Imperial College London

Department of Electrical and Electronic Engineering

Digital Image Processing

PART I

IMAGE TRANSFORMS

Academic responsible

Dr. Tania STATHAKI, Meng PhD DIC

Associate Professor in Signal Processing

Room 812

Ext. 46229

Email: **t.stathaki@imperial.ac.uk**

http://www.commsp.ee.ic.ac.uk/~tania/

https://www.researchgate.net/profile/Tania_Stathaki

Abstract

Transform theory plays a fundamental role in image processing, as working with the transform of an image instead of the image itself may give us more insight into the properties of the image. Two-dimensional transforms are applied to image enhancement, restoration, encoding and description.

1. UNITARY TRANSFORMS

1.1 One-dimensional signals

For a one-dimensional sequence $\{f(x), 0 \leq x \leq N-1\}$ of size N, represented as a column vector $\underline{f} = [f(0)\ f(1)\ \dots\ f(N-1)]^T$, a transformation may be written as

$$g(u) = \sum_{x=0}^{N-1} T(u,x)f(x)\,, 0 \leq u \leq N-1$$

where $g(u)$ is the transform (or transformation) of $f(x)$, and $T(u,x)$ is the so-called **forward transformation kernel function**.

If we represent the sequence $\{g(u), 0 \leq u \leq N-1\}$ of size N, as a column vector $\underline{g} = [g(0)\ g(1)\ \dots\ g(N-1)]^T$ as well, the transformation may be written in a matrix form as follows

$$\underline{g} = \underline{T} \cdot \underline{f}$$

where the square matrix $\underline{T}$ of size $N \times N$ contains the forward transformation kernel function values.

Similarly, the inverse transform is given by the relationship

$$f(x) = \sum_{u=0}^{N-1} I(x,u)g(u),\ 0 \leq x \leq N-1$$

where $I(x,u)$ is the so-called **inverse transformation kernel function**.

In matrix form the above is written as

$$\underline{f} = \underline{I} \cdot \underline{g} = \underline{T}^{-1} \cdot \underline{g}$$

where the matrix $\underline{I}$ of size $N \times N$ contains the inverse transformation kernel function values.

In order to recover $\underline{f}$ from $\underline{g}$ the matrix $\underline{I} = \underline{T}^{-1}$ is required. If

$$\underline{I} = \underline{T}^{-1} = \underline{T}^{*T}$$

the square matrix $\underline{T}$ is called **unitary**, and the transformation is called unitary as well. It can be proven that the columns (or rows) of an $N \times N$ unitary matrix are **orthonormal** and therefore, form a complete set of **basis vectors** in the $N-$dimensional vector space. In other words, both the column and row vectors of a unitary matrix are orthogonal (perpendicular to each other) and of unit length. If we denote the columns of $\underline{T}$ with $\underline{t_x} = [T(0,x)\ T(1,x)\ \dots T(N-1,x)]$, $0 \leq x \leq N-1$, the following relationship holds:

$$\underline{t_i}^{*T} \cdot \underline{t_j} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

with $0 \leq i,j \leq N-1$.

In the case of a unitary transform, we can recover the original signal as follows:

$$\underline{f} = \underline{T}^{*T} \underline{\cdot\, g} \Rightarrow f(x) = \sum_{u=0}^{N-1} T^{*}(u,x)g(u)$$

The columns of $\underline{T}^{*T}$ are the vectors $\underline{T}_u^* = [T^*(u,0)\ T^*(u,1)\ \ldots T^*(u,N-1)]^T$.

1.2 Two-dimensional signals (images)

As a one-dimensional signal of size N can be represented by a set of N orthonormal basis vectors of size N, an image can also be expanded to a discrete set of **basis arrays** called **basis images** through a **two-dimensional (image) transform**.

For an $M \times N$ image $f(x,y)$ the forward and inverse transforms are given below

$$g(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} T\,(u,v,x,y) f(x,y)$$

$$f(x,y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} I\,(x,y,u,v) g(u,v)$$

where, again, $T(u,v,x,y)$ and $I(x,y,u,v)$ are called the **forward and inverse transformation kernel functions**, respectively.

The forward kernel is said to be **separable** if

$$T(u,v,x,y) = T_1(u,x)T_2(v,y)$$

It is said to be **symmetric** if T_1 is functionally equal to T_2 such that

$$T(u,v,x,y) = T_1(u,x)T_1(v,y)$$

The same comments are valid for the inverse kernel.

For the property of symmetry to be valid the condition $M = N$ must hold.

