Lesson 05

Image moments, LBP, HoG

Ing. Marek Hrúz, Ph.D.

Katedra Kybernetiky
Fakulta aplikovaných věd
Západočeská univerzita v Plzni

DEPARTMENT OF CYBERNETICS

Statistical description of textures

LBP

HoG

Haar and Face Detection

DEPARTMENT OF CYBERNETICS

First order statistics

- use the histogram of the image - namely the relative histogram

$$P(I) = \frac{\text{pixels with intensity } I}{\text{total pixels in region}} \tag{1}$$

- image moments

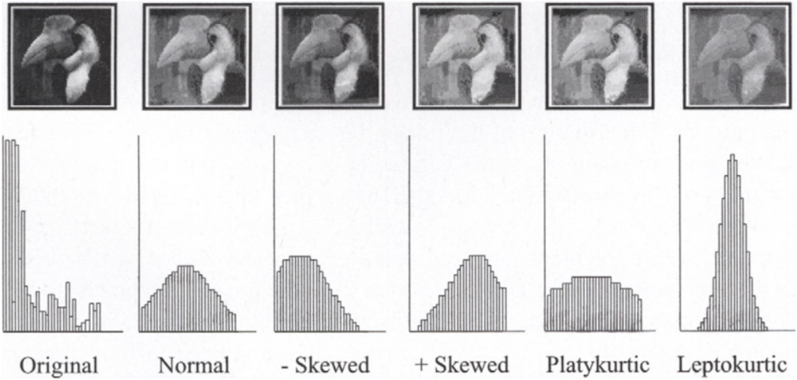
$$m_i = E[I^i] = \sum_{I=0}^{N_g-1} I^i P(I) \tag{2}$$

- image central moments

$$\mu_i = E[(I - E[I])^i = \sum_{I=0}^{N_g-1} (I - m_1)^i P(I) \tag{3}$$

- entropy

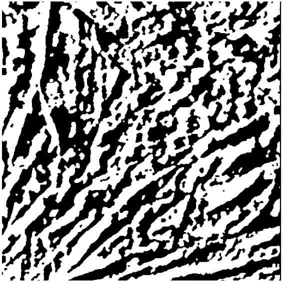
$$H = -E[log_2 P(I)] = -\sum_{I=0}^{N_g-1} P(I) log_2 P(I) \tag{4}$$

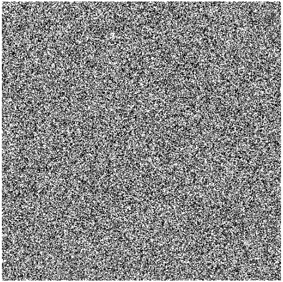


	Orig	Norm	-Skew	+Skew	Plat	Lept
μ_3	587	0	-169	169	0	0
μ_4	16609	7365	7450	7450	9774	1007
H	4.61	4.89	4.81	4.81	4.96	4.12

Second order statistics

- the second order statistics consider the structure of the data





- the images above have the same histogram, hence the same first order statistics

- **adjacency matrix** - for a given direction θ and distance d tells me how many times are two intensities in relation

$$0^\circ : P(I(m,n) = I_1, I(m \pm d, n) = I_2) = \\ = \frac{\text{number of pairs with intensities } I_1,\ I_2}{\text{total number of possible pairs}} \tag{5}$$

- θ is discretized into $\{0, 45, 90, 135\}$
- simpler form of the adjacency matrix is independent on θ and d
- in such case $P(i, j)$ tells us how many times pixels with intensity i are neighbors with pixels with intensity j (in the terms of probability)

