MIT CSAIL

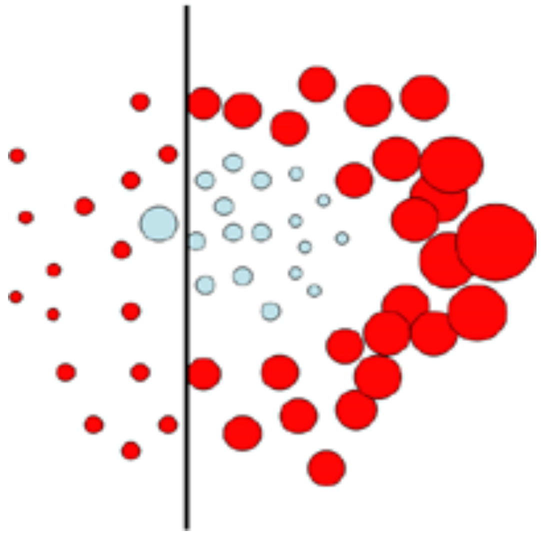
6.869: Advances in Computer Vision

MIT COMPUTER VISION

Lecture 16

Object recognition II

A simple object detector



• Simple but contains some of same basic elements of many state of the art detectors.

• Based on boosting which makes all the stages of the training and testing easy to understand.

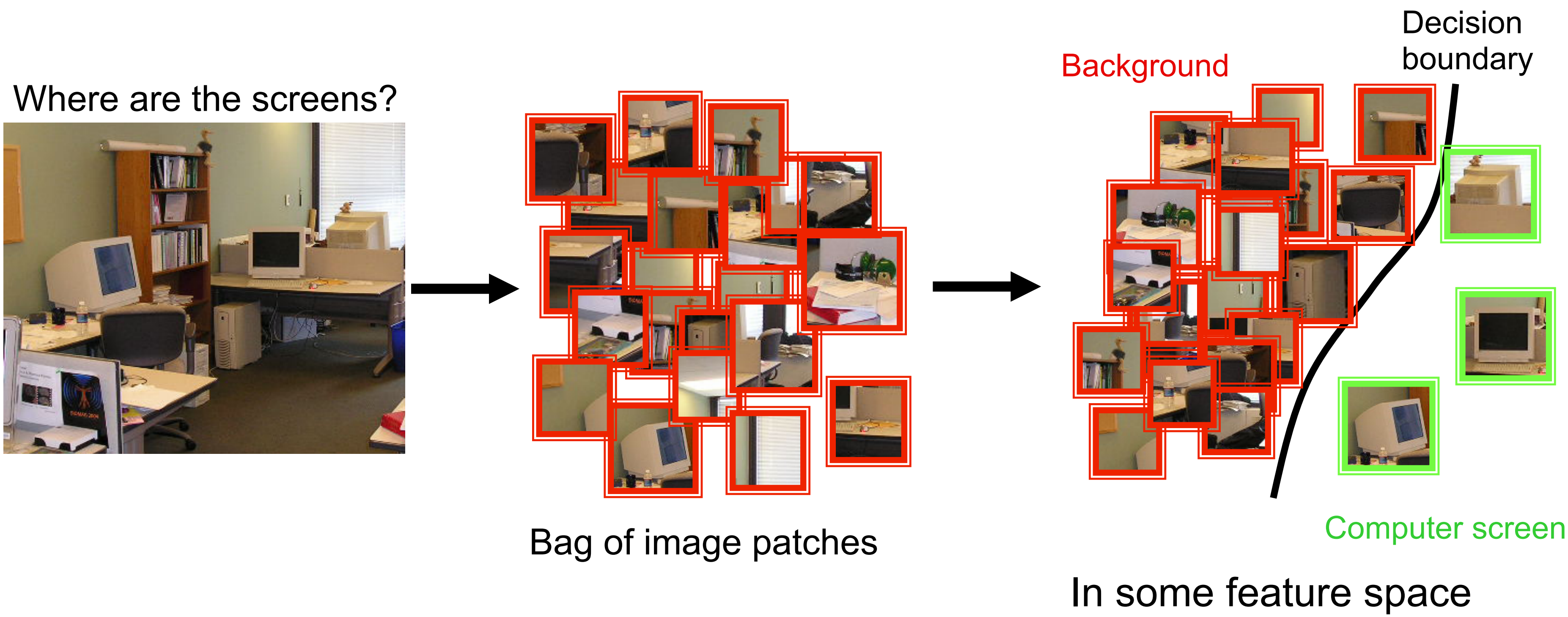
Most of the slides are from the ICCV 05 short course
http://people.csail.mit.edu/torralba/shortCourseRLOC/

Discriminative methods

Object detection and recognition is formulated as a classification problem.

The image is partitioned into a set of overlapping windows

… and a decision is taken at each window about if it contains a target object or not.



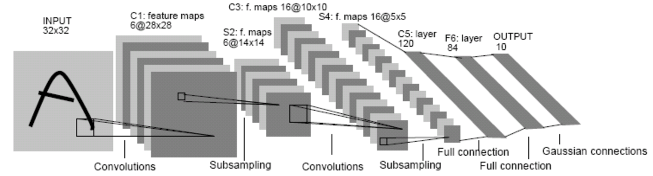
Discriminative methods

Nearest neighbor

10^6 examples

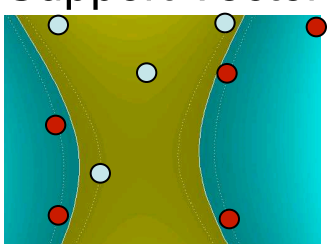
Shakhnarovich, Viola, Darrell 2003
Berg, Berg, Malik 2005
…

Neural networks



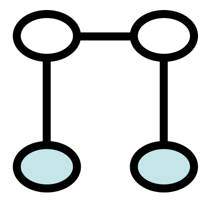
LeCun, Bottou, Bengio, Haffner 1998
Rowley, Baluja, Kanade 1998
…

Support Vector Machines and Kernels



Guyon, Vapnik
Heisele, Serre, Poggio, 2001
…

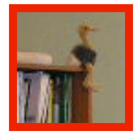
Conditional Random Fields



McCallum, Freitag, Pereira 2000
Kumar, Hebert 2003
…

Formulation

- Formulation: binary classification

   …      

Features	x =	x_1	x_2	x_3	…	x_N		x_{N+1}	x_{N+2}	…	x_{N+M}
Labels	y =	-1	+1	-1		-1		?	?		?

Training data: each image patch is labeled as containing the object or background

Test data

- Classification function

$$\hat{y} = F(x)$$ Where $F(x)$ belongs to some family of functions

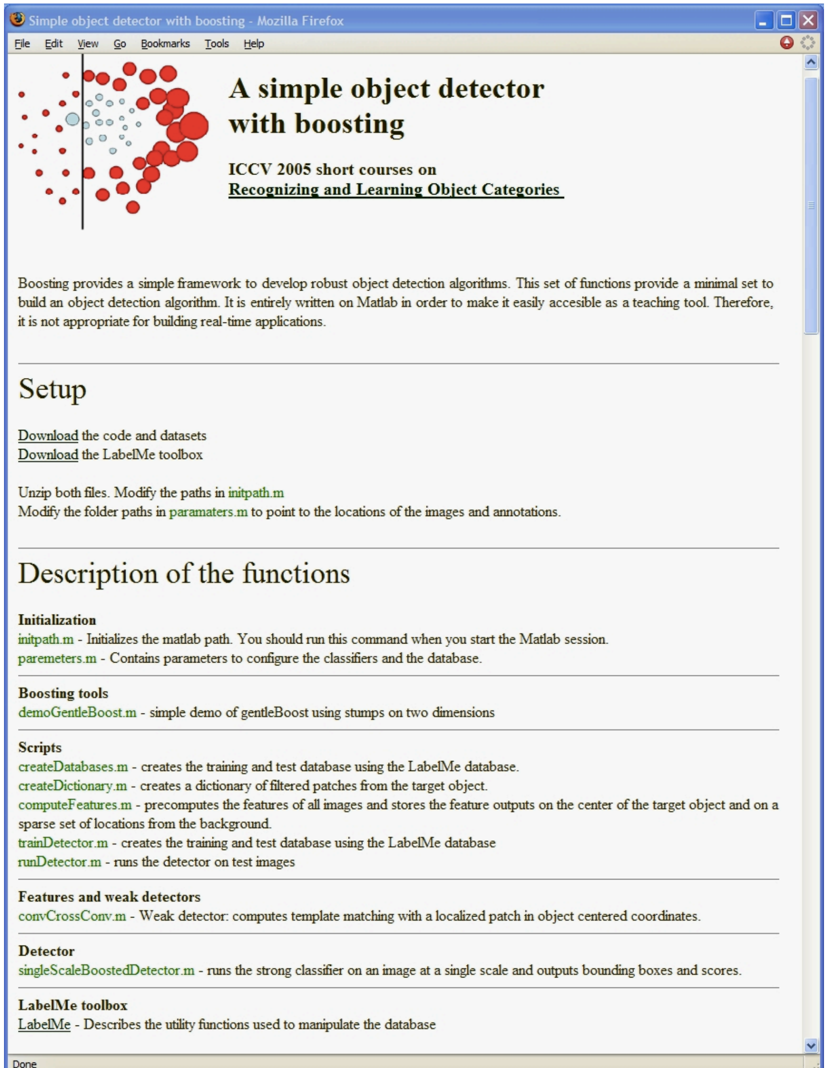
- Minimize misclassification error

(Not that simple: we need some guarantees that there will be generalization)

Overview of section

- Object detection with classifiers
- **Boosting**
  - Gentle boosting
  - Weak detectors
  - Object model
  - Object detection

A simple object detector with Boosting



http://people.csail.mit.edu/torralba/iccv2005/

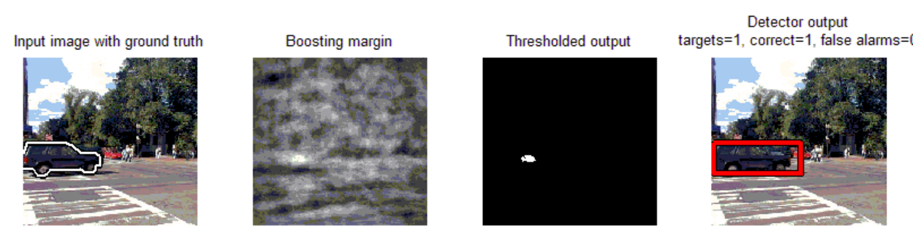
Download

- Toolbox for manipulating dataset
- Code and dataset

Matlab code

- Gentle boosting
- Object detector using a part based model

Dataset with cars and computer monitors



Why boosting?

- A simple algorithm for learning robust classifiers
  - Freund & Shapire, 1995
  - Friedman, Hastie, Tibshhirani, 1998
- Provides efficient algorithm for sparse visual feature selection
  - *Tieu & Viola, 2000*
  - *Viola & Jones, 2003*
- Easy to implement, not requires external optimization tools.

For a description of several methods:
Friedman, J. H., Hastie, T. and Tibshirani, R.
Additive Logistic Regression: a Statistical View of Boosting. 1998

Boosting

- Defines a classifier using an additive model:

$$F(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x) + \alpha_3 f_3(x) + \ldots$$

Strong classifier

Weak classifier

Weight

Features vector

Boosting

- Defines a classifier using an additive model:

$$F(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x) + \alpha_3 f_3(x) + \ldots$$

Strong classifier

Features vector

Weight

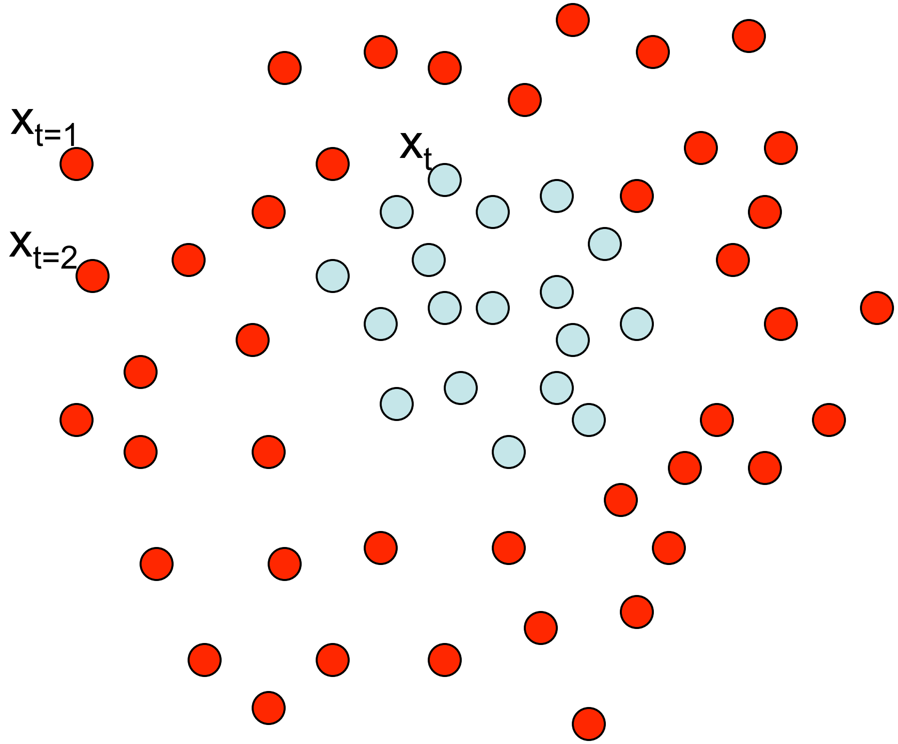
Weak classifier

- We need to define a family of weak classifiers

$f_k(x)$ from a family of weak classifiers

Boosting

- It is a sequential procedure:



Each data point has

a class label:

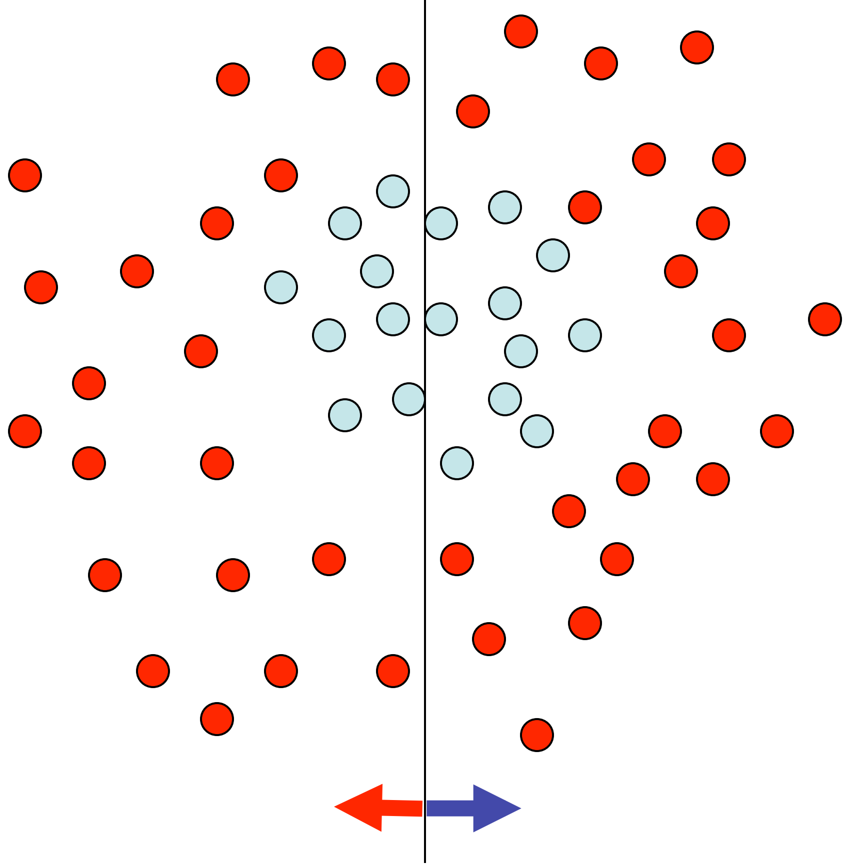
$$y_t = \begin{cases} +1 \ (\bullet) \\ -1 \ (\circ) \end{cases}$$

and a weight:

$w_t = 1$

Toy example

Weak learners from the family of lines



Each data point has

a class label:

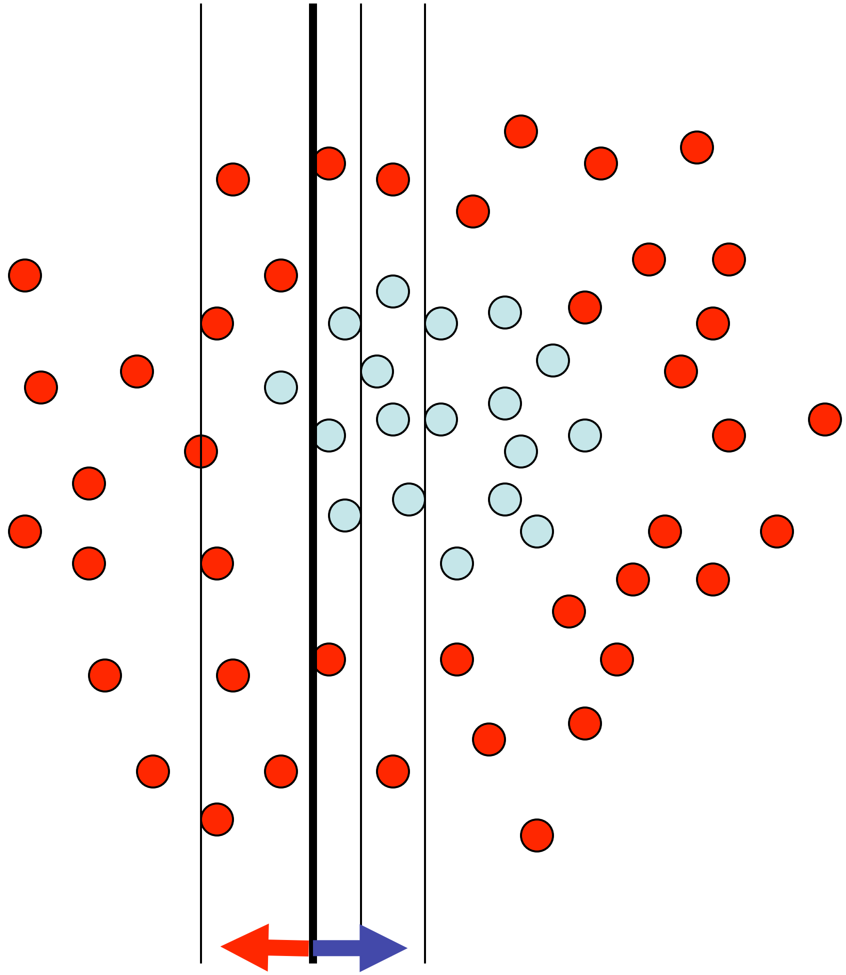
$$y_t = \begin{cases} +1 \\ -1 \end{cases}$$

and a weight:

$w_t = 1$

h => p(error) = 0.5 it is at chance

Toy example



Each data point has

a class label:

$$y_t = \begin{cases} +1 \ (\bullet) \\ -1 \ (\circ) \end{cases}$$

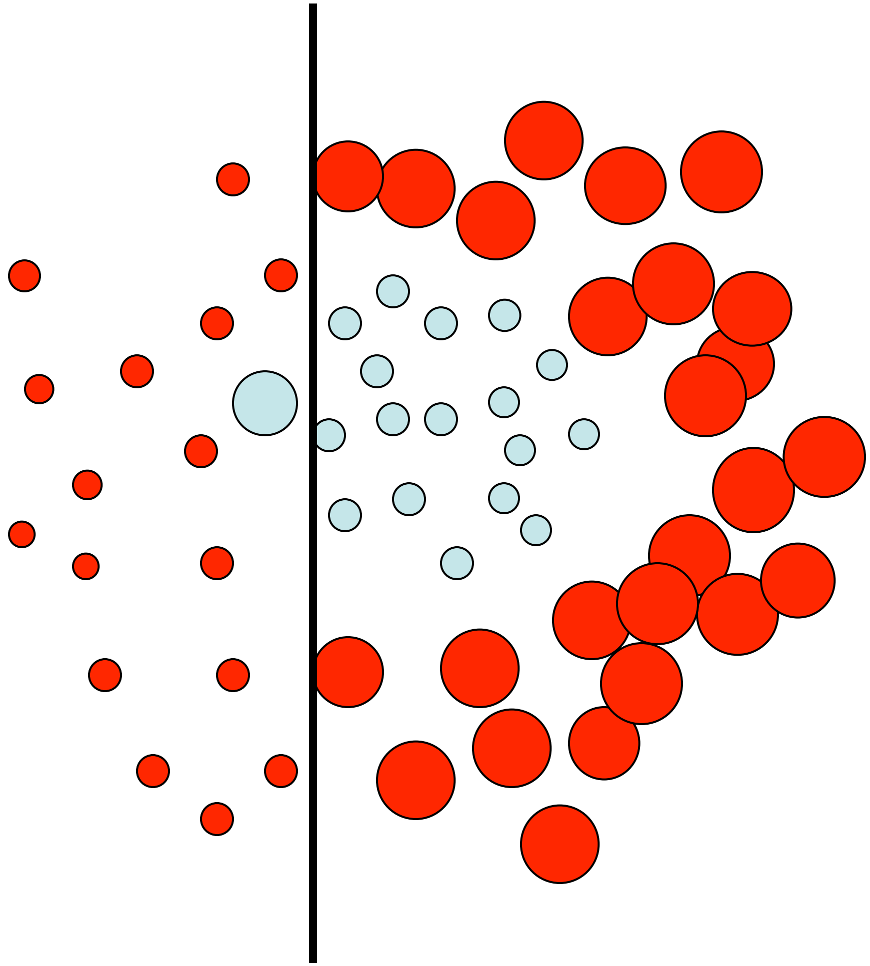
and a weight:

$w_t = 1$

This one seems to be the best

This is a ‘**weak classifier**’: It performs slightly better than chance.

Toy example



Each data point has a class label:

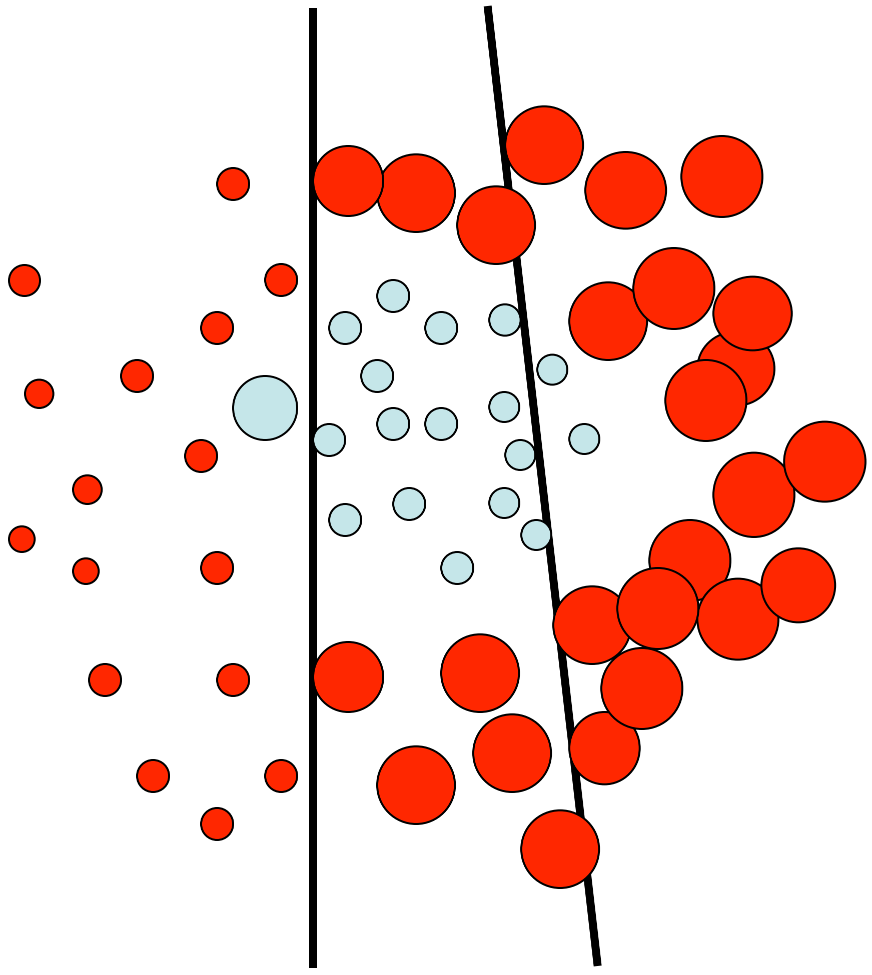
$$y_t = \begin{cases} +1 & (\bullet) \\ -1 & (\circ) \end{cases}$$

We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t H_t\}$$

We set a new problem for which the previous weak classifier performs at chance again

Toy example



Each data point has

a class label:

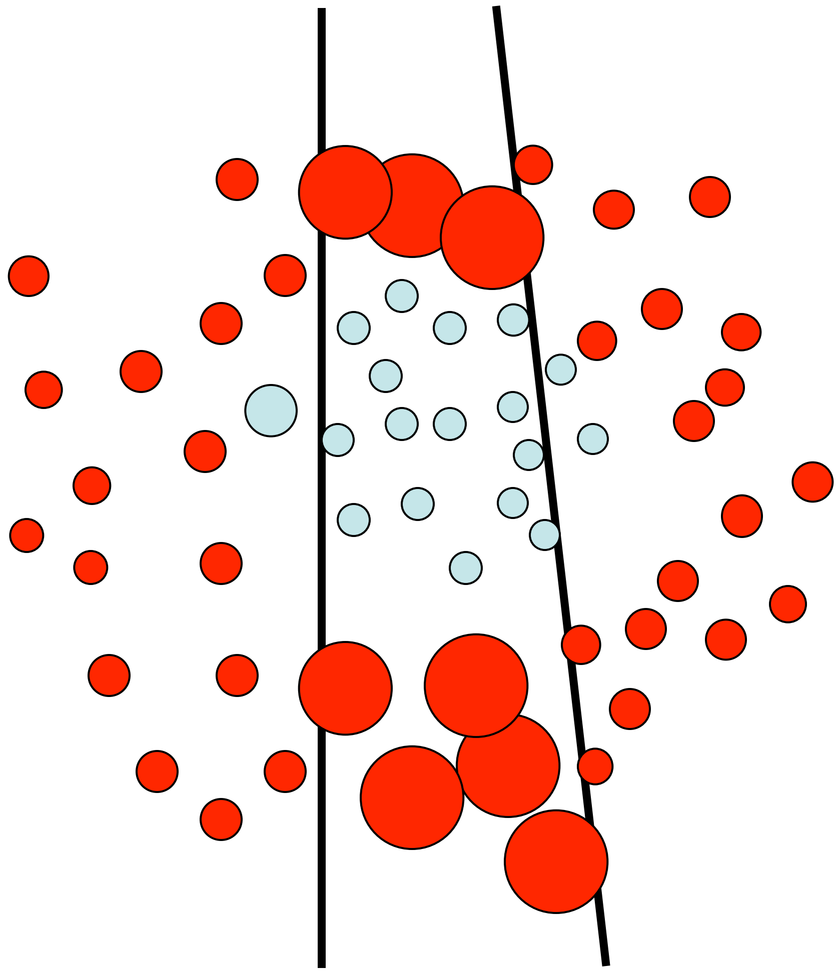
$$y_t = \begin{cases} +1 \ (\text{red}) \\ -1 \ (\text{blue}) \end{cases}$$

We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t H_t\}$$

We set a new problem for which the previous weak classifier performs at chance again

Toy example



Each data point has

a class label:

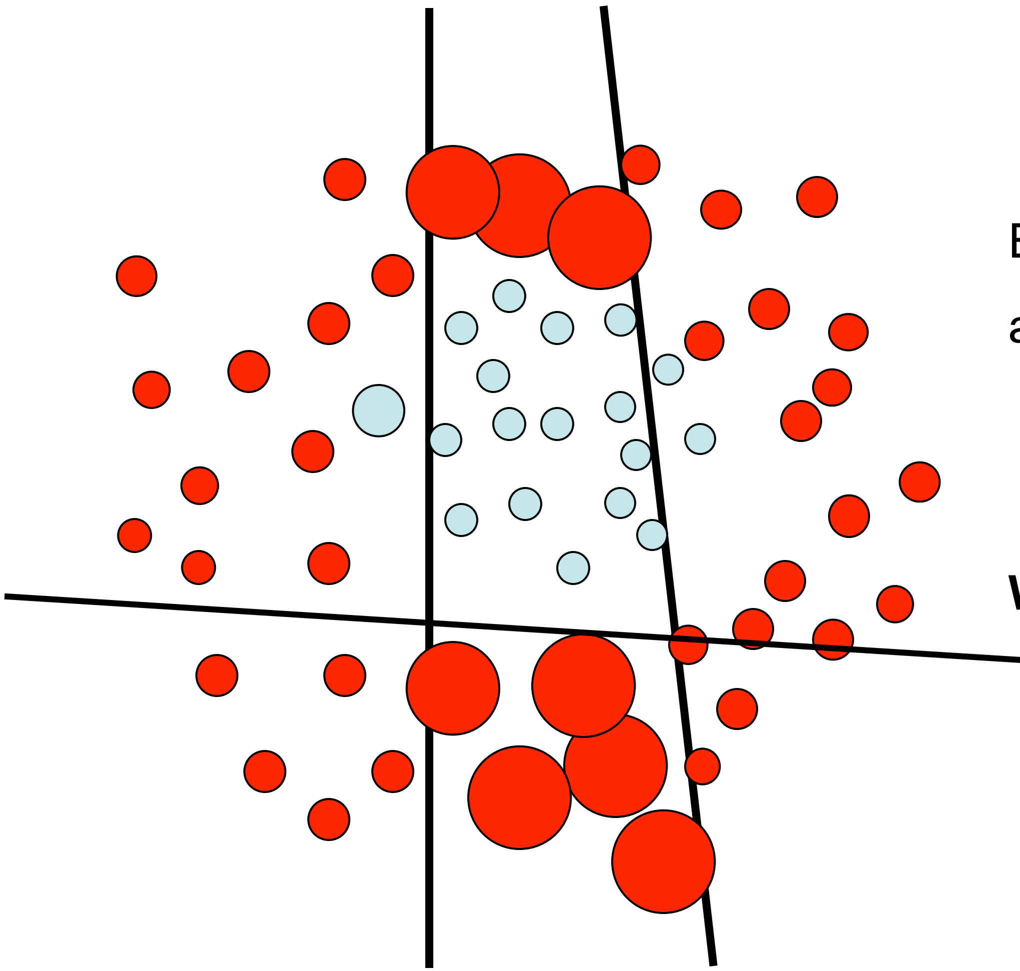
$$y_t = \begin{cases} +1 \ (\bullet) \\ -1 \ (\circ) \end{cases}$$

We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t H_t\}$$

We set a new problem for which the previous weak classifier performs at chance again

Toy example



Each data point has

a class label:

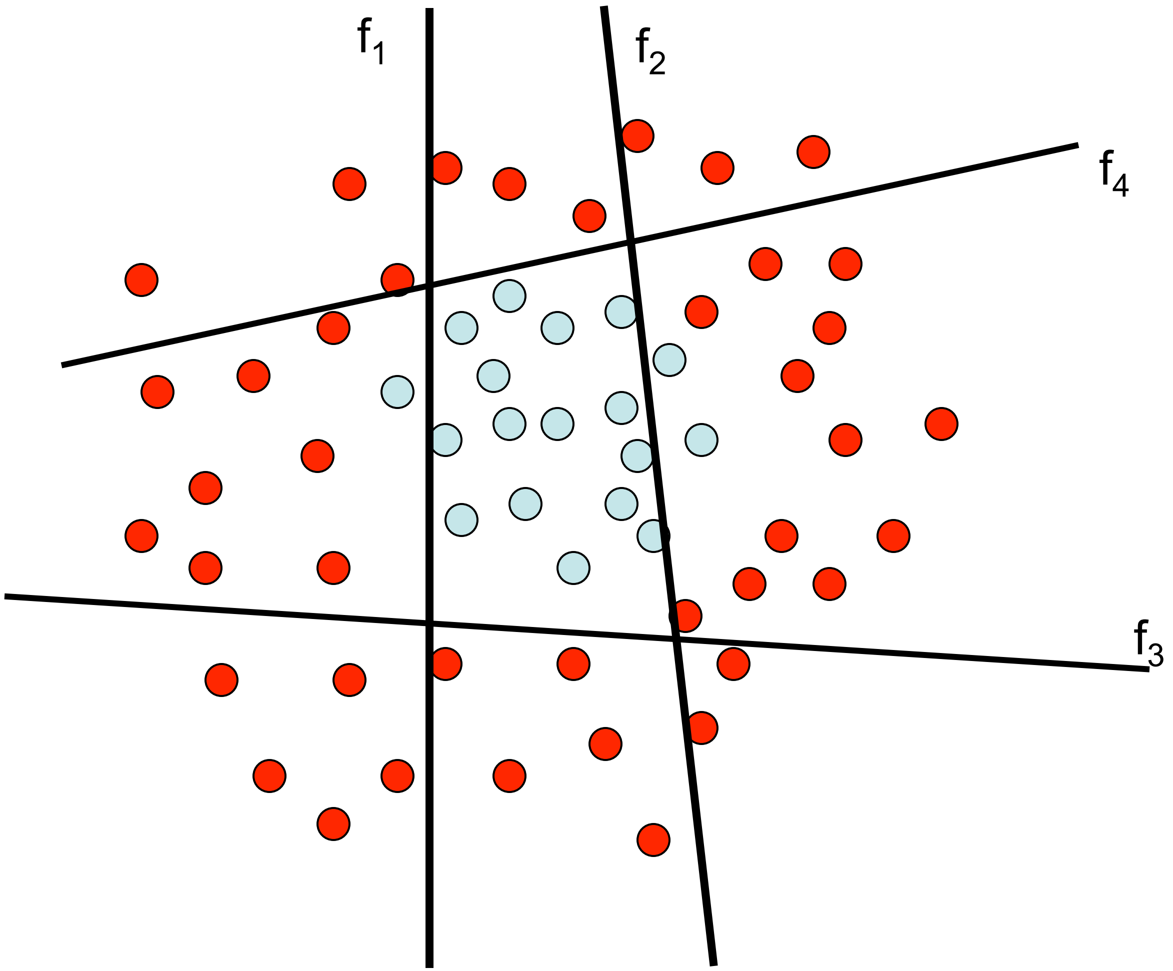
$$y_t = \begin{cases} +1 \ (\quad) \\ -1 \ (\quad) \end{cases}$$

We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t H_t\}$$

We set a new problem for which the previous weak classifier performs at chance again

Toy example



The strong (non- linear) classifier is built as the combination of all the weak (linear) classifiers.

Boosting

- Different cost functions and minimization algorithms result is various flavors of Boosting
- In this demo, I will use gentleBoosting: it is simple to implement and numerically stable.

Overview of section

- Object detection with classifiers
- Boosting
  - **Gentle boosting**
  - Weak detectors
  - Object model
  - Object detection

Boosting

Boosting fits the additive model

$$F(x) = f_1(x) + f_2(x) + f_3(x) + \dots$$

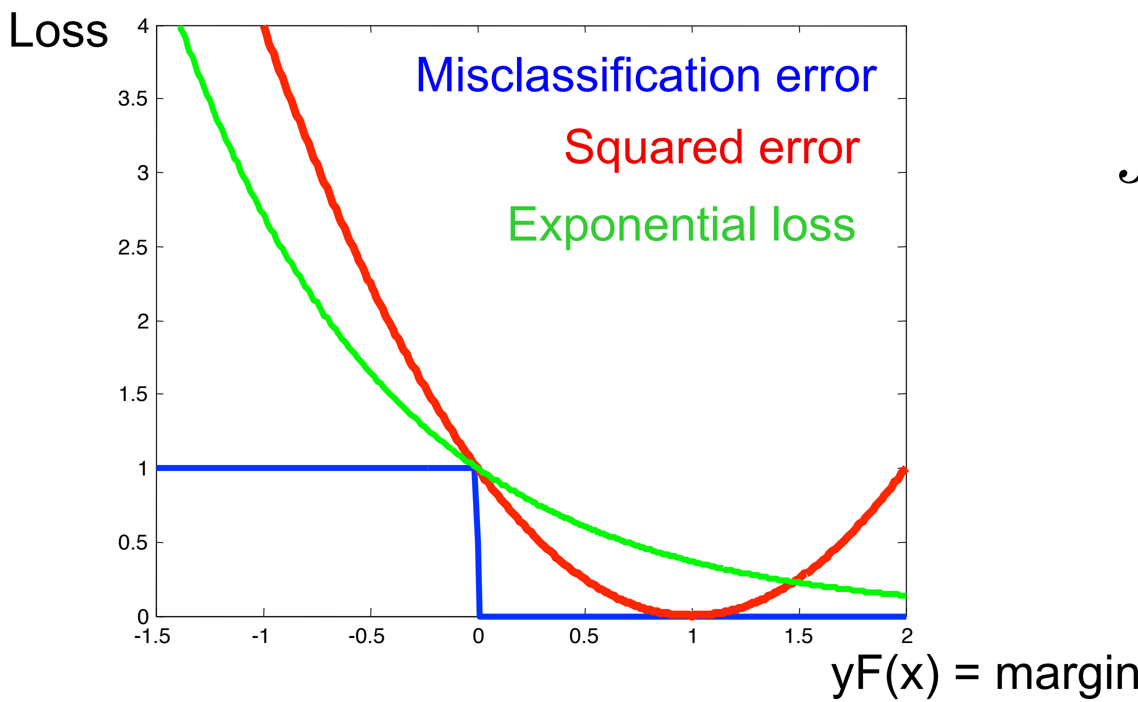
by minimizing the exponential loss

$$J(F) = \sum_{t=1}^{N} e^{-y_t F(x_t)}$$

Training samples

The exponential loss is a differentiable upper bound to the misclassification error.

