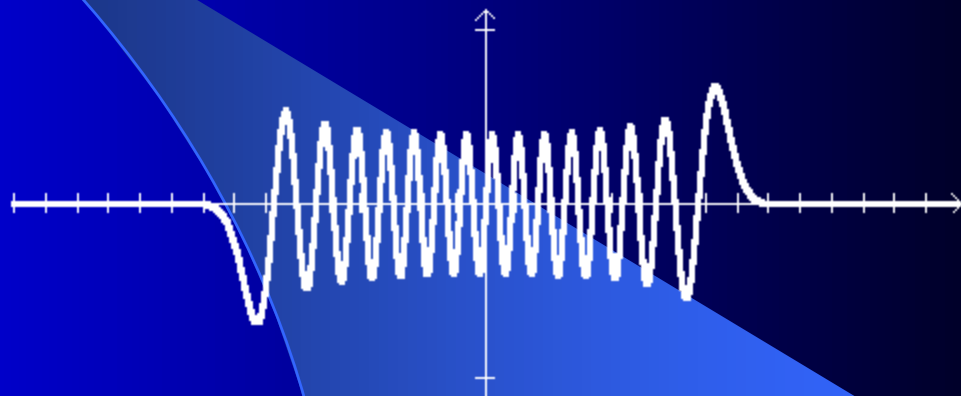
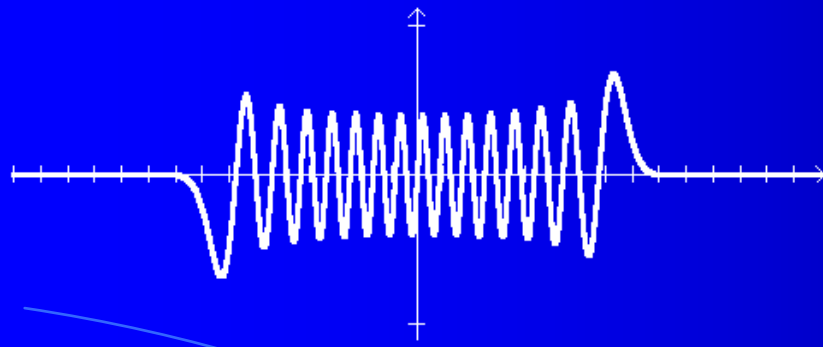


Лекция 7

Проекционный метод обращения преобразования Фурье с использованием функций Эрмита



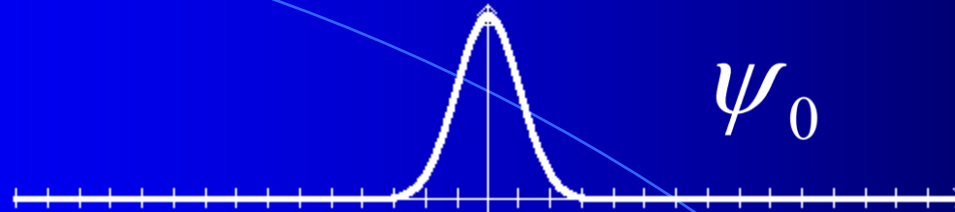


Проекционный метод обращения преобразования Фурье с использованием функций Эрмита

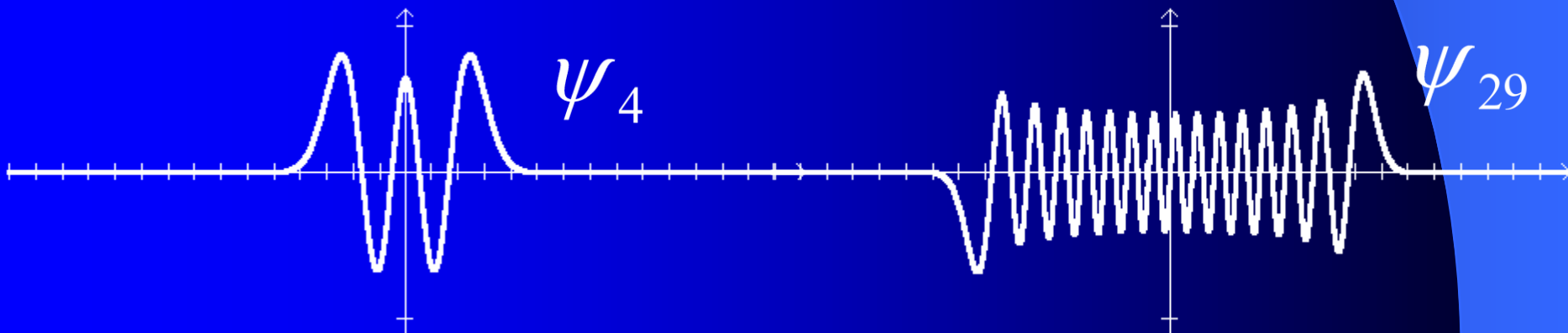
Outline:

- Projection Method (Hermite series approach)
- Applications
 1. Image filtering and deblocking by projection filtering
 2. Image matching
 3. Texture matching
 4. Low-level methods for audio
 5. Hermite foveation

Hermite transform



The proposed method is based on the features of Hermite functions. An expansion of signal information into a series of these functions enables one to perform information analysis of the signal and its Fourier transform at the same time.



A) $\hat{\psi}_n = i^n \psi_n$

B) They derivate a full orthonormal in $L_2(-\infty, \infty)$ system of functions.

The Hermite functions are defined as:

$$\psi_n(x) = \frac{(-1)^n e^{x^2/2}}{\sqrt{2^n n!} \sqrt{\pi}} \cdot \frac{d^n (e^{-x^2})}{dx^n}$$

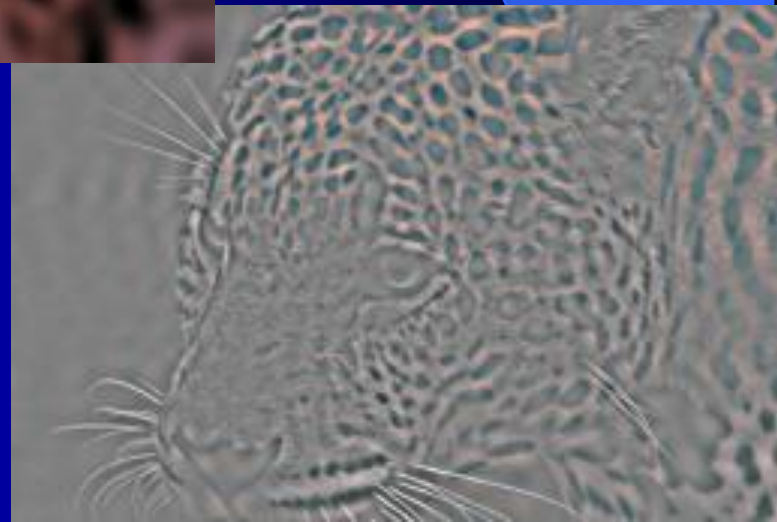


Original image

2D decoded image by
45 Hermite functions at
the first pass and 30
Hermite functions at
the second pass



Difference image
(+50% intensity)



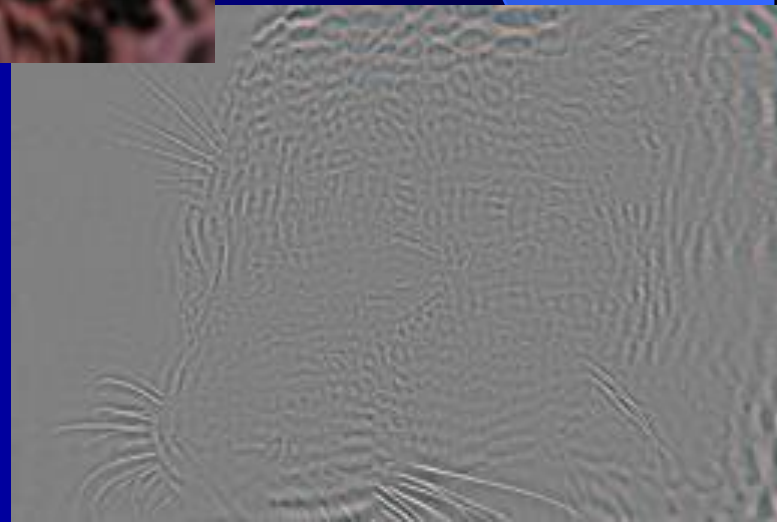


Original image

2D decoded image by
90 Hermite functions at
the first pass and 60
Hermite functions at
the second pass



Difference image
(+50% intensity)



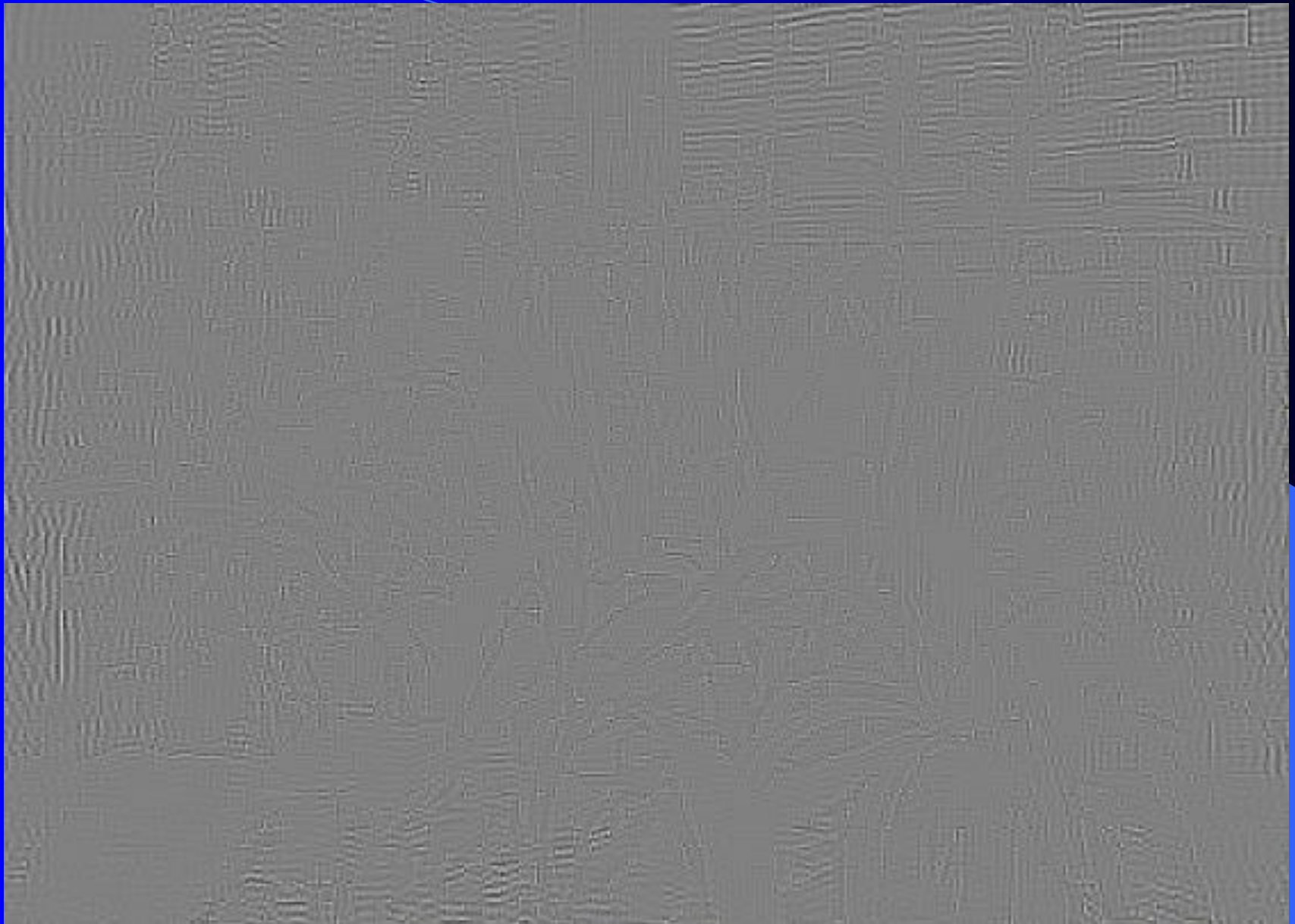
*Image filtering and deblocking by
projection filtering*



Original lossy JPEG image



Enhanced image



Difference image (Subtracted high frequency information)

Zoomed in:



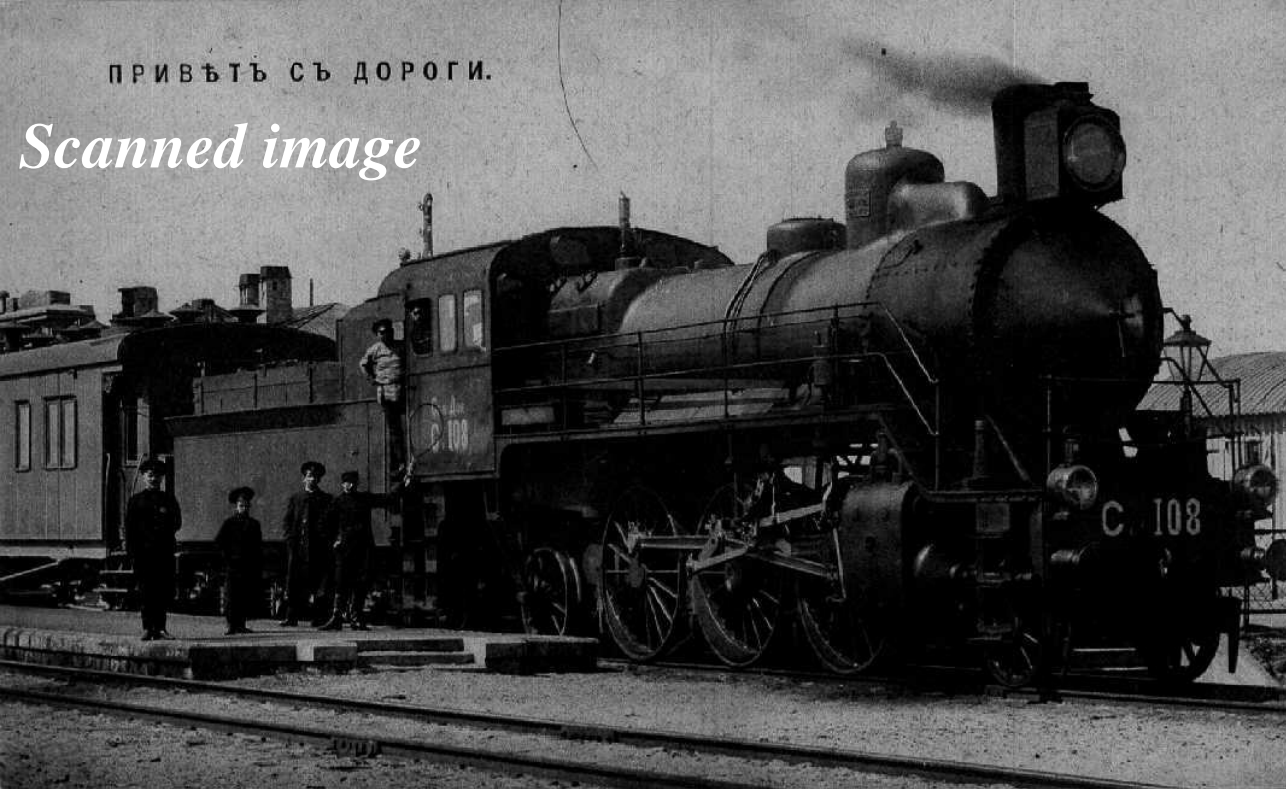
Original image



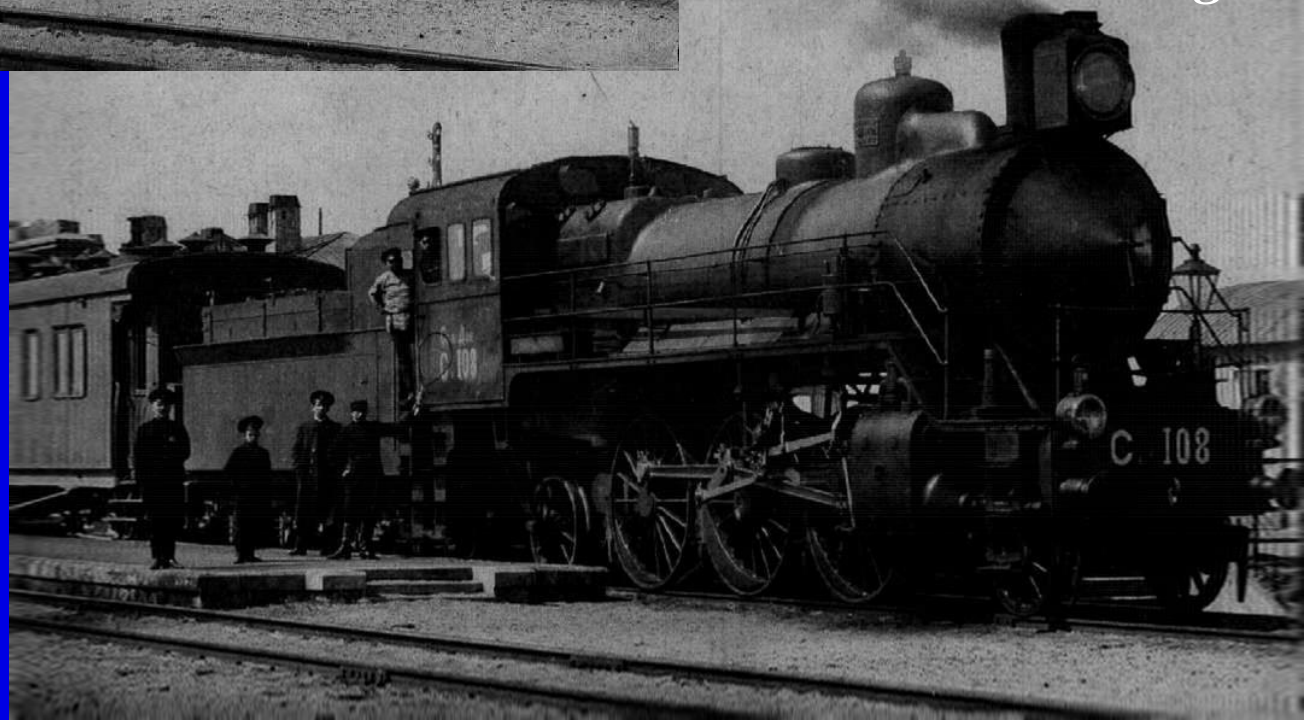
Enhanced image

ПРИВЪТЪ СЪ ДОРОГИ.

Scanned image



Enhanced image



Zoomed in:

Enhanced image

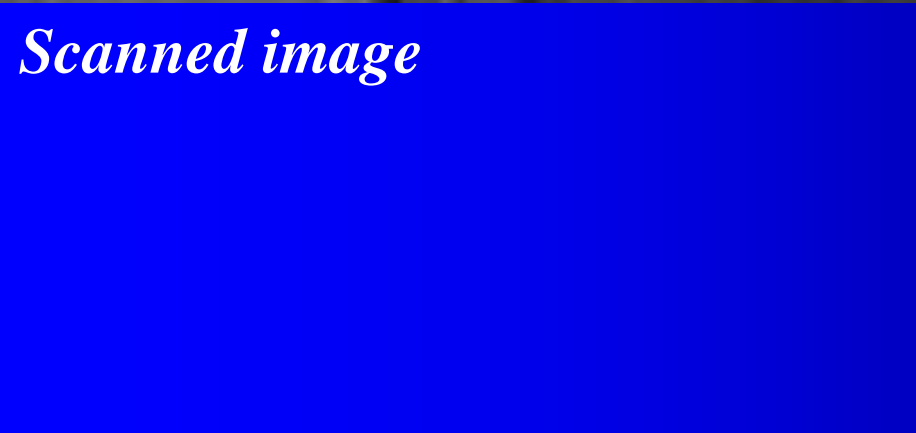


Image matching

Information parameterization for image database retrieval

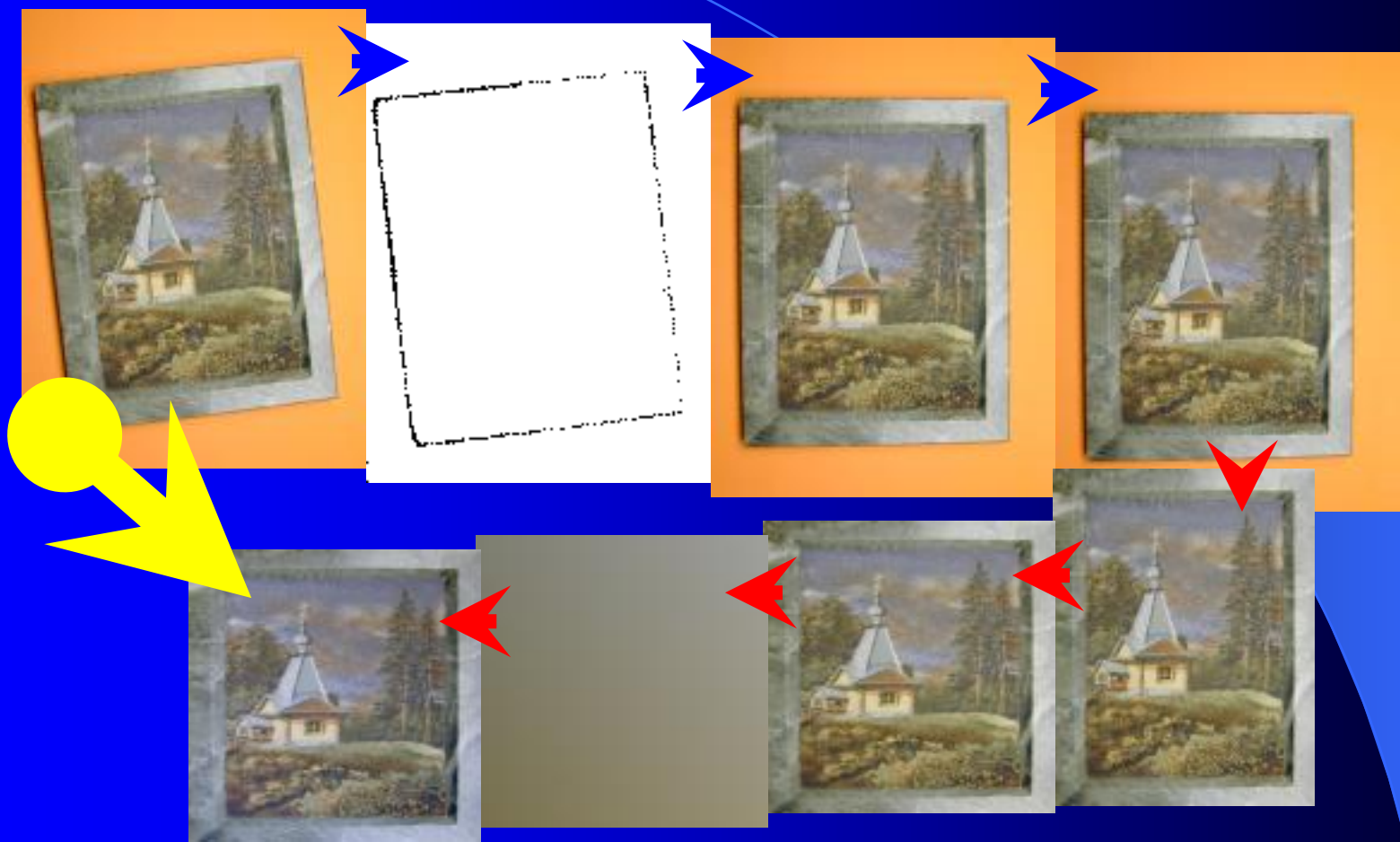


Image normalizing

Graphical information parameterization

Parameterized image retrieval

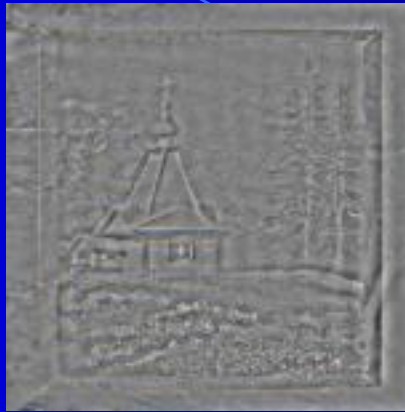
Algorithm

Information parameterization for image database retrieval



*Normalized
image*

=

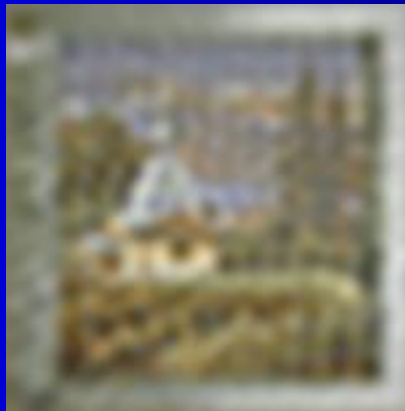


HF component

+



LF component



*Recovered image with the
recovered image plane*

Image normalizing

Graphical information parameterization

Parameterized image retrieval

Algorithm

Information parameterization for image database retrieval

Database size	– 768 images (4.12Gb)
Initial images format	– 1600x1200x24bit (5.5Mb)
Normalized images format	– 512x512x24bit (0.75Mb)
Number of parameterization coefficients	– 32x32x3
Error rate	– <0.14%
Search time	– 4 sec. (for K7-750)

