4 Transforms

4.3: DC and KL Transforms

Transforms

4.3

1. **2D Discrete Fourier Transform (2D-DCT)**
   - Basic properties
   - Basic signal transforms
2. **2D Linear Filtering**
   - Filter design in the spatial domain
   - Filter design in the frequency domain
3. **Other transforms**
   - Discrete Cosine Transform (DCT)
   - Karhunen-Loeve Transform (KLT)
4. **Short-Term Fourier Transform (STFT)**
   - STFT as a filter bank
   - Spectogram: Time-frequency analysis

Unit Structure

4.3

1. **Introduction:**
   - Canonical representation and Discrete Fourier Transform (DFT)
2. **Discrete Cosine Transform (DCT):**
   - Basic properties
3. **Karhunen-Loeve Transform (KLT):**
   - Basic properties
4. **Compression application:**
   - Baseline JPEG
   - JPEG progressive transmision
5. **Biometry application:**
   - KLT: Face recognition
6. **Summary and Conclusions**

Linear transforms

- In the **space/frequency image model**, the image $x[m,n]$ is understood and represented as a **linear combination** of simpler (elementary) functions :

$x[m,n]$

$$= t[0,0]\; a_{00}[m,n] + t[0,1]\; a_{01}[m,n] + t[1,0]\; a_{10}[m,n] + \ldots$$

- Therefore, the image can be represented by a generic expression in terms of these elementary functions:

$$x[m,n] = \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} t[k,l]\, a_{k,l}[m,n] \quad \text{with} \quad \begin{cases} a_{k,l}[m,n] & \text{Set of elementary functions} \\ t[k,l] & \text{Coefficients associated to this image and to this set of elementary functions} \end{cases}$$

Linear transforms

4.3

So far, we have analyzed two sets of elementary functions:

- **Impulse functions**, as in the **canonical representation**.

$$x[m,n] = \sum_{m'=0}^{M-1} \sum_{n'=0}^{N-1} x[m',n']\delta[m-m',n-n']$$

- **Complex exponentials**, as in the **Fourier representation**.

$$x[m,n] = \frac{1}{MN} \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} X[k,l] e^{j2\pi\left(\frac{k}{M}m + \frac{l}{N}n\right)}$$

For any of these two representations, the set of functions being used:

- Allows perfect reconstruction of the original image. These functions form a **basis of the MxN dimensional image space.**
- Contains exactly MxN elements. The representation is said to be **complete.**
- Is formed by **orthogonal elements**. It is an orthogonal basis:
  - The Fourier transform can be normalized and become orthonormal

Linear transforms

4.3

- Every representation presents **different characteristics** that can be useful in different situations/applications:
  - **Canonical representation**: allows spatial detection and **convolution**
  - **Fourier representation**: allows **frequency analysis**
- Other linear representations are possible. They present other characteristics while preserving the completeness and orthogonality.
- To obtain the coefficients to create the representation, it is necessary to compute a **linear transform** on the input data:

Example: The DFT coefficients $X[k,l]$ representing an MxN sequence $x[m,n]$ in terms of a set of MxN complex exponentials are obtained using the **DFT transform**

$$X[k,l] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} x[m,n] e^{-j2\pi\left(\frac{k}{M}m+\frac{l}{N}n\right)}$$

Direct linear transform

4.3

The expression that, given an orthonormal space representation $a_{k.l}[m,n]$ and an image $x[m,n]$, obtains the coefficients representing the image in this space is the **direct transform**:

$$t[k,l] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} x[m,n] a_{k,l}^{*}[m,n]$$

Signal analysis Forward transform Coefficient computation		Signal synthesis Inverse transform Signal computation
$t[k,l] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} x[m,n] a_{k,l}^{*}[m,n]$	$\Leftrightarrow$	$x[m,n] = \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} t[k,l] a_{k,l}[m,n]$

Separable transform

4.3

- If 2D elementary functions can be expressed as a product of two 1D elementary functions, the transform is said to be **separable**.
- If the two 1D elementary functions are the same, the transform is said to be **symmetric**.

$$a_{k,l}[m,n] = a_k[m]b_l[n] = a_k[m]a_l[n]$$

- Symmetric and separable 2D transforms allow a **simpler expression**:

$$\boxed{t[k,l] = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} x[m,n]a_{k,l}^*[m,n] = \sum_{n=0}^{N-1} a_l^*[n] \sum_{m=0}^{M-1} a_k^*[m]x[m,n]}$$

Energy preservation

4.3

- Depending on the normalizations of the direct and inverse transforms, orthogonal and complete transforms can be forced to have elementary functions of unitary norm. In these cases, the transform is said to be an **unitary transform (orthonormal basis)**.

- In the case of unitary transforms, the **signal energy is preserved** when applying the transform. This is an extension of the **Parseval's theorem**:

$$\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\|x[m,n]\|^2 = \sum_{k=0}^{M-1}\sum_{l=0}^{N-1}\|t[k,l]\|^2$$

- Nevertheless, if the transform is not unitary only due to a normalization constant, the **relative relevance in terms of energy** between the various transformed coefficients is kept:
  - In several applications non-unitary transform definitions are preferred since they present other useful properties, e.g.: **fast implementations**

Unit Structure

4.3

1. **Introduction:**
   - Canonical representation and Discrete Fourier Transform (DFT)
2. **Discrete Cosine Transform (DCT):**
   - Basic properties
3. **Karhunen-Loeve Transform (KLT):**
   - Basic properties
4. **Compression application:**
   - Baseline JPEG
   - JPEG progressive transmision
5. **Biometry application:**
   - KLT: Face recognition
6. **Summary and Conclusions**

Definition of the DCT

4.3

- The **Discrete Fourier Transform** allows analyzing the image in the frequency domain but implies the use of **complex exponentials** as elementary functions, which produces complex transformed coefficients.

- In turn, the **Discrete Cosine Transform** (**DCT**) uses as elementary functions cosine functions which produce **real transformed coefficients**.

- There exist several ways of defining the representation **in terms of cosine functions**, depending on the type of symmetry and normalization that are to be used. One of the most common ones is the so-called **DCT-type II**:

DCT-I [editar]

$$f_j = \frac{1}{2}(x_0 + (-1)^j x_{n-1}) + \sum_{k=1}^{n-2} x_k \cos\left[\frac{\pi}{n-1} nj\right]$$

DCT-II [editar]

$$f_j = \sum_{k=0}^{n-1} x_k \cos\left[\frac{\pi}{n} j \left(k + \frac{1}{2}\right)\right]$$

Es la forma más típicamente utilizada

DCT-III [editar]

$$f_j = \frac{1}{2} x_0 + \sum_{k=1}^{n-1} x_k \cos\left[\frac{\pi}{n}\left(j + \frac{1}{2}\right) k\right]$$

DCT-IV [editar]

$$f_j = \sum_{k=0}^{n-1} x_k \cos\left[\frac{\pi}{n}\left(j + \frac{1}{2}\right)\left(k + \frac{1}{2}\right)\right]$$

$$x[m,n] = \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} \alpha(k)\alpha(l) X_{DCT}[\mathrm{k}, l] cos\left(\frac{(2m+1)}{2M} k\pi\right) cos\left(\frac{(2n+1)}{2N} l\pi\right)$$

where

$$\alpha(k=0) = \sqrt{1/M}, \alpha(k \neq 0) = \sqrt{2/M}$$

$$\alpha(l=0) = \sqrt{1/N}, \alpha(l \neq 0) = \sqrt{2/N}$$

DCT elementary functions

4.3

- The elementary DCT functions of type-II are **separable and symmetric**

$$a_{k,l}[m,n] = a_k[m]a_l[n] \qquad a_k[m] = \alpha(k)\, cos\left(\frac{(2m+1)}{2M}k\pi\right) \qquad \begin{array}{l}\alpha(k=0) = \sqrt{1/M} \\ \alpha(k \neq 0) = \sqrt{2/M}\end{array}$$

  and the direct Discrete Cosine Transform is defined as:

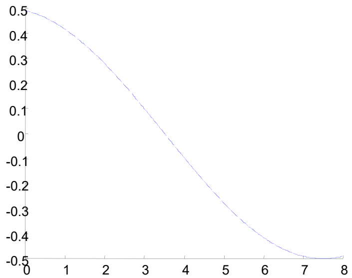
$$\boxed{X_{DCT}[k,l] = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} \alpha(k)\alpha(l)x[m,n] cos\left(\frac{(2m+1)}{2M}k\pi\right) cos\left(\frac{(2n+1)}{2N}l\pi\right)}$$

- This representation in terms of separable and symmetric functions enables **fast implementations**:
  - Two 1D transforms can be used
  - The use of images of size 2^B and the symmetries of the cosine transform allows **FFT-like implementations**.

