Visuelle Perzeption für Mensch-Maschine Schnittstellen

Vorlesung, WS 2013/14

Prof. Dr. Rainer Stiefelhagen
Arne Schumann

Institut für Anthropomatik
Karlsruher Institut für Technologie

http://cvhci.ira.uka.de
rainer.stiefelhagen@kit.edu
arne.schumann@kit.edu

Computer Vision for Human-Computer Interaction
Research Group, Universität Karlsruhe (TH)
cv:hci

Computer Vision:

People Detection I

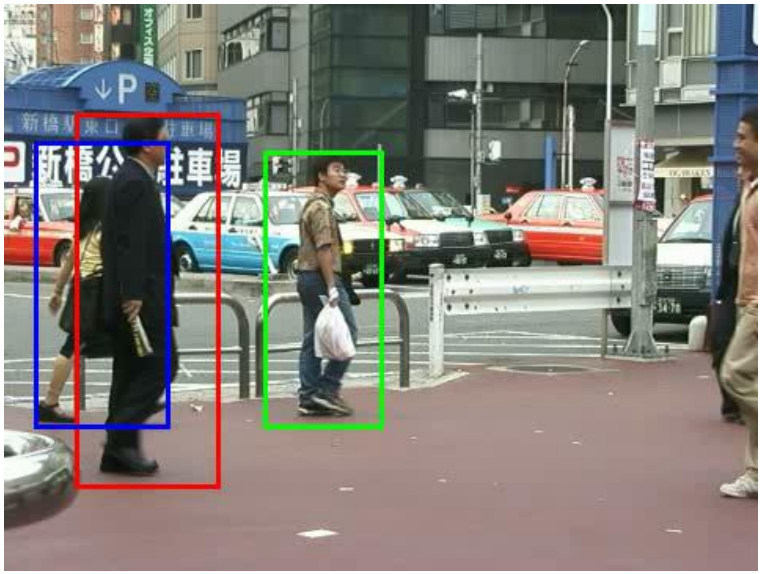
WS 2013/14

Arne Schumann

arne.schumann@kit.edu

Today

- People Detection
  - a brief introduction



- Global Approaches
  - contour based methods
  - Histograms of Oriented Gradients (HOG) (Dalal and Triggs 2005)
  - Silhouette Matching (Gavrila et al. 1999+)

Still images vs. videos

- People detection research can be divided into two categories
  - still image based techniques
  - video based techniques
- Still image based detection
  - is typically more difficult (there is only a single frame to generate a decision)
  - performs poorer than video based techniques
  - is applicable in a wider variety of applications

Still image based techniques

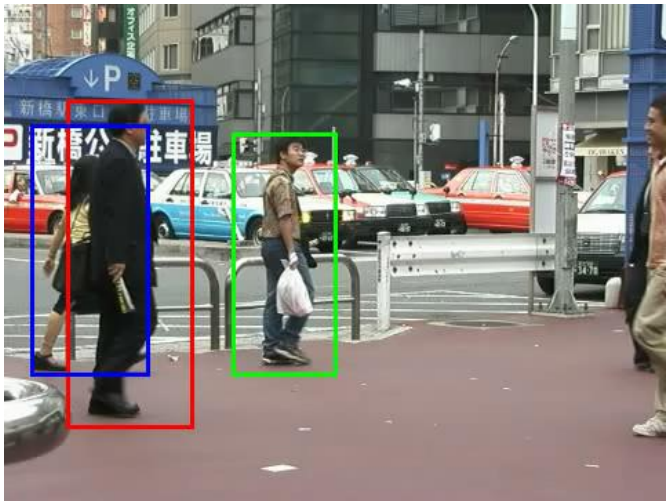
- Still image based techniques are required for
  - e.g. photo/web search applications
  - initialization or recovery of video based techniques
- Used information cues include
  - gray/color information
  - (infra-red)
  - (radar)
  - (stereo)
- Applications
  - e.g. pedestrian detection from moving vehicles

Video based techniques

- Video based techniques typically use
  - background modeling
  - temporal information (speed, position of person in earlier frames)
  - or optical flow
- Often hard to apply in unconstrained environments
  - i.e. moving camera (independent motion)
  - or changing background structures
- Applications
  - e.g. Surveillance
  - or Human-Robot Interaction

In this lecture

- Focus of this lecture mainly
  - on still image based techniques
  - with gray/color based images (infra-red approaches typically apply similar techniques)
  - in realistic environments



History

- People detection is a relatively new field
- First relevant approaches date to the late 1990s
- Interestingly earlier approaches are based on video
  - typically indoor (controlled and constant lighting)
  - static camera and background
- People detection is tightly coupled with general object detection techniques

Challenges

- Detection in Crowded Scenes
  - learn object variability
    - changes in appearance, scale, viewpoint (3D!) and articulation
  - compensate for clutter, overlap, and occlusion

Discriminative vs. generative models

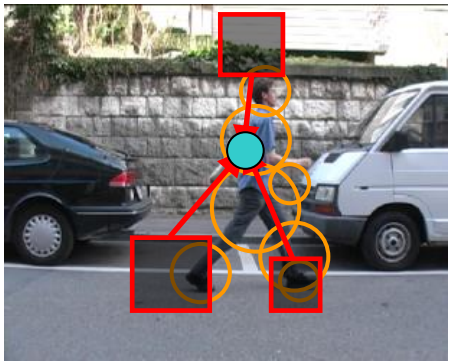
- Generative
  - ✚ possibly interpretable
  - ✚ models the object class/can draw samples
  - ▬ model variability unimportant to classification task
  - ▬ often hard to build good model with few parameters
  - Example: Implicit Shape Model (ISM) (next lecture)
- Discriminative
  - ✚ appealing when infeasible to model data itself
  - ✚ currently often excel in practice
  - ▬ often can't provide uncertainty in predictions
  - ▬ non-interpretable
  - Example: HOG (this lecture)

Global vs. part-based people detection

- We distinguish global people detectors and part-based people detectors

- Global approaches
  - a single feature description for the complete person

- Part-based approaches
  - individual feature descriptors for body parts / local parts



Advantages and disadvantages

- Global approaches
  - ✚ typically simple, i.e. training a discriminative classifier on top of the feature descriptions (e.g. SVM)
  - ✚ work well for low resolutions
  - ▬ problems with occlusions
  - ▬ problems with articulations
  - Example: HOG
- Part-based approaches
  - ✚ deal better with moving body parts
  - ✚ able to handle occlusions, overlaps
  - ✚ sharing of training data
  - ▬ require more complex reasoning
  - ▬ problems with low resolutions
  - Example: ISM

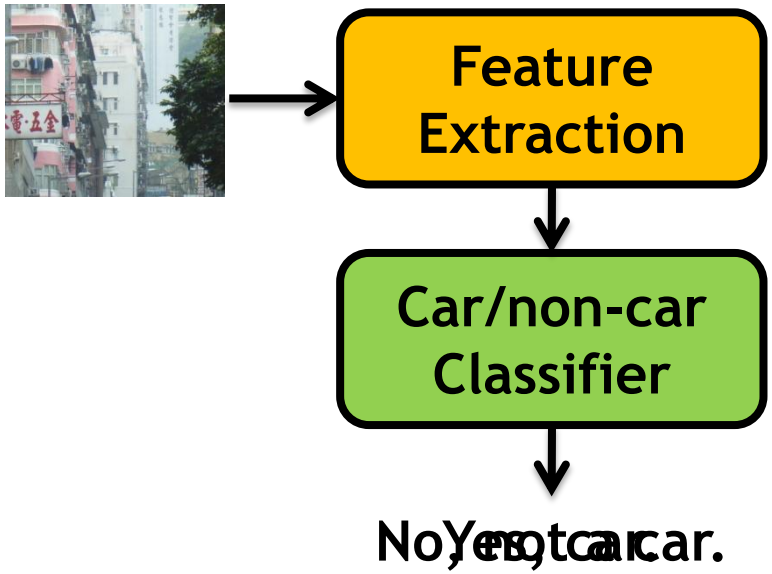
Typical components of global approaches

- Detection via classification, i.e. a binary classifier

- Gradient based
  - e.g. HOG + SVM
- Edge based
  - e.g. Chamfer Silhouette Matching + Decision Stump (threshold)
- Wavelet based
  - e.g. Wavelet Features [Oren, Papageorgiou, Poggio, 1997+] + SVM (similar to the Viola & Jones face detector)

