universität innsbruck

Lecture 10. Object Recognition II: Object detection. Instances. Viola and Jones.

703142. Computer Vision

Assoz.Prof. Antonio Rodríguez-Sánchez, PhD.

Outline

Shapes: Snakes

Instance recognition

Large databases

Object detection

Face detection: Viola and Jones

Outline

Shapes: Snakes

Instance recognition

Large databases

Object detection

Face detection: Viola and Jones

Shape representation

- Shape *is all the geometrical information that remains when location, scale and rotational effects are filtered out from an object* (Kendall, 1984)
  - Shape consists of whatever geometric information is invariant to a similarity transform

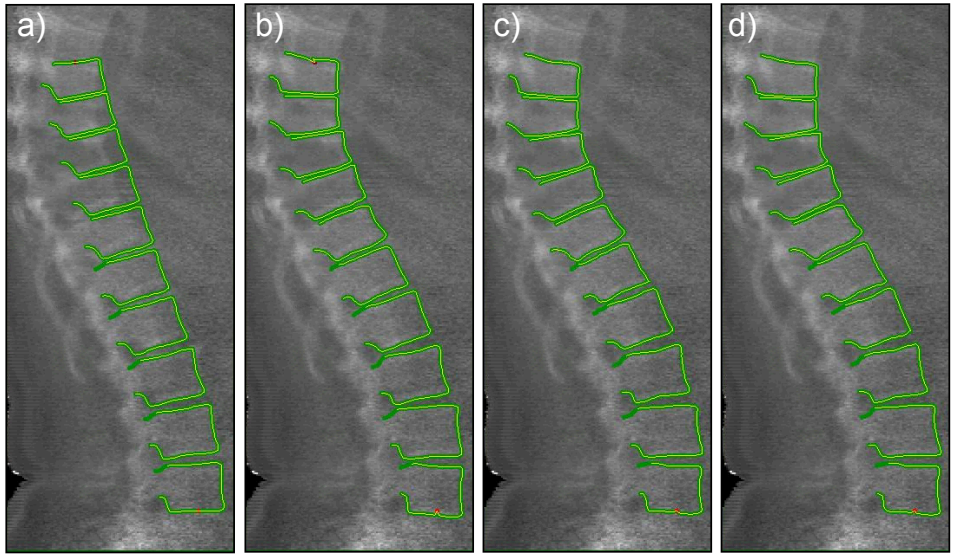
Shape representation

- Shape *is all the geometrical information that remains when location, scale and rotational effects are filtered out from an object* (Kendall, 1984)
  - Shape consists of whatever geometric information is invariant to a similarity transform
- We can define an algebraic expression describing a contour
  - Includes circles, ellipses, parabolas and hyperbolas

$$\begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} \alpha & \beta & \gamma \\ \beta & \delta & \epsilon \\ \gamma & \epsilon & \zeta \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0.$$

Shape representation

- Shape *is all the geometrical information that remains when location, scale and rotational effects are filtered out from an object*
  - Shape consists of whatever geometric information is invariant to a similarity transform
- We can define an algebraic expression describing a contour
  - Includes circles, ellipses, parabolas and hyperbolas
  - Limited application



Shape representation

- Shape *is all the geometrical information that remains when location, scale and rotational effects are filtered out from an object*
  - Shape consists of whatever geometric information is invariant to a similarity transform
- Define shape using landmark points
  - Discrete samples from one or more underlying continuous contours
  - Connectivity varies according to the model
  - Contour can be reconstituted from the landmark points by interpolating between them according to the connectivity

Shape representation

- Define shape using landmark points
  - Discrete samples from one or more underlying continuous contours
  - Connectivity varies according to the model
  - Contour can be reconstituted from the landmark points by interpolating between them according to the connectivity

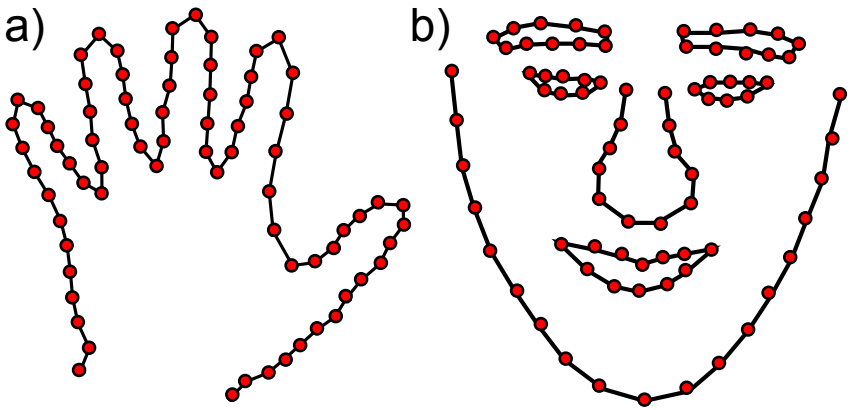


Figure 17.2 Landmark points. Object shape can be represented with sets of landmark points. a) Here the landmark points (red dots) define a single open contour that describes the shape of a hand. b) In this example the landmark points are connected into sets of open and closed contours that describe the regions of the face.

Shape representation

The Rules of Probability

sum rule $$p(X) = \sum_{Y} p(X, Y)$$

product rule $$p(X, Y) = p(Y|X)p(X).$$

Note: $p(X = x_i)$ is usually shortened to $p(x_i)$,

e.g. $p(B=r)$ is $p(B)$ for distribution over random variable B

$p(r)$ is the distribution for the particular value r

Note: $p(X,Y)$ is verbalized as "the probability of X and Y"

Note: $p(Y \mid X)$ is verbalized as "the probability of Y given X"

Note: $p(X)$ is verbalized as "the probability of X"

Shape representation

$$p(Y|X) = \frac{p(X|Y)p(Y)}{p(X)}$$

- $p(X)$ is the **prior** probability
- $p(Y \mid X)$ is the **posterior** probability

- Bayes rule:

$$\underbrace{\frac{p(zebra \mid image)}{p(no\ zebra \mid image)}}_{\text{posterior ratio}} = \underbrace{\frac{p(image \mid zebra)}{p(image \mid no\ zebra)}}_{\text{likelihood ratio}} \cdot \underbrace{\frac{p(zebra)}{p(no\ zebra)}}_{\text{prior ratio}}$$

Shape representation

$$\underbrace{\frac{p(zebra \mid image)}{p(no\ zebra \mid image)}}_{\text{posterior ratio}} = \underbrace{\frac{p(image \mid zebra)}{p(image \mid no\ zebra)}}_{\text{likelihood ratio}} \cdot \underbrace{\frac{p(zebra)}{p(no\ zebra)}}_{\text{prior ratio}}$$

- **Discriminative methods model posterior**
- **Generative methods model likelihood and prior**

$$p(X,Y) = p(Y|X)p(X)$$

Snakes

- Active contour models
  - Provide only very weak a priori geometric information
    - They assume that we know the topology of the contour (i.e., open or closed) and that it is smooth
  - They are suitable for situations where little is known about the contents of the image

Snakes

- Active contour models
  - Provide only very weak a priori geometric information
    - They assume that we know the topology of the contour (i.e., open or closed) and that it is smooth
  - They are suitable for situations where little is known about the contents of the image
  - We will consider a closed contour defined by a set of N 2D landmark points $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_N]$, which are unknown
  - Goal is to find the configuration of landmark points that best explains the shape of an object in the image

Snakes

- Active contour models
  - We will consider a closed contour defined by a set of N 2D landmark points $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_N]$, which are unknown
  - Goal is to find the configuration of landmark points that best explains the shape of an object in the image
  - Construct a generative model for the landmark positions, which is determined by a likelihood term $Pr(\mathbf{x}|\mathbf{W})$
    - Likelihood of the RGB image data $\mathbf{x}$ given the landmark points $\mathbf{W}$ should be high when the landmark points $\mathbf{w}$ lie on or close to edges in the image, and low when they lie in flat regions

Snakes

- Active contour models
  - We will consider a closed contour defined by a set of N 2D landmark points $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_N]$, which are unknown
  - Goal is to find the configuration of landmark points that best explains the shape of an object in the image
  - Construct a generative model for the landmark positions, which is determined by a likelihood term $Pr(\mathbf{x}|\mathbf{W})$

$$Pr(\mathbf{x}|\mathbf{W}) \propto \prod_{n=1}^{N} \exp\left[-(\text{dist}\left[\mathbf{x}, \mathbf{w}_n\right])^2\right]$$

Snakes

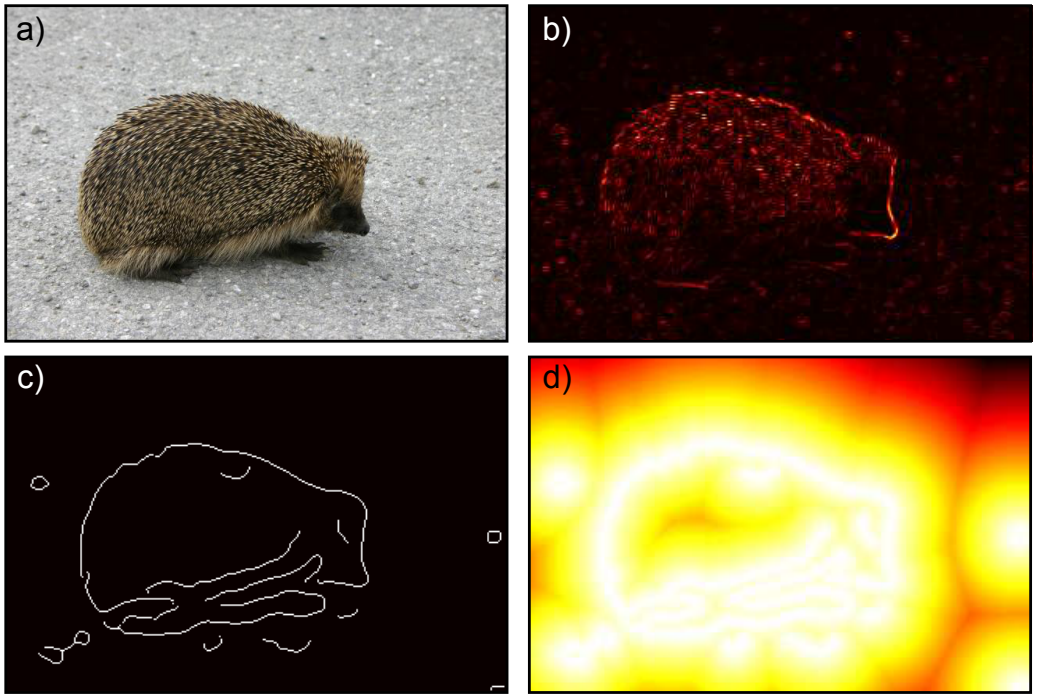


Figure 17.3 Likelihood for landmark points. a) Original image. b) Output of Sobel edge operator - one possible scheme is to assign the landmark points a high likelihood if the Sobel response is strong at their position. This encourages the landmark points to lie on boundaries in the image, but the response is flat in regions away from the boundaries, and this makes it difficult to apply gradient methods to fit the model c) Results of applying Canny edge detector. d) Negative distance from nearest Canny edge. This function is also high at boundaries in the image but varies smoothly in regions away from the boundaries.

$$Pr(\mathbf{x}|\mathbf{W}) \propto \prod_{n=1}^{N} \exp\left[-(\text{dist}\left[\mathbf{x}, \mathbf{w}_n\right])^2\right]$$

Snakes

- Active contour models
  - We will consider a closed contour defined by a set of N 2D landmark points $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_N]$, which are unknown
  - Goal is to find the configuration of landmark points that best explains the shape of an object in the image
  - Construct a generative model for the landmark positions, which is determined by a likelihood term $Pr(\mathbf{x}|\mathbf{W})$
  - Define a prior that favors smooth contours with low curvature

$$Pr(\mathbf{W}) \propto \prod_{n=1}^{N} \exp\left[\alpha \,\text{space}[\mathbf{w}, n] + \beta \,\text{curve}[\mathbf{w}, n]\right]$$

