Graphical Models in Computer Vision

Gerard Pons-Moll

Max Planck Institute for Intelligent Systems
Perceiving Systems

June 20, 2016

MAX-PLANCK-GESELLSCHAFT

Syllabus

11.04.2016	Introduction
18.04.2016	Graphical Models 1
25.04.2016	Graphical Models 2 (Sand 6/7)
02.05.2016	Graphical Models 3
09.05.2016	Graphical Models 4
23.05.2016	Body Models 1
30.05.2016	Body Models 2
06.06.2016	Body Models 3
13.06.2016	Body Models 4
20.06.2016	Object Detection 1
27.06.2016	Object Detection 2
04.07.2016	Stereo
11.07.2016	Optical Flow
18.07.2016	Segmentation

Todays topic

Recognition

- Motivation
- Image Categorization
  - Bag-of-Words Model
  - Spatial Pyramids
- Object Detection
  - Implicit Shape Model (ISM)
  - Sliding Window Detection
  - Viola-Jones Detector
  - Histogram of Oriented Gradients (HOG)

What is object detection?

What is object detection?

Object detection vs. Categorization

Categorization:

- Determine what is in an image
  (*e.g.*, swiss alps)
- Ambiguous if multiple objects are present
  (*e.g.*, flying dogs, fence)

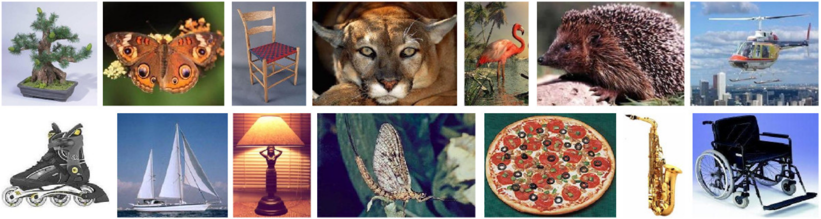
Object detection:

- Determine where an object is in an image
  (*e.g.*, we can draw bounding boxes around each dog)
- Possible for well-defined objects
  (*e.g.*, complex shapes can't be well approximated with boxes)
- Not possible for "stuff" regions (*e.g.*, grass, mountain, sky)
  (but we can give labels to individual pixel $\Rightarrow$ semantic segmentation)

How many visual object categories are there?

Dataset: Caltech 101

- $\sim$ 101 categories, $40 - 800$ images per category [Fei-Fei, 2004]

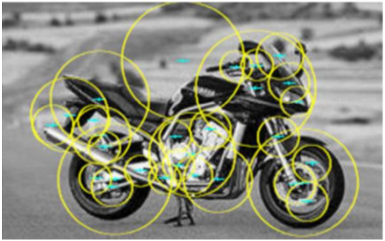


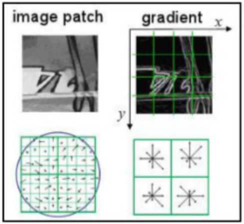
Multi-class classification results (30 training images per class)

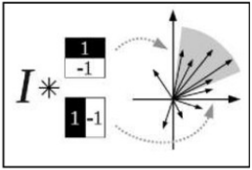
	Weak features (16)		Strong features (200)	
Level	Single-level	Pyramid	Single-level	Pyramid
0	15.5 ±0.9		41.2 ±1.2	
1	31.4 ±1.2	32.8 ±1.3	55.9 ±0.9	57.0 ±0.8
2	47.2 ±1.1	49.3 ±1.4	63.6 ±0.9	**64.6** ±0.8
3	52.2 ±0.8	**54.0** ±1.1	60.3 ±0.9	64.6 ±0.7

Categorization: Image Description by Local Features

Classical Descriptors: SIFT and SURF



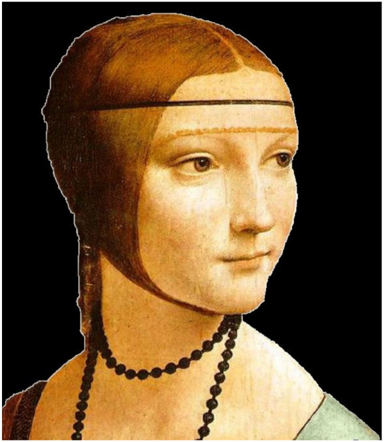




SIFT Feature Extraction: [Lowe, 2004]

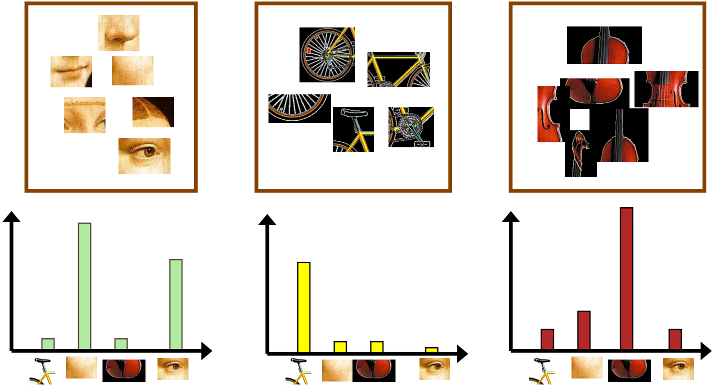
- Detect keypoints (*e.g.*, blobs in scale-space)
- Extract descriptor
  - Extract patch
  - Calculate gradients
  - Create local histograms
  - Normalize
- Robust wrt. slight transformations (translation, rotation, intensity)

Categorization: Bag-of-Words



- Represent image by bag of patches/features [Fei-Fei, 2003]

Categorization: Bag-of-Words



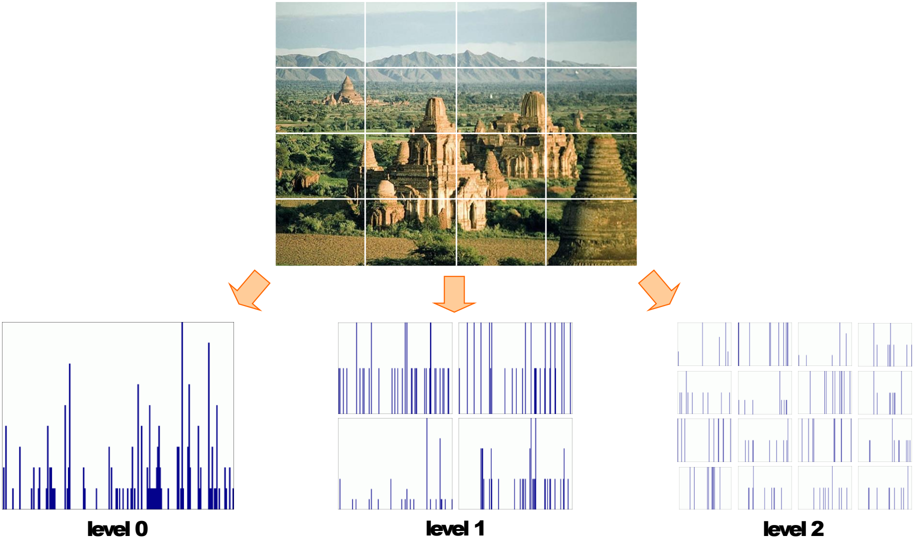
Bag-of-Words Approach:

- Learn "visual vocabulary" from large set of features
- Quantize all features in the image using this vocabulary
- Represent images by frequencies of "visual words"
- What is the problem with this representation?

Categorization: Bag-of-Words

- Spatial information has been lost (*i.e.*, where the patches came from)
- All images above are treated as being the same!

