Object Category Detection: Sliding Windows

Computer Vision

CS 543 / ECE 549

University of Illinois

Derek Hoiem

Administrative

- Remember to e-mail project descriptions (by Thurs) and talk to Ian or me (by Fri)

- Clarifications about the homework?

Today's class: Object Category Detection

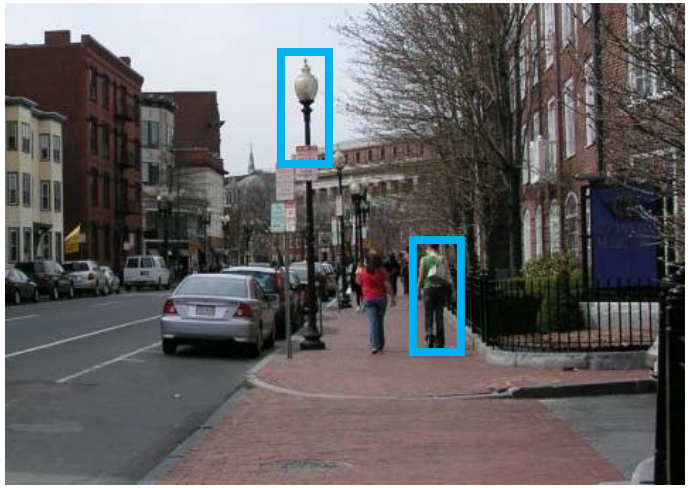
- Statistical template matching with sliding window detector
  - Schneiderman Kanade detector
  - Viola Jones detector
- Broader overview of object category detection

Object category detection in computer vision

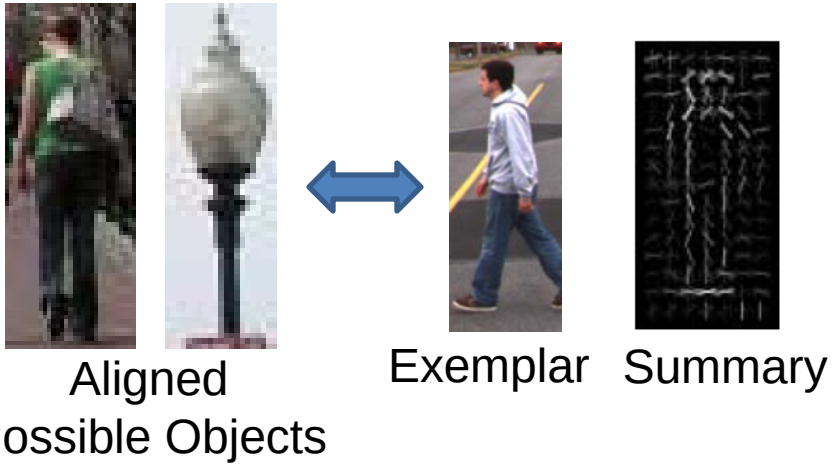
Goal: detect all pedestrians, cars, monkeys, etc in image

Basic Steps of Category Detection

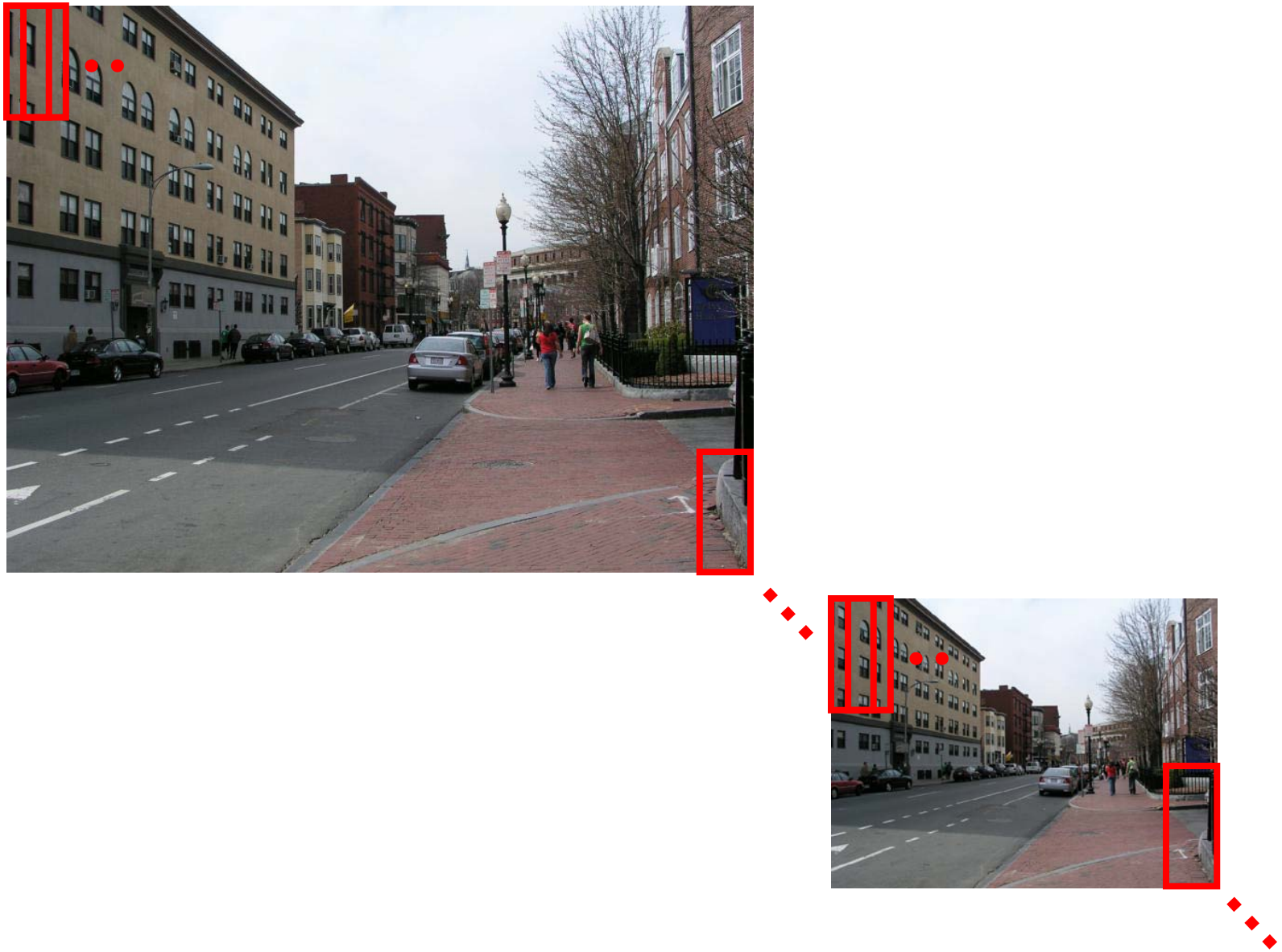
1. Align
   - E.g., choose position, scale orientation
   - How to make this tractable?



2. Compare
   - Compute similarity to an example object or to a summary representation
   - Which differences in appearance are important?



Sliding window: a simple alignment solution



Each window is separately classified

Statistical Template

- Object model = sum of scores of features at fixed positions

+3 +2 -2 -1 -2.5 = -0.5 ? > 7.5

Non-object

+4 +1 +0.5 +3 +0.5 = 10.5 ? > 7.5

Object

Design challenges

- How to efficiently search for likely objects
  - Even simple models require searching hundreds of thousands of positions and scales
- Feature design and scoring
  - How should appearance be modeled? What features correspond to the object?
- How to deal with different viewpoints?
  - Often train different models for a few different viewpoints
- Implementation details
  - Window size
  - Aspect ratio
  - Translation/scale step size
  - Non-maxima suppression

Schneiderman and Kanade

Schneiderman and Kanade. A Statistical Method for 3D Object Detection. (2000)

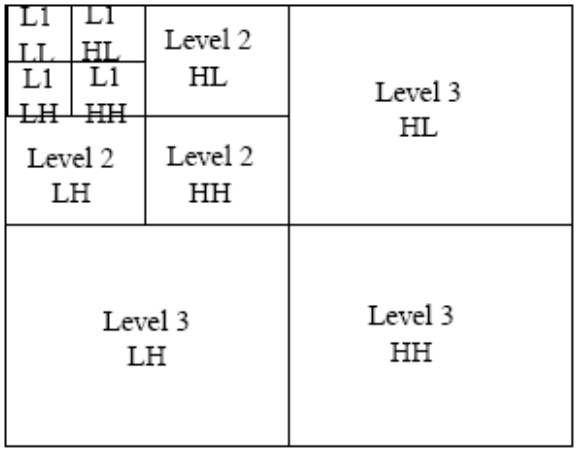
Schneiderman and Kanade

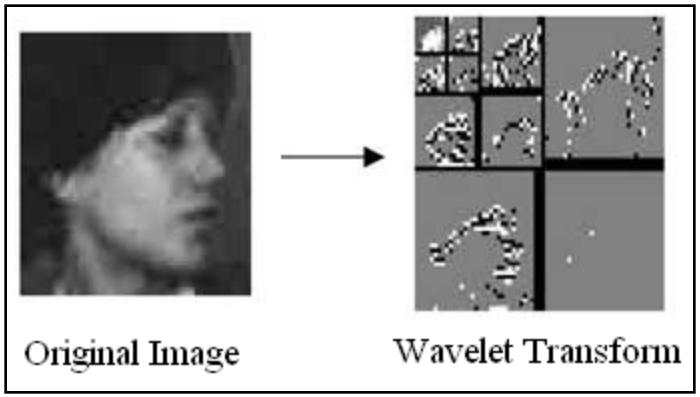
Decision function for statistical template matching:

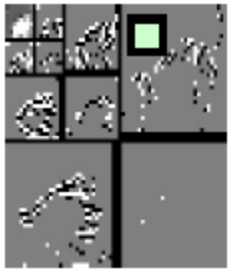
$$\frac{P(image|\text{object})}{P(image|\text{non-object})} > \lambda \qquad \left(\lambda = \frac{P(\text{non-object})}{P(\text{object})}\right)$$

Appearance model

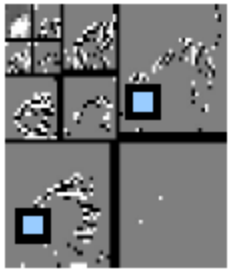
- Each feature is a group of quantized wavelet coefficients that are statistically dependent



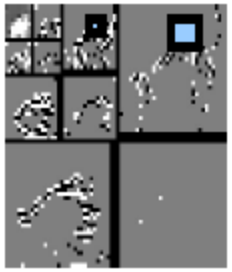




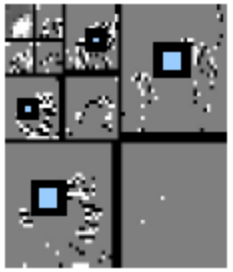
Intra-subband



Inter-orientation



Inter-frequency



Inter-frequency/
Inter-orientation

Learning to classify (feature likelihoods)

- Class-conditional likelihood ratio

$$\frac{\prod_{x,y \in \text{region}} \prod_{k=1}^{17} P_k(pattern_k(x,y), x, y | \text{object})}{\prod_{x,y \in \text{region}} \prod_{k=1}^{17} P_k(pattern_k(x,y), x, y | \text{non-object})} > \lambda$$

- Estimate P(pattern|object) and P(pattern|non-object) by counting over examples

$$P(pattern | object) = \frac{count(pattern \,\&\, object)}{count(object)}$$

- Tune weights discriminatively using Adaboost

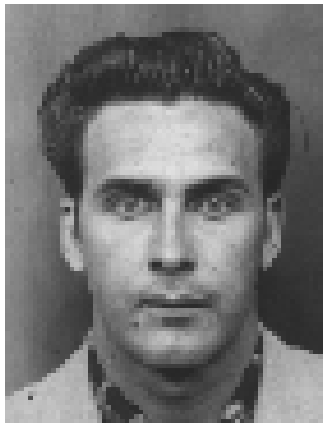
Training

1) Create training data
   a) Prepare each image: pre-process (optional), compute wavelet coefficients, discretize
   b) Extract positive windows and sample of negative windows
   c) Compute feature values for each example window

2) Learn scores for all possible feature values
   a) Compute ratios of histograms by counting for positive and negative examples
   b) Reweight examples using Adaboost

3) Get high-scoring negative examples (bootstrapping)

Training multiple viewpoints



Train new detector for each viewpoint.

Testing

1) Processing:
   a) Lighting correction (optional)
   b) Compute wavelet coefficients, quantize
2) Slide window over each position/scale (2 pixels, $2^{1/4}$ scale)
   a) Compute feature values
   b) Look up scores
   c) Sum scores over features
   d) Threshold
3) Use faster classifier to prune patches (cascade…more on this later)
4) Non-maximum suppression

Results: faces

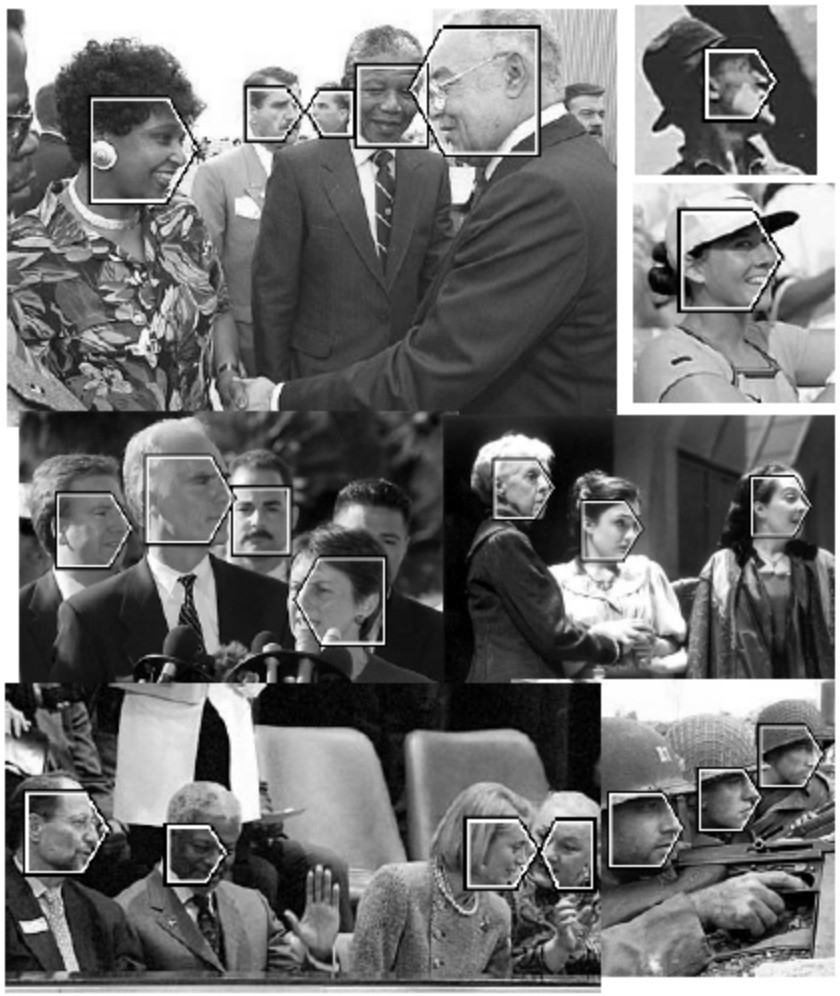


Table 1. Face detection with out-of-plane rotation

γ	Detection (all faces)	Detection (profiles)	False Detections
0.0	92.7%	92.8%	700
1.5	85.5%	86.4%	91
2.5	75.2%	78.6%	12

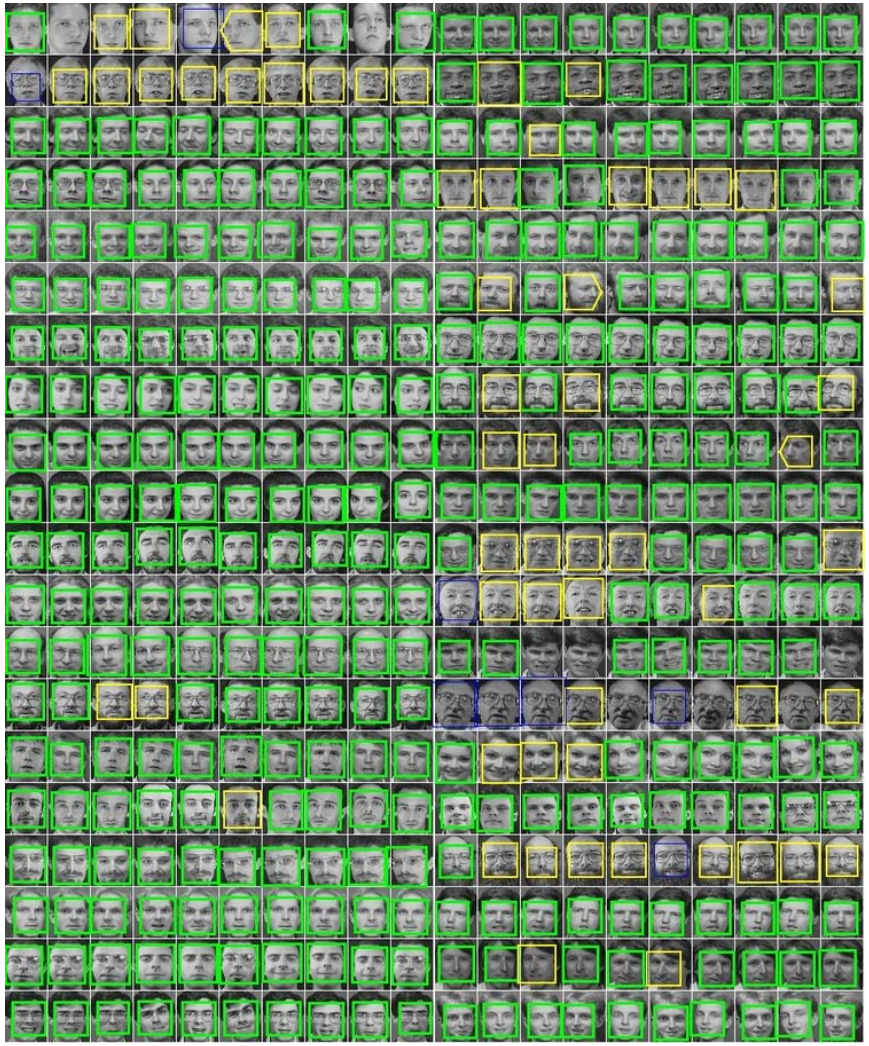
208 images with 441 faces, 347 in profile