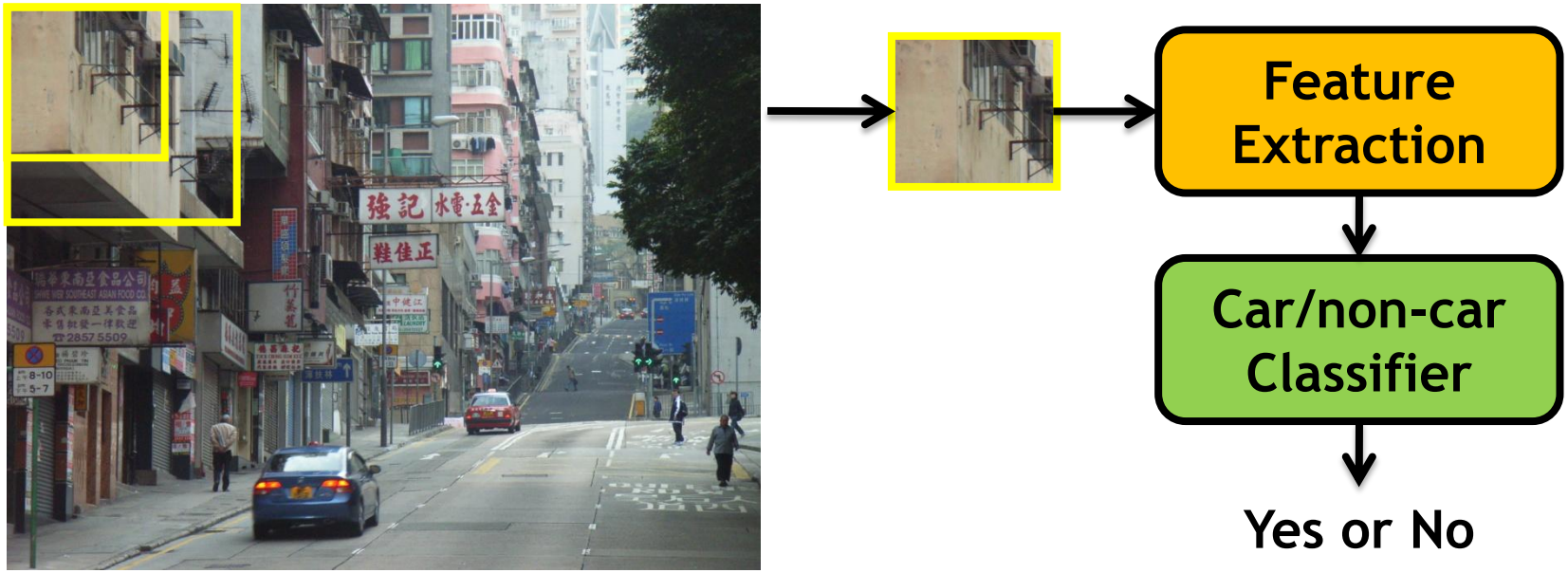
Detection via classification: Main idea

Basic component: a binary classifier



Detection via classification: Main idea

If an object may be in a cluttered scene, slide a window over the image (at different positions and scales) looking for it.



Gradient Histograms

- Have become extremely popular and successful in the vision community
- Avoid hard decisions (compared to edge based features)
- Examples:
  - SIFT (Scale-Invariant Feature Transform)
  - GLOH (Gradient Location and Orientation Histogram)
  - HOG (Histogram of Oriented Gradients)

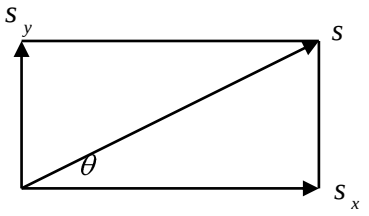
Computing gradients

- One sided: $f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$
- Two sided: $f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x-h)}{2h}$

- Filter masks in x-direction
  - One sided: | -1 | 1 |
  - Two sided: | -1 | 0 | 1 |

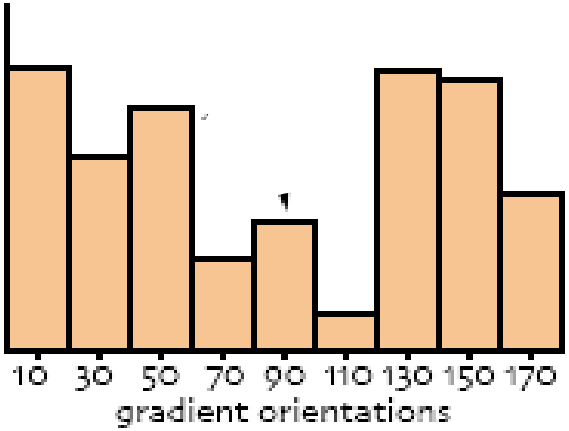
```
cv::Mat img = cvtScaleRange(loadImageGray(argv[1]), CV_32FC1);
cv::Mat dx_kernel = (cv::Mat_<float>(1, 3) << -1, 0, 1);
cv::Mat dy_kernel = (cv::Mat_<float>(3, 1) << -1, 0, 1);

cv::Mat x_grad, y_grad, magnitudes, orientations;
cv::filter2D(img, x_grad, -1, dx_kernel);
cv::filter2D(img, y_grad, -1, dy_kernel);

cv::cartToPolar(x_grad, y_grad, magnitudes, orientations);
```

- Gradient magnitude: $s = \sqrt{s_x^2 + s_y^2}$
- Gradient orientation: $\theta = \arctan(\frac{s_y}{s_x})$



Histograms

- Gradient histograms measure the orientation and strength of gradients within an image region
  - histogram bins can be filled by absolute number of pixels or weighted according to gradient magnitude



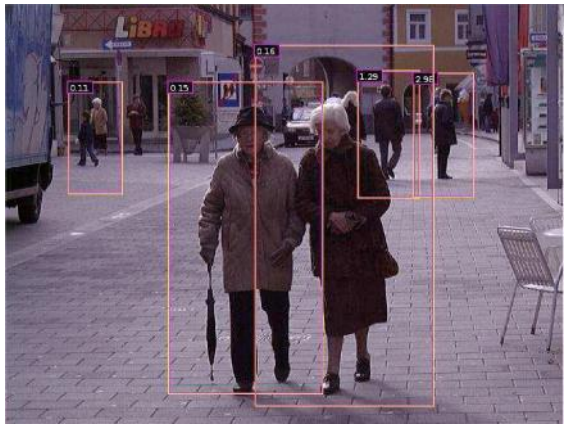
The HOG People Detector

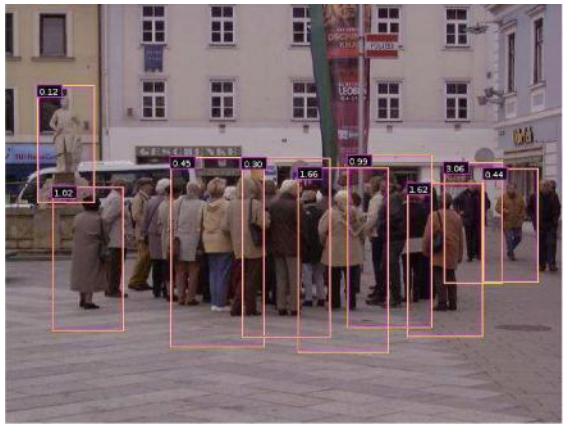
HOG - Histogram of Oriented Gradients

- Gradient-based feature descriptor developed for people detection
- Global descriptor for the complete body
- Very high-dimensional
  - typically ~4000 dimensions
- Dalal, N.; Triggs, B.; Histograms of oriented gradients for human detection; CVPR 2005

