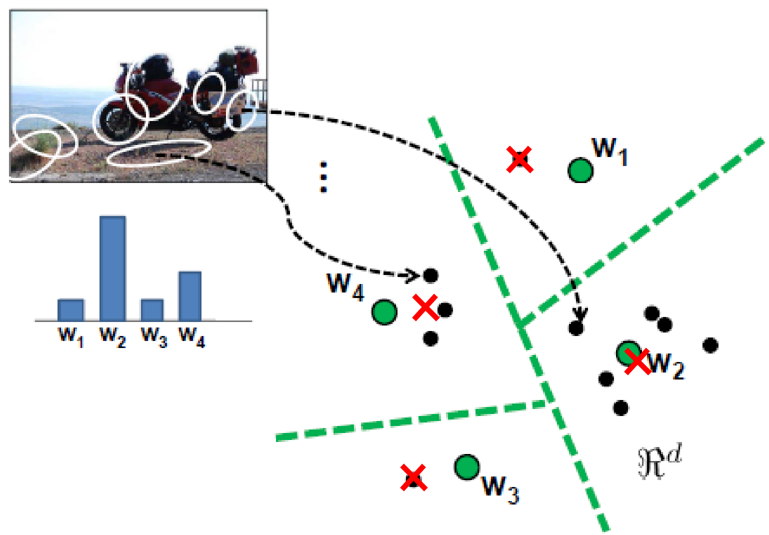
Recap: Advanced Feature Encoding

Bag of Visual Words is only about **counting** the number of local descriptors assigned to each Voronoi region (0^{th} order statistics)

Why not including **other statistics**? For instance:

- mean of local descriptors (first order statistics) ×



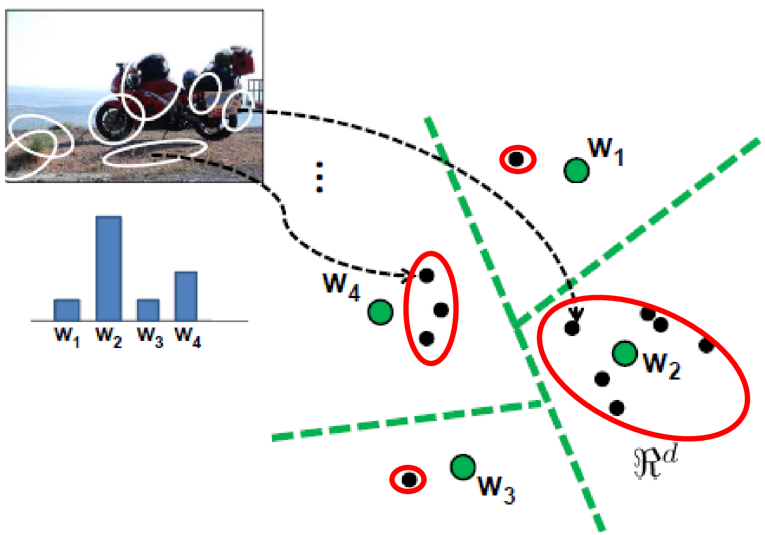
http://www.cs.utexas.edu/~grauman/courses/fall2009/papers/bag_of_visual_words.pdf

Recap: Advanced Feature Encoding

Bag of Visual Words is only about **counting** the number of local descriptors assigned to each Voronoi region (0th order statistics)

Why not including **other statistics**? For instance:

- mean of local descriptors (first order statistics)
- (co)variance of local descriptors



http://www.cs.utexas.edu/~grauman/courses/fall2009/papers/bag_of_visual_words.pdf

Recap: Advanced Feature Encoding

- We’ve looked at methods to better characterize the distribution of visual words in an image:
  - Soft assignment (a.k.a. Kernel Codebook)
  - VLAD
  - Fisher Vector

- Mixtures of Gaussians could be thought of as a soft form of kmeans which can better model the data distribution.

Modern Object Detection

Computer Vision

CS 143

Brown

James Hays

Many slides from Derek Hoiem

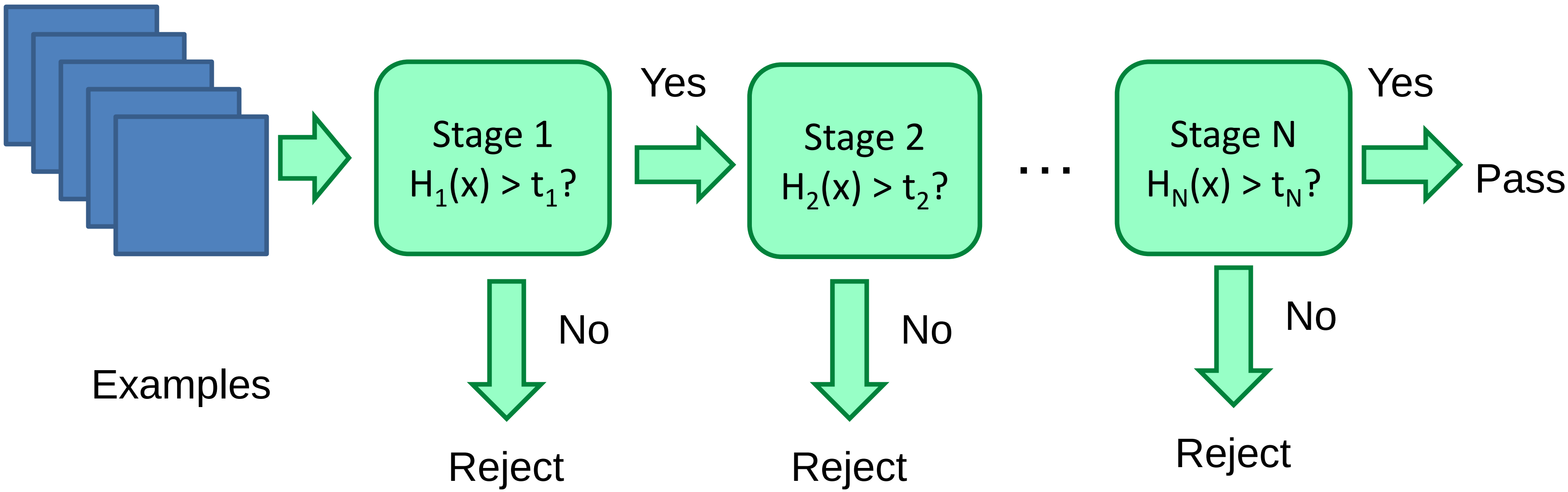
Recap: Viola-Jones sliding window detector

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).

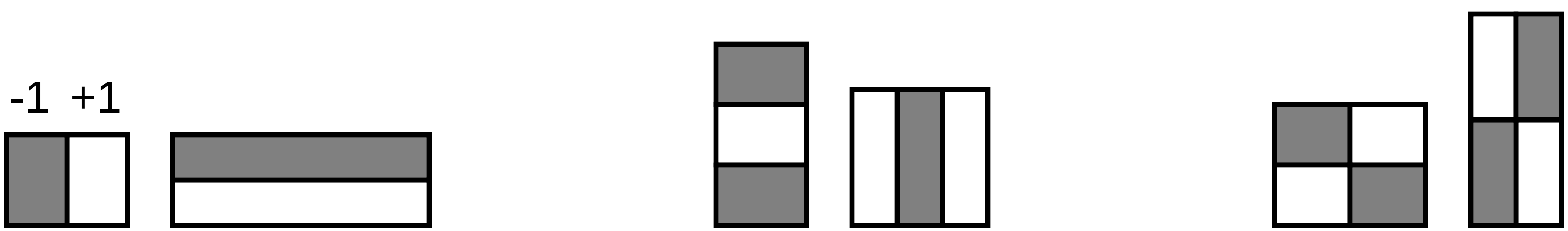
Cascade for Fast Detection



- 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



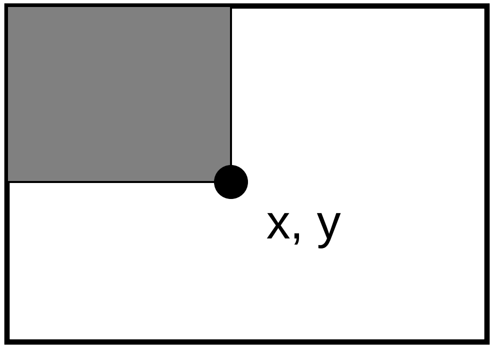
Two-rectangle features

Three-rectangle features

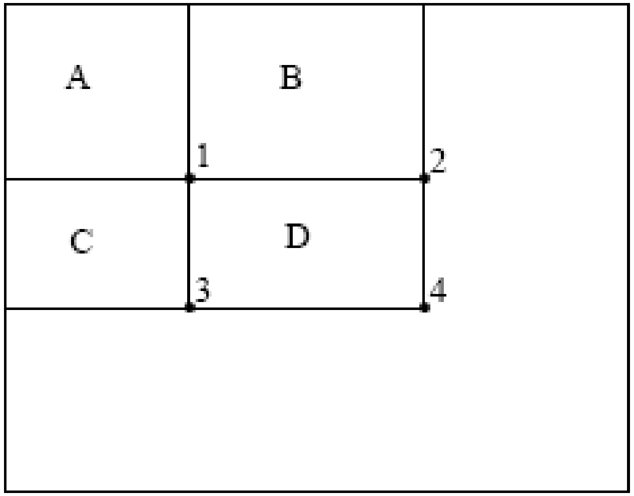
Etc.

