Volume 91 Issue 11 November 2011 ISSN 0165-1684
ELSEVIER
SIGNAL PROCESSING
An International Journal
A publication of the European Association for Signal Processing (EURASIP)
EURASIP

Signal Processing 91 (2011) 2443–2464

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Review

A tutorial overview on the properties of the discrete cosine transform for encoded image and video processing

Nuno Roma [*], Leonel Sousa

IST/TU Lisbon/INESC-ID, Rua Alves Redol, 9, 1000-029 Lisboa, Portugal

A R T I C L E I N F O

Article history:
Received 24 August 2010
Received in revised form
18 February 2011
Accepted 11 April 2011
Available online 17 April 2011

Keywords:
Discrete cosine transform
Image and video processing
Multiplication-convolution property

A B S T R A C T

Discrete trigonometric transforms, such as the discrete cosine transform (DCT) and the discrete sine transform (DST), have been extensively used in signal processing for transform-based coding. The even type-II DCT, used in image and video coding, became specially popular to decorrelate the pixel data and minimize the spatial redundancy. Albeit this DCT tends to be the most often used, it integrates a broader family of transforms composed of eight DCTs and eight DSTs. However, even though most applications require little knowledge more than the actual DCT definition and its inverse, it is often widely regarded that the implementation of more complex operations on transformed data sequences (transcoding) requires a more in-depth knowledge about its precise definitions and formal mathematical properties. One of such relations is the multiplication-convolution property, often required to implement more specific and complex manipulations. Considering that such information is still spread into several documents and manuscripts, the main purpose of this article is to provide a broad set of practical and useful information in a single and self-contained source, embracing a wide range of definitions and properties related to the DCT and DST families, with a special emphasis on its application to image and video processing.

Contents

* Corresponding author.
E-mail addresses: Nuno.Roma@inesc-id.pt (N. Roma), Leonel.Sousa@inesc-id.pt (L. Sousa).

0165-1684/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.sigpro.2011.04.015

1. Introduction

Transform-based coding has been extensively used in image and video coding. The adoption of transform functions to encode pixel data relies on the general premise that adjacent pixels exhibit a significant level of spatial correlation, which can be highly exploited to predict the value of a given pixel from its corresponding neighbors. As a consequence, most digital image and video coding methods that have been proposed take advantage of these transform functions to map the spatial correlated data into a set of less correlated transform-domain coefficients.

As an illustrative example, in Fig. 1 it is depicted the general structure of a digital image encoding and decoding system. The main objective of the *source encoder* is to exploit the spatial redundancies and the irrelevancies of the pixel data, in order to obtain the highest possible compression level. Such objective is attained by reducing the contents entropy, thus decreasing the average number of bits required to represent each image.

To maximize the compression, each component of this particular encoder exploits a different redundancy level of the pixel data:

- *Transform* module—decorrelates the pixel data in order to minimize the spatial redundancy between adjacent pixels. Several different transforms can be adopted in the implementation of this module; a thorough and detailed discussion about the transform functions that are usually used by such module will be provided in the following sections.
- *Quantizer* module—takes advantage of the inability of the human visual system (HVS) to perceive small differences between pixel values. Accordingly, such small differences are often regarded as irrelevant and can actually be discarded without introducing serious visual artifacts. Such irrelevancy is often denoted by psychovisual redundancy. This idea is often extended and exploited in low bit rate transmission systems, characterized by strict bandwidth restrictions, where visual quality has to be sacrificed in order to reduce the bit rate to the available bandwidth. Hence, even though other sources of degradation may also have to be considered, this module is usually the main source of irreversible degradation in lossy encoding schemes.
- *Entropy encoder*—fulfills a final lossless compression stage in the encoding scheme. Among the several possible alternatives, Huffman and arithmetic run-length encoders have been extensively adopted [1–4]. The main purpose of this module is to represent each symbol of the quantizer output with the minimum amount of bits as possible.

On the other hand, and contrasting to the described image encoding structure, typical digital video transmission systems also incorporate a temporal prediction loop, in order to decorrelate the pixel data and minimize the temporal redundancies that exist between consecutive frames of the video sequence (see Fig. 2). To implement such prediction, an additional component needs to be implemented in order to exploit this additional redundancy level of the pixel data:

- *Motion compensation* module—compensates for the displacement of moving objects, from one frame to another, in order to minimize the magnitude of the resulting difference signal. The implementation of this module in the encoder side is tightly coupled to a *motion estimation* module, which computes the motion vectors that describe the relative displacement of a given block of pixels.

Finally, to enhance the reliability of the transmission, the *channel encoder* may add a certain redundancy level to the output bit stream of the source encoder [5], in order to allow the receiver to recover from eventual transmission errors.

Independently of the