Recognizing Specific Objects & Faces

Lectures 15 and 16

Many slides from: S. Savarese, S. Lazebnik, D. Lowe, P. Viola

Overview

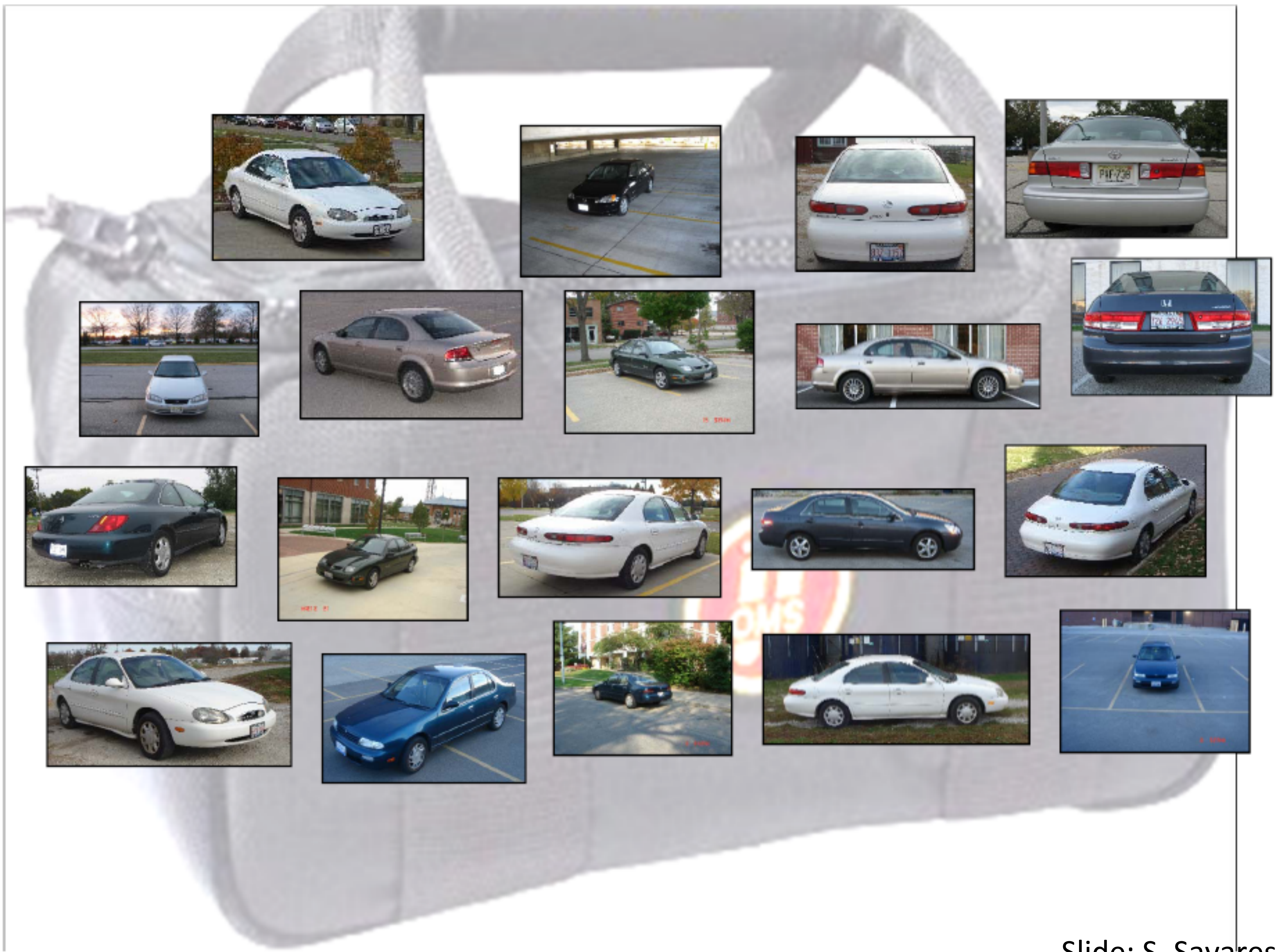
- Specific Object Recognition
  - Lowe ‘04
  - Rothganger et al. ‘03

- Face Dectection
  - Sub-space methods
  - Neural Network-based approaches
  - Viola & Jones

- Face Recognition

Specific Object Recognition

Slides from S. Savarese, D. Lowe and F. Rothganger



Slide: S. Savarese

Single 3D object recognition

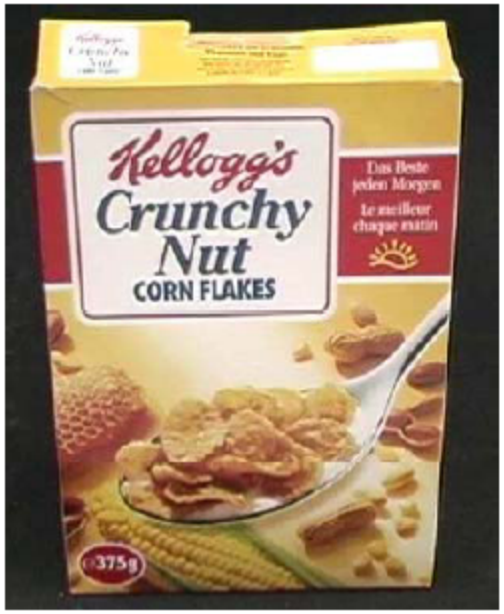
- Ballard, '81
- Grimson & L.-Perez, '87
- Lowe, '87

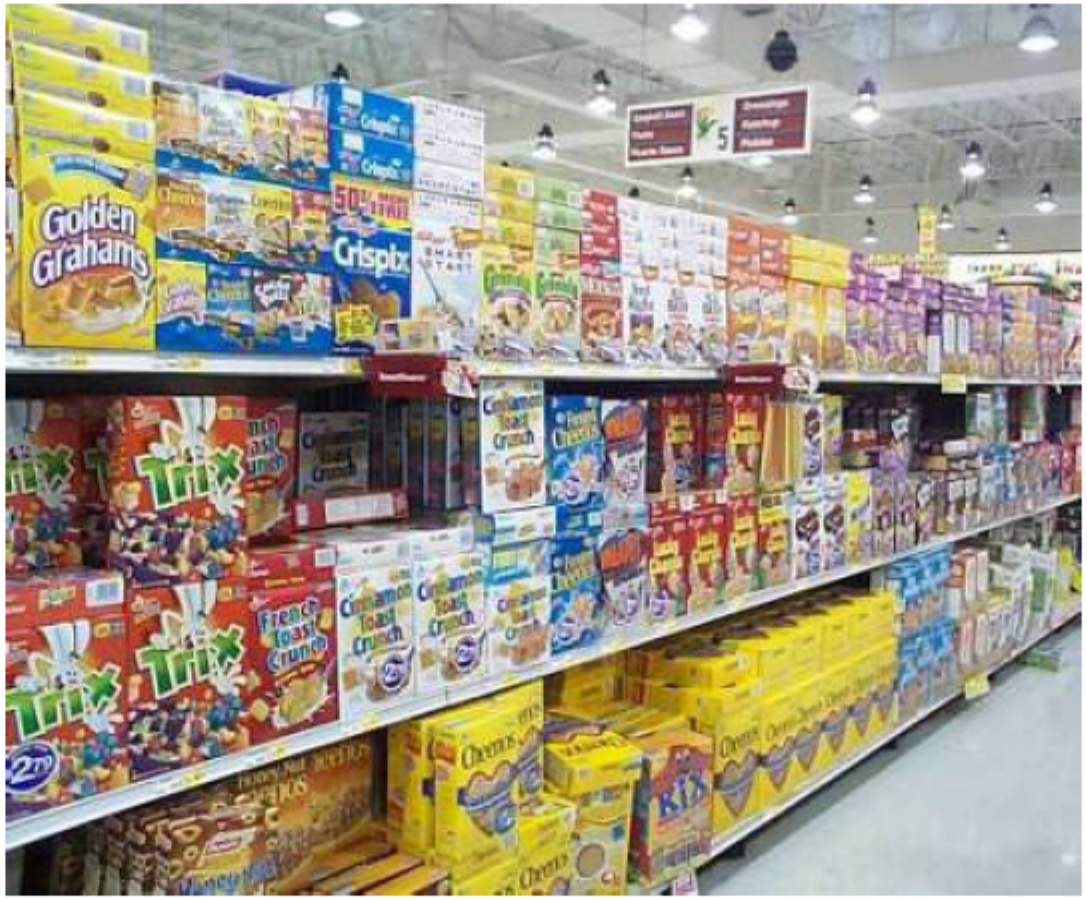
- Edelman et al. '91
- Ullman & Barsi, '91
- Rothwell '92
- Linderberg, '94
- Murase & Nayar '94

- Zhang et al '95
- Schmid & Mohr, '96
- Schiele & Crowley, '96
- **Lowe, '99**
- **Jacob & Barsi, '99**
- Mahamud and Herbert, 00

- **Rothganger et al., '04**
- **Ferrari et al, '05**
- Moreels and Perona, 05
- **Brown & Lowe '05**
- **Snavely et al '06**
- **Yin & Collins, '07**

Where is the crunchy nut?





Challenges:

- Variability due to:
  - View point
  - Illumination
  - Occlusions

  - But not intra-class variation

Recognition of single 3D objects

- Rothganger et al. '04, '06
- Brown et al, '05
- Lowe '99, '04
- Ferrari et al. '04, '06
- Lazebnick et al '04

-Representation

-Features

-2D/3D Geometrical constraints

-Model learning

-Recognition

-Hypothesis generation

-Validation

Representation

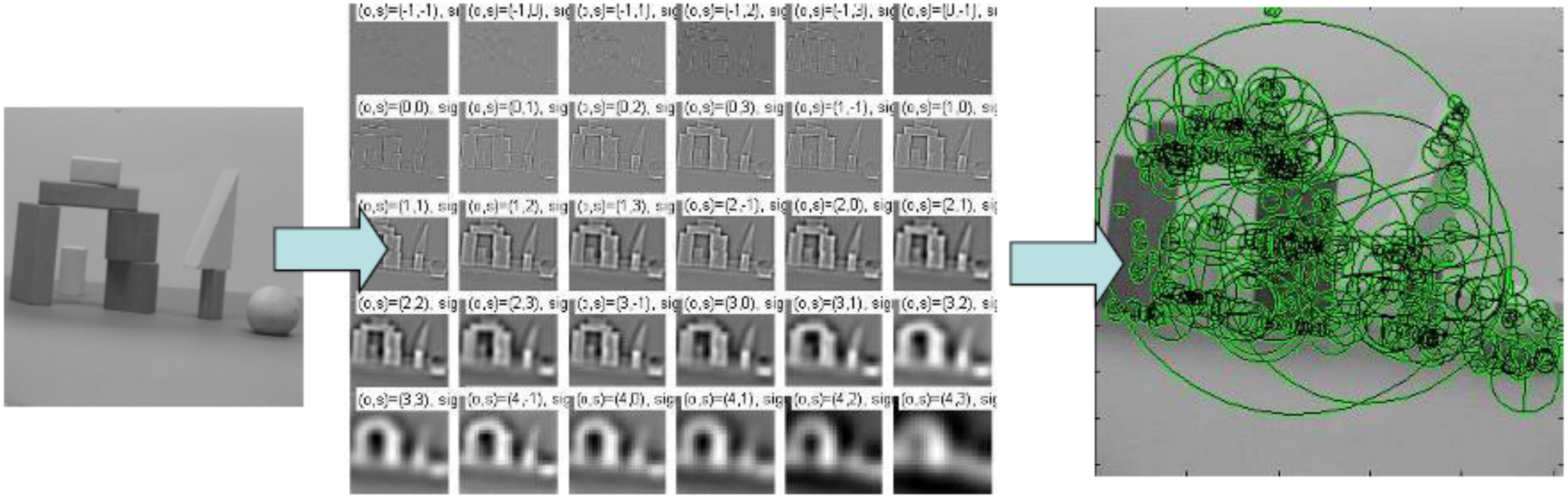
Interest points -- or Regions (group of interest points)

- Detection
  - Difference of Gaussian (DOG) [Lowe '99]
  - Harris-Laplacian [Mikolajczyk & Schmid '01]
  - Kadir-Brady [Kadir et al. '01]
  - Laplacian [Gårding & Lindeberg, '96]
- Adaptation [invariants]
  - Scale, rotation
  - Affine
- Description
  - SIFT
  - Color histograms

Geometrical constraints

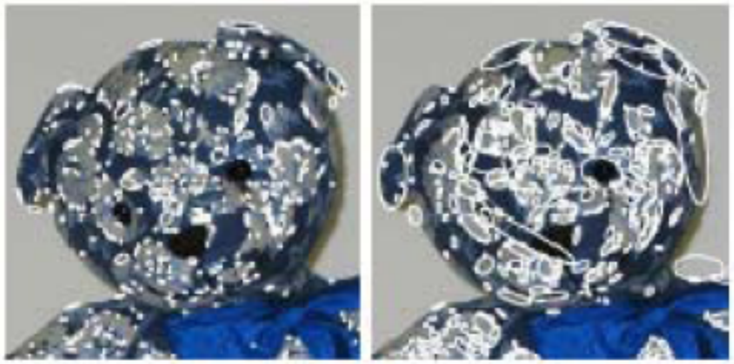
- 2D spatial layout of keypoints
- Tracks of keypoints (regions) across views
- 3D locations and/or surface normals

Difference of Gaussian (DOG): used in Lowe 99, Brown et al '05



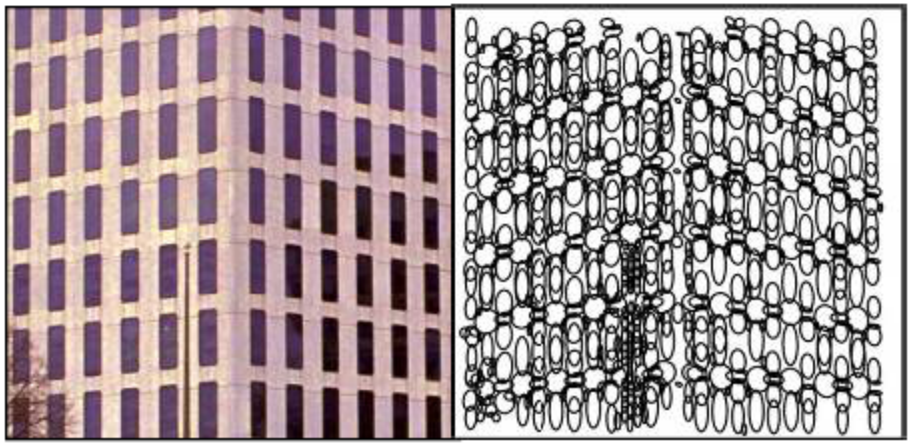
Courtesy of D. Lowe

Harris-Laplace: used in Rothganger et al. '06



Courtesy of Rothganger et al

Laplacian: used in Lazebnik et al. '04



Representation

Interest points -- or Regions (group of interest points)

- Detection
  - Difference of Gaussian (DOG) [Lowe '99]
  - Harris-Laplacian [Mikolajczyk & Schmid '01]
  - Kadir-Brady [Kadir et al. '01]
  - Laplacian [Gårding & Lindeberg, '96]

$\longrightarrow$

- x,y
- Scale
- Orientation
- Affine structure

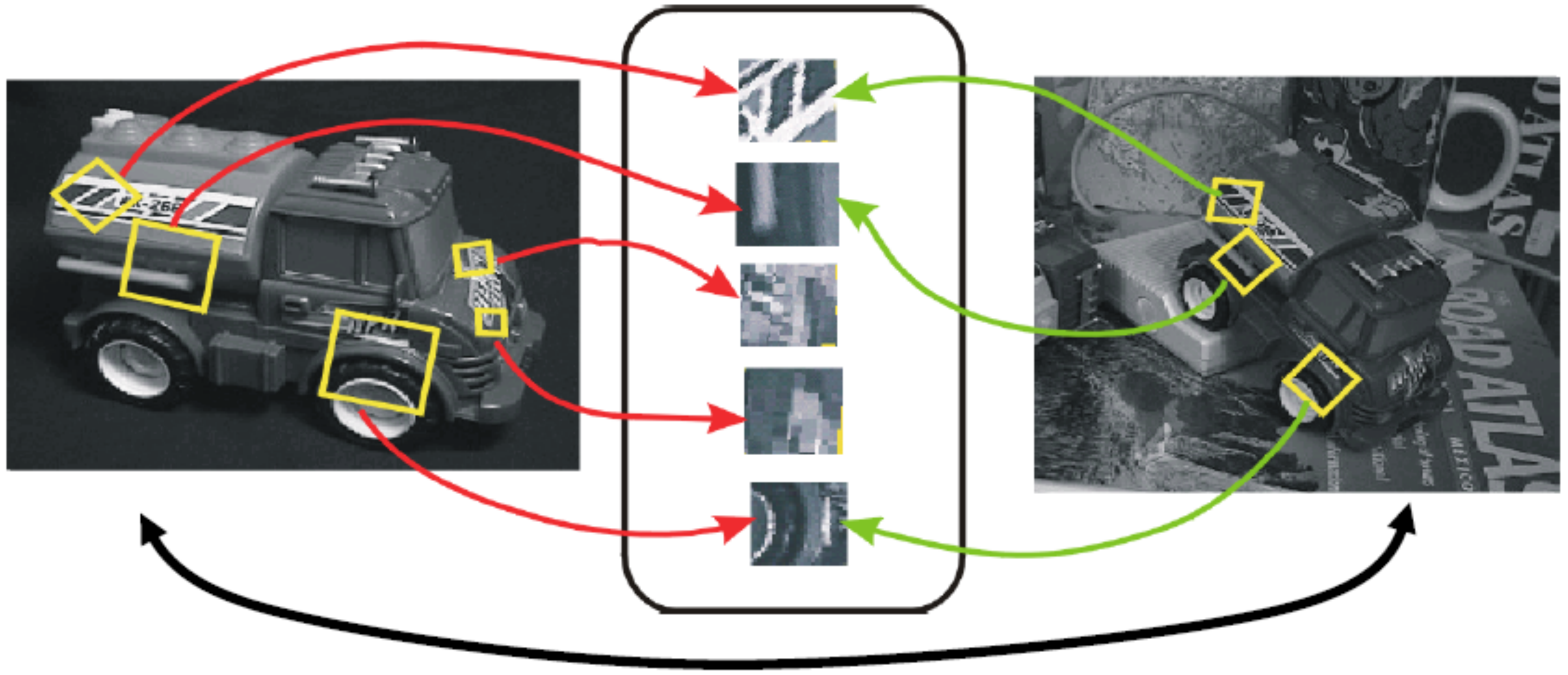
- Adaptation [invariants]
  - Scale, rotation
  - Affine
- Description
  - SIFT
  - Color histograms

Geometrical constraints

- 2D spatial layout of keypoints
- Tracks of keypoints (regions) across views
- 3D locations and/or surface normals

Adaptation

- keypoints are transformed in order to be invariant to translation, rotation, scale, and other geometrical parameters


Courtesy of D. Lowe

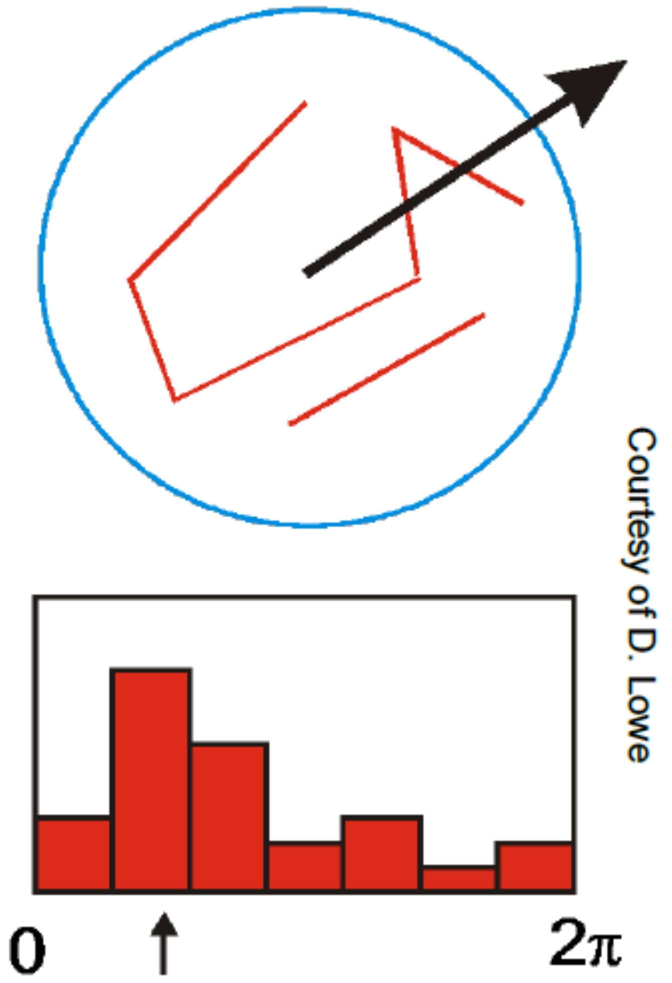
Change of scale, pose, illumination…

Scale & orientation adaptation

[used in Lowe '99]

detector $\longrightarrow$
- x,y
- Scale
- Orientation

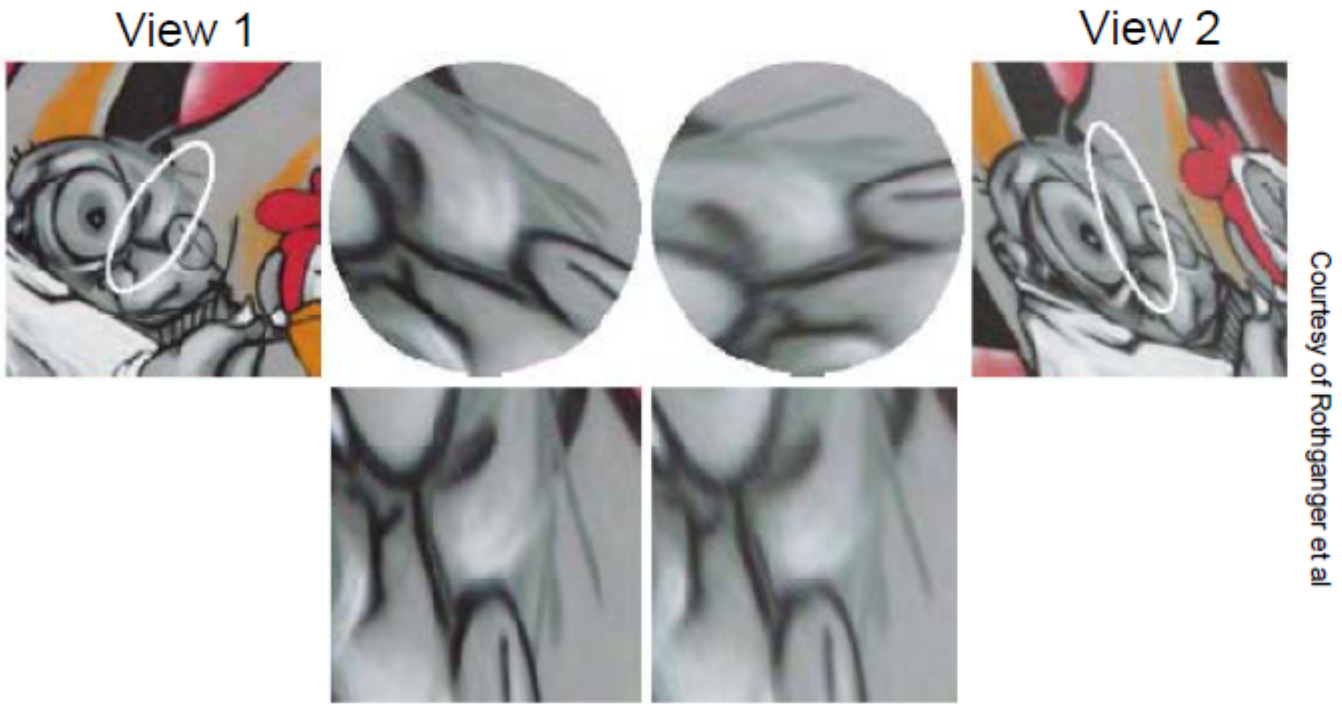
- SIFT: Create histogram of local gradient directions computed at selected scale
- Assign canonical orientation at peak of smoothed histogram



Affine adaptation

[used in Rothganger et al. '03, '06]
[Lazebnick et al '04]

1. Define elliptical region **using second moment matrix**
2. Use main canonical orientation to remove orientation ambiguity
3. Map ellipsis onto unit square



Representation

Interest points -- or Regions (group of interest points)

- Detection
  - Difference of Gaussian (DOG) [Lowe '99]
  - Harris-Laplacian [Mikolajczyk & Schmid '01]
  - Kadir-Brady [Kadir et al. '01]
  - Laplacian [Gårding & Lindeberg, '96]

$\rightarrow$

- x,y
- Scale
- Orientation
- Affine structure

- Adaptation [invariants]
  - Scale, rotation
  - Affine
- Description
  - SIFT
  - Color histograms

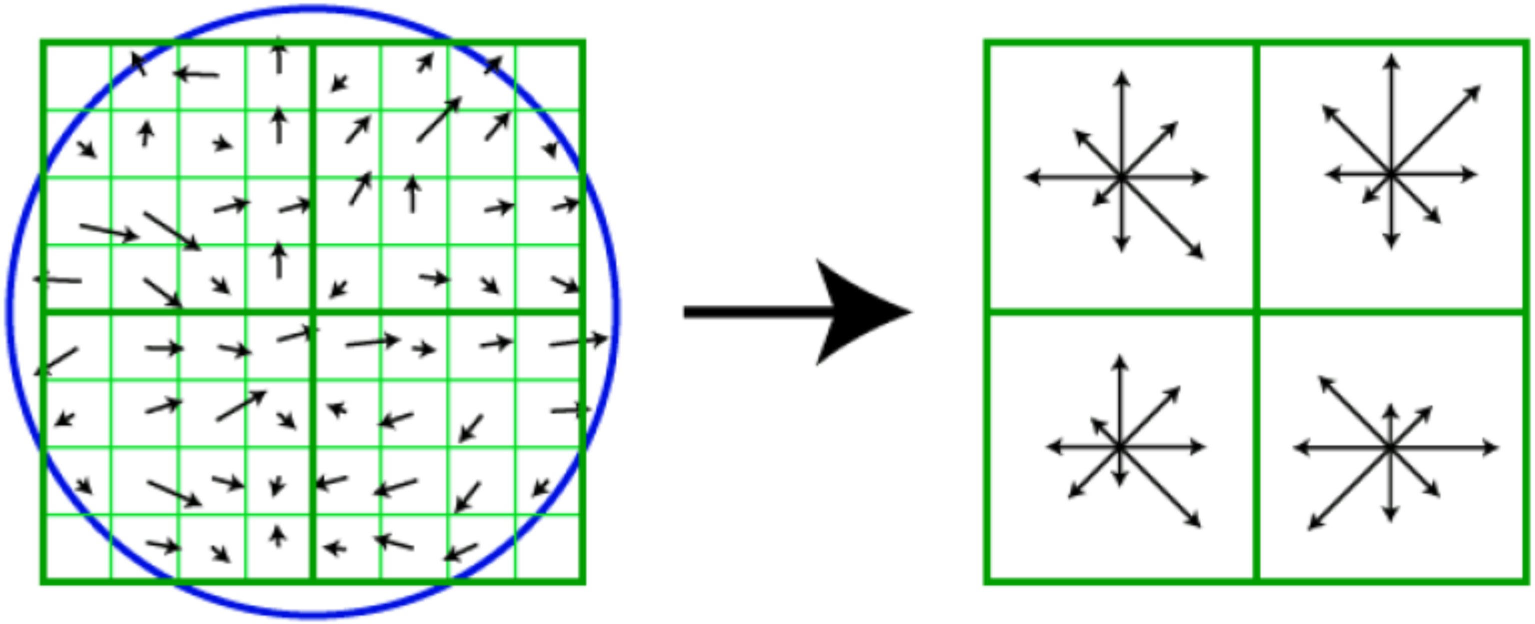
Geometrical constraints

- 2D spatial layout of keypoints
- Tracks of keypoints (regions) across views
- 3D locations and/or surface normals

Keypoint description

[Lowe '99]

- Thresholded image gradients are sampled over 16x16 array of locations in scale space
- Create array of orientation histograms
- 8 orientations x 4x4 histogram array = 128 dimensions



Representation

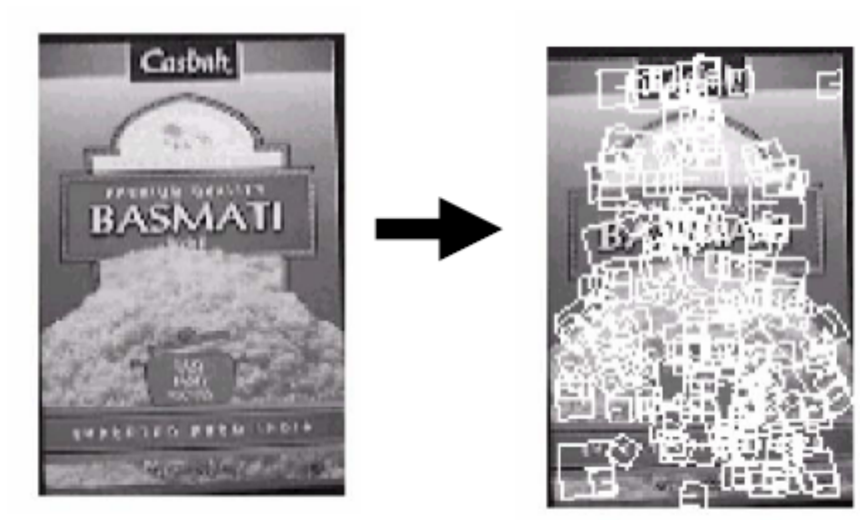
Interest points -- or Regions (group of interest points)

- Detection
  - Difference of Gaussian (DOG) [Lowe '99]
  - Harris-Laplacian [Mikolajczyk & Schmid '01]
  - Kadir-Brady [Kadir et al. '01]
  - Laplacian [Gårding & Lindeberg, '96]
- Adaptation [invariants]
  - Scale, rotation
  - Affine
- Description
  - SIFT
  - Color histograms

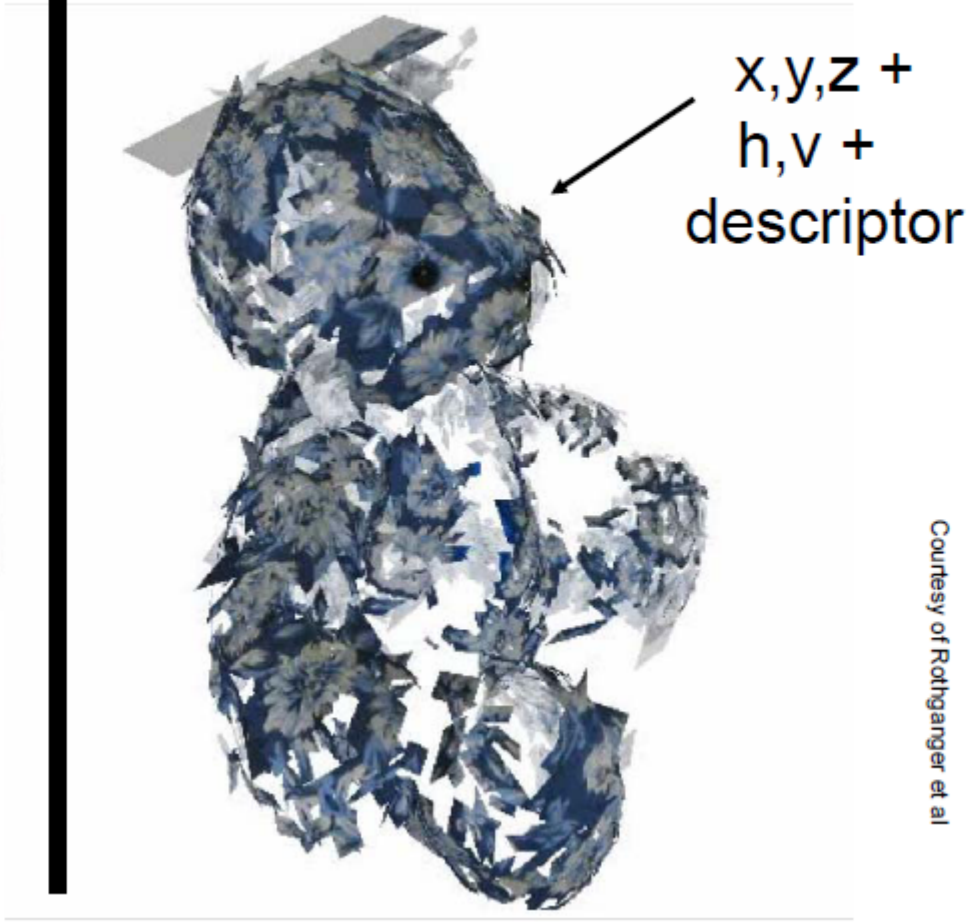
Geometrical constraints
- 3D locations and/or surface normals
- 2D spatial layout of keypoints [collections of views]
- Tracks of keypoints (regions) across views