Snakes

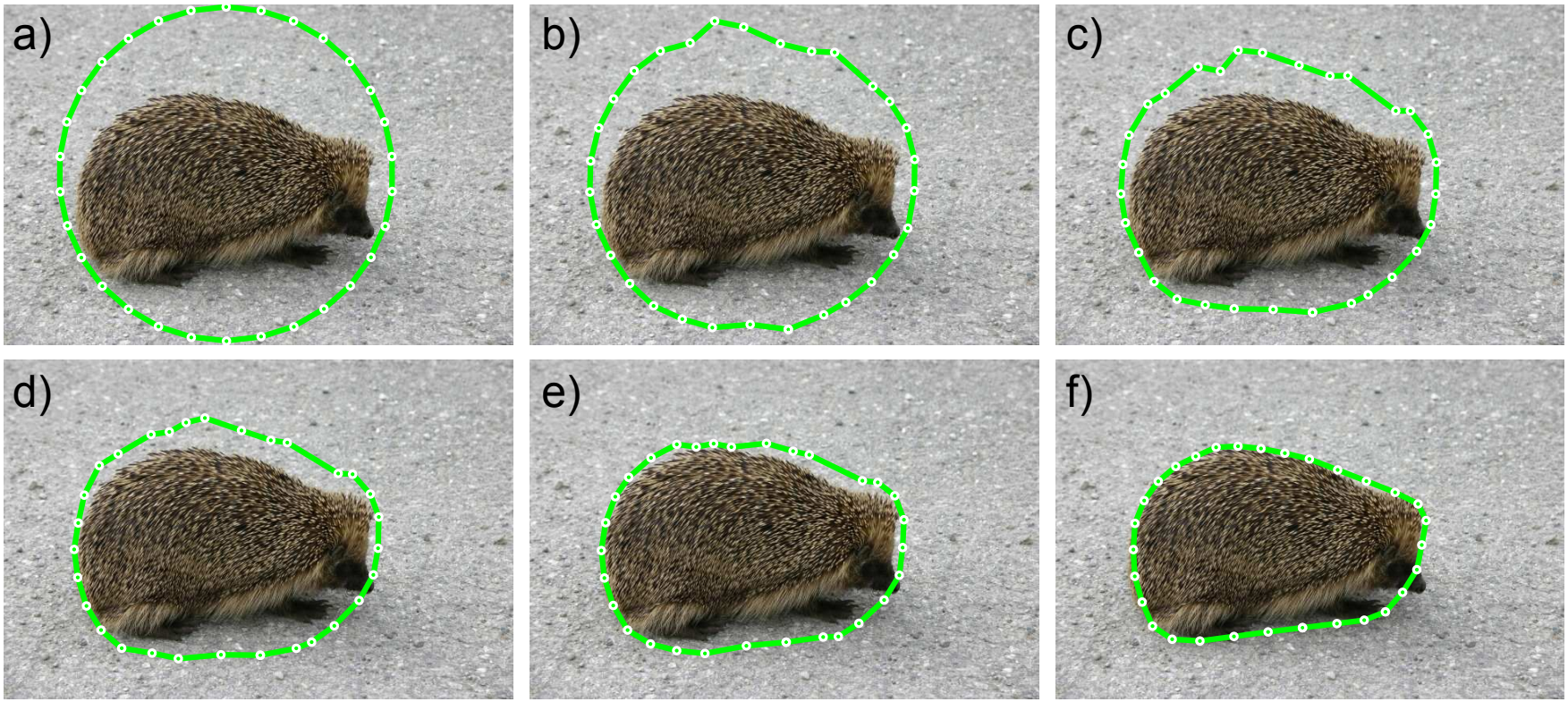
- Active contour models
  - We will consider a closed contour defined by a set of N 2D landmark points $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_N]$, which are unknown
  - Goal is to find the configuration of landmark points that best explains the shape of an object in the image
  - Construct a generative model for the landmark positions, which is determined by a likelihood term $Pr(\mathbf{x}|\mathbf{W})$
  - Define a prior that favors smooth contours with low curvature

$$Pr(\mathbf{W}) \propto \prod_{n=1}^{N} \exp\left[\alpha \,\text{space}[\mathbf{w}, n] + \beta \,\text{curve}[\mathbf{w}, n]\right]$$

$$\text{space}[\mathbf{w}, n] = -\left(\frac{\sum_{n=1}^{N}\sqrt{(\mathbf{w}_n - \mathbf{w}_{n-1})^T(\mathbf{w}_n - \mathbf{w}_{n-1})}}{N} - \sqrt{(\mathbf{w}_n - \mathbf{w}_{n-1})^T(\mathbf{w}_n - \mathbf{w}_{n-1})}\right)^2, \quad (17.5)$$

$$\text{curve}[\mathbf{w}, n] = -(\mathbf{w}_{n-1} - 2\mathbf{w}_n + \mathbf{w}_{n+1})^T(\mathbf{w}_{n-1} - 2\mathbf{w}_n + \mathbf{w}_{n+1})$$

Snakes



$$\hat{\mathbf{W}} = \underset{\mathbf{W}}{\operatorname{argmax}}\,[Pr(\mathbf{W}|\mathbf{x})] = \underset{\mathbf{W}}{\operatorname{argmax}}\,[Pr(\mathbf{x}|\mathbf{W})Pr(\mathbf{W})]$$
$$= \underset{\mathbf{W}}{\operatorname{argmax}}\,[\log[Pr(\mathbf{x}|\mathbf{W})] + \log[Pr(\mathbf{W})]]$$

Snakes

- Limitations
  - It is not useful when we know the object shape but not its position within the image
  - It is not useful when we know the class of the object (e.g. a face) but not the particular instance (e.g., whose face)
  - The snake model is 2D and does not understand that some contours are created by projecting a 3D surface through the camera model
  - It cannot model articulated objects such as the human body

Outline

Shapes: Snakes

Instance recognition

Large databases

Object detection

Face detection: Viola and Jones

Instance recognition

- To recognize one or more instances of some known objects, the recognition system:
  - First extracts a set of interest points in each database image and
  - stores the associated descriptors (and original positions) in an indexing structure

Instance recognition

- To recognize one or more instances of some known objects, the recognition system:
  - First extracts a set of interest points in each database image and
  - stores the associated descriptors (and original positions) in an indexing structure
  - At recognition time, features are extracted from the new image and compared against the stored object features
  - Whenever a sufficient number of matching features (say, three or more) are found for a given object, the system then invokes a match verification stage,
  - whose job is to determine whether the spatial arrangement of matching features is consistent with those in the database image

Instance recognition

- To recognize one or more instances of some known objects, the recognition system:
  - First extracts a set of interest points in each database image and
  - stores the associated descriptors (and original positions) in an indexing structure
  - At recognition time, features are extracted from the new image and compared against the stored object features
  - Whenever a sufficient number of matching features (say, three or more) are found for a given object, the system then invokes a match verification stage,
  - whose job is to determine whether the spatial arrangement of matching features is consistent with those in the database image

Geometric alignment

- Because images can be highly cluttered and similar features may belong to several objects, the original set of feature matches can have a large number of outliers

Geometric alignment

- Because images can be highly cluttered and similar features may belong to several objects, the original set of feature matches can have a large number of outliers
- Solution (Lowe, 2004): Use Hough transform
  - Accumulate votes for likely geometric transformations
  - Affine transformation between the database object and the collection of scene features

Hough transform

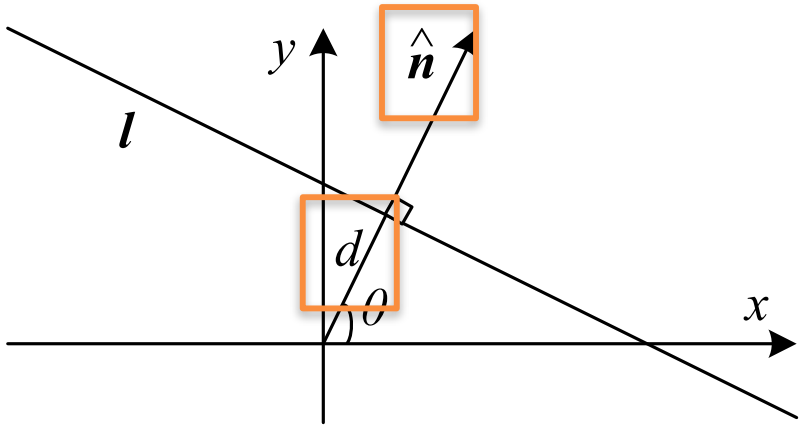
- Group edgels into line segments even across gaps and occlusions
  - Curve approximation with polylines can often lead to successful line extraction, lines in the real world are sometimes broken up into disconnected components

Hough transform

- Edges “vote” for plausible line locations
  - We must first choose a suitable representation

Hough transform

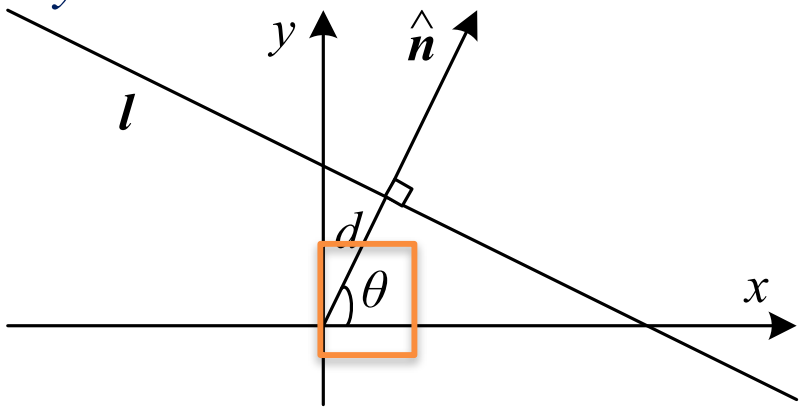
- Edges "vote" for plausible line locations
  - We must first choose a suitable representation
  - The line normal $\hat{n}$ points in the same direction as the image gradient $J(x) = \nabla I(x)$



Hough transform

- Edges "vote" for plausible line locations
  - We must first choose a suitable representation
  - The line normal $\hat{n}$ points in the same direction as the image gradient $J(x) = \nabla I(x)$
  - convert the normal vector into an angle

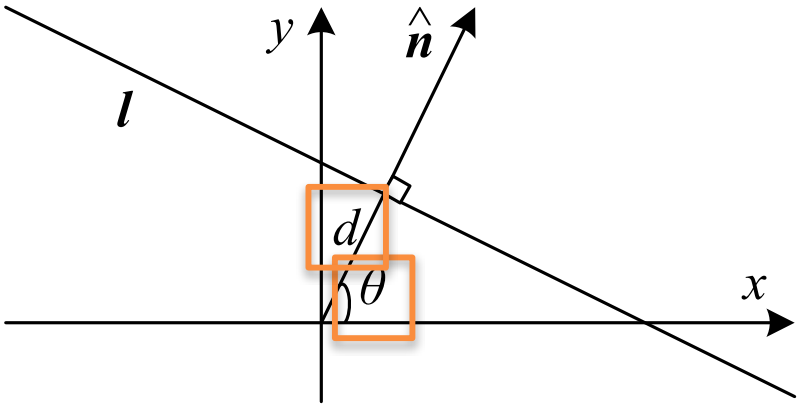
$$\theta = tan^{-1} n_y / n_x$$



Hough transform

- Edges "vote" for plausible line locations
  - We must first choose a suitable representation
  - The range of possible (θ, d) values is