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

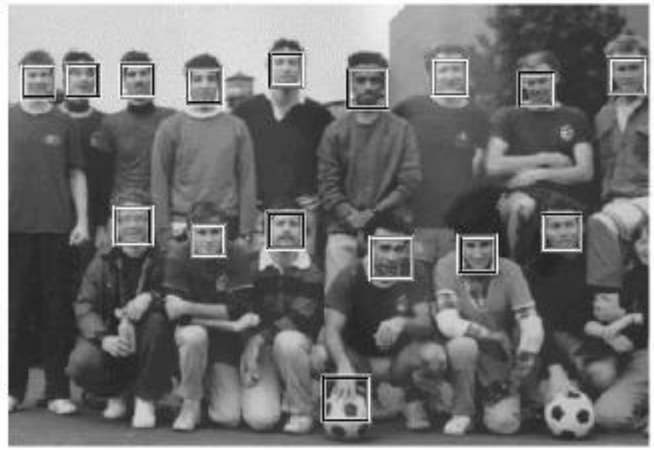
- Create a large pool of features (180K)
- Select features that are discriminative and work well together
  - "Weak learner" = feature + threshold + parity

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

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

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%)	-	-

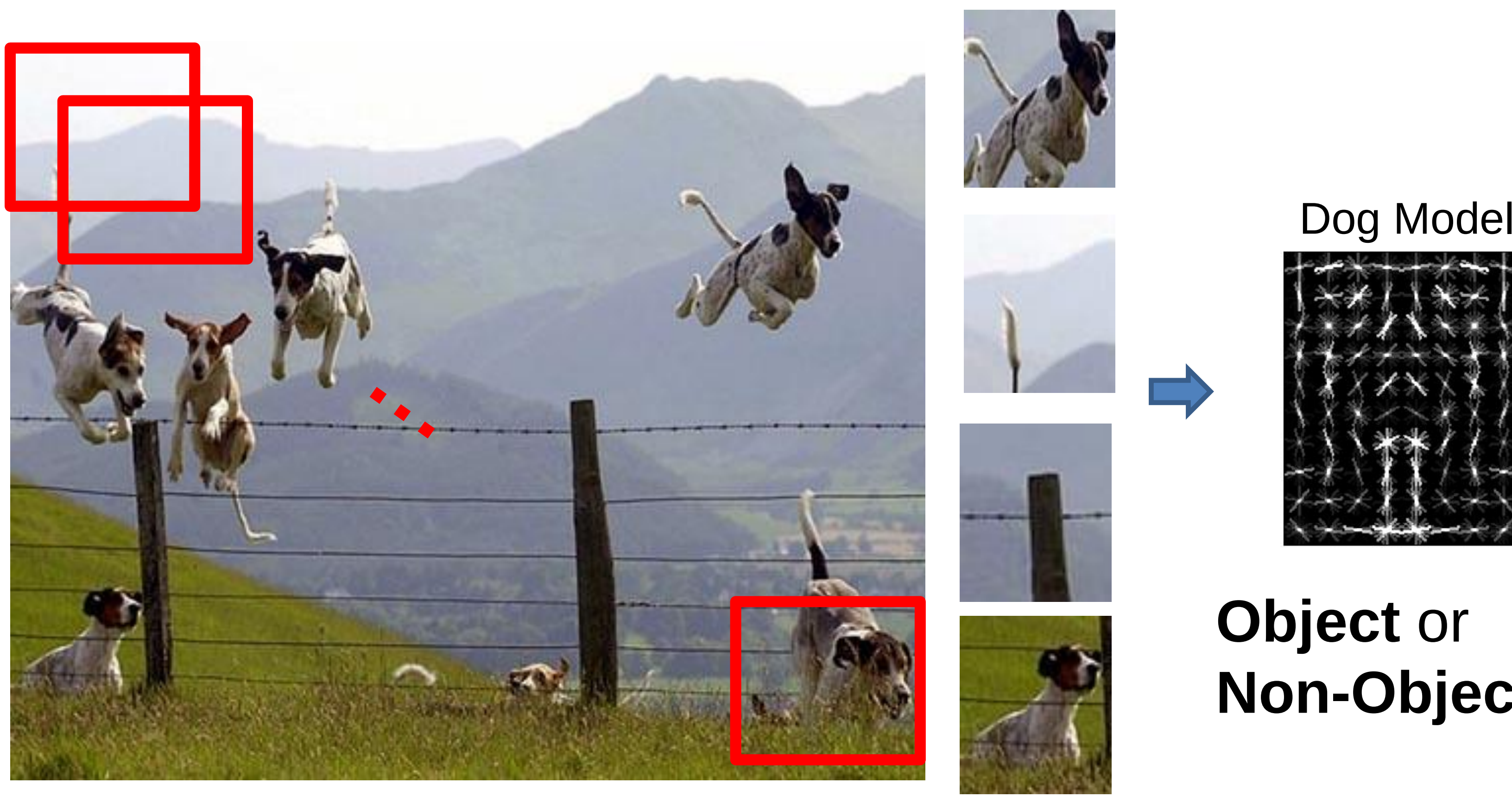
MIT + CMU face dataset

Today's class: Modern Object Category Detection

- Recap of Viola Jones
- Overview of object category detection
- Statistical template matching with sliding window detector
  - Dalal-Triggs pedestrian detector

Object Category Detection

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



Object or
Non-Object?

Challenges in modeling the object class

Illumination

Object pose

Clutter

Occlusions

Intra-class appearance

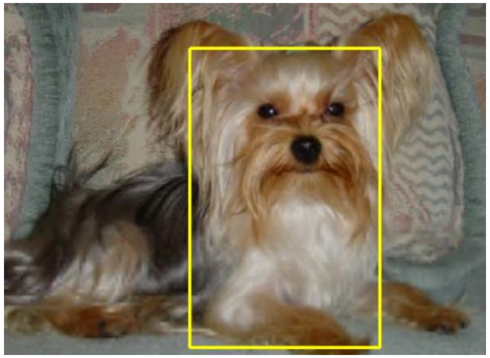
Viewpoint

Slide from K. Grauman, B. Leibe

Challenges in modeling the non-object class

True Detections

Bad Localization

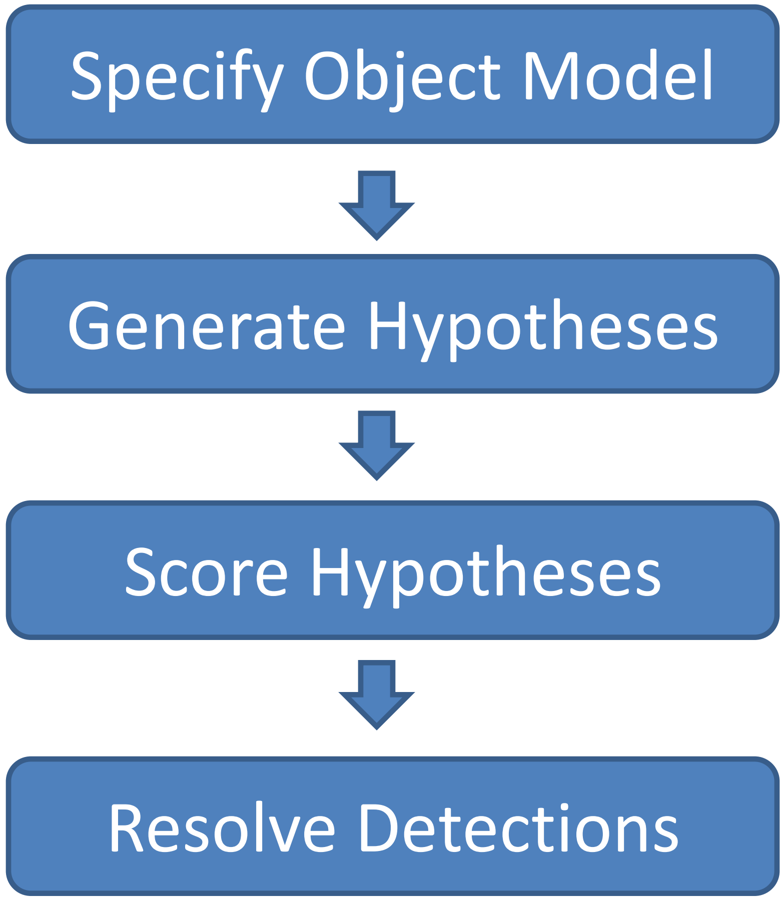


Confused with Similar Object

Misc. Background

Confused with Dissimilar Objects

General Process of Object Recognition

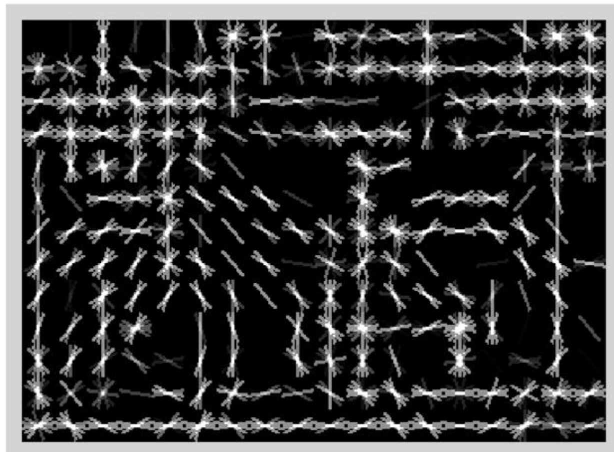


What are the object parameters?