Results: cars

Table 3. Car detection

γ	Detections	False Detections
1.05	83%	7
1.0	86%	10
0.9	92%	71

Results: faces today



http://demo.pittpatt.com/

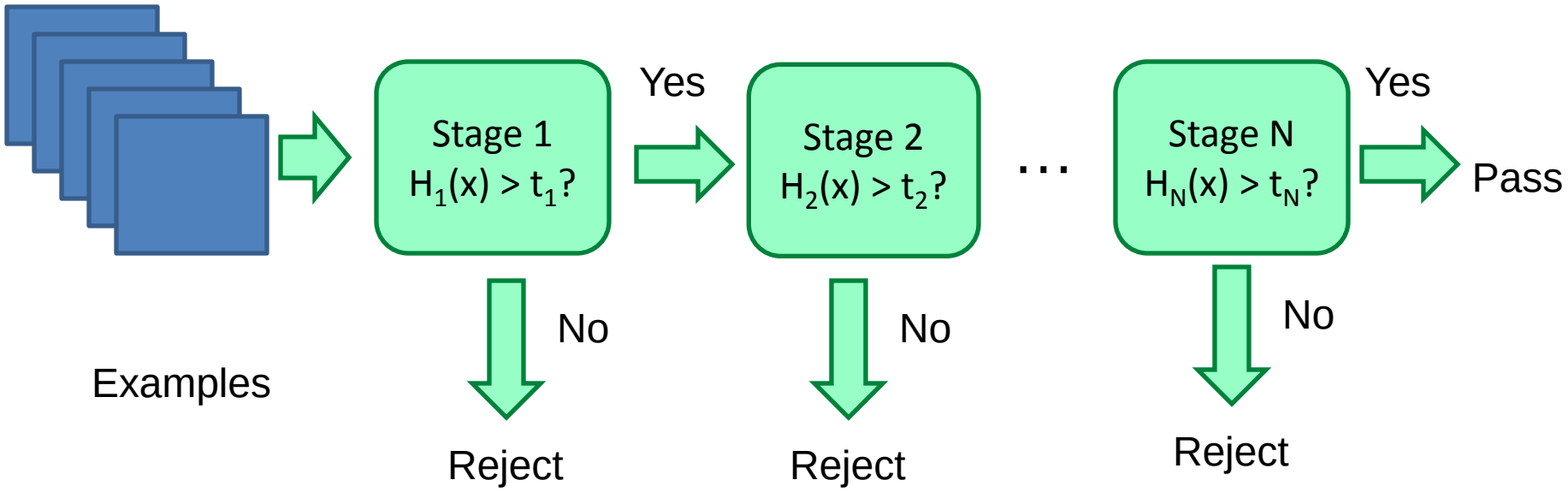
Viola-Jones sliding window detector

Fast detection through two mechanisms

- Quickly eliminate unlikely windows
- Use features that are fast to compute

Viola and Jones. Rapid Object Detection using a Boosted Cascade of Simple Features (2001).

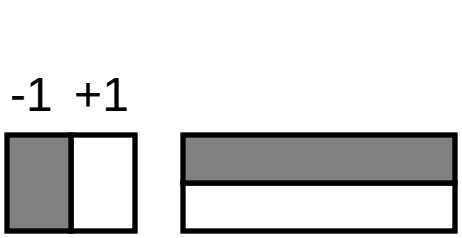
Cascade for Fast Detection



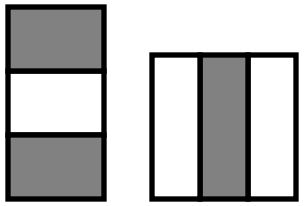
- Choose threshold for low false negative rate
- Fast classifiers early in cascade
- Slow classifiers later, but most examples don't get there

Features that are fast to compute

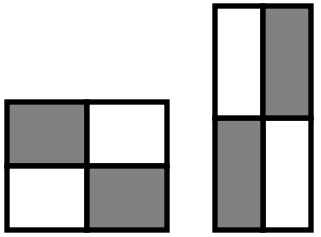
- "Haar-like features"
  - Differences of sums of intensity
  - Thousands, computed at various positions and scales within detection window



Two-rectangle features



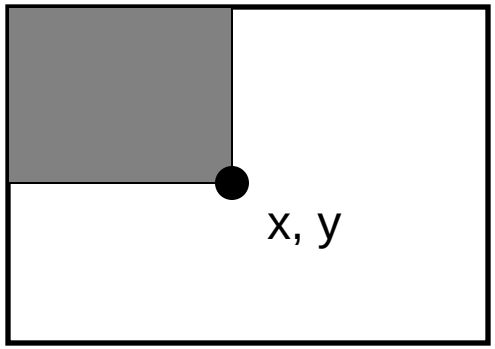
Three-rectangle features



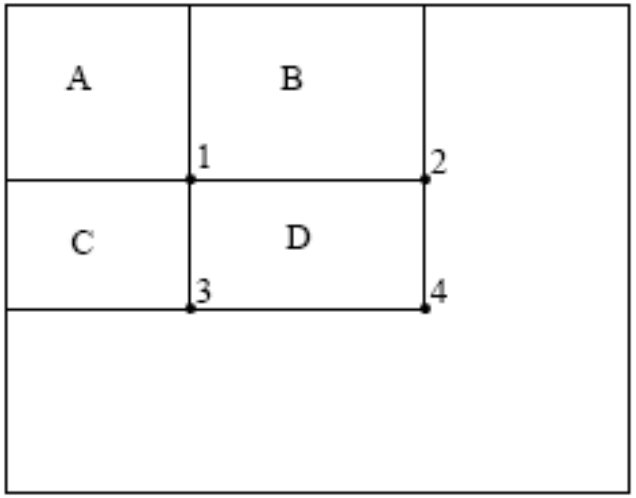
Etc.

Integral Images

- ii = cumsum(cumsum(Im, 1), 2)



ii(x,y) = Sum of the values in the grey region



How to compute B-A?

How to compute A+D-B-C?

Feature selection with Adaboost

- Create a large pool of features (180K)
- Select features that are discriminative and work well together
  - "Weak learner" = feature + threshold + parity

$$h_j(x) = \begin{cases} 1 & \text{if } p_j f_j(x) < p_j \theta_j \\ 0 & \text{otherwise} \end{cases}$$

  - Choose weak learner that minimizes error on the weighted training set
  - Reweight

Adaboost

- Given example images $(x_1, y_1), \ldots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For $t = 1, \ldots, T$:
  1. Normalize the weights,

     $$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$$

     so that w_t is a probability distribution.
  2. For each feature, j, train a classifier h_j which is restricted to using a single feature. The error is evaluated with respect to w_t, $\epsilon_j = \sum_i w_i \left| h_j(x_i) - y_i \right|$.
  3. Choose the classifier, h_t, with the lowest error ϵ_t.
  4. Update the weights:

     $$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$$

     where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
- The final strong classifier is:

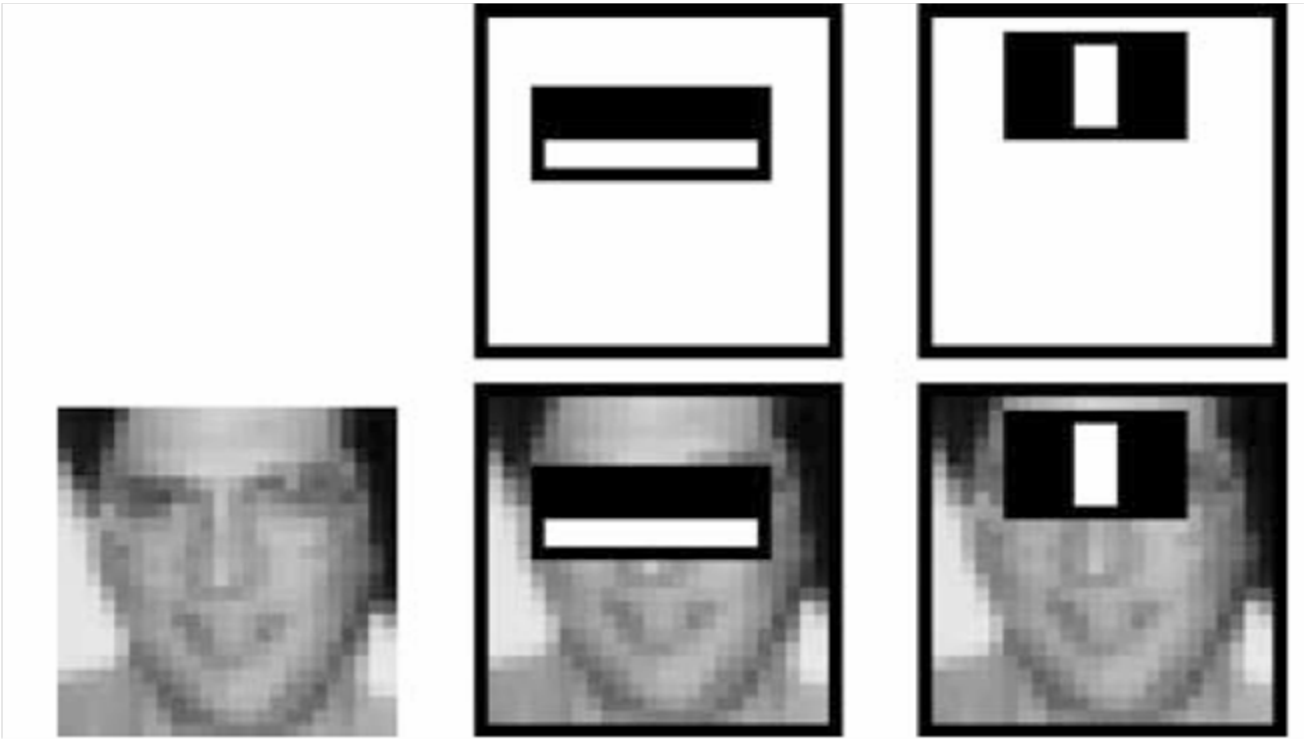
$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

where $\alpha_t = \log \frac{1}{\beta_t}$

Interpretations of Adaboost

- Additive logistic regression (Friedman et al. 2000)
  - LogitBoost from Collins et al. 2002 does this more explicitly
- Margin maximization (Schapire et al. 1998)
  - Ratch and Warmuth 2002 do this more explicitly

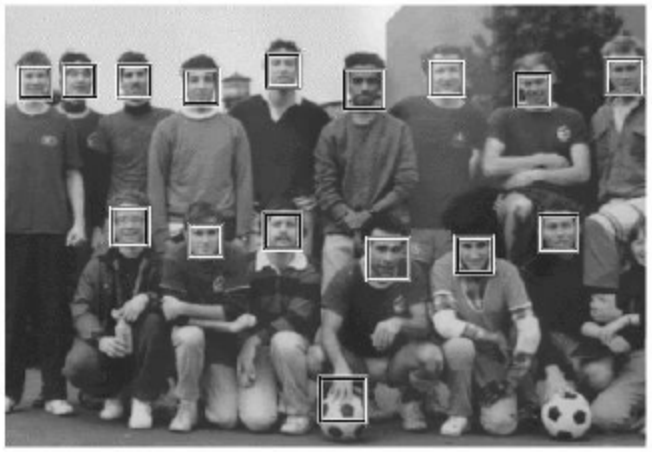
Top 2 selected features



Viola-Jones details

- 38 stages with 1, 10, 25, 50 … features
  - 6061 total used out of 180K candidates
  - 10 features evaluated on average
- Examples
  - 4916 positive examples
  - 10000 negative examples collected after each stage
- Scanning
  - Scale detector rather than image
  - Scale steps = 1.25, Translation 1.0*s to 1.5*s
- Non-max suppression: average coordinates of overlapping boxes
- Train 3 classifiers and take vote

Viola Jones Results


Detector \ False detections	10	31	50	65	78	95	167
Viola-Jones	76.1%	88.4%	91.4%	92.0%	92.1%	92.9%	93.9%
Viola-Jones (voting)	81.1%	89.7%	92.1%	93.1%	93.1%	93.2 %	93.7%
Rowley-Baluja-Kanade	83.2%	86.0%	-	-	-	89.2%	90.1%
Schneiderman-Kanade	-	-	-	94.4%	-	-	-
Roth-Yang-Ahuja	-	-	-	-	(94.8%)	-	-

MIT + CMU face dataset

Schneiderman later results

Schneiderman 2004

Viola-Jones 2001

Roth et al. 1999

Schneiderman-Kanade 2000

	89.7%	93.1%	94.4%	94.8%	95.7%
Bayesian Network *	1	8	19	36	56
Semi-Naïve Bayes*	6	19	29	35	46
[6]	31	65	--	--	--
[7]*	--	--	--	78	--
[16]*	--	--	65	--	--

Table 2. False alarms as a function of recognition rate on the MIT-CMU Test Set for Frontal Face Detection. * indicates exclusion of the 5 images of hand-drawn faces.

Speed: frontal face detector

- Schneiderman-Kanade (2000): 5 seconds

- Viola-Jones (2001): 15 fps

Strengths and Weaknesses of Statistical Template Approach

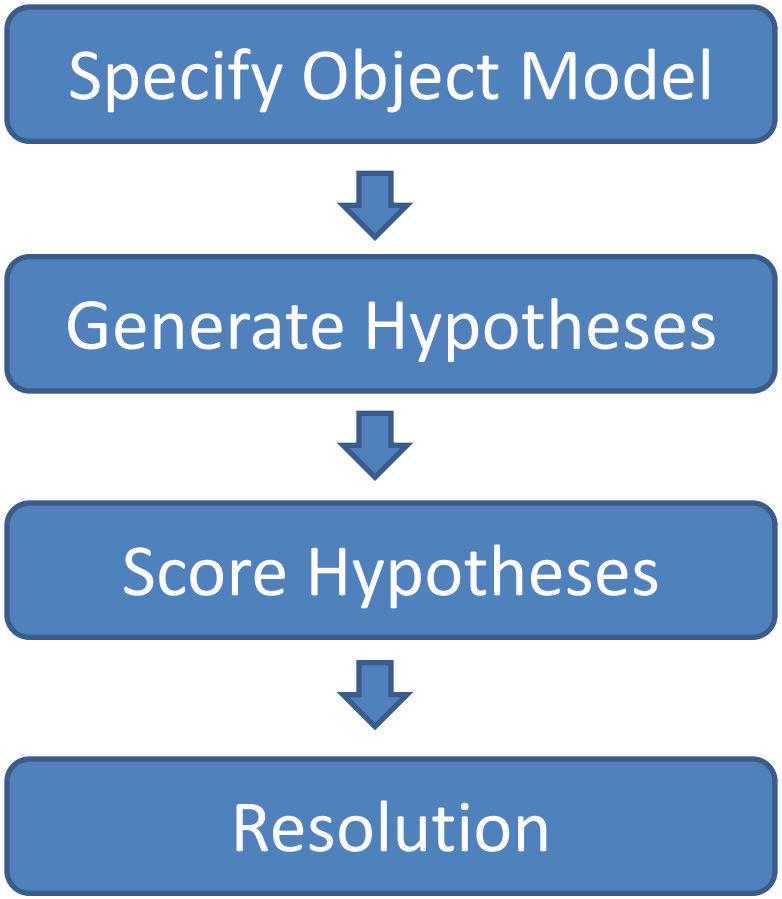
Strengths

- Works very well for non-deformable objects: faces, cars, upright pedestrians
- Fast detection

Weaknesses

- Not so well for highly deformable objects
- Not robust to occlusion
- Requires lots of training data

General Process of Object Recognition

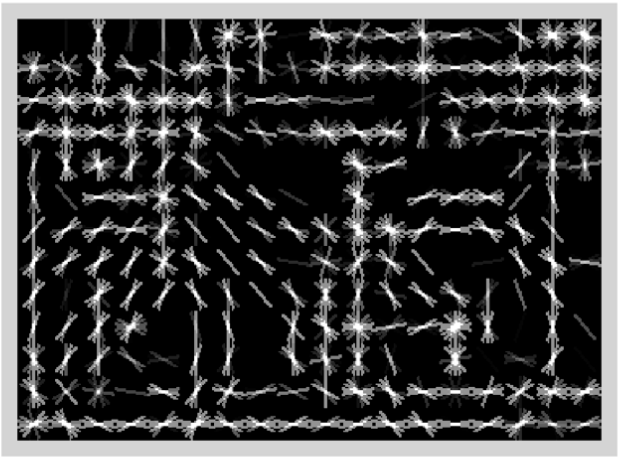


What are the object parameters?