If the kernel $T(u,v,x,y)$ of an image transform is separable and symmetric, then the transform $g(u,v) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} T\,(u,v,x,y) f(x,y) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} T_1\,(u,x) T_1(v,y) f(x,y)$ can be written in matrix form as follows

$$\underline{g} = \underline{T_1} \cdot \underline{f} \cdot \underline{T_1}^T$$

where $\underline{f}$ is the original image of size $N \times N$, and $\underline{T_1}$ is an $N \times N$ transformation matrix. Basically $\underline{T_1}$ is the transformation matrix of the one-dimensional version of the transform under consideration. If $\underline{T_1}$ is a unitary matrix then the transform is called **separable, symmetric and unitary** and the original image is recovered through the relationship

$$\underline{f} = \underline{T_1}^{*T} \cdot \underline{g} \cdot \underline{T_1}^*$$

1.3 Fundamental properties of unitary transforms

Most of the signal transformations that we use in engineering are unitary and possess the following two very important properties.

1.3.1 The property of energy preservation

For a unitary transformation

$$\underline{g} = \underline{T} \cdot \underline{f}$$

and

$$\underline{g}^{*T} = (\underline{T}^* \cdot \underline{f}^*)^T = \underline{f}^{*T} \cdot \underline{T}^{*T}$$

and therefore, by using the relation $\underline{T}^{-1} = \underline{T}^{*T}$ we have that

$$\underline{g}^{*T} \cdot \underline{g} = \left(\underline{f}^{*T} \cdot \underline{T}^{*T}\right) \cdot \left(\underline{T} \cdot \underline{f}\right) = \underline{f}^{*T} \cdot (\underline{T}^{*T} \cdot \underline{T}) \cdot \underline{f} = \underline{f}^{*T} \cdot \underline{f} \Rightarrow \left\|\underline{g}\right\|^2 = \left\|\underline{f}\right\|^2$$

Thus, a unitary transformation preserves the signal energy. In mathematics we say that a unitary transformation preserves the inner product. This property is called **energy preservation property**.

This means that every unitary transformation is simply a rotation of the vector $\underline{f}$ in the N- dimensional vector space.

For the 2-D case the energy preservation property is written as

$$\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}|f(x,y)|^2 = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1}|g(u,v)|^2$$

1.3.2 The property of energy compaction

Most unitary transforms pack a large fraction of the energy of the image into relatively few of the transform coefficients. This means that relatively few of the transform coefficients have significant values and these are the coefficients that are close to the origin (small index coefficients).

This property is called **energy compaction** and it is very useful for compression purposes. This is because we can only keep a few of the transformed signal values, namely the large values and discard the rest of the values (the small values). We can recover the original signal using the inverse transform by replacing the values we discarded with zeros, without losing a significant amount of signal information.

2. THE TWO-DIMENSIONAL FOURIER TRANSFORM

2.1 Continuous space and continuous frequency

The Fourier Transform (FT) is extended to a function $f(x,y)$ of two variables. If $f(x,y)$ is continuous and integrable and $F(u,v)$ is integrable, the following Fourier Transform pair exists:

$$F(u,v) = \int^{\infty}\int_{-\infty} f(x,y)e^{-j2\pi(ux+vy)}\,dxdy$$

$$f(x,y) = \frac{1}{(2\pi)^2}\int^{\infty}\int_{-\infty} F(u,v)e^{j2\pi(ux+vy)}\,dudv$$

In general, $F(u,v)$ is a complex-valued function of two real frequency variables u,v and hence, it can be written as:

$$F(u,v) = R(u,v) + jI(u,v)$$

The amplitude spectrum, phase spectrum and power spectrum, respectively, are defined as follows.

$$|F(u,v)| = \sqrt{R^2(u,v) + I^2(u,v)}$$

$$\varphi(u,v) = \tan^{-1}\left[\frac{I(u,v)}{R(u,v)}\right]$$

$$P(u,v) = |F(u,v)|^2 = R^2(u,v) + I^2(u,v)$$

At this stage we have assumed that the independent variables are continuous in both space and frequency domain.

2.2 Discrete space and continuous frequency

The Discrete Space Fourier Transform is the member of the 2D Fourier Transform family that operates on aperiodic, 2D discrete images. It is the 2D equivalent of the Discrete Time Fourier Transform

(DTFT). For the case of a discrete image $f(x,y)$, we can define the 2D Discrete Space Fourier Transform pair as follows

$$F(u,v) = \sum_{x}\sum_{y} f(x,y)e^{-j(xu+vy)}$$

$$f(x,y) = \frac{1}{(2\pi)^2}\int_{u=-\pi}^{\pi}\int_{v=-\pi}^{\pi} F(u,v)\, e^{j(xu+vy)}dudv$$

$F(u,v)$ is again a complex-valued function of two real frequency variables u, v and it is periodic with a period $2\pi \times 2\pi$, that is to say $F(u,v) = F(u+2\pi, v) = F(u, v+2\pi)$.

The Fourier Transform of $f(x,y)$ is said to converge uniformly when $F(u,v)$ is finite and

$\lim_{N_1\to\infty}\lim_{N_2\to\infty} \sum_{x=-N_1}^{N_1}\sum_{y=-N_2}^{N_2} f(x,y)e^{-j(xu+vy)} = F(u,v)$ for all u, v.