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

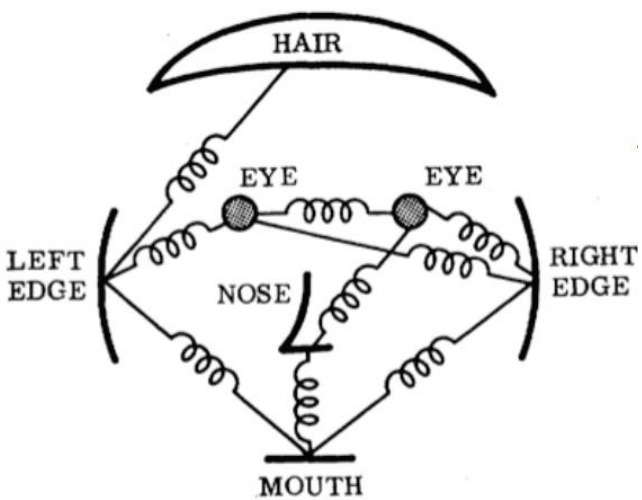


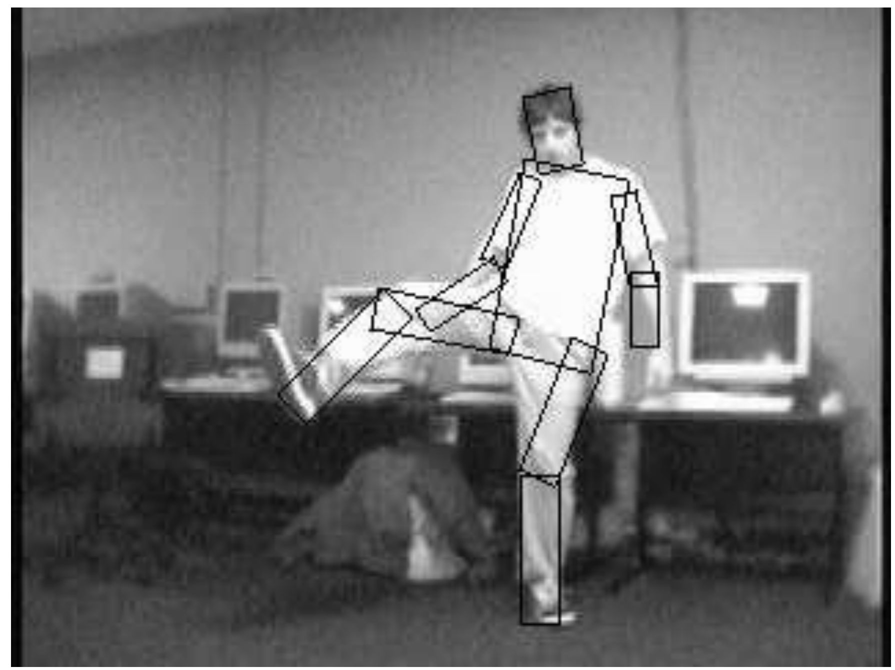
Template Visualization

Images from Felzenszwalb

Specifying an object model

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



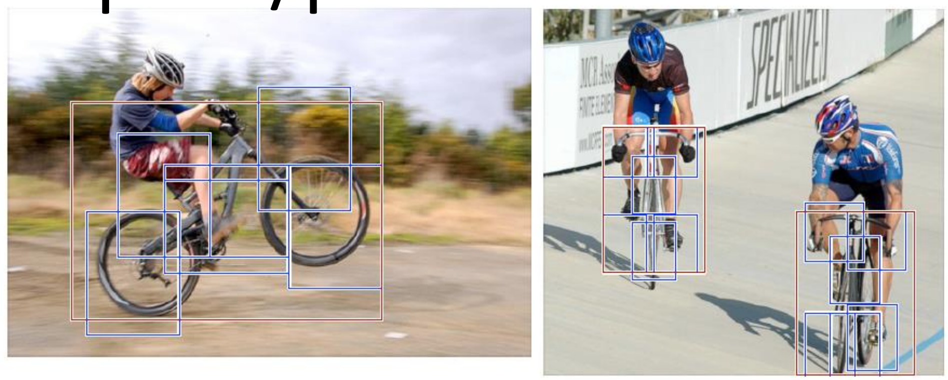


Images from Felzenszwalb

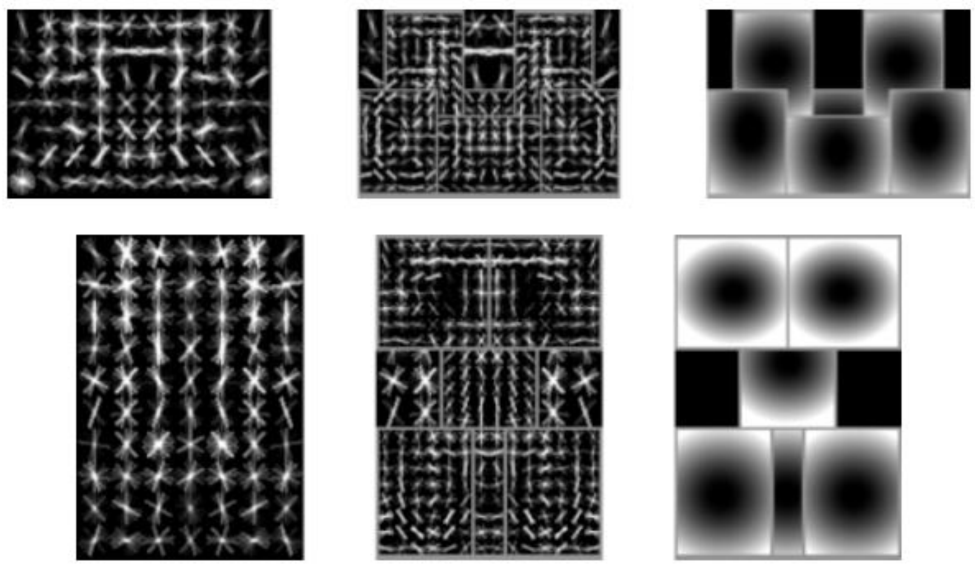
Specifying an object model

3. Hybrid template/parts model

Detections



Template Visualization



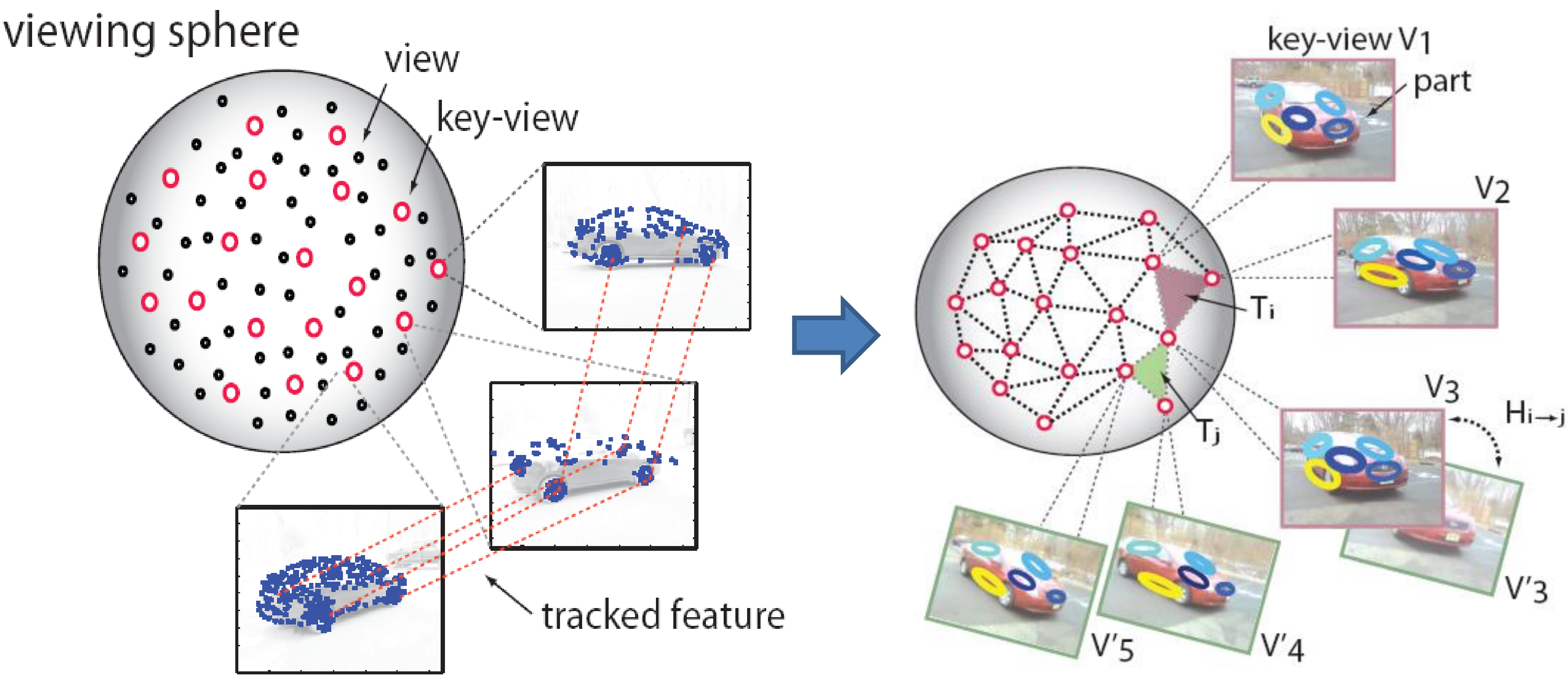
root filters coarse resolution	part filters finer resolution	deformation models

