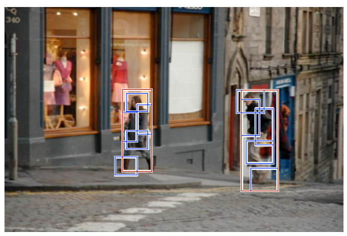
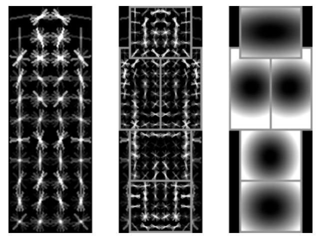
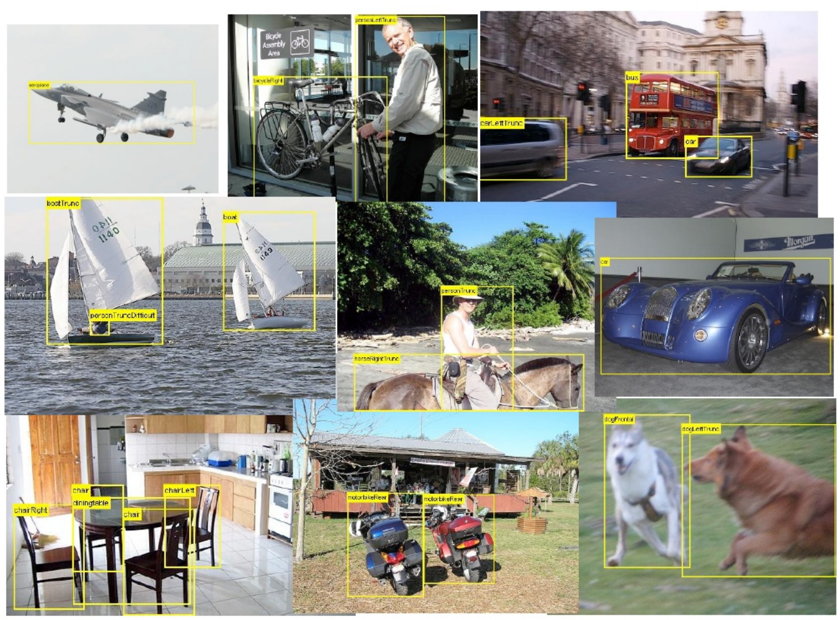
More sliding window detection: Discriminative part-based models





Many slides based on [P. Felzenszwalb]

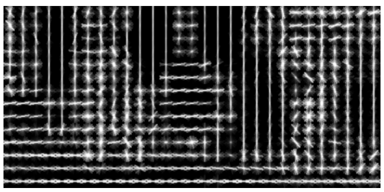
Challenge: Generic object detection



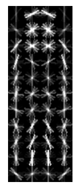
Pedestrian detection

- Features: Histograms of oriented gradients (HOG)
  - Partition image into 8x8 pixel blocks and compute histogram of gradient orientations in each block
- Learn a pedestrian template using a linear support vector machine
  - At test time, convolve feature map with template

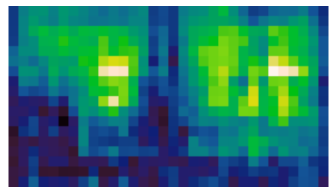
HOG feature map



Template

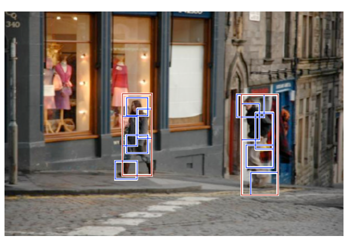


Detector response map

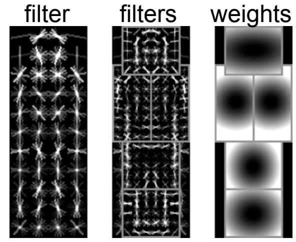


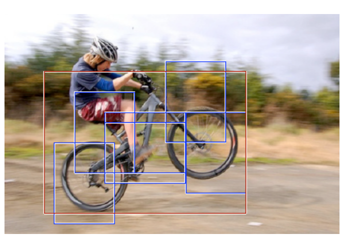
N. Dalal and B. Triggs,
[Histograms of Oriented Gradients for Human Detection](), CVPR 2005

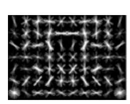
Discriminative part-based models

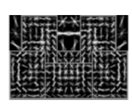


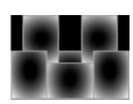
Root filter | Part filters | Deformation weights









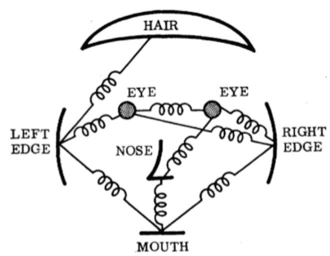


P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan,
[Object Detection with Discriminatively Trained Part Based Models](), PAMI 32(9), 2010

Part-based representation

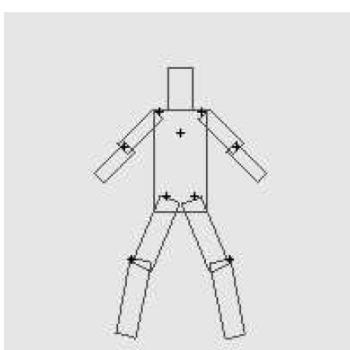
Objects are decomposed into parts and spatial relations among parts

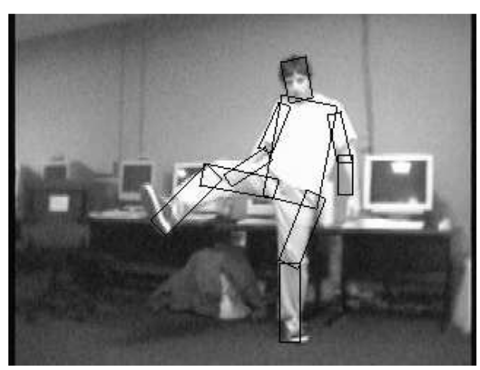
E.g. Face model by Fischler and Elschlager '73