HOG People Detector

Very promising results on challenging data sets



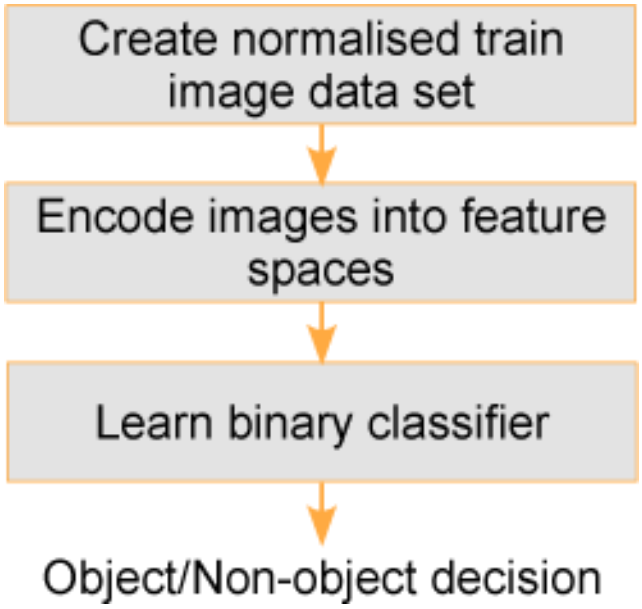


Phases

1. Learning Phase
2. Detection Phase

HOG Detector: Learning Phase

1. Learning Phase



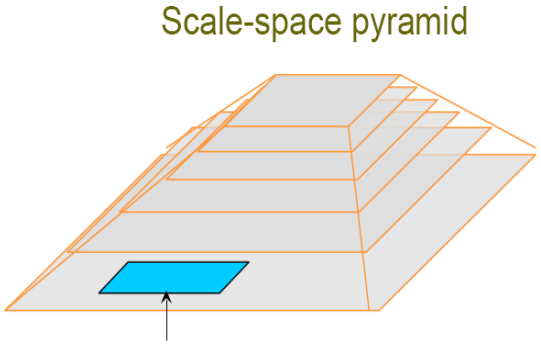
Set of cropped images containing pedestrians in normal environments

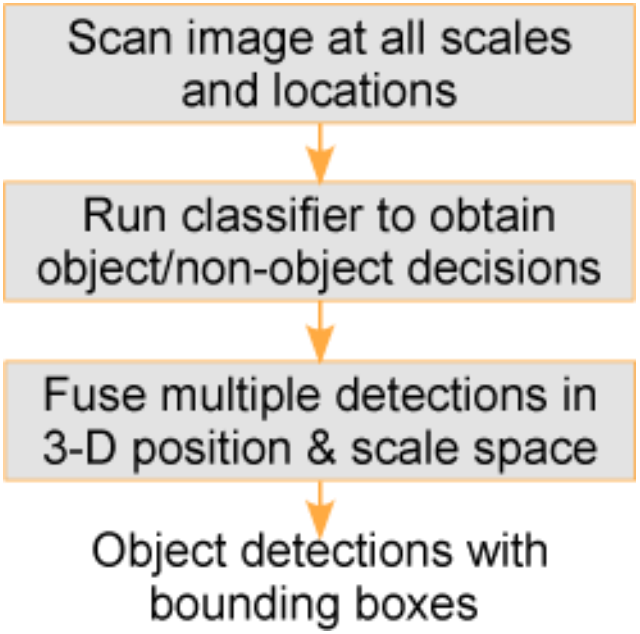
Global descriptor (~4000 dimensional vector) for each input image/person

Train a linear SVM

HOG Detector: Detection Phase

2. Detection Phase





Sliding window over each scale

Simple SVM prediction

Clustering

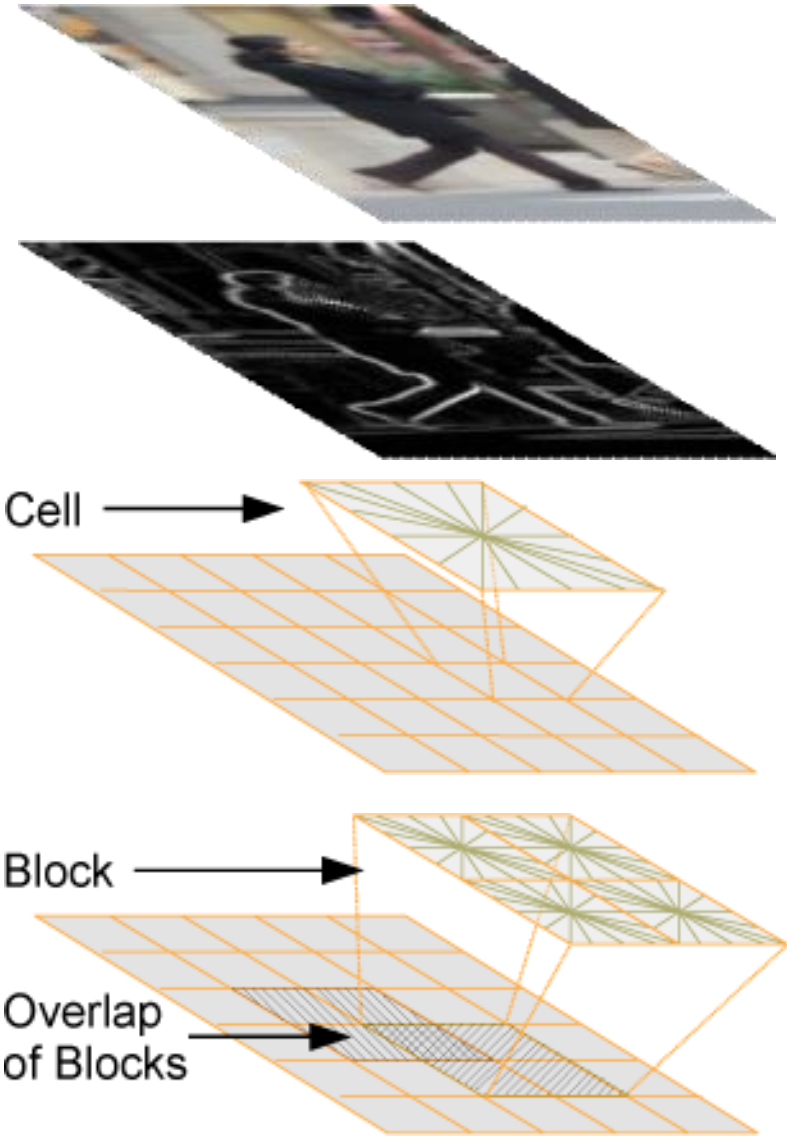
HOG Descriptor

1. Compute gradients on an image region of 64x128 pixels

2. Compute gradient orientation histograms on *cells* of 8x8 pixels (in total 8x16 cells).
   typical histogram size: 9 bins

3. Normalize histograms within overlapping *blocks* of 2x2 cells (in total 7x15 blocks)
   block descriptor size: 4x9 = 36