Object representation: 2D or 3D location of key points

[Lowe '99]



Rothganger et al. '06



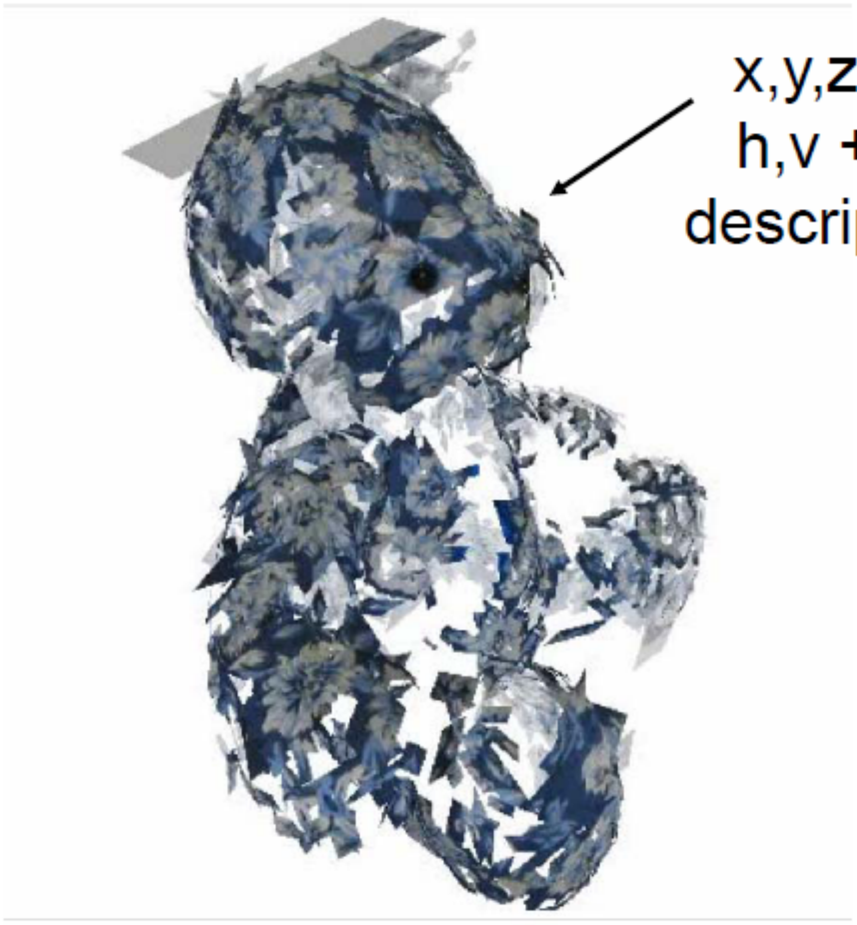
Courtesy of Rothganger et al

Basic scheme

- -Representation
  - -Features
  - -2D/3D Geometrical constraints
- -Model learning
- -Recognition
  - -hypothesis generation
  - -validation

Model learning

Rothganger et al. '03 '06



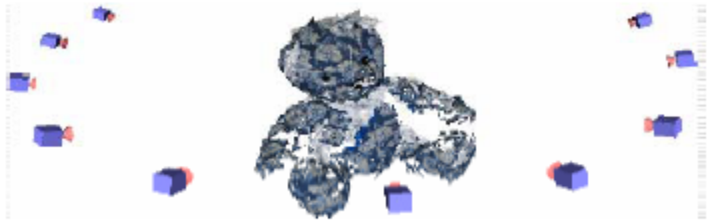
Courtesy of Rothganger et al

Model learning

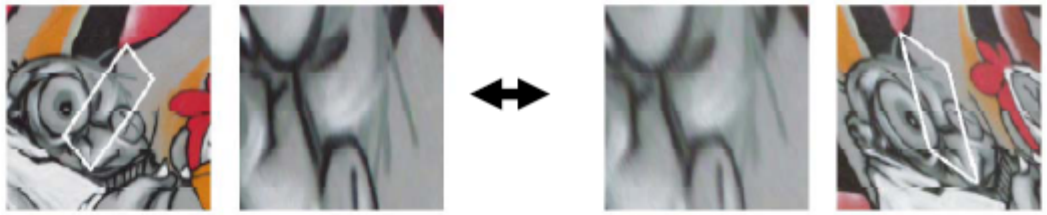
Rothganger et al. '03 '06

Build a 3D model:

- N images of object from N different view points



- Match key points between consecutive views
[create sample set]

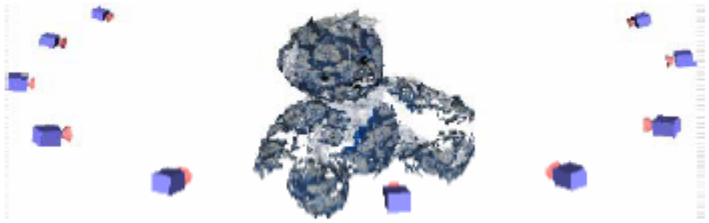


- Use affine structure from motion to compute 3D location and orientation + camera locations

- Affine factorization Tomasi & Kanade '92
- RANSAC
- 2 matches are needed rather than 4 thanks to affine invariant patches

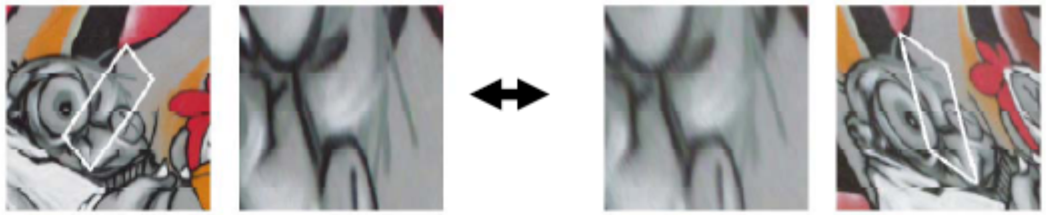
Model learning Rothganger et al. '03 '06

Build a 3D model:



• N images of object from N different view points

• Match key points between consecutive views
[create sample set]



Use affine structure from motion to compute 3D location and orientation + camera locations

• Affine factorization Tomasi & Kanade '92
• RANSAC
• 2 matches are needed rather than 4 thanks to affine invariant patches

• Find connected components

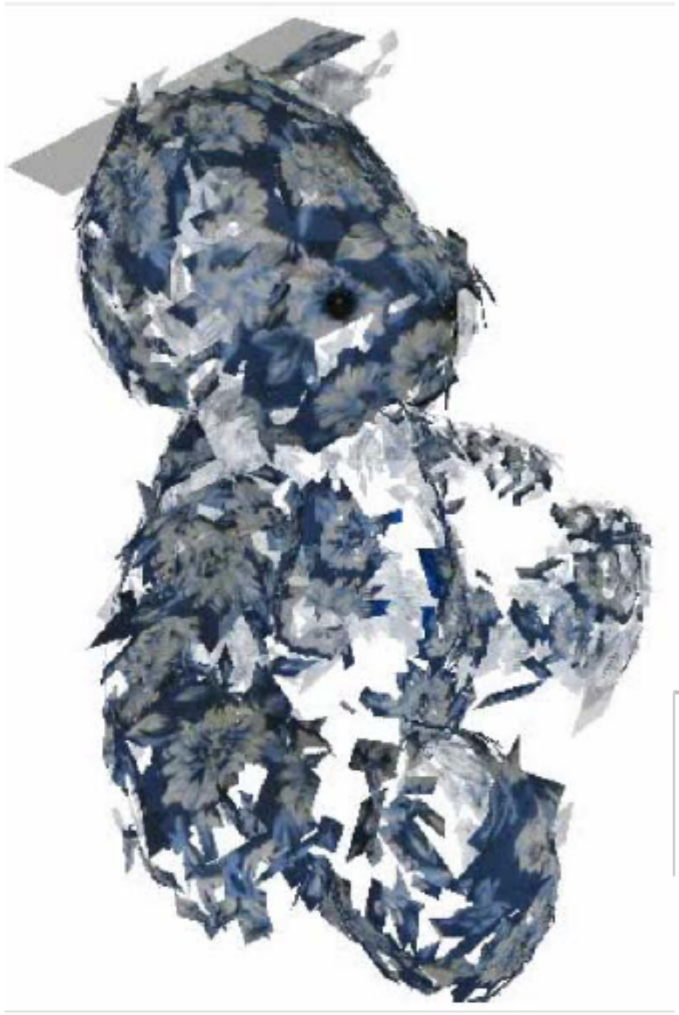
• Use bundle adjustment to refine model

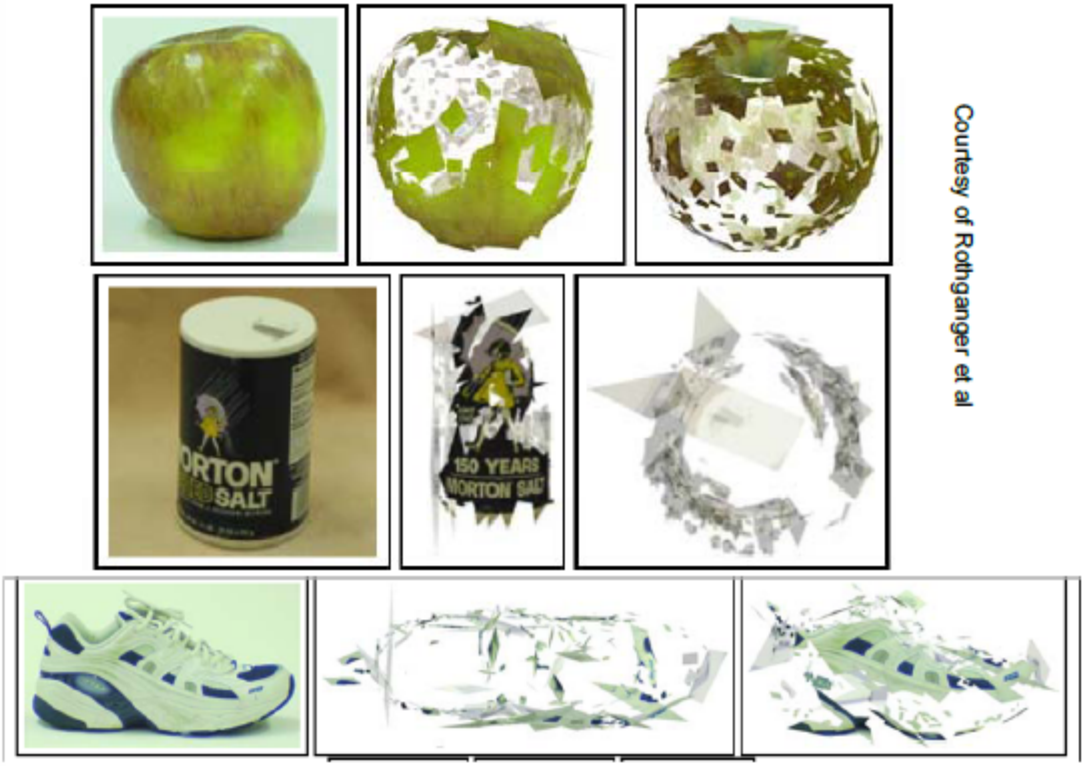
$$E = \sum_{j=1}^{n} \sum_{i \in I_j} |\mathcal{S}_{ij} - \mathcal{M}_i \mathcal{N}_j|^2,$$

• Upgrade model to Euclidean assuming zero skew and square pixels

Learnt models

Rothganger et al. '03 '06

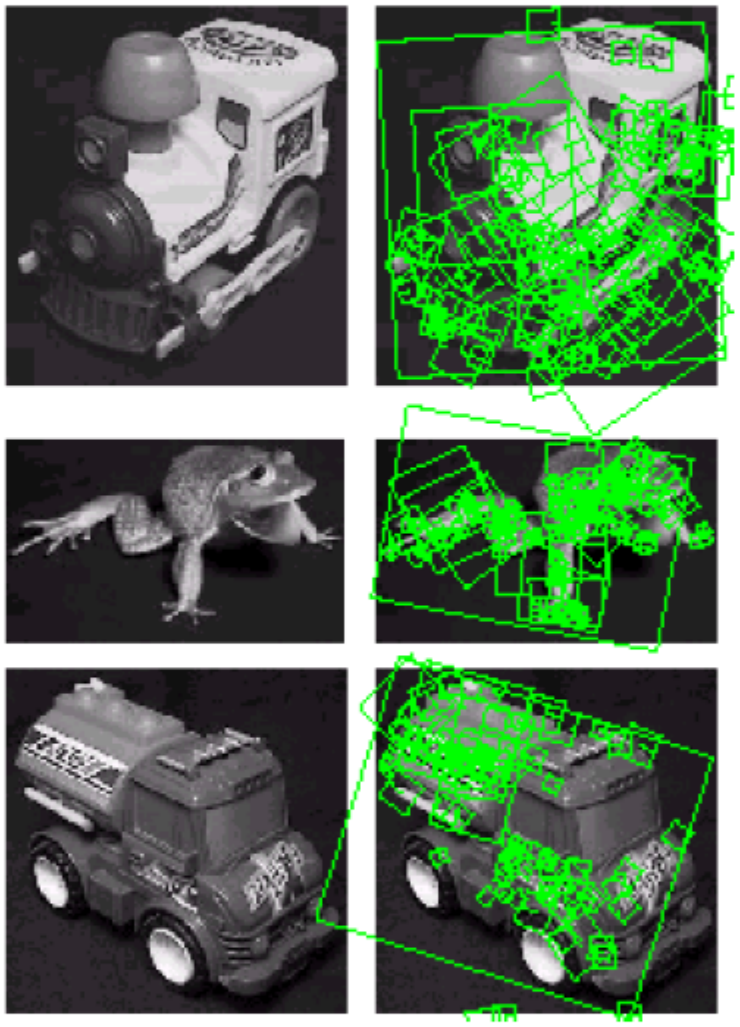




Courtesy of Rothganger et al

Learnt models

[Lowe '99]



Basic scheme

- -Representation
  - -Features
  - -2D/3D Geometrical constraints
- -Model learning
- -Recognition [object instance from object model]
  - -hypothesis generation
  - -Model verification

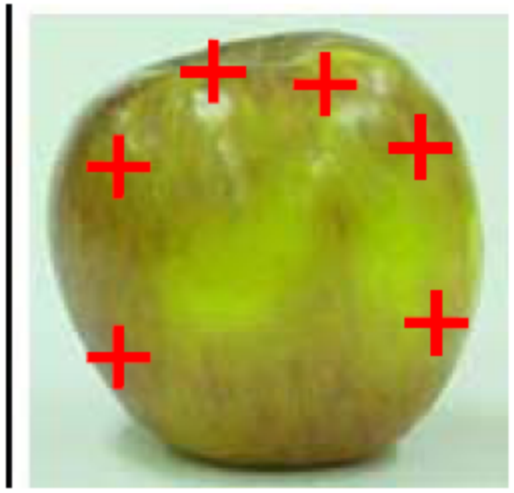
Recognition

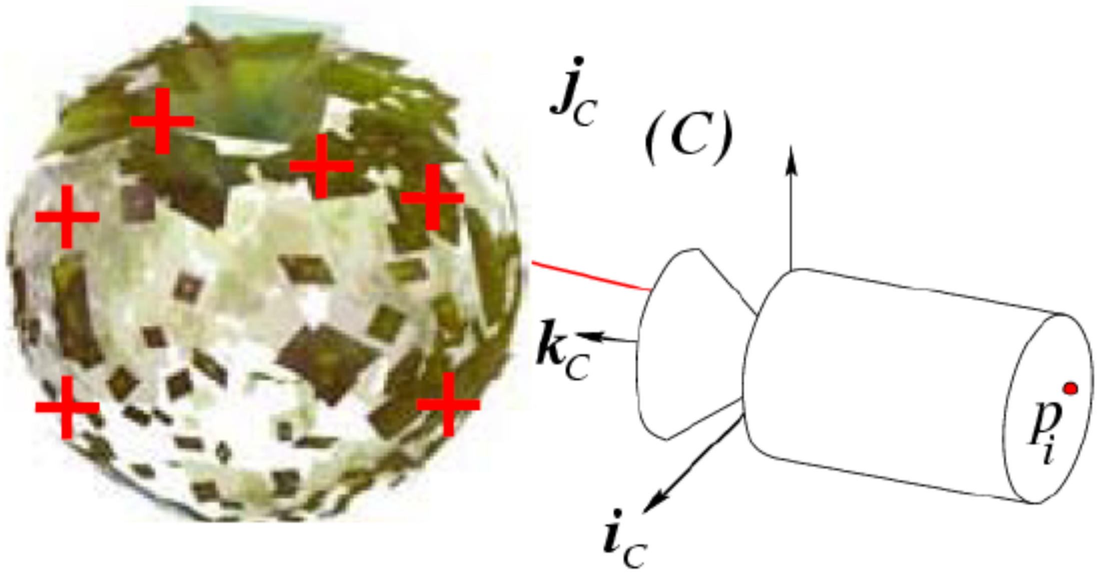
Goal: given a query image I, identify object model in the image I (match learned model to I)

- Generate hypothesis
- Verify hypothesis
- Select hypothesis with lowest fitting error
- Generate recognition results

Recognition

[Rothganger et al. ‘03 ’06]

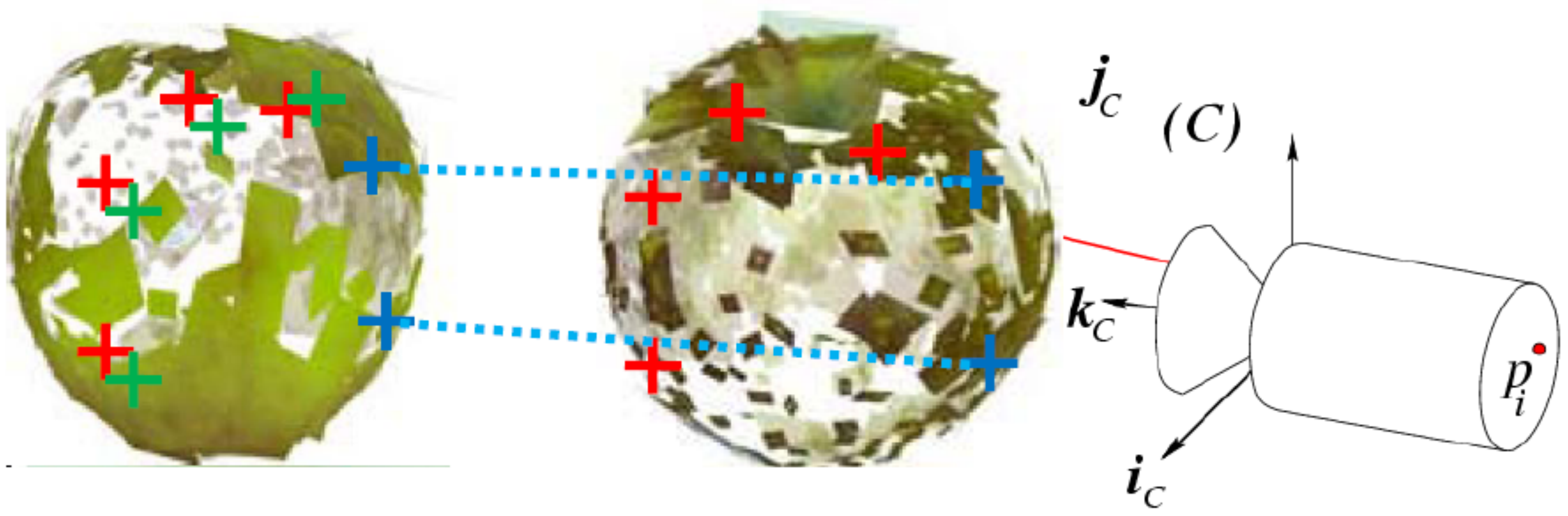




1. Find matches between model and test image features

Recognition

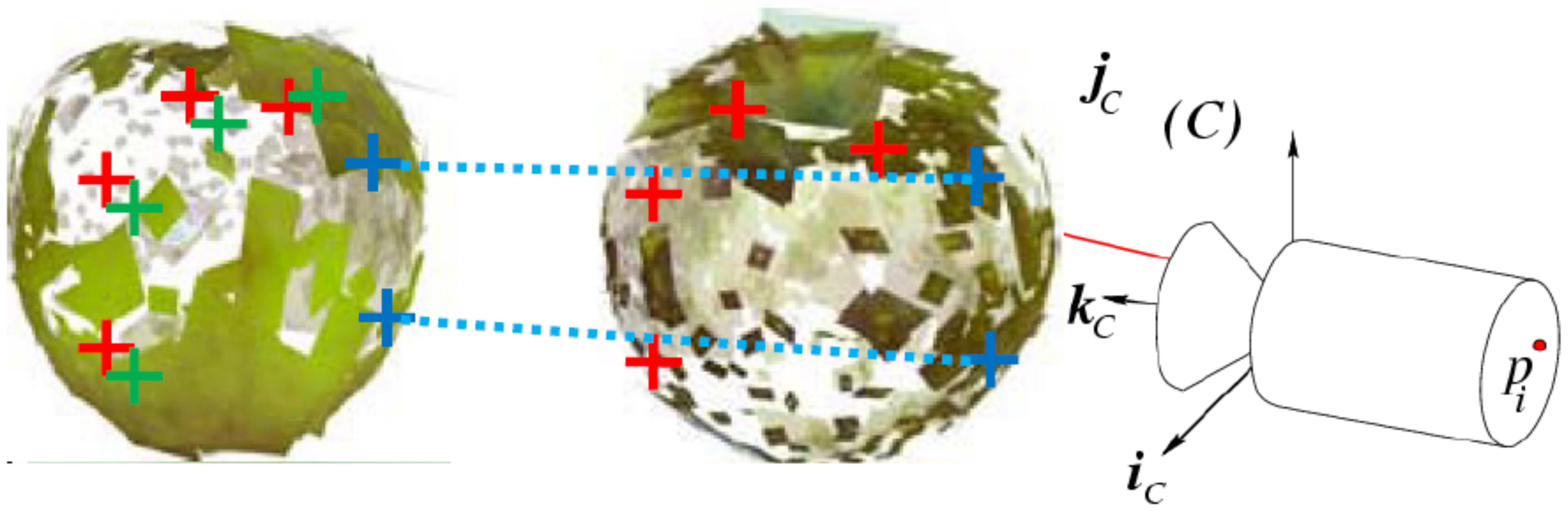
[Rothganger et al. '03 '06]



1. Find matches between model and test image features
2. Generate hypothesis:
   - Compute transformation M from N matches (N=2; affine camera; affine key points)
3. Model verification
   - Use M to project other matched 3D model features into test image
   - Compute residual = D(projections, measurements)

Recognition

[Rothganger et al. '03 '06]



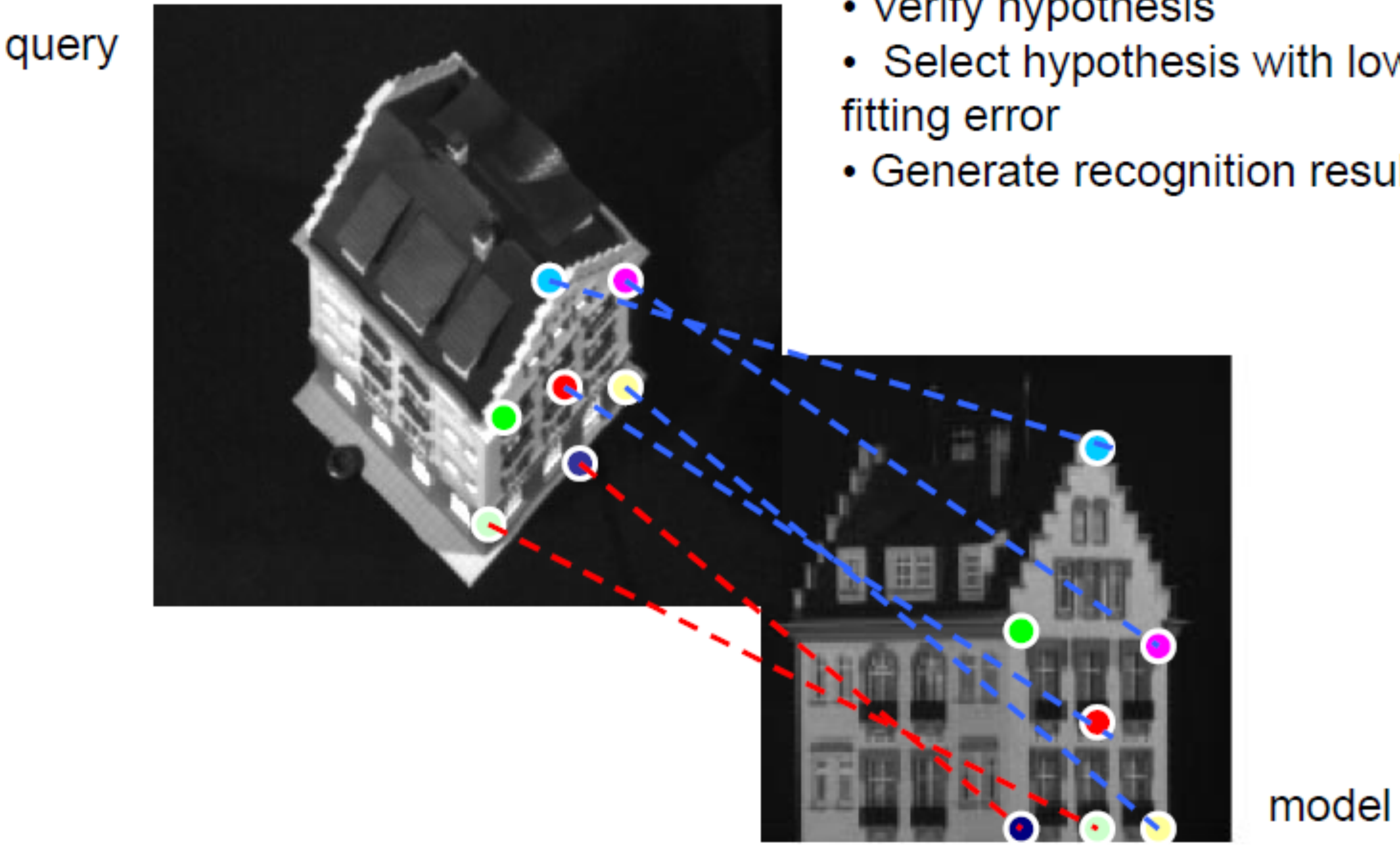
Goal:

Estimate (fit) the best M in presence of outliers

Recognition

Goal: given a query image I, identify object model in the image I (match learned model to I)

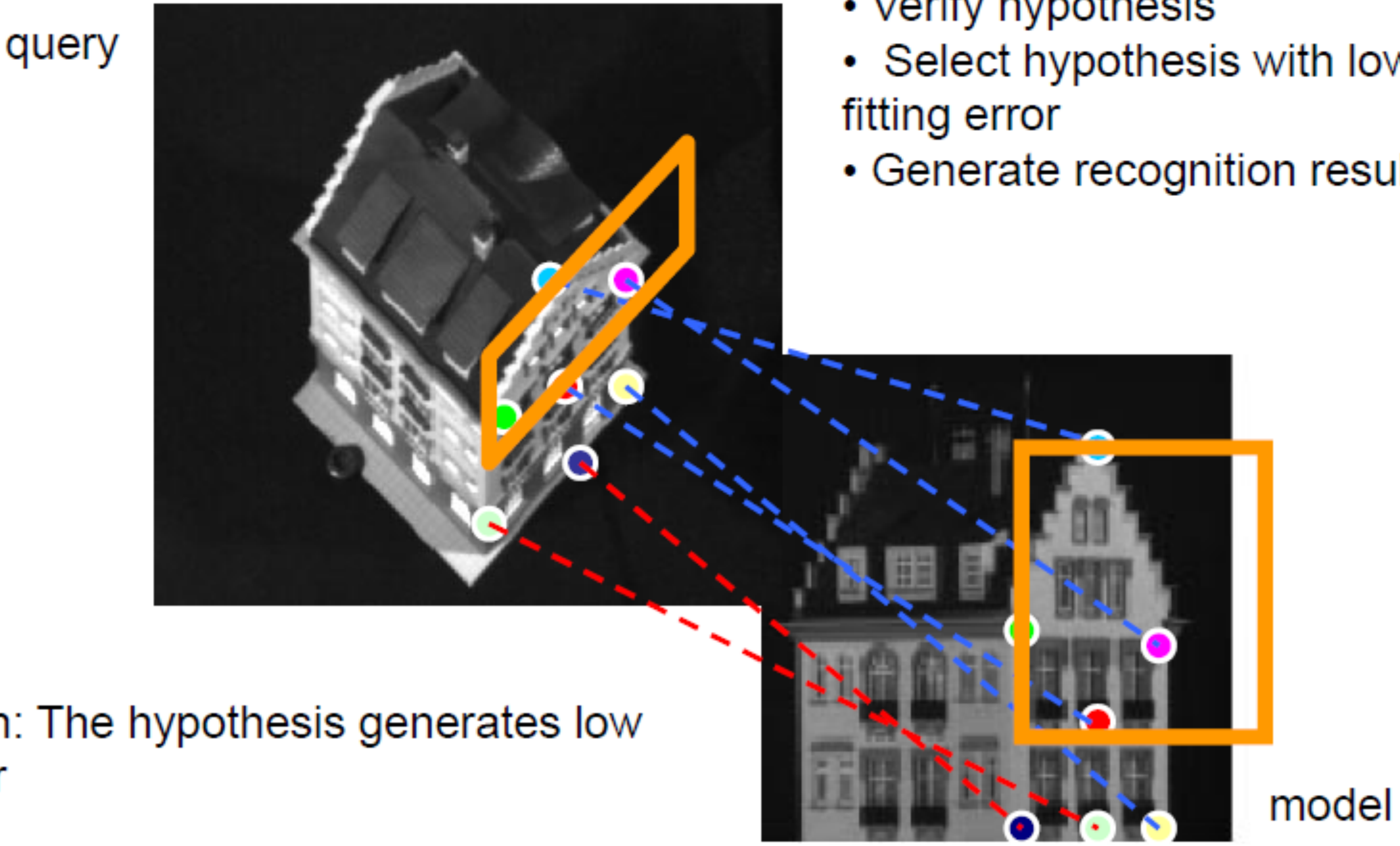
- Generate hypothesis
- Verify hypothesis
- Select hypothesis with lowest fitting error
- Generate recognition results



Recognition

Goal: given a query image I, identify object model in the image I (match learned model to I)