Felzenszwalb et al. 2008

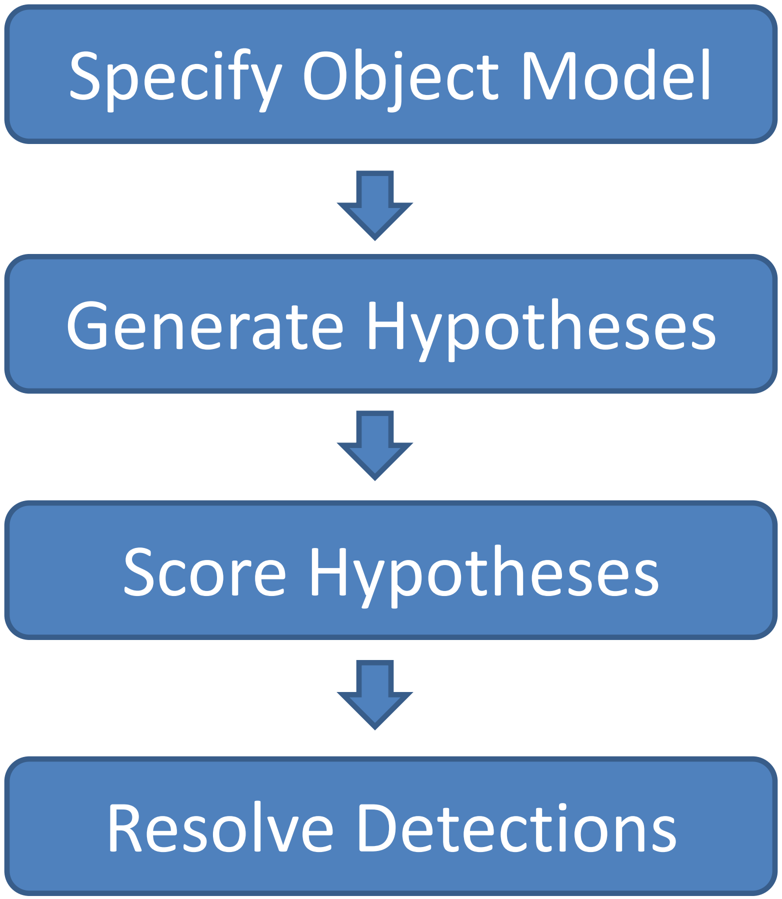
Specifying an object model

4. 3D-ish model

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



General Process of Object Recognition



Propose an alignment of the model to the image

Generating hypotheses

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

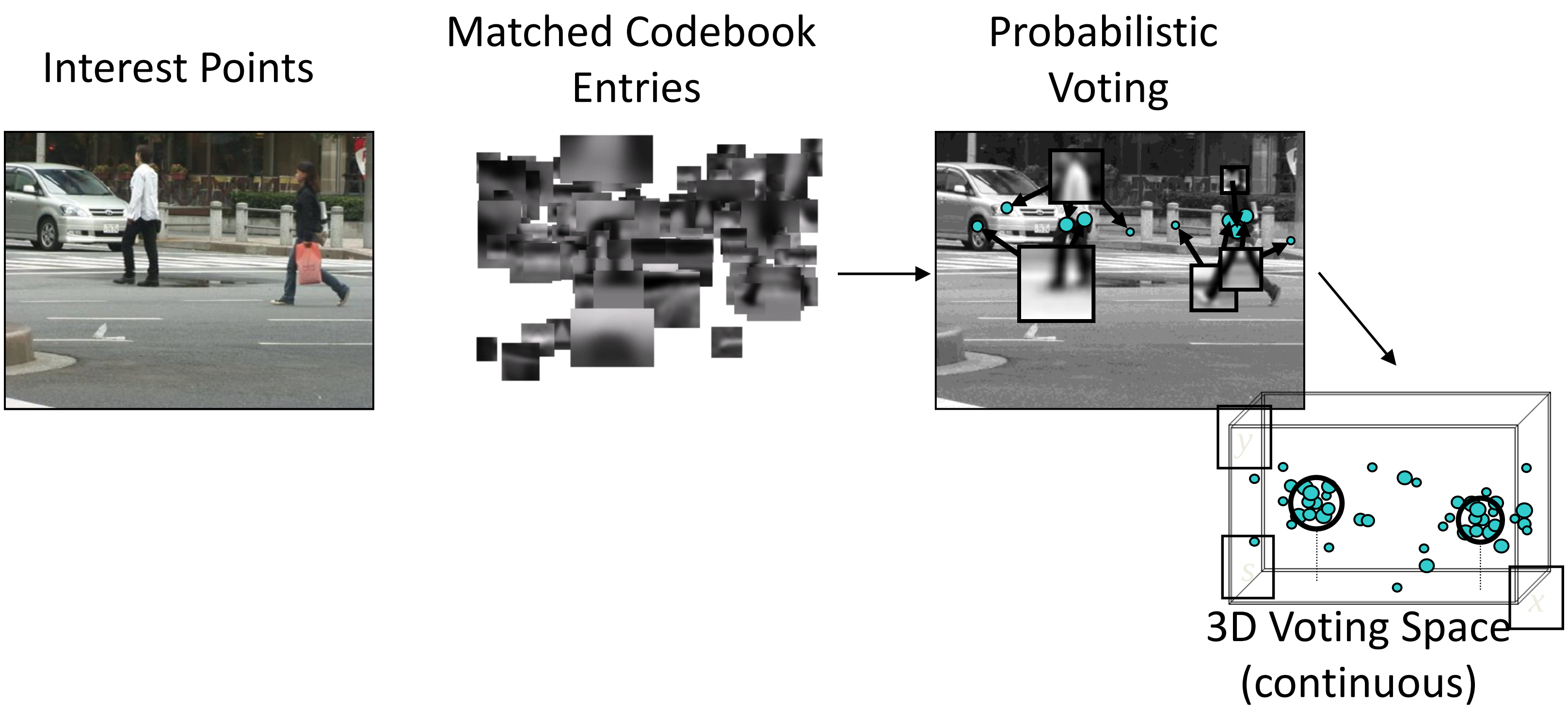
Generating hypotheses

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

Note – Template did not change size

Generating hypotheses

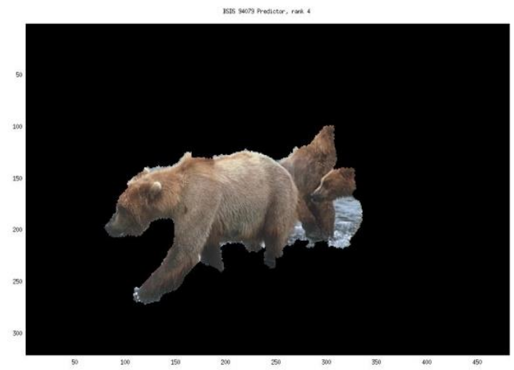
2. Voting from patches/keypoints

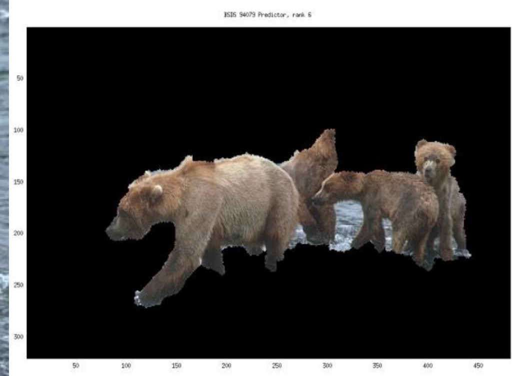


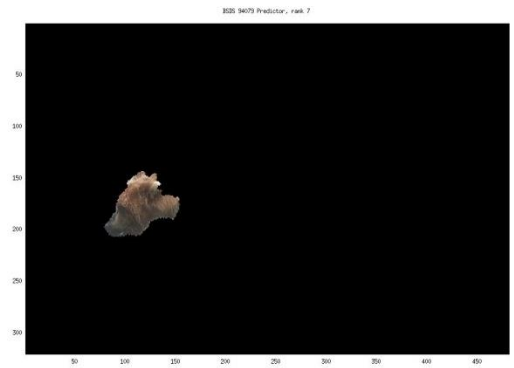
ISM model by Leibe et al.