$$[-180°, 180°] \times [-\sqrt{2}, \sqrt{2}]$$



Hough transform

- Edges “vote” for plausible line locations

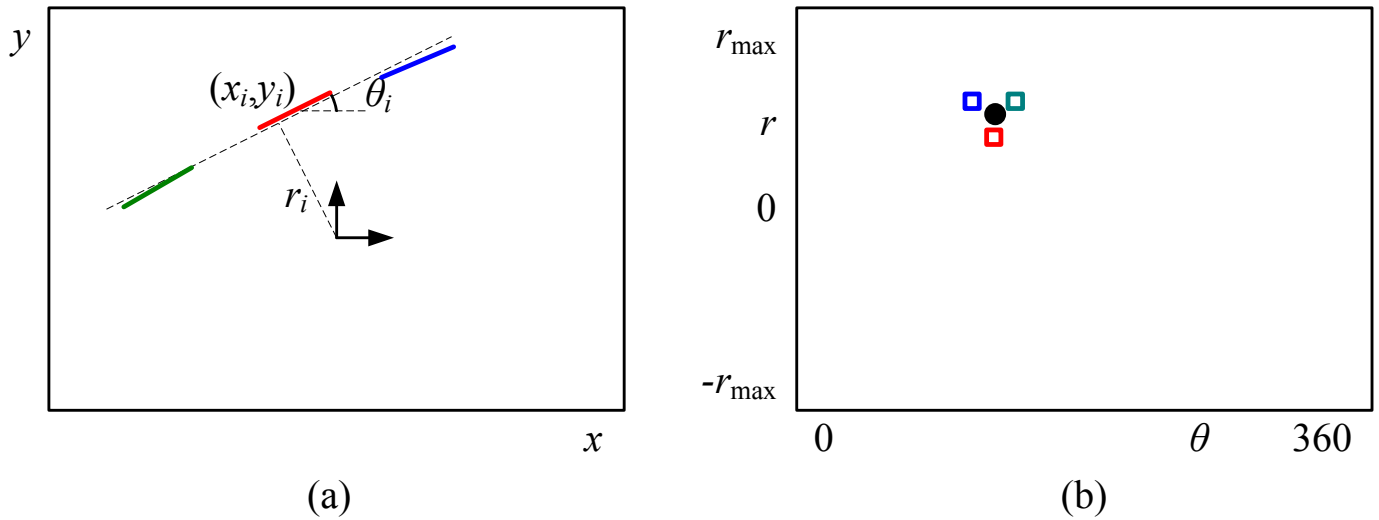


Figure 4.42 Oriented Hough transform: (a) an edgel re-parameterized in polar (r, θ) coordinates, with $\hat{\boldsymbol{n}}_i = (\cos\theta_i, \sin\theta_i)$ and $r_i = \hat{\boldsymbol{n}}_i \cdot \boldsymbol{x}_i$; (b) (r, θ) accumulator array, showing the votes for the three edgels marked in red, green, and blue.

Hough transform

procedure *Hough*$(\{(x, y, \theta)\})$:

1. Clear the accumulator array.
2. For each detected edgel at location (x, y) and orientation $\theta = \tan^{-1} n_y/n_x$, compute the value of
$$d = x\, n_x + y\, n_y$$
and increment the accumulator corresponding to (θ, d).
3. Find the peaks in the accumulator corresponding to lines.
4. Optionally re-fit the lines to the constituent edgels.

Algorithm 4.2 Outline of a Hough transform algorithm based on oriented edge segments.

Geometric alignment

- Because images can be highly cluttered and similar features may belong to several objects, the original set of feature matches can have a large number of outliers
- Identify objects with the fewest possible number of feature matches

Geometric alignment

- Because images can be highly cluttered and similar features may belong to several objects, the original set of feature matches can have a large number of outliers
- SIFT features carry with them their own location, scale, and orientation

Geometric alignment

- The Hough transform identifies clusters of features with a consistent interpretation by using each feature to vote for all object poses that are consistent with the feature
  - When clusters of features are found to vote for the same pose of an object, the probability of the interpretation being correct is much higher than for any single feature
- SIFT features carry with them their own location, scale, and orientation

Geometric alignment

- The Hough transform identifies clusters of features with a consistent interpretation by using each feature to vote for all object poses that are consistent with the feature
- SIFT features carry with them their own location, scale, and orientation
  - Each matched keypoint in the database has a record of the keypoint's parameters relative to the training image in which it was found
  - We can create a Hough transform entry predicting the model location, orientation, and scale from the match hypothesis

Geometric alignment

- The Hough transform is used to identify all clusters with at least 3 entries in a bin
- An affine transformation correctly accounts for 3D rotation of a planar surface under orthographic projection

Geometric alignment

- The Hough transform is used to identify all clusters with at least 3 entries in a bin
- An affine transformation correctly accounts for 3D rotation of a planar surface under orthographic projection
- A more general solution would be to solve for the fundamental matrix

Geometric alignment

- The Hough transform is used to identify all clusters with at least 3 entries in a bin
- An affine transformation correctly accounts for 3D rotation of a planar surface under orthographic projection
- A more general solution would be to solve for the fundamental matrix
  - A fundamental matrix solution requires at least 7 point matches as compared to only 3 for the affine solution

Geometric alignment

- The Hough transform is used to identify all clusters with at least 3 entries in a bin
- An affine transformation correctly accounts for 3D rotation of a planar surface under orthographic projection
- Use affine transformation model

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Geometric alignment

- The Hough transform is used to identify all clusters with at least 3 entries in a bin
- An affine transformation correctly accounts for 3D rotation of a planar surface under orthographic projection
- Use affine transformation model

$$\begin{bmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \\ & & \cdots & & & \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_x \\ t_y \end{bmatrix} = \begin{bmatrix} u \\ v \\ \vdots \end{bmatrix}$$

Geometric alignment

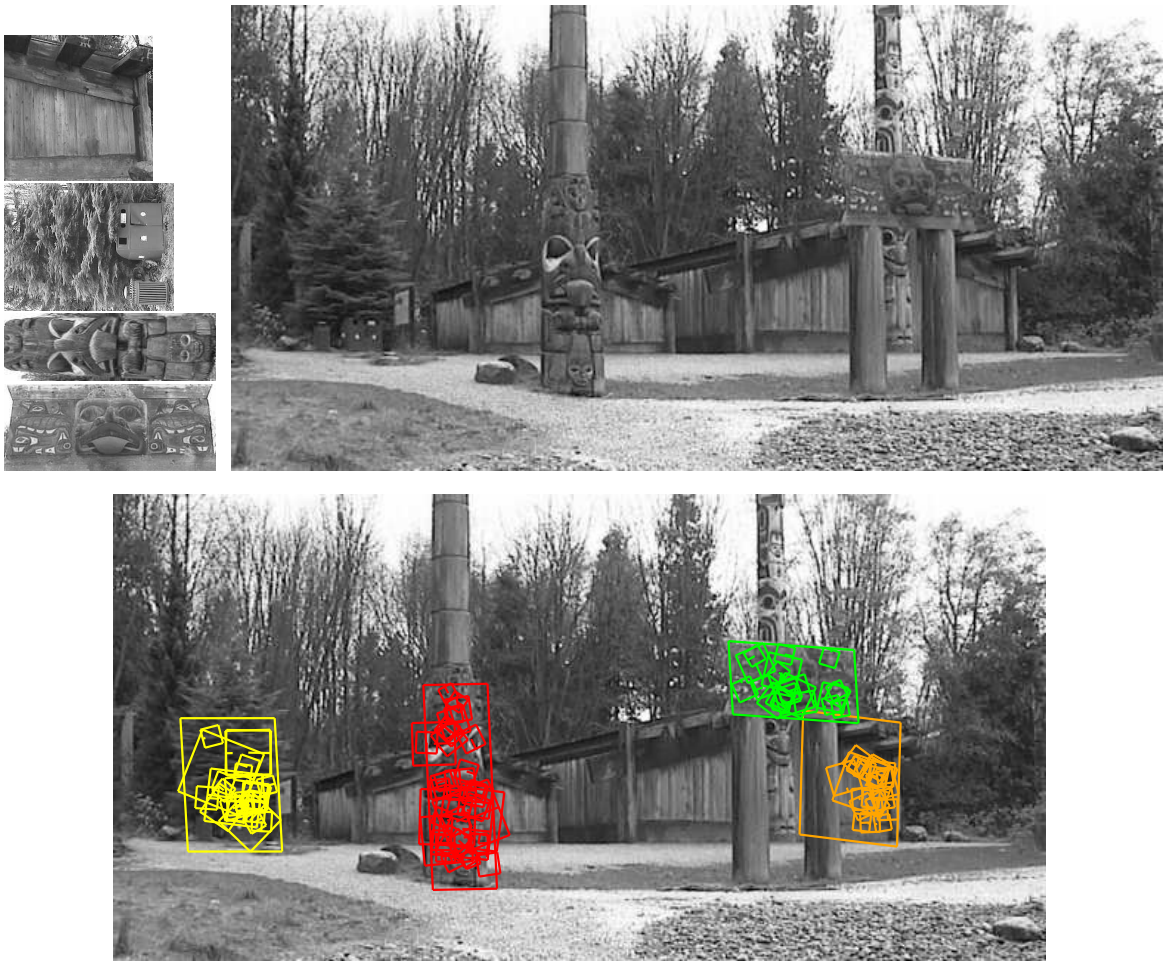


Figure 13: This example shows location recognition within a complex scene. The training images for locations are shown at the upper left and the 640x315 pixel test image taken from a different viewpoint is on the upper right. The recognized regions are shown on the lower image, with keypoints shown as squares and an outer parallelogram showing the boundaries of the training images under the affine transform used for recognition.

Outline

Shapes: Snakes

Instance recognition

Large databases

Object detection

Face detection: Viola and Jones

Large databases

- As the number of objects in the database starts to grow large (say, millions of objects or video frames being searched), the time it takes to match a new image against each database image can become prohibitive

Large databases

- As the number of objects in the database starts to grow large (say, millions of objects or video frames being searched), the time it takes to match a new image against each database image can become prohibitive
  - Instead of comparing the images one at a time, techniques are needed to quickly narrow down the search to a few likely images, which can then be compared using a more detailed and conservative verification stage

Large databases

- Information Retrieval
  - Re-compute an inverted index between individual words and the documents (or Web pages or news stories) where they occur
  - The frequency of occurrence of particular words in a document is used to quickly find documents that match a particular query

Large databases

- Sivic and Zisserman (2009) Video Google
  - Affine invariant features are first detected in all the video frames they are indexing using both shape adapted regions around Harris feature points
  - 128-dimensional SIFT descriptors are computed from each normalized region

Large databases

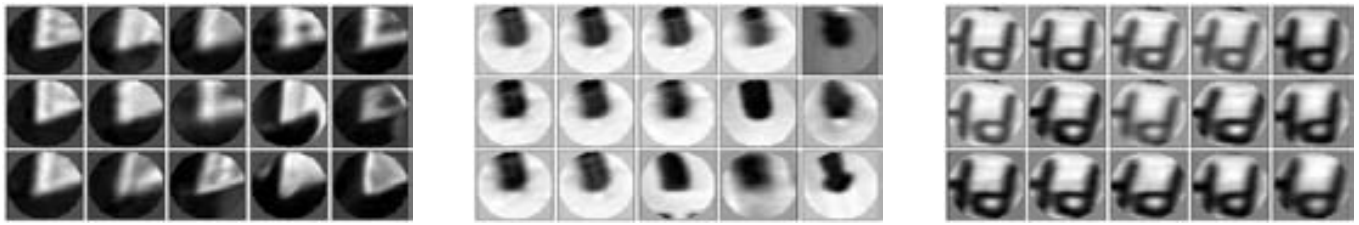
- Sivic and Zisserman (2009) Video Google
  - Affine invariant features are first detected in all the video frames they are indexing using both shape adapted regions around Harris feature points
  - 128-dimensional SIFT descriptors are computed from each normalized region
  - High-dimensional feature descriptors that occur in each image must first be mapped into discrete visual words
    - E.g. mapping using k-means clustering

Large databases

- Sivic and Zisserman (2009) Video Google
  - Affine invariant features are first detected in all the video frames they are indexing using both shape adapted regions around Harris feature points
  - 128-dimensional SIFT descriptors are computed from each normalized region
  - High-dimensional feature descriptors that occur in each image must first be mapped into discrete visual words
    - E.g. mapping using k-means clustering
  - The frequency of occurrence of particular words in a document is used to quickly find documents that match a particular query