Exponential loss



Squared error

$$J = \sum_{t=1}^{N} [y_t - F(x_t)]^2$$

Exponential loss

$$J = \sum_{t=1}^{N} e^{-y_t F(x_t)}$$

Boosting

Sequential procedure. At each step we add

$$F(x) \leftarrow F(x) + f_m(x)$$

to minimize the residual loss

$$(\phi_m) = \arg\min_{\phi} \sum_{t=1}^{N} J\left(y_i, F(x_t) + f(x_t; \phi)\right)$$

Parameters weak classifier

Desired output

input

For more details: Friedman, Hastie, Tibshirani. "Additive Logistic Regression: a Statistical View of Boosting" (1998)

gentleBoosting

- At each iteration:

We chose $f_m(x)$ that minimizes the cost:

$$J(F + f_m) = \sum_{t=1}^{N} e^{-y_t(F(x_t)+f_m(x_t))}$$

Instead of doing exact optimization, gentle Boosting minimizes a Taylor approximation of the error:

$$J(F) \propto \sum_{t=1}^{N} \boxed{e^{-y_t F(x_t)}} (y_t - f_m(x_t))^2$$

Weights at this iteration

At each iterations we just need to solve a weighted least squares problem

For more details: Friedman, Hastie, Tibshirani. “Additive Logistic Regression: a Statistical View of Boosting” (1998)

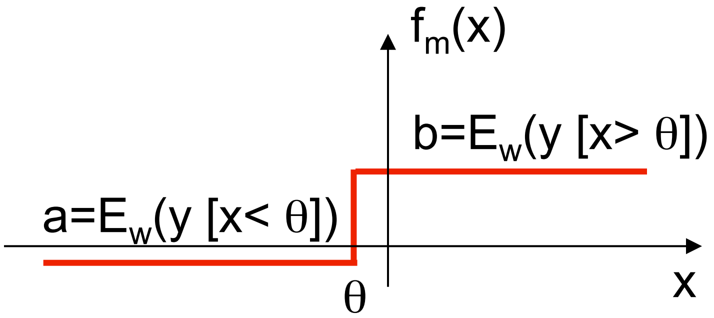
Weak classifiers

- The input is a set of weighted training samples (x,y,w)

- Regression stumps: simple but commonly used in object detection.

$$f_m(x) = a[x_k < \theta] + b[x_k \geq \theta]$$

Four parameters: $[a, b, \theta, k]$



fitRegressionStump.m

gentleBoosting.m

```
function classifier = gentleBoost(x, y, Nrounds)

…

for m = 1:Nrounds

    fm = selectBestWeakClassifier(x, y, w);

    w = w .* exp(- y .* fm);

    % store parameters of fm in classifier
    …
end
```

Initialize weights w = 1

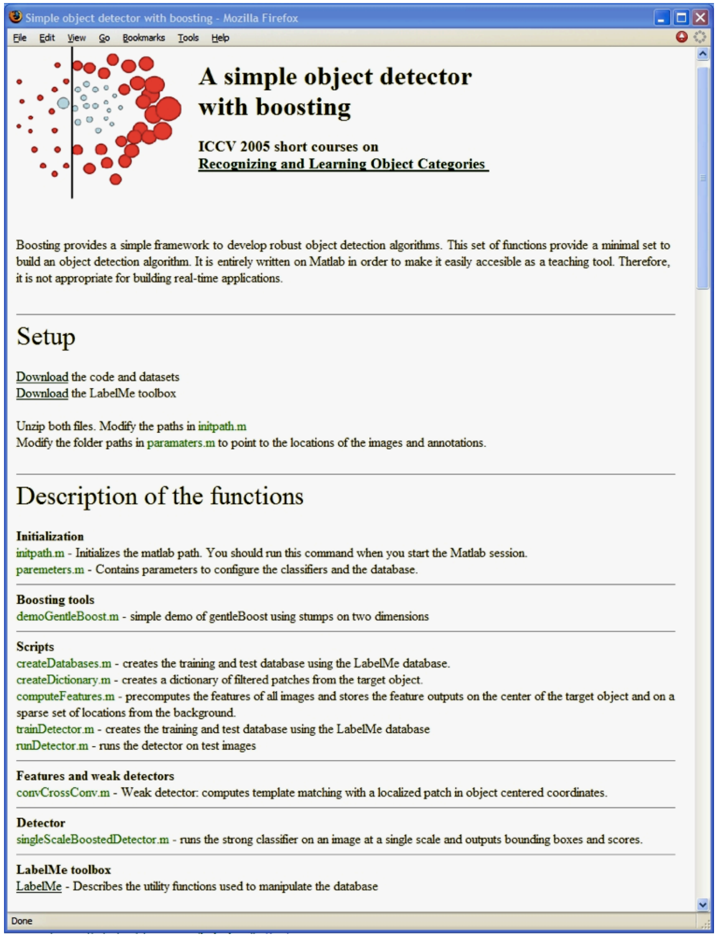
Solve weighted least-squares

Re-weight training samples

Demo gentleBoosting

Demo using Gentle boost and stumps with hand selected 2D data:

> demoGentleBoost.m



Flavors of boosting

- AdaBoost (Freund and Shapire, 1995)
- Real AdaBoost (Friedman et al, 1998)
- LogitBoost (Friedman et al, 1998)
- Gentle AdaBoost (Friedman et al, 1998)
- BrownBoosting (Freund, 2000)
- FloatBoost (Li et al, 2002)
- …

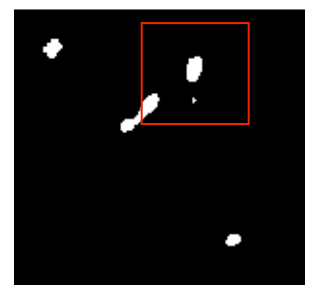
Overview of section

- Object detection with classifiers
- Boosting
  - Gentle boosting
  - **Weak detectors**
  - Object model
  - Object detection

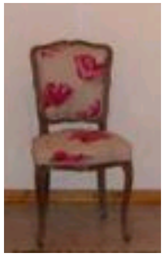
From images to features: Weak detectors

We will now define a family of visual features that can be used as weak classifiers ("weak detectors")

$$\longrightarrow h_i(I, x, y) \longrightarrow$$



Takes image as input and the output is binary response.
The output is a weak detector.



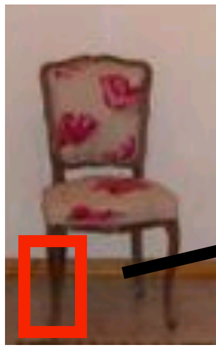
Object recognition
Is it really so hard?

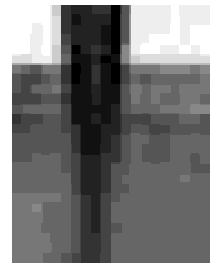
Find the chair in this image

But what if we use smaller patches? Just a part of the chair?

Parts

But what if we use smaller patches? Just a part of the chair?





Find a chair in this image

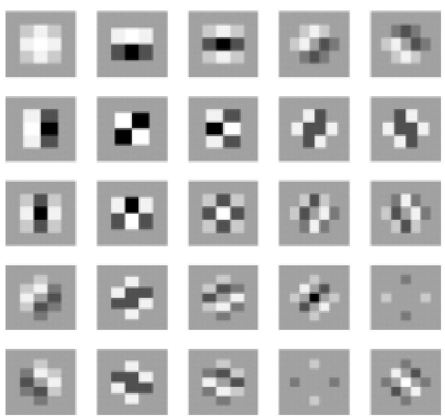
Seems to fire on legs… not so bad

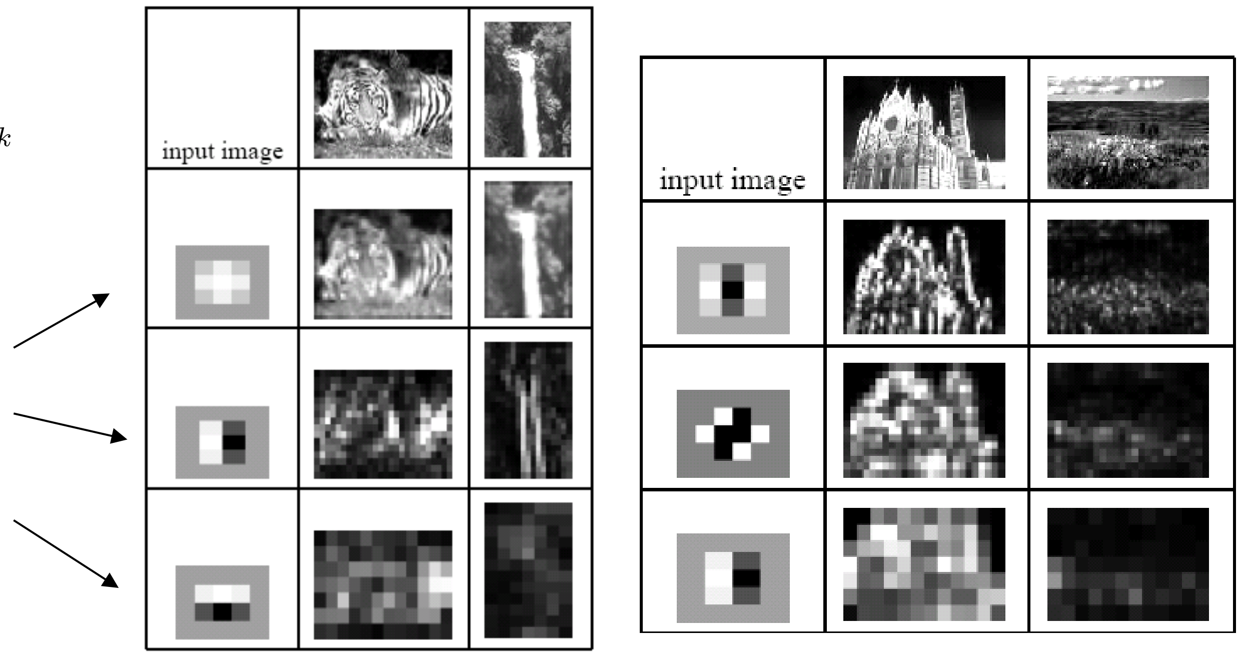
Weak detectors

Textures of textures

Tieu and Viola, CVPR 2000. One of the first papers to use boosting for vision.

$$g_{i,j,k} = \sum_{pixels} ||I * f_i| \downarrow_2 * f_j| \downarrow_2 * f_k$$





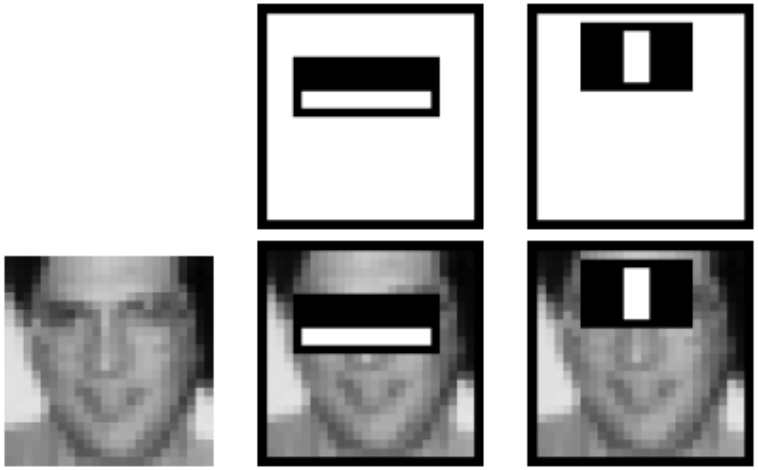
Every combination of three filters generates a different feature

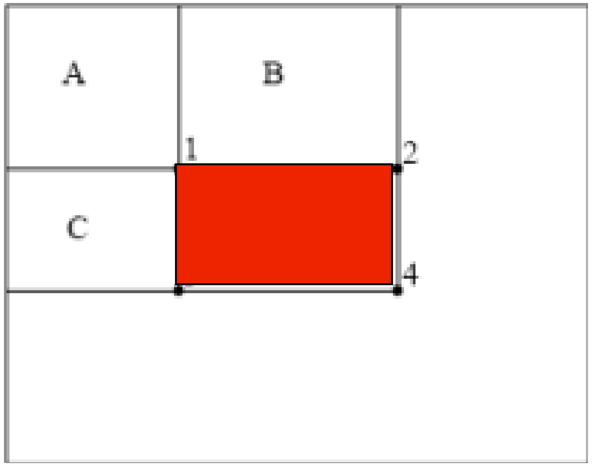
This gives thousands of features. Boosting selects a sparse subset, so computations on test time are very efficient. Boosting also avoids overfitting to some extend.

Weak detectors

Haar filters and integral image

Viola and Jones, ICCV 2001





The average intensity in the block is computed with four sums independently of the block size.

Edge fragments

J. Shotton, A. Blake, R. Cipolla.
Multi-Scale Categorical Object Recognition
Using Contour Fragments. In *IEEE Trans. on PAMI*, 30(7):1270-1281, July 2008.

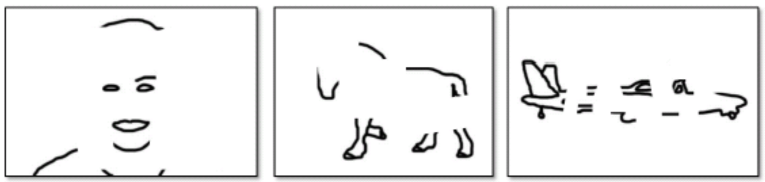
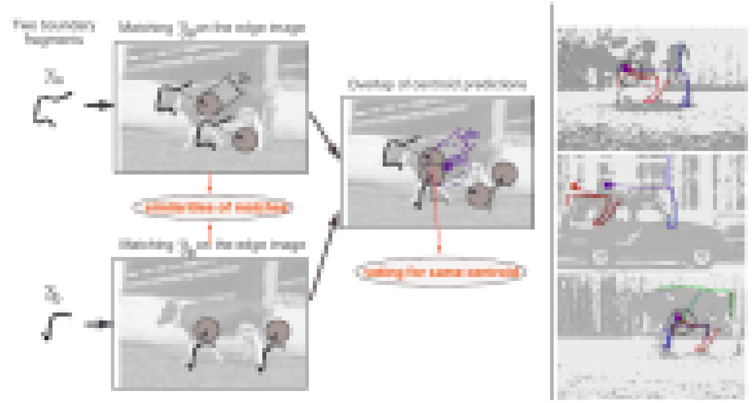
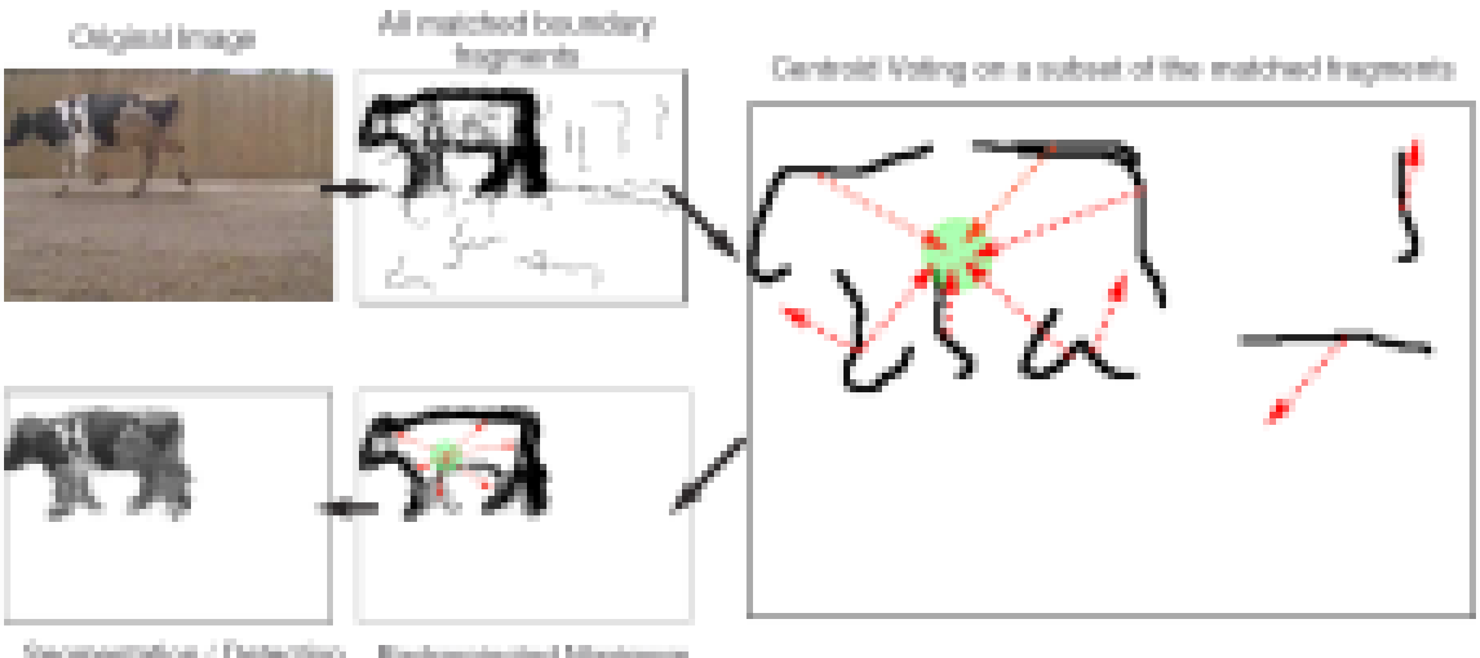


Fig. 1. **Object recognition using contour fragments.** Our innate biological vision system is able to interpret spatially arranged local fragments of contour to recognize the objects present. In this work we show that an automatic computer vision system can also successfully exploit the cue of contour for object recognition.

Opelt, Pinz, Zisserman, ECCV 2006





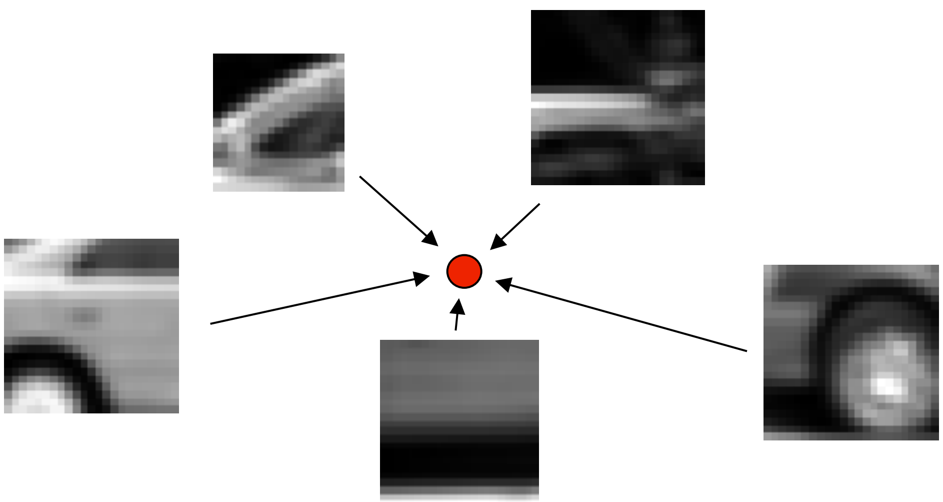
Weak detectors

Other weak detectors:

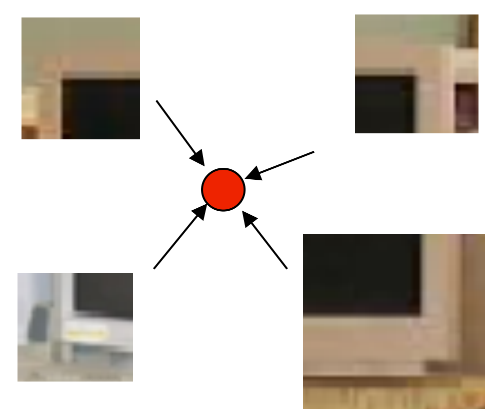
- Carmichael, Hebert 2004
- Yuille, Snow, Nitzbert, 1998
- Amit, Geman 1998
- Papageorgiou, Poggio, 2000
- Heisele, Serre, Poggio, 2001
- Agarwal, Awan, Roth, 2004
- Schneiderman, Kanade 2004
- …

Weak detectors

Part based: similar to part-based generative models. We create weak detectors by using parts and voting for the object center location



Car model



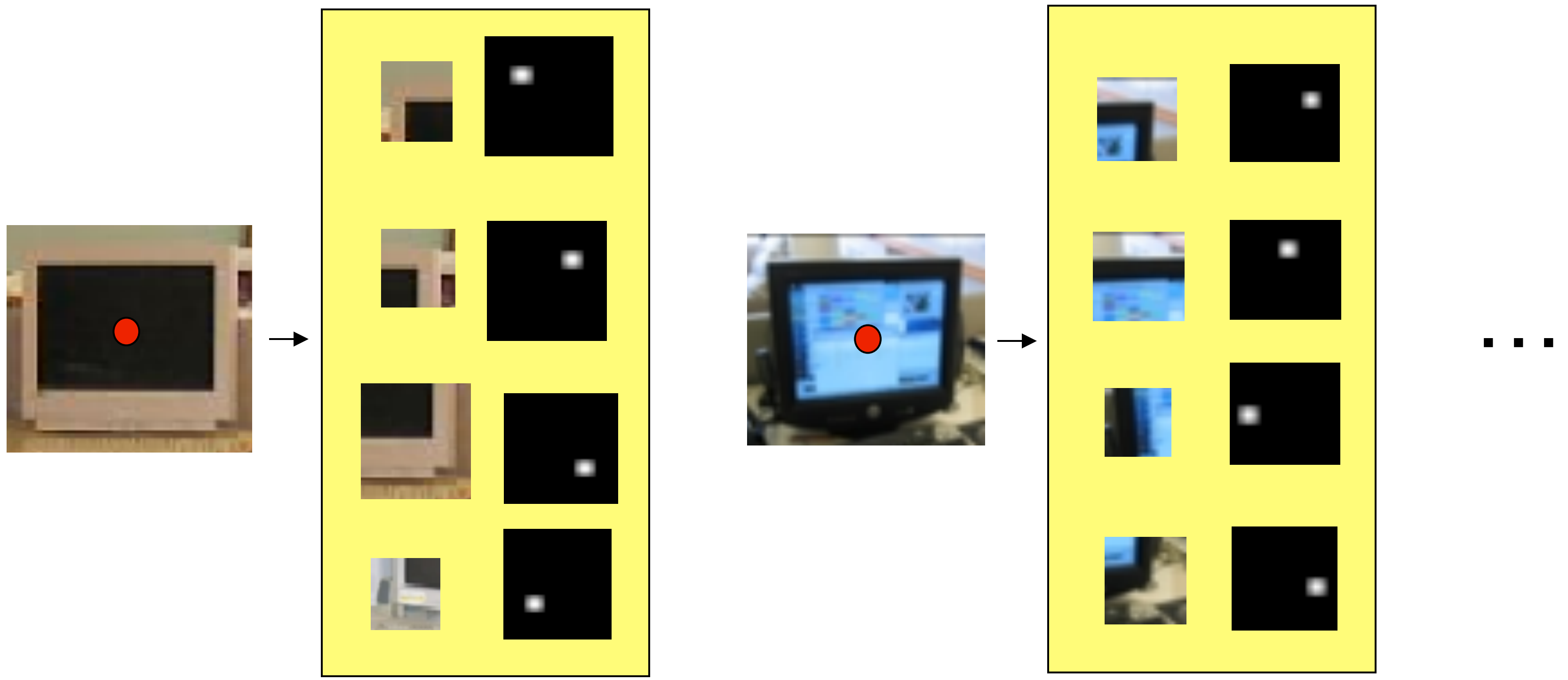
Screen model

These features are used for the detector on the course web site.

Weak detectors

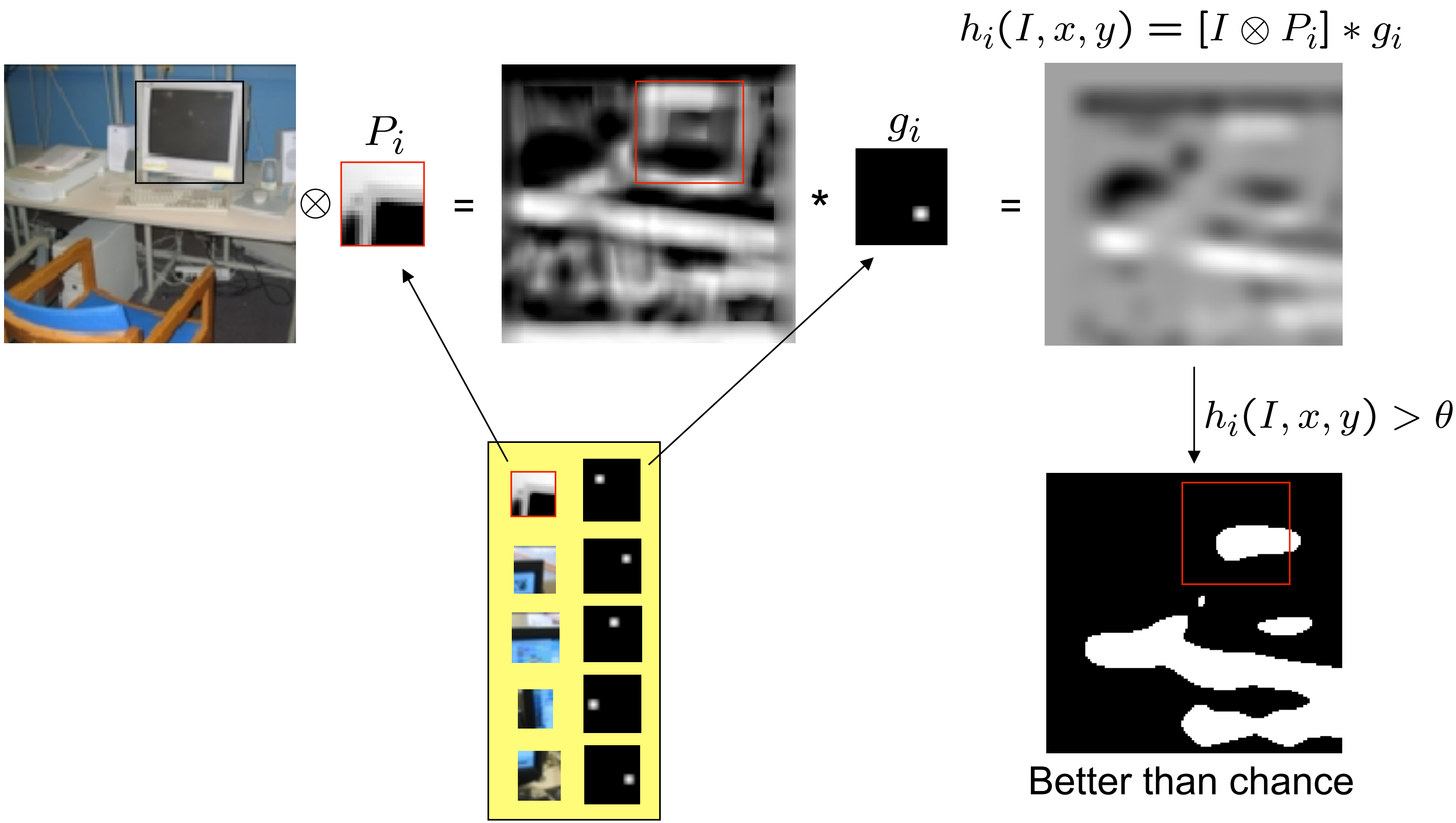
First we collect a set of part templates from a set of training objects.

Vidal-Naquet, Ullman (2003)



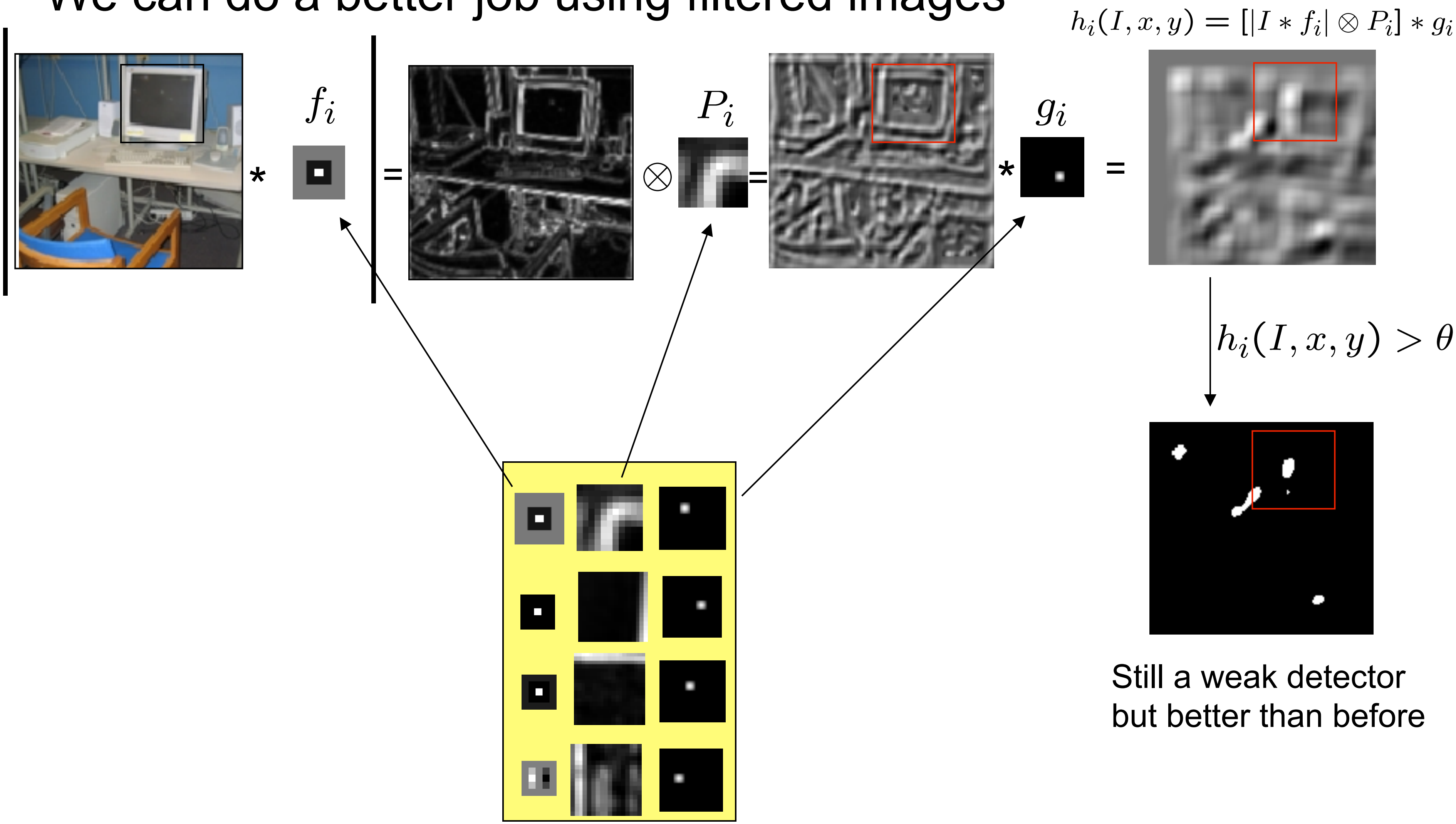
Weak detectors

We now define a family of “weak detectors” as:



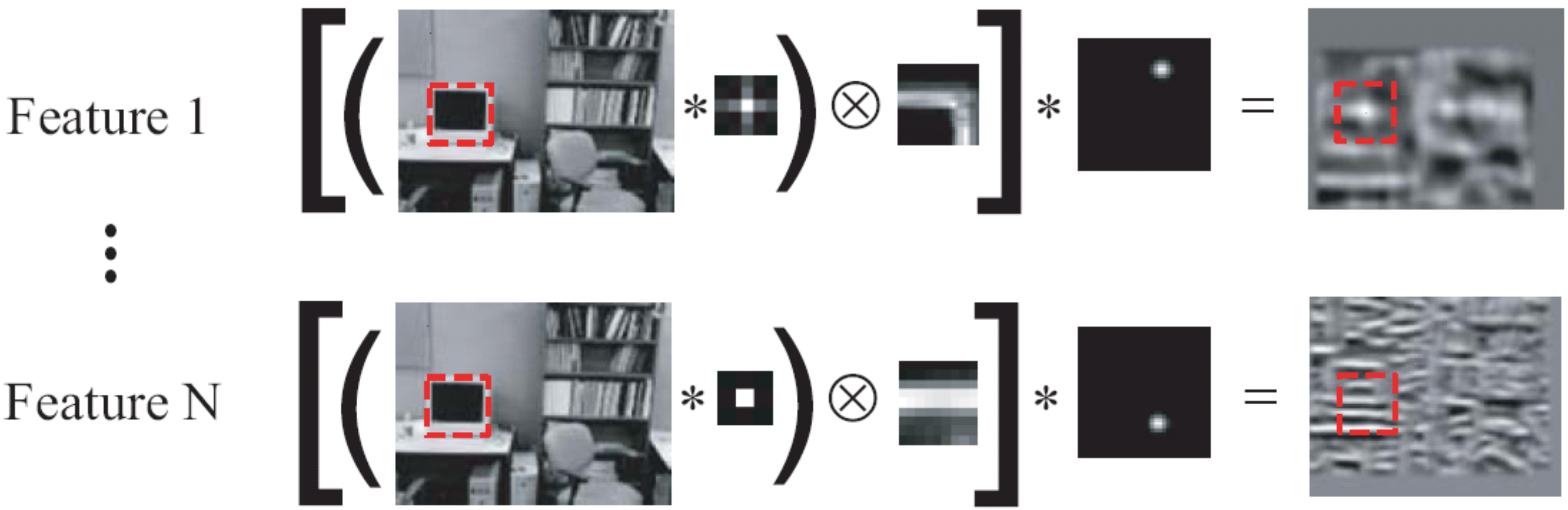
Weak detectors

We can do a better job using filtered images

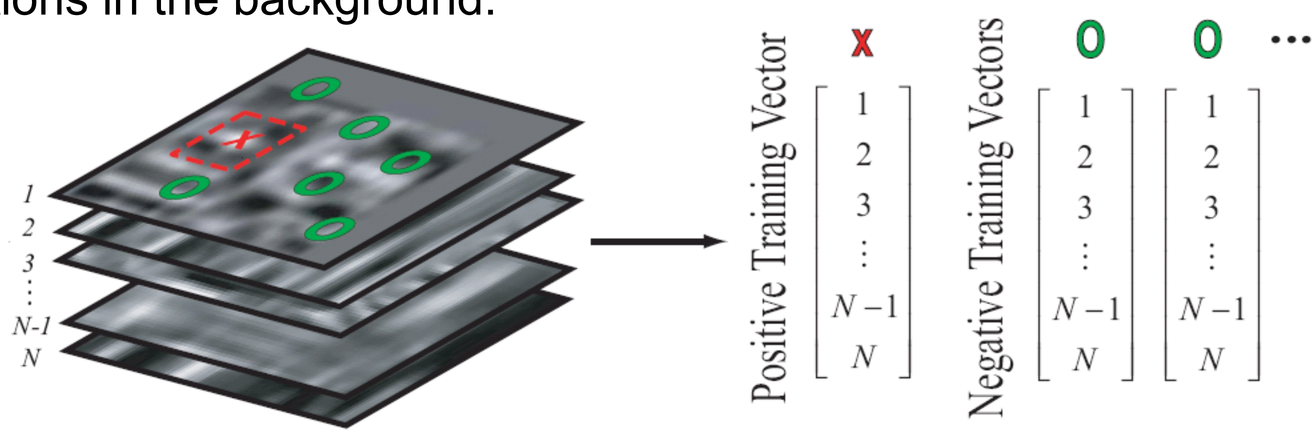


Training

First we evaluate all the N features on all the training images.



Then, we sample the feature outputs on the object center and at random locations in the background:



gentleBoosting.m

```
function classifier = gentleBoost(x, y, Nrounds)

…

for m = 1:Nrounds

    fm = selectBestWeakClassifier(x, y, w);

    w = w .* exp(- y .* fm);

    % store parameters of fm in classifier
    …
end
```

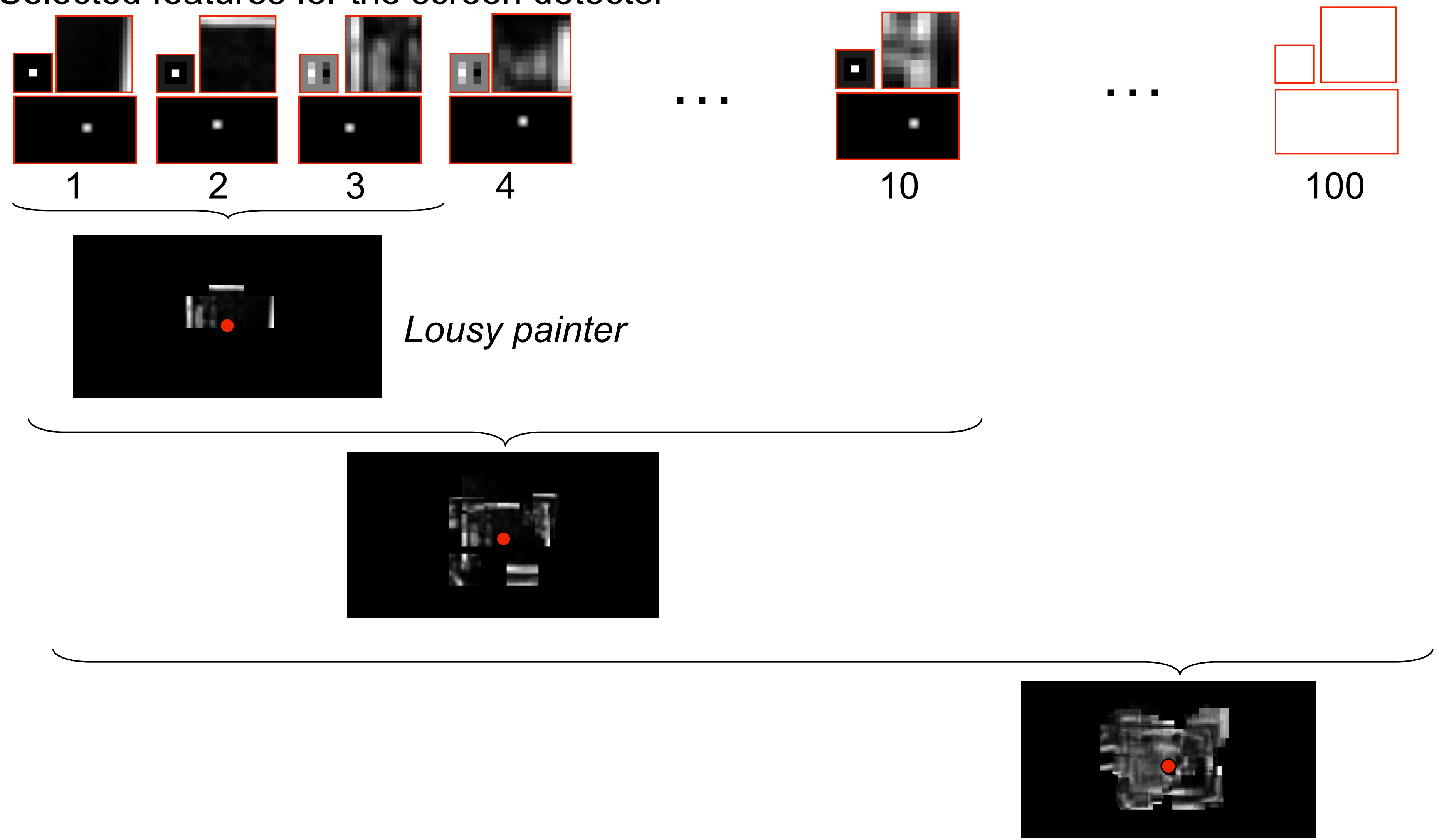
Initialize weights w = 1

Solve weighted least-squares

Re-weight training samples

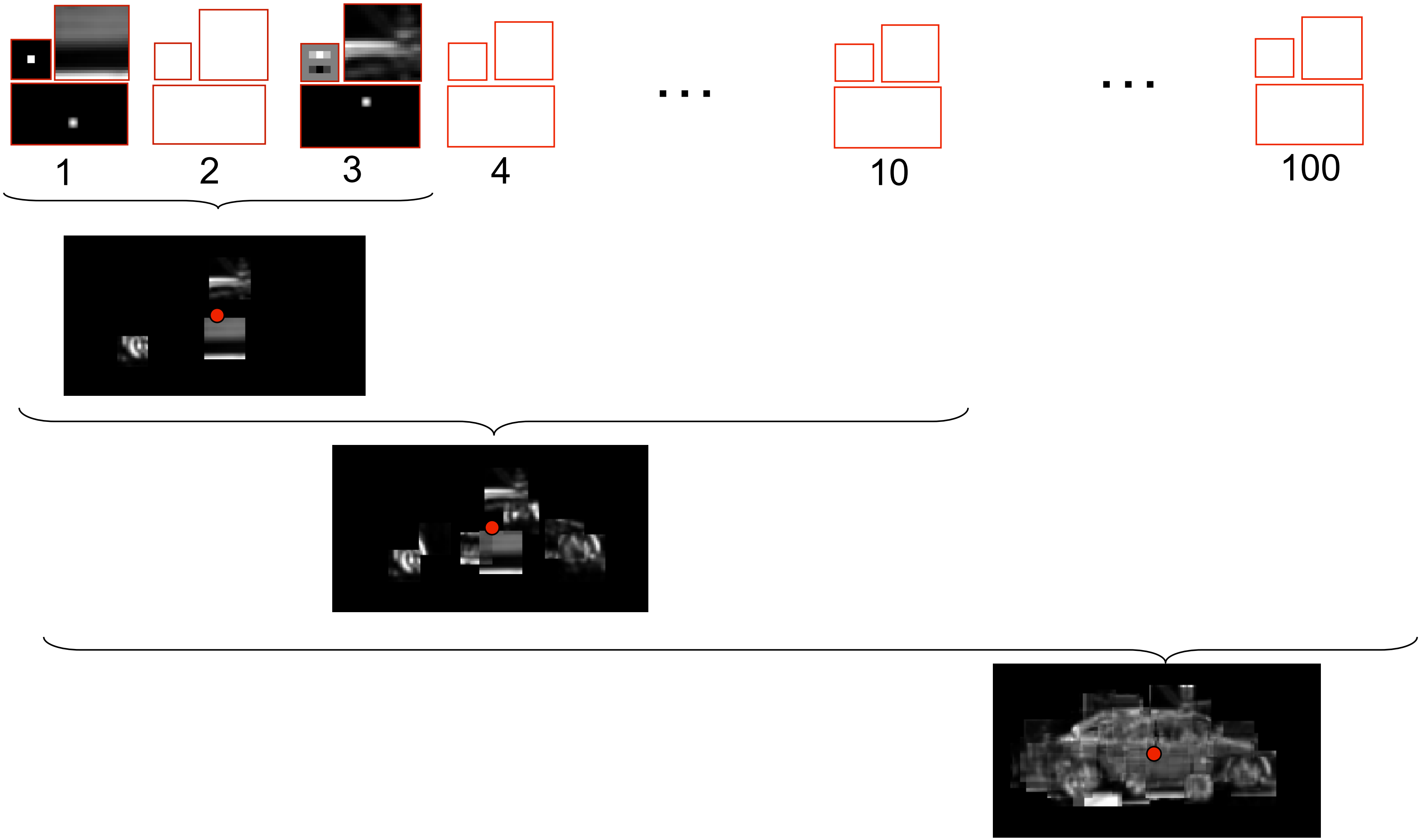
Representation and object model

Selected features for the screen detector



Representation and object model

Selected features for the car detector



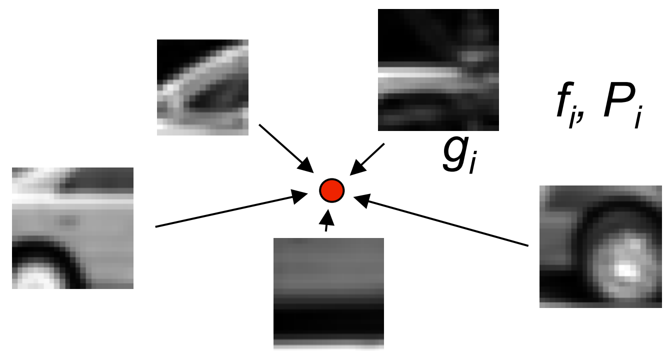
Overview of section

- Object detection with classifiers

- Boosting
  - Gentle boosting
  - Weak detectors
  - Object model
  - **Object detection**

Object model

- Voting

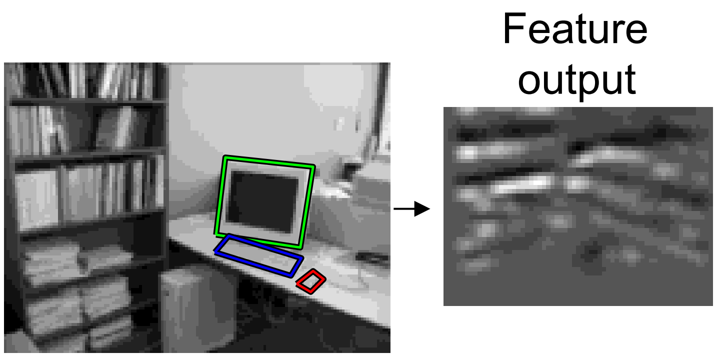


- Invariance: search strategy

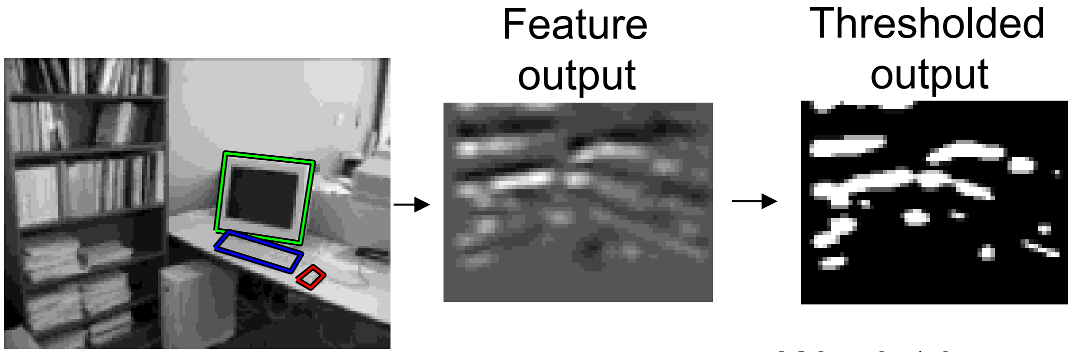
Here, invariance in translation and scale is achieved by the search strategy: the classifier is evaluated at all locations (by translating the image) and at all scales (by scaling the image in small steps).

