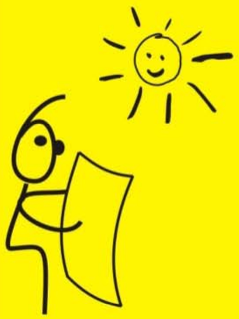


SSIP 2011

19th Summer School on Image Processing

7 July - 16 July 2011 • Szeged, Hungary

Image processing in the compressed domain

Camelia Florea, Ph.D

Camelia.Florea@com.utcluj.ro

Centre for Multimedia Technologies and Distance Education,
Technical University of Cluj-Napoca,
15, C. Daicoviciu Street, 400020, Cluj Napoca, ROMANIA.

Brief Overview

- In recent years,
  - the resolution of images increased

- Digital libraries manipulate huge amounts of image data
  - due to the limitations of space and time,
    the images are represented in compressed domain

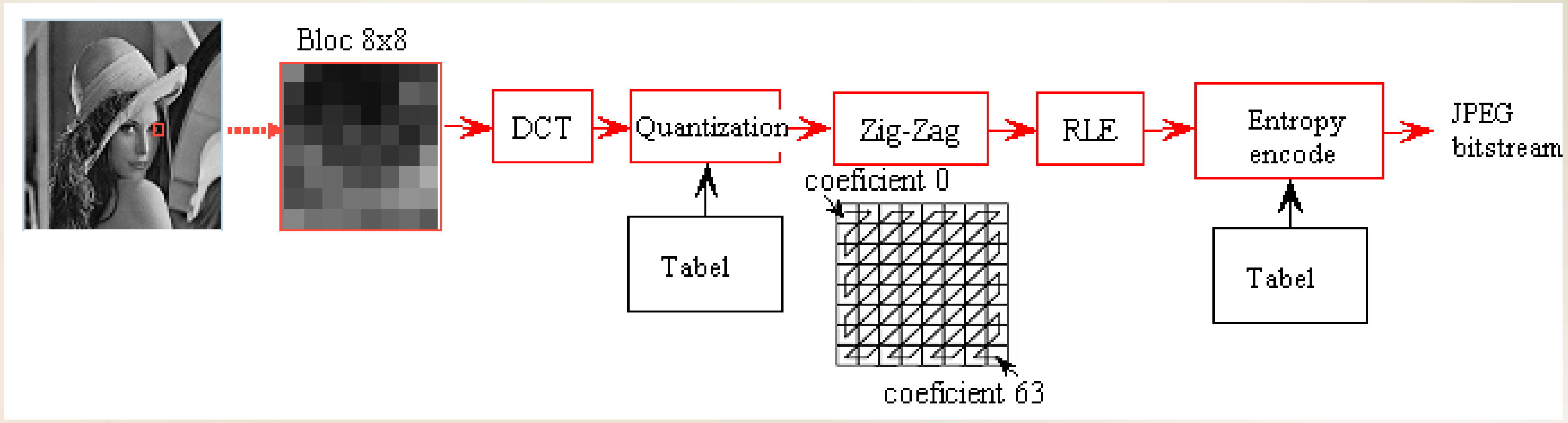
=> techniques used for processing, segmentation, classification or indexing images ***directly in the compressed domain*** have become one of the most important topics in digital libraries.

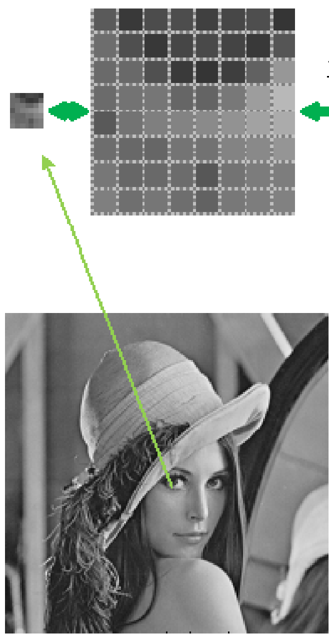
Brief Overview

- Among compressed domain, JPEG format has been used more than others (more than 95 % of images on the web are in JPEG compressed format)
- Even medical images are stored in JPEG format, by reason of the advantages offered by this format:
  - low storage capacity needed, and
  - better performances in information transmission (via internet).
- Compressed domain image processing algorithms in encoded JPEG image domain - provide a powerful computational alternative to classical.
  - no need to decompress/ recompress the whole image prior to processing/after processing.
  - we compute less data (after quantization many of the DCT coefficients are zero).

JPEG standard coding

- The color space of the image is converted (RGB to YUV)
- Values are scaled symmetrical towards 0 (from [0, 255] to [-128, 127])
- The image is divided into 8×8 blocks
- Each 8×8 block is processed for compression
  - DCT is applied on each block => obtain the DCT coefficients (DC and AC)
  - DCT coefficients are quantized - small coefficients are quantized to zero
  - zig-zag scan of the DCT blocks
  - RLE (Run Length Encoding) is performed
  - finally - entropy coding





Y

90	57	76	77	64	71	78	53
109	79	56	70	73	68	57	85
106	105	80	58	53	62	97	151
107	115	118	102	100	120	157	184
89	115	127	131	138	146	162	171
109	104	102	116	114	131	137	149
119	103	105	108	87	115	124	125
127	111	118	110	98	120	125	129

Y_S

-38	-71	-52	-51	-64	-57	-50	-75
-19	-49	-72	-58	-55	-60	-71	-43
-22	-23	-48	-70	-75	-66	-31	23
-21	-13	-10	-26	-28	-8	29	56
-39	-13	-1	3	10	18	34	43
-19	-24	-26	-12	-14	3	9	21
-9	-25	-23	-20	-41	-13	-4	-3
-1	-17	-10	-18	-30	-8	-3	1

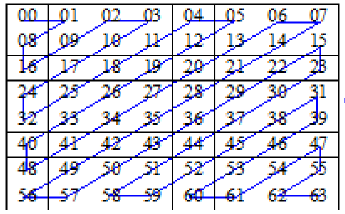
Y_{DCT}

-181.12	-58.32	73.49	-18.54	7.62	11.75	9.20	-7.35
-134.15	30.04	12.30	3.86	15.69	-5.92	-2.77	6.69
-107.32	70.10	-19.33	33.69	6.99	23.20	10.06	-1.74
6.09	-11.82	-55.59	21.18	-3.29	15.1	-2.20	-0.31
53.62	-23.28	-27.78	-7.03	-19.12	11.52	0.54	10.92
-6.22	-2.55	20.13	-10.91	-12.62	15.78	1.67	2.66
-7.90	-3.07	15.56	-9.11	-3.03	-0.86	-0.41	3.14
-8.01	0.18	-0.99	-11.05	-7.15	-0.77	-2.87	1.97

4	3	2	4	6	10	13	16
3	3	3	5	7	14	15	14
2	3	4	6	10	14	17	14
4	5	6	9	13	22	20	16
6	7	10	13	17	28	26	19
10	14	14	22	28	26	28	23
13	15	17	20	26	28	30	26
16	14	14	16	19	23	26	25

Y_Q

-45	-19	37	-5	1	1	1	0
-45	10	4	1	2	0	0	0
-54	23	-5	6	1	2	1	0
2	-2	-9	2	0	1	0	0
9	-3	-3	-1	-1	0	0	1
-1	0	1	0	0	1	0	0
-1	0	1	0	0	0	0	0
-1	0	0	-1	0	0	0	0



00	01	02	03	04	05	06	07
08	09	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63

Y_{ZZ}

[-45 -19 -45 -54 10 37 -5 4 23 2 9 -2 -5 1 1 1 2 6 -9 -3 -1 -1 0 -3 2 1 0 1 0 0 2 0 -1 1 0 -1 0 1 0 -1 1 1 0 0 0 0 0 0 0 -1 0 1 0 0 1 0 0 0 0 0 0 0 0 0]

Y_{RLE}

[-45 0 -19 0 -45 0 -54 0 10 0 37 0 -5 0 4 0 23 0 2 0 9 0 -2 0 -5 0 1 0 1 0 1 0 2 0 6 0 -9 0 -3 0 -1 0 -1 1 -3 0 2 0 1 1 1 2 2 1 -1 0 1 1 -1 1 1 1 -1 0 1 0 1 7 -1 1 1 2 1 0 0]

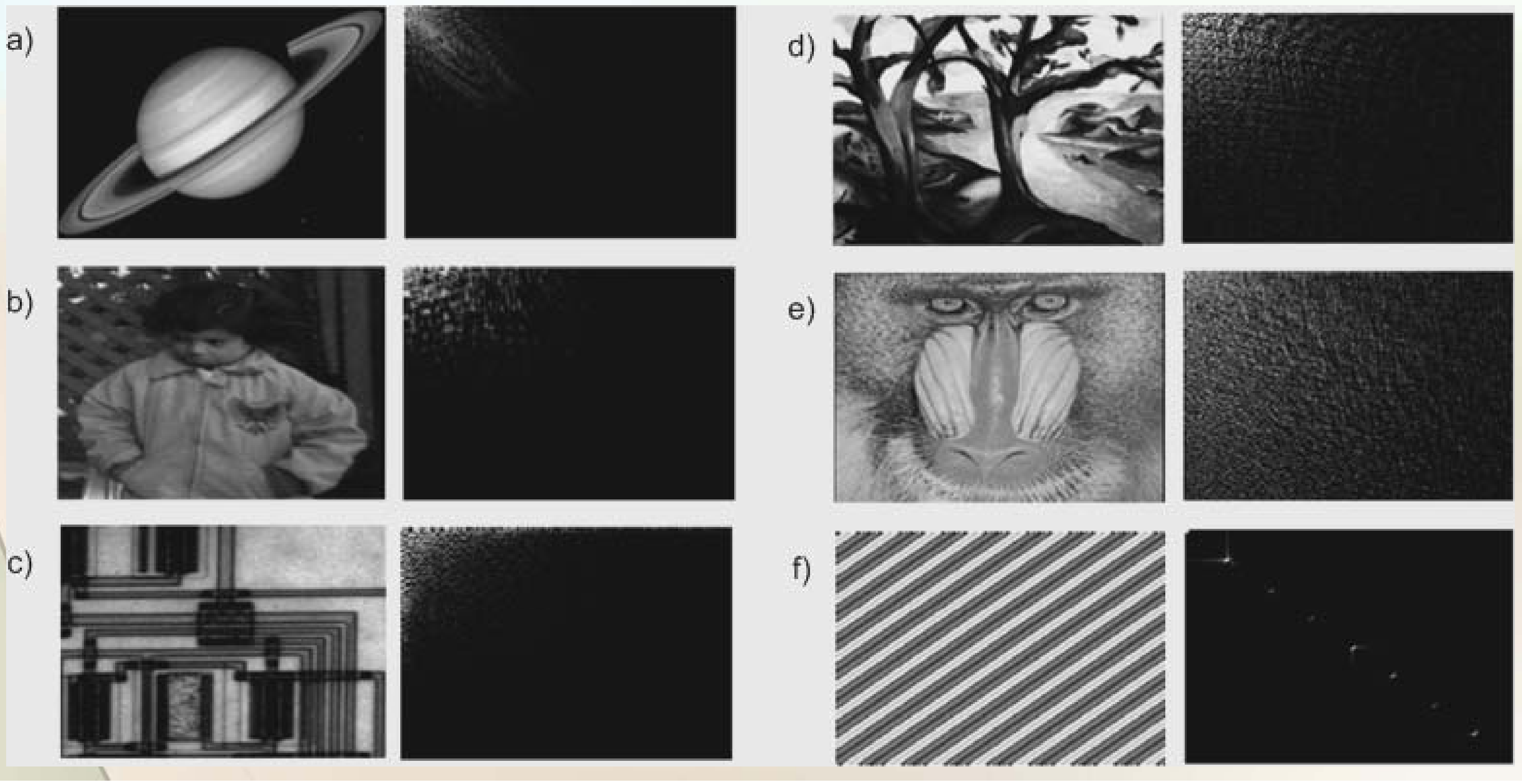
DCT Coefficients

- There are many local texture features embedded in the DCT coefficients, reflecting color and texture:
    - the DC coefficient - the average color in a block of pixels,
    - the AC coefficients - the variance of luminance and chrominance

