EE 4211 Computer Vision

Lecture 11A: Object detection with traditional method

Semester B, 2021-2022

Schedules

Week	Date	Topics
1	Jan. 11 (face to face)	Introduction/Imaging
2	Jan. 18 (online)	Image enhancement in spatial domain
3	Jan. 25 (online)	Image enhancement in frequency domain (HW1 out)
4	Feb. 8 (online)	Morphological processing
5	Feb. 15 (online)	Image restoration (HW1 due)
6	Feb. 22 (online)	Midterm (no tutorials this week)
7	Mar. 1 (online)	Edge detection (HW2 out)
8	Mar. 8 (online)	Image segmentation
9	Mar. 15 (online)	Face recognition with PCA, LDA (tutorial on segmentation) (HW2 due)
10	Mar. 22 (online)	Face recognition based on deep learning (Quiz on two code questions, 1 hour) Image segmentation based on deep learning
11	Mar. 29 (online)	Object detection with traditional methods Object detection based on deep learning (tutorial on detection)
12	Apr. 5	Events / Public Holidays
13	Apr. 12 (online)	Invited project presentation and Summary

Outline

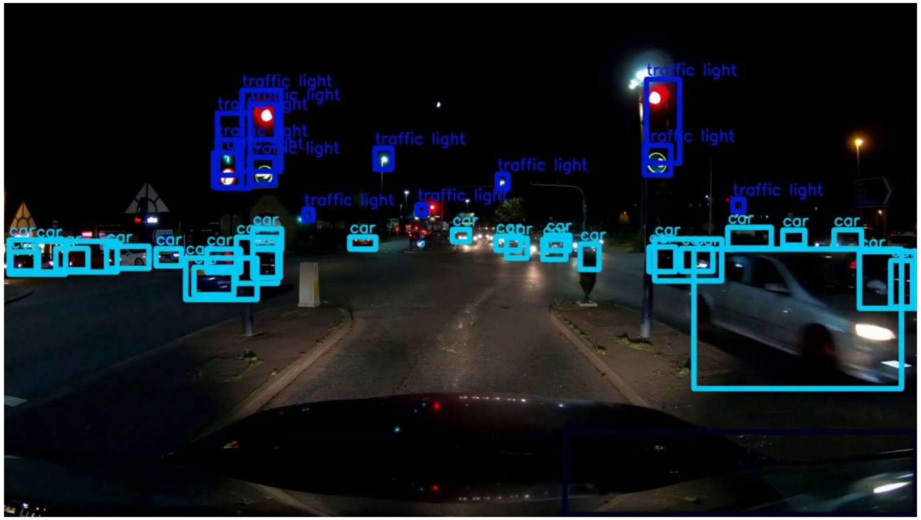
- Basic concepts with object detection
- HOG detector
- BPM detector

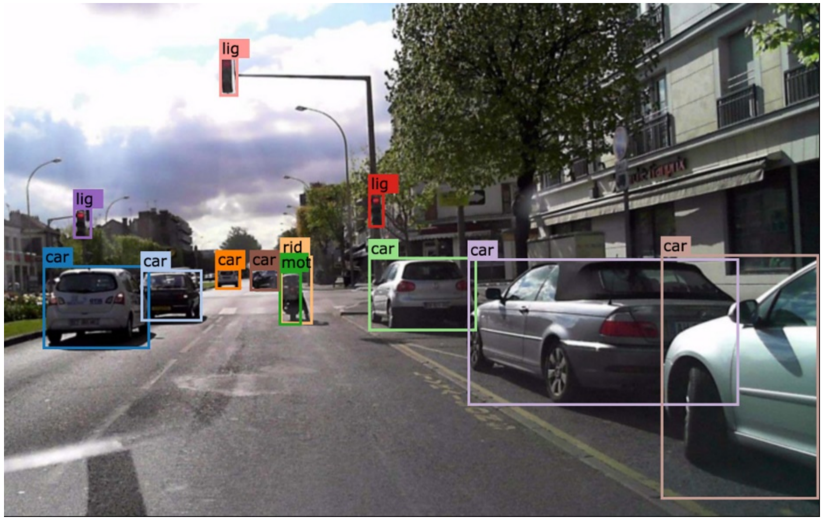
Object category detection

- Goal: detect all pedestrians, cars, tree, building, etc.

Why is it hard?

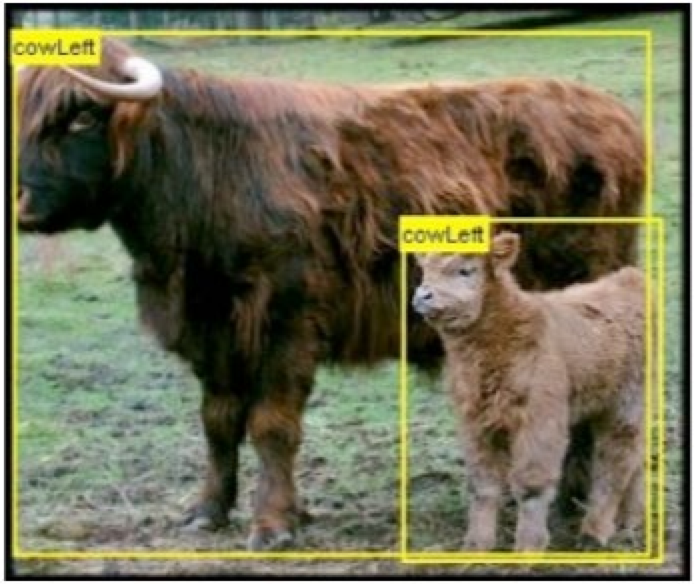
- Objects in a category have highly variable appearance
- Photometric variation
- Viewpoint variation
- Intra-class variability
  - Cars come in a variety of shapes (sedan, minivan, etc)
  - People wear different clothes and take different poses





PASCAL Challenge

- Objects from 20 categories - person, car, bicycle, bus, airplane, sheep, cow, table, ...
- Objects are annotated with bounding boxes



How to build a good object detector

- There may be overlapping instances or detections
  - We need a detection postprocessing strategy
- The method is likely to be based on learning and will need to be validated
  - We need labelled training and validation sets
- Computational cost or embeddability may be an issue
  - We need to review the whole system for efficiency

A Naive Image Scanning Detector

- **Template Matching**

Scan image at all scales and locations

↓

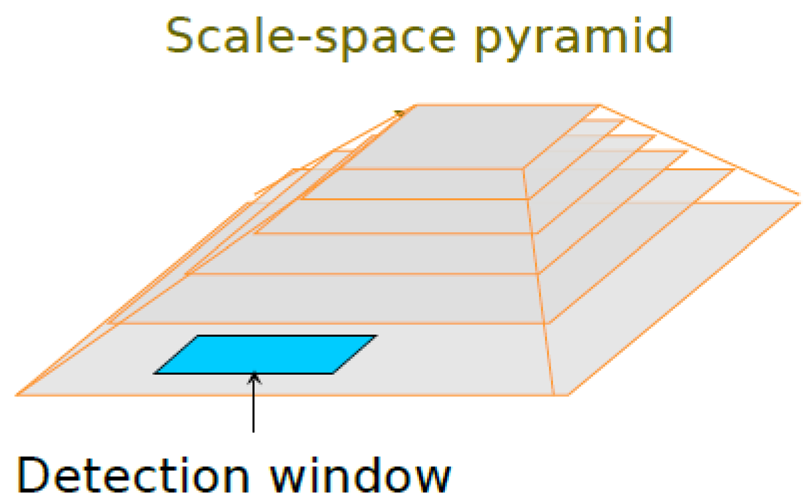
Match window against a rigid template, e.g. by correlation

↓

Return above-threshold matches as detections

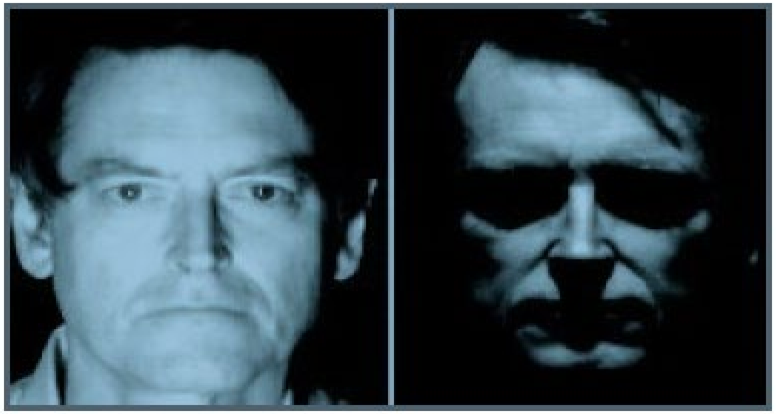
↓

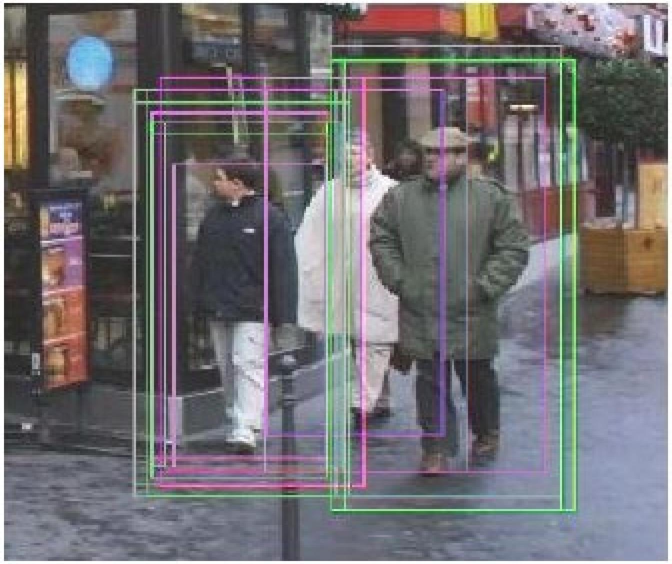
Object detections



Problems with this approach

- It is photometrically too rigid to resist changes in lighting and appearance variations
- It is geometrically too rigid to resist shape variations
- It does not have a strategy for overlapping detections