When the Fourier Transform of $f(x,y)$ converges uniformly, $F(u,v)$ is an analytic function and is infinitely differentiable with respect to u and v.

2.3 Discrete space and discrete frequency: The two-dimensional Discrete Fourier Transform (2-D DFT)

Obviously working with digital computers arise the need of dealing exclusively with discrete signals. If $f(x,y)$ is an $M \times N$ array, such as that obtained by sampling a continuous function of two dimensions at dimensions M and N on a rectangular grid, then its two-dimensional Discrete Fourier Transform (DFT) is the array given by

$$F(u,v) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(ux/M+vy/N)}$$

$u = 0, \dots, M-1$, $v = 0, \dots, N-1$

and the inverse DFT (IDFT) is

$$f(x,y) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)e^{j2\pi(ux/M+vy/N)}$$

When images are sampled in a square array, $M = N$ and

$$F(u,v) = \frac{1}{N}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(ux/M+vy/N)}$$

$$f(x,y) = \frac{1}{N}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)e^{j2\pi(ux/M+vy/N)}$$

It is straightforward to prove that the two-dimensional Discrete Fourier Transform is separable, symmetric and unitary.

2.3.1 Properties of the 2-D DFT

Most of them are straightforward extensions of the properties of the 1-D Fourier Transform. Advise any introductory book on Image Processing.

2.3.2 The importance of the phase in 2-D DFT. Image reconstruction from amplitude or phase only.

The Fourier Transform of a sequence is, in general, complex-valued, and the unique representation of a sequence in the Fourier Transform domain requires both the phase and the magnitude of the Fourier Transform. In various contexts it is often desirable to reconstruct a signal from only partial domain information. Consider a 2-D sequence $f(x,y)$ with Fourier Transform $F(u,v) = \mathcal{F}\{f(x,y)\}$ so that

$$F(u,v) = \mathcal{F}\{f(x,y\} = |F(u,v)|e^{j\varphi_f(u,v)}$$

It has been observed that a straightforward signal synthesis from the Fourier Transform phase $\varphi_f(u,v)$ alone, often captures most of the intelligibility of the original image $f(x,y)$. A straightforward synthesis from the Fourier Transform magnitude $|F(u,v)|$ alone, however, does not generally capture the original signal's intelligibility. To illustrate this, we can synthesise the phase-only signal $f_p(x,y)$ and the magnitude-only signal $f_m(x,y)$ by

$$f_p(x,y) = \mathcal{F}^{-1}\left[1 \cdot e^{j\varphi_f(u,v)}\right]$$

$$f_m(x,y) = \mathcal{F}^{-1}\left[|F(u,v)| \cdot e^{j0}\right]$$

and observe the two results (**Try this exercise in MATLAB**).

An experiment which more dramatically illustrates the observation that phase-only signal synthesis captures more of the signal intelligibility than magnitude-only synthesis, can be performed as follows. Consider two images $f(x,y)$ and $g(x,y)$. From these two images, we synthesise two other images $f(x,y)$ and $g_1(x,y)$ by mixing the amplitudes and phases of the original images as follows:

$$f_1(x,y) = \mathcal{F}^{-1}\left[|G(u,v)|e^{j\varphi_f(u,v)}\right]$$

$$g_1(x,y) = \mathcal{F}^{-1}\left[|F(u,v)|e^{j\varphi_g(u,v)}\right]$$

In this experiment $f(x,y)$ captures the intelligibility of $f(x,y)$, while $g_1(x,y)$ captures the intelligibility of $g(x,y)$ (**Try this exercise in MATLAB**).

---

3. THE DISCRETE COSINE TRANSFORM (DCT)

3.1 One-dimensional signals

A **Discrete Cosine Transform** (**DCT**) expresses a finite sequence of data points in terms of a sum of cosine functions at different frequencies. DCTs are important to numerous applications in science and engineering, from compression of images (e.g. JPEG), video (e.g. MPEG) and audio (e.g. MP3), to spectral methods for the numerical solution of partial differential equations.

There DCT is not uniquely defined. There is a number of variants of it, each one possessing slightly different properties. The DCT we will learn in this course is defined below.

$$C(u) = a(u)\sum_{x=0}^{N-1} f(x)\, cos\left[\frac{(2x+1)u\pi}{2N}\right], u = 0,1,\dots,N-1$$

with $a(u)$ a parameter that is defined below.

$$a(u) = \begin{cases} \sqrt{1/N} & u = 0 \\ \\ \sqrt{2/N} & u = 1,\dots,N-1 \end{cases}$$

The inverse DCT (IDCT) is defined below.

$$f(x) = \sum_{u=0}^{N-1} a(u)C(u)\,cos\left[\frac{(2x+1)u\pi}{2N}\right]$$

The DCT in MATLAB is the orthogonal DCT which, for both forward and inverse transform, has a scale factor of $\sqrt{1/N}$ for the DC term an of $\sqrt{2/N}$ for the other terms. Because this is orthogonal, it preserves the energy of the signal.