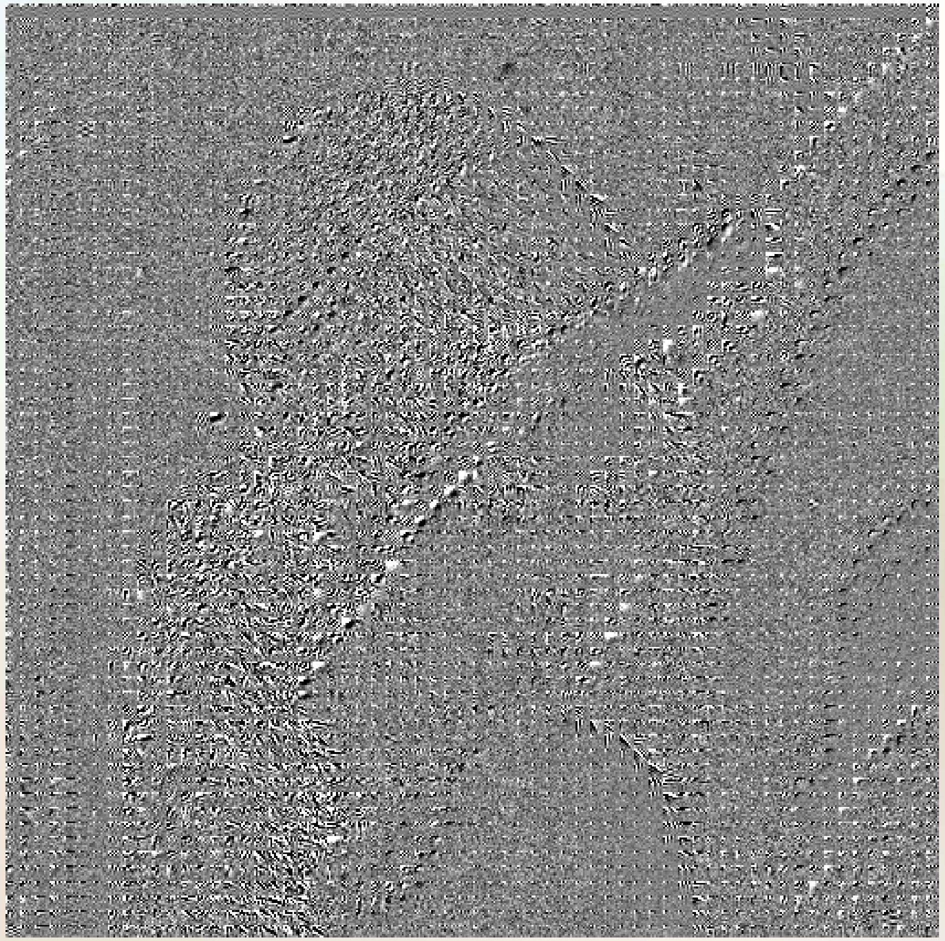
$$y(k,l)=\frac{2}{\sqrt{N\cdot M}}\cdot c(k)\cdot c(l)\cdot\left[\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}x(i,j)\cdot\cos\frac{(2\cdot i+1)\cdot k\cdot\pi}{2\cdot M}\cdot\cos\frac{(2\cdot j+1)\cdot l\cdot\pi}{2\cdot N}\right]$$

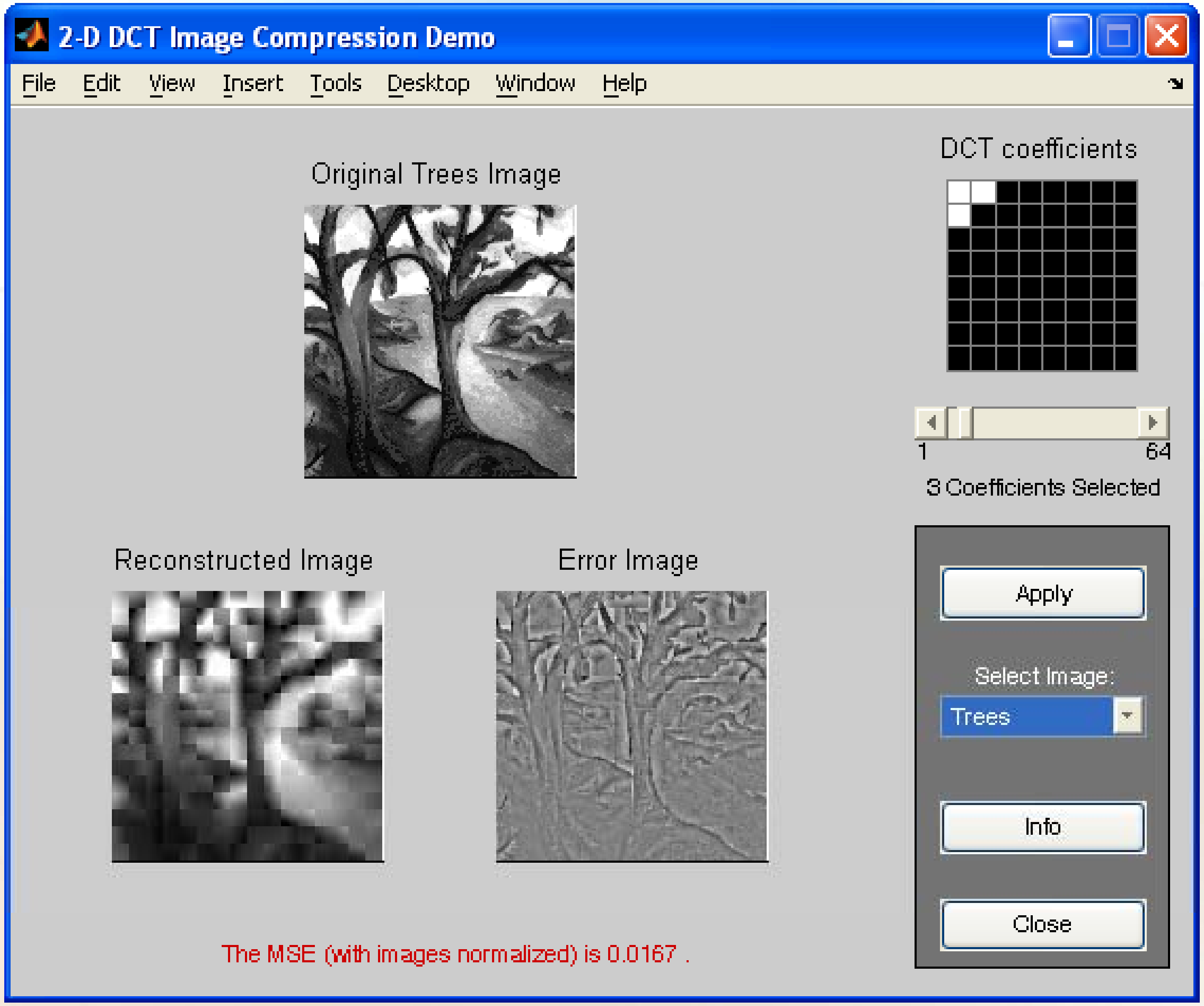
$$\textit{where}\quad k=0,1,\ldots,M-1;\ l=0,1,\ldots,N-1,$$

$$\textit{and}\quad c(k)=\begin{cases}\frac{1}{\sqrt{2}}, & \textit{if } k=0;\\ 1, & \textit{else}\end{cases}$$



The image in the JPEG compressed domain

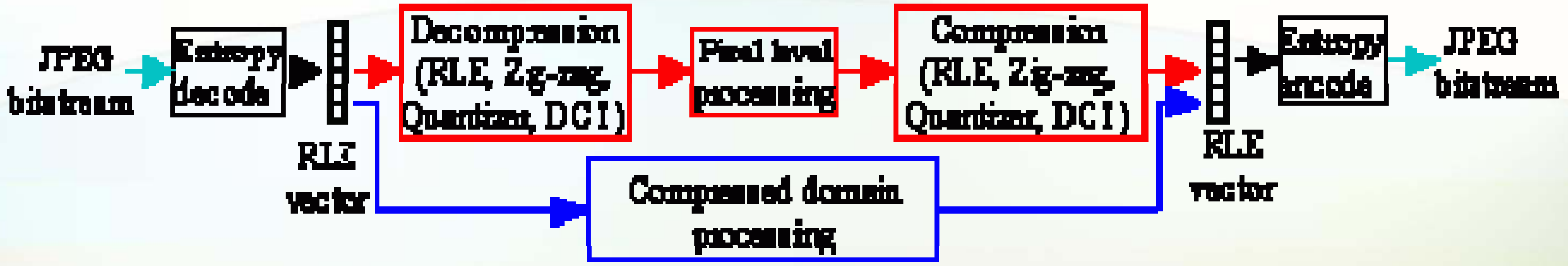




3 Coefficients Selected

The MSE (with images normalized) is 0.0167 .

The two ways to process JPEG compressed images


JPEG bitstream
Entropy decode
RLE vector
Decompression (RLE, Zig-zag, Quantizer, DCT)
Pixel level processing
Compression (RLE, Zig-zag, Quantizer, DCT)
Compressed domain processing
RLE vector
Entropy encode
JPEG bitstream

- Having a JPEG compressed image is more efficiently to process it, without:
  - performing decompression, pixel level processing, and recompression.

- Processing in the compressed domain is made over the RLE vectors.
- RLE vector contains data about the variation and mean of luminance/color

Linear and nonlinear image processing operations in compressed domain

- To obtain in the compressed domain the same processing results as the one given by the pixel-level approach

  - the algorithm must be reformulated as a block level processing.

