ELEC 2885:
Image Processing and Computer Vision

Artificial Intelligence in computer vision.

christophe.devleeschouwer@uclouvain.be

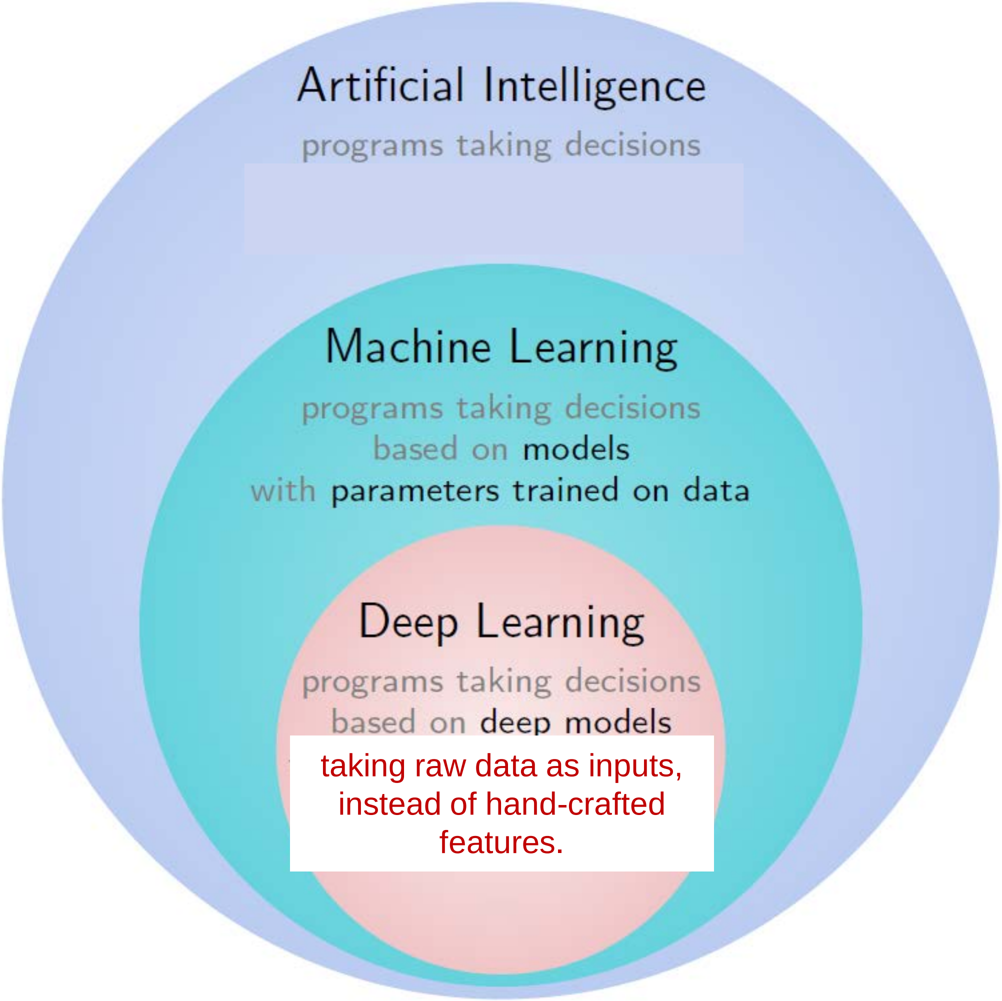
Outline

- **Introduction**
  - **Terminology: AI / ML / DL;**
  - AI4Vision:
    - From pixels to semantic;
    - Two paradigms: (AI or ML) vs. DL.

- Hand-crafted image features
  - Pixel- or patch-based features;
  - Interpreting the features:
    - AI: 3D Geometry;
    - AI: Graph-based topology;
    - ML: Random ferns, AdaBoost, Support Vector Machines.

- DL in vision : Convolutional Neural Networks (CNNs)
  - Paradigm shift & revolution;
  - CNNs basics;
  - CNNs heads & backbones: network architecture examples;
  - CNNs limitations.

Terminology



Artificial Intelligence

programs taking decisions

Machine Learning

programs taking decisions
based on models
with parameters trained on data

Deep Learning

programs taking decisions
based on deep models

taking raw data as inputs,
instead of hand-crafted
features.

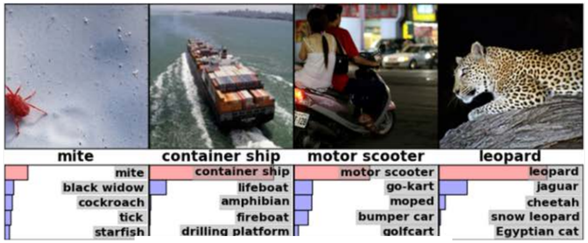
Outline

- **Introduction**
  - Terminology: AI / ML / DL;
  - **AI4Vision:**
    - From pixels to semantic;
    - Two paradigms: (AI or ML) vs. DL.

- Hand-crafted image features
  - Pixel- or patch-based features;
  - Interpreting the features:
    - AI: 3D Geometry;
    - AI: Graph-based topology;
    - ML: Random ferns, AdaBoost, Support Vector Machines.

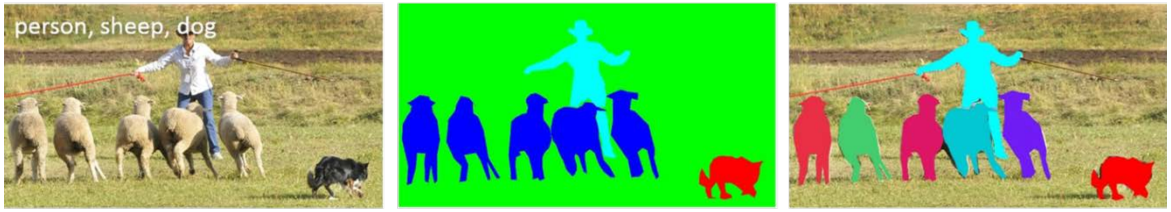
- DL in vision : Convolutional Neural Networks (CNNs)
  - Paradigm shift & revolution;
  - CNNs basics;
  - CNNs heads & backbones: network architecture examples;
  - CNNs limitations.

AI4vision: a variety of tasks



Classification

Detection



Class or instance segmentation

Pose estimation

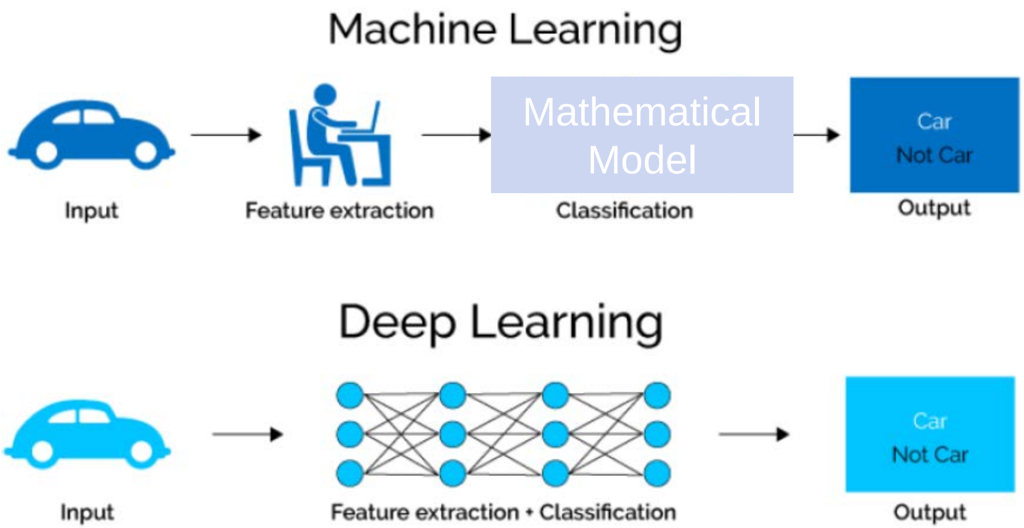
AI4vision: implementation

- General principle: from pixels to semantic
  - collect pixel-wise features (foreground mask, HoG, local binary features, feature map CNN);
  - build on topology / geometry / *training* to turn low-level local information into high-level semantic knowledge.

- Two paradigms

Hand-crafted features

End-to-end learning



Outline

- Introduction
  - Terminology: AI / ML / DL;
  - AI4Vision:
    - From pixels to semantic;
    - Two paradigms: (AI or ML) vs. DL.

- **Hand-crafted image features**
  - **Pixel- or patch-based features;**
  - Interpreting the features:
    - AI: 3D Geometry;
    - AI: Graph-based topology;
    - ML: Random ferns, AdaBoost, Support Vector Machines.

- DL in vision : Convolutional Neural Networks (CNNs)
  - Paradigm shift & revolution;
  - CNNs basics;
  - CNNs heads & backbones: network architecture examples;
  - CNNs limitations.