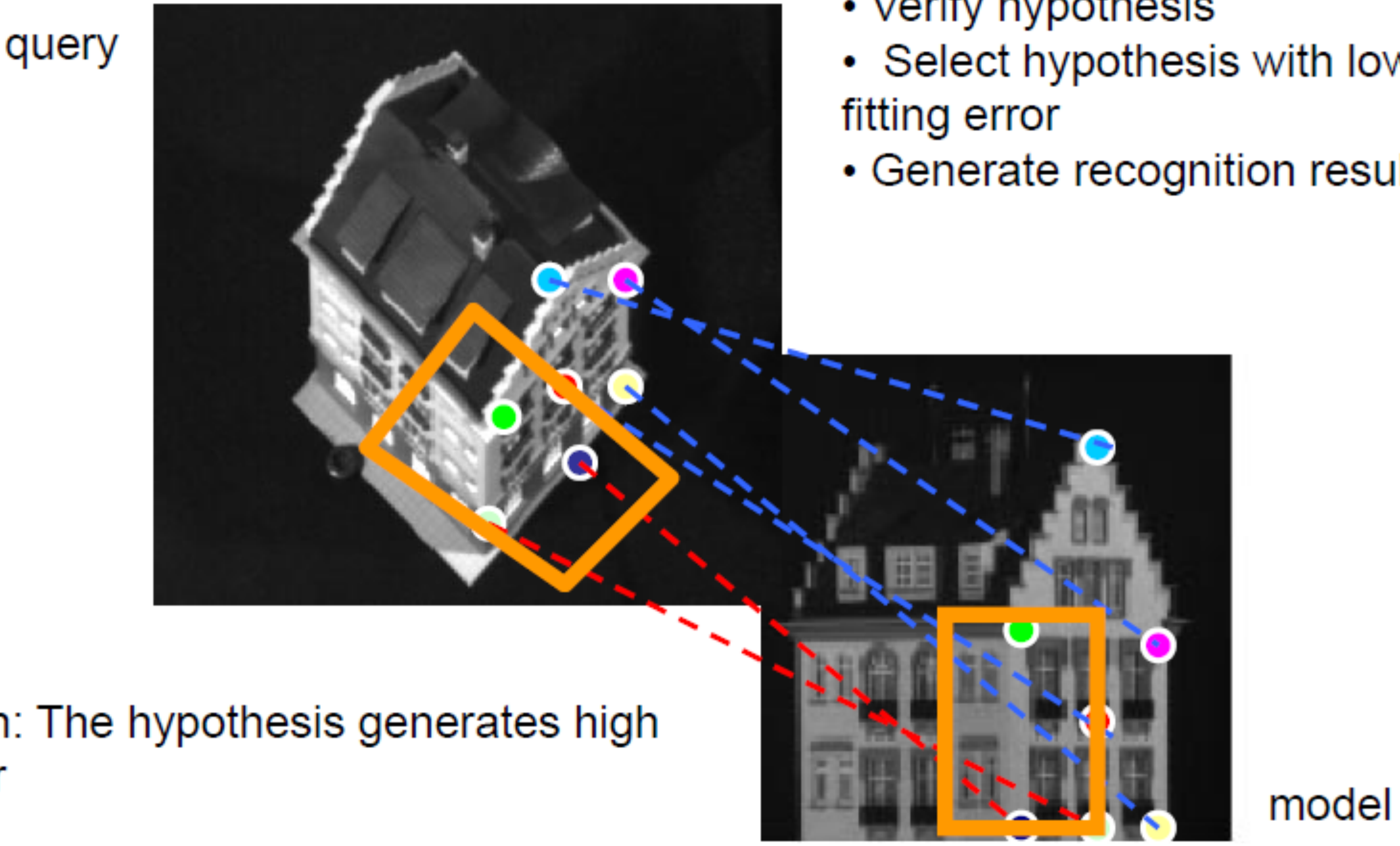
- Generate hypothesis
- Verify hypothesis
- Select hypothesis with lowest fitting error
- Generate recognition results



Verification: The hypothesis generates low fitting error

Recognition

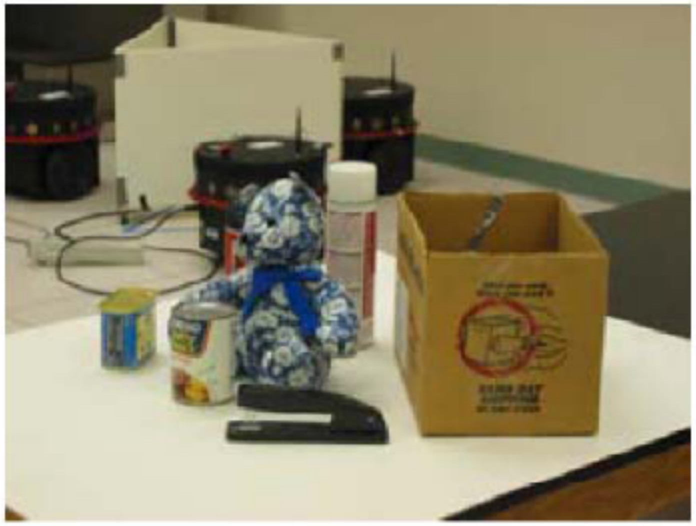
Goal: given a query image I, identify object model in the image I (match learned model to I)



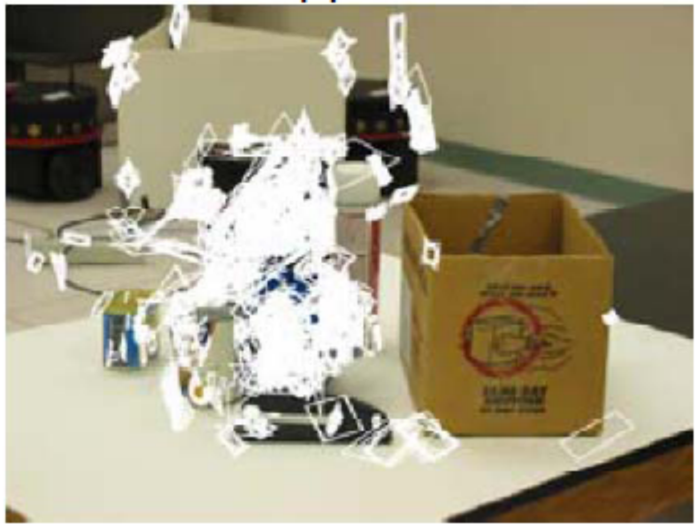
- Generate hypothesis
- Verify hypothesis
- Select hypothesis with lowest fitting error
- Generate recognition results

Verification: The hypothesis generates high fitting error

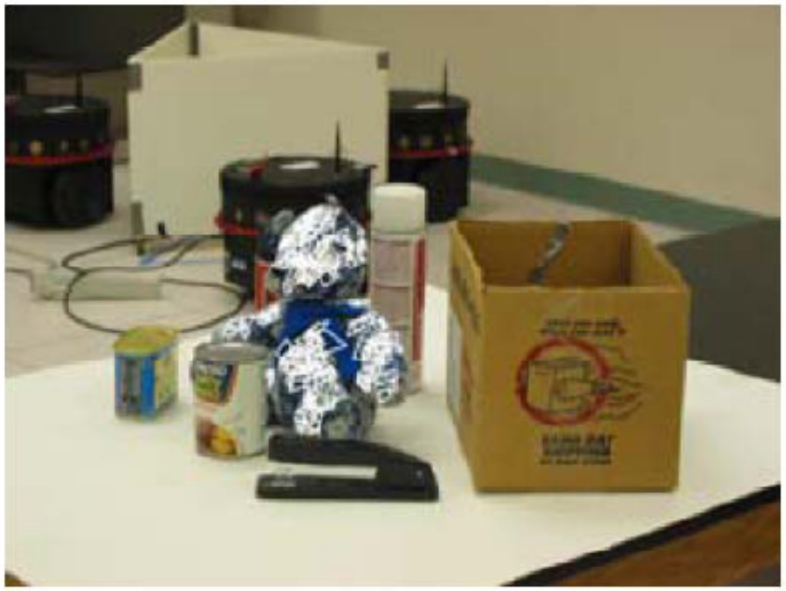
Object to recognize



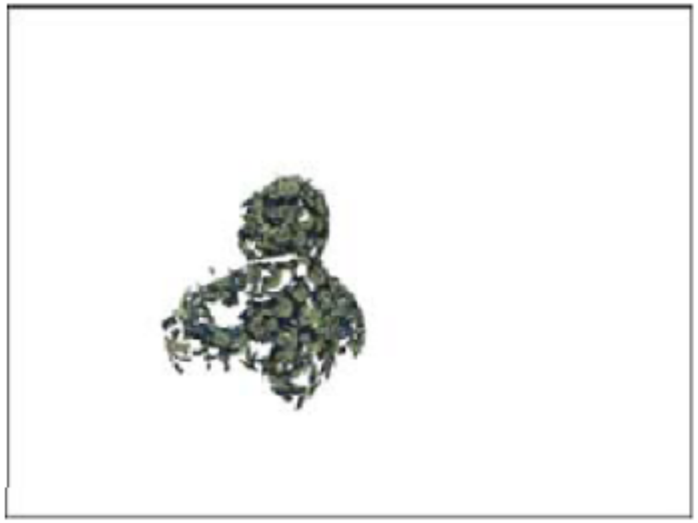
Initial matches based on appearance



Matches verified with geometrical constraints



Recovered pose



Courtesy of Rothganger et al

Basic scheme

-Representation
- -Features
- -2D/3D Geometrical constraints

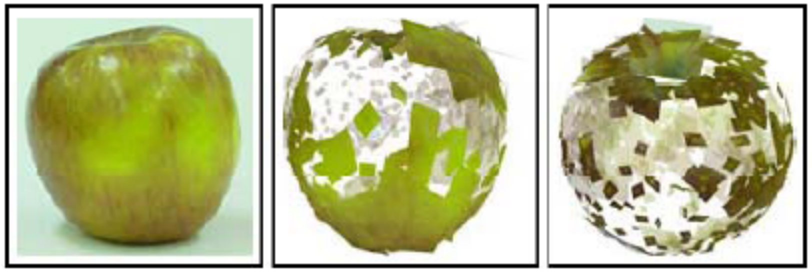
-Model learning

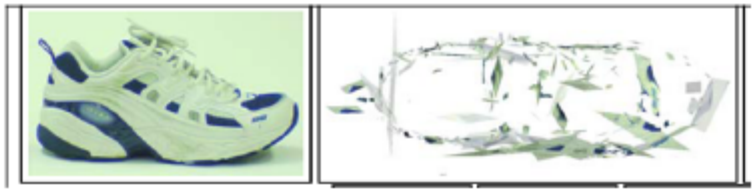
Let's see some results!

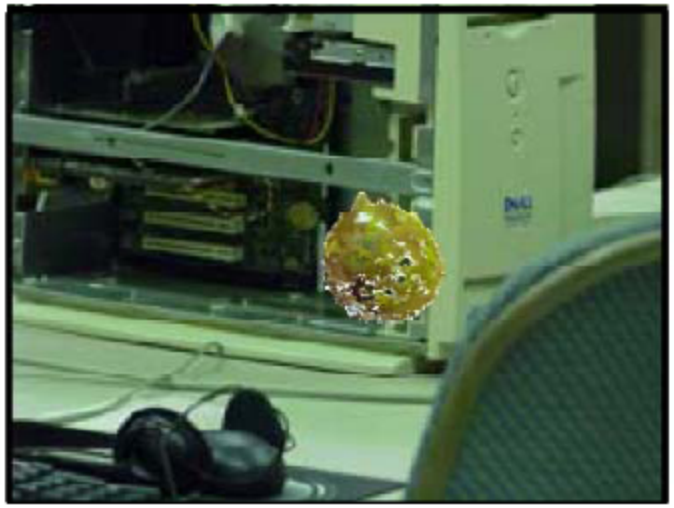
-Recognition [object instance from a single image]
- -hypothesis generation
- -Model verification

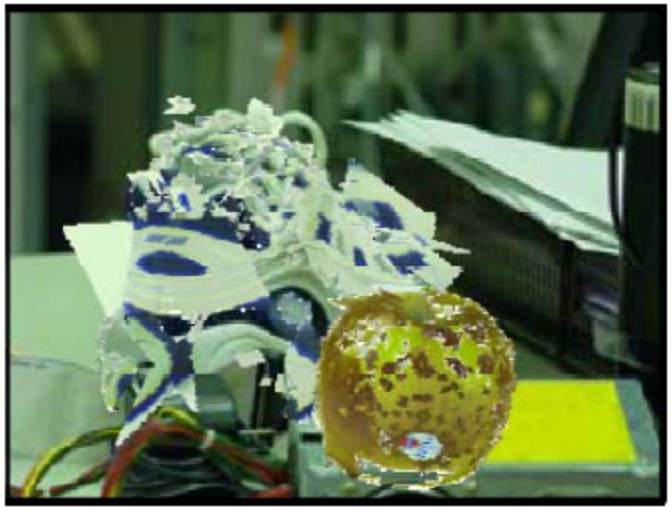
3D Object Recognition results

Rothganger et al. '03 '06







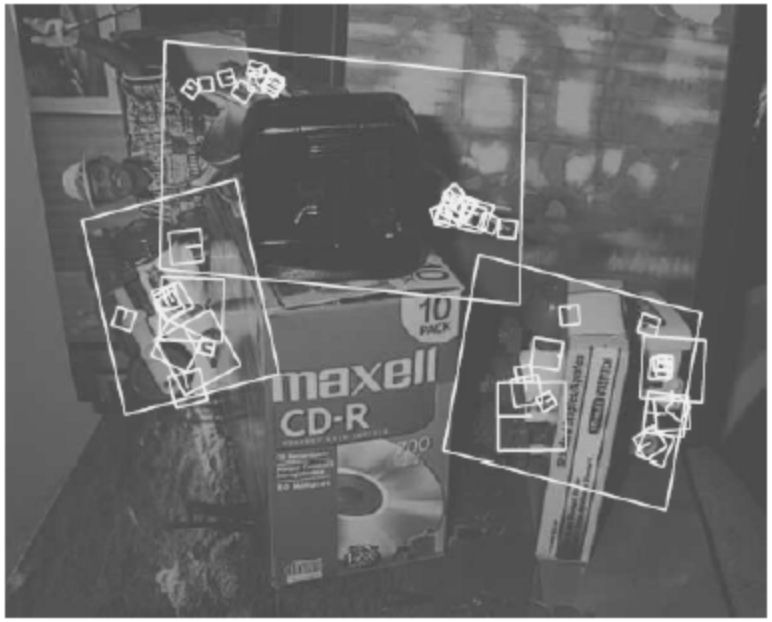


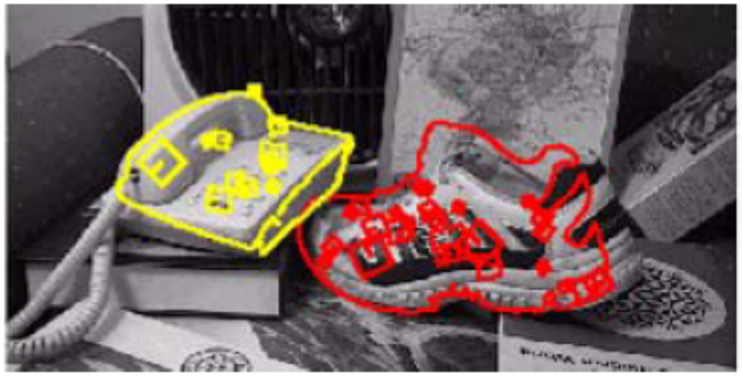
Courtesy of Rothganger et al

•Handle severe clutter

3D Object Recognition results

Lowe. '99, '04





Courtesy of D. Lowe

- •Handle severe occlusions
- •Fast!

3D Object Recognition results

[Ferrari et al '04]

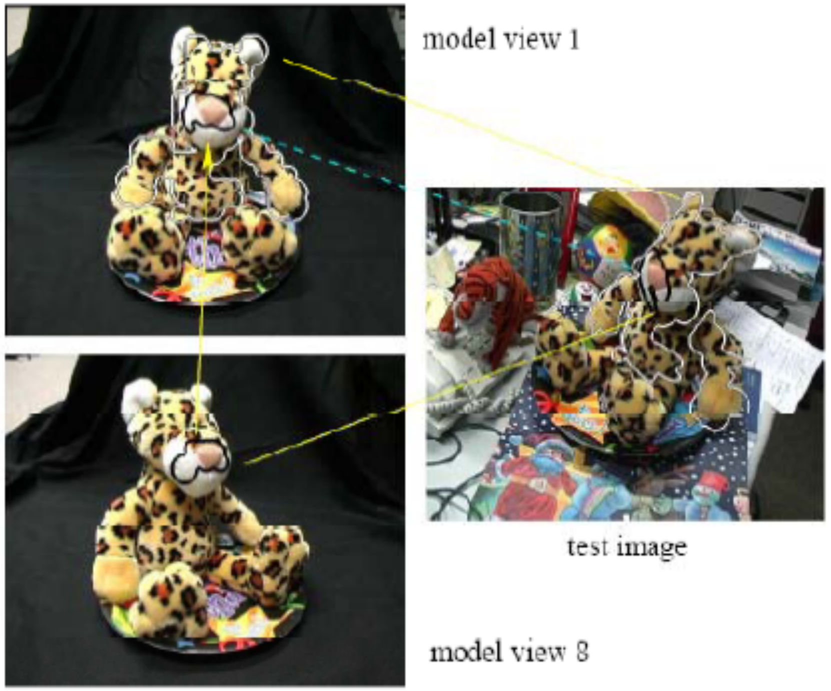
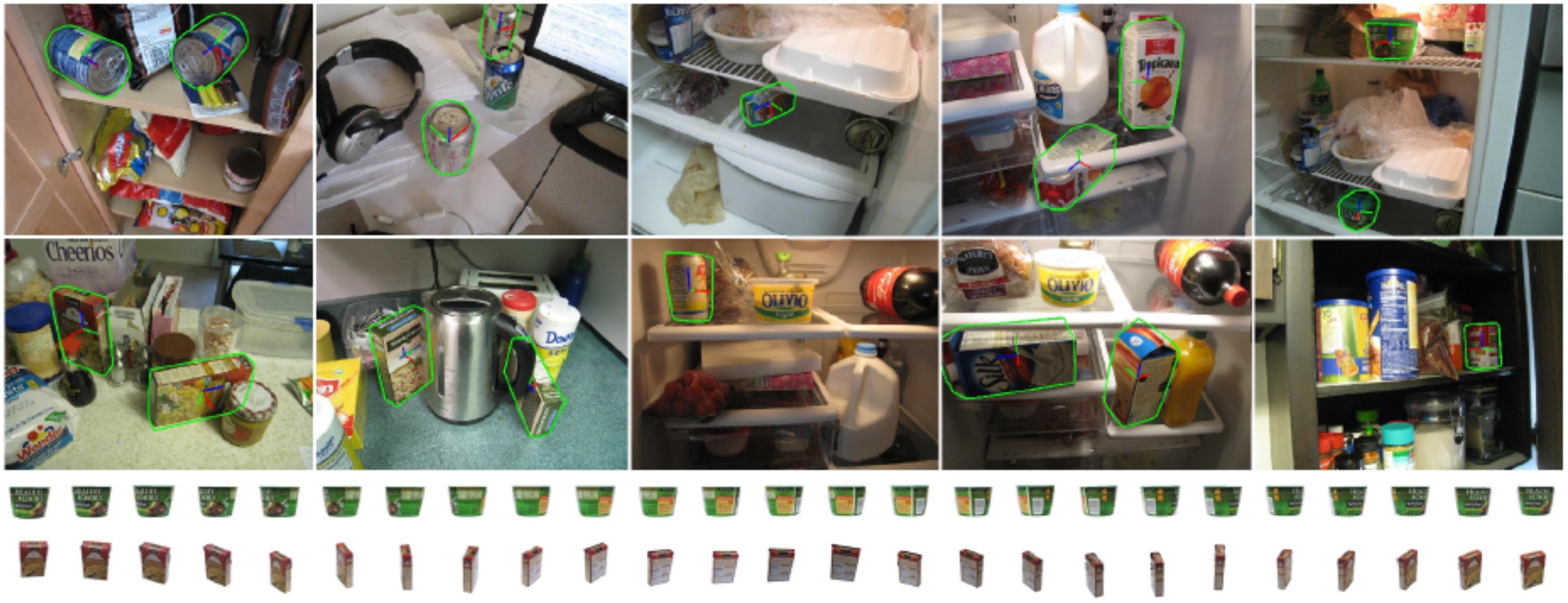


Figure 17: *Two compatible (and correct) GAMs. The nose GAM (black) is initially matched from model view 8, and is transferred to model view 1. Note how the other GAM (white) is very large and covers the head, arms and chest. A GAM can extend over multiple facets when the combination of viewpoints and surface orientations make the affine transformations of the region matches vary smoothly even across facet edges. In these cases, the resulting GAMs are larger and therefore more reliable and relevant.*

Courtesy of Ferrari et al

3D Object Recognition results

Edward Hsiao, Alvaro Collet and Martial Hebert. **Making specific features less discriminative to improve point-based 3D object recognition**. *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, June, 2010.



Google Goggles

http://www.google.com/support/mobile/bin/answer.py?hl=en&answer=166331

Overview

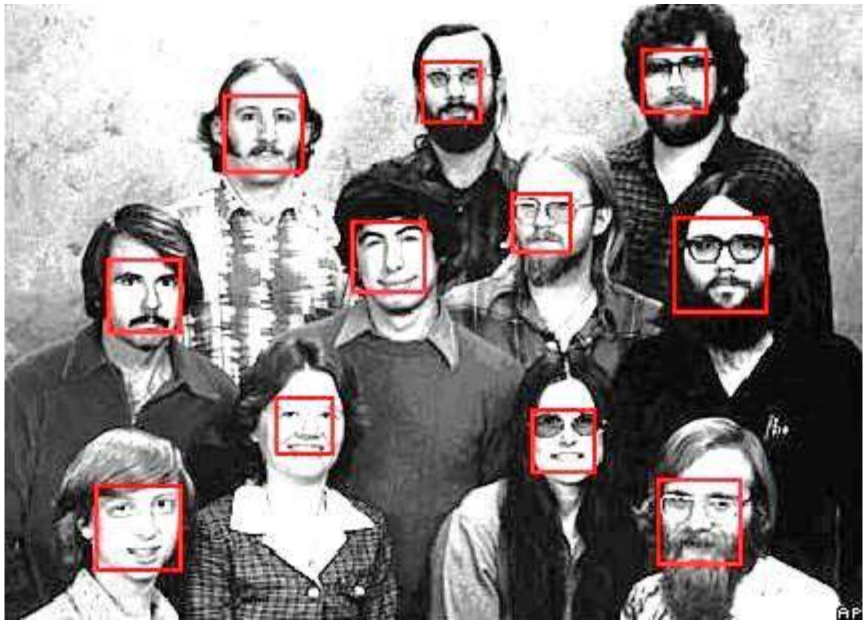
- Specific Object Recognition
  - Lowe ‘04
  - Rothganger et al. ‘03
- Face Dectection
  - Sub-space methods
  - Neural Network-based approaches
  - Viola & Jones
- Face Recognition

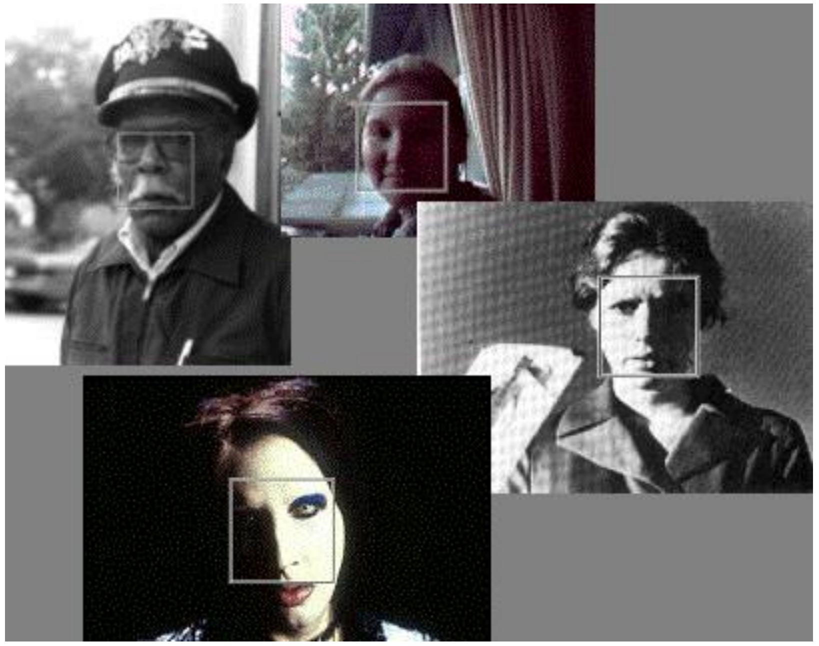
Face Detection & Recognition

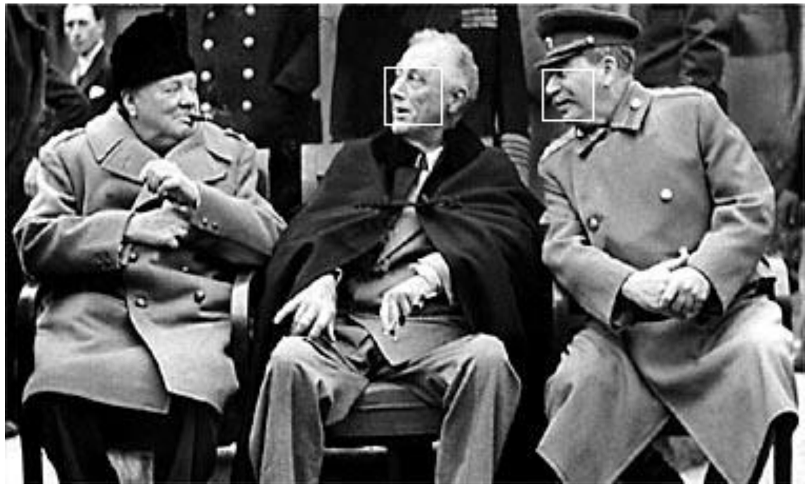
Slides from

P. Viola, S. Lazebnik, A. Torralba

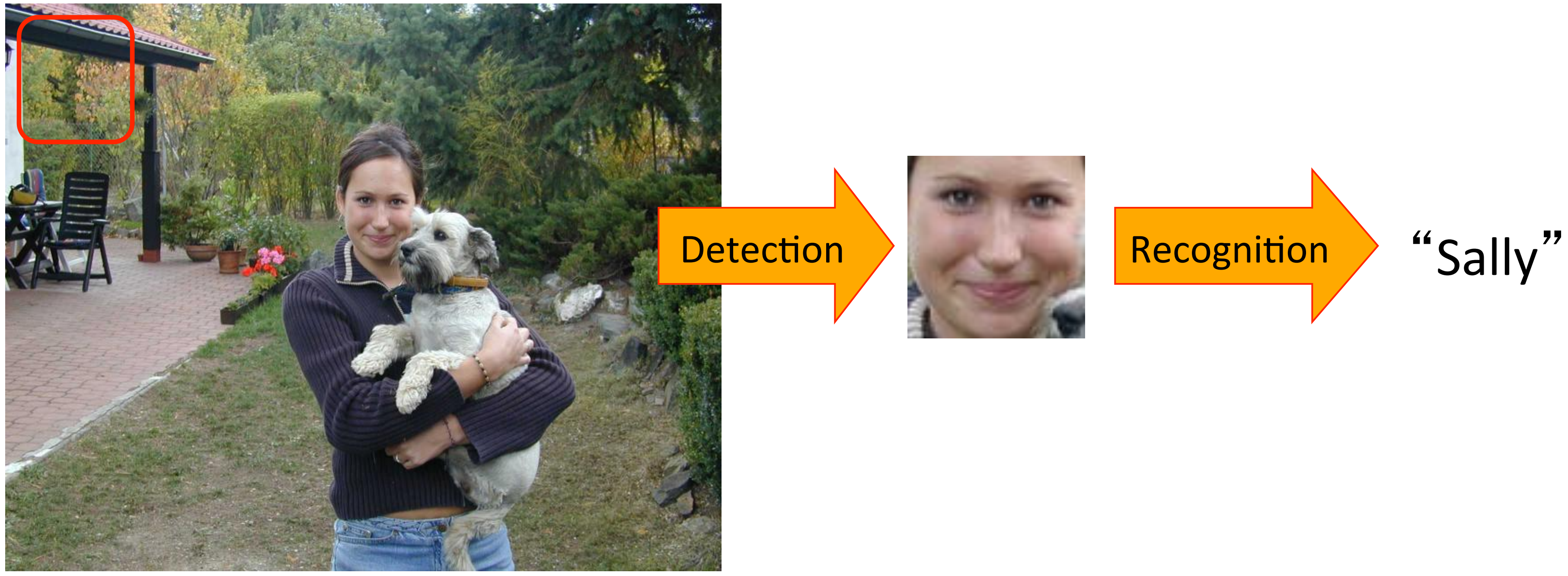
Face detection



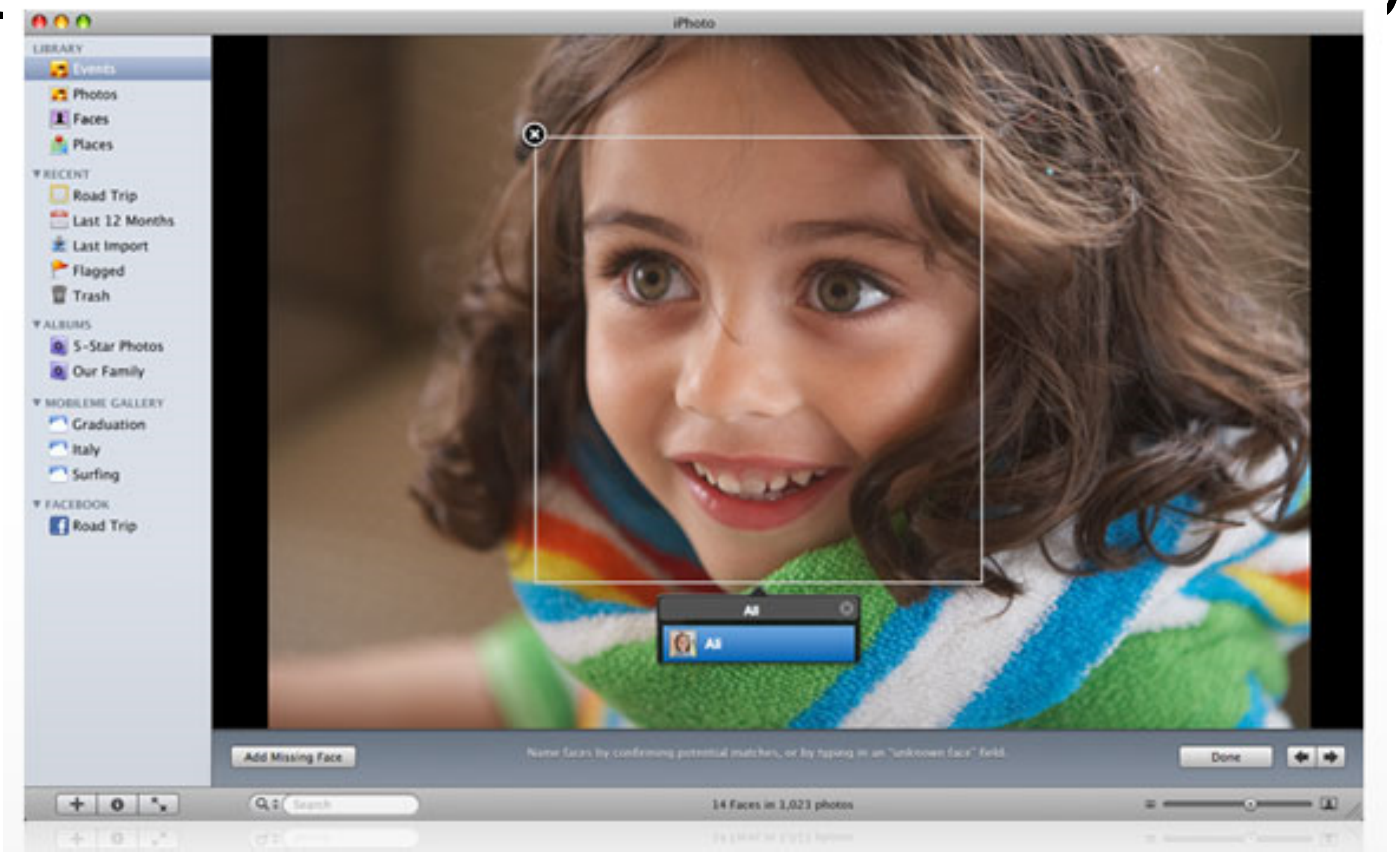




Face detection and recognition



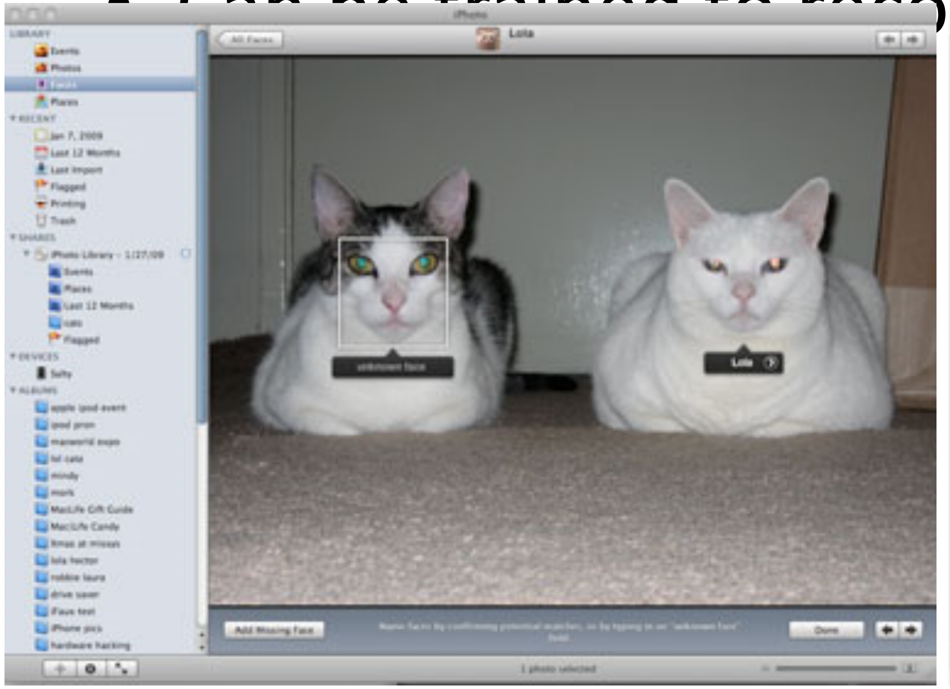
Consumer application: iPhoto 2009

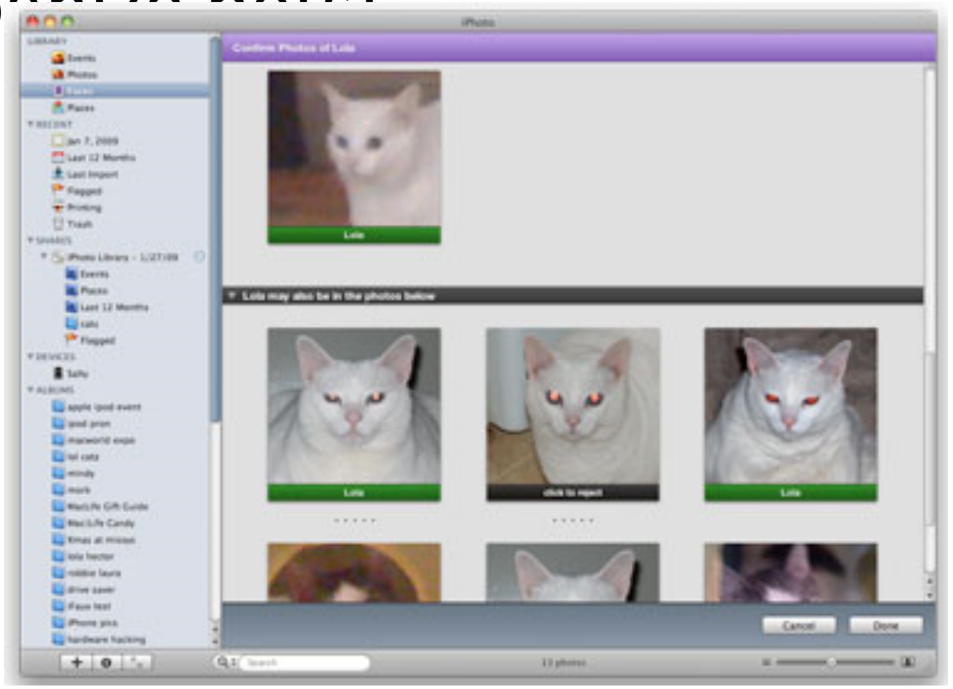


http://www.apple.com/ilife/iphoto/

Consumer application: iPhoto 2009

- Can be trained to recognize pets!