4. Concatenate block descriptors
   7 x 15 x 4 x 9 = 3780 dimensional feature vector



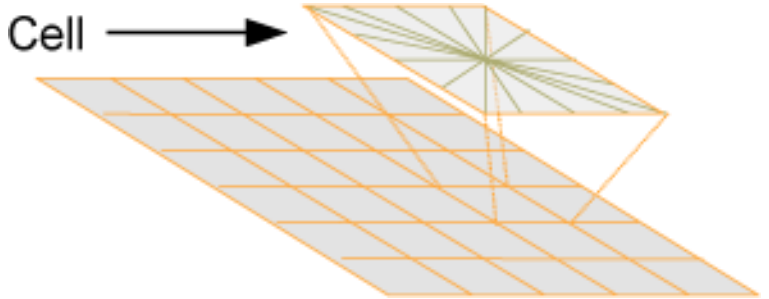
Step 1 - Gradients

- Convolution with [-1 0 1] filters
- No smoothing
- Compute gradient magnitude and direction
- Per pixel: color channel with greatest magnitude is used for final gradient

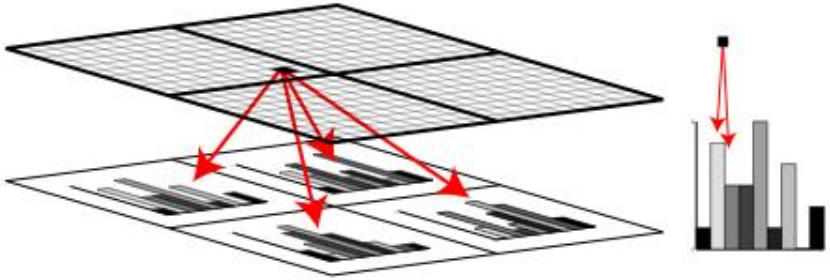
RGB Gray R G B

Step 2 - Cell histograms

- 9 bins for gradient orientations (0-180 degrees)
- Filled with magnitudes



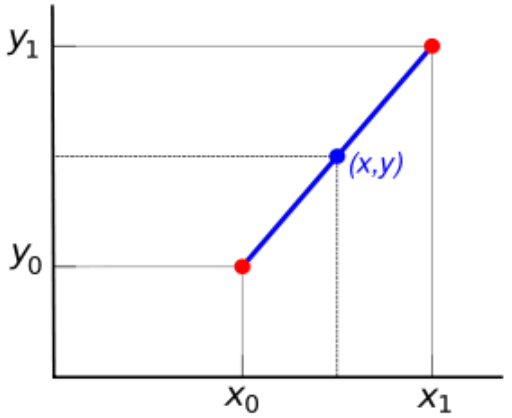
- Interpolated trilinearly:
  - bilinearly into spatial cells
  - linearly into orientation bins



Linear and Bilinear interpolation for subsampling

Linear:

$$y = y_0 + (x - x_0)\frac{y_1 - y_0}{x_1 - x_0}$$

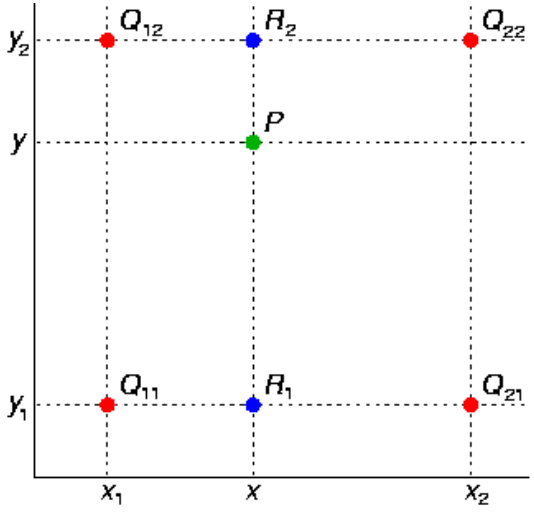


Bilinear:

$$f(R_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}) \quad \text{where} \quad R_1 = (x, y_1),$$

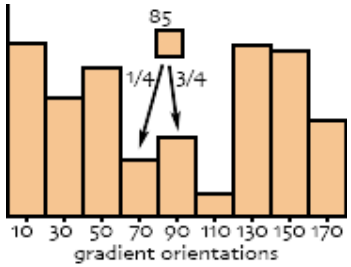
$$f(R_2) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22}) \quad \text{where} \quad R_2 = (x, y_2).$$

$$f(P) \approx \frac{y_2 - y}{y_2 - y_1} f(R_1) + \frac{y - y_1}{y_2 - y_1} f(R_2).$$

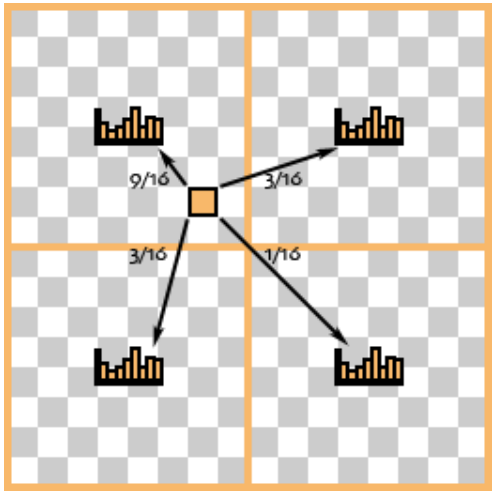


Histogram interpolation example

- Task: assign new pixel to cell histogram
  - θ =85 degrees, magnitude=48
- Linear orientation interpolation by distance to bin centers
  - bin 70: distance of 15 degrees
  - bin 90: distance of 5 degrees
  - weights: 5/20=1/4, 15/20=3/4



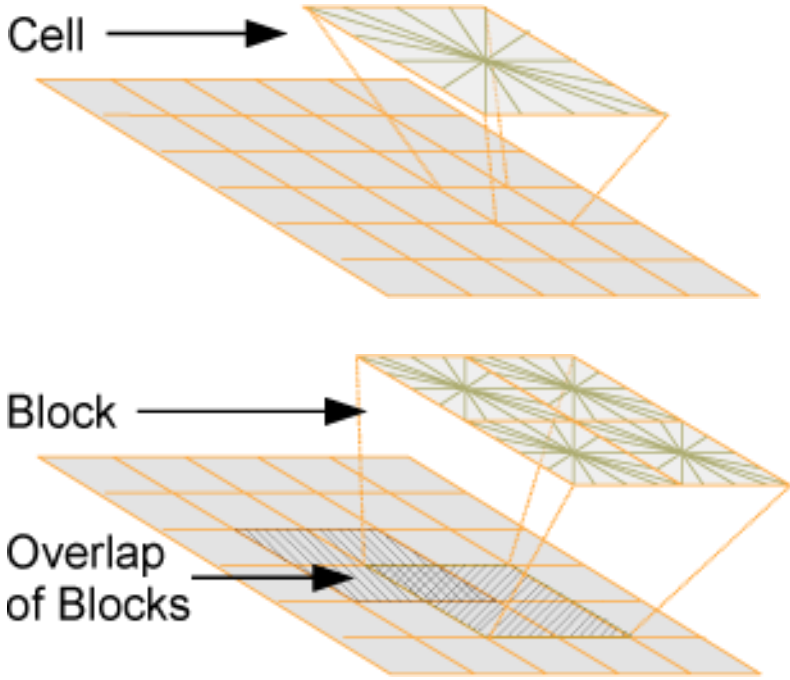
- Bilinear interpolation by distance to cell centers
  - left: 2, right: 6
  - top: 2, bottom: 6
  - ratio between left and right: 6/8, 2/8
  - ratio between top and bottom: 6/8, 2/8
  - weights:
    - top left: 6/8*6/8 = 36/64 = 9/16
    - top right: 6/8*2/8 = 12/64 = 3/16
    - bottom left: 2/8*6/8 = 12/64 = 3/16
    - bottom right: 2/8*2/8 = 4/64 = 1/16