Anatomy of a Modern Object Detector

- Strong image preprocessing and feature normalization for resistance to illumination changes
- Local rectification and pooling for resistance to small shape variations
- Overcomplete feature set for rich description
- Machine learning based decision rule to capture statistics and variability of real application
- Postprocessing to fuse multiple detections

Image Scanning Detectors

Scan image(s) at all scales and locations

↓

Extract features over windows

↓

Run window classifier at all locations

↓

Fuse multiple detections in 3-D position & scale space

↓

Object detections with bounding boxes

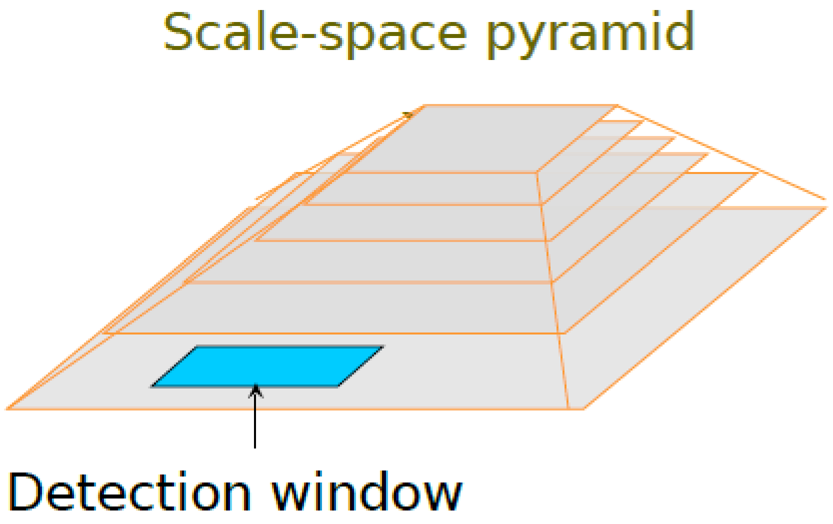
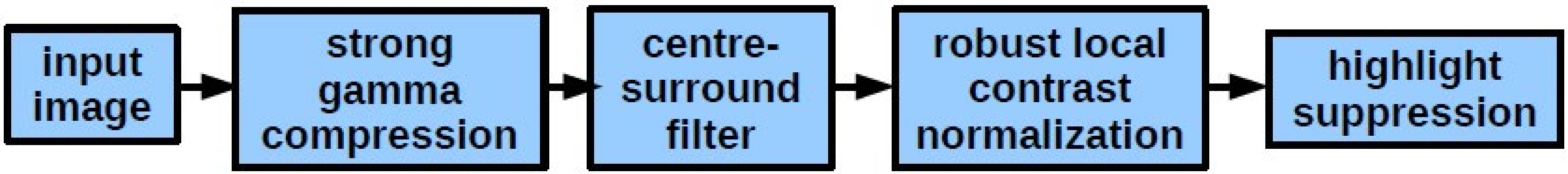
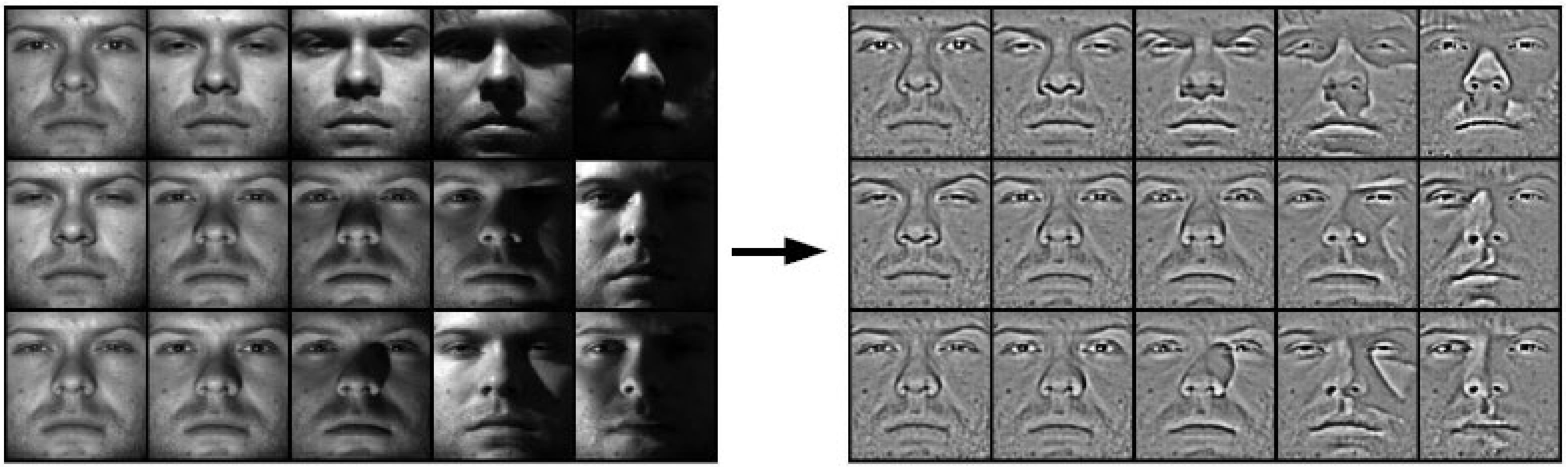


Image Preprocessing

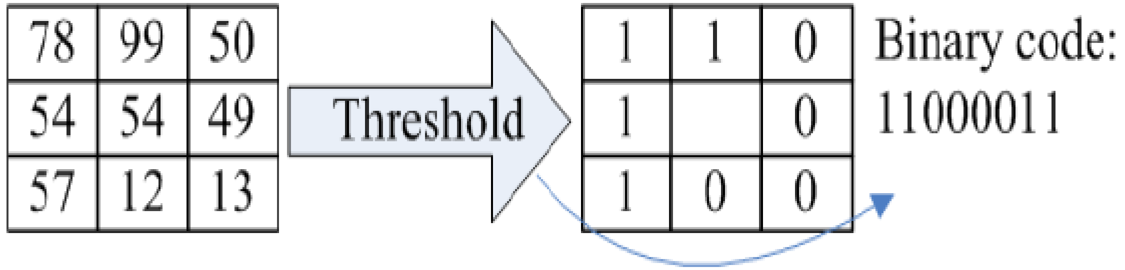
- Preprocessing is often neglected but it can make a huge difference in performance
- One example of a preprocessing chain





Local Binary Pattern Features

- Descriptors based on local thresholding or ranking of pixel or edge intensities are very resistant to illumination changes
- Local Binary Patterns
  - Threshold the pixels at value of central pixel
  - Locally histogram resulting binary codes
  - Used to be one of the best descriptors for face recognition



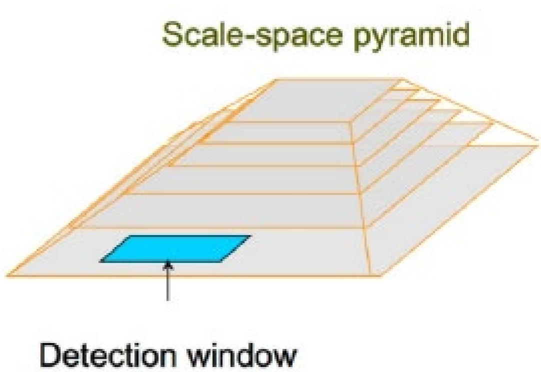
Outline

- Basic concepts with object detection
- HOG detector
- BPM detector

HOG Detector: pipline

- Sliding window

locations



scales

Scan image(s) at all scales and locations

↓

Extract features over windows

↓

Run window classifier at all locations

↓

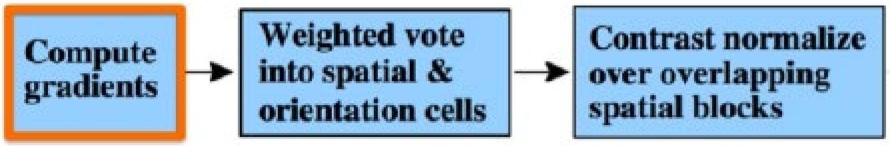
Fuse multiple detections in 3-D position & scale space

