Object Category Detection: Statistical Templates

Computer Vision

CS 543 / ECE 549

University of Illinois

Derek Hoiem

Logistics

- HW 5 due next Monday

- Final project
  - Posters on May 8, 7-10pm (final exam period)
  - Papers due following Monday (one per group)

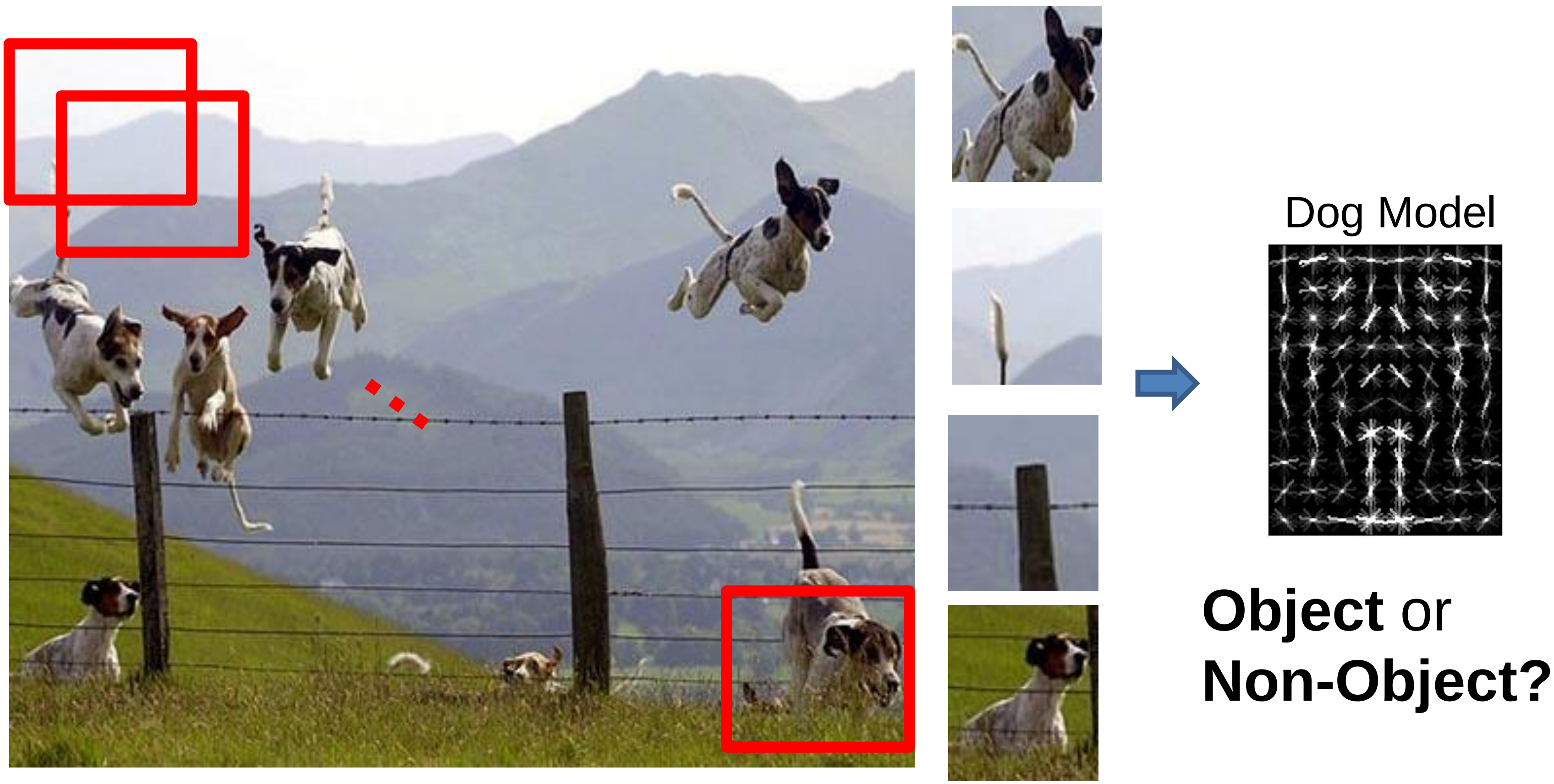
- Remaining classes
  - Object detection/tracking: next three classes
  - Action recognition
  - 3D scenes/context
  - Summary lecture and feedback (2^{nd} to last day)
  - I need to miss last class – Jiabin will teach convolutional neural networks

Today's class: Object Category Detection

- Overview of object category detection
- Statistical template matching
  - Dalal-Triggs pedestrian detector (basic concept)
  - Viola-Jones detector (cascades, integral images)
  - R-CNN detector (object proposals/CNN)

Object Category Detection

- Focus on object search: “Where is it?”
- Build templates that quickly differentiate object patch from background patch



Challenges in modeling the object class

Illumination

Object pose

Clutter

Occlusions

Intra-class appearance

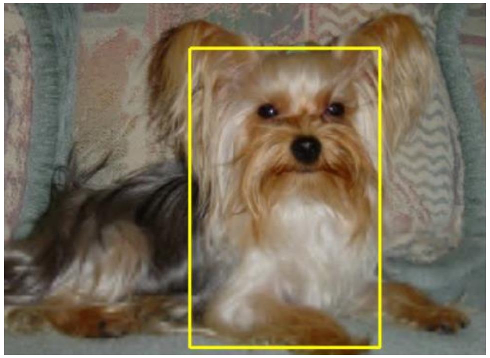
Viewpoint

Slide from K. Grauman, B. Leibe

Challenges in modeling the non-object class

True Detections

Bad Localization

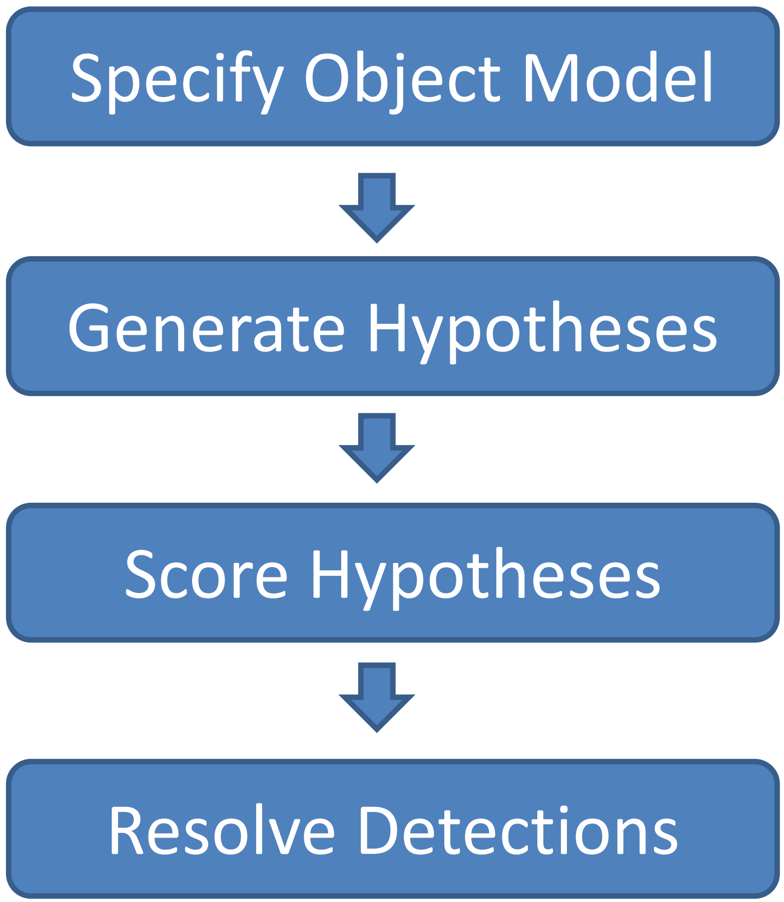


Confused with Similar Object

Misc. Background

Confused with Dissimilar Objects

General Process of Object Recognition

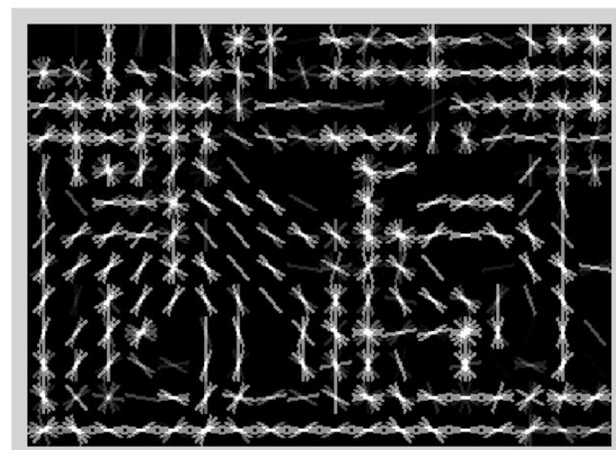


What are the object parameters?