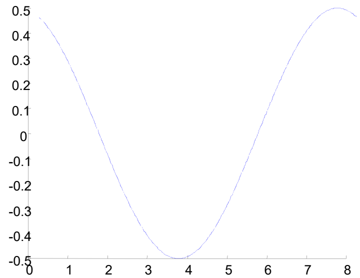
DCT elementary functions (M=8)

4.3

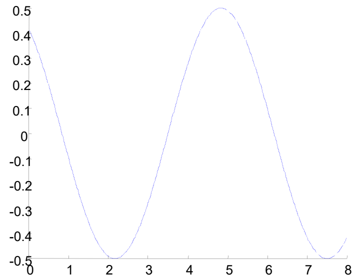
$a_0(n)$



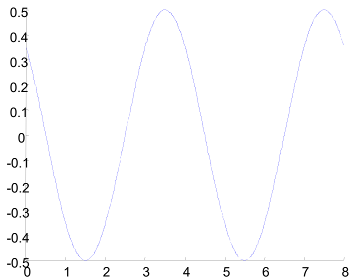
$a_1(n)$



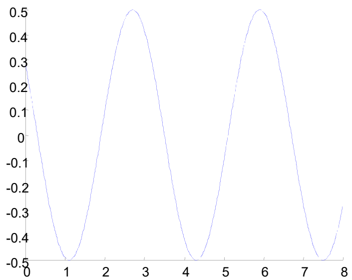
$a_2(n)$



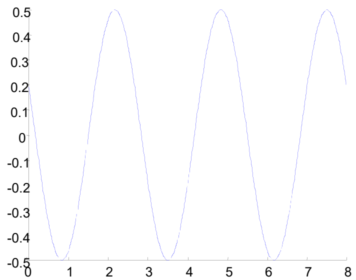
$a_3(n)$



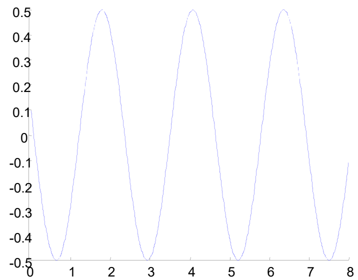
$a_4(n)$



$a_5(n)$



$a_6(n)$

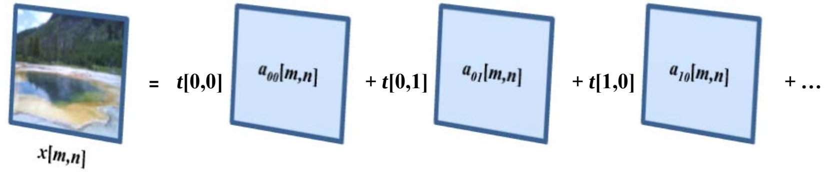


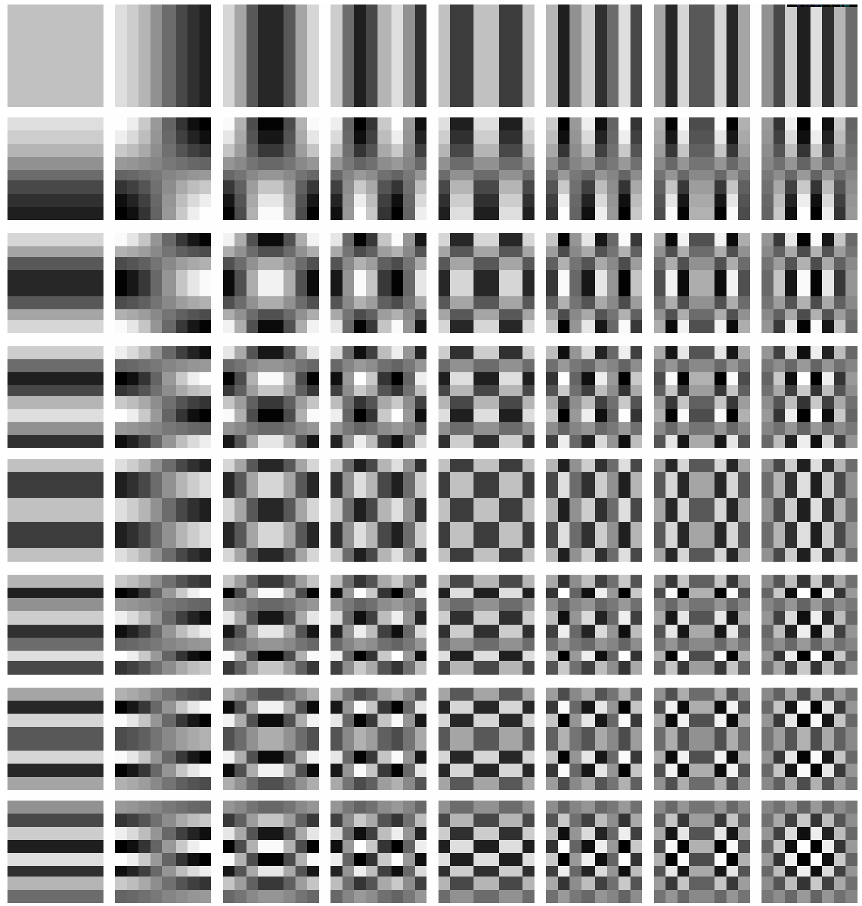
$a_7(n)$

$$a_k[m] = \alpha(k)\, cos\left(\frac{(2m+1)}{2M}k\pi\right)$$

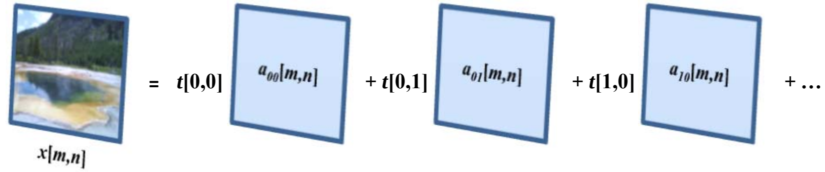
DCT elementary functions

- Given the real nature of the DCT elementary functions, they can be **visualized** (after normalization and quantization).
- Elementary functions are shown in increasing ***k* value by rows** and in increasing ***l* value by columns**.
- In this representation, images have been assumed to be of size $M = N = 8$.
- This is a common practice since usually (mainly in compression) **square, 2^B size sub-images** are analyzed rather than the whole image at once.

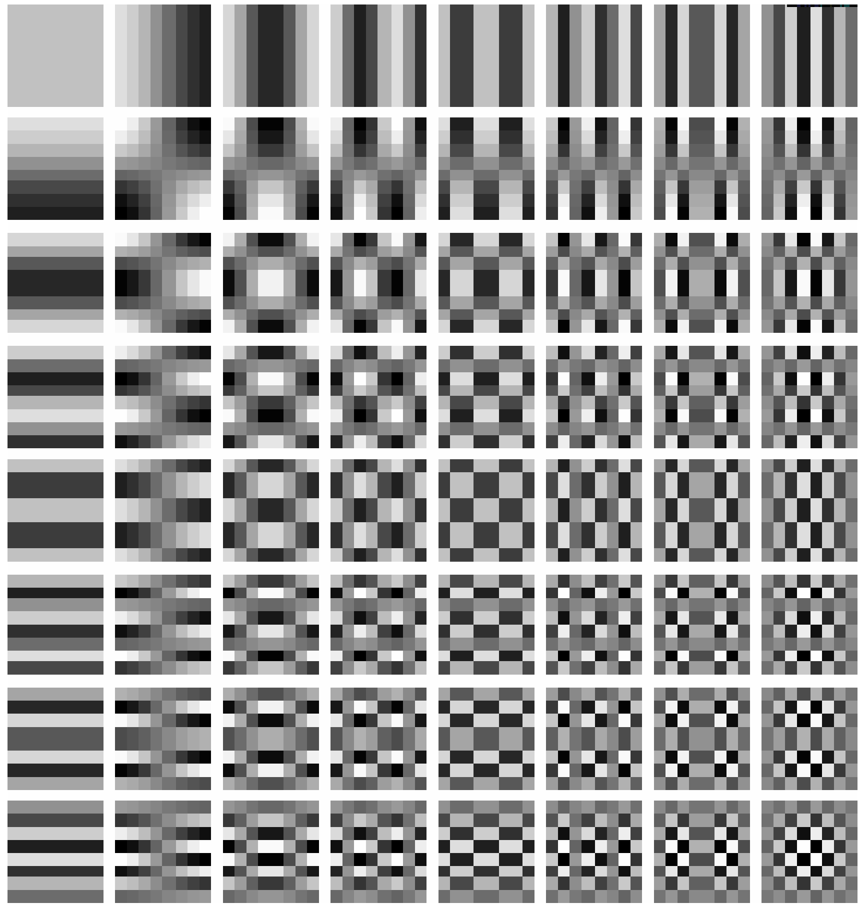




DCT elementary functions



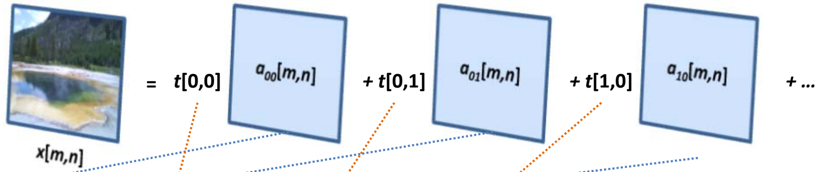
- As $M = N = 8$, a **64 Dimensional-space** has been defined
- The 64 elementary functions form **the DCT basis of the 64D-space**.
  - Note that each elementary function has 64 samples (pixels)
- The DCT basis is formed by a set of **orthogonal vectors**.
  - The cosine functions that form the elementary functions are orthogonal.

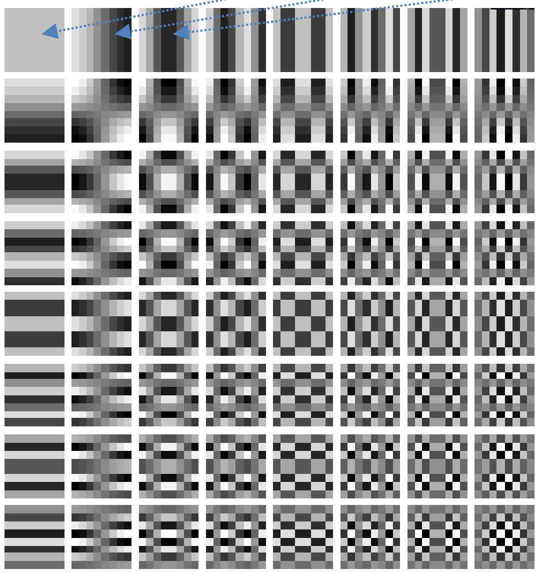


DCT elementary functions

4.3

- As $M = N = 8$, a 64 Dimensional-space has been defined





Basis: 64 images (vectors)