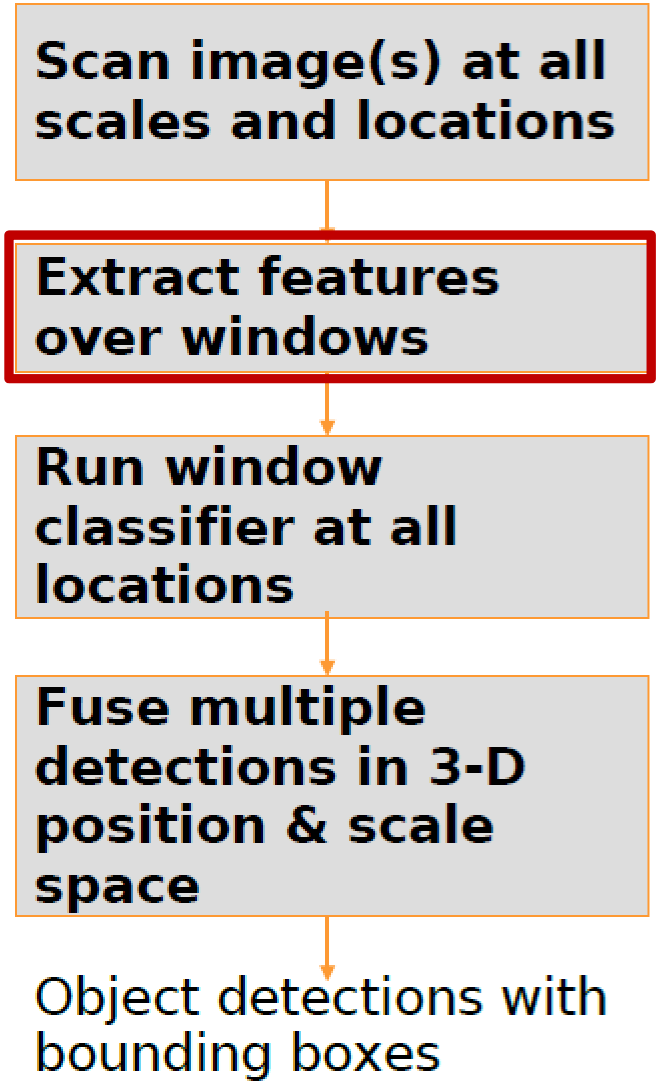
↓

Object detections with bounding boxes

HOG detector

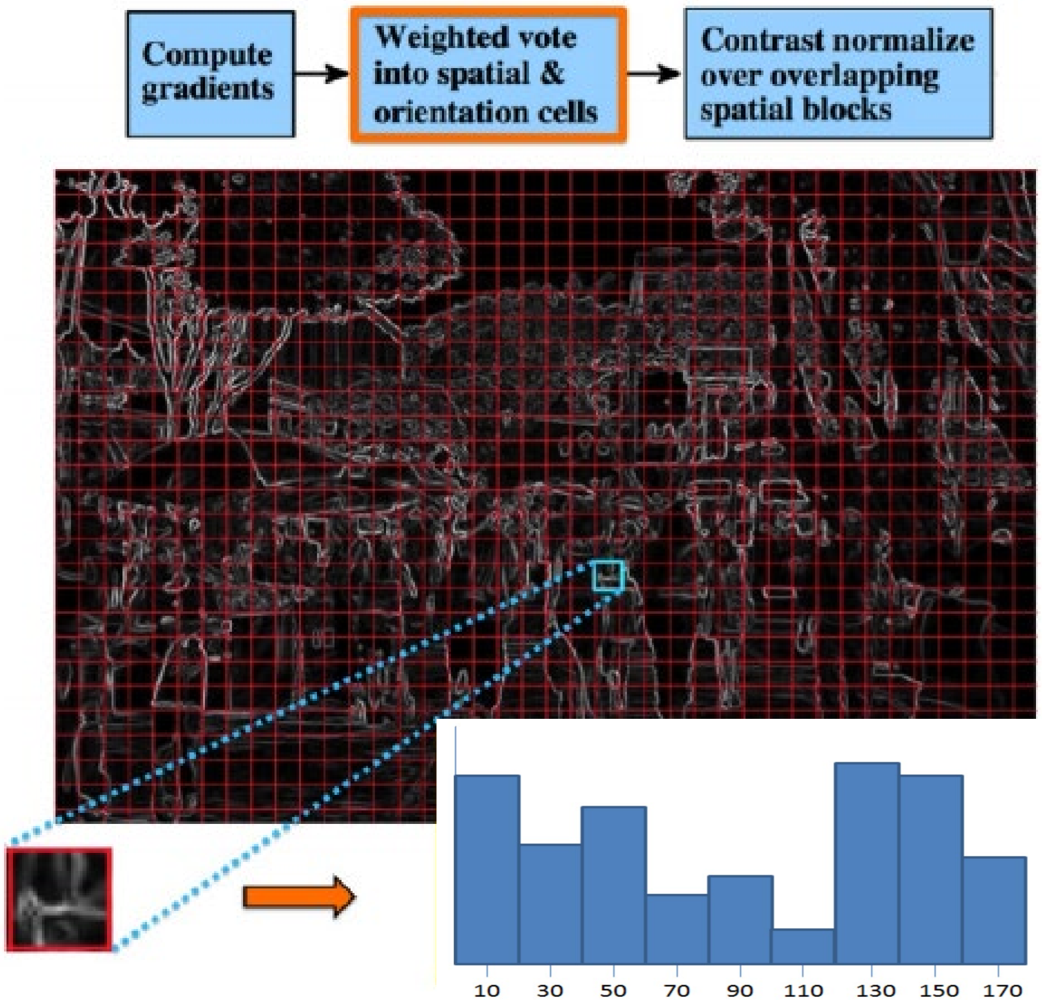
- HOG feature extraction





HOG detector

- HOG feature extraction



Scan image(s) at all scales and locations

Extract features over windows

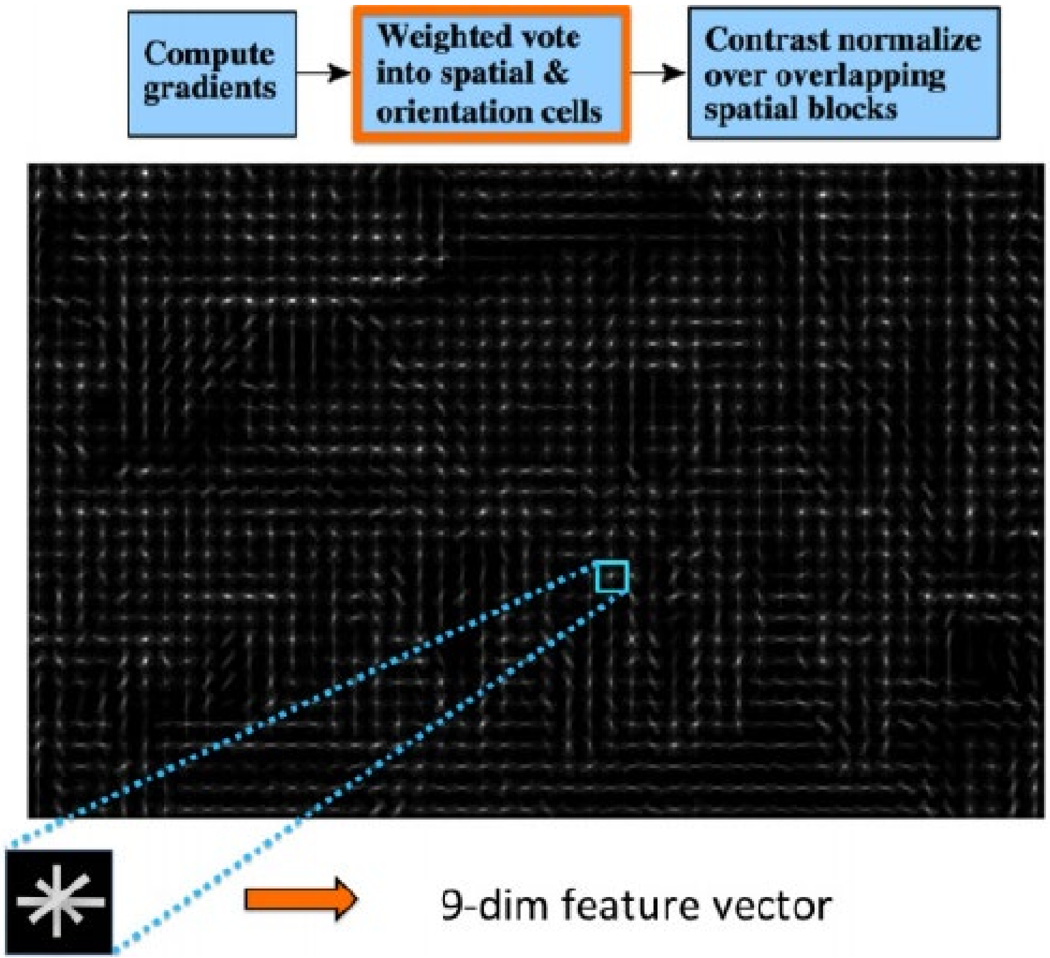
Run window classifier at all locations

Fuse multiple detections in 3-D position & scale space

Object detections with bounding boxes

HOG detector

- HOG feature extraction



Scan image(s) at all scales and locations

Extract features over windows

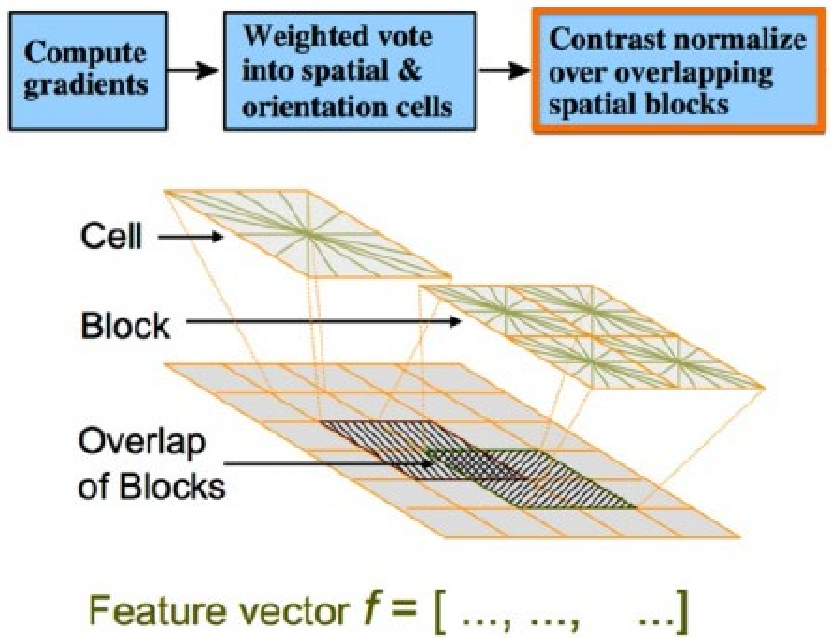
Run window classifier at all locations

Fuse multiple detections in 3-D position & scale space

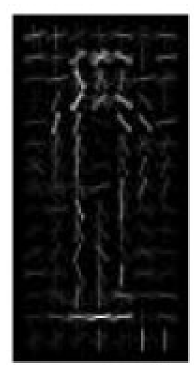
Object detections with bounding boxes

HOG detector

- HOG feature extraction
- Invariant to changes in lighting, small deformations, etc.



