RWTH AACHEN UNIVERSITY

Computer Vision – Lecture 10

Sliding-Window based Object Detection

27.11.2014

Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de

leibe@vision.rwth-aachen.de

Course Outline

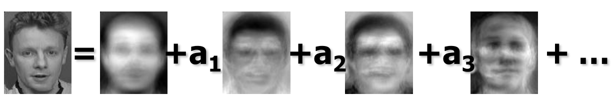
- Image Processing Basics
- Segmentation & Grouping
- Recognition
  - Global Representations
- Object Categorization I
  - Sliding Window based Object Detection
- Local Features & Matching
- Object Categorization II
  - Part based Approaches
- 3D Reconstruction
- Motion and Tracking

Recap: Subspace Methods

Subspace methods

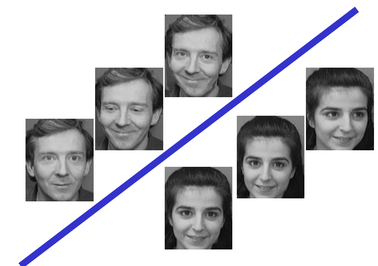
Reconstructive | Discriminative

PCA, ICA, NMF



representation

FLD, SVM, CCA



classification
regression

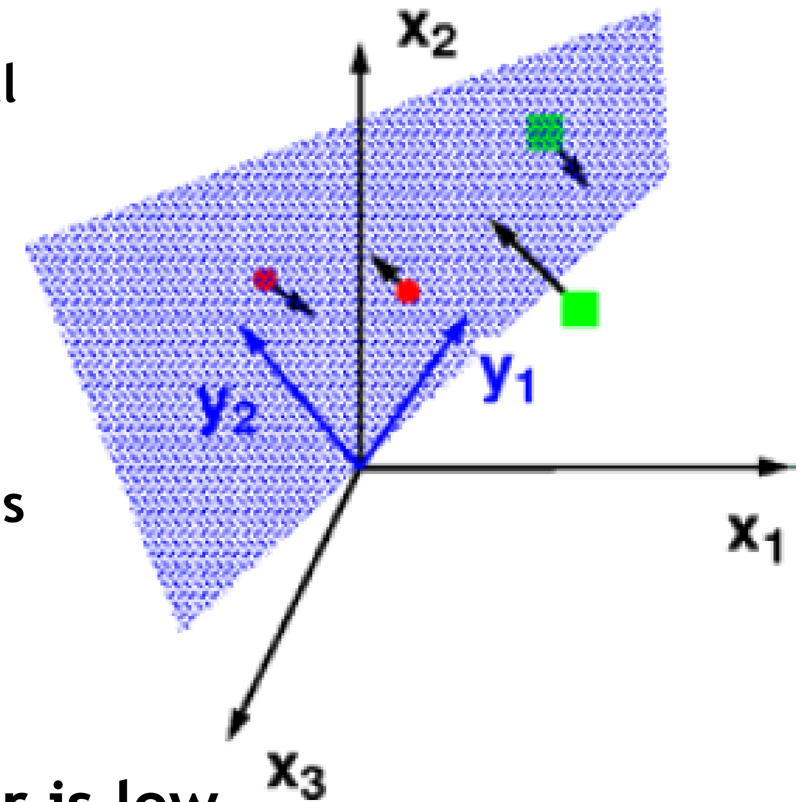
Recap: Obj. Detection by Distance TO Eigenspace

- For each test image, compute the reprojection error
  - An n-pixel image $x \in \mathbb{R}^n$ can be projected to the low-dimensional feature space $y \in \mathbb{R}^m$ by

$$y = Ux$$

  - From $y \in \mathbb{R}^m$, the reconstruction of the point is $\underline{U}^T y$
  - The error of the reconstruction is

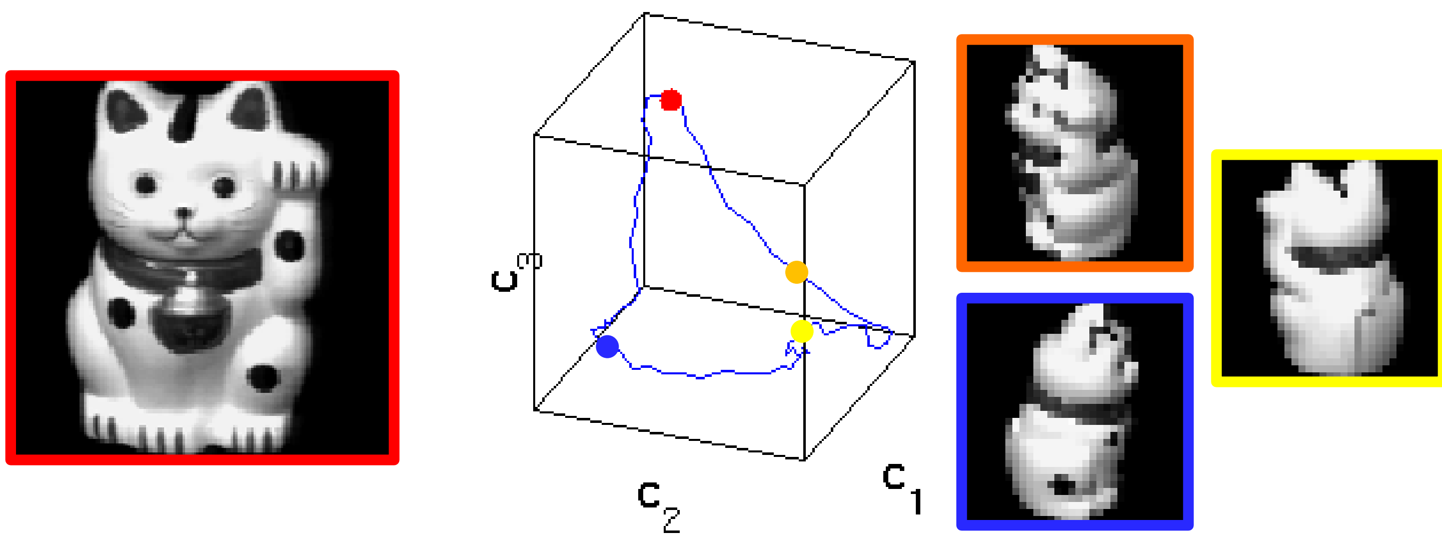
$$\left\| x - U^T U x \right\|$$



- Accept a detection if this error is low.
  - Assumption: subspace is optimized to the target object (class).
  - Other classes are not represented well ⇒ large error.

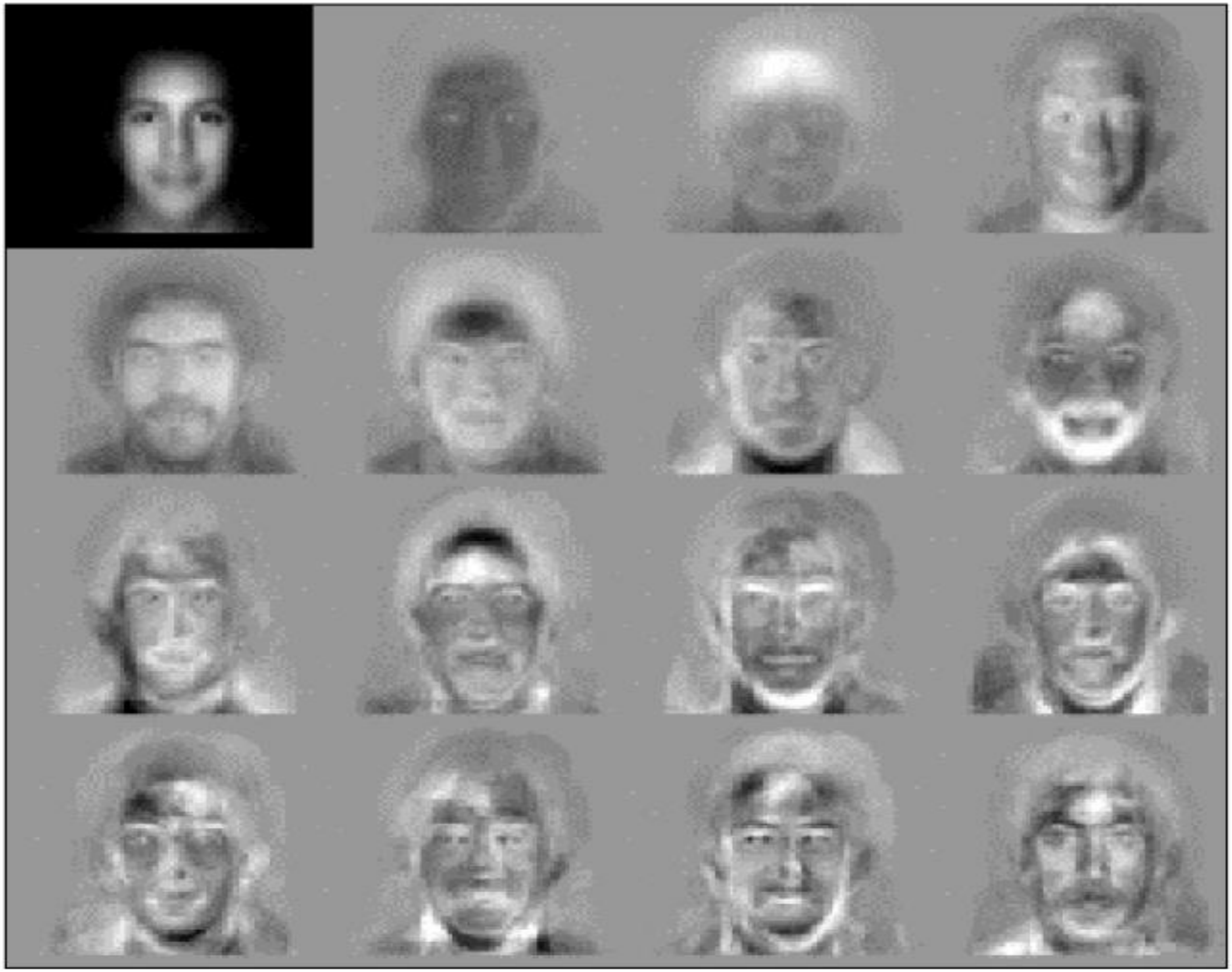
Recap: Obj Identification by Distance IN Eigenspace

- **Objects are represented as coordinates in an n-dim. eigenspace.**
- **Example:**
  - **3D space with points representing individual objects or a manifold representing parametric eigenspace (*e.g.*, orientation, pose, illumination).**



- **Estimate parameters by finding the NN in the eigenspace**

Recap: Eigenfaces



Slide credit: Peter Belhumeur

Important Footnote

- We've derived PCA by computing the eigenvectors of Σ…
- *Don't implement PCA this way!*
  - Why?

1. How big is Σ?
   - $n \times n$, where n is the number of pixels in an image!
   - However, we only have m training examples, typically $m << n$.

   $\Rightarrow \Sigma$ will at most have rank m!

2. You only need the first k eigenvectors

Slide credit: Peter Belhumeur

Singular Value Decomposition (SVD)

- **Any m×n matrix A may be factored such that**

$$A = U\Sigma V^T$$

$$[m\times n] = [m\times m][m\times n][n\times n]$$

- U: $m\times m$, **orthogonal matrix**
  - **Columns of** U **are the eigenvectors of** AA^T
- V: $n\times n$, **orthogonal matrix**
  - **Columns are the eigenvectors of** A^TA
- Σ: $m\times n$, **diagonal with non-negative entries** ($\sigma_1, \sigma_2, \ldots, \sigma_s$) **with** $s = \min(m,n)$ **are called the singular values.**
  - **Singular values are the square roots of the eigenvalues of both** AA^T **and** A^TA. ***Columns of*** U ***are corresponding eigenvectors!***
  - **Result of SVD algorithm:** $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_s$

Slide credit: Peter Belhumeur

Performing PCA with SVD

- Singular values of A are the square roots of eigenvalues of both AA^T and A^TA.
  - Columns of U are the corresponding eigenvectors.

- And $\sum_{i=1}^{n} a_i a_i^T = [a_1 \ \ldots \ a_n][a_1 \ \ldots \ a_n]^T = AA^T$

- Covariance matrix

$$\Sigma = \frac{1}{n}\sum_{i=1}^{n}(\vec{x}_i - \vec{\mu})(\vec{x}_i - \vec{\mu})^T$$

- So, ignoring the factor $1/n$, subtract mean image μ from each input image, create data matrix $A = (\vec{x}_i - \vec{\mu})$, and perform SVD on the data matrix.
  - And you're done.