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

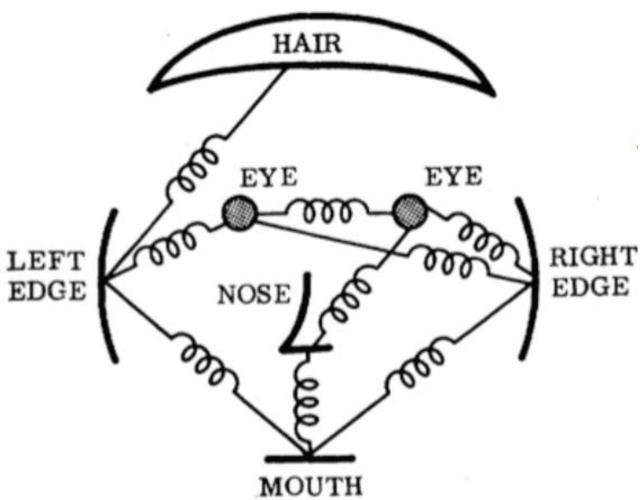


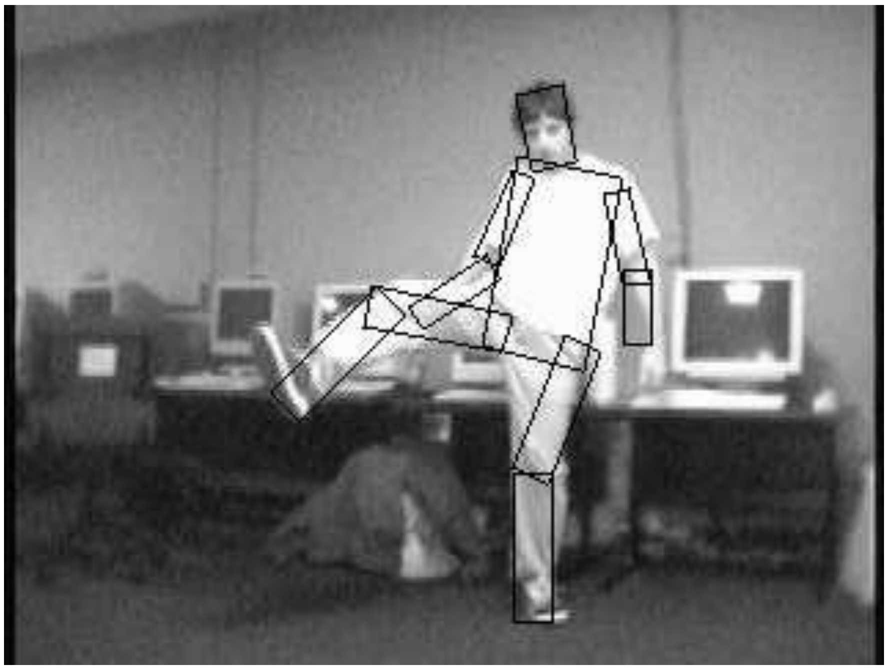
Template Visualization

Images from Felzenszwalb

Specifying an object model

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



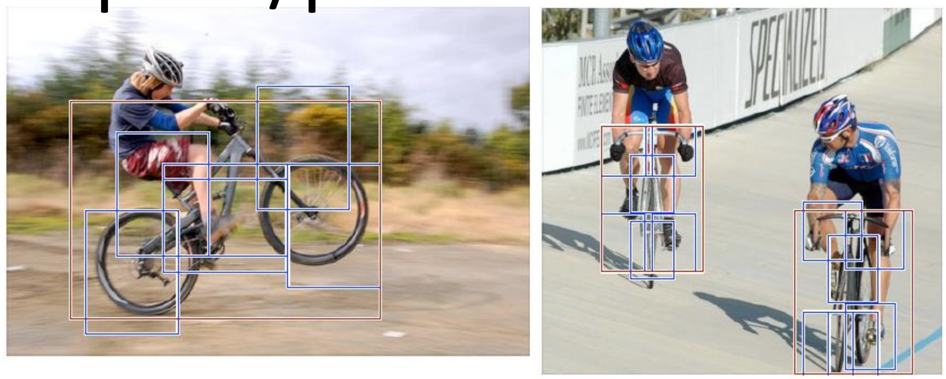


Images from Felzenszwalb

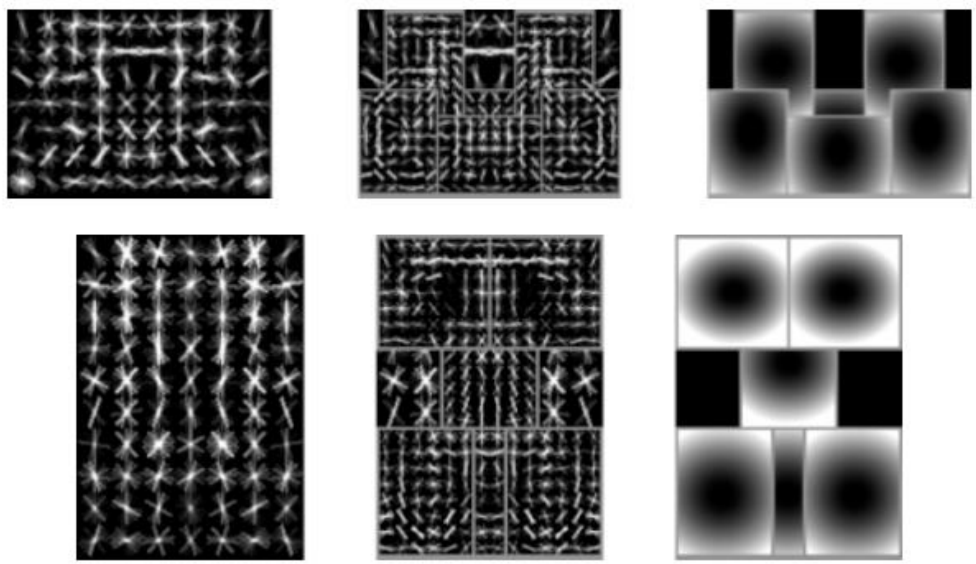
Specifying an object model

3. Hybrid template/parts model

Detections



Template Visualization



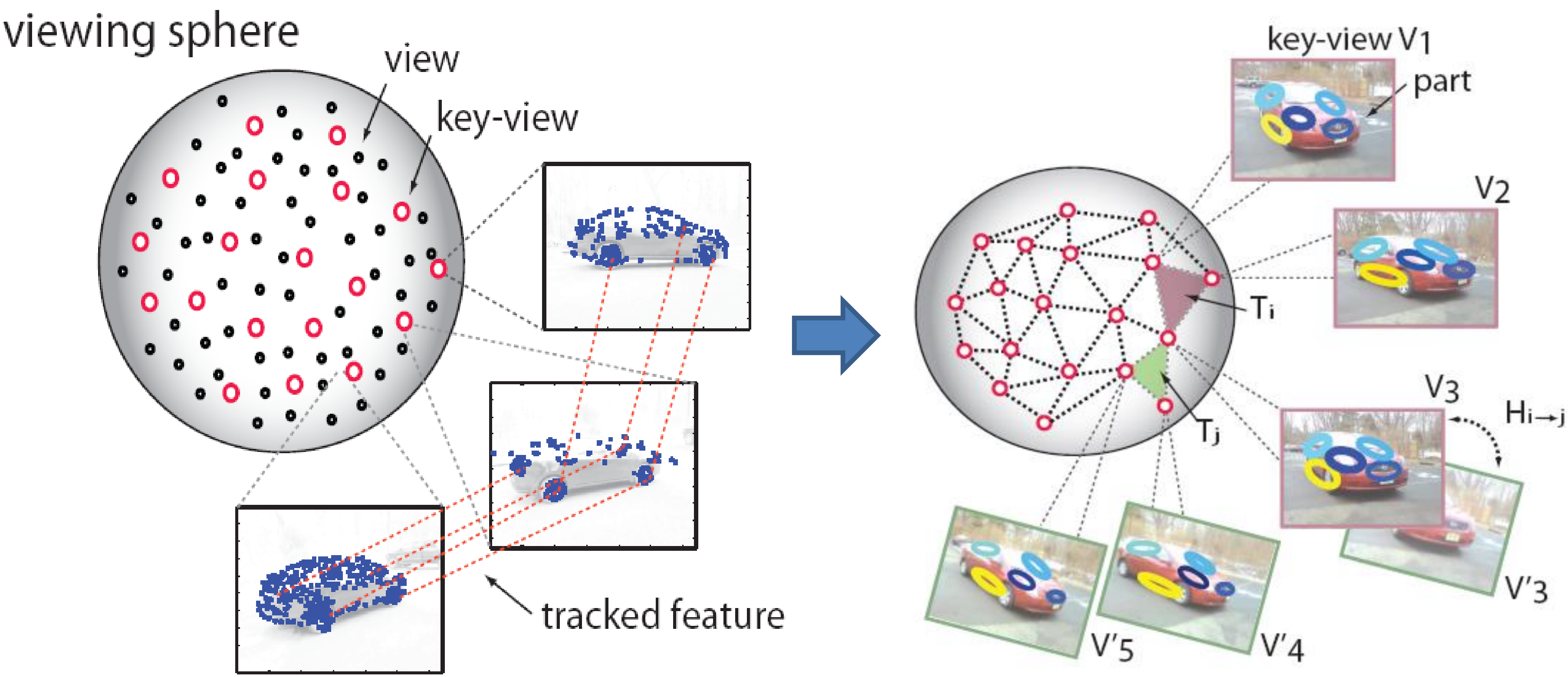
root filters coarse resolution | part filters finer resolution | deformation models

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

Specify Object Model

↓

Generate Hypotheses

↓

Score Hypotheses

↓

Resolve Detections

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

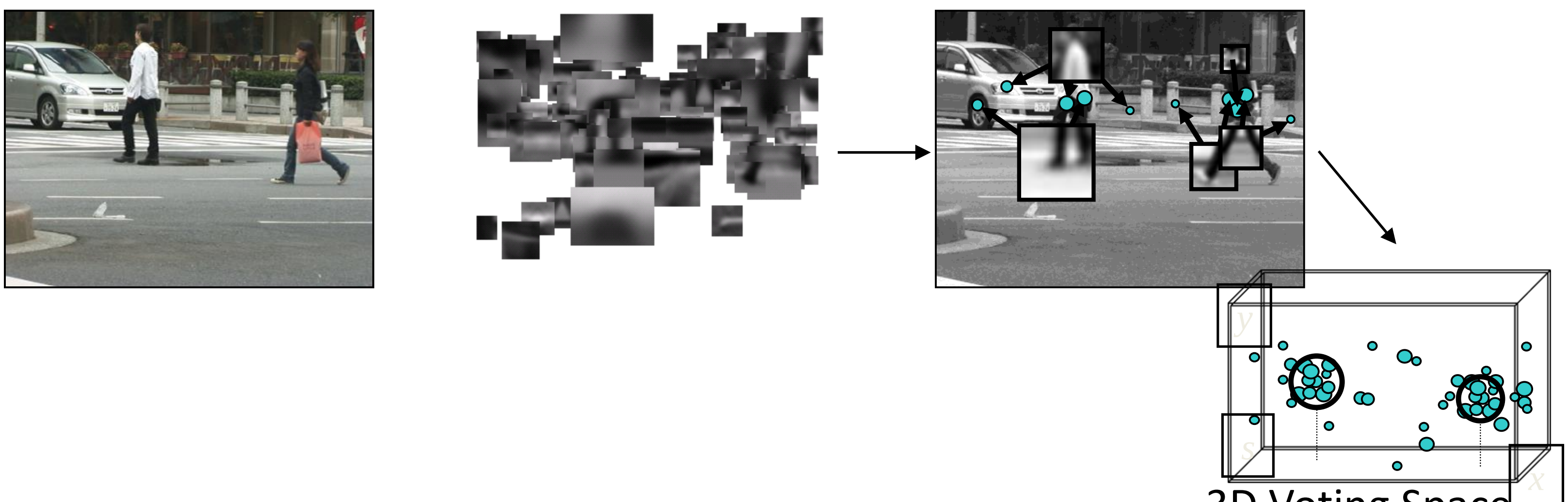
Generating hypotheses

2. Voting from patches/keypoints

Interest Points

Matched Codebook Entries

Probabilistic Voting

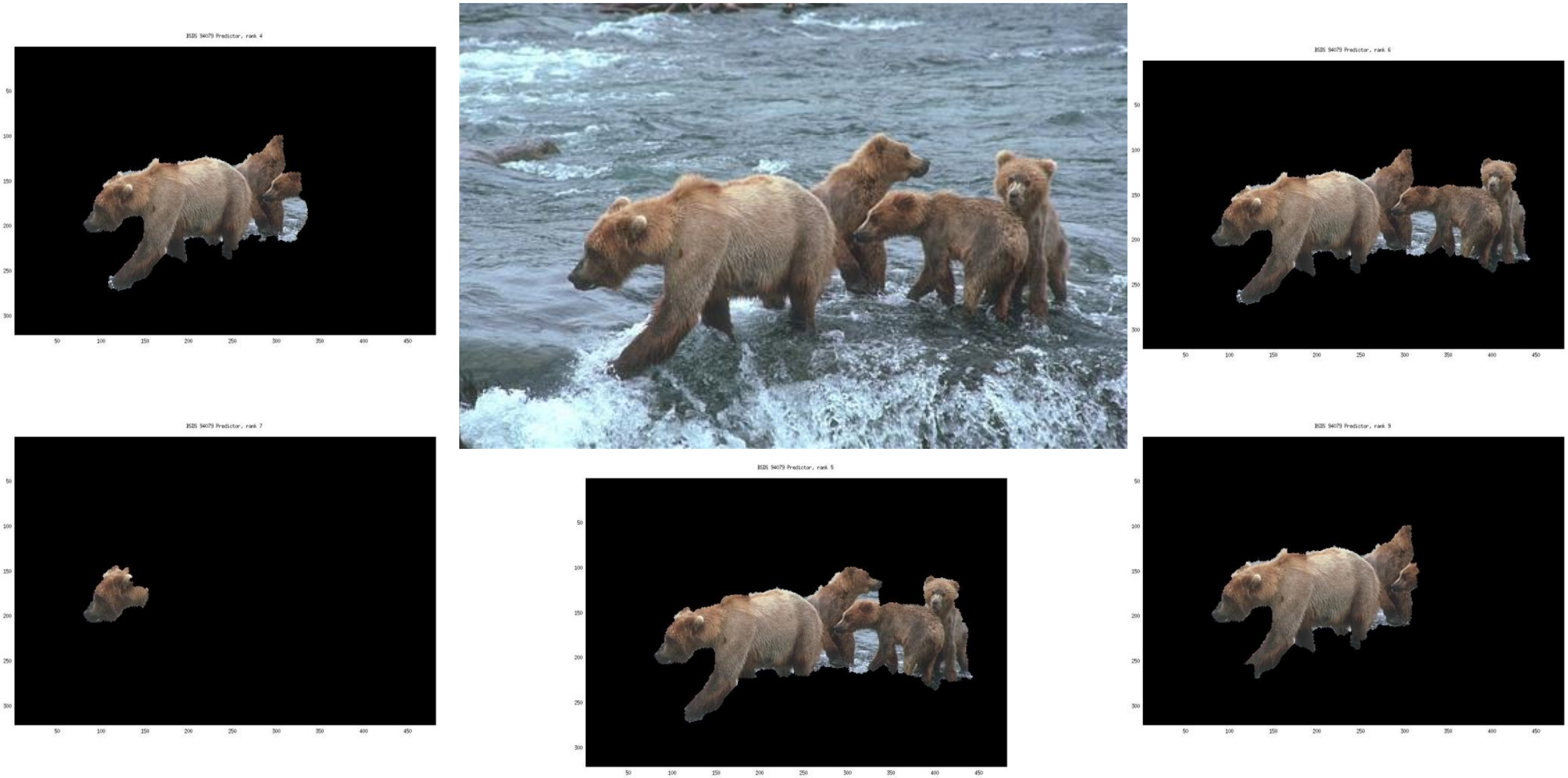


3D Voting Space (continuous)

ISM model by Leibe et al.