L2 normalization in each block: $$\mathbf{f} = \frac{\mathbf{f}}{\sqrt{||\mathbf{f}||_2^2 + \epsilon^2}}$$



Scan image(s) at all scales and locations

Extract features over windows

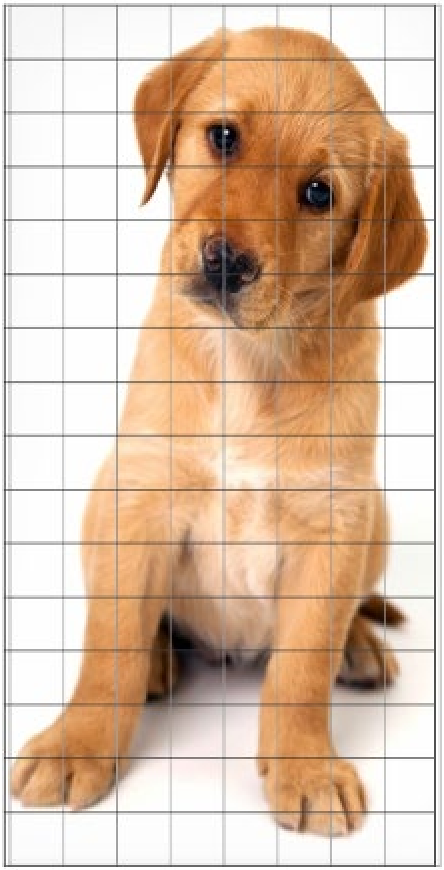
Run window classifier at all locations

Fuse multiple detections in 3-D position & scale space

Object detections with bounding boxes

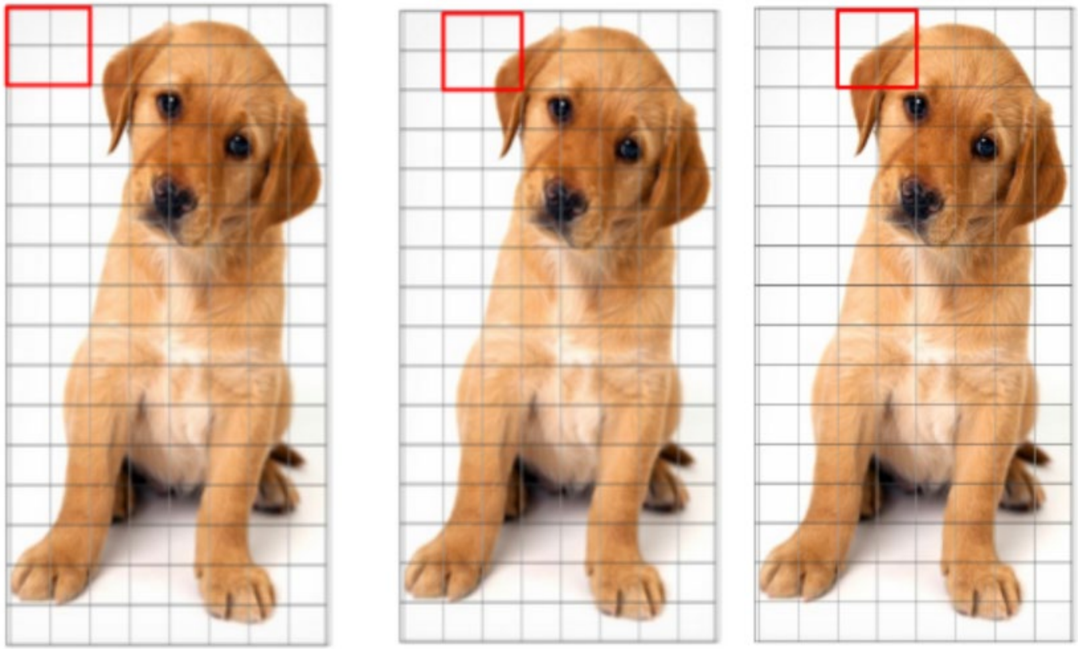
HOG feature example

- Tile 64 x 128 pixel window into 8 x 8 pixel cells
- Each cell represented by histogram over 9 orientation bins (i.e. angles in range 0-180 degrees)
- Reduce this lighting variation by normalize gradients in 16 $\times$ 16 blocks (Combine four 8 $\times$ 8 cells)
- Each 8 $\times$ 8 cell has a 9 $\times$ 1 matrix for a histogram. So, we would have four 9 $\times$ 1 matrices or a single 36 $\times$ 1 matrix.



HOG feature example

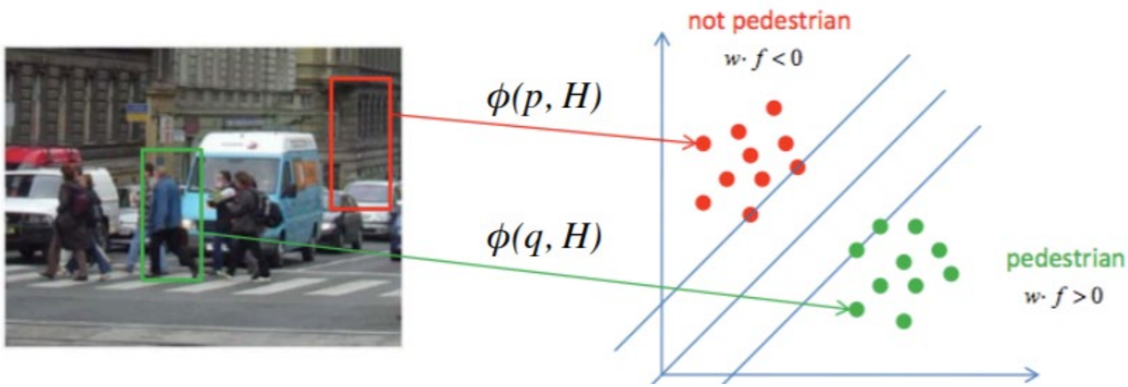
- We would have 105 (7 × 15) blocks of 16 × 16. Each of these 105 blocks has a vector of 36 × 1 as features. Hence, the total features for the image would be 105 x 36 × 1 = 3780 features.



HOG detector

- Training:
  - Train a classifier (person vs no person)

Train classifier → Predict presence/absence of object class in each image window



Scan image(s) at all scales and locations

↓

Extract features over windows

↓

Run window classifier at all locations

↓

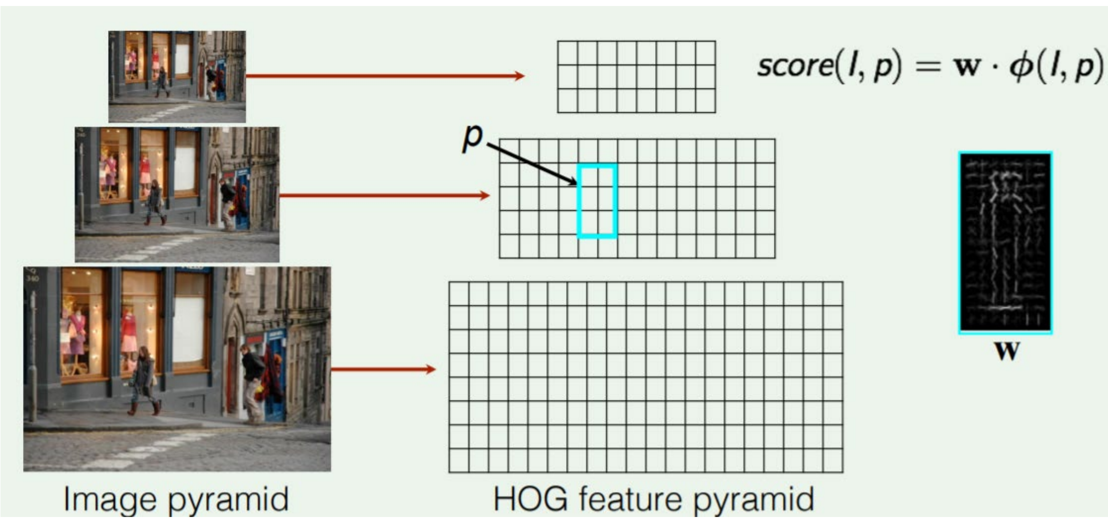
Fuse multiple detections in 3-D position & scale space