http://www.maclife.com/article/news/iphotos_faces_recognizes_cats

Challenges:

- Variability due to:
  - Intra-class variation (but fairly small)
  - Illumination
  - Occlusions (eyeglasses, facial hair, hair)

  - Fairly limited viewpoint (frontal, profile)

Practical Challenges of face detection

- Sliding window detector must evaluate tens of thousands of location/scale combinations
- Faces are rare: 0–10 per image
  - For computational efficiency, we should try to spend as little time as possible on the non-face windows
  - A megapixel image has ~10^6 pixels and a comparable number of candidate face locations
  - To avoid having a false positive in every image image, our false positive rate has to be less than 10^{-6}

Overview

- Specific Object Recognition
  - Lowe '04
  - Rothganger et al. '03
- Face Dectection
  - Sub-space methods
  - Neural Network-based approaches
  - Viola & Jones
- Face Recognition

Subspace Methods

- PCA (“Eigenfaces”, Turk and Pentland)
- PCA (Bayesian, Moghaddam and Pentland)
- LDA/FLD (“Fisherfaces”, Belhumeur & Kreigman)
- ICA

Principal Component Analysis

- Given: N data points $\mathbf{x_1}, \dots, \mathbf{x_N}$ in R^d

- We want to find a new set of features that are linear combinations of original ones:

$$u(\mathbf{x}_i) = \mathbf{u}^T(\mathbf{x}_i - \boldsymbol{\mu})$$

(**μ**: mean of data points)

- What unit vector **u** in R^d captures the most variance of the data?

Principal Component Analysis

- Direction that maximizes the variance of the projected data:

$$
\begin{aligned}
var(u) &= \frac{1}{N}\sum_{i=1}^{N} \mathbf{u}^{\mathrm{T}}\underbrace{(\mathbf{x}_i - \mu)(\mathbf{u}^{\mathrm{T}}}_{\text{Projection of data point}}(\mathbf{x}_i - \mu))^{\mathrm{T}} \\
&= \mathbf{u}^{\mathrm{T}}\underbrace{\Big[\sum_{i=1}^{N}(\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^{\mathrm{T}}\Big]}_{\text{Covariance matrix of data}}\mathbf{u} \\
&= \mathbf{u}^{\mathrm{T}}\Sigma\mathbf{u}
\end{aligned}
$$

The direction that maximizes the variance is the eigenvector associated with the largest eigenvalue of Σ

Eigenfaces: Key idea

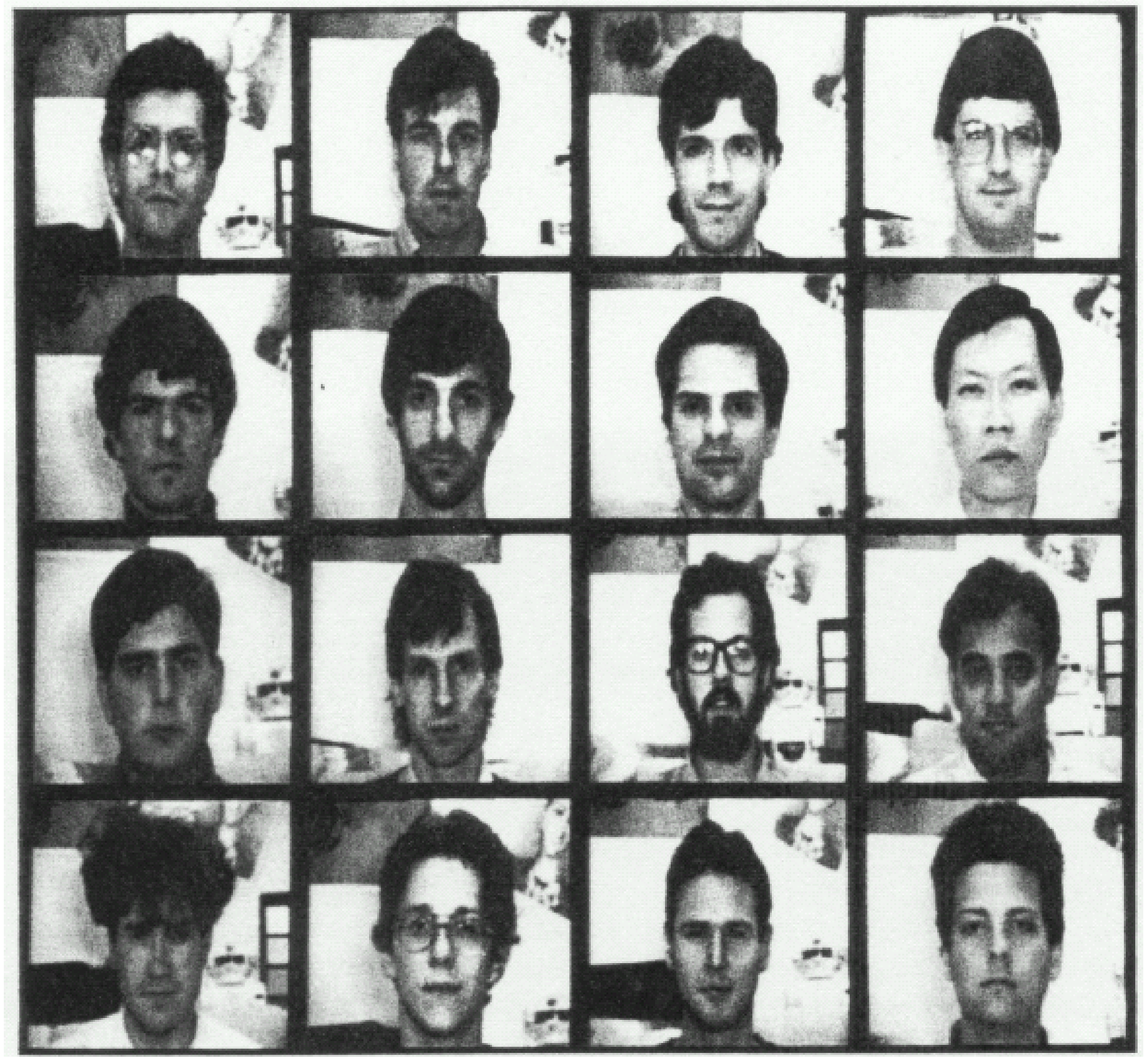
- Assume that most face images lie on a low-dimensional subspace determined by the first k ($k<d$) directions of maximum variance
- Use PCA to determine the vectors $\mathbf{u}_1,\ldots\mathbf{u}_k$ that span that subspace:
  $\mathbf{x} \approx \boldsymbol{\mu} + w_1\mathbf{u}_1 + w_2\mathbf{u}_2 + \ldots + w_k\mathbf{u}_k$
- Represent each face using its "face space" coordinates $(w_1,\ldots w_k)$
- Perform nearest-neighbor recognition in "face space"

M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

Eigenfaces example

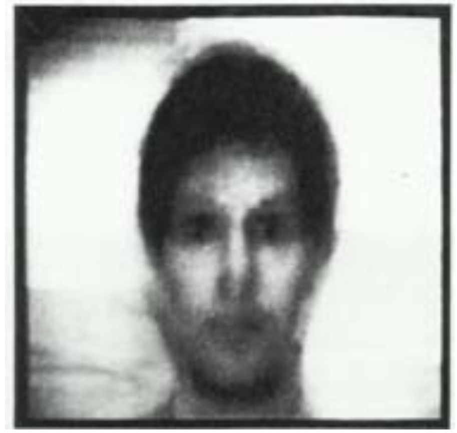
Training images

$\mathbf{x}_1,\ldots,\mathbf{x}_N$

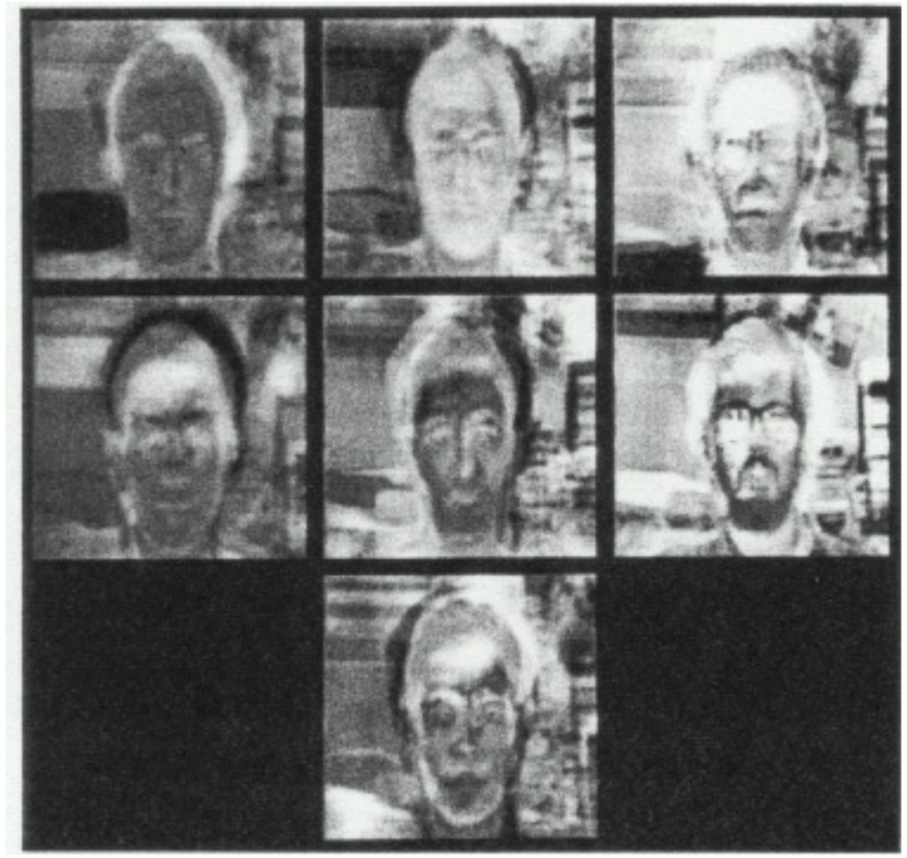


Eigenfaces example

Mean: **μ**



Top eigenvectors: $\mathbf{u}_1, \dots \mathbf{u}_k$



Eigenfaces example

- Face **x** in "face space" coordinates:

$$\mathbf{x} \rightarrow [\mathbf{u}_1^{\mathrm{T}}(\mathbf{x} - \mu), \ldots, \mathbf{u}_k^{\mathrm{T}}(\mathbf{x} - \mu)]$$

$$= \quad w_1, \ldots, w_k$$

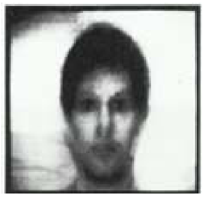
Eigenfaces example

- Face **x** in "face space" coordinates:

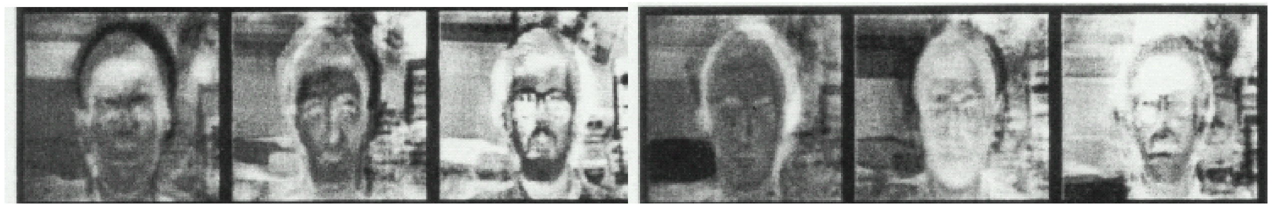
$$\mathbf{x} \rightarrow [\mathbf{u}_1^{\mathrm{T}}(\mathbf{x}-\mu), \ldots, \mathbf{u}_k^{\mathrm{T}}(\mathbf{x}-\mu)]$$
$$= \quad w_1, \ldots, w_k$$

- Reconstruction:

=



+



$\hat{\mathbf{x}} \quad = \quad \boldsymbol{\mu} \quad + \quad w_1\mathbf{u}_1 + w_2\mathbf{u}_2 + w_3\mathbf{u}_3 + w_4\mathbf{u}_4 + \ldots$

Summary: Recognition with eigenfaces

Process labeled training images:

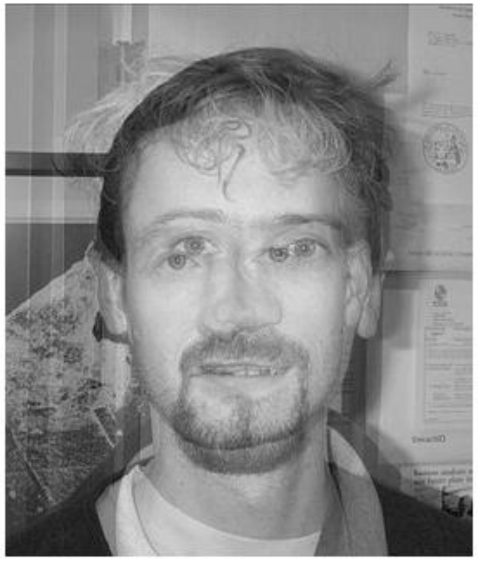
- Find mean $\mathbf{\mu}$ and covariance matrix $\mathbf{\Sigma}$
- Find k principal components (eigenvectors of $\mathbf{\Sigma}$) $\mathbf{u}_1,\ldots\mathbf{u}_k$
- Project each training image $\mathbf{x}_i$ onto subspace spanned by principal components:
  $(w_{i1},\ldots,w_{ik}) = (\mathbf{u}_1^T(\mathbf{x}_i - \mathbf{\mu}), \ldots, \mathbf{u}_k^T(\mathbf{x}_i - \mathbf{\mu}))$

Given novel image $\mathbf{x}$:

- Project onto subspace:
  $(w_1,\ldots,w_k) = (\mathbf{u}_1^T(\mathbf{x} - \mathbf{\mu}), \ldots, \mathbf{u}_k^T(\mathbf{x} - \mathbf{\mu}))$
- Optional: check reconstruction error $\mathbf{x} - \hat{\mathbf{x}}$ to determine whether image is really a face
- Classify as closest training face in k-dimensional subspace

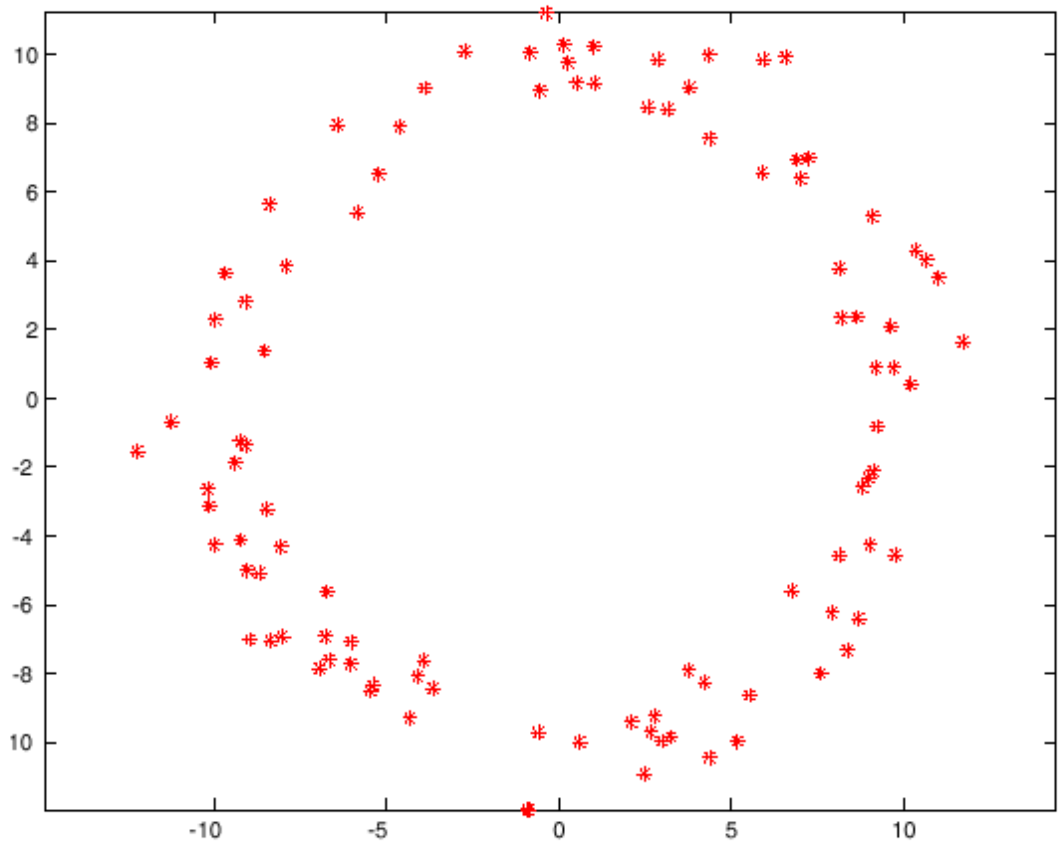
Limitations

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



Limitations

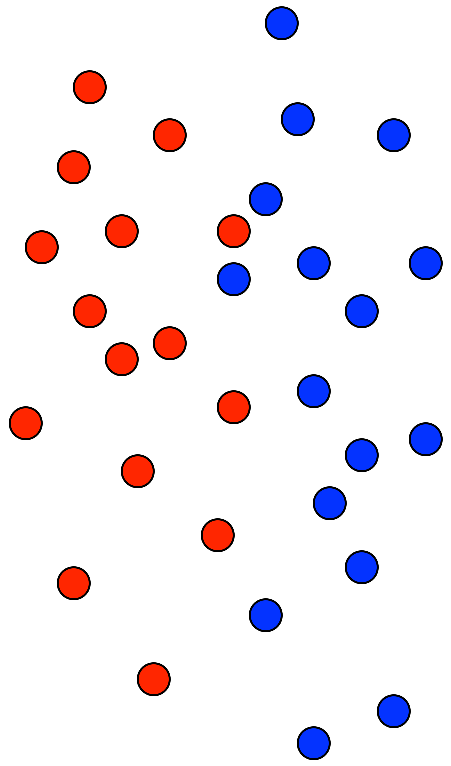
- PCA assumes that the data has a Gaussian distribution (mean μ, covariance matrix Σ)



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

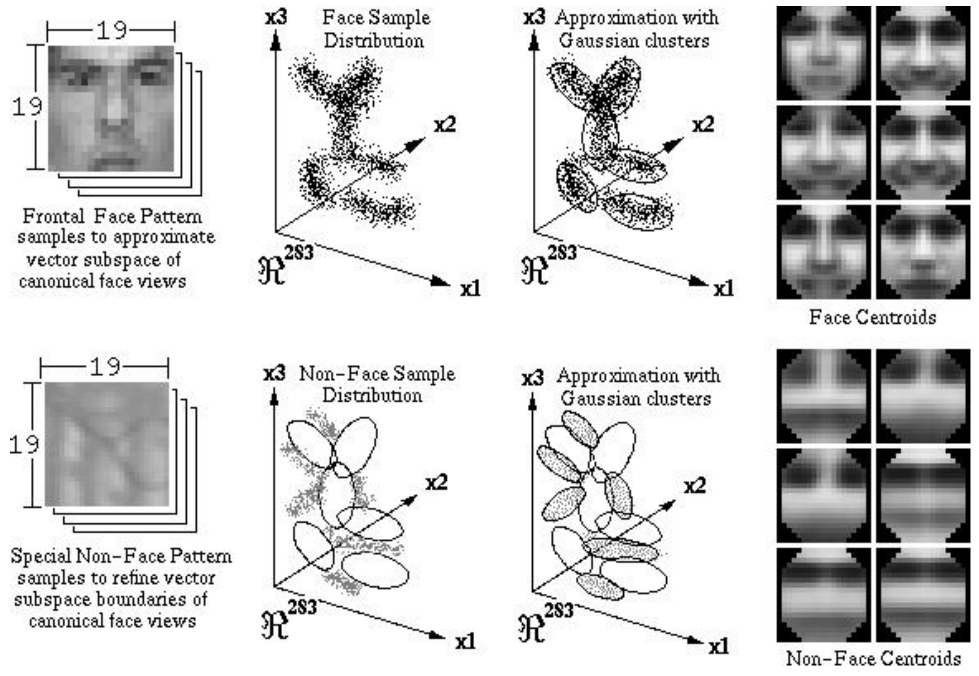
Limitations

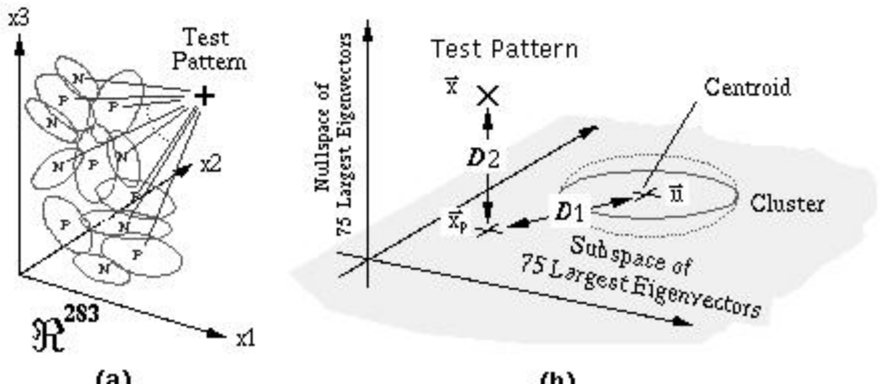
- The direction of maximum variance is not always good for classification



Distribution-Based Face Detector

- Learn face and nonface models from examples [Sung and Poggio 95]
- Cluster and project the examples to a lower dimensional space using Gaussian distributions and PCA
- Detect faces using distance metric to face and nonface clusters

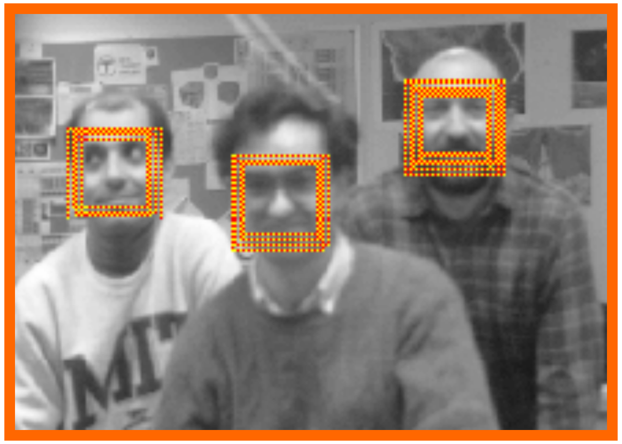

19
19
Frontal Face Pattern samples to approximate vector subspace of canonical face views
x3
Face Sample Distribution
x2
x1
$\Re^{283}$
Approximation with Gaussian clusters
Face Centroids
Special Non-Face Pattern samples to refine vector subspace boundaries of canonical face views
Non-Face Sample Distribution
Approximation with Gaussian clusters
Non-Face Centroids

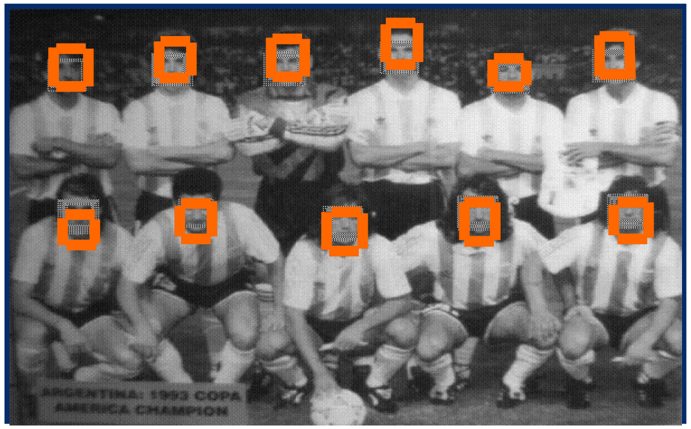


(a) (b)

Distribution-Based Face Detector

- Learn face and nonface models from examples [Sung and Poggio 95]





Training Database

1000+ Real, 3000+ *VIRTUAL*

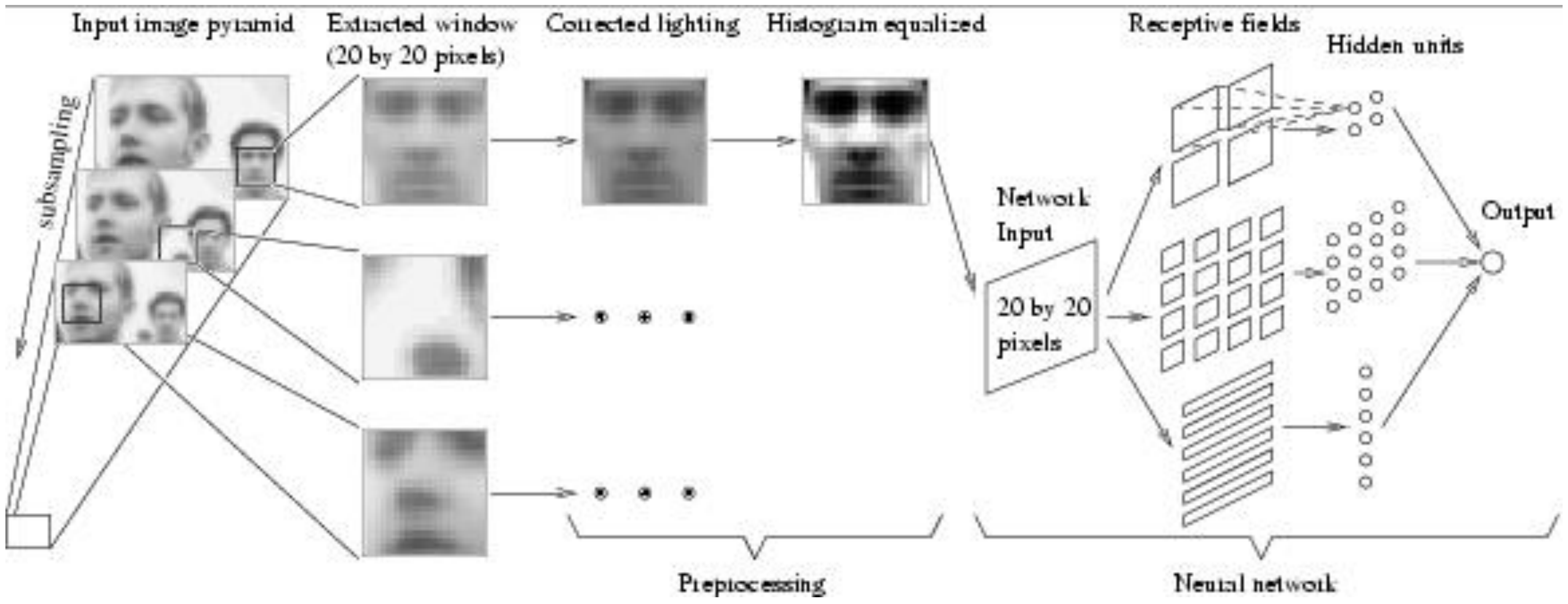
50,0000+ Non-Face Pattern

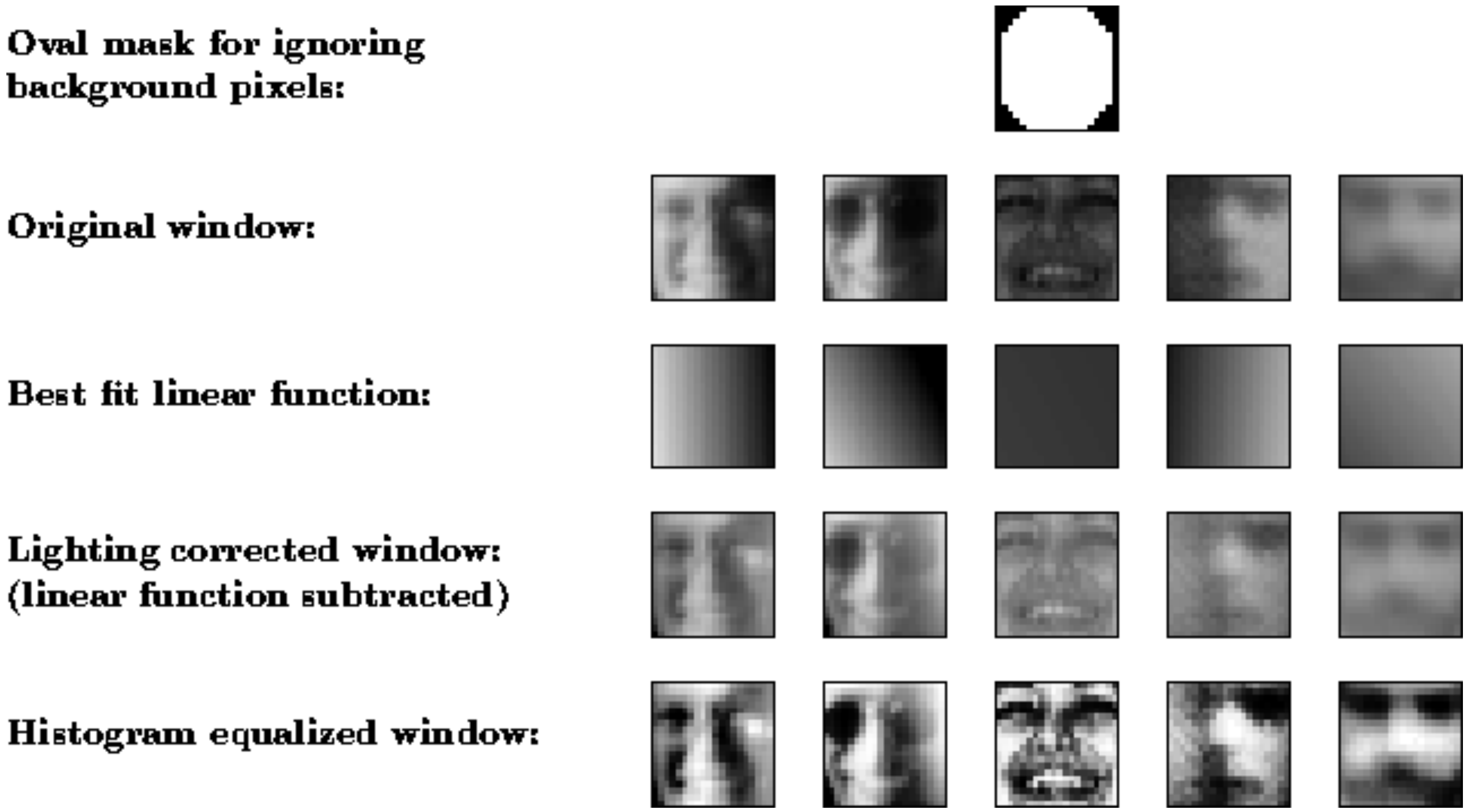
Overview

- Specific Object Recognition
  - Lowe '04
  - Rothganger et al. '03
- Face Dectection
  - Sub-space methods
  - Neural Network-based approaches
  - Viola & Jones
- Face Recognition