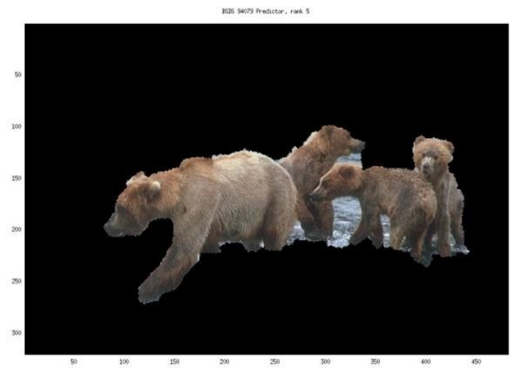
Generating hypotheses

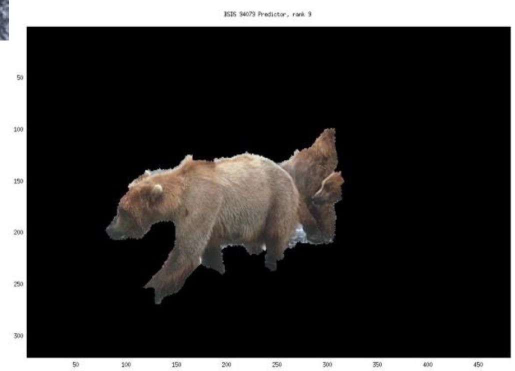
3. Region-based proposal











Endres Hoiem 2010

General Process of Object Recognition

Specify Object Model

↓

Generate Hypotheses

↓

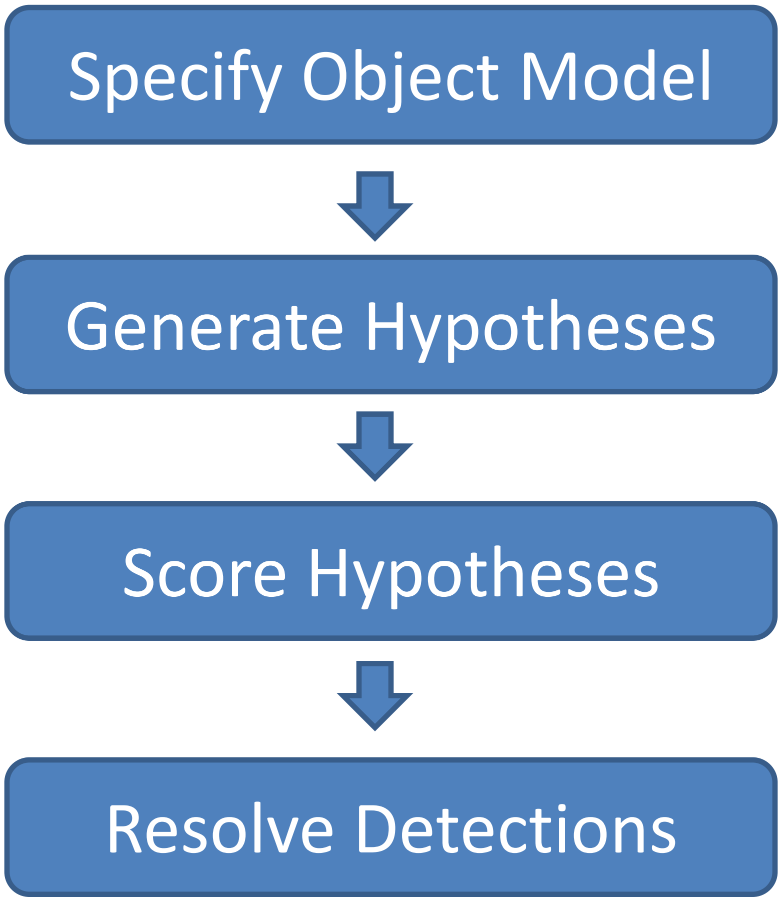
Score Hypotheses

↓

Resolve Detections

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

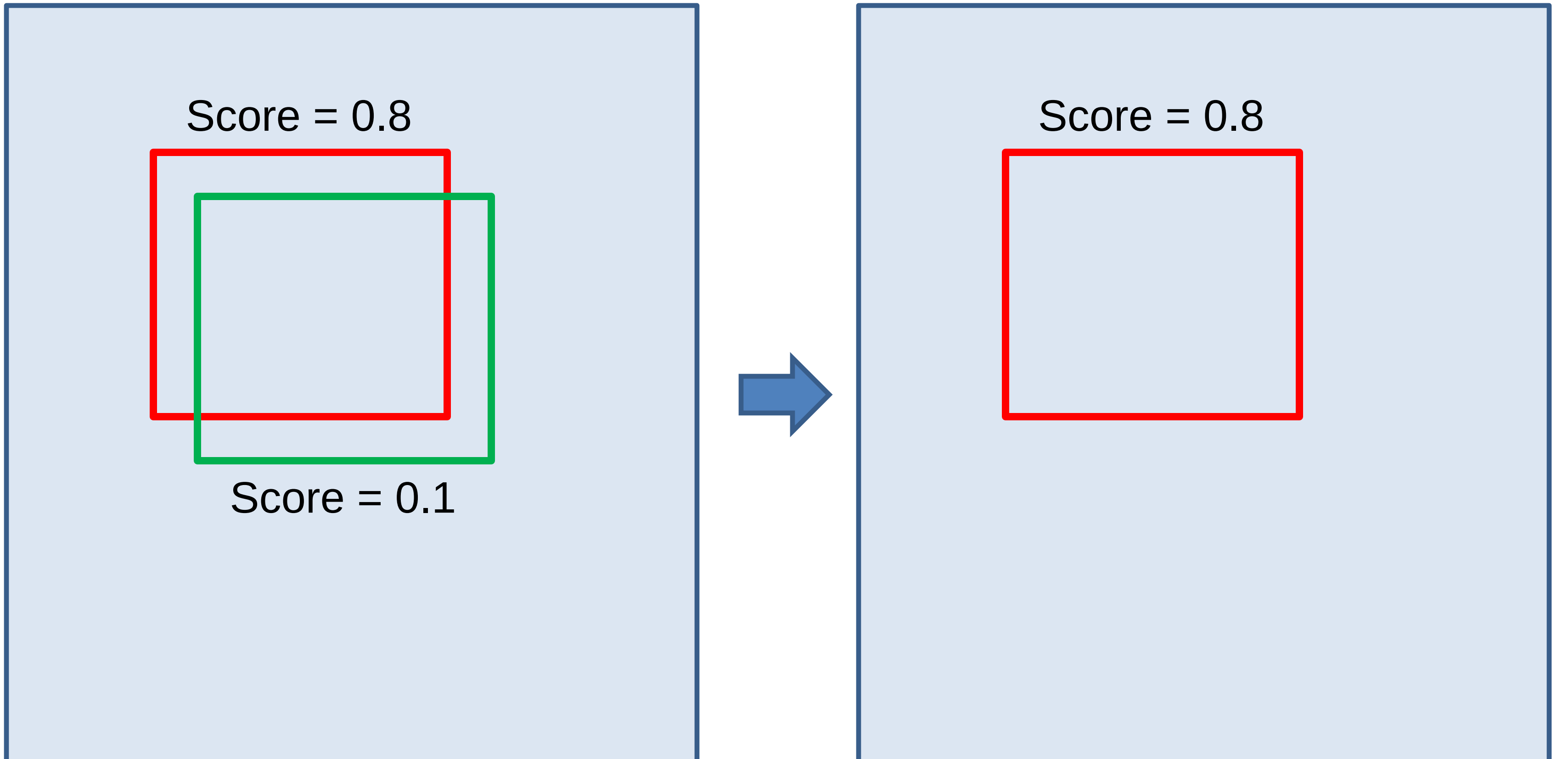
General Process of Object Recognition



Rescore each proposed
object based on whole set

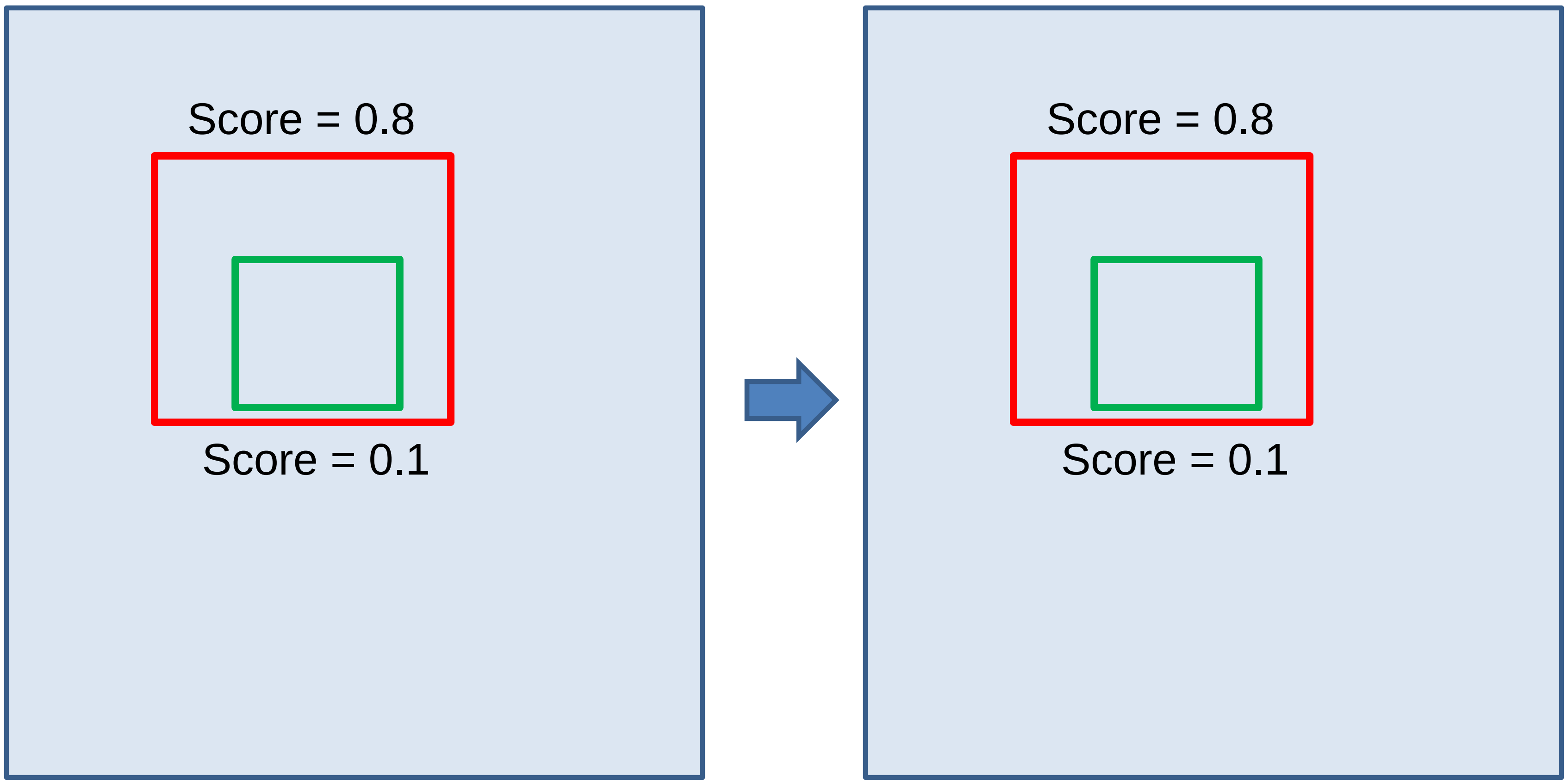
Resolving detection scores

1. Non-max suppression



Resolving detection scores

1. Non-max suppression

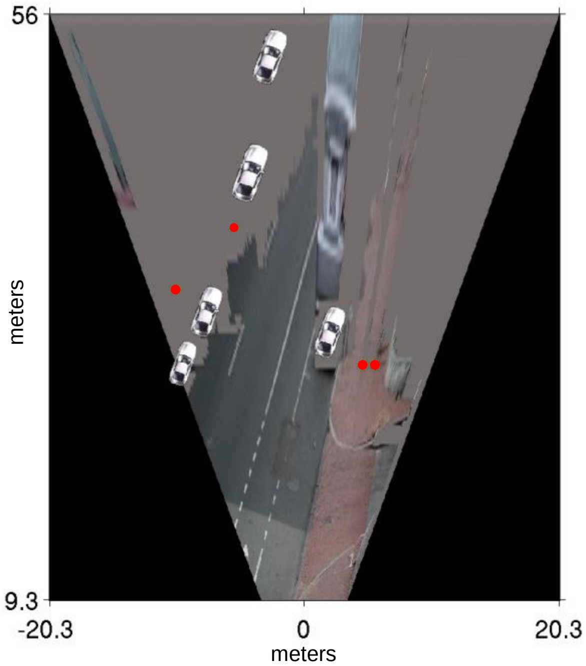


"Overlap" score is below some threshold

Resolving detection scores

2. Context/reasoning

(g) Car Detections: Local (h) Ped Detections: Local



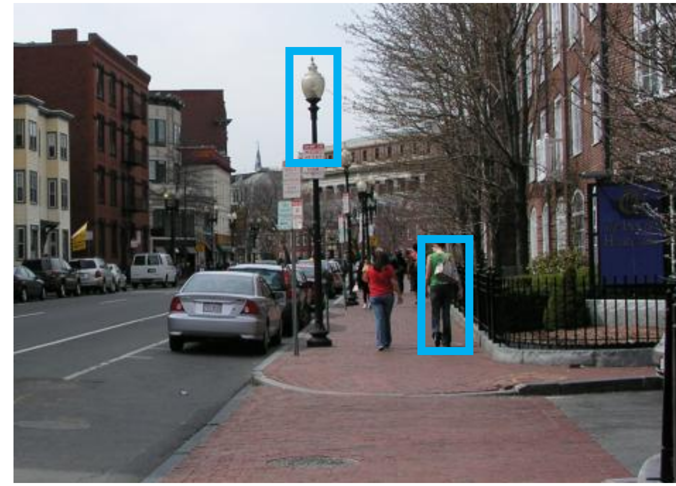
Hoiem et al. 2006

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?