↓

Object detections with bounding boxes

HOG detector

- Detection:
  - Use the trained classifier to predict presence/absence of object class in each window in the image
  - Filters are rectangular template defining weights for features
  - Score is dot product of filter and subwindow of HOG pyramind



Scan image(s) at all scales and locations

↓

Extract features over windows

↓

Run window classifier at all locations

↓

Fuse multiple detections in 3-D position & scale space

↓

Object detections with bounding boxes

HOG detector

- Non-maxima suppression (NMS)
- Remove all boxes that overlap more than a criteria (typically 50%) with the chosen box

$$\text{overlap} = \frac{\text{area}(box_1 \cup box_2)}{\text{area}(box_1 \cap box_2)} > 0.5 \;\Rightarrow\; \boxed{\text{remove } box_2}$$

Scan image(s) at all scales and locations

↓

Extract features over windows

↓

Run window classifier at all locations

↓

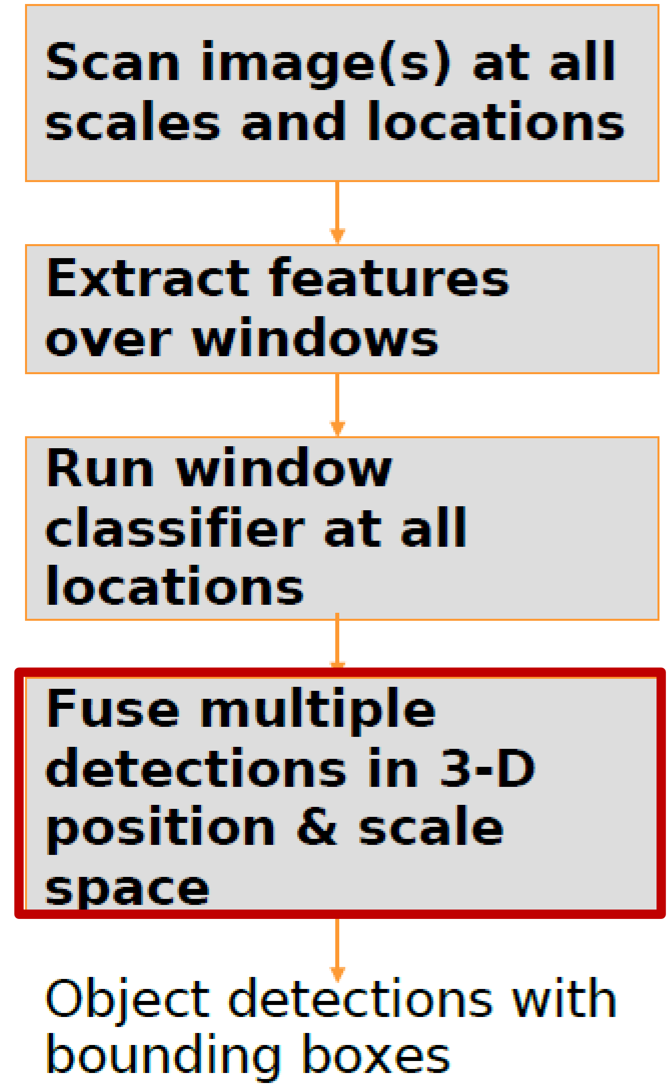
Fuse multiple detections in 3-D position & scale space

↓

Object detections with bounding boxes

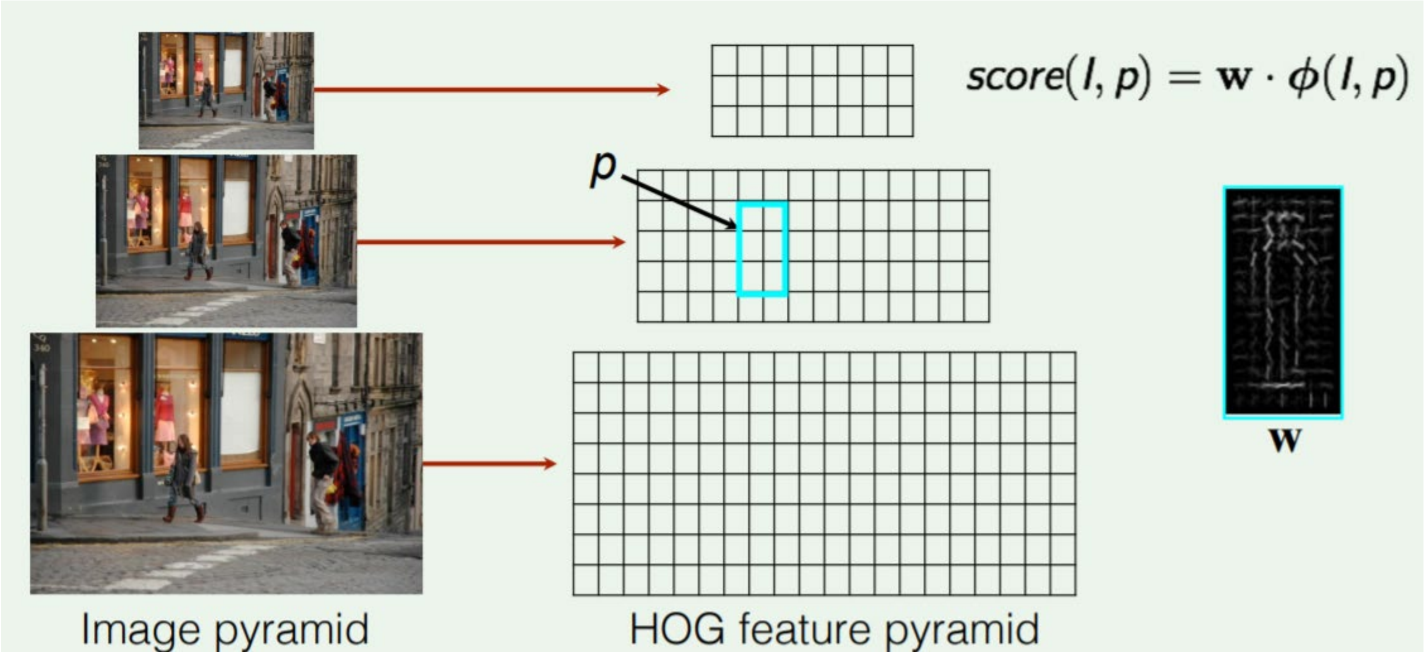
HOG detector

- Non-maxima suppression (NMS)
  - Greedy algorithm
  - At each iteration, pick the highest scoring box
  - Remove all boxes that overlap more than a criteria (typically 50%) with the chosen box



Analysis with HOG

- Compute HOG of the whole image at multiple resolutions
- Score every window of the feature pyramid
- Choose windows and apply non-maximal suppression

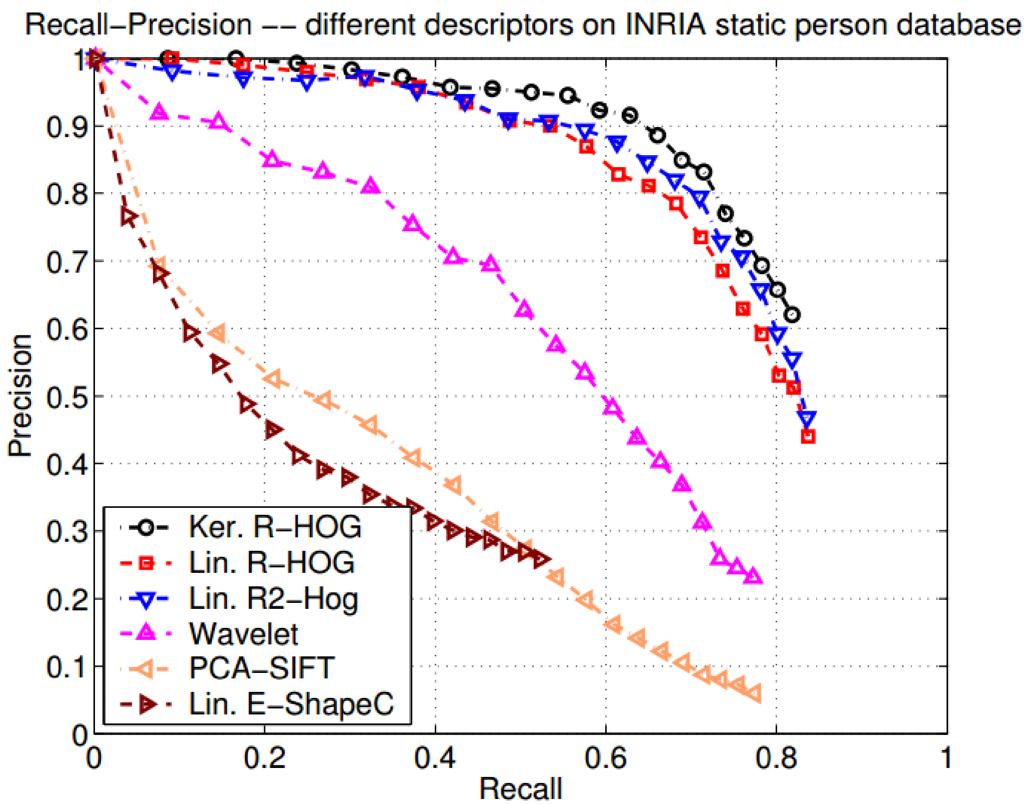


Analysis with HOG

- Example:
- number of locations p ~ 250,000 per image
- test set has ~ 5000 images
- >> 1.3x109 windows to classify
- typically only ~ 1,000 true positive locations

Analysis with HOG

- Average Precision (AP) = 75%!
- Very good!