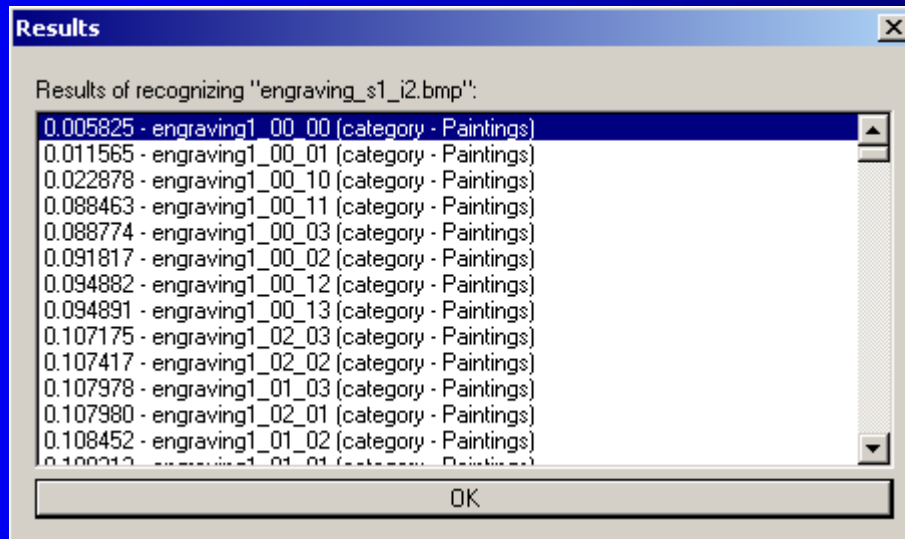


Image normalizing

→ Graphical information parameterization

→ Parameterized image retrieval

Algorithm



Graphics: Polyline Coef All image

Pass: X X&Y

String #: Professional settings

Number of functions (X): Auto

Number of functions (Y):



Image matching results

Graphics:

Polyline Coef

All image

Pass:

X

X&Y

String #:

Number of functions (X):

Number of functions (Y):

Auto



Compare Dscn0012.bmp with:

Dscn0009.bmp : 0.044278

Dscn0010.bmp : 0.044161

Dscn0011.bmp : 0.038277



Image matching results



Graphics: Polyline Coef All image

Pass: X X&Y

String #: Professional settings

Number of functions (X):

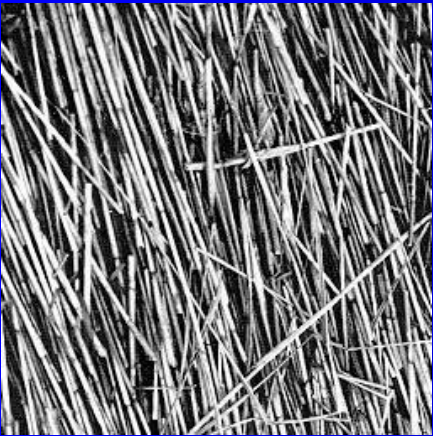
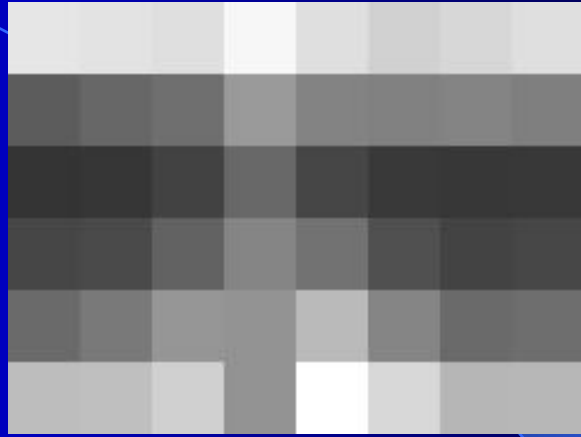
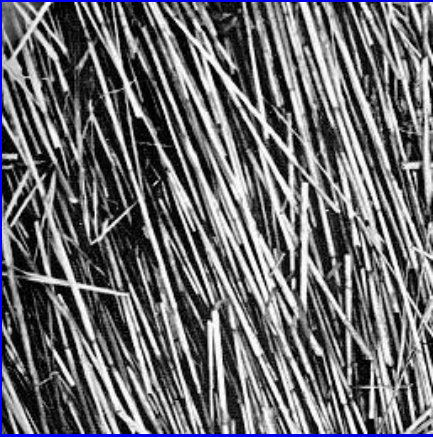
Number of functions (Y): Auto

Compare Dscn0040.bmp with:

- Dscn0001.bmp : 0.259785
- Dscn0002.bmp : 0.265052
- Dscn0005.bmp : 0.173370
- Dscn0006.bmp : 0.172548
- Dscn0007.bmp : 0.171292
- Dscn0008.bmp : 0.167293
- Dscn0009.bmp : 0.181936
- Dscn0010.bmp : 0.181217
- Dscn0015.bmp : 0.177050
- Dscn0016.bmp : 0.174577
- Dscn0039.bmp : 0.007523
- Dscn0041.bmp : 0.028101
- Dscn0042.bmp : 0.174565
- Dscn0043.bmp : 0.175095
- Dscn0044.bmp : 0.179361



Texture matching



A method of obtaining the texture feature vectors

Input function

$$f(x) = \sum_{i=0}^{\infty} \alpha_i \cdot \Psi_i(x)$$

Fourier coefficients

$$\alpha_i = \int_{-\infty}^{\infty} \Psi_i(x) \cdot f(x) dx$$

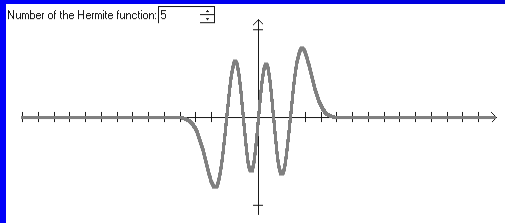
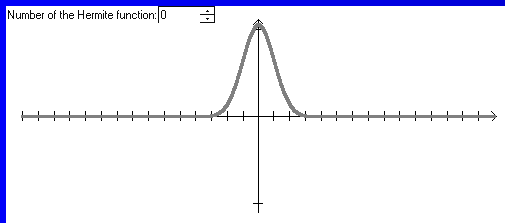
1-D to 2-D expansion

$$\psi_{n_1 n_2}(x, y) = \psi_{n_1}(x) \cdot \psi_{n_2}(y),$$

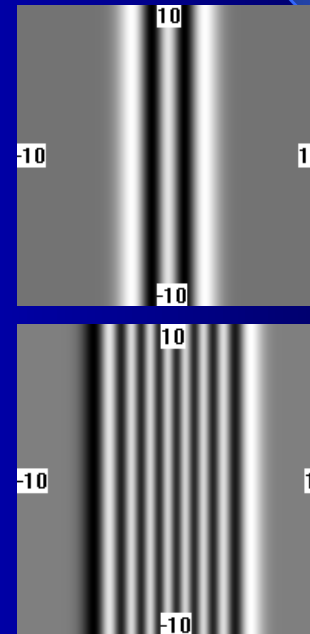
$$\psi_n(x, y) = \psi_n(x) \cdot 1$$

A method of obtaining the texture feature vectors

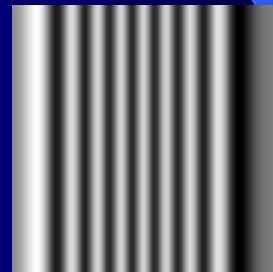
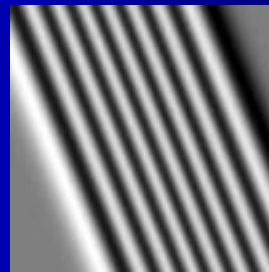
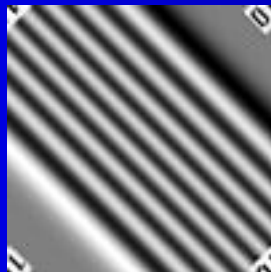
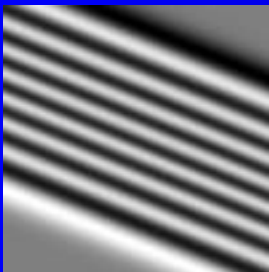
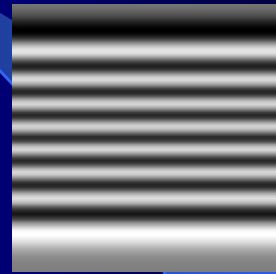
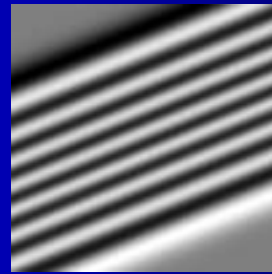
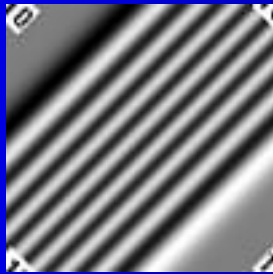
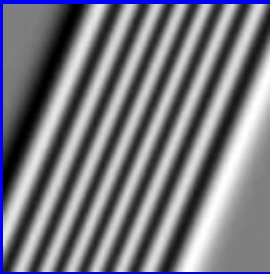
1-D Hermite functions:



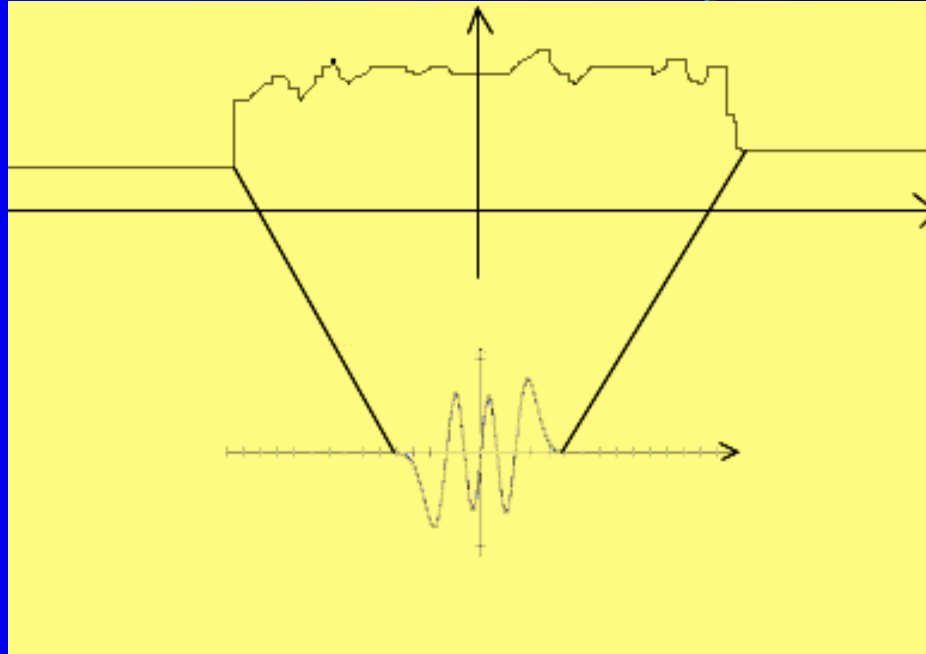
1-D to 2-D expanded Hermite functions:



Orientations



Localization problem



Decomposition process is optimal, if localization segments of the input function and filtering functions are equal.