Part-based representation

Tree model ➔ Efficient inference by dynamic programming





Pictorial Structure

Matching = Local part evidence + Global constraint

$$L^* = \arg\min_L \left(\sum_{i=1}^{n} m_i(l_i) + \sum_{(v_i,v_j)\in E} d_{ij}(l_i,l_j) \right)$$

$m_i(l_i)$: matching cost for part I

$d_{ij}(l_i,l_j)$: deformable cost for connected pairs of parts

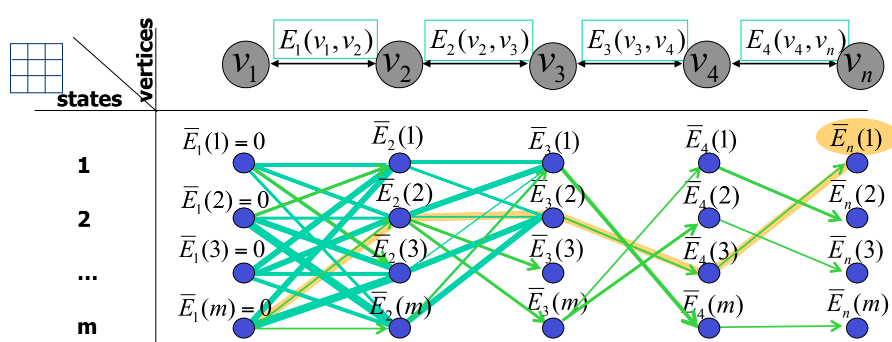
(v_i,v_j): connection between part i and j

7

Viterbi algorithm

Main idea: determine optimal position (state) of predecessor, for each possible position of self. Then backtrack from best state for last vertex.

$$E_{total} = E_1(v_1,v_2) + E_2(v_2,v_3) + \ldots + E_{n-1}(v_{n-1},v_n)$$

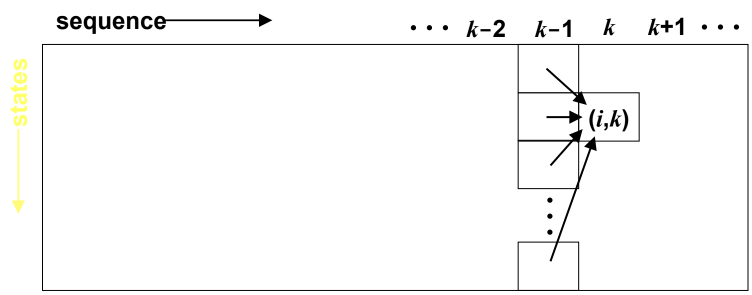


Complexity: $O(nm^2)$ **vs. brute force search ____?**

Example adapted from Y. Boyk

The Viterbi Algorithm

$$V(i,k)=\begin{cases}\max\limits_{j} V(j,k-1)P_t(q_i\mid q_j)P_e(x_k,q_i) & \text{if } k>0,\\ P_t(q_i\mid q_0)P_e(x_0\mid q_i) & \text{if } k=0.\end{cases}$$



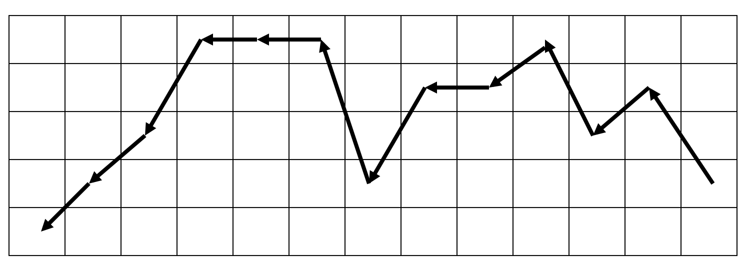
$$\phi_{\max}=\underset{\phi_{i,L-1}}{\arg\max}\, V(i,L-1)P_t(q_0\mid q_i)$$

Viterbi: Traceback

$$V(i,k)=\begin{cases}\max\limits_{j} V(j,k-1)P_t(q_i\mid q_j)P_e(x_k\mid q_i) & \text{if } k>0,\\ P_t(q_i\mid q^0)P_e(x_0\mid q_i) & \text{if } k=0.\end{cases}$$

$$T(i,k)=\begin{cases}\underset{j}{\operatorname{argmax}}\, V(j,k-1)P_t(q_i\mid q_j)P_e(x_k\mid q_i) & \text{if } k>0,\\ 0 & \text{if } k=0.\end{cases}$$

$T(\,T(\,T(\,\ldots\,T(\,T(i,L-1),L-2)\,\ldots,2),1),0)=0$



Viterbi Algorithm in Pseudocode

$\lambda_{trans}[q_i]=\{q_j \mid P_t(q_i|q_j)>0\}$

$\lambda_{emit}[s] = \{q_i \mid P_e(s|q_i)>0\}$

```
procedure viterbi(Q,α,Pt,Pe,S,λtrans,λemit)
    for k←0 up to |S|-1 do
      for i←0 up to |Q|-1 do
        V[i][k]←-∞;
        T[i][k]←NIL;
    for i←1 up to |Q|-1 do
      V[i][0]←log(Pt(qi|q0))+log(Pe(S[0]|qi));
      if V[i][0]>-∞ then T[i][0]←0;
    for k←1 up to |S|-1 do
      foreach qi∈λemit[S[k]] do
        foreach qj∈λtrans[ qi] do
          v←V[j][k-1]+log(Pt(qi|qj))+
                           log(Pe(S[k]|qi));
          if v>V[i][k] then
            V[i][k]←v;
            T[i][k]←j;
    y←1;
    push φ,0;
    for i←2 up to |Q|-1 do
      if V[i][|S|-1]+log(Pt(q0|qi)) >
         V[y][|S|-1]+log(Pt(q0|qy)) then y←i;
    for k←|S|-1 down to 0 do
      push φ,y;
      y←T[y][k];
    push φ,0;
    return φ;
```

initialization

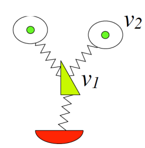
fill out main part of DP matrix

choose best state from last column in DP matrix

traceback

Matching on tree structure

$$E(L) = \sum_{i=1}^{n} m_i(l_i) + \sum_{(v_i,v_j)\in E} d_{ij}(l_i,l_j)$$



For each l_1, find best l_2:

$$\text{Best}_2(l_1) = \min_{l_2} \left[m_2(l_2) + d_{12}(l_1,l_2)\right]$$

Remove v_2, and repeat with smaller tree, until only a single part

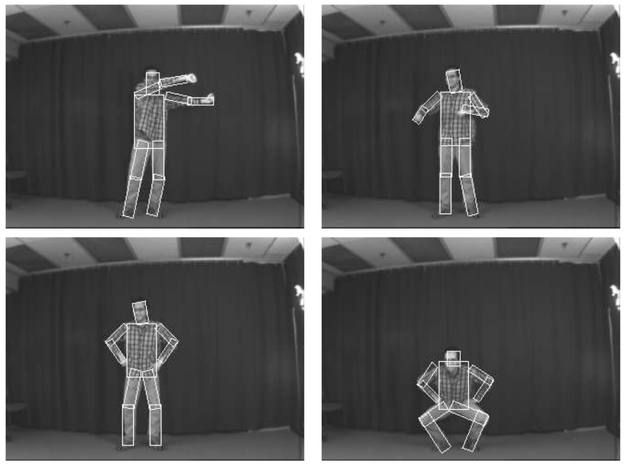
Complexity: $O(nk^2)$: n parts, k locations per part

$$B_j(l_i) = \min_{l_j}(m_j(l_j) + d(l_i,l_j) + \sum_{v\in C_j} B_c(l_j))$$

For root no 2nd term, for leaves no 3rd term

12

Sample result on matching human



13

Pictorial Structures

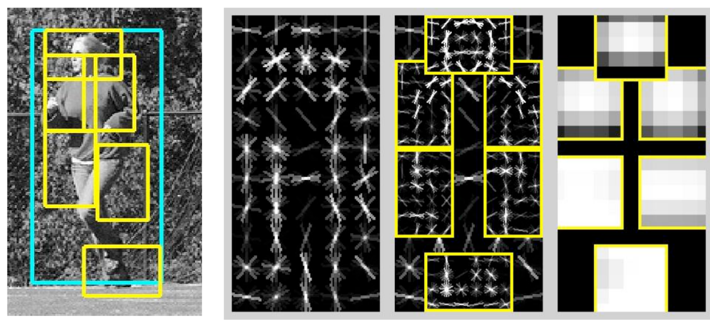
We can efficiently solve the above optimization

Problem using distance transform in linear *O(nk)*

$$L^* = \arg\min_L \left(\sum_{i=1}^{n} m_i(l_i) + \sum_{(v_i,v_j)\in E} d_{ij}(l_i, l_j) \right)$$

Pictorial structures combine local appearance scores with global spatial constraints

Discriminatively trained part based models

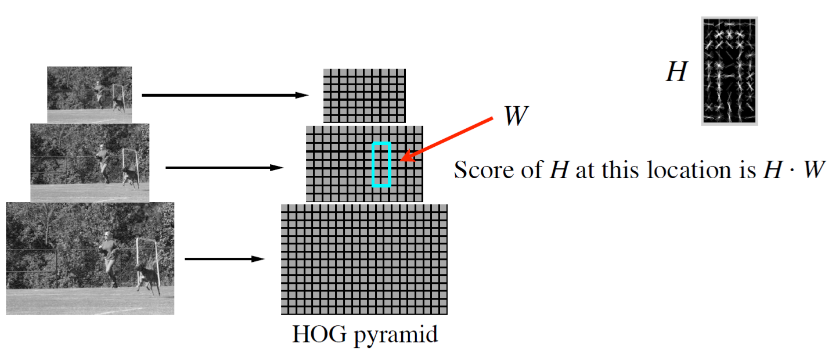


detection root filter part filters deformation models

15

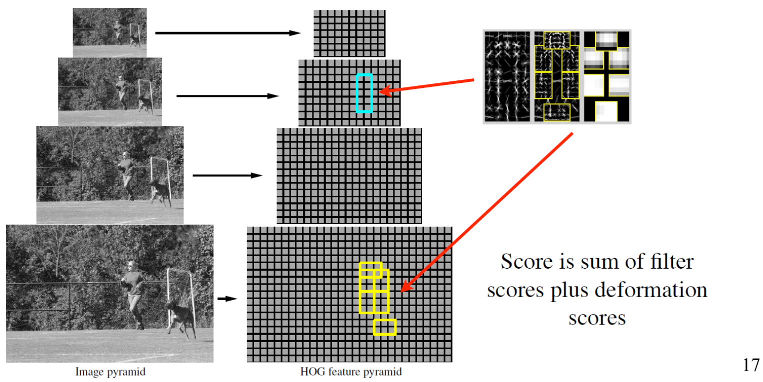
Filters

Filters are rectangular templates defining weights for features



Object hypothesis

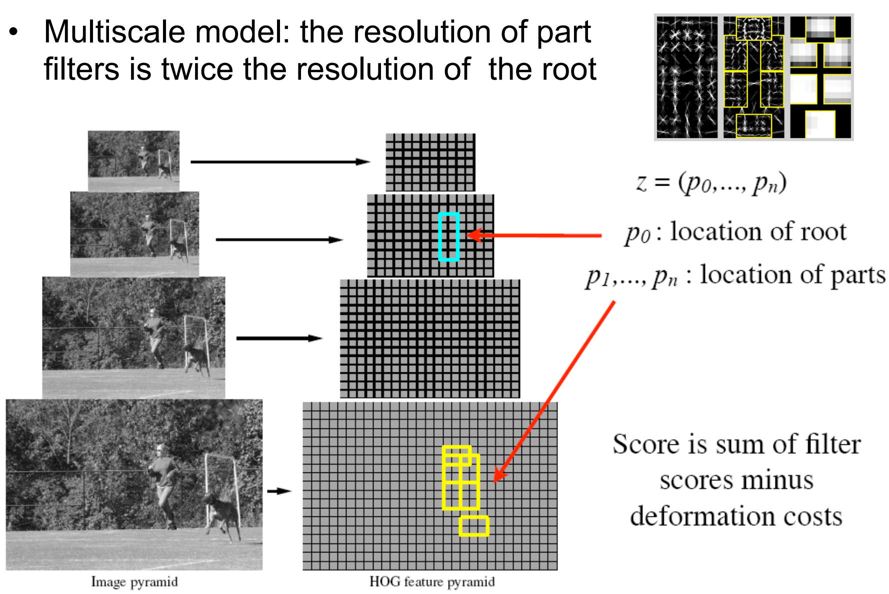
Coarser level for the root filter (whole object) and higher level for part filters