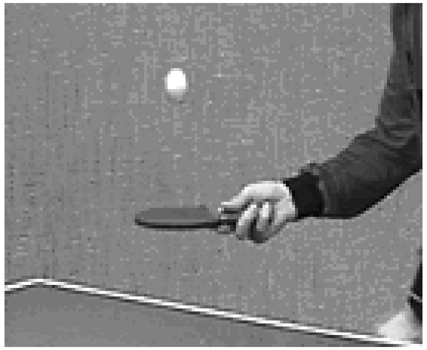
$t[0,0]$	$t[0,1]$	$t[0,2]$	....	$t[0,5]$	$t[0,6]$	$t[0,7]$
$t[1,0]$	$t[1,1]$	$t[1,2]$	....	$t[1,5]$	$t[1,6]$	$t[1,7]$
$t[2,0]$	$t[2,1]$	$t[2,2]$	....	$t[2,5]$	$t[2,6]$	$t[2,7]$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$t[5,0]$	$t[5,1]$	$t[5,2]$	....	$t[5,5]$	$t[5,6]$	$t[5,7]$
$t[6,0]$	$t[6,1]$	$t[6,2]$	....	$t[6,5]$	$t[6,6]$	$t[6,7]$
$t[7,0]$	$t[7,1]$	$t[7,2]$	....	$t[7,5]$	$t[7,6]$	$t[7,7]$

Coefficients: 64 DCT coefficients

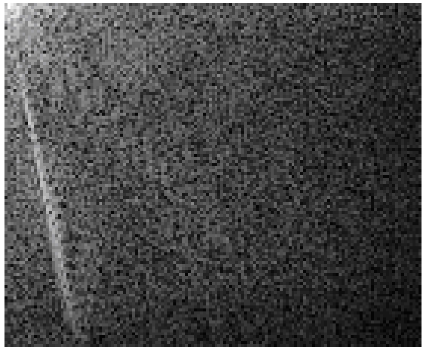
Good compactness properties

4.3

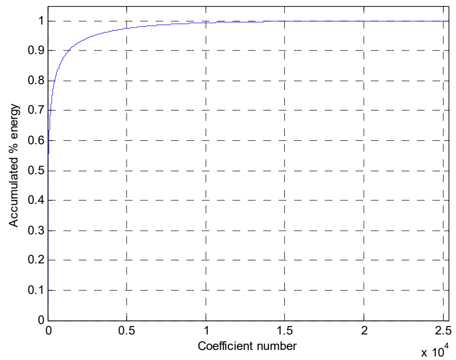
- The DCT **compacts the energy** of the MxN ($176x144 = 25.344$) original samples into a much lower number of transformed samples:
  - **Selective coding** example (higher energy):



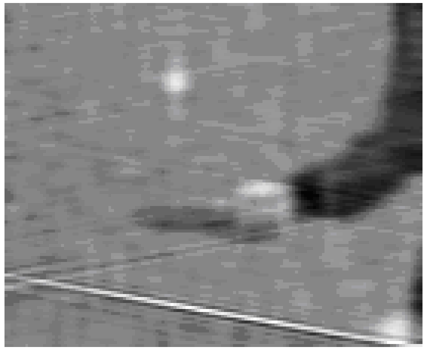
Original Image



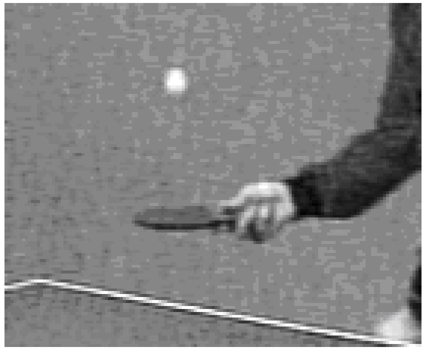
DCT coefficients



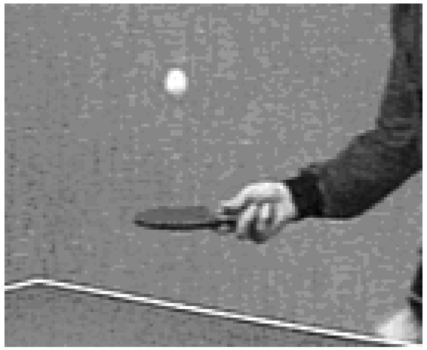
Energy distribution



N = 396 (79%)



N = 1.584 (91%)

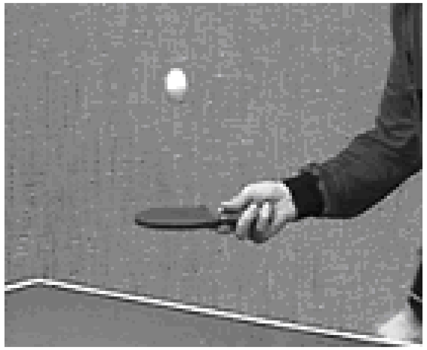


N = 3.168 (95%)

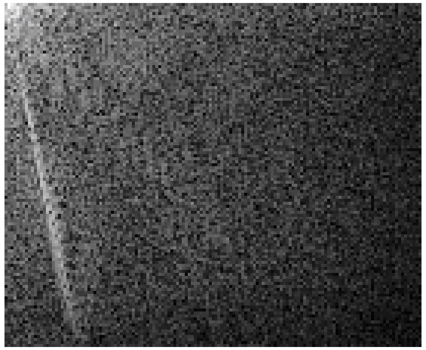
Good compactness properties

4.3

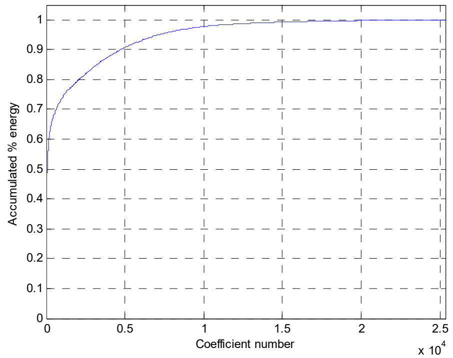
- The DCT **compacts the energy** of the MxN ($176x144 = 25.344$) original samples into a much lower number of transformed samples:
  - **Zonal coding** example (specific ordering):



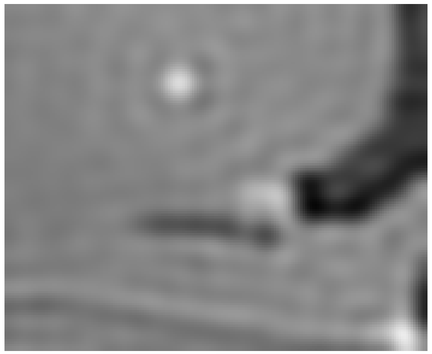
Original Image



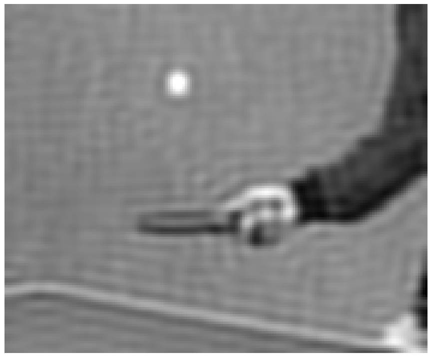
DCT coefficients



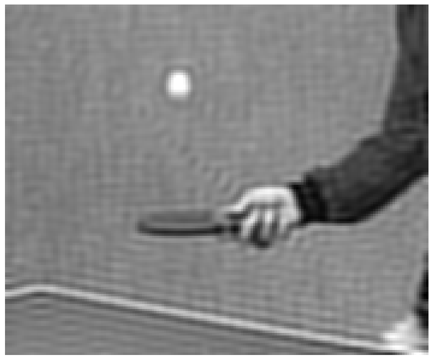
Energy distribution



N = 396 (67%)



N = 1.584 (77%)



N = 3.168 (85%)

Unit Structure

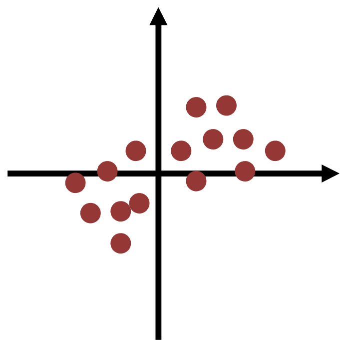
4.3

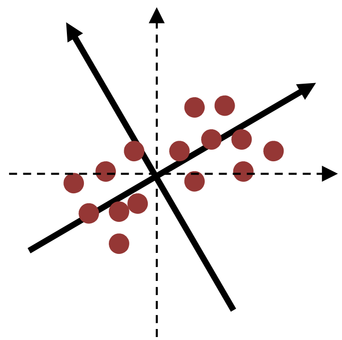
1. **Introduction:**
   - Canonical representation and Discrete Fourier Transform (DFT)
2. **Discrete Cosine Transform (DCT):**
   - Basic properties
3. **Karhunen-Loeve Transform (KLT):**
   - Basic properties
4. **Compression application:**
   - DCT: Image compression
5. **Biometry application:**
   - KLT: Face recognition
6. **Summary and Conclusions**

Optimal compactness

4.3

- Even though the DCT compacts the energy of the information into a few coefficients, **it is not the optimal transform** in the sense of being the transform that most energy compacts into the smallest number of transformed coefficients.
- The optimal transform in terms of **energy compaction** is the so-called Karhunen-Loeve Transform (**KLT**) and it is a transform that takes advantage of the **statistical characteristics** of the signal to be processed.
- The main concept consists in selecting the **representation that better adapts to the variability of the data**



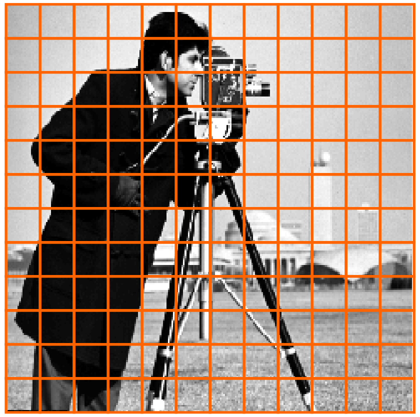


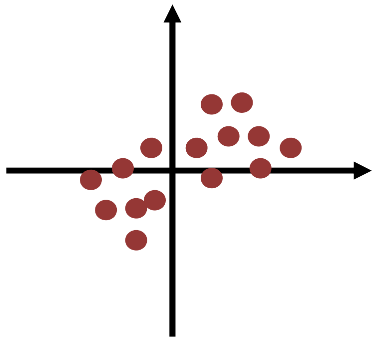
Optimal compactness

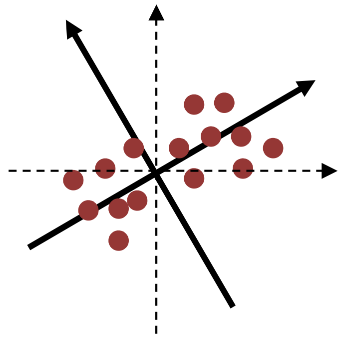
4.3

The idea is to analyze the data distribution and to select, as vectors of the basis, those vectors that are in the directions of **maximal variance of the data distribution**.