3.2 Two-dimensional signals (images)

For 2-D signals it is defined as

$$C(u,v) = a(u)a(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f\,(x,y)\,cos\left[\frac{(2x+1)u\pi}{2N}\right] cos\left[\frac{(2y+1)v\pi}{2N}\right]$$

$$f(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} a(u)a(v)C\,(u,v)\,cos\left[\frac{(2x+1)u\pi}{2N}\right] cos\left[\frac{(2y+1)v\pi}{2N}\right]$$

$a(u)$ is defined as above and $u, v = 0,1, \dots, N-1$

3.3 Properties of the DCT

- The DCT is a real transform. This property makes it attractive in comparison to the Fourier Transform.
- The DCT has excellent energy compaction properties. For that reason, it is widely used in image compression standards (as for example JPEG standards).
- There are fast algorithms to compute the DCT, similar to the FFT algorithm for computing the DFT.

---

4. WALSH TRANSFORM (WT)

4.1 One-dimensional signals

This transform is slightly different from the transforms you have met so far. Suppose we have a function $f(x), x = 0, \dots, N-1$ where $N = 2^n$ and its Walsh Transform $W(u)$.

If we use binary representation for the values of the independent variables x and u, we need n bits to represent them. Hence, for the binary representation of x and u we can write:

$(x)_{10} = (b_{n-1}(x)b_{n-2}(x) \dots b_0(x))_2$, $(u)_{10} = (b_{n-1}(u)b_{n-2}(u) \dots b_0(u))_2$

with $b_i(x)$ 0 or 1 for $i = 0, \dots, n-1$.

Example

If $f(x), x = 0, \dots, 7,$ (8 samples) then $n = 3$ and for $x = 6$, 6=$(110)_2 \Rightarrow b_2(6) = 1, b_1(6) = 1, b_0(6) = 0$

We define now the 1-D Walsh Transform as

$$W(u) = \tfrac{1}{N}\textstyle\sum_{x=0}^{N-1} f(x)\left[\prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)}\right] \text{ or}$$

$$W(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)(-1)^{\sum_{i=0}^{n-1} b_i(x)b_{n-1-i}(u)}$$

The array formed by the Walsh Transform kernels is again a symmetric matrix having orthogonal rows and columns. Therefore, the Walsh Transform kernel elements are of the form $T(u,x) = \prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)}$. You can immediately observe that $T(u,x) = -1$ or 1 depending on the values of $b_i(x)$ and $b_{n-1-i}(u)$. If the Walsh Transform is written in a matrix form

$$\underline{W} = \underline{T} \cdot \underline{f}$$

the rows of the matrix $\underline{T}$ which are the vectors $[T(u,0)\ T(u,1)\ \dots\ T(u,N-1)]$ have the form of square waves.

For $u = 0$ we see that $(0)_{10} = (b_{n-1}(0)b_{n-2}(0) \dots b_0())_2 = (00 \dots 0)_2$ and hence, $b_{n-1-i}(u) = 0$, for any i. Thus, $T(0,x) = 1$ and $W(0) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)$. Therefore, we see that the first element of the Walsh Transform is the mean of the original function $f(x)$ (the DC value) as it is in the case of Fourier Transform.

However, as a general rule, we see that as the variable u (which represents the index of the transform) increases, the corresponding basis function's (a square wave in that case) "frequency" does not necessarily increases as well.

The inverse Walsh Transform is defined as follows.

$$f(x) = \sum_{u=0}^{N-1} W(u)\left[\prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)}\right] \text{ or}$$

$$f(x) = \sum_{u=0}^{N-1} W(u)(-1)^{\sum_{i=0}^{n-1} b_i(x)b_{n-1-i}(u)}$$

4.2 Two-dimensional signals

The Walsh Transform is defined as follows for two-dimensional signals.

$$W(u,v) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f\,(x,y)\left[\prod_{i=0}^{n-1}(-1)^{(b_i(x)b_{n-1-i}(u)+b_i(y)b_{n-1-i}(v))}\right] \text{ or}$$

$$W(u,v) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f\,(x,y)(-1)^{\sum_{i=0}^{n-1}(b_i(x)b_{n-1-i}(u)+b_i(y)b_{n-1-i}(v))}$$

The inverse Walsh Transform is defined as follows for two-dimensional signals.

$$f(x,y) = \frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} W(u,v)\left[\prod_{i=0}^{n-1}(-1)^{(b_i(x)b_{n-1-i}(u)+b_i(y)b_{n-1-i}(v))}\right] \text{ or}$$

$$f(x,y) = \frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} W(u,v)\,(-1)^{\sum_{i=0}^{n-1}(b_i(x)b_{n-1-i}(u)+b_i(y)b_{n-1-i}(v))}$$