- The linear operations can be easily implemented

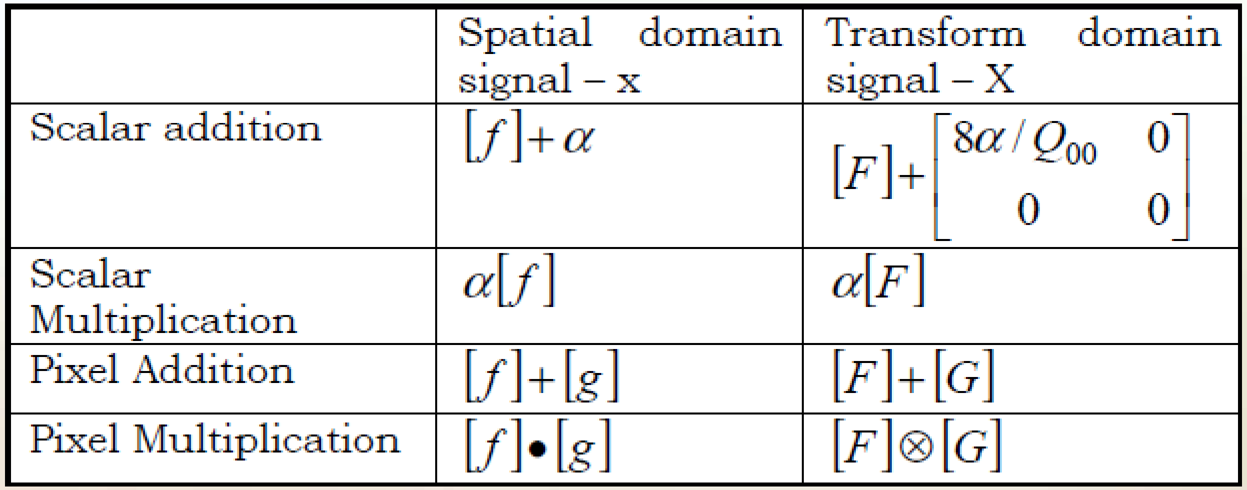
- The implementation of operations involving nonlinear operators

  is challenging, but not impossible, and

  when the right combination is found, the processing is much faster

Linear Operations in the compressed domain

- Mostly linear algorithms - pixel arithmetic:
  - Pointwise image addition/substraction
  - Constant addition/substraction to each spatial position
  - Constant multiplication to each spatial position
  - Pointwise image multiplication

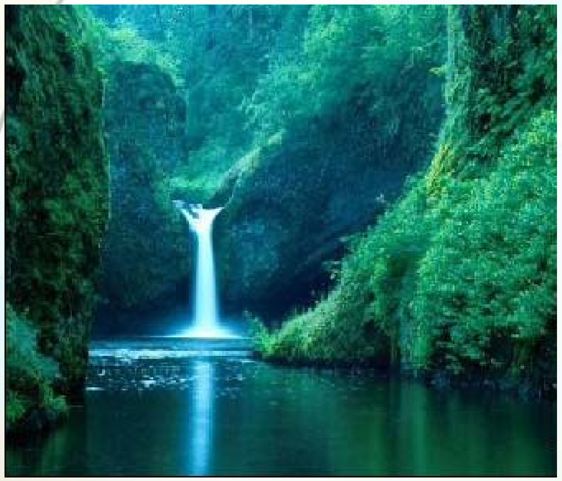


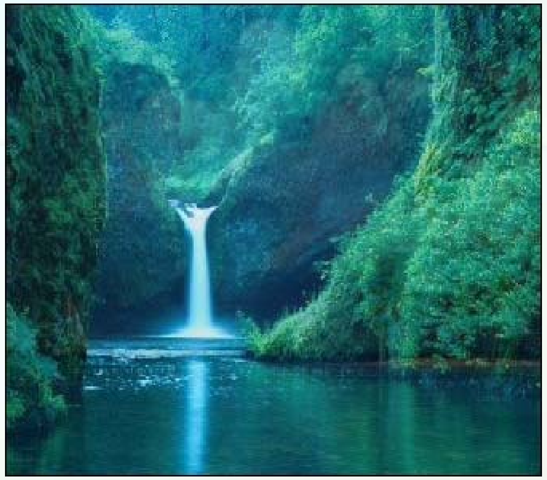
	Spatial domain signal – x	Transform domain signal – X
Scalar addition	$[f]+\alpha$	$[F]+\begin{bmatrix} 8\alpha/Q_{00} & 0 \\ 0 & 0 \end{bmatrix}$
Scalar Multiplication	$\alpha[f]$	$\alpha[F]$
Pixel Addition	$[f]+[g]$	$[F]+[G]$
Pixel Multiplication	$[f]\bullet[g]$	$[F]\otimes[G]$

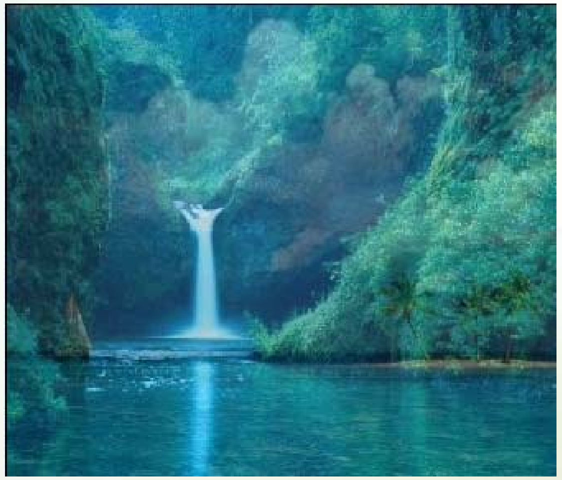
- Pixel arithmetic can be used to implement a number of operations
  - cross-fade between two images or video sequences
  - image composition - overlaying a forecaster on a weather map
  - implementation of contrast enhancement algorithms , etc.

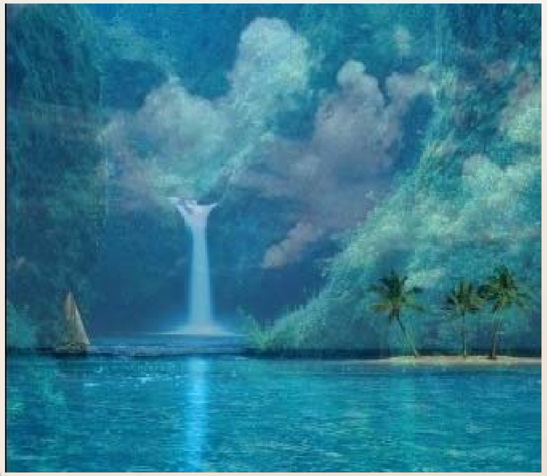
Alpha-blending between two images

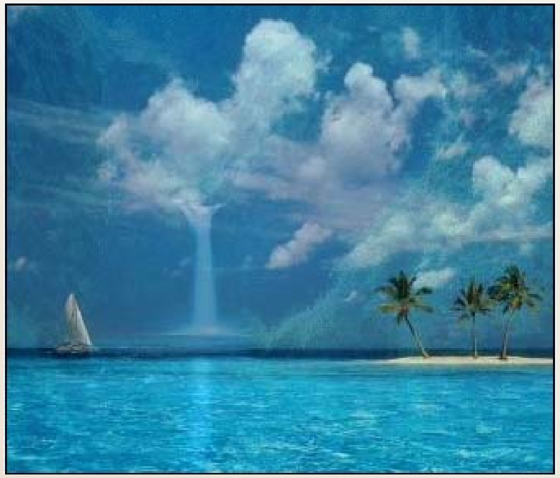
$$g(x, y) = \alpha f_2(x, y) + (1 - \alpha) f_1(x, y)$$


$\alpha = 0$


$\alpha = 0.2$


$\alpha = 0.4$


$\alpha = 0.6$


$\alpha = 0.8$

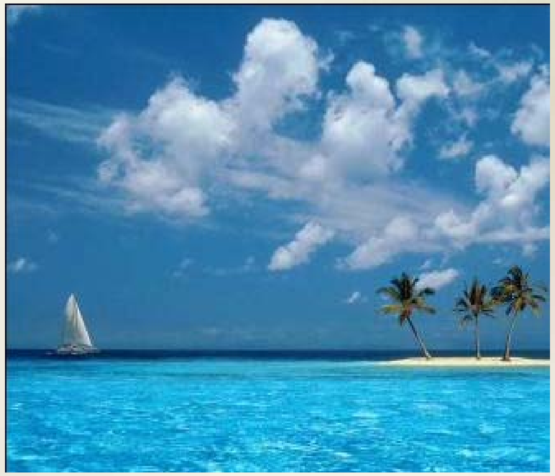
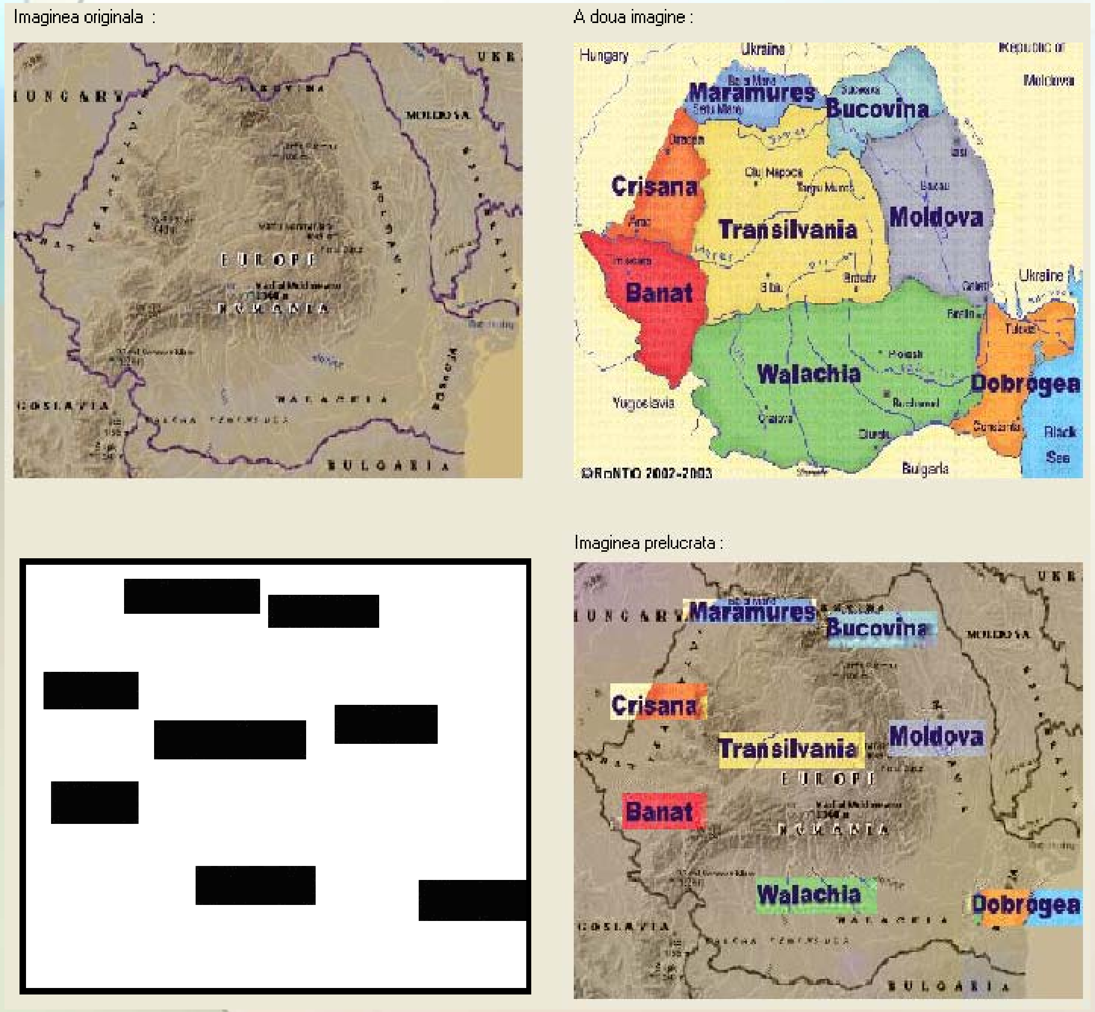

$\alpha = 1$

Image composition

$$g(x,y) = m(x,y)f(x,y) + (1 - m(x,y))b(x,y)$$



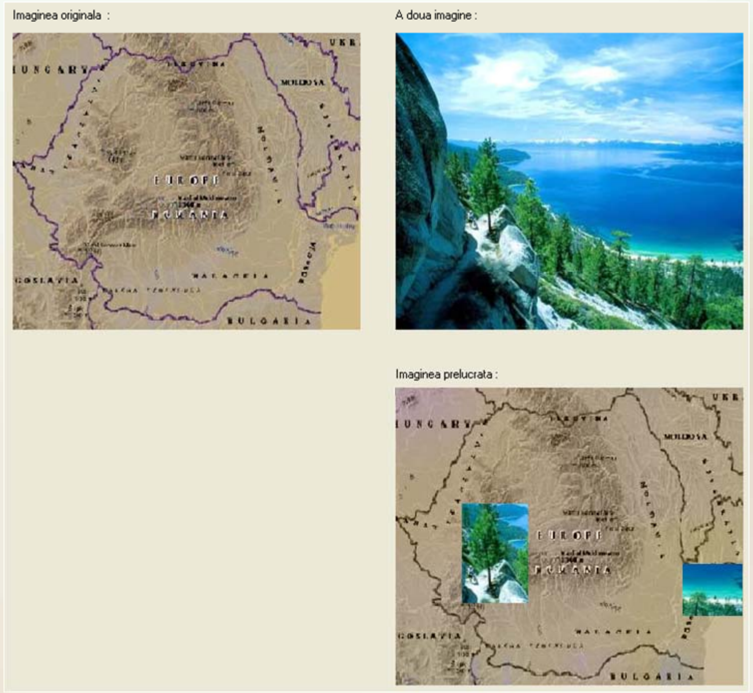


Image Invert

$$g(x,y) = 256 - f(x,y) \quad | \quad -128$$
$$g(x,y) - 128 = 256 - f(x,y) - 128$$
$$g(x,y) - 128 = 256 - f(x,y) - 128$$
$$g(x,y) - 128 = 128 - f(x,y)$$
$$g(x,y) - 128 = -(f(x,y) - 128)$$
$$DCT(g - 128) = -DCT(f - 128)$$