- Putting all together:
  - top left, 70 deg bin: 48 * 1/4 * 9/16 = 27/4
  - top left, 90 deg bin: 48 * 3/4 * 9/16 = 81/4
  - top right, 70 deg bin: 48 * 1/4 * 3/16 = 9/4
  - …

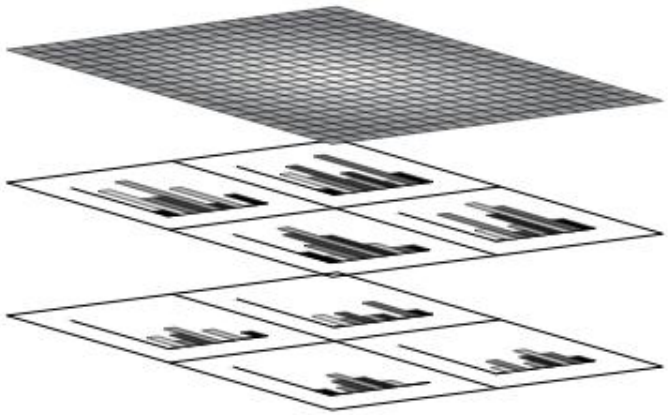
Step 3 - Blocks

- Overlapping blocks of 2x2 cells
- Cell histograms are concatenated and then normalized
  - note that each cell has several occurrences with different normalization in final descriptor
- Normalization
  - different norms possible (L2, L2hys etc.)
  - add a normalization epsilon to avoid division by zero



Step 3 – Blocks (contd.)

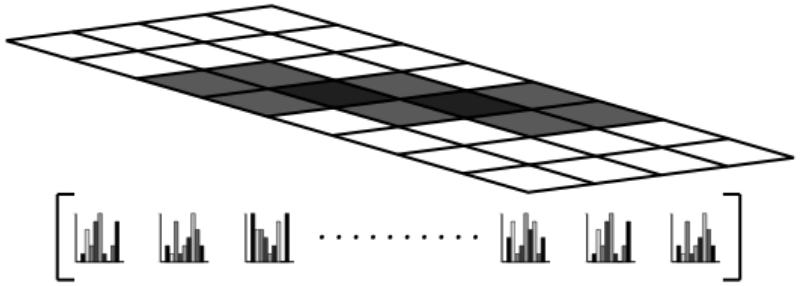
- Gradient magnitudes are weighted according to a Gaussian spatial window



- Distant gradients contribute less to the histogram
- The Gaussian spans over the size of a block but the weight already has to be considered when generating the cell histograms

Step 4 – The final HOG descriptor

- Concatenation of block descriptors



- Visualization:

Engineering

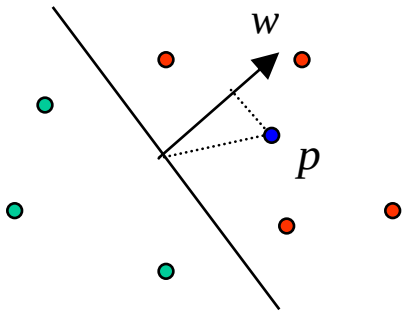
- Developing a feature descriptor requires a lot of engineering
  - testing of parameters (e.g. size of cells, blocks, number of cells in a block, size of overlap)
  - normalization schemes (e.g. L1, L2-Norms etc., gamma correction, pixel intensity normalization)
- An extensive evaluation of different choices was performed, when the descriptor was proposed
- It’s not only the idea, but also the engineering effort

From feature to detector

- Simple linear SVM on top of the HOG Features
  - fast (one inner product per evaluation window)
    - for an entire image it's a vector-matrix multiplication
  - hyperplane normal vector:

$$w = \sum \alpha_i y_i x_i$$ with y_i in {0,1} and x_i the support vectors

$$f(p) = \sum \alpha_i y_i \langle p, x_i \rangle = p^T w$$



  - decision: $sign\ (p^T w)$

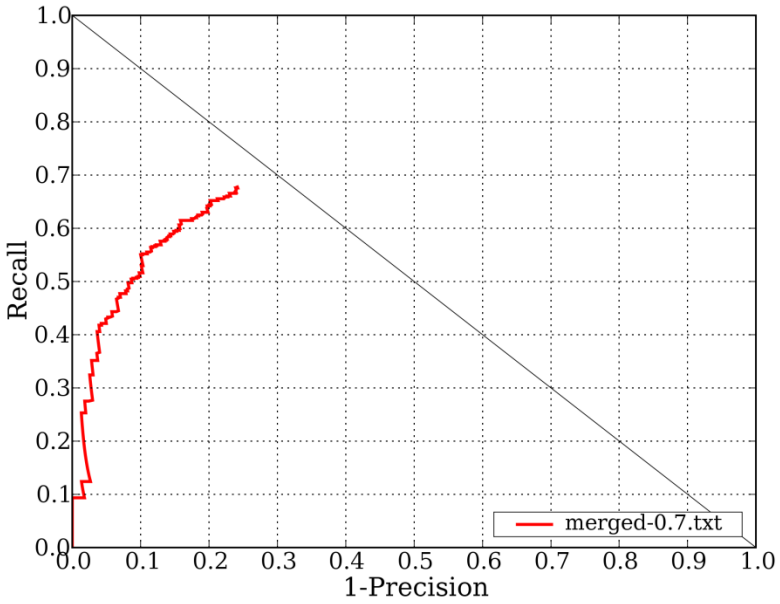
- Gaussian kernel SVM
  - slightly better classification accuracy
  - but considerable increase in computation time

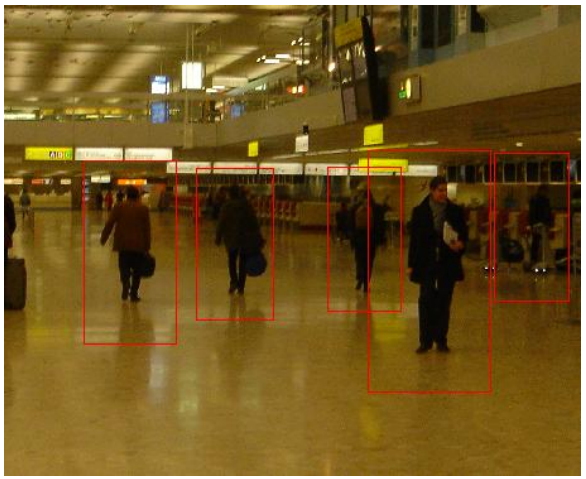
Training Set

- More than 2000 positive & 2000 negative training images (96x160 px)
- Carefully aligned and resized
- Wide variety of backgrounds