SVD Properties

- Matlab: `[u s v] = svd(A)`
  - where $A = u*s*v'$
- `r = rank(A)`
  - Number of non-zero singular values
- U, V give us orthonormal bases for the subspaces of A
  - first r columns of U: *column space* of A
  - last m-r columns of U: *left nullspace* of A
  - first r columns of V: *row space* of A
  - last n-r columns of V: *nullspace* of A
- For $d \leq r$, the first d columns of U provide the best d-dimensional basis for columns of A in least-squares sense

Limitations

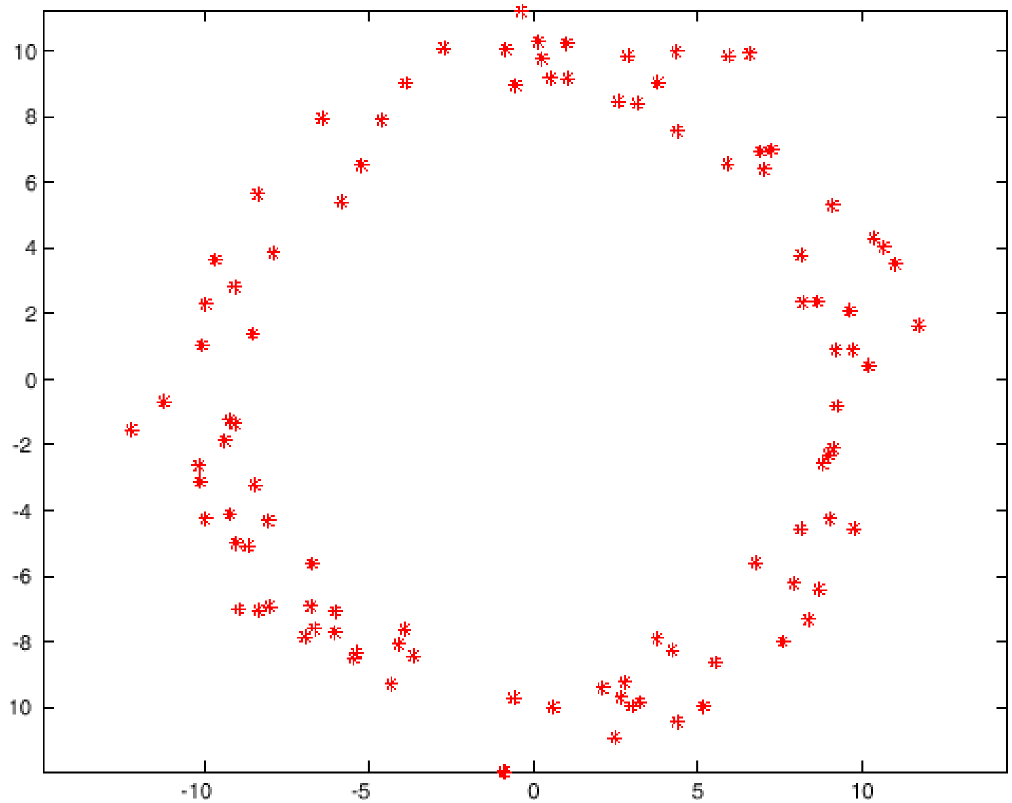
- Global appearance method: not robust to misalignment, background variation

- Easy fix (with considerable manual overhead)
  - Need to align the training examples

RWTH AACHEN UNIVERSITY

Limitations

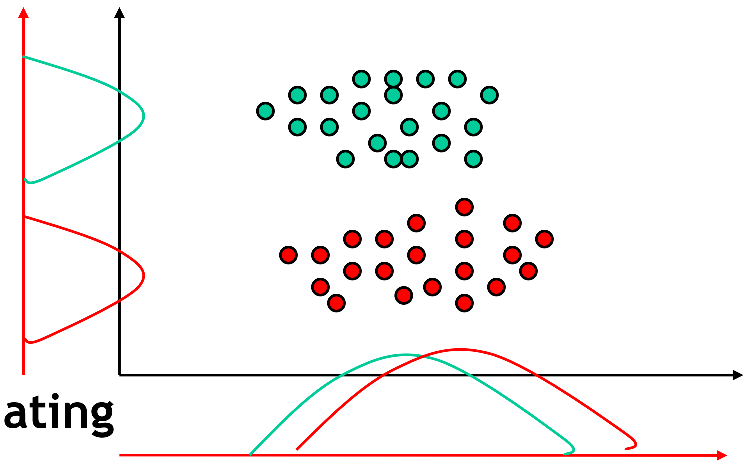
- **PCA assumes that the data has a Gaussian distribution (mean μ, covariance matrix Σ)**



  - **The shape of this dataset is not well described by its principal components**

Restrictions of PCA

- PCA minimizes projection error

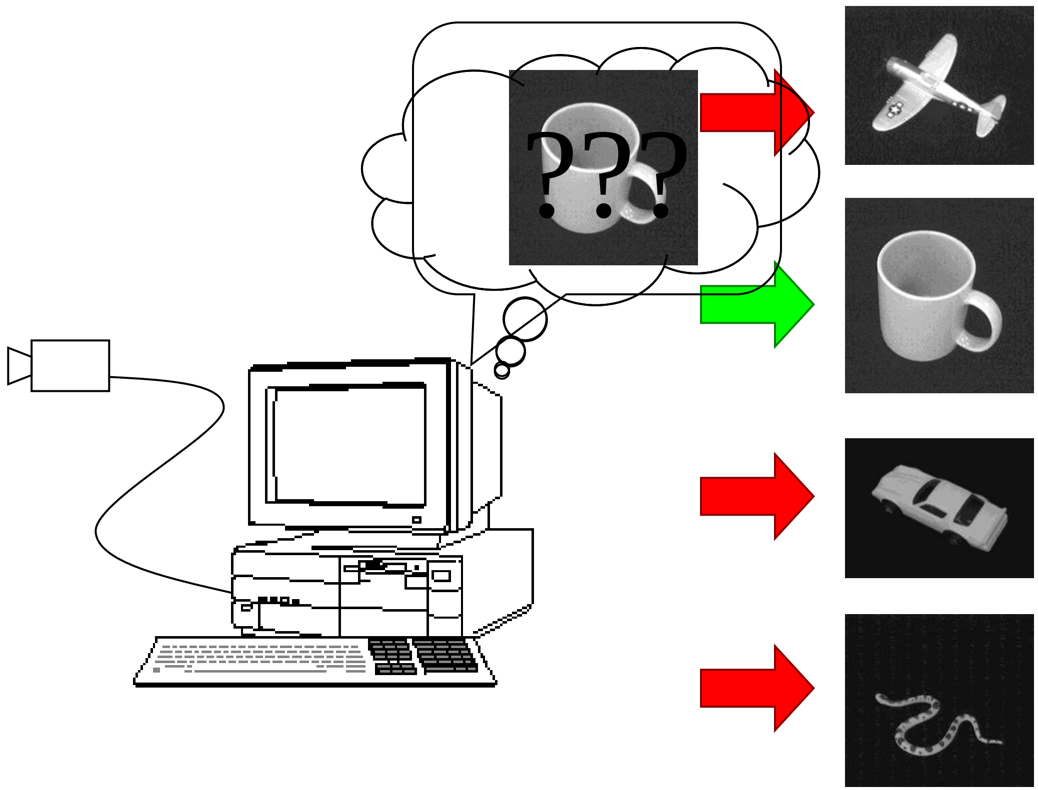


- PCA is „unsupervised“ no information on classes is used
- Discriminating information might be lost

Topics of This Lecture

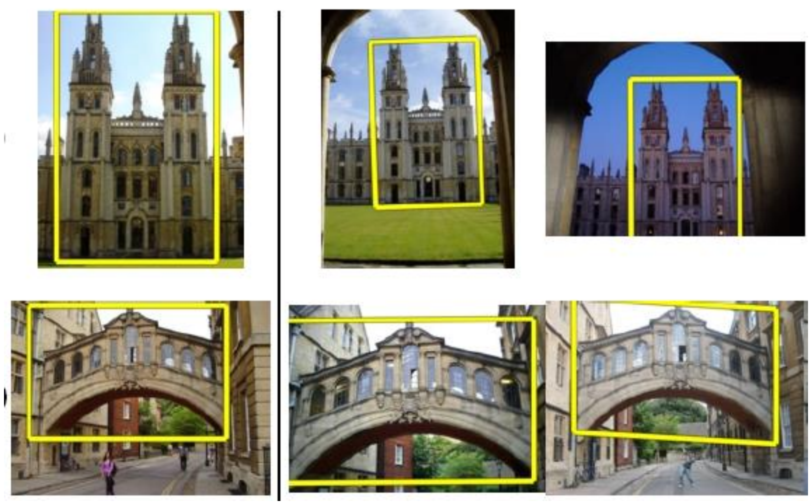
- Object Categorization
  - Problem Definition
  - Challenges
- Sliding-Window based Object Detection
  - Detection via Classification
  - Global Representations
  - Classifier Construction
- Classification with Boosting
  - AdaBoost
  - Viola-Jones Face Detection
- Classification with SVMs
  - Support Vector Machines
  - HOG Detector

Identification vs. Categorization



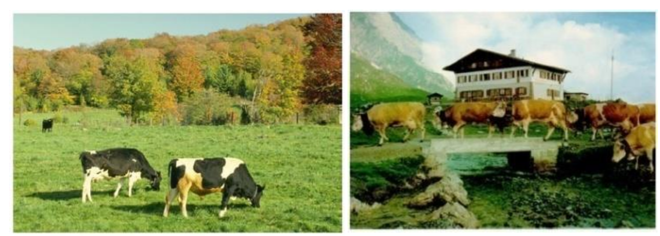
Identification vs. Categorization

- Find *this particular* object



- Recognize ANY car

- Recognize ANY cow



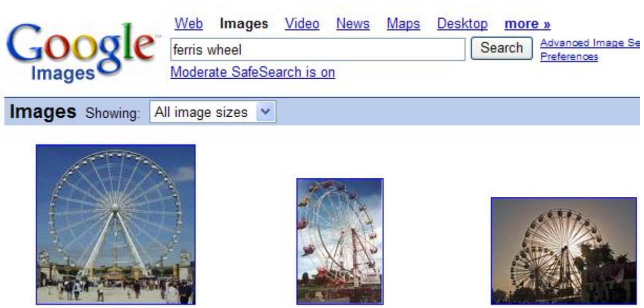
Object Categorization - Potential Applications

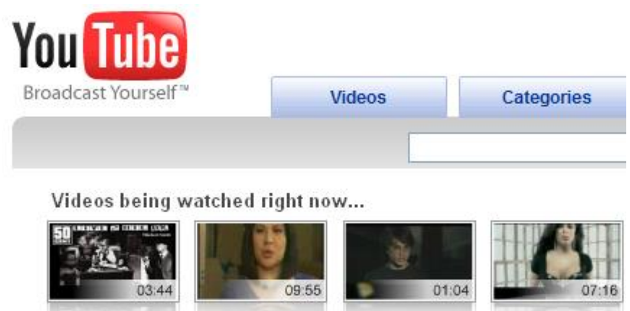
There is a wide range of applications, including.

Autonomous robots

Navigation, driver safety

Consumer electronics



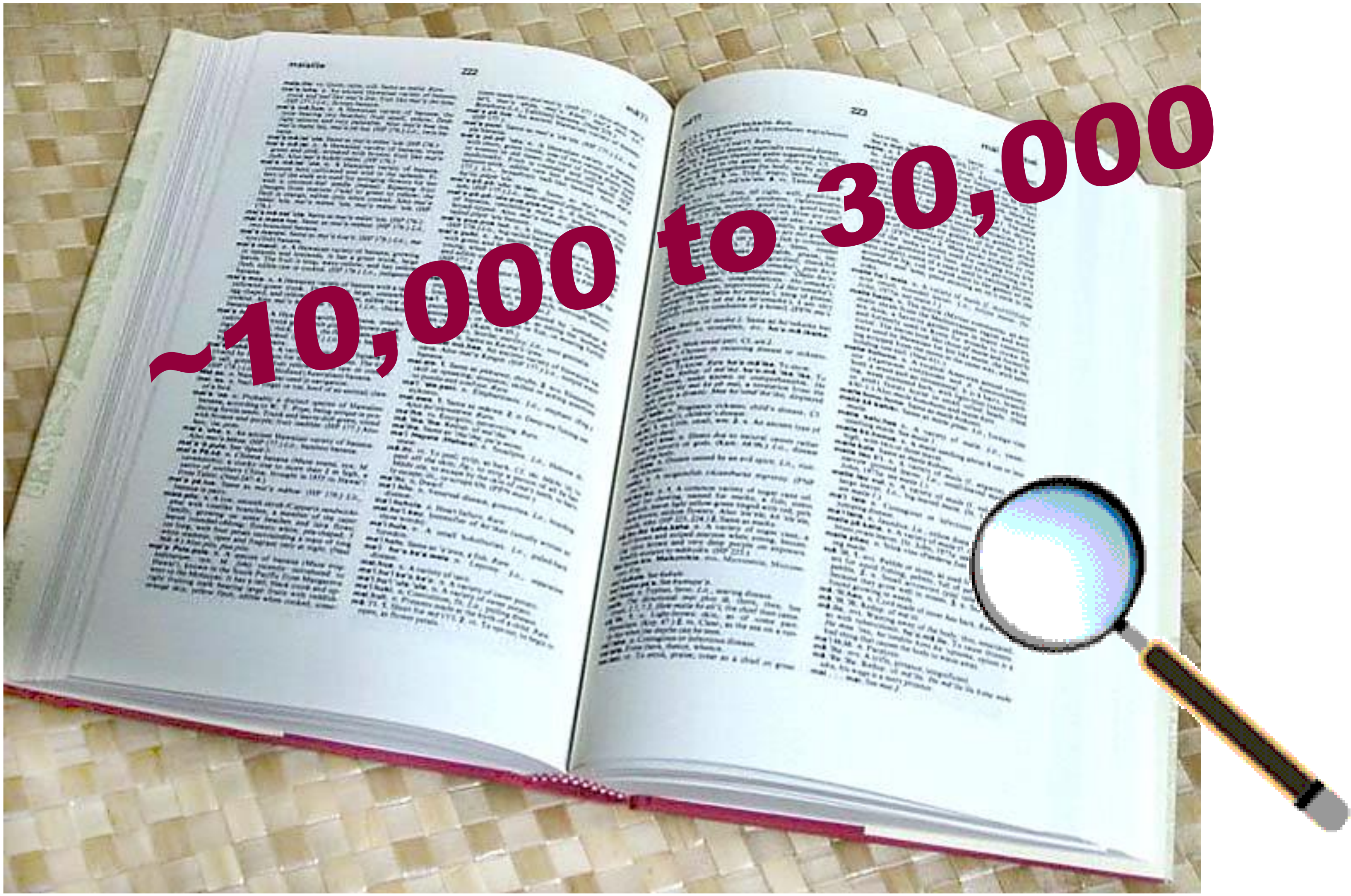


Content-based retrieval and analysis for images and videos

Medical image analysis

Slide adapted from Kristen Grauman

How many object categories are there?