4.3 Properties of the Walsh Transform

- Unlike the Fourier Transform, which is based on trigonometric terms, the Walsh Transform consists of a series expansion of basis functions whose values are only -1 or 1 and they have the form of square waves. These functions can be implemented more efficiently in a digital environment than the exponential basis functions of the Fourier Transform.
- The forward and inverse Walsh kernels are identical except for a constant multiplicative factor of $\frac{1}{N}$ for 1-D signals.
- The forward and inverse Walsh kernels are identical for 2-D signals. This is because the array formed by the kernels is a symmetric matrix having orthogonal rows and columns, so its inverse array is the same as the array itself.

- The concept of frequency exists also in Walsh Transform basis functions. We can think of frequency as the number of zero crossings or the number of transitions in a basis vector and we call this number **sequency**.

---

5. HADAMARD TRANSFORM (HT)

5.1 Definition

In a similar form as the Walsh Transform, the 2-D Hadamard Transform is defined as follows.

Forward

$$H(u,v) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f\,(x,y)\left[\prod_{i=0}^{n-1}(-1)^{(b_i(x)b_i(u)+b_i(y)b_i(v))}\right],\ N = 2^n \text{or}$$

$$H(u,v) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f\,(x,y)(-1)^{\sum_{i=0}^{n-1}(b_i(x)b_i(u)+b_i(y)b_i(v))}$$

Inverse

$$f(x,y) = \frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} H\,(u,v)\left[\prod_{i=0}^{n-1}(-1)^{(b_i(x)b_i(u)+b_i(y)b_i(v))}\right] \text{ etc.}$$

5.2 Properties of the Hadamard Transform

- **Most of the comments made for Walsh Transform are valid here.**
- The Hadamard Transform differs from the Walsh Transform only in the <u>order of basis functions</u>.
- An important property of Hadamard Transform is that, letting H_N represent the matrix of order N, the recursive relationship is given by the expression

$$H_{2N} = \begin{bmatrix} H_N & H_N \\ H_N & -H_N \end{bmatrix}$$

Extended versions of the Walsh and Hadamard Transforms are their so-called ordered versions. In these versions the order of basis functions has been rearranged so that the sequency of the basis function increases by its index. Only the ordered versions of the above transforms exhibit the property of energy compaction and therefore, only these versions are used in real life applications.

For the fast computation of the Ordered Walsh or the Ordered Hadamard Transform there exist algorithms called **Fast Walsh Transform (FWT) and Fast Hadamard Transform (FHT),** respectively. These algorithms are straightforward modification of the FFT. Advise any introductory book for your own interest.

---

6. KARHUNEN-LOEVE (KLT) or HOTELLING TRANSFORM

The Karhunen-Loeve Transform or KLT was originally introduced as a series expansion for continuous random processes by Karhunen and Loeve. For discrete signals Hotelling first studied what was called a method of principal components, which is the discrete equivalent of the KL series expansion.

Consequently, the KL Transform is also called the Hotelling Transform or the method of principal components. The term KLT is the most widely used.

6.1 The case of many realisations of a signal or image (Gonzalez/Woods)

The concepts of **eigenvalue** and **eigevector** are necessary to understand the KL transform.

If $\underline{C}$ is a matrix of dimension $n \times n$, then a scalar λ is called an eigenvalue of $\underline{C}$ if there is a nonzero column vector $\underline{e}$ in R^n such that

$$\underline{C} \cdot \underline{e} = \lambda \underline{e}$$

The vector $\underline{e}$ is called an eigenvector of the matrix $\underline{C}$ corresponding to the eigenvalue λ.

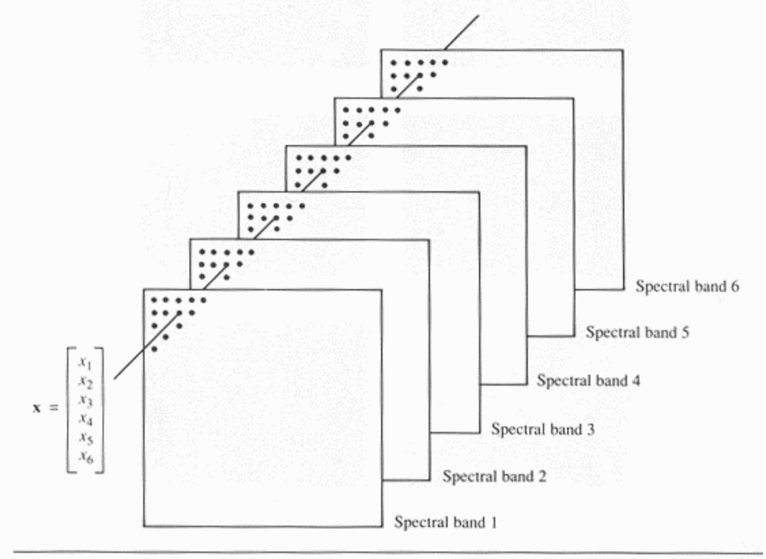
Consider a population of random column vectors of the form