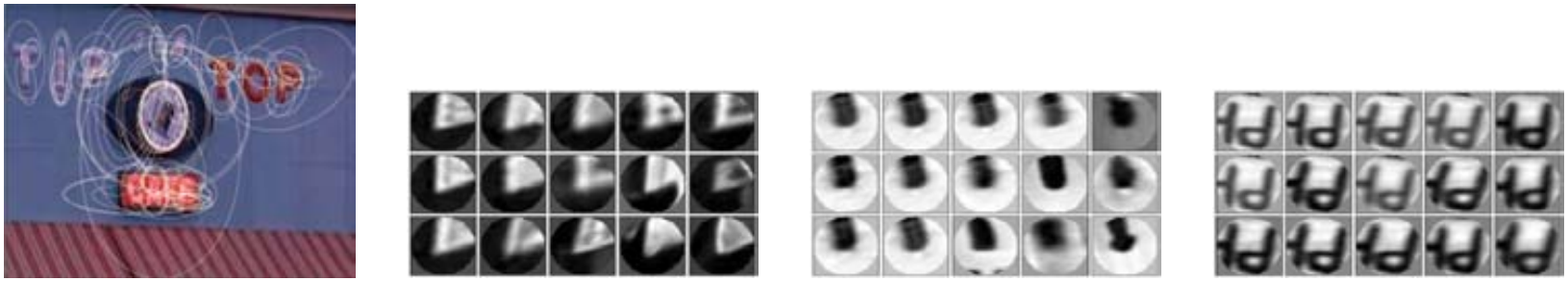
Large databases

- Sivic and Zisserman (2009) Video Google
  - High-dimensional feature descriptors that occur in each image must first be mapped into discrete visual words
    - Vector quantization, e.g. mapping using k-means clustering
  - The frequency of occurrence of particular words in a document is used to quickly find documents that match a particular query



Large databases

- Sivic and Zisserman (2009) Video Google
  - At visual query time, each feature in a new query region is mapped to its corresponding visual word
  - Once a query image or region has been mapped into its constituent visual words, likely matching images or video frames must then be retrieved from the database



Large databases

- Sivic and Zisserman (2009) Video Google
  - At visual query time, each feature in a new query region is mapped to its corresponding visual word
  - Once a query image or region has been mapped into its constituent visual words, likely matching images or video frames must then be retrieved from the database
    - Information retrieval systems do this by matching word distributions (term frequencies) n_{id}/n_d between the query and target documents
    - To downweight words that occur frequently and to focus the search on rarer (and hence, more informative) terms, an inverse document frequency weighting $\log N/N_i$ is applied

Large databases

- Sivic and Zisserman (2009) Video Google
  - Once a query image or region has been mapped into its constituent visual words, likely matching images or video frames must then be retrieved from the database
    - Information retrieval systems do this by matching word distributions (term frequencies) n_{id}/n_d between the query and target documents
    - To downweight words that occur frequently and to focus the search on rarer (and hence, more informative) terms, an inverse document frequency weighting $\log N/N_i$ is applied
    - *tf-idf*: Term frequency-inverse document frequency

$$t_i = \frac{n_{id}}{n_d} log \frac{N}{N_i}$$

Large databases

- Sivic and Zisserman (2009) Video Google
  - Once a query image or region has been mapped into its constituent visual words, likely matching images or video frames must then be retrieved from the database
    - Information retrieval systems do this by matching word distributions (term frequencies) n_{id}/n_d between the query and target documents
    - To downweight words that occur frequently and to focus the search on rarer (and hence, more informative) terms, an inverse document frequency weighting $\log N/N_i$ is applied
    - *tf-idf*: Term frequency-inverse document frequency

$$t_i = \frac{n_{id}}{n_d} log \frac{N}{N_i}$$

    - Each document (or query region) is represented by its *tf-idf* vector

$$\mathbf{t} = (t_1, \ldots, t_i, \ldots t_m)$$

Large databases

- Sivic and Zisserman (2009) Video Google
  - Once a query image or region has been mapped into its constituent visual words, likely matching images or video frames must then be retrieved from the database
    - Information retrieval systems do this by matching word distributions (term frequencies) n_{id}/n_d between the query and target documents
    - To downweight words that occur frequently and to focus the search on rarer (and hence, more informative) terms, an inverse document frequency weighting $\log N/N_i$ is applied
    - *tf-idf*: Term frequency-inverse document frequency

$$t_i = \frac{n_{id}}{n_d} log \frac{N}{N_i}$$

    - The similarity between two documents is measured by the dot product between their corresponding normalized vectors

$$\hat{\mathbf{t}} = \mathbf{t}/\|\mathbf{t}\|$$

Large databases

1. **Vocabulary construction (off-line)**

   (a) Extract affine covariant regions from each database image.

   (b) Compute descriptors and optionally whiten them to make Euclidean distances meaningful (Sivic and Zisserman 2009).

   (c) Cluster the descriptors into visual words, either using k-means (Sivic and Zisserman 2009), hierarchical clustering (Nistér and Stewénius 2006), or randomized k-d trees (Philbin, Chum, Isard *et al.* 2007).

   (d) Decide which words are too common and put them in the stop list.

Large databases

1. **Vocabulary construction (off-line)**

   (a) Extract affine covariant regions from each database image.

   (b) Compute descriptors and optionally whiten them to make Euclidean distances meaningful (Sivic and Zisserman 2009).

   (c) Cluster the descriptors into visual words, either using k-means (Sivic and Zisserman 2009), hierarchical clustering (Nistér and Stewénius 2006), or randomized k-d trees (Philbin, Chum, Isard *et al.* 2007).

   (d) Decide which words are too common and put them in the stop list.

2. **Database construction (off-line)**

   (a) Compute term frequencies for the visual word in each image, document frequencies for each word, and normalized *tf-idf* vectors for each document.

   (b) Compute inverted indices from visual words to images (with word counts).

Large databases

2. **Database construction (off-line)**

   (a) Compute term frequencies for the visual word in each image, document frequencies for each word, and normalized *tf-idf* vectors for each document.

   (b) Compute inverted indices from visual words to images (with word counts).

3. **Image retrieval (on-line)**

   (a) Extract regions, descriptors, and visual words, and compute a *tf-idf* vector for the query image or region.

   (b) Retrieve the top image candidates, either by exhaustively comparing sparse *tf-idf* vectors (Sivic and Zisserman 2009) or by using inverted indices to examine only a subset of the images (Nistér and Stewénius 2006).

   (c) Optionally re-rank or verify all the candidate matches, using either spatial consistency (Sivic and Zisserman 2009) or an affine (or simpler) transformation model (Philbin, Chum, Isard *et al.* 2007).

   (d) Optionally expand the answer set by re-submitting highly ranked matches as new queries (Chum, Philbin, Sivic *et al.* 2007).

Outline

Shapes: Snakes

Instance recognition

Large databases

Object detection

Face detection: Viola and Jones

Object detection

- We are given an image to analyze
  - Goal: construct special-purpose *detectors*, whose job is to rapidly find likely regions where particular objects might occur.

Object detection

- We are given an image to analyze
  - Goal: construct special-purpose *detectors*, whose job it is to rapidly find likely regions where particular objects might occur.
  - Face detection
    - Viola and Jones (later)
  - Pedestrian detection

Pedestrian detection

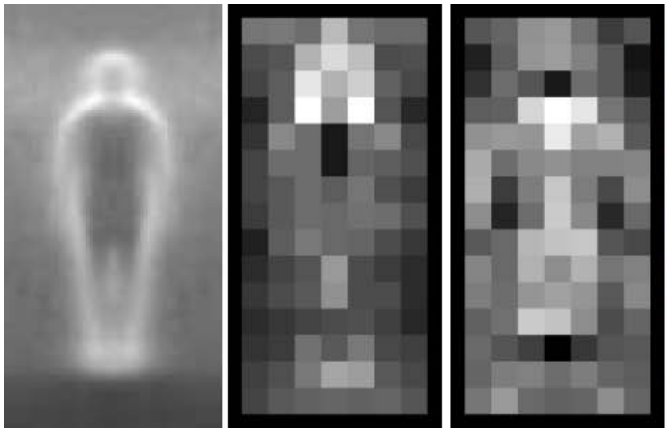
- Dalal and Triggs (2005)
  - Overlapping histogram of oriented gradients (HOG) Face detection
  - A support vector machine (SVM) is trained on the resulting high-dimensional continuous descriptor vectors

Pedestrian detection

- Dalal and Triggs (2005)
  - Overlapping histogram of oriented gradients (HOG) Face detection
  - A support vector machine (SVM) is trained on the resulting high-dimensional continuous descriptor vectors

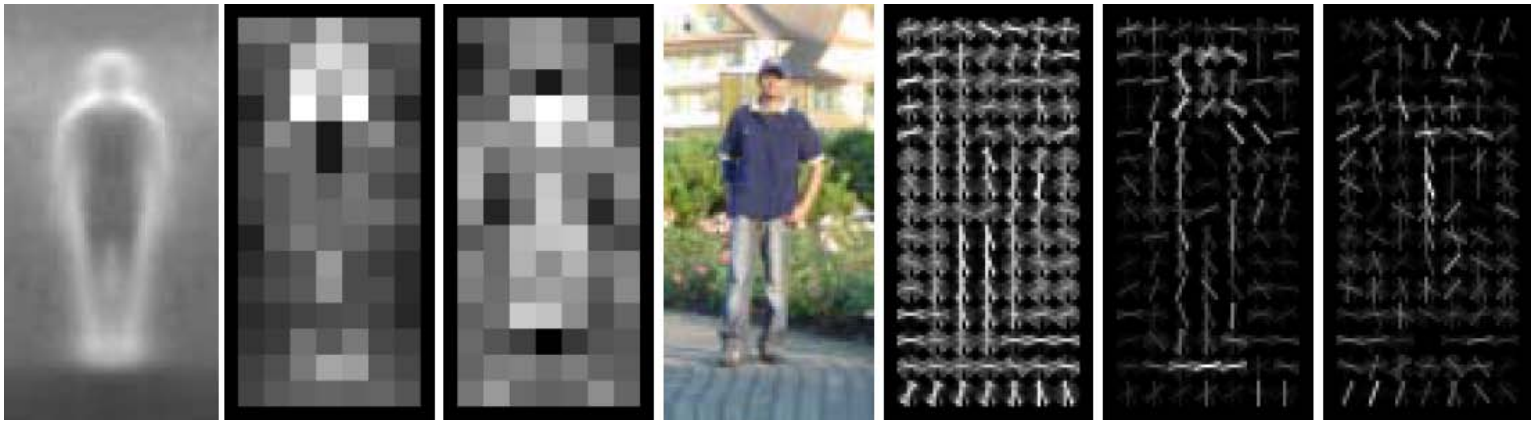
Pedestrian detection

- Dalal and Triggs (2005)
  - Overlapping histogram of oriented gradients (HOG) Face detection
  - A support vector machine (SVM) is trained on the resulting high-dimensional continuous descriptor vectors



Pedestrian detection

- Dalal and Triggs (2005)
  - Overlapping histogram of oriented gradients (HOG) Face detection
  - A support vector machine (SVM) is trained on the resulting high-dimensional continuous descriptor vectors

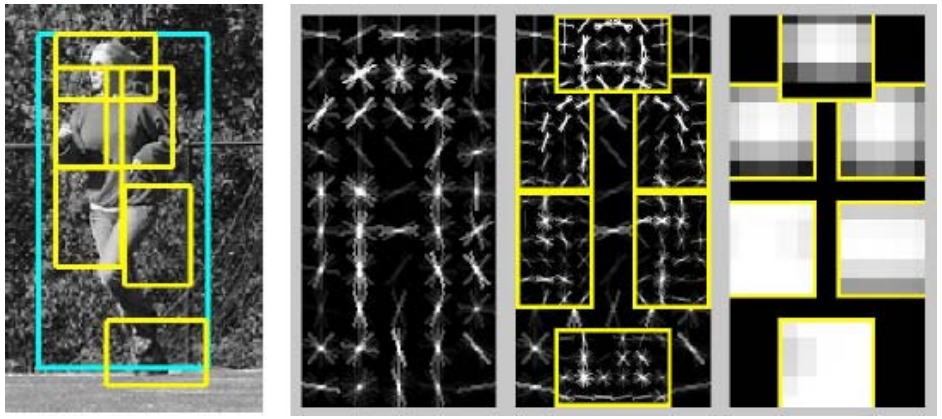


Pedestrian detection

- Felzenszwalb, McAllester, and Ramanan 2008)
  - Flexlible parts
  - Each part is trained and detected on HOGs evaluated at two pyramid levels below the overall object model and the locations of the parts relative to the parent node