17

Object hypothesis

- Multiscale model: the resolution of part filters is twice the resolution of the root



Score of the filter : inner products between the filter and features

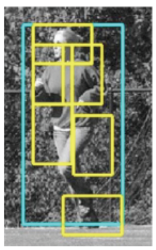
Scoring an object hypothesis

- The score of a hypothesis is the sum of filter scores at the locations minus the sum of deformation costs

Subwindow features — Displacements

$$score(p_0,\ldots,p_n) = \sum_{i=0}^{n} F_i \cdot H(p_i) - \sum_{i=1}^{n} D_i \cdot (dx_i, dy_i, dx_i^2, dy_i^2)$$

Filters — Deformation weights



Scoring an object hypothesis

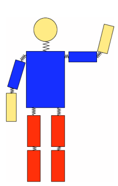
- The score of a hypothesis is the sum of filter scores minus the sum of deformation costs

Subwindow features — Displacements

$$score(p_0,\ldots,p_n) = \sum_{i=0}^{n} F_i \cdot H(p_i) - \sum_{i=1}^{n} D_i \cdot (dx_i, dy_i, dx_i^2, dy_i^2)$$

Filters — Deformation weights

- Recall: pictorial structures



$$E(l_1,\ldots,l_n) = \underbrace{\sum_i m_i(l_i)}_{\textbf{Matching cost}} + \underbrace{\sum_{i,j} d_{ij}(l_i,l_j)}_{\textbf{Deformation cost}}$$

Scoring an object hypothesis

- The score of a hypothesis is the sum of filter scores minus the sum of deformation costs

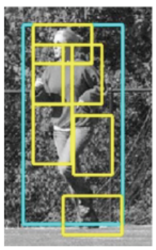
Subwindow features

Displacements

$$score(p_0,\ldots,p_n) = \sum_{i=0}^{n} F_i \cdot H(p_i) - \sum_{i=1}^{n} D_i \cdot (dx_i, dy_i, dx_i^2, dy_i^2)$$

Filters

Deformation weights



$$score(z) = w \cdot H(z)$$

Concatenation of filter and deformation weights

Concatenation of subwindow features and displacements

Detection

- Define the score of each root filter location as the score given the best part placements:

$$score(p_0) = \max_{p_1,\ldots,p_n} score(p_0,\ldots,p_n)$$

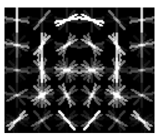
Detection

- Define the score of each root filter location as the score given the best part placements:

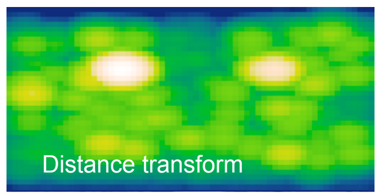
$$score(p_0) = \max_{p_1,\ldots,p_n} score(p_0,\ldots,p_n)$$

- Efficient computation: *generalized distance transforms*
  - For each "default" part location, find the best-scoring displacement

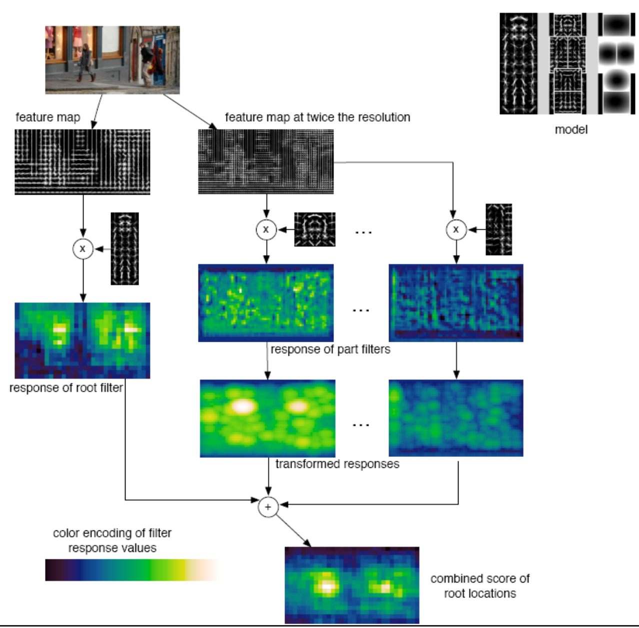
$$R_i(x,y) = \max_{dx,dy}\Big(F_i \cdot H(x+dx, y+dy) - D_i \cdot (dx, dy, dx^2, dy^2)\Big)$$