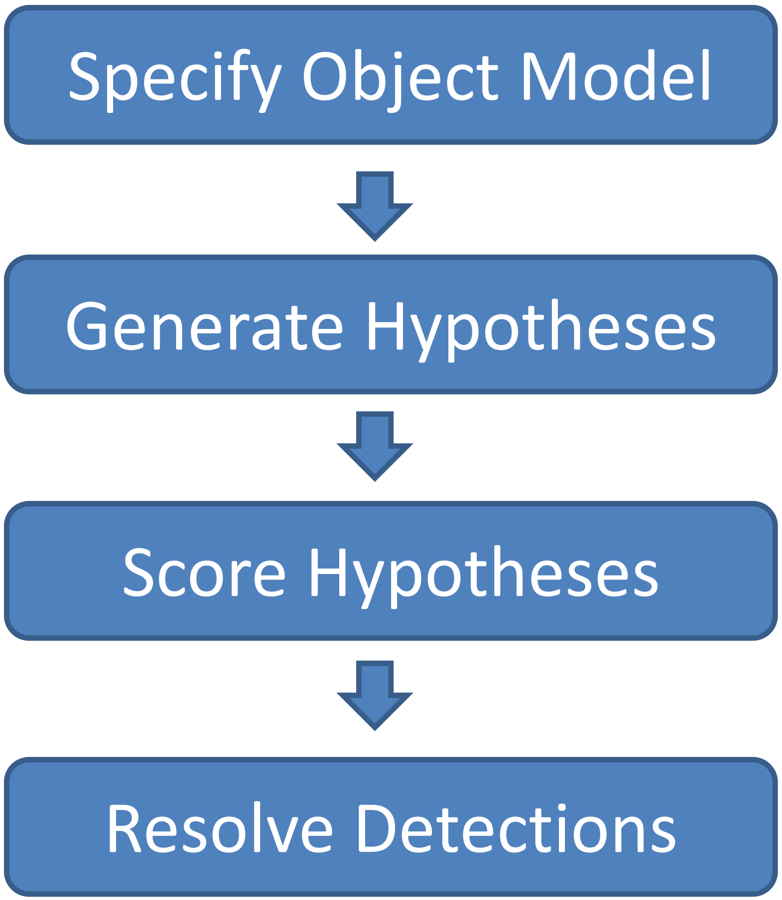
Generating hypotheses

3. Region-based proposal



Endres Hoiem 2010

General Process of Object Recognition



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

General Process of Object Recognition

Specify Object Model

↓

Generate Hypotheses

↓

Score Hypotheses

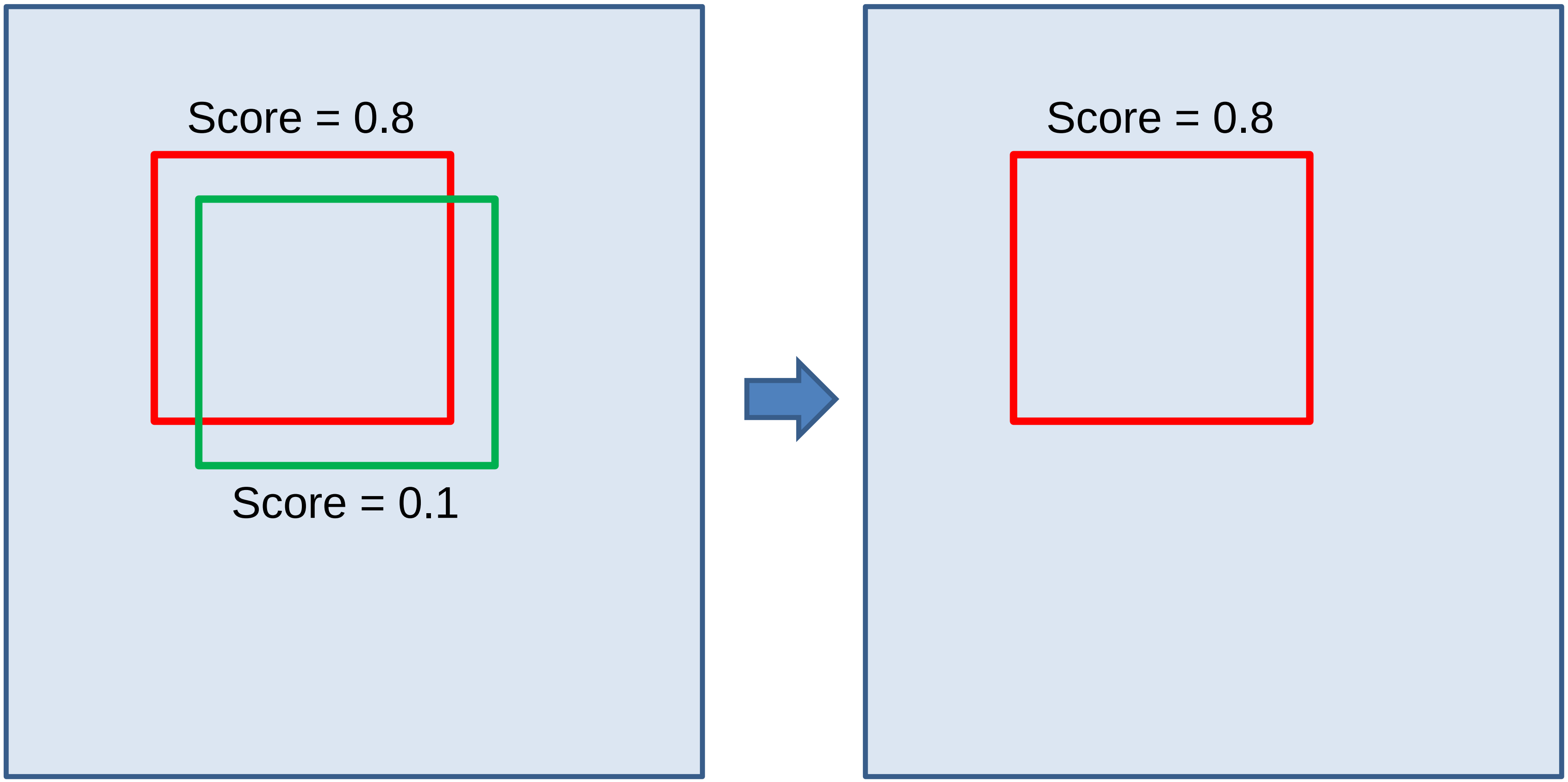
↓

Resolve Detections

Rescore each proposed object based on whole set

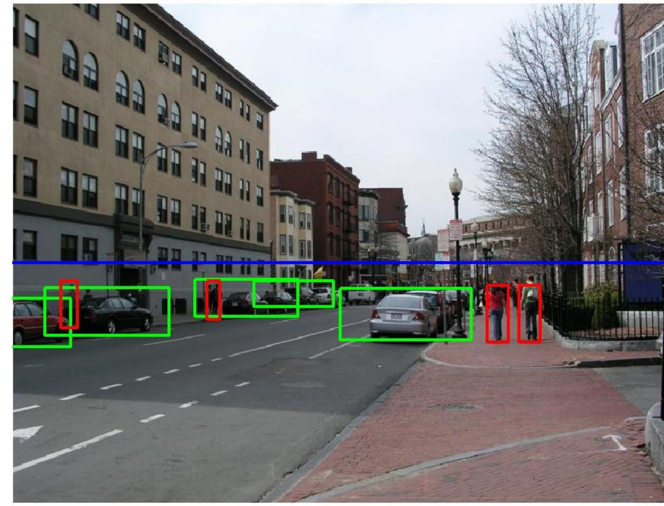
Resolving detection scores

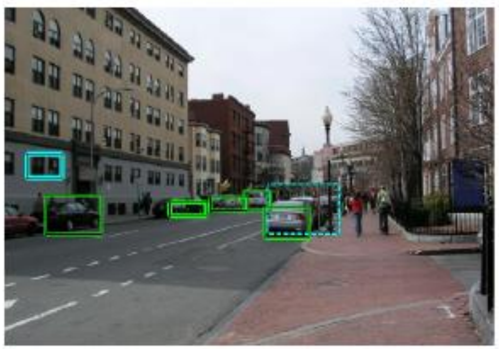
1. Non-max suppression



Resolving detection scores

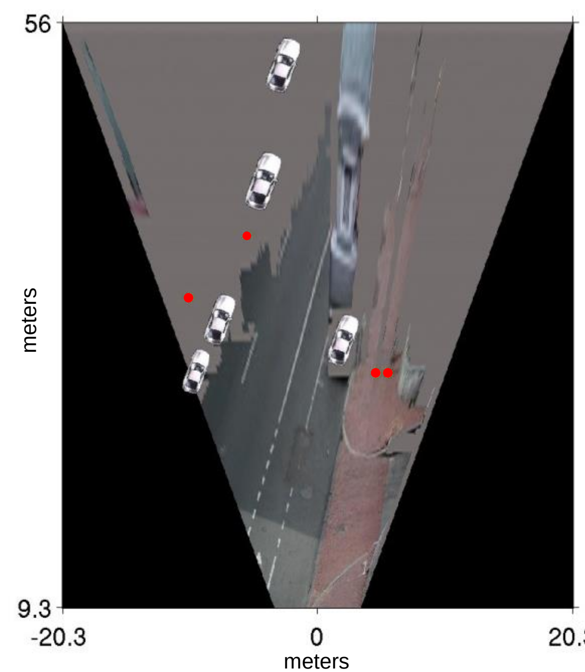
2. Context/reasoning





(g) Car Detections: Local

(h) Ped Detections: Local



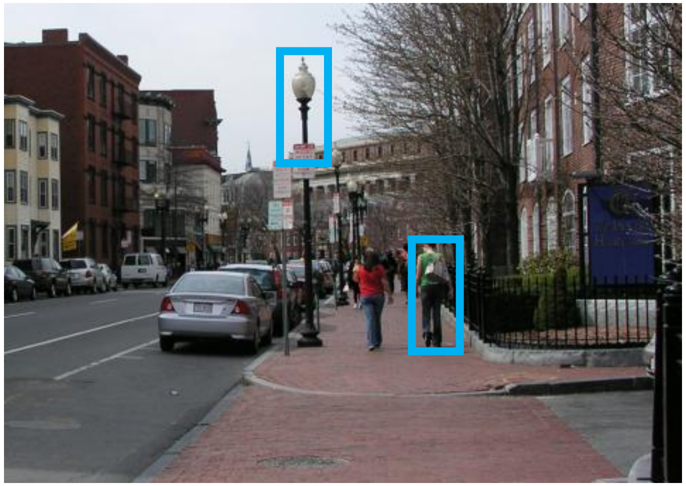
Hoiem et al. 2006

Object category detection in computer vision

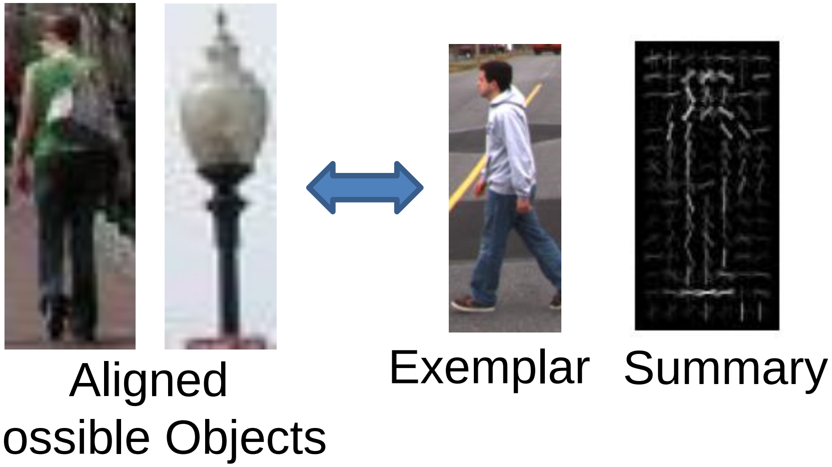
Goal: detect all pedestrians, cars, monkeys, etc in image