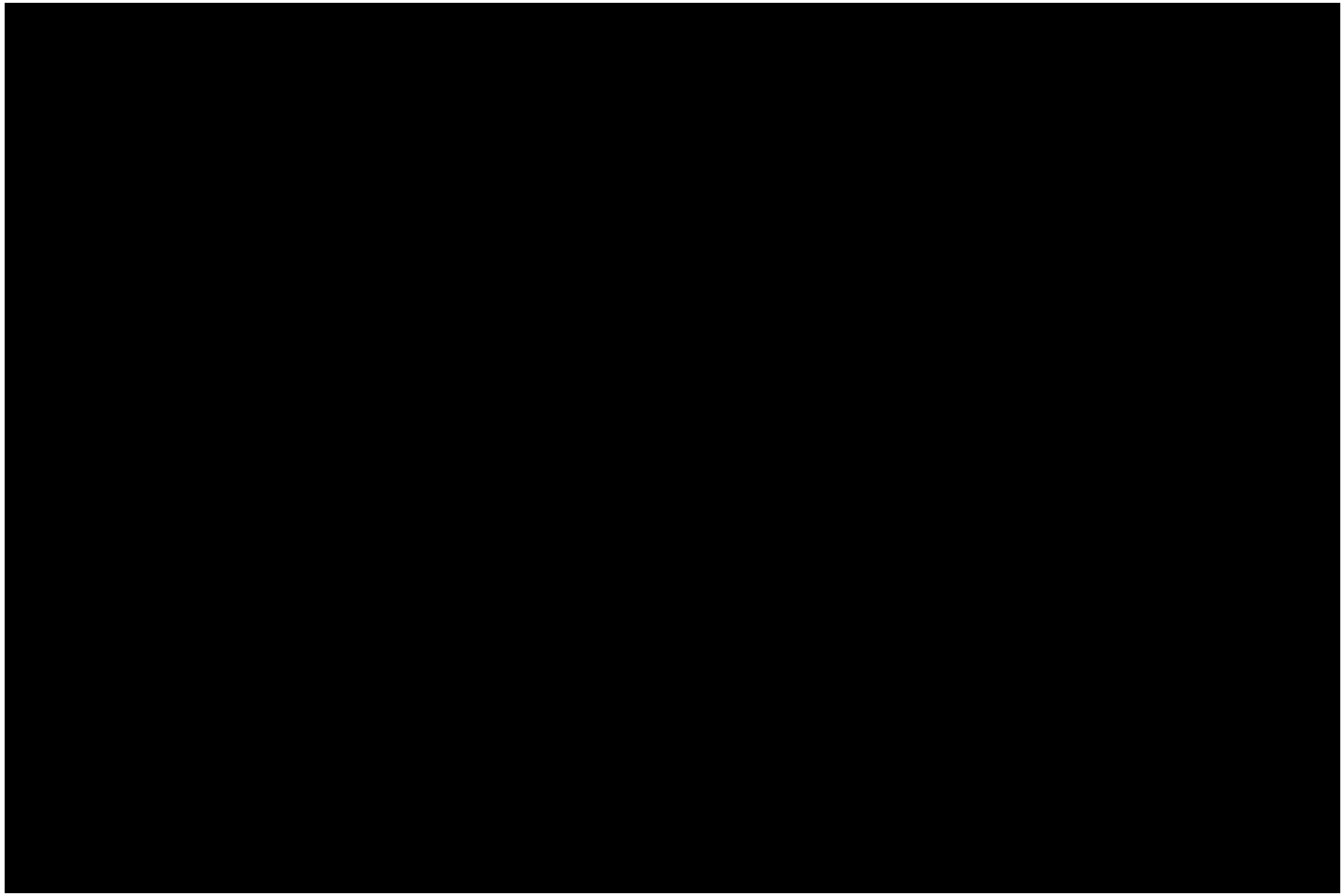
Result on INRIA database

- Test Set contains 287 images
- Resolution ~640x480
- 589 persons
- Avg. size: 288 pixels





Demo



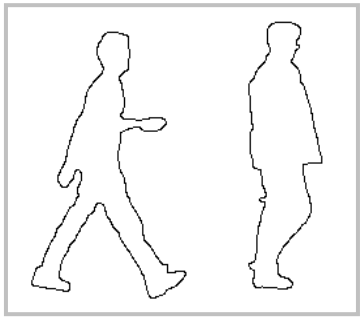
HOG in OpenCV (2.2.0)

```
cv::Mat img = cv::imread(argv[1]);

cv::Size padding(32,32);                //size of the margin added to the image before detection
cv::Size win_size(64,128);              //sliding window size
cv::Size win_stride(8,8);               //sliding window stride (1 cell)
cv::Size block_size(16,16);             //size of the blocks (2x2 cells)
cv::Size block_stride(8,8);             //stride between two blocks (1 cell)
cv::Size cell_size(8,8);                //size of a cell
int nbins = 9;                          //number of bins for the cell gradient histograms
int deriv_aperture = 1;                 //unused parameter
double win_sigma = -1.0;                //controls the size of the gaussian weighting
int norm_type = HOGDescriptor::L2Hys;   //block normalization scheme (L2 hysteresis)
double norm_thr = 0.2;                  //threshold for L2 hysteresis
bool gamma_correction = true;           //turn on sqrt() gamma correction
float scale_factor = 1.1;               //factor between two scale levels in the image pyramid
int nlevels = 64;                       //maximum number of levels allowed in the image pyramid
float score_thr = 0.0;                  //minimum score required for a detection
int group_thr = 2;                      //threshold for grouping person detections

//initialize HOG detector
cv::HOGDescriptor hog(win_size, block_size, block_stride, cell_size, nbins, deriv_aperture, win_sigma,
norm_type, norm_thr, gamma_correction, nlevels);
hog.setSVMDetector(hog.getDefaultPeopleDetector());

//detect persons
vector<cv::Rect> person_bounds;
hog.detectMultiScale(img, person_bounds, score_thr, win_stride, padding, scale_factor, group_thr);

//visualize detections
cv::Mat img_paint = img.clone();
for (size_t ii = 0; ii < person_bounds.size(); ++ii) {
   cv::rectangle(img_paint, person_bounds[ii], cv::Scalar(0,255,0), 1);
}
cv::imshow("person detections", img_paint);
cv::waitKey();
```

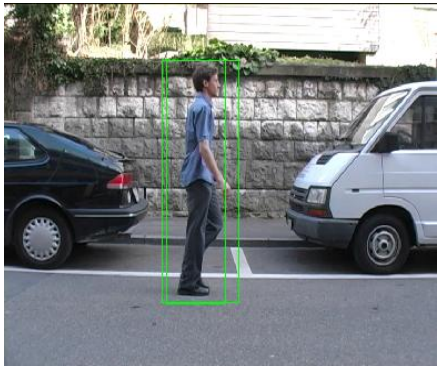
Silhouette Matching

Chamfer Matching

- Goal
  - align known object shapes with image



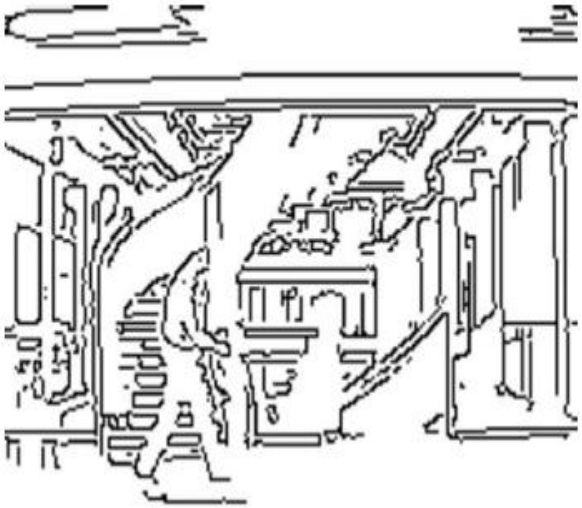
Object shapes



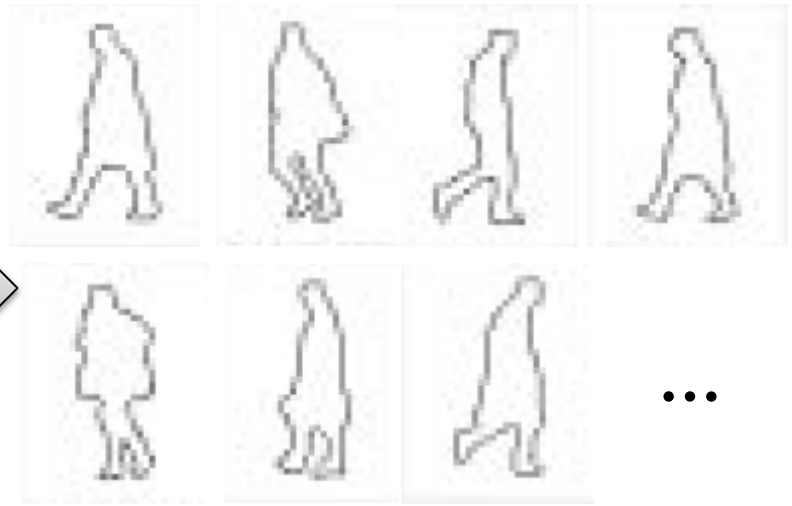
Real-world image of object

- Requirements for an alignment algorithm
  - high detection rate
  - few false positives
  - robustness
  - computationally inexpensive
- Gavrila, D.M.; Philomin, V.; Real-time object detection for “smart” vehicles; ICCV 1999