Categorization: Spatial Pyramid Matching

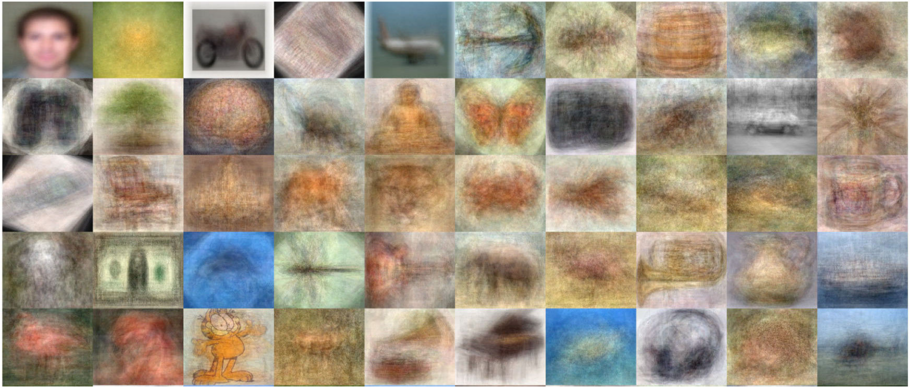


- Locally orderless representation at multiple levels [Lazebnik, 2006]

Caltech 101 - Average Images

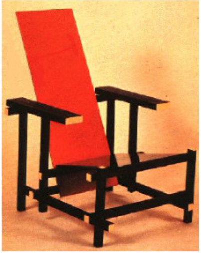
Is Caltech 101 a challenging/realistic dataset?

- No or little clutter
- Objects are centered in the image
- Most objects presented in stereotypical pose

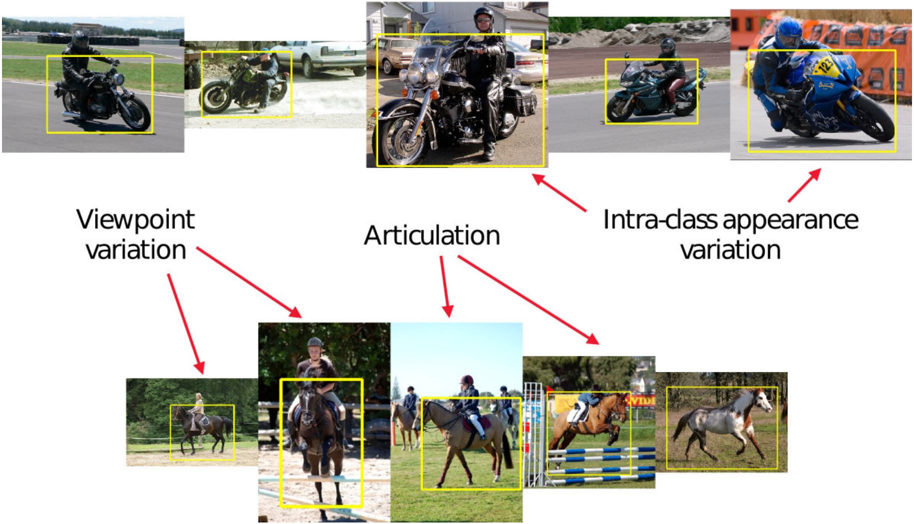


Challenges in Object Detection



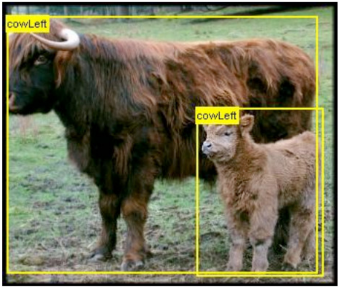



Challenges in Object Detection



- Not centered, complex backgrounds, complex lighting, occlusions, ...

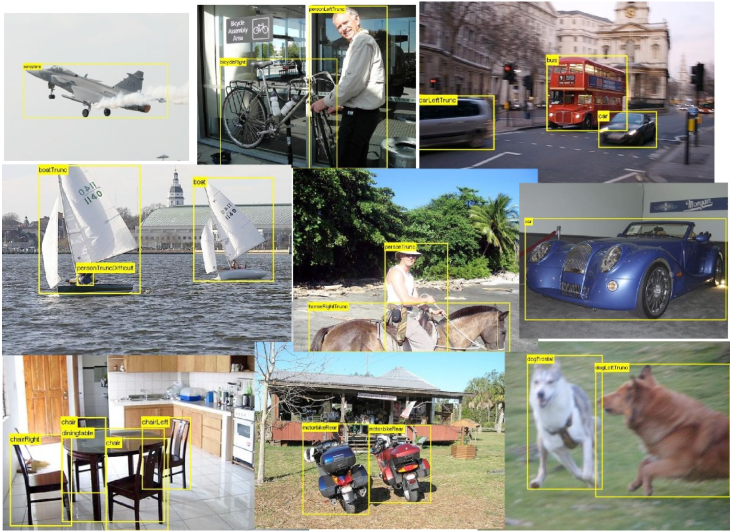
PASCAL VOC



PASCAL VOC Dataset:

- $\sim 10,000$ images with $\sim 25,000$ objects
- Large photometric/viewpoint variation and intra-class variability
- Objects from 20 categories (person, car bicycle, cow, table, ...)
- Objects are annotated with labeled bounding boxes

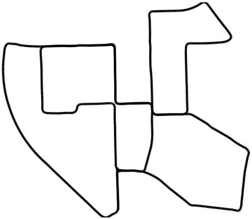
PASCAL VOC

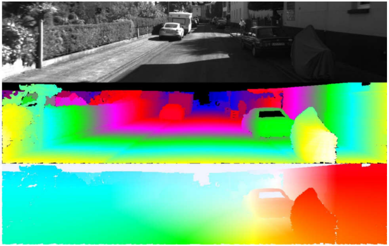


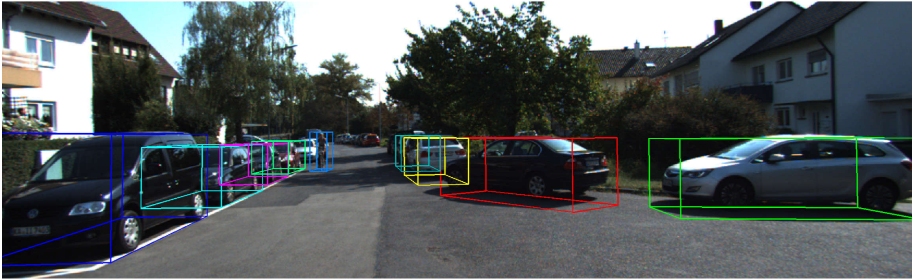
[Everingham et al., 2005-2012]

http://pascallin.ecs.soton.ac.uk/challenges/VOC/

KITTI







[Geiger et al., 2012] `http://www.cvlibs.net/datasets/kitti/`

Object Detection

From Image Categorization to Object Detection:

- Can we transfer ideas from categorization to detection? How?
- Yes, "categorize" each possible rectangle!
- However, objects convey more structural regularity than scenes, thus such a model will not perform very well
- We need something more rigid, which can capture the local and global shape of an object!

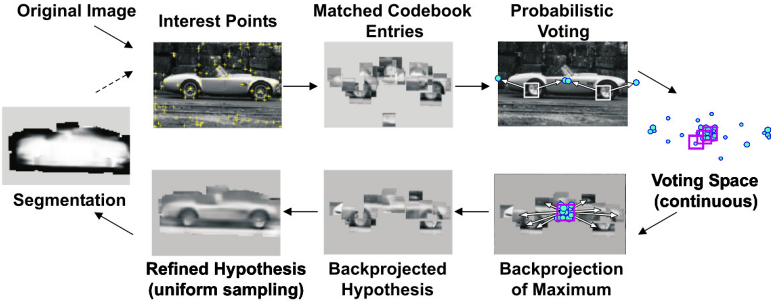
Object Detection Overview

Object Detection Methods:

- Feature-based methods
  - Implicit Shape Model
- Sliding-window-based methods
  - Viola-Jones
  - Dalal-Triggs
  - DPM
- Proposal Regions + complex predictor (CNN)
  - More about this in the last lecture!