Specifying an object model

1. Statistical Template in Bounding Box
   - Object is some (x,y,w,h) in image
   - Features defined wrt bounding box coordinates

Image

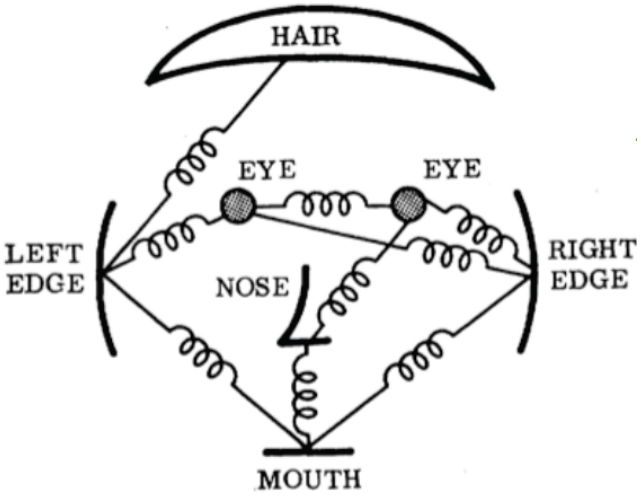


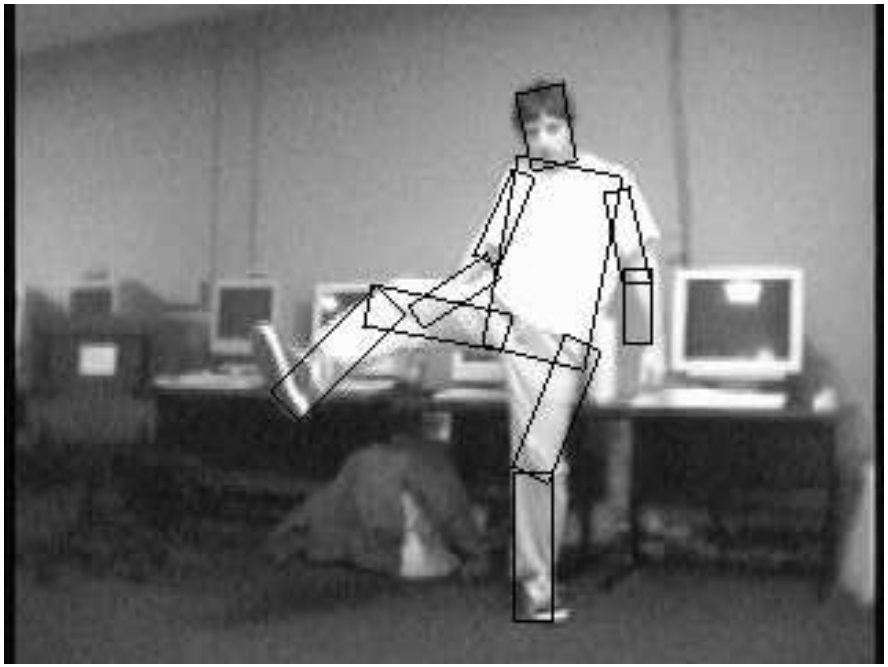
Template Visualization

Images from Felzenszwalb

Specifying an object model

2. Articulated parts model
   - Object is configuration of parts
   - Each part is detectable



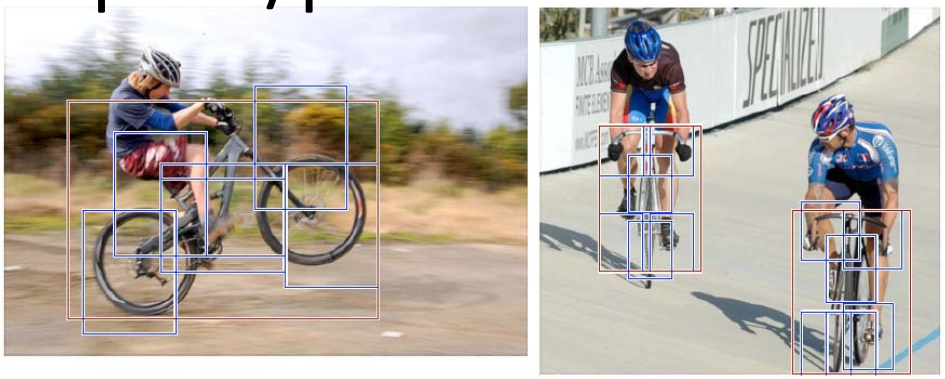


Images from Felzenszwalb

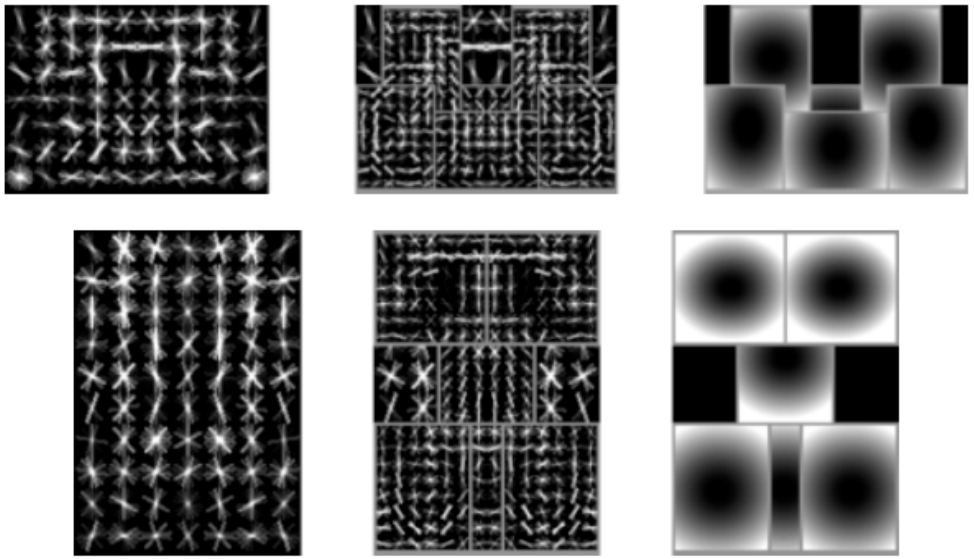
Specifying an object model

3. Hybrid template/parts model

Detections



Template Visualization



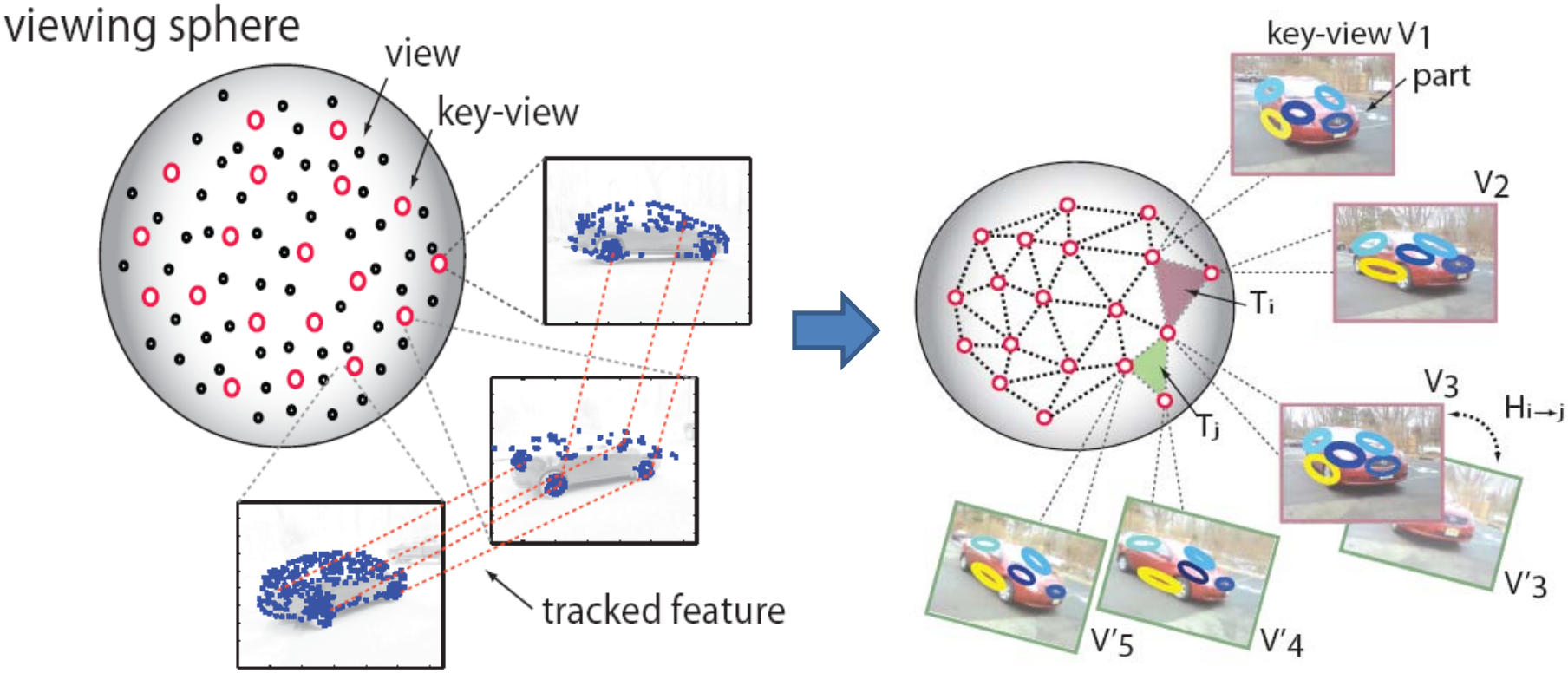
root filters coarse resolution | part filters finer resolution | deformation models