The search cost can be reduced using a cascade.

Example: screen detection



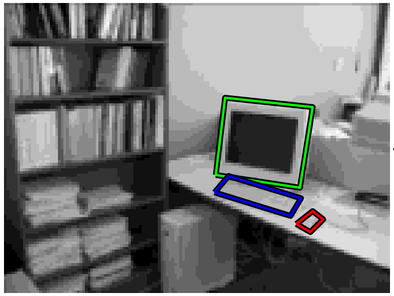
Example: screen detection



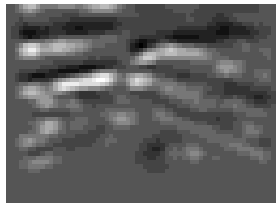
Weak ‘detector’

Produces many false alarms.

Example: screen detection



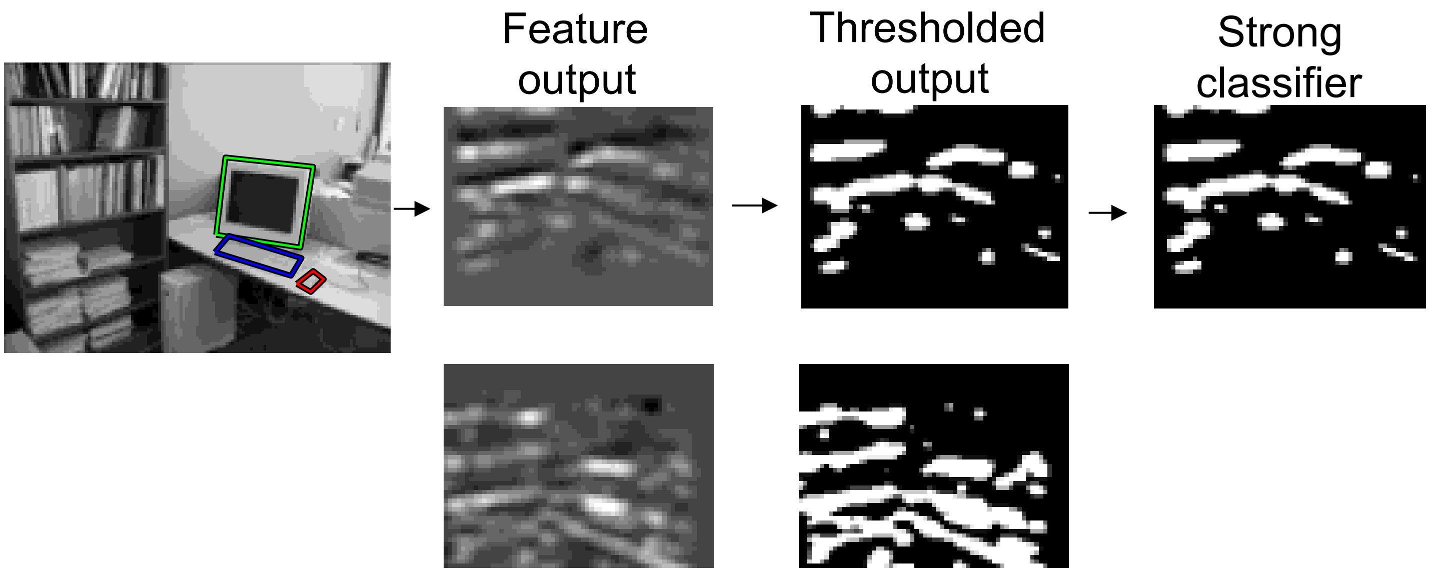
Feature output



Thresholded output

Strong classifier at iteration 1

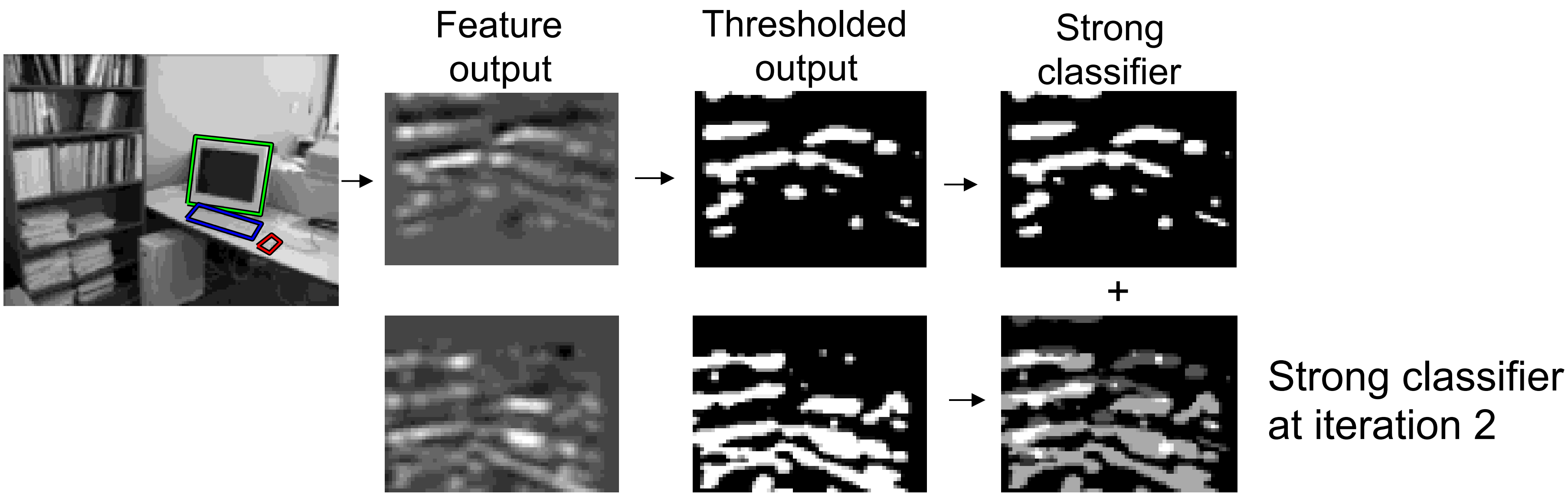
Example: screen detection



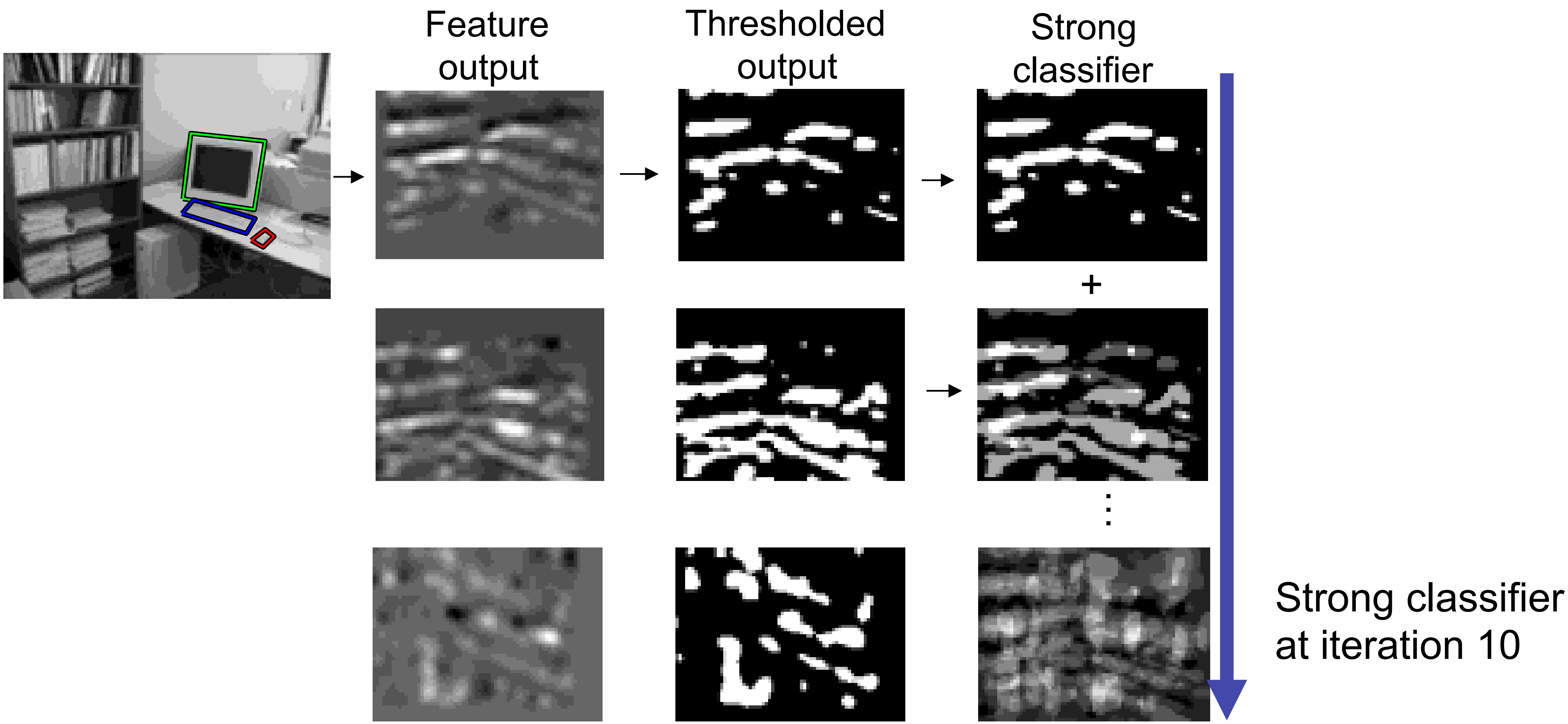
Second weak ‘detector’

Produces a different set of
false alarms.

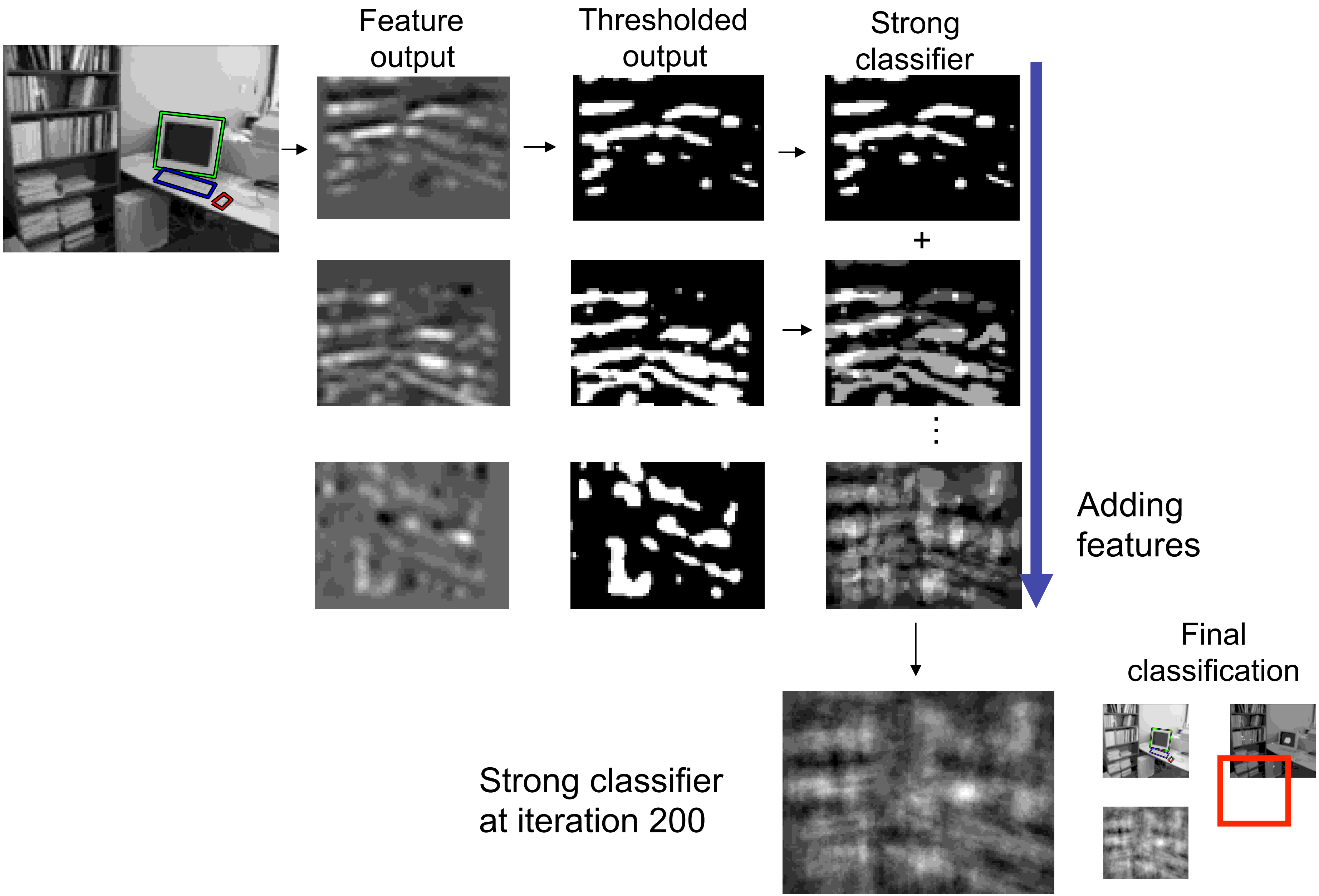
Example: screen detection



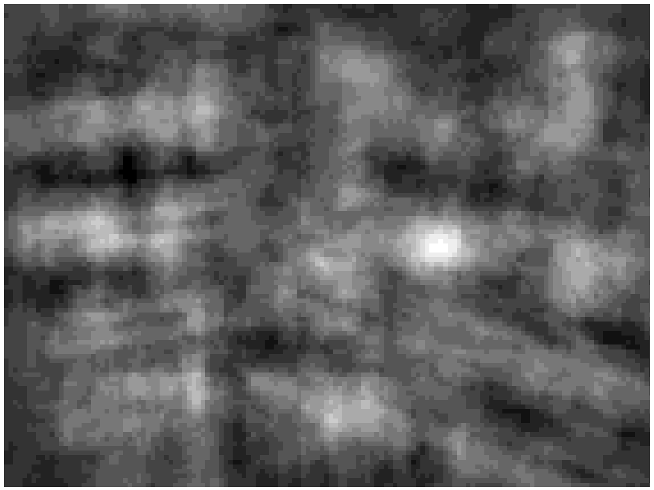
Example: screen detection

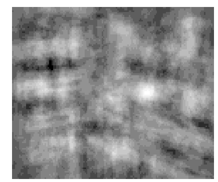


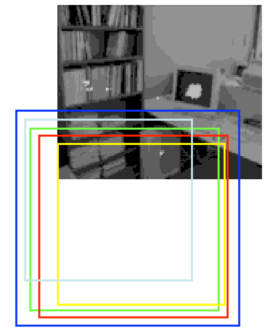
Example: screen detection



Maximal suppression







Detect local maximum of the response. We are only allowed detecting each object once. The rest will be considered false alarms.

This post-processing stage can have a very strong impact in the final performance.

Evaluation

When do we have a correct detection?

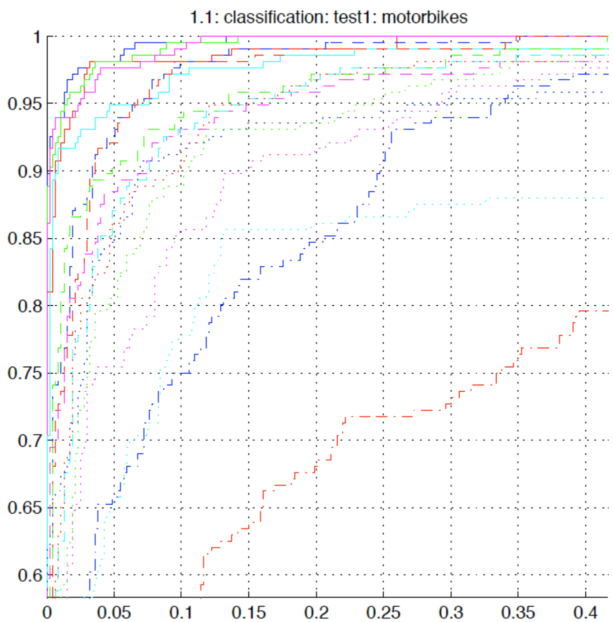
Is this correct?

$$\frac{\text{Area intersection}}{\text{Area union}} > 0.5$$

- ROC
- Precision-recall

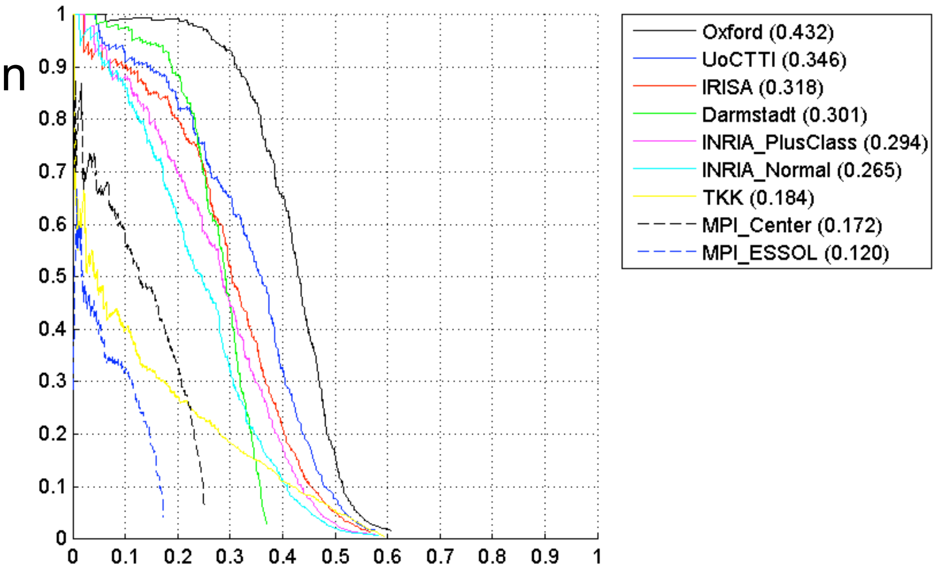
ROC and Precision-Recall

Detection rate



False alarm rate

Precision



Recall

Plots from PASCAL competition

Rapid Object Detection Using a Boosted Cascade of Simple Features

Paul Viola Michael J. Jones
Mitsubishi Electric Research Laboratories (MERL)
Cambridge, MA

Most of this work was done at Compaq CRL before the authors moved to MERL

Manuscript available on web:

http://citeseer.ist.psu.edu/cache/papers/cs/23183/http:zSzzSzwww.ai.mit.eduzSzpeoplezSzviolazSzresearchzSzpublicationszSzICCV01-Viola-Jones.pdf/viola01robust.pdf

What is novel about this approach?

- Feature set (… is huge about 16,000,000 features)
- Efficient feature selection using AdaBoost
- New image representation: Integral Image
- Cascaded Classifier for rapid detection
  - Hierarchy of Attentional Filters

What is new is the combination of these ideas. This yields the fastest known face detector for gray scale images.

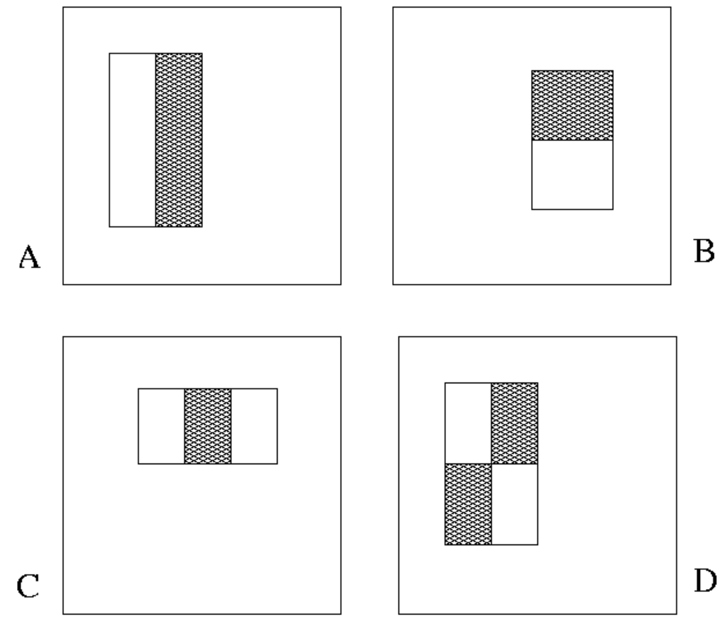
Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Image Features

“Rectangle filters”

Similar to Haar wavelets

Differences between sums of pixels in adjacent rectangles



$$h_t(x) = \begin{cases} +1 & \text{if } f_t(x) > \theta_t \\ -1 & \text{otherwise} \end{cases}$$

$$\boxed{160{,}000 \times 100 = 16{,}000{,}000}$$

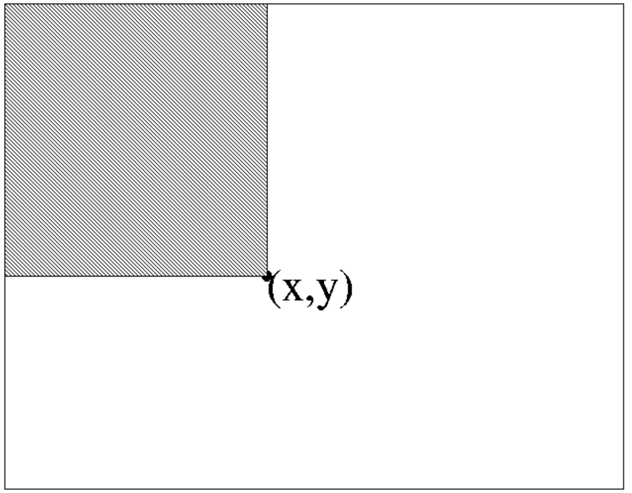
Unique Features

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Integral Image

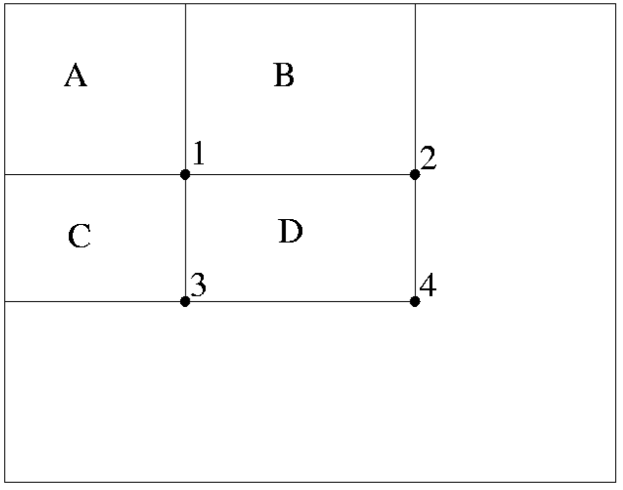
- Define the Integral Image

$$I'(x,y) = \sum_{\substack{x' \le x \\ y' \le y}} I(x', y')$$



- Any rectangular sum can be computed in constant time:

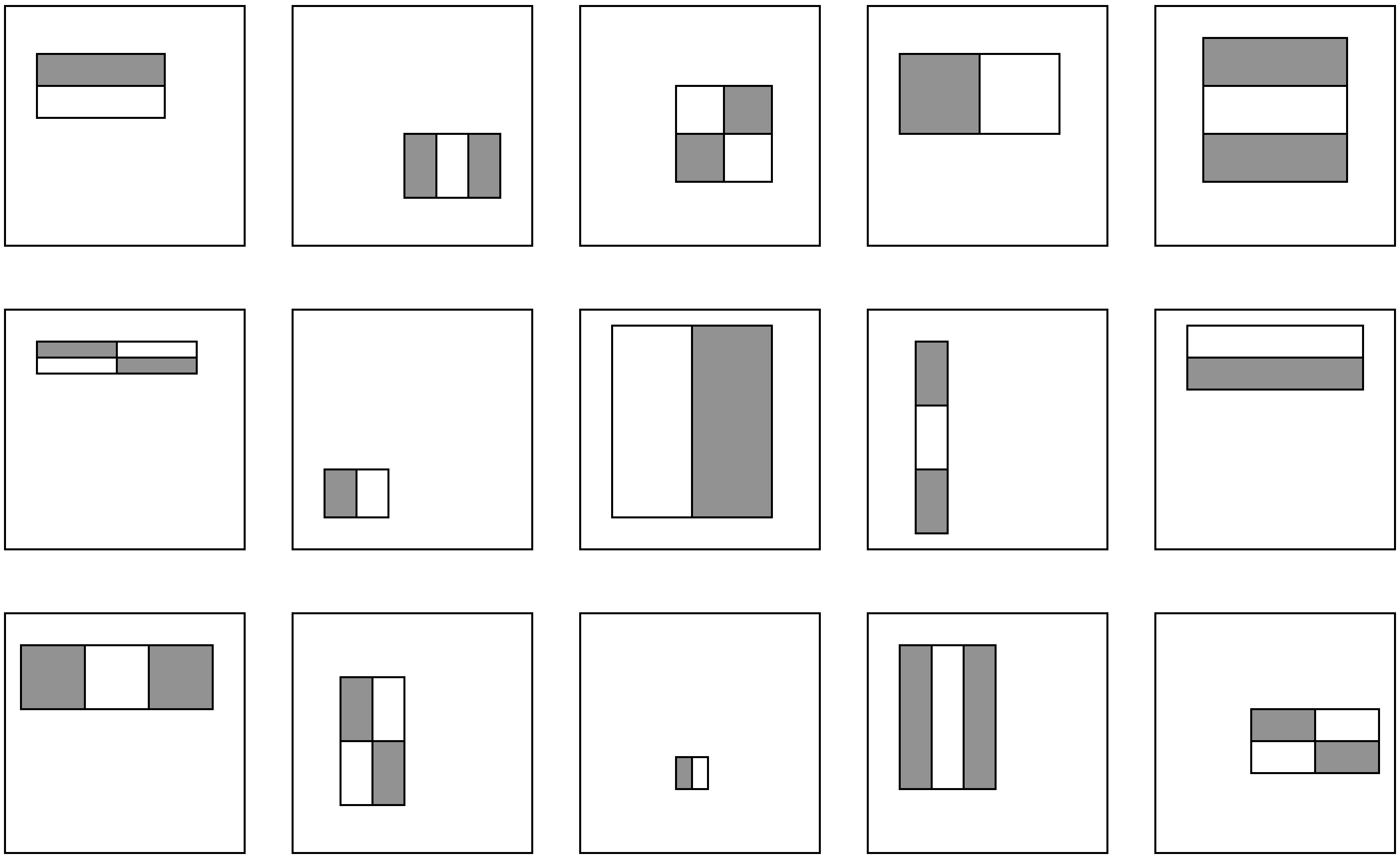
$$\begin{aligned} D &= 1 + 4 - (2 + 3) \\ &= A + (A + B + C + D) - (A + C + A + B) \\ &= D \end{aligned}$$



- Rectangle features can be computed as differences between rectangles

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Huge "Library" of Filters



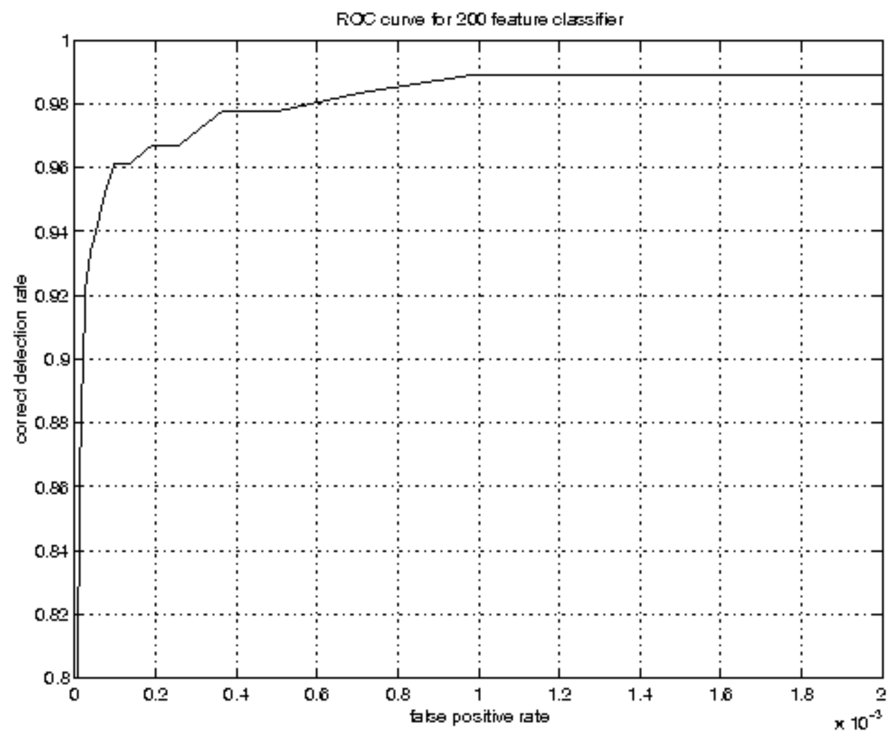
Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Example Classifier for Face Detection

A classifier with 200 rectangle features was learned using AdaBoost

95% correct detection on test set with 1 in 14084 false positives.

Not quite competitive.
Need to add more features,
but then that slows it down.



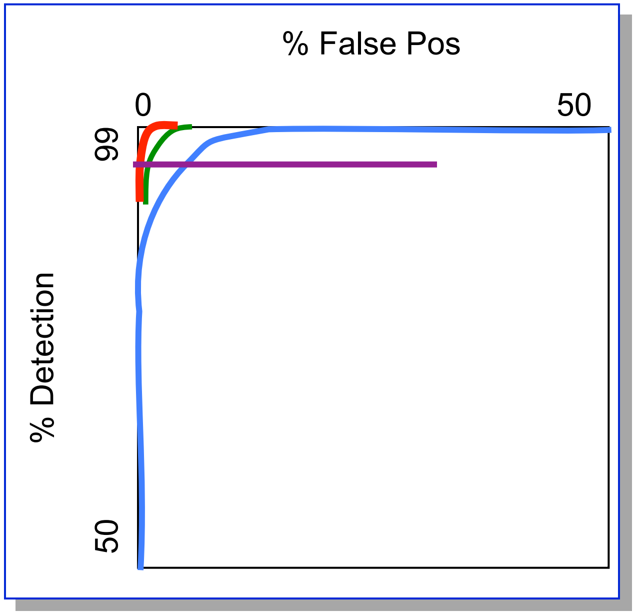
ROC curve for 200 feature classifier

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

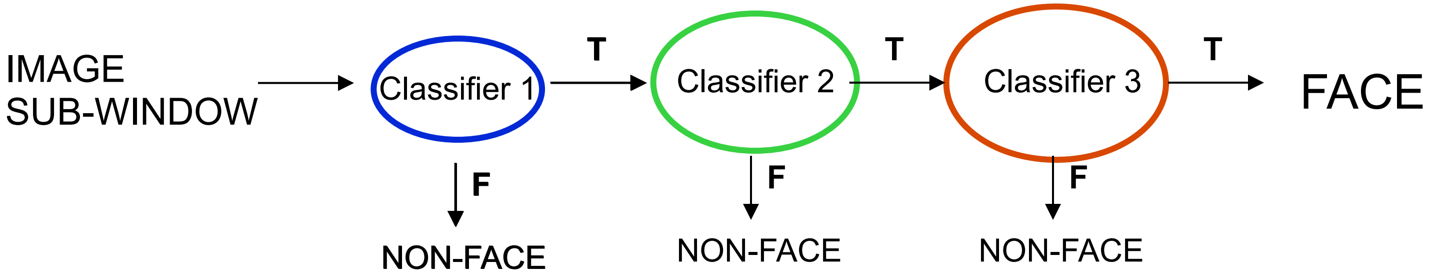
Fast and accurate classifier using a cascade

Fleuret and Geman 2001, Viola and Jones 2001

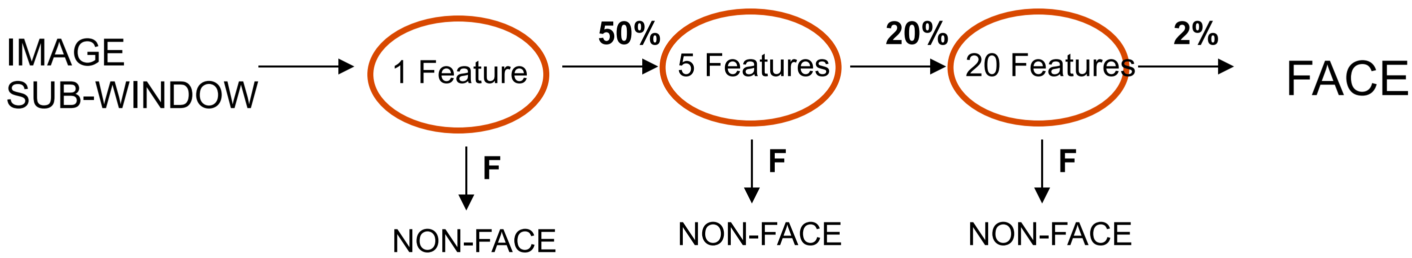
- Given a nested set of classifier hypothesis classes



- Cascade



Cascaded Classifier



- A 1 feature classifier achieves 100% detection rate and about 50% false positive rate.
- A 5 feature classifier achieves 100% detection rate and 40% false positive rate (20% cumulative)
  - using data from previous stage.
- A 20 feature classifier achieve 100% detection rate with 10% false positive rate (2% cumulative)

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

A Real-time Face Detection System

Training faces: 4916 face images (24 x 24 pixels) plus vertical flips for a total of 9832 faces

Training non-faces: 350 million sub-windows from 9500 non-face images

Final detector: 38 layer cascaded classifier
The number of features per layer was 1, 10, 25, 25, 50, 50, 50, 75, 100, …, 200, …

Final classifier contains 6061 features.

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Speed of Face Detector

Speed is proportional to the average number of features computed per sub-window.

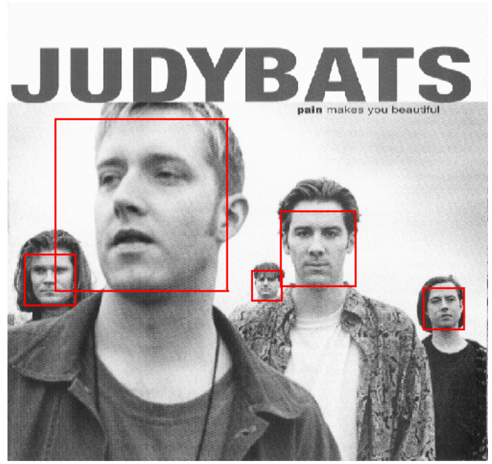
On the MIT+CMU test set, an average of 9 features out of a total of 6061 are computed per sub-window.

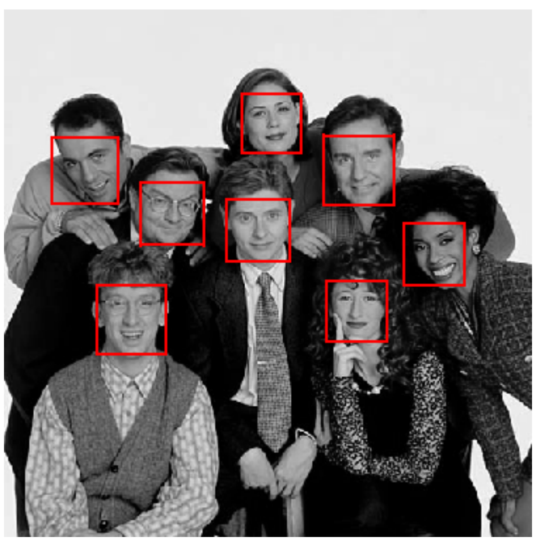
On a 700 Mhz Pentium III, a 384x288 pixel image takes about 0.067 seconds to process (15 fps).