- **Example**: An image is partitioned into sub-images of size 8x8 pixels (blocks). **Each one of these blocks is a point in a 64D-space** and, for a given image (or collection of images), these points have a specific structure.
- By **analyzing the covariance matrix** of the set of points, the directions of maximal variance can be established. They are represented by the **eigenvectors** of the matrix.
- The first direction provides the best representation of the set of points with only one component: it **minimizes the energy of the representation error**. The process can be iterated to obtain a complete orthogonal basis.







Optimal transform

4.3

- The concept is to find the transform leading to **optimum energy concentration**:
  - Given a fix number of coefficients to represent a signal ($K < M$), find the transform that leads to the **minimum mean square error**.

$$\underline{\mathbf{x}} = \sum_{q=0}^{M-1} y_q \underline{\mathbf{a}}_q \quad \rightarrow \quad \underline{\tilde{\mathbf{x}}} = \sum_{q=0}^{K-1} y_q \underline{\mathbf{a}}_q \quad \rightarrow \quad \underline{\mathbf{x}} - \underline{\tilde{\mathbf{x}}} = \sum_{q=K}^{M-1} y_q \underline{\mathbf{a}}_q$$

$$\min_{\{\underline{\mathbf{a}}_q\}}(\varepsilon^2) = \min_{\{\underline{\mathbf{a}}_q\}} \left(E\left\{ \left(\underline{\mathbf{x}} - \underline{\tilde{\mathbf{x}}}\right)^T \left(\underline{\mathbf{x}} - \underline{\tilde{\mathbf{x}}}\right) \right\} \right)$$

- The optimal transform is the so-called **Karhunen-Loewe Transform** (KLT) or Hotellin Transform

Mean square error

4.3

- Let us develop the mean square error expression:

$$\varepsilon^2 = E\left\{\left(\underline{\mathbf{x}} - \underline{\tilde{\mathbf{x}}}\right)^T\left(\underline{\mathbf{x}} - \underline{\tilde{\mathbf{x}}}\right)\right\} = E\left\{\sum_{i=K}^{M-1} y_i \underline{\mathbf{a}}_i^T \sum_{j=K}^{M-1} y_j \underline{\mathbf{a}}_j\right\}$$

$$\varepsilon^2 = E\left\{\sum_{i=K}^{M-1}\sum_{j=K}^{M-1} y_i \underline{\mathbf{a}}_i^T y_j \underline{\mathbf{a}}_j\right\} = \left[\underline{\mathbf{a}}_i^T \underline{\mathbf{a}}_j = \delta_{ij}\right] = E\left\{\sum_{j=K}^{M-1} y_j y_j\right\}$$

$$\boxed{\varepsilon^2 = \left[y_j = \underline{\mathbf{a}}_j^T \underline{\mathbf{x}}\right] = E\left\{\sum_{j=K}^{M-1} \underline{\mathbf{a}}_j^T \underline{\mathbf{x}}\underline{\mathbf{x}}^T \underline{\mathbf{a}}_j\right\} = \sum_{j=K}^{M-1} \underline{\mathbf{a}}_j^T E\left\{\underline{\mathbf{x}}\underline{\mathbf{x}}^T\right\} \underline{\mathbf{a}}_j = \sum_{j=K}^{M-1} \underline{\mathbf{a}}_j^T \underline{\underline{\mathbf{R}}}_x \underline{\mathbf{a}}_j}$$

Optimization step

4.3

- In order to minimize the mean square error, while avoiding the obvious solution, an **optimization with restrictions** is proposed:

$$\left.\begin{aligned}&\min_{\{\underline{\mathbf{a}}_q\}}(\varepsilon^2) = \min_{\{\underline{\mathbf{a}}_q\}}\left(\sum_{j=K}^{M-1} \underline{\mathbf{a}}_j^T \underline{\underline{\mathbf{R}}}_x \underline{\mathbf{a}}_j\right) \\ &\text{subject to } \underline{\mathbf{a}}_j^T \underline{\mathbf{a}}_j = 1\end{aligned}\right\} \Rightarrow \mathcal{L}(\underline{\mathbf{x}}, \lambda) = f(\underline{\mathbf{x}}) - \lambda g(\underline{\mathbf{x}})$$

- A **new function is defined**, whose minimum is coincident with the minimum of the previous constrained problem:

$$\boxed{U = \varepsilon^2 - \sum_{q=0}^{M-1} \lambda_q \left(\underline{\mathbf{a}}_q^T \underline{\mathbf{a}}_q - 1\right) \quad \Rightarrow \quad \nabla U = \underline{\mathbf{0}}}$$

Optimal transform and minimum MSE

4.3

- The previous optimization problem leads to the **following solution**:
  - The vectors forming the basis of the optimum transform are the **eigenvectors of the correlation matrix**.
  - As the optimal transform depends on the autocorrelation matrix of the process, for each source **a new transform has to be computed**.

$$\nabla U = \underline{\mathbf{0}} \quad \Rightarrow \quad \underline{\underline{\mathbf{R}}}_x \underline{\mathbf{a}}_q - \lambda_q \underline{\mathbf{a}}_q = 0 \quad \Rightarrow \quad \underline{\underline{\mathbf{R}}}_x \underline{\mathbf{a}}_q = \lambda_q \underline{\mathbf{a}}_q$$

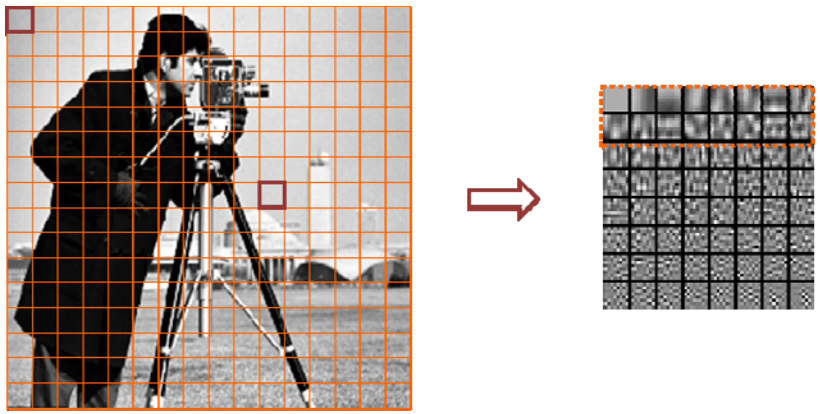
- The minimum mean square error is provided by **the sum of the eigenvalues** associated to the rejected eigenvectors:

$$\varepsilon^2 = \sum_{j=K}^{M-1} \underline{\mathbf{a}}_j^T \underline{\underline{\mathbf{R}}}_x \underline{\mathbf{a}}_j = \left[\underline{\underline{\mathbf{R}}}_x \underline{\mathbf{a}}_q = \lambda_q \underline{\mathbf{a}}_q\right] = \sum_{j=K}^{M-1} \underline{\mathbf{a}}_j^T \lambda_j \underline{\mathbf{a}}_j = \left[\underline{\mathbf{a}}_j^T \underline{\mathbf{a}}_j = 1\right] = \sum_{j=K}^{M-1} \lambda_j$$

Minimum MSE and KLT

4.3

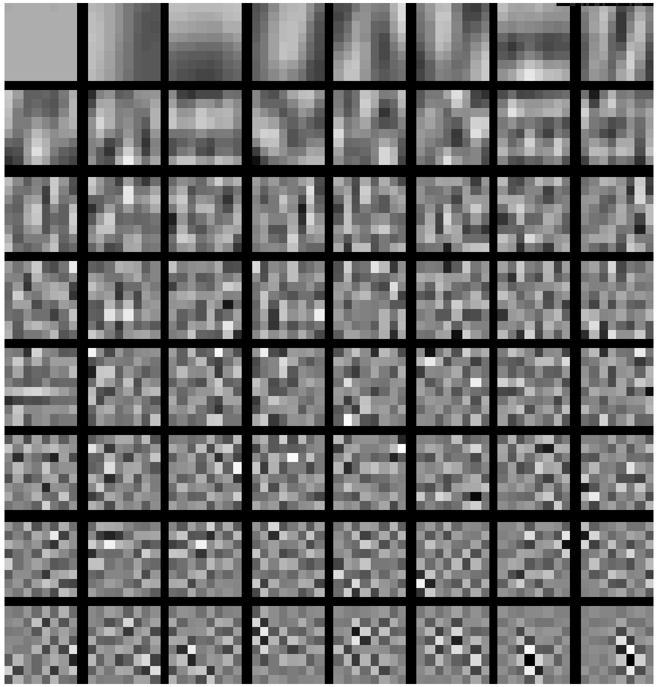
- The methodology to, given a fix number of coefficients to represent a signal (K < M), find the transform that leads to the **minimum mean square error** is:
    1. Compute (estimate) the autocorrelation matrix of the signal (process) R_x
    2. Compute the set of eigenvalues $\{\lambda_q\}$ and eigenvectors $\{\underline{a}_q\}$
    3. Order the eigenvectors $\{\underline{a}_q\}$ from λ_{max} to λ_{min} and re-index $\{\underline{a}_q\}$
    4. Select the K first autovectors