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

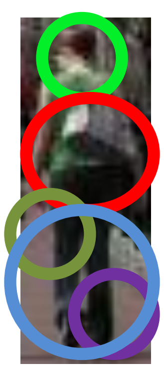
Each window is separately classified

Statistical Template

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

$$+3 +2 -2 -1 -2.5 = -0.5 \overset{?}{>} 7.5$$

Non-object



$$+4 +1 +0.5 +3 +0.5 = 10.5 \overset{?}{>} 7.5$$

Object

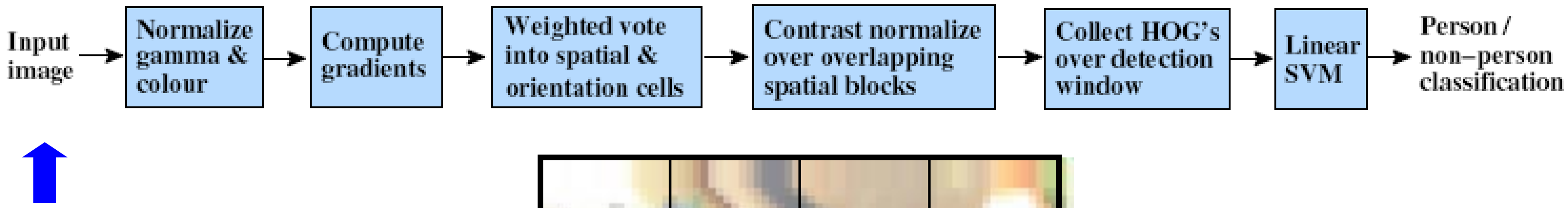
Design challenges

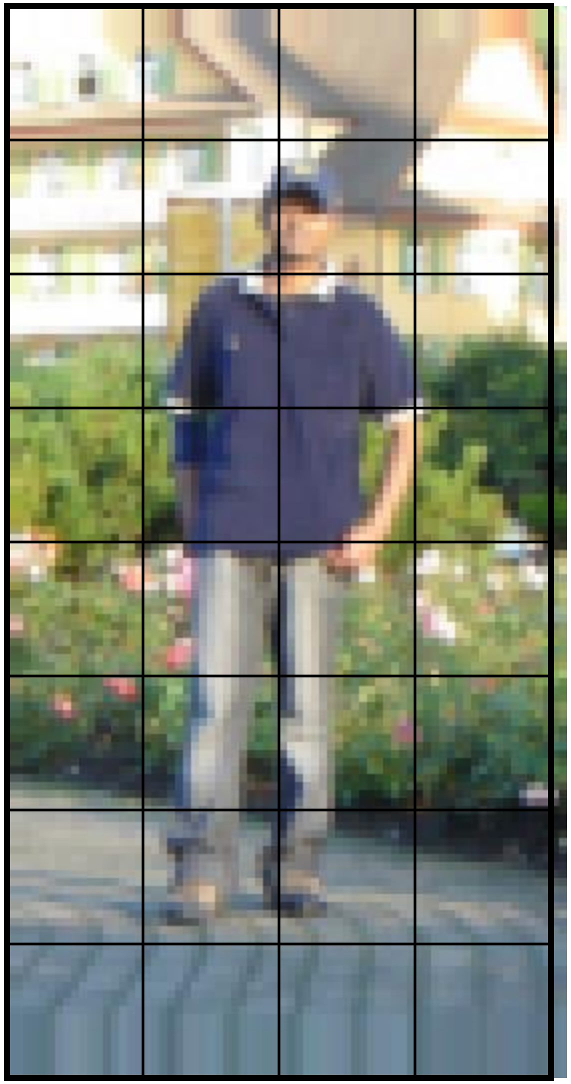
- How to efficiently search for likely objects
  - Even simple models require searching hundreds of thousands of positions and scales
- Feature design and scoring
  - How should appearance be modeled? What features correspond to the object?
- How to deal with different viewpoints?
  - Often train different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

Example: Dalal-Triggs pedestrian detector

1. Extract fixed-sized (64x128 pixel) window at each position and scale
2. Compute HOG (histogram of gradient) features within each window
3. Score the window with a linear SVM classifier
4. Perform non-maxima suppression to remove overlapping detections with lower scores

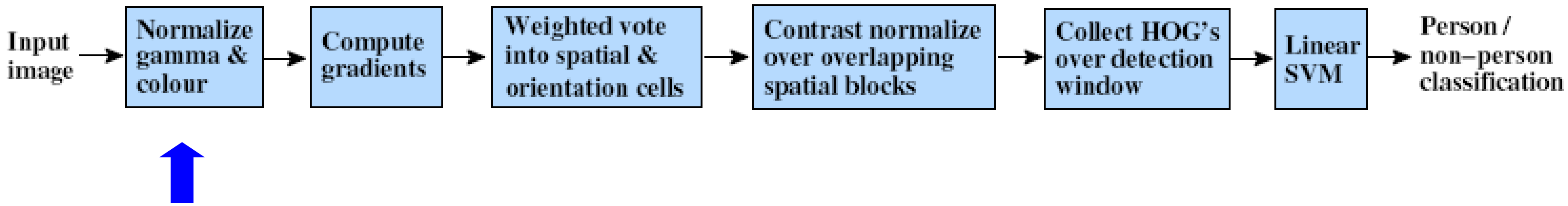
Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05





Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

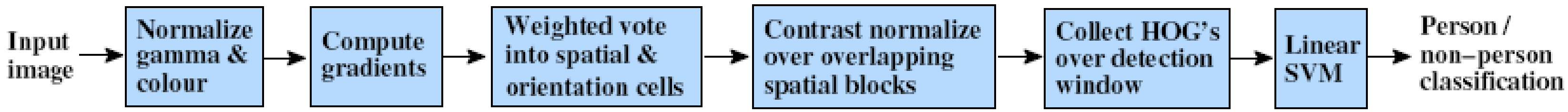


- Tested with
  - RGB
  - LAB
  - Grayscale

Slightly better performance vs. grayscale

- Gamma Normalization and Compression
  - Square root
  - Log

Very slightly better performance vs. no adjustment



Outperforms

-1	0	1

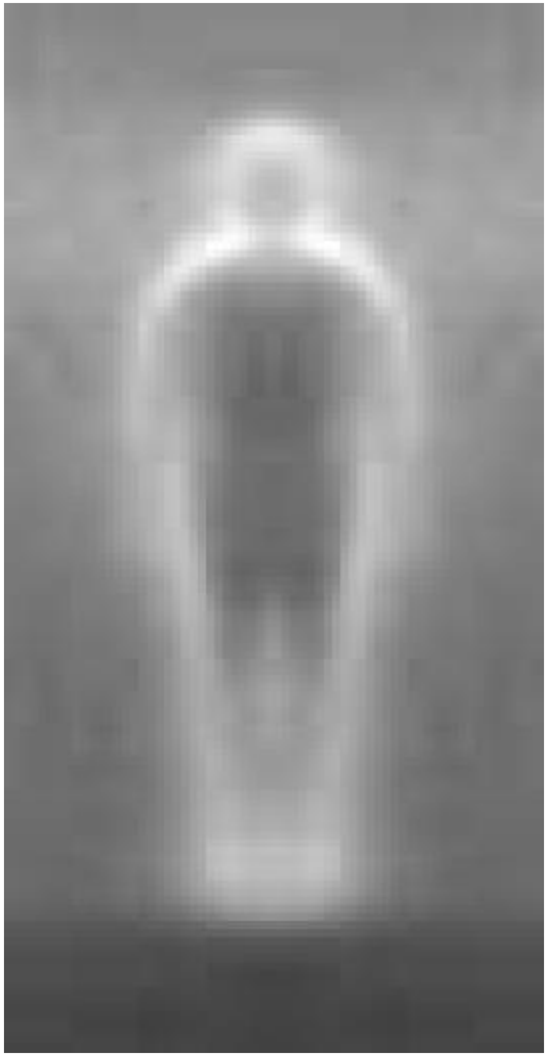
centered

-1	1

uncentered

1	-8	0	8	-1

cubic-corrected



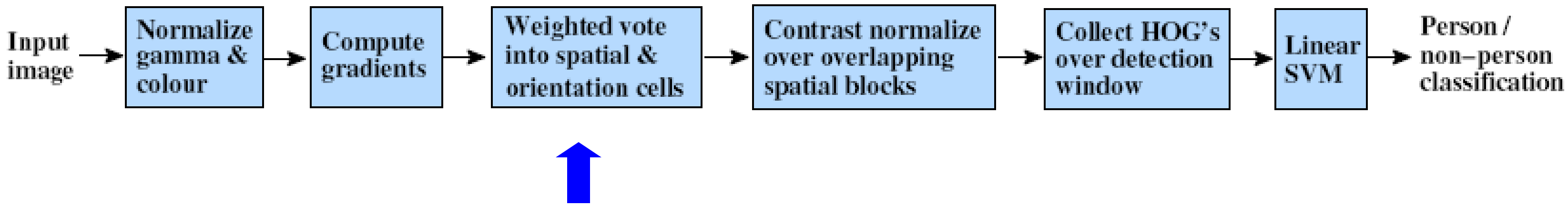
0	1
-1	0

diagonal

-1	0	1
-2	0	2
-1	0	1

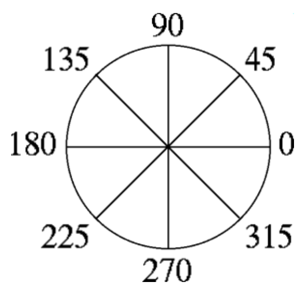
Sobel

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

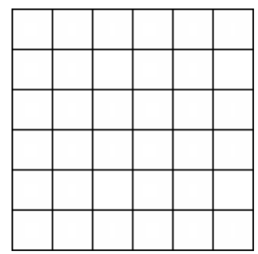


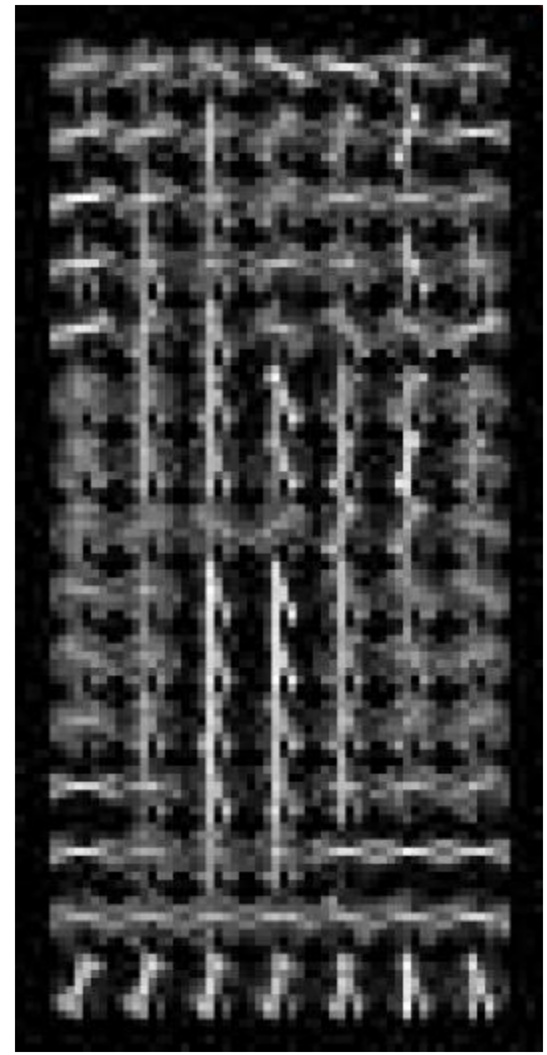
- Histogram of gradient orientations

Orientation: 9 bins
(for unsigned angles)



Histograms in
8x8 pixel cells

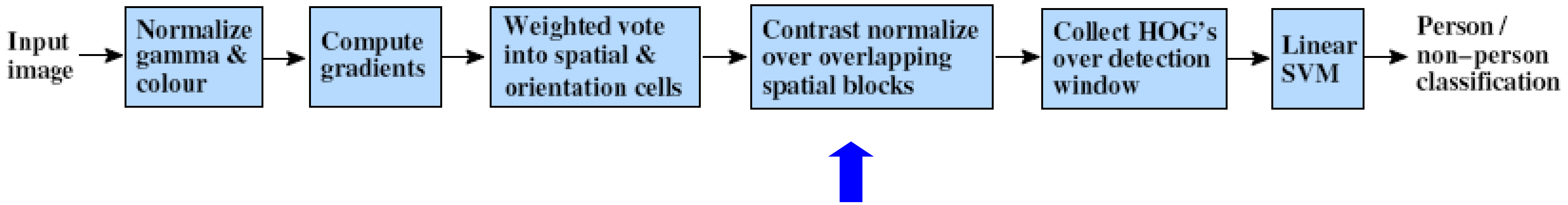




  - Votes weighted by magnitude
  - Bilinear interpolation between cells

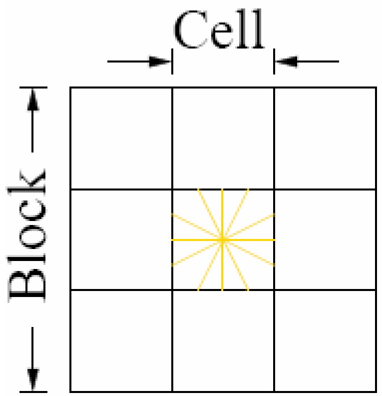
Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05



R-HOG

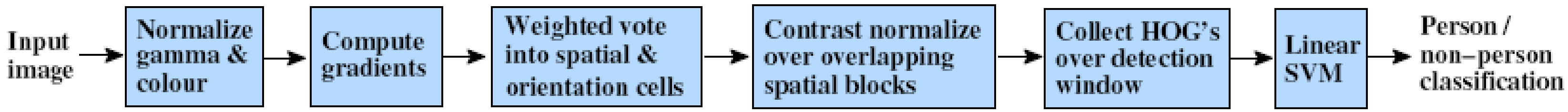
Normalize with respect to surrounding cells