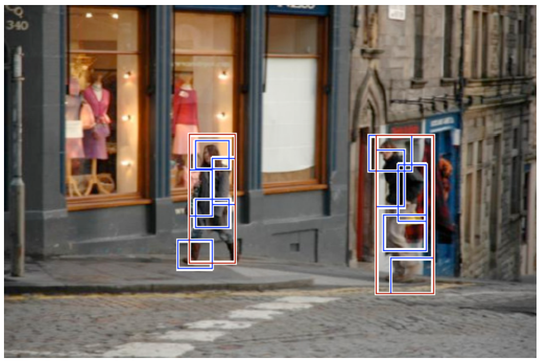
Head filter

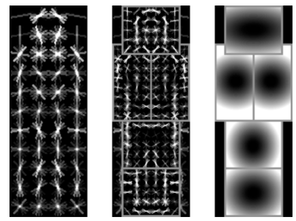


Detection



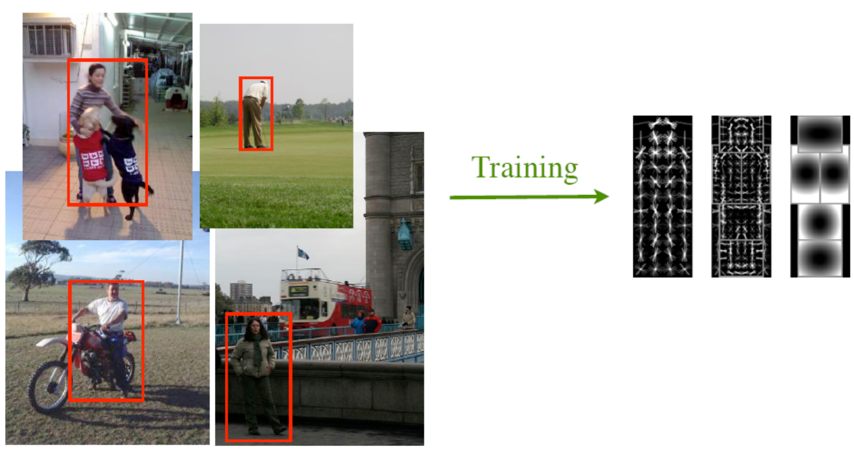
Matching result





Training

- Training data consists of images with labeled bounding boxes
- Need to learn the filters and deformation parameters



Training

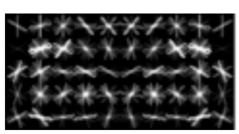
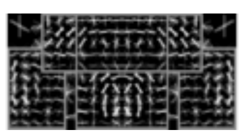
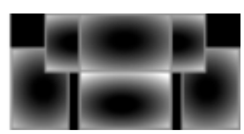
- The classifier has the form

$$f(x) = \max_z w \cdot H(x,z)$$

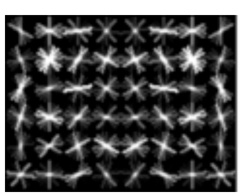
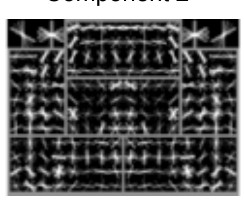
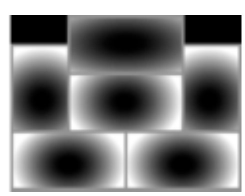
- *w* are model parameters (filters and deformation parameters, *z* are *latent* hypotheses)
- *x* is detection window, *z* are features and filter placements
- **Latent SVM** training:
  - Initialize *w* and iterate:
    - Fix *w* and find the best *z* for each training example (detection)
    - Fix *z* and solve for *w* (standard SVM training)
- Issue: too many negative examples
  - Do "data mining" to find "hard" negatives