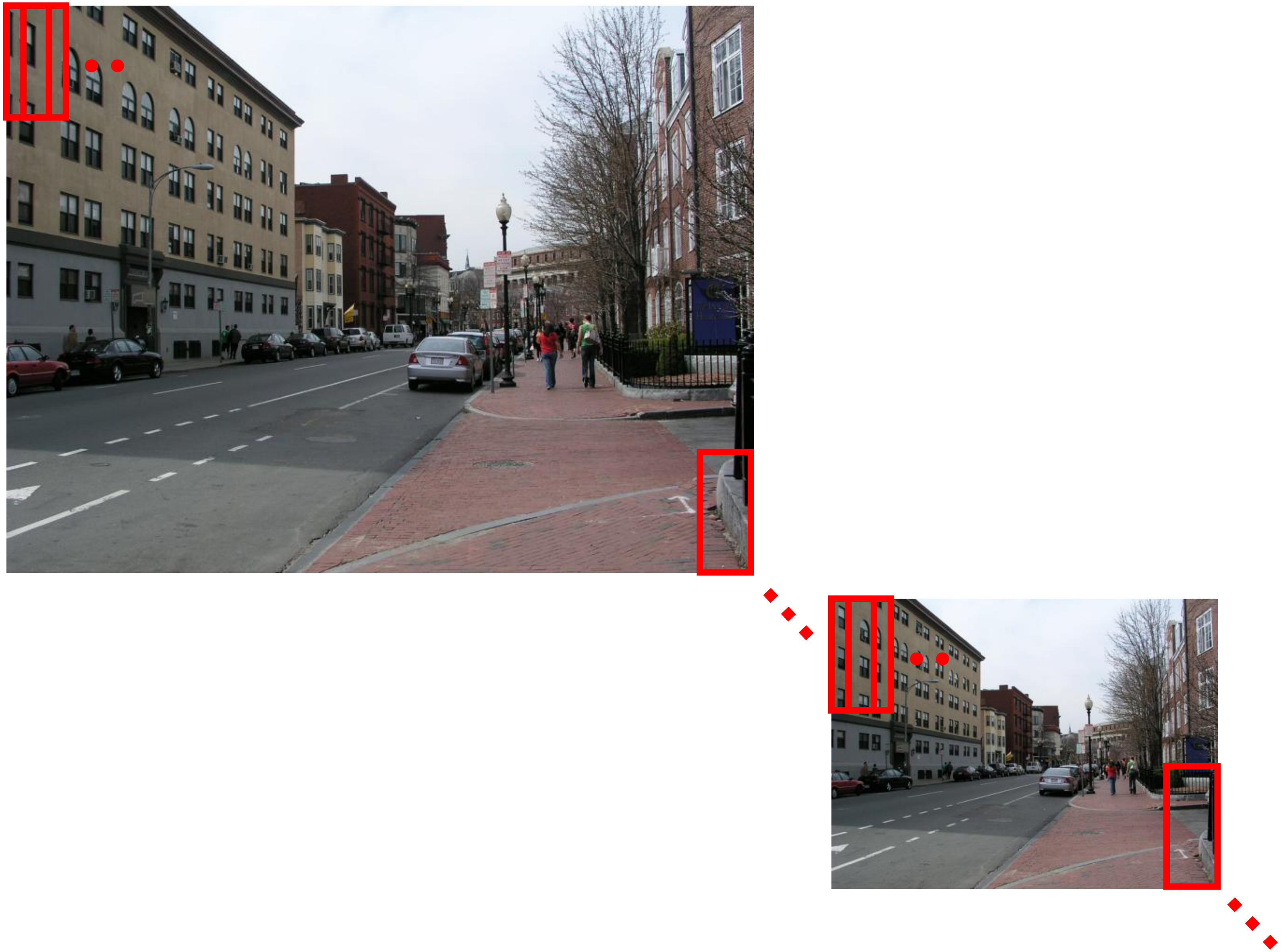
2. Compare
   - Compute similarity to an example object or to a summary representation
   - Which differences in appearance are important?

Aligned Possible Objects

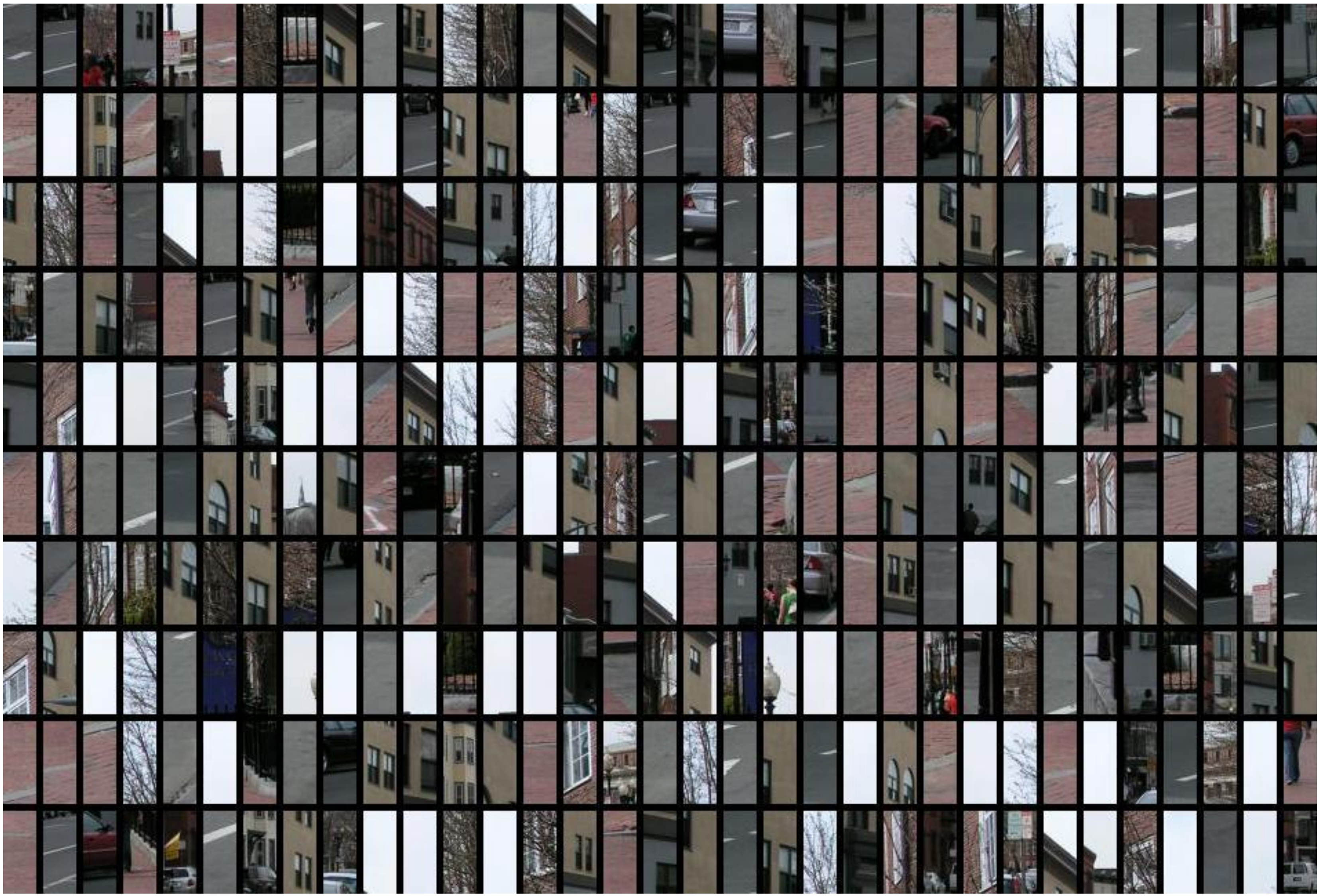
Exemplar

Summary

Sliding window: a simple alignment solution



Each window is separately classified

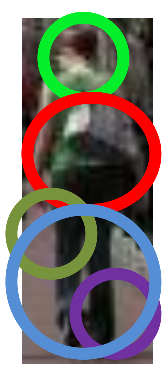


Statistical Template

- 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

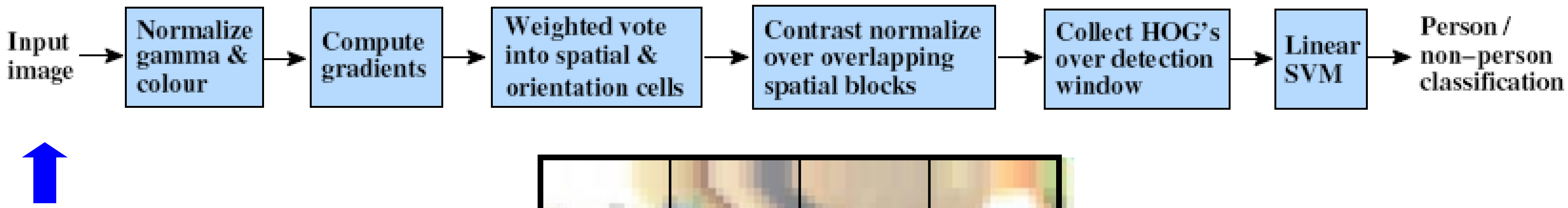
Design challenges

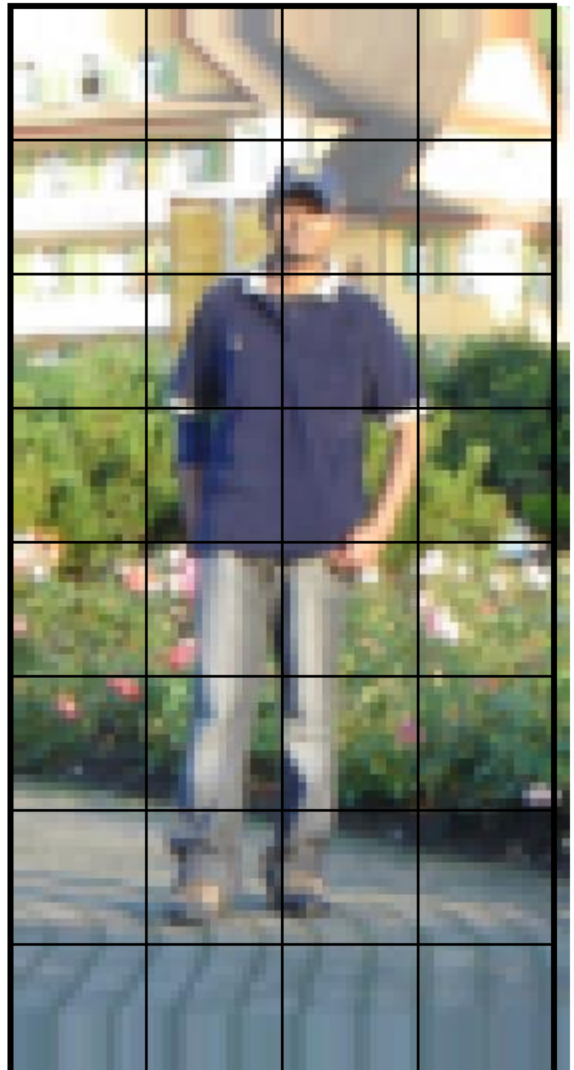
- How to efficiently search for likely objects
  - Even simple models require searching hundreds of thousands of positions and scales
- Feature design and scoring
  - How should appearance be modeled? What features correspond to the object?
- How to deal with different viewpoints?
  - Often train different 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