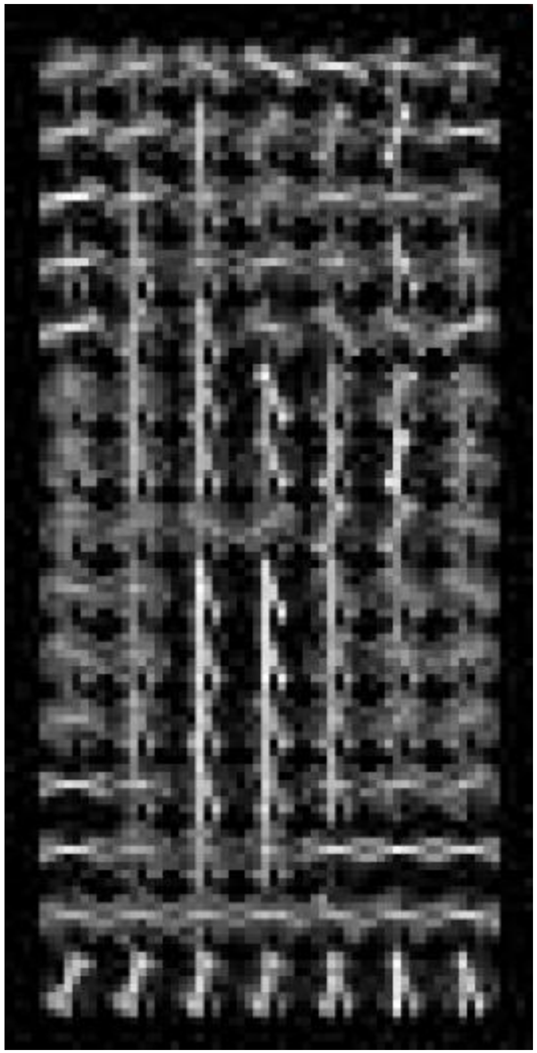
$$L2-norm : v \longrightarrow v/\sqrt{||v||_2^2+\epsilon^2}$$

Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05



X=



orientations

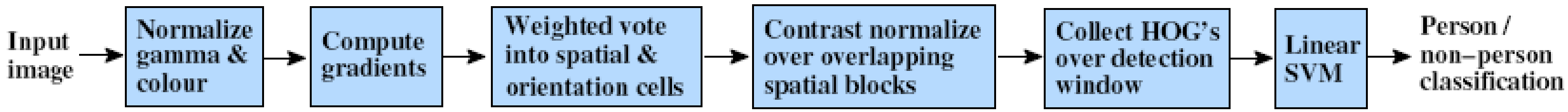
features = 15 x 7 x 9 x 4 = 3780

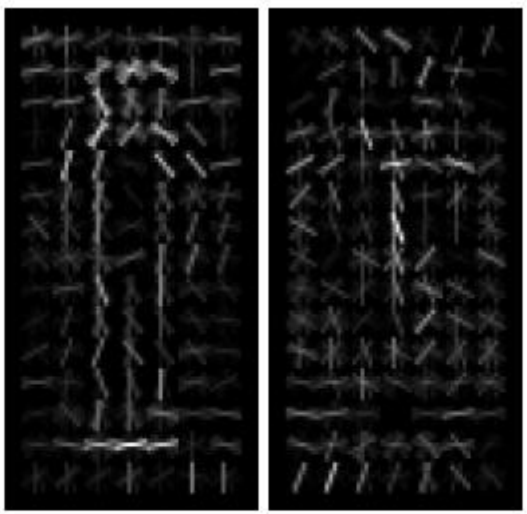
cells

normalizations by neighboring cells

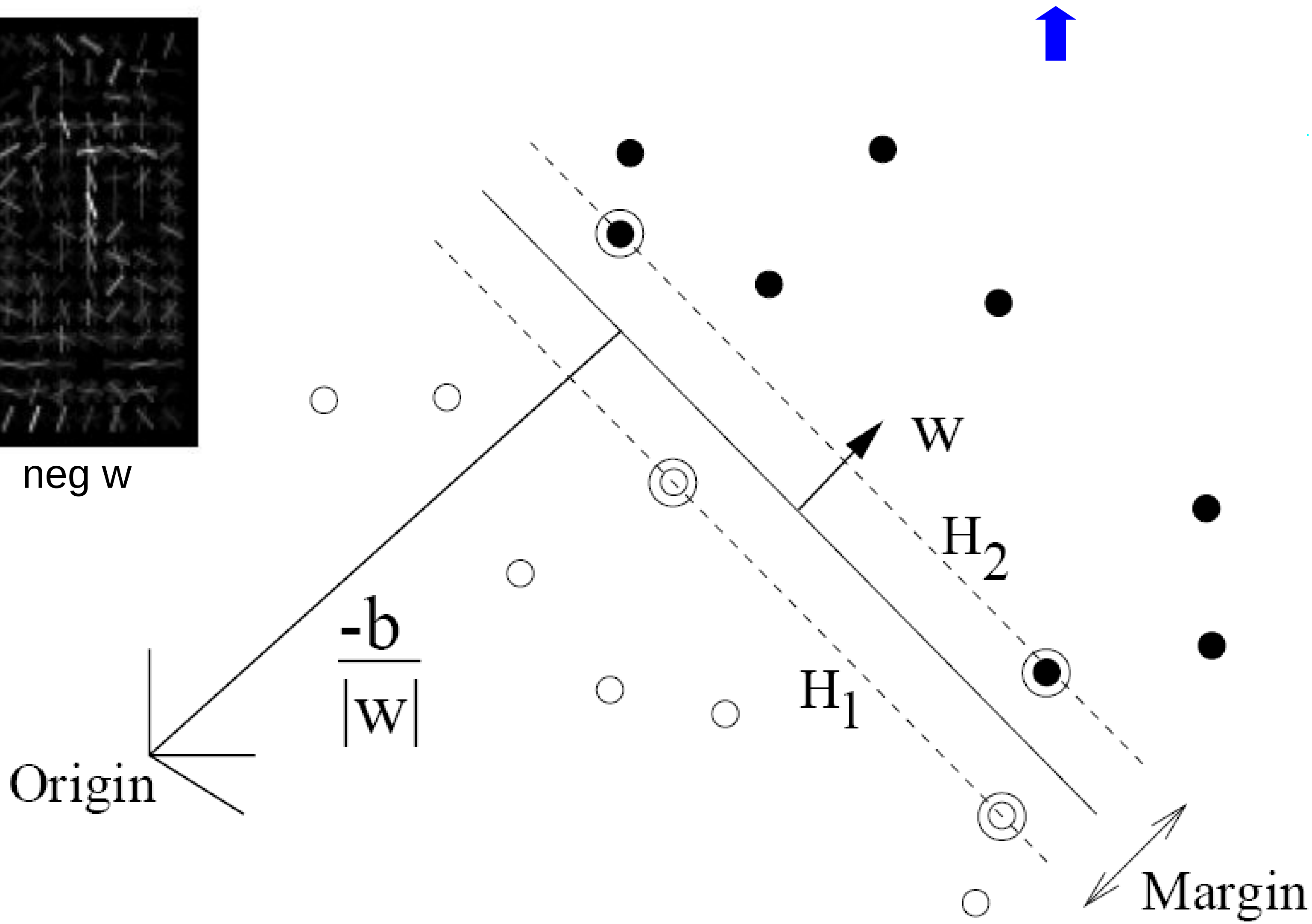
Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05



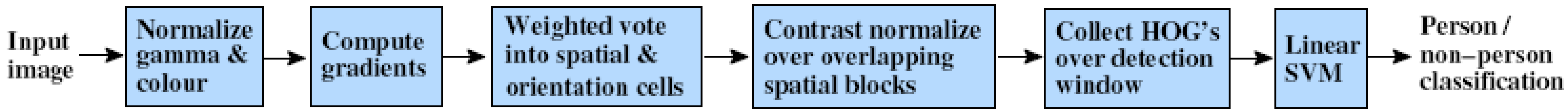


pos w neg w



Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05



$$0.16 = w^T x - b$$

$$sign(0.16) = 1$$

$=>$ pedestrian

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

Detection examples

2 minute break

Something to think about…

- Sliding window detectors work
  - *very well* for faces
  - *fairly well* for cars and pedestrians
  - *badly* for cats and dogs
- Why are some classes easier than others?

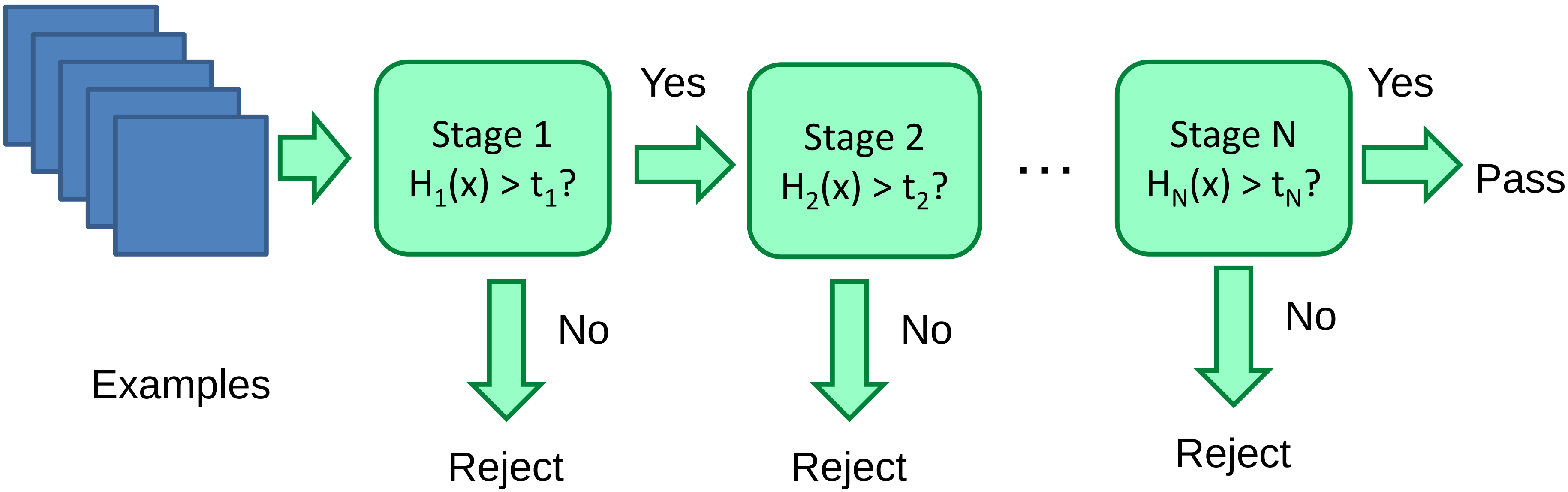
Viola-Jones sliding window detector

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).