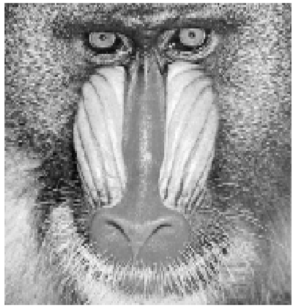
KLT elementary functions

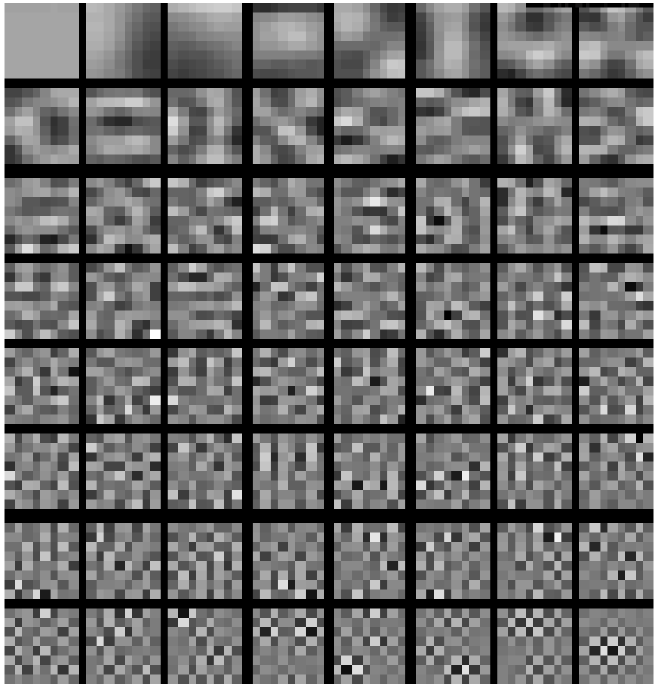
4.3

- The obtained bases (elementary images) depend on the **source statistics:**
- Bases for different sources (image collections) are not equal … **but may be similar.**



Elementary functions (KLT bases) for the Cameraman image, ordered by descending λ value.

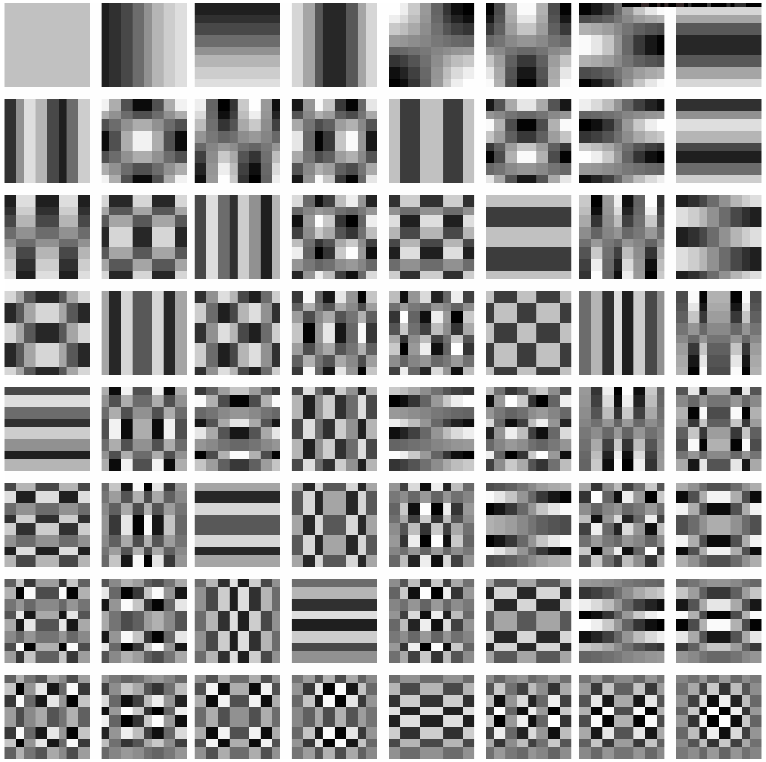




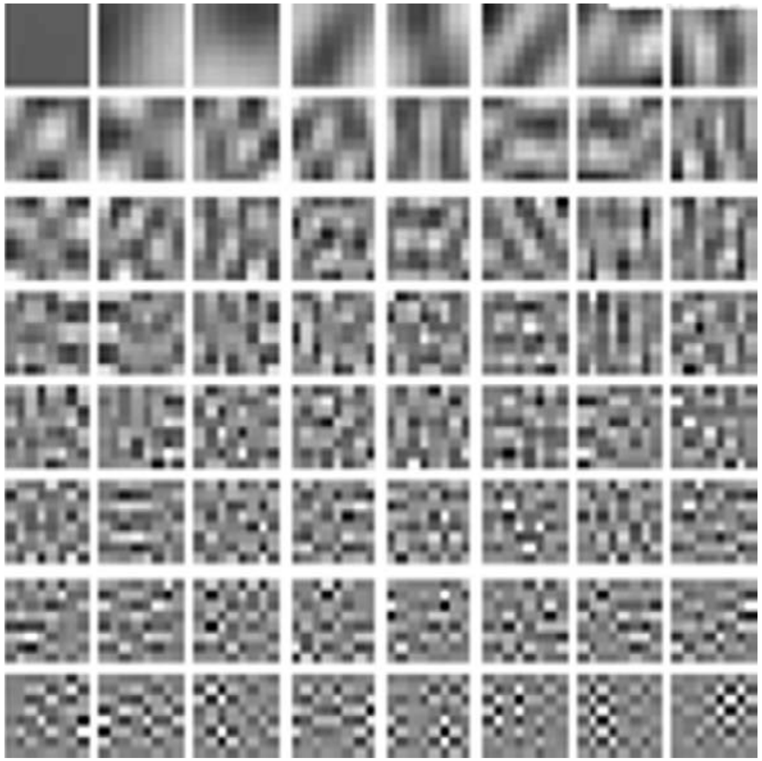
Elementary functions (KLT bases) for the Baboon image, ordered by descending λ value.

How does DCT compare with KLT?

4.3



DCT basis



KLT basis

Weak and strong points of KLT

4.3

- The KLT is the **optimal transform** in terms of compacting the energy in the minimum number of transformed samples.
- Nevertheless, the KLT basis is not a fixed one but **it is data dependent**. It exploits the statistical properties of the data.
  - For each collection of data, a **new basis has to be computed**.
  - The computation of the basis is **neither simple nor recursive**.
- Usually, given this data dependency, the optimal vectors **cannot be represented in terms of analytical functions** and, therefore, the elementary functions are not easy to handle:
  - KLT elementary functions are **not separable**
  - Fast implementations of the KLT are **not possible**

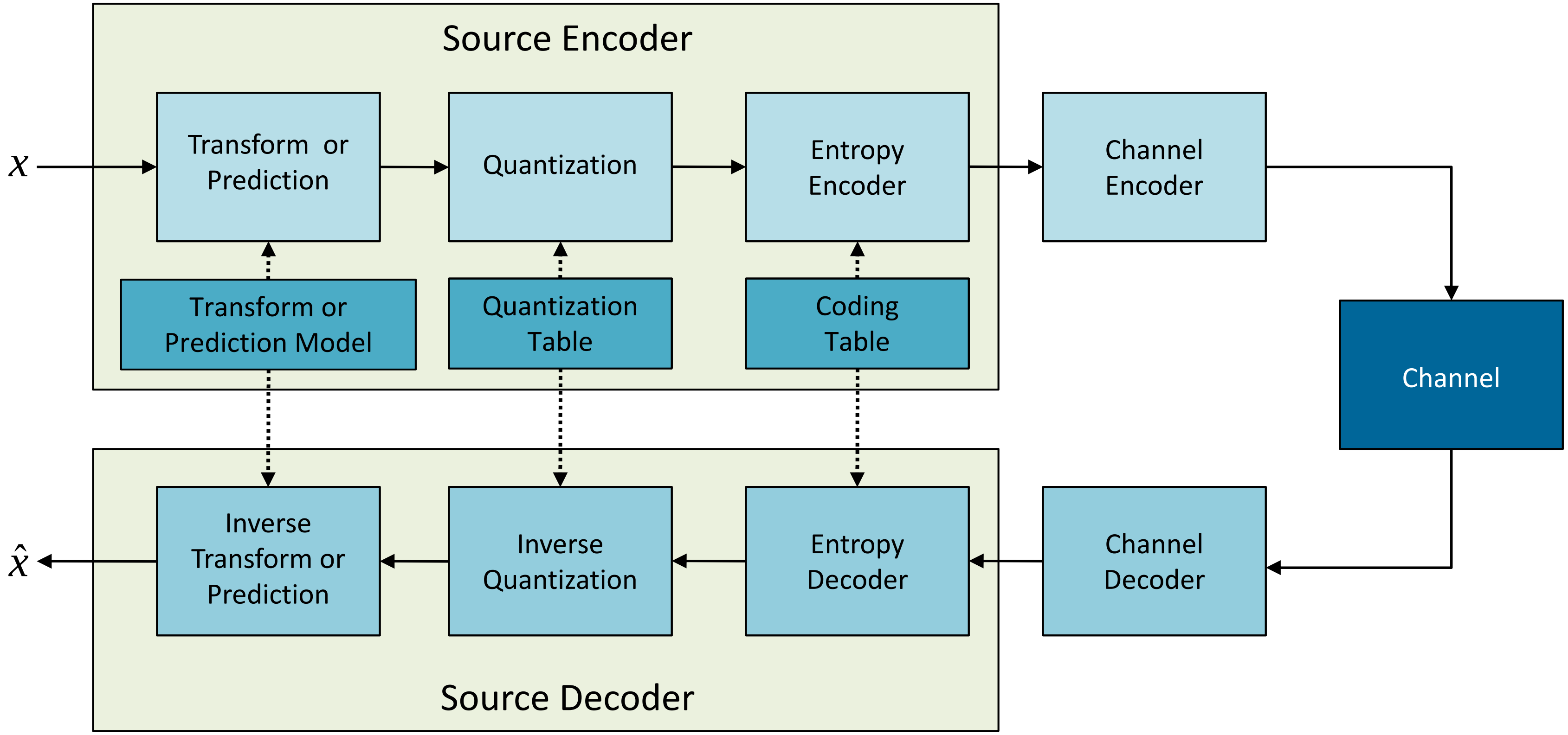
Unit Structure

4.3

1. **Introduction:**
   - Canonical representation and Discrete Fourier Transform (DFT)
2. **Discrete Cosine Transform (DCT):**
   - Basic properties
3. **Karhunen-Loeve Transform (KLT):**
   - Basic properties
4. **Compression application:**
   - DCT: Image compression
5. **Biometry application:**
   - KLT: Face recognition
6. **Summary and Conclusions**