Feature vectors

Standard coding

In this approach to get the feature vectors we consider the functions $\psi_n(x,y)$ where $n=0..64$, and 6 energy coefficients are calculated as:

$$E_1 = (\alpha_0)^2 + (\alpha_1)^2,$$

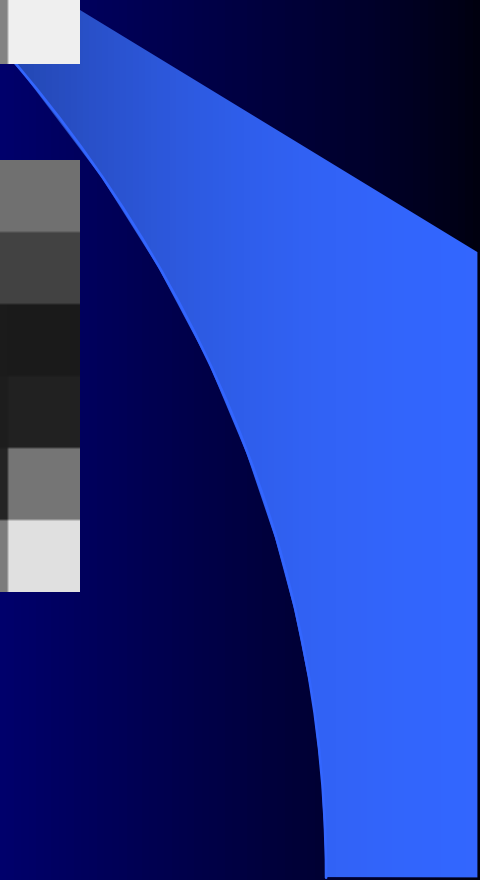
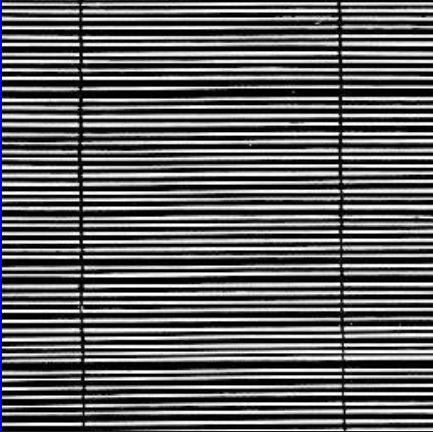
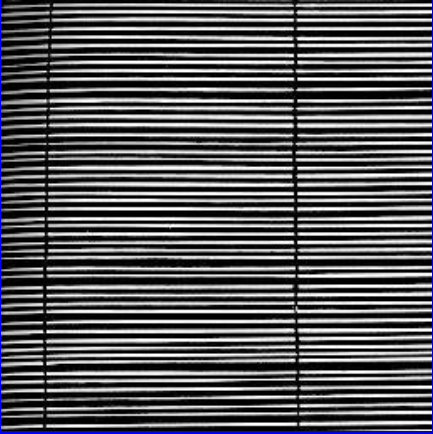
$$E_2 = (\alpha_2)^2 + (\alpha_3)^2 + (\alpha_4)^2,$$

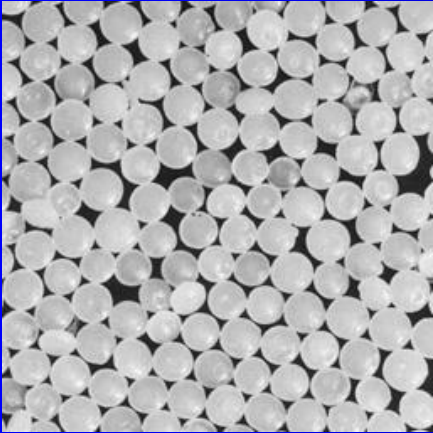
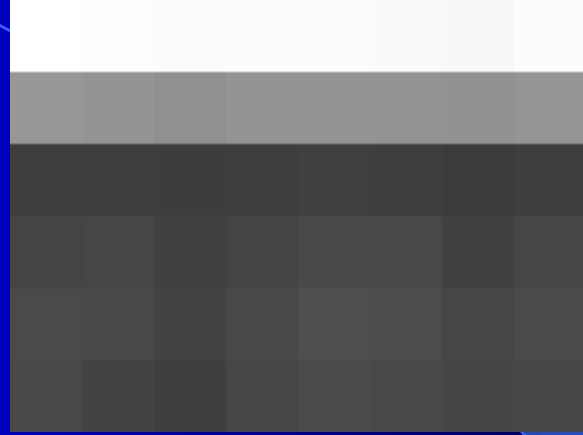
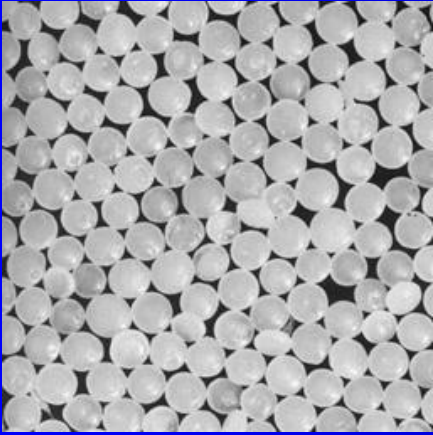
$$E_3 = (\alpha_5)^2 + (\alpha_6)^2 + (\alpha_7)^2 + (\alpha_8)^2,$$

...

$$E_6 = (\alpha_{33})^2 + (\alpha_{34})^2 + \dots + (\alpha_{63})^2 + (\alpha_{64})^2,$$

$f(x,y)$ is the source image.





Feature vectors

Hierarchical coding

$$E_1 = (\alpha_0^{(1)})^2 + (\alpha_1^{(1)})^2,$$

$$E_2 = (\alpha_0^{(2)})^2 + (\alpha_1^{(2)})^2 + (\alpha_2^{(2)})^2 + (\alpha_3^{(2)})^2,$$

...

$$E_6 = (\alpha_0^{(6)})^2 + (\alpha_1^{(6)})^2 + \dots + (\alpha_{62}^{(6)})^2 + (\alpha_{63}^{(6)})^2,$$

$$\alpha_i^{(1)} = \frac{1}{\sqrt{A_j}} \int_{-\infty}^{\infty} dy \int_{-\infty}^{\infty} \psi_i(x, y) \cdot f(x, y) dx$$

$$\alpha_i^{(j)} = \frac{1}{\sqrt{A_j}} \int_{-\infty}^{\infty} dy \int_{-\infty}^{\infty} \psi_i(x, y) \cdot (f(x, y) - \sum_{l=2}^j f^{(l-1)}(x, y)) dx, j > 1$$

$f(x, y)$ is the source image.

Feature vectors

Hierarchical coding without subtractions

$$\begin{aligned} E_1 &= (\alpha_0^{(1)})^2 + (\alpha_1^{(1)})^2, \\ E_2 &= (\alpha_0^{(2)})^2 + (\alpha_1^{(2)})^2 + (\alpha_2^{(2)})^2 + (\alpha_3^{(2)})^2, \end{aligned}$$

...

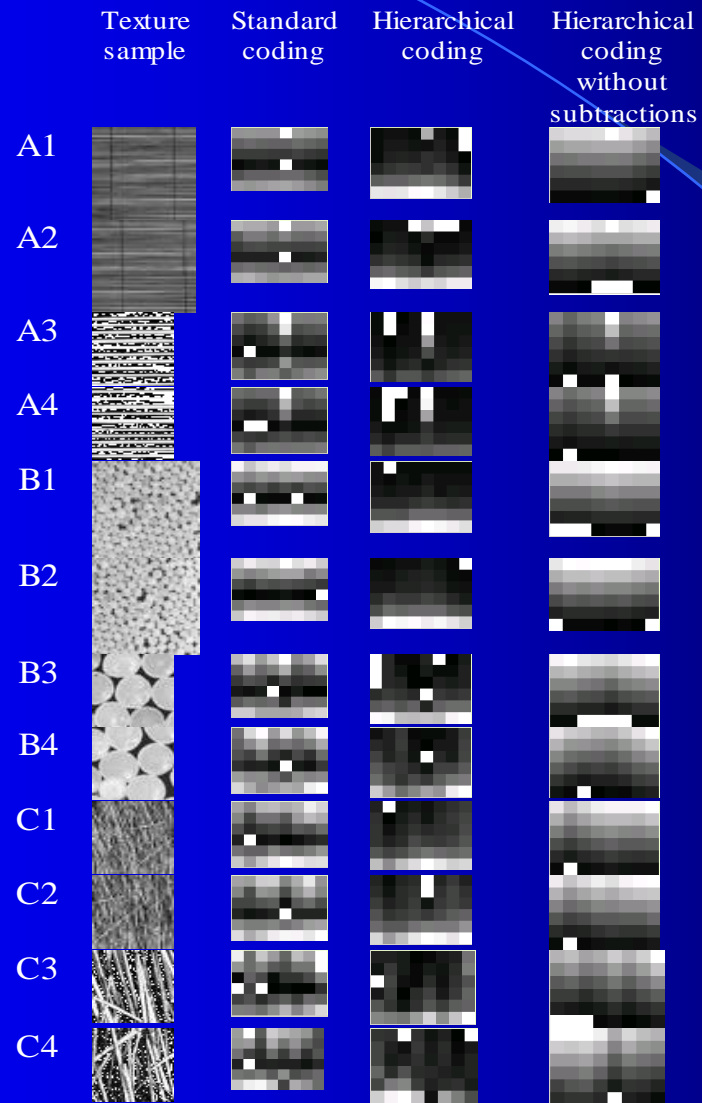
$$E_6 = (\alpha_0^{(6)})^2 + (\alpha_1^{(6)})^2 + \dots + (\alpha_{62}^{(6)})^2 + (\alpha_{63}^{(6)})^2,$$

$$\alpha_i^{(1)} = \frac{1}{\sqrt{A_j}} \int_{-\infty}^{\infty} dy \int_{-\infty}^{\infty} \psi_i(x, y) \cdot f(x, y) dx$$

$$\alpha_i^{(j)} = \frac{1}{\sqrt{A_j}} \int_{-\infty}^{\infty} dy \int_{-\infty}^{\infty} \psi_i(x, y) \cdot f(x, y) dx, j > 1$$

$f(x, y)$ is the source image.

Feature vectors



Feature vectors

<i>a</i>	<i>b</i>	Standard coding	Hierarchical coding	Hierarchical coding without subtractions
a1	a2	0.004505	0.005349	0.003739
b1	b2	0.009905	0.008160	0.007742
c1	c2	0.017778	0.011848	0.009946
a3	a4	0.008807	0.006047	0.002422
b3	b4	0.020075	0.010667	0.006275
c3	c4	0.128322	0.101591	0.080176
a1	b1	0.488420	0.492238	0.491653
b1	c1	0.315644	0.304597	0.305442