DEPARTMENT OF CYBERNETICS

- **Angular Second Moment** - the level of smoothness

$$ASM = \sum_{i=0}^{N_g-1} \sum_{j=0}^{N_g-1} P(i,j)^2 \tag{6}$$

- **Contrast** - big values for large contrast

$$CON = \sum_{i=0}^{N_g-1} \sum_{j=0}^{N_g-1} |i-j|^2 \log_2 P(i,j) \tag{7}$$

- **Homogeneity** - big values for small contrast

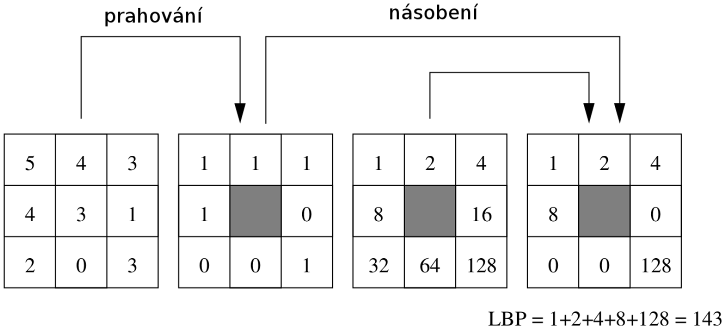
$$HOM = \sum_{i=0}^{N_g-1} \sum_{j=0}^{N_g-1} \frac{P(i,j)}{1+|i-j|^2} \tag{8}$$

- **Entropy** - big values for small contrast

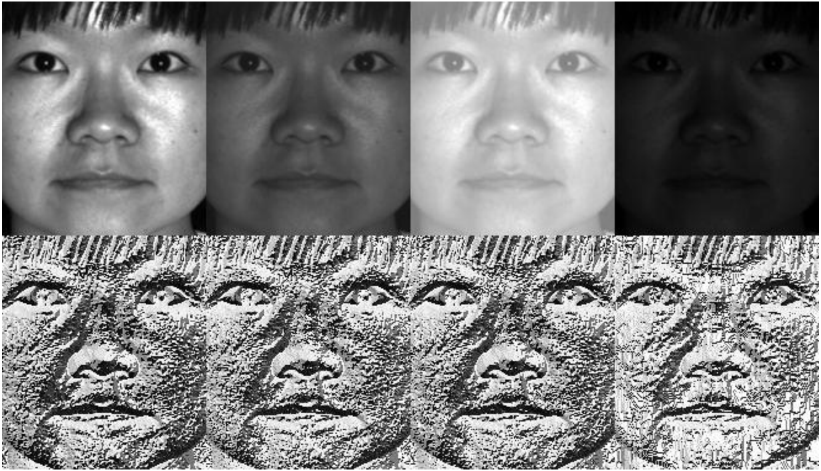
$$H_{xy} = -\sum_{i=0}^{N_g-1} \sum_{j=0}^{N_g-1} P(i,j) \log_2 P(i,j) \tag{9}$$

Local Binary Patterns

- Ojala 1996: How to describe texture around a pixel with one scalar?
- the basic version uses the 8-neighborhood of a pixel
- from this neighborhood a binary representation is build



- for a given image patch a histogram of LBP codes is constructed and used as a feature



Classification of LBP codes

- **Classification with non-parametric test** - the difference between an unknown sample and a model is computed (Kullback–Leibler divergence)

$$KL(S, M) = \sum_{b=1}^{B} S_b \log \frac{S_b}{M_b} \tag{10}$$

- and since S_b is constant we can simplify to

$$L(S, M) = -\sum_{b=1}^{B} S_b \log M_b \tag{11}$$

- which is cross entropy

Classification of LBP codes

- if low number of samples is available we can use equation:

$$\chi^2 = \sum_{b=1}^{B} S_b \frac{(S_b - M_b)^2}{S_b + M_b} \tag{12}$$

- or if we need computational efficiency

$$H(S, M) = \sum_{b=1}^{B} \min(S_b, M_b) \tag{13}$$

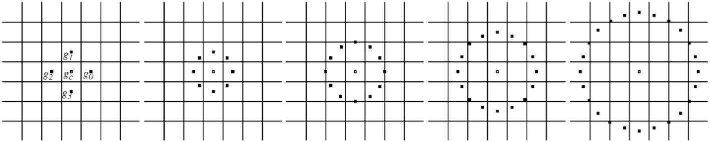
Extensions of LBP

- the basic LBP is invariant to brightness and contrast changes
- they are variant with scale and rotation - this is an issue
- texture definition:

$$T = t(g_c, g_0, \ldots, g_{P-1}) \tag{14}$$

- the position of pixels in the neighborhood is defined as:

$$g_p = (-R\sin(2\pi p/P), R\cos(2\pi p/P)) \tag{15}$$



(P=4,R=1.0) (P=8,R=1.0) (P=12,R=1.5) (P=16,R=2.0) (P=24,R=3.0)

$$LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c)2^p \tag{16}$$

Rotation invariance and uniformity

- is achieved by rotating the local neighborhood

$$LBP_{P,R}^{ri} = \min\left(ROR(LBP_{P,R}, i) | i = 0, 1, \ldots, P-1\right) \quad (17)$$