Computational Complexity



input edge image

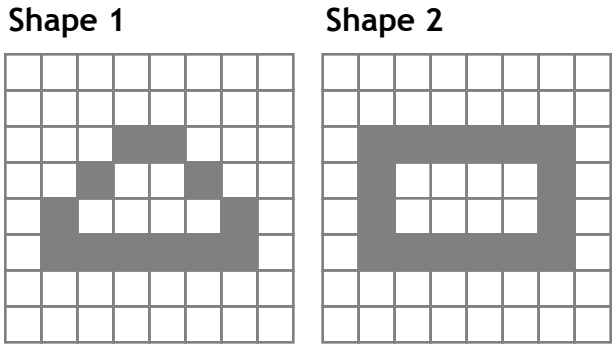


silhouette database

- Which template?
- Where in the image?
- Complexity
  - O(#positions * #templates * # contourpixels * sizeof(searchregion))

Distance Transform

- Used to compare/align two (typically binary) shapes



Distance = ?

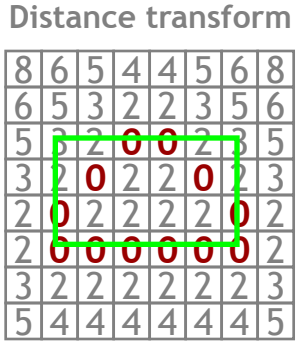
1. Compute the distance from each pixel to the nearest edge pixel
   - here the euclidean distances are approximated by the 2-3 distance

Distance transform

8	6	5	4	4	5	6	8
6	5	3	2	2	3	5	6
5	3	2	**0**	**0**	2	3	5
3	2	**0**	2	2	**0**	2	3
2	**0**	2	2	2	2	**0**	2
2	**0**	**0**	**0**	**0**	**0**	**0**	2
3	2	2	2	2	2	2	3
5	4	4	4	4	4	4	5

Distance Transform

2. Overlay second shape over distance transform



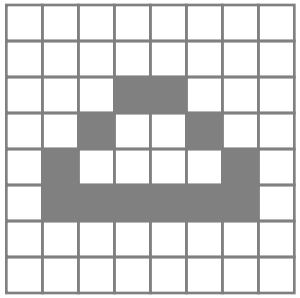
Distance = 14

3. Accumulate distances along shape 2
4. Find best matching position by an exhaustive search

- 2-3 distance is not symmetric
- 2-3 distance has to be normalized w.r.t. the length of the shapes

Chamfer Matching

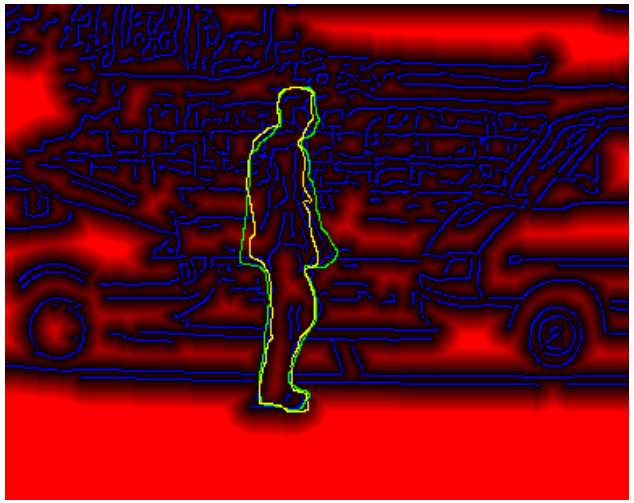
Binary image



Distance transform

8	6	5	4	4	5	6	8
6	5	3	2	2	3	5	6
5	3	2	**0**	**0**	2	3	5
3	2	**0**	2	2	**0**	2	3
2	**0**	2	2	2	2	**0**	2
2	**0**	**0**	**0**	**0**	**0**	**0**	2
3	2	2	2	2	2	2	3
5	4	4	4	4	4	4	5

Distance transform of a real-world image



Chamfer Matching

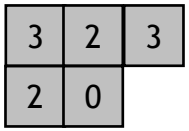
- Compute distance transform (DT)
- For each possible object location
  - position known object shape over DT
  - accumulate distances along the contour
  - keep instances where the accumulated distance is below some threshold

Distance measure

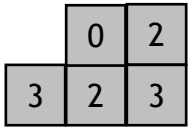
$$dist = \frac{1}{N} \sum_{i \in F} dt(i) \qquad F : features \quad N = |F|$$

Efficient implementation

- The distance transform can be efficiently computed by two scans over the complete image
- Forward-Scan
  - starts in the upper-left corner and moves from left to right, top to bottom
  - uses the following mask



- Backward-Scan
  - starts in the lower-right corner and moves from right to left, bottom to top
  - uses the following mask



Forward scan

	d	s	d
s	c	

- We can choose different values for the filter mask
- The local distances, *d*, s and c, in the mask are added to the pixel values of the distance map and the new value of the zero pixel is the minimum of the five sums
- Example:

3	2	3	5
2	0	?	?
?	?	?	?

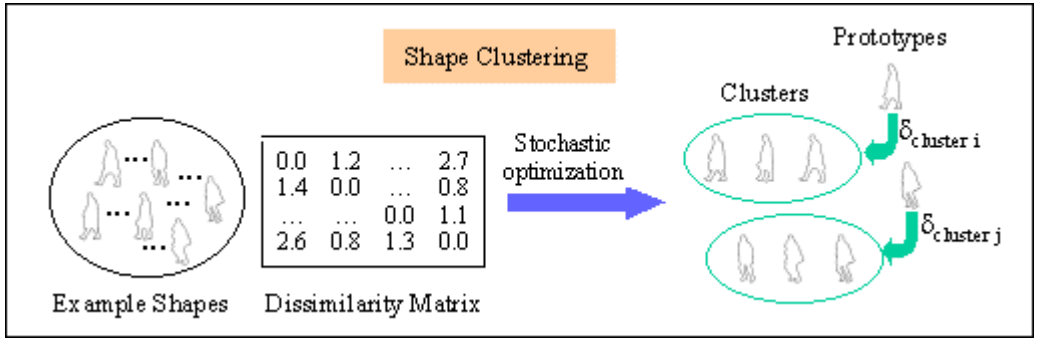
3	3+2	2+3	3+5
2	2+0	2	?
?	?	?	?