Compression application

4.3



Basic structure of a coder / decoder (codec) system

JPEG standard

4.3

JPEG (ISO/IEC IS 10918)

Joint Pictures Expert Group

Jointly published (1990) as an ISO/IEC and ITU-T standard

Digital Compression and Coding of

Continuous-tone Still Images

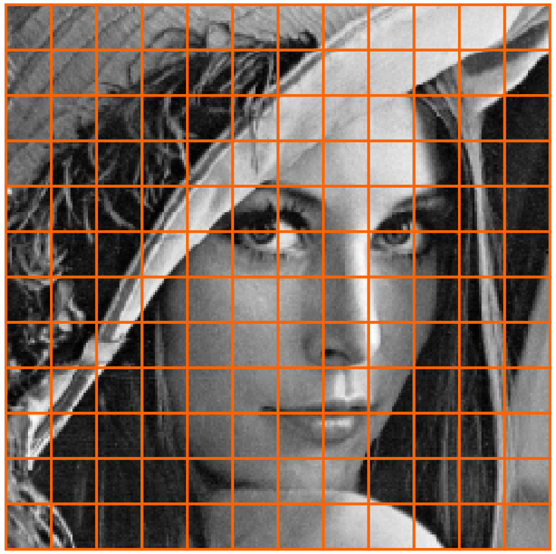
It contains several parts:

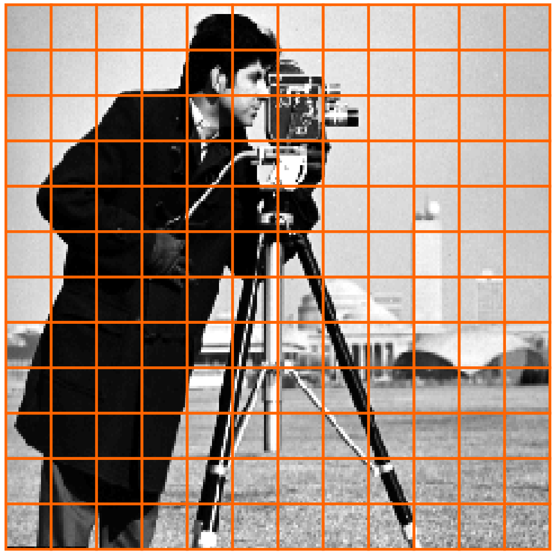
- **Part 1: Requirements and guidelines**
- Part 2: Compliance testing: Soft
- Part 3: Extensions: including the SPIFF file format
- Part 4: Registration: Defines parameters used to extend JPEG

DCT-based image compression

4.3

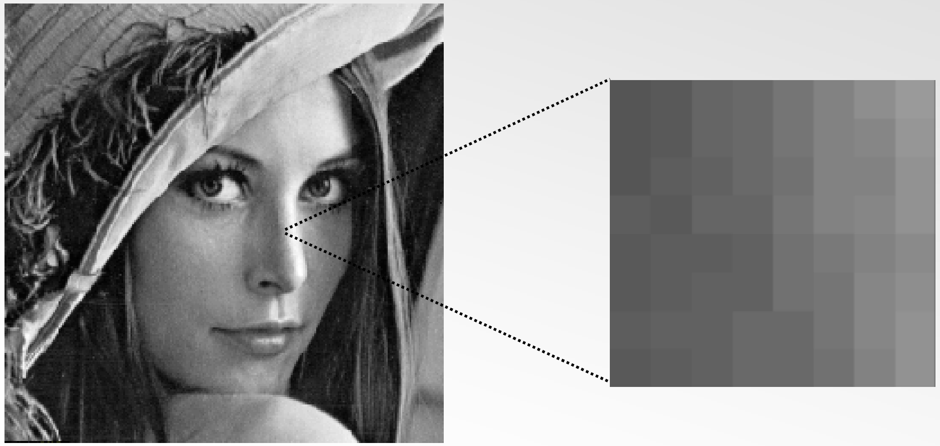
- Images are (strongly) **non stationary data**:
  - Division of the image into stationary regions
  - Arbitrary regions (dependent on the image) will require the transmission of the partition
  - Fix partition independent of the image: **Division into blocks**.
  - The partition may be visible in the decoded image: **Blocking effect**





DCT-based image compression

- The block dimensions are a **trade-off**:
  - If the signal is stationary, the larger the block, the more compaction is obtained. But, in an image, it is difficult to find large stationary blocks.
  - Commonly, blocks of **8x8 pixels** are used (e.g.: in the JPEG standard)



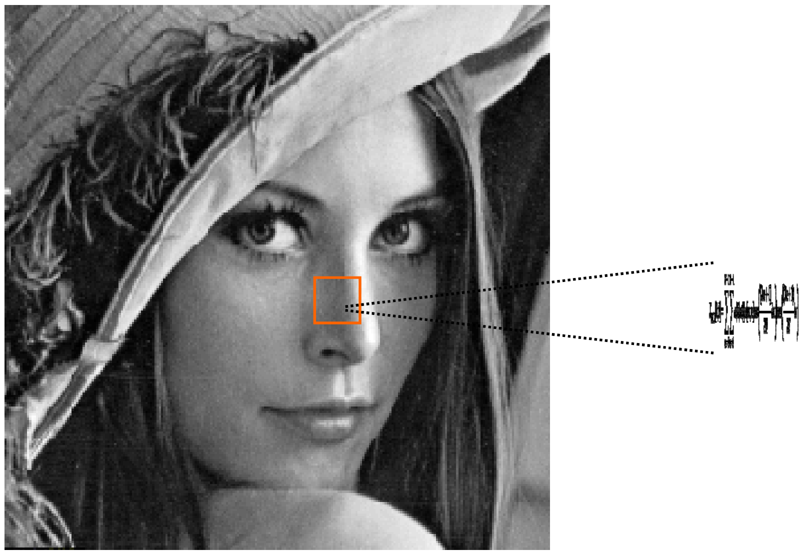
$$\mathbf{x} = \begin{bmatrix} 87 & 89 & 101 & 106 & 118 & 130 & 142 & 155 \\ 85 & 91 & 101 & 105 & 116 & 129 & 135 & 149 \\ 86 & 92 & 96 & 105 & 112 & 128 & 131 & 144 \\ 92 & 88 & 102 & 101 & 116 & 129 & 135 & 147 \\ 88 & 94 & 94 & 98 & 113 & 122 & 130 & 139 \\ 88 & 95 & 98 & 97 & 113 & 119 & 133 & 141 \\ 92 & 99 & 98 & 106 & 107 & 118 & 135 & 145 \\ 89 & 95 & 98 & 107 & 104 & 112 & 130 & 144 \end{bmatrix}$$

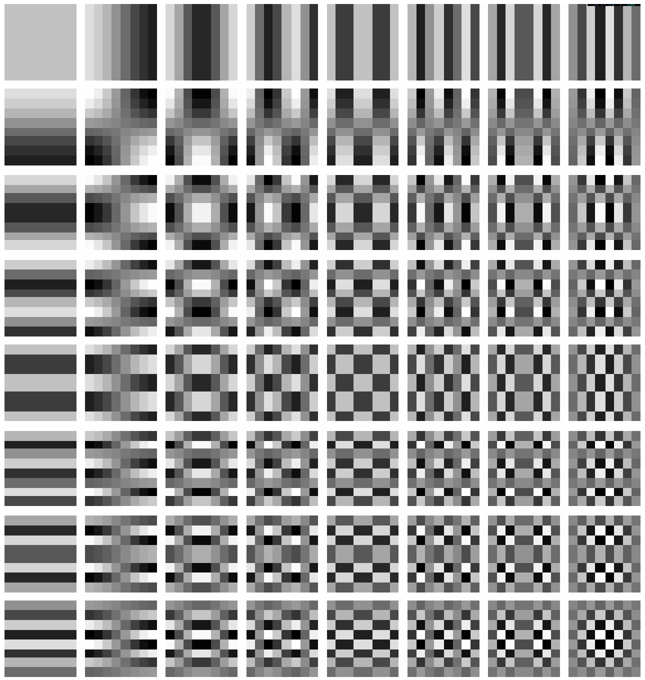
DCT-based image compression

4.3

- The DCT transform of each block is computed. That is, each block in the image is **projected** onto the DCT elementary functions.

$$X_{DCT}[k,l] = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} \alpha(k)\alpha(l)x[m,n]cos\left(\frac{(2m+1)}{2M}k\pi\right)cos\left(\frac{(2n+1)}{2N}l\pi\right)$$





DCT: JPEG implementation

4.3

$$\mathbf{x} = \begin{bmatrix} 87 & 89 & 101 & 106 & 118 & 130 & 142 & 155 \\ 85 & 91 & 101 & 105 & 116 & 129 & 135 & 149 \\ 86 & 92 & 96 & 105 & 112 & 128 & 131 & 144 \\ 92 & 88 & 102 & 101 & 116 & 129 & 135 & 147 \\ 88 & 94 & 94 & 98 & 113 & 122 & 130 & 139 \\ 88 & 95 & 98 & 97 & 113 & 119 & 133 & 141 \\ 92 & 99 & 98 & 106 & 107 & 118 & 135 & 145 \\ 89 & 95 & 98 & 107 & 104 & 112 & 130 & 144 \end{bmatrix} \xrightarrow{\boxed{\text{DCT}}} \mathbf{X} = \begin{bmatrix} 898.0000 & -149.5418 & 26.6464 & -14.0897 & 0.7500 & -5.7540 & 3.5750 & 0.0330 \\ 12.1982 & -16.5235 & -7.6122 & 5.2187 & -0.2867 & -1.9909 & 8.4265 & 1.2591 \\ 5.3355 & -2.6557 & 2.3410 & -9.9277 & 2.4614 & 4.4558 & -3.1945 & -3.1640 \\ 1.9463 & -2.7271 & 1.5106 & 2.8421 & -2.1336 & -2.7203 & -2.7510 & 5.4051 \\ 0.7500 & -2.0745 & 0.8610 & 0.2085 & 2.5000 & 1.8446 & 2.0787 & 2.4750 \\ 7.9536 & -2.6624 & 2.6308 & 0.4010 & 0.4772 & 3.3000 & 1.7394 & 0.3942 \\ -4.1042 & -0.1650 & -0.6945 & 0.0601 & 0.0628 & -0.7874 & -0.8410 & 0.3496 \\ -3.4688 & 2.3804 & 0.1559 & 0.8696 & 0.1142 & -0.5240 & -3.9974 & -5.6187 \end{bmatrix}$$

- In the **JPEG standard**, the DCT is not implemented as a unitary transform:
  - Due to old implementation issues, the **mean value is multiplied by 8**.