Felzenszwalb et al. 2008

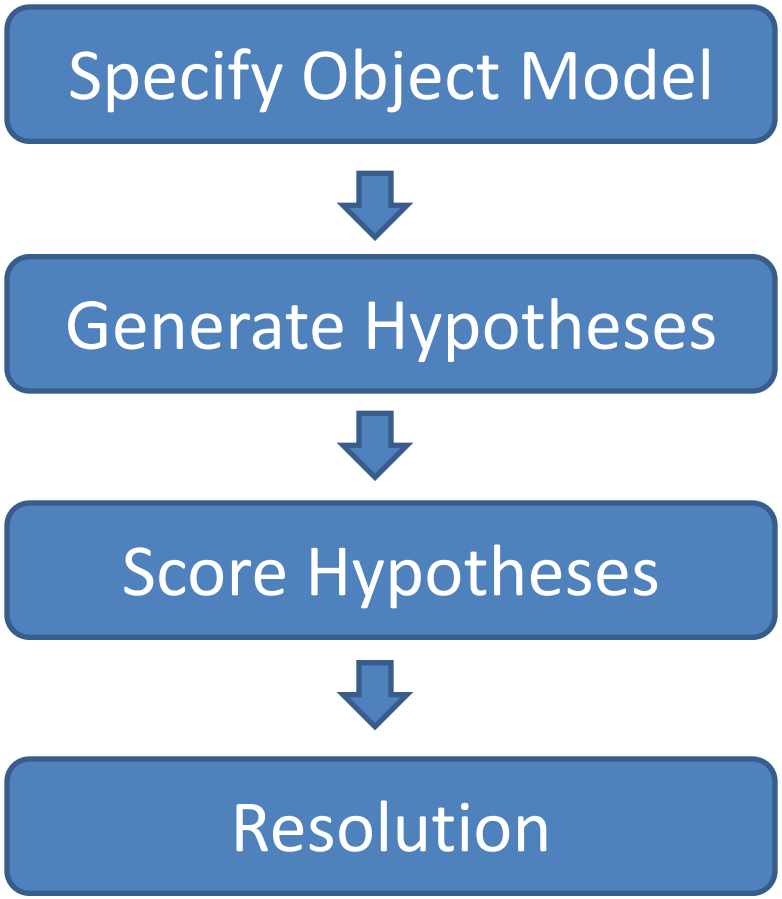
Specifying an object model

4. 3D-ish model

- Object is collection of 3D planar patches under affine transformation



General Process of Object Recognition



Propose an alignment of the model to the image

Generating hypotheses

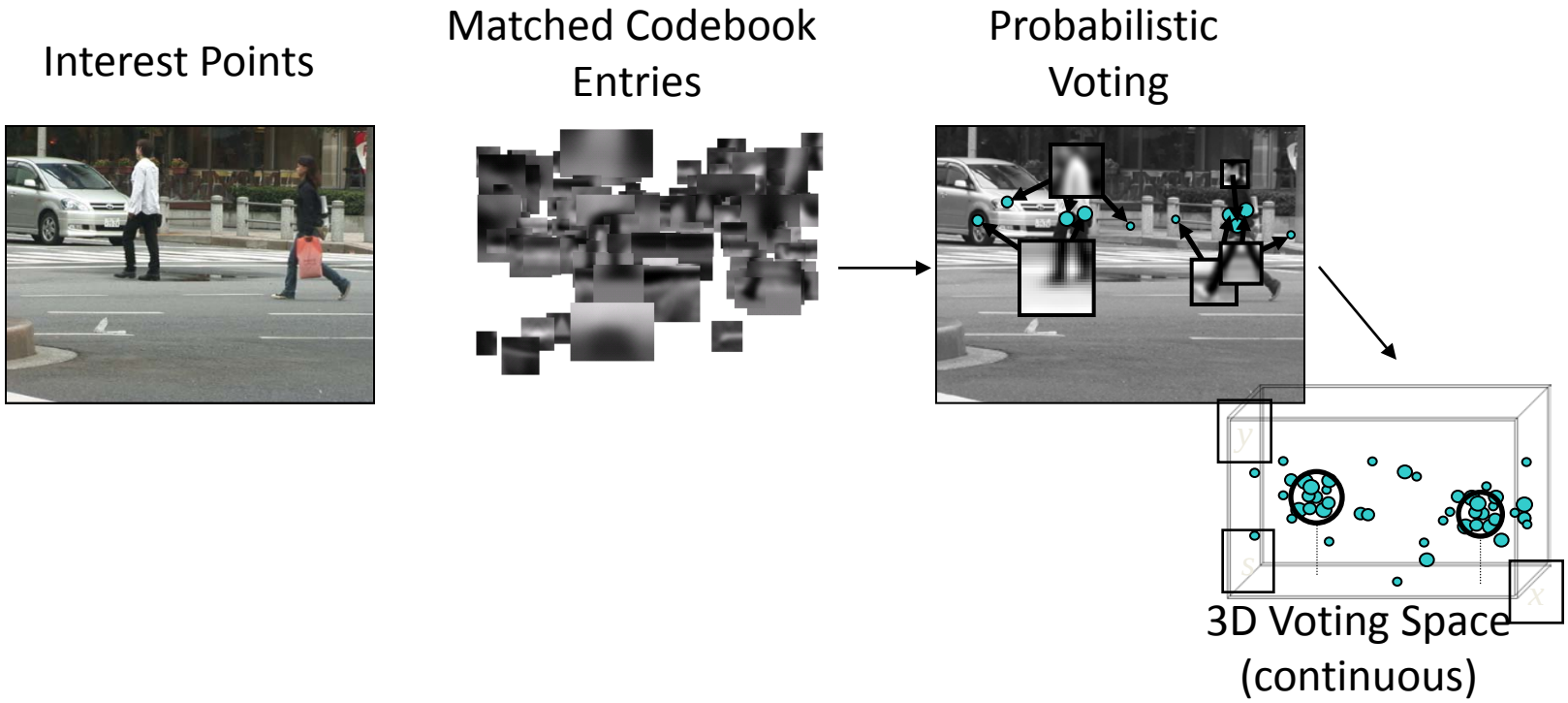
1. Sliding window
   - Test patch at each location and scale