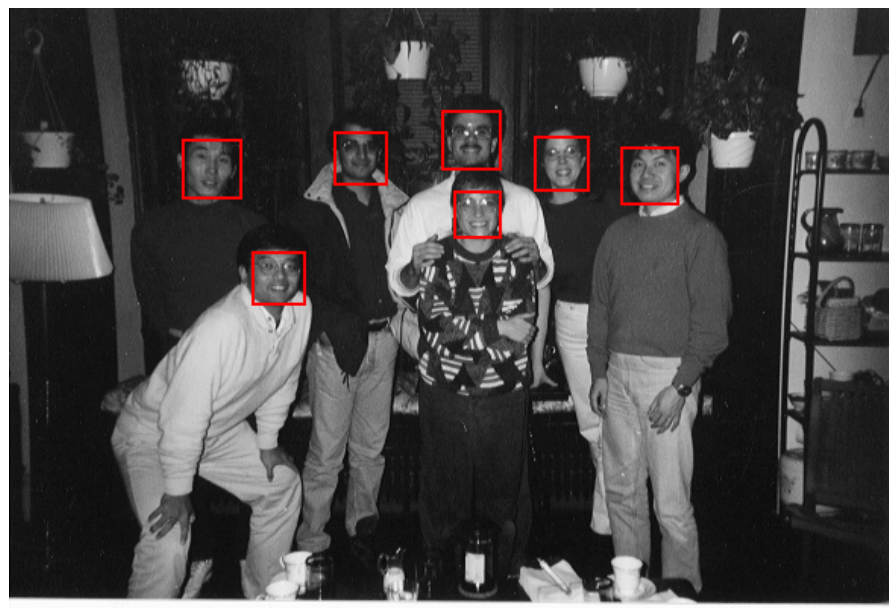
Roughly 15 times faster than Rowley-Baluja-Kanade and 600 times faster than Schneiderman-Kanade.

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

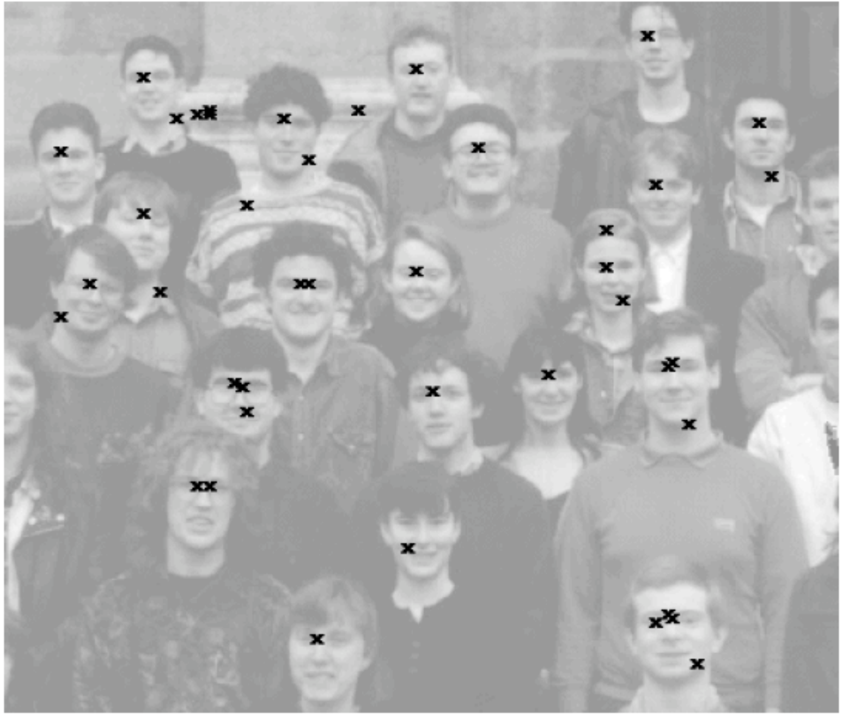
Output of Face Detector on Test Images

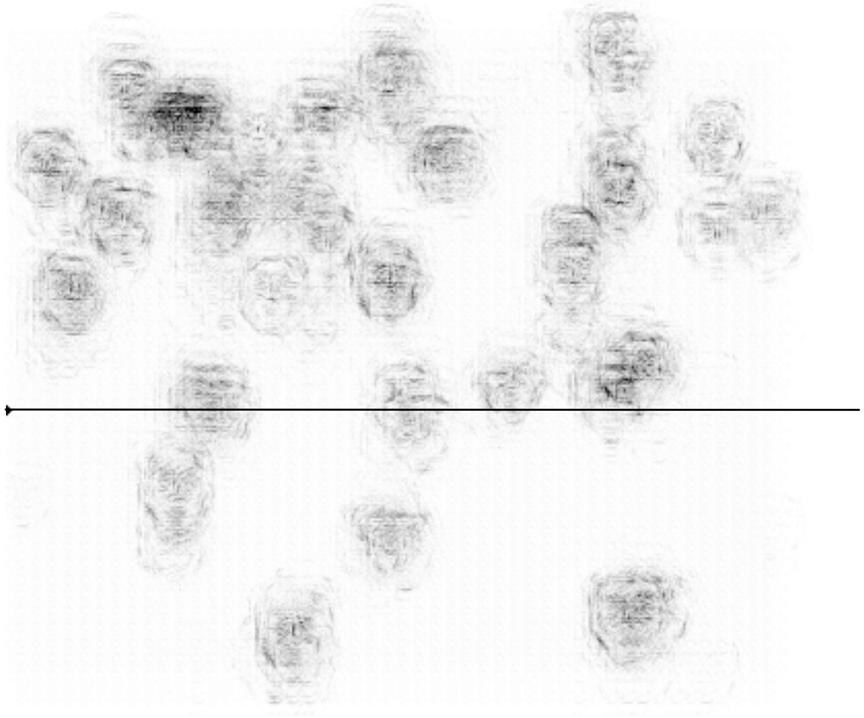


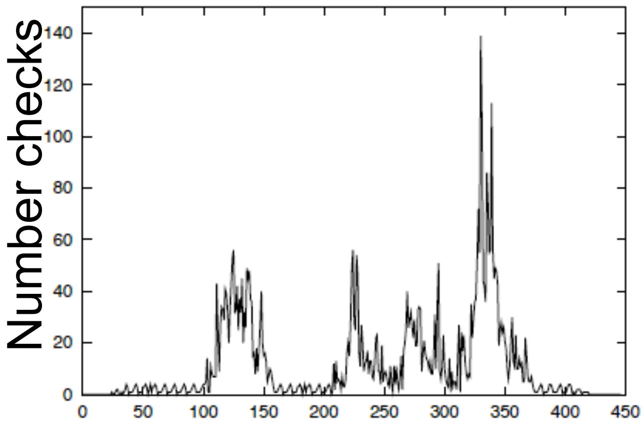




Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001







Fleuret and Geman 2001

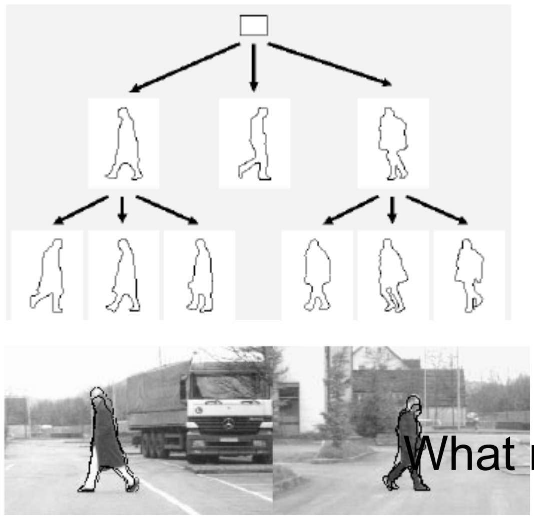
Cascade of classifiers

- Perhaps, enough efficiency can overcome combinatorics…

Fleuret and Geman 2001

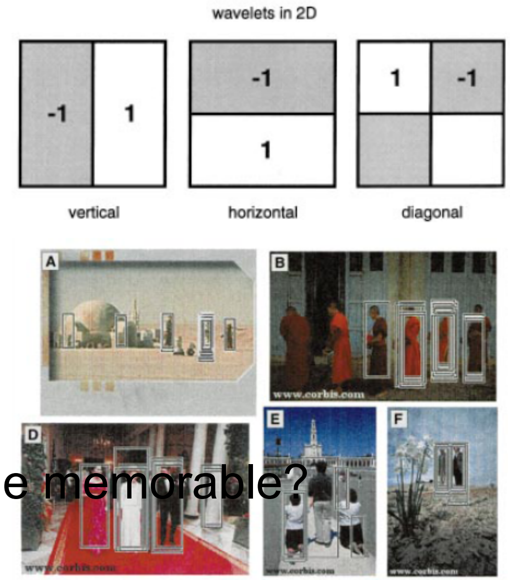
Edge based descriptors

Edge based descriptors

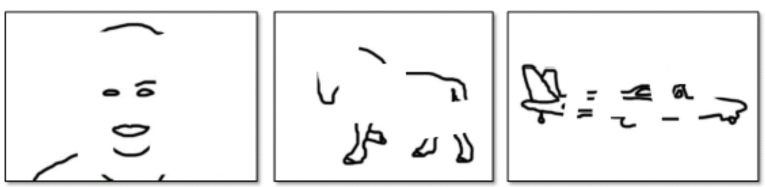


Gavrila, Philomin, ICCV 1999

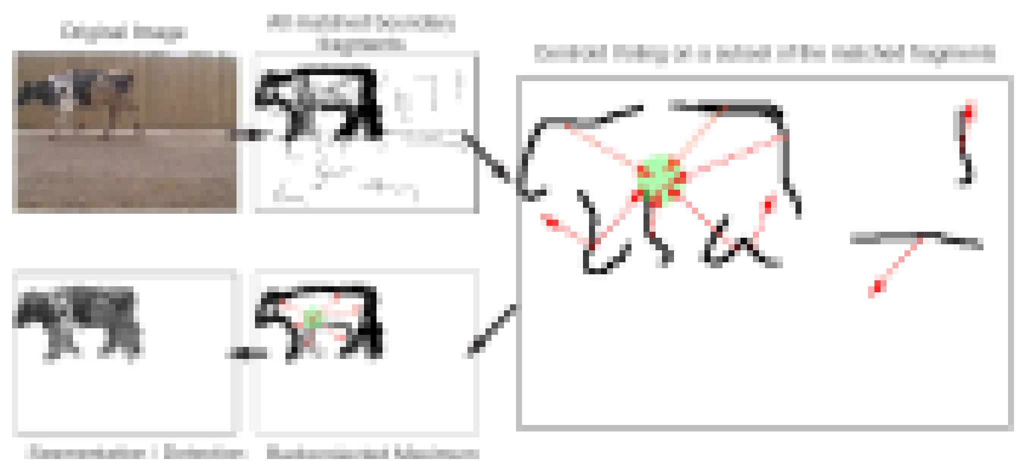
What makes an image memorable?



Papageorgiou & Poggio (2000)

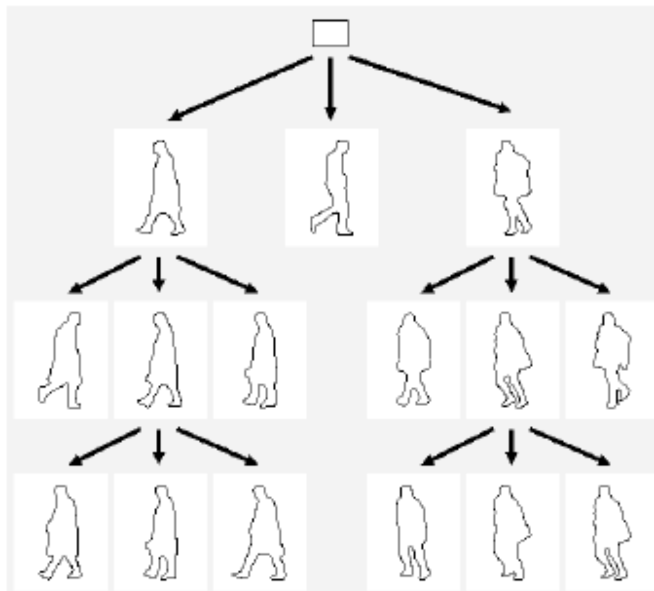


J. Shotton, A. Blake, R. Cipolla. PAMI 2008.



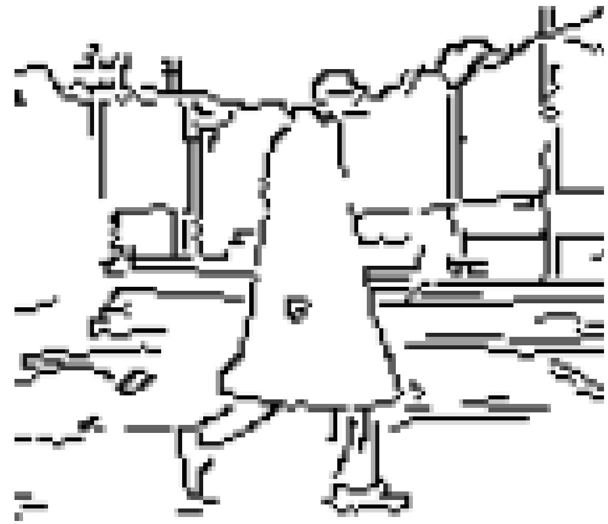
Opelt, Pinz, Zisserman, ECCV 2006

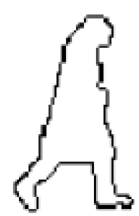
Edges and chamfer distance



Gavrila, Philomin, ICCV 1999

Edges and chamfer distance





Template

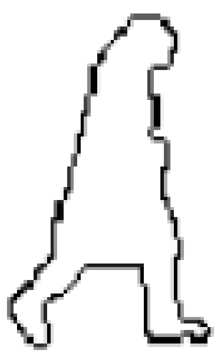
Gavrila, Philomin, ICCV 1999

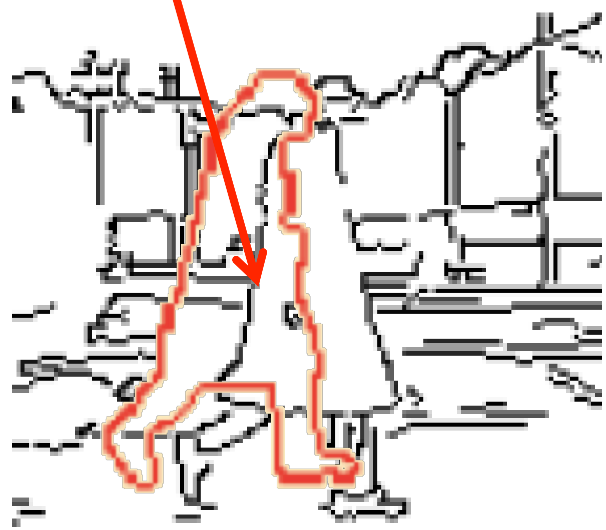
Chamfer distance

$$d_{chamfer}(\mathbf{x}) = \sum_{\mathbf{u} \in F} \min_{\mathbf{v} \in E} \|(\mathbf{u} + \mathbf{x}) - \mathbf{v}\|_2$$

Find closest edge location after displacement x

Sum over pixels on the edge template F



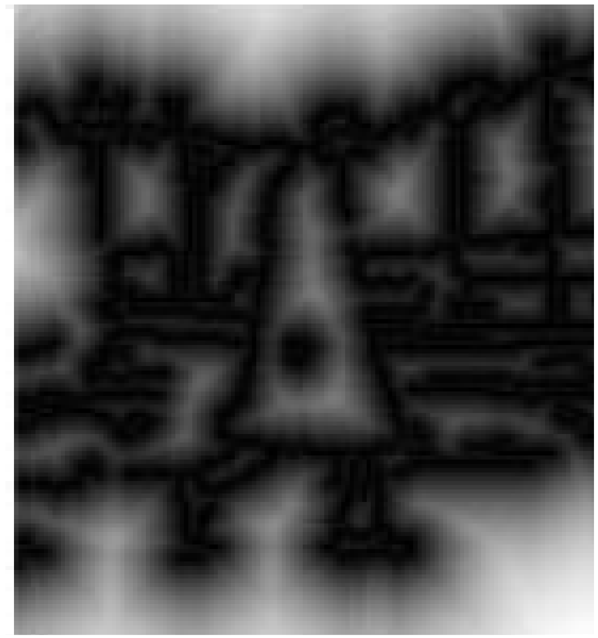


E = edge map of the image

Chamfer distance

Edges

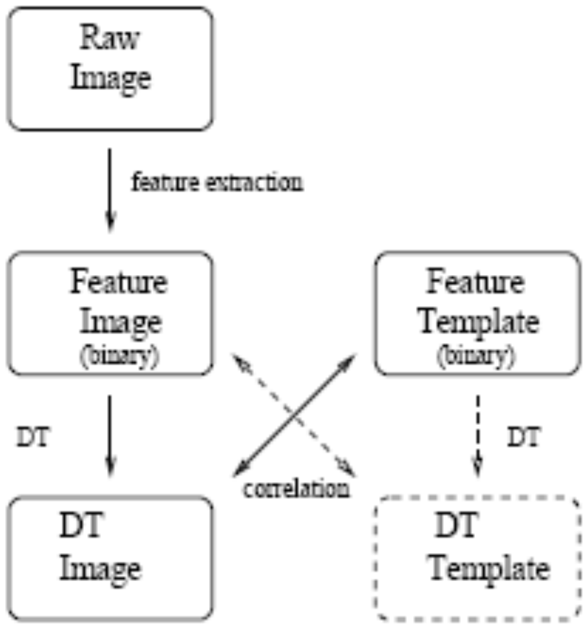
Distance transform

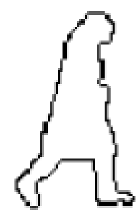


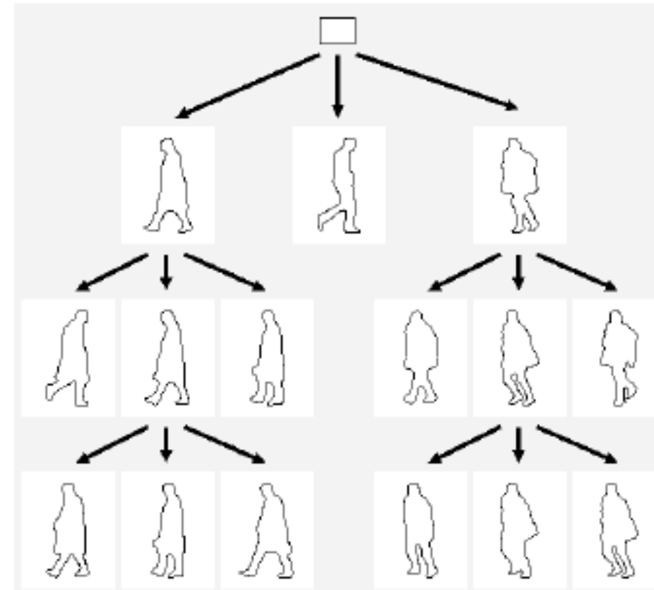
DT(E) = Function that assigns to each pixel the distance to the nearest edge.

Using the distance transform, the Chamfer distance can be written as a convolution

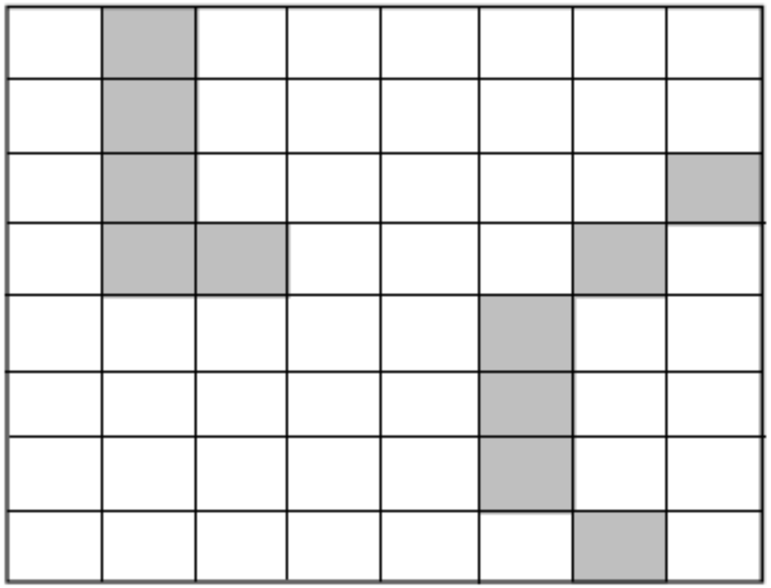
Edges and chamfer distance





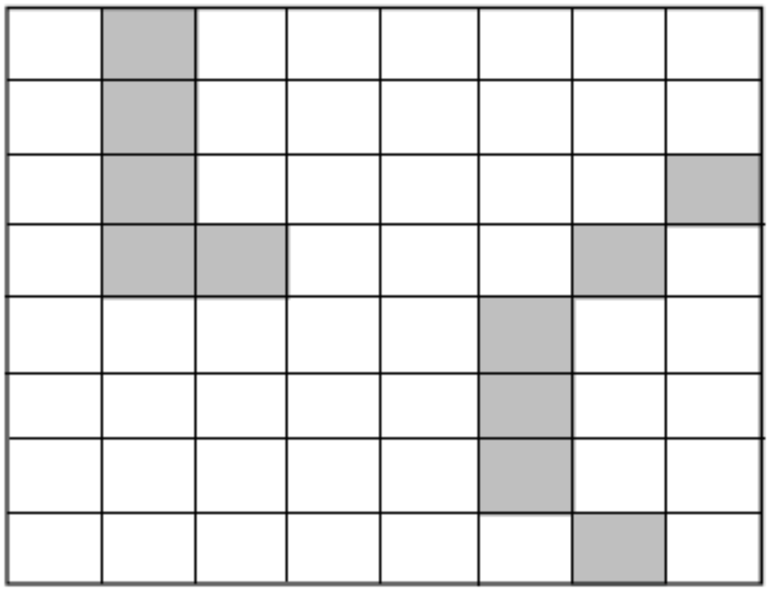


Distance transform



Edges

Distance transform



Edges

1	0	1	2	3	4	3	2
1	0	1	2	3	3	2	1
1	0	1	2	3	2	1	0
1	0	0	1	2	1	0	1
2	1	1	2	1	0	1	2
3	2	2	2	1	0	1	2
4	3	3	2	1	0	1	2
5	4	4	3	2	1	0	1

Distance transform
(with Manhattan distance)

Efficient computation of DT

P = set of edge pixels.

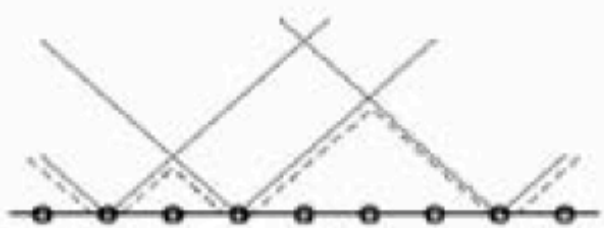
Two pass O(n) algorithm for 1D L_1 norm

1. Initialize: For all j
   $D[j] \leftarrow 1_P[j]$ // 0 if j is in **P**, infinity otherwise
2. Forward: For j from 1 up to n-1
   $D[j] \leftarrow \min(D[j], D[j-1]+1)$

1	0

3. Backward: For j from n-2 down to 0
   $D[j] \leftarrow \min(D[j], D[j+1]+1)$

0	1



∞	0	∞	0	∞	∞	∞	0	∞
∞	0	1	0	1	2	3	0	1
1	0	1	0	1	2	1	0	1

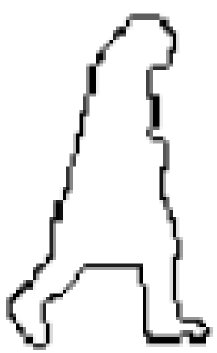
Adapted from D. Huttenlocher

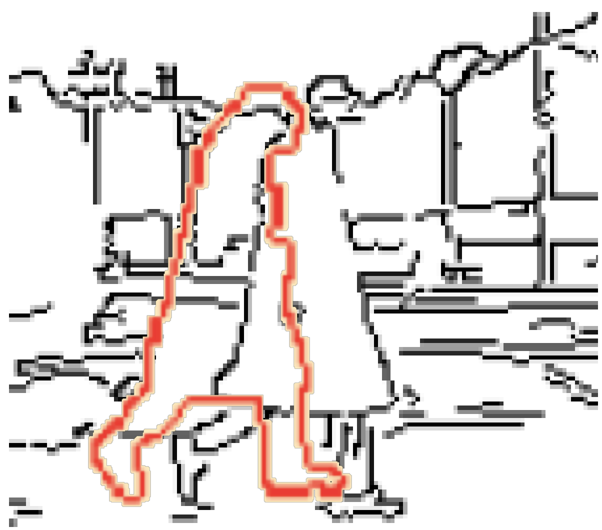
Chamfer distance

$$d_{chamfer}(\mathbf{x}) = \sum_{\mathbf{u} \in F} \min_{\mathbf{v} \in E} \|(\mathbf{u} + \mathbf{x}) - \mathbf{v}\|_2 \quad = F \; * \; DT(E)$$

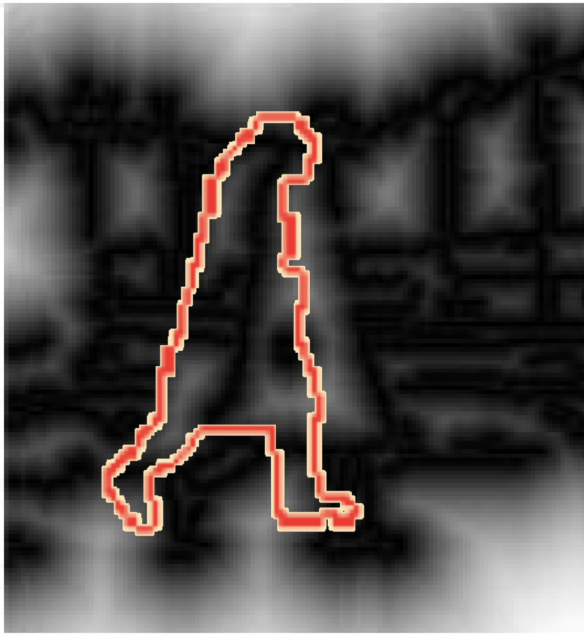
Find closest edge location after displacement x

Sum over pixels on the edge template F





E = edge map of the image



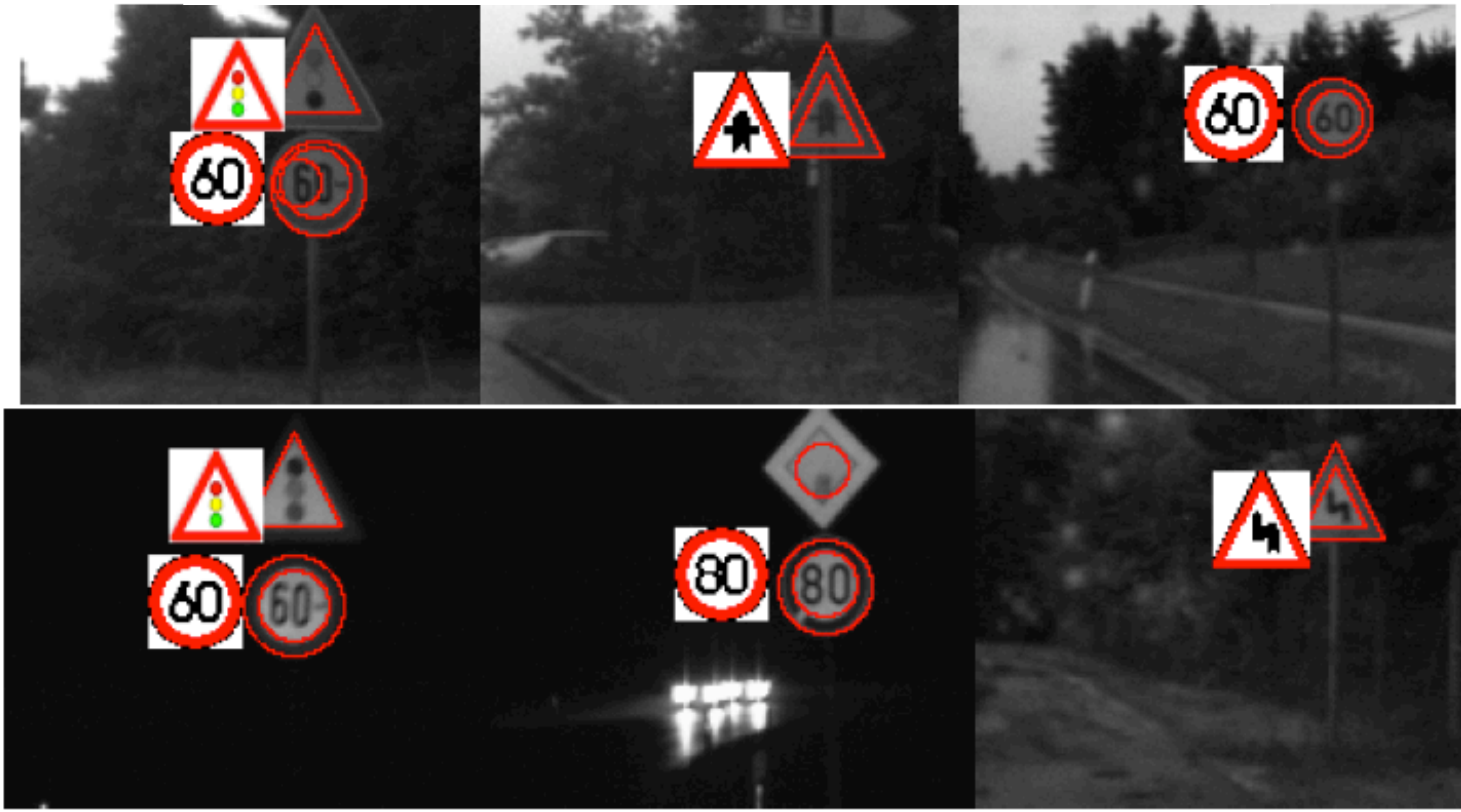
REAL-TIME OBJECT DETECTION FOR "SMART" VEHICLES

D.M. Gavrila

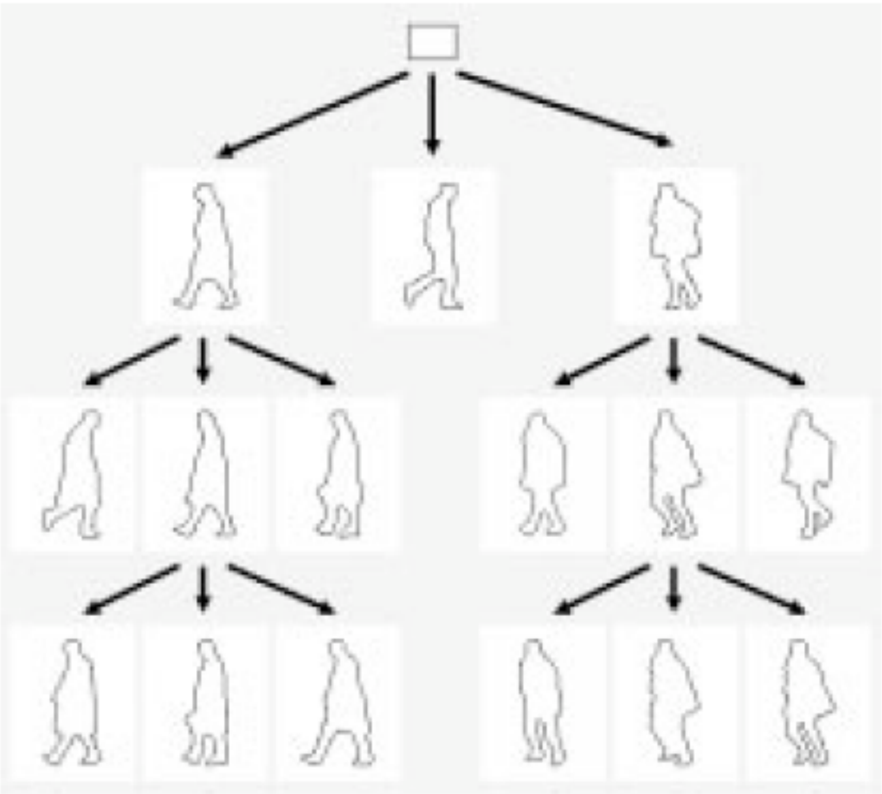
Image Understanding Systems
DaimlerChrysler Research
Ulm 89081, Germany
dariu.gavrila@DaimlerChrysler.com

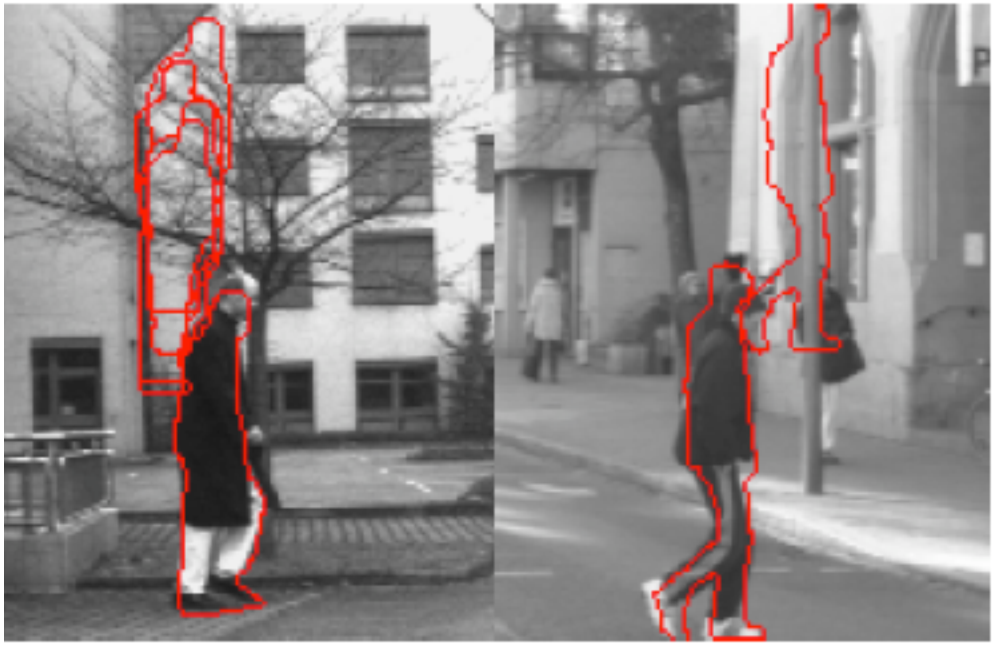
V. Philomin

Computer Vision Laboratory
University of Maryland
College Park, MD 20742, U.S.A.
vasi@cs.umd.edu



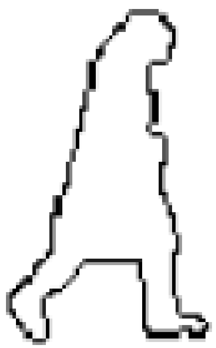
To deal with multiple appearances…



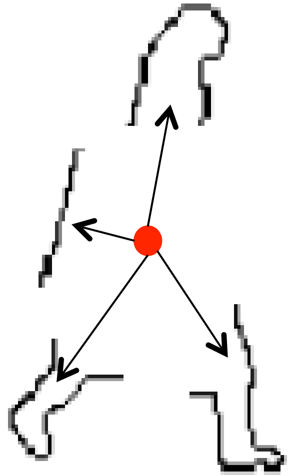


Issues

Global templates are sensitive to:
- Partial occlusions
- Non-rigid deformations



Constellation of local edge fragments



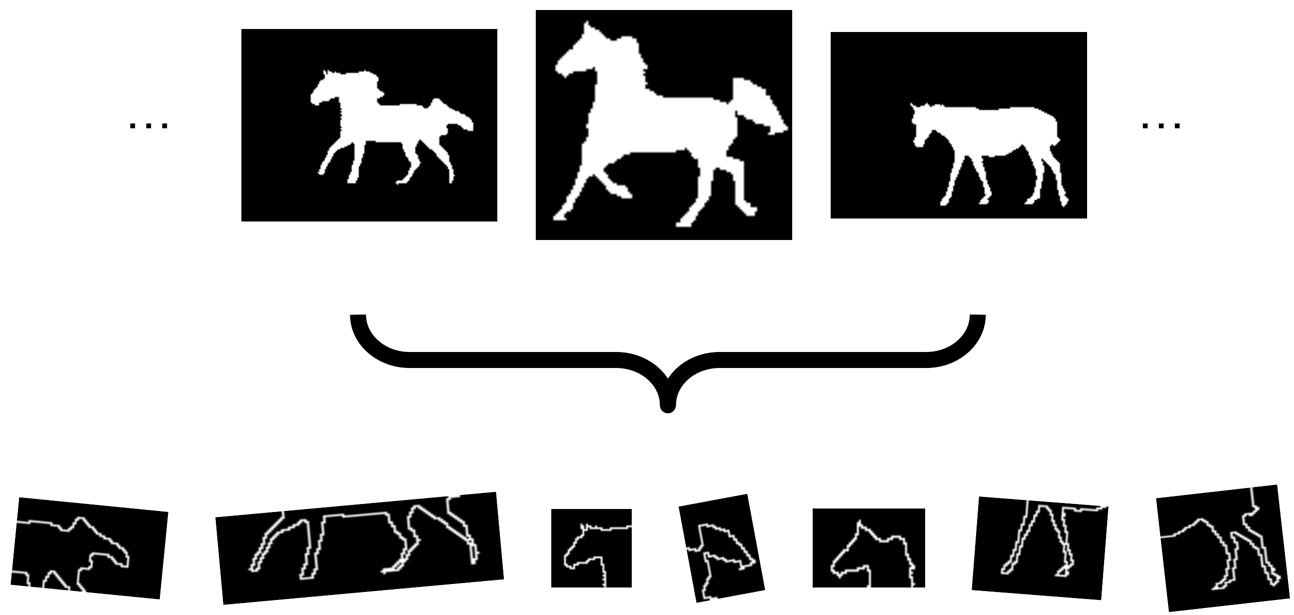
Building a Fragment Dictionary

Masks …

(~10 images)

Contour Fragments T_n …

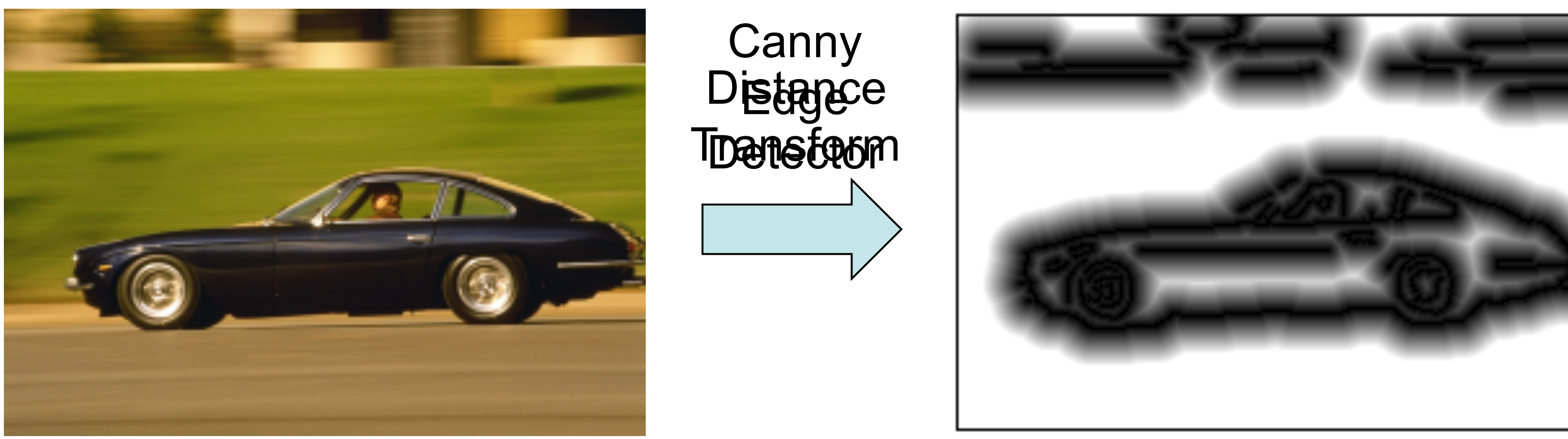
(~1000 fragments)