$$\underline{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

The quantity x_i may represent the value (grey level) of an image i. We have n images, all of equal size $M \times N$. Each of the above vectors refers to the exact same location across the n images (please look at the next figure).
Therefore, it is more accurate to write

$$\underline{x}_{(k,l)} = \begin{bmatrix} x_1(k,l) \\ x_2(k,l) \\ \vdots \\ x_n(k,l) \end{bmatrix}$$

with $k \in [0 \ldots M-1]$ and $l \in [0 \ldots N-1]$.



Formation of a vector from corresponding pixels in six images.

The mean vectors of the population are defined as:

$$\underline{m}_{\underline{x}_{(k,l)}} = E\{\underline{x}_{(k,l)}\} = [m_{1,(k,l)} \quad m_{2,(k,l)} \quad \cdots \quad m_{n,(k,l)}]^T$$

As you can see, we assume that the mean of each pixel (k,l) in each image i is different.

In that case we would require a large number of realizations of each image i in order to calculate the means $m_{i,(k,l)}$. However, if we assume that each image signal is **ergodic** we can calculate a single mean value for all pixels from a single realization using the entire collection of pixels of this particular image. (Note that in signal processing, a **stochastic process** is said to be **ergodic** if its statistical properties can be deduced from a single, sufficiently long, random sample of the process. The reasoning is that any collection of random samples from a process must represent the average statistical properties of the entire process. In other words, regardless of what the individual samples are, a birds-eye view of the collection of samples must represent the whole process. Conversely, a process that is not ergodic is a process that changes erratically at an inconsistent rate.

In that case:

$$m_{i,(k,l)} = m_i = \frac{1}{MN}\sum_{k=0}^{M-1}\sum_{l=0}^{N-1} x_{i,(k,l)}$$

and

$$\underline{m}_{\underline{x}(k,l)} = \underline{m}_{\underline{x}} = [m_1 \quad m_2 \quad \dots \quad m_n]^T = [E\{x_1\} \quad E\{x_2\} \quad \dots \quad E\{x_n\}]^T = E\{\underline{x}\}$$

The operator E refers to the **expected value** of the population, calculated theoretically using the probability density functions (pdf) of the elements x_i.

The **covariance matrix** of the population is defined as

$$\underline{C}_{\underline{x}} = E\{(\underline{x} - \underline{m}_{\underline{x}})(\underline{x} - \underline{m}_{\underline{x}})^T\}$$

The operator E is now calculated theoretically using the probability density functions (pdf) of the elements x_i and the joint probability density functions between the elements x_i and x_j.

Because $\underline{x}$ is n -dimensional $(\underline{x} - \underline{m}_{\underline{x}})(\underline{x} - \underline{m}_{\underline{x}})^T$ and $\underline{C}_x$ are matrices of order $n \times n$. The diagonal element c_{ii} of matrix $\underline{C}_{\underline{x}}$ is the variance of x_i, and the element c_{ij} of $\underline{C}_{\underline{x}}$ is the covariance between the elements x_i and x_j. If the elements x_i and x_j are uncorrelated, their covariance is zero and, therefore, $c_{ij} = c_{ji} = 0$. The covariance matrix $\underline{C}_{\underline{x}}$ can be written as follows.

$$\underline{C}_{\underline{x}} = E\{(\underline{x} - \underline{m}_{\underline{x}})(\underline{x} - \underline{m}_{\underline{x}})^T\} = E\{(\underline{x} - \underline{m}_{\underline{x}})(\underline{x}^T - \underline{m}_{\underline{x}}^T)\} = E\{\underline{x}\underline{x}^T - \underline{x}\underline{m}_{\underline{x}}^T - \underline{m}_{\underline{x}}\underline{x}^T + \underline{m}_{\underline{x}}\underline{m}_{\underline{x}}^T\}$$

It can be easily shown that

$$E\{\underline{x}\underline{m}_{\underline{x}}^T\} = E\{\underline{m}_{\underline{x}}\underline{x}^T\}$$

Therefore,

$$E\{\underline{x}\underline{x}^T - \underline{x}\underline{m}_{\underline{x}}^T - \underline{m}_{\underline{x}}\underline{x}^T + \underline{m}_{\underline{x}}\underline{m}_{\underline{x}}^T\} = E\{\underline{x}\underline{x}^T - 2\underline{m}_{\underline{x}}\underline{x}^T + \underline{m}_{\underline{x}}\underline{m}_{\underline{x}}^T\}$$
$$= E\{\underline{x}\underline{x}^T\} - E\{2\underline{m}_{\underline{x}}\underline{x}^T\} + E\{\underline{m}_{\underline{x}}\underline{m}_{\underline{x}}^T\}$$