- The transformed coefficients ($X_{DCT}[k,l]$) with highest absolute values (energy) **appear gathered in the upper left corner**, around the coefficient (0, 0).
  - This is a common behavior for most images. In general, the energy is compacted in a few coefficients.

JPEG Quantization

4.3

- The transform coefficients are quantized taking into account their specific relevancy for the **human visual system**:
  - Quantization tables have been empirically determined: **Losheller tables**.
- The **compression can be controlled** by using a different table:
  - iQ with $i > 1$

$$\mathbf{Q} = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

$$\hat{X}(k,l) = round\left[\frac{X(k,l)}{Q(k,l)}\right]$$

JPEG Quantization

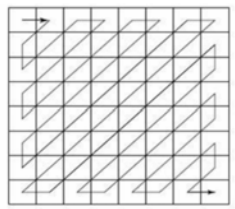
- The *round* [.] operation produces a **large number of zeros** in the quantized transformed coefficients. These zeros lead to the **losses in quality** but allow for a strong **reduction of the amount of necessary bits** to represent the image.
- **Larger *i* values** increase the number of quantized transformed coefficients equal to zero, increasing the compression and the losses in quality.

$$\hat{\mathbf{X}} = \begin{bmatrix} 56 & -14 & 3 & -1 & 0 & 0 & 0 & 0 \\ 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

***i* = 1**

$$\hat{\mathbf{X}} = \begin{bmatrix} 28 & -7 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

***i* = 2**



Zigzag scanning

JPEG typical compression results

Original

i = 1   C = 8.54   0.94 bpp   SNR = 25.3 dB

JPEG typical compression results

4.3

Original

i = 2 C = 13.2 0.61 bpp SNR = 23.3 dB

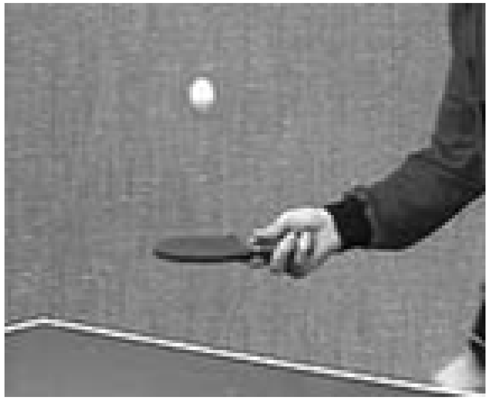
JPEG typical compression results

4.3

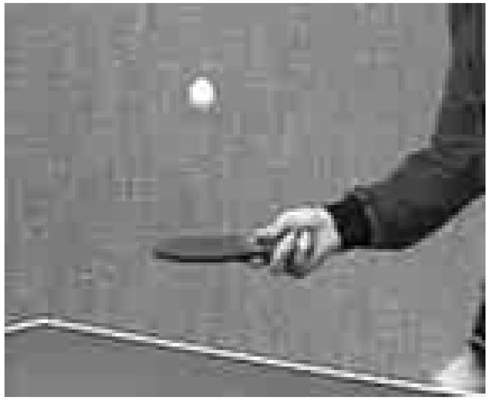
Original

i **= 4   C = 20.6   0.39 bpp   SNR = 21 dB**

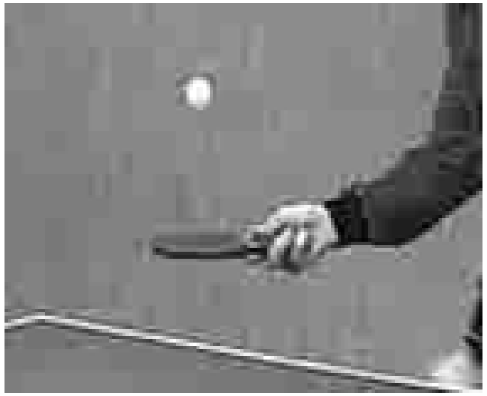
JPEG typical compression results (QCIF)



i = 1; PSNR = 32.5 dB
bits/pixel = 0.89
compression ratio = 8.99



i = 3; PSNR = 29.1 dB
bits/pixel = 0.36
compression ratio = 22.10



i = 5; PSNR = 27.8 dB
bits/pixel = 0.25
compression ratio = 32.51

i = 1; PSNR = 30.0 dB
bits/pixel = 0.95
compression ratio = 25.1

i = 3; PSNR = 27.2 dB
bits/pixel = 0.41
compression ratio = 59.1

i = 5; PSNR = 25.9 dB
bits/pixel = 0.30
compression ratio = 78.8

JPEG typical compression results

4.3

$i = 1$; PSNR = 35.2 dB
bits/pixel = 0.72
compress. ratio = 11.2

$i = 3$; PSNR = 31.8 dB
bits/pixel = 0.35
compress. ratio = 22.7

$i = 5$; PSNR = 30.0 dB
bits/pixel = 0.25
compress. ratio = 30.9

T. Dutoit, F. Marques, Applied Signal Processing: A Matlab-based proof of concept, Springer, 2009. ISBN: 978-0-387-74534-3

JPEG typical compression results

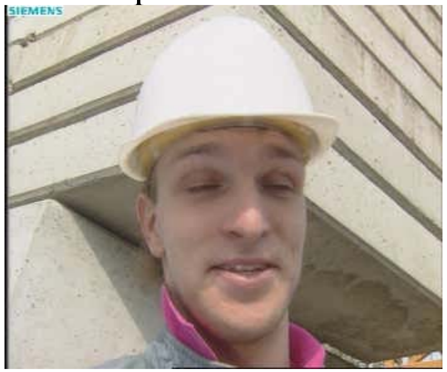
4.3

i = 1; PSNR = 31.3 dB
bits/pixel = 1.12
compress. ratio = 41.4

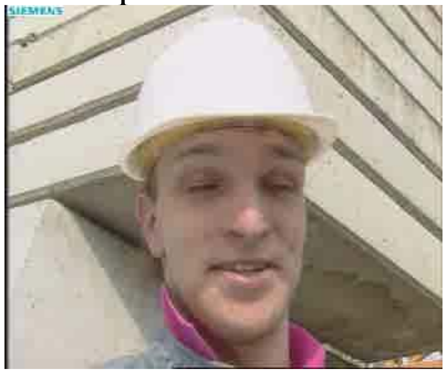
i = 3; PSNR = 27.8 dB
bits/pixel = 0.56
compress. ratio = 42.5

i = 5; PSNR = 26.9 dB
bits/pixel = 0.42
compress. ratio = 57.3

QCIF: 176 × 144



i = 1; PSNR = 33.6 dB
bits/pixel = 0.76
compress. ratio = 31.7



i = 3; PSNR = 30.3 dB
bits/pixel = 0.39
compress. ratio = 61.8

i = 5; PSNR = 28.5 dB
bits/pixel = 0.30
compress. ratio = 80.4

CIF: 352 × 288

JPEG progressive transmission

4.3

- It allows transmitting a **coarse version** of the image at low rate and, **progressively**, improve its quality by the subsequent transmissions:
- **Spectral selection** takes advantage of the “spectral" (spatial frequency spectrum) characteristics of the DCT coefficients. **Higher AC components** provide detail information. For each block:
    - Scan 1: Encode DC and first few AC components; e.g., AC1, AC2.
    - Scan 2: Encode a few more AC components; e.g., AC3, AC4, AC5.
    - ...
    - Scan k: Encode the last few ACs; e.g., AC61, AC62, AC63.

❖ **Successive approximation** takes advantage of the **bit plane information**. All DCT coefficients are encoded simultaneously but with their most significant bits (MSBs) first. For each block:

- Scan 1: Encode the first few MSBs, e.g., Bits 7, 6, 5, 4.
- Scan 2: Encode a few more less significant bits, e.g., Bit 3.
- ...
- Scan m: Encode the least significant bit (LSB), Bit 0

Example of spectral selection

4.3

- **Spectral selection** takes advantage of the spatial frequency spectrum characteristics of the DCT coefficients. **Higher AC components** provide detail information:
  - **Scan 1: Encode DC and first few AC components; e.g., AC1, AC2.**
  - Scan 2: Encode a few more AC components; e.g., AC3, AC4, AC5.
  - ...
  - Scan k: Encode the last few ACs; e.g., AC61, AC62, AC63.

JPEG progressive transmission

4.3

- **Spectral selection** takes advantage of the spatial frequency spectrum characteristics of the DCT coefficients. **Higher AC components** provide detail information:
  - Scan 1: Encode DC and first few AC components; e.g., AC1, AC2.
  - **Scan 2: Encode a few more AC components; e.g., AC3, AC4, AC5.**
  - **...**
  - Scan k: Encode the last few ACs; e.g., AC61, AC62, AC63.