Generating hypotheses

1. Sliding window
   - Test patch at each location and scale

Generating hypotheses

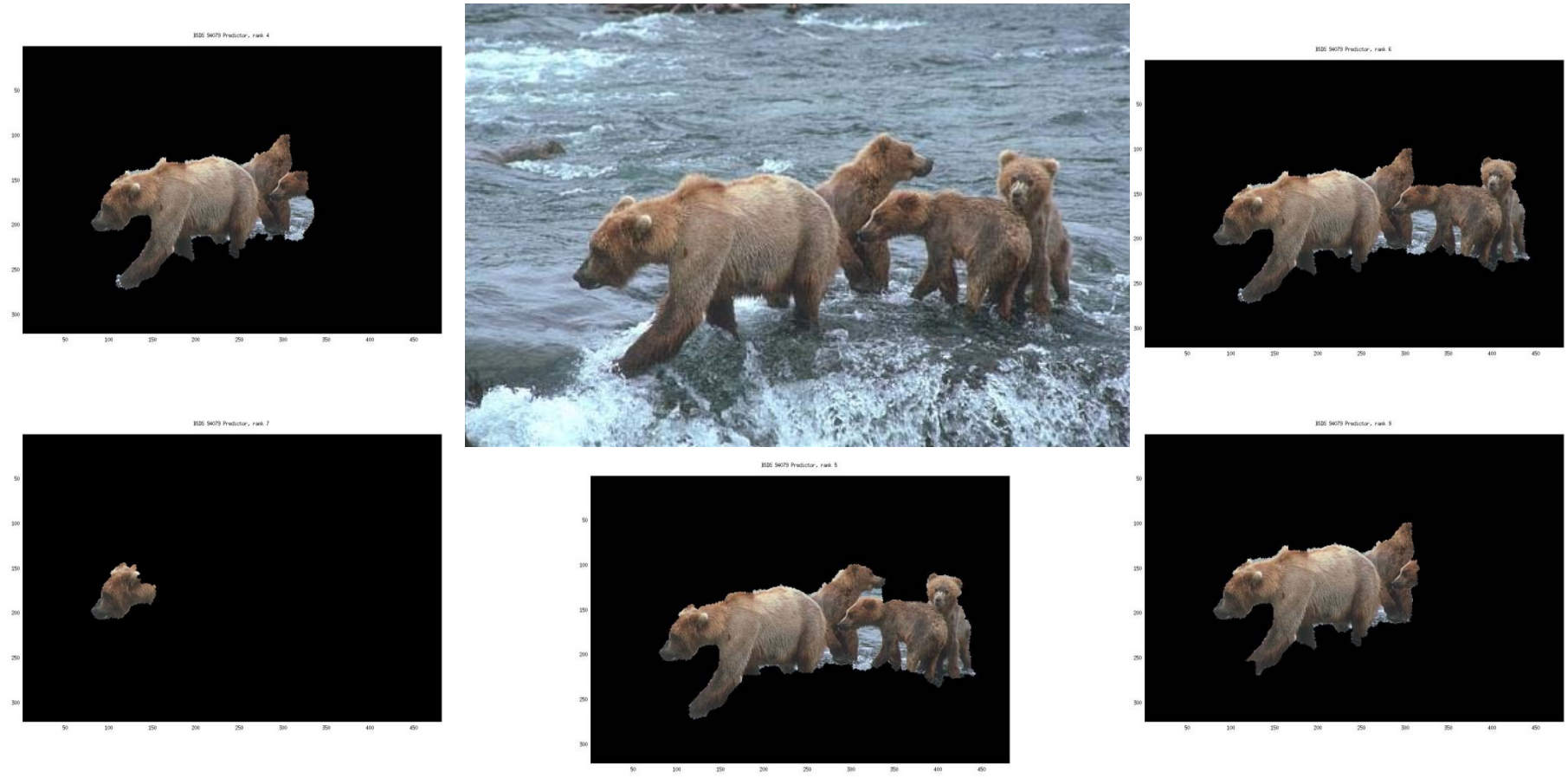
2. Voting from patches/keypoints



ISM model by Leibe et al.

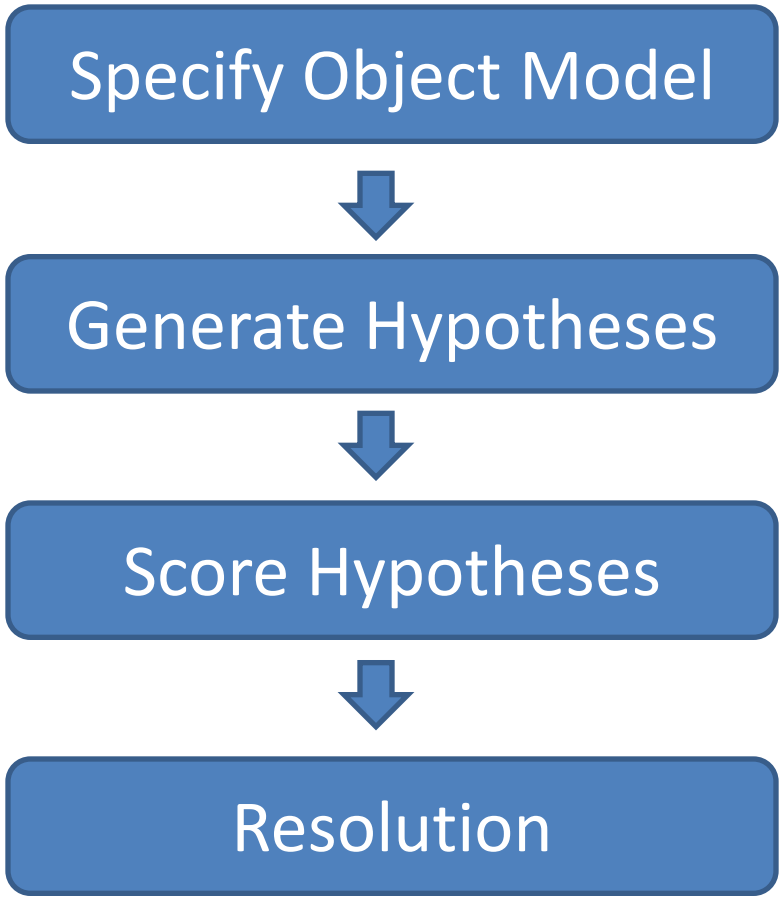
Generating hypotheses

3. Region-based proposal



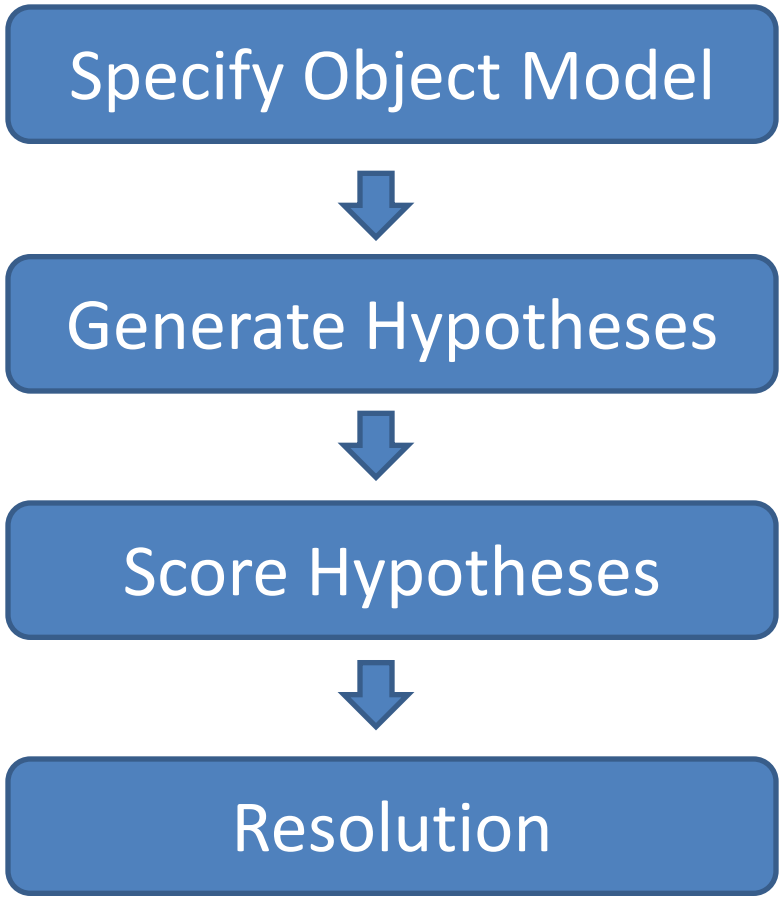
Endres Hoiem 2010

General Process of Object Recognition



Mainly-gradient based features, usually based on summary representation, many classifiers