Since the vector $\underline{m}_{\underline{x}}$ and the matrix $\underline{m}_{\underline{x}}\underline{m}_{\underline{x}}^T$ contain constant quantities, we can write

$$E\{\underline{x}\underline{x}^T - 2\underline{m}_{\underline{x}}\underline{x}^T + \underline{m}_{\underline{x}}\underline{m}_{\underline{x}}^T\} = E\{\underline{x}\underline{x}^T\} - 2\underline{m}_{\underline{x}}E\{\underline{x}^T\} + \underline{m}_{\underline{x}}\underline{m}_{\underline{x}}^T$$

Knowing that

$$E\{\underline{x}^T\} = \underline{m}_{\underline{x}}^T$$

we have

$$\underline{C}_{\underline{x}} = E\{\underline{x}\underline{x}^T\} - 2\underline{m}_{\underline{x}}E\{\underline{x}^T\} + \underline{m}_{\underline{x}}\underline{m}_{\underline{x}}^T = E\{\underline{x}\underline{x}^T\} - 2\underline{m}_{\underline{x}}\underline{m}_{\underline{x}}^T + \underline{m}_{\underline{x}}\underline{m}_{\underline{x}}^T \Rightarrow$$
$$\underline{C}_{\underline{x}} = E\{\underline{x}\underline{x}^T\} - \underline{m}_{\underline{x}}\underline{m}_{\underline{x}}^T$$

In most real-life applications, the probability density functions (pdf) of the elements x_i and the joint probability density functions between the elements x_i and x_j are not known.

For M vectors from a random population, where M is large enough, the mean vector $\underline{m}_{\underline{x}}$ and the covariance matrix $\underline{C}_{\underline{x}}$ can be approximately calculated from the available vectors by using the following relationships where all the expected values are approximated by summations

$$\underline{m_x} = E\{\underline{x}\} \cong \frac{1}{M}\sum_{k=1}^{M} \underline{x_k}$$

$$\underline{C_x} = E\{\underline{x}\underline{x}^T\} - \underline{m_x}\underline{m_x^T} \cong \frac{1}{M}\sum_{k=1}^{M} \underline{x_k}\,\underline{x_k^T} - \underline{m_x}\underline{m_x^T}$$

Very easily it can be seen that, for real images, $\underline{C_x}$ is real and symmetric. Let $\underline{e_i}$ and λ_i, $i = 1,2,\dots,n$, be a set of orthonormal eigenvectors and corresponding eigenvalues of $\underline{C_x}$, arranged in descending order so that $\lambda_i \geq \lambda_{i+1}$ for $i = 1,2,\dots,n-1$. Suppose that $\underline{e_i}$ are column vectors.

(Note that real and symmetric matrices of dimension $n \times n$, always have a set of n orthonormal eigenvectors.)

Let $\underline{A}$ be a matrix whose rows are formed from the eigenvectors of $\underline{C_x}$, ordered so that the first row of $\underline{A}$ is the eigenvector corresponding to the largest eigenvalue, and the last row the eigenvector corresponding to the smallest eigenvalue. Therefore,

$$\underline{A} = \begin{bmatrix} \underline{e_1^T} \\ \underline{e_2^T} \\ \vdots \\ \underline{e_n^T} \end{bmatrix} \text{ and } \underline{A}^T = [\underline{e_1} \quad \underline{e_2} \quad \cdots \quad \underline{e_n}]$$

Suppose that $\underline{A}$ is a transformation matrix that maps the vectors $\underline{x}$ into vectors $\underline{y}$ by using the following transformation:

$$\underline{y} = \underline{A}(\underline{x} - \underline{m_x})$$

The above transform is called the **Karhunen-Loeve** or **Hotelling** transform. The mean of the $\underline{y}$ vectors resulting from the above transformation is zero, since

$$E\{\underline{y}\} = E\{\underline{A}(\underline{x} - \underline{m_x})\} = \underline{A}E\{\underline{x} - \underline{m_x}\} = \underline{A}(E\{\underline{x}\} - \underline{m_x}) = \underline{A}(\underline{m_x} - \underline{m_x}) = \underline{0} \Rightarrow$$

$$\underline{m_y} = \underline{0}$$

The covariance matrix of the $\underline{y}$ vectors is

$$\underline{C_y} = E\{(\underline{y} - \underline{m_y})(\underline{y} - \underline{m_y})^T\} = E\{\underline{y}\underline{y}^T\}$$