Neural Network-Based Face Detector

- Train a set of multilayer perceptrons and arbitrate a decision among all outputs [Rowley et al. 98]




Oval mask for ignoring background pixels:
Original window:
Best fit linear function:
Lighting corrected window: (linear function subtracted)
Histogram equalized window:

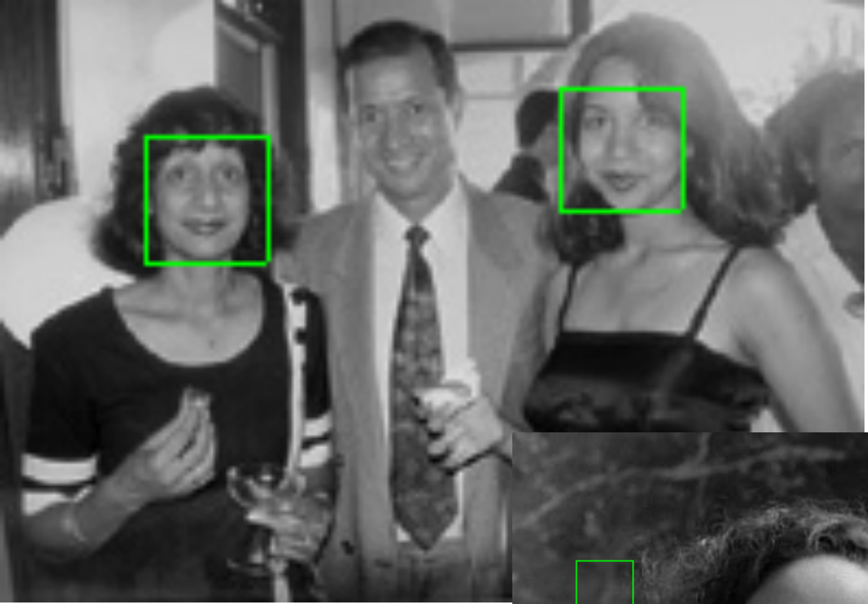
The steps in preprocessing a window. First, a linear function is fit to the intensity values in the window, and then subtracted out, correcting for some extreme lighting conditions. Then, histogram equalization is applied, to correct for different camera gains and to improve contrast. For each of these steps, the mapping is computed based on pixels inside the oval mask, while the mapping is applied to the entire window.

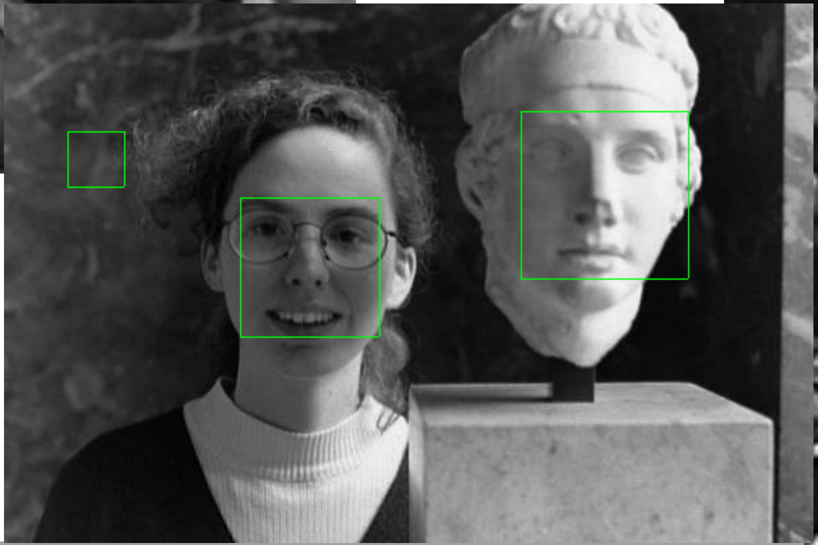
From: http://www.ius.cs.cmu.edu/IUS/har2/har/www/CMU-CS-95-158R/

Example CMU face detector results

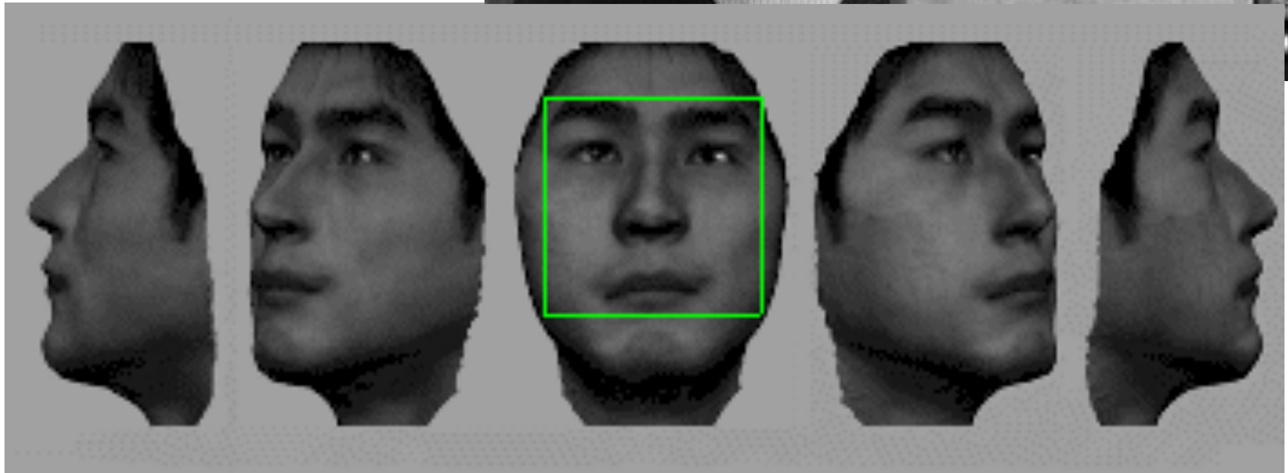
input

All images from: http://www.ius.cs.cmu.edu/demos/facedemo.html

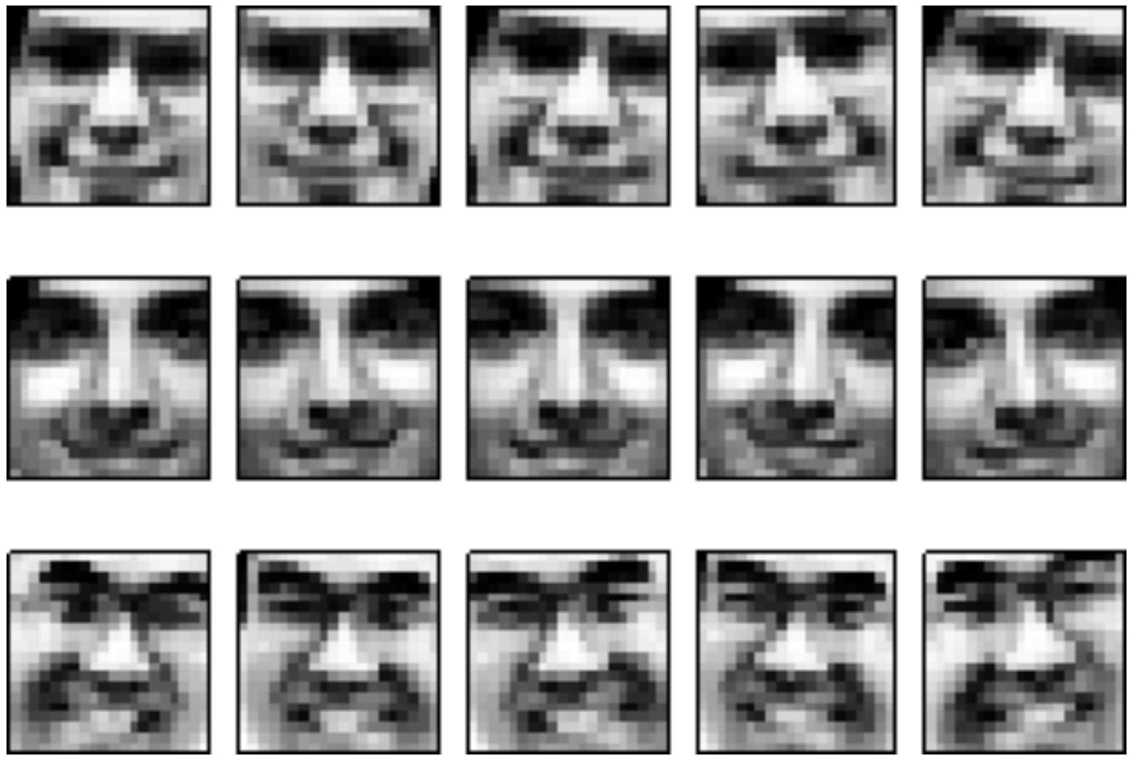




The novelization of the ultimate
Deep Space Nine adventure!
STAR TREK
DEEP SPACE NINE
THE WAY OF THE WARRIOR
A Novel by Diane Carey
d on The Way of the Warrior written by
Steven Behr & Robert Hewitt Wolfe

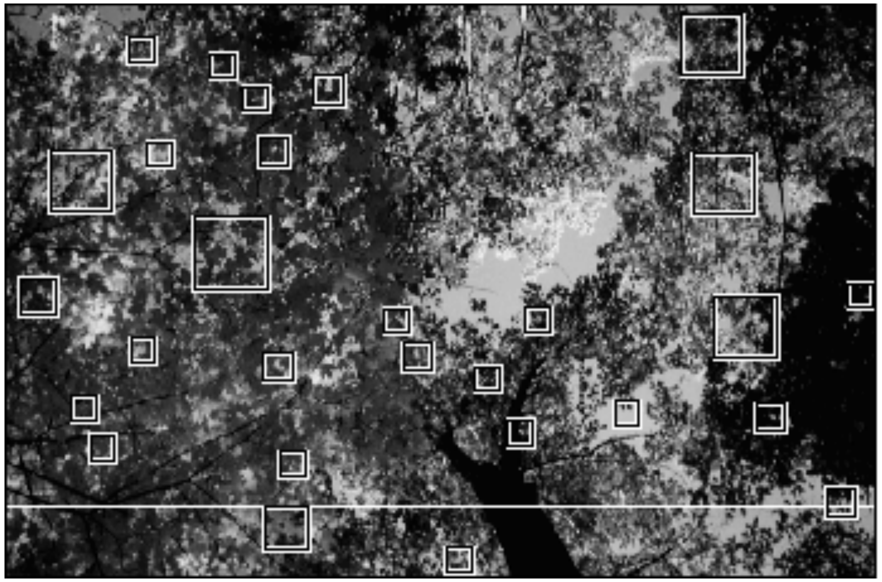
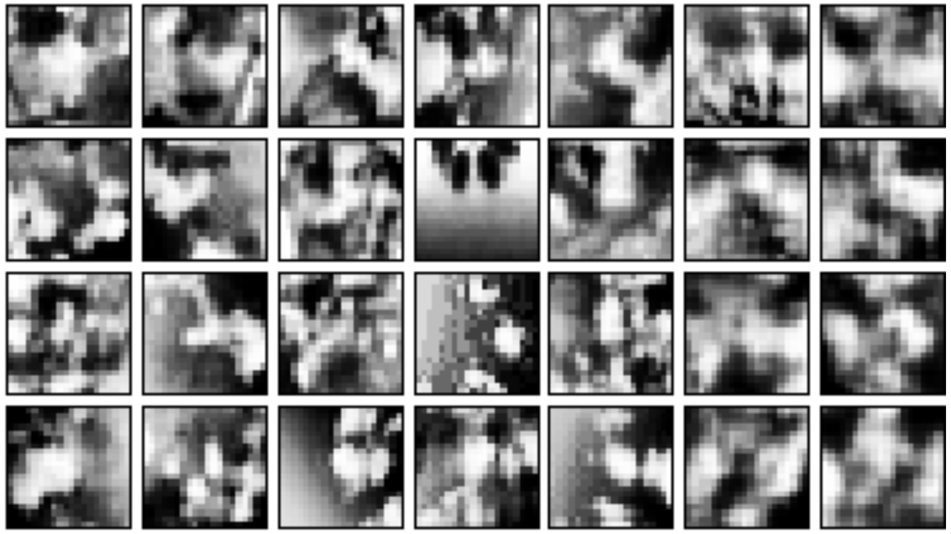


Example face images, randomly mirrored, rotated, translated, and scaled by small amounts (photos are of the three authors).



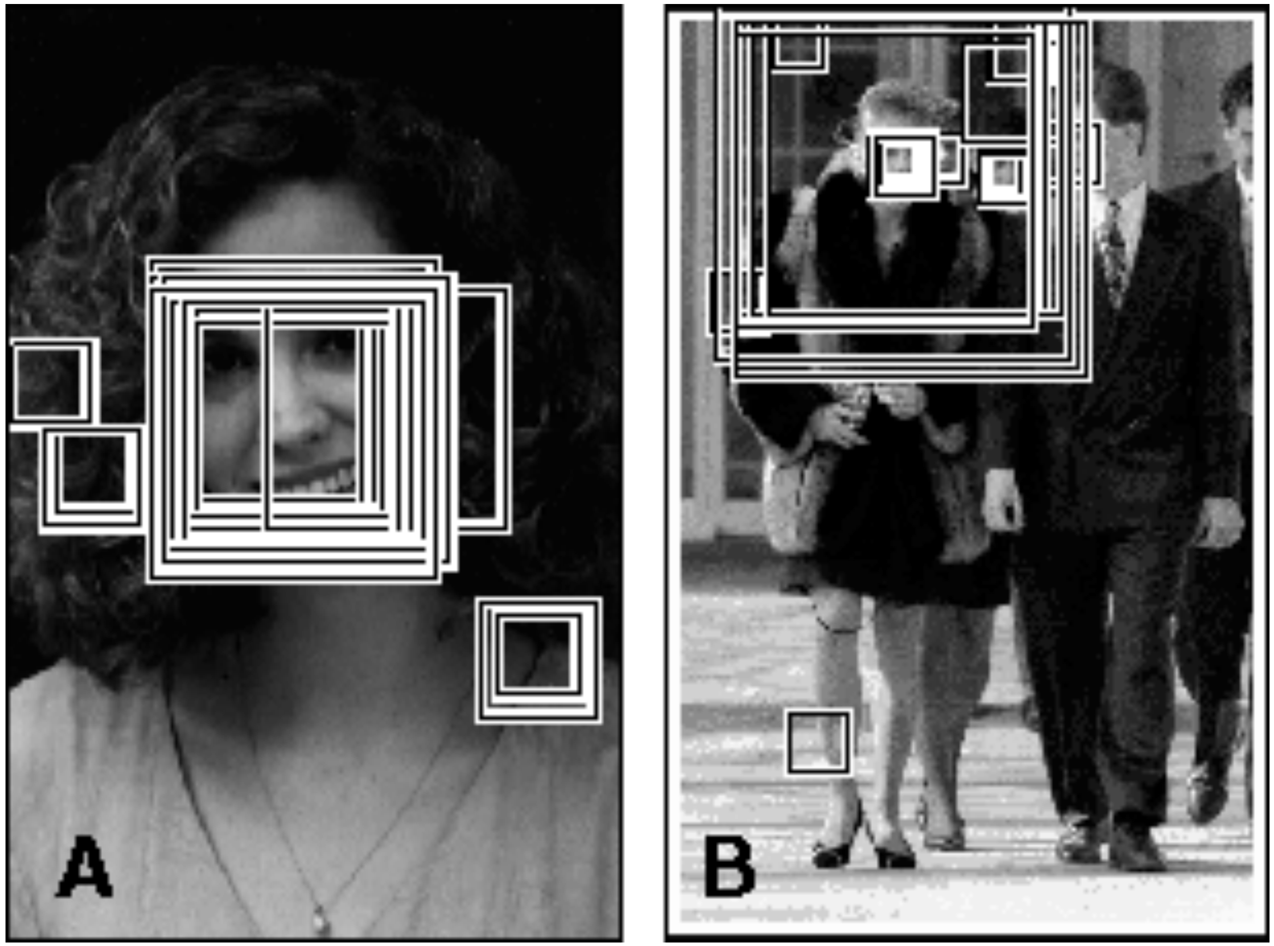
From: http://www.ius.cs.cmu.edu/IUS/har2/har/www/CMU-CS-95-158R/

During training, the partially-trained system is applied to images of scenery which do not contain faces (like the one on the left). Any regions in the image detected as faces (which are expanded and shown on the right) are errors, which can be added into the set of negative training examples.

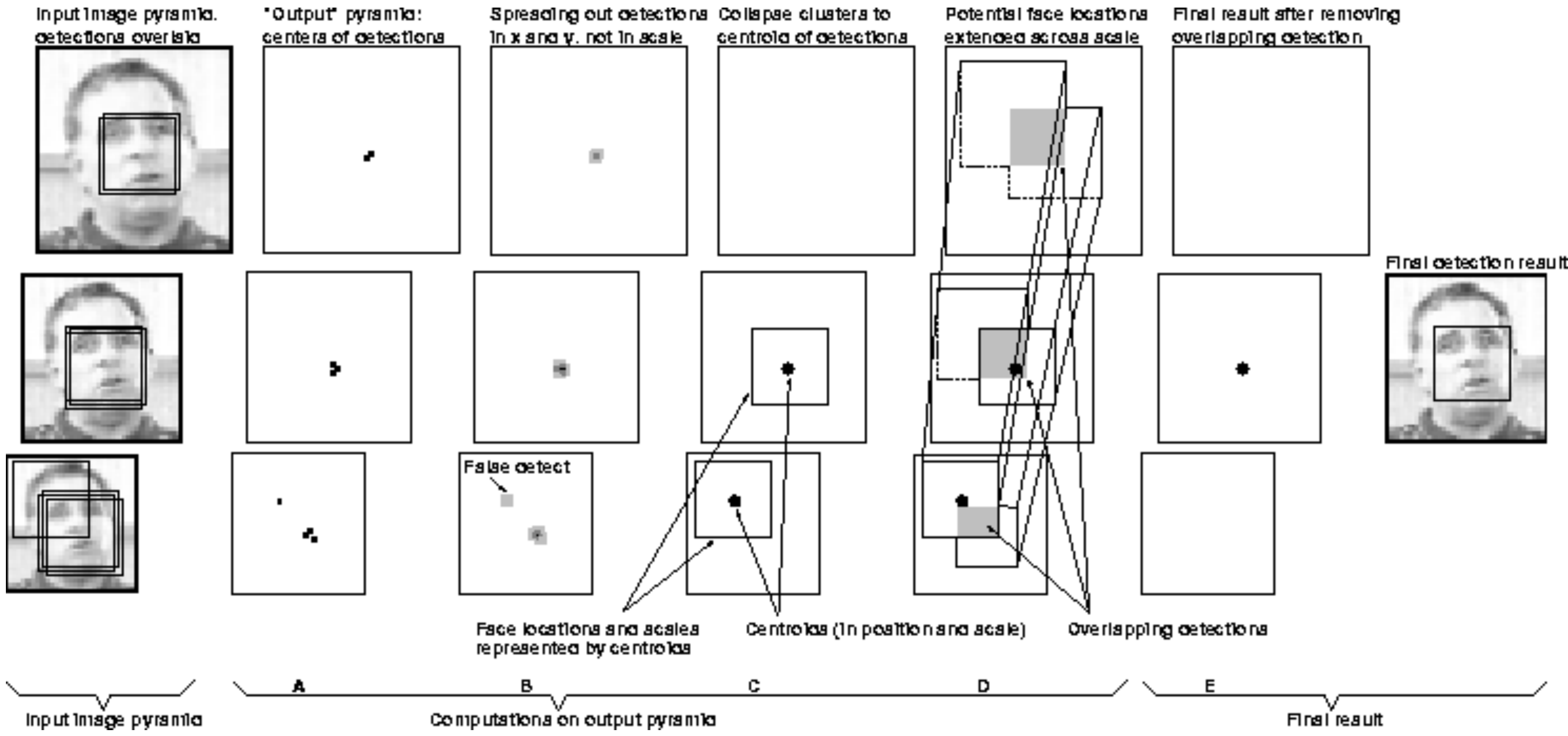



From: http://www.ius.cs.cmu.edu/IUS/har2/har/www/CMU-CS-95-158R/

Images with all the above threshold detections indicated by boxes.



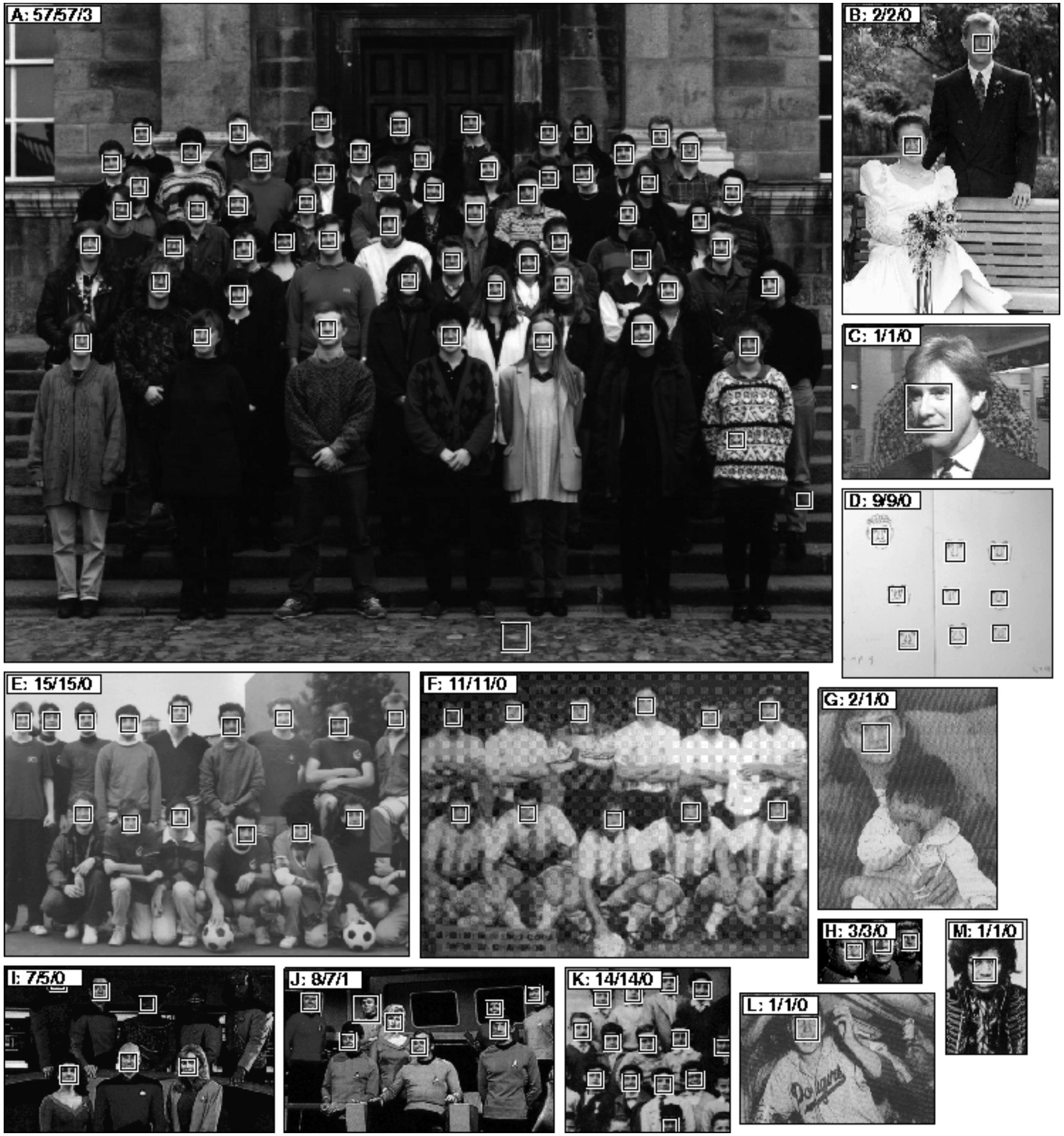
From: http://www.ius.cs.cmu.edu/IUS/har2/har/www/CMU-CS-95-158R/

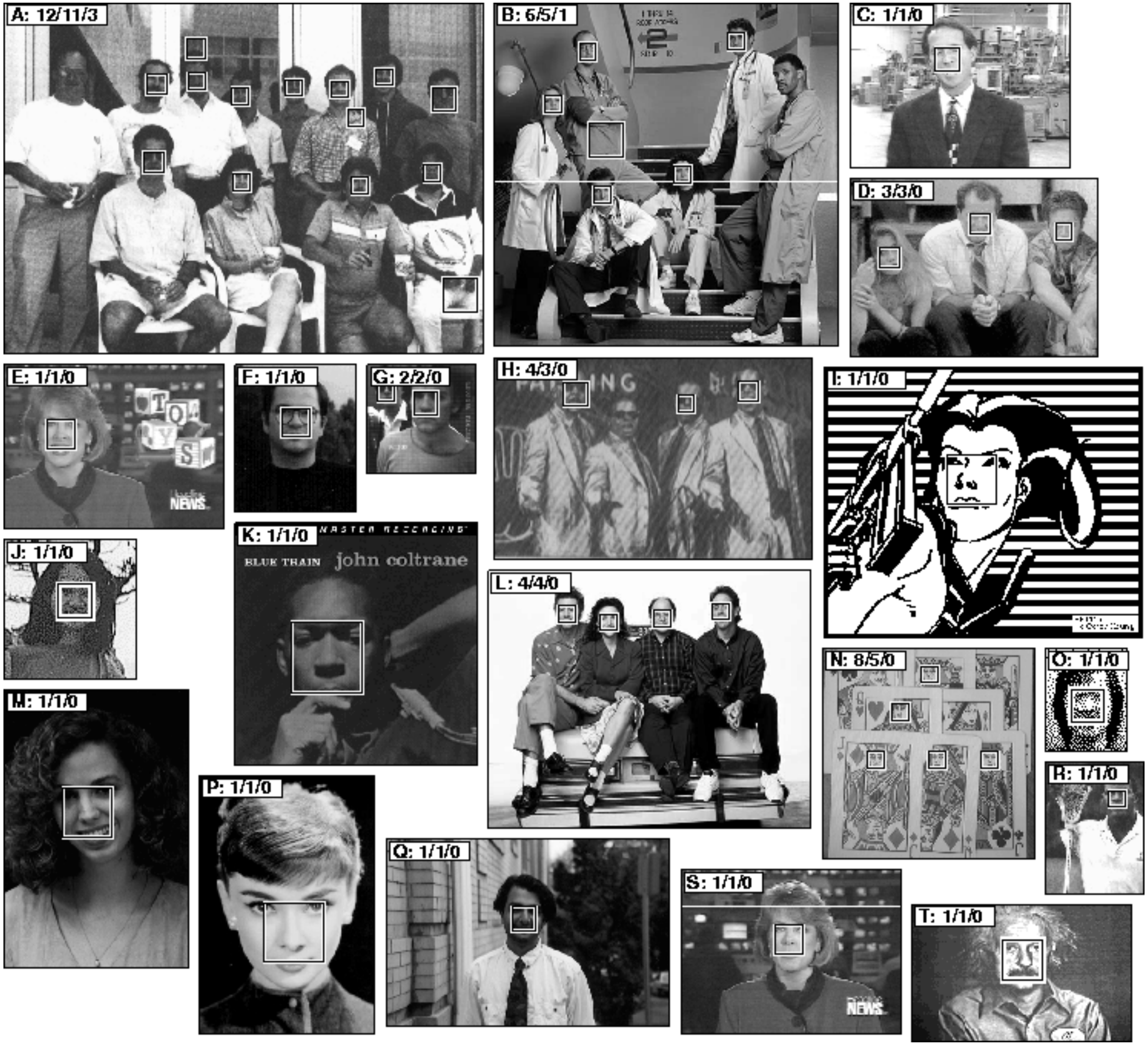


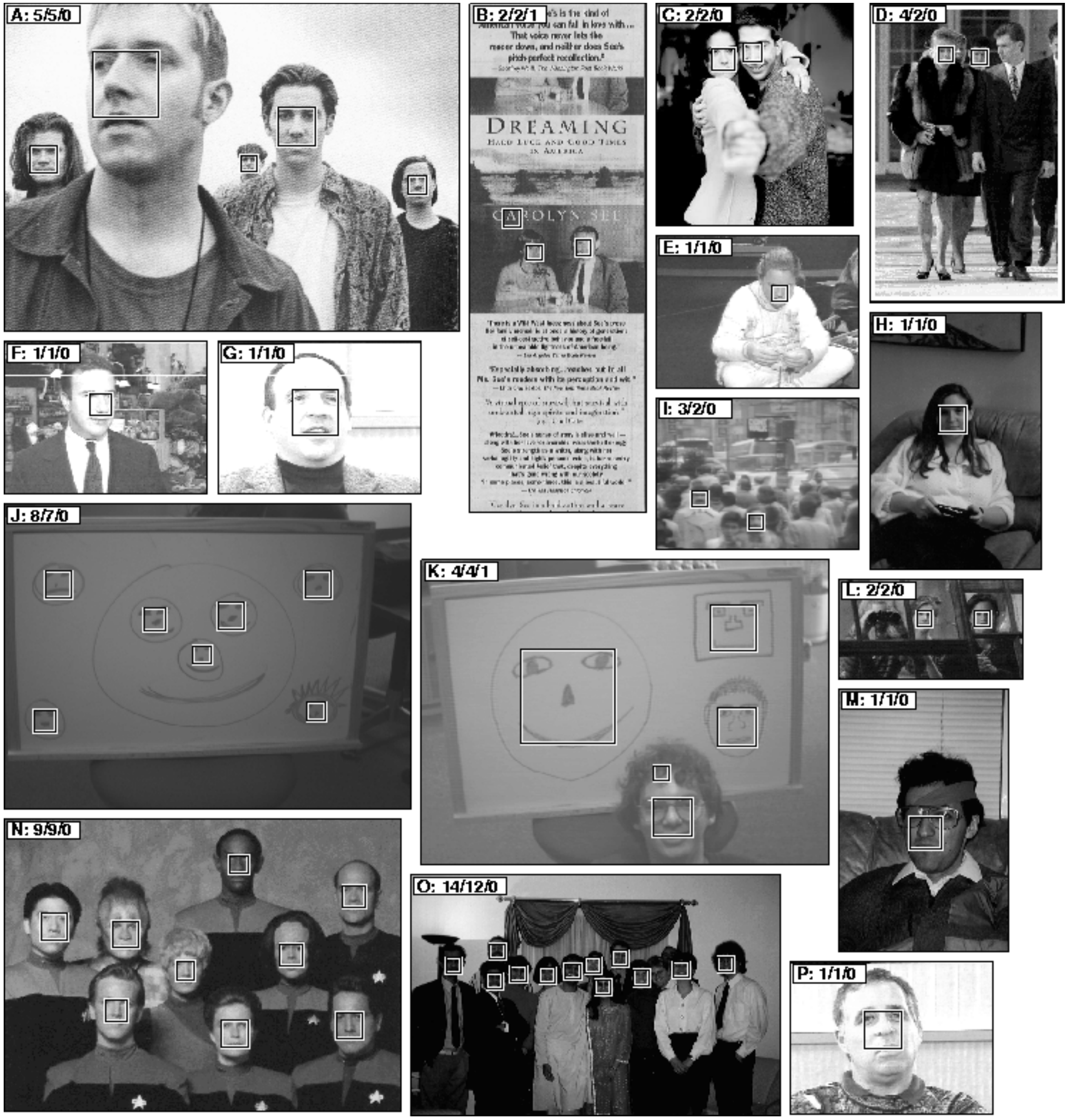
Input image pyramid, detections overlaid
"Output" pyramid: centers of detections
Spreading out detections in x and y, not in scale
Collapse clusters to centroid of detections
Potential face locations extended across scale
Final result after removing overlapping detection
Final detection result
False detect
Face locations and scales represented by centroids
Centroids (in position and scale)
Overlapping detections
A
B
C
D
E
Input image pyramid
Computations on output pyramid
Final result

The framework used for merging multiple detections from a single network: A) The detections are recorded in an image pyramid. B) The detections are ``spread out'' and a threshold is applied. C) The centroids in scale and position are computed, and the regions contributing to each centroid are collapsed to single points. In the example shown, this leaves only two detections in the output pyramid. D) The final step is to check the proposed face locations for overlaps, and E) to remove overlapping detections if they exist. In this example, removing the overlapping detection eliminates what would otherwise be a false positive.