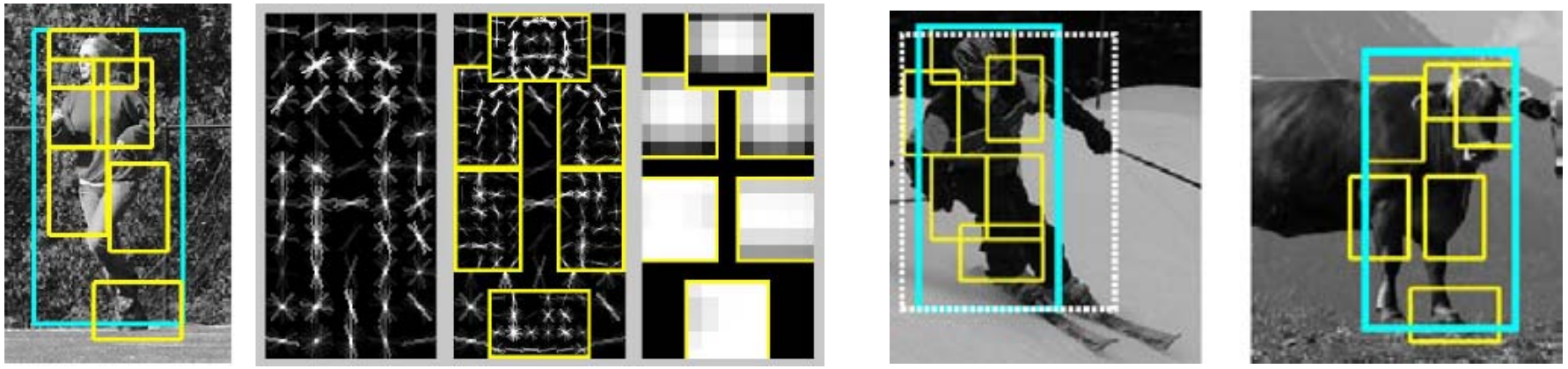
Pedestrian detection

- Felzenszwalb, McAllester, and Ramanan 2008)
  - Flexlible parts
  - Each part is trained and detected on HOGs evaluated at two pyramid levels below the overall object model and the locations of the parts relative to the parent node



Pedestrian detection

- Felzenszwalb, McAllester, and Ramanan 2008)
  - Flexlible parts
  - Each part is trained and detected on HOGs evaluated at two pyramid levels below the overall object model and the locations of the parts relative to the parent node



Pedestrian detection

- When video sequences are available, the additional information present in the optic flow and motion discontinuities can greatly aid in the detection task

Outline

Shapes: Snakes

Instance recognition

Large databases

Object detection

Face detection: Viola and Jones

Objective

- Face detection

Contributions

- Integral Image
- Adaboost classification
- Cascade of classifiers

Integral image

- Similar to Haar basis functions (Papageorgiou et al. 1998)

Integral image

- Similar to Haar basis functions (Papageorgiou et al. 1998)

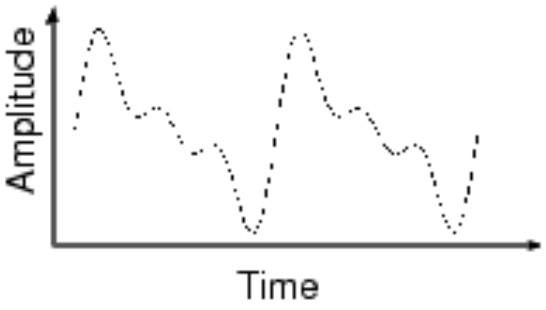
Wavelets

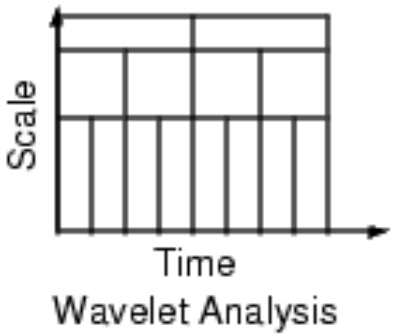
- A wavelet is a windowing technique with variable-size regions

Integral image

- Similar to Haar basis functions (Papageorgiou et al. 1998)

Wavelets



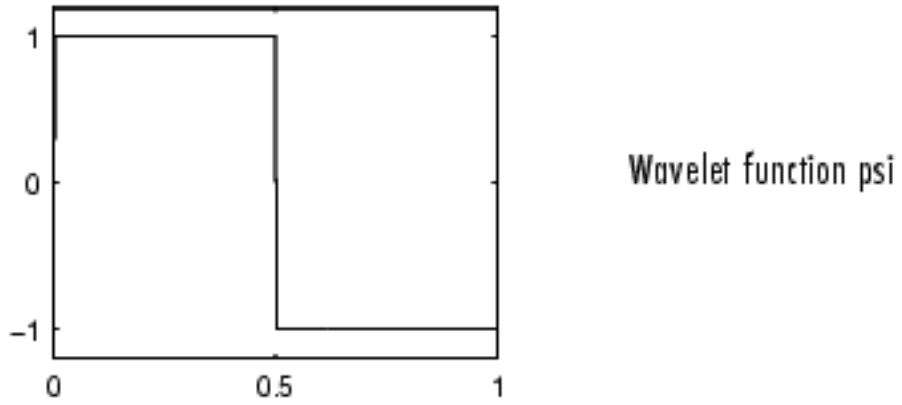


Integral image

- Similar to Haar basis functions (Papageorgiou et al. 1998)

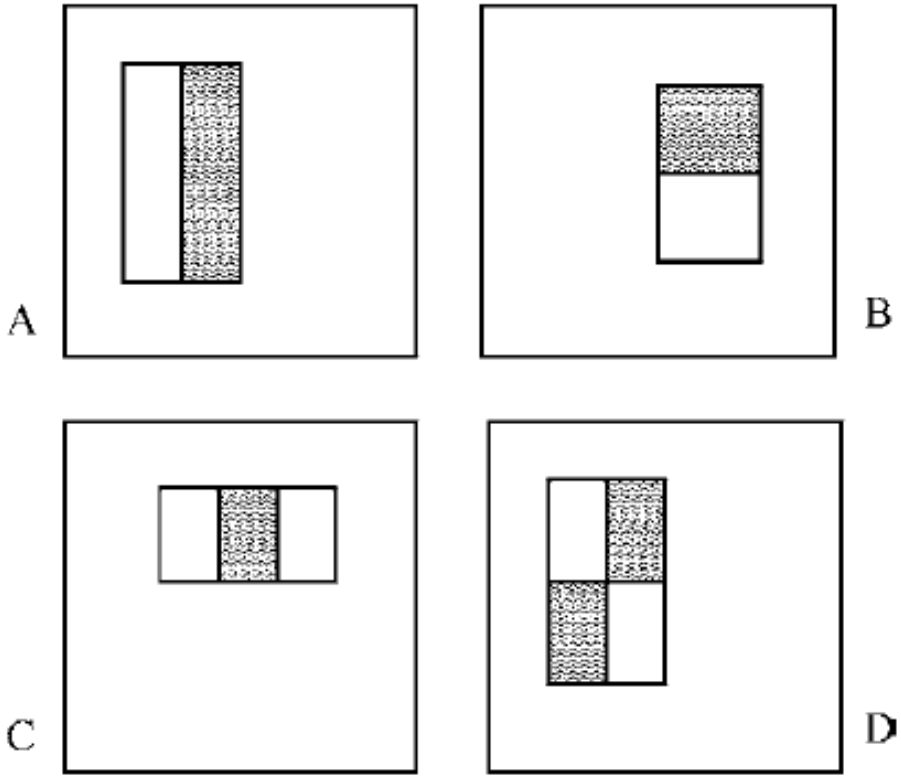
Haar wavelet

- Is discontinuous, and resembles a step function.



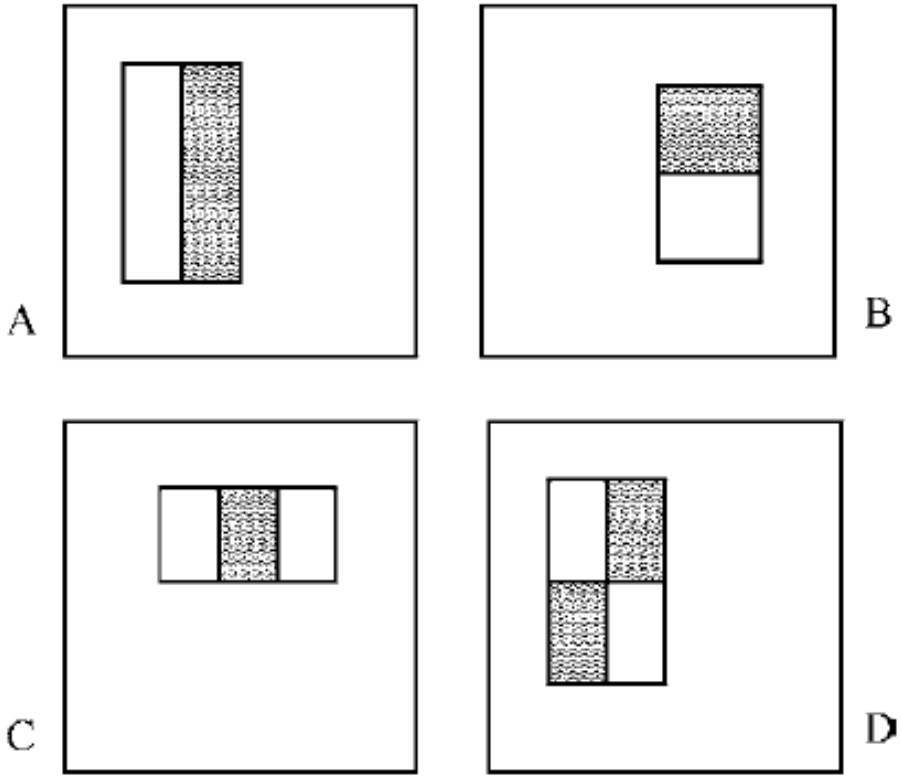
Integral image

- Similar to Haar basis functions (Papageorgiou et al. 1998)