…

Matching Features

- Gaussian weighted oriented chamfer matching
  - aligns features to image

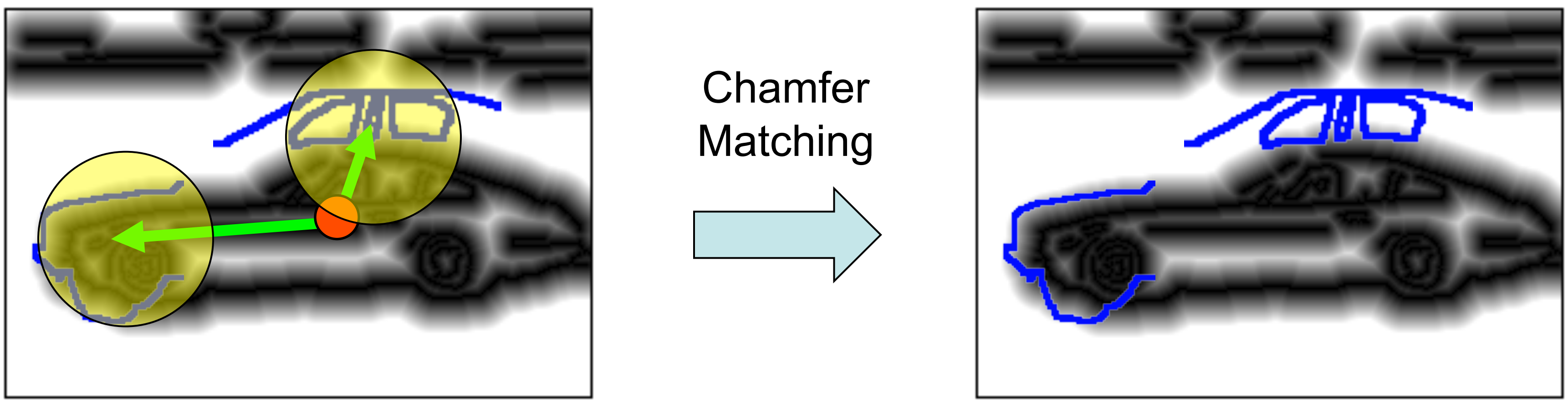


Opelt, Pinz, Zisserman, ECCV 2006

J. Shotton, A. Blake, R. Cipolla. PAMI 2008.

Matching Features

- Gaussian weighted oriented chamfer matching
  - aligns features to image



$v(F_m, E|c)$ feature **match score** at optimal position

$r(F_m, E|c)$ optimal **position**

Location Sensitive Classification

- Feature match scores make detection simple
- Detection uses a boosted classification function $K(\mathbf{c})$:

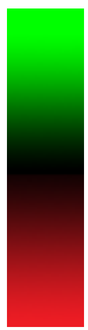
$$K(\mathbf{c}) = \sum_{m=1}^{M} a_m \delta(v(F_m, E|\mathbf{c}) > \theta_m) + b_m$$

M	number of features
F_m	feature m
E	canny edge map
$\mathbf{c}$	object centroid

θ_m	weak learner threshold
a_m	weak learner confidence
b_m	weak learner confidence
δ	0-1 indicator function

Object Detection

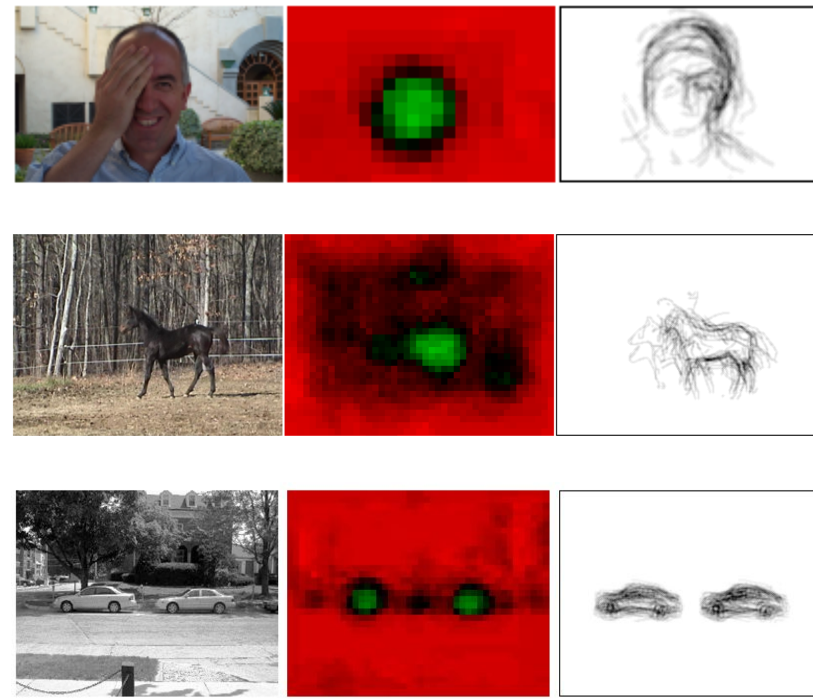
- Evaluate $K(\mathbf{c})$ for all $\mathbf{c}$ gives a *classification map*
  - confidence as function of position



object

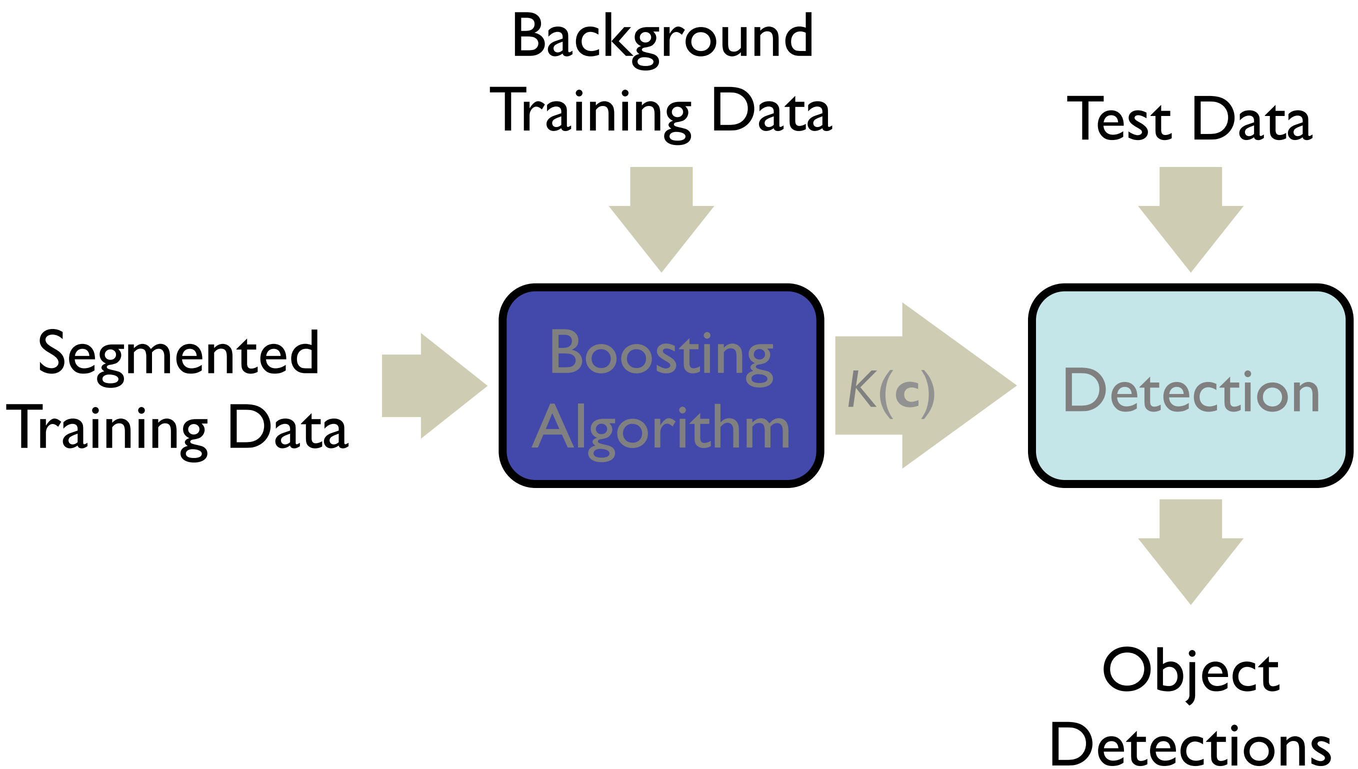
no object

- Globally thresholded local maxima give final detections

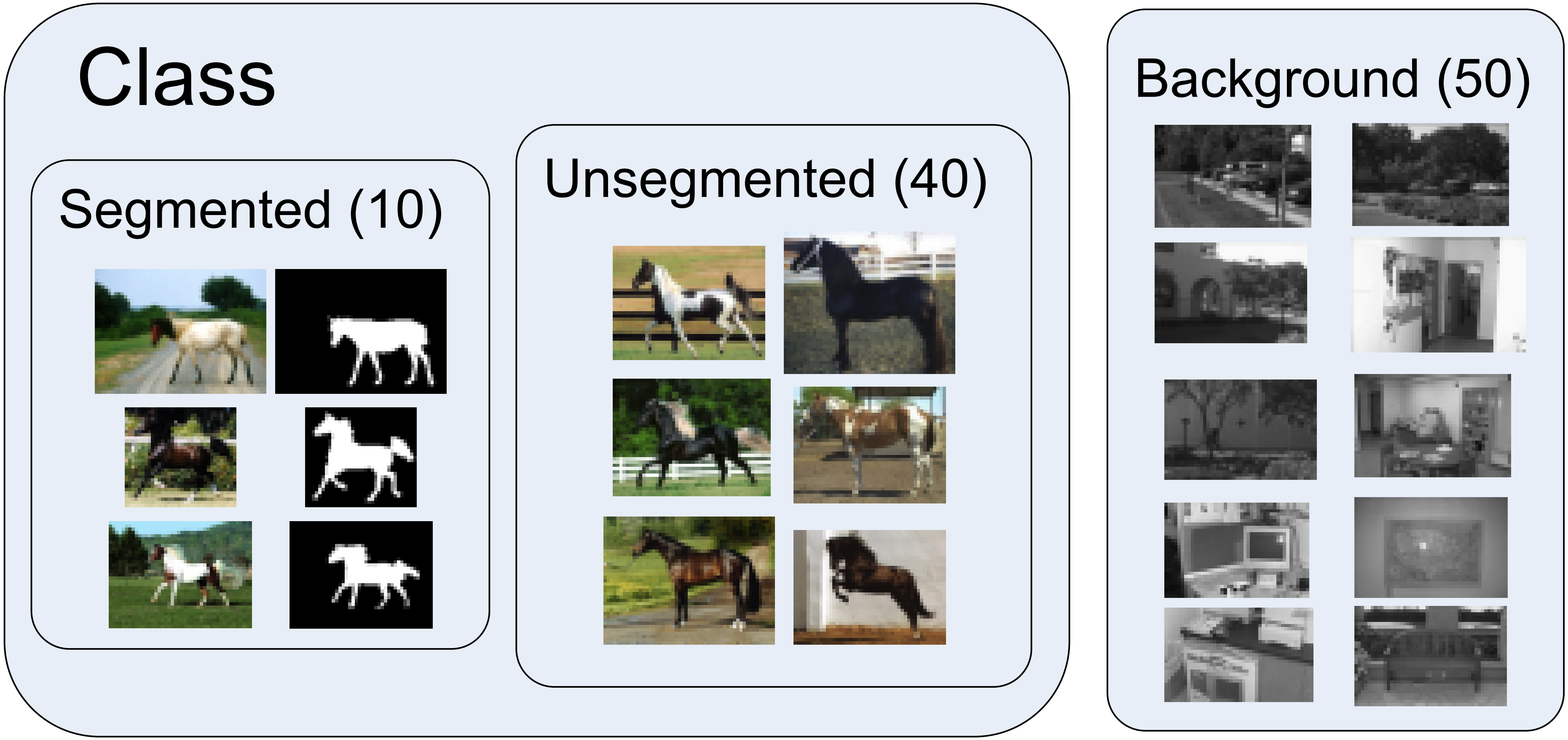


test image | classification map | contours

Learning System

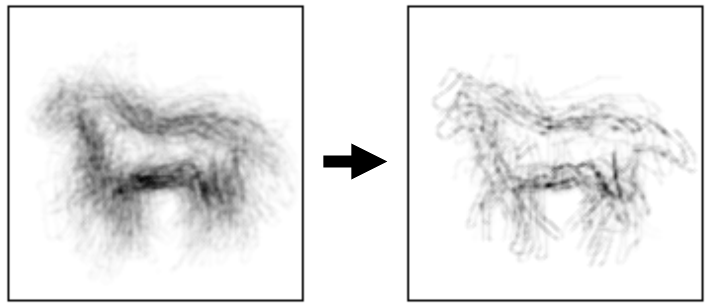


Training Data



Boosting as Feature Selection

1. Fragment Selection



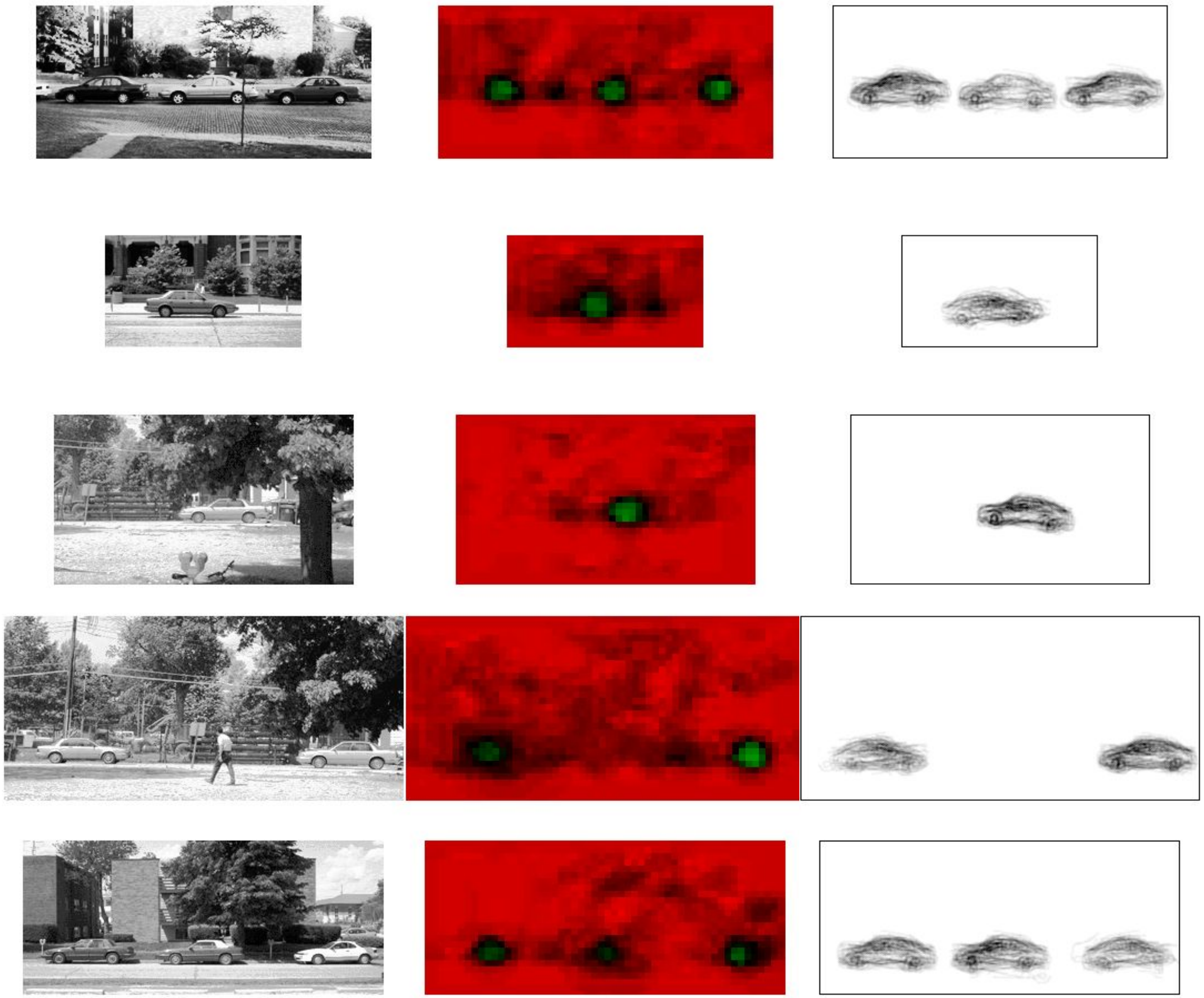
1000 random fragments

50 discriminative fragments

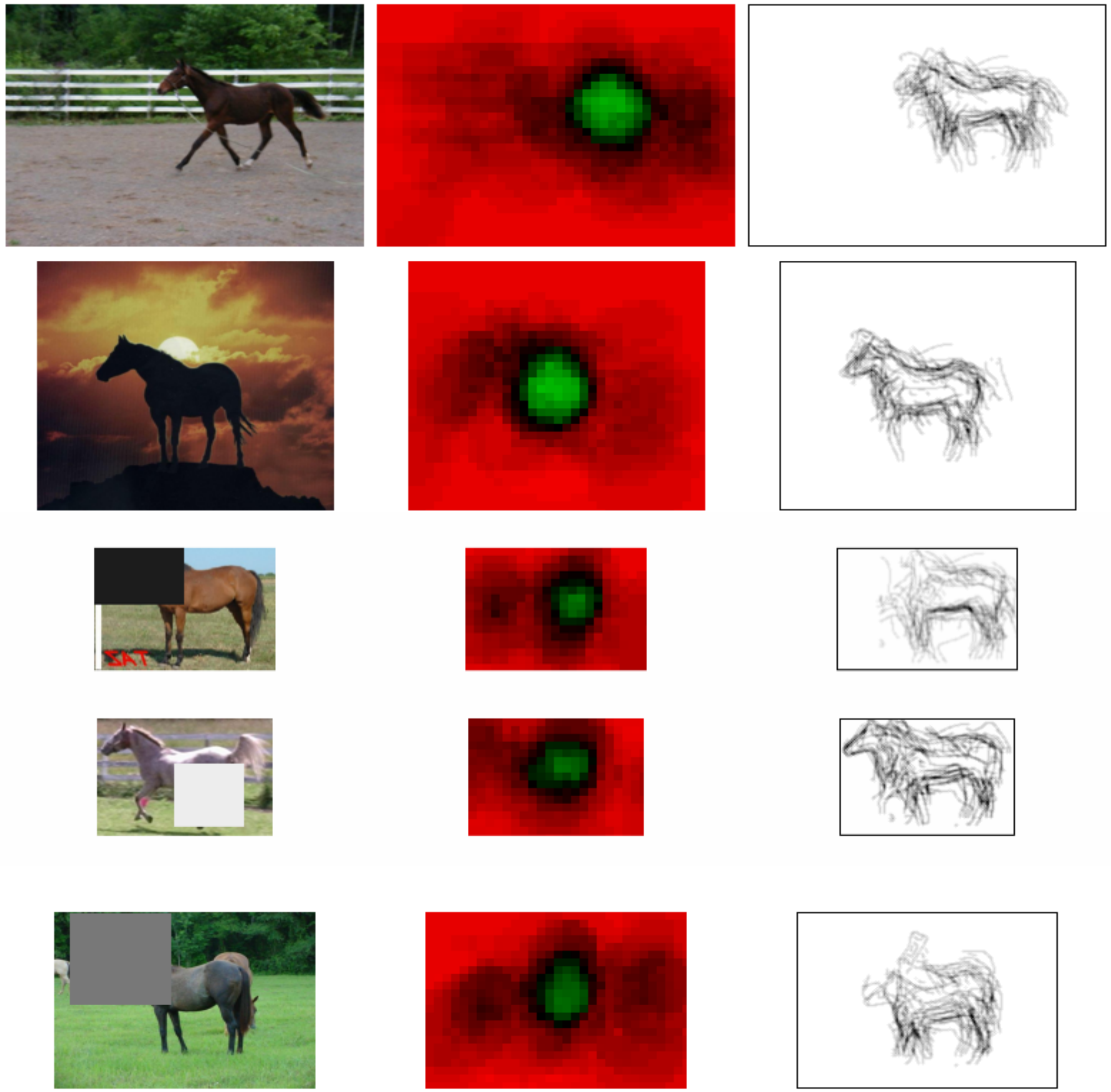
2. Model Parameter Estimation
   Select σ, λ for each feature

3. Weak-Learner Estimation
   Select θ, *a*, *b* for each feature

Contour Results



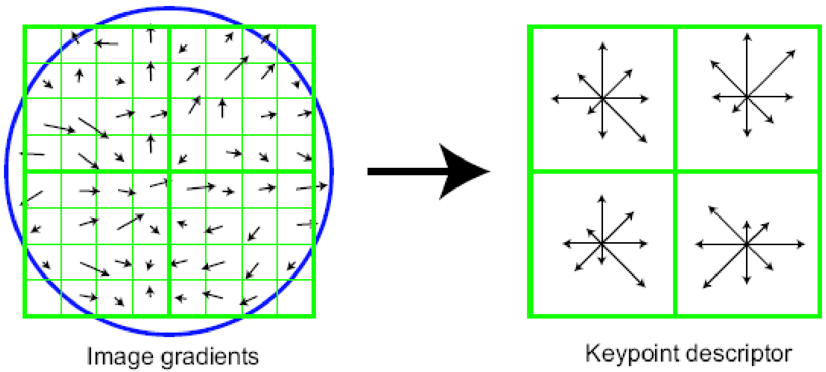
Contour Results



Histograms of oriented gradients

Histograms of oriented gradients

SIFT, D. Lowe, ICCV 1999



Shape context

Belongie, Malik, Puzicha, NIPS 2000

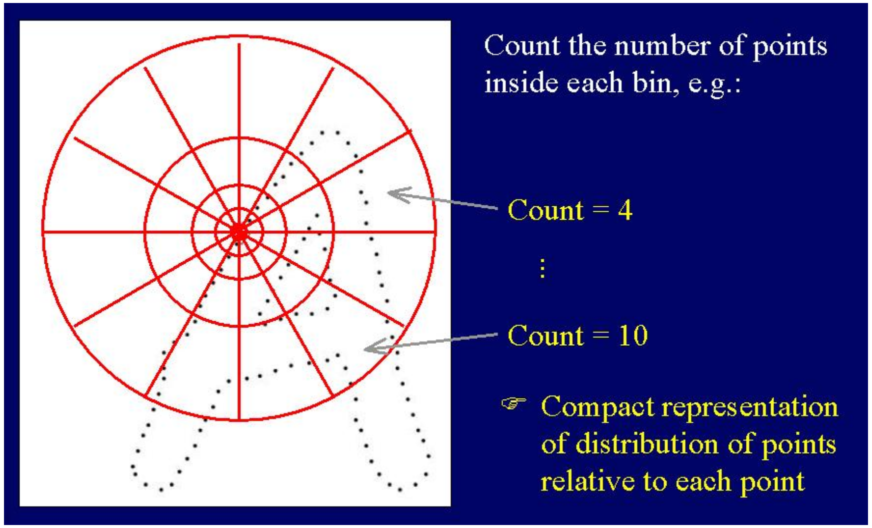
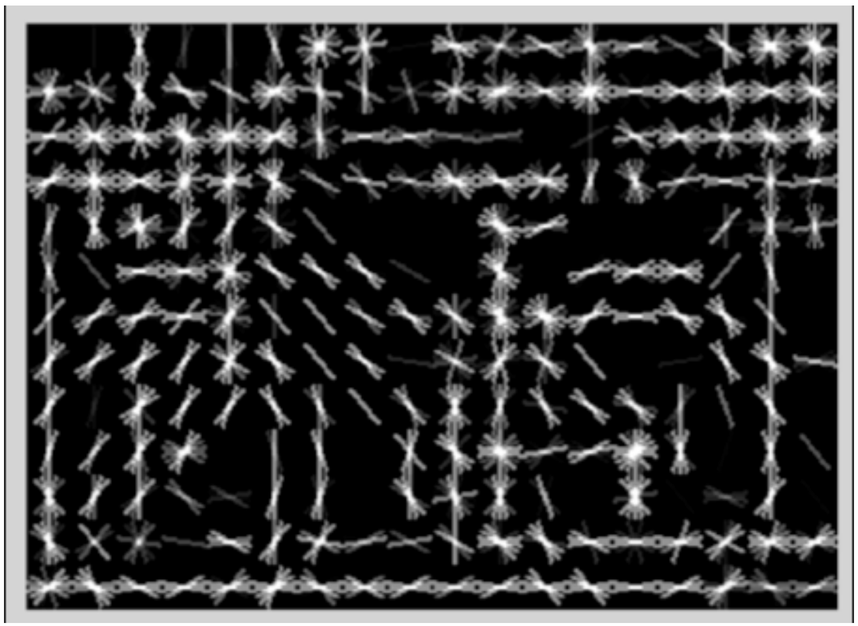


Image features:

Histograms of oriented gradients (HOG)



Bin gradients from 8x8 pixel neighborhoods into 9 orientations

(Dalal & Triggs CVPR 05)

Source: Deva Ramanan

Histograms of Oriented Gradients for Human Detection

Navneet Dalal and Bill Triggs
INRIA Rhône-Alps, 655 avenue de l'Europe, Montbonnot 38334, France
{Navneet.Dalal,Bill.Triggs}@inrialpes.fr, http://lear.inrialpes.fr

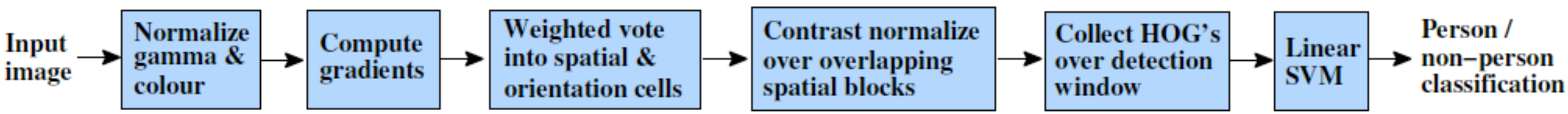


Figure 1. An overview of our feature extraction and object detection chain. The detector window is tiled with a grid of overlapping blocks in which Histogram of Oriented Gradient feature vectors are extracted. The combined vectors are fed to a linear SVM for object/non-object classification. The detection window is scanned across the image at all positions and scales, and conventional non-maximum suppression is run on the output pyramid to detect object instances, but this paper concentrates on the feature extraction process.

Histograms of Oriented Gradients for Human Detection

Navneet Dalal and Bill Triggs

INRIA Rhône-Alps, 655 avenue de l'Europe, Montbonnot 38334, France

{Navneet.Dalal,Bill.Triggs}@inrialpes.fr, http://lear.inrialpes.fr

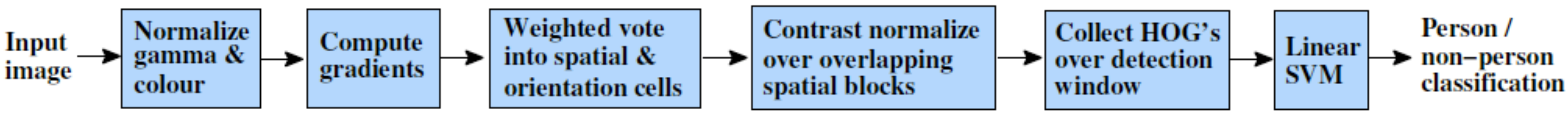
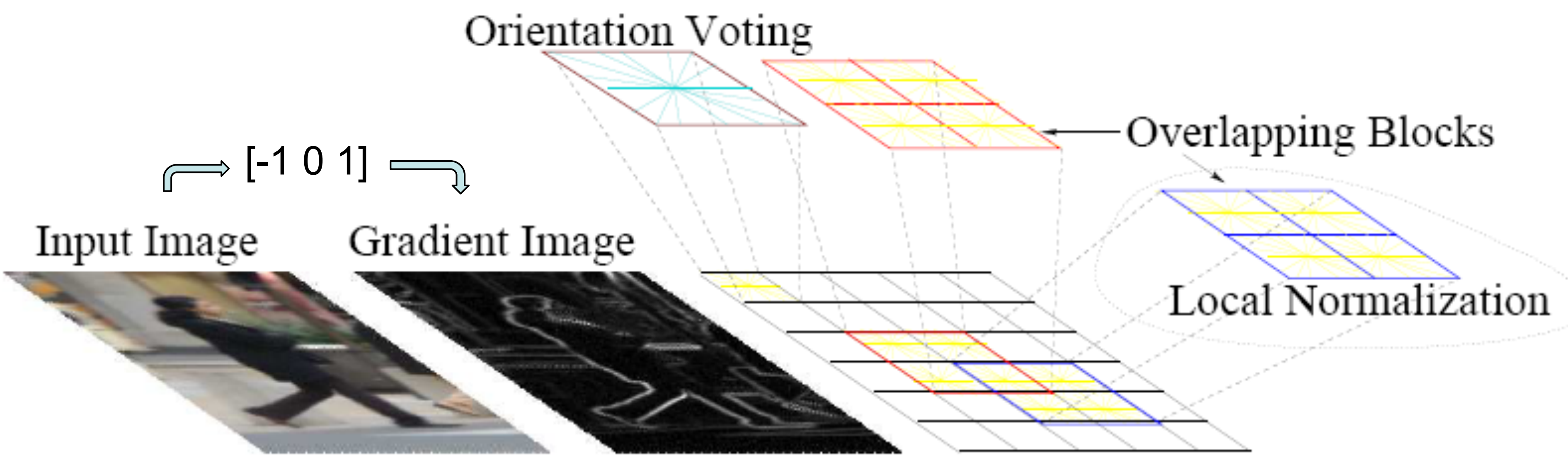
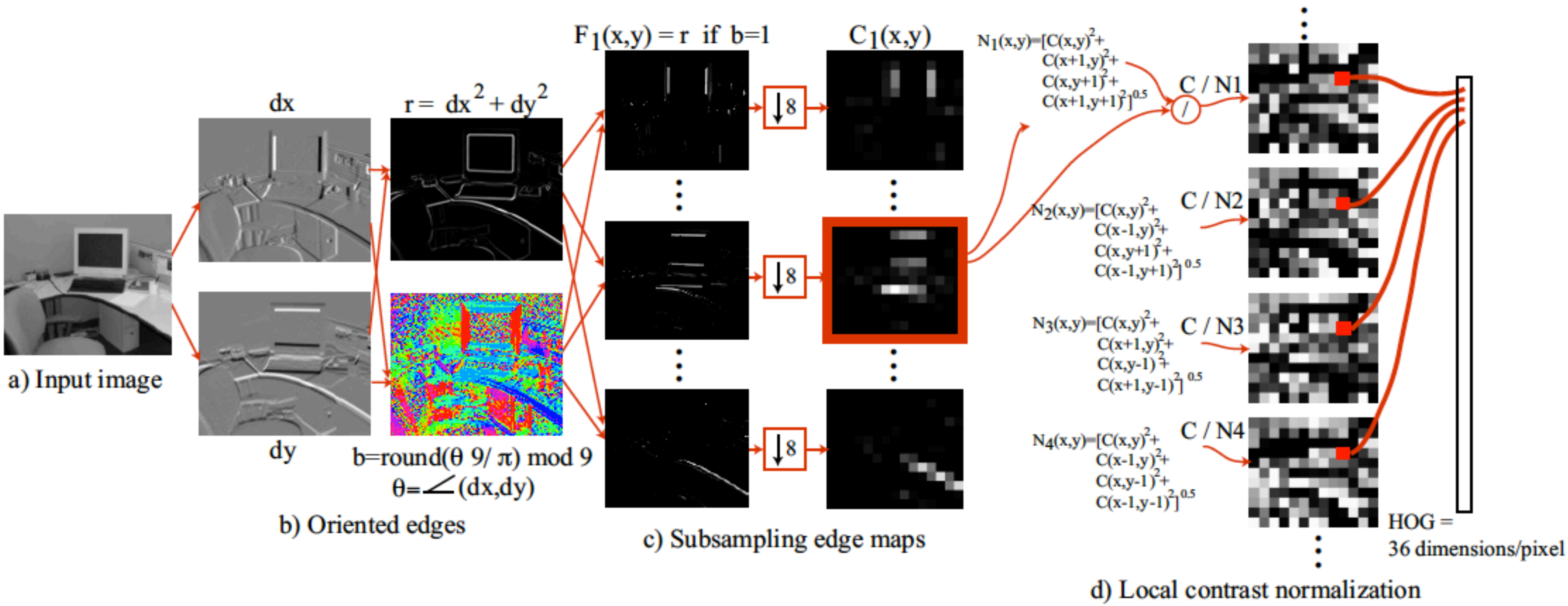


Figure 1. An overview of our feature extraction and object detection chain. The detector window is tiled with a grid of overlapping blocks in which Histogram of Oriented Gradient feature vectors are extracted. The combined vectors are fed to a linear SVM for object/non-object classification. The detection window is scanned across the image at all positions and scales, and conventional non-maximum suppression is run on the output pyramid to detect object instances, but this paper concentrates on the feature extraction process.



HOG



a) Input image
dx
dy
r = dx 2 + dy 2
b=round(θ 9/ π) mod 9
θ=∠(dx,dy)
b) Oriented edges
F 1(x,y) = r if b=1
8
8
8
C1(x,y)
c) Subsampling edge maps
N1(x,y)=[C(x,y)2+ C(x+1,y)2+ C(x,y+1)2+ C(x+1,y+1)2]0.5
N2(x,y)=[C(x,y)2+ C(x-1,y)2+ C(x,y+1)2+ C(x-1,y+1)2]0.5
N3(x,y)=[C(x,y)2+ C(x+1,y)2+ C(x,y-1)2+ C(x+1,y-1)2]0.5
N4(x,y)=[C(x,y)2+ C(x-1,y)2+ C(x,y-1)2+ C(x-1,y-1)2]0.5
C / N1
C / N2
C / N3
C / N4
/
HOG =
36 dimensions/pixel
d) Local contrast normalization

SVM

A Support Vector Machine (SVM) learns a classifier with the form:

$$H(x) = \sum_{m=1}^{M} a_m y_m k(x, x_m)$$

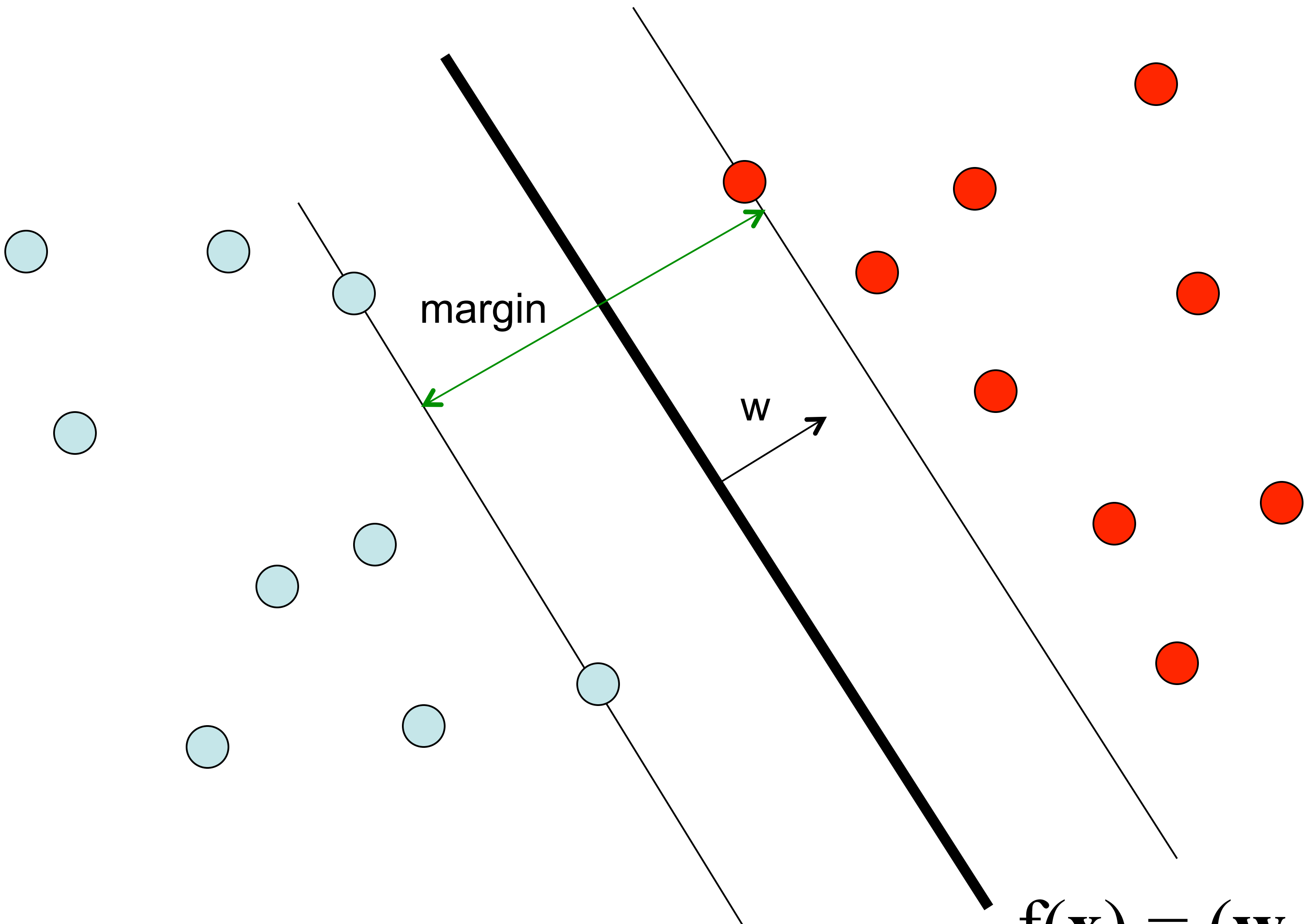
Where $\{x_m, y_m\}$, for m = 1 . . .M, are the training data with x_m being the input feature vector and y_m = +1,-1 the class label. $k(x, x_m)$ is the kernel and it can be any symmetric function satisfying the Mercer Theorem.

The classification is obtained by thresholding the value of H(x).