Using the relationships

$$\underline{y} = \underline{A}(\underline{x} - \underline{m_x})$$

$$\underline{y}^T = [\underline{A}(\underline{x} - \underline{m_x})]^T = (\underline{x} - \underline{m_x})^T\underline{A}^T$$

we get

$$\underline{y}\underline{y}^T = \underline{A}(\underline{x} - \underline{m_x})(\underline{x} - \underline{m_x})^T\underline{A}^T \Rightarrow E\{\underline{y}\underline{y}^T\} = E\{\underline{A}(\underline{x} - \underline{m_x})(\underline{x} - \underline{m_x})^T\underline{A}^T\}$$

$$= \underline{A}E\{(\underline{x} - \underline{m_x})(\underline{x} - \underline{m_x})^T\}\underline{A}^T$$

$$\underline{C_y} = \underline{A}\,\underline{C_x}\underline{A}^T$$

$$\underline{C_x}\underline{A}^T = \underline{C_x}[\underline{e_1} \quad \underline{e_2} \quad \cdots \quad \underline{e_n}] = [\lambda_1\underline{e_1} \quad \lambda_2\underline{e_2} \quad \cdots \quad \lambda_n\underline{e_n}]$$

$$\underline{C_y} = \underline{A}\underline{C_x}\underline{A}^T = \begin{bmatrix} \underline{e_1^T} \\ \underline{e_2^T} \\ \vdots \\ \underline{e_n^T} \end{bmatrix} [\lambda_1\underline{e_1} \quad \lambda_2\underline{e_2} \quad \dots \quad \lambda_n\underline{e_n}]$$

Because $\underline{e_i}$ is a set of orthonormal eigenvectors we have that:

$$\underline{e_i^T}\,\underline{e_i} = 1, i = 1,\dots,n$$

$$\underline{e_i^T}\,\underline{e_j} = 0, i,j = 1,\dots,n \text{ and } i \neq j$$

and therefore, $\underline{C_y}$ is a diagonal matrix whose elements along the diagonal are the eigenvalues of $\underline{C_x}$

$$\underline{C}_{\underline{y}} = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_n \end{bmatrix}$$

The off-diagonal elements of the covariance matrix of the population of vectors $\underline{y}$ are 0, and therefore, the elements of the $\underline{y}$ vectors are uncorrelated.

Let's try to reconstruct any of the original vectors $\underline{x}$ from its corresponding $\underline{y}$. Because the rows of $\underline{A}$ are orthonormal vectors we have

$$\underline{A}\underline{A}^T = \begin{bmatrix} \underline{e}_1^T \\ \underline{e}_2^T \\ \vdots \\ \underline{e}_n^T \end{bmatrix} [\underline{e}_1 \quad \underline{e}_2 \quad \cdots \quad \underline{e}_n] = I$$

with I the unity matrix. Therefore, $\underline{A}^{-1} = \underline{A}^T$, and any vector $\underline{x}$ can by recovered from its corresponding vector $\underline{y}$ by using the relation

$$\underline{x} = \underline{A}^T \underline{y} + \underline{m}_{\underline{x}}$$

Suppose that instead of using all the eigenvectors of $\underline{C}_{\underline{x}}$ we form matrix $\underline{A}_K$ from the K eigenvectors corresponding to the K largest eigenvalues,

$$\underline{A}_K = \begin{bmatrix} \underline{e}_1^T \\ \underline{e}_2^T \\ \vdots \\ \underline{e}_K^T \end{bmatrix}$$

yielding a transformation matrix of order $K \times n$. The $\underline{y}$ vectors would then be K dimensional, and the reconstruction of any of the original vectors would be approximated by the following relationship

$$\underline{\hat{x}} = \underline{A}_K^T \underline{y} + \underline{m}_{\underline{x}}$$

The mean square error between the perfect reconstruction $\underline{x}$ and the approximate reconstruction $\underline{\hat{x}}$ is given by the expression

$$e_{ms} = \sum_{j=1}^{n} \lambda_j - \sum_{j=1}^{K} \lambda_j = \sum_{j=K+1}^{n} \lambda_j .$$

By using $\underline{A}_K$ instead of $\underline{A}$ for the KL transform we achieve compression of the available data.

6.2 Properties of the Karhunen-Loeve Transform

Despite its favourable theoretical properties, the KLT is not used often in practice for the following reasons.

- Its basis functions depend on the covariance matrix of the image, and hence they have to recomputed and transmitted for every image.
- Perfect decorrelation is not possible, since images can rarely be modelled as realisations of ergodic fields.
- There aren't any computationally fast algorithms for its implementation.

REFERENCES

1. Digital Image Processing by R. C. Gonzales and R. E. Woods, Addison-Wesley Publishing Company, 1992.
2. Two-Dimensional Signal and Image Processing by J. S. Lim, Prentice Hall, 1990.
3. Porat, B. (1994). Digital Processing of Random Signals: Theory & Methods. Prentice Hall. p. 14.
4. Papoulis, Athanasios (1991). Probability, random variables, and stochastic processes. New York: McGraw-Hill. pp. 427–442.