From: http://www.ius.cs.cmu.edu/IUS/har2/har/www/CMU-CS-95-158R/

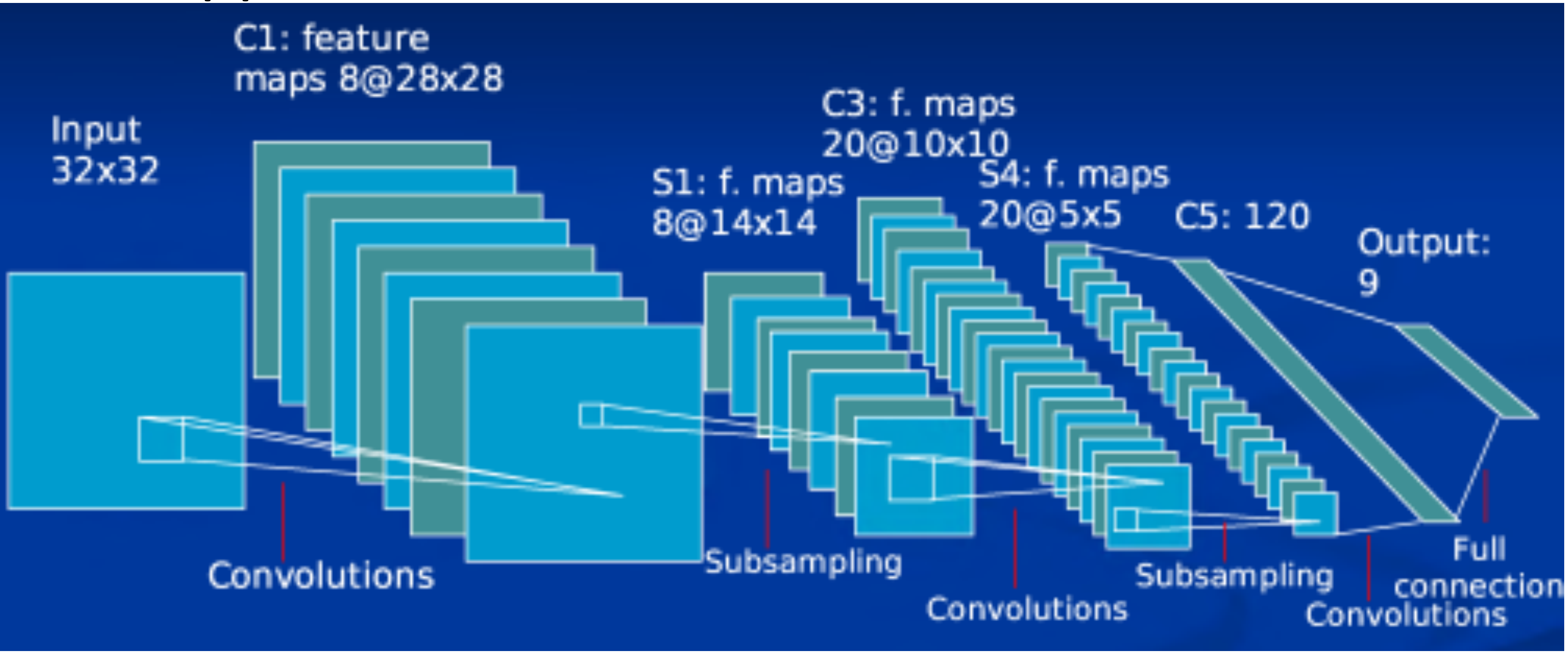

A: 57/57/3
B: 2/2/0
C: 1/1/0
D: 9/9/0
E: 15/15/0
F: 11/11/0
G: 2/1/0
H: 3/3/0
M: 1/1/0
I: 7/5/0
J: 8/7/1
K: 14/14/0
L: 1/1/0


A: 12/11/3
B: 6/5/1
C: 1/1/0
D: 3/3/0
E: 1/1/0
F: 1/1/0
G: 2/2/0
H: 4/3/0
I: 1/1/0
J: 1/1/0
K: 1/1/0
BLUE TRAIN john coltrane
L: 4/4/0
M: 1/1/0
N: 8/5/0
O: 1/1/0
P: 1/1/0
Q: 1/1/0
R: 1/1/0
S: 1/1/0
T: 1/1/0


A: 5/5/0
B: 2/2/1
DREAMING
HARD LUCK AND GOOD TIMES IN AMERICA
CAROLYN SEE
C: 2/2/0
D: 4/2/0
E: 1/1/0
F: 1/1/0
G: 1/1/0
H: 1/1/0
I: 3/2/0
J: 8/7/0
K: 4/4/1
L: 2/2/0
M: 1/1/0
N: 9/9/0
O: 14/12/0
P: 1/1/0

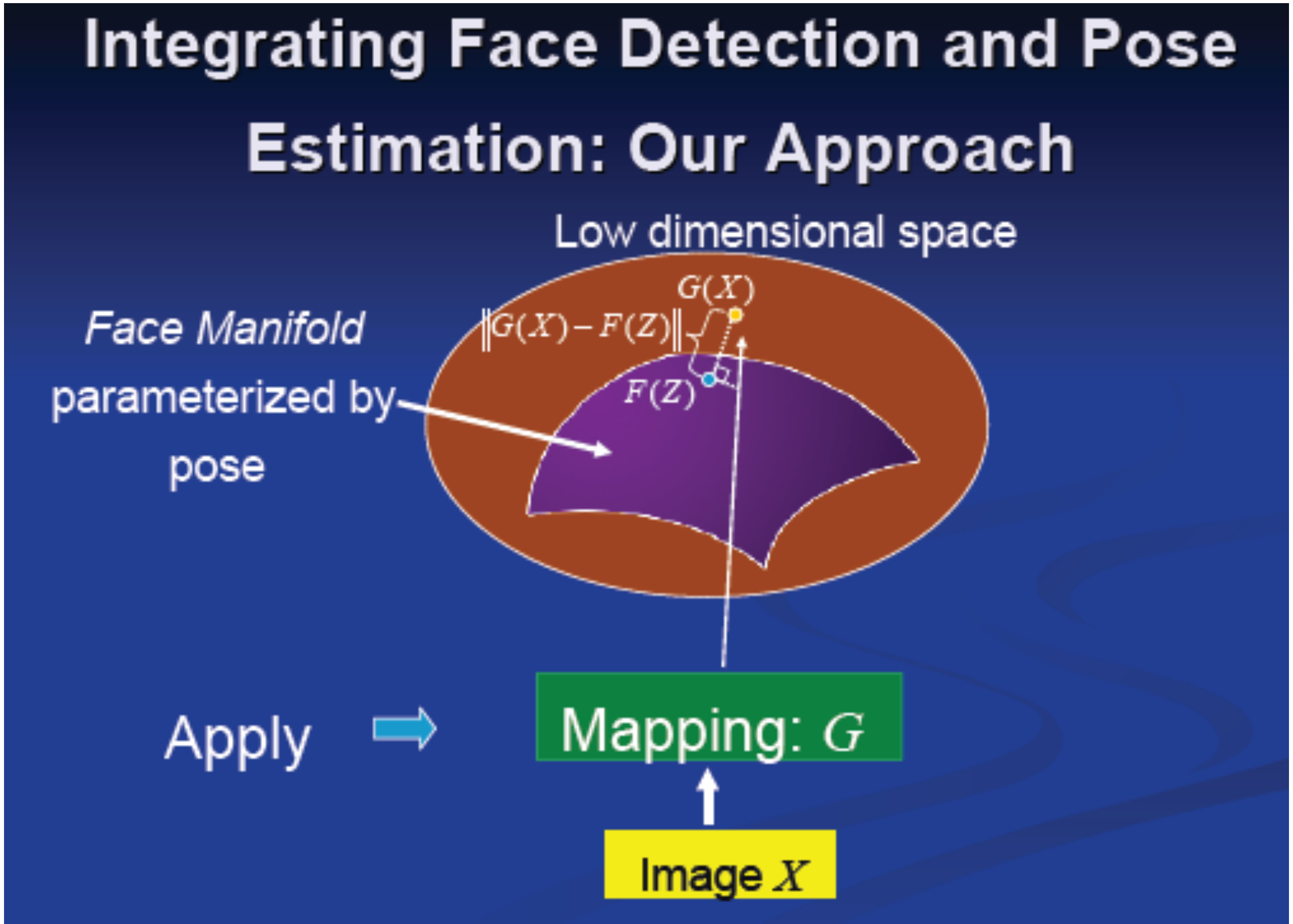
Osadchy, Miller, LeCun.
Face Detection and Pose Estimation, 2004

- Application of Convolutional Neural Networks

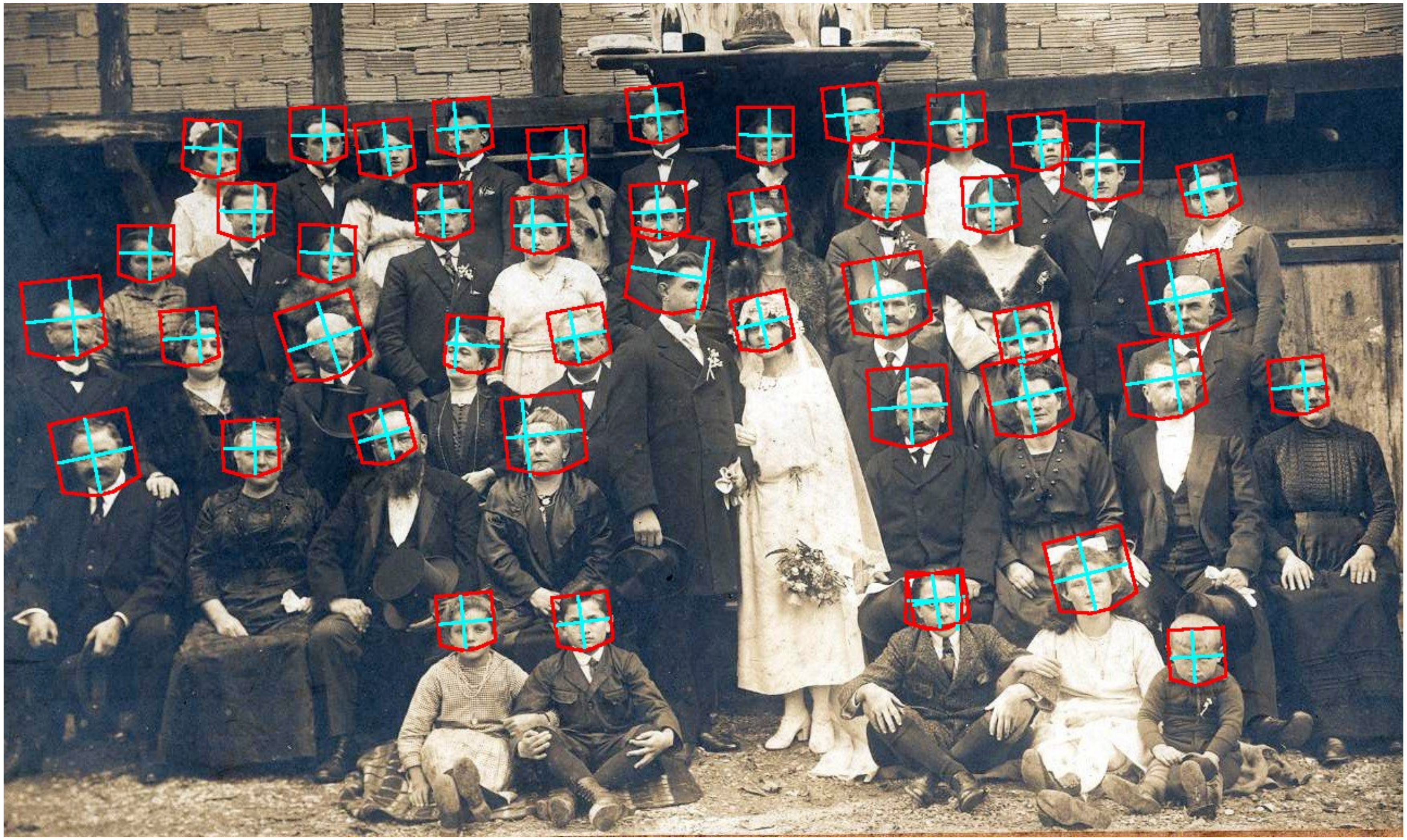


Osadchy, Miller, LeCun.
Face Detection and Pose Estimation, 2004

- Non-linear dimensionality reduction


Integrating Face Detection and Pose Estimation: Our Approach
Low dimensional space
$G(X)$
$\|G(X)-F(Z)\|$
$F(Z)$
Face Manifold parameterized by pose
Apply
Mapping: G
Image X

Osadchy, Miller, LeCun.
Face Detection and Pose Estimation, 2004



Overview

- Specific Object Recognition
  - Lowe '04
  - Rothganger et al. '03
- Face Dectection
  - Sub-space methods
  - Neural Network-based approaches
  - Viola & Jones
- Face Recognition

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

The Viola/Jones Face Detector

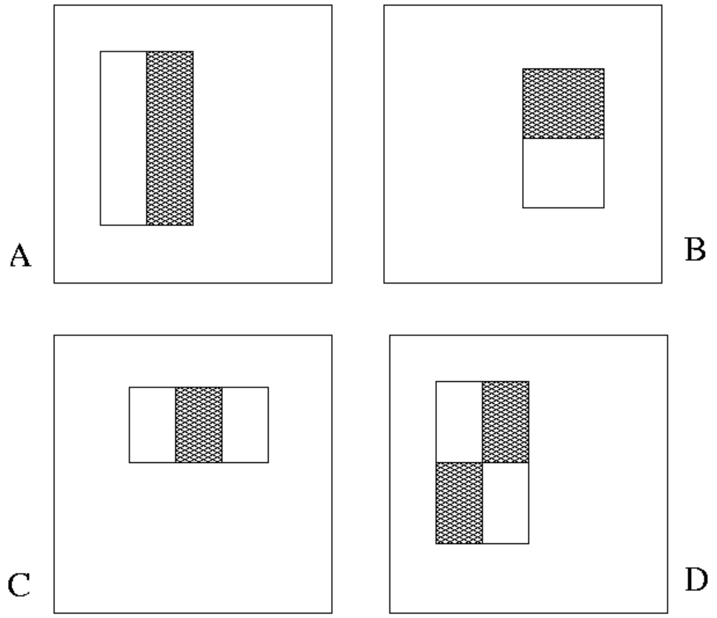
- A seminal approach to real-time object detection
- Training is slow, but detection is very fast
- Key ideas
  - *Integral images* for fast feature evaluation
  - *Boosting* for feature selection
  - *Attentional cascade* for fast rejection of non-face windows

P. Viola and M. Jones. *Rapid object detection using a boosted cascade of simple features.* CVPR 2001.

P. Viola and M. Jones. *Robust real-time face detection.* IJCV 57(2), 2004.

Image Features

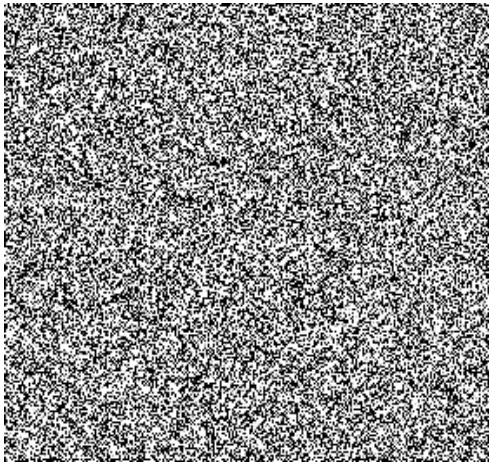
"Rectangle filters"



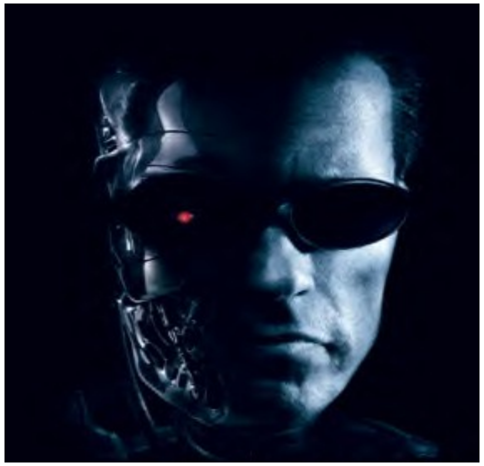
Value =

∑ (pixels in white area) –
∑ (pixels in black area)

Example

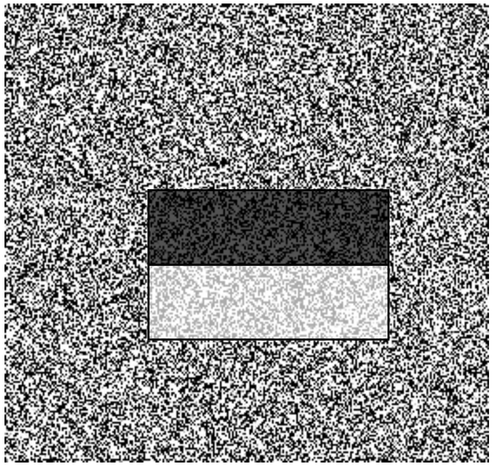


Source

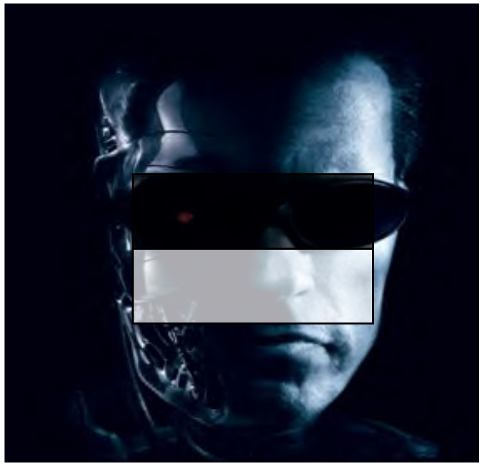


Result

✗



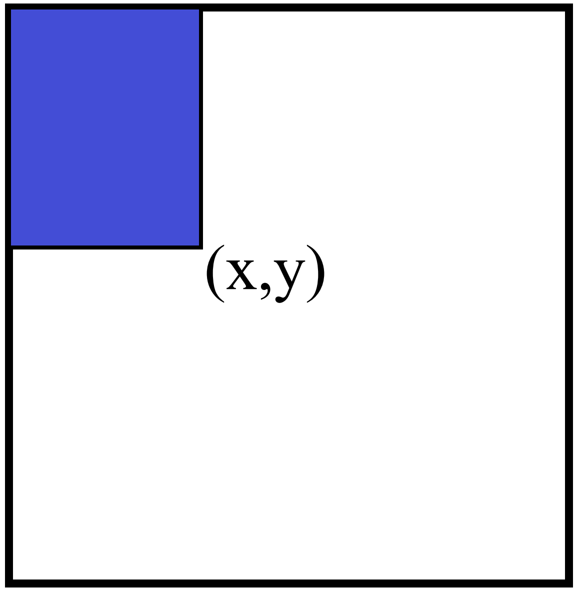
✓



Slide: S. Lazebnik

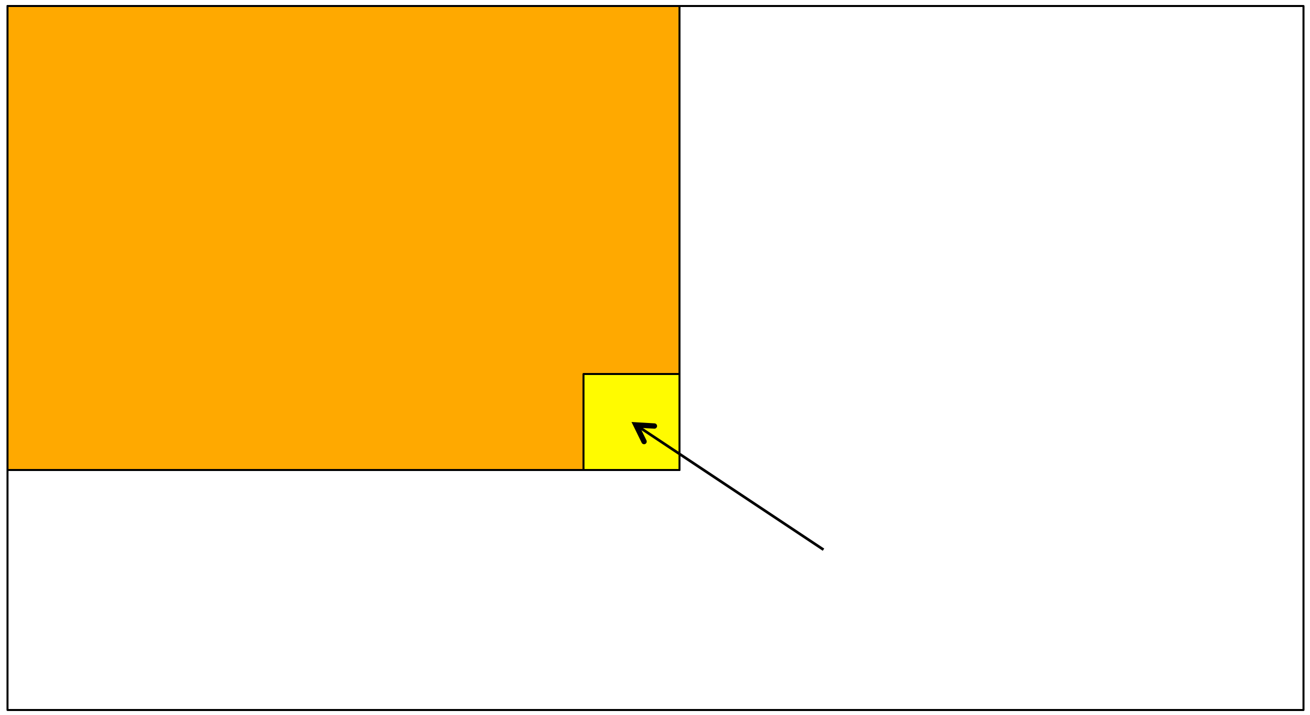
Fast computation with integral images

- The *integral image* computes a value at each pixel (*x*,*y*) that is the sum of the pixel values above and to the left of (*x*,*y*), inclusive
- This can quickly be computed in one pass through the image



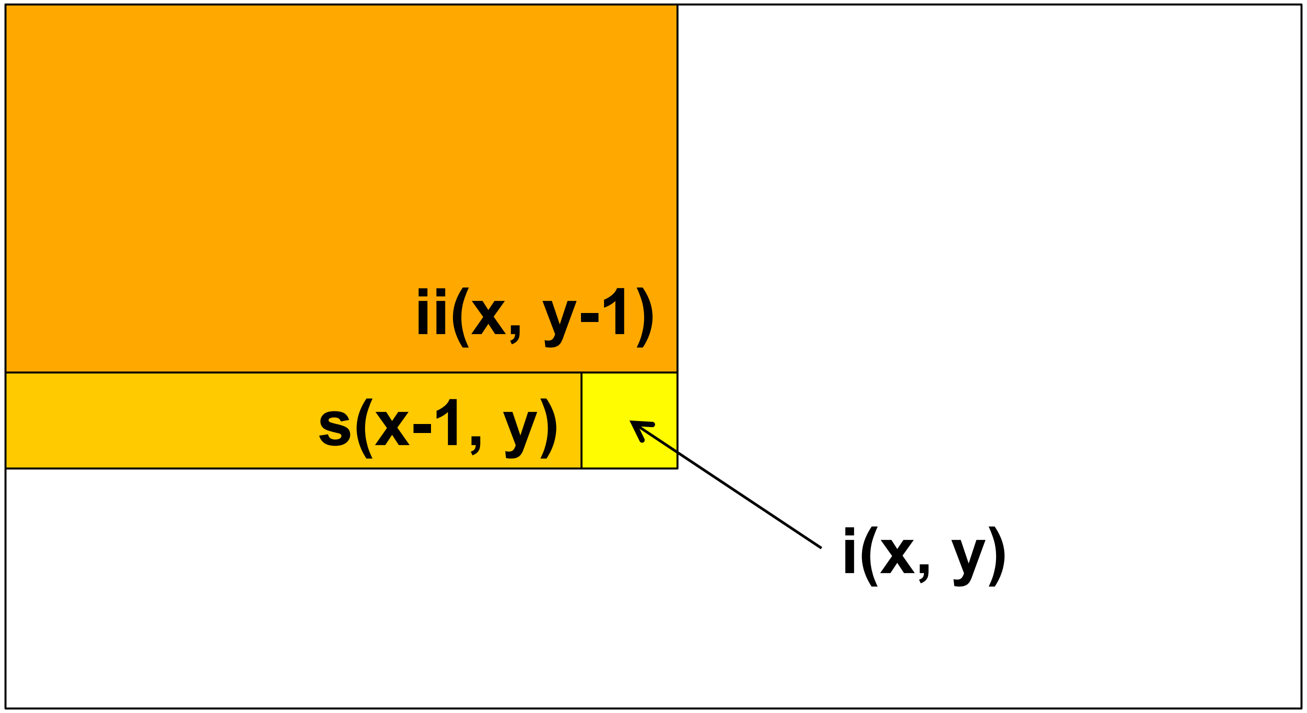
Slide: S. Lazebnik

Computing the integral image



Slide: S. Lazebnik

Computing the integral image



Cumulative row sum: $s(x, y) = s(x-1, y) + i(x, y)$

Integral image: $ii(x, y) = ii(x, y-1) + s(x, y)$

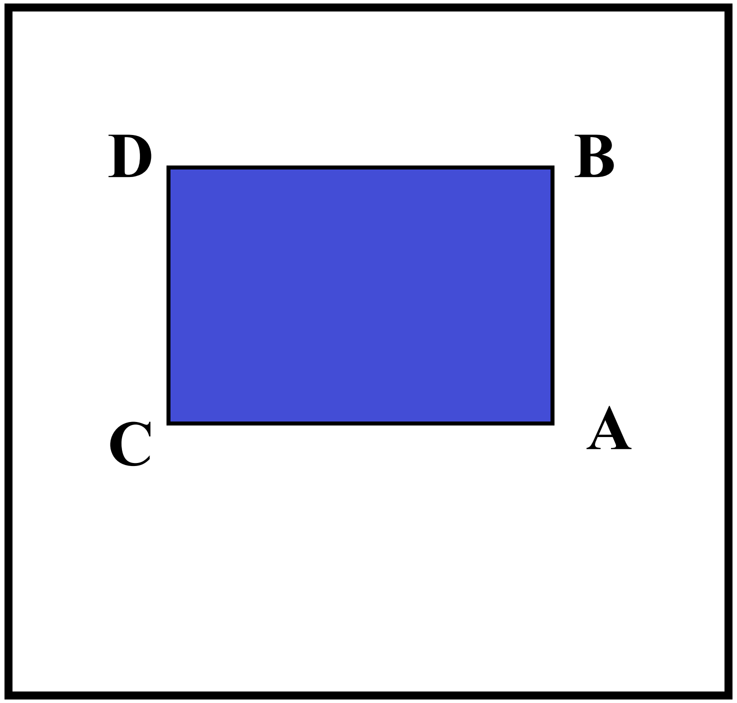
MATLAB: `ii = cumsum(cumsum(double(i)), 2);`

Slide: S. Lazebnik

Computing sum within a rectangle

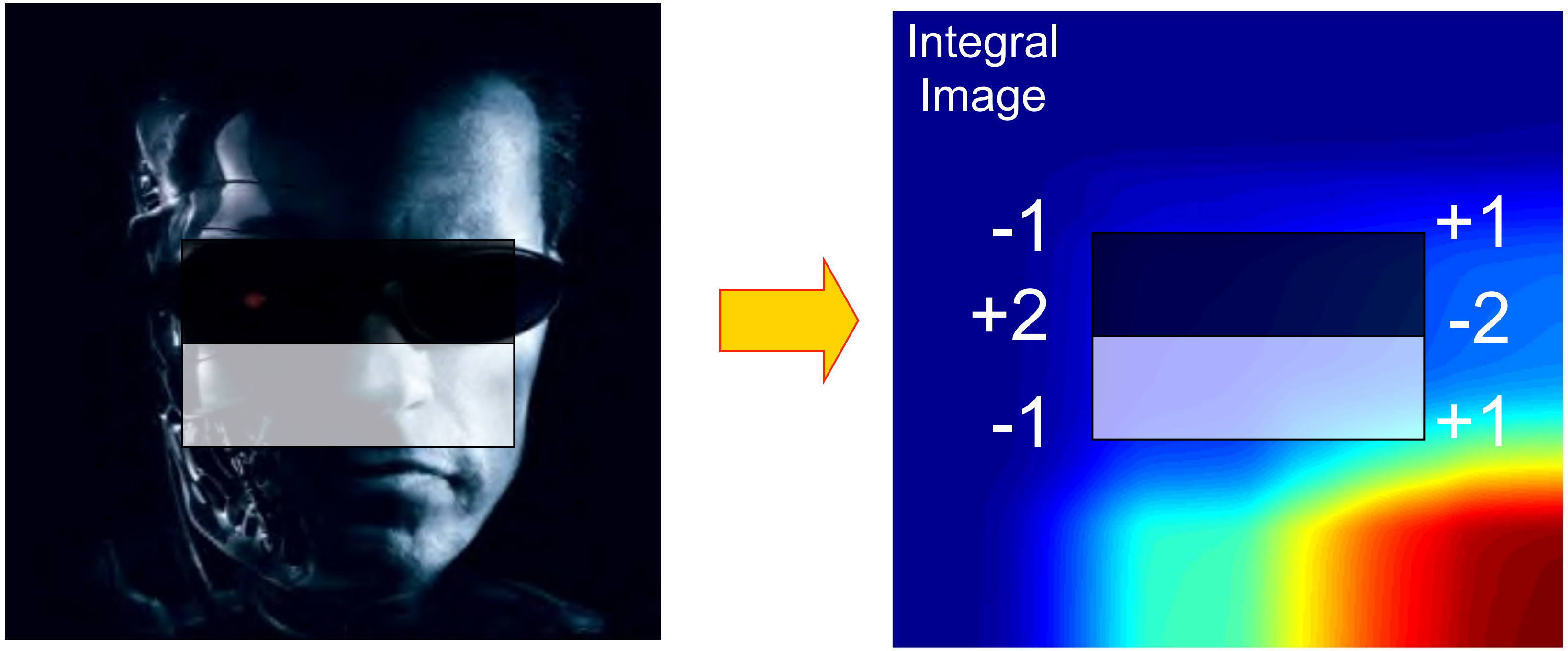
- Let A,B,C,D be the values of the integral image at the corners of a rectangle
- Then the sum of original image values within the rectangle can be computed as:

  sum = A – B – C + D
- Only 3 additions are required for any size of rectangle!