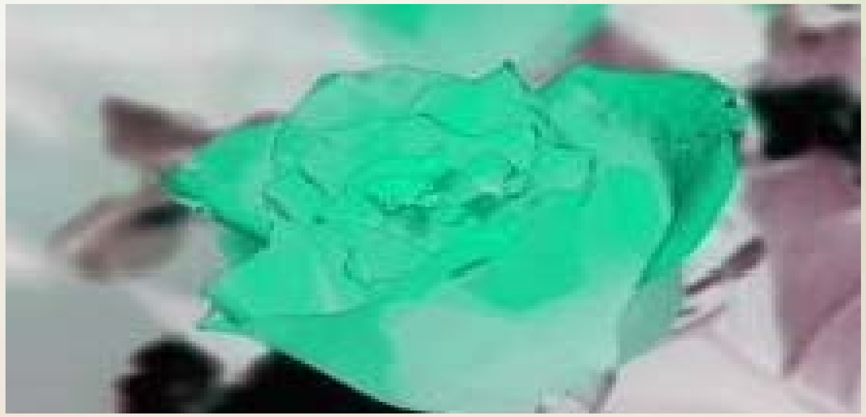
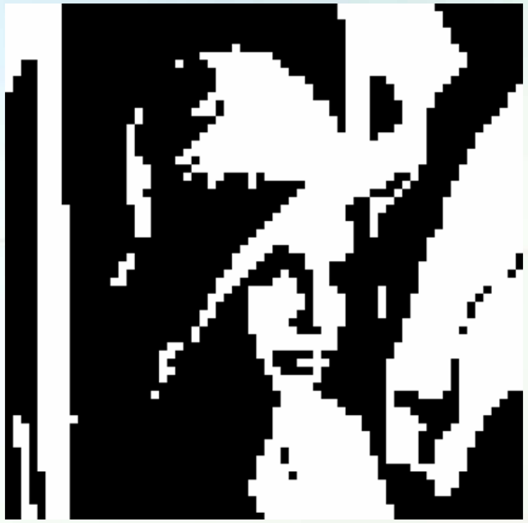
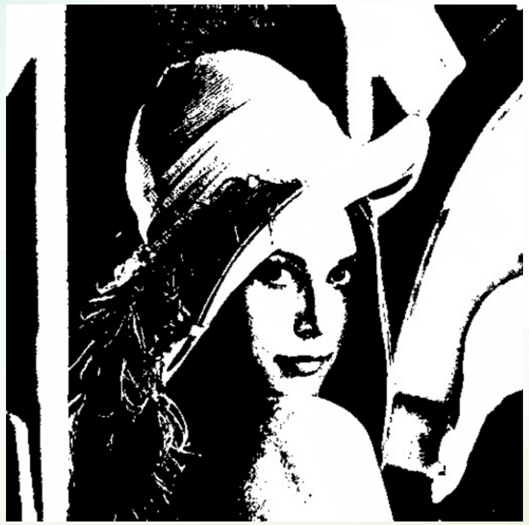


Image Binarization – nonlinear operation

Original Image
“Lena.jpg”;


Compressed domain
image processing


Pixel level image
processing

- no need to decompress/ recompress the whole image prior to processing/after processing.

- for the 8×8 blocks - instead of 64 comparisons needed at pixel level, for a block processed in the compressed domain the processing implies a single comparison of the coefficient.

An adaptive algorithm

- The DC coefficient gives the average brightness in the block

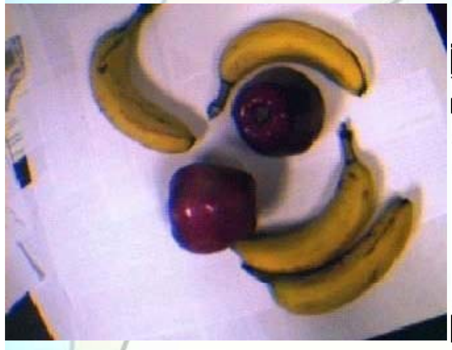
  - is used as an estimate for selecting the processing rule for all the pixels in the blocks with small AC energy.

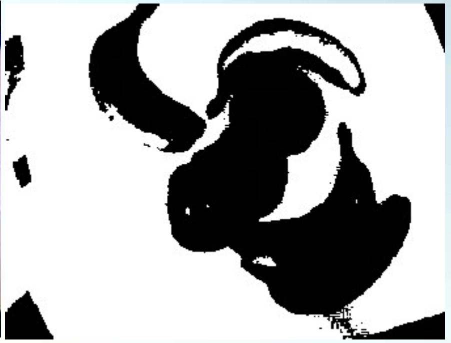
- In this algorithm an adaptive minimal decompression is used:

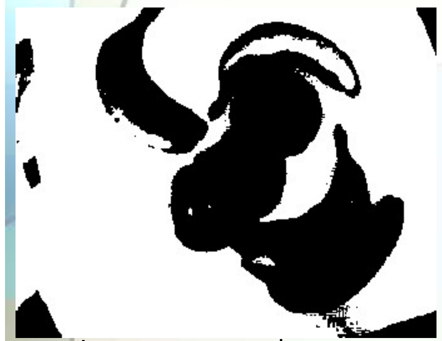
  - full decompression is no longer needed,

  - but, decompression is used for the block having many details, for an improved accuracy of processing.

Image processed in compressed domain







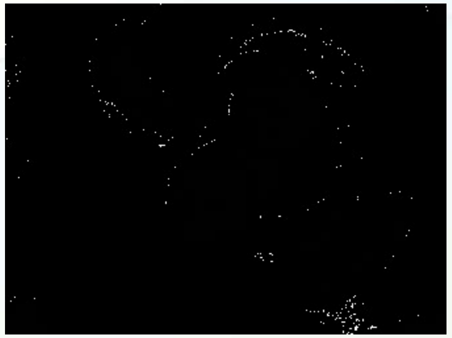
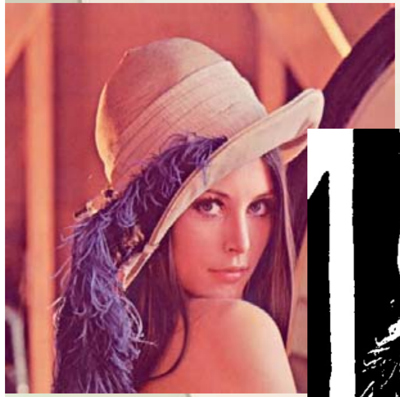


Image processed at pixel level

Image	EffBlocks [%]	MSE [%]
Img1.jpg	13.5	1.45
Img2.jpg	16.07	0.48
Img3.jpg	24	0.012
Img4.jpg	18.24	1.45

- The Mean Squared Error (MSE) between the pixel level processed images and the images processed in the compressed domain was used as quality performance measure.

- The efficiency (*EffBlocks*) of the method formulated above for the compressed domain, is evaluated by examining the number of blocks processed at pixel level as percent from the total number of 8×8 pixels blocks in the image.



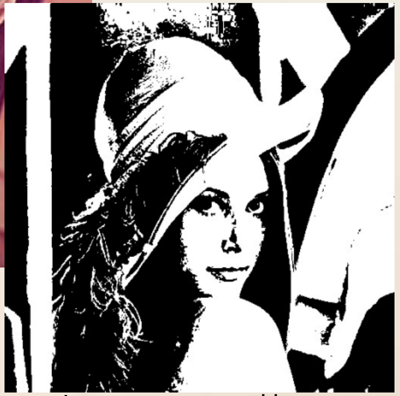


Image processed in compressed domain

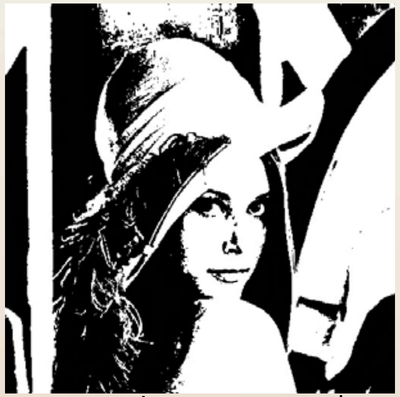


Image processed at pixel level

The DC histogram in the compressed domain

- Roughly speaking, if the DC coefficients would be the only ones used to reconstruct the pixel level representation (without any AC information),

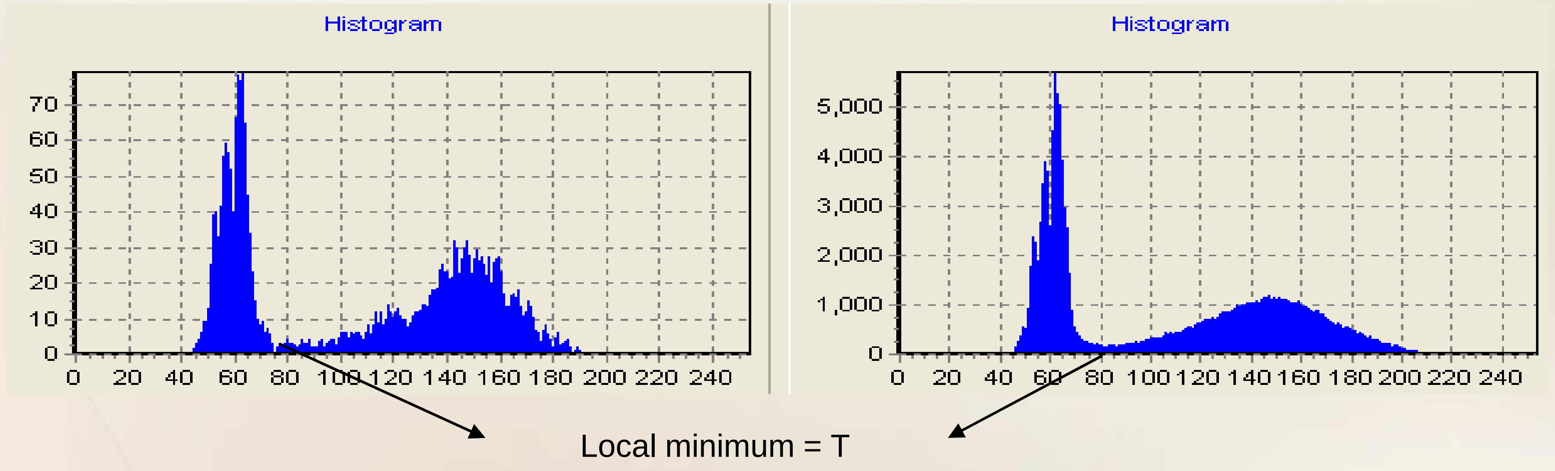
  - they would give an approximation of the image,

  - with some block boundary effects and some loss of details,

  - but, however still preserving the significant visual information.