There is a large number of possible kernels, each yielding a different family of decision boundaries:

- Linear kernel: $k(x, x_m) = x^T x_m$
- Radial basis function: $k(x, x_m) = \exp(-|x - x_m|^2/\sigma^2)$.
- Histogram intersection: $k(x,x_m) = \text{sum}_i(\min(x(i), x_m(i)))$

Linear SVM

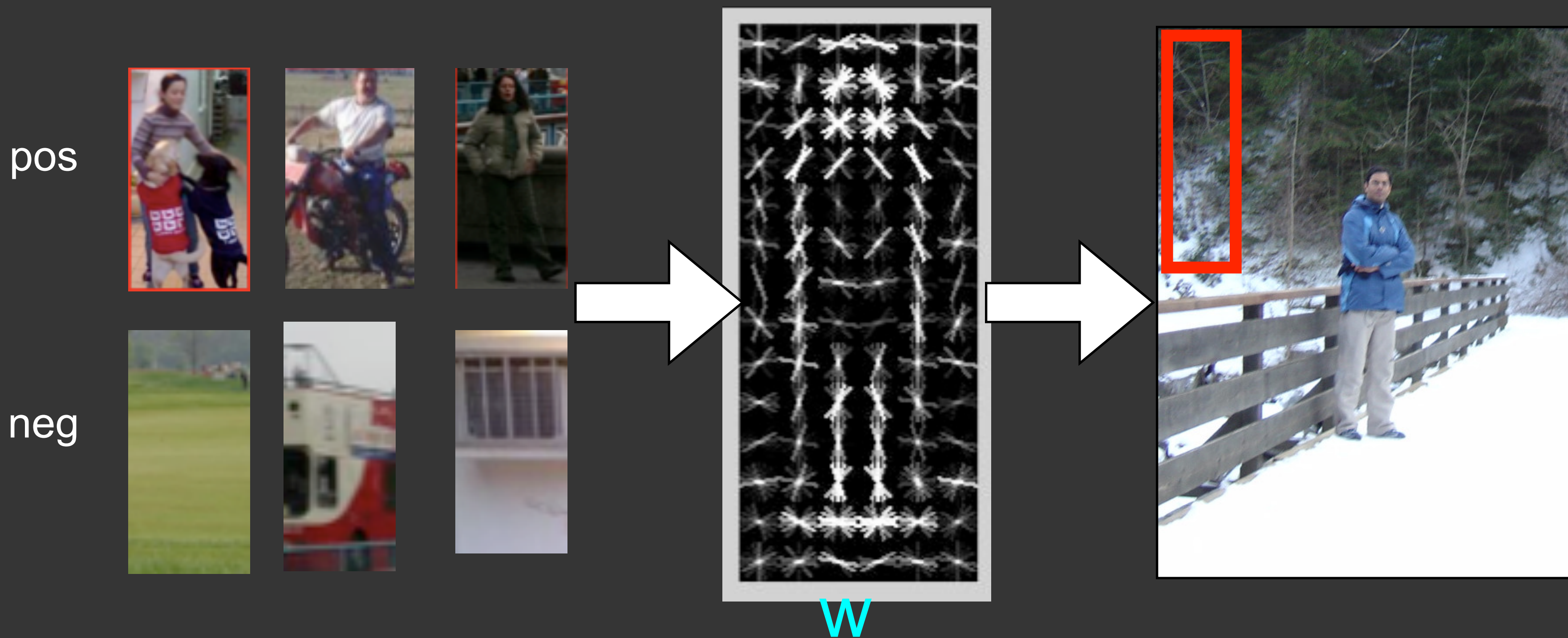


$$f(\mathbf{x}) = (\mathbf{w} \,.\, \mathbf{x} + b)$$

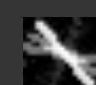
Scanning-window templates

Dalal and Triggs CVPR05 (HOG)

Papageorgiou and Poggio ICIP99 (wavelets)



w = weights for orientation and spatial bins



$w \cdot x > 0$

Train with a linear classifier (perceptron, logistic regression, SVMs...)

Source: Deva Ramanan

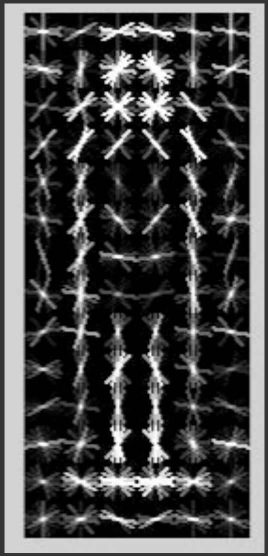
How to interpret positive and negative weights?

$$w \cdot x > 0$$

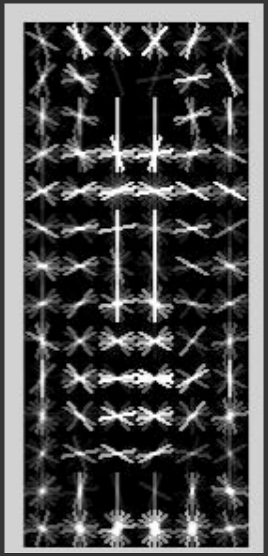
$$(w_{pos} - w_{neg}) \cdot x > 0$$

$$w_{pos} \cdot x > w_{neg} \cdot x$$

Pedestrian template



>



Pedestrian background template

w_{pos}, w_{neg} = weighted average of positive, negative support vectors

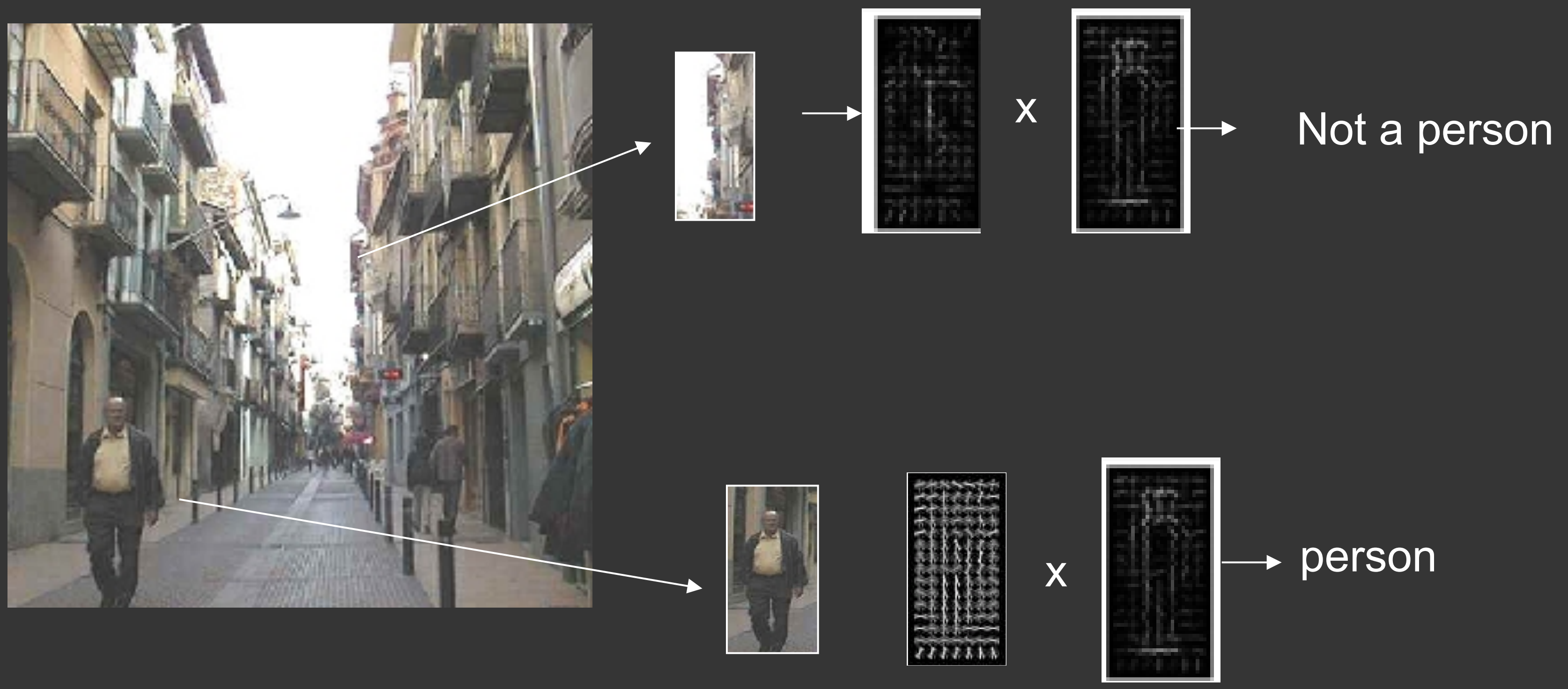
Right approach is to compete pedestrian, pillar, doorway... models

Background class is hard to model - easier to penalize particular vertical edges

Source: Deva Ramanan

Histograms of oriented gradients

Dalal & Trigs, 2006



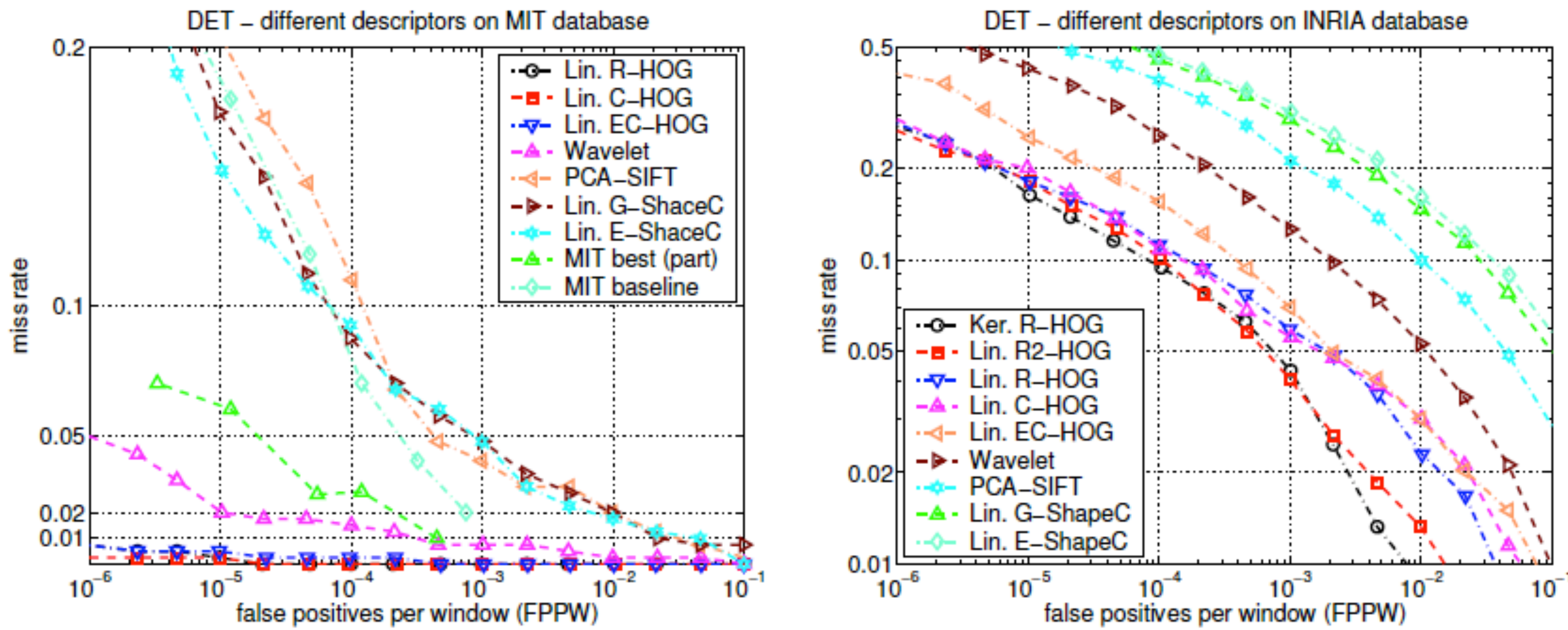
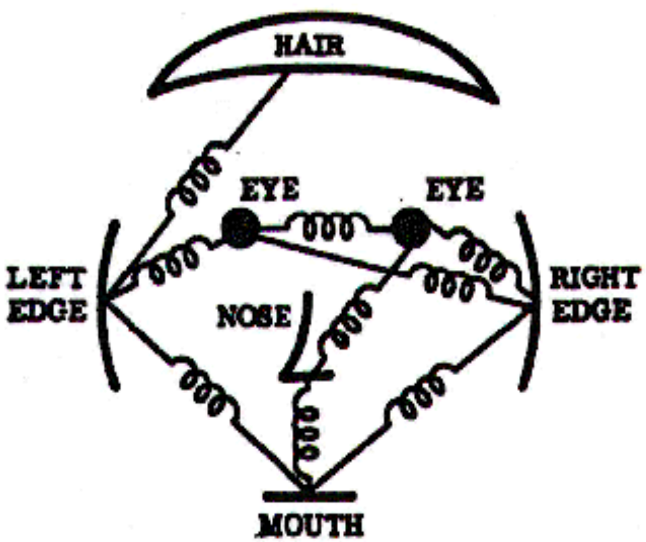


Figure 3. The performance of selected detectors on (left) MIT and (right) INRIA data sets. See the text for details.

Constellation models



Source: short course on object recognition. Fergus, Fei-fei, Torralba

Representation

- Object as set of parts
  - Generative representation
- Model:
  - Relative locations between parts
  - Appearance of part
- Issues:
  - How to model location
  - How to represent appearance
  - Sparse or dense (pixels or regions)
  - How to handle occlusion/clutter

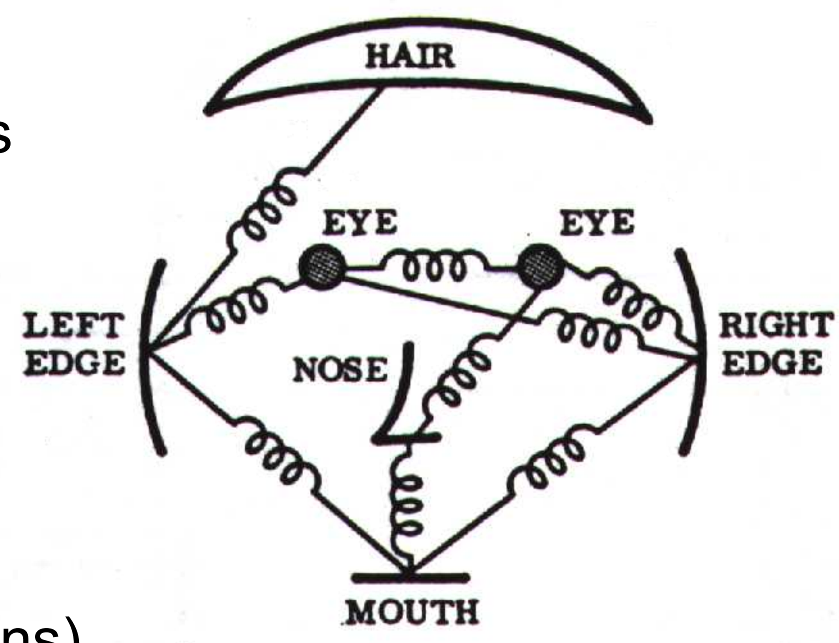
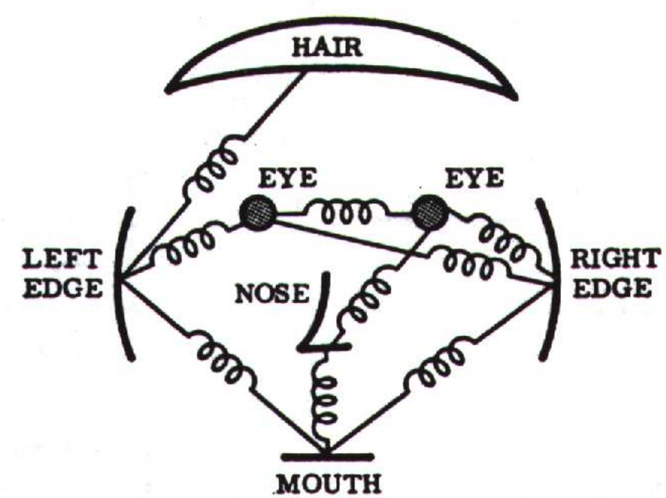


Figure from [Fischler & Elschlager 73]

History of Parts and Structure approaches



- Fischler & Elschlager 1973

- Yuille ‘91
- Brunelli & Poggio ‘93
- Lades, v.d. Malsburg et al. ‘93
- Cootes, Lanitis, Taylor et al. ‘95
- Amit & Geman ‘95, ‘99
- Perona et al. ‘95, ‘96, ’98, ’00, ’03, ‘04, ‘05
- Felzenszwalb & Huttenlocher ’00, ’04
- Crandall & Huttenlocher ’05, ’06
- Leibe & Schiele ’03, ’04

- Many papers since 2000

The Representation and Matching of Pictorial Structures

MARTIN A. FISCHLER AND ROBERT A. ELSCHLAGER

Abstract—The primary problem dealt with in this paper is the following. Given some description of a visual object, find that object in an actual photograph. Part of the solution to this problem is the specification of a descriptive scheme, and a metric on which to base the decision of "goodness" of matching or detection.

We offer a combined descriptive scheme and decision metric which is general, intuitively satisfying, and which has led to promising experimental results. We also present an algorithm which takes the above descriptions, together with a matrix representing the intensities of the actual photograph, and then finds the described object in the matrix. The algorithm uses a procedure similar to dynamic programming in order to cut down on the vast amount of computation otherwise necessary.

One desirable feature of the approach is its generality. A new programming system does not need to be written for every new description; instead, one just specifies descriptions in terms of a certain set of primitives and parameters.

There are many areas of application: scene analysis and description, map matching for navigation and guidance, optical tracking, stereo compilation, and image change detection. In fact, the ability to describe, match, and register scenes is basic for almost any image processing task.

Index Terms—Dynamic programming, heuristic optimization, picture description, picture m
tation.

Manuscript received November 30, 1971; revised May 22, 1972, and August 21, 1972.

The authors are with the Lockheed Palo Alto Research Laboratory, Lockheed Missiles & Space Company, Inc., Palo Alto, Calif. 94304.

INT

THE PRIMARY
paper is the follo
a visual object, fin
graph. The object mig
complicated, such as an
can be linguistic, picto
photograph will be cal
dimensional array of gr
being sought is called t

This ability to find a
equivalently, to match
scenes, is basic for alm
Application to such are
tion, map matching for

Martin A. Fischler (S'57–M'58) was born in New York, N. Y., on February 15, 1932. He received the B.E.E. degree from the City College of New York, New York, in 1954 and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, Calif., in 1958 and 1962, respectively.

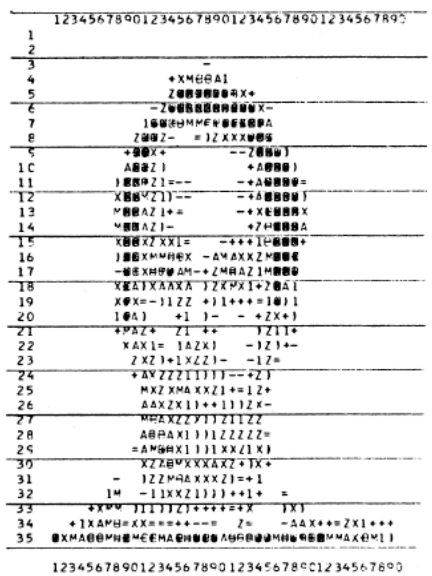
He served in the U. S. Army for two years and held positions at the National Bureau of Standards and at Hughes Aircraft Corporation during the period 1954 to 1958. In 1958 he joined the technical staff of the Lockheed Missiles & Space Company, Inc., at the Lockheed Palo Alto Research Laboratory, Palo Alto, Calif., and currently holds the title of Staff Scientist. He has conducted research and published in the areas of artificial intelligence, picture processing, switching theory, computer organization, and information theory.

Dr. Fischler is a member of the Association for Computing Machinery, the Pattern Recognition Society, the Mathematical Association of America, Tau Beta Pi, and Eta Kappa Nu. He is currently an Associate Editor of the journal *Pattern Recognition* and is a past Chairman of the San Francisco Chapter of the IEEE Society on Systems, Man, and Cybernetics.

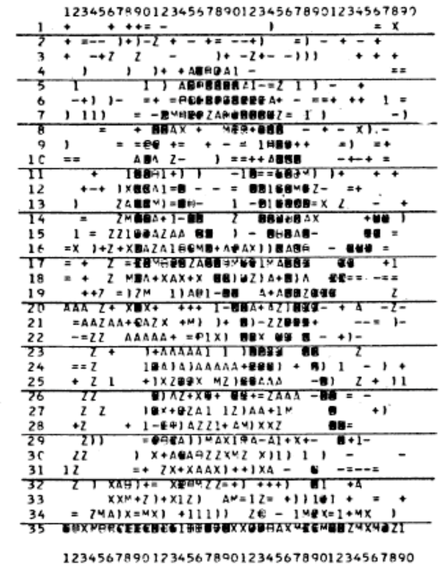
Robert A. Elschlager was born in Chicago, Ill., on May 25, 1943. He received the B.S. degree in mathematics from the University of Illinois, Urbana, in 1964, and the M.S. degree in mathematics from the University of California, Berkeley, in 1969.

Since then he has been an Associate Scientist with the Lockheed Missiles & Space Company, Inc., at the Lockheed Palo Alto Research Center, Palo Alto, Calif. His current interests are picture processing, operating systems, computer languages, and computer understanding.

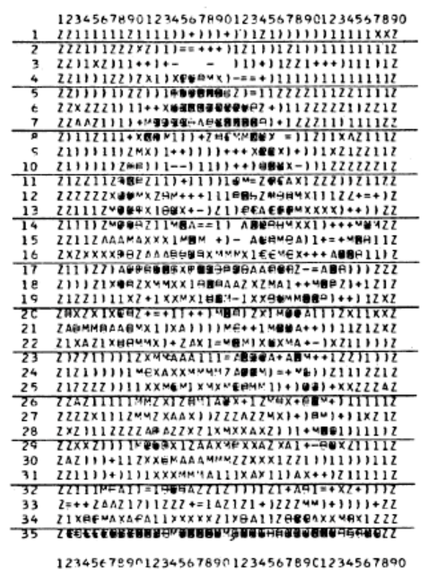
Mr. Elschlager is a member of the American Mathematical Society, the Mathematical Association of America, and the Association for Symbolic Logic.



Original picture.



Noisy picture (sensed scene) as used in experiment.



L(EV)A for eye. (Density at a point is proportional to probability that an eye is present at that location.)

HAIR WAS LOCATED AT (6, 18)
L/EDGE WAS LOCATED AT (18, 10)
R/EDGE WAS LOCATED AT (18, 25)
L/EYE WAS LOCATED AT (17, 13)
R/EYE WAS LOCATED AT (17, 21)
NOSE WAS LOCATED AT (22, 18)
MOUTH WAS LOCATED AT (24, 17)

(a)

Fig. 4. Examples of image-matching experiments using faces. (a) Successful embedding under coherent noise.

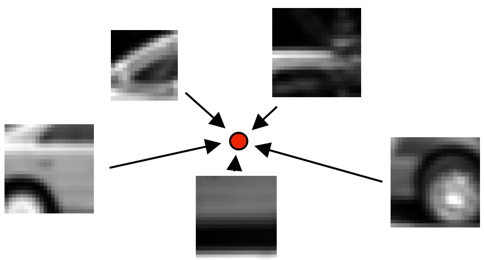
Sparse representation

+ Computationally tractable (10^5 pixels → 10^1 -- 10^2 parts)
+ Generative representation of class
+ Avoid modeling global variability
+ Success in specific object recognition

- Throw away most image information
- Parts need to be distinctive to separate from other classes

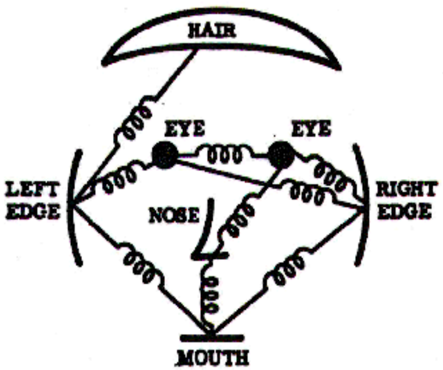
Structure models

Voting models



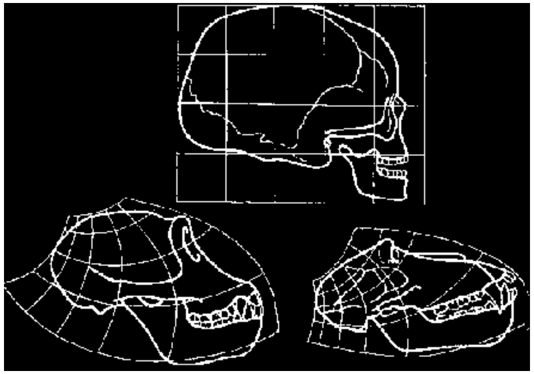
- Many patches (>100)

Constellation models



- Few parts (~6)

Deformable models

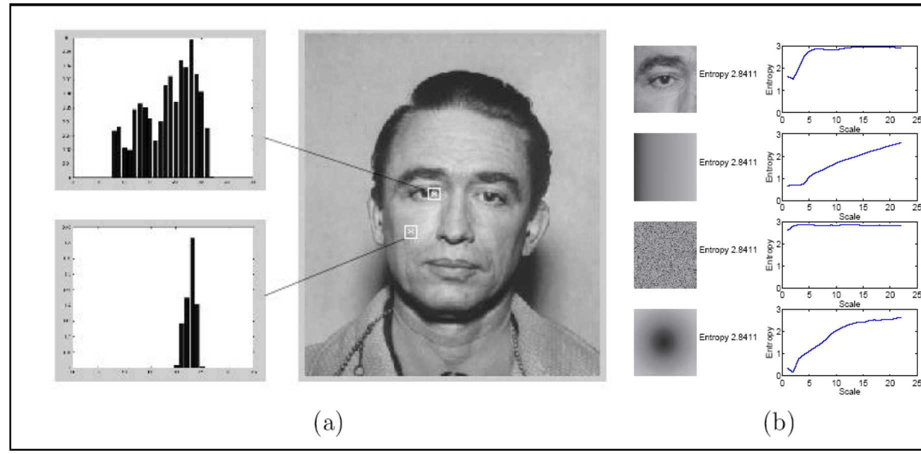


- No parts

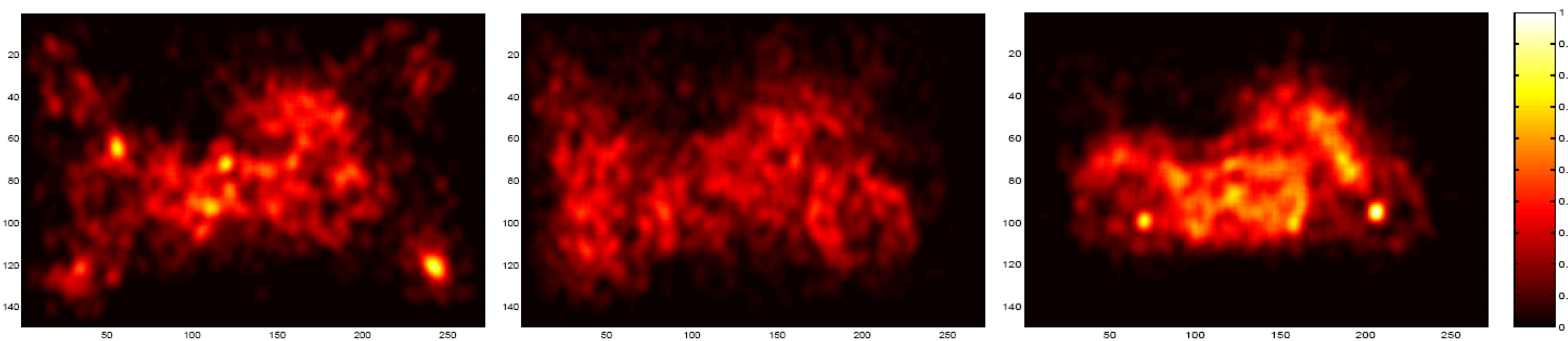
Region operators

– Local maxima of interest operator function

– Can give scale/ orientation invariance



(a) (b)

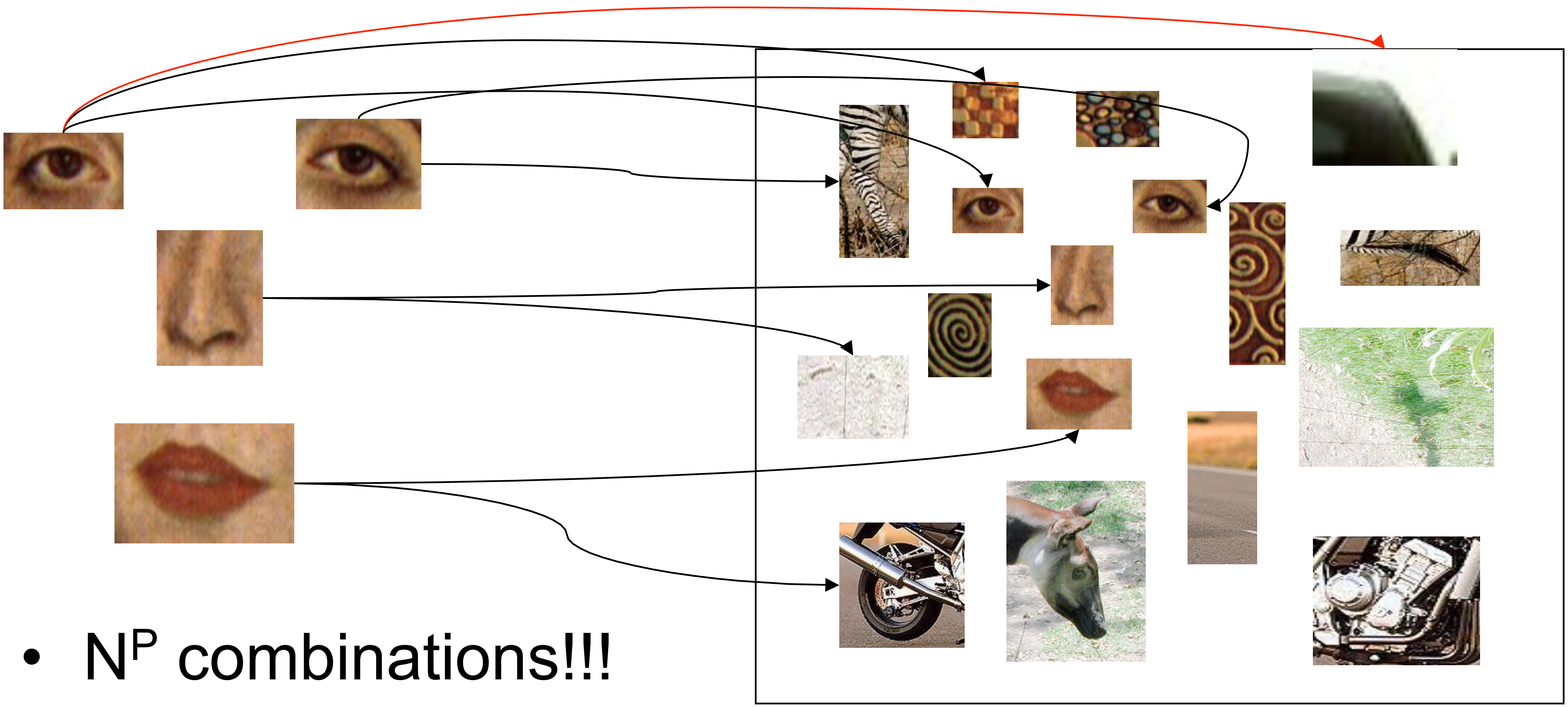


MultiScale Harris   Difference-of-Gaussian   Saliency

Figures from [Kadir, Zisserman and Brady 04]

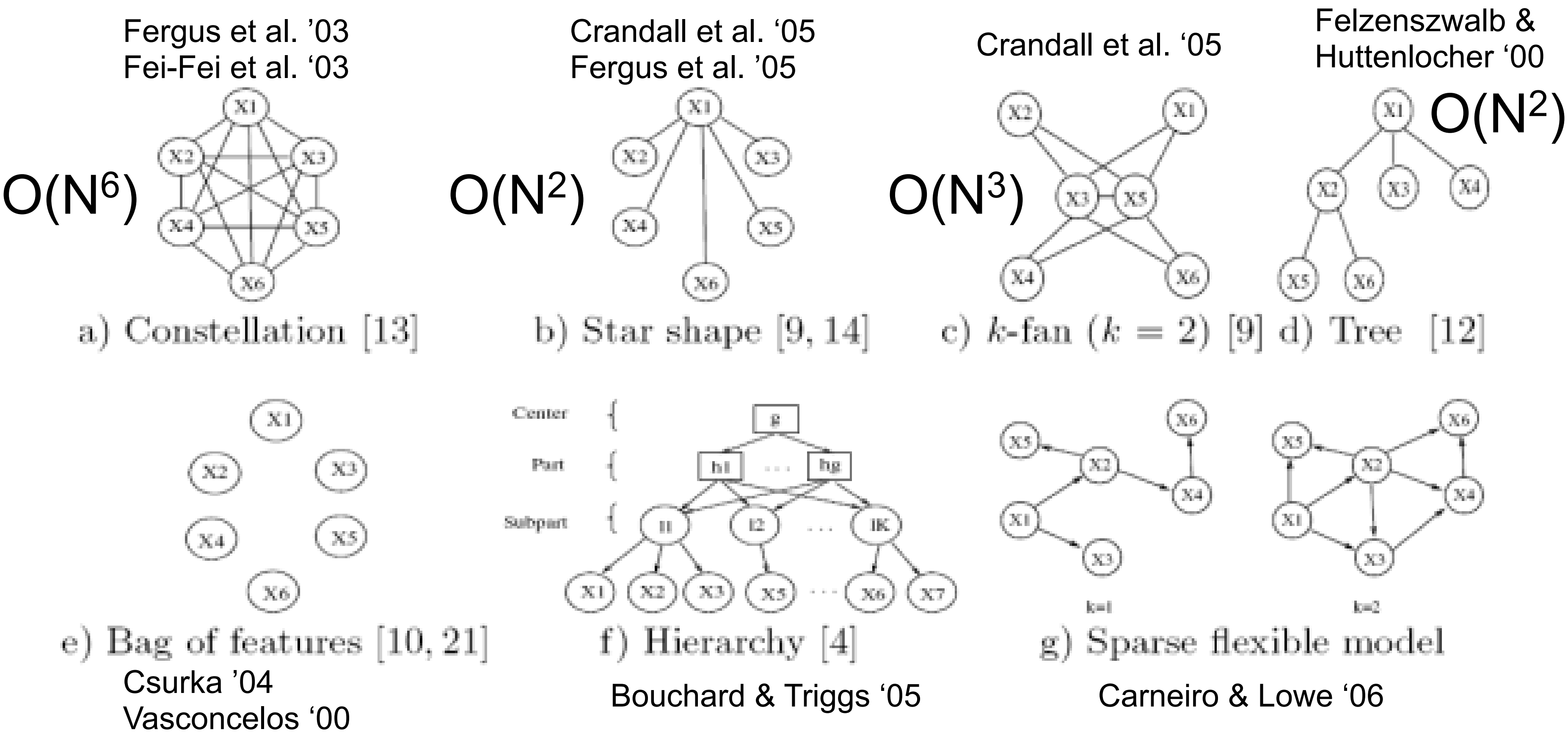
The correspondence problem

- Model with P parts
- Image with N possible assignments for each part
- Consider mapping to be 1-1



- N^P combinations!!!

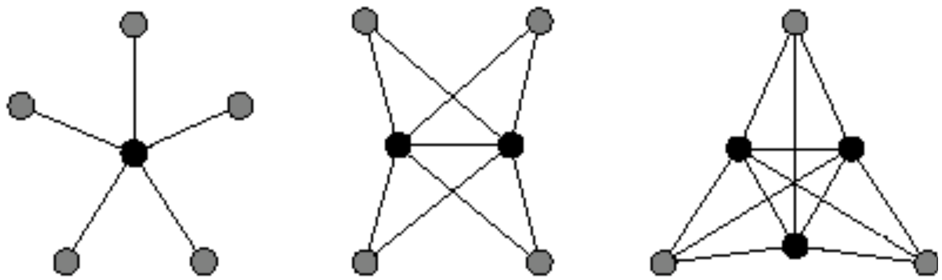
Different connectivity structures

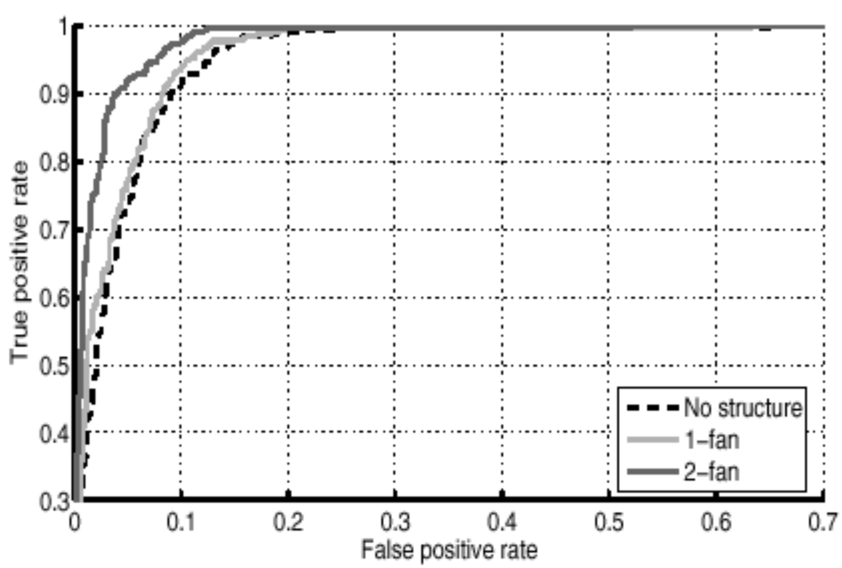


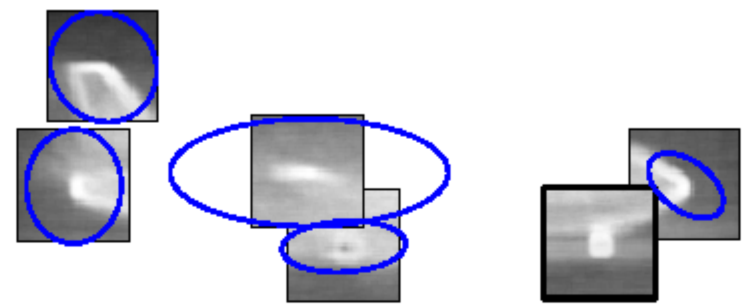
from Sparse Flexible Models of Local Features
Gustavo Carneiro and David Lowe, ECCV 2006

How much does shape help?