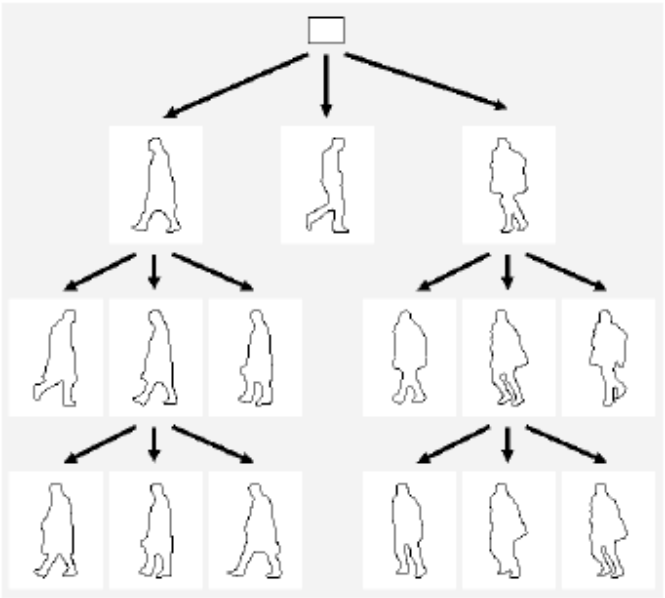
Advantages and disadvantages

- Fast
  - distance transform has to be computed only once
  - comparison for each shape location is cheap
- Good performance on uncluttered images (with few background structures)
- Bad performance for cluttered images
- Needs a huge number of people silhouettes
  - e.g. 5500 silhouette templates over five scales
  - but computation effort increases with the number of silhouettes

Template Hierachy

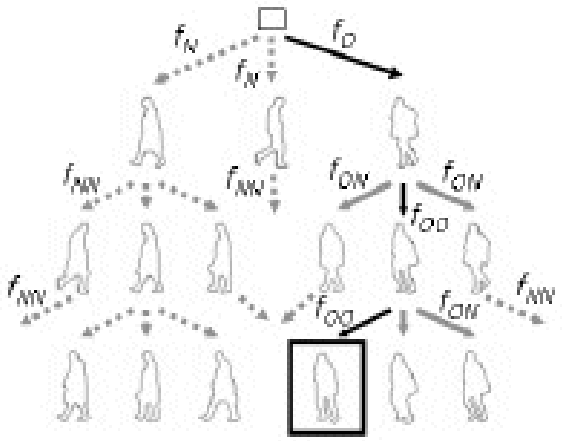
- To reduce the number of silhouettes to consider, they can be organized in a template hierarchy
- For this, the shapes are clustered by similarity





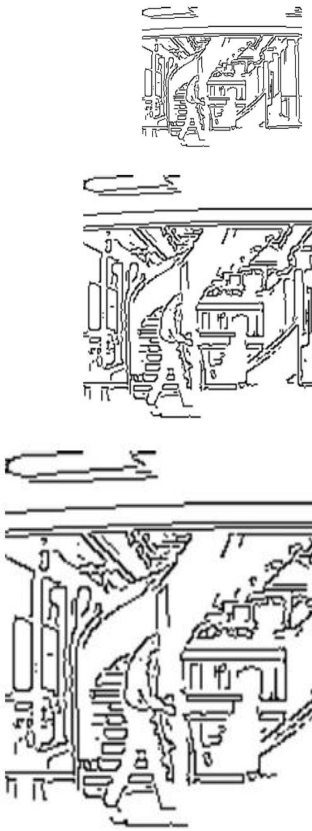
Search in the hierarchy

- Matching the shapes now corresponds to a traversal of the template hierarchy
- How can we prune search branches to speed up matching?
  - using thresholds which depend on:
    - edge detector (likelihood of gaps)
    - silhouette sizes
    - hierarchy level
    - allowed shape variation
  - thresholds are set statistically during training



Coarse-To-Fine Search

- Goal: Reduce search effort by discarding unlikely regions with minimal computational effort
- Idea:
  - subsample the image and search first at a coarse scale
  - only consider regions with a low distance when searching for a match on finer scales
- Again, we have to find reasonable thresholds

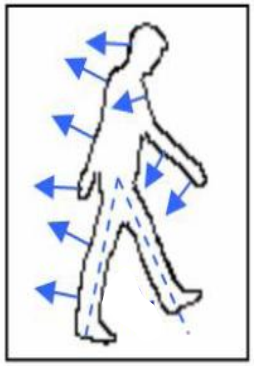


Adding edge orientation

- So far edge orientation has been completely ignored

Distance = small

- Idea: Consider edge orientation for each pixel



Edge orientation - The math

- Given two shapes S, C, we can express the chamfer distance in the following manner

$$d_{chamfer}(S,C) = \frac{1}{n}\sum_{s_i \in S} \min_{c_j \in C} \|s_i - c_j\|$$

- The orientation correspondence between two points is then measured by

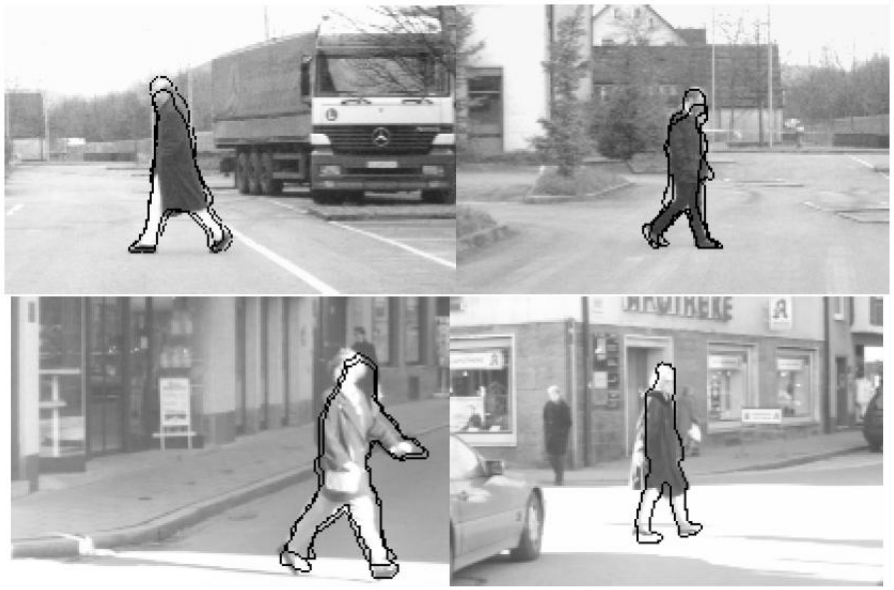
$$p(s_i, c_j) = K * [\tan(\alpha_{s_i} - \beta_{c_j})]^2$$

- The combined distance measure:

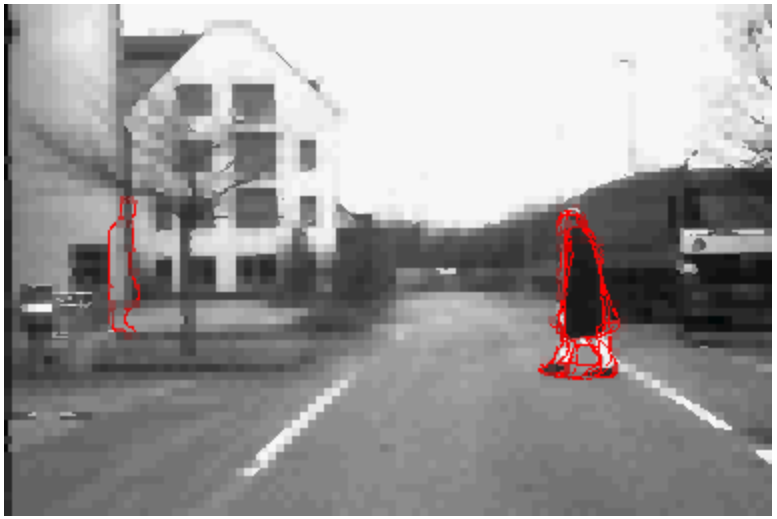
$$d_{chamfer}(S,C) = \frac{1}{n}\sum_{s_i \in S} \rho\left(\frac{1}{k}\|s_i - c(s_i)\| + p(s_i, c(s_i))\right)$$

where $c(s_i)$ is the closest contour point to point s_i

Example Detections



Video



End of lecture