Biederman 1987

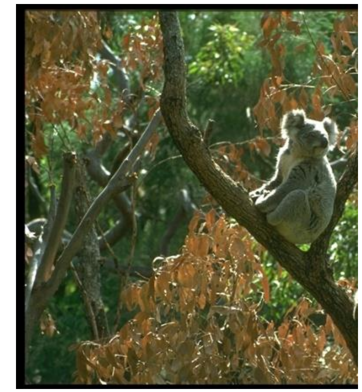
Source: Fei-Fei Li, Rob Fergus, Antonio Torralba.


~10,000 to 30,000

Challenges: Robustness

Illumination

Object pose



Clutter

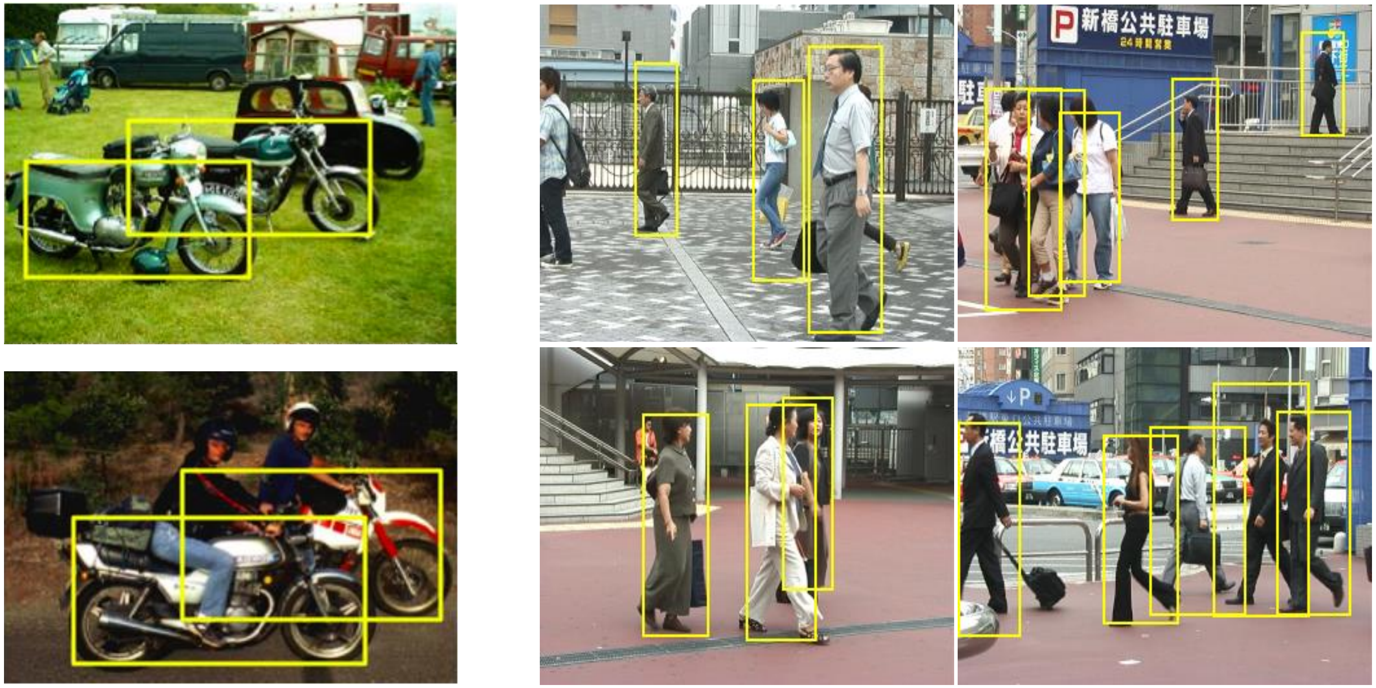
Occlusions

Intra-class appearance

Viewpoint

Slide credit: Kristen Grauman

Challenges: Robustness



- **Detection in crowded, real-world scenes**
  - **Learn object variability**
    - **Changes in appearance, scale, and articulation**
  - **Compensate for clutter, overlap, and occlusion**

[Leibe, Seemann, Schiele, CVPR'05]

Topics of This Lecture

- Object Categorization
  - Problem Definition
  - Challenges
- **Sliding-Window based Object Detection**
  - **Detection via Classification**
  - **Global Representations**
  - **Classifier Construction**
- Classification with Boosting
  - AdaBoost
  - Viola-Jones Face Detection
- Classification with SVMs
  - Support Vector Machines
  - HOG Detector

Detection via Classification: Main Idea

- **Basic component: a binary classifier**

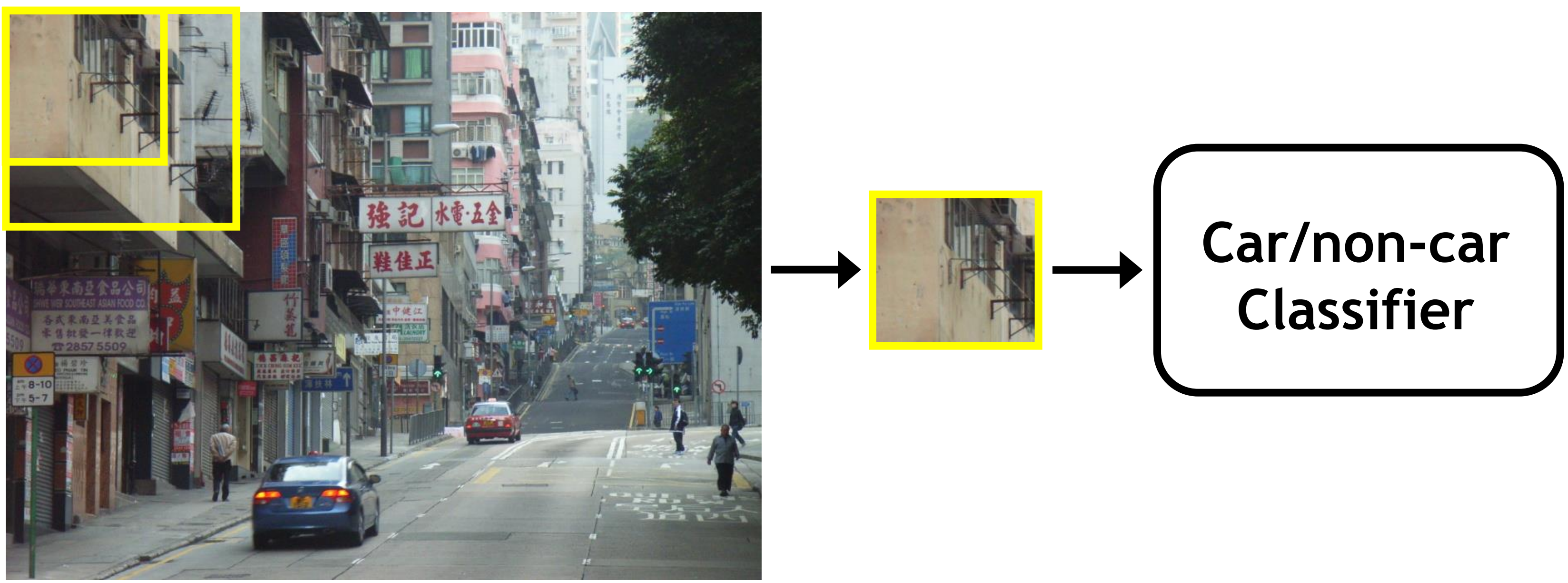
Car/non-car Classifier

No, not a car.

Yes, car.

Detection via Classification: Main Idea

- If the object may be in a cluttered scene, slide a window around looking for it.



- Essentially, this is a brute-force approach with many local decisions.

What *is* a Sliding Window Approach?

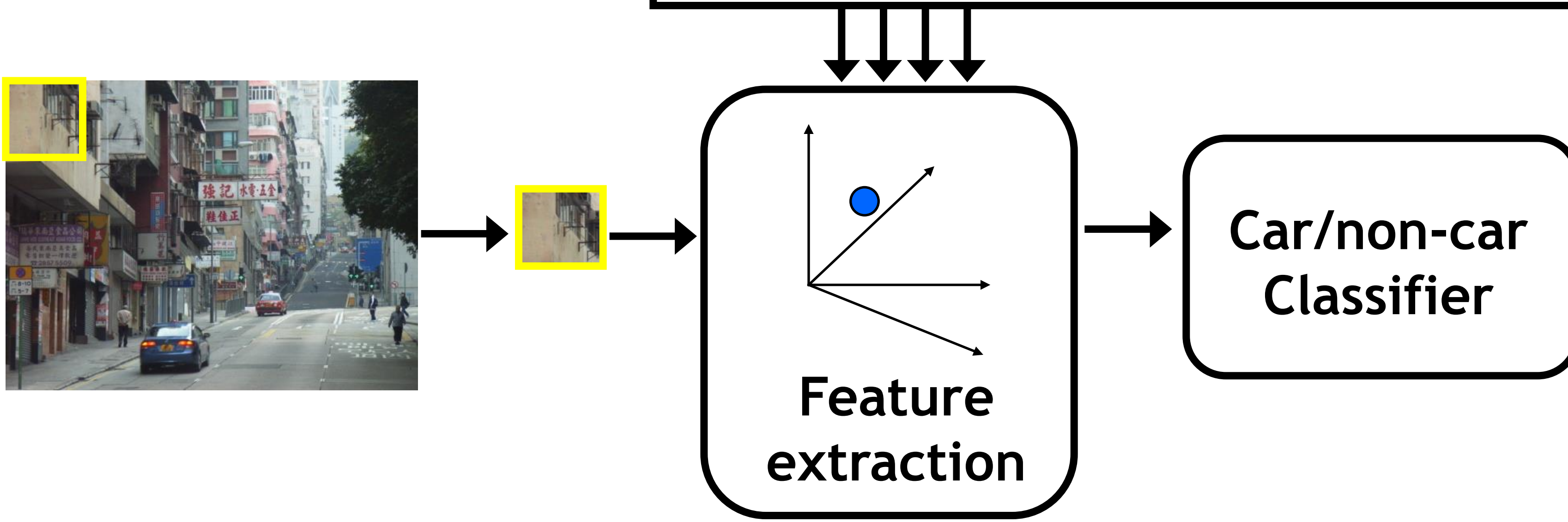
- Search over space and scale

- Detection as subwindow classification problem
- *"In the absence of a more intelligent strategy, any global image classification approach can be converted into a localization approach by using a sliding-window search."*

Detection via Classification: Main Idea

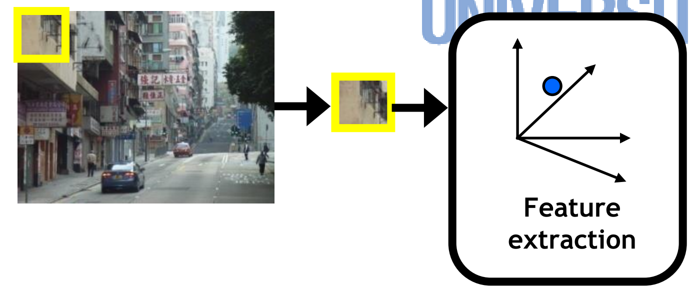
Fleshing out this pipeline a bit more, we need to:

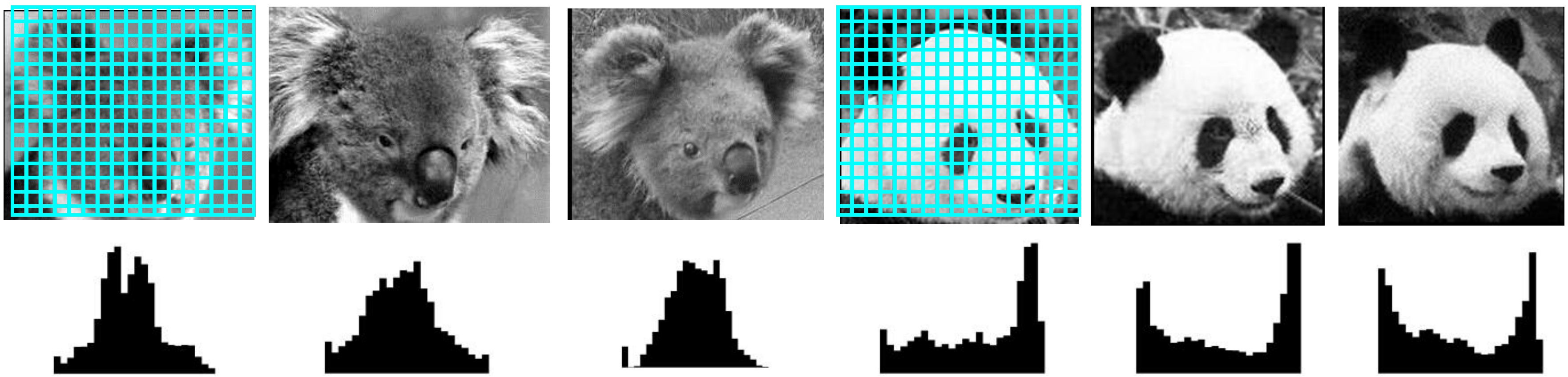
1. Obtain training data
2. Define features
3. Define classifier



Slide credit: Kristen Grauman

Feature extraction: Global Appearance





- Simple holistic descriptions of image content
  - Grayscale / color histogram
  - Vector of pixel intensities

Eigenfaces: Global Appearance Description

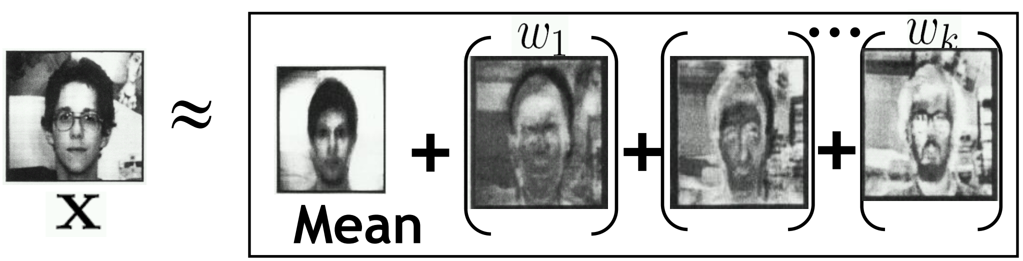
This can also be applied in a sliding-window framework…

Training images

Mean

Eigenvectors computed from covariance matrix

Generate low-dimensional representation of appearance with a linear subspace.