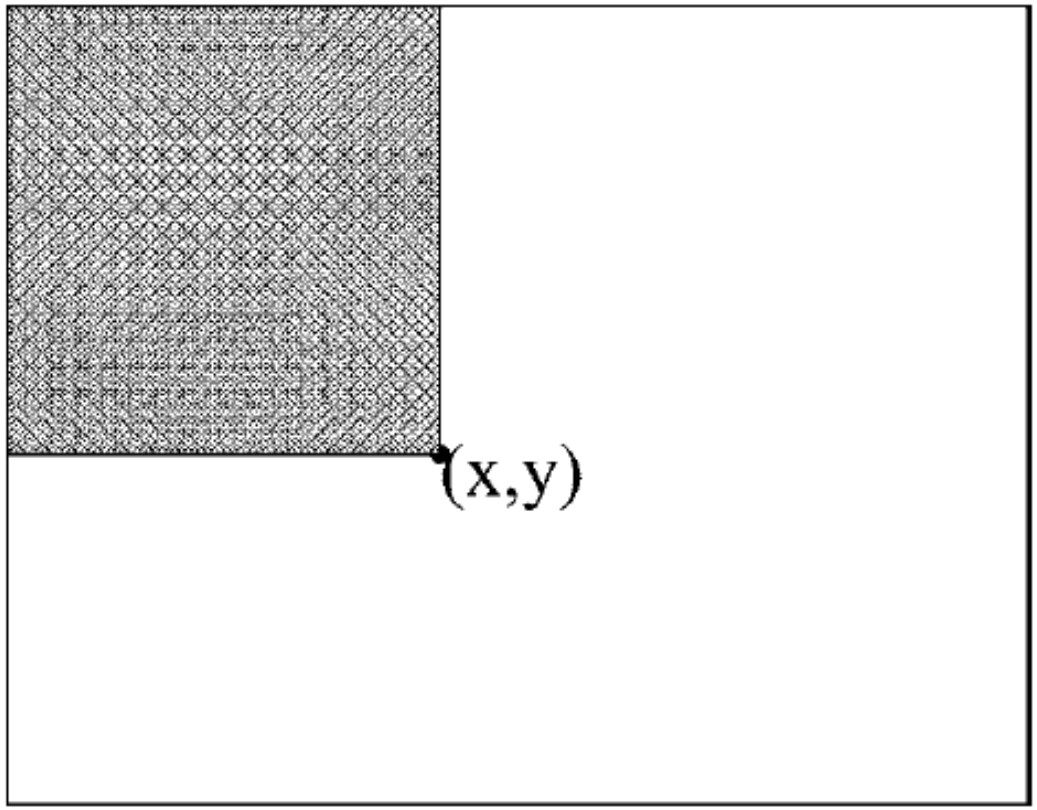
Integral image

- Similar to Haar basis functions (Papageorgiou et al. 1998)



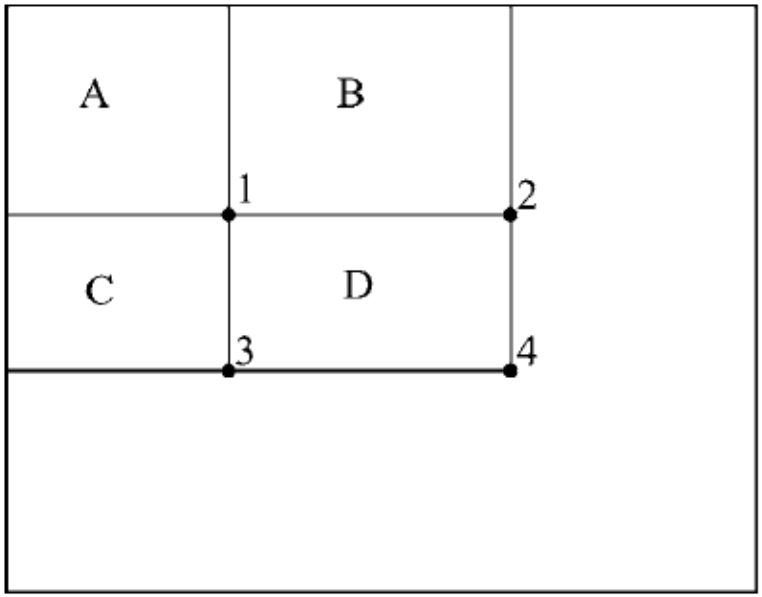
Integral image

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y'),$$



Integral image

- Convolution can be accelerated



The sum of the pixels within rectangle D can be computed with four array references. The value of the integral image at location 1 is the sum of the pixels in rectangle A. The value at location 2 is A + B, at location 3 is A + C, and at location 4 is A + B + C + D. The sum within D can be computed as 4 + 1 − (2 + 3).

Learning

- Around 45,000 rectangle features within each subwindow
- Hypothesis: Use a small number of features to form an effective classifier
- Variant of Adaboost

Learning: Adaboost

- Boosting
  - Goal: Improve the accuracy of any given learning algorithm
  - Train successive classifiers with a subset of the training data that is most informative given the current set of component classifiers

Learning: Adaboost

- Boosting
  - Consider creating three component classifiers for a two-category problem

Learning: Adaboost

- Boosting
  - Consider creating three component classifiers for a two-category problem
  1. Select a set $n_1 < n$ patterns from the full training set D, call this set D_1

Learning: Adaboost

- Boosting
  - Consider creating three component classifiers for a two-category problem
  1. Select a set $n_1 < n$ patterns from the full training set D, call this set D_1
  2. Train first classifier C_1
     - Only need to be weak learner (better than chance)

Learning: Adaboost

- Boosting
  - Consider creating three component classifiers for a two-category problem
  1. Select a set $n_1 < n$ patterns from the full training set D, call this set D_1
  2. Train first classifier C_1
  3. Seek a second training set D_2
     - "Most informative" given C_1
     - Half the patterns should be classified correctly, half incorrectly
       - Select samples from D, present it to C_1 until wrong
       - Add those to D_2 along with 50% correctly classified by C_1

Learning: Adaboost

- Boosting
  - Consider creating three component classifiers for a two-category problem
  1. Select a set $n_1 < n$ patterns from the full training set D, call this set D_1
  2. Train first classifier C_1
  3. Seek a second training set D_2
  4. Train second classifier C_2

Learning: Adaboost

- Boosting
  - Consider creating three component classifiers for a two-category problem
  1. Select a set $n_1 < n$ patterns from the full training set D, call this set D_1
  2. Train first classifier C_1
  3. Seek a second training set D_2
  4. Train second classifier C_2
  5. Seek a third training set D_3 which is not well classified by C_1 and C_2

Learning: Adaboost

- Boosting
  - Consider creating three component classifiers for a two-category problem

1 – 3. Select D_1, train C_1, select D_1

4. Train second classifier C_2
5. Seek a third training set D_3 which is not well classified by C_1 and C_2
   - If C_1 and C_2 disagree, add the pattern to the dataset, otherwise ignore it

Learning: Adaboost

- Boosting
  - Consider creating three component classifiers for a two-category problem

1 – 3. Select D_1, train C_1, select D_1

4. Train second classifier C_2 with D_2
5. Seek a third training set D_3 which is not well classified by C_1 and C_2
6. Train third classifier C_3 with D_3

Learning: Adaboost

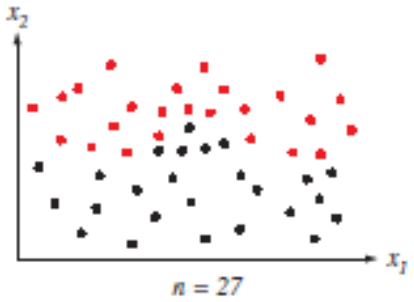
- Boosting
  - Consider creating three component classifiers for a two-category problem
  1. Select a set $n_1 < n$ patterns from the full training set D, call this set D_1
  2. Train first classifier C_1
  3. Seek a second training set D_2
  4. Train second classifier C_2
  5. Seek a third training set D_3 which is not well classified by C_1 and C_2
  6. Train third classifier C_3 with D_3

Learning: Adaboost

- Boosting
  - Consider creating three component classifiers for a two-category problem
  - Classification
    - Based on the votes of the component classifiers
    - e.g. If C_1 and C_2 agree on the category label x, we use that label, if they disagree, we use C_3

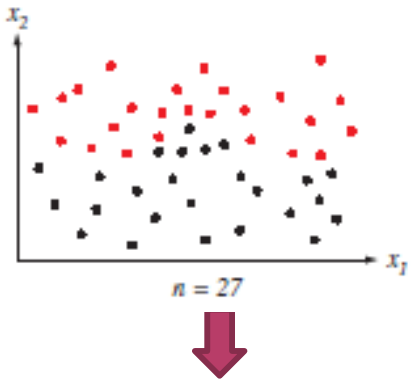
Learning: Adaboost

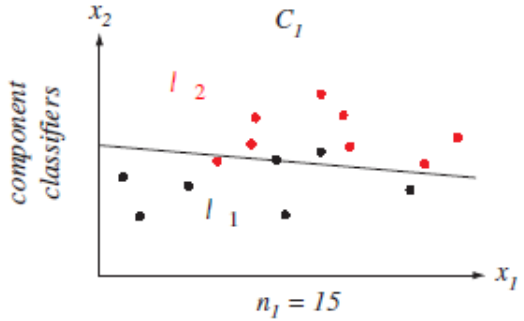
- Boosting
  - Consider creating three component classifiers for a two-category problem
  - Example



Learning: Adaboost

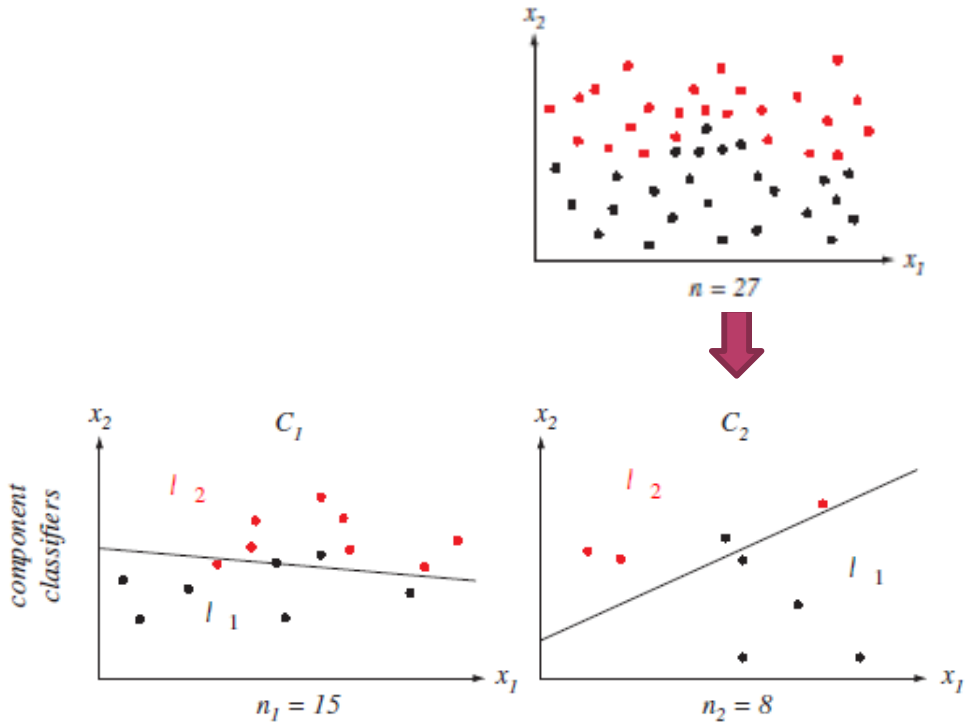
- Boosting
  - Consider creating three component classifiers for a two-category problem
  - Example