Analysis with HOG

- AP = 12%

Outline

- Basic concepts with object detection
- HOG detector
- BPM detector

Motivation

- Great variations in objects
  - Non-rigid deformations: e.g. human in different poses
  - Intra-class variability, e.g. cars in various shape
  - Variations caused by different viewpoints and illumination
- A better representation of objects- Deformable Part Model (DPM)
  - HOG detector acts as the root filter in DPM

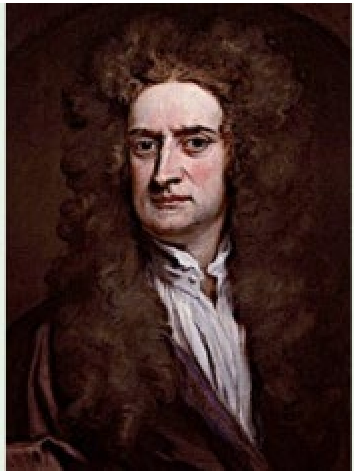
What is DPM?

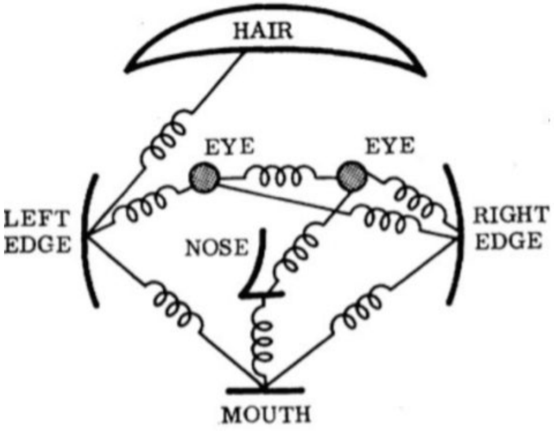
- Deformable Part Model (DPM) is a discriminatively trained, multi-scale model for image training that aim at making possible the effective use of more latent information such as hierarchical models and models involving latent three dimensional pose.
- Definition
  - Root : Catch Roughly appearance of object
  - Part : Catch local appearance of object
  - Spring : spatial connections between parts

Object Detection with Discriminatively Trained Part Based Models
http://cs.brown.edu/people/pfelzens/latent-release3/

DPM

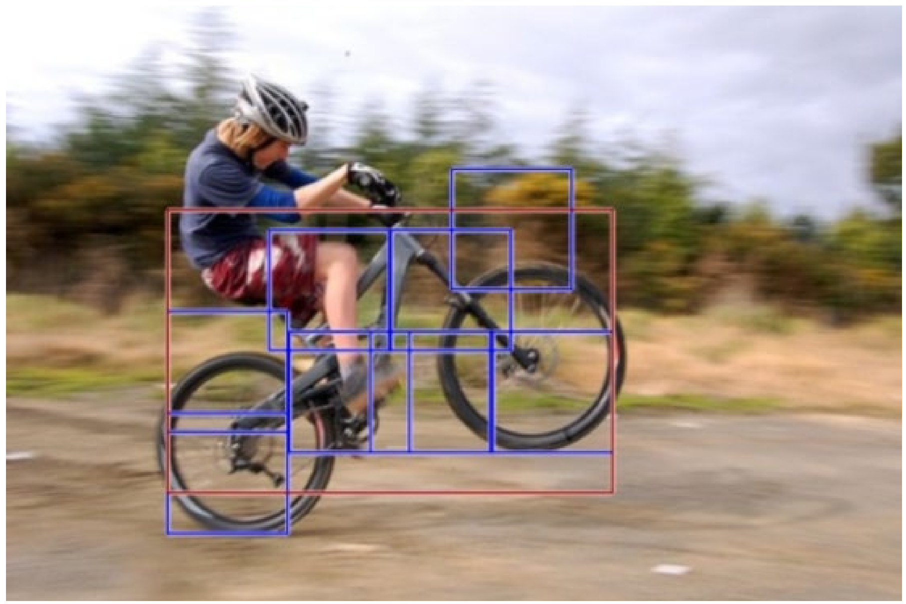
- Part based model
  - Each part represents local visual properties
  - Springs capture spatial relationships
- Goal: alignment of part model with features in an image.

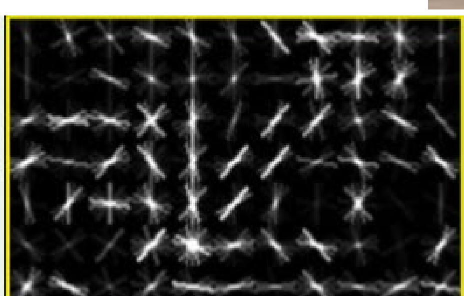




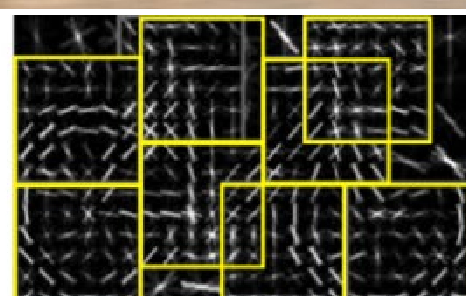
DPM

- Root filter models coarse whole-object appearance
- Part filters model finer-scale appearance of smaller patches

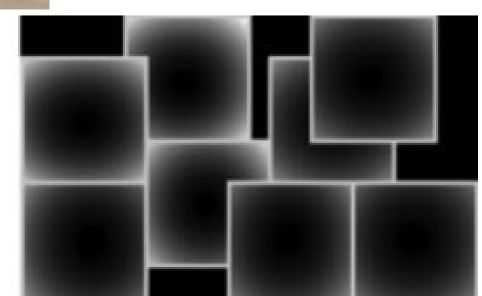




Root filter



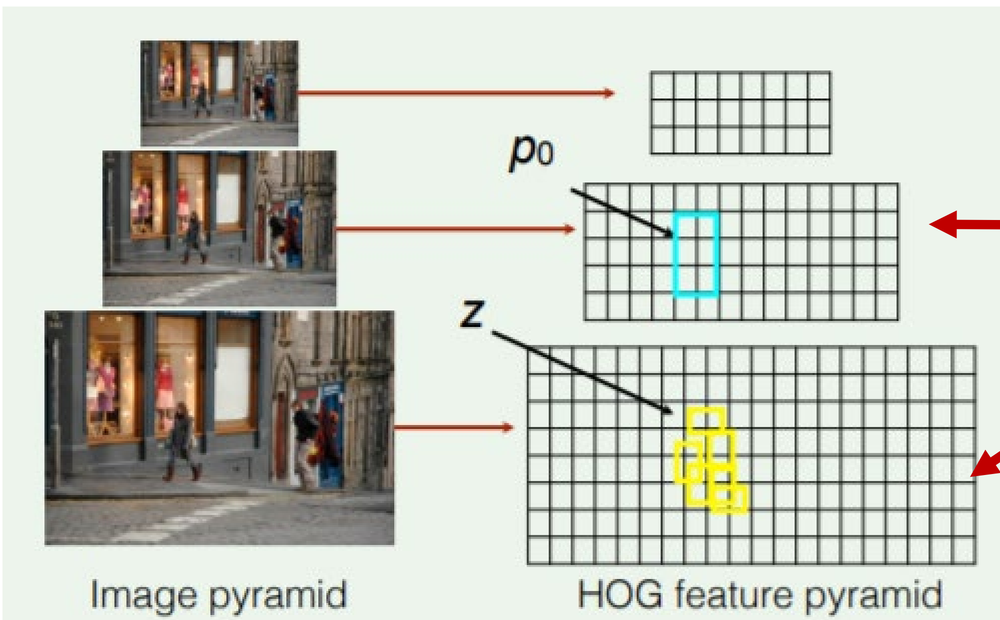
Part filters

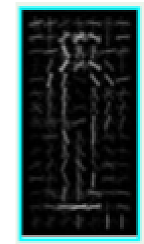


Deformation cost

DPM detector

- Add parts to the HOG detector
- Linear filters / sliding-window detector
- Discriminative training





$z = (p_0, \ldots, p_n)$

p_0 : location of root

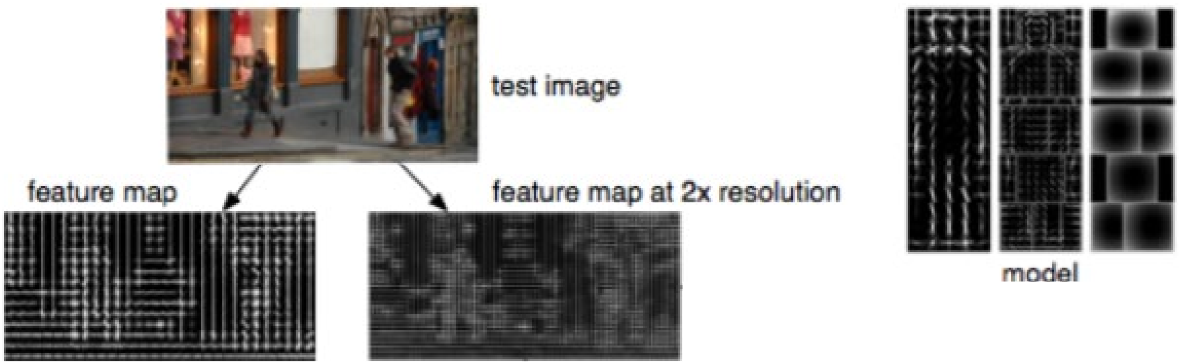
$p_1, \ldots, p_n$: location of parts

$z = (p_1, \ldots, p_n)$

$$\text{score}(I, p_0) = \max_{p_1, \ldots, p_n} \underbrace{\sum_{i=0}^{n} m_i(I, p_i)}_{\text{Filter scores}} - \underbrace{\sum_{i=1}^{n} d_i(p_0, p_i)}_{\text{Spring costs}}$$