Project new images to “face space”.

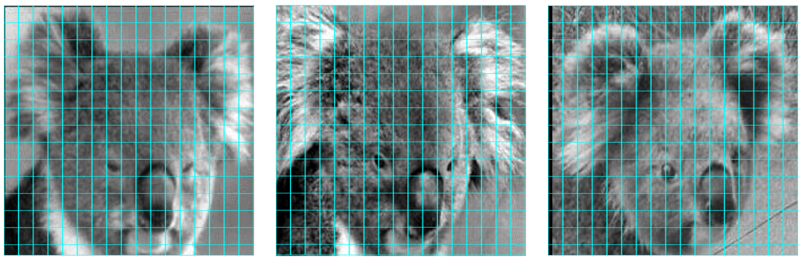
Detection via distance **TO** eigenspace

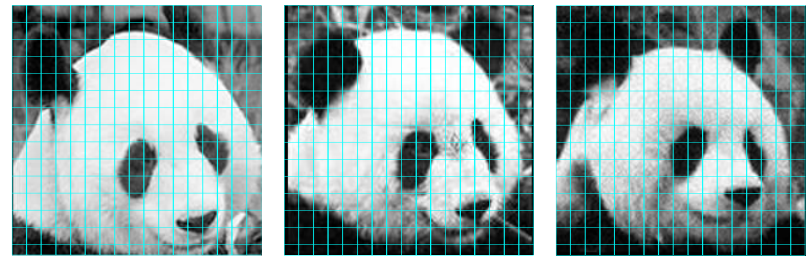
Identification via distance **IN** eigenspace

[Turk & Pentland, 1991]

Feature Extraction: Global Appearance

- Pixel-based representations are sensitive to small shifts



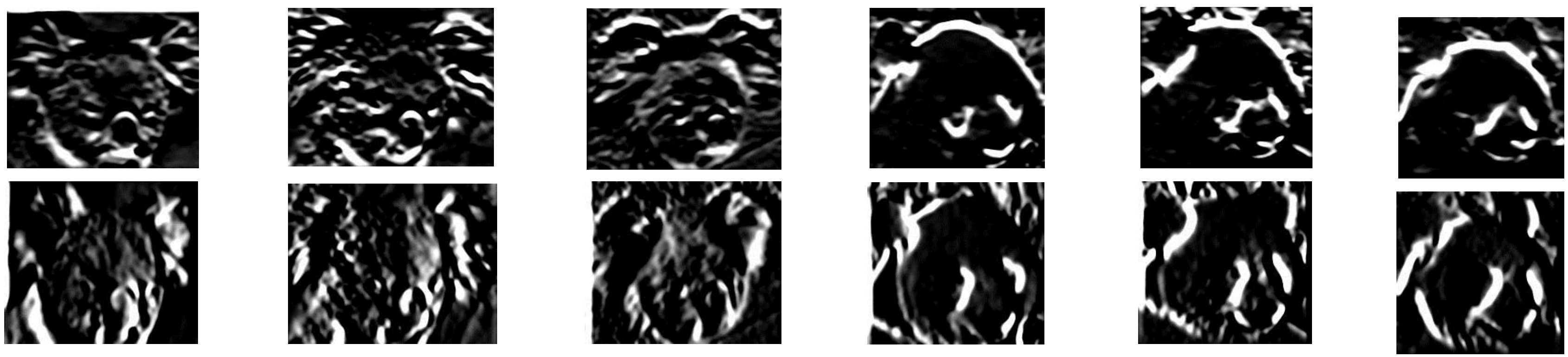


- Color or grayscale-based appearance description can be sensitive to illumination and intra-class appearance variation

Cartoon example:
an albino koala

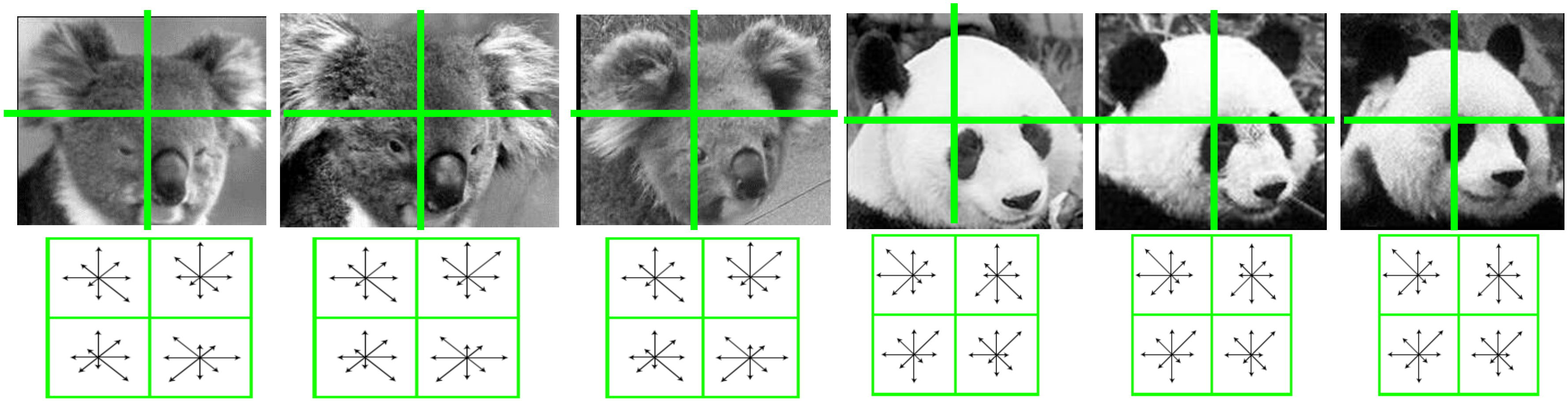
Gradient-based Representations

- **Idea**
  - Consider edges, contours, and (oriented) intensity gradients



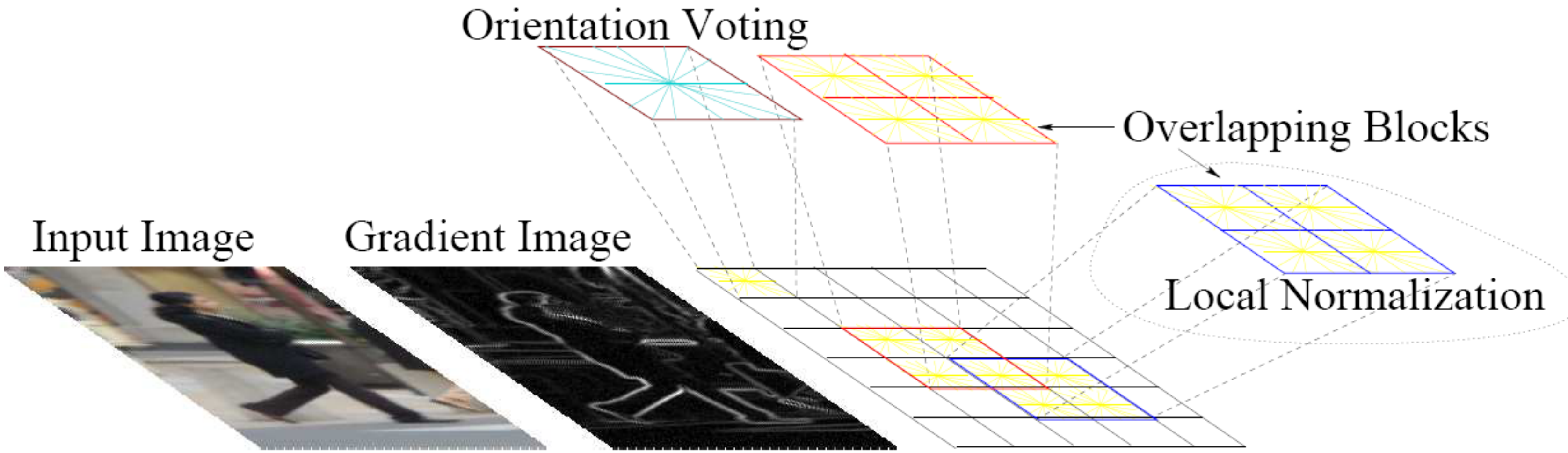
Gradient-based Representations

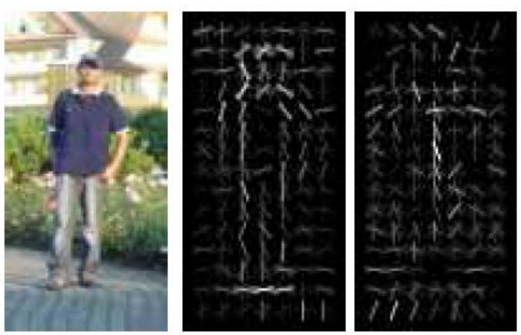
- Idea
  - Consider edges, contours, and (oriented) intensity gradients



- Summarize local distribution of gradients with histogram
  - Locally orderless: offers invariance to small shifts and rotations
  - Localized histograms offer more spatial information than a single global histogram (tradeoff invariant vs. discriminative)
  - Contrast-normalization: try to correct for variable illumination

Gradient-based Representations: Histograms of Oriented Gradients (HoG)





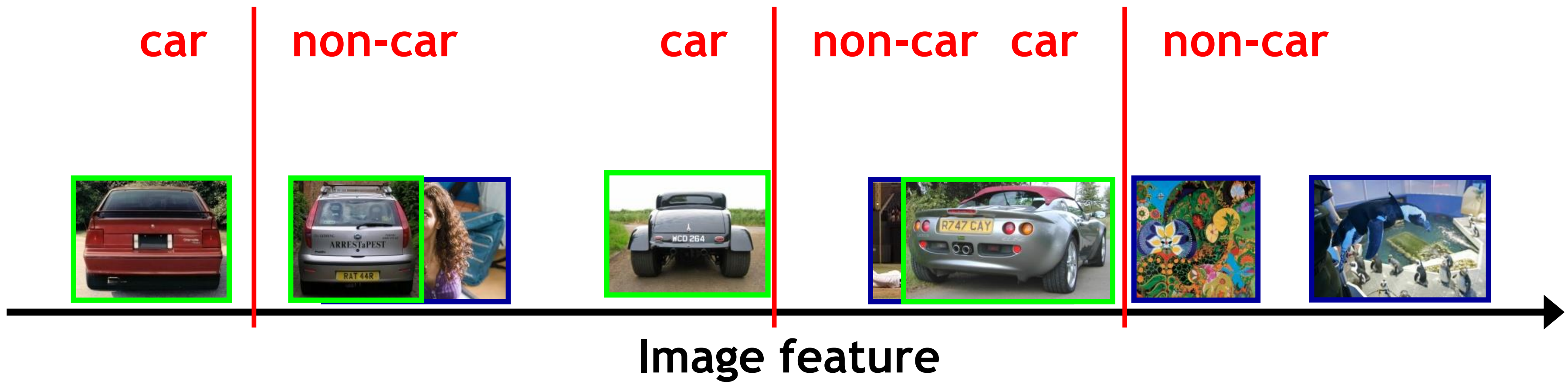
- **Map each grid cell in the input window to a histogram counting the gradients per orientation.**
- **Code available: http://pascal.inrialpes.fr/soft/olt/**

Slide credit: Kristen Grauman

[Dalal & Triggs, CVPR 2005]

Classifier Construction

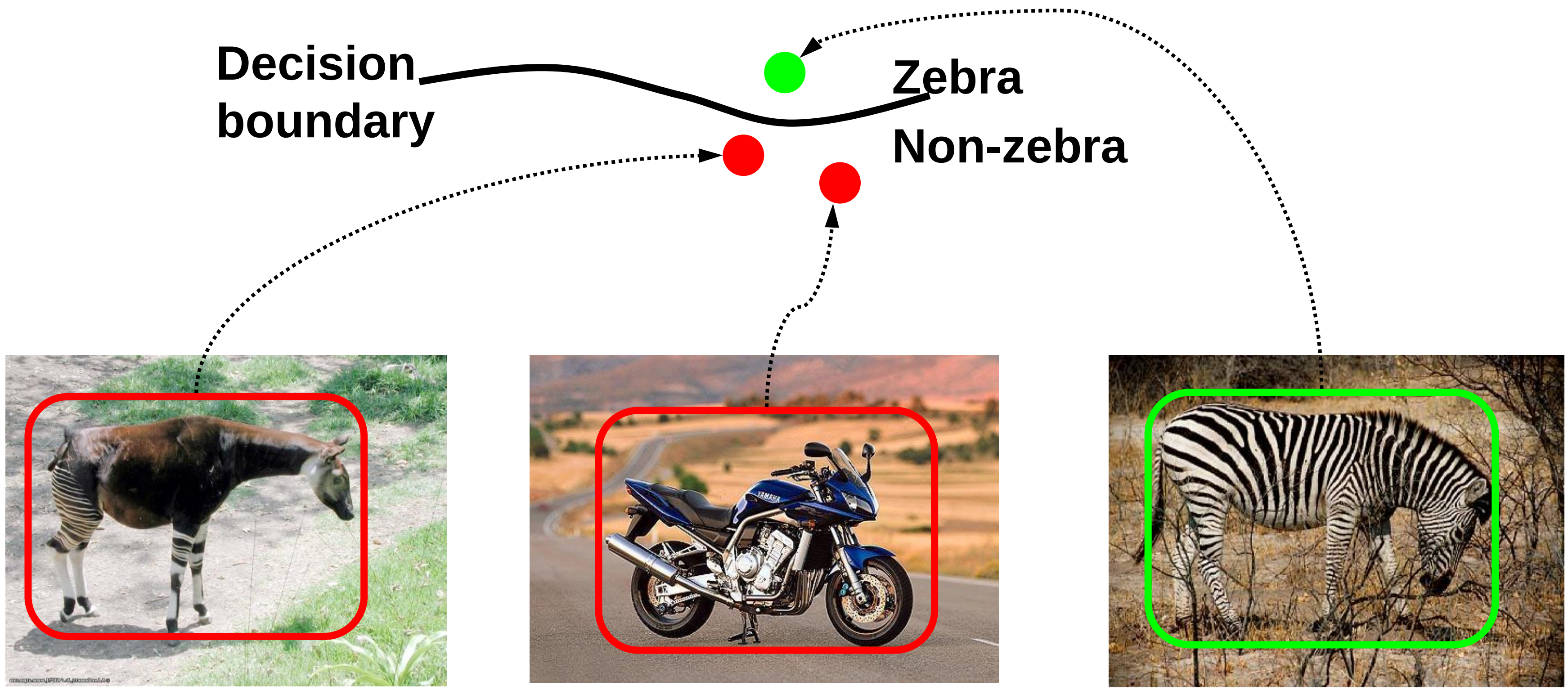
- How to compute a decision for each subwindow?