- ROR is a bitwise rotation operator
- **uniform patterns** - are patterns with at most 2 changes between 0 and 1
- there are a total of 58 uniform patterns, while the rest are put into 59^{th} bin
- Multi-resolution analysis - is used to cope with the resolution (scale) of the image

$$L_N = \sum_{n=1}^{N} L(S^n, M^n) \quad (18)$$

- with different P and R for each n

DEPARTMENT OF CYBERNETICS

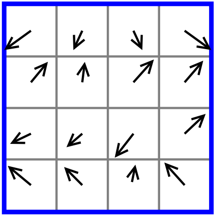
Histogram of Oriented Gradients

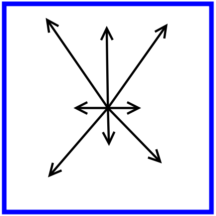
- method for describing images via histogram analysis
- the results of the method are directly dependent on the gradient operator, many had been tested
- the best results were obtained for simple gradient approximation

$$\begin{aligned} I_x &= I * [-1, 0, 1] \\ I_y &= I * [-1, 0, 1]^T \end{aligned} \tag{19}$$

- for every pixel the size and orientation of the gradient is computed
- a histogram is constructed from these values
- the histogram is parametrized by interval i and number of sectors s
- the interval i is mostly $i =< 0, \pi >, i =< 0, 2\pi >$
- the magnitude of the histogram is added to each bin and moreover bilinearly distributed into neighboring bins

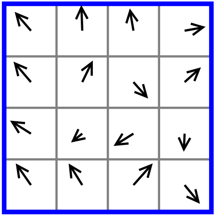
DEPARTMENT OF CYBERNETICS

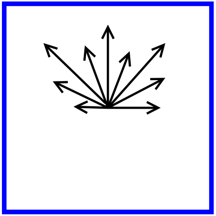




$i = \langle 0, 2\pi \rangle$

$s = 8$





$i = \langle 0, \pi \rangle$

$s = 9$

Histogram Normalization

- the normalization is useful to cope with brightness transformations
- the most used normalizations are

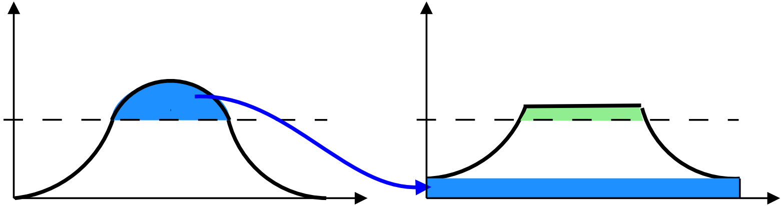
$$L^1 - norm = \frac{v}{(\|v\|_1 + e)}, \quad (20)$$

$$L^1 - sqrt = \sqrt{\frac{v}{(\|v\|_1 + e)}}, \quad (21)$$

$$L^2 - norm = \frac{v}{\sqrt{(\|v\|_2^2 + e^2)}}, \quad (22)$$

- v is the histogram to be normalized and e is a small constant
- a special case of normalization $L_2 - Hys$ - the normalized vector is clipped as in CLAHE

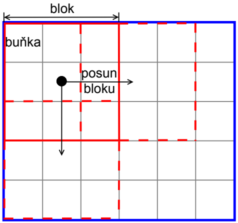
DEPARTMENT OF CYBERNETICS



- the values (after normalization) above a given threshold are distributed into all the bins
- the process is repeated until no value is above the threshold