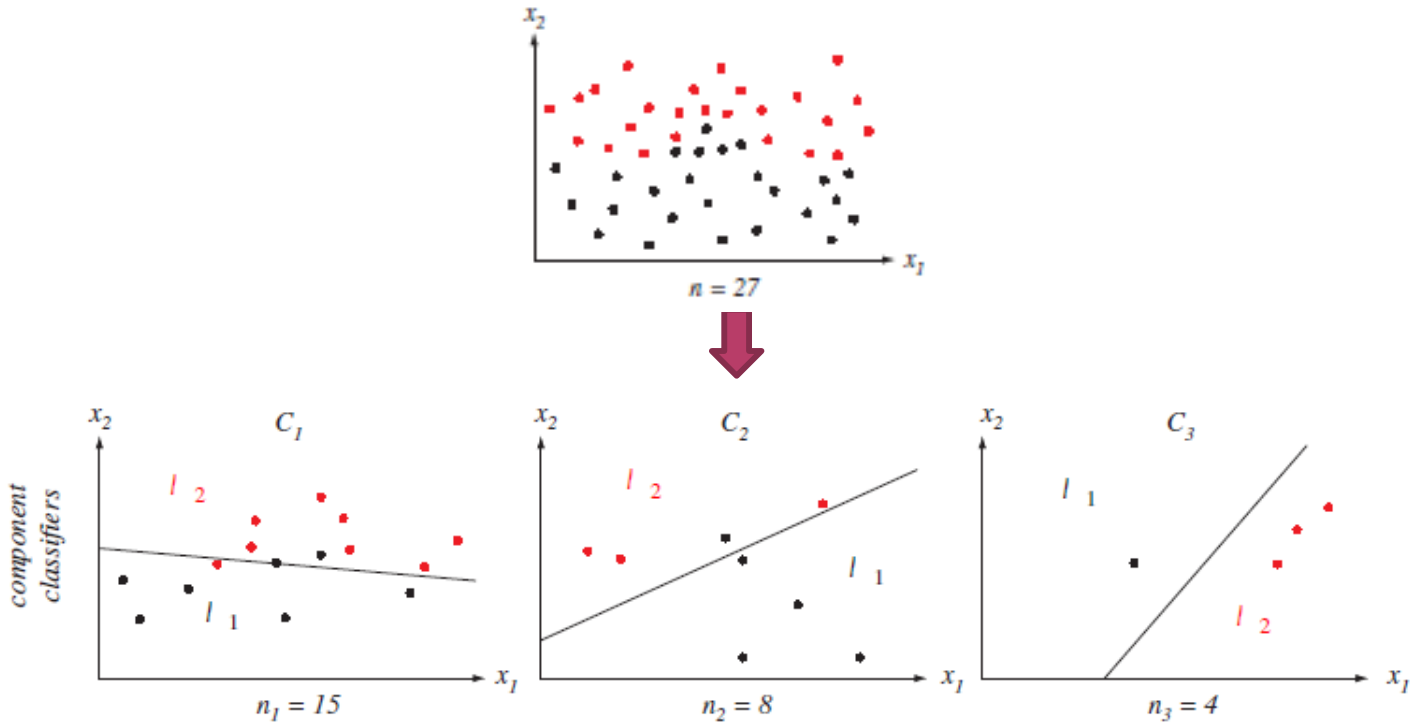
Learning: Adaboost

- Boosting
  - Consider creating three component classifiers for a two-category problem
  - Example



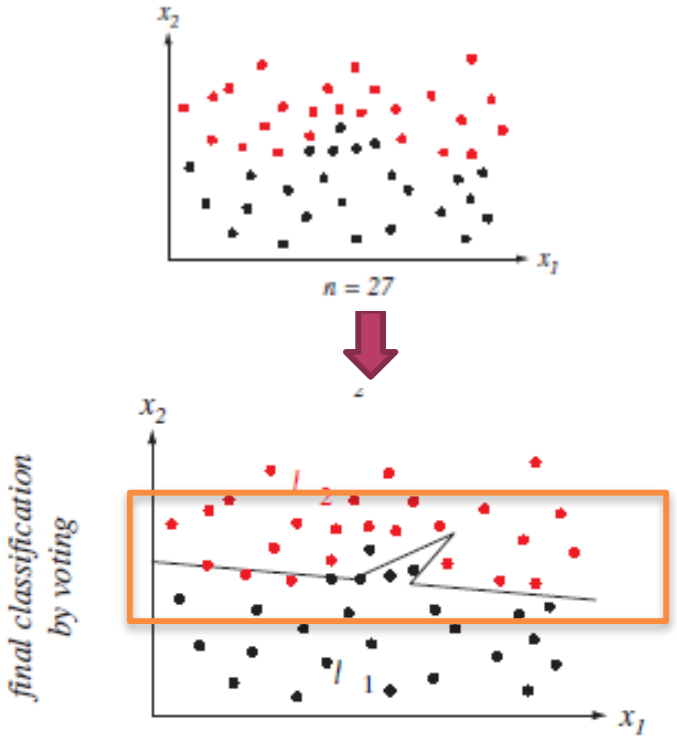
Learning: Adaboost

- Boosting
  - Consider creating three component classifiers for a two-category problem
  - Example



Learning: Adaboost

- Boosting
  - Consider creating three component classifiers for a two-category problem
  - Classification by voting



Learning: Adaboost

- Boosting
  - Considerations
    - Selecting n_1 is critical
      - If n_1 is too simple, C_1 will solve the classification problem
      - And $n_1 \approx n$
      - If n_2 is too complicated, C_1 will be of little use
      - And $n_2 \approx n$ or $n_3 \approx n$
    - Usually try to get size $n_1 \approx n_2 \approx n_3 \approx n/3$
      - Run boosting several times

Learning: Adaboost

- Variation of boosting ("Adaptive" boosting)
  - Allows the designer to add weak learners until some training error
  - Each training pattern receives a weight that determines the probability of being selected from a classifier
    - If a training pattern is accurately classified, its chance of being used again by other classifier is reduced
    - If it is not accurately classified, its chance is raised
  - Adaboost focuses on informative or difficult patterns

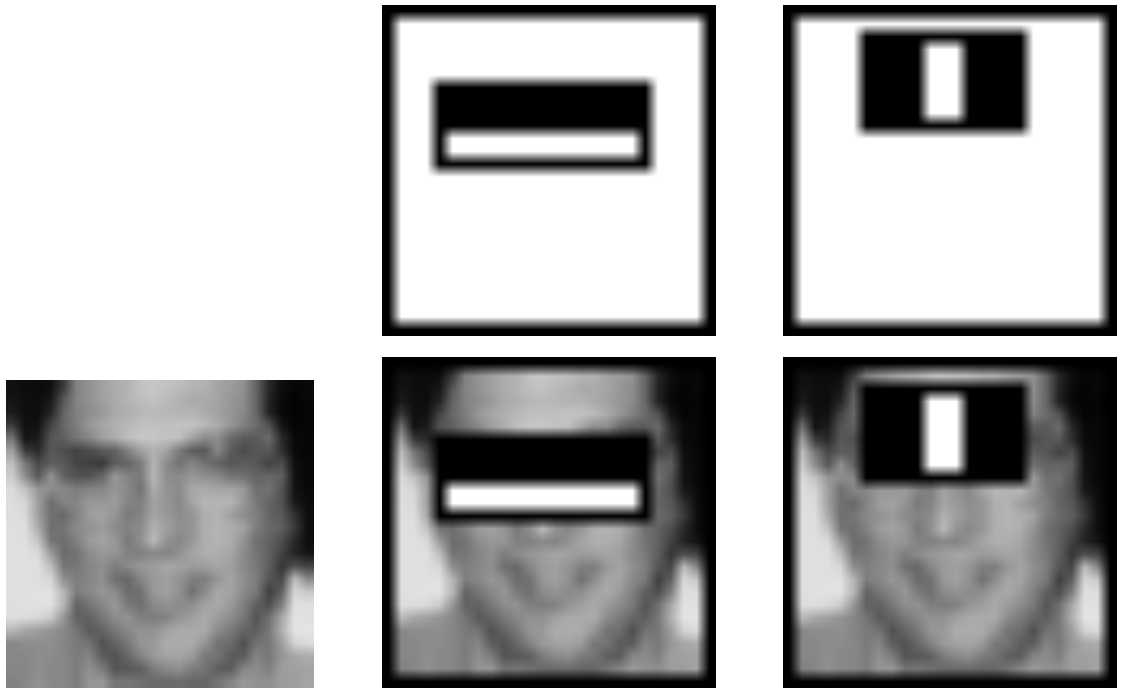
Learning: Adaboost

- Remember: 45,000 rectangle features within each subwindow
- Viola and Jones use Adaboost to select a number of “good features”
- The weak algorithm is designed to select the single rectangle feature which best separates the positive and negative examples
- Add a weak classifier such that when added we achieve lowest error

Learning: Adaboost

- Remember: 45,000 rectangle features within each subwindow
- Viola and Jones use Adaboost to select a number of “good features”
- The weak algorithm is designed to select the single rectangle feature which best separates the positive and negative examples
- Add a weak classifier such that when added we achieve lowest error
- Reduction to 200 features ! (95% detection rate)

Learning: Adaboost



Attentional cascade

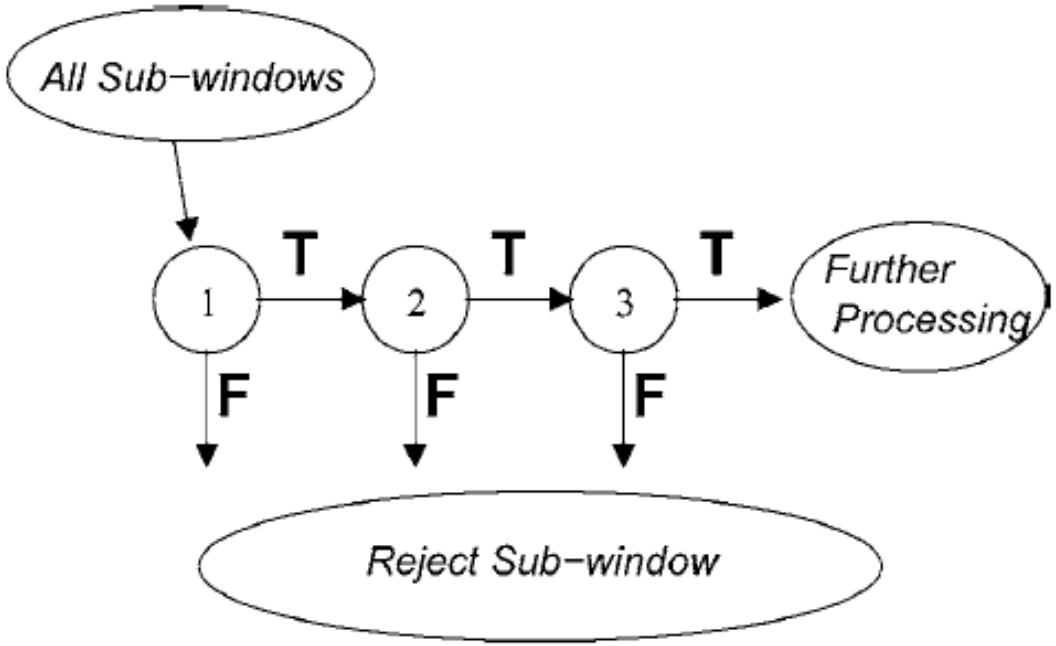
- Goal: Construct a cascade of classifiers to reduce computing time

Attentional cascade

- Goal: Construct a cascade of classifiers to reduce computing time
- Simpler classifiers are used to reject sub-windows
  - A positive result from the first classifier triggers the second classifier
  - A positive result from the second triggers the third
  - And so on

Attentional cascade

- Goal: Construct a cascade of classifier to reduce computing time
- Simpler classifiers are used to reject sub-windows

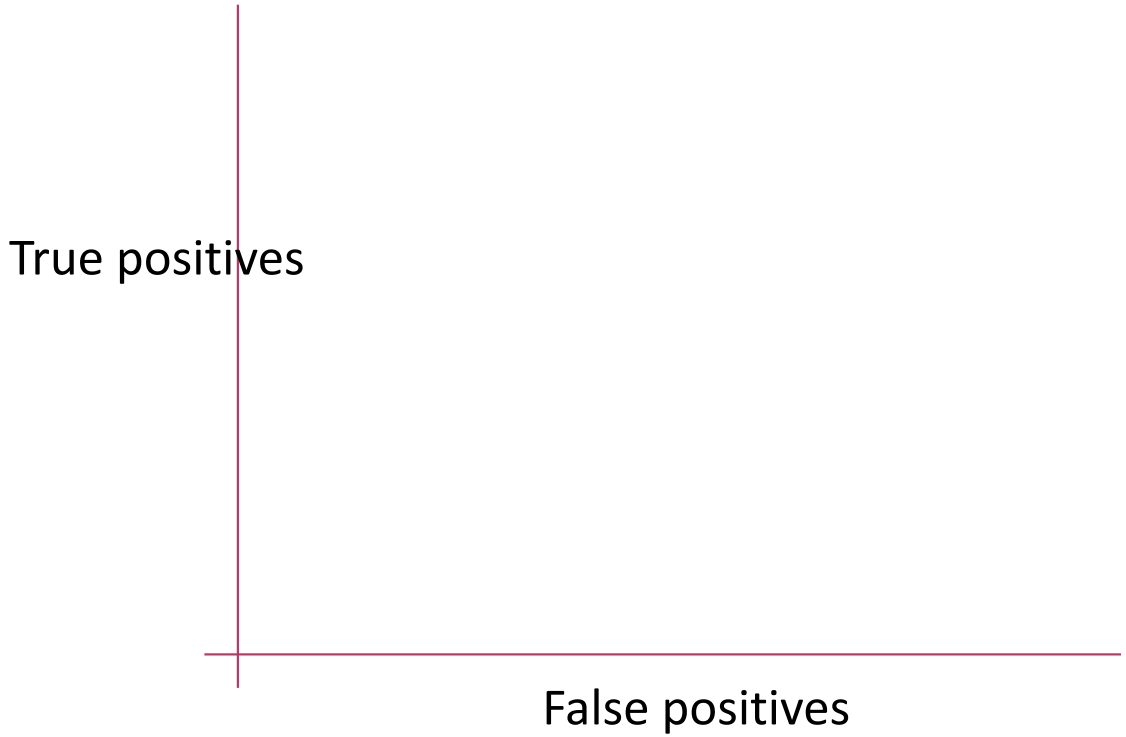


Attentional cascade

- Goal: Construct a cascade of classifier to reduce computing time
- Simpler classifiers are used to reject sub-windows
  - So, we are selecting windows that are highly informative of containing a face
- Attentional cascade is 10 times faster than using the 200 features at once

