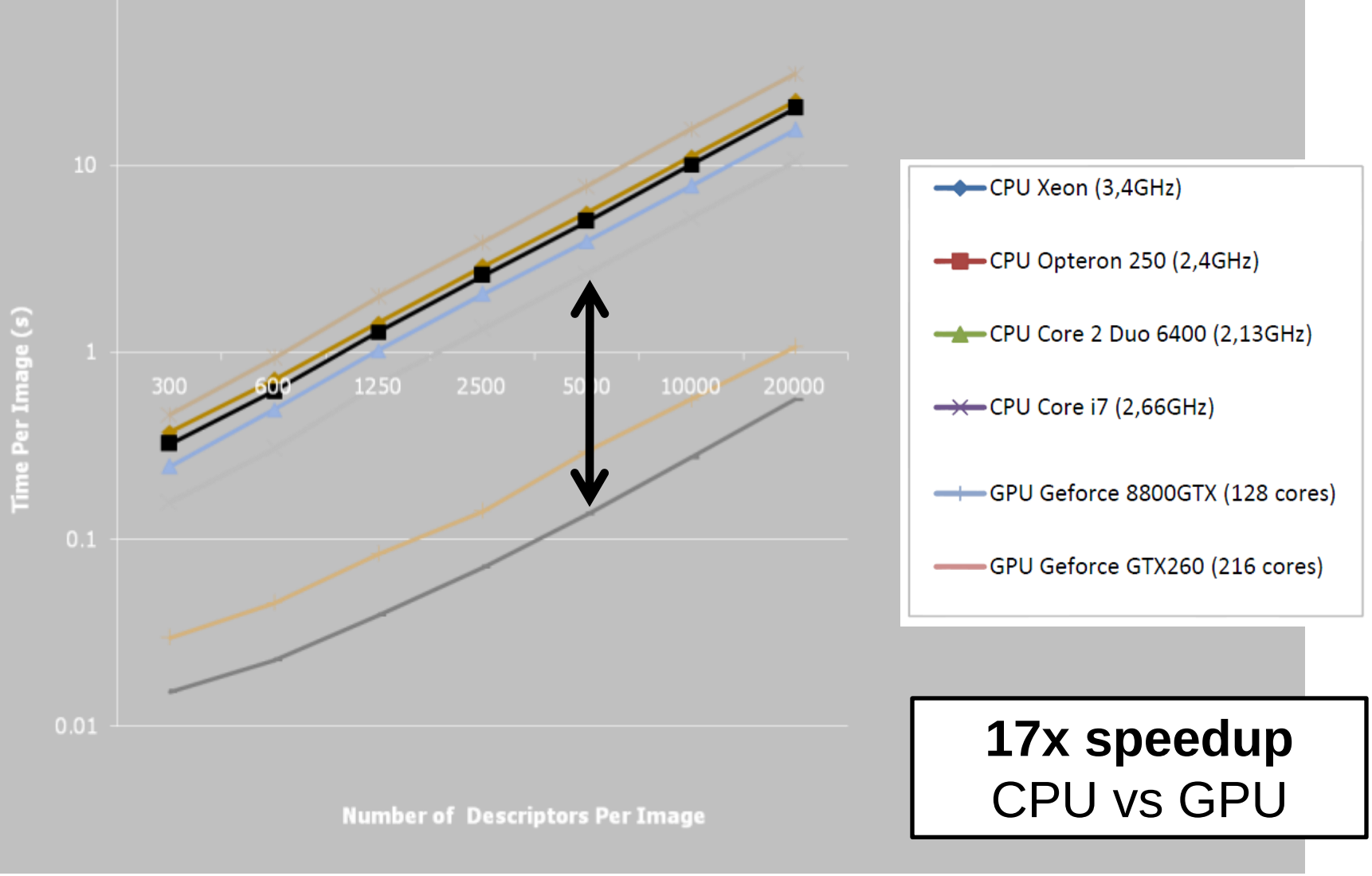
- Bag-of-word approach works
- Good local descriptors: SIFT, OpponentSIFT, rgSIFT/WSIFT, RGB SIFT
- Combining these color features gives state-of-the-art performance: 1ste position in VOC08 and second in VOC09.
- Drawback: computational costs of bag-of-word approach

GPU-Accelerated Feature Extraction

- Single bag-of-words feature up to 15s/frame (CPU-time)
- TRECVid 2008 / PASCAL VOC 2008 consortium entries used 10 of these features
- More than 80% of time spent in vector quantization



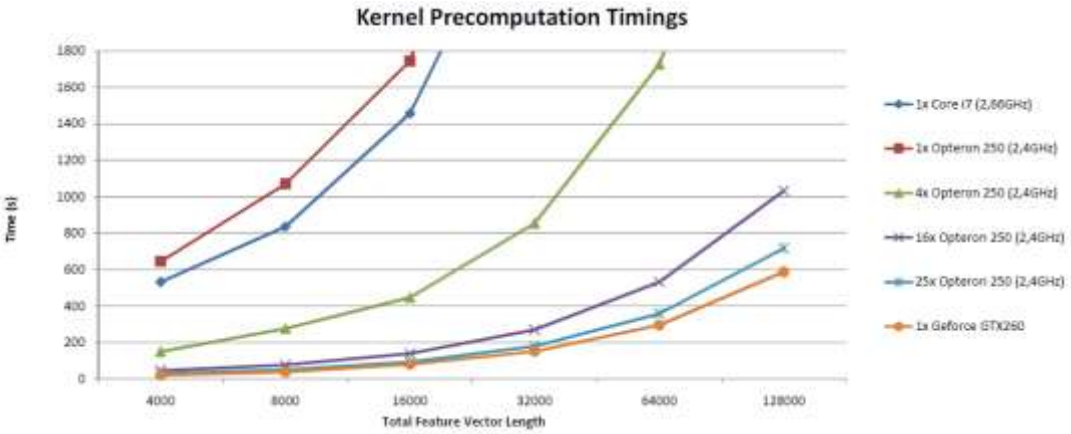
Vector Quantization Timings for ColorSIFT



17x speedup
CPU vs GPU

Kernel Value Precomputation

- Step from image feature vectors to kernel-based classifiers from WP5 (SVM/SR-KDA)
- Computes χ^2 distance between pairs of images
- Suitable for GPU implementation: **22x speedup**
- TRECVid 2008 processing time: 800 CPU hours vs. 37 GPU hours



- Process datasets order of magnitude larger

or

- Single GPU replaces medium-sized cluster

Conclusions

- Large scale datasets with annotations
- Color and photometric invariance needed
- Balance between discriminative power and invariance
- Color add information to classification achieving best performance in VOC08/VOC09, TRECVid08/TRECVid09 and ImageCLEF.
- Speed up is required (e.g. GPU)
- Higher semantics like aggression, emotions etc.