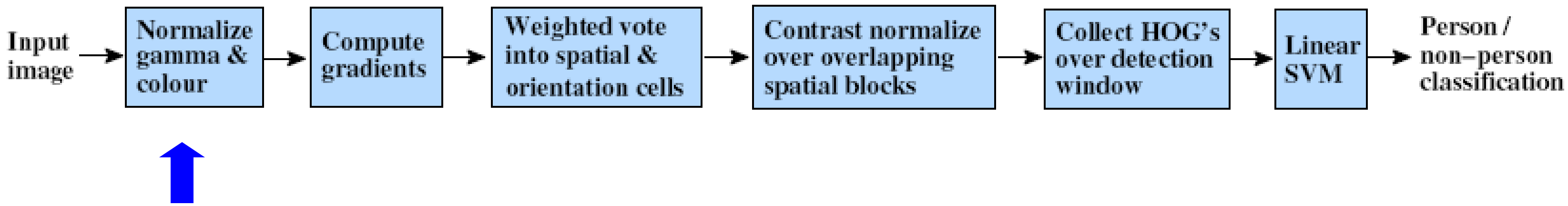
Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05





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

    Very slightly better performance vs. no adjustment
  - Log

Input image → Normalize gamma & colour → Compute gradients → Weighted vote into spatial & orientation cells → Contrast normalize over overlapping spatial blocks → Collect HOG's over detection window → Linear SVM → Person / non–person classification

Outperforms

-1	0	1

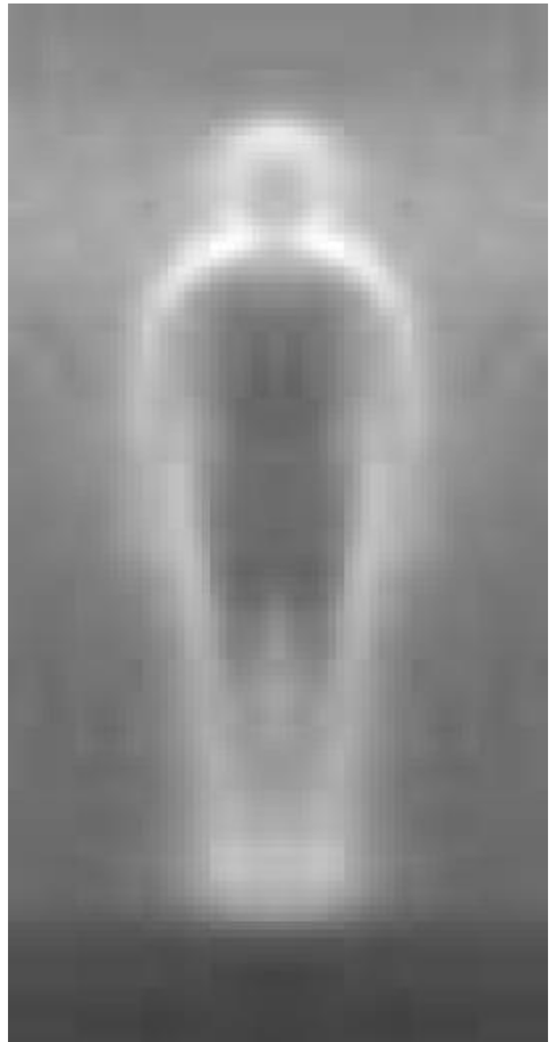
centered

-1	1

uncentered

1	-8	0	8	-1

cubic-corrected



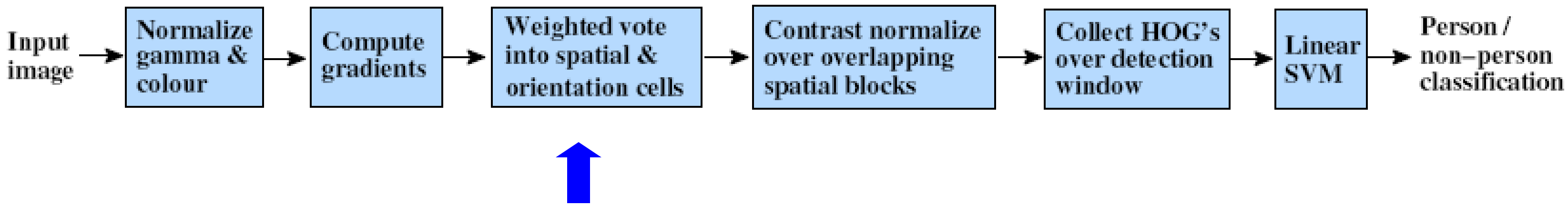
0	1
-1	0

diagonal

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

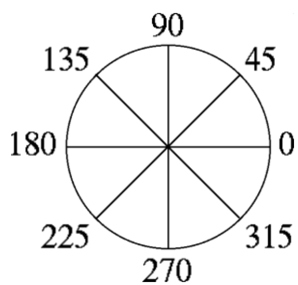
Sobel

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

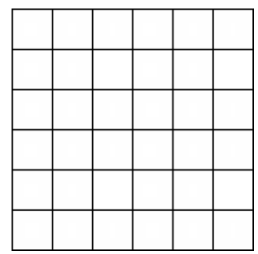


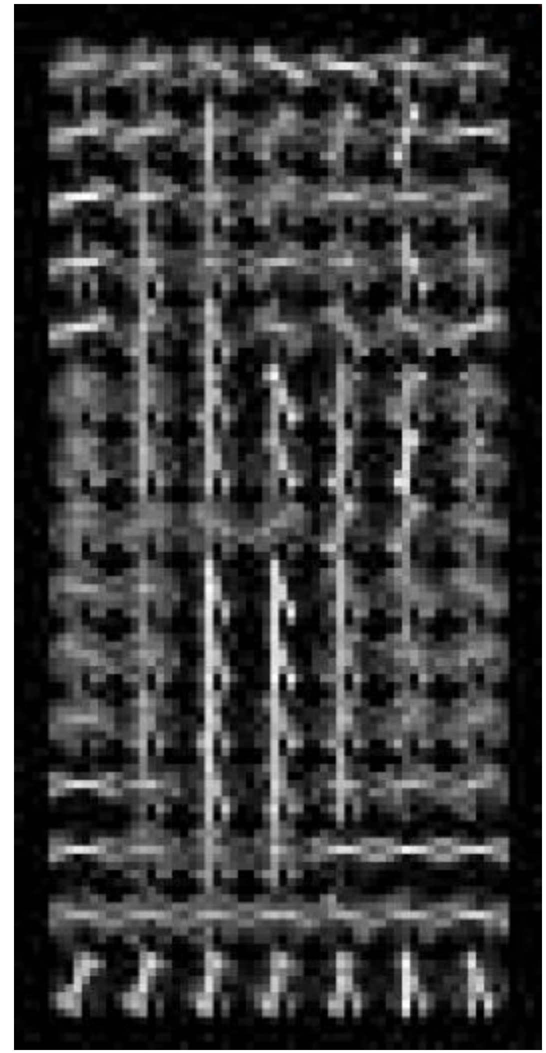
- Histogram of gradient orientations

Orientation: 9 bins
(for unsigned angles)



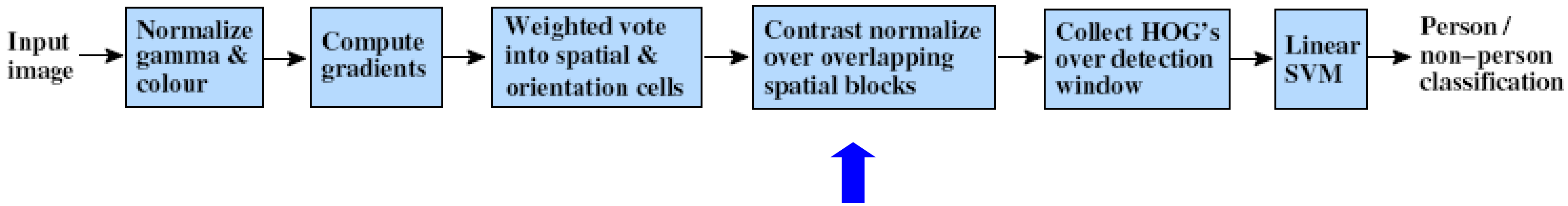
Histograms in
k x k pixel cells



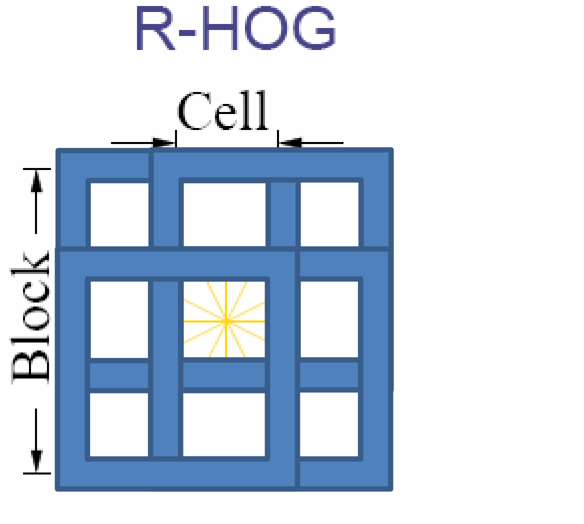


  – Votes weighted by magnitude
  – Bilinear interpolation between cells

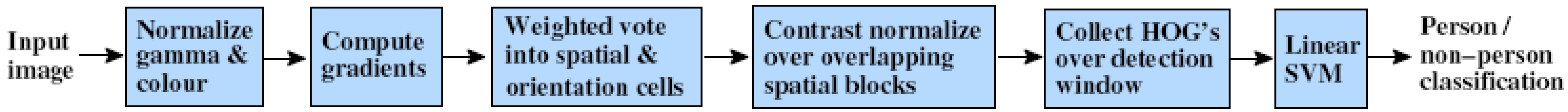
Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