Slide: S. Lazebnik

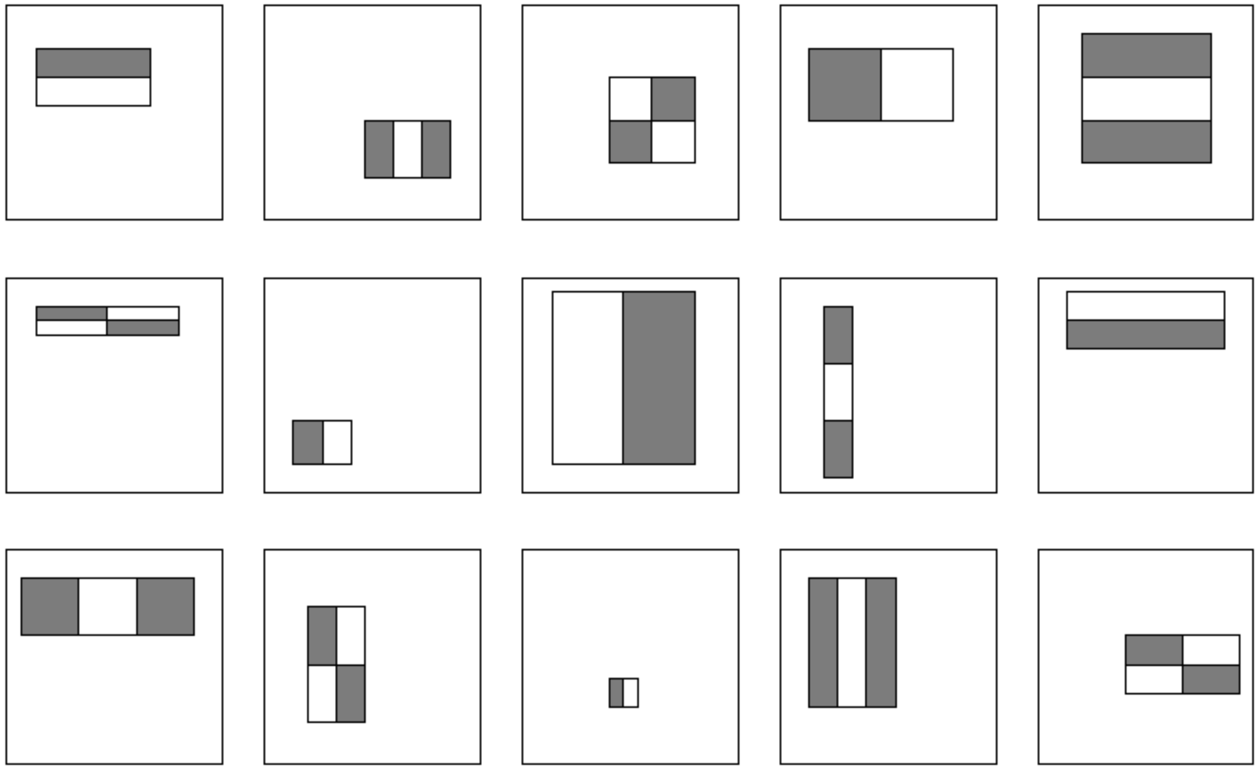
Example



Slide: S. Lazebnik

Feature selection

- For a 24x24 detection region, the number of possible rectangle features is ~160,000!



Feature selection

- For a 24x24 detection region, the number of possible rectangle features is ~160,000!
- At test time, it is impractical to evaluate the entire feature set
- Can we create a good classifier using just a small subset of all possible features?
- How to select such a subset?

Boosting

- Boosting is a classification scheme that works by combining *weak learners* into a more accurate ensemble classifier
- Training consists of multiple *boosting rounds*
  - During each boosting round, we select a weak learner that does well on examples that were hard for the previous weak learners
  - "Hardness" is captured by weights attached to training examples

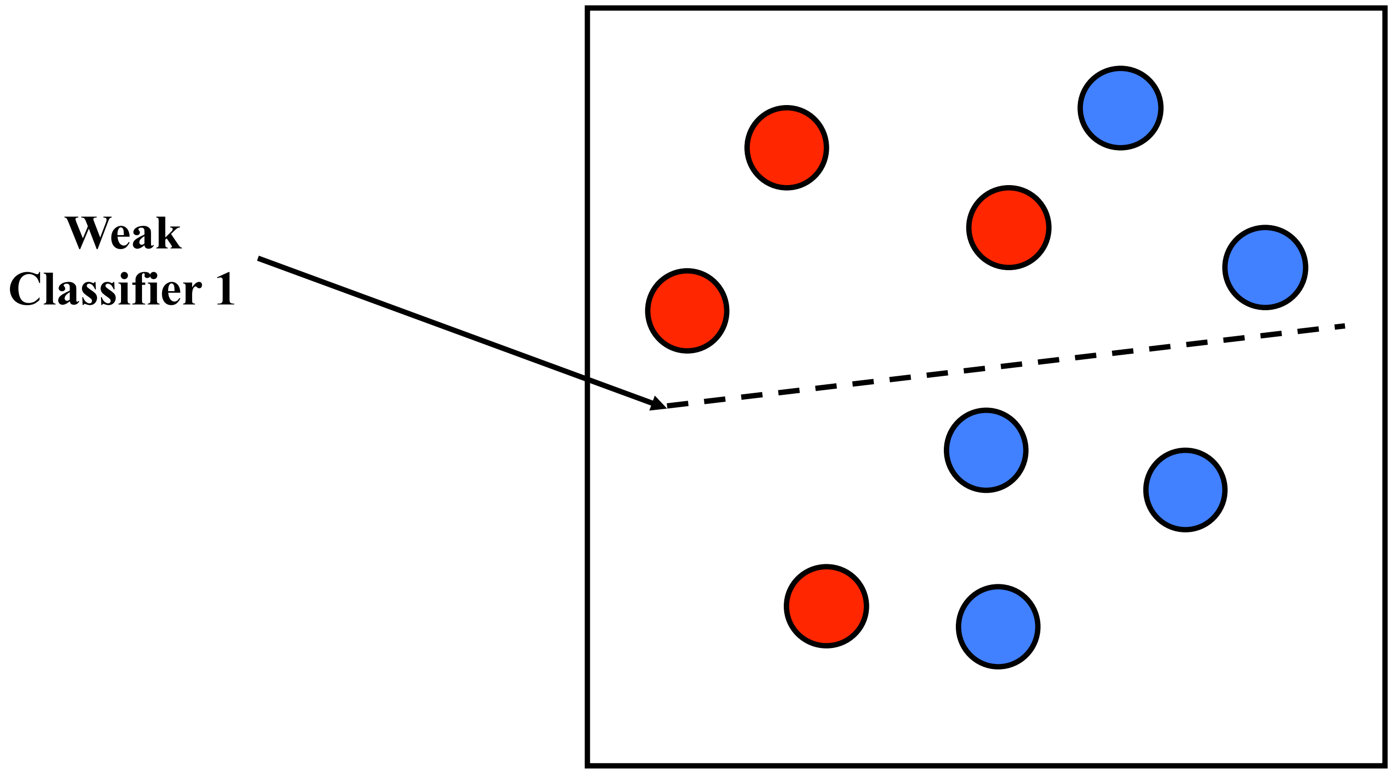
Y. Freund and R. Schapire, A short introduction to boosting, *Journal of Japanese Society for Artificial Intelligence*, 14(5):771-780, September, 1999.

Training procedure

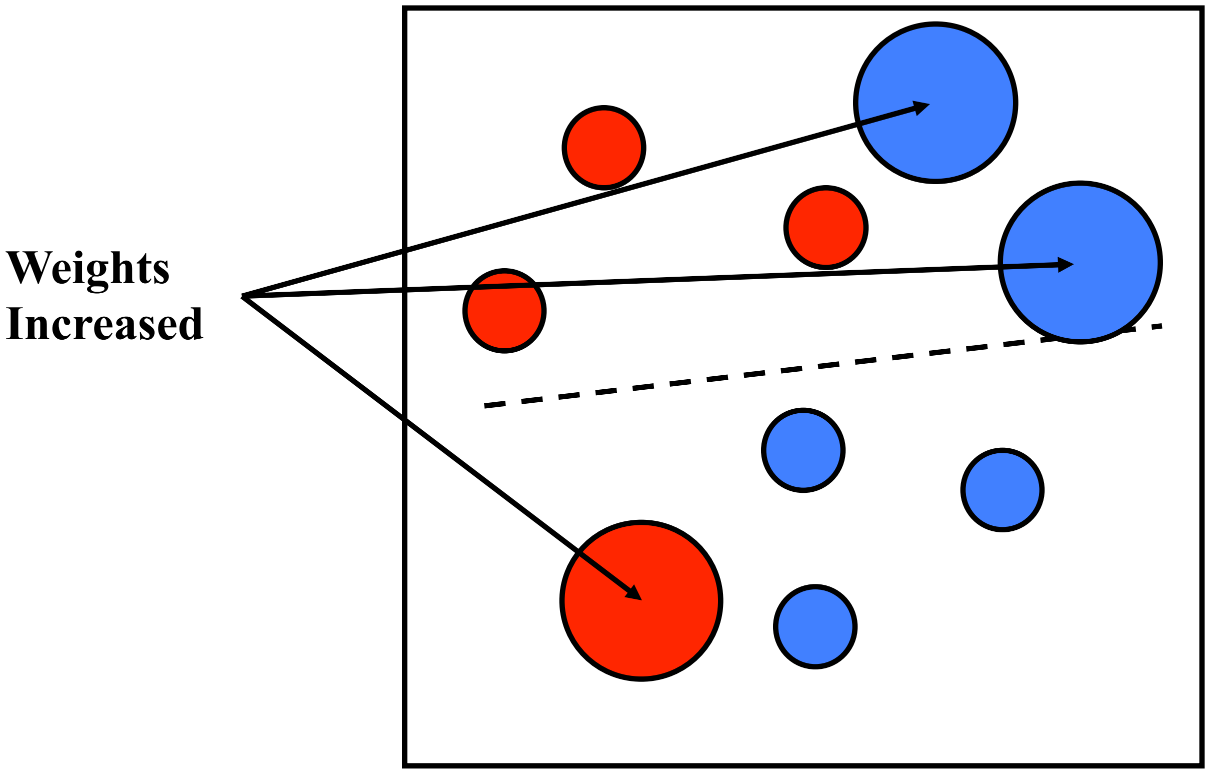
- Initially, weight each training example equally
- In each boosting round:
  - Find the weak learner that achieves the lowest *weighted* training error
  - Raise the weights of training examples misclassified by current weak learner
- Compute final classifier as linear combination of all weak learners (weight of each learner is directly proportional to its accuracy)
- Exact formulas for re-weighting and combining weak learners depend on the particular boosting scheme (e.g., AdaBoost)

Y. Freund and R. Schapire, A short introduction to boosting, *Journal of Japanese Society for Artificial Intelligence*, 14(5):771-780, September, 1999.

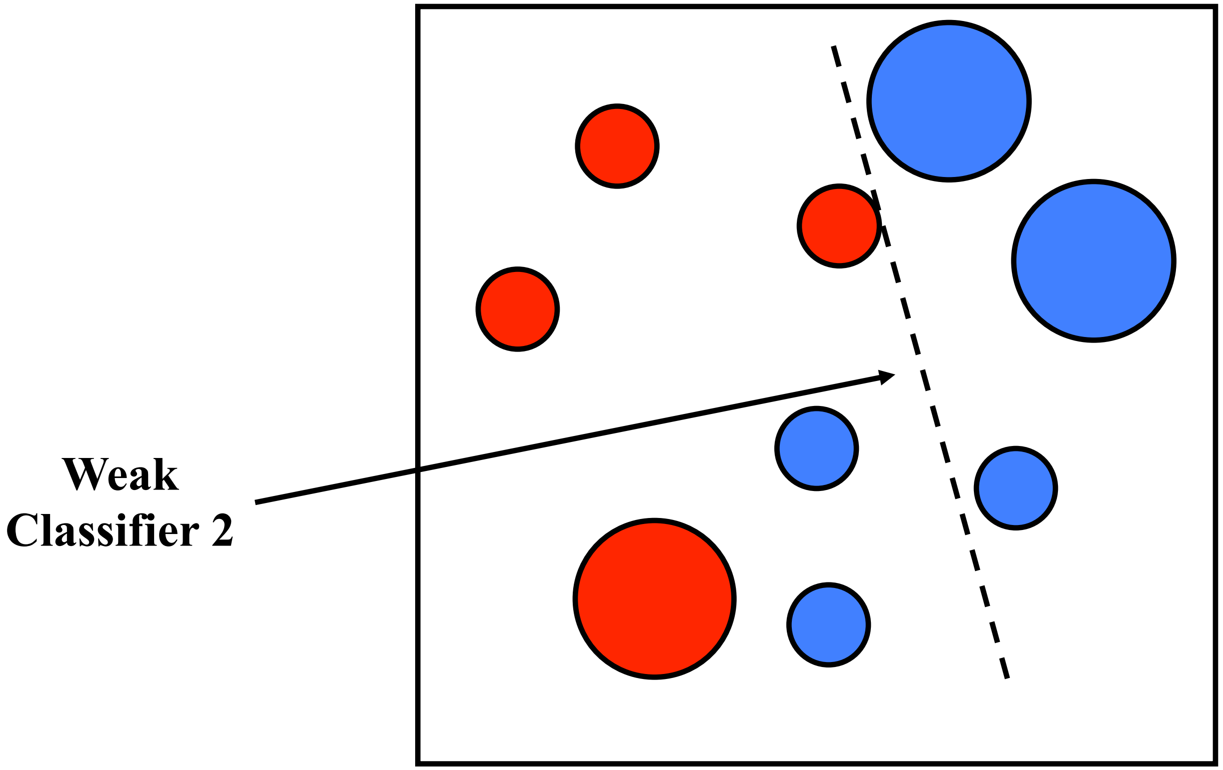
Boosting



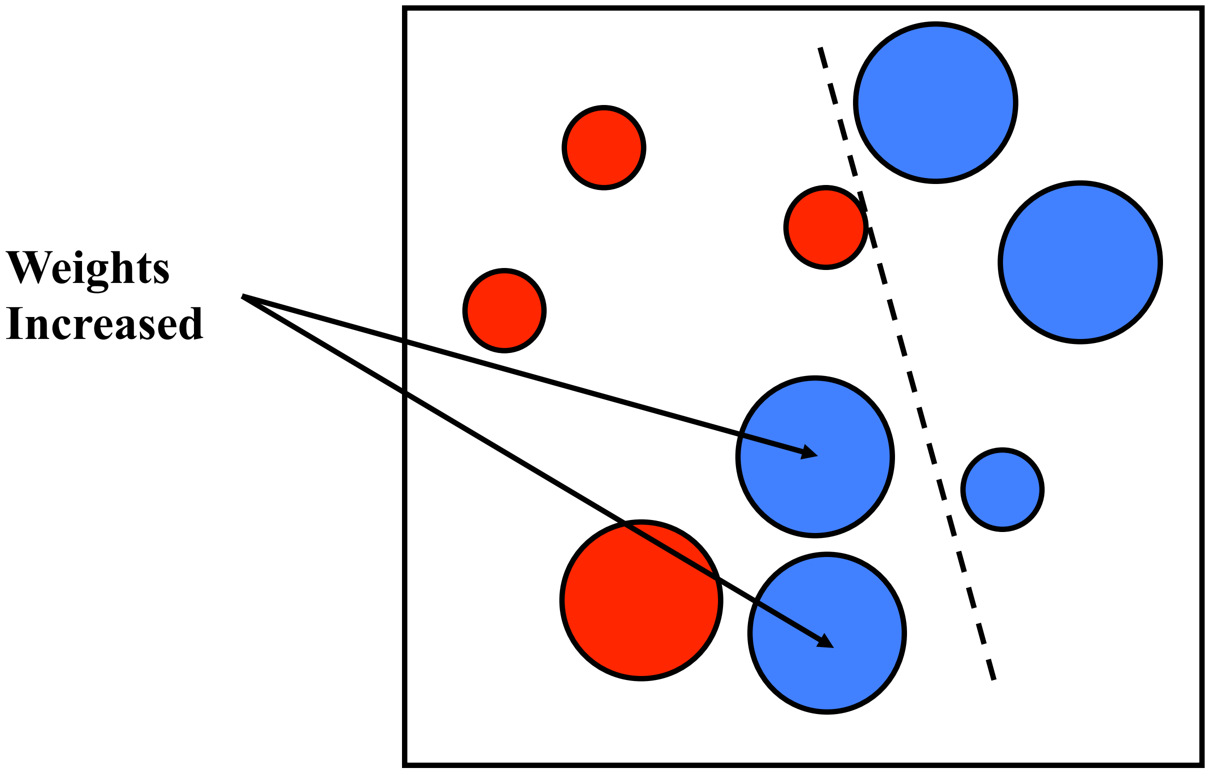
Boosting



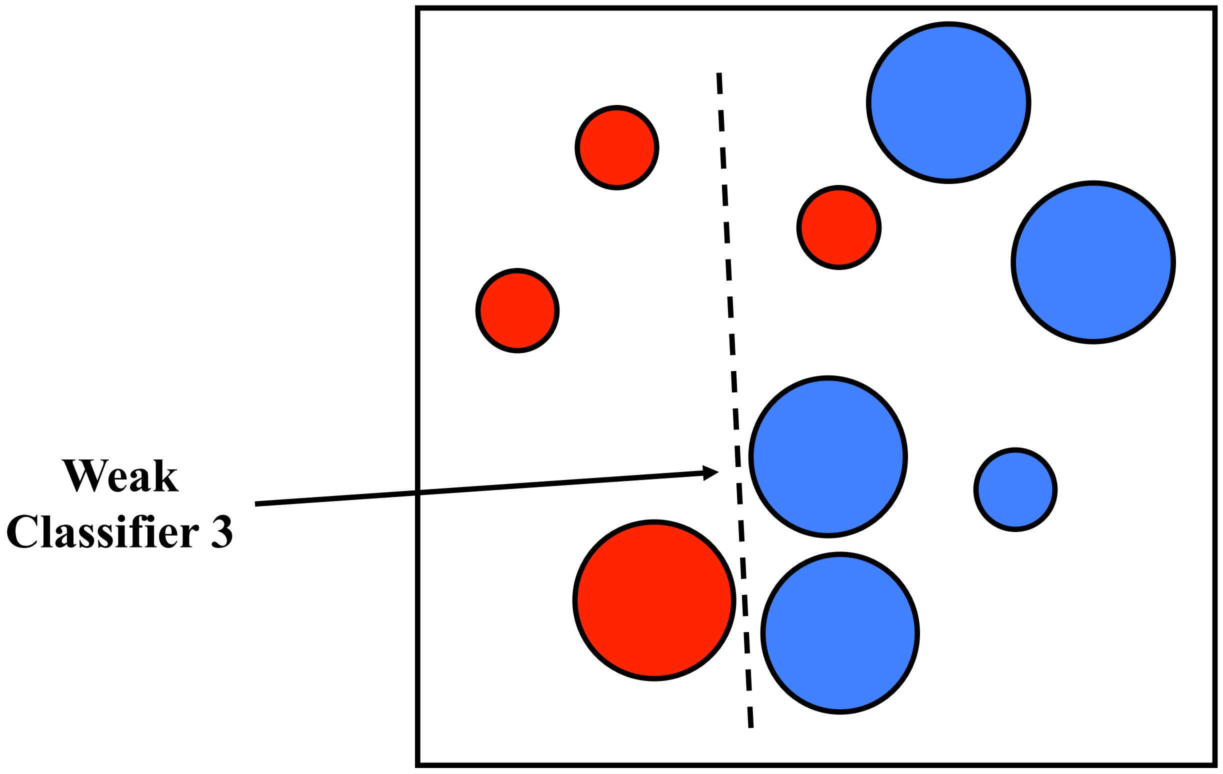
Boosting



Boosting

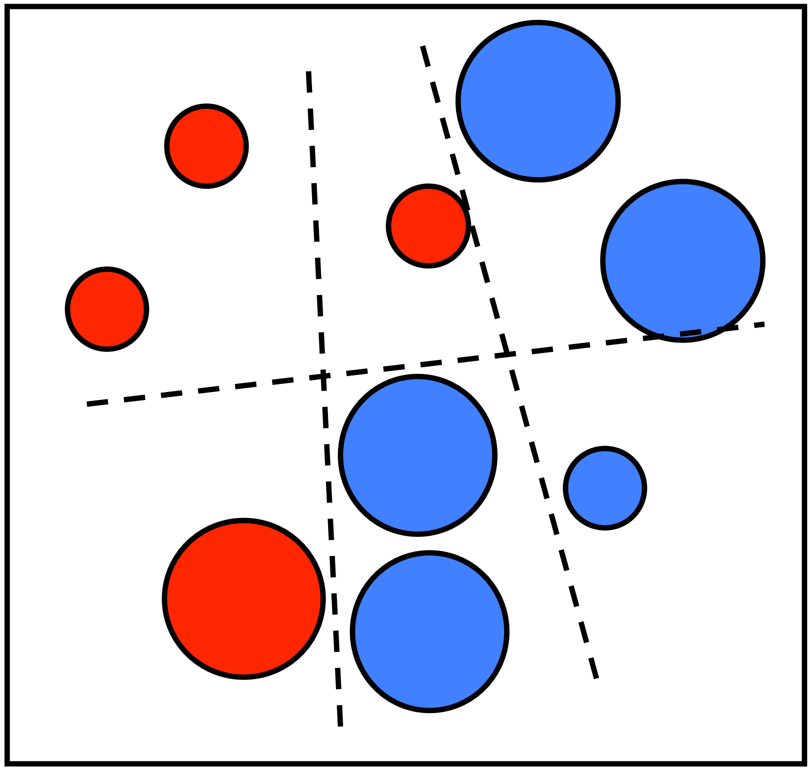


Boosting



Boosting

Final classifier is linear combination of weak classifiers



Boosting vs. SVM

- Advantages of boosting
  - Integrates classifier training with feature selection
  - Complexity of training is linear instead of quadratic in the number of training examples
  - Flexibility in the choice of weak learners, boosting scheme
  - Testing is fast
  - Easy to implement
- Disadvantages
  - Needs many training examples
  - Training is slow
  - Often doesn’t work as well as SVM (especially for many-class problems)

Boosting for face detection

- Define weak learners based on rectangle features

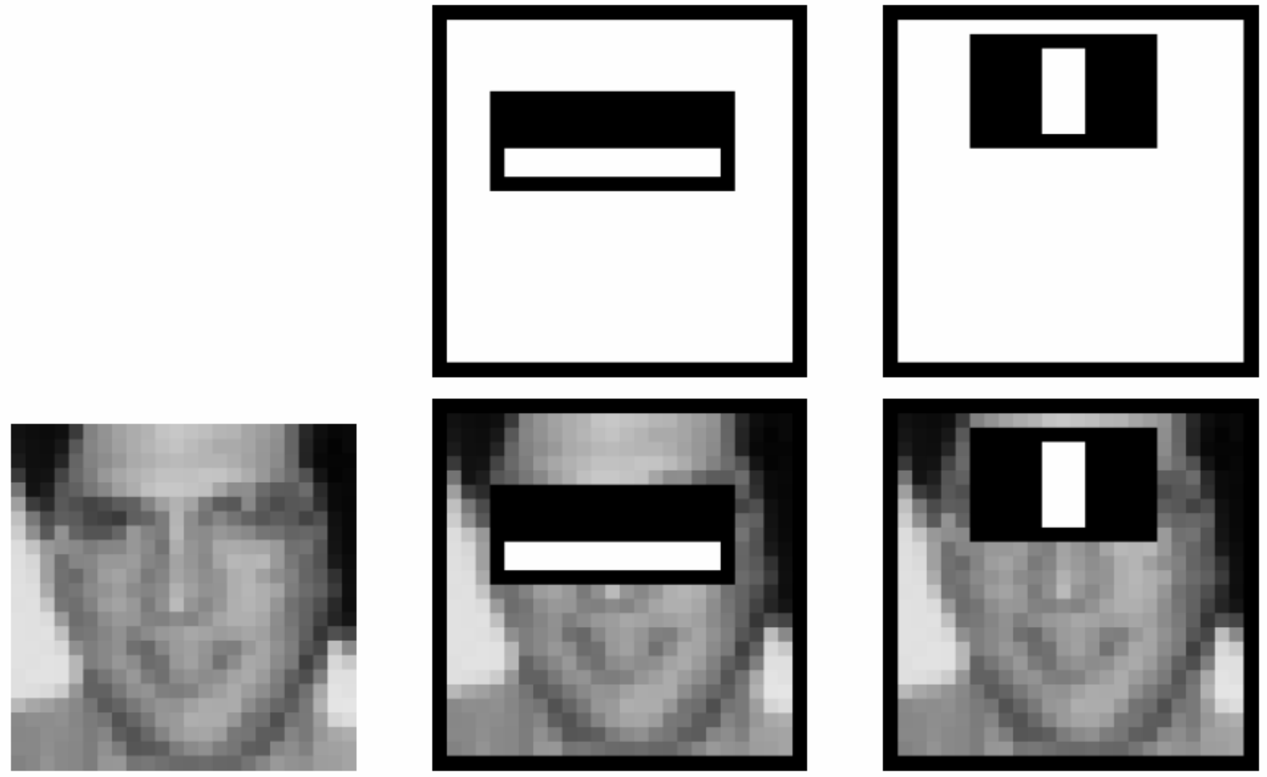
$$h_t(\underset{\substack{\uparrow \\ \text{window}}}{x}) = \begin{cases} 1 & \text{if } p_t \overset{\substack{\text{value of rectangle feature} \\ \downarrow}}{f_t(x)} > \underset{\substack{\uparrow \\ \text{parity}}}{p_t}\,\underset{\substack{\uparrow \\ \text{threshold}}}{\theta_t} \\ 0 & \text{otherwise} \end{cases}$$

Boosting for face detection

- Define weak learners based on rectangle features
- For each round of boosting:
  - Evaluate each rectangle filter on each example
  - Select best filter/threshold combination based on weighted training error
  - Reweight examples

Boosting for face detection

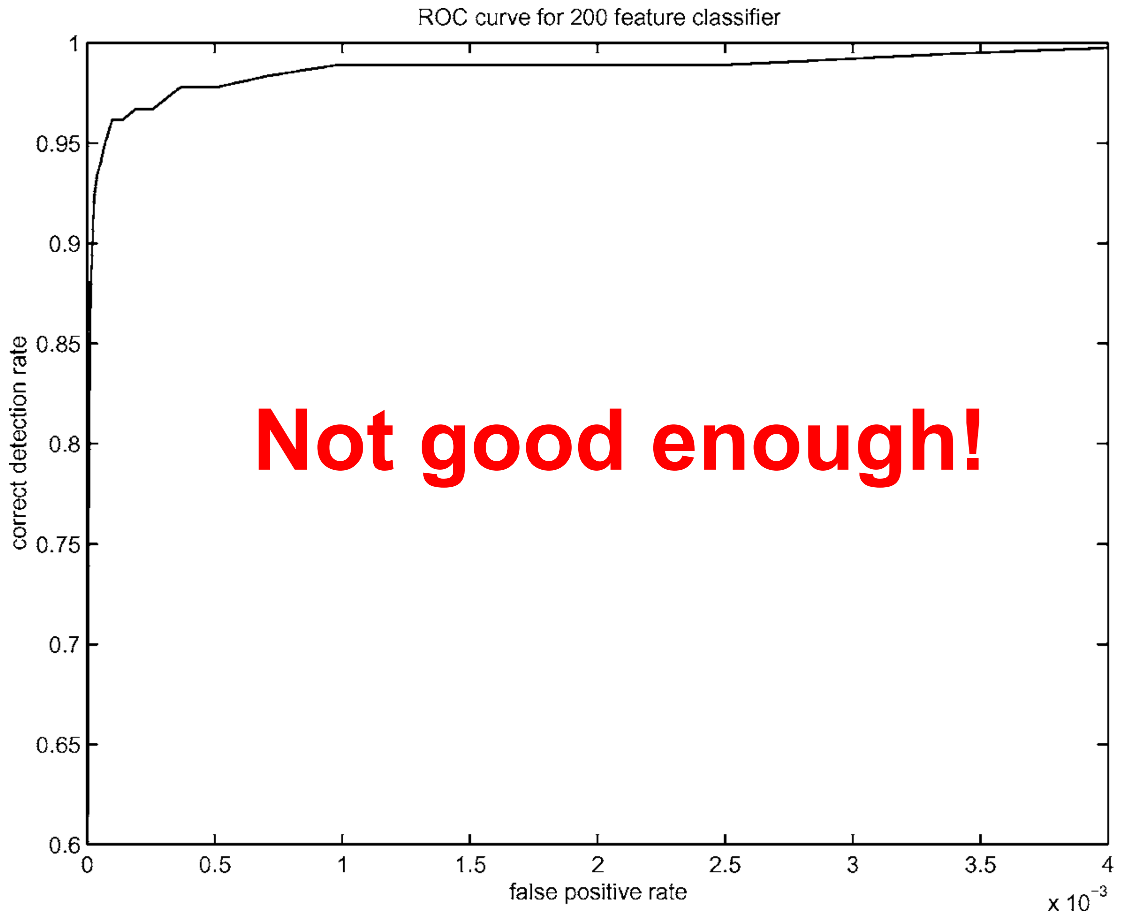
- First two features selected by boosting:



This feature combination can yield 100% detection rate and 50% false positive rate

Boosting for face detection

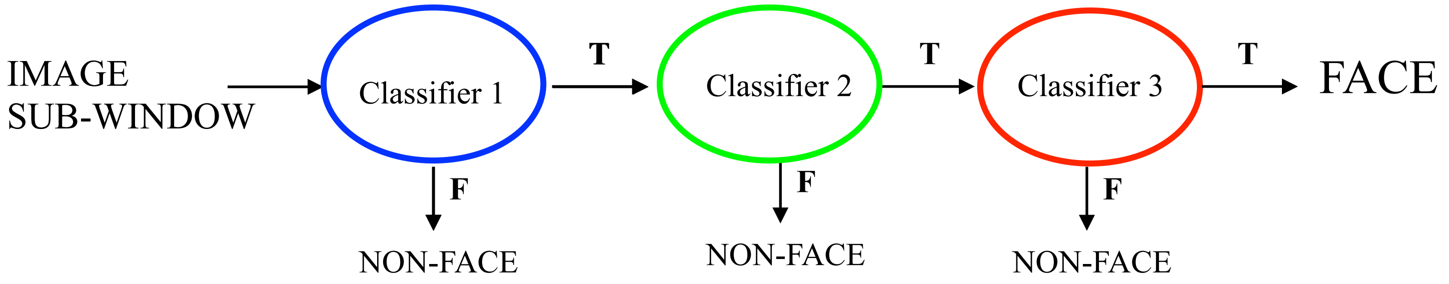
- A 200-feature classifier can yield 95% detection rate and a false positive rate of 1 in 14084