JPEG progressive transmission

4.3

- **Spectral selection** takes advantage of the spatial frequency spectrum characteristics of the DCT coefficients. **Higher AC components** provide detail information:
  - Scan 1: Encode DC and first few AC components; e.g., AC1, AC2.
  - Scan 2: Encode a few more AC components; e.g., AC3, AC4, AC5.
  - ...
  - **Scan k: Encode the last few ACs; e.g., AC61, AC62, AC63.**

Unit Structure

4.3

1. **Introduction:**
   - Canonical representation and Discrete Fourier Transform (DFT)
2. **Discrete Cosine Transform (DCT):**
   - Basic properties
3. **Karhunen-Loeve Transform (KLT):**
   - Basic properties
4. **Compression application:**
   - DCT: Image compression
5. **Biometry application:**
   - KLT: Face recognition
6. **Summary and Conclusions**

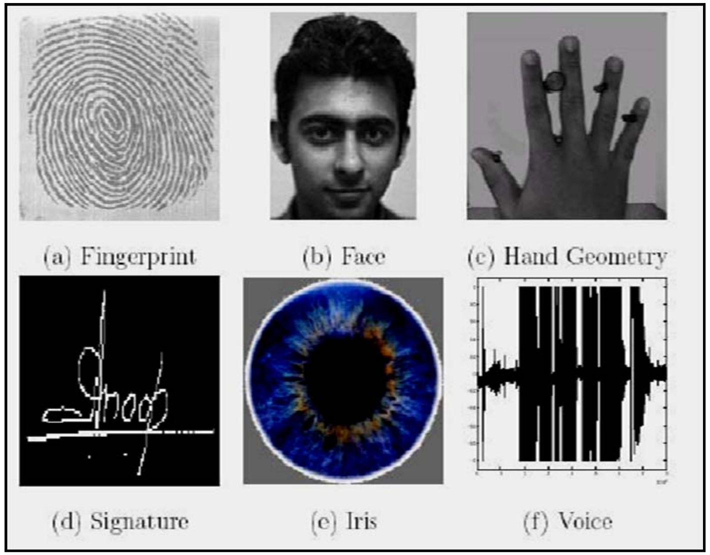
Biometry

4.3

- **Speech biometrics**
  - Text-dependent speaker recognition
  - Text-independent speaker recognition
  - Speaker indexing

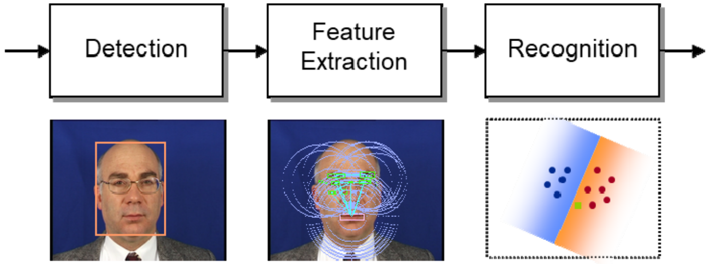
- **Image biometrics**
  - **Face recognition**
  - Fingerprint recognition
  - Iris recognition
  - Human activity recognition (behaviour)
    - Gait
    - Signature
  - Others biometric approaches

- **Multimodal biometric systems**



(a) Fingerprint (b) Face (c) Hand Geometry

(d) Signature (e) Iris (f) Voice

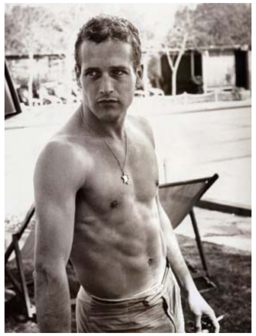


Attributes of a biometric system

- **Universality**: Each person must have the proposed characteristic:
  - ✓ OK for faces.

- **Uniqueness:** Two persons should not be identified as the same one
  - o More complicated. Extreme case: twins

- **Permanence:** Characteristic is invariant over time
  - ➢ Short term and long term variations.



Attributes of a biometric system

4.3

- **Collectability:** Characteristic should be measured with a sensing device
  - ✓ A simple camera can be used (be careful with the simplicity).

- **Circumvention:** Ease of use of a substitute

Circumvention: Paycheck (2003)

The modern burglar

Circumvention: The Saint (1997)

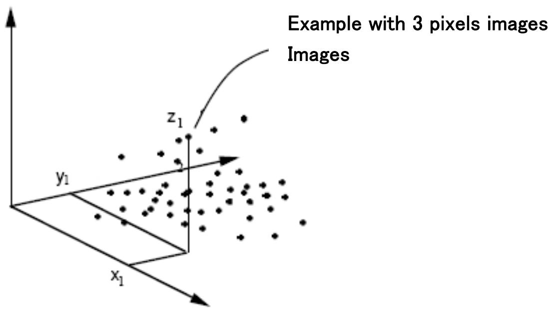
Other attributes

4.3

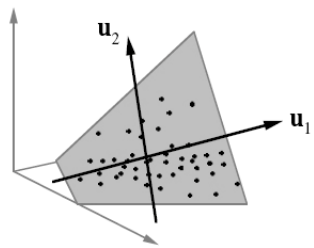
- **Low-cost:** Expensive systems have no chance of large scale deployment
  - ✓ Face recognition systems are not expensive
- **Ordinary-people friendly:** Problems deploying in large scale
  - o Face recognition: Cultural or climatic
- **Non-invasiveness:** People will NOT cooperate in invasive procedures
  - ➢ Face recognition: Invasiveness improve performance
- **Easy to enroll:** Users should be easily enrolled
  - ✓ Face recognition: Just a few pictures
- **Fast & easy authentication:** Long delay systems cannot be deployed
  - ✓ Face recognition: Fast algorithms for recognition
- **Easy to archive & audit trail:** Be able to quickly trace, locate and retrieve data
  - ✓ Face recognition: Images or just parameters to be retrieved
- **Connotation:** Even the best technology cannot be accepted if it has a negative reputation
  - o Face recognition: Camera networks are accepted nowadays … but …

KLT-based face recognition

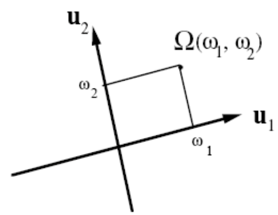
- Face images (with common pose) are very redundant. The face data lies on a low dimensional subspace. The KLT gives a global compact representation of this class of images: **Face subspace**.



3D space



2D subspace

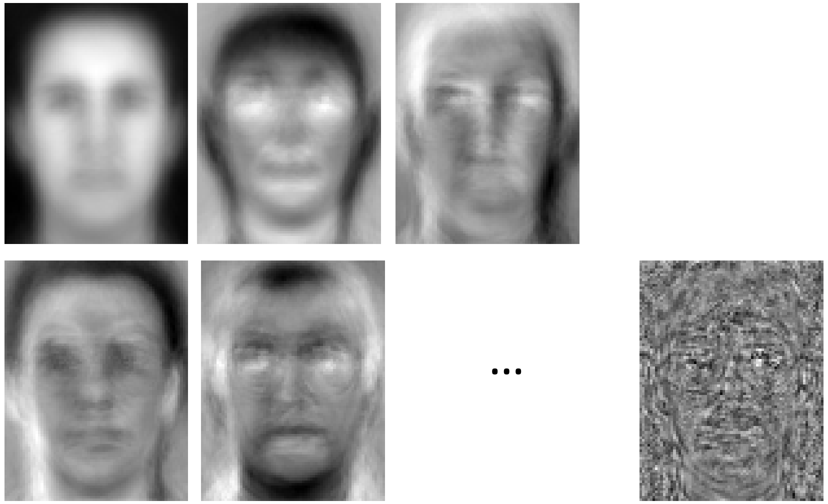


Eigenvectors

- In the case of **face recognition**, the usual way to work is:
  - Use face images of the same size L = MxN.
  - Use the KLT to estimate the **face representation space** of dimension M << N.
    - The M eigenvectors that span the face space are usually referred to as **eigenfaces**.
  - A linear combination of eigenfaces can describe any face image

KLT-based face recognition

- Face images **do not perfectly lie** on an M-dimensional space.
- All eigenvectors and their associated eigenvalues are studied. The **M vectors with higher associated eigenvalues** are assumed to form the face space (eigenfaces). Those of lower energy are said to describe the noise and rejected.



- A new face image is represented as a linear combination of these eigenfaces :
  - Every person to be recognized is represented by a set of M transformed coefficients values: a point in the face space.

KLT-based face recognition

4.3

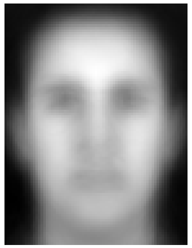
- **Space reduction:** The face image is approximated in the new space.
- The M coefficients represent this face image.

$$\mathbf{m_x} = \mathbf{E}[\mathbf{x}] \qquad \mathbf{R_{\tilde{x}}} = \mathbf{E}\left[\mathbf{\tilde{x}\tilde{x}^T}\right] \qquad \mathbf{V} = \left[\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_M, \cdots \mathbf{v}_N\right]$$

$$\mathbf{\tilde{x}} = \mathbf{x} - \mathbf{m_x} \qquad \mathbf{R_{\tilde{x}} v_i} = \lambda_i \mathbf{v_i} \qquad \mathbf{W} = \left[\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_M\right]$$

System Input

y_0



+

y_1

+

y_2

+ … +

y_M

Representation of the individual after projection in the face space.

$$\mathbf{y} = \mathbf{W^T \tilde{x}}$$

Unit Structure

4.3

1. **Introduction:**
   - Canonical representation and Discrete Fourier Transform (DFT)
2. **Discrete Cosine Transform (DCT):**
   - Basic properties
3. **Karhunen-Loeve Transform (KLT):**
   - Basic properties
4. **Compression application:**
   - DCT: Image compression
5. **Biometry application:**
   - KLT: Face recognition
6. **Summary and Conclusions**