Slide credit: Kristen Grauman

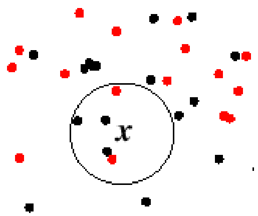
Discriminative Methods

- **Learn a decision rule (classifier) assigning image features to different classes**



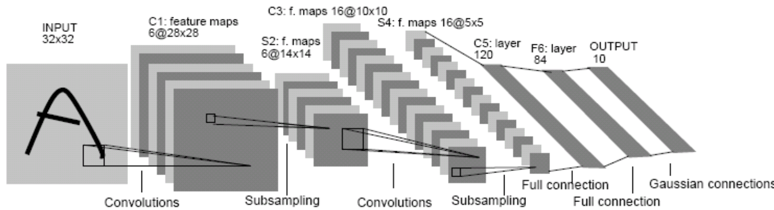
Classifier Construction: Many Choices…

Nearest Neighbor



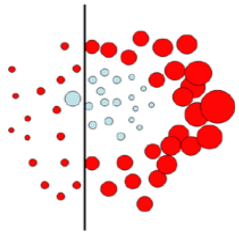
Berg, Berg, Malik 2005,
Chum, Zisserman 2007,
Boiman, Shechtman, Irani 2008, …

Neural networks



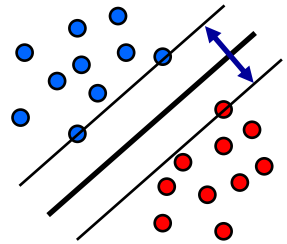
LeCun, Bottou, Bengio, Haffner 1998
Rowley, Baluja, Kanade 1998
…

Boosting



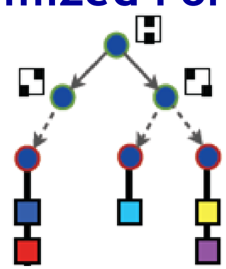
Viola, Jones 2001,
Torralba et al. 2004,
Opelt et al. 2006,
Benenson 2012, …

Support Vector Machines



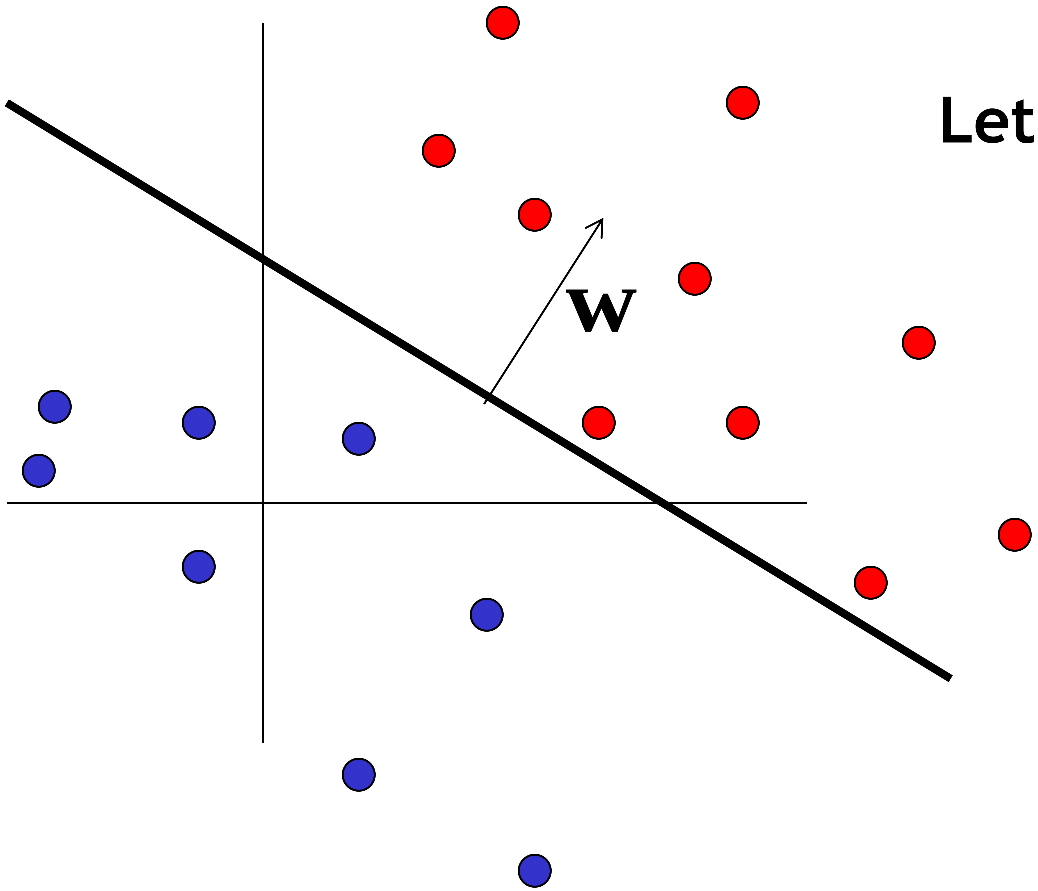
Vapnik, Schölkopf 1995,
Papageorgiou, Poggio '01,
Dalal, Triggs 2005,
Vedaldi, Zisserman 2012

Randomized Forests



Amit, Geman 1997,
Breiman 2001,
Lepetit, Fua 2006,
Gall, Lempitsky 2009,…

Linear Classifiers



Let $\mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

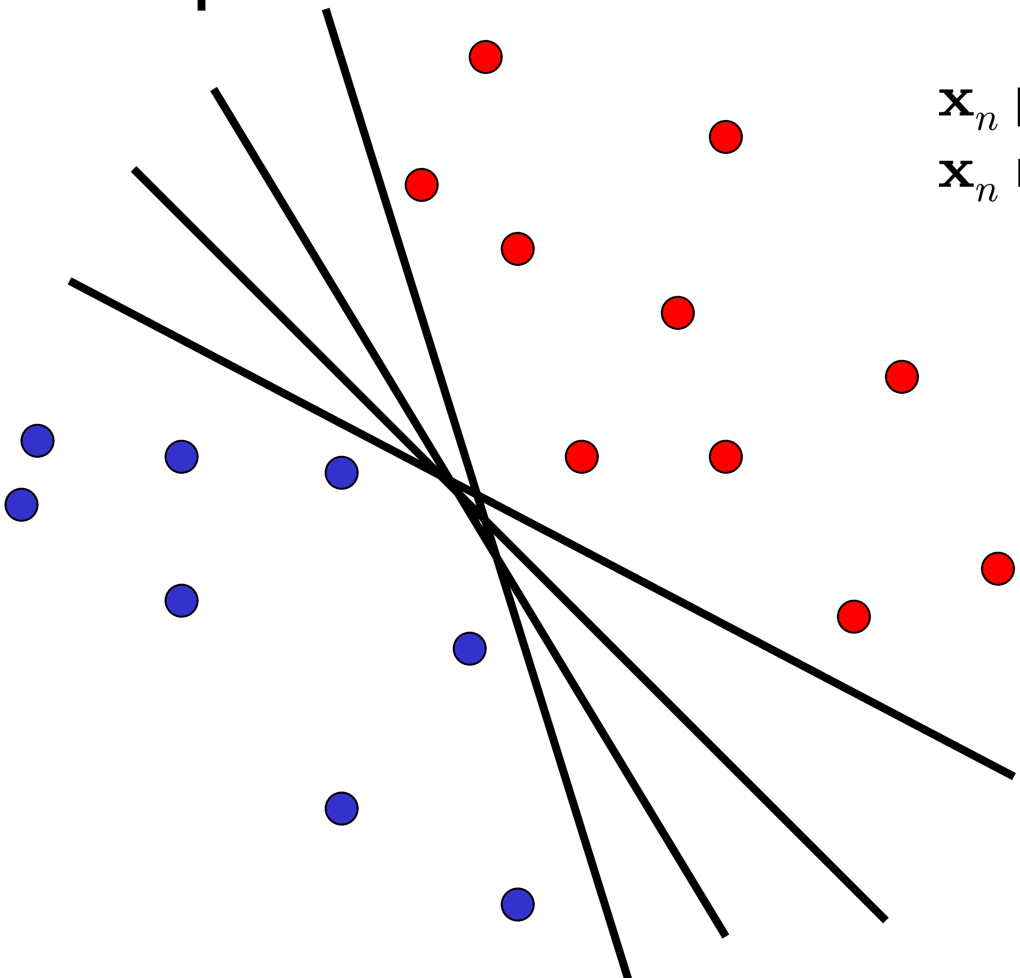
$$w_1 x_1 + w_2 x_2 + b = 0$$

$$\updownarrow$$

$$\mathbf{w}^T \mathbf{x} + b = 0$$

Linear Classifiers

- **Find linear function to separate positive and negative examples**



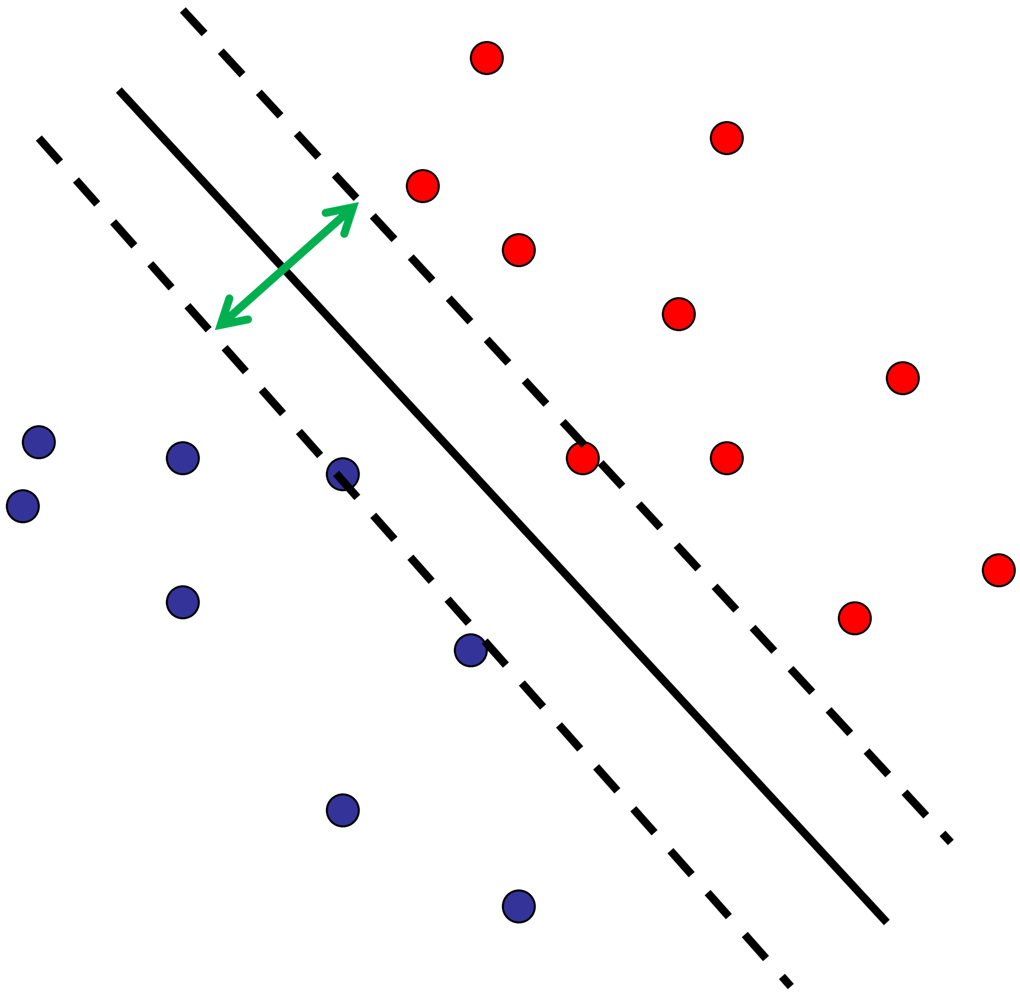
$\mathbf{x}_n$ **positive:** $\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b \geq 0$

$\mathbf{x}_n$ **negative:** $\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b < 0$

Which line is best?