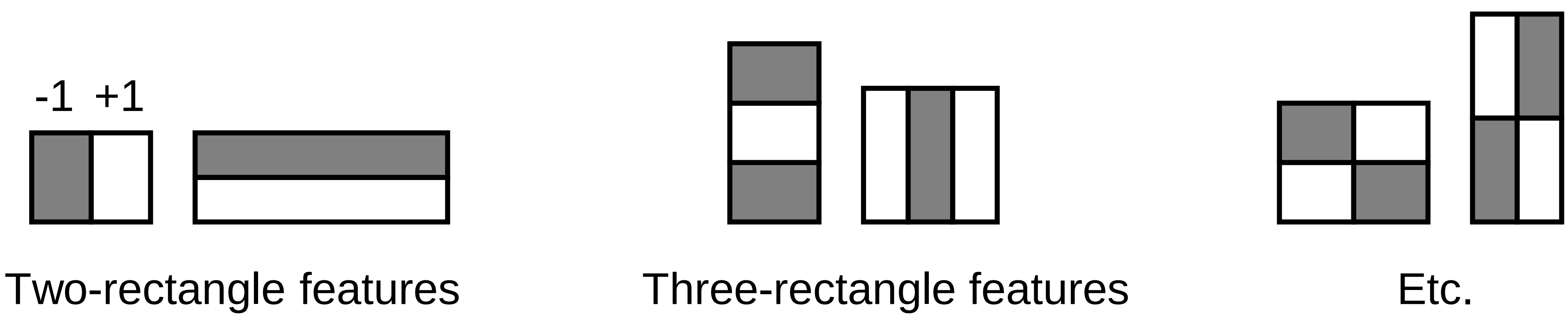
Cascade for Fast Detection



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

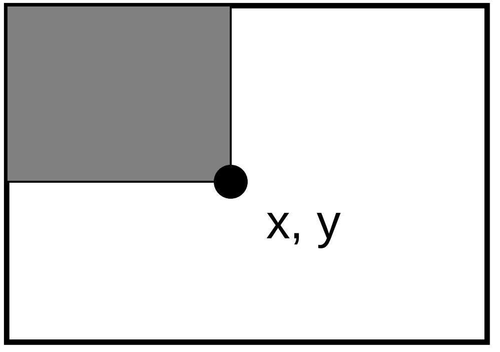
Features that are fast to compute

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

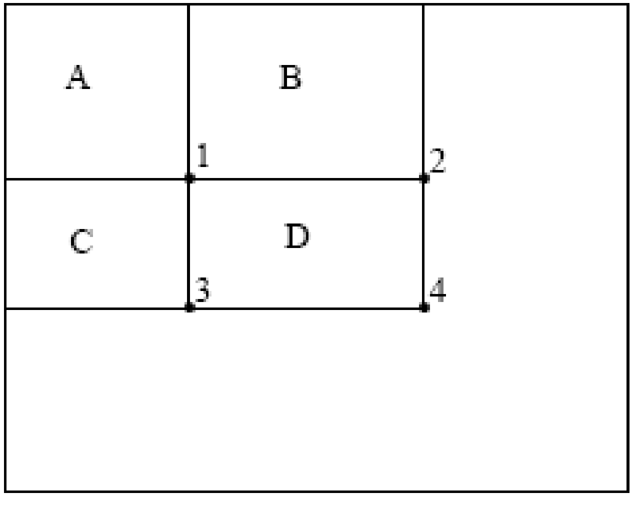


Integral Images

- `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

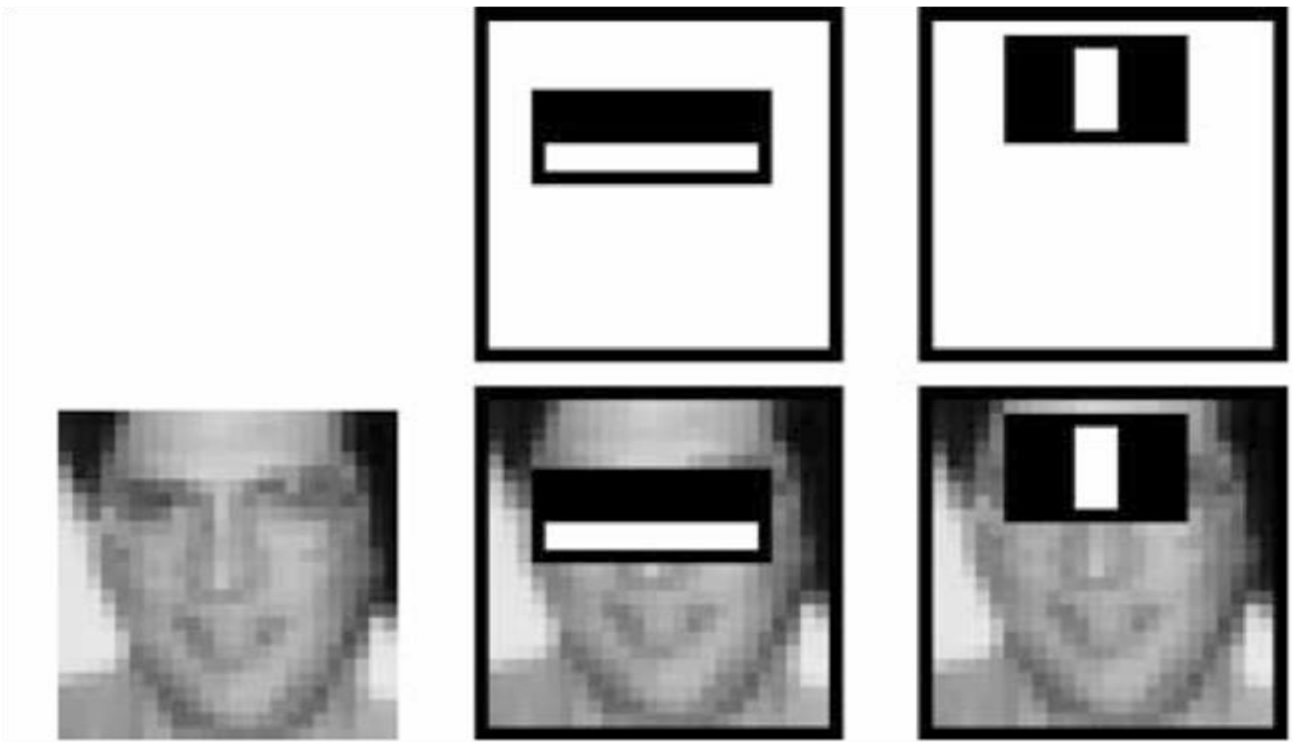
Adaboost

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

Top 2 selected features

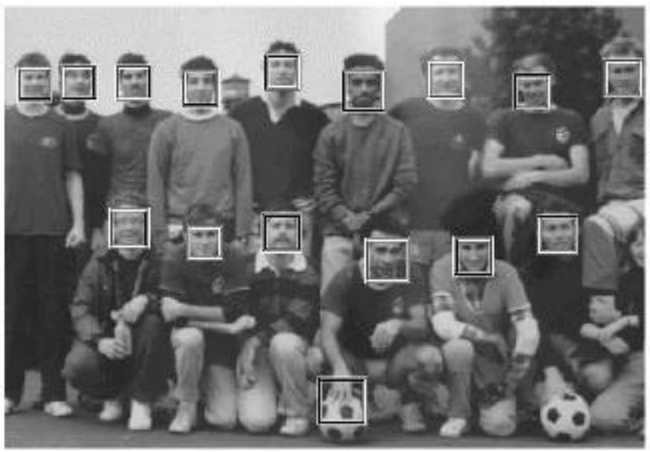


Viola-Jones details

- 38 stages with 1, 10, 25, 50 … features
  - 6061 total used out of 180K candidates
  - 10 features evaluated on average
- Training Examples
  - 4916 positive examples
  - 10000 negative examples collected after each stage
- Scanning
  - Scale detector rather than image
  - Scale steps = 1.25 (factor between two consecutive scales)
  - Translation 1*scale (# pixels between two consecutive windows)
- Non-max suppression: average coordinates of overlapping boxes
- Train 3 classifiers and take vote

Viola Jones Results

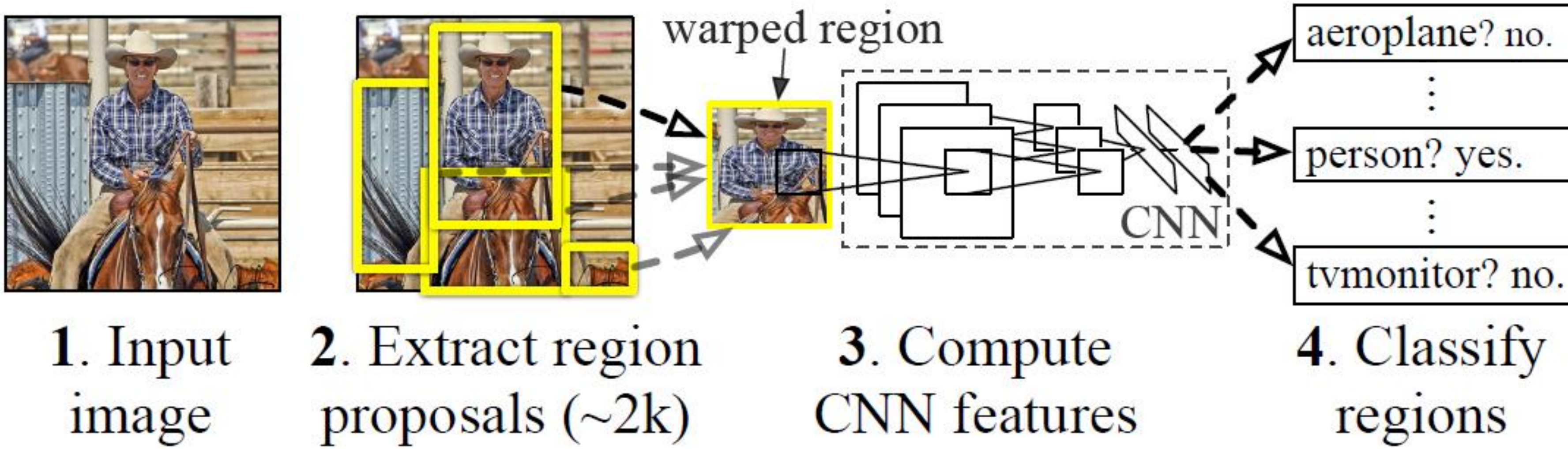
Speed = 15 FPS (in 2001)



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

R-CNN (Girshick et al. CVPR 2014)



- Replace sliding windows with “selective search” region proposals (Uijilings et al. IJCV 2013)
- Extract rectangles around regions and resize to 227x227
- Extract features with fine-tuned CNN (that was initialized with network trained on ImageNet before training)
- Classify last layer of network features with SVM

http://arxiv.org/pdf/1311.2524.pdf

Sliding window vs. region proposals

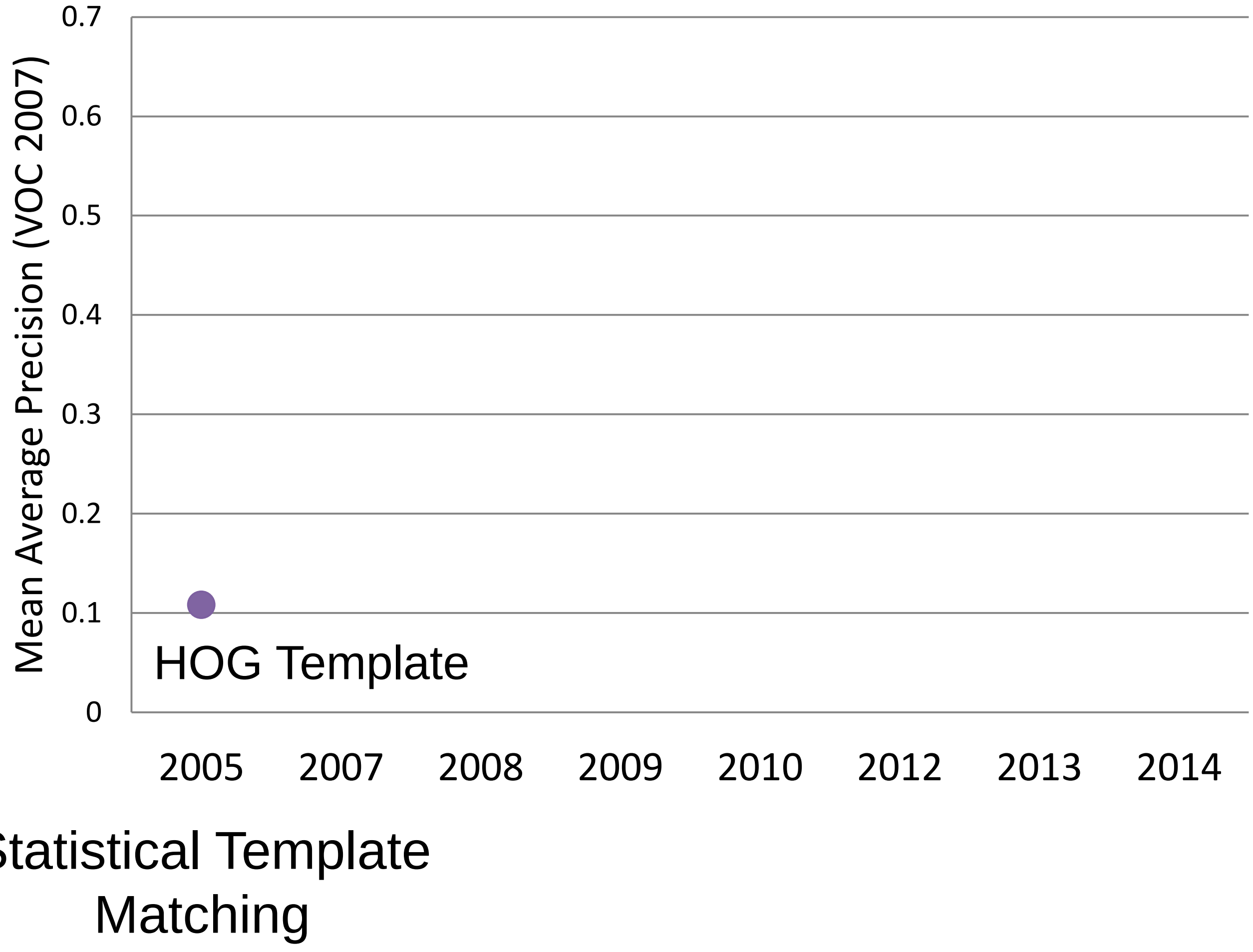
Sliding window

- Comprehensive search over position, scale (sometimes aspect, though expensive)
- Typically 100K candidates
- Simple
- Speed boost through convolution often possible
- Repeatable
- Even with many candidates, may not be a good fit to object