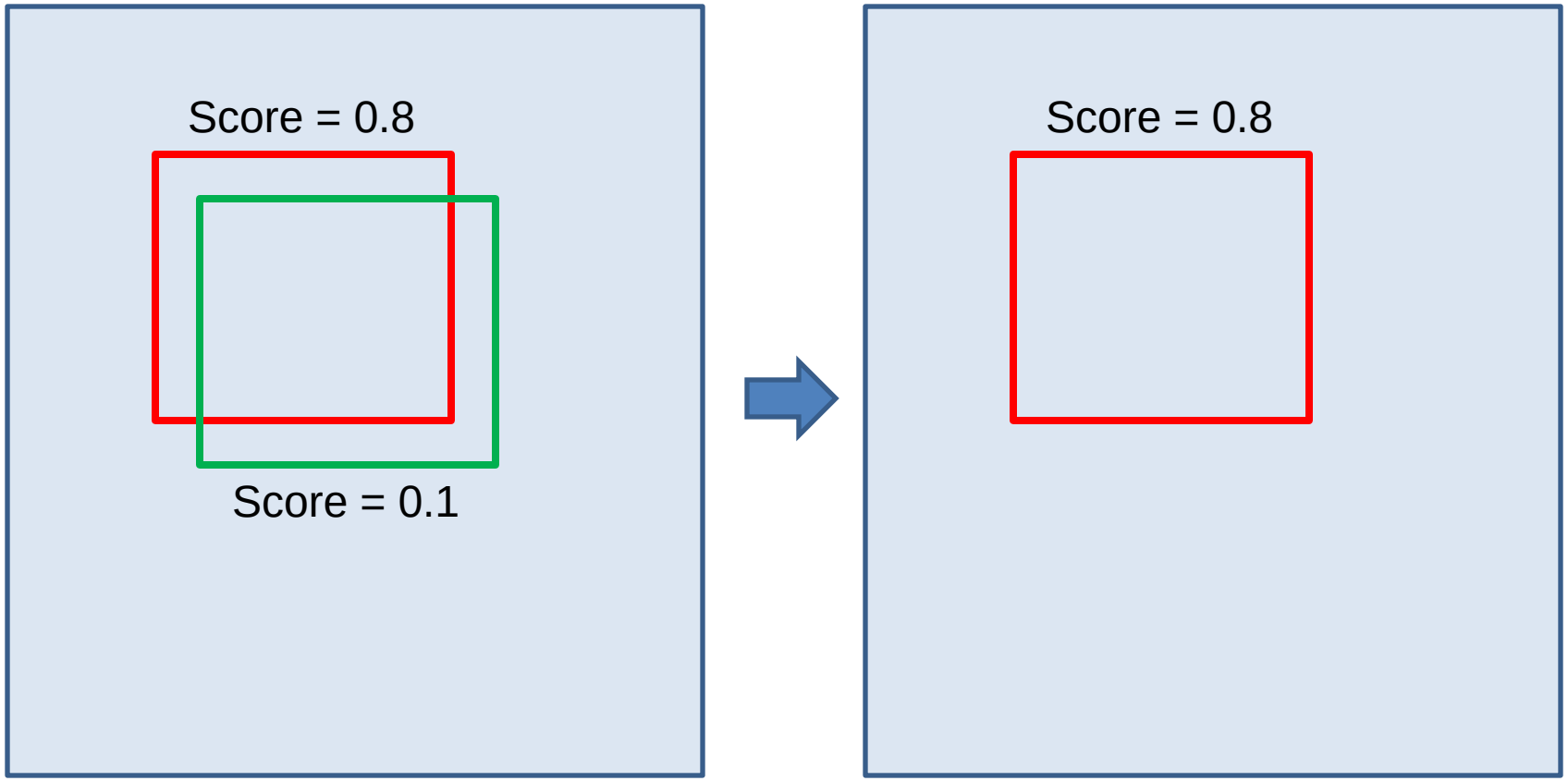
General Process of Object Recognition



Rescore each proposed object based on whole set

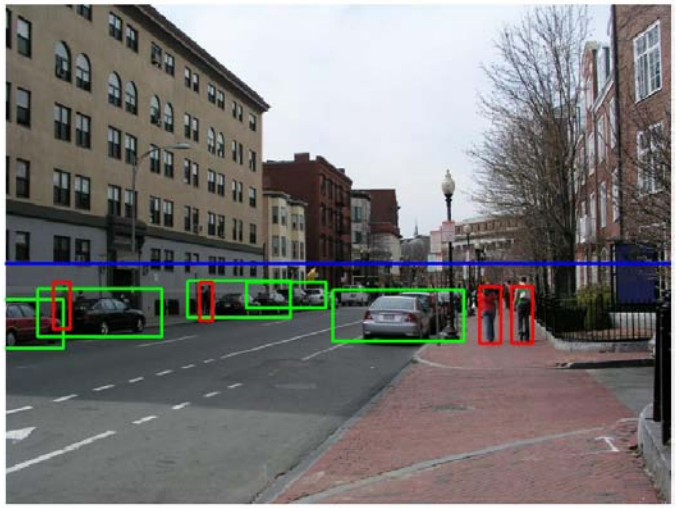
Resolving detection scores

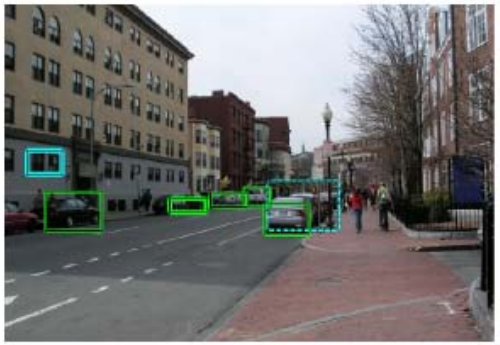
1. Non-max suppression



Resolving detection scores

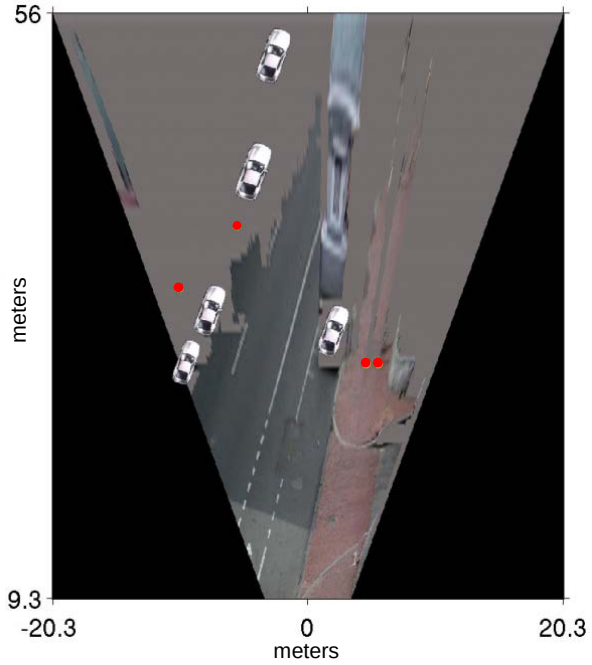
2. Context/reasoning





(g) Car Detections: Local

(h) Ped Detections: Local



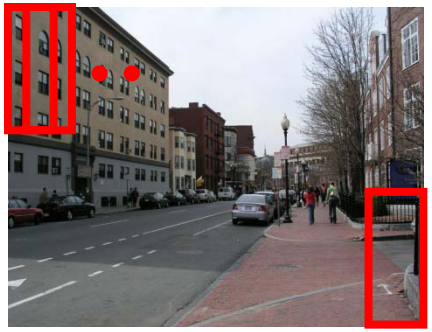
Hoiem et al. 2006

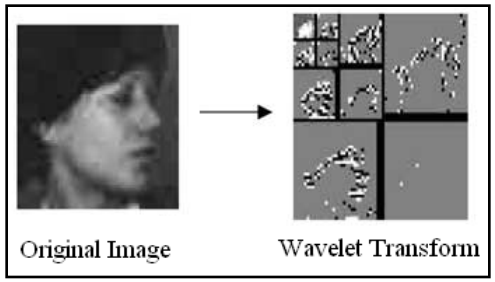
Influential Works in Detection

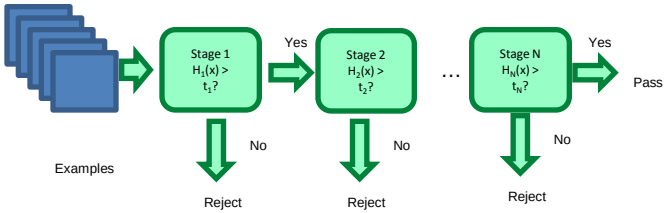
- Sung-Poggio (1994, 1998) : ~1450 citations
  - Basic idea of statistical template detection (I think), bootstrapping to get "face-like" negative examples, multiple whole-face prototypes (in 1994)
- Rowley-Baluja-Kanade (1996-1998) : ~2900
  - "Parts" at fixed position, non-maxima suppression, simple cascade, rotation, pretty good accuracy, fast
- Schneiderman-Kanade (1998-2000,2004) : ~1250
  - Careful feature engineering, excellent results, cascade
- Viola-Jones (2001, 2004) : ~6500
  - Haar-like features, Adaboost as feature selection, hyper-cascade, very fast, easy to implement
- Dalal-Triggs (2005) : ~2000
  - Careful feature engineering, excellent results, HOG feature, online code
- Felzenszwalb-Huttenlocher (2000): ~800
  - Efficient way to solve part-based detectors
- Felzenszwalb-McAllester-Ramanan (2008)? ~350
  - Excellent template/parts-based blend

Things to remember

- Sliding window for search
- Features based on differences of intensity (gradient, wavelet, etc.)
  - Excellent results require careful feature design
- Boosting for feature selection (also L1-logistic regression)
- Integral images, cascade for speed
- Bootstrapping to deal with many, many negative examples

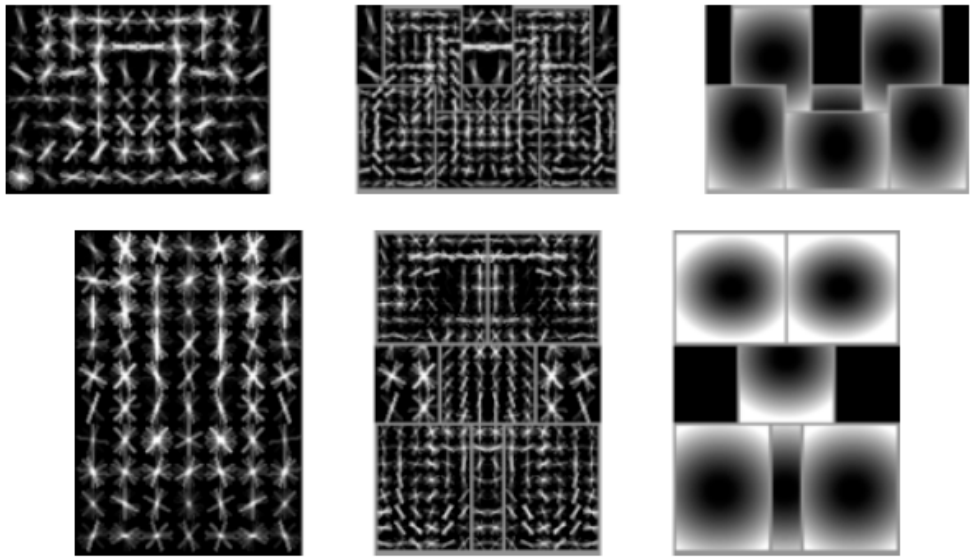






Next class

- Deformable parts models and the distance transform



root filters
coarse resolution

part filters
finer resolution

deformation
models