Slide credit: Kristen Grauman

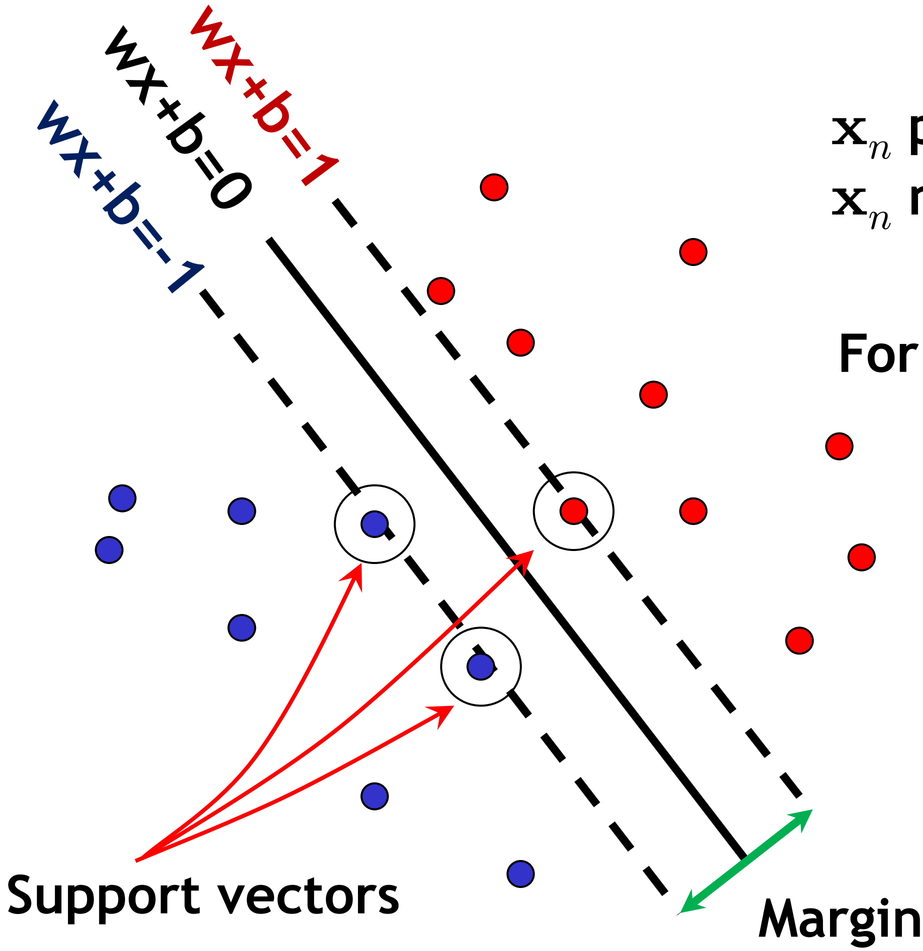
Support Vector Machines (SVMs)



- Discriminative classifier based on *optimal separating hyperplane (i.e. line for 2D case)*
- Maximize the *margin* between the positive and negative training examples

Support Vector Machines

- Want line that maximizes the margin.



$\mathbf{x}_n$ positive $(t_n = 1)$: $\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b \geq 1$
$\mathbf{x}_n$ negative $(t_n = \text{-}1)$: $\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b < \text{-}1$

For support, vectors, $\mathbf{w}^T\mathbf{x}_n + b = \pm 1$

Quadratic optimization problem

Minimize $\frac{1}{2}\mathbf{w}^T\mathbf{w}$

Subject to $t_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1$

Packages available for that…

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

Finding the Maximum Margin Line

- **Solution:**

$$\mathbf{w} = \sum_{n=1}^{N} a_n t_n \mathbf{x}_n$$

Learned weight

Support vector

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

Finding the Maximum Margin Line

- **Solution:** $$\mathbf{w} = \sum_{n=1}^{N} a_n t_n \mathbf{x}_n$$

- **Classification function:**

$$f(\mathbf{x}) = \operatorname{sign}(\mathbf{w}^T\mathbf{x} + b)$$
$$= \operatorname{sign}\left(\sum_{n=1}^{N} a_n t_n \boxed{\mathbf{x}_n^T\mathbf{x}} + b\right)$$

If f(x) < 0, classify as neg., if f(x) > 0, classify as pos.

  - Notice that this relies on an *inner product* between the test point $\mathbf{x}$ and the support vectors $\mathbf{x}_n$
  - (Solving the optimization problem also involves computing the inner products $\mathbf{x}_n{}^T\mathbf{x}_m$ between all pairs of training points)

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

Questions

- What if the features are not 2d?
- What if the data is not linearly separable?
- What if we have more than just two categories?

Questions

- What if the features are not 2d?
  - Generalizes to d-dimensions - replace line with “hyperplane”

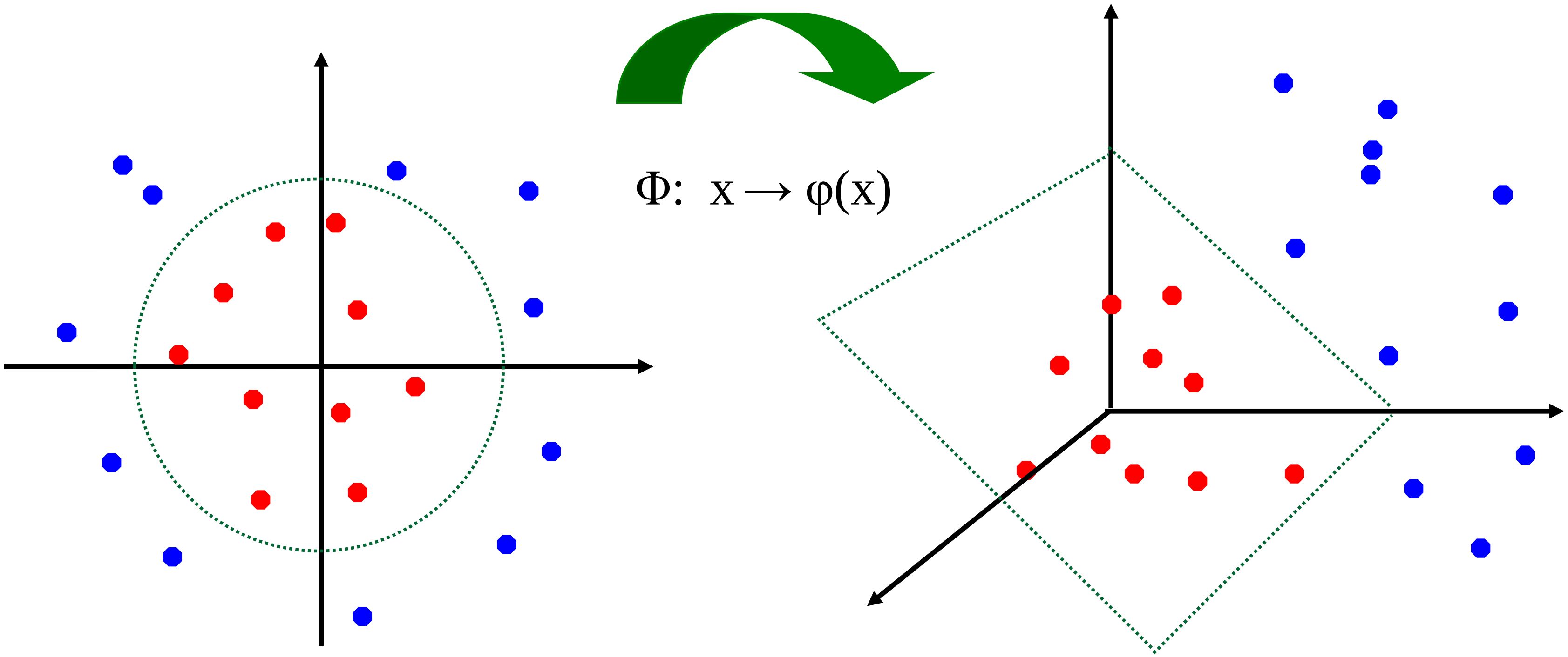
- What if the data is not linearly separable?
- What if we have more than just two categories?

Questions

- **What if the features are not 2d?**
  - Generalizes to d-dimensions - replace line with “hyperplane”
- **What if the data is not linearly separable?**
  - Non-linear SVMs with special kernels
- **What if we have more than just two categories?**

Non-Linear SVMs: Feature Spaces

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



More on that in the Machine Learning lecture…

Slide from Andrew Moore's tutorial: http://www.autonlab.org/tutorials/svm.html

Nonlinear SVMs

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

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

- This gives a nonlinear decision boundary in the original feature space:

$$\sum_n a_n t_n K(\mathbf{x}_n, \mathbf{x}) + b$$

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

Some Often-Used Kernel Functions

- **Linear:**

$$K(x_i, x_j) = x_i^T x_j$$

- **Polynomial of power p:**

$$K(x_i, x_j) = (1 + x_i^T x_j)^p$$

- **Gaussian (Radial-Basis Function):**

$$K(\mathbf{x_i}, \mathbf{x_j}) = \exp\left(-\frac{\left\|\mathbf{x_i} - \mathbf{x_j}\right\|^2}{2\sigma^2}\right)$$

Slide from Andrew Moore's tutorial: http://www.autonlab.org/tutorials/svm.html

Questions

- What if the features are not 2d?
  - Generalizes to d-dimensions - replace line with “hyperplane”
- What if the data is not linearly separable?
  - Non-linear SVMs with special kernels
- What if we have more than just two categories?

Multi-Class SVMs

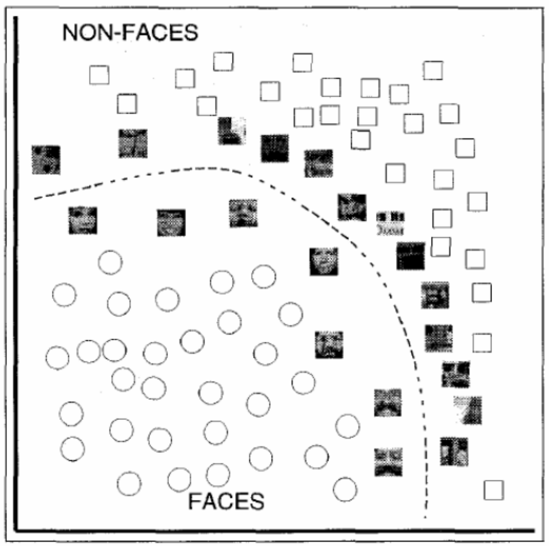
- Achieve multi-class classifier by combining a number of binary classifiers

- One vs. all
  - Training: learn an SVM for each class vs. the rest
  - Testing: apply each SVM to test example and assign to it the class of the SVM that returns the highest decision value
- One vs. one
  - Training: learn an SVM for each pair of classes
  - Testing: each learned SVM "votes" for a class to assign to the test example

Slide credit: Kristen Grauman

SVMs for Recognition

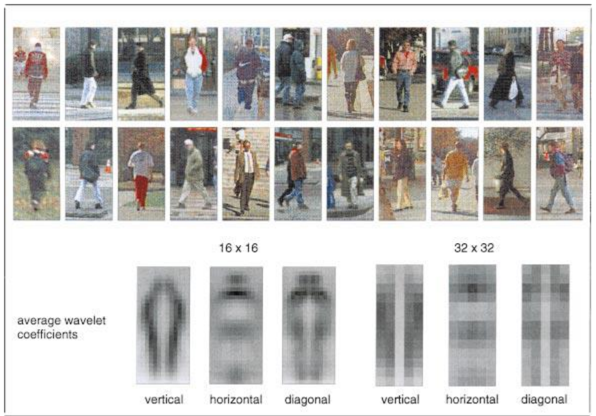
1. Define your representation for each example.
2. Select a kernel function.
3. Compute pairwise kernel values between labeled examples
4. Given this “kernel matrix” to SVM optimization software to identify support vectors & weights.
5. To classify a new example: compute kernel values between new input and support vectors, apply weights, check sign of output.