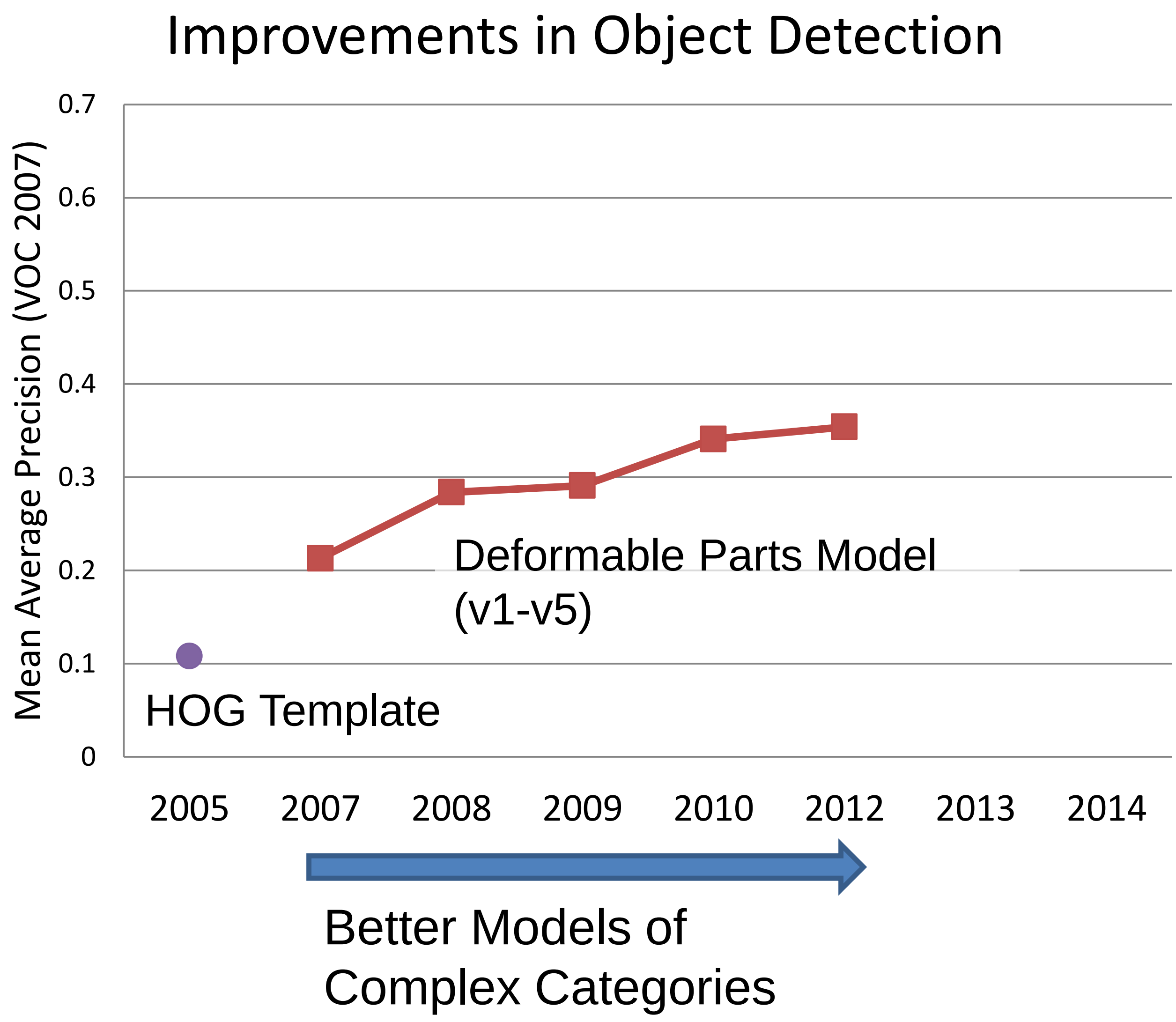
Region proposals

- Search over regions guided by image contours/patterns with varying aspect/size
- Typically 2-10K candidates
- Random (not repeatable)
- Requires a preprocess (currently 1-5s)
- Often requires resizing patch to fit fixed size
- More likely to provide candidates with very good object fit

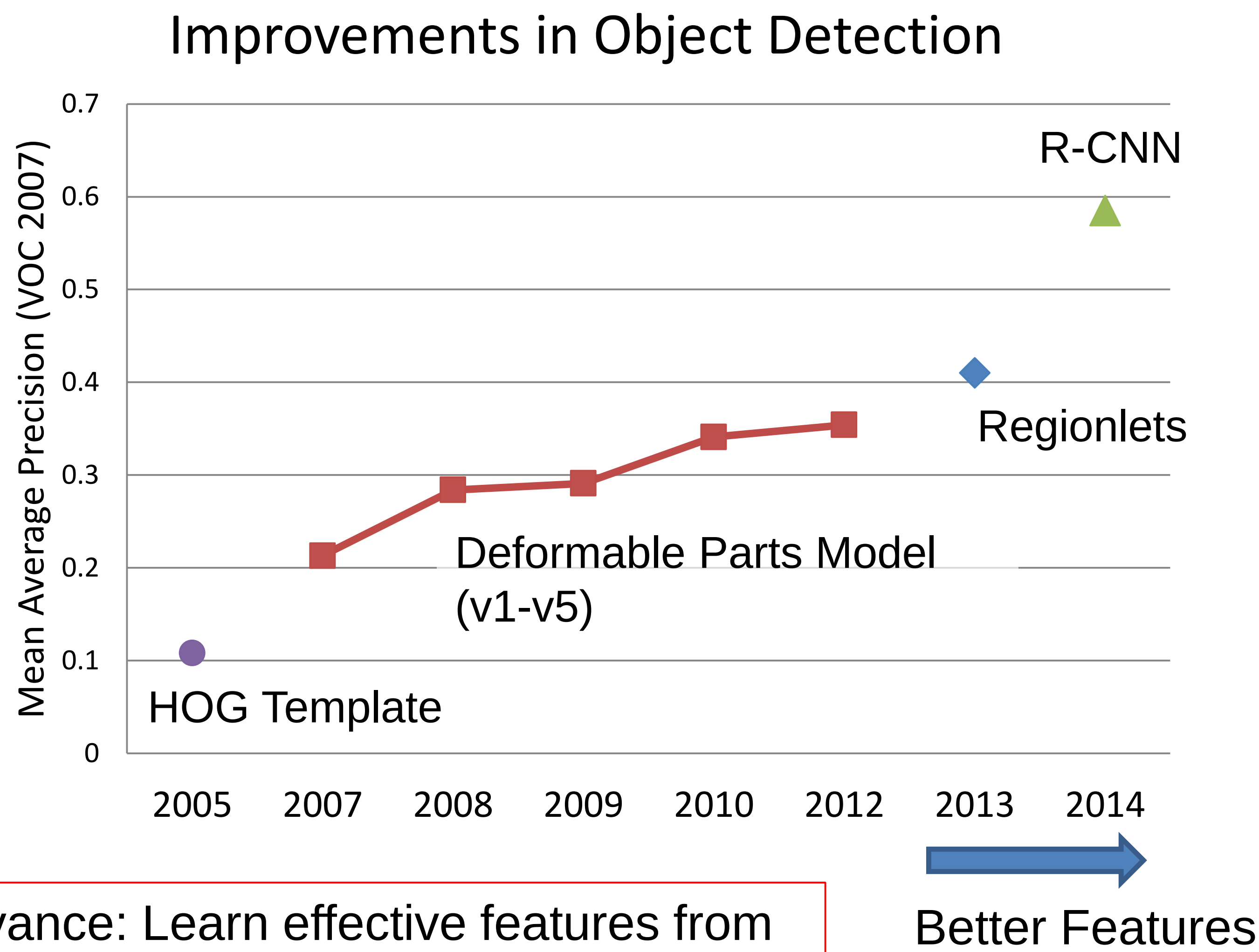
Improvements in Object Detection



HOG: Dalal-Triggs 2005



HOG: Dalal-Triggs 2005 DPM: Felzenszwalb et al. 2008-2012

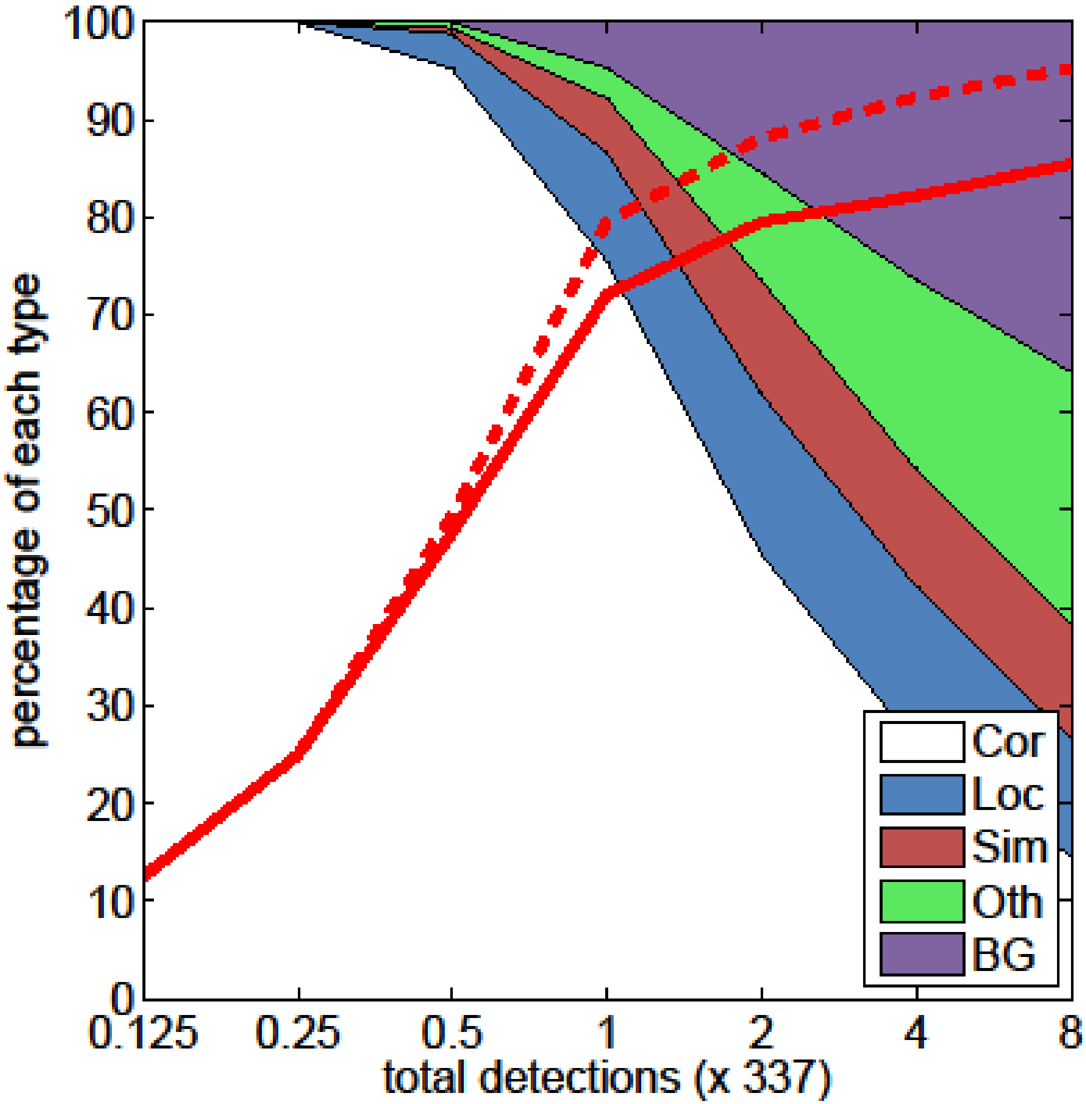


Key Advance: Learn effective features from massive amounts of labeled data *and* adapt to new tasks with less data

HOG: Dalal-Triggs 2005 DPM: Felzenszwalb et al. 2008-2012 Regionlets: Wang et al. 2013 R-CNN: Girshick et al. 2014