- Therefore,

  - the histogram built only from the DC coefficients will have approximately the same shape as the grey level histogram.



Histogram of DC coefficients, and at pixel level

Compressed domain implementation of fuzzy rule-based contrast enhancement

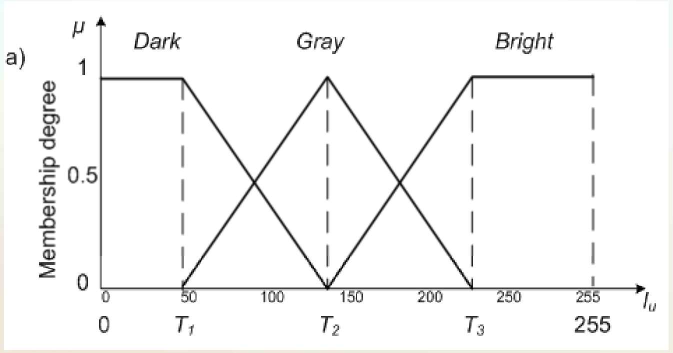
The fuzzy rule base of the Takagi-Sugeno fuzzy systems comprises the following 3 rules:

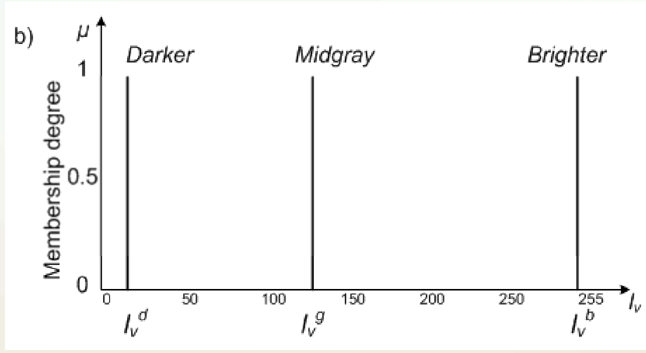
R1: IF l_u is *Dark* THEN l_v is *Darker* R1: IF l_u is *Dark* THEN l_v=l_v^d

R2: IF l_u is *Gray* THEN l_v is *Midgray* ⇔ **R2: IF l_u is *Gray* THEN l_v=l_v^g**

R3: IF l_u is *Bright* THEN l_v is *Brigter*, **R3: IF l_u is *Bright* THEN l_v=l_v^b**

Input and output membership functions for fuzzy rule-based contrast enhancement:





For any value l_u^* at the input of our Takagi-Sugeno contrast enhancement fuzzy system, in the output image, the corresponding brightness l_v^* is obtained by applying the Takagi-Sugeno fuzzy inference, as:

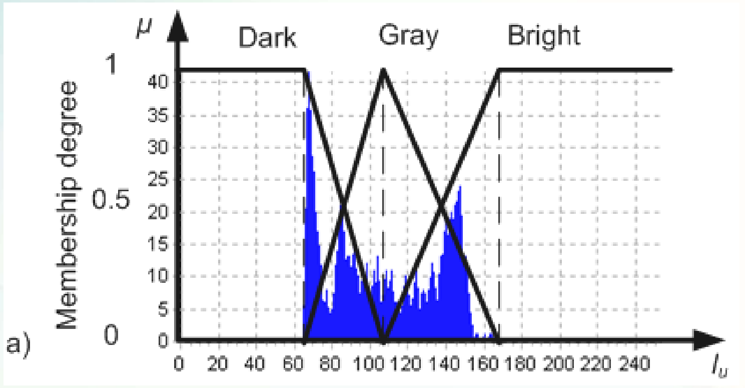
$$l_v^* = \frac{\mu_{Dark}(l_u^*)\cdot l_v^d + \mu_{Gray}(l_u^*)\cdot l_v^g + \mu_{Bright}(l_u^*)\cdot l_v^b}{\mu_{Dark}(l_u^*) + \mu_{Gray}(l_u^*) + \mu_{Bright}(l_u^*)},$$

$$\mu_{Dark}(l_u) + \mu_{Gray}(l_u) + \mu_{Bright}(l_u) = 1$$

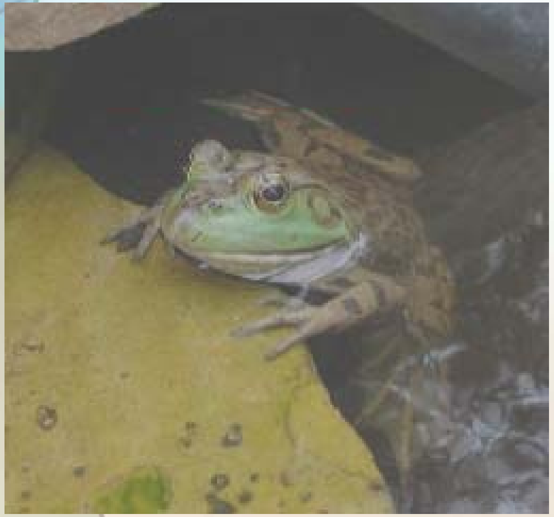
Where: $\mu_{Dark}(l_u^*), \mu_{Gray}(l_u^*), \mu_{Bright}(l_u^*)$ denote the membership degrees of the currently processed brightness to the input fuzzy sets *Dark, Gray* and *Bright.*

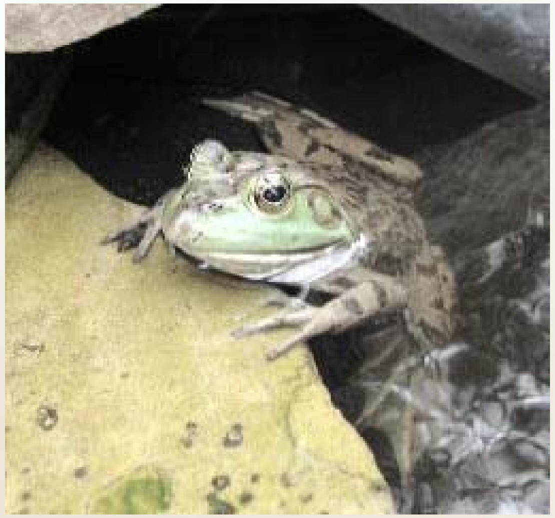
Experimental results

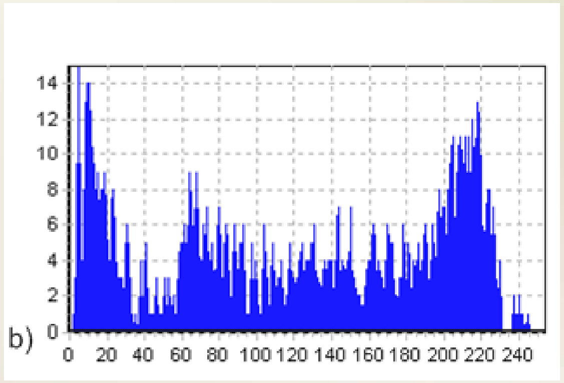
- The algorithm is applied only on the luminance component.
- However, it can be used to enhance color images as well, with no change of the chrominance components



Input membership function superimposed on the DC histogram of the Y component.
A reasonable choice for the thresholds values would be the minimum, the mean and the maximum grey level from the image histogram.





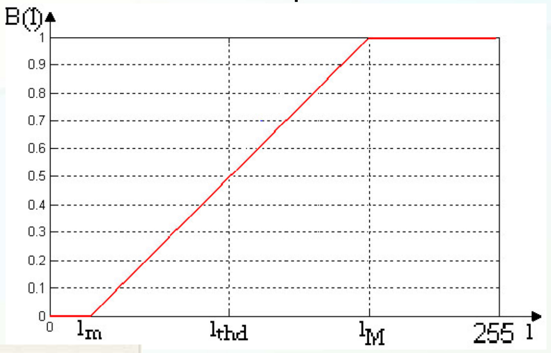


DC histogram of *the Y* component after fuzzy contrast enhancement

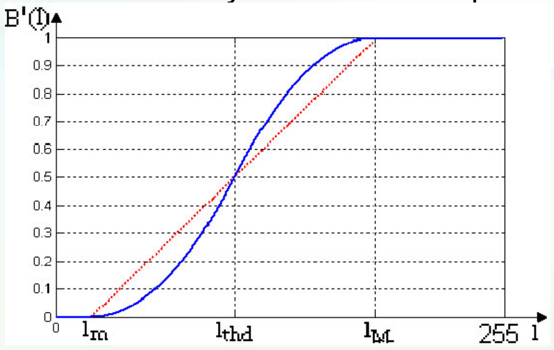
Contrast enhancement with fuzzy INT

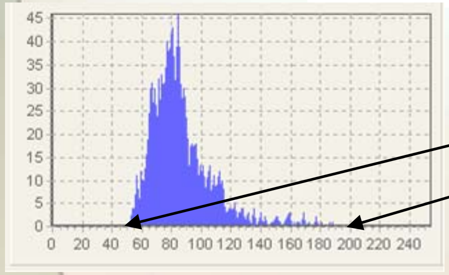
Analytical and graphical representation of the fuzzy contrast enhanced function:

The linear slope



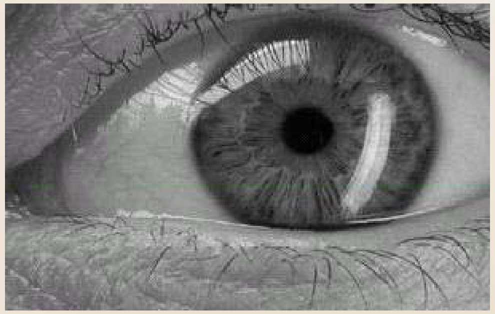
The fuzzy intensification operator

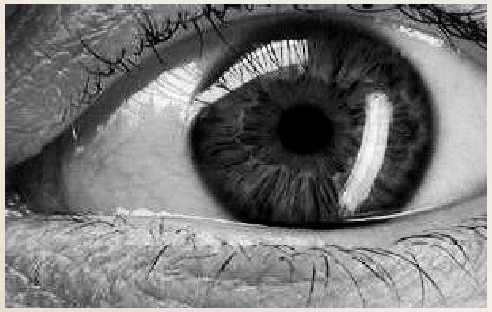




$$[l_{\min}, l_{Max}] \subseteq [l_m, l_M],$$

$$B'(l) = INT\{B(l)\} = \begin{cases} 2 \cdot (a \cdot l + b)^2, & dacă\ 0 \le a \cdot l + b \le 0.5 \\ 1 - 2 \cdot (-a \cdot l + 1 - b)^2, & dacă\ 0.5 \le a \cdot l + b \le 1 \end{cases}$$





Geometrical Transformation of Images

- Techniques for manipulating image in the transform compressed domain for special visualization effects: e.g., image scaling, translation, rotation, warping.
  - It works by rearranging the compressed data (DCT coefficients), without ever fully decoding the image.
- These algorithms are applicable to any orthogonal separable transforms such as DCT, DFT, ...