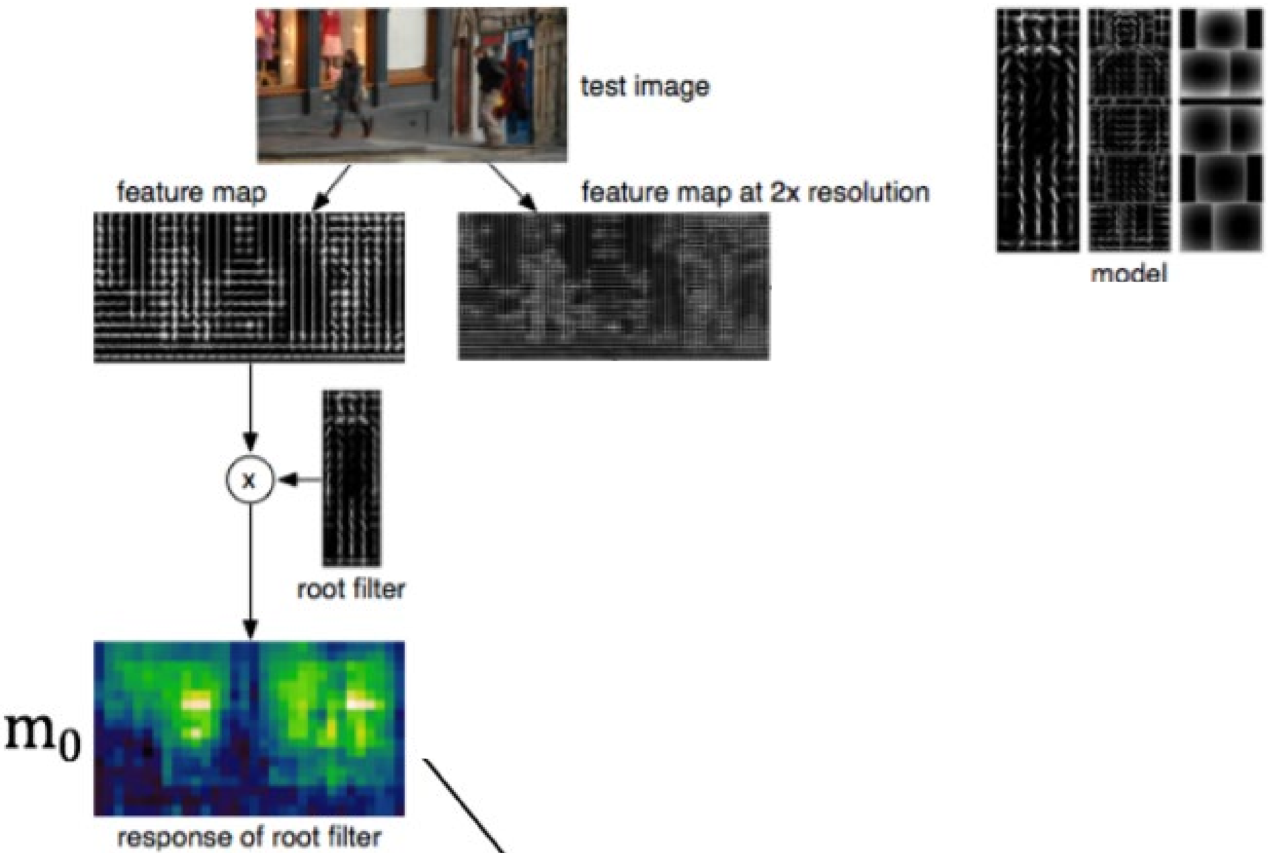
Score is sum of filter scores minus deformation costs