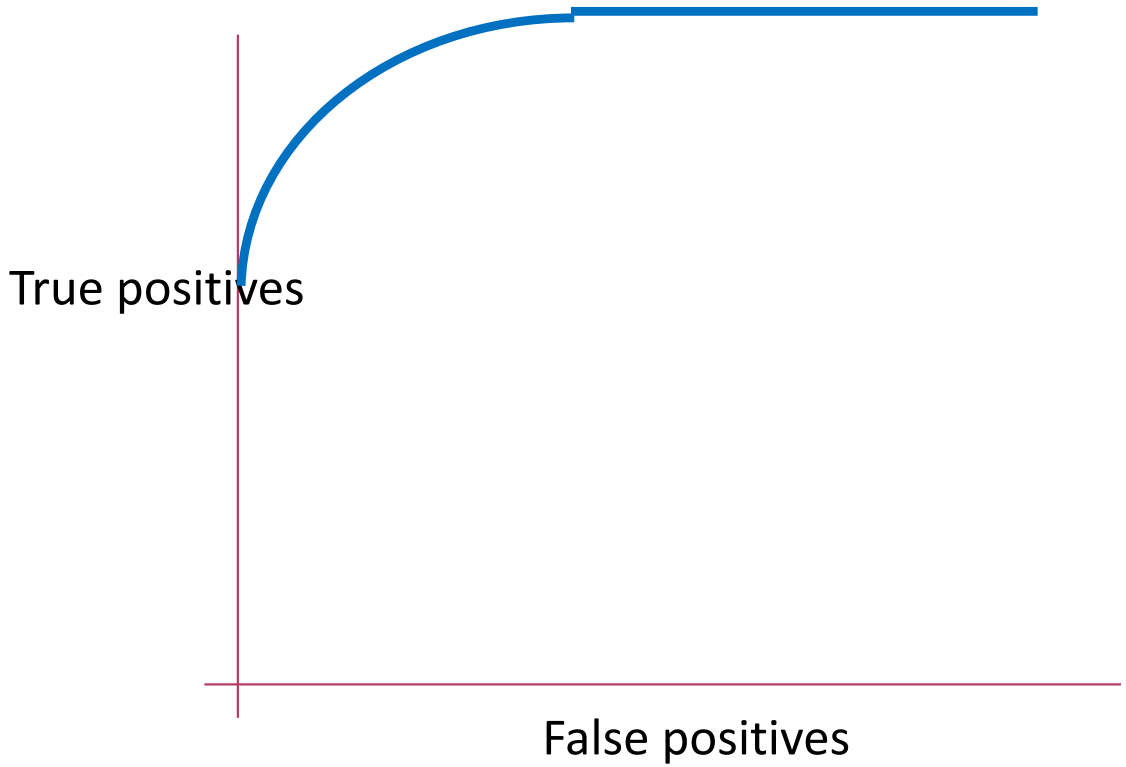
Note: ROC curves

- Receiver Operator Characteristic

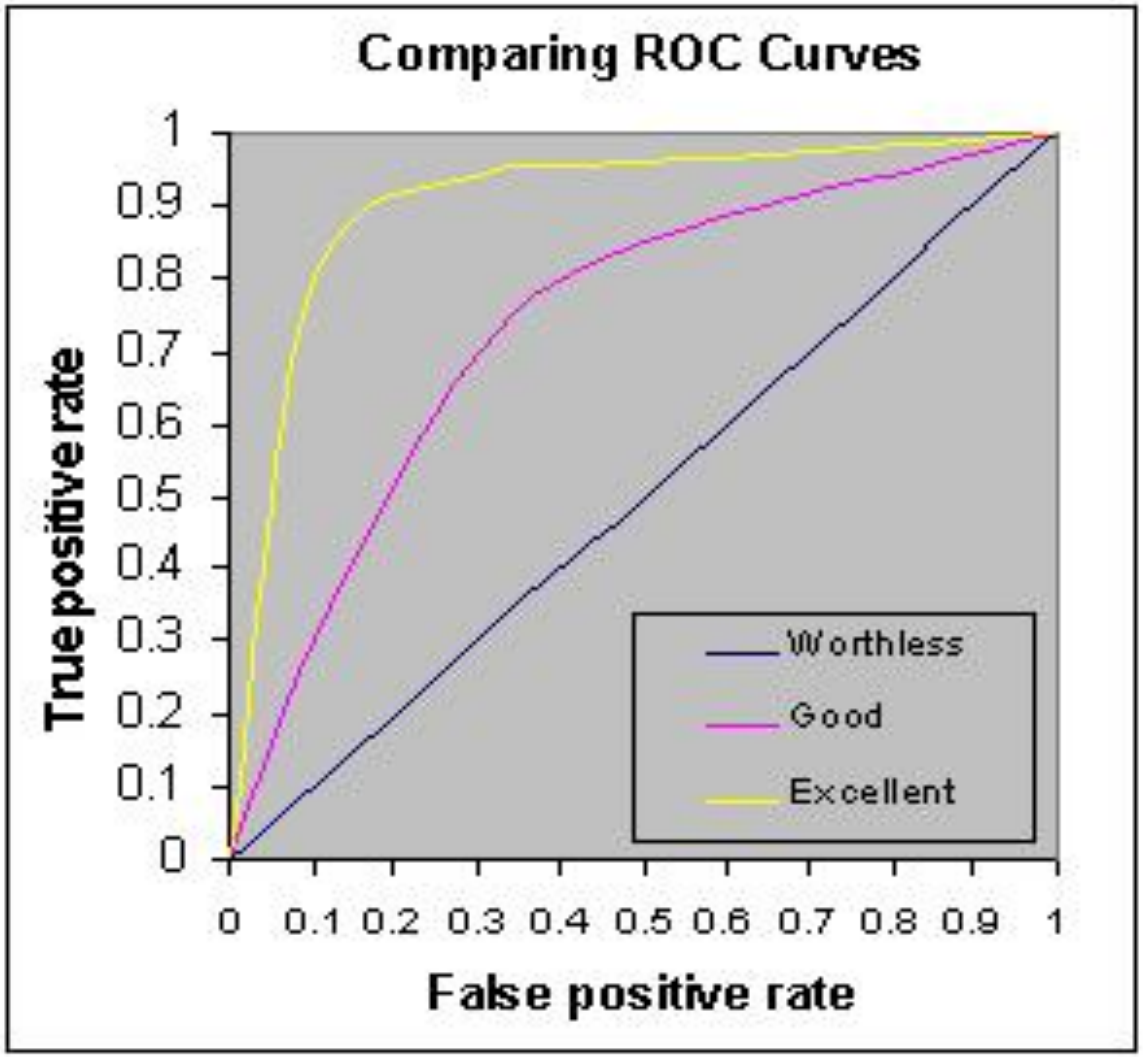


Note: ROC curves

- Receiver Operator Characteristic



Note: ROC curves



Results

- Training set 1
  - 4916 labeled faces
  - 9500 non faces
- Detector: 38 layer cascade with 6060 features
  - First classifier
    - 2 features. 50% non-faces correctly detected
  - Second classifier
    - 10 features. 80% non-faces correctly detected
  - Third and four classifier
    - 25 features.
  - ….

Results

- Training set 2
  - 130 images with 507 faces

Results

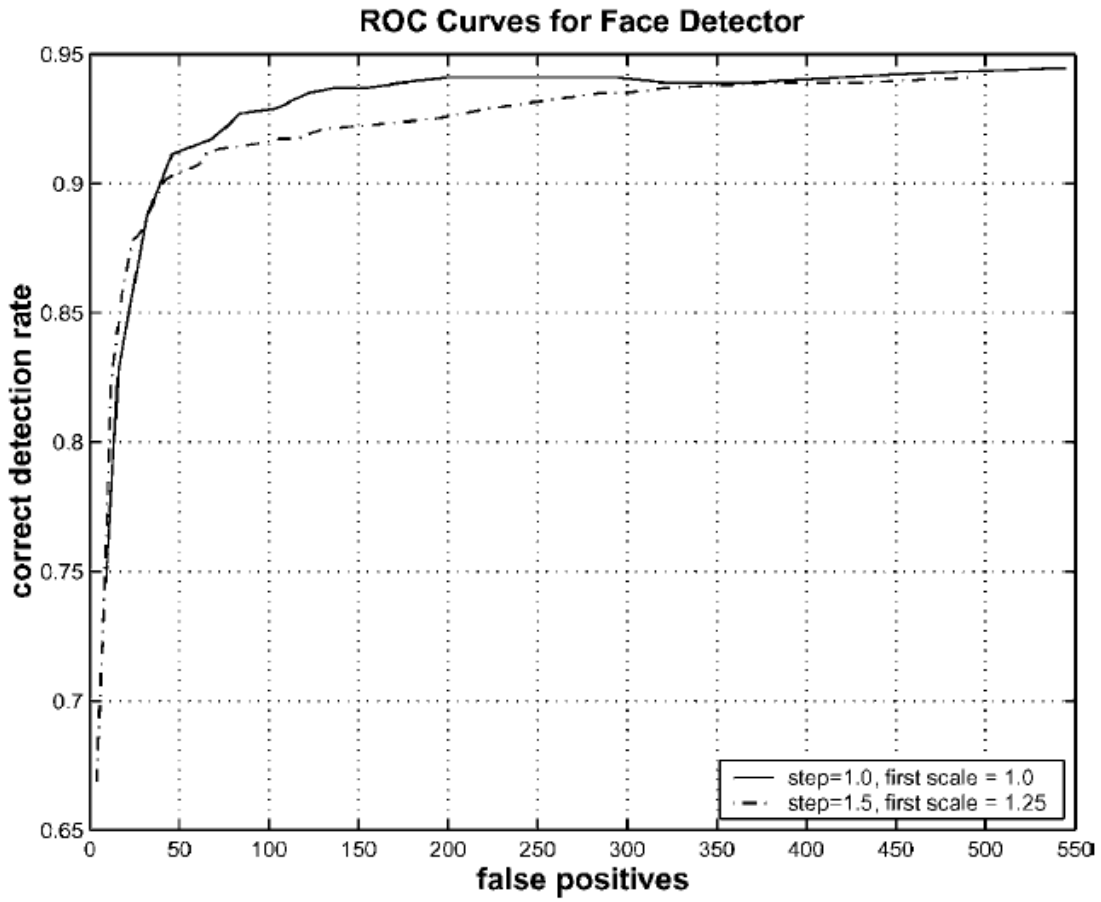
- Training set 2
  - 130 images with 507 faces

Results

- Training set 2
  - 130 images with 507 faces

Results

- Training set 2
  - 130 images with 507 faces



Conclusions

- 15 times faster than face detection up to date of publication
- High detection rate
- Adaboost for feature selection
- Limitations
  - Rotation
  - Backlighting
  - Occlusions

Summary

Shapes: Snakes

Instance recognition

Large databases

Object detection

Face detection: Viola and Jones