Image segmentation task: Brodatz textures

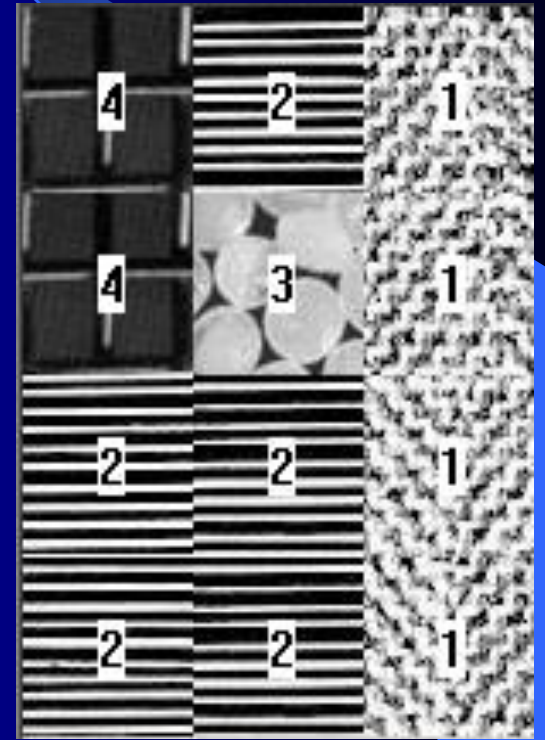
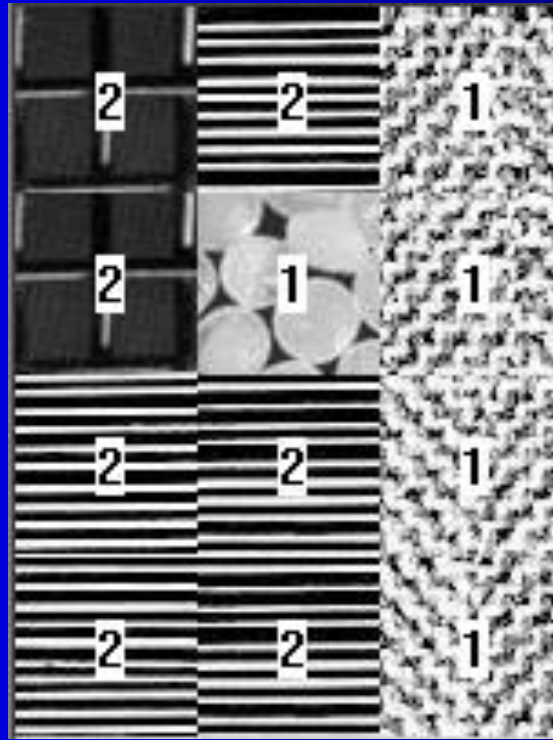
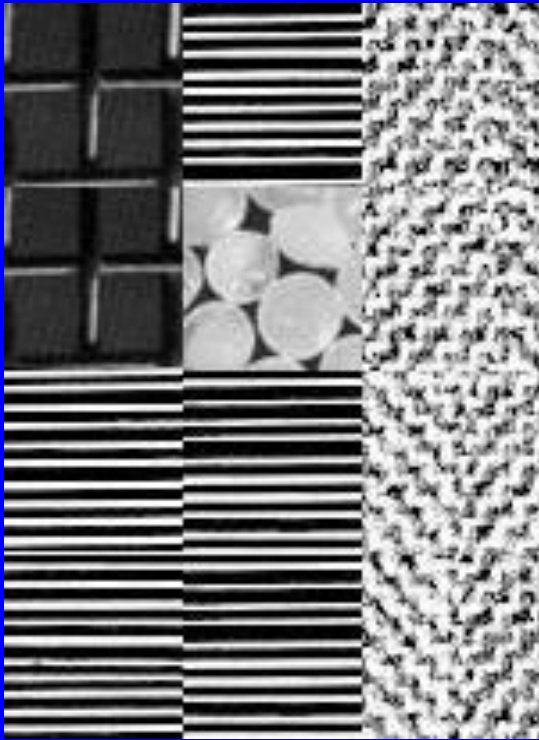
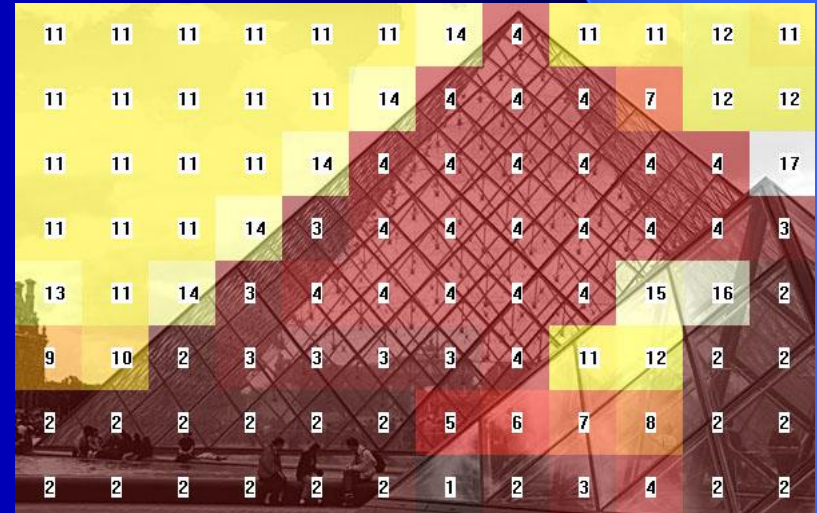
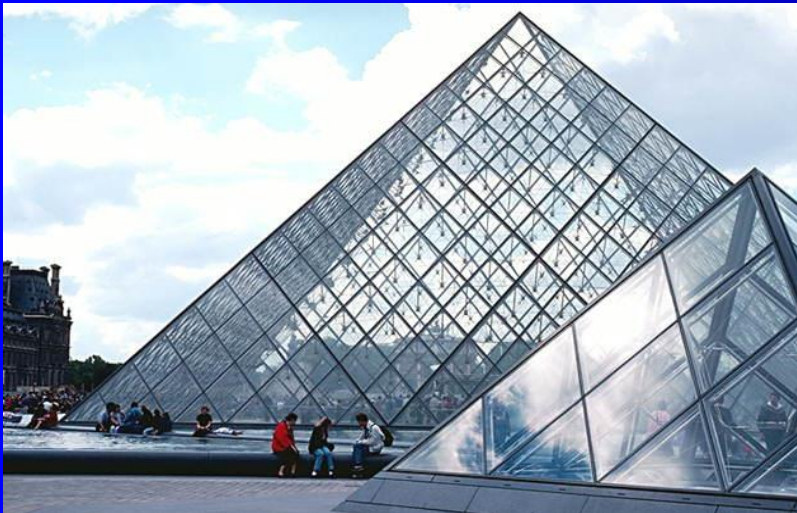
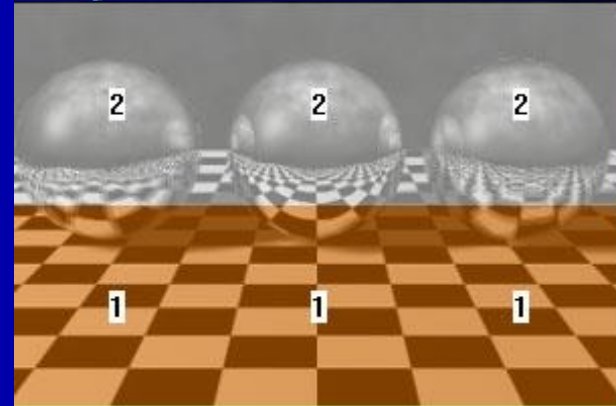
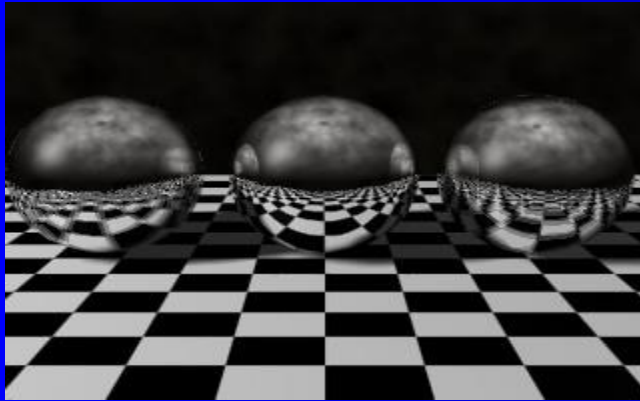


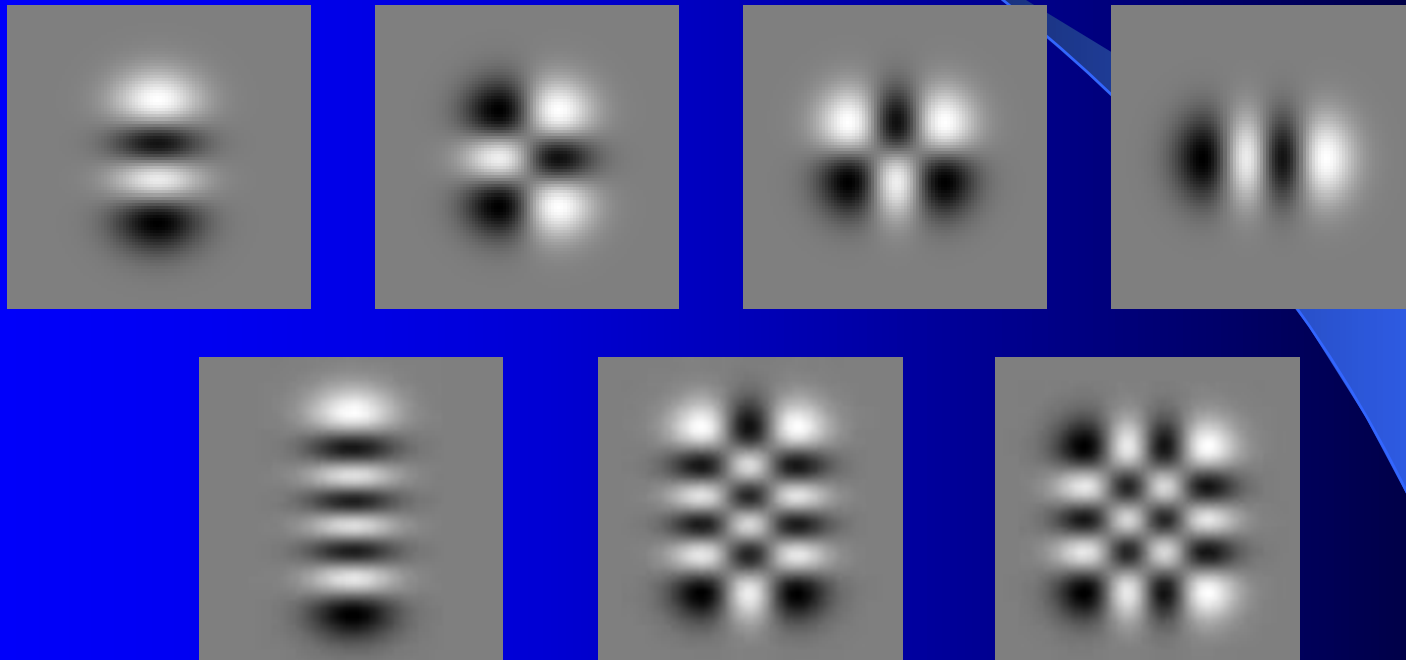
Image segmentation task



Image segmentation task

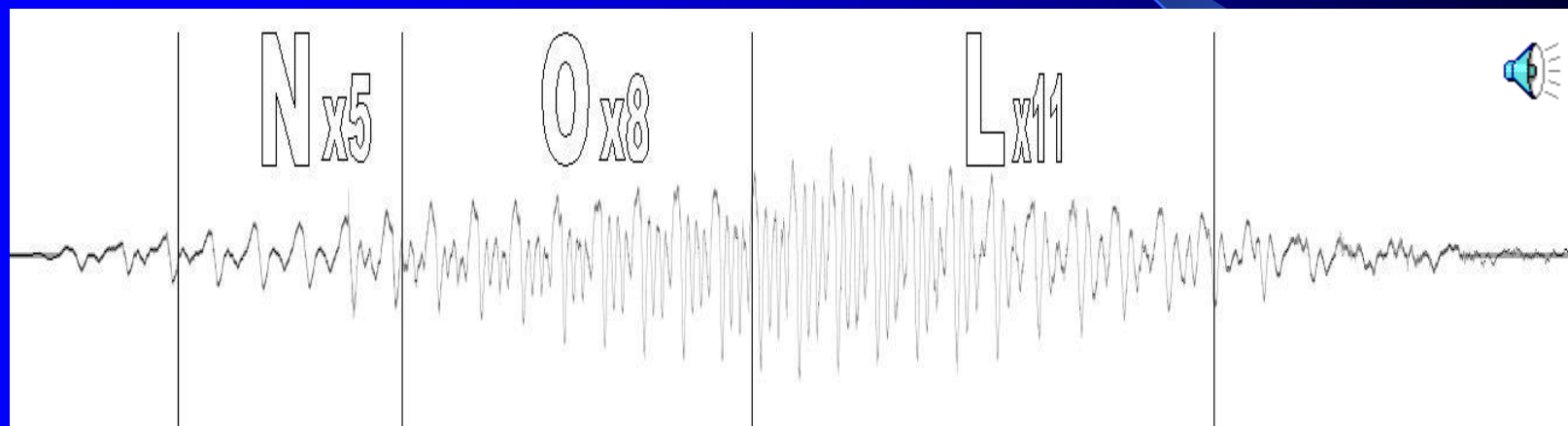


Texture parameterization using 2-D Hermite functions

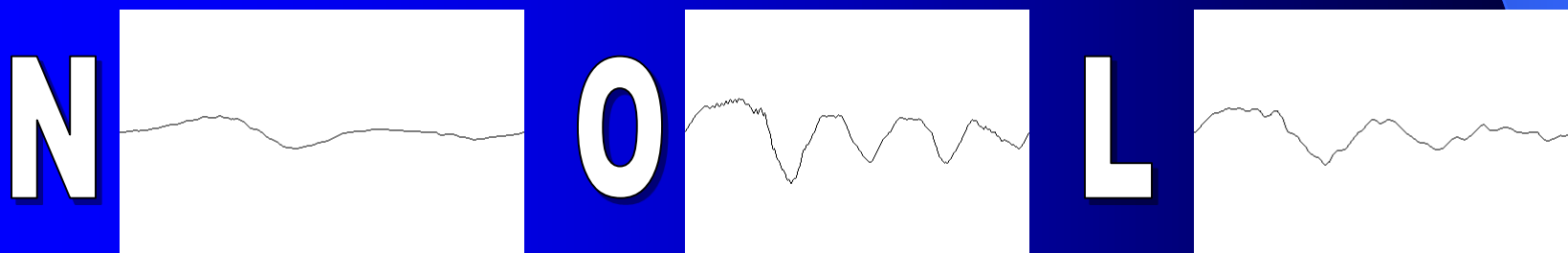


$$\psi_{0,3}(x, y), \psi_{1,2}(x, y), \psi_{2,1}(x, y), \psi_{3,0}(x, y), \psi_{0,7}(x, y), \psi_{2,5}(x, y), \psi_{3,4}(x, y)$$

Low-level methods for audio signal processing



Quasiperiod's waveforms



Areas of Hermite transform application:

- Signal filtering
- Speaker indexing
- Speaker recognition using database
- Source separation

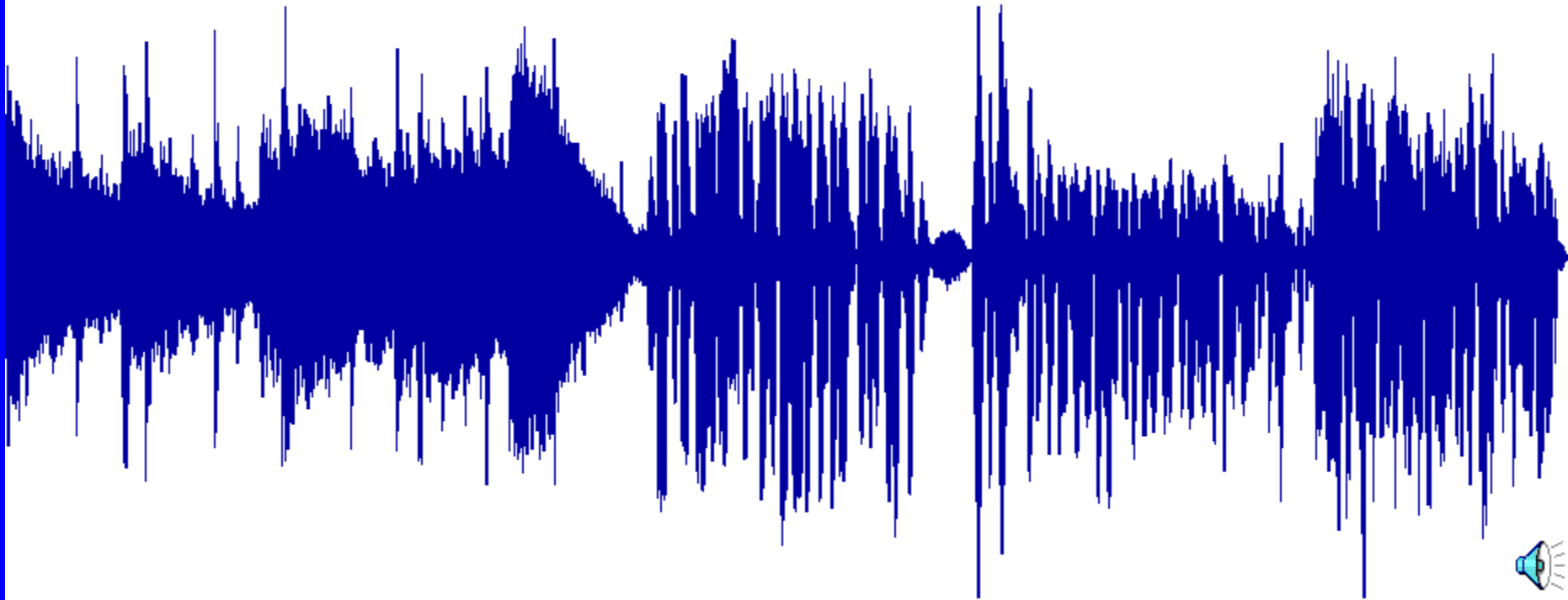
Audio sample

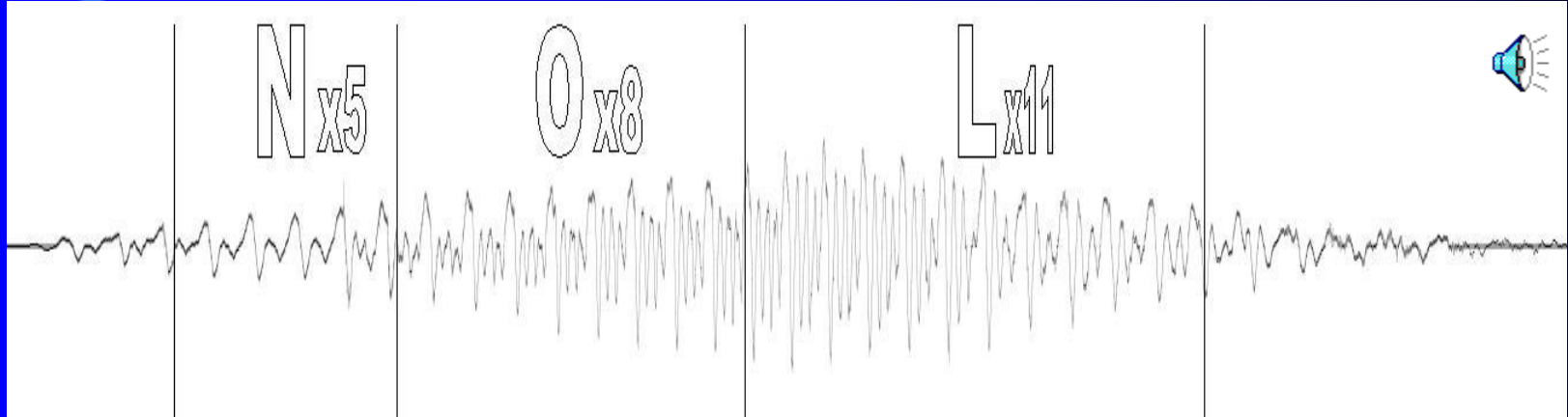
Music

Speaker 1

Speaker 2

2 speakers

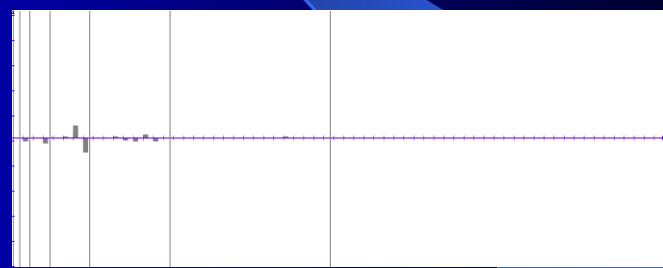
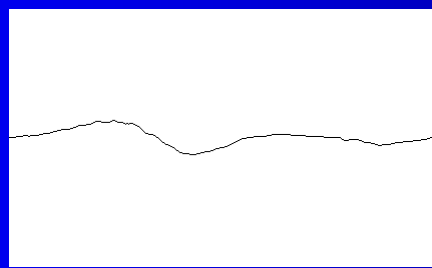




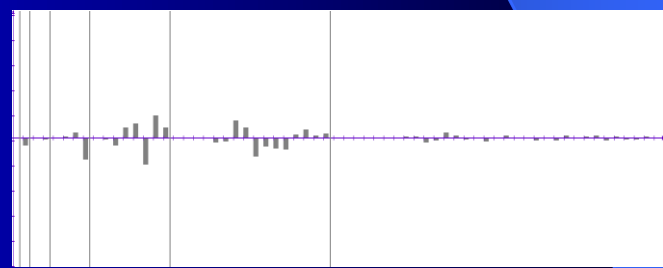
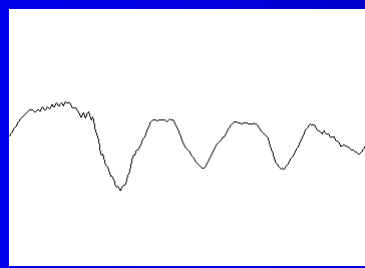
Quasiperiod waveform

Hermite histogram

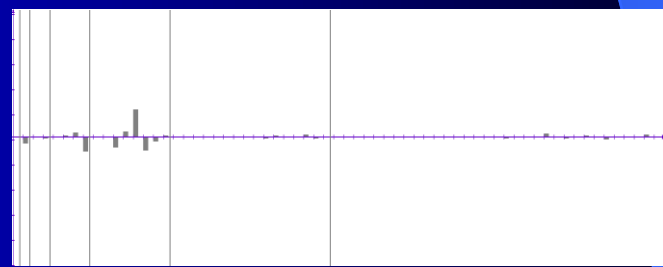
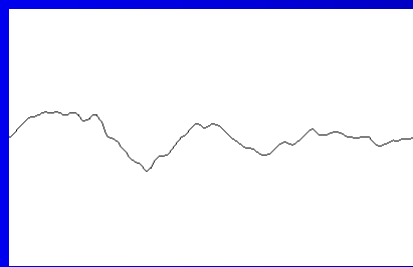
N



O



L



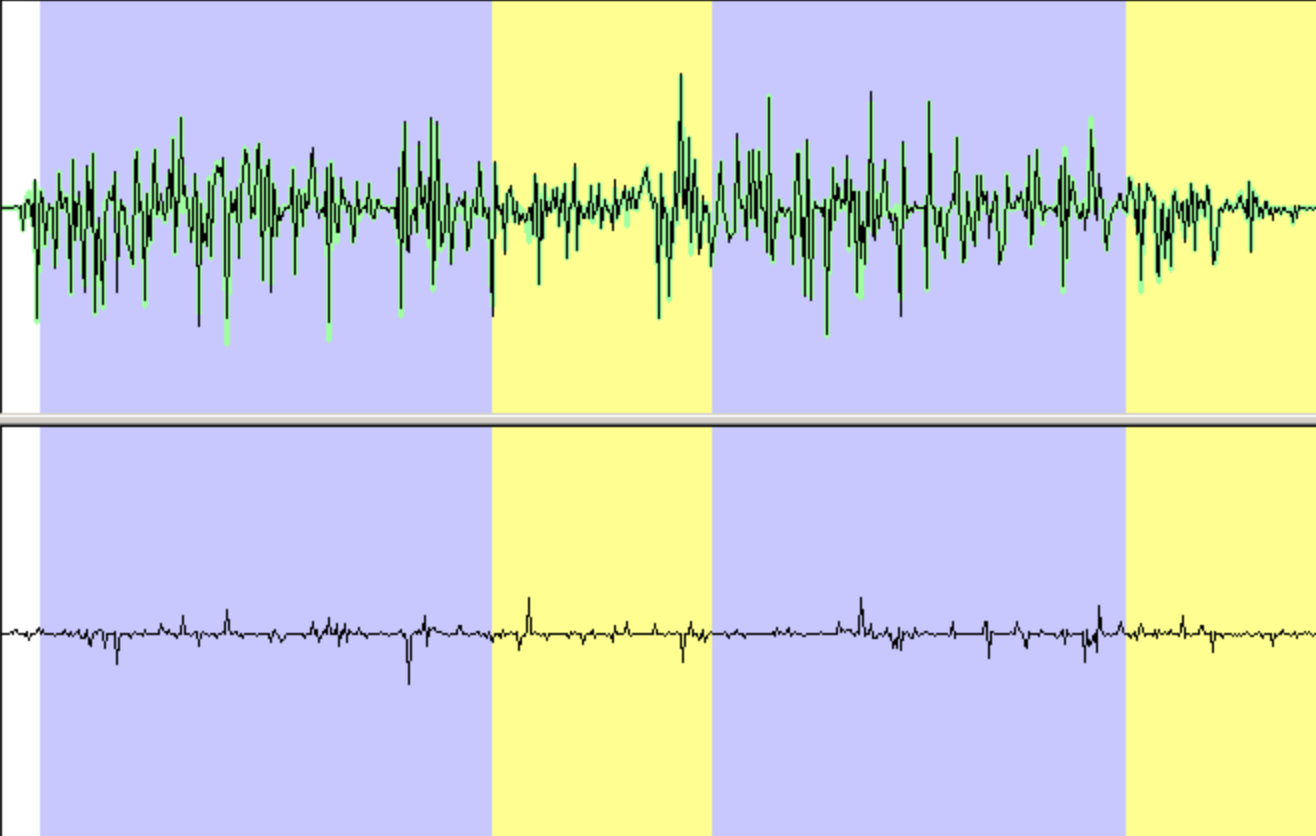
Speaker indexing

Hermite coder pro - [Dialog1]

File View Wave Settings Window Help

Play original sound Stop Number of blocks: 2287 Professional settings Draw
Play decoded sound Number of functions: 63 Auto detect blocks

Play difference sound



The main window displays two waveforms. The top waveform is a green audio signal with a light blue background. The bottom waveform is a black audio signal with a light blue background. Both waveforms are overlaid with vertical yellow bars, indicating speaker indexing blocks. The bars are positioned at approximately 1/3, 1/2, and 2/3 of the waveform's duration.