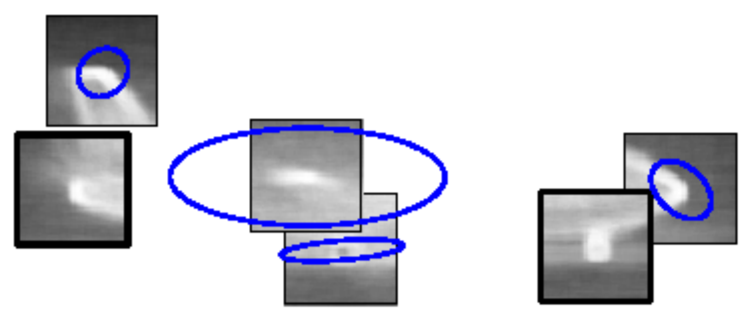
- Crandall, Felzenszwalb, Huttenlocher CVPR'05
- Shape variance increases with increasing model complexity
- Do get some benefit from shape







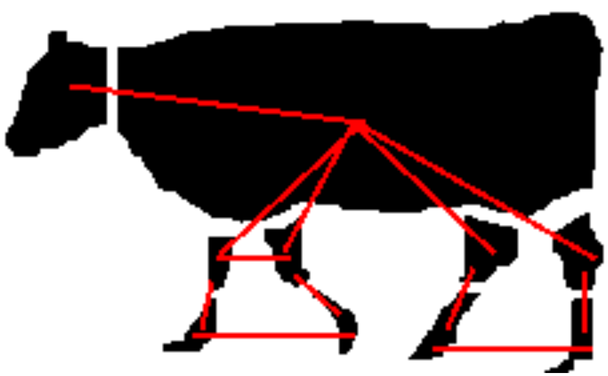
(a) Airplane, 1-fan

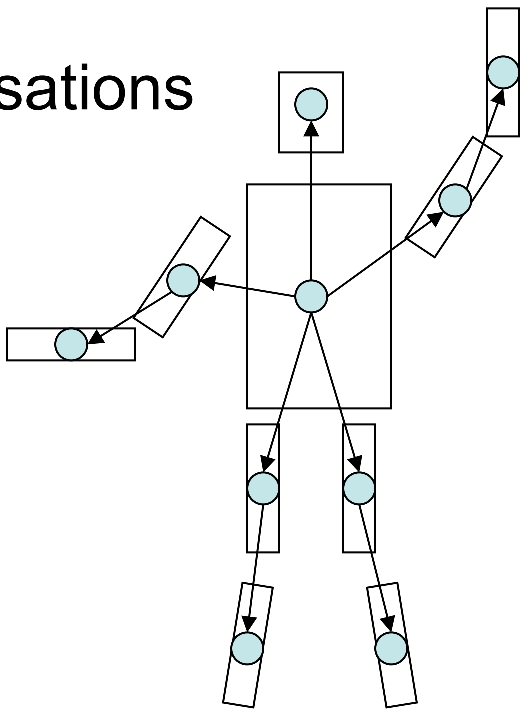


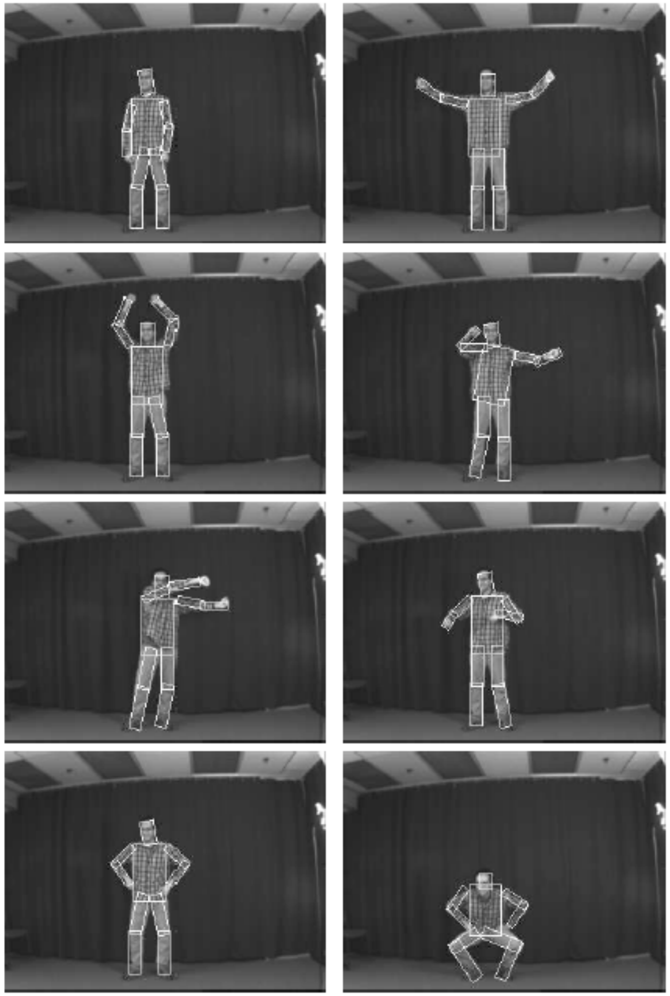
(b) Airplane, 2-fan

Some class-specific graphs

- Articulated motion
  - People
  - Animals
- Special parameterisations
  - Limb angles



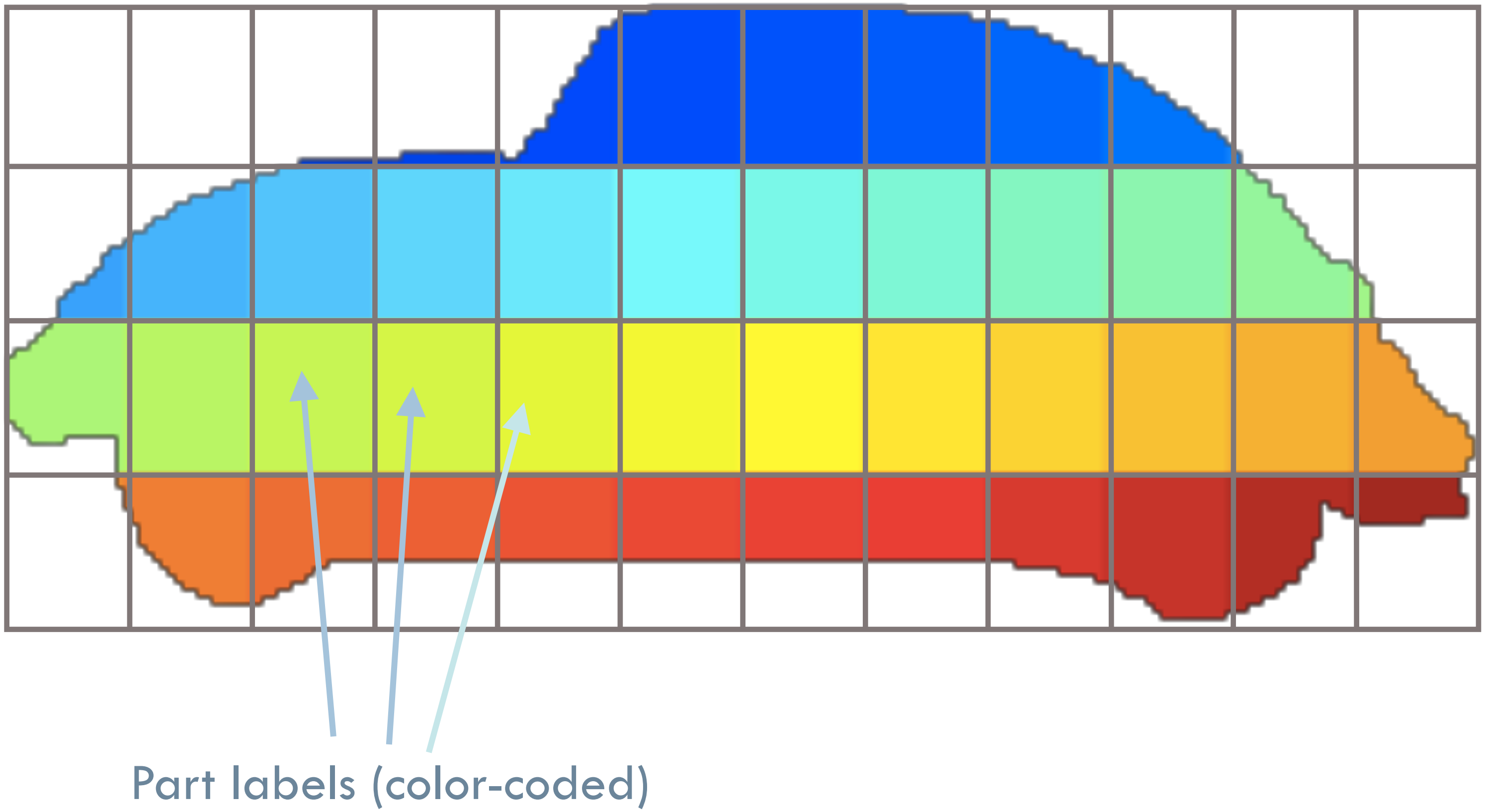




Images from [Kumar, Torr and Zisserman 05, Felzenszwalb & Huttenlocher 05]

Dense layout of parts

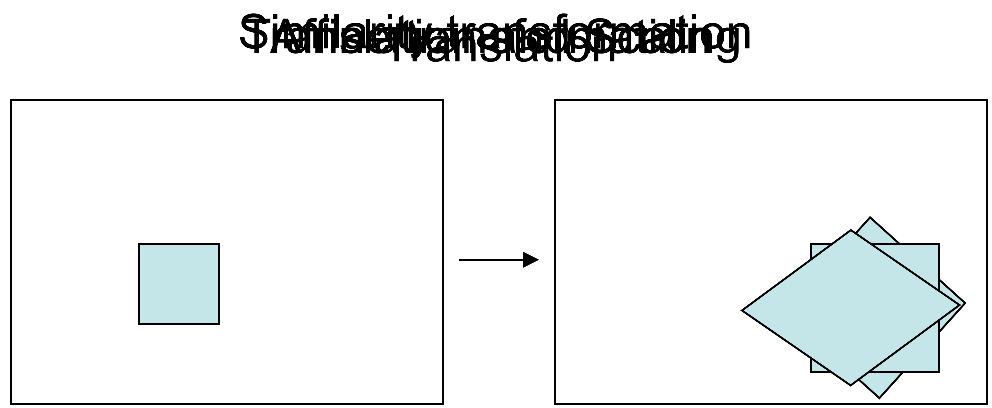
Layout CRF: Winn & Shotton, CVPR '06



How to model location?

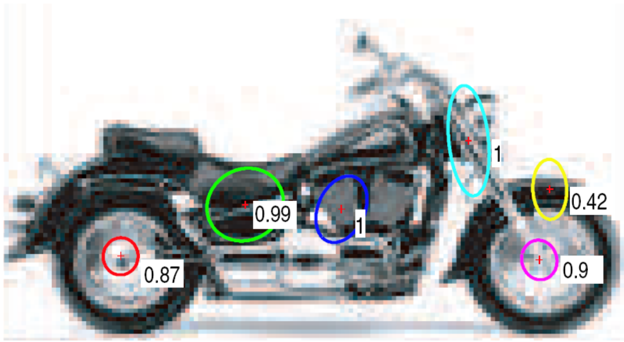
- Explicit: Probability density functions
- Implicit: Voting scheme

- Invariance
  - Translation
  - Scaling
  - Similarity/affine
  - Viewpoint



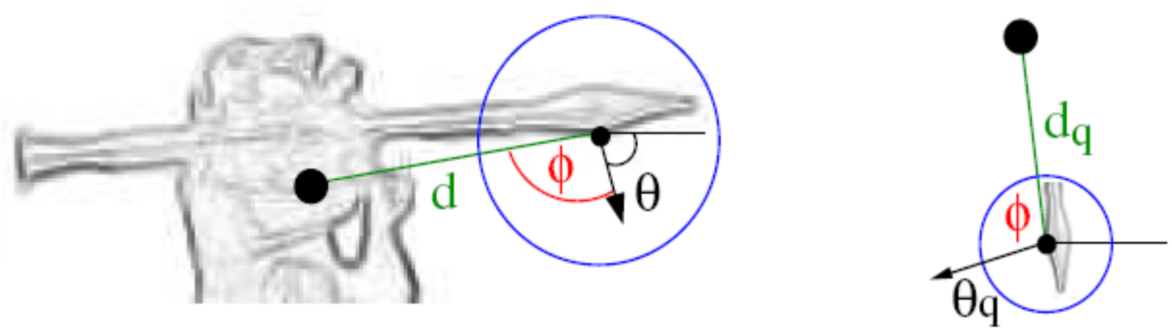
Explicit shape model

- Cartesian
  - E.g. Gaussian distribution
  - Parameters of model, μ and Σ
  - Independence corresponds to zeros in Σ
  - Burl et al. '96, Weber et al. '00, Fergus et al. '03



$$\mu = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix} \quad \Sigma = \begin{pmatrix} x_1x_1 & x_1x_2 & x_1x_3 & x_1y_1 & x_1y_2 & x_1y_3 \\ x_2x_1 & x_2x_2 & x_2x_3 & x_2y_1 & x_2y_2 & x_2y_3 \\ x_3x_1 & x_3x_2 & x_3x_3 & x_3y_1 & x_3y_2 & x_3y_3 \\ y_1x_1 & y_1x_2 & y_1x_3 & y_1y_1 & y_1y_2 & y_1y_3 \\ y_2x_1 & y_2x_2 & y_2x_3 & y_2y_1 & y_2y_2 & y_2y_3 \\ y_3x_1 & y_3x_2 & y_3x_3 & y_3y_1 & y_3y_2 & y_3y_3 \end{pmatrix}$$

- Polar
  - Convenient for invariance to rotation



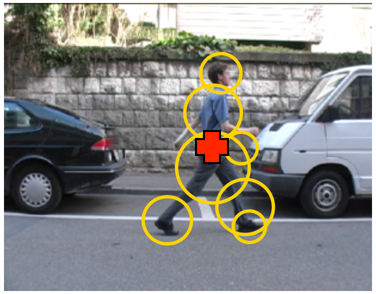
Mikolajczyk et al., CVPR '06

Implicit shape model

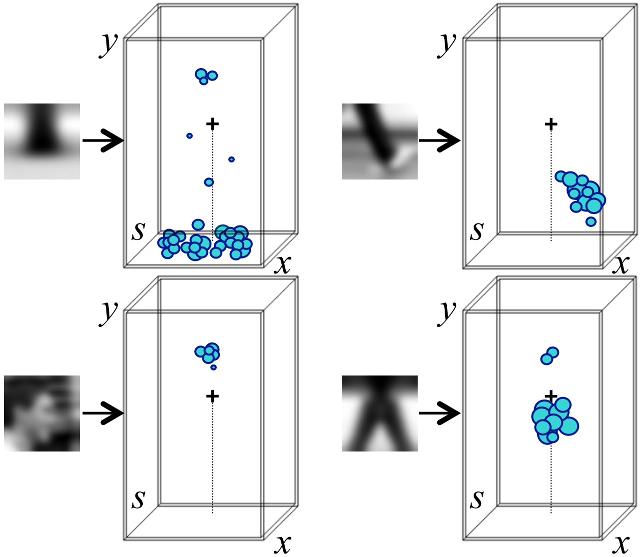
- Use Hough space voting to find object
- Leibe and Schiele ’03,’05

Learning

- Learn appearance codebook
  - Cluster over interest points on training images



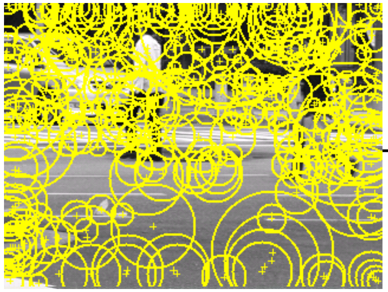
- Learn spatial distributions
  - Match codebook to training images
  - Record matching positions on object
  - Centroid is given



Spatial occurrence distributions

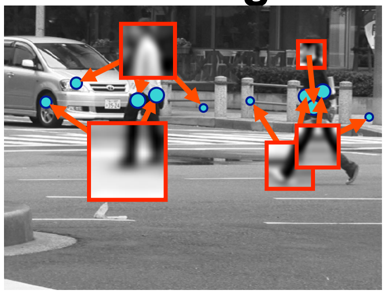
Recognition

Interest Points



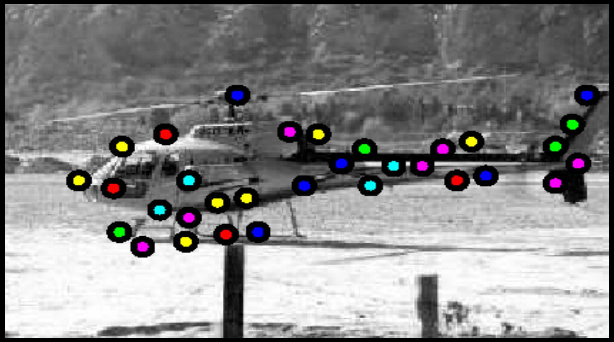
Matched Codebook Entries

Probabilistic Voting

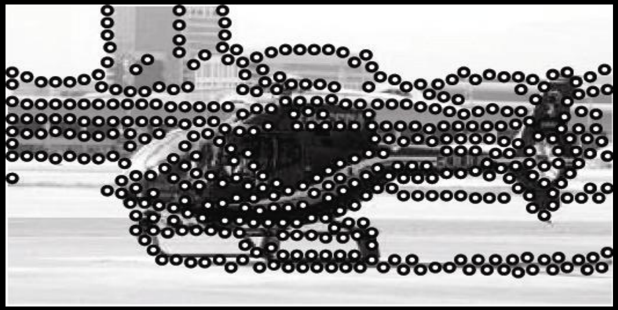


Deformable Template Matching

Berg, Berg and Malik CVPR 2005



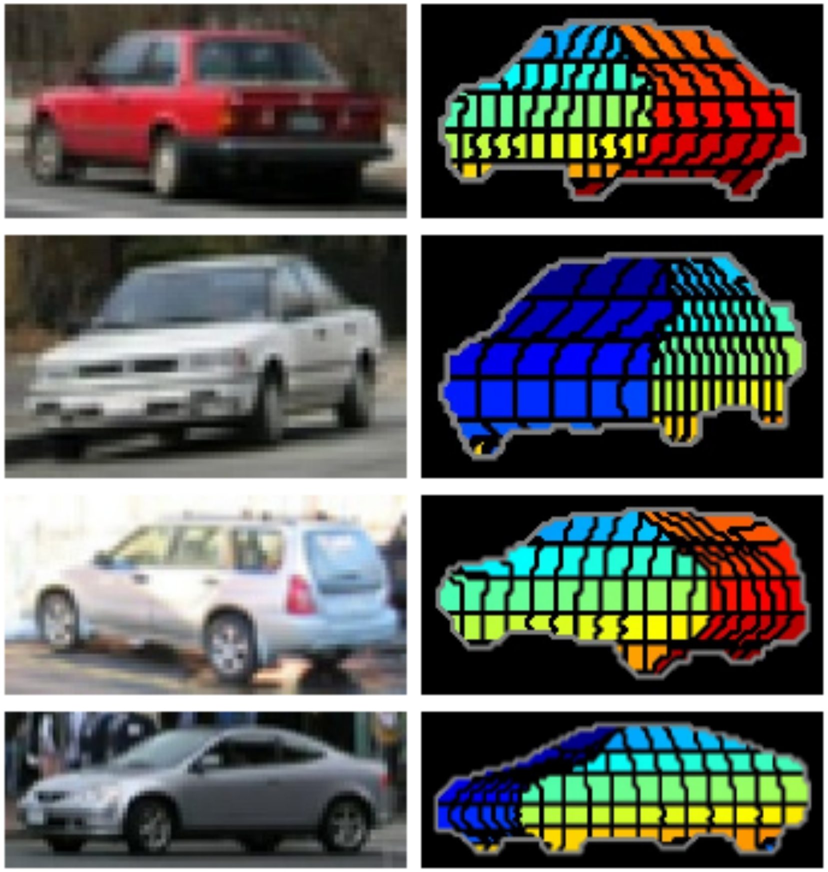
Template



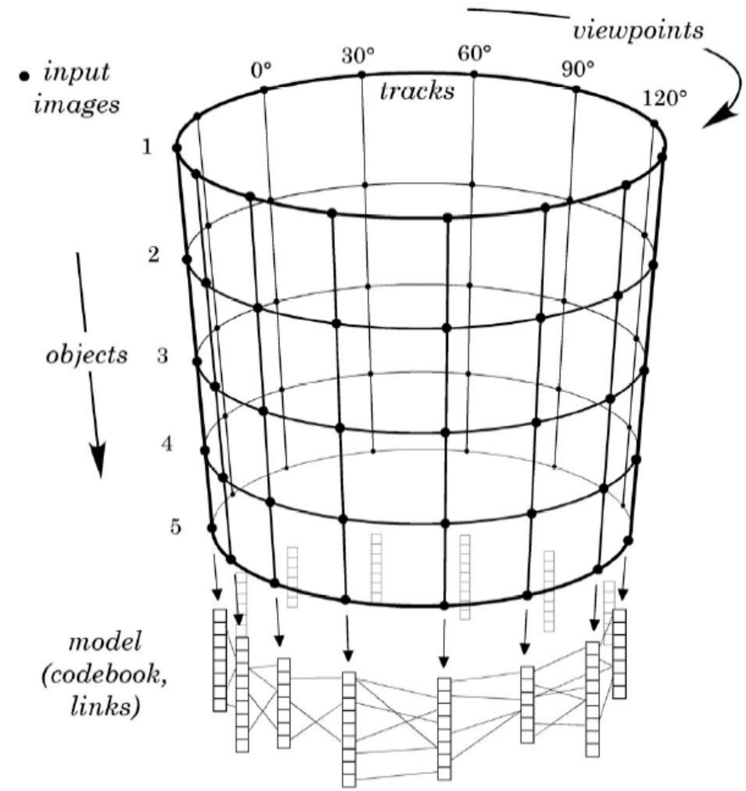
Query

- Formulate problem as Integer Quadratic Programming
- $O(N^P)$ in general
- Use approximations that allow P=50 and N=2550 in <2 secs

Multiple view points



Hoiem, Rother, Winn, 3D LayoutCRF for Multi-View Object Class Recognition and Segmentation, CVPR ‘07



Thomas, Ferrari, Leibe, Tuytelaars, Schiele, and L. Van Gool. Towards Multi-View Object Class Detection, CVPR 06

Representation of appearance

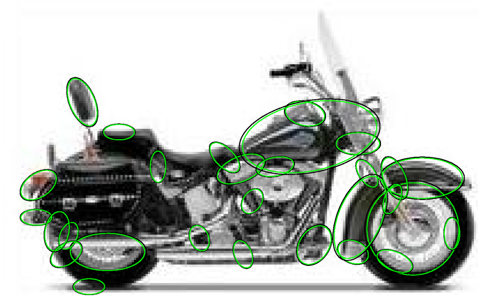
- Needs to handle intra-class variation
  - Task is no longer matching of descriptors
  - Implicit variation (VQ to get discrete appearance)
  - Explicit model of appearance (e.g. Gaussians in SIFT space)

- Dependency structure
  - Often assume each part's appearance is independent
  - Common to assume independence with location

Representation of appearance

- Invariance needs to match that of shape model

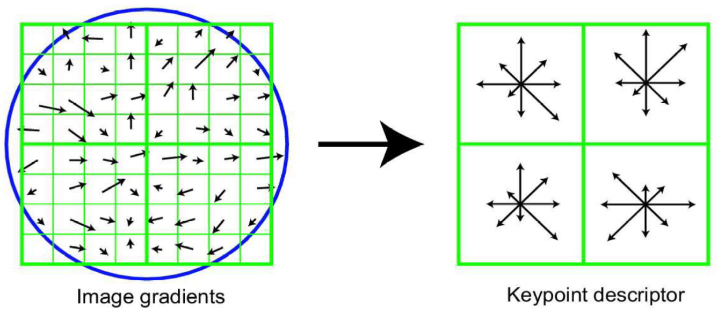
- Insensitive to small shifts in translation/scale
  - Compensate for jitter of features
  - e.g. SIFT



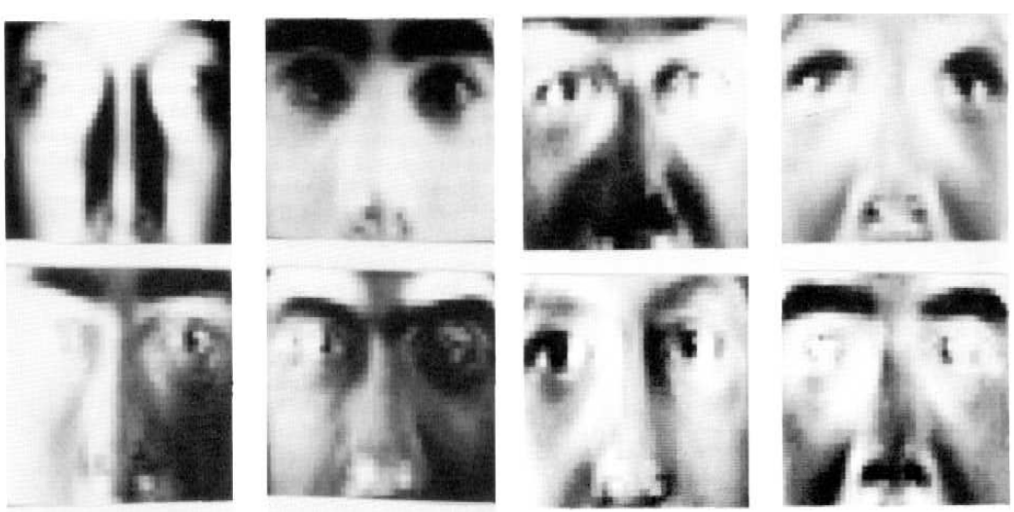
- Illumination invariance
  - Normalize out

Appearance representation

- SIFT



- PCA



- Decision trees

[Lepetit and Fua CVPR 2005]

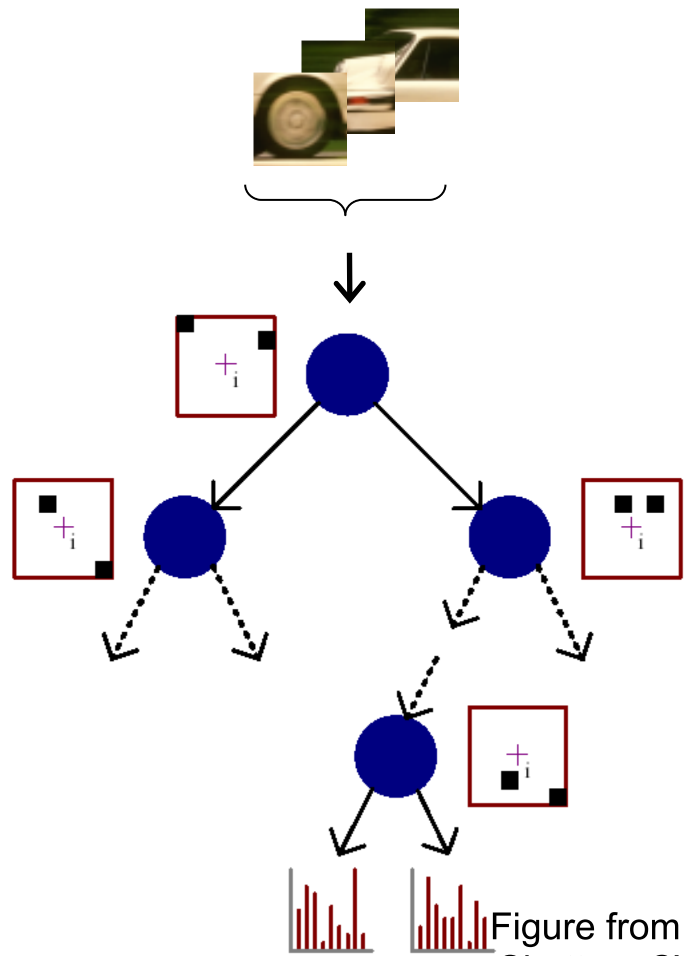
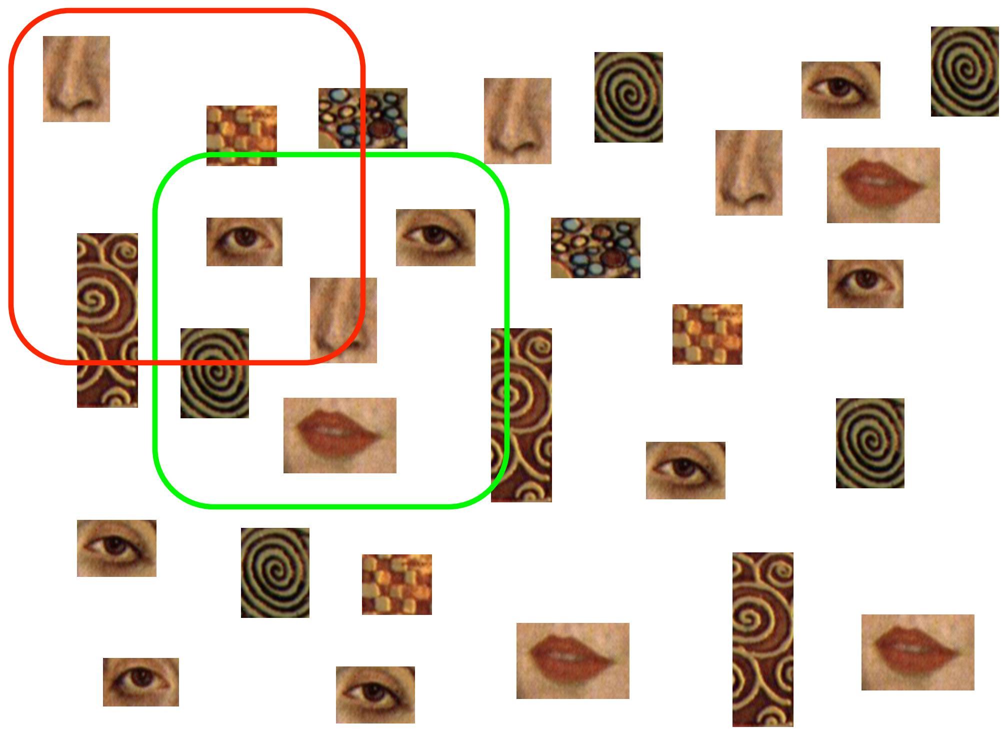


Figure from Winn & Shotton, CVPR '06

Background clutter

- Explicit model
  - Generative model for clutter as well as foreground object
- Use a sub-window
  - At correct position, no clutter is present



Demo Web Page

A simple parts and structure object detector - Microsoft Internet Explorer provided by Insight Broadband

File Edit View Favorites Tools Help

Address http://people.csail.mit.edu/fergus/iccv2005/partsstructure.html

A simple parts and structure object detector

ICCV 2005 short courses on
Recognizing and Learning Object Categories

An intuitive way to represent objects is as a collection of distinctive parts. Such schemes model both the relative positions of the parts as well as their appearance, giving a sparse representation that captures the essence of the object.

This simple demo illustrates the concepts behind many such "parts and structure" approaches. For simplicity, training is manually guided with the user hand-clicking on the distinctive parts of a few training images. A simple model is then built for use in recognition. Two different recognition approaches are provided: one relying on feature points [1]; the other using the efficient methods of Felzenswalb and Huttenlocher [2].

The code consists of Matlab scripts (which should run under both Windows and Linux). The Image Processing toolbox is required. The code is for teaching/research purposes only. If you find a bug, please email me at fergus where csail point mit point edu.

Download

Download the code and datasets (24 Mbytes)

Operation of code

To run the demos:

Demo (2)

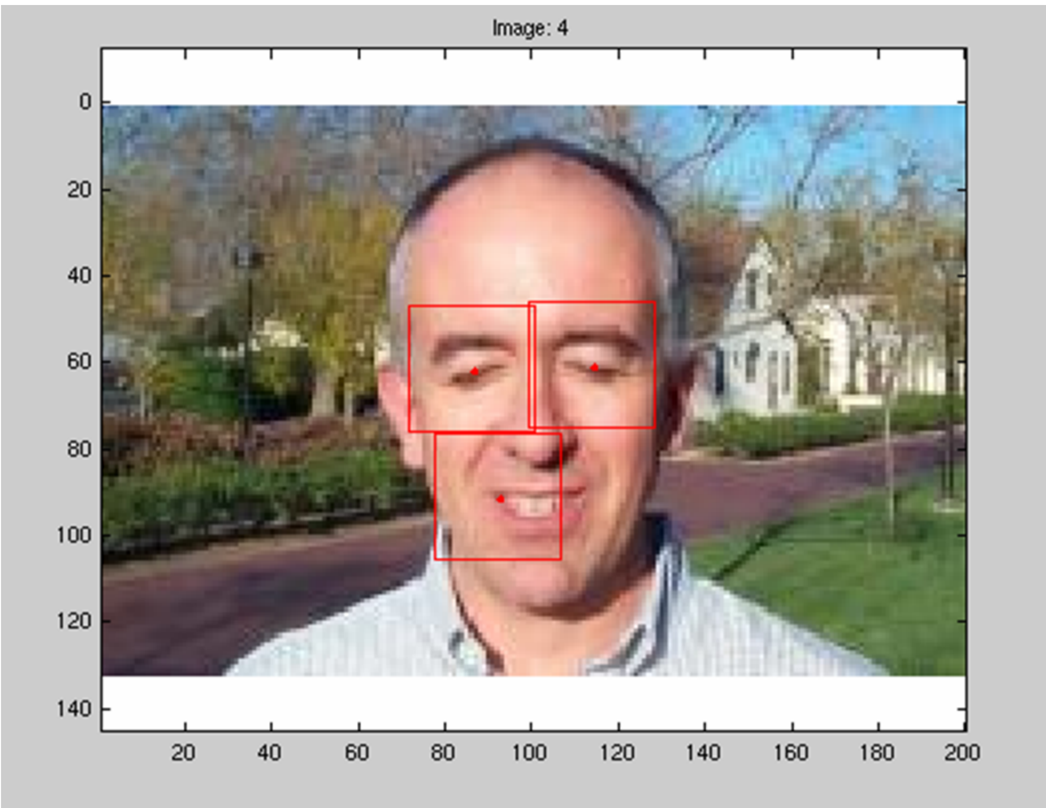
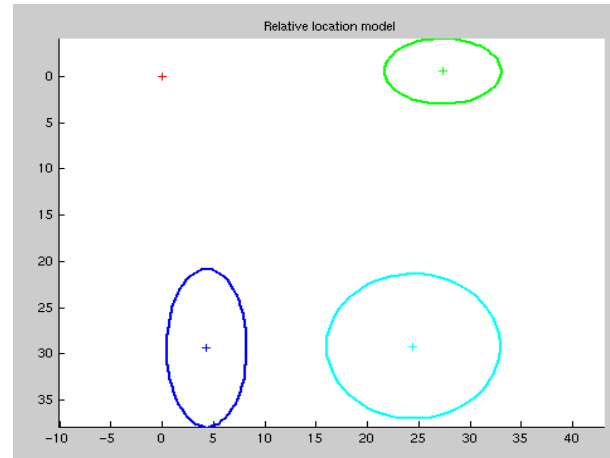
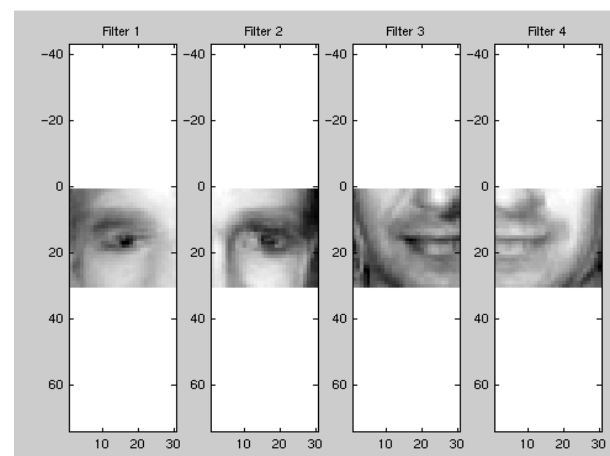
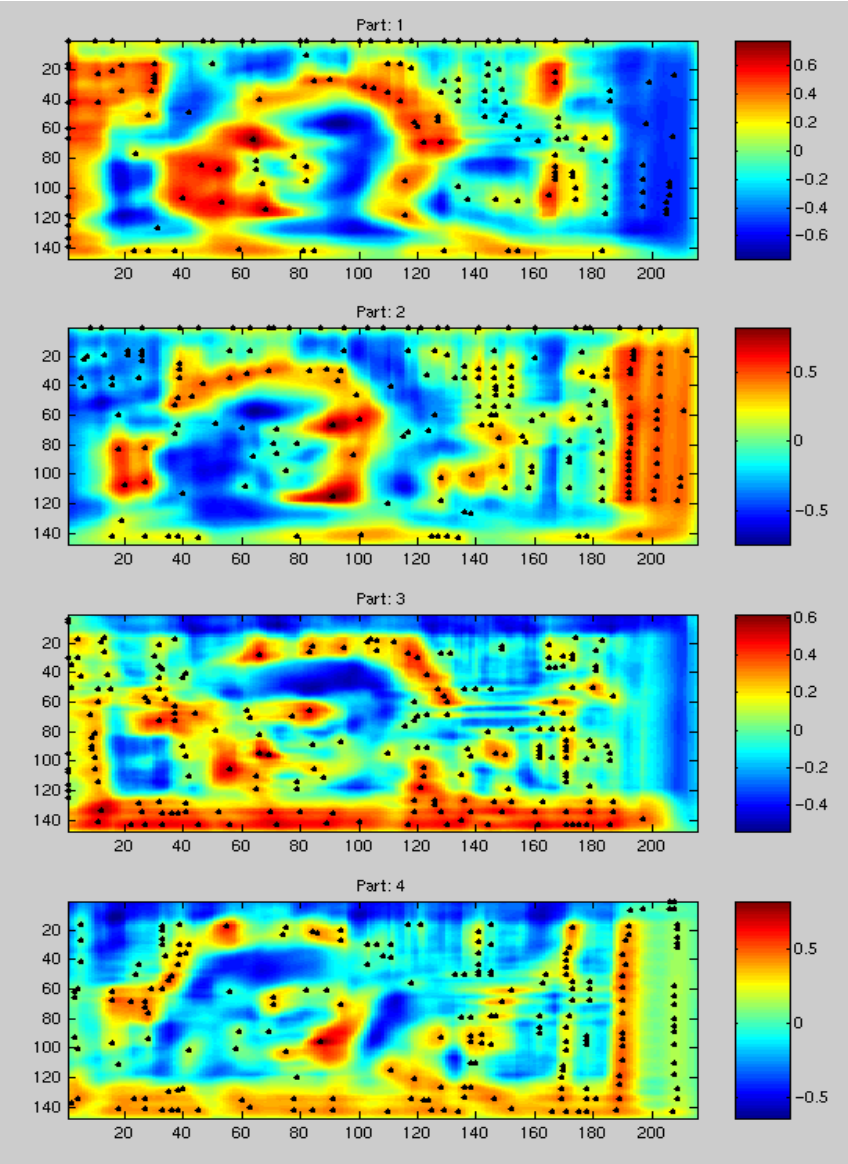


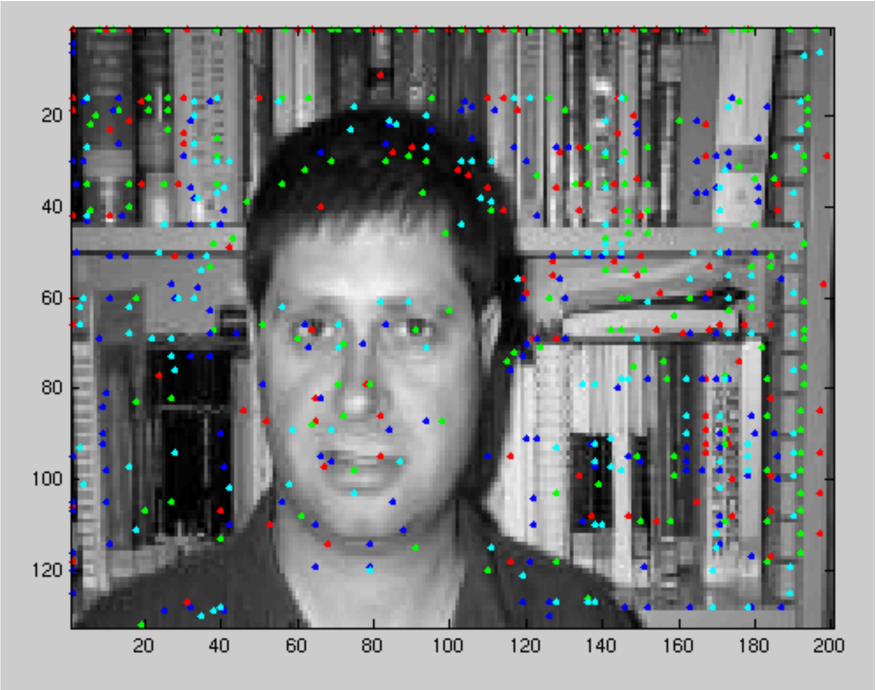
Image: 4


Relative location model

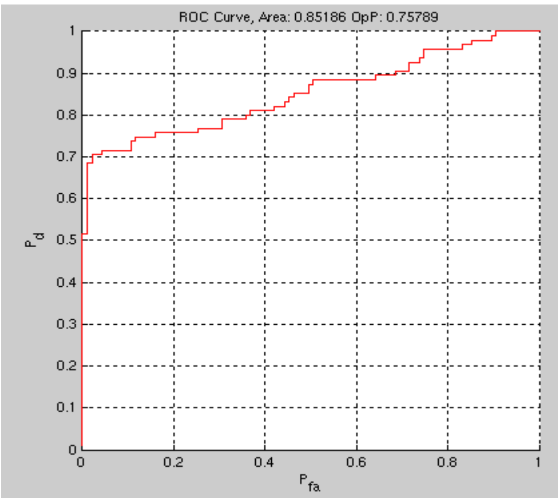

Filter 1
Filter 2
Filter 3
Filter 4

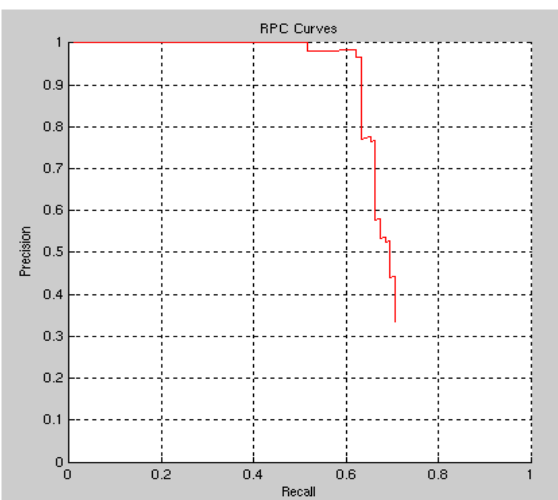
Demo (3)