Slide credit: Kristen Grauman

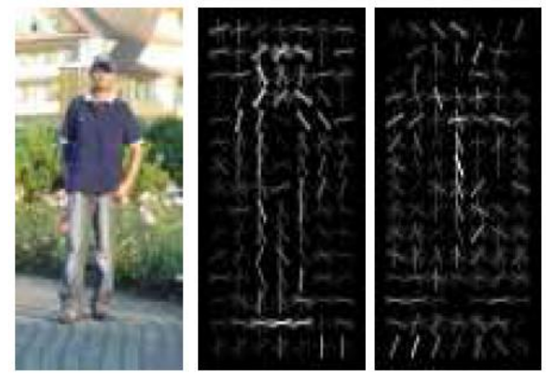
Pedestrian Detection

- **Detecting upright, walking humans using sliding window's appearance/texture; e.g.,**



SVM with Haar wavelets [Papageorgiou & Poggio, IJCV 2000]

Space-time rectangle features [Viola, Jones & Snow, ICCV 2003]



SVM with HoGs [Dalal & Triggs, CVPR 2005]

HOG Descriptor Processing Chain

Image Window

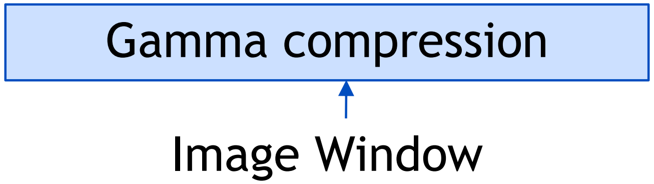
Slide adapted from Navneet Dalal

HOG Descriptor Processing Chain

- Optional: Gamma compression
  - Goal: Reduce effect of overly strong gradients
  - Replace each pixel color/intensity by its square-root

$$x \mapsto \sqrt{x}$$

$\Rightarrow$ Small performance improvement



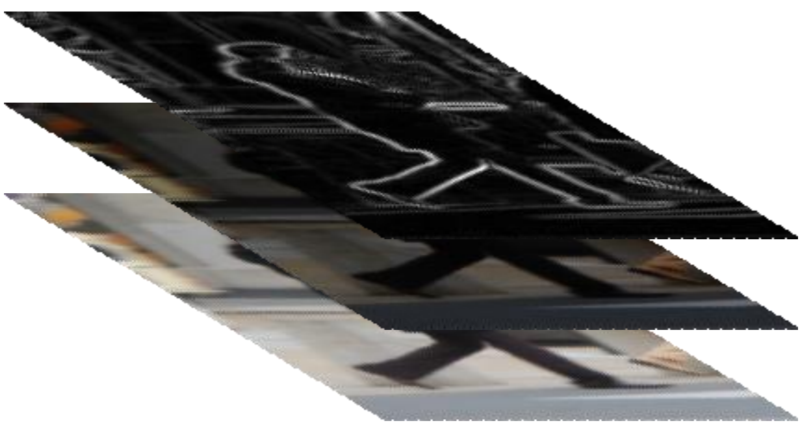
Slide adapted from Navneet Dalal

RWTH AACHEN UNIVERSITY

HOG Descriptor Processing Chain

- Gradient computation
  - Compute gradients on all color channels and take strongest one
  - Simple finite difference filters work best (no Gaussian smoothing)

$$\begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$



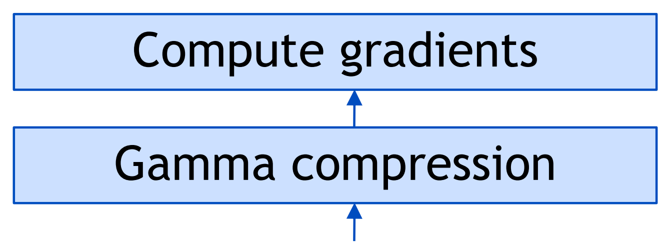
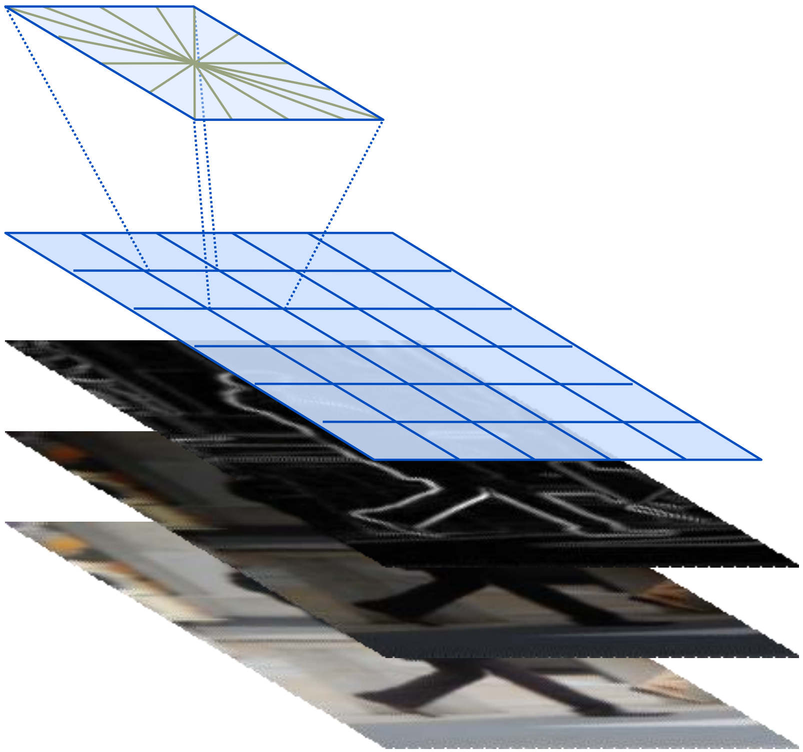


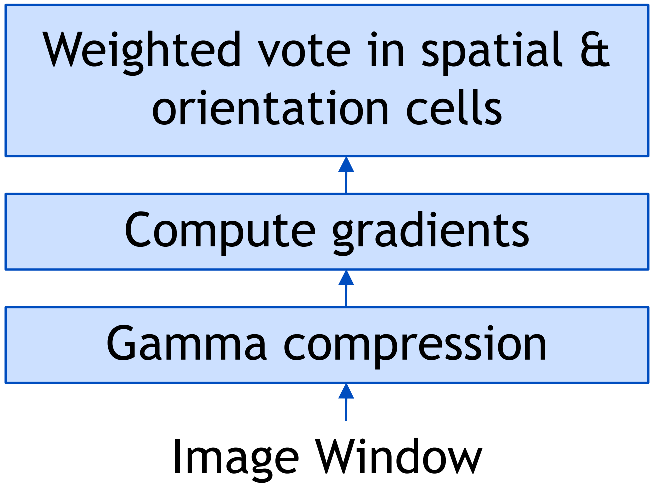
Image Window

Slide adapted from Navneet Dalal

HOG Descriptor Processing Chain

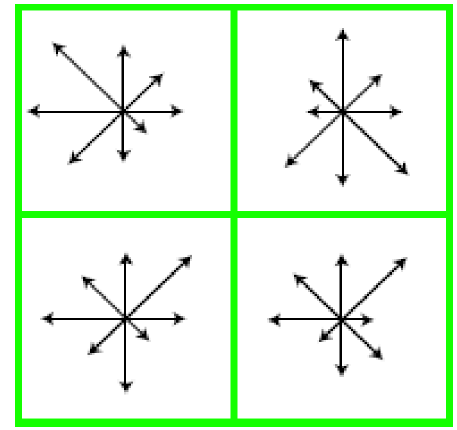
- Spatial/Orientation binning
  - Compute localized histograms of oriented gradients
  - Typical subdivision: 8×8 cells with 8 or 9 orientation bins





Slide adapted from Navneet Dalal

HOG Cell Computation Details

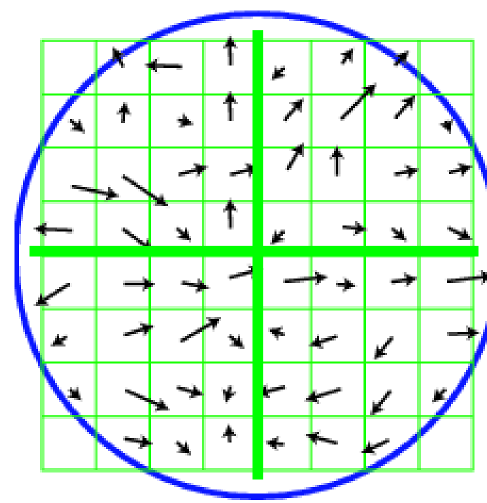


- Gradient orientation voting
  - Each pixel contributes to localized gradient orientation histogram(s)
  - Vote is weighted by the pixel's gradient magnitude

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

- Block-level Gaussian weighting
  - An additional Gaussian weight is applied to each 2×2 block of cells
  - Each cell is part of 4 such blocks, resulting in 4 versions of the histogram.



HOG Cell Computation Details (2)

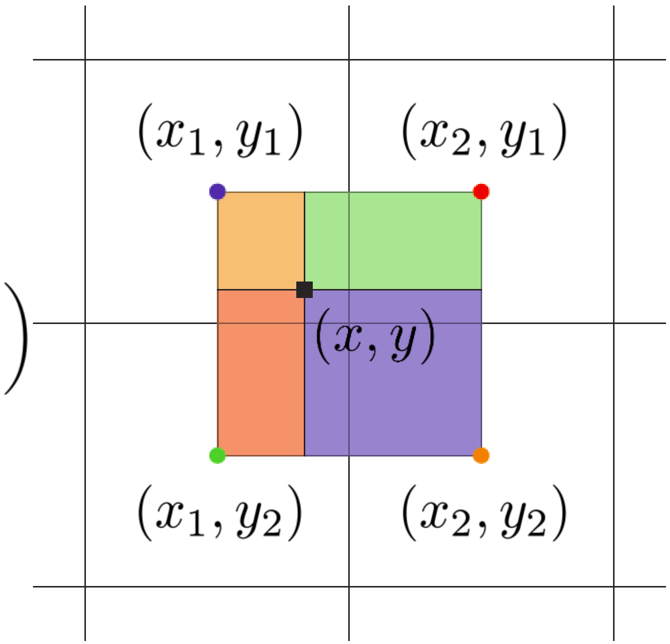
- **Important for robustness: Tri-linear interpolation**
  - Each pixel contributes to (up to) 4 neighboring cell histograms
  - Weights are obtained by bilinear interpolation in image space:

$$h(x_1, y_1) \leftarrow w \cdot \left(1 - \frac{x - x_1}{x_2 - x_1}\right)\left(1 - \frac{y - y_1}{y_2 - y_1}\right)$$

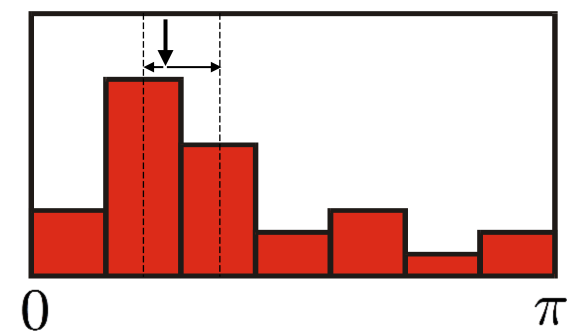
$$h(x_1, y_2) \leftarrow w \cdot \left(1 - \frac{x - x_1}{x_2 - x_1}\right)\left(\frac{y - y_1}{y_2 - y_1}\right)$$

$$h(x_2, y_1) \leftarrow w \cdot \left(\frac{x - x_1}{x_2 - x_1}\right)\left(1 - \frac{y - y_1}{y_2 - y_1}\right)$$

$$h(x_2, y_2) \leftarrow w \cdot \left(\frac{x - x_1}{x_2 - x_1}\right)\left(\frac{y - y_1}{y_2 - y_1}\right)$$

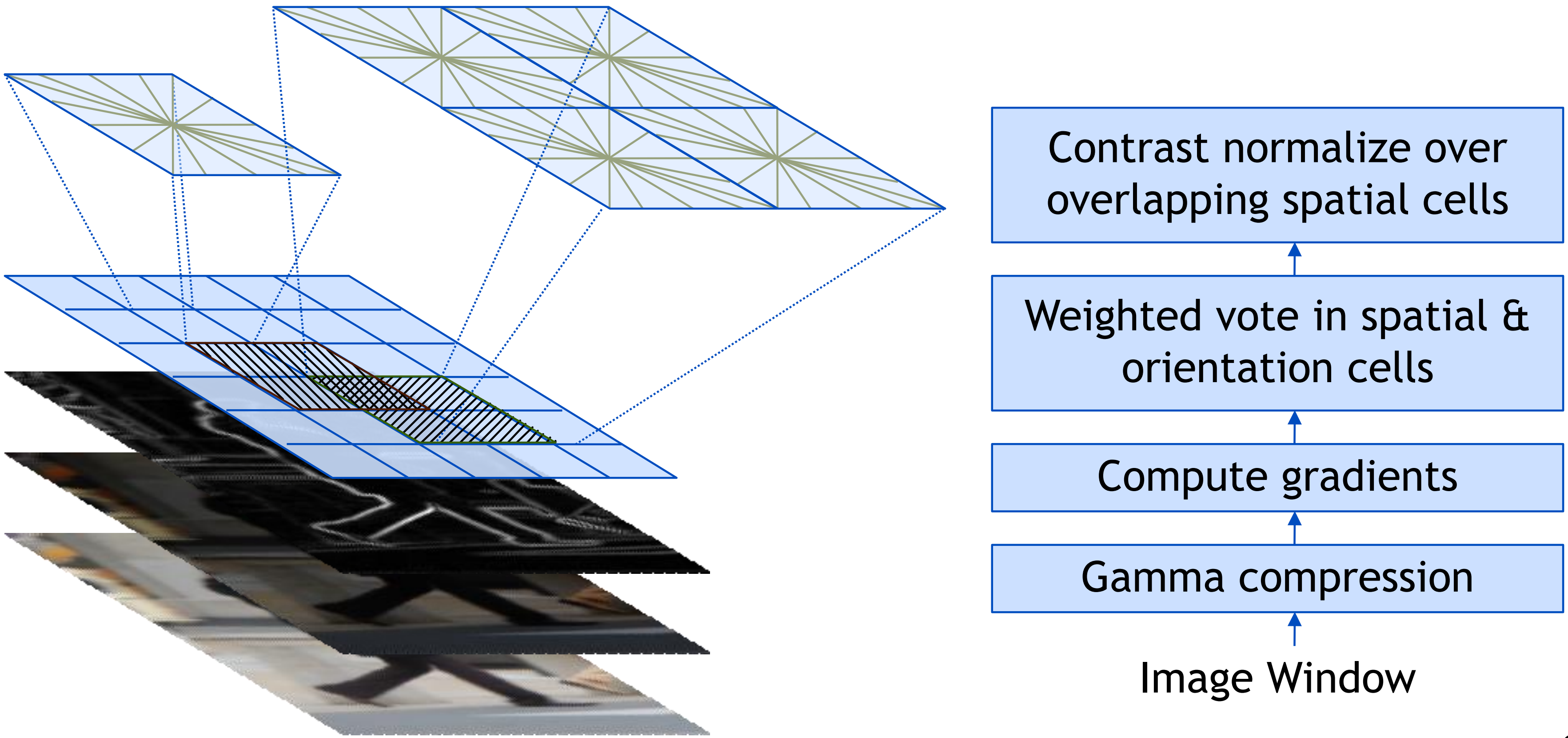


  - Contribution is further split over (up to) 2 neighboring orientation bins via linear interpolation over angles.