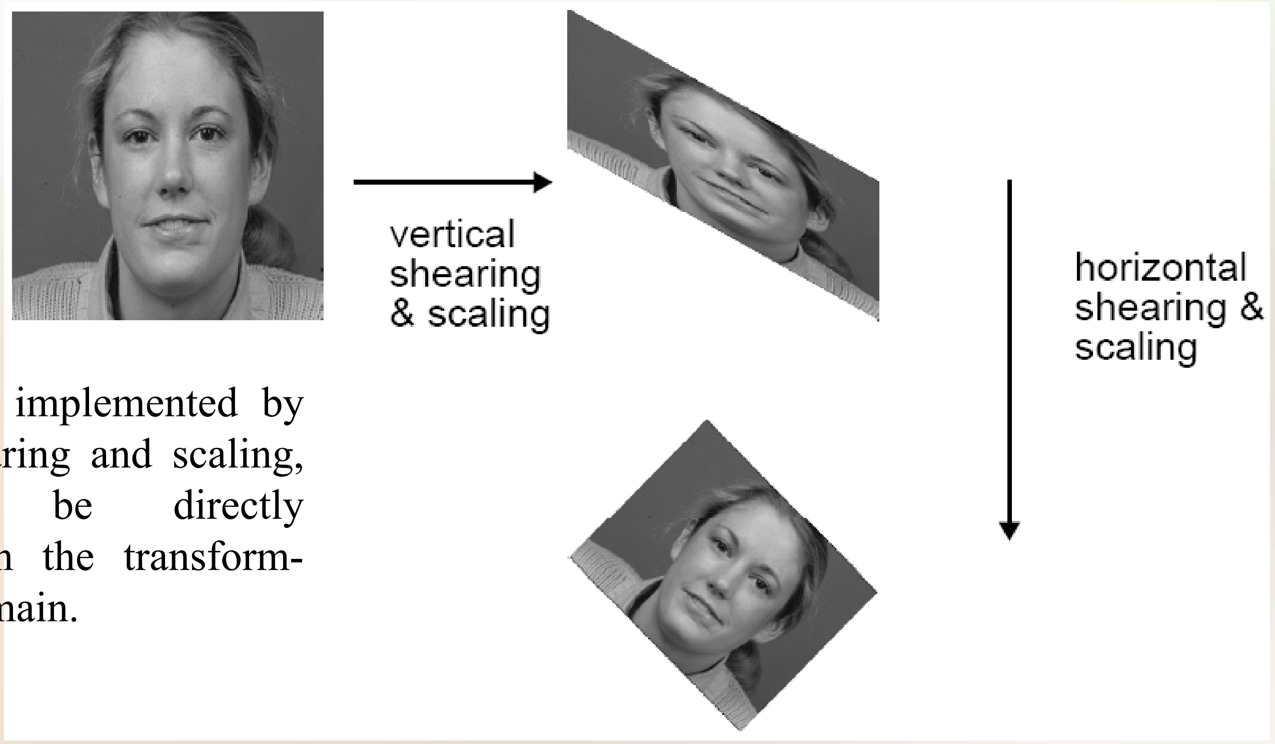


Image rotation implemented by multi-pass shearing and scaling, which can be directly implemented in the transform-compressed domain.

JPEG features space for image segmentation

- There are many local texture features embedded in the DCT coefficients, reflecting color and texture in spatial image windows:
  - The DC coefficient represents the average luminance /color in a block of pixels, while
  - the AC coefficients reflect the variance of luminance and chrominance changes in pixels within the same block.

- The DCT - is one of the best filters for feature extraction in the frequency domain

- Image segmentation is the technique of partitioning an image into units which are homogeneous with respect to one or more characteristics.
  - This can be done on pixel level for highest accuracy,
  - but also on JPEG block level, in the compressed domain, considering the local color and texture information roughly needed, which is completely present in this representation.

- Dimensionality reduction is essential for extracting effective features and reducing computational complexity in classification stage.

- This reduced dimensionally feature space makes easier the training and implementation of rather complex classifiers,
    - as e.g. systems based on the PCA and LDA; the Bayesian classifier with class probabilities modeled by Gaussian mixtures; Support Vector Machine classifier; etc.

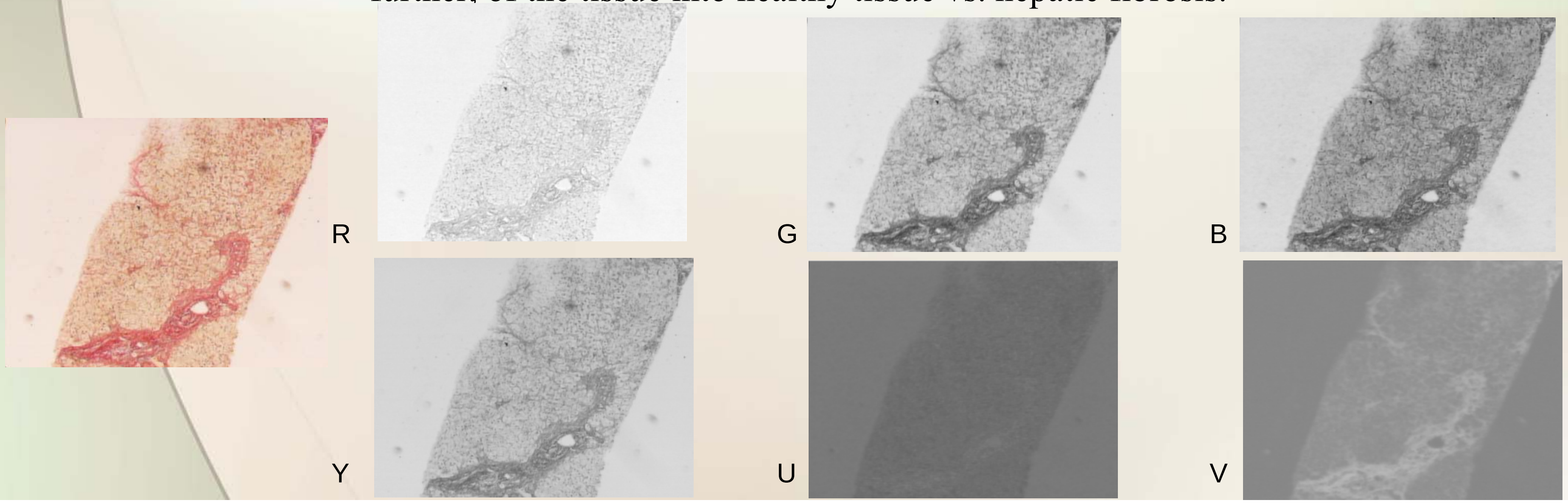
- There are several advantages if such systems can be implemented in the block DCT domain:

(1) If the database is stored as compressed JPEG images to reduce storage requirement, DCT coefficients of JPEG images can be directly used;

(2) In JPEG images, some redundant information has been removed by quantization - such that the dimensionality of data can be initially reduced by removing coefficients with less information.

Color in YUV vs. RGB space

- Many image representation spaces can be used in segmentation process.
- The YUV representation yields certain advantages over RGB:
  - The YUV is the representation used in the JPEG standard,
  - The YUV provides a clear separation between the luminance representation (Y) and color (U,V),
  - The luminance information Y, the color information U and V exhibits poor correlation
- The image storage format itself provides the information needed for an accurate identification/segmentation
  - e.g.: segmentation of the hepatic biopsies into tissue vs. microscopic slide and further, of the tissue into healthy tissue vs. hepatic fibrosis.



Bayesian classification of pixel block in the discrete cosine transform domain

- A powerful yet simple model for blocks classification is the Multivariate Gaussian model:

$$p(X \mid C_i) = \frac{1}{\sqrt{(2\pi)^{64}|\Sigma_i|}} e^{-\frac{1}{2}(X-m_i)^T \Sigma_i^{-1} (X-m_i)} ,$$

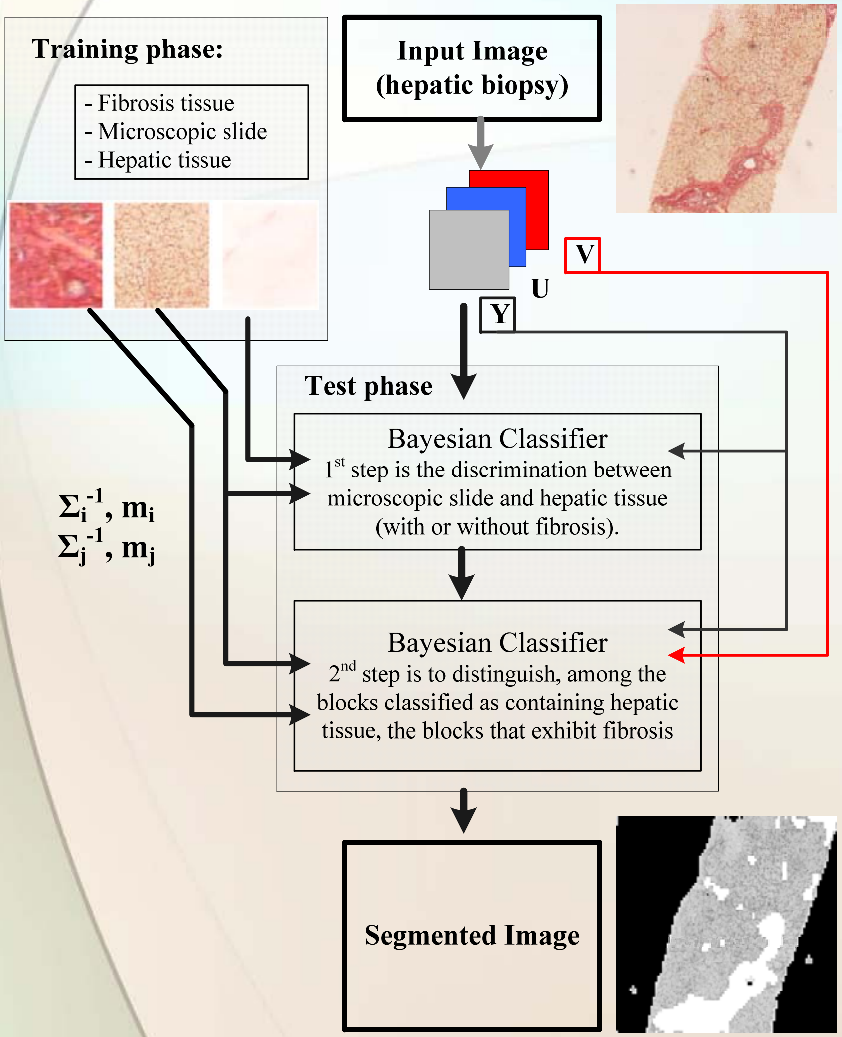
where: $$m_i[64\times 1], m_i = \frac{1}{N_i}\sum_{k=0}^{N_i-1} X_{k,i} \;, \quad \Sigma_i[64\times 64], \Sigma_i = \frac{1}{N_i}\sum_{k=0}^{N_i-1}(X_{k,i}-m_i)(X_{k,i}-m_i)^T \;,$$

- The Bayes decision rules for minimal cost are the following:

$$if \qquad p(C_i \mid X) < p(C_j \mid X) \Rightarrow X \in C_j; \quad else \quad X \in C_i$$

$$p(C_i \mid X) = \frac{p(X \mid C_i)\cdot P(C_i)}{p(X)}$$

Identification and quantification of the hepatic fibrosis from liver biopsies



Training phase:
- Fibrosis tissue
- Microscopic slide
- Hepatic tissue
Input Image (hepatic biopsy)
V
U
Y
Test phase
Σ_i^{-1}, m_i
Σ_j^{-1}, m_j
Bayesian Classifier
1st step is the discrimination between microscopic slide and hepatic tissue (with or without fibrosis).
Bayesian Classifier
2nd step is to distinguish, among the blocks classified as containing hepatic tissue, the blocks that exhibit fibrosis
Segmented Image

- Two classification tasks must be solved:

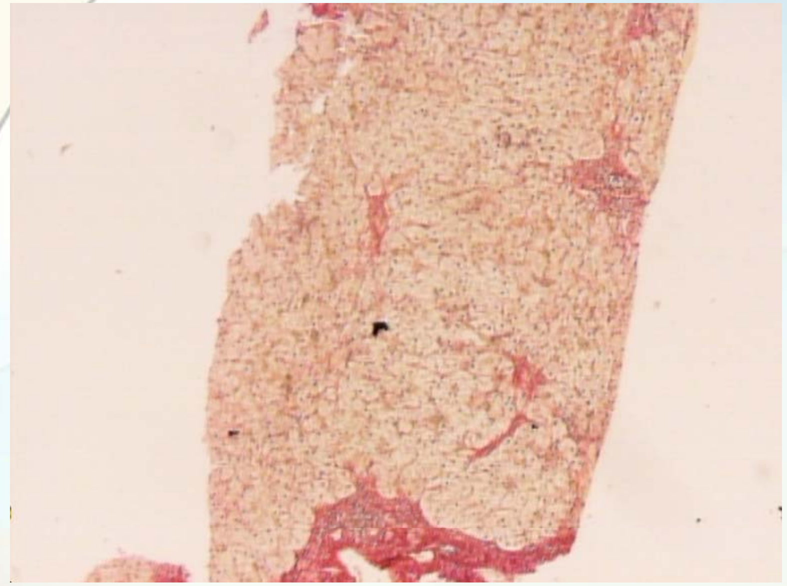
  - textured (hepatic tissue) vs. microscopic slide

  - in the textured hepatic tissue => fibrosis (reddish) vs. healthy (beige)

- We use the Bayes decision rule to classify a DCT block into microscopic slide, healthy tissue or fibrosis

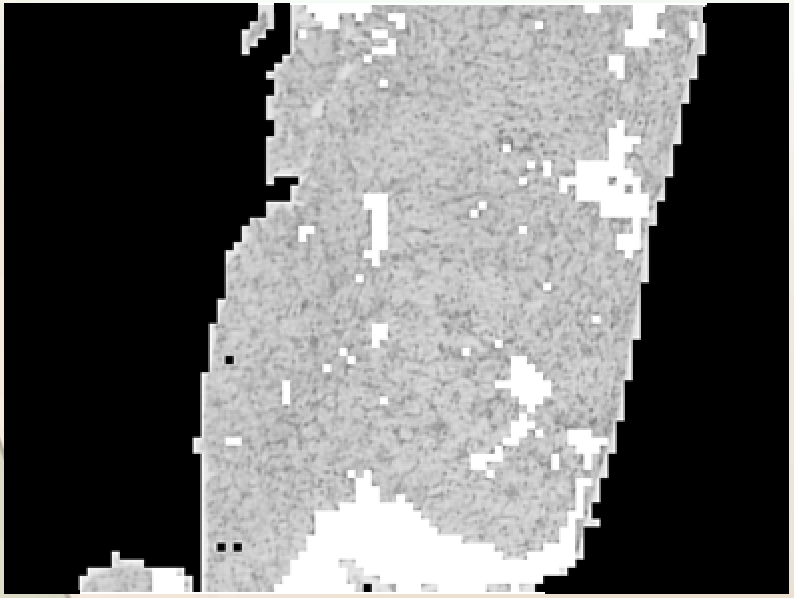
- Features space = zig-zag scanned quantized DCT coefficients.

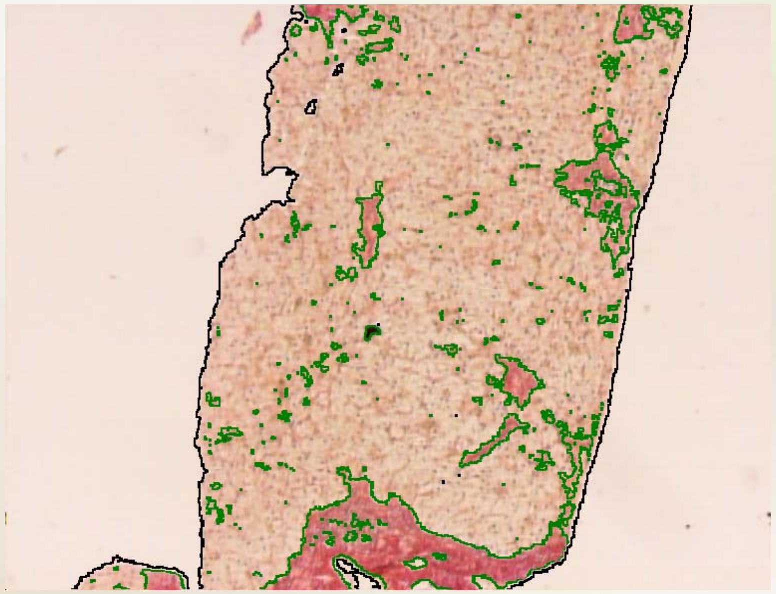
Experimental results



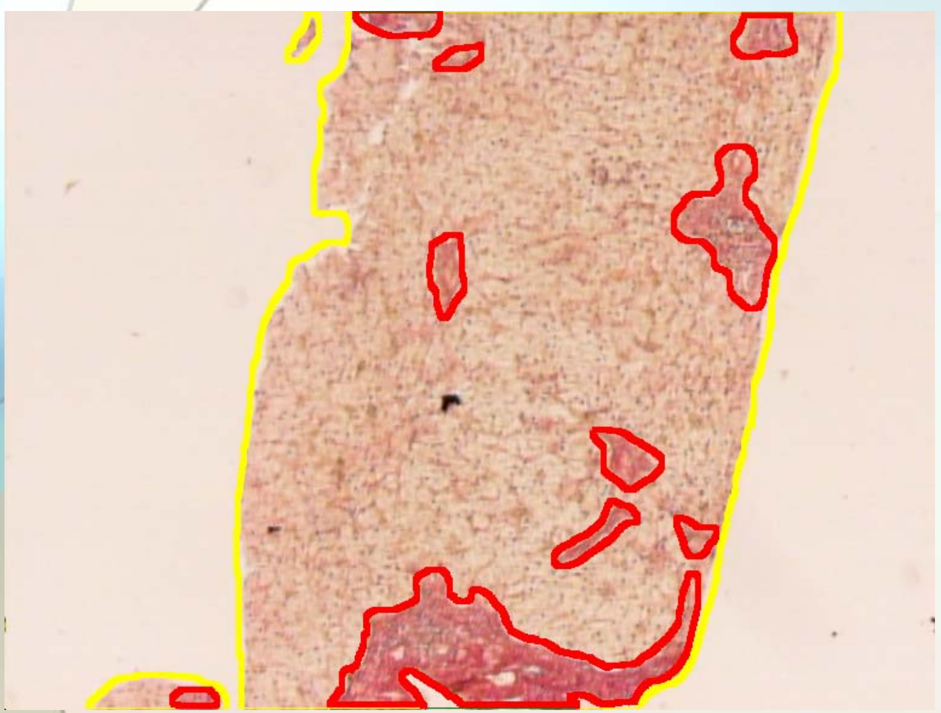
Classifications results using our algorithm and pixel level algorithm:

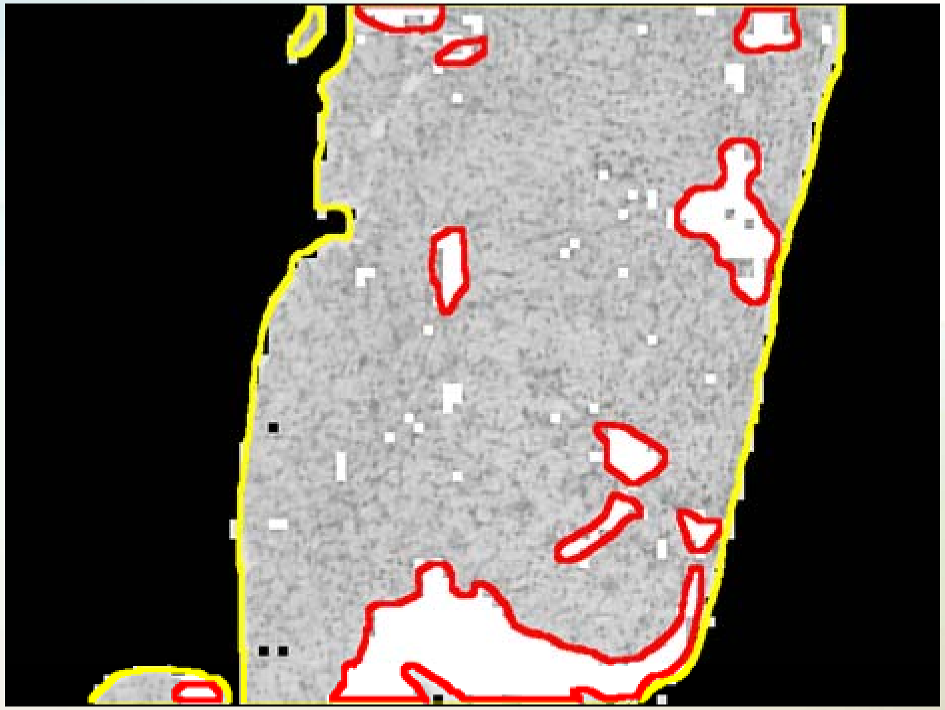
Patient	DCT algorithm [%]	Pixel level algorithm [%]	Scores of fibrosis
P1	4.15	4.47	1
P2	5.32	7.12	1
P3	7.05	6.23	1
P4	10.7	11.2	2
P5	14.95	14.6	2
P6	16.71	20.5	3
P7	17.84	21	3





Experimental results

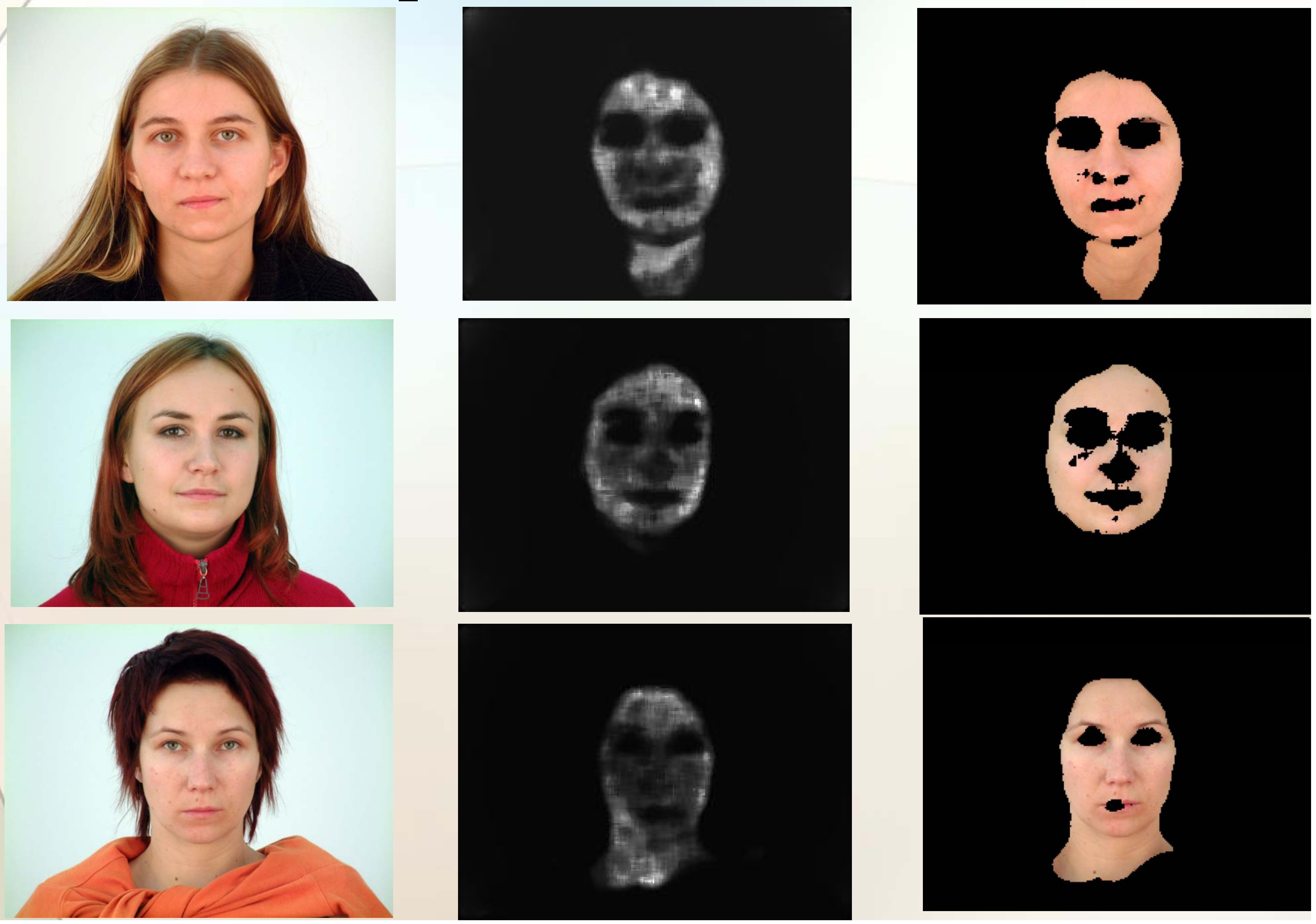




False acceptance rate (FAR) and false rejection rate (FRR), for one patient:

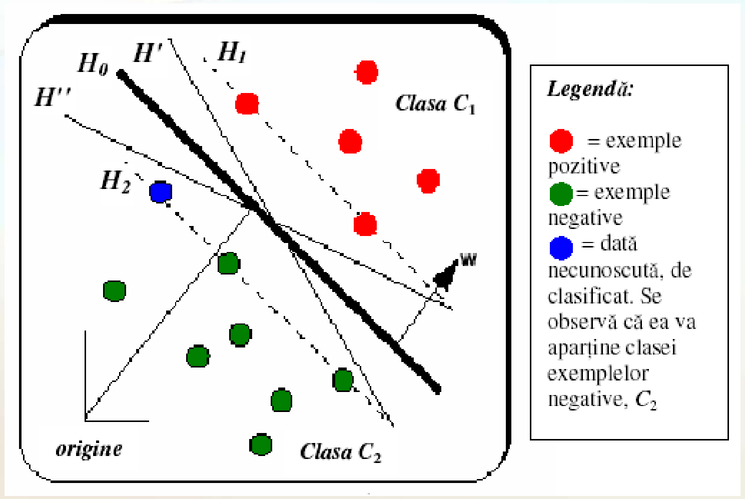
	1st classifier		2nd classifier	
	FAR	**FRR**	**FAR**	**FRR**
Average	0.75	0.99	2.14	1.81
Worse case FAR	1.41	0.98	3.72	1.95
Worse case FRR	0.82	1.93	1.74	3.59

Skin Segmentation of Facial Images in the Compressed Domain



Support vector machines

- Support vector machines (SVM) = are characterized by the ability to learn from a relatively small number of examples (they have a very good generalization capacity), and by the ability to separate highly correlated data by projecting them in a higher dimensional space in which the data becomes linearly separable.
- The main mathematical operation between the feature vectors in the classification phase of a support vector machine is the computation of the dot product of two vectors.



For a linear SVM, the decision function is :

$$f(\mathbf{x}) = \sum_{i=1}^{N_S} \alpha_i y_i \mathbf{x}^T \mathbf{x}_i + b,$$

unde:

x_i = support vectors of SVM

α_i = non-zero Lagrange multipliers

y_i = labels of support vectors xi, yi = +1, -1

b - polarization term of the optimal separating hyperplane

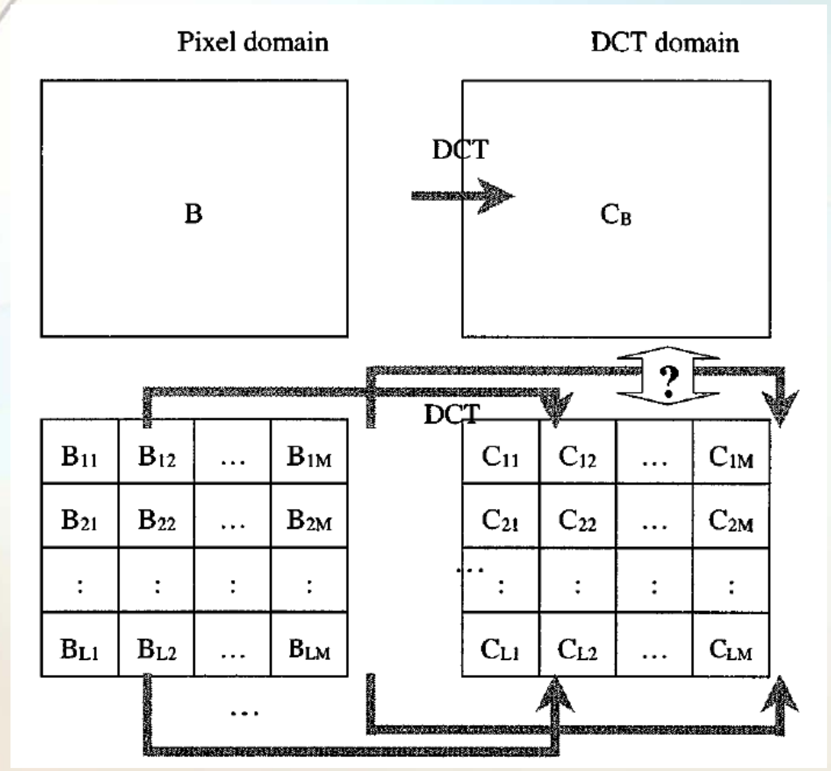
The class assignment is determined by the sign of the decision function of the SVM classifier:

$$y = sign(f(x))$$

$\mathbf{x}^T \mathbf{x}_i$ It was proven mathematically that the dot products is the same in both the original feature space and in the discrete cosine transform induced feature space.

- => the implementation of support vector machines in the JPEG compressed domain is feasible and it can be expected to lead to similar performances like the application of support vector machines in the original data space (the differences in the classification performances are due to the quantization, but in some cases this will be an advantage for the classification).

The relations between a DCT coefficient block and its sub-blocks



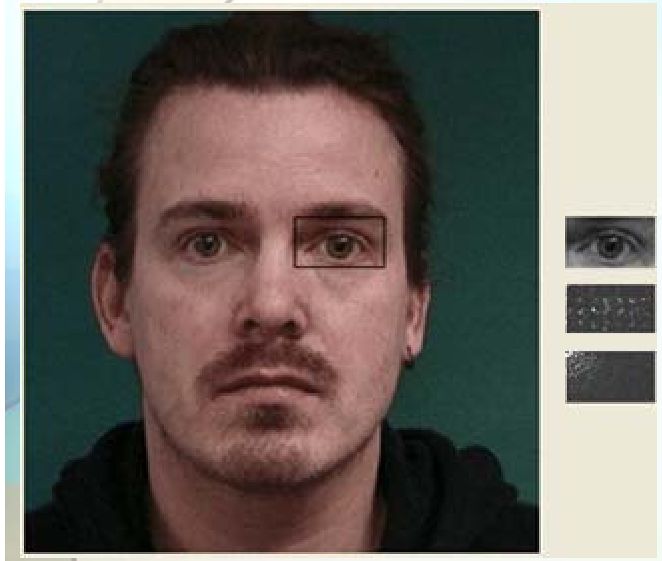
$$C_B = \frac{1}{M} A^{*} \begin{pmatrix} C_{0,0} & \cdots & C_{0,M-1} \\ \vdots & \ddots & \vdots \\ C_{L-1,0} & \cdots & C_{L-1,M-1} \end{pmatrix} A^{*T}$$

$$\begin{pmatrix} C_{0,0} & \cdots & C_{0,M-1} \\ \vdots & \ddots & \vdots \\ C_{L-1,0} & \cdots & C_{L-1,M-1} \end{pmatrix} = M A^{*-1} C_B A^{*T-1}$$

• In the compressed domain we have access only at 8×8 pixels block – this is a restricted dimension for analyses regions of interest

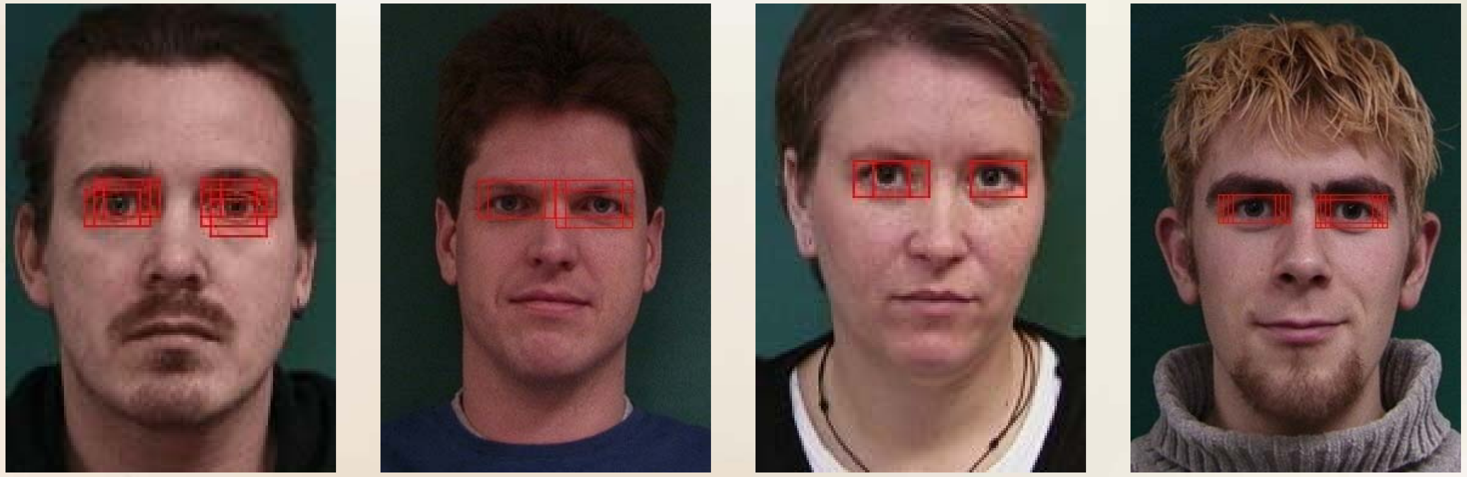
• To obtain the quantized DCT coefficients feature vectors for any given size window (corresponding to the object that is to be recognized), starting from the 8×8 DCT coefficient blocks used in JPEG encoding, it is necessary to use the relations between a DCT coefficient block and its sub-blocks.

An application for object recognition and localization in the compressed domain



A support vector machine classifier with binary decision function for object localization in a JPEG image, considered decomposed in a set of partially overlapping rectangular windows of any dimension (obtained by scanning the image with a rectangular window of an a-priori chosen size, with a randomly chosen scanning step - measured in pixels).

The performances achieved, compared to the application based on pixel data, support the theoretical reasoning. In the case of using quantized DCT coefficients, the performances are slightly superior to the ones based on the spatial representation of the image.



Conclusions

- The main advantage of processing in the compressed domain is given by the possibility of using the available information straight from the JPEG compressed images (the DCT coefficients), without the need for decompressing the digital image to pixel level in order to apply the processing.

- Another advantage is that, by using a feature space with a very good energy compacting property, a large number of values in the feature vectors will be zero, thus significantly reducing the number of multiplications and summations involved in the computation.

Thank you for your attention !

Assistant Eng. Camelia FLOREA, Ph.D.
Technical University of Cluj-Napoca
Str. C. Daicoviciu 15, 400020, Cluj-Napoca, Romania
E-mail: Camelia.Florea@com.utcluj.ro
Phone: 00 40 264 401309

Cluj Napoca - Views