Demo (4)


ROC Curve, Area: 0.85186 OpP: 0.75789
P_d
P_{fa}


RPC Curves
Precision
Recall

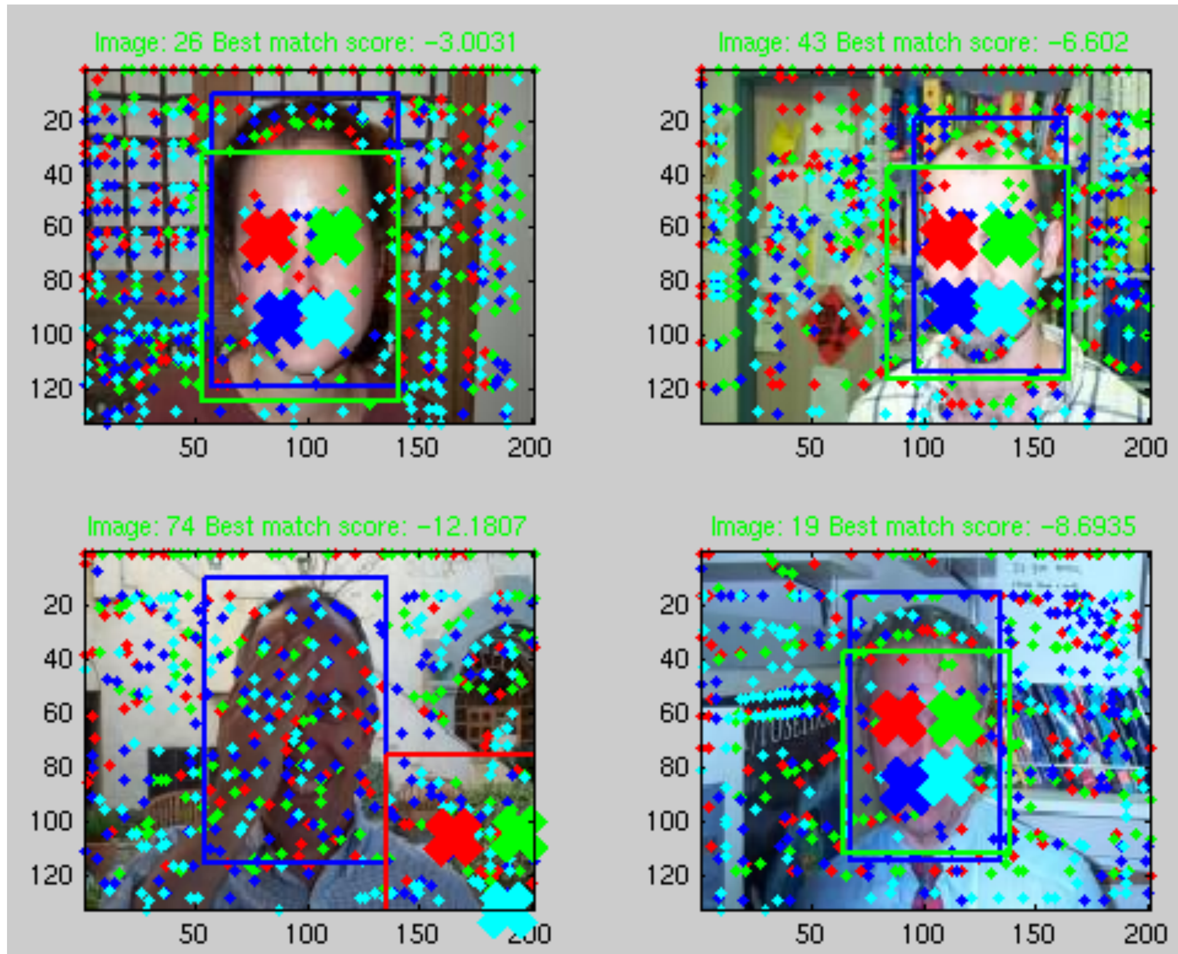


Image: 26 Best match score: -3.0031
Image: 43 Best match score: -6.602
Image: 74 Best match score: -12.1807
Image: 19 Best match score: -8.6935

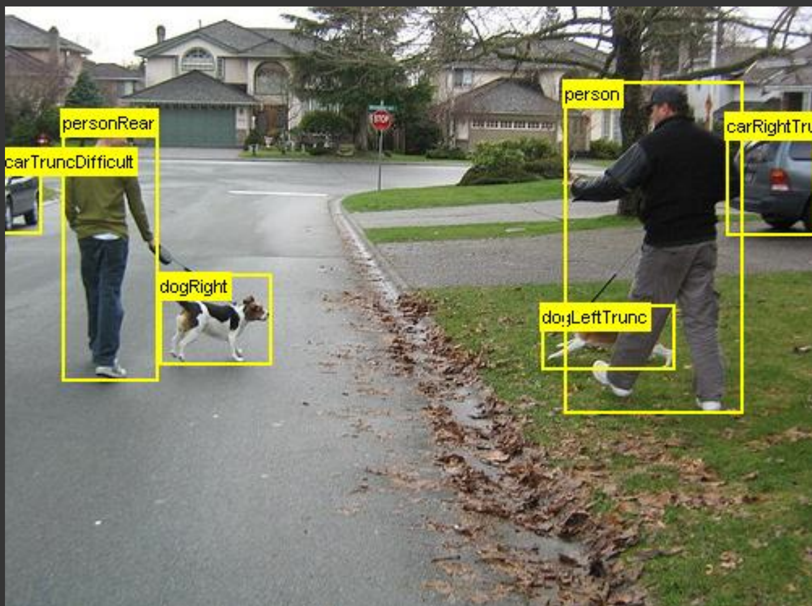
Object Detection with Discriminatively Trained Part Based Models

Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester and Deva Ramanan

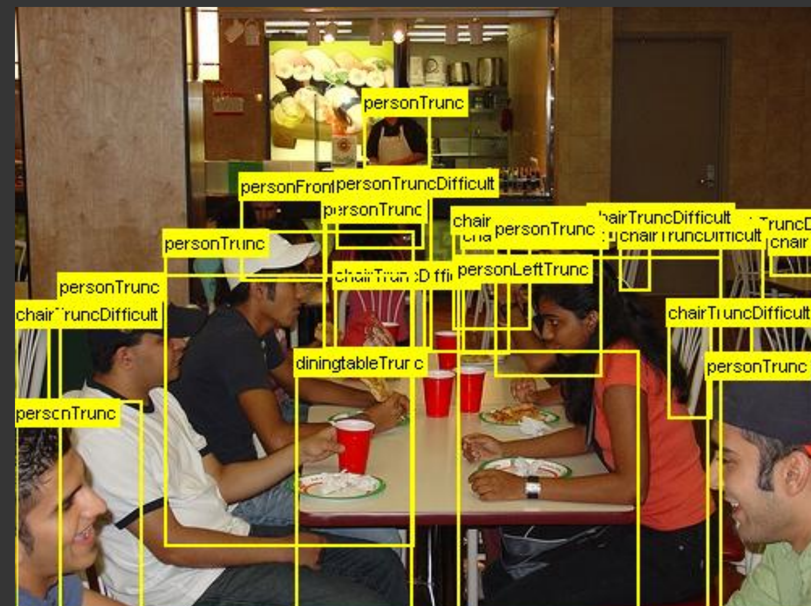
Abstract—We describe an object detection system based on mixtures of multiscale deformable part models. Our system is able to represent highly variable object classes and achieves state-of-the-art results in the PASCAL object detection challenges. While deformable part models have become quite popular, their value had not been demonstrated on difficult benchmarks such as the PASCAL datasets. Our system relies on new methods for discriminative training with partially labeled data. We combine a margin-sensitive approach for data-mining hard negative examples with a formalism we call *latent SVM*. A latent SVM is a reformulation of MI-SVM in terms of latent variables. A latent SVM is semi-convex and the training problem becomes convex once latent information is specified for the positive examples. This leads to an iterative training algorithm that alternates between fixing latent values for positive examples and optimizing the latent SVM objective function.

Index Terms—Object Recognition, Deformable Models, Pictorial Structures, Discriminative Training, Latent SVM

PASCAL Visual Object Challenge



5000 training images



5000 testing images

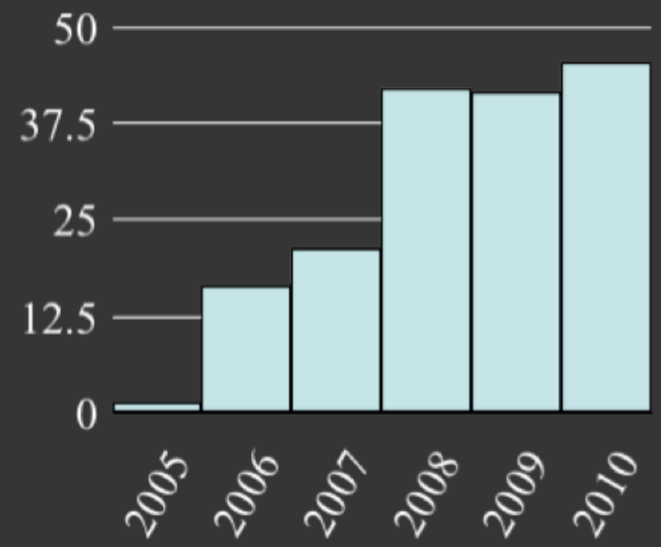
20 everyday object categories

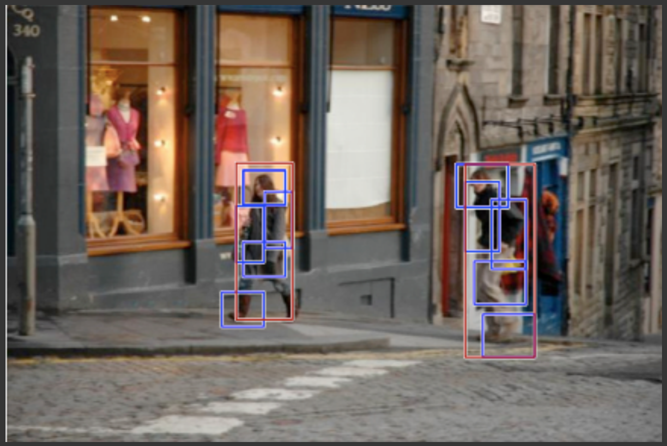
aeroplane bike bird boat bottle bus car cat chair cow table
dog horse motorbike person plant sheep sofa train tv

Source: Deva Ramanan

5 years of PASCAL people detection

average precision





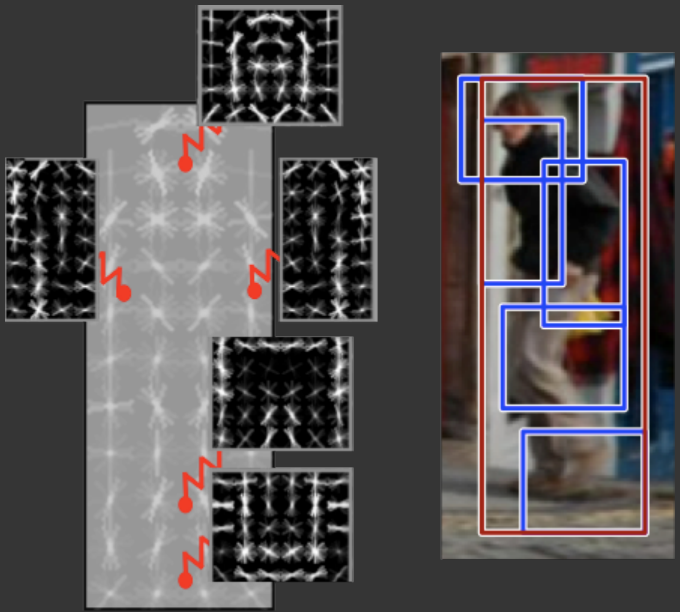
1% to 45% in 5 years

Discriminative mixtures of star models 2007-2010 Felzenszwalb, McAllester, Ramanan CVPR 2008

Felzenszwalb, Girshick, McAllester, and Ramanan PAMI 2009

Source: Deva Ramanan

Deformable part models



Model encodes local appearance + pairwise geometry

Source: Deva Ramanan

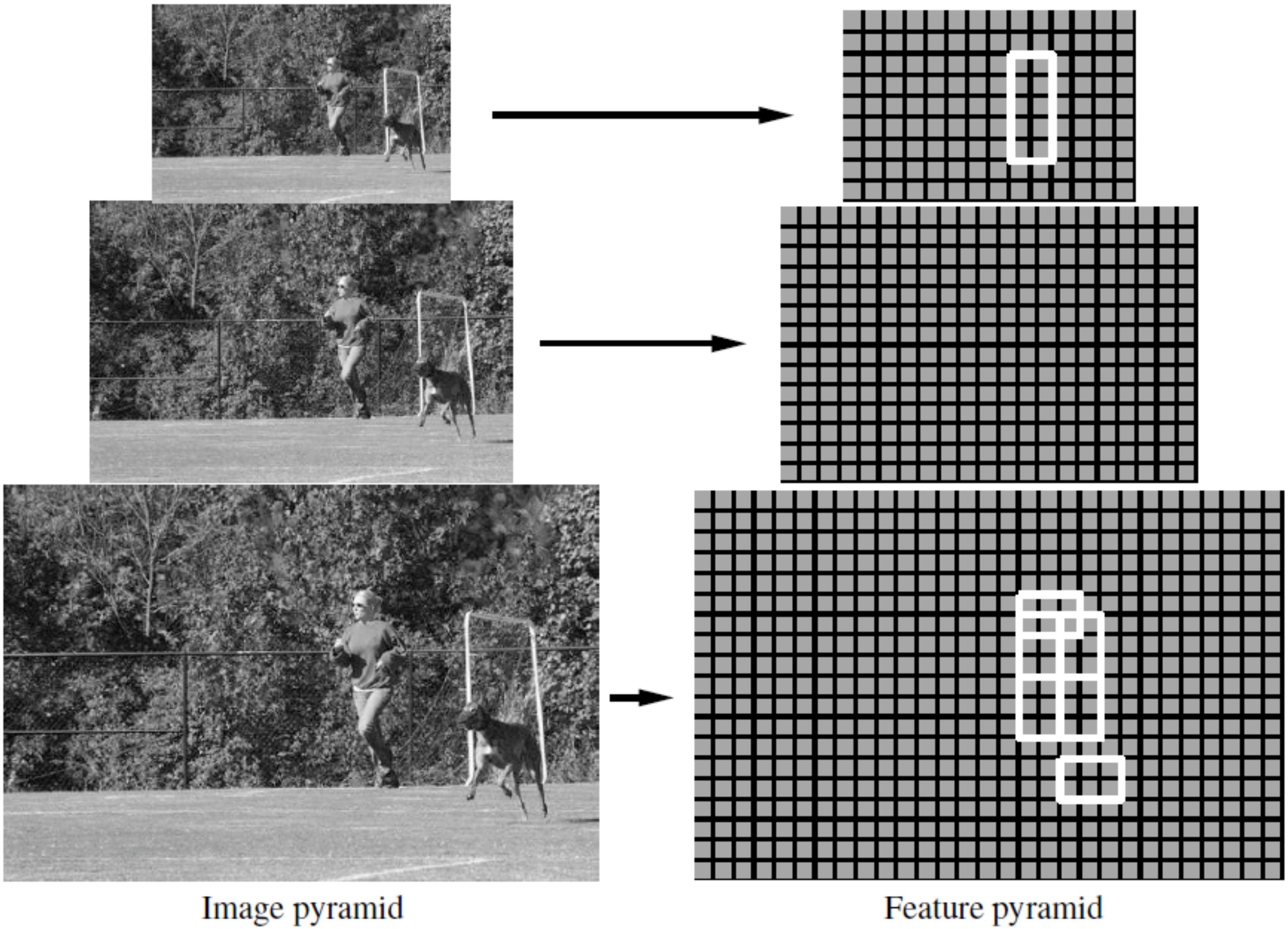
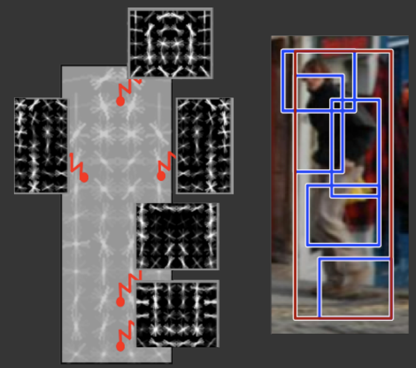


Image pyramid
Feature pyramid

Scoring function

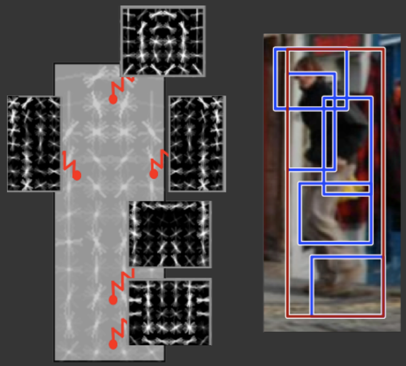


score(x,z)

x = image
$z_i = (x_i, y_i)$
$z = \{z_1, z_2 ...\}$

Source: Deva Ramanan

Scoring function



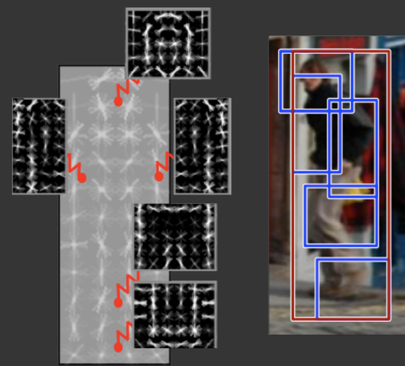
$$\text{score}(x,z) = \sum_i w_i\, \phi\,(x, z_i) +$$

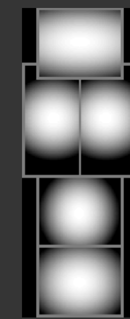
x = image
$z_i = (x_i, y_i)$
$z = \{z_1, z_2 \ldots\}$

part template
scores

Source: Deva Ramanan

Scoring function





$$\text{score}(x,z) = \sum_i w_i \, \phi(x, z_i) + \sum_{i,j} w_{ij} \, \Psi(z_i, z_j)$$

x = image
$z_i = (x_i, y_i)$
$z = \{z_1, z_2 \ldots\}$

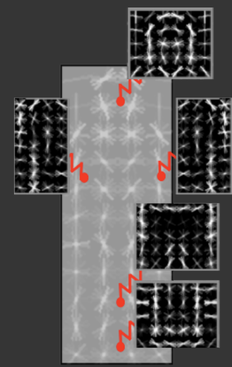
part template scores

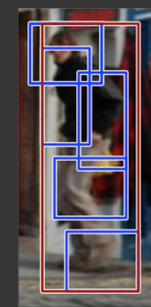
spring deformation model

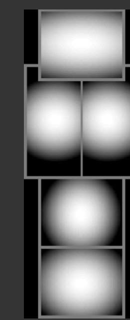
E = relational graph

Source: Deva Ramanan

Scoring function







$$\text{score}(x,z) = \sum_i w_i \, \phi(x, z_i) + \sum_{i,j} w_{ij} \, \Psi(z_i, z_j)$$

x = image
$z_i = (x_i, y_i)$
$z = \{z_1, z_2 \ldots\}$

part template scores

spring deformation model

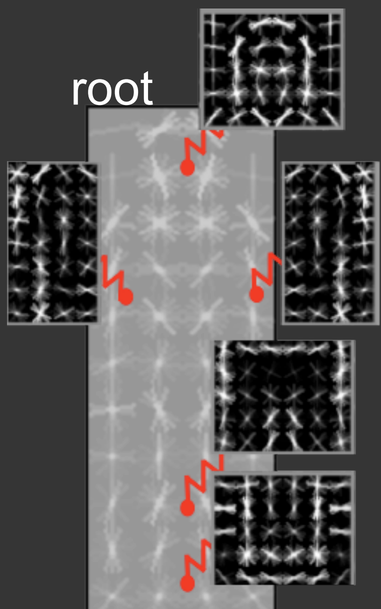
Score is linear in local templates w_i and spring parameters w_{ij}

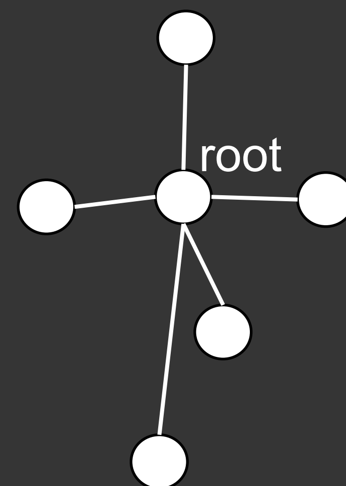
$$\text{score}(x,z) = w \cdot \Phi(x, z)$$

Source: Deva Ramanan

Inference: $\max_{z} \text{score}(x,z)$

Felzenszwalb & Huttenlocher 05





Star model: the location of the root filter is the anchor point
Given the root location, all part locations are independent

Source: Deva Ramanan

Classification

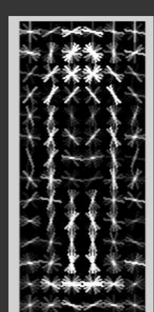
$f_w(x)>0$

$f_w(x)=w \cdot \Phi(x)$

Source: Deva Ramanan

Latent-variable classification

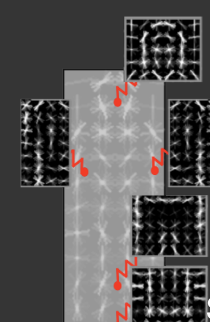
$f_w(x)=w \cdot \Phi(x)$



$f_w(x)>0$

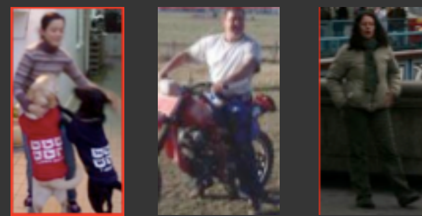
$f_w(x)=\max_z S(x,z)$

$=\max_z w \cdot \Phi(x, z)$

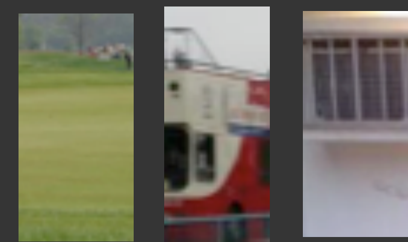


Source: Deva Ramanan

Latent SVMs



pos



neg

Given positive and negative training windows $\{x_n\}$

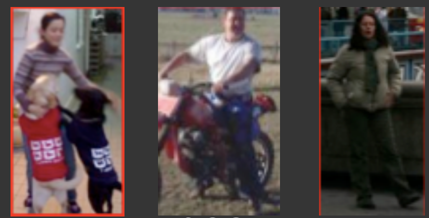
$$L(w) = ||w||^2 + \sum_{n \in \text{pos}} \max(0, 1 - f_w(x_n)) + \sum_{n \in \text{neg}} \max(0, 1 + f_w(x_n))$$

$$f_w(x) = \max_z w \cdot \Phi(x, z)$$

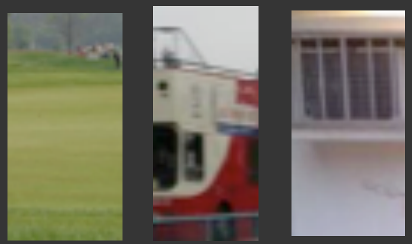
L(w) is “almost” convex

Source: Deva Ramanan

Latent SVMs



pos



neg

Given positive and negative training windows $\{x_n\}$

$$L(w) = ||w||^2 + \sum_{n \in \text{pos}} \max(0, 1 - \cancel{f_w(x_n)}) + \sum_{n \in \text{neg}} \max(0, 1 + f_w(x_n))$$

$$w \cdot \Phi(x_n, z_n)$$

$$f_w(x) = \max_z w \cdot \Phi(x, z)$$

L(w) is convex if we fix latent values for positives

Source: Deva Ramanan

Coordinate descent

1) Given positive part locations, learn w with a convex program

$$w = \underset{w}{\operatorname{argmin}}\, L(w) \quad \text{with fixed} \quad \{z_n : n \in \text{pos}\}$$

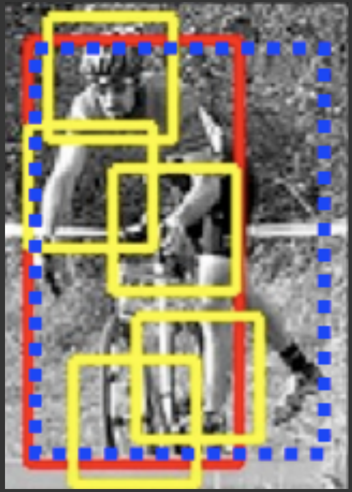
2) Given w, estimate part locations on positives

$$z_n = \underset{z}{\operatorname{argmax}}\, w \cdot \Phi(x_n, z) \quad \forall n \in \text{pos}$$

The above steps perform coordinate descent on a joint loss

Source: Deva Ramanan

Treat ground-truth labels as partially latent



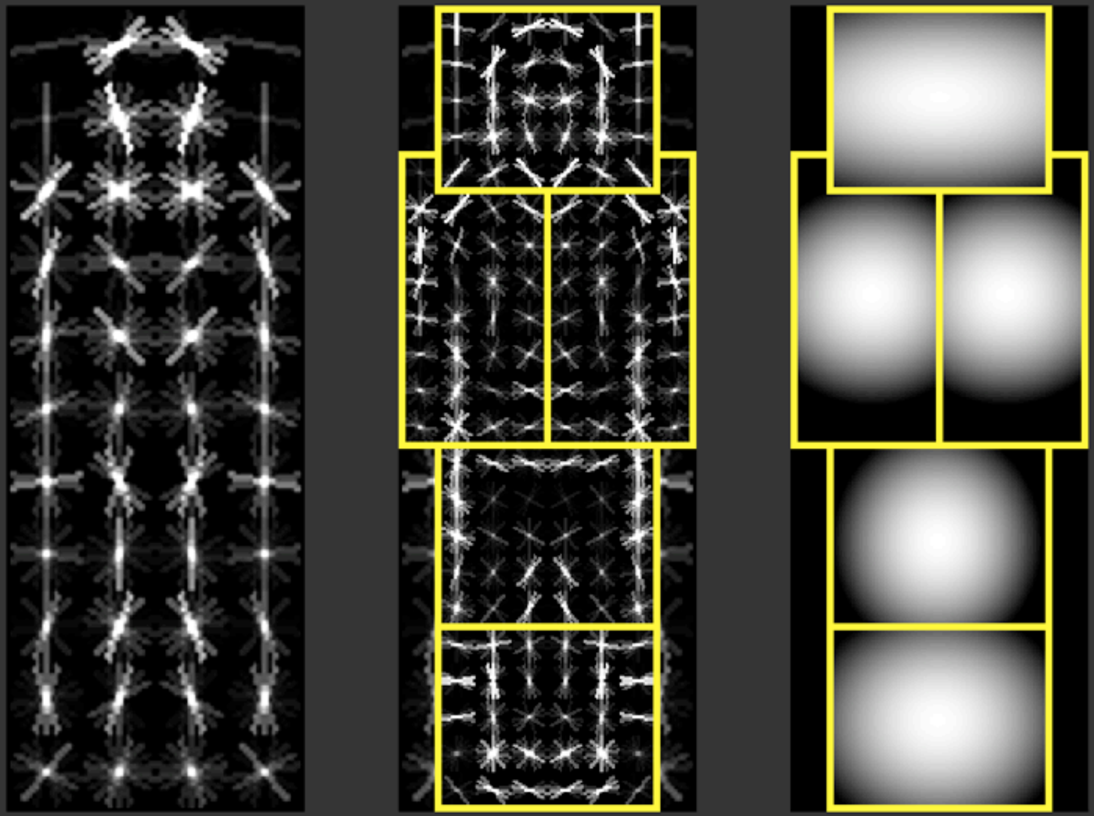
Allows for “cleaning up” of noisy labels (in blue) during iterative learning

Source: Deva Ramanan

Initialization

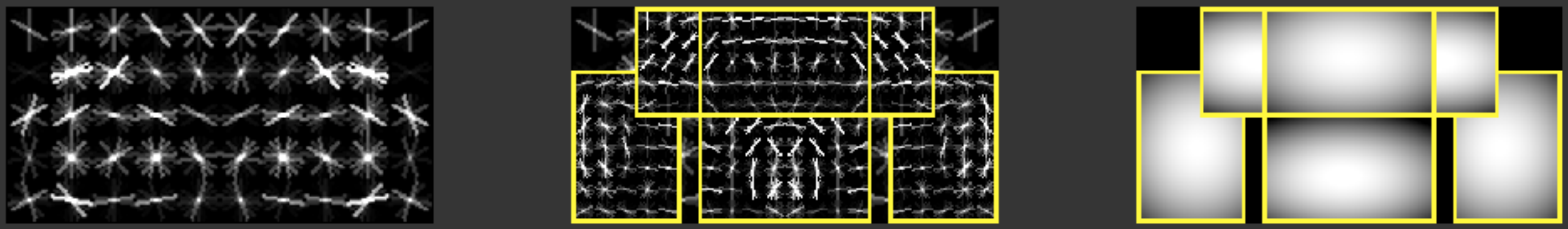
Learn root filter with SVM

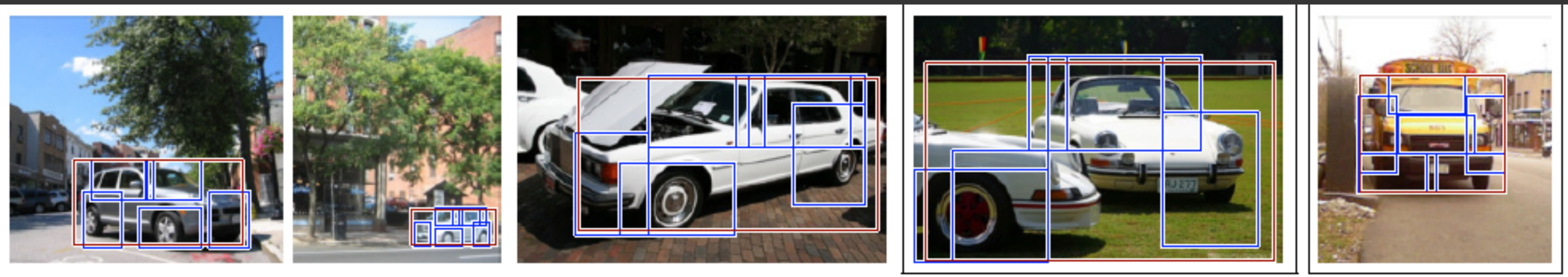
Initialize part filters to regions in root filter with lots of energy



Source: Deva Ramanan

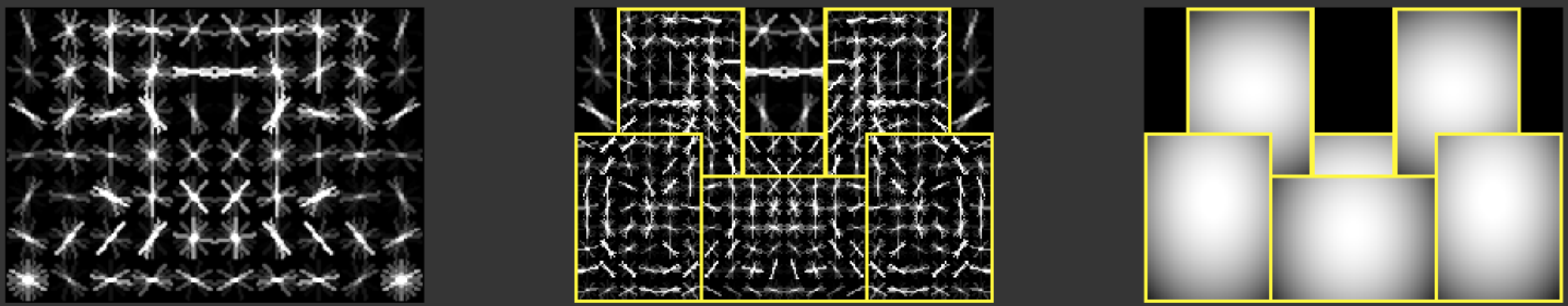
Example models





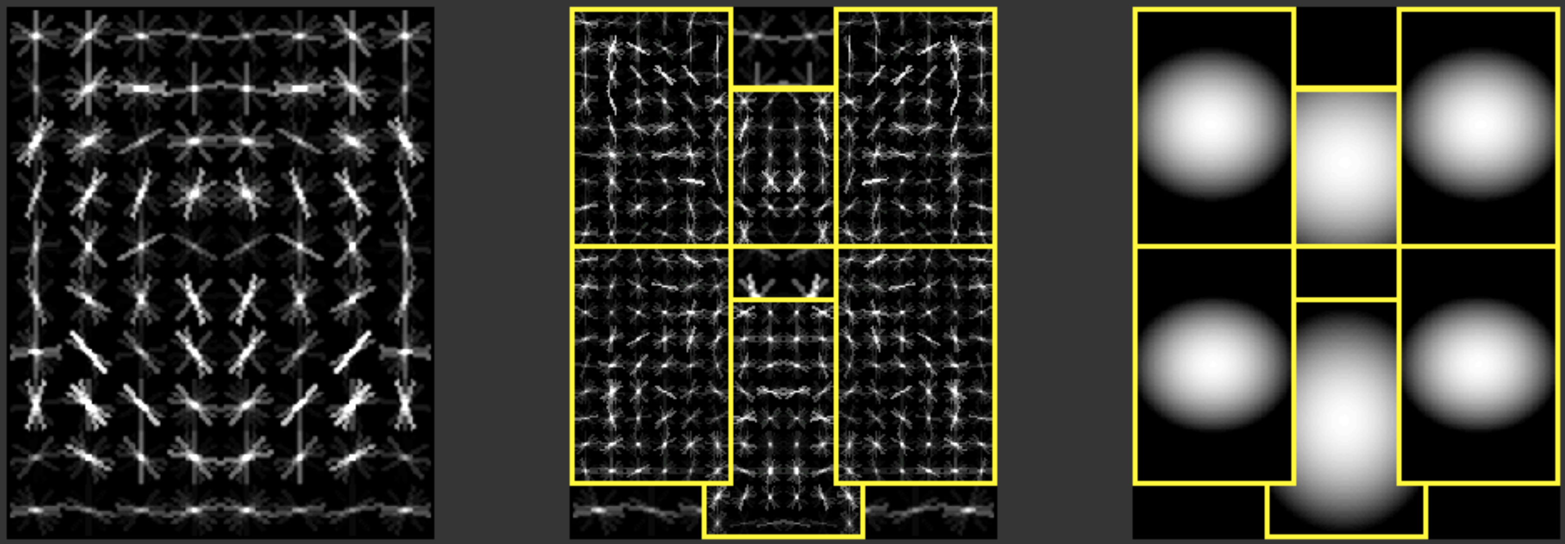
Source: Deva Ramanan

Example models

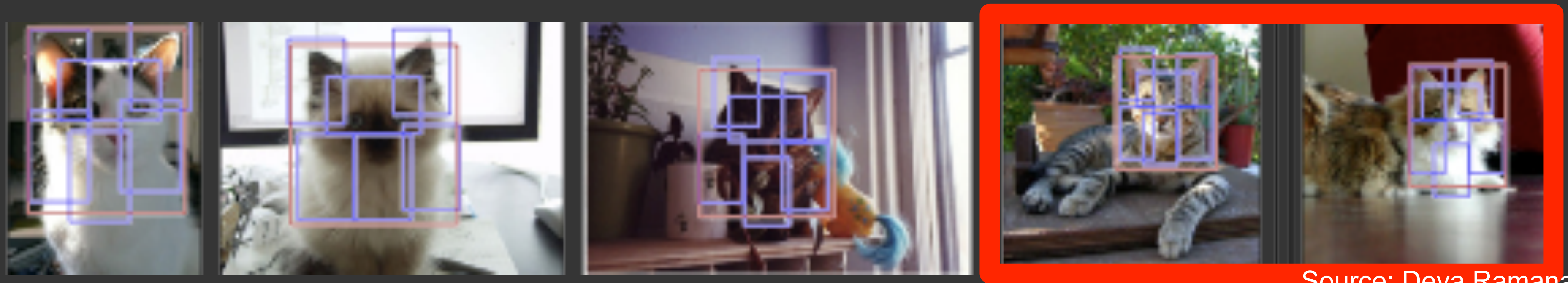


Source: Deva Ramanan

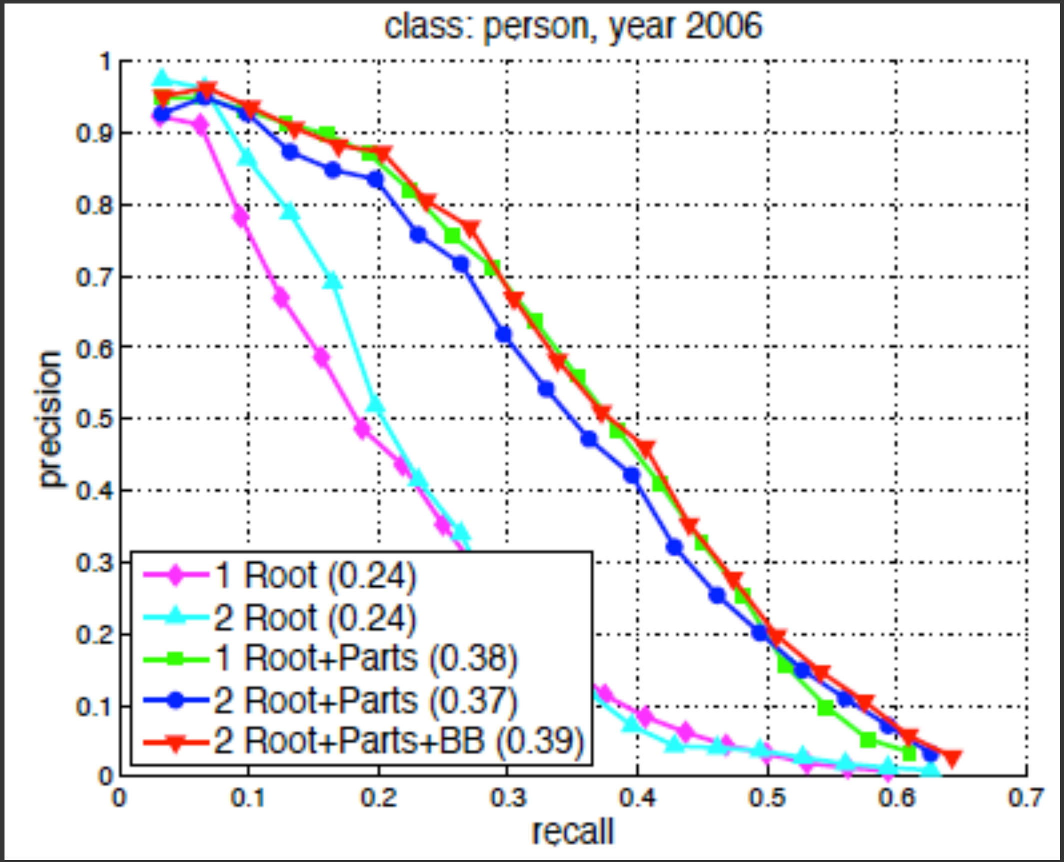
Example models



False positive due to imprecise bounding box



Source: Deva Ramanan



Other tricks:

•Mining hard negative examples

•Noisy annotations


horse
sofa
bottle
cat