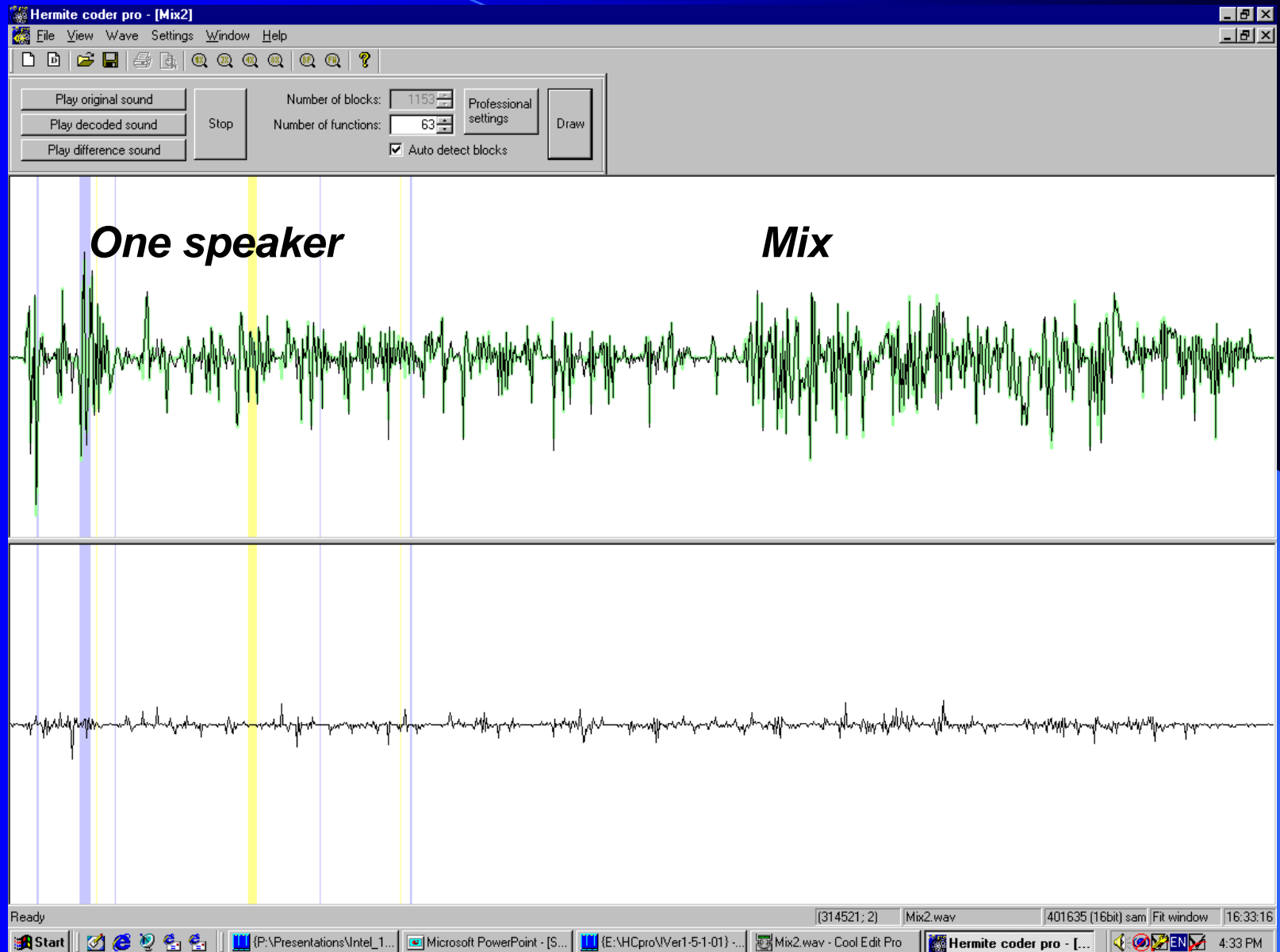
Database bar

Names:
Norkin
Osokin
New Edit Delete

Characters:
*
New Edit Delete
View
Hide characters
Show detected characters for selected name
Show detected characters and speakers

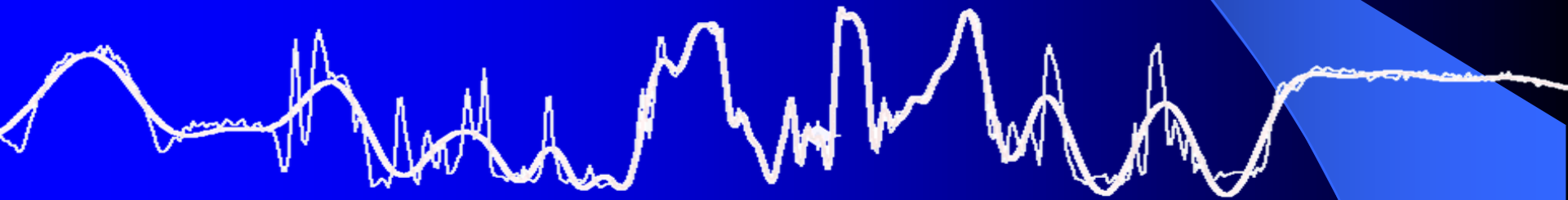
(687653; 235) Dialog1.wav 690797 (16bit) sam Fit window 14:33:36

Mix detection



Faculty of Computational Mathematics and Cybernetics
Moscow State University

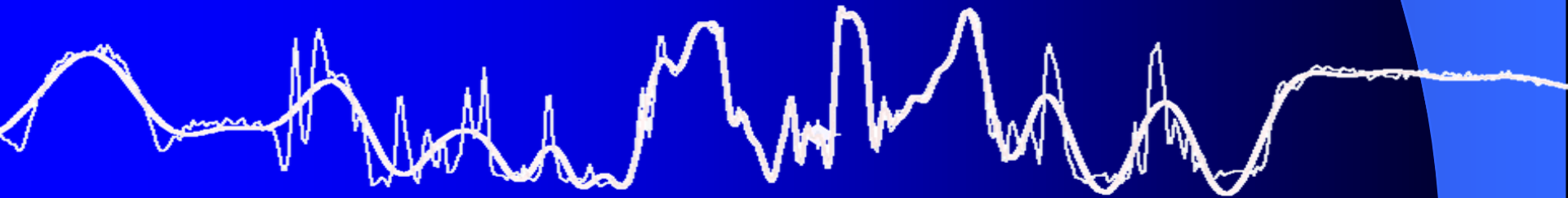
Hermite Foveation



A foveated image is a non-uniform resolution image whose resolution is highest at a point (fovea), but falls off away from the fovea.

$$(Tf)(x) = \int_{-\infty}^{\infty} k(x, t) f(t) dt$$

$$k(x, t) = \frac{1}{\alpha|x - \gamma| + \beta} g\left(\frac{t - x}{\alpha|x - \gamma| + \beta}\right)$$

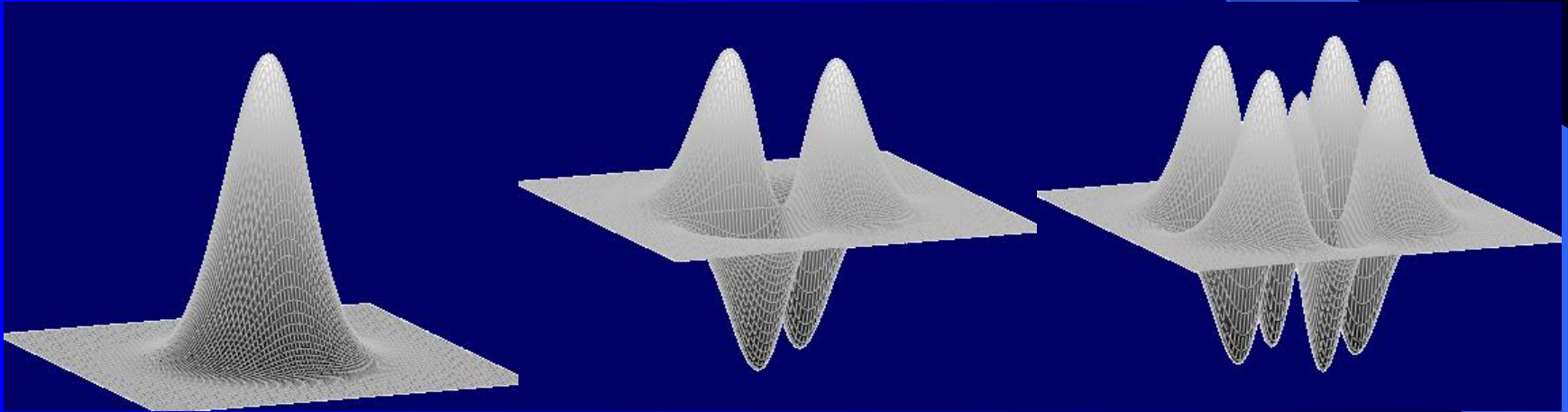


For foveation we used eigenfunctions of the Fourier transform (2D Hermite functions ψ_{nm}).

$$F(\psi_{nm}) = i^{n+m} \psi_{nm}$$

$$\psi_{nm}(x, y) = \frac{(-1)^{n+m} e^{x^2/2+y^2/2}}{\sqrt{2^{n+m} n! m! \pi}} \cdot \frac{d^n (e^{-x^2})}{dx^n} \cdot \frac{d^m (e^{-y^2})}{dy^m}$$

The graphs of the 2D Hermite functions look like the following:



$$\psi_{0,0}(x, y)$$

$$\psi_{1,1}(x, y)$$

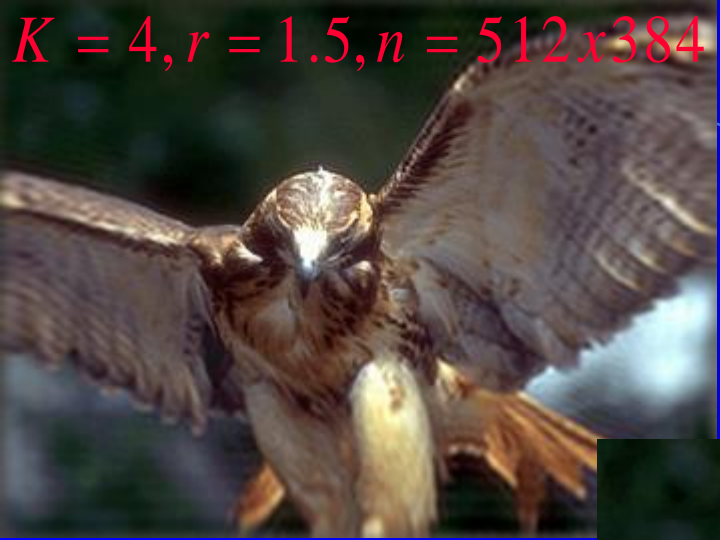
$$\psi_{2,2}(x, y)$$

The kernel for Hermite foveation was defined as:

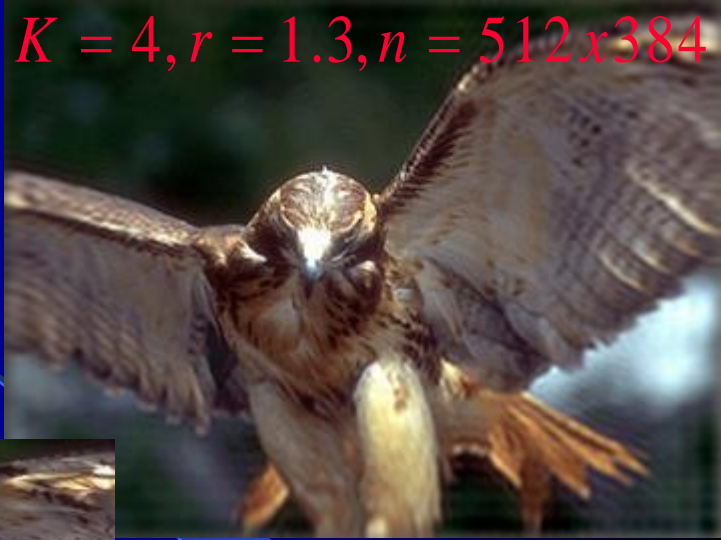
$$k(x, t) = \sum_{i=0}^{\frac{n}{K}-1} \psi_i \left(A_{\frac{n}{K}-1} \frac{2x - w + 1}{w} \right) \psi_i \left(A_{\frac{n}{K}-1} \frac{2t - w + 1}{w} \right) +$$

$$+ \sum_{j=1}^{K-1} \left(\max \left(\min \left(\frac{r}{r-1} \left(1 - \frac{2r^j |\gamma - x|}{w} \right), 1 \right), 0 \right) \cdot \sum_{i=0}^{\frac{n}{K}-1} \psi_i \left(A_{\frac{n}{K}-1} \frac{2x - w + 1}{wr^j} \right) \psi_i \left(A_{\frac{n}{K}-1} \frac{2t - w + 1}{wr^j} \right) \right)$$