Car model

Component 1

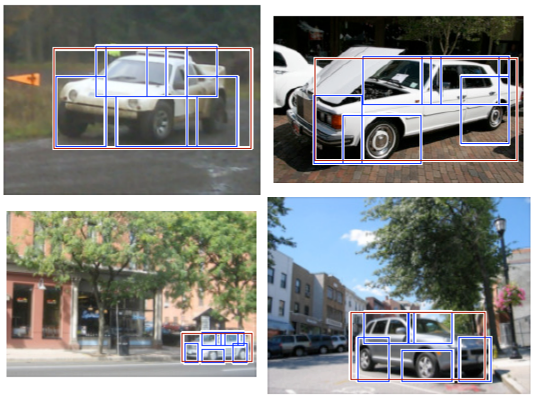
Component 2

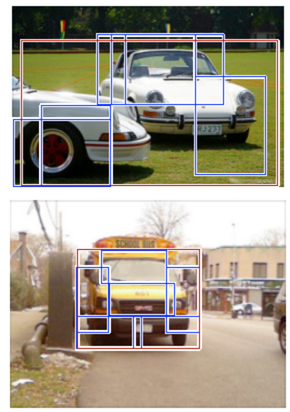




Car detections

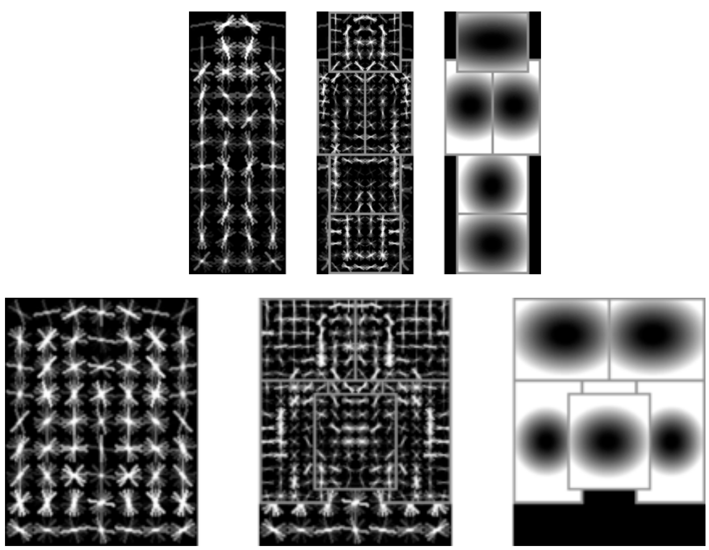
high scoring true positives

high scoring false positives





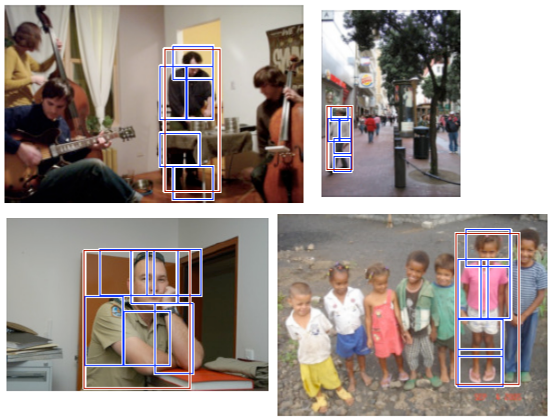
Person model



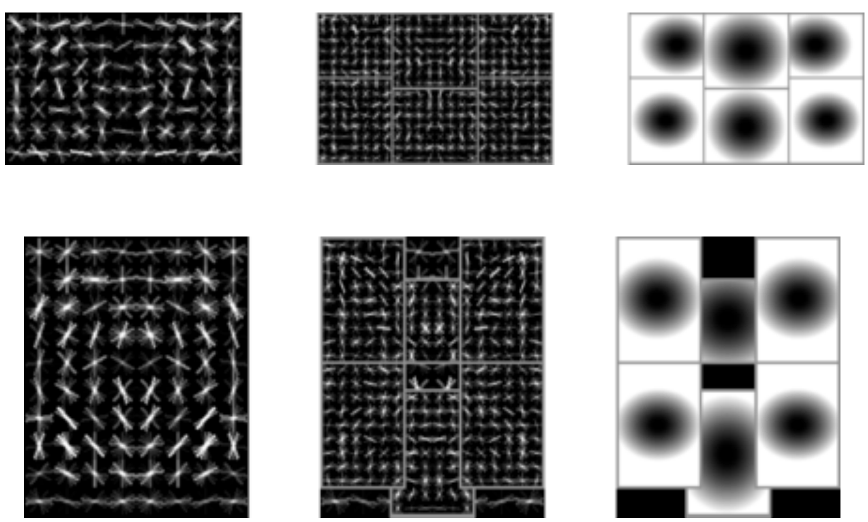
Person detections

high scoring true positives

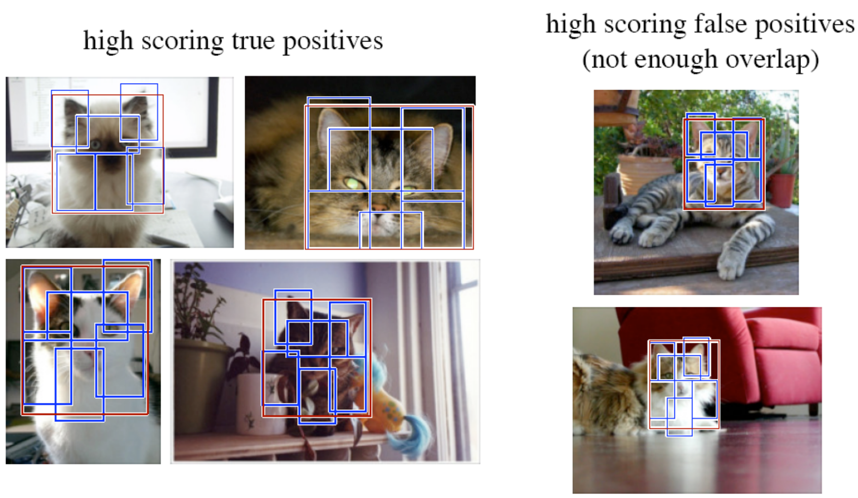
high scoring false positives
(not enough overlap)



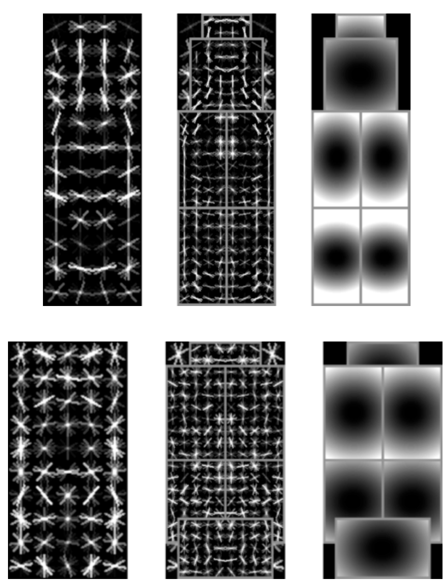
Cat model



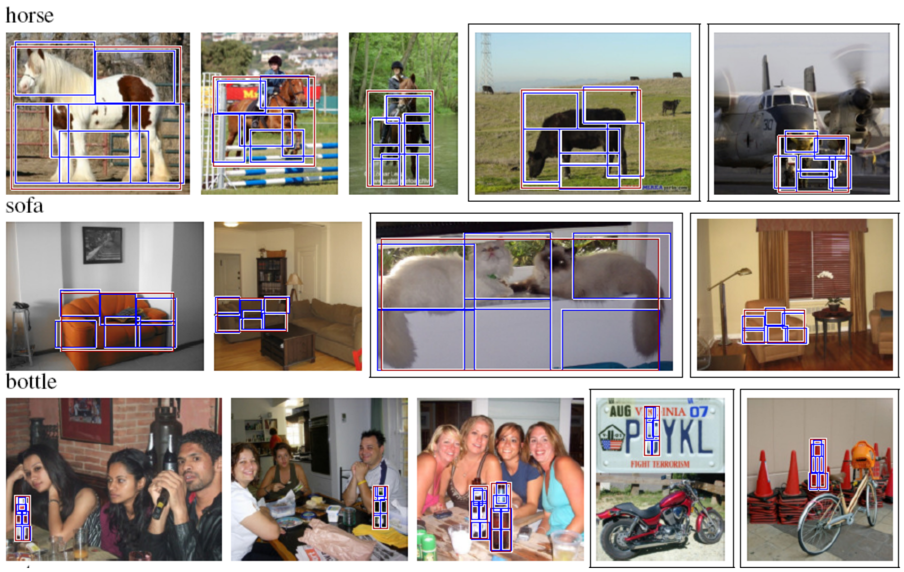
Cat detections



Bottle model

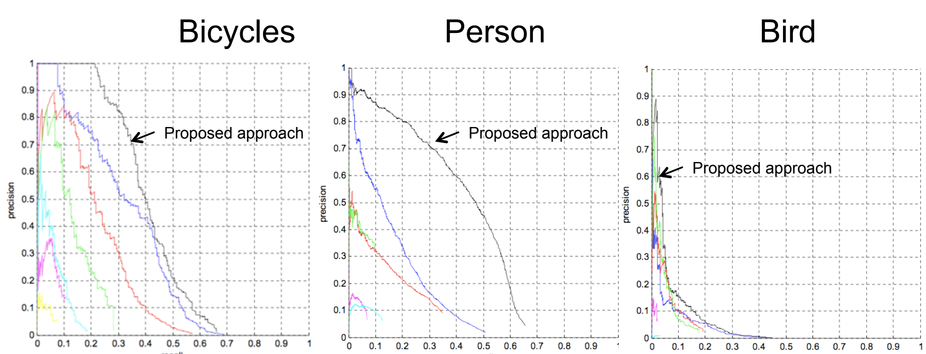


More detections



Quantitative results (PASCAL 2008)