Implicit Shape Model

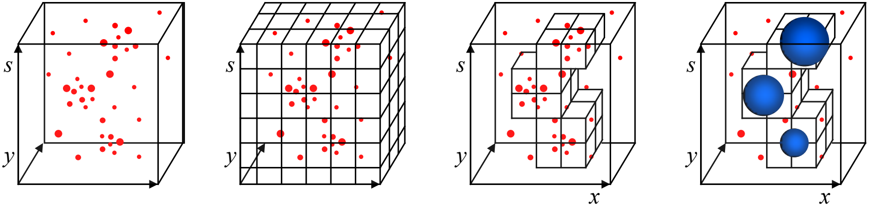
Implicit Shape Model



[Leibe, 2004]

- Detect interest points, extract descriptors, match to codebook
- Cast vote according to associated spatial uncertainty
- Probabilistic Generalized Hough Transform (scale = 3rd dim.)
- Find modes using the mean shift algorithm

Implicit Shape Model



Scale votes | Binned accum. array | Candidate maxima | Refinement

Efficient Continuous Generalized Hough Transform:

- Binned accumulator array similar to standard hough transform
- Quickly identify candidate maxima locations
- Refine locations by Mean-Shift (search only around identified maxima)
- Avoid quantization effects by keeping exact vote locations