DPM detector



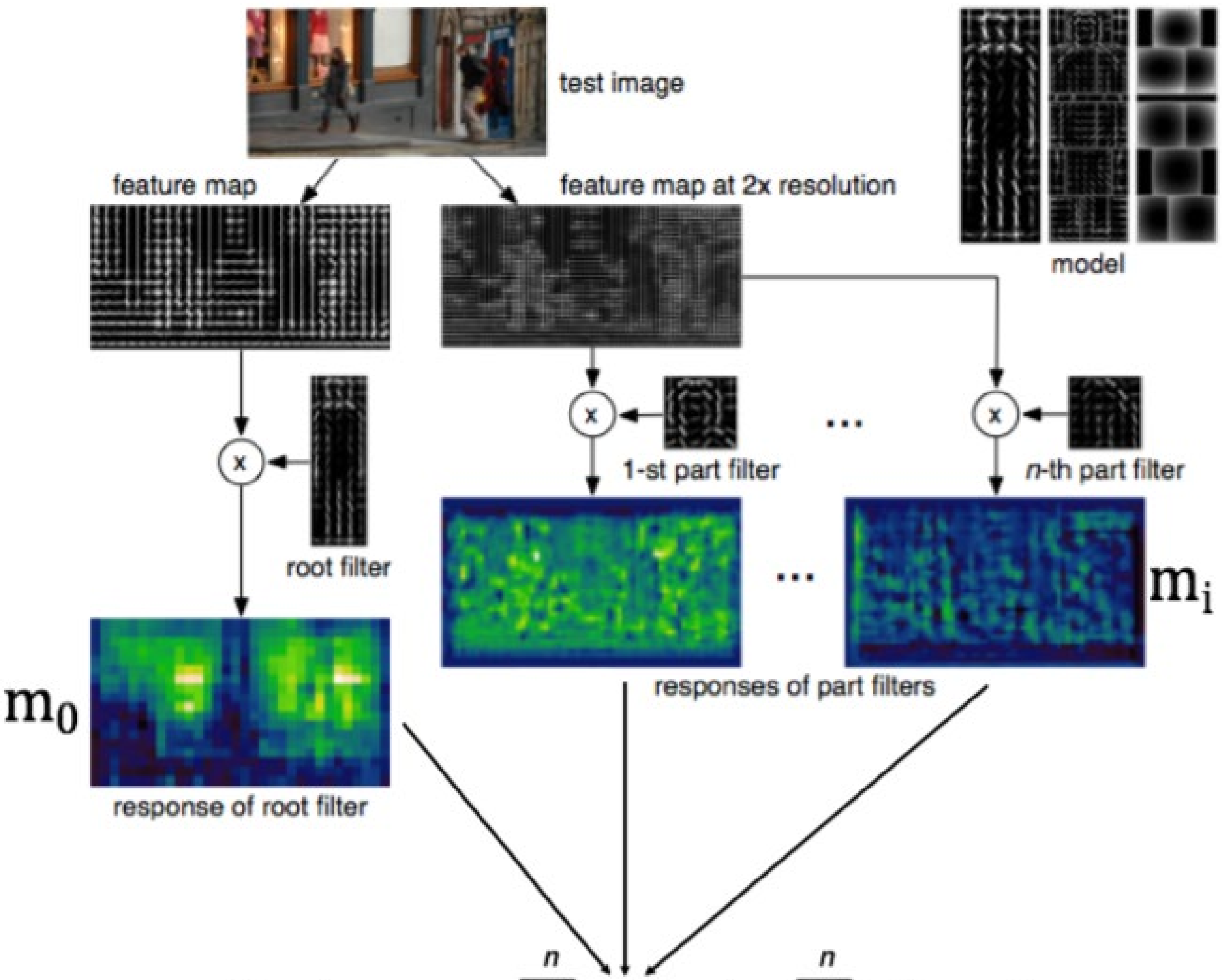
Root scale Part scale

DPM detector



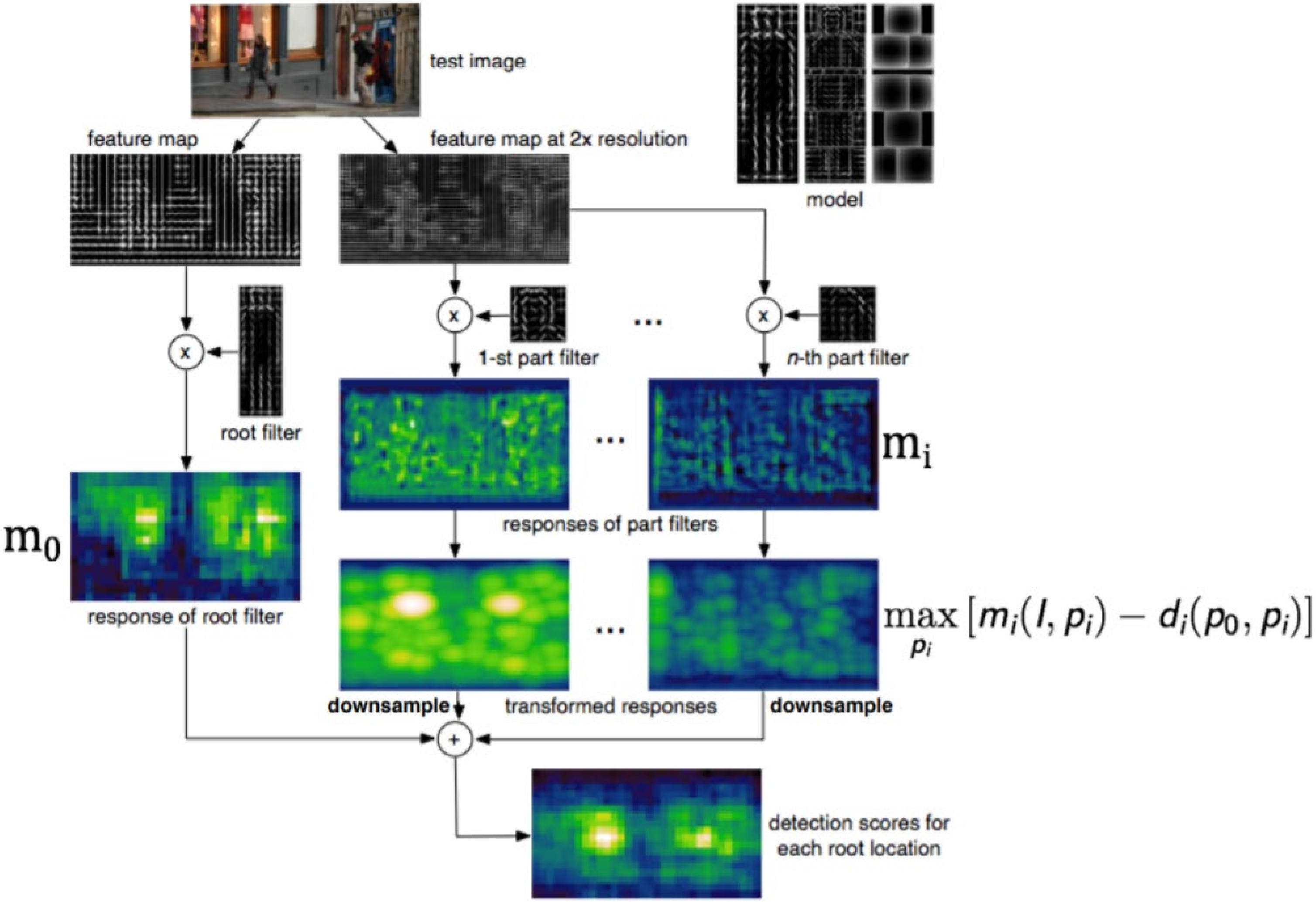
$$\text{score}(I, p_0) = \max_{p_1,\ldots,p_n} \sum_{i=0}^{n} m_i(I, p_i) - \sum_{i=1}^{n} d_i(p_0, p_i)$$

DPM detector



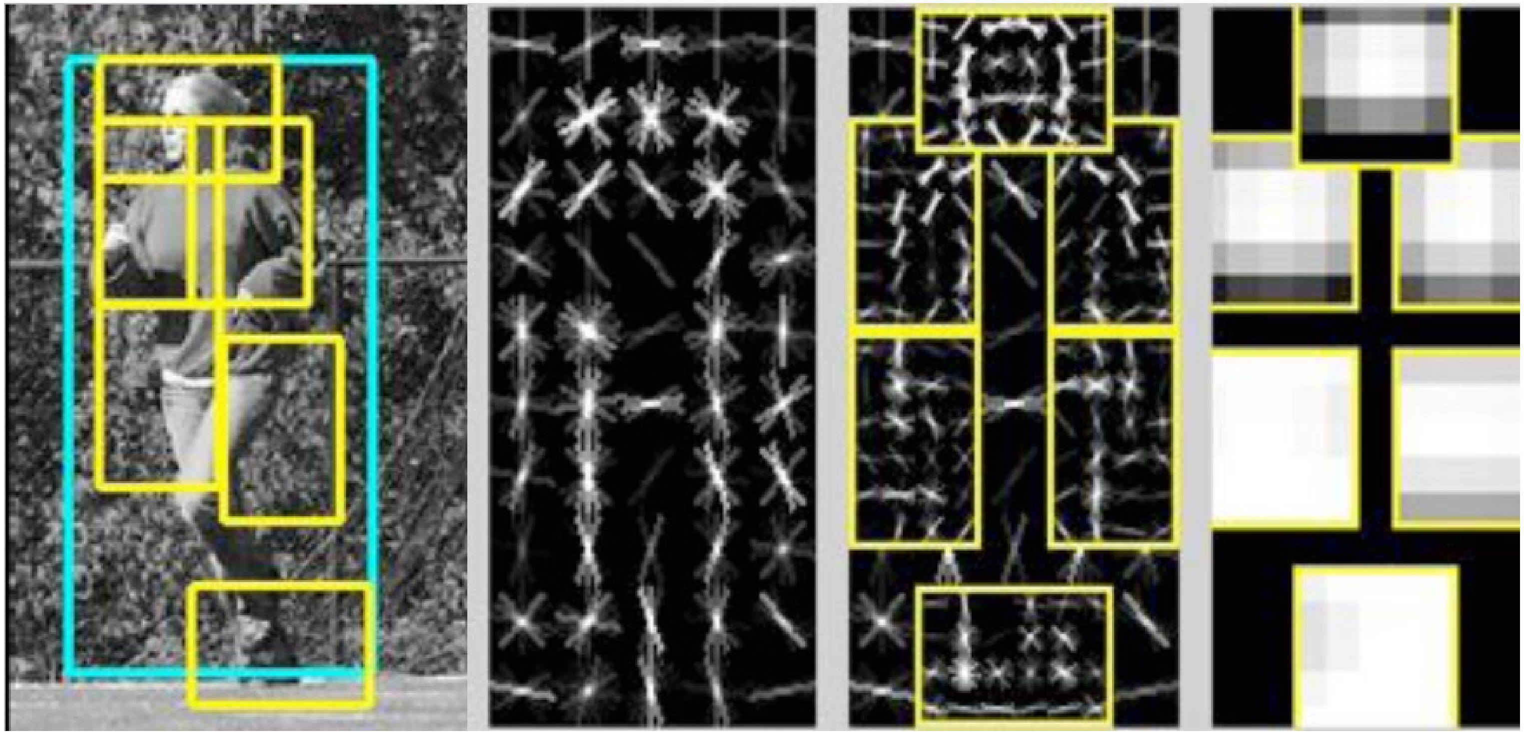
$$\text{score}(I, p_0) = \max_{p_1,\ldots,p_n} \sum_{i=0}^{n} m_i(I, p_i) - \sum_{i=1}^{n} d_i(p_0, p_i)$$

DPM detector



DPM detector

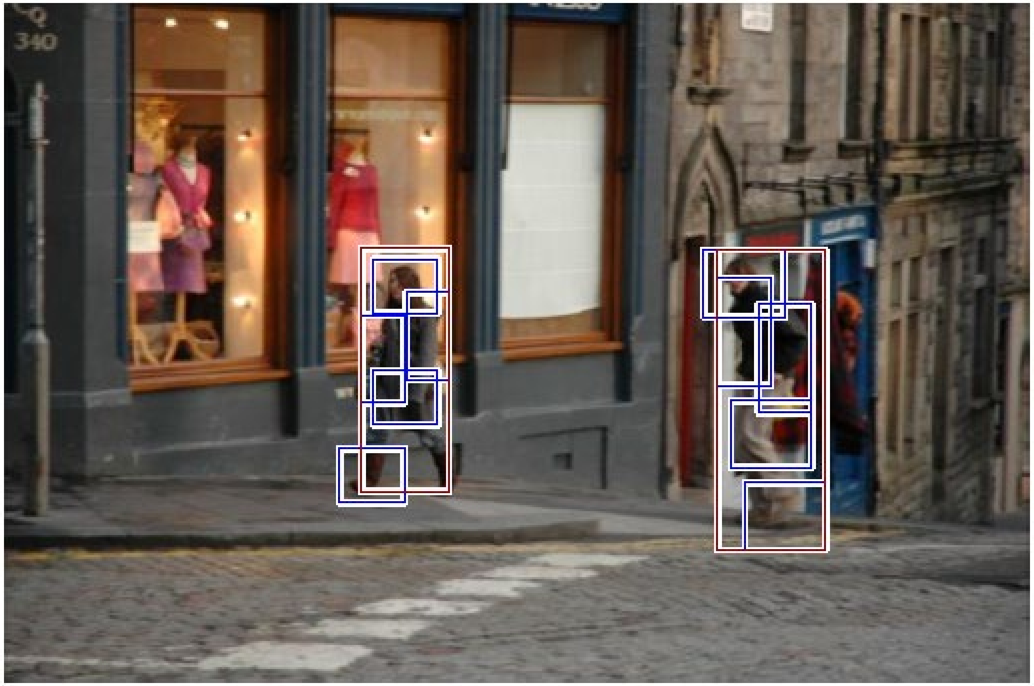
- The deformable model includes both a coarse global template covering an entire object and higher resolution part templates. The templates represent histogram of gradient features

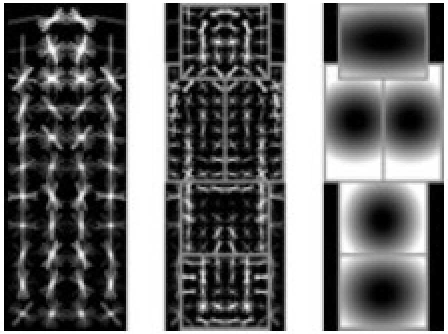


(a) Person detection example (b1) coarse template (b2) part templates (b3) spatial model

Matching results

- ~1 second to search all scales on a multi-core computer





DPM detector

- Experiment results

cat