HOG Descriptor Processing Chain

- **2-Stage contrast normalization**
  - L2 normalization, clipping, L2 normalization

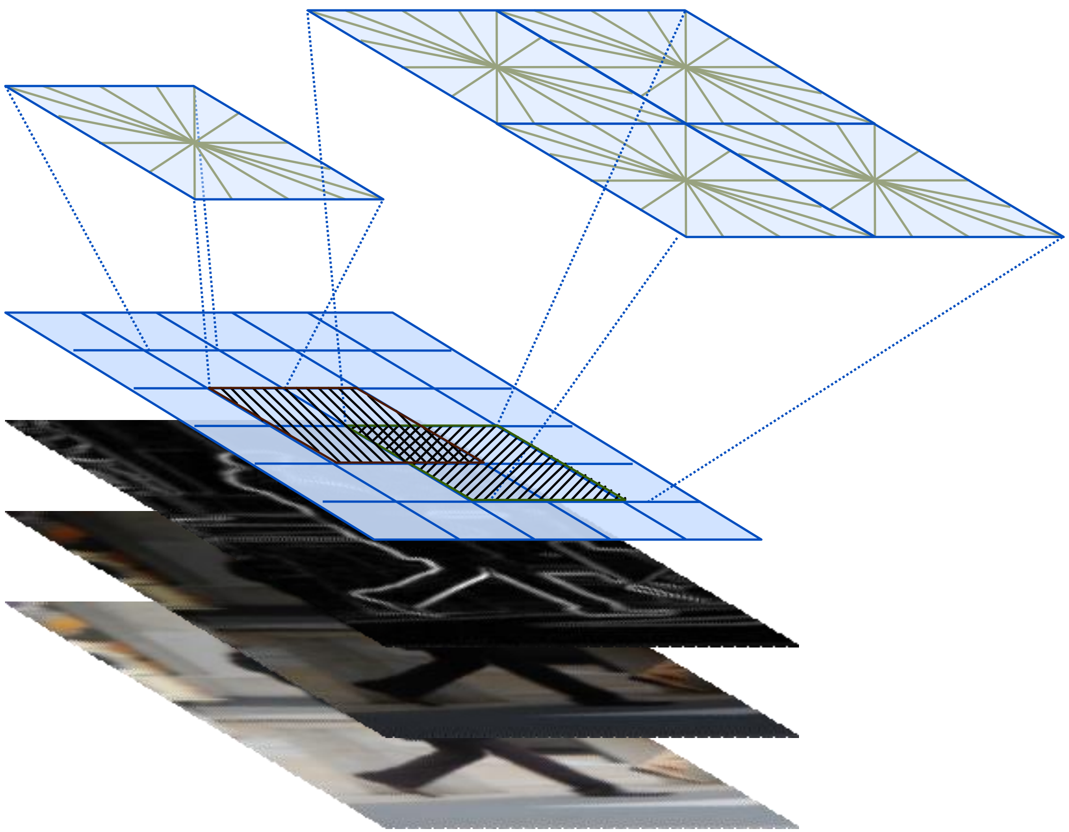


Slide adapted from Navneet Dalal

HOG Descriptor Processing Chain

- **Feature vector construction**
  - Collect HOG blocks into vector

[..., ..., ..., ...]



Collect HOGs over detection window

↑

Contrast normalize over overlapping spatial cells

↑

Weighted vote in spatial & orientation cells

↑

Compute gradients

↑

Gamma compression

↑

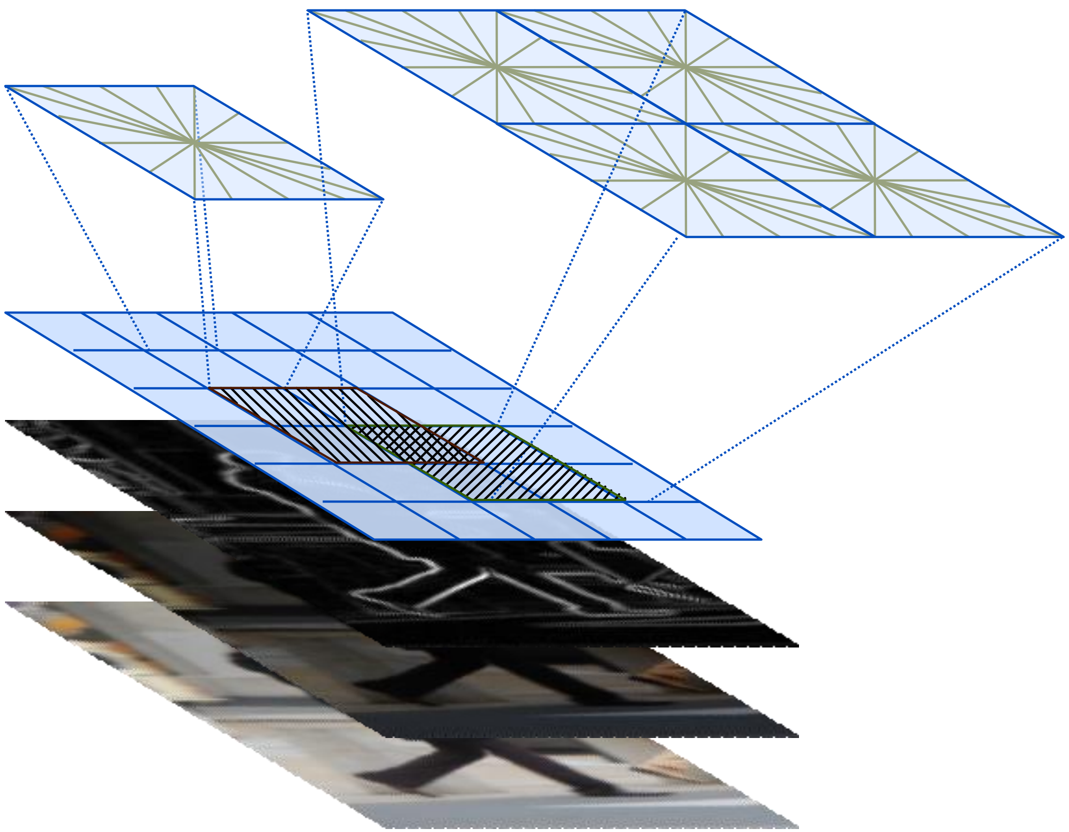
Image Window

Slide adapted from Navneet Dalal

HOG Descriptor Processing Chain

- **SVM Classification**
  - Typically using a linear SVM

[…, …, …, …]



Object/Non-object

↑

Linear SVM

↑

Collect HOGs over detection window

↑

Contrast normalize over overlapping spatial cells

↑

Weighted vote in spatial & orientation cells

↑

Compute gradients

↑

Gamma compression

↑

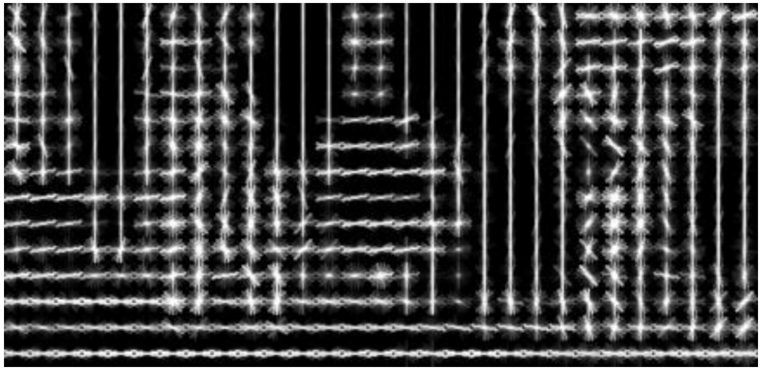
Image Window

Slide adapted from Navneet Dalal

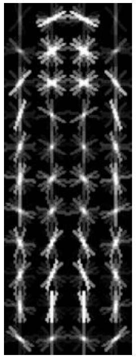
Pedestrian Detection with HOG

- Train a pedestrian template using a linear SVM
- At test time, convolve feature map with template

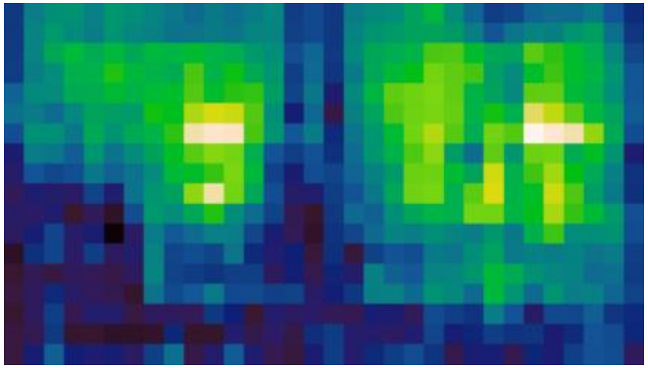
HOG feature map



Template



Detector response map



N. Dalal and B. Triggs, Histograms of Oriented Gradients for Human Detection, CVPR 2005

Slide credit: Svetlana Lazebnik

Non-Maximum Suppression

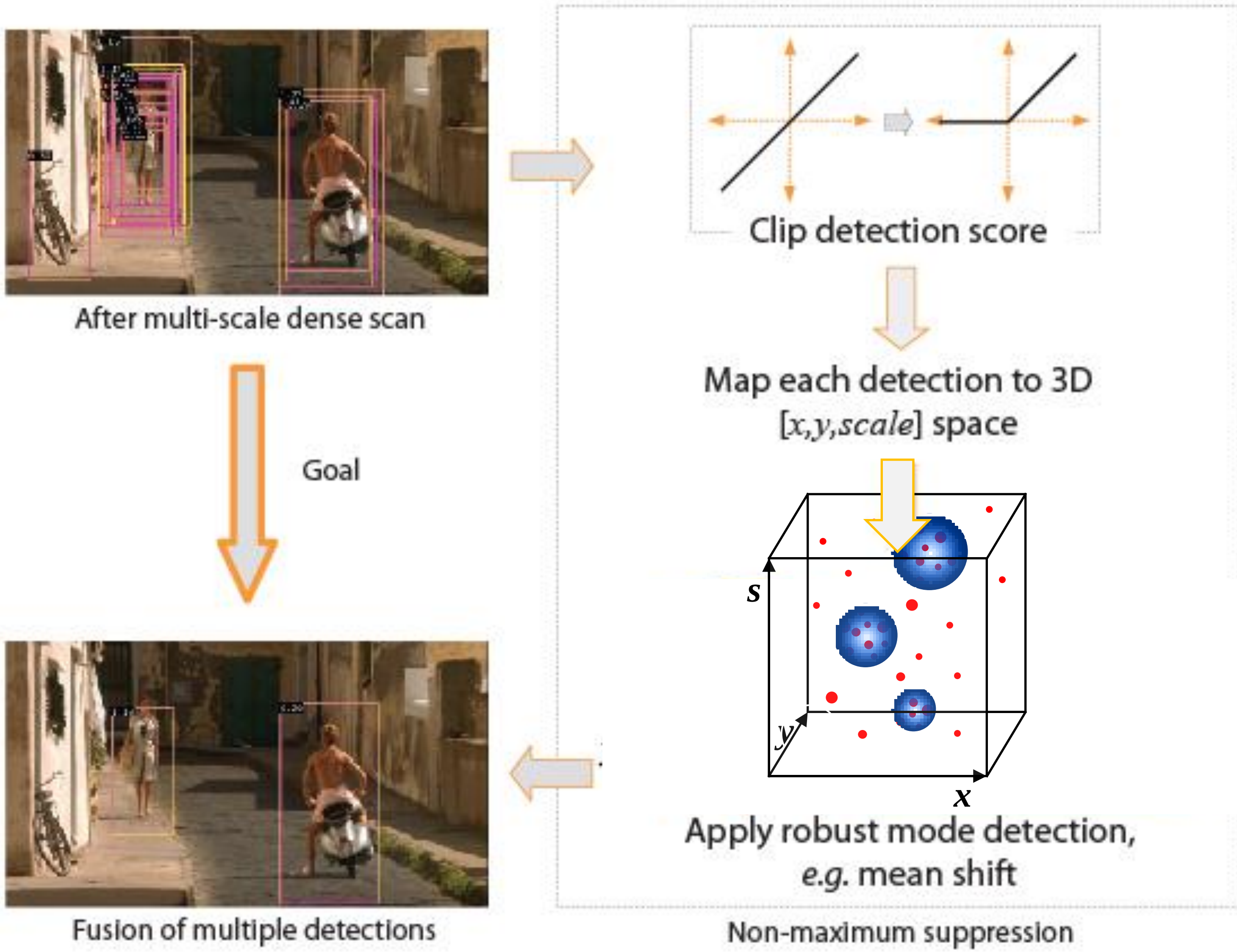


Image source: Navneet Dalal, PhD Thesis

Pedestrian detection with HoGs & SVMs

- Navneet Dalal, Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR 2005

References and Further Reading

- **Read the HOG paper**
  - N. Dalal, B. Triggs,
    Histograms of Oriented Gradients for Human Detection,
    CVPR, 2005.

- **HOG Detector**
  - Code available: http://pascal.inrialpes.fr/soft/olt/