Receiver operating characteristic (ROC) curve

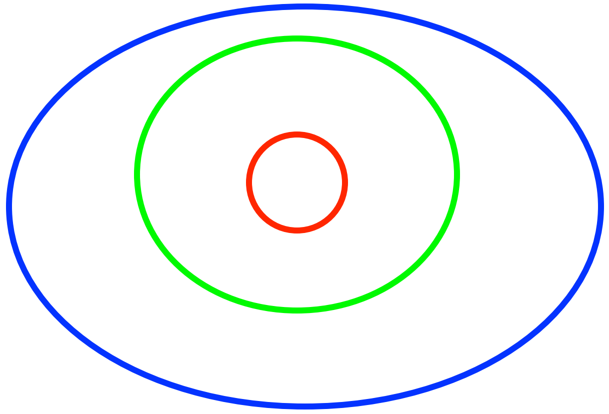
Attentional cascade

- We start with simple classifiers which reject many of the negative sub-windows while detecting almost all positive sub-windows
- Positive response from the first classifier triggers the evaluation of a second (more complex) classifier, and so on
- A negative outcome at any point leads to the immediate rejection of the sub-window

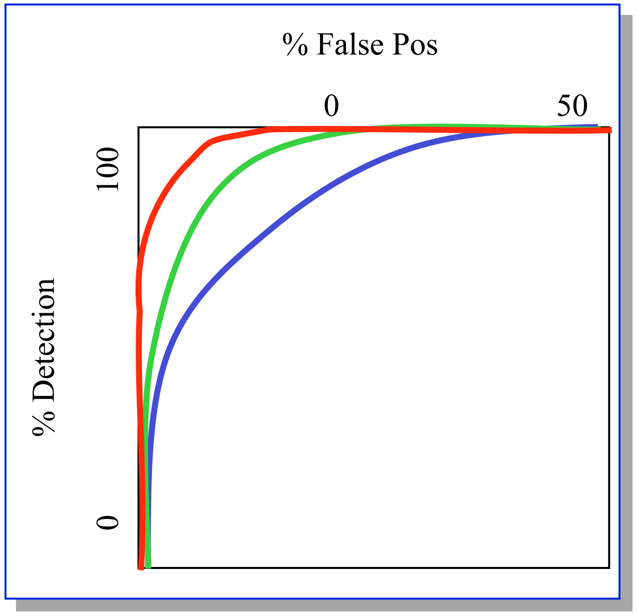


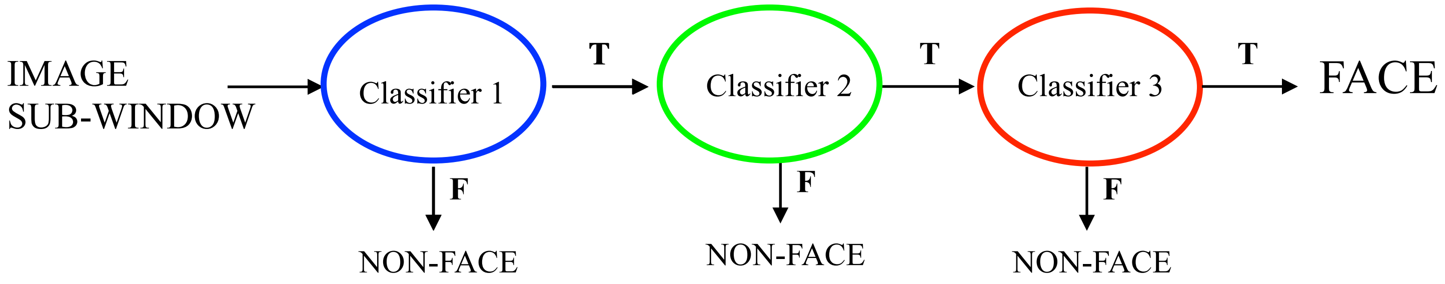
Attentional cascade

- Chain classifiers that are progressively more complex and have lower false positive rates:



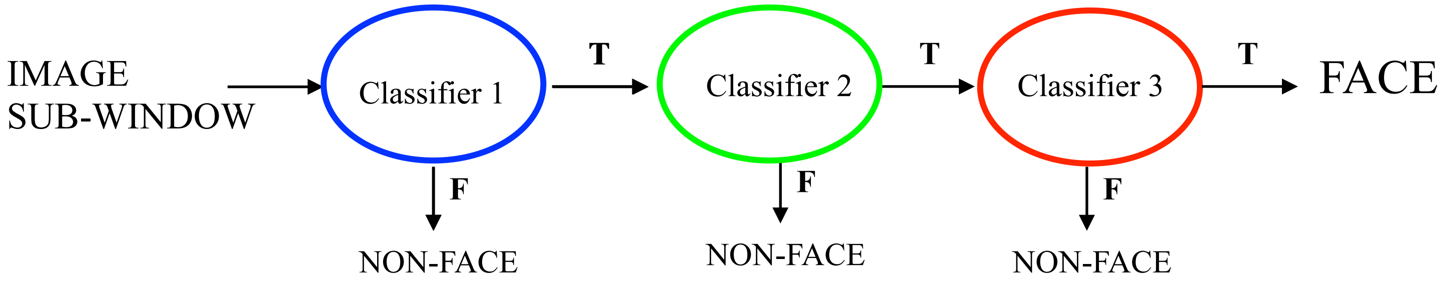
Receiver operating characteristic





Attentional cascade

- The detection rate and the false positive rate of the cascade are found by multiplying the respective rates of the individual stages
- A detection rate of 0.9 and a false positive rate on the order of 10^{-6} can be achieved by a 10-stage cascade if each stage has a detection rate of 0.99 ($0.99^{10} \approx 0.9$) and a false positive rate of about 0.30 ($0.3^{10} \approx 6\times10^{-6}$)



Training the cascade

- Set target detection and false positive rates for each stage
- Keep adding features to the current stage until its target rates have been met
  - Need to lower AdaBoost threshold to maximize detection (as opposed to minimizing total classification error)
  - Test on a *validation set*
- If the overall false positive rate is not low enough, then add another stage
- Use false positives from current stage as the negative training examples for the next stage

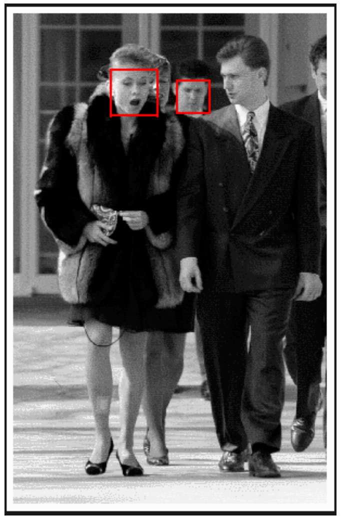
The implemented system

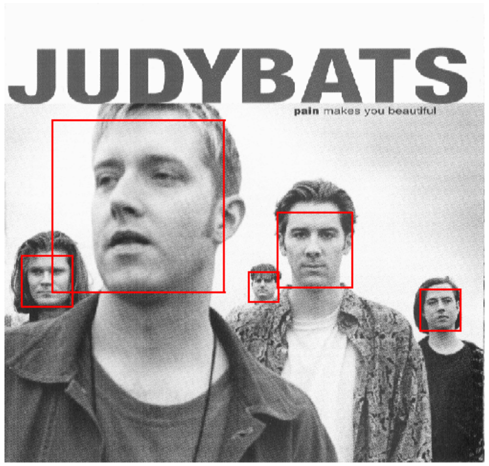
- Training Data
  - 5000 faces
    - All frontal, rescaled to 24x24 pixels
  - 300 million non-faces
    - 9500 non-face images
  - Faces are normalized
    - Scale, translation
- Many variations
  - Across individuals
  - Illumination
  - Pose

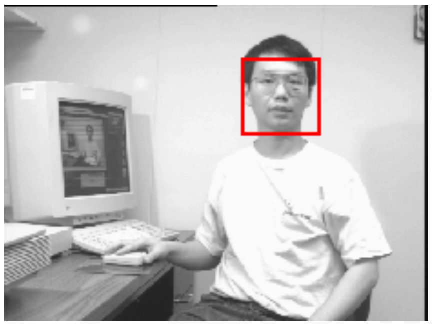
System performance

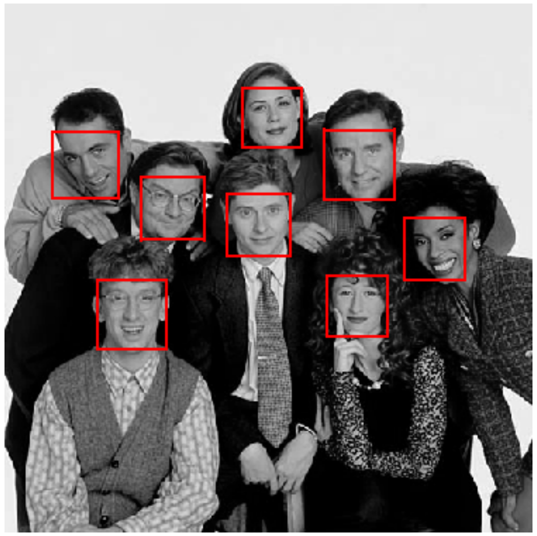
- Training time: “weeks” on 466 MHz Sun workstation
- 38 layers, total of 6061 features
- Average of 10 features evaluated per window on test set
- “On a 700 Mhz Pentium III processor, the face detector can process a 384 by 288 pixel image in about .067 seconds”
    - 15 Hz
    - 15 times faster than previous detector of comparable accuracy (Rowley et al., 1998)

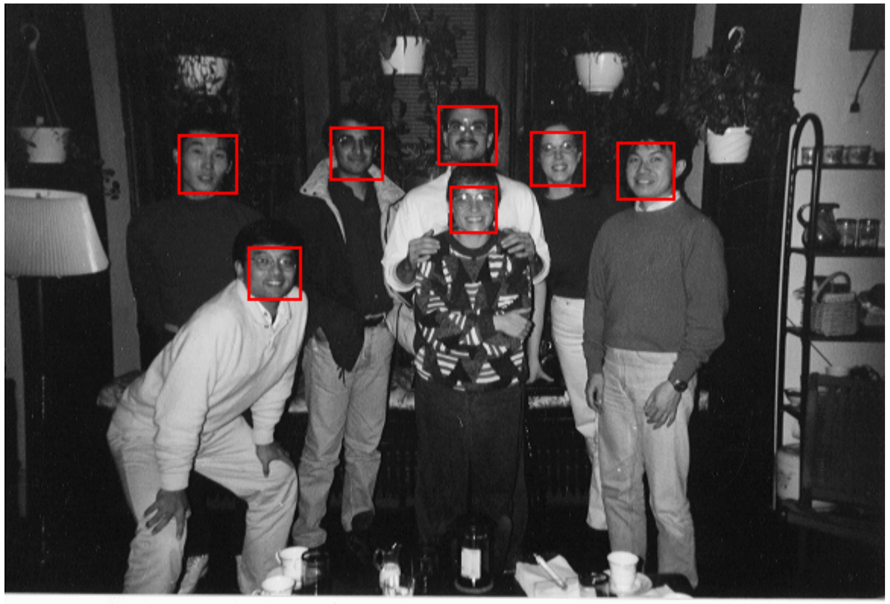
Output of Face Detector on Test Images





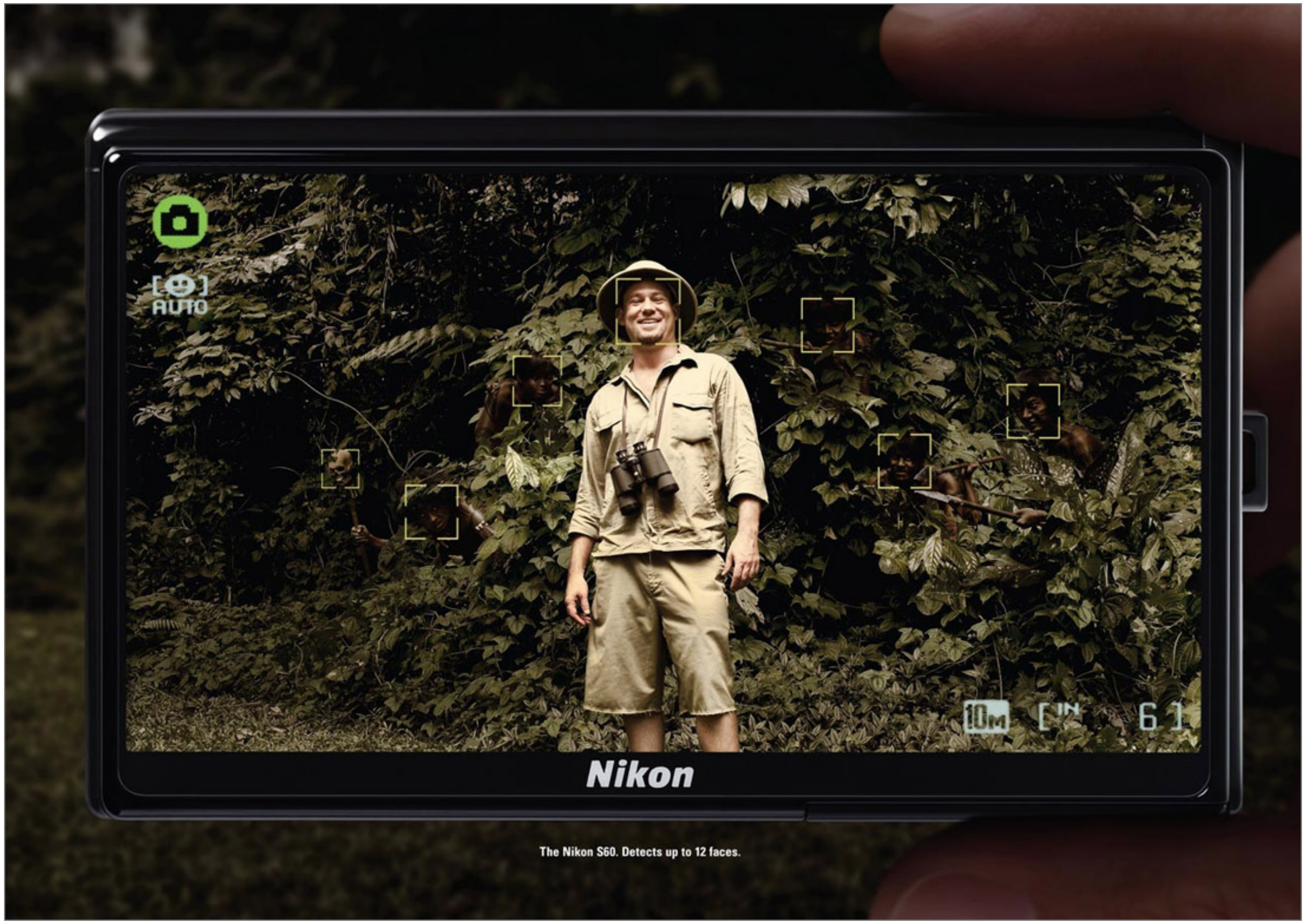






Funny Nikon ads

"The Nikon S60 detects up to 12 faces."



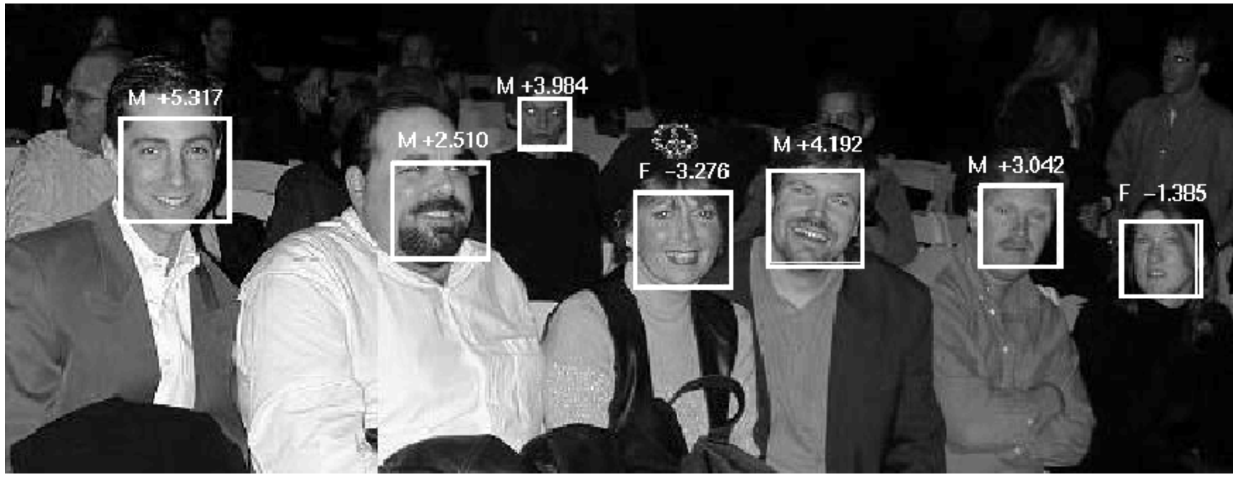
Slide: S. Lazebnik

Other detection tasks

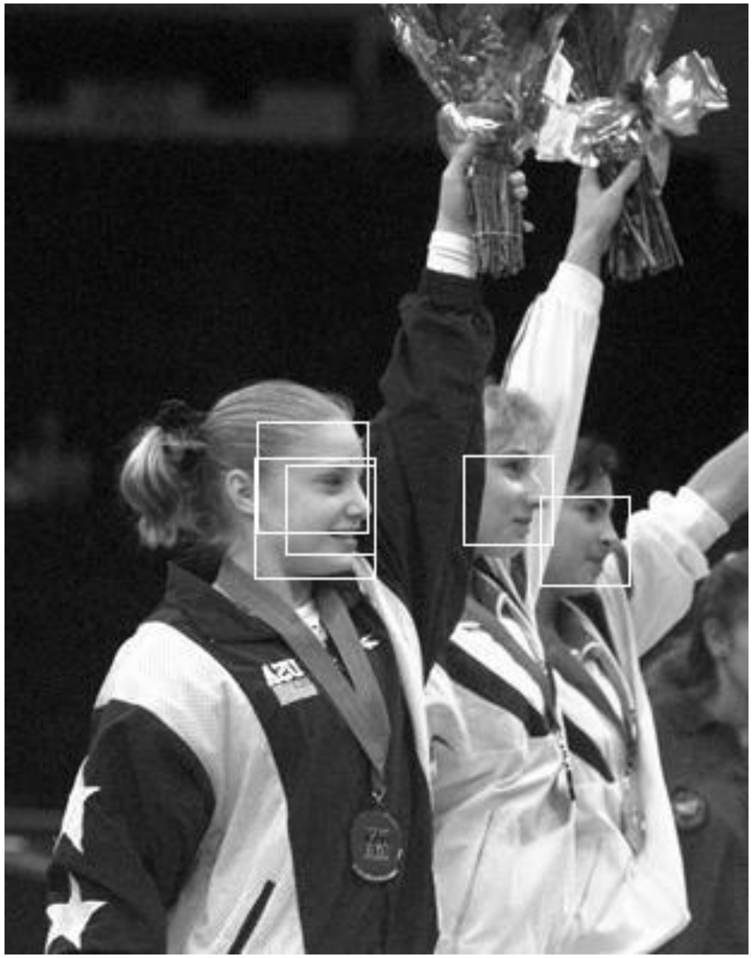
Facial Feature Localization

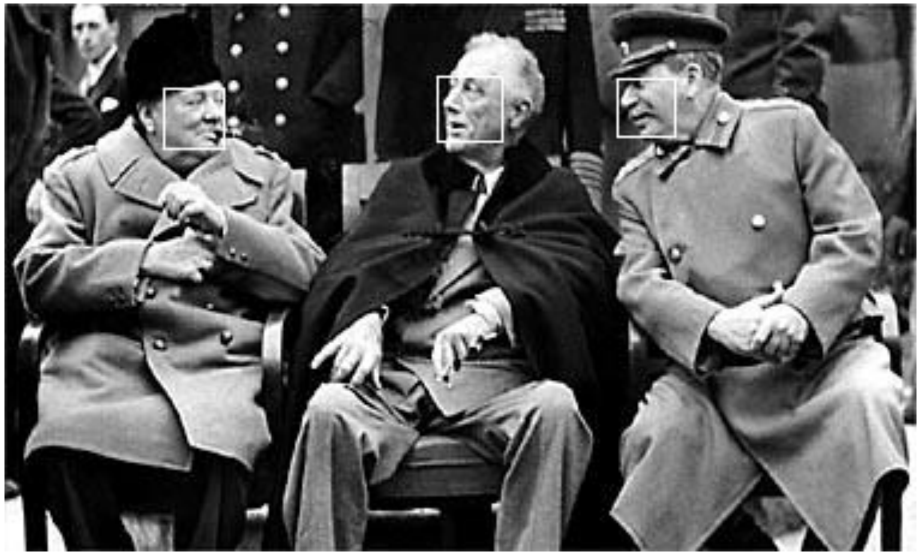
Profile Detection

Male vs. female

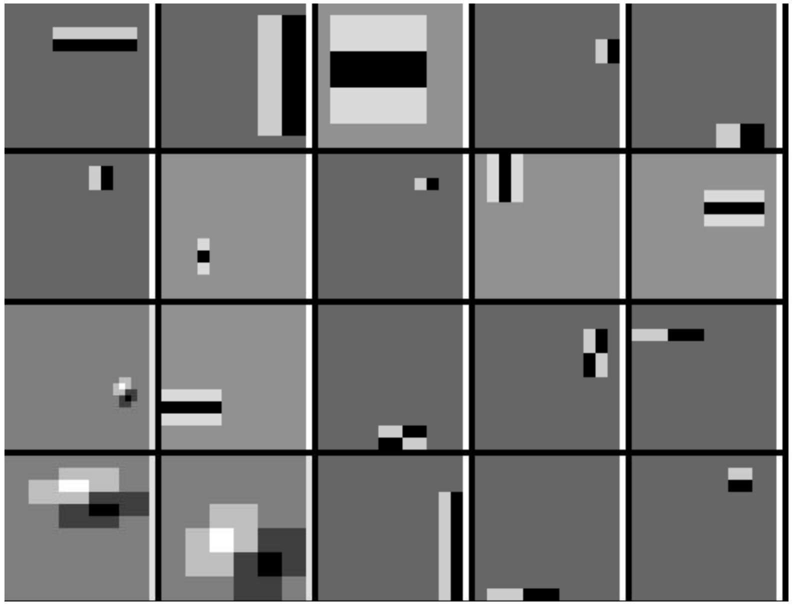


Profile Detection





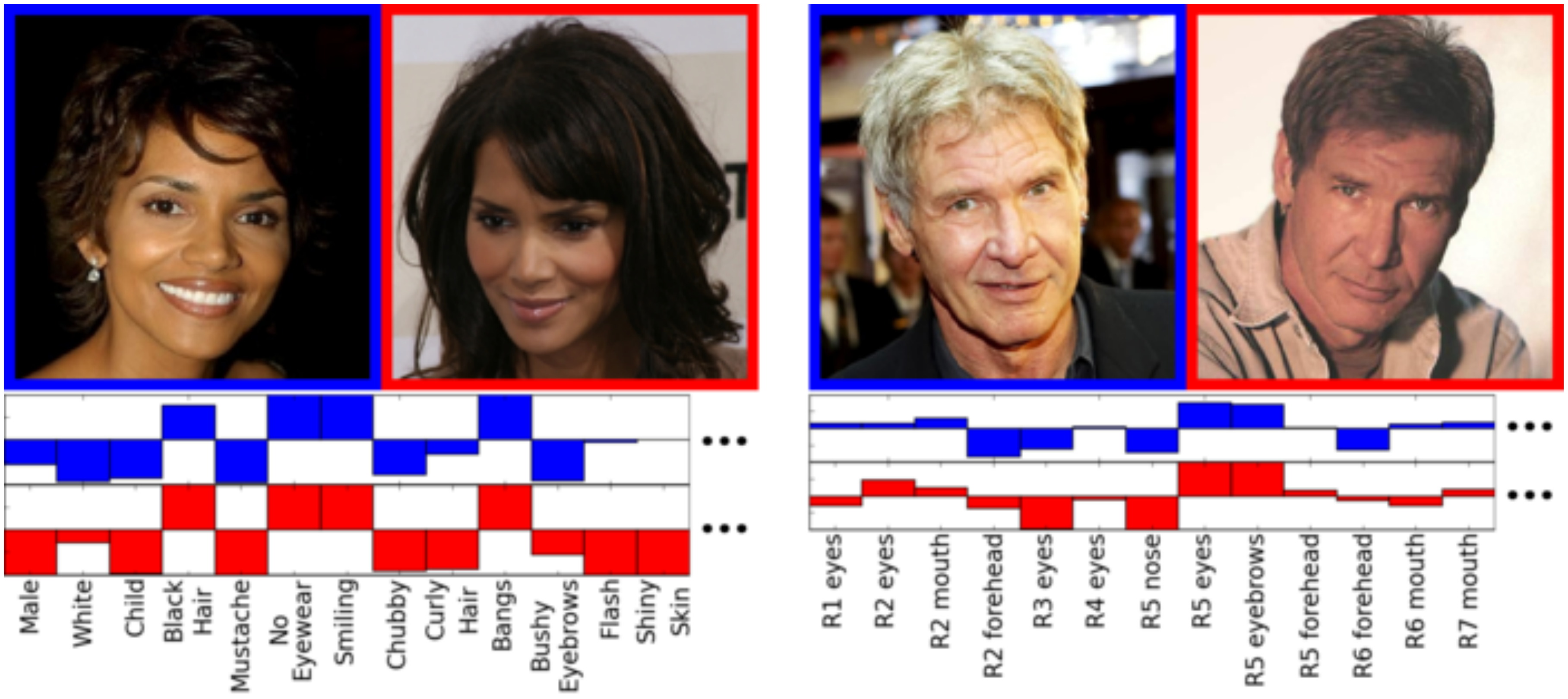
Profile Features



Summary: Viola/Jones detector

- Rectangle features
- Integral images for fast computation
- Boosting for feature selection
- Attentional cascade for fast rejection of negative windows

Face Recognition

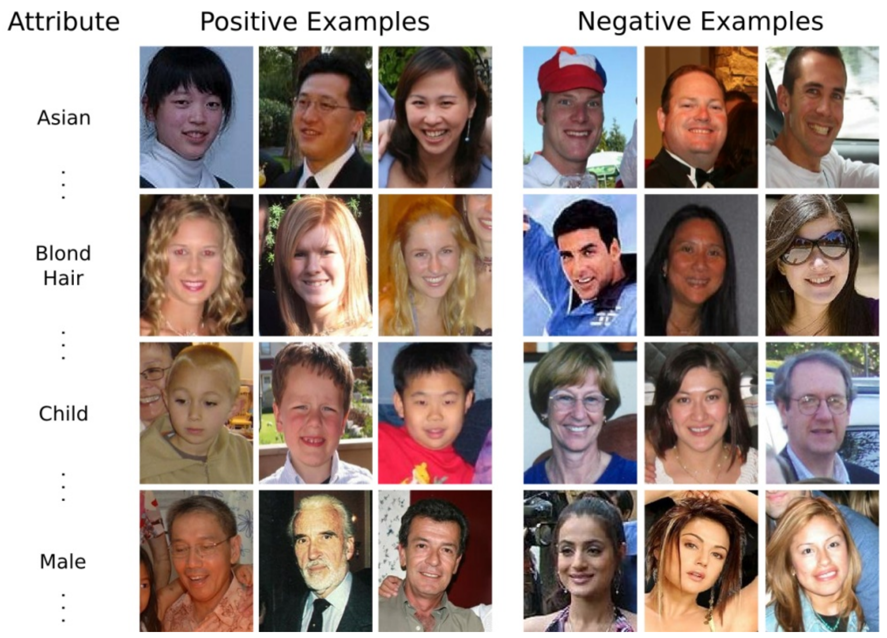


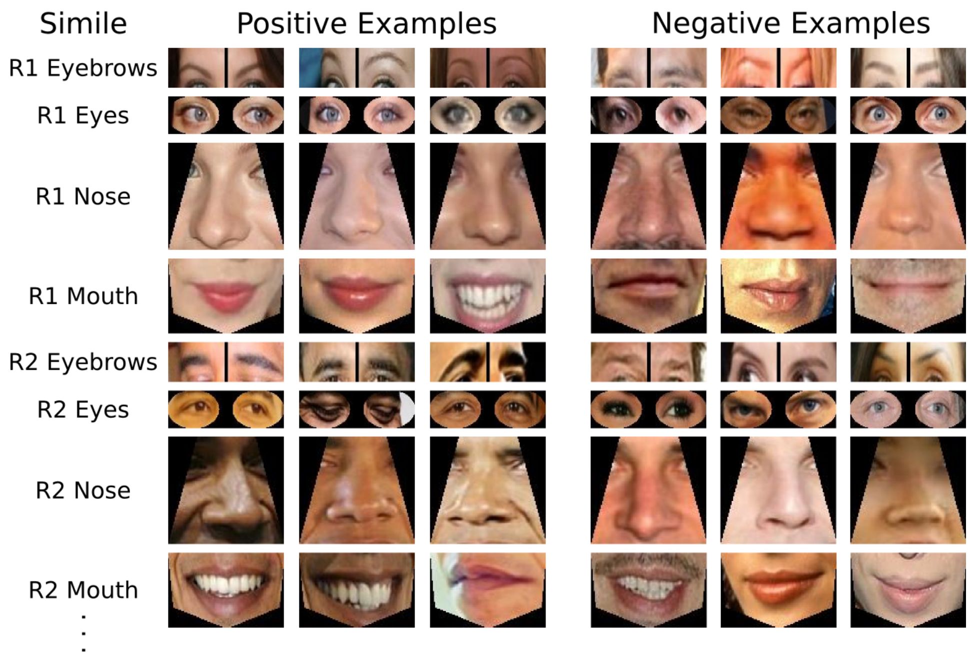
N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar, **"Attribute and Simile Classifiers for Face Verification,"** ICCV 2009

Face Recognition

Attributes for training

Similes for training

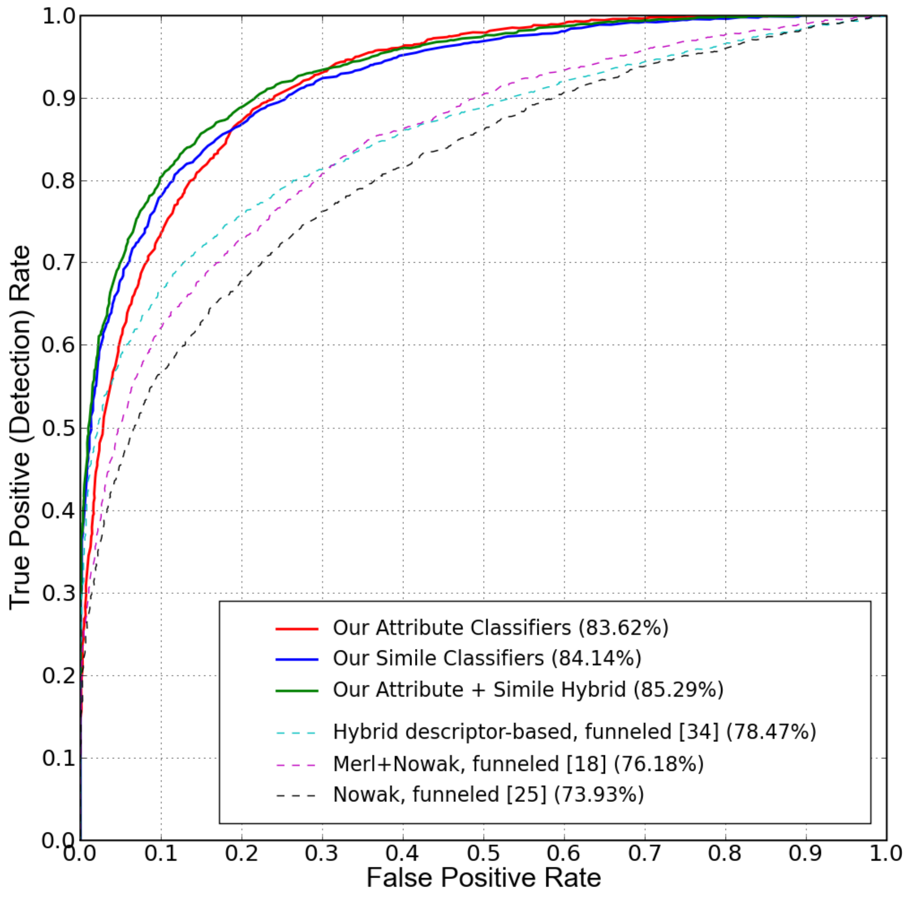




N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar, **"Attribute and Simile Classifiers for Face**

Face Recognition

Results on Labeled Faces in the Wild Dataset



N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar, **"Attribute and Simile Classifiers for Face**