- 7 systems competed in the 2008 challenge
- Out of 20 classes, first place in 7 classes and second place in 8 classes

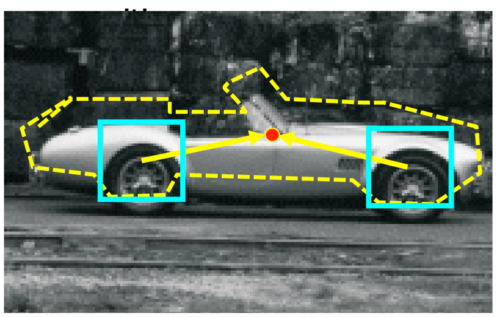


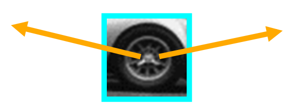
Summary

- Deformable model for object detection
  - Coarse root filter and finer part filter
  - Learn from weakly labeled data
  - Fast algorithm for matching
  - State-of-the-art results on PASCAL challenge

Implicit shape models

- Combining the edge based Hough Transform style voting with appearance codebooks
- Visual codebook is used to index votes for object





visual codeword with displacement vectors

training image annotated with object localization info

B. Leibe, A. Leonardis, and B. Schiele, Combined Object Categorization and Segmentation with an Implicit Shape Model, ECCV Workshop on Statistical Learning in Computer Vision 2004

Implicit shape models

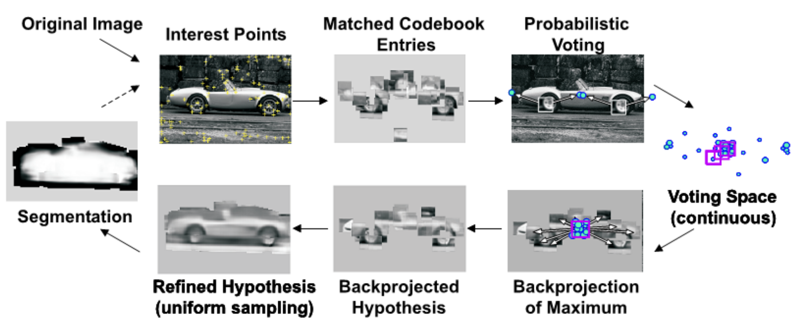
- Visual codebook is used to index votes for object position

test image

Idea Implicit Shape Model

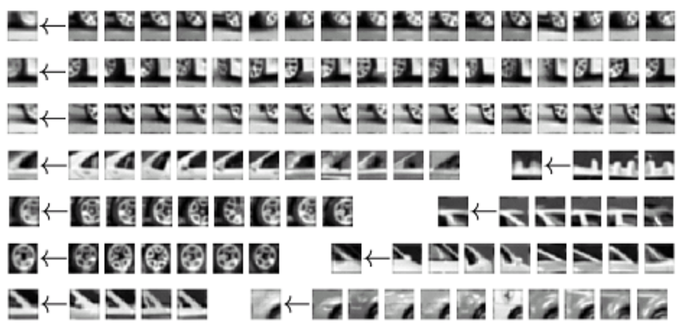
Faces rectangular templates – detection windows

Does not generalize to more complex object with different shapes



Initial Recognition Approach

First Step: Generate hypotheses from local features
Training: Agglomerative Clustering



$$similarity(C_1, C_2) = \frac{\sum_{p \in C_1, q \in C_2} NGC(p,q)}{|C_1| \times |C_2|} > t, \qquad NGC(p,q) = \frac{\sum_i (p_i - \overline{p_i})(q_i - \overline{q_i})}{\sqrt{\sum_i (p_i - \overline{p_i})^2 \sum_i (q_i - \overline{q_i})^2}}$$

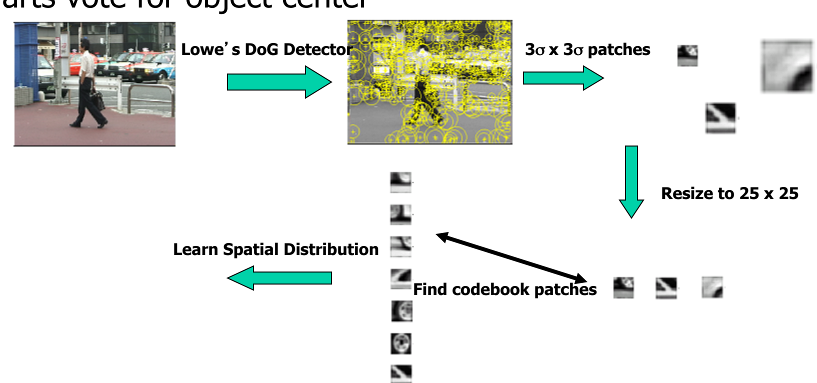
- **How to decide when to merge two clusters**
- **Average NCC of patches**
- **NCC between two patches**

Initial Recognition Approach

Codebook words - spatial information is lost

For each codebook entry store all positions it was activated in relative to object center (positions parametrized by r and theta)

Parts vote for object center



Pedestrian Detection

1. Interleaved Object Categorization and Segmentation, BMVC' 03
2. Combined Object Categorization and Segmentation with an Implicit Shape Model. Bastian Leibe, Ales Leonardis, and Bernt Schiele. In ECCV'04 Workshop on Statistical Learning in Computer Vision, Prague, May 2004.