$K = 4, r = 1.5, n = 512 \times 384$



$K = 4, r = 1.3, n = 512 \times 384$



Original image



$K = 8, r = 1.3, n = 512 \times 384$



$K = 16, r = 1.2, n = 512 \times 384$



$K = 4, r = 1.5, n = 512 \times 384$



$K = 4, r = 1.3, n = 512 \times 384$



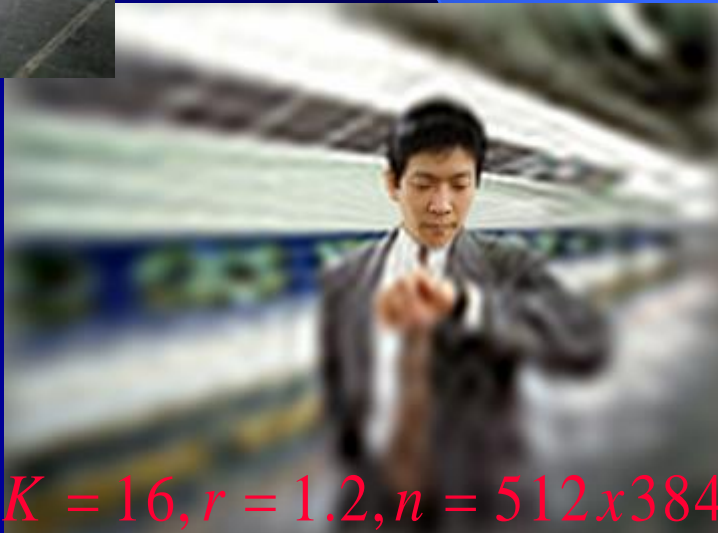
Original image



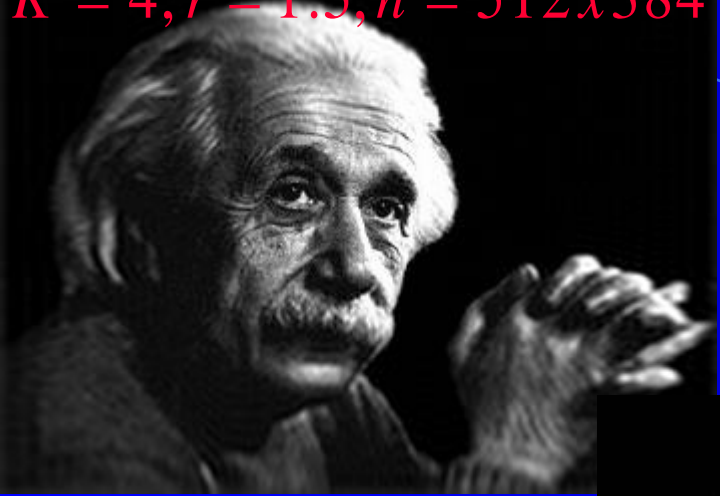
$K = 8, r = 1.3, n = 512 \times 384$



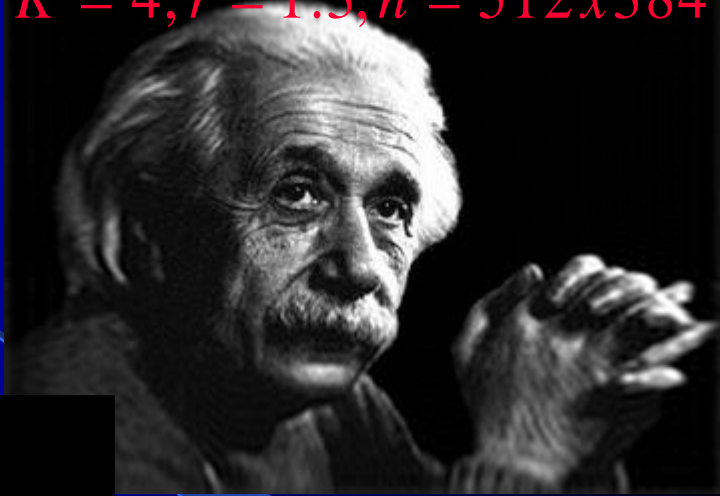
$K = 16, r = 1.2, n = 512 \times 384$



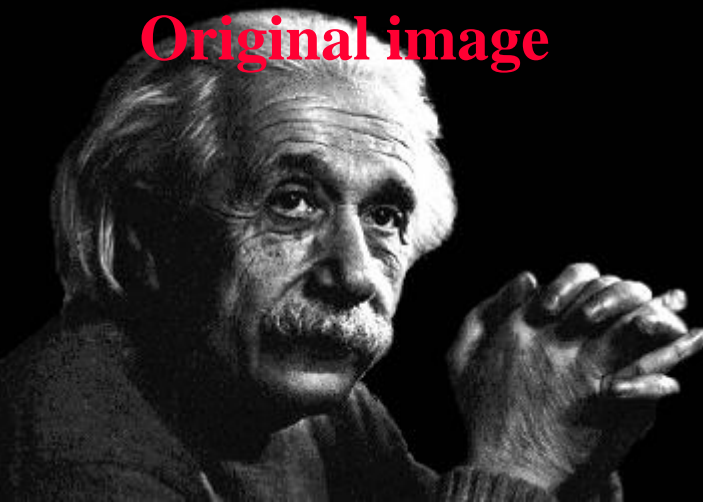
$K = 4, r = 1.5, n = 512 \times 384$



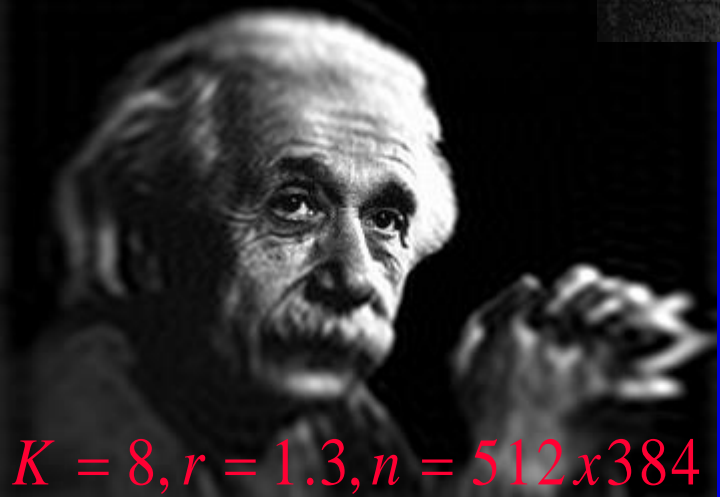
$K = 4, r = 1.3, n = 512 \times 384$



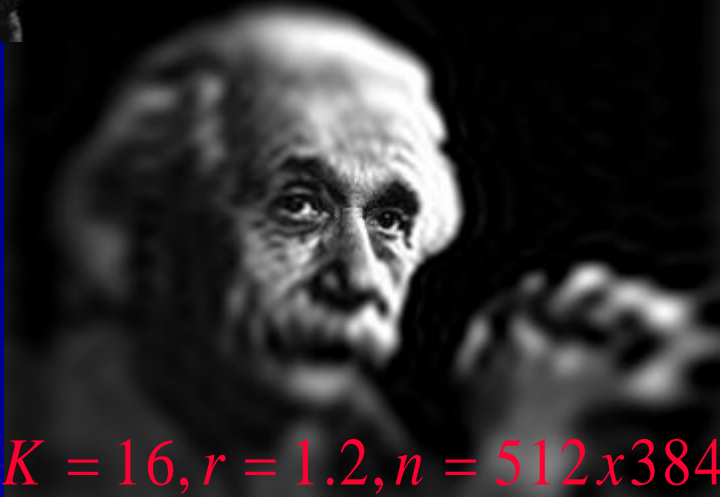
Original image



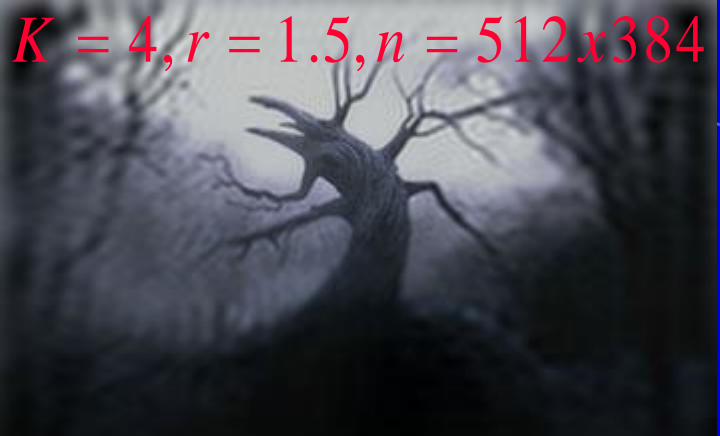
$K = 8, r = 1.3, n = 512 \times 384$



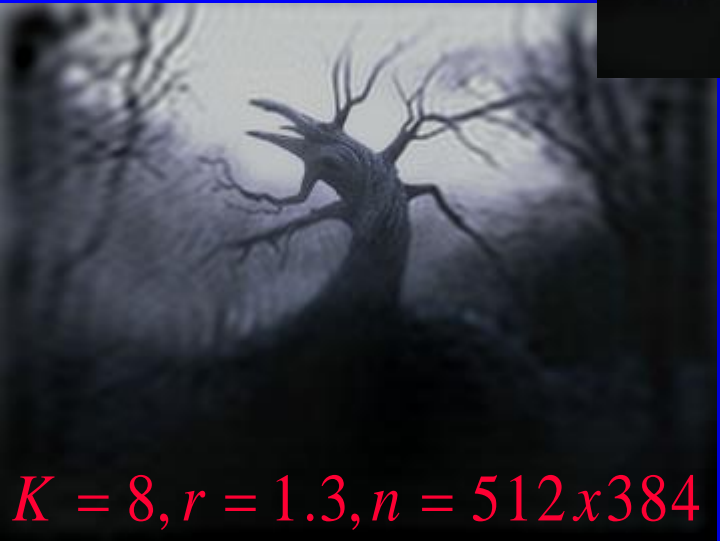
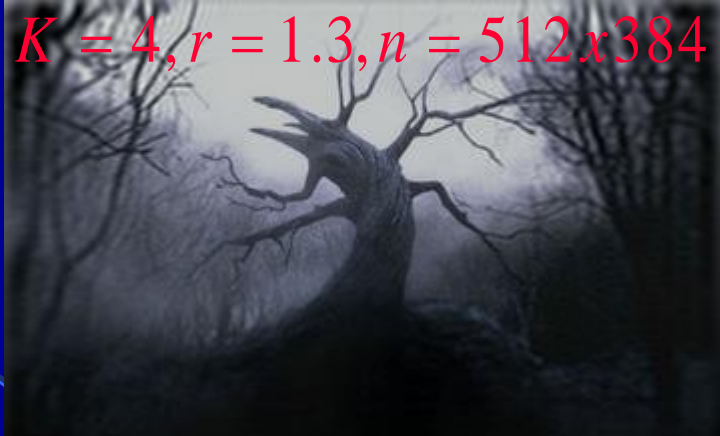
$K = 16, r = 1.2, n = 512 \times 384$



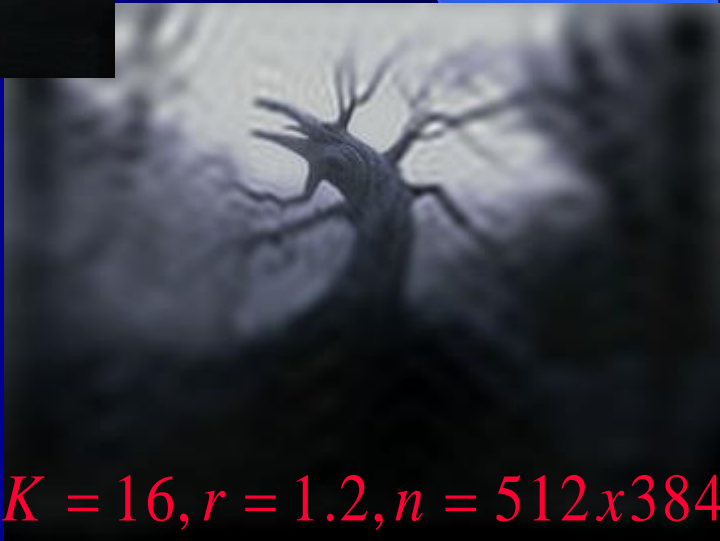
$K = 4, r = 1.5, n = 512 \times 384$



$K = 4, r = 1.3, n = 512 \times 384$



$K = 8, r = 1.3, n = 512 \times 384$



$K = 16, r = 1.2, n = 512 \times 384$

Conclusion

Hermite foveation allows us to compress useful data, to improve performance of coding/decoding and to use advantages of a time-frequency analysis.