Mistakes are often reasonable

Bicycle: AP = 0.73



Confident Mistakes

bicycle (loc): ov=0.44 1−r=0.70

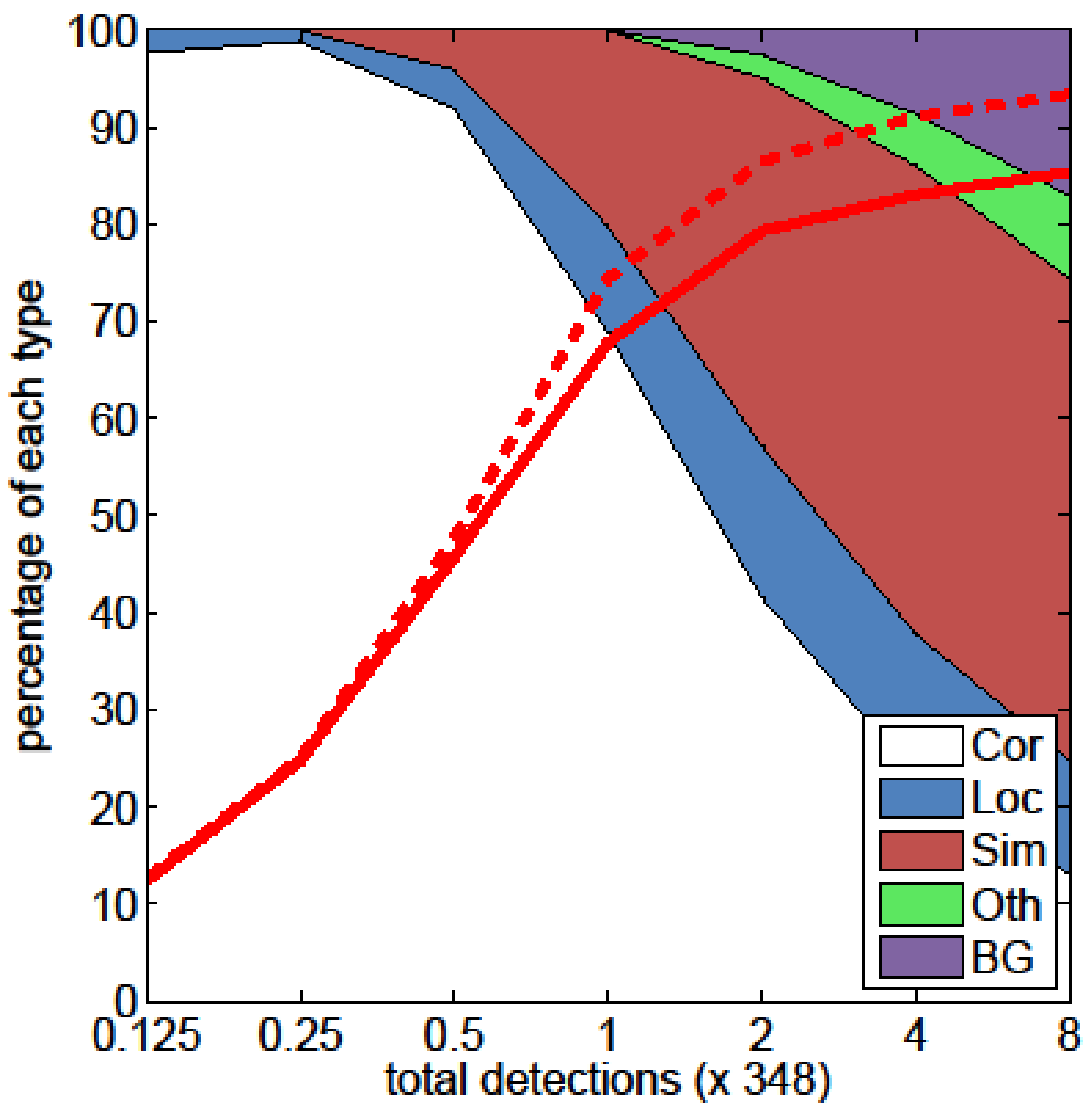
bicycle (sim): ov=0.00 1−r=0.56

bicycle (bg): ov=0.00 1−r=0.47

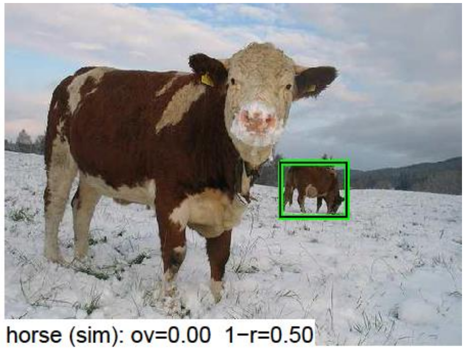
R-CNN results

Mistakes are often reasonable

Horse: AP = 0.69

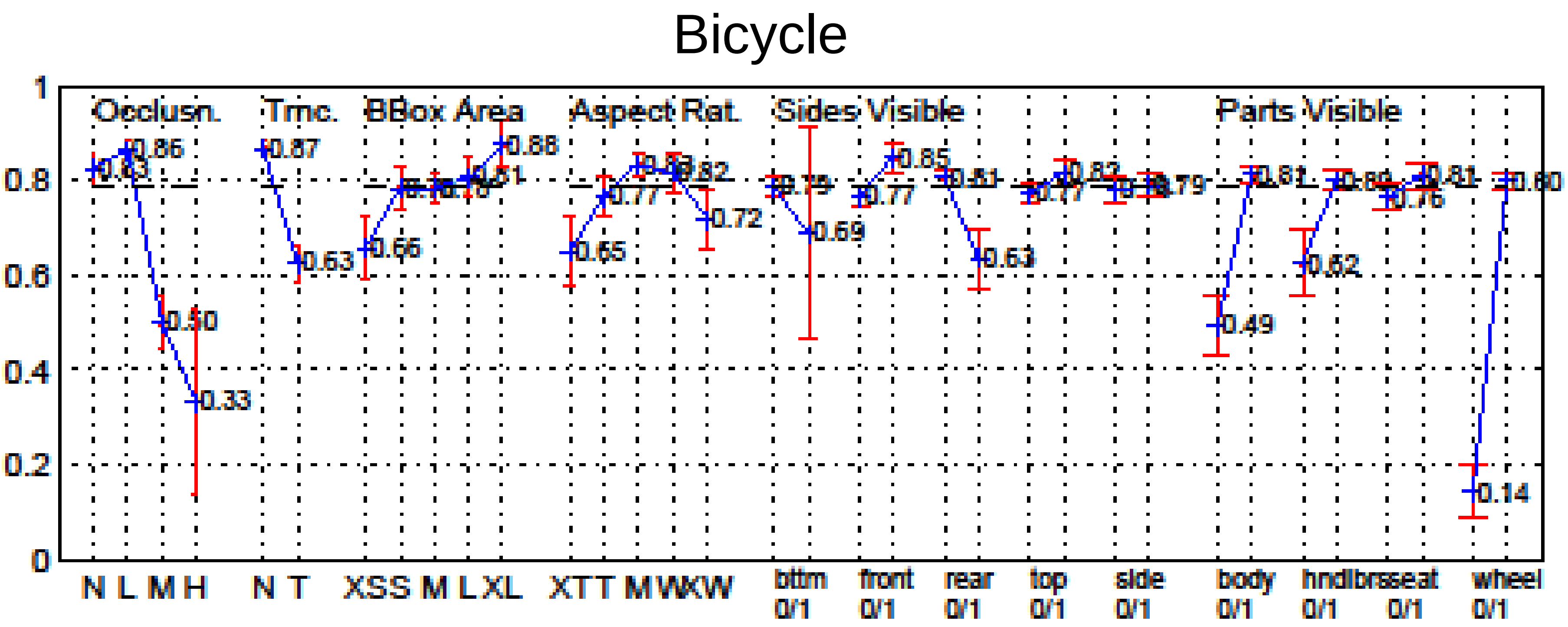


Confident Mistakes



R-CNN results

Misses are often predictable



Small objects, distinctive parts absent
or occluded, unusual views

R-CNN results

Strengths and Weaknesses of Statistical Template Approach

Strengths

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

Weaknesses

- Sliding window has difficulty with deformable objects (proposals works with flexible features works better)
- Not robust to occlusion
- Requires lots of training data

Tricks of the trade

- Details in feature computation really matter
  - E.g., normalization in Dalal-Triggs improves detection rate by 27% at fixed false positive rate
- Template size
  - Typical choice for sliding window is size of smallest detectable object
  - For CNNs, typically based on what pretrained features are available
- "Jittering" to create synthetic positive examples
  - Create slightly rotated, translated, scaled, mirrored versions as extra positive examples
- Bootstrapping to get hard negative examples
  1. Randomly sample negative examples
  2. Train detector
  3. Sample negative examples that score > -1
  4. Repeat until all high-scoring negative examples fit in memory

Influential Works in Detection

- Sung-Poggio (1994, 1998) : ~2100 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) : ~4200
  - “Parts” at fixed position, non-maxima suppression, simple cascade, rotation, pretty good accuracy, fast
- Schneiderman-Kanade (1998-2000,2004) : ~2250
  - Careful feature/classifier engineering, excellent results, cascade
- Viola-Jones (2001, 2004) : ~20,000
  - Haar-like features, Adaboost as feature selection, hyper-cascade, very fast, easy to implement
- Dalal-Triggs (2005) : ~11000
  - Careful feature engineering, excellent results, HOG feature, online code
- Felzenszwalb-Huttenlocher (2000): ~1600
  - Efficient way to solve part-based detectors
- Felzenszwalb-McAllester-Ramanan (2008,2010)? ~4000
  - Excellent template/parts-based blend
- Girshick-Donahue-Darrell-Malik (2014) ~300
  - Region proposals + fine-tuned CNN features (marks significant advance in accuracy over hog-based methods)

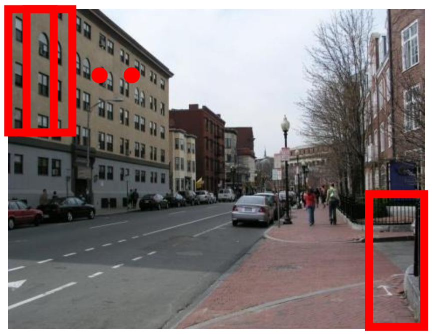
Fails in commercial face detection

- [Things iPhoto thinks are faces]()

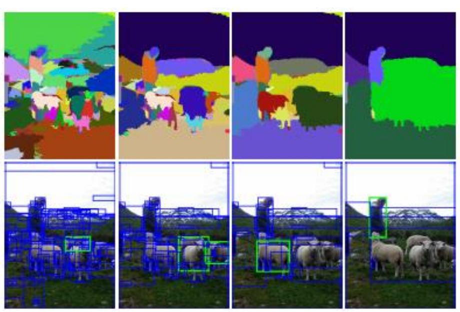
http://www.oddee.com/item_98248.aspx

Summary: statistical templates

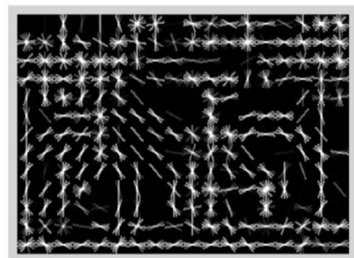
Propose Window → Extract Features → Classify → Post-process



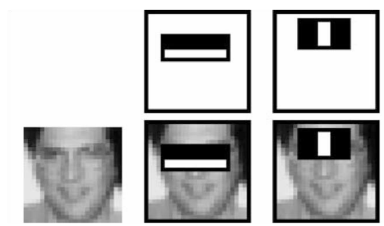
Sliding window: scan image pyramid



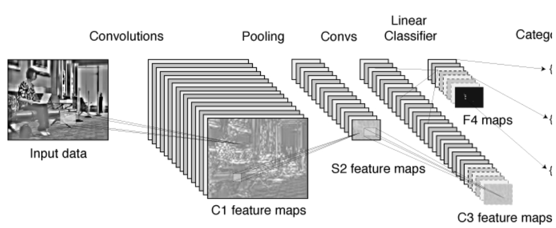
Region proposals: edge/region-based, resize to fixed window



HOG



Fast randomized features



CNN features

SVM

Boosted stubs

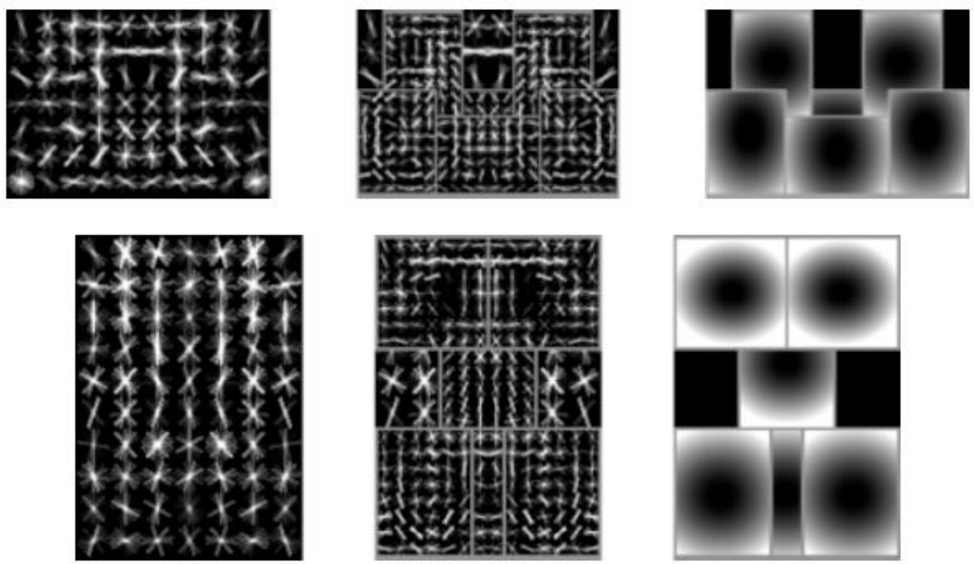
Neural network

Non-max suppression

Segment or refine localization

Next class

- Part-based models and pose estimation



root filters coarse resolution

part filters finer resolution

deformation models