43

Pedestrian Detection

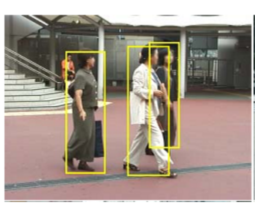
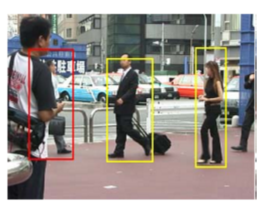
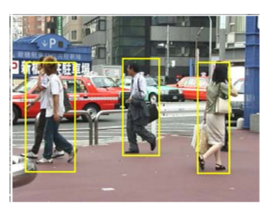
Many applications

Large variation in shape, appearance

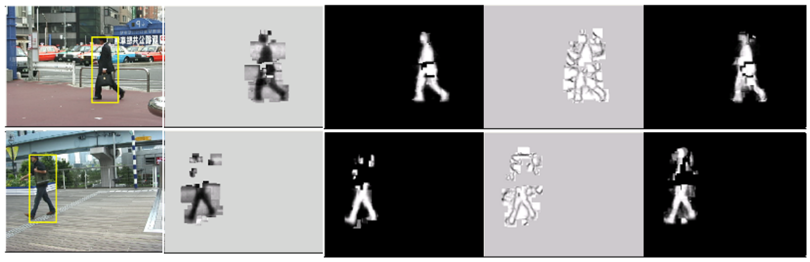
Need to combine different representations

Basic Premise: "[Such a] problem is too difficult for any type of feature or model alone"

Probabilistic bottom up, top down segmentation

Open Question: How would you do pedestrian detection/segmentation?



Original image **Segmentation from local features**

Solution: integrate as many cues as possible from many sources

Support of Segmentation from local features

Support of segmentation from global features (Chamfer Matching)

Goal: Localize AND count pedestrians in a given image

Datasets

Training Set: 35 people walking parallel to the image plane

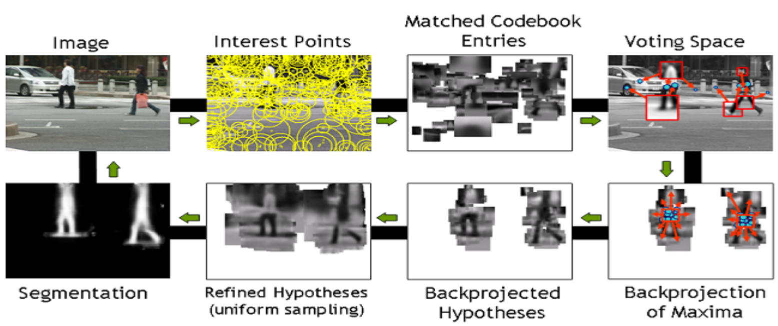
Testing Set (Much harder!): 209 images of 595 annotated pedestrians

Initial Recognition Approach

First Step: Generate hypotheses from local features
(Intrinsic Shape Models)

Testing:

Initial Hypothesis: Overall

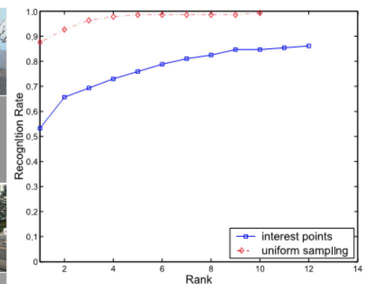


Initial Recognition Approach

First Step: Generate hypotheses from local features (Intrinsic Shape Models)

Testing:

Initial Hypothesis: Overall



Initial Recognition Approach

Second Step: Segmentation based Verification (Minimum Description Length)

Caveat: it leads to another set of problems

Or four legs and three arms

ISM doesn't know a person doesn't have three legs!

Global Cues are needed

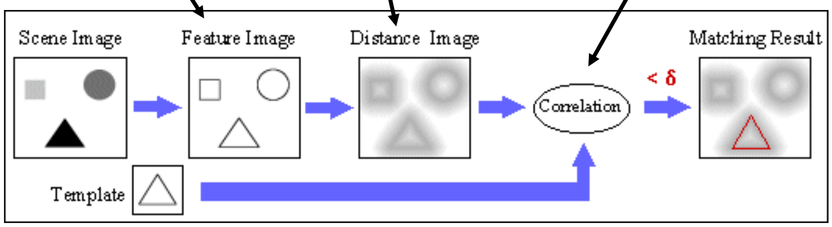
Assimilation of Global Cues

Distance Transform, Chamfer Matching

$$D_{Chamfer}(T,I) = \frac{1}{|T|}\sum_{t\in T}\min(DT_I(t),\tau)$$

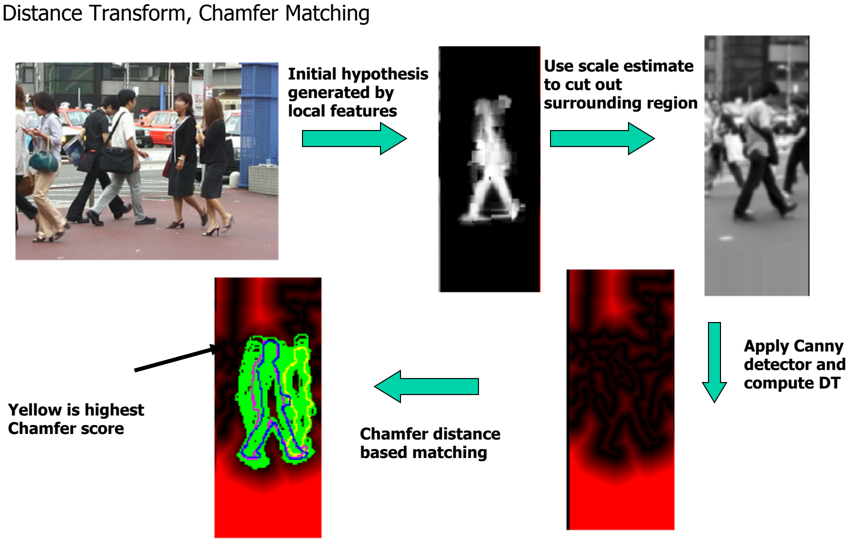
get Feature Image by an edge detector **Chamfer Distance between template and DT image**

get DT image by computing distance to nearest feature point



Assimilation of Global Cues (Attempt 1)

Distance Transform, Chamfer Matching



Results