HoG descriptor

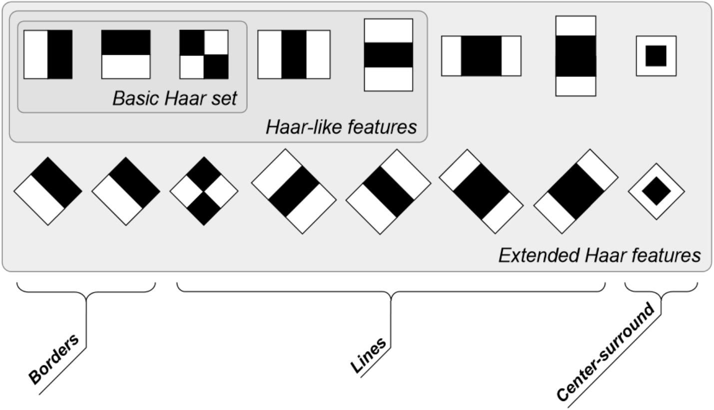
- the image is divided into blocks of size (k, k)
- individual blocks are divided into cells of size (l, l)



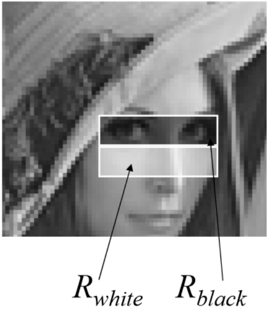
- for each cell the normalized histograms of gradients are computed
- for each block the cell histograms are averaged
- then the block shifts by some pixels and the process is repeated
- the averaged block histograms are concatenated to obtain the descriptor

Haar-like features

- Haar-like features are used for image transformation similar to cosine transform

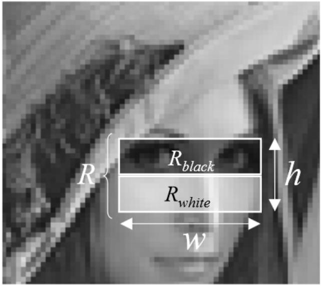


- the Haar-like filters can be computed efficiently by using integral image
- the black regions are subtracted from the white regions



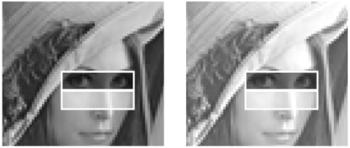
$$F_{Haar} = E(R_{white}) - E(R_{black})$$

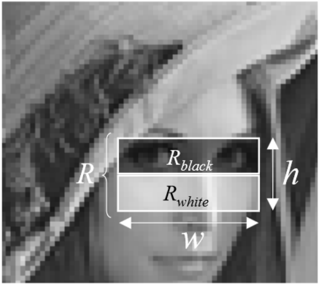
Normalization (monotonic illumination changes)



$$F_{Haar} = \frac{E(R_{black}) - E(R_{white})}{\underbrace{\sqrt{\left|E(R_{\mu})^2 - E(R_{\mu}{}^2)\right|}}_{Standard\ Deviation}}$$

Normalization (monotonic illumination changes and scale)

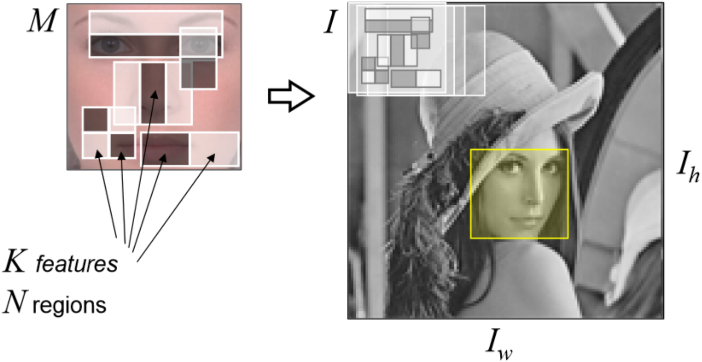




$$F_{Haar} = \frac{E(R_{black}) - E(R_{white})}{w \cdot h \cdot \sqrt{\left| E(R_{\mu})^2 - E(R_{\mu}{}^2) \right|}}$$

Face detection

- a sliding window in different scales is used to compute responses on different Haar filters



- a boosted classifier is used to train the right responses to certain (most informative) filters