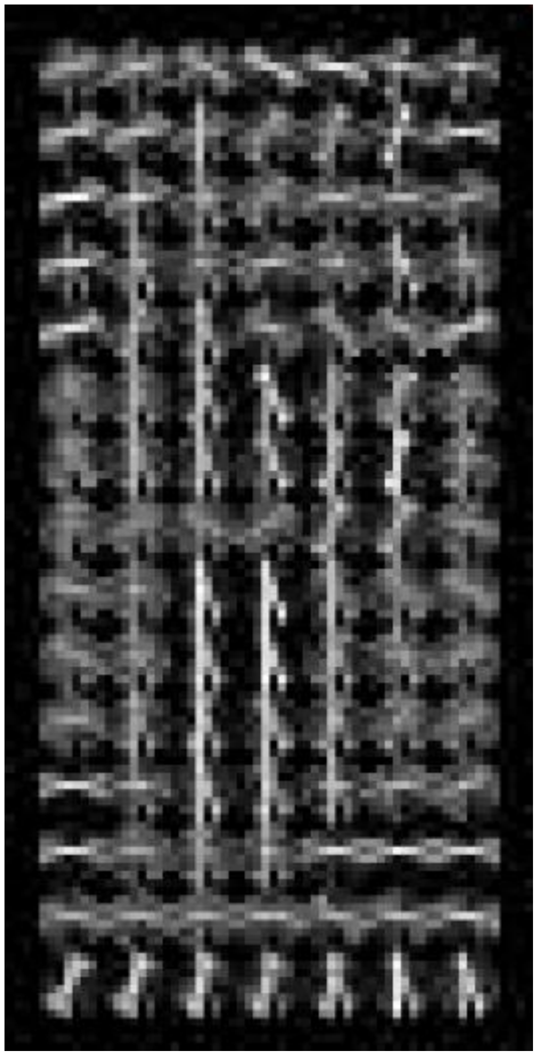


Normalize with respect to surrounding cells



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



X= 

Original Formulation

orientations

features = 15 x 7 x 9 x 4 = 3780

cells

normalizations by neighboring cells

UoCTTI variant

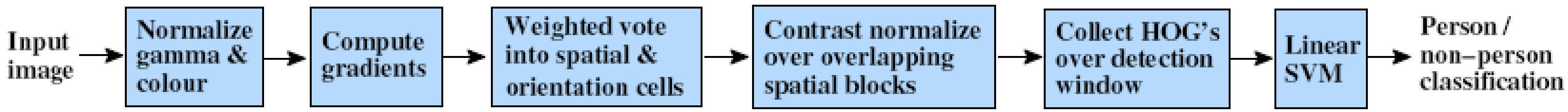
orientations

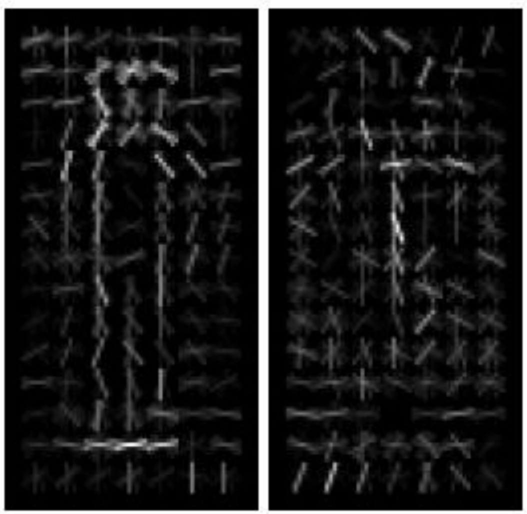
features = 15 x 7 x (3 x 9) + 4 = 3780

cells

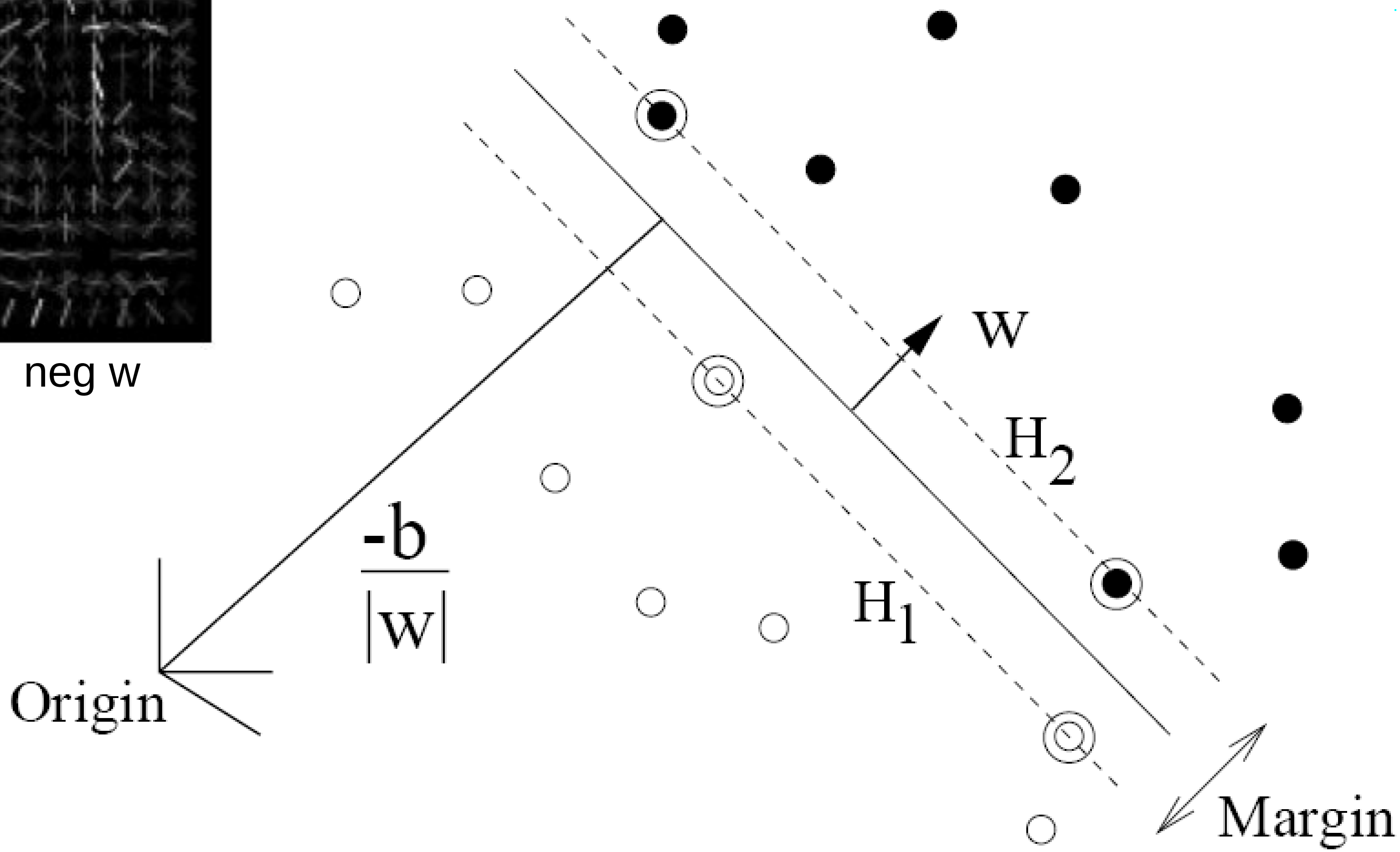
magnitude of neighbor cells

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



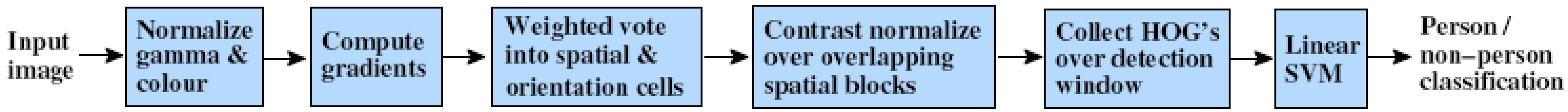


pos w　　neg w



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

Detection examples

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?

Strengths and Weaknesses of Statistical Template Approach

Strengths

- Works very well for non-deformable objects with canonical orientations: faces, cars, pedestrians
- Fast detection

Weaknesses

- Not so well for highly deformable objects or “stuff”
- 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 is size of smallest detectable object
- “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) : ~2000 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) : ~3600
  - "Parts" at fixed position, non-maxima suppression, simple cascade, rotation, pretty good accuracy, fast
- Schneiderman-Kanade (1998-2000,2004) : ~1700
  - Careful feature engineering, excellent results, cascade
- Viola-Jones (2001, 2004) : ~11,000
  - Haar-like features, Adaboost as feature selection, hyper-cascade, very fast, easy to implement
- Dalal-Triggs (2005) : ~6500
  - Careful feature engineering, excellent results, HOG feature, online code
- Felzenszwalb-Huttenlocher (2000): ~2100
  - Efficient way to solve part-based detectors
- Felzenszwalb-McAllester-Ramanan (2008): ~1300
  - Excellent template/parts-based blend