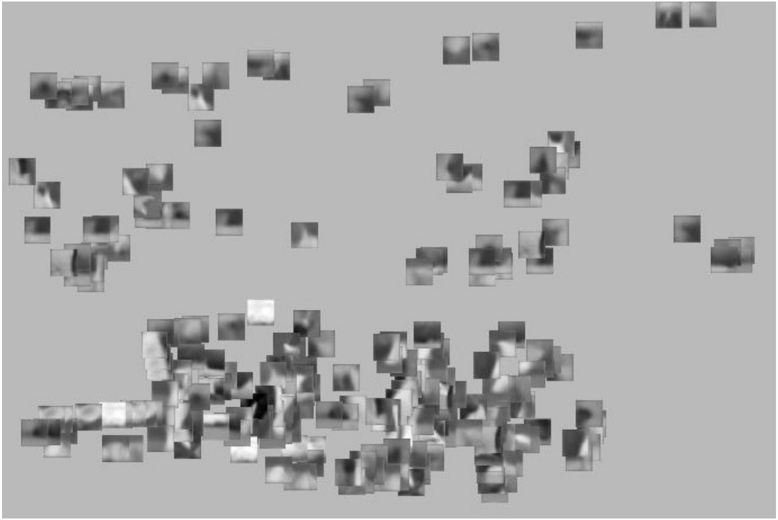
Implicit Shape Model: Example

Input Image

Implicit Shape Model: Example

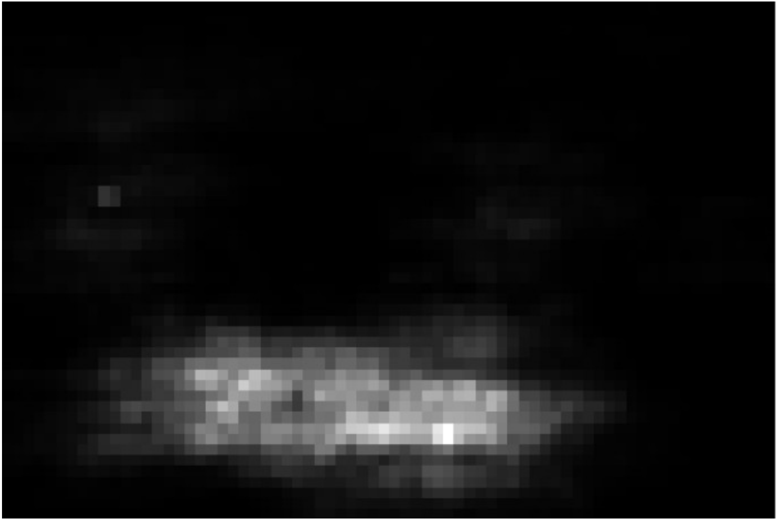
Interest Points

Implicit Shape Model: Example



Matched Patches

Implicit Shape Model: Example



Prob. Votes

Implicit Shape Model: Example

1st Hypothesis

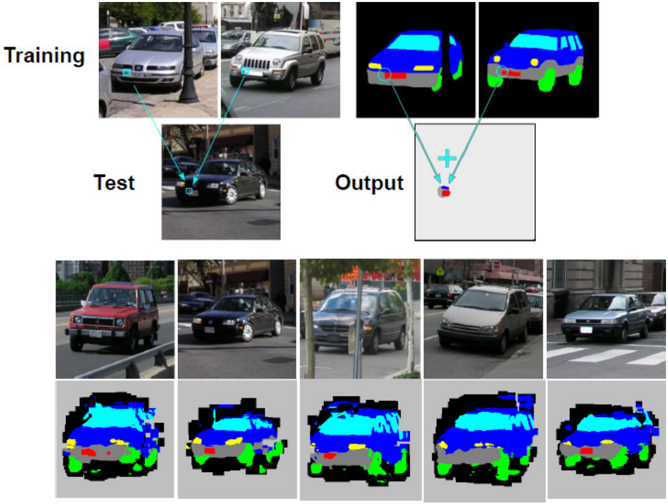
Implicit Shape Model: Example

2nd Hypothesis

Implicit Shape Model: Example

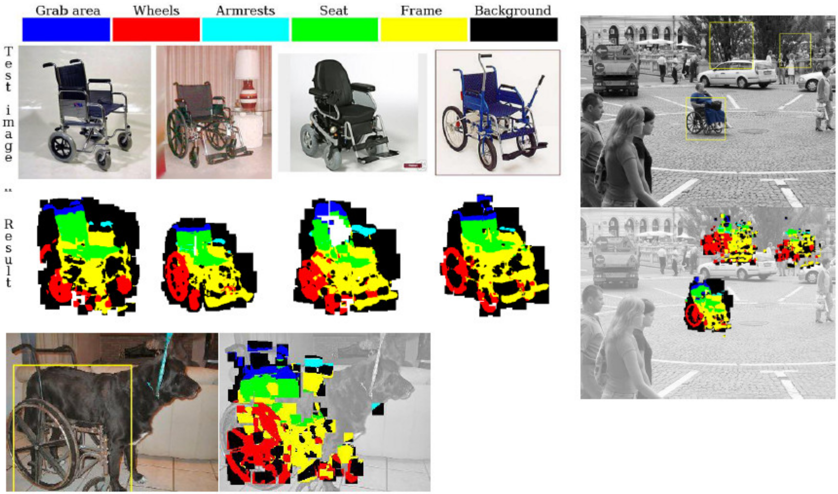
3rd Hypothesis

Implicit Shape Model: Predicting other Modalities



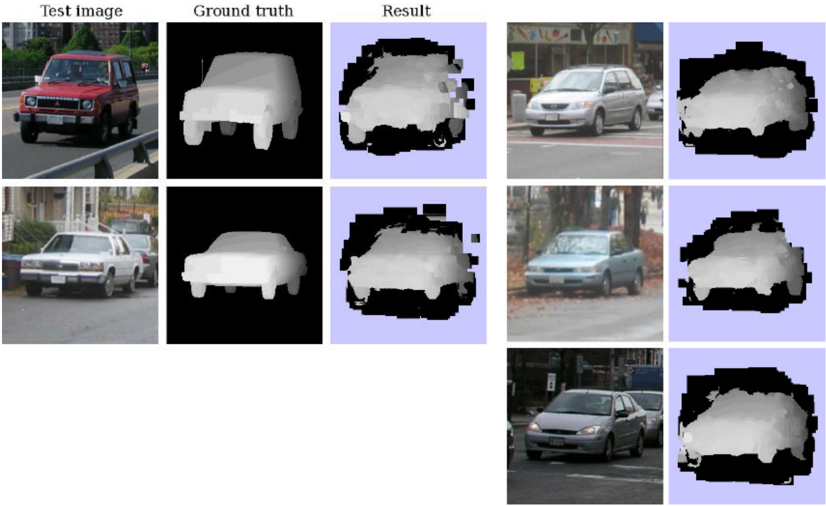
Vote for Semantic Labels

Implicit Shape Model: Predicting other Modalities



Vote for Semantic Labels

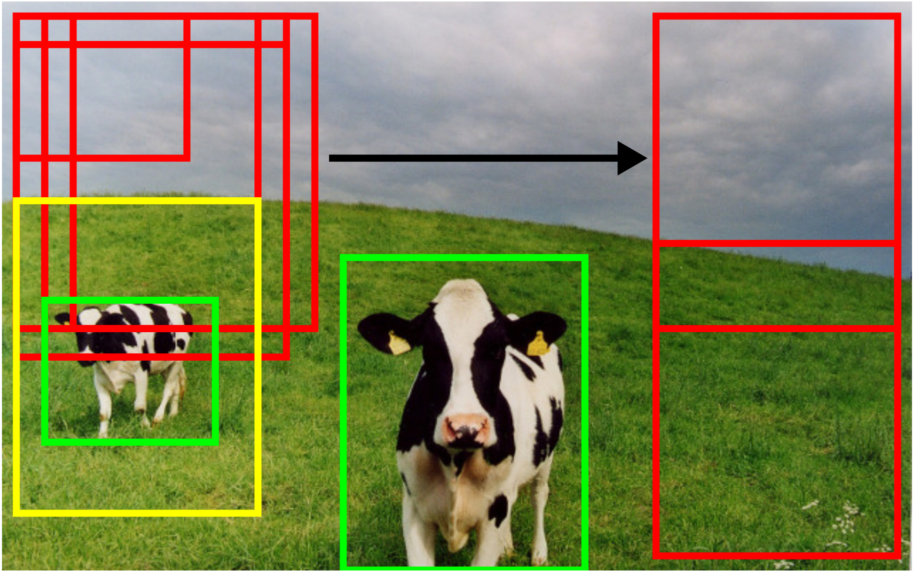
Implicit Shape Model: Predicting other Modalities



Vote for Depth

Sliding Window Detection

Sliding Window Object Detection



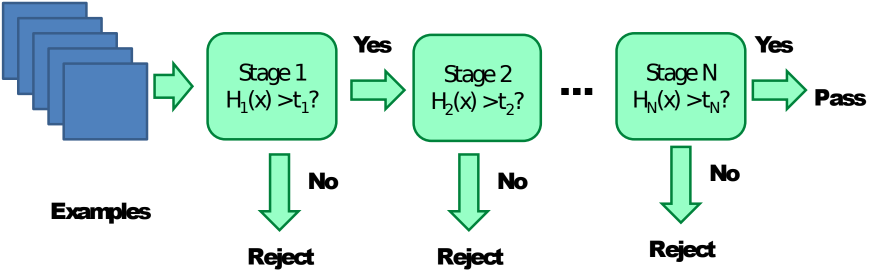
Viola-Jones Face Detector

Face Detection

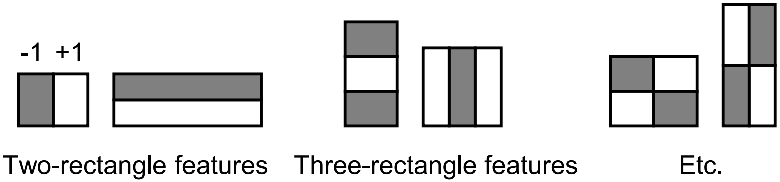
Viola-Jones Detector

Viola Jones Face Detection: [Viola and Jones, 2001]

- Sliding window detector
- Idea 1: Use features/classifiers that are very fast to compute
- Idea 2: Quickly reject unlikely windows by cascade of decision



Haar Features

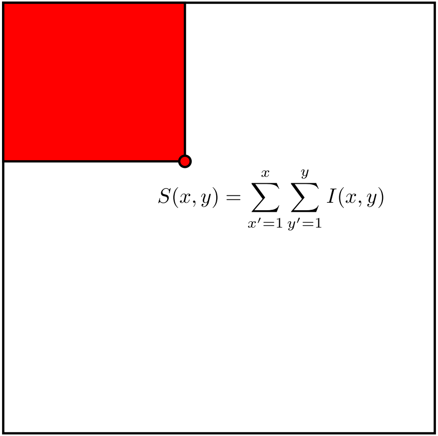


Haar Features:

- Differences of sums of intensities
- Large pool of possible features (24×24 window $\Rightarrow 160k$ features)
- Very fast to calculate! Why?

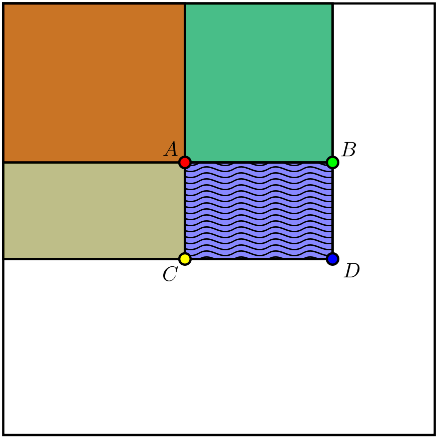
Integral Images

- The integral image computes a value $S(x, y)$ at each pixel which is the sum of the pixel values above and to the left
- This can be quickly computed in one pass through the image



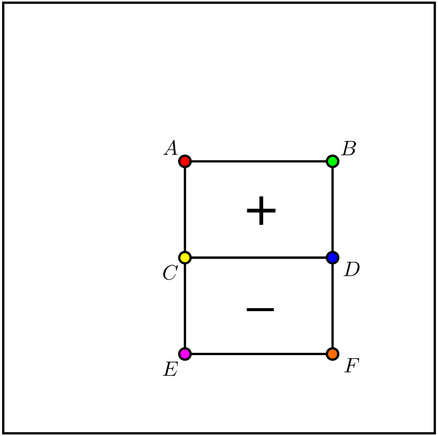
Integral Images

- Given the integral image, how can we quickly calculate the sum of pixels within an arbitrary rectangle?
- Consider the integral values at the four corners of the rectangle
- We have: $D-(B-A)-(C-A)-A = D-B-C+A$
- Only 3 operations required for any rectangle!
- $\Rightarrow$ Fast at all scales!



Integral Images

- So how about this simple Haar feature?



Adaboost – Algorithm

Given: Dataset $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ with labels $y_i \in \{0, 1\}$

1. Initialize weights $\mathbf{w}$ uniformly
2. For $t = 1 \ldots T$ do:
   - 2.1 Normalize weights $\mathbf{w} \leftarrow \mathbf{w}/\bar{\mathbf{w}}$
   - 2.2 Train a weak classifier ($f_j(\mathbf{x})$ = feature j evaluated on image $\mathbf{x}$)

     $$h_j(\mathbf{x}) = [p_j\, f_j(\mathbf{x}) > p_j\, \theta_j]$$

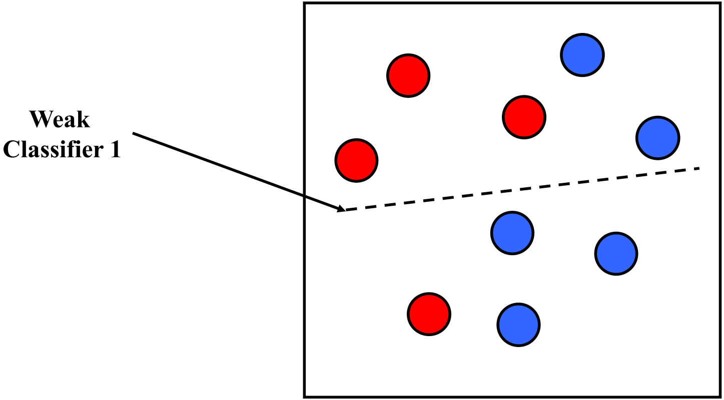
     per feature dimension j wrt. the weighted 0/1 error (loss):

     $$E_j \leftarrow \sum_i w_i |h_j(\mathbf{x}_i) - y_i|$$

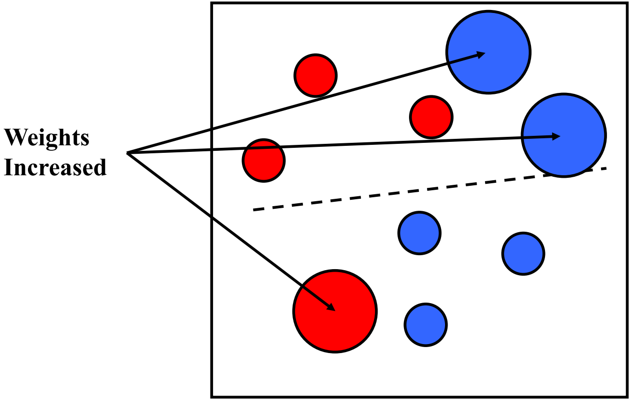
   - 2.3 Choose the classifier h_t^* with the lowest error E_t^*
   - 2.4 Update weights $w_i \leftarrow w_i \cdot \beta_t^{1-e_i}$ where $e_i = 0$ if $\mathbf{x}_i$ is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{E_t^*}{1-E_t^*}$
3. Final strong classifier:

$$H(\mathbf{x}) = \left[\sum_{t=1}^{T} \log \frac{1}{\beta_t} \cdot h_t^*(\mathbf{x}) \geq \frac{1}{2} \sum_{t=1}^{T} \log \frac{1}{\beta_t} \right]$$

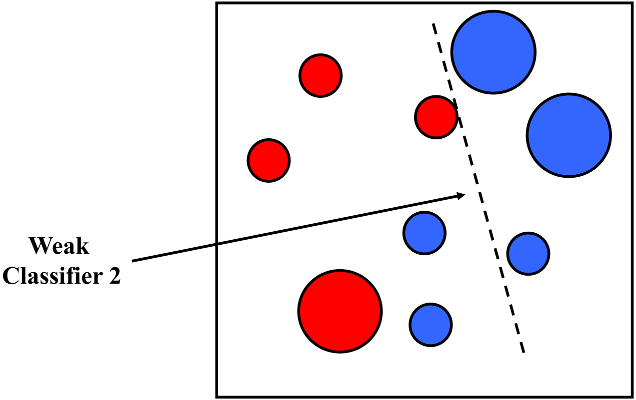
Adaboost – Illustration



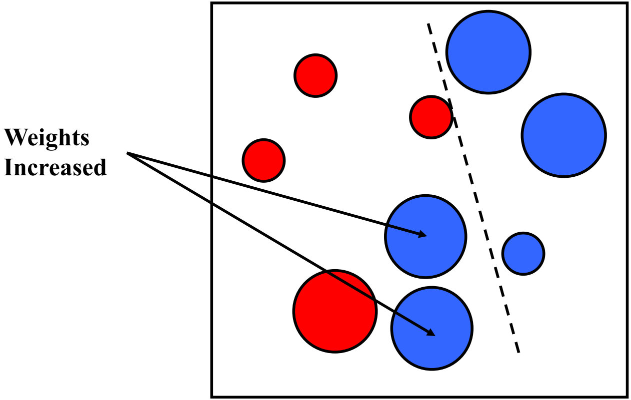
Adaboost – Illustration



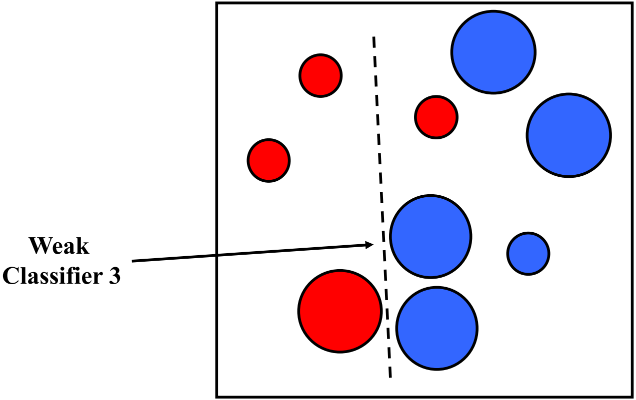
Adaboost – Illustration



Adaboost – Illustration

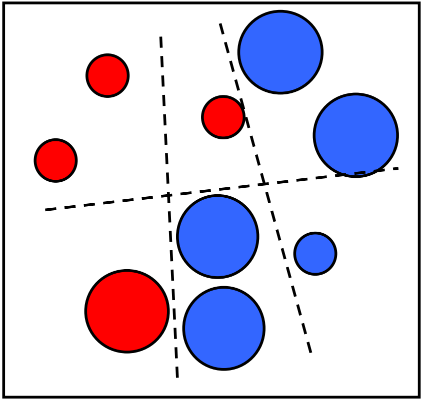


Adaboost – Illustration



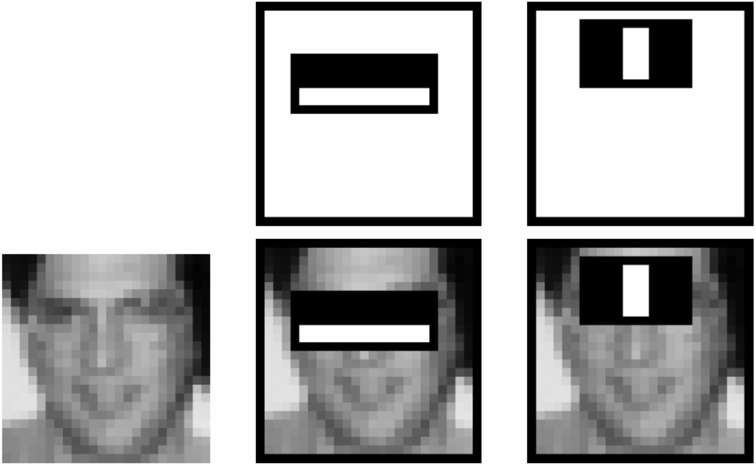
Adaboost – Illustration

Final classifier is a combination of weak classifiers



Feature Selection

The two most important features selected by the Adaboost algorithm:



Boosting vs. SVM

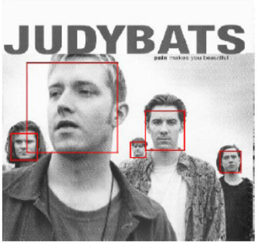
Advantages of Boosting

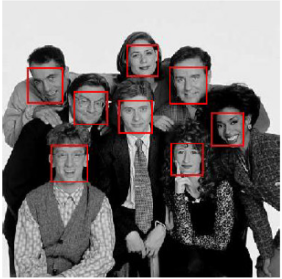
- Feature selection during training
- Flexible in the choice of weak learners / boosting scheme
- Testing is very fast
  (50 ms / 384 $\times$ 288 Px image on Pentium III @ 700Mhz)
- Easy to implement

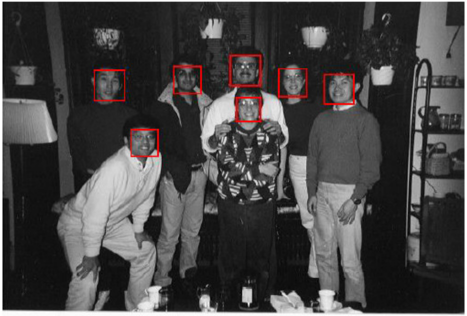
Disadvantages of Boosting

- Many training samples required
- Training is slow
- Performance often a bit worse than SVM

Viola-Jones Detection Results







Histogram of oriented Gradients

Overview

Dalal-Triggs Method [Dalal and Triggs, 2005]:

- Goal: Detect and localize people in images
- Assumption: People are upright and fully visible
- Annotated dataset exists (supervised training)
- Difficulties: Pose+appearance variability, background, illumination
- Simple idea: Combine robust orientation histograms (popular in the context of sparse feature descriptors) with linear SVM classifier



Overview

Detection Phase

Scan image(s) at all scales and locations

↓

Extract features over windows

↓

Run linear SVM classifier on all locations

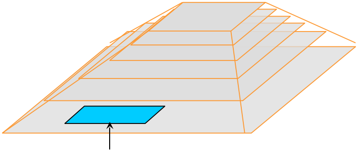
↓

Fuse multiple detections in 3-D position & scale space

↓

Object detections with bounding boxes

Scale-space pyramid

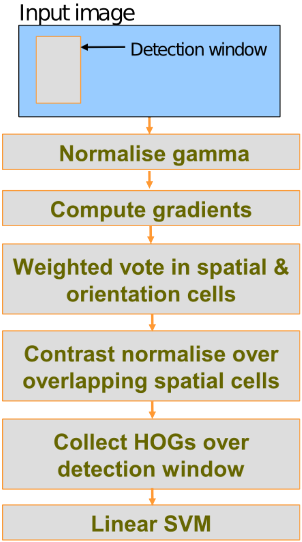


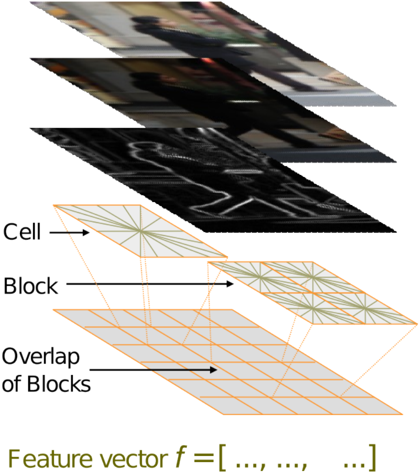
Detection window

Focus on building robust feature sets (static & motion)

5

Feature Extraction





HoG Descriptor

Feature vector
$x = [\dots , \dots , \dots , \dots]$

HoG Features:

- Sliding window of 8×8 Px cells (stride: 8 Px)
- For each cell record distribution of gradients
- Cells combined into $n \times n$ blocks and renormalized
- Why not simply using a pixel intensity-based descriptor?
- Histograms of Gradients are invarieant to slight transformations

HoG Descriptor

Parameters

- Gradient scale
- Orientation bins
- Percentage of block overlap

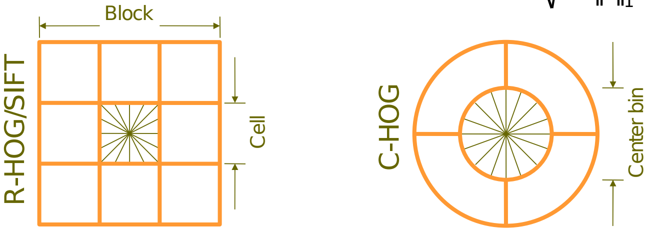
Schemes

- RGB or Lab, colour/gray-space
- Block normalisation

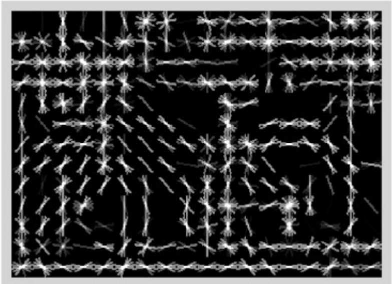
$L2$-norm, $v \leftarrow v/\sqrt{\|v\|_2^2+\varepsilon}$

or

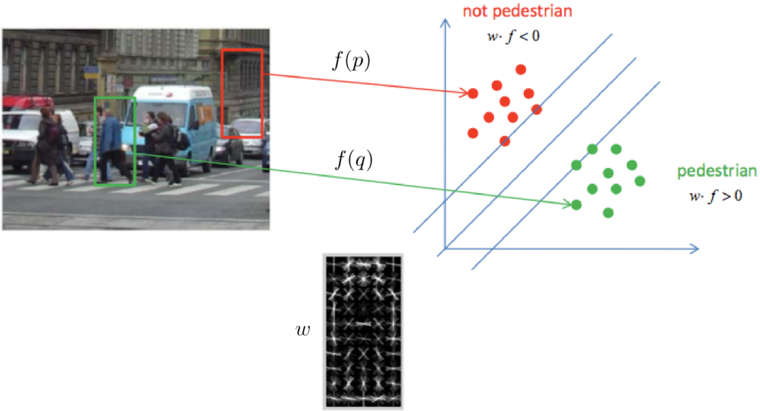
$L1$-norm, $v \leftarrow \sqrt{v/(\|v\|_1+\varepsilon)}$



HoG Descriptor



Linear SVM



Linear Support Vector Machine Classifier:

- Learn a linear SVM classifier from an annotated dataset
- This yields the model parameters (feature weights) w

Hard Example Mining (Search for False Positives)

Learning phase

Input: Annotations on training images

Create fixed-resolution normalised training image data set

Encode images into feature spaces

Learn binary classifier

Resample negative training images to create hard examples

Encode images into feature spaces

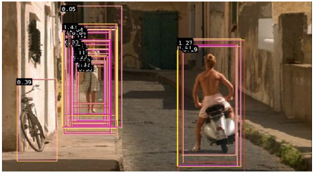
Learn binary classifier

Object/Non-object decision

Retraining reduces false positives by an order of magnitude!

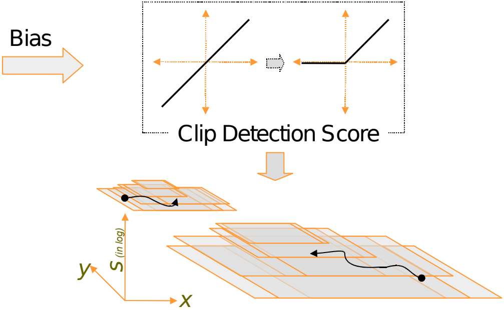
9

Multi-Scale Object Localization



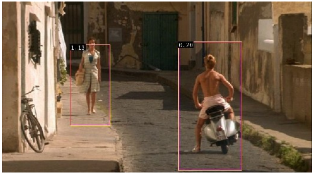
Multi-scale dense scan of detection window

Bias



Threshold

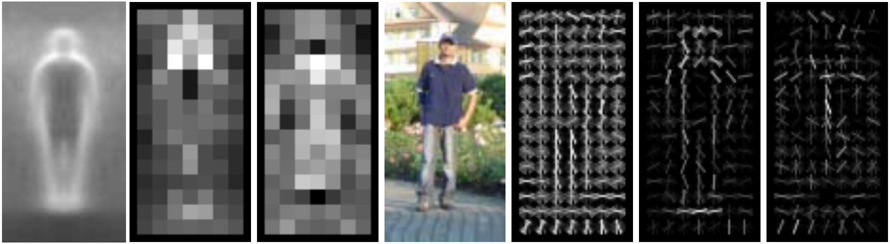
Apply robust mode detection, like mean shift



Final detections

Classification Score Map

HoG Descriptor Weights

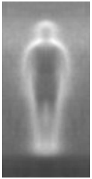


(a) (b) (c) (d) (e) (f) (g)

Figure 6. Our HOG detectors cue mainly on silhouette contours (especially the head, shoulders and feet). The most active blocks are centred on the image background just *outside* the contour. (a) The average gradient image over the training examples. (b) Each "pixel" shows the maximum positive SVM weight in the block centred on the pixel. (c) Likewise for the negative SVM weights. (d) A test image. (e) It's computed R-HOG descriptor. (f,g) The R-HOG descriptor weighted by respectively the positive and the negative SVM weights.

- Most important cues are head, shoulder, leg silhouettes
- Vertical gradients inside a person count negative
- Overlapping blocks around the contour are most important

Datasets

MIT pedestrian database		INRIA person database	
 		 	
Train	507 positive windows Negative data unavailable	Train	1208 positive windows 1218 negative images
Test	200 positive windows Negative data unavailable	Test	566 positive windows 453 negative images
Overall 709 annotations+ reflections		Overall 1774 annotations+ reflections	

Importance of Cell Size

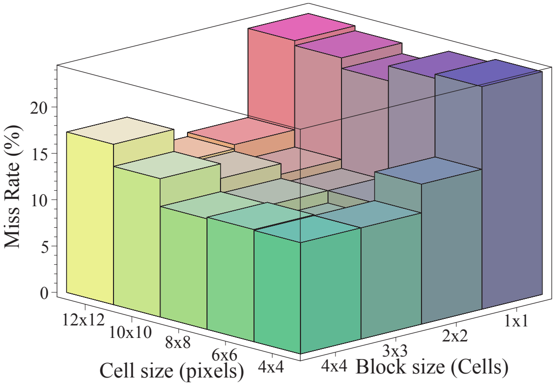
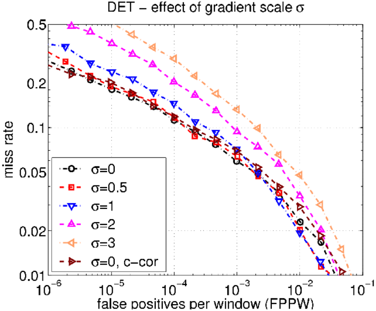


Figure 5. The miss rate at 10^{-4} FPPW as the cell and block sizes change. The stride (block overlap) is fixed at half of the block size. 3×3 blocks of 6×6 pixel cells perform best, with 10.4% miss rate.

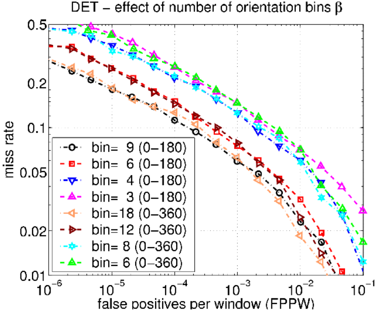
Influence of Parameters

Gradient smoothing, σ



Reducing gradient scale from 3 to 0 decreases false positives by 10 times

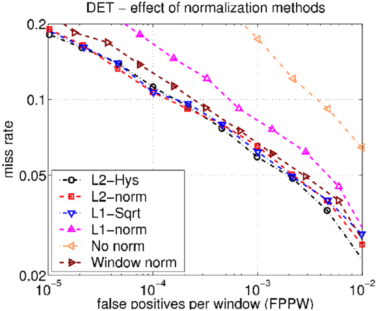
Orientation bins, β



Increasing orientation bins from 4 to 9 decreases false positives by 10 times

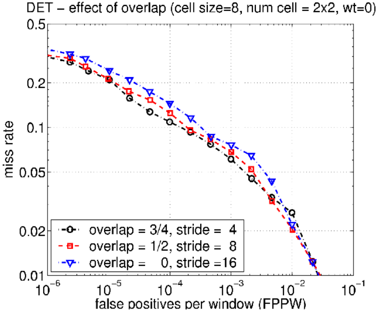
Influence of Parameters

Normalisation method



Strong local normalisation is essential

Block overlap



Overlapping blocks improve performance, but descriptor size increases

Results

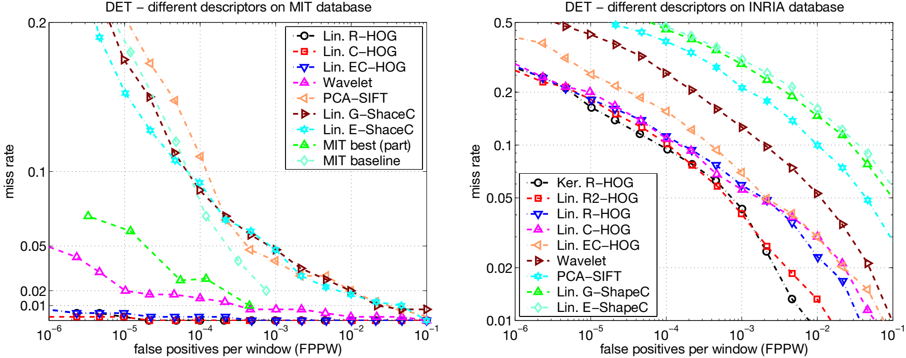
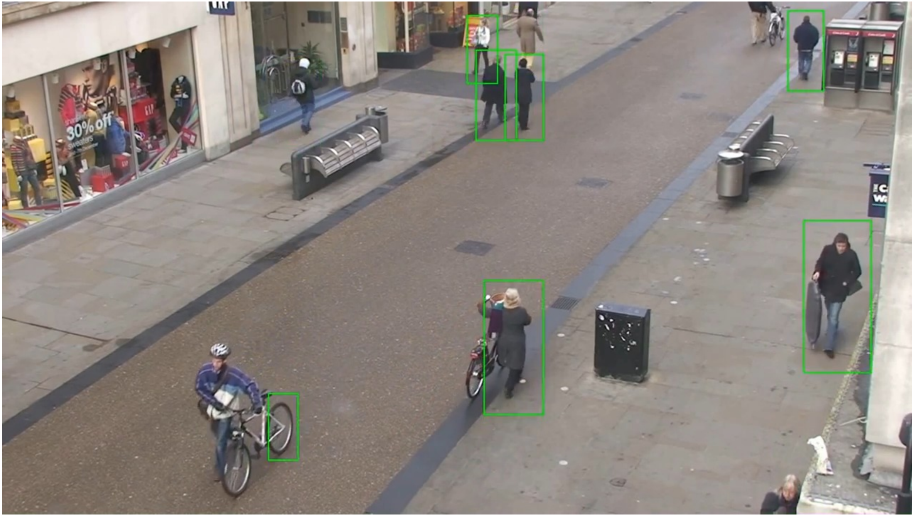


Figure 3. The performance of selected detectors on (left) MIT and (right) INRIA data sets. See the text for details.

Results



More Results ...

Failure Cases


(a) Unusual articulation (b) Difficult contrast (c) Occlusion (d) Person carrying goods

149 missing detections on INRIA people dataset:

- 44 due to difficult contrast & backgrounds
- 43 due to occlusion & carried bags
- 37 due to unusual articulations
- 18 due to over-/underexposed images
- 7 due to images at wrong scale

Failure Cases

(e) Detection on parts (f) Too large scale (g) Detection on vertical structures (h) Cluttered background (i) Missing annotation

149 false positives on INRIA people dataset:

- 54 due to vertical structure / street signs
- 31 due to cluttered background
- 28 due to too small scale (only body parts)
- 24 due to too large scale detections
- 12 due to people that are not annotated :-)

HOGgles

When do HoG features fail? [Vondrick et al., 2013]

HOGgles: Visualizing Object Detection Features

Carl Vondrick Aditya Khosla Tomasz Malisiewicz Antonio Torralba
Massachusetts Institute of Technology

Oral presentation at ICCV 2013

We introduce algorithms to visualize feature spaces used by object detectors. The tools in this paper allow a human to put on "HOG goggles" and perceive the visual world as a HOG based object detector sees it.

Check out this page for a few of our experiments, and read our paper for full details. Code is available to make your own visualizations.

Quick Jump:

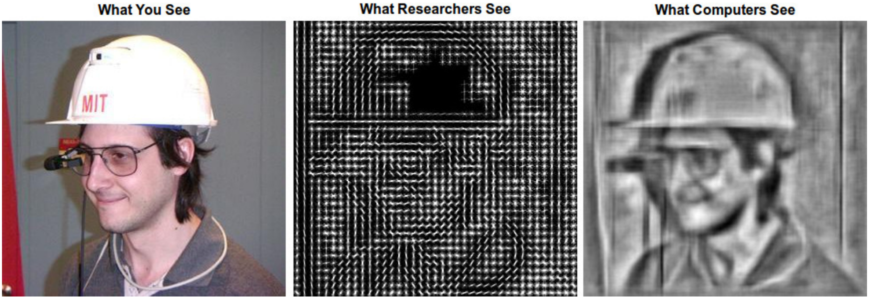
1. Code
2. Overview
3. Why did my detector fail?
4. Visualizing Top Detections
5. What does HOG see?
6. Eye Glass
7. Visualizing Learned Models
8. Recovering Color
9. Videos
10. HOGgles

Download Paper

Read about it in the MIT news!
Download slides or watch

http://web.mit.edu/vondrick/ihog/

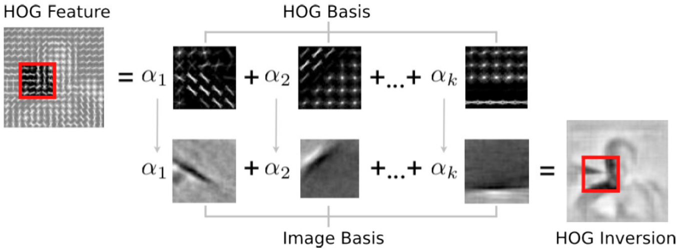
HOGgles



HOGgles:

- Tool to visualize (high-dimensional) feature spaces
- Idea: Invert feature descriptors back to a natural image
- Provides intuitions about object detection features

HOGgles

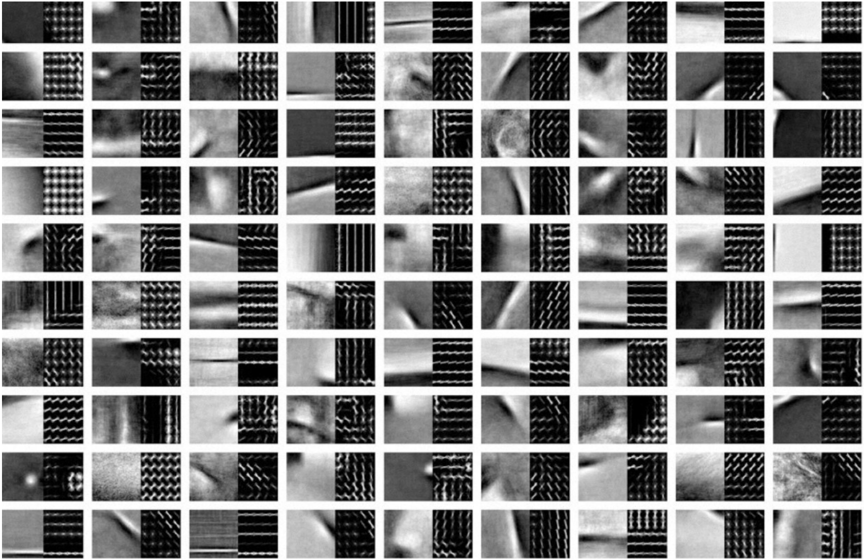


- Jointly learn a coupled basis of HoG features and natural images
- At test time:
  - Project HoG vector onto a HoG basis
  - Transfer coefficients to image basis
  - Reconstruct natural image

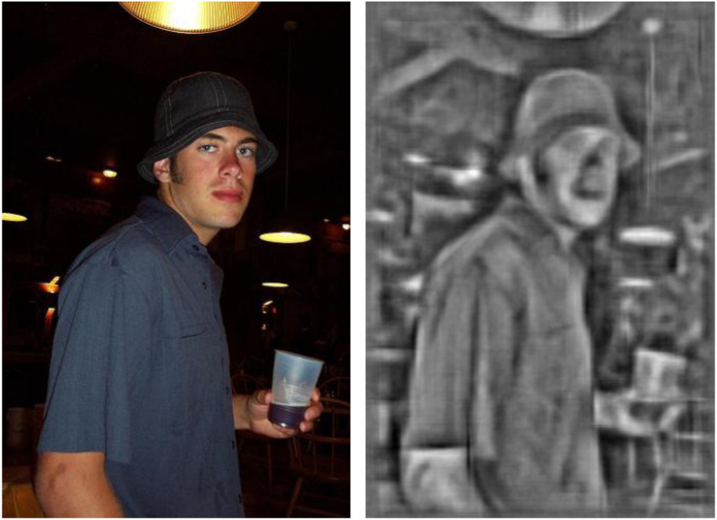
$\mathbf{x} = \mathbf{U}\boldsymbol{\alpha}$ $\quad \mathbf{f} = \mathbf{V}\boldsymbol{\alpha}$ $\quad$ $\mathbf{x}$: image patch, $\mathbf{f}$: HoG feature vector

$\boldsymbol{\alpha}^* = \underset{\boldsymbol{\alpha}}{\text{argmin}} \|\mathbf{V}\boldsymbol{\alpha} - \mathbf{f}\|_2^2$ $\quad$ $\mathbf{U}, \mathbf{V}$: linear bases, $\boldsymbol{\alpha}$: coefficients

HOGgles



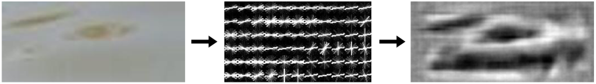
HOGgles



(a) Human Vision (b) HOG Vision

HOGgles

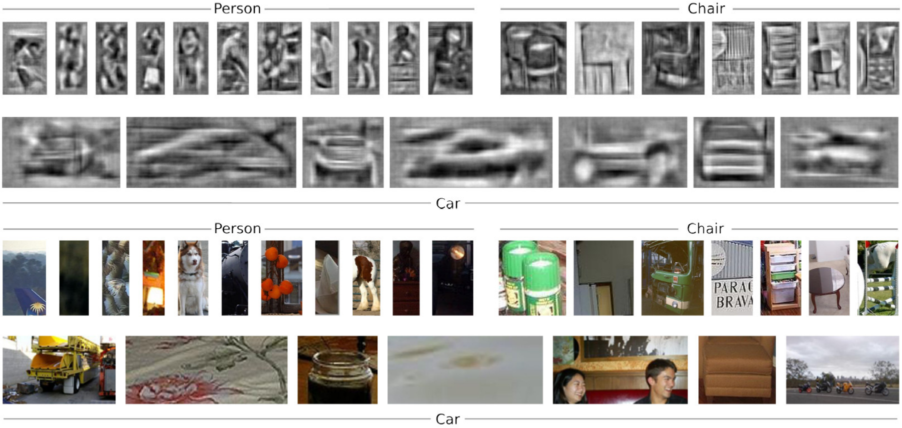
How many cars do you see in this image?



Car Detection HOG Features Our Visualization

HOGgles

Which of these high scoring detections are false alarms?



HOGgles

What do we loose by using the HoG representation?