Image pixel features

- Color components
- Gradient vector (see ‘Segmentation’ lecture)
- Foreground label

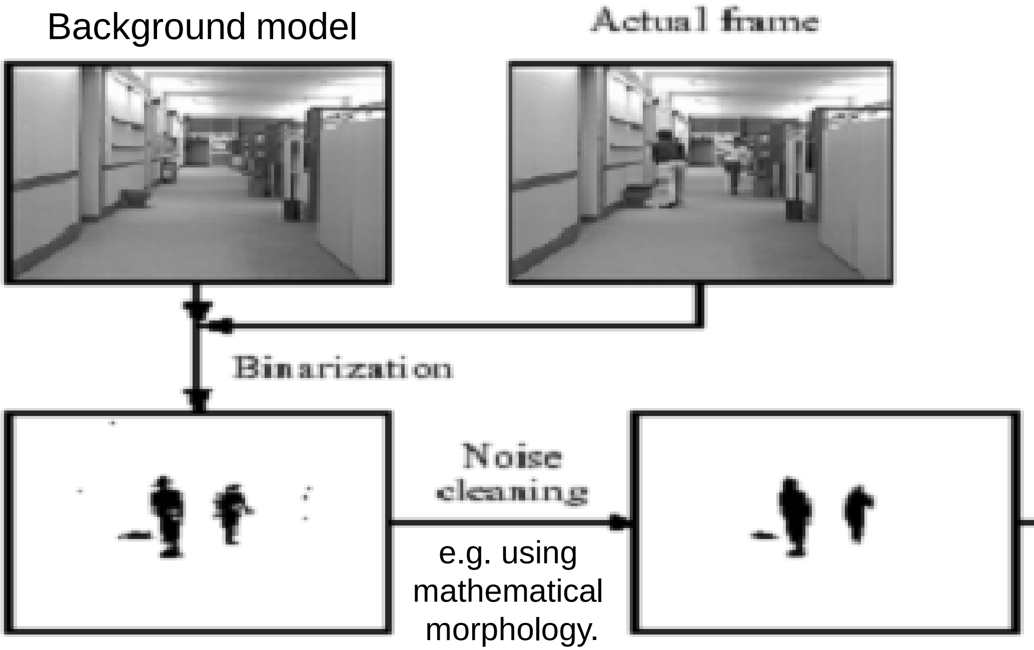
Foreground pixels in video

- Key assumption: still cameras !!!
  - Realistic in many practical scenarios.

- Principle:
  - Background pixel is estimated as the most frequent observation among past ones;
  - Foreground pixel detected if the observed pixel does not fit the background model.

- Key advantage:

  low complexity.


Background model

e.g. using mathematical morphology.

Background model

Exponentially weighted moving average :

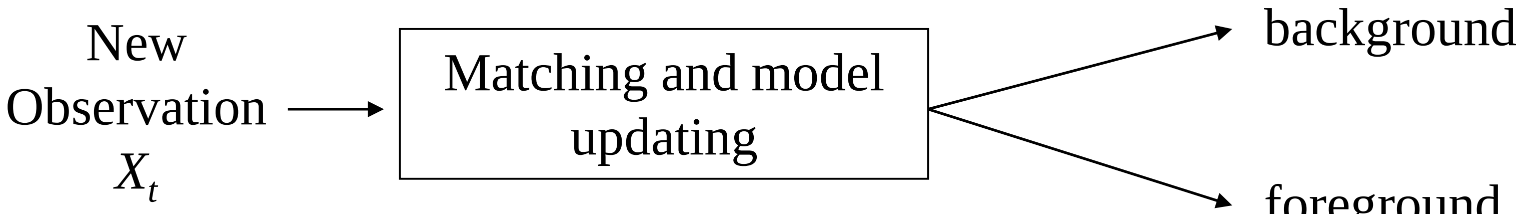
- Construct a background image B as average of few images, e.g. using an exponentially weighted moving average: $B(t) = (1-\alpha)\, B(t-1) + \alpha\, I(t)$;
- For each frame I, classify individual pixels as foreground if $|B-I| > \tau$.

Gaussian Mixture Model :

Not covered this year.

(Adaptive background mixture models for real-time tracking, Stauffer and Grimson CVPR98)

- Advantage: is able to model multiple appearances for a background pixel;
- Each pixel is modeled as the sum of K weighted Gaussians, with weight $w_{k,t}$, mean $\mu_{k,t}$, and standard deviation $\sigma_{k,t}$, $k < K = 3$~5.
- The weights reflects the frequency of occurrence. Hence, large weights correspond to background modes ;
- Model is updated adaptively with new observations, and some learning rate α <1.


New Observation X_t

Matching and model updating

background

foreground

○ Matching Criterion

$$\left\| X_t - \mu_{k,t} \right\|^2 < \beta \cdot \sigma^2_{k,t}$$

- If no match found: *foreground*
- If match found:
  - *B first modes with highest* w_k *are background,*

    $$B = argmin_b \left(\sum_{k=1}^{b} \omega_k > T \right)$$

  - *others are foreground.*

○ Update Formula

- Update active (M_k=1) and non-active (M_k=0) weights :

$$w_k = (1-\alpha) \cdot w_k + \alpha \cdot M_k$$

- Update Gaussian for active mode *m*:

Match found:

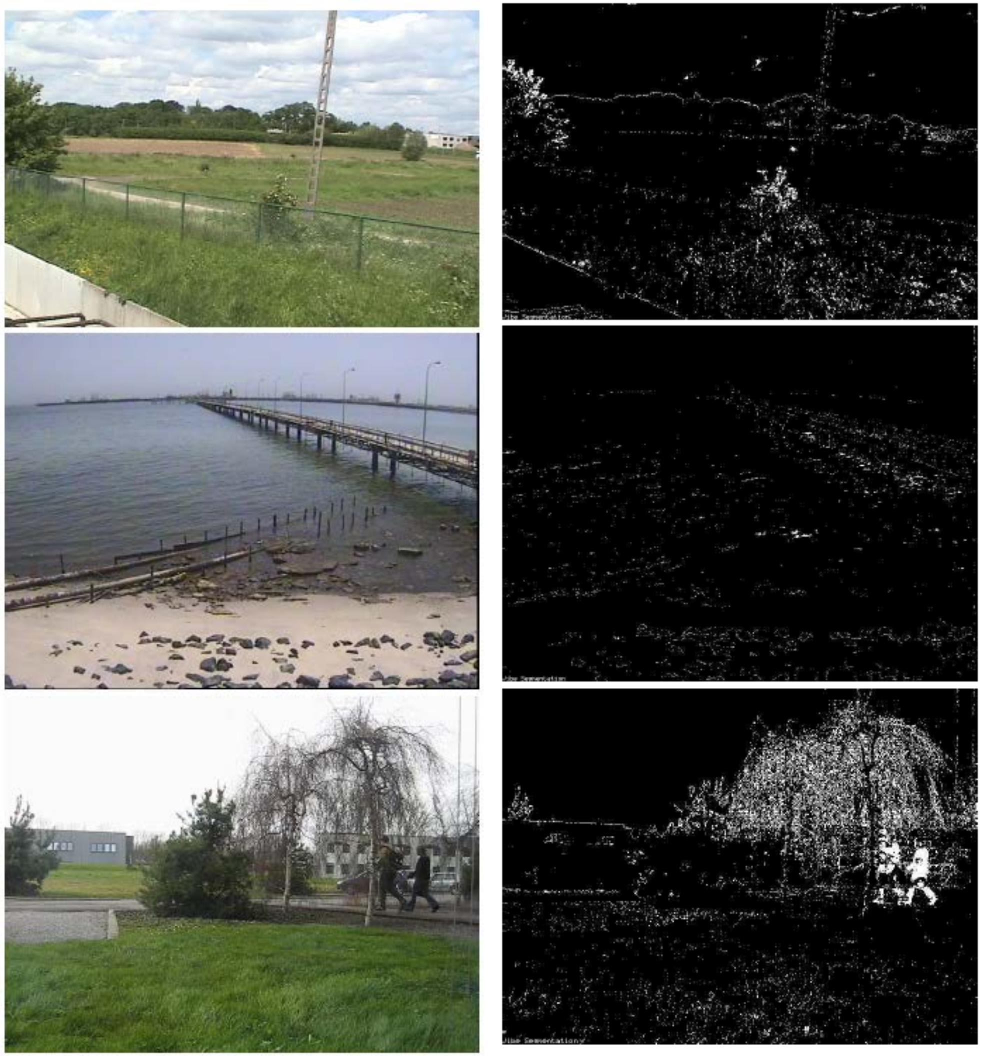
$$\mu_{m,t+1} = (1-\alpha) \cdot \mu_{m,t} + \alpha \cdot X_t$$

$$\sigma^2_{m,t+1} = (1-\alpha) \cdot \sigma^2_{m,t} + \alpha (X_t - \mu_{m,t})^T (X_t - \mu_{m,t})$$

No Match found:

Replace Gaussian with smallest weight by new observation.

Foreground mask weakness



Noise !

- illumination changes;
- shadows;
- motion, e.g.due to wind.

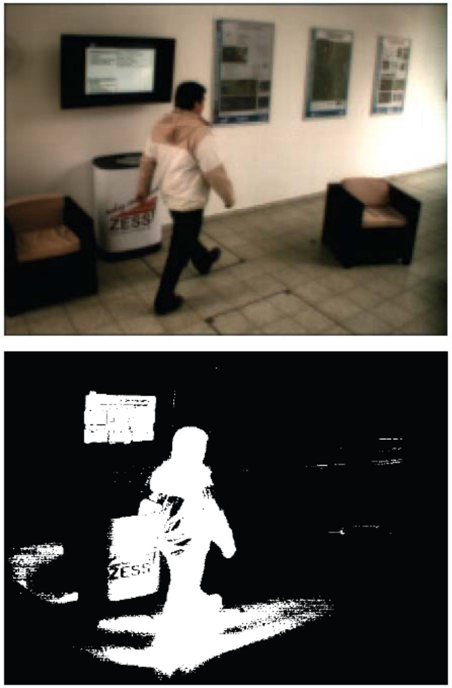
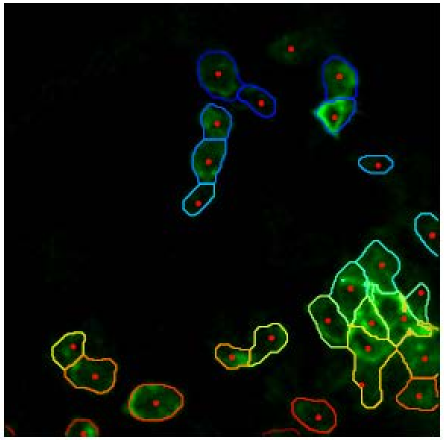
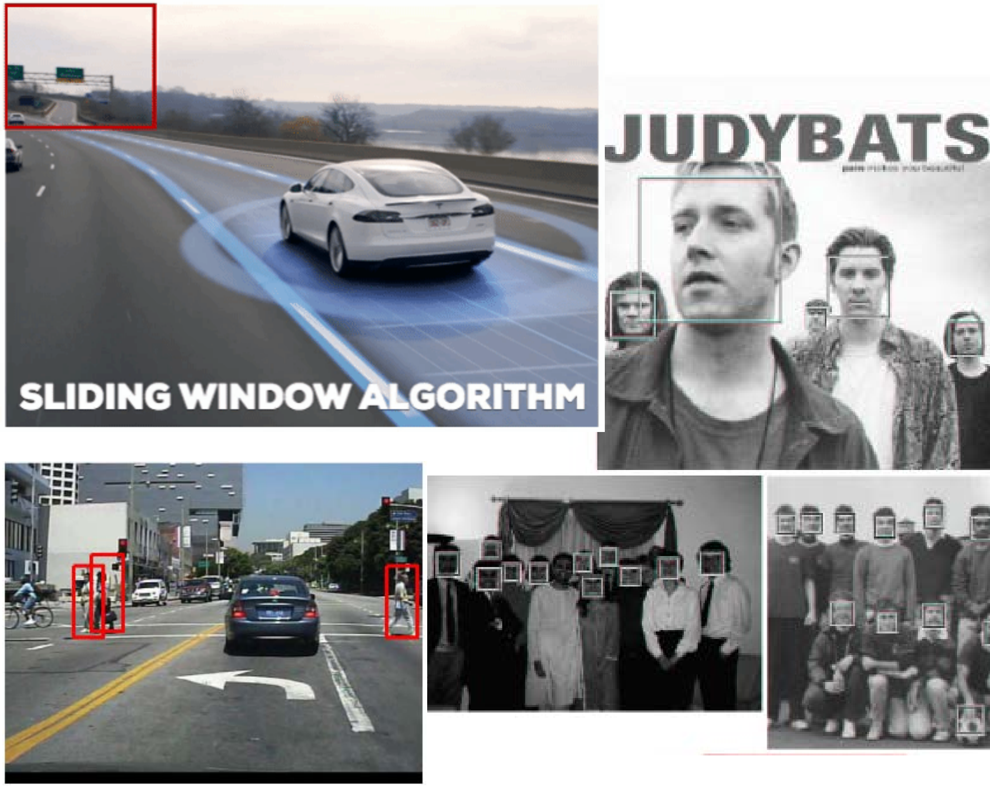


Image patch features

Why image patches ?

The patch might correspond to a <u>pixel</u> neighborhood, and the purpose is to decide whether the pixel lies inside, outsise or between two cells.



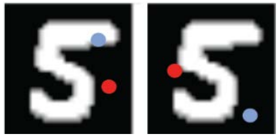


The patch might be a <u>window of semantic interest</u> scanned across the image at multiple scales and locations to detect objects (eg. faces or pedestrians).

https://towardsdatascience.com/evolution-of-object-detection-and-localization-algorithms-e241021d8bad

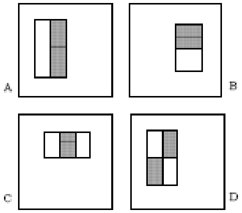
Image patch *binary tests*, considered as weak classifiers

- Pixel comparison (as for BRIEF):

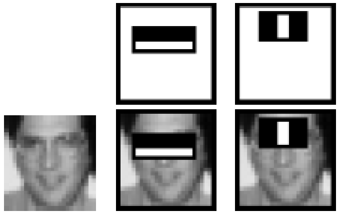


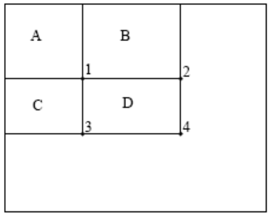
$$b_i = \begin{cases} 1 & \mathit{if}\ I(\bullet) < I(\bullet) \\ 0 & \mathit{else} \end{cases}$$

- Integral image features (as for SURF):



Binary feature $h_j(x)$ defined by selecting a filter and a threshold in a window around pixel x.

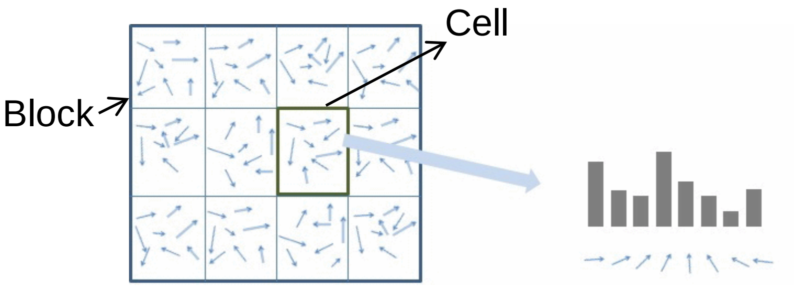




Efficient computation through integral images.

Image patch *feature vectors*, to feed ML models

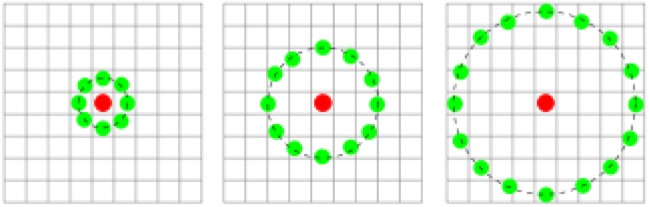
- Histogram of Oriented Gradients



! normalization on blocks of cells.

Histograms of Oriented Gradients for Human Detection, Dalal and Triggs, CVPR2005.

- Local binary patterns:



Histogram of eight-bit vector (256 bins),
each vector being associated to
one pixel in the histogram cell/bock.

Dynamic texture recognition using local binary patterns with an application to facial expressions, Zhao, Guoying, and Matti Pietikainen. IEEE TPAMI, 2007.

Outline

- Introduction
  - Terminology: AI / ML / DL;
  - AI4Vision:
    - From pixels to semantic;
    - Two paradigms: (AI or ML) vs. DL.

- Hand-crafted image features
  - Pixel- or patch-based features;
  - **Interpreting the features:**
    - **AI: 3D Geometry;**
    - AI: Graph-based topology;
    - ML: Random ferns, AdaBoost, Support Vector Machines.

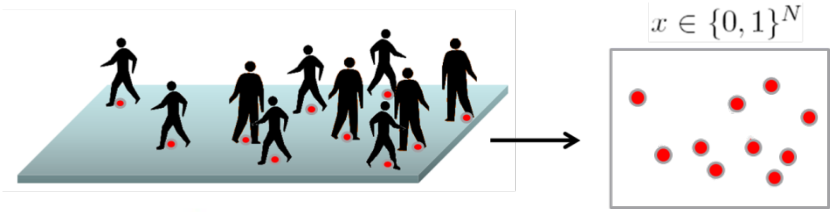
- DL in vision : Convolutional Neural Networks (CNNs)
  - Paradigm shift & revolution;
  - CNNs basics;
  - CNNs heads & backbones: network architecture examples;
  - CNNs limitations.

Detecting objects from foreground mask

- Assumption: ***Prior knowledge about the 3D shape of the object + camera calibration information***.

- Principle = Detect from object silhouette models :
  - Ground occupancy map: Measure the likelihood that an object is located in a ground plane position X.
  - The contribution of one view to the ground occupancy map in position X is obtained by integrating the foreground mask over the projected silhouette of the (expected) 3D object in X.

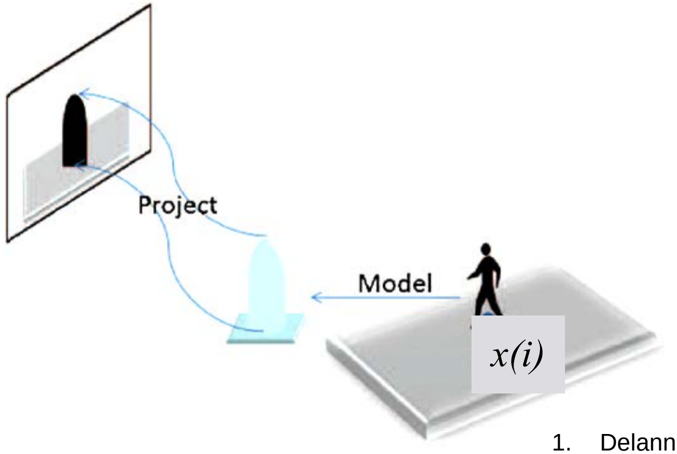
- This interpretation allows for
  - o An efficient implementation, in which accumulation is based on integral images [1].
  - o An inverse problem formulation to reconstruct the sparse ground occupancy map [2].



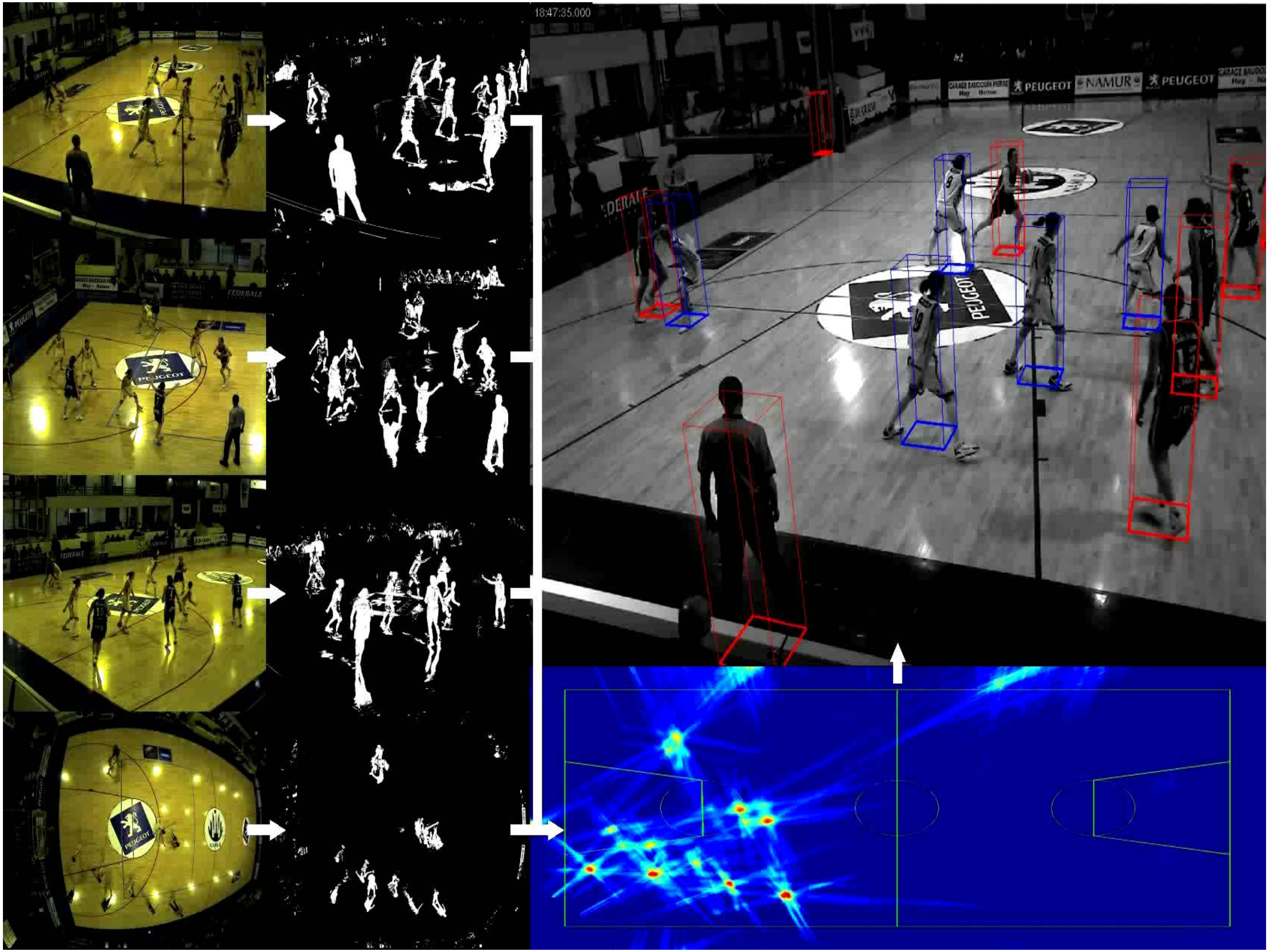
$$\arg\min_{x\in\{0,1\}^N} \|x\|_0 \quad \text{s.t.} \quad \|y - Q(Dx)\|_2^2 < \varepsilon$$

$$\arg\min_{x\in\{0,1\}^N} \|y - Q(Dx)\|_2^2 \quad \text{s.t.} \quad \|x\|_0 < \varepsilon_p$$

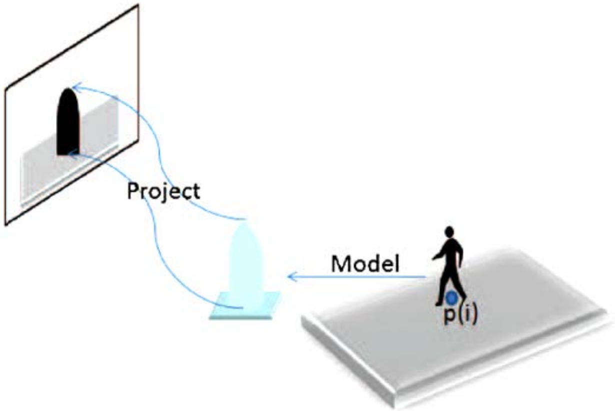
y = foreground mask;
D = silhouette projection operator;
Q = quantization (because silhouettes are not additive)



1. Delannay and al., Detection and recognition of sports (wo)men from multiple views, ICDSC 2009.
2. Alahi and al., Sparsity driven people localization with a heterogeneous Network of Cameras, JMIV, 2009.
3. I. Matthews, P. Carr, Y. Sheikh, Monocular Object Detection Using 3D Geometric Primitives, ECCV 2012.

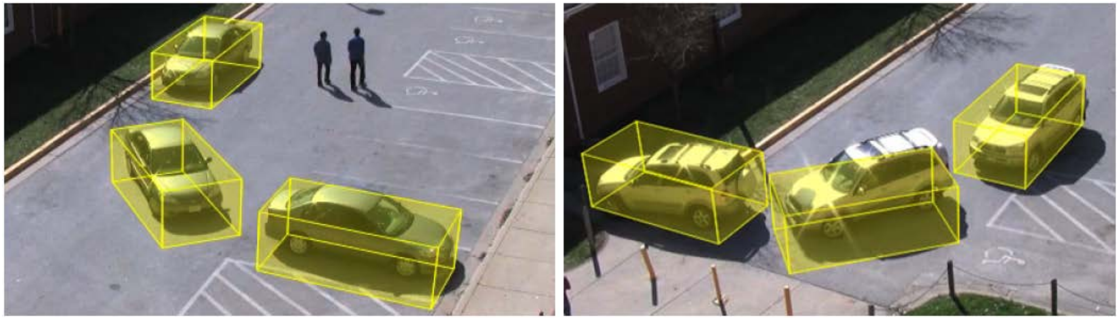


VIDEO



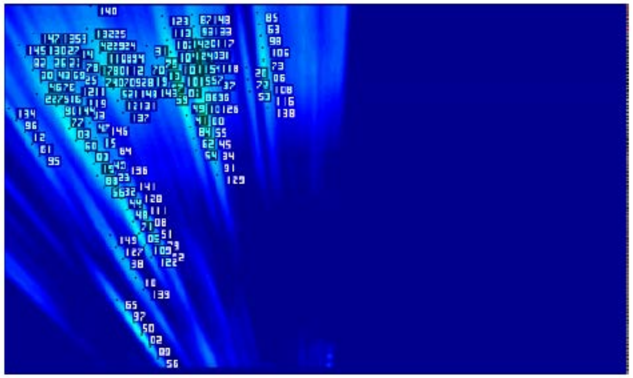
Mature & efficient but…

… requires **multiple models** for non-isotropic 3D objects;



… unreliable in single-view case.

Ground occupancy mask is ambiguous.



Outline

- Introduction
  - Terminology: AI / ML / DL;
  - AI4Vision:
    - From pixels to semantic;
    - Two paradigms: (AI or ML) vs. DL.
- Hand-crafted image features
  - Pixel- or patch-based features;
  - **Interpreting the features:**
    - AI: 3D Geometry;
    - **AI: Graph-based topology;**
    - ML: Random ferns, AdaBoost, Support Vector Machines.
- DL in vision : Convolutional Neural Networks (CNNs)
  - Paradigm shift & revolution;
  - CNNs basics;
  - CNNs heads & backbones: network architecture examples;
  - CNNs limitations.

Towards semantic with graphs

- Mathematical morphology and Watershed (see ‘Segmentation’ lecture)

  → e.g. to clean a foreground mask → e.g. to segment a region of connected pixels with similar color.

- Energy minimization via graph-cuts [Boykov, ECCV 2006]

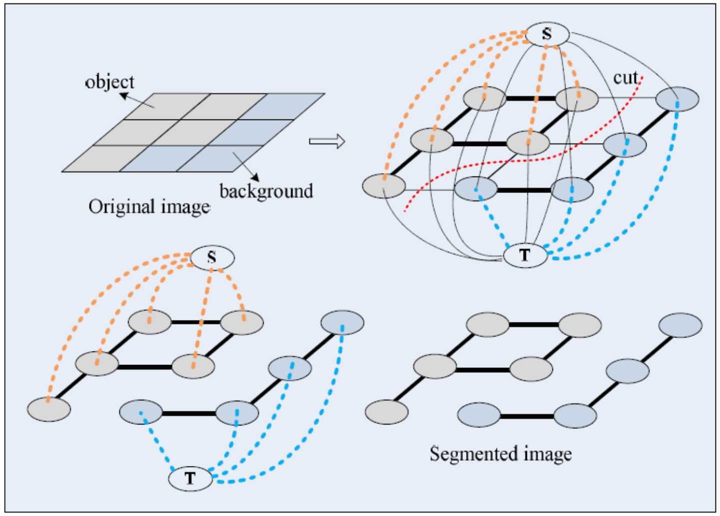
Image segmentation can be regarded as computing a min-cut (or a max flow) in a graph.
The nodes of our graphs represent image pixels. S and T represent ‘object’ and ‘background’ labels (extendable to N labels).
Each pixel is connected to each terminal node (with t-links). Neighbouring pixels are interconnected by edges (n-links).

A s-t cut is a subset of edges such that the terminals S ad T become completely separated in the induced graph.
A minimum s-t cut minimizes the sum of the cost associated to severed edges.
This is equivalent to finding an object/background pixel assignment $A = (A_1, \ldots, A_p, \ldots, A_{|\mathcal{P}|})$ at minimizes :

$$E(A) = \lambda \cdot R(A) + B(A)$$

with $R(A) = \sum_{p \in \mathcal{P}} R_p(A_p)$ *(regional term)*

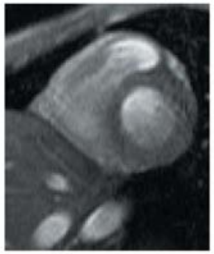
$$B(A) = \sum_{\{p,q\} \in \mathcal{N}} B_{p,q} \cdot \delta_{A_p \neq A_q} \quad \textit{(boundary term)}$$

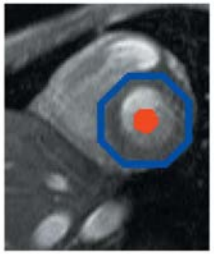


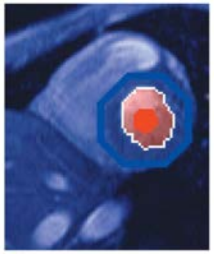
R_p(.) measures how the pixel *p* matches the object/background histogram.
$B_{p,q}$ coresponds to a penalty in case of label discontinuity between *p* and *q*.

For example : $R_p(\text{“obj”}) = -\ln \Pr(I_p|\text{“obj”})$ and $B_{p,q} \propto \exp\left(-\frac{(I_p - I_q)^2}{2\sigma^2}\right)$

! NP-hard problem → approximation via iterative expansion & swap moves
See Y. Boykov, et al., Fast Approximate Energy Minimization via Graph Cuts. IEEE TPAMI, 2001


(a) Original image


(b) Initialization


(c) Segmentation

Outline

- Introduction
  - Terminology: AI / ML / DL;
  - AI4Vision:
    - From pixels to semantic;
    - Two paradigms: (AI or ML) vs. DL.

- Hand-crafted image features
  - Pixel- or patch-based features;
  - **Interpreting the features:**
    - AI: 3D Geometry;
    - AI: Graph-based topology;
    - **ML: Random ferns, AdaBoost, Support Vector Machines.**

- DL in vision : Convolutional Neural Networks (CNNs)
  - Paradigm shift & revolution;
  - CNNs basics;
  - CNNs heads & backbones: network architecture examples;
  - CNNs limitations.

Machine Learning in Vision: outline

- Combining weak binary classifiers (= binary tests):
  - Random ferns (Note: RFs share pros/cons with random trees)
  - AdaBoost

- Discriminating feature vectors :
  - SVM ;
  - Linear Discriminant Analysis. *Not covered this year.*

Binary tests

Combining weak classifiers with **random ferns**.

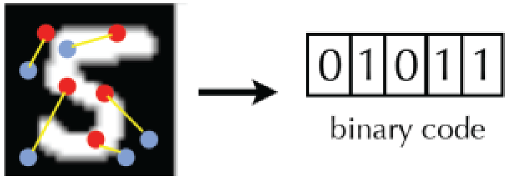
Goal: associate a class label to a patch.

Bayesian approach, with uniform prior:

$$p(c|o) = \frac{p(o|c) \cdot p(c)}{p(o)} \Rightarrow p(c|o) \sim p(o|c)$$

Estimated from training samples

$$\hat{c} = \underset{c \in \{0,1\}}{\operatorname{argmax}} P(C = c | b_1, \ldots, b_N) = \underset{c \in \{0,1\}}{\operatorname{argmax}} P(b_1, \ldots, b_N | C = c)$$



Problem: observation space dimensionality.

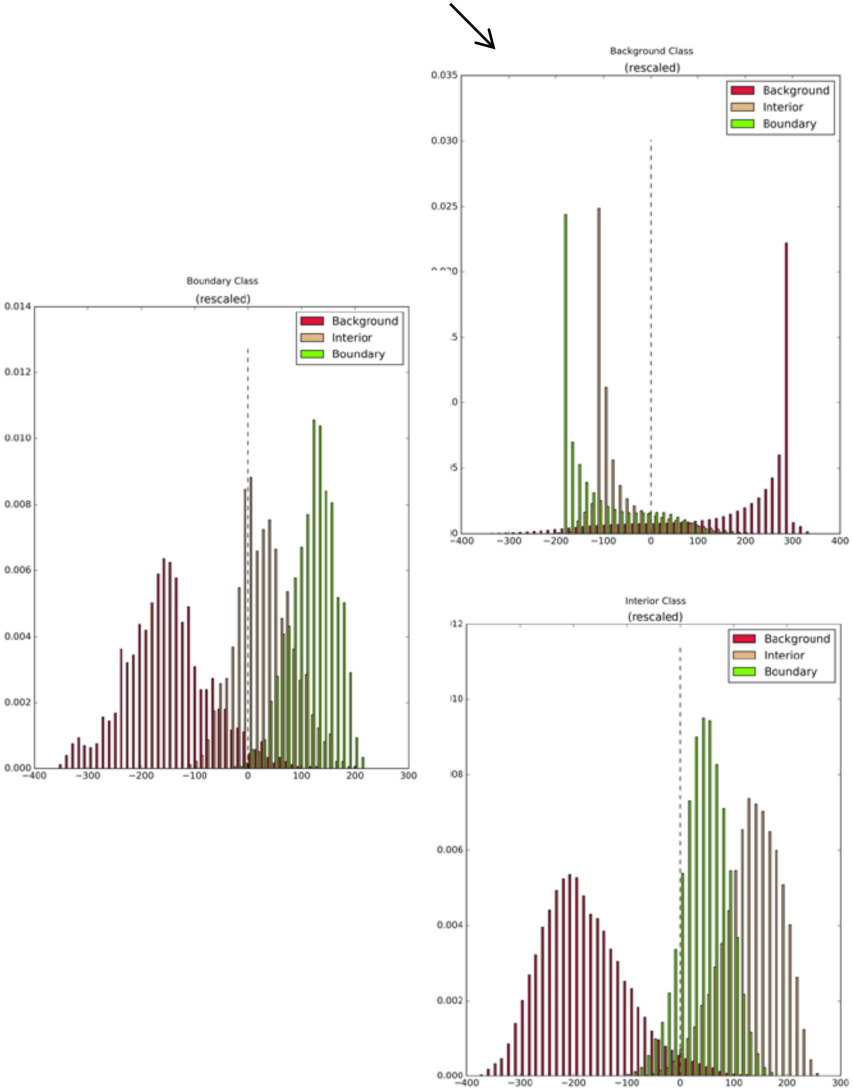
Semi-Naive Bayesian approach: group the *N* binary tests into *M* sets, named ferns, of size *N/M,* and assume independence between ferns:

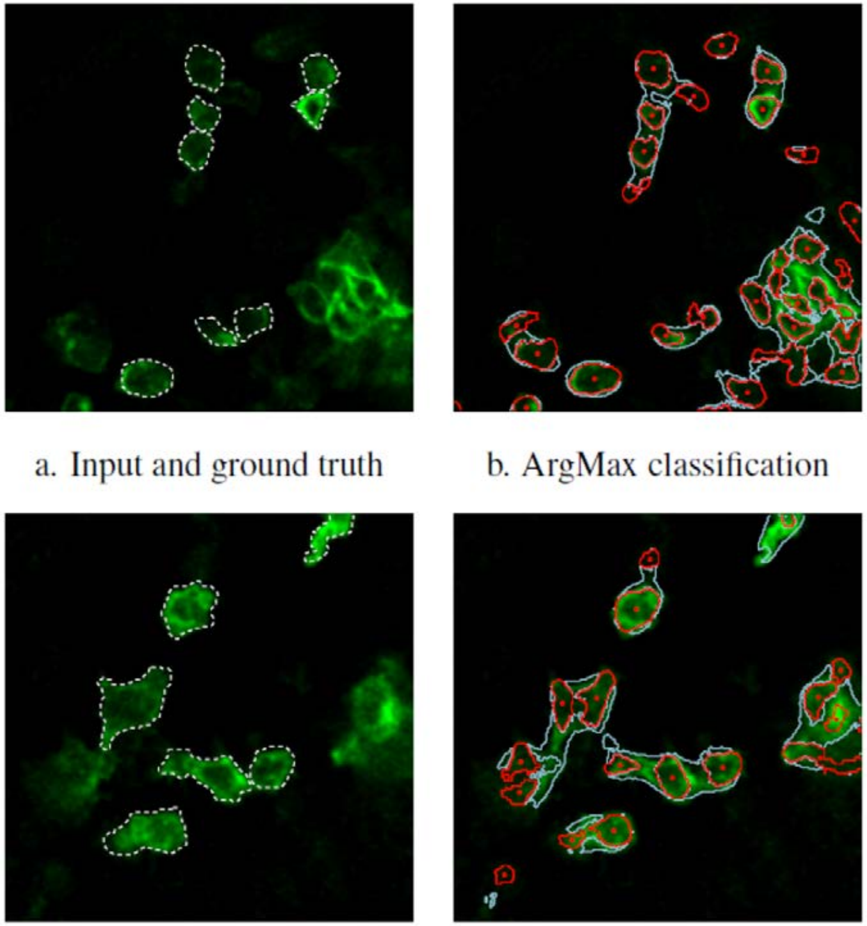
$$P(b_1, \ldots, b_N | C = c) = \prod_{k=1}^{M} P(F_k | C = c)$$

Example : Binary test usage for cell segmentation. https://arxiv.org/abs/1602.05439

Assign a label (interior/boundary/exterior)
to each pixel based on binary tests
in a small squared neighborhood.

Estimated probability (log scale, zero mean)
of being in the background class
for 3 types of pixels

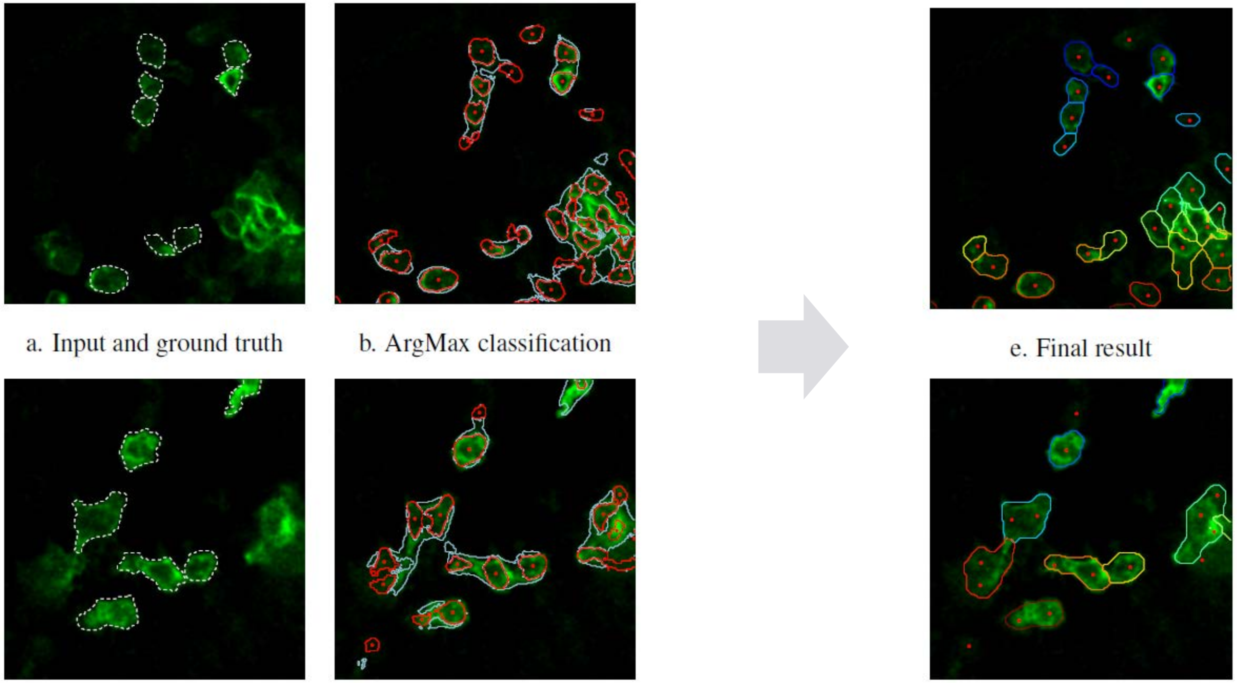




a. Input and ground truth

b. ArgMax classification

Note: Graph cuts can be used to improve argmax segmentation. [Browet, IEEE ICIP 2016]



a. Input and ground truth

b. ArgMax classification

e. Final result

Seeds = centers of argmax interior regions.

One label per seed, and graph-cut to define label assignment f_p to pixel p, so as to minimize an energy function:

$$\overbrace{E(f) = \sum_{p\in\mathcal{P}} D_p(f_p) +}^{D_p(f_p)\text{ is small when seed }f_p\text{ is close to }p\text{ ('inside' path).}} \overbrace{\sum_{(p,q)\in\mathcal{N}} W(p,q)\left(1-\delta(f_p,f_q)\right)}^{\text{Penalizes transitions occuring in non-boundary pixels.}} + \overbrace{\sum_{l\in\mathcal{L}} h_l(f)}^{\text{Penalizes the use of many labels.}}$$

Combining weak classifiers with **Adaboost**

‘Learning object waveforms: Face detection’ Viola & Jones, 2001

Goal: Define $H^T(x)=\sum_{t=1}^{T}\alpha_t\cdot h_t(x)$ to maximize $\sum_i y_i\cdot\mathrm{sign}\left(H^T(x_i)\right)$ when summing over *N* training samples (*x$_i$*,*y$_i$*), with *y$_i$* = 1 or -1.

Trick: Find $H^T(x)$ to minimize $E_T=\sum_i e^{-y_i\cdot H^T(x_i)}$

Exponential in E_T implies that the energy change caused by an additional term in the sum of weak classifiers is larger for wrongly classified samples than for correctly classified ones. At each iteration, this favors the selection of a weak classifier h_m that classifies correctly the samples that are wrongly classified by H^{m-1}.

- Proceed iteratively, knowing $H^{m-1}(x)$,

select $h_m(x)$ and α_m such that $H^m(x)=H^{m-1}(x)+\alpha_m\cdot h_m(x)$ minimizes E_m .

- $$E_m=\sum_i e^{-y_i\cdot H^{m-1}(x_i)}\cdot e^{-y_i\cdot\alpha_m\cdot h_m(x_i)}$$

$$=\sum_{h_m(x_i)=y_i} w^m(i)\cdot e^{-\alpha_m}+\boxed{\sum_{h_m(x_i)\neq y_i} w^m(i)}\,e^{\alpha_m}\qquad\text{with}\quad w^m(i)\equiv e^{-y_i\cdot H^{m-1}(x_i)}$$

To minimize since $e^{\alpha_m} > e^{-\alpha_m}$

- $$h_m=\boxed{\underset{h\in\mathrm{H}}{\mathrm{argmin}}\sum_i w^m(i)\left[h(x_i)\neq y_i\right]}\quad\text{and}\quad\frac{\partial E_m}{\partial\alpha_m}=0\;\rightarrow\alpha_m=\frac{1}{2}\cdot\ln\left(\frac{\sum_{h_m(x_i)=y_i} w^m(i)}{\sum_{h_m(x_i)\neq y_i} w^m(i)}\right)>0$$

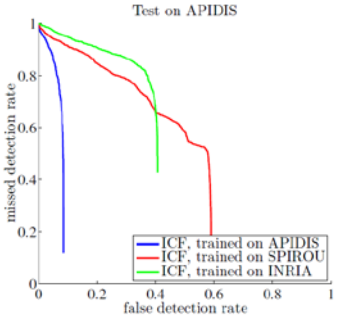
Random subset if H is huge.

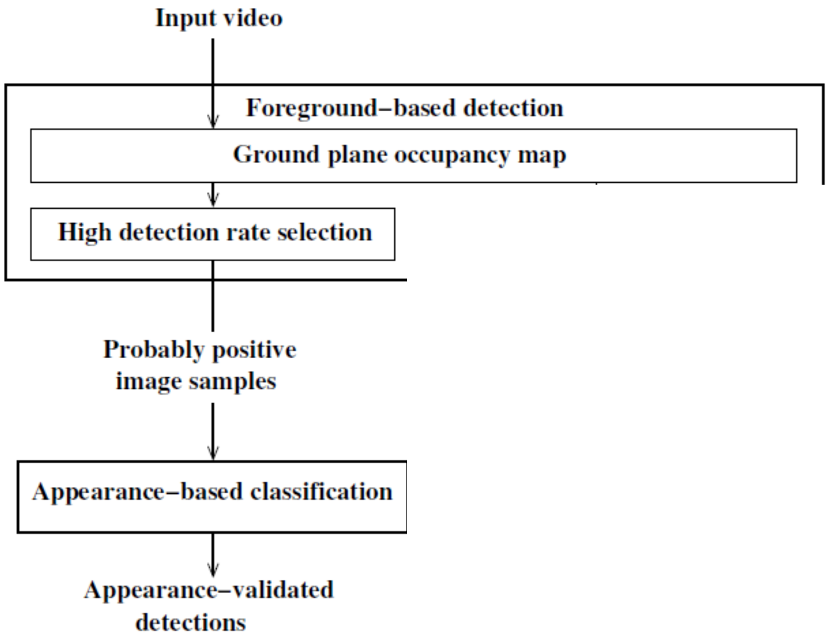
Minimizing the sum requires to select a h_m that classifies correctly samples that are wrongly classified by H^{m-1} , because they have a large $w^m(i)$.

Ensemble of classifiers: Adaboost or Random Ferns ?

- **Key concern: label corruption.**
- Example:

Foreground detector + *SCENE-SPECIFIC* appearance-based classifier.




Input video

Foreground–based detection

Ground plane occupancy map

High detection rate selection

Probably positive image samples

Appearance–based classification

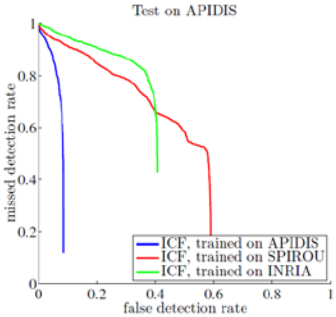
Appearance–validated detections

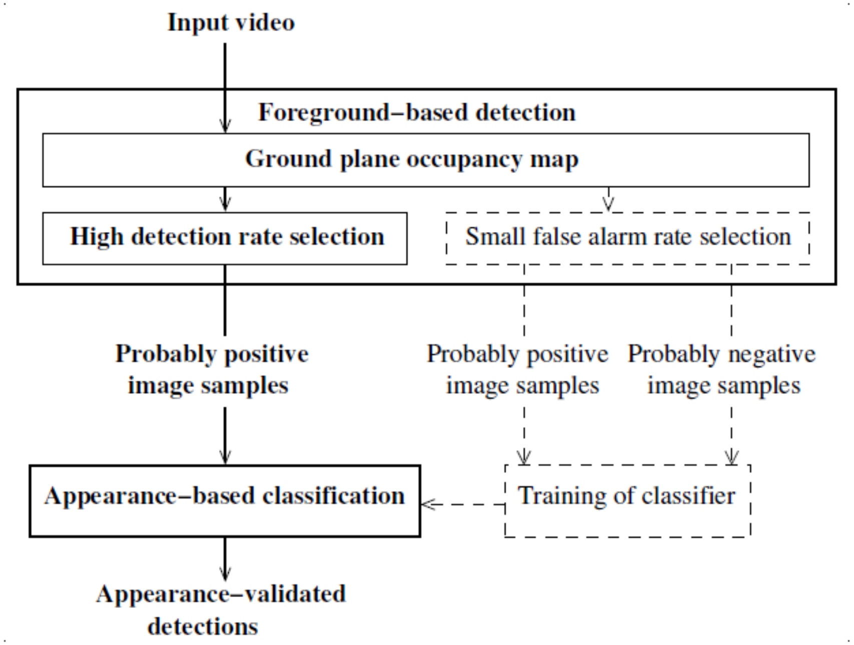
Scene-specific classifier for effective and efficient team sport players detection from a single calibrated camera, CVIU2016.

Ensemble of classifiers: Adaboost or Random Ferns ?

- **Key concern: label corruption.**
- Real-life example:

Foreground detector + *SCENE-SPECIFIC* appearance-based classifier.





Input video

Foreground–based detection

Ground plane occupancy map

High detection rate selection

Small false alarm rate selection

Probably positive image samples

Probably positive image samples

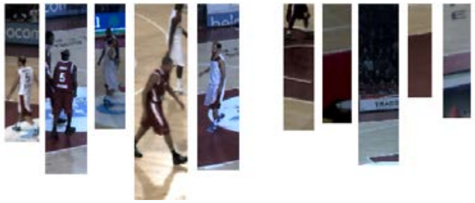
Probably negative image samples

Appearance–based classification

Training of classifier

Appearance–validated detections

To adapt the classifier to the game/scene at hand, the training is based on samples labelled *with high confidence* by the foreground detector.



Scene-specific classifier for effective and efficient team sport players detection from a single calibrated camera, CVIU2016.

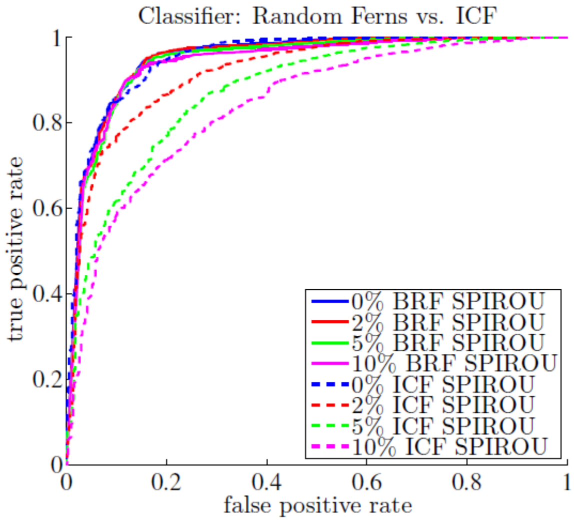
ICF = Integral (color) Channel Features, combined with AdaBoost [BMVC2009; CVPR2012; TPAMI2014].

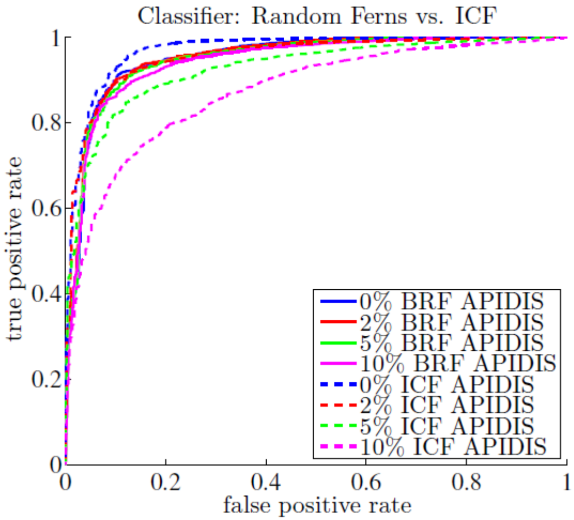
BRF = Block Random Ferns:

Bounding box investigated by the classifier is split in a grid of square blocks;

All the binary tests (= pixels value comparison) in a given ferns are related to the same block.

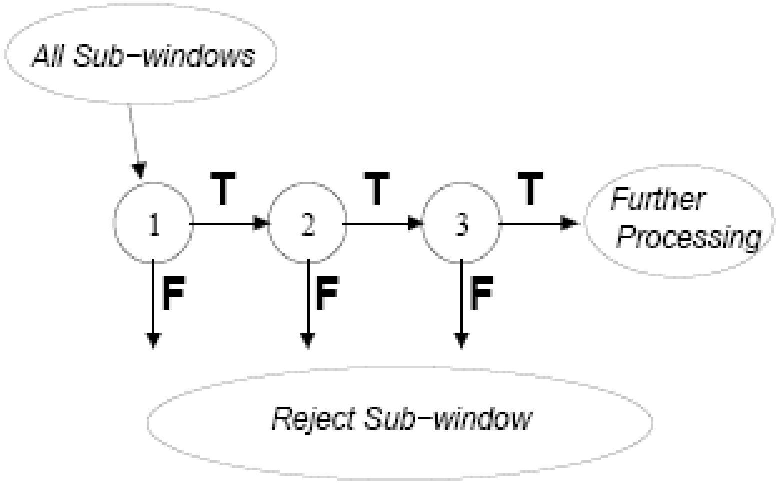
[CVIU2016] : Scene-specific classifier for effective and efficient team sport players detection from a single calibrated camera.





Ensemble of classifiers & **computational complexity**.

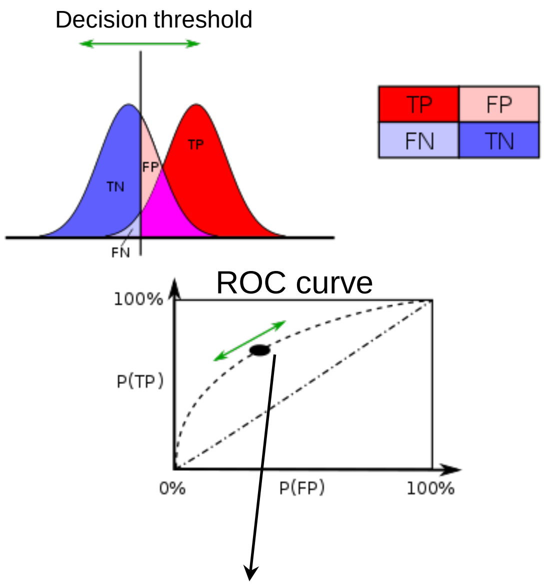
Cascade of classifiers allows for very fast testing:



Schematic depiction of a the detection cascade. A series of classifiers are applied to every sub-window. The initial classifier eliminates a large number of negative examples with very little processing. Subsequent layers eliminate additional negatives but require additional computation. After several stages of processing the number of sub-windows have been reduced radically. Further processing can take any form such as additional stages of the cascade (as in our detection system) or an alternative detection system.

Note: imbalance of training samples. 5000 vs. 350.000.000!

How to select operating points for the binary classifiers in the cascade?



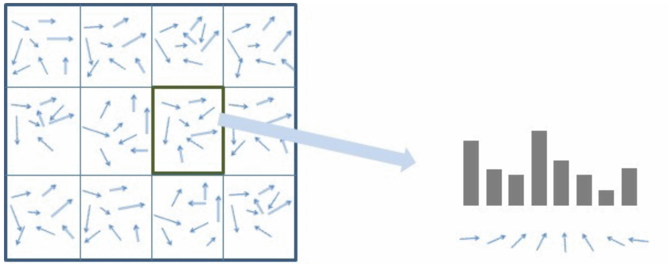
The operating point moves along the ROC curve when changing the classifier decision threshold.

Note: ROC = receiver operating characteristic.

Figure from 'Learning object waveforms: Face detection' Viola & Jones, 2001, and https://en.wikipedia.org/wiki/Receiver_operating_characteristic

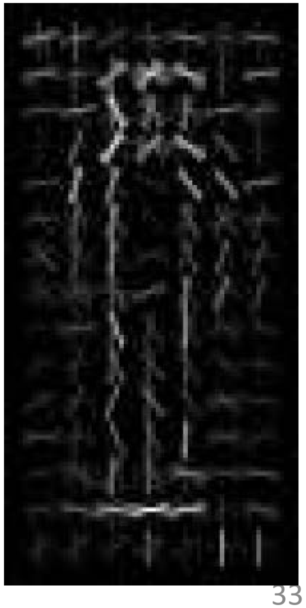
Discriminating feature vectors

- Histogrammes d’orientations de gradients (HoG)



Fast and Accurate Digit Classification, Maji and Malik, 2009.

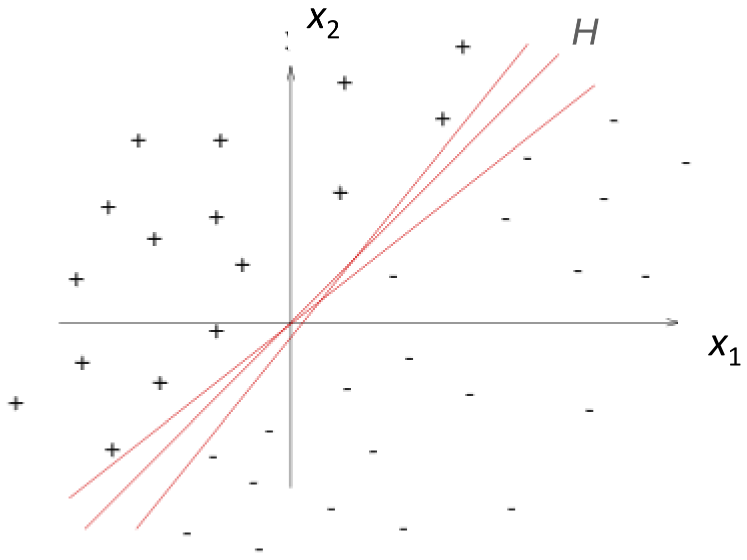
Image window
=
object of a
given class
when gradient
orientations in
window blocks
correspond to the
class model.



Support Vector Machine

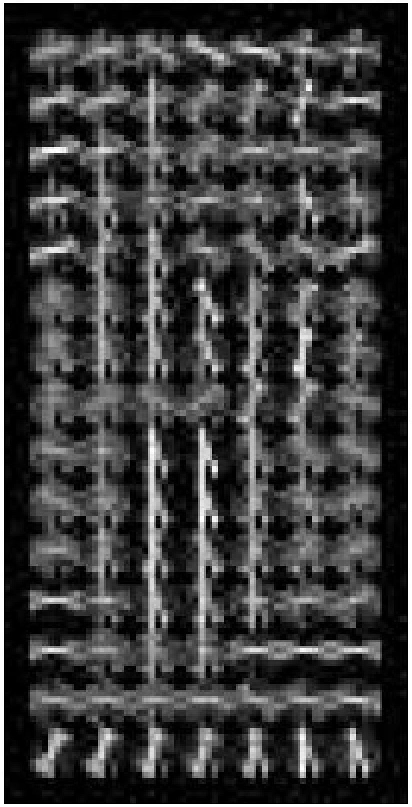
- Linear SVM:

$$\text{decision} = \text{sign}(\mathbf{w}^T \cdot \mathbf{x} + b)$$

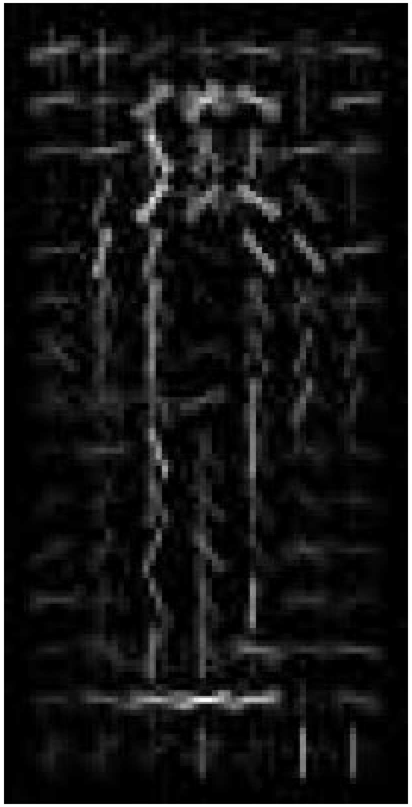


The SVM optimization problem is formulated so as to find the hyperplane that best separate the training data
into their two classes.

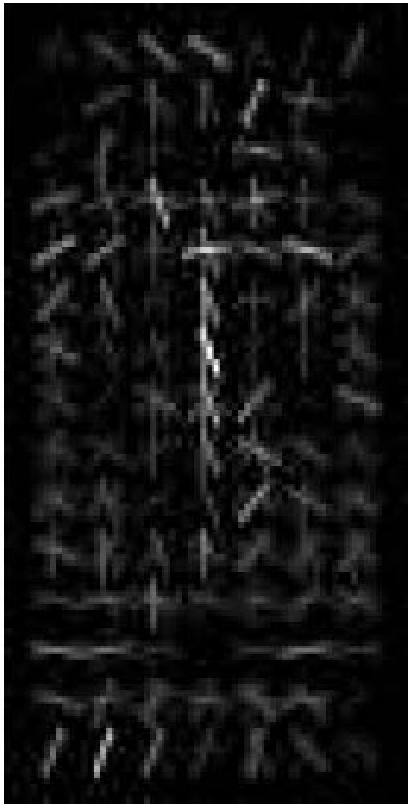
O HOG + Linear SVM :



HoG



HoG components corresponding to positive SVM weights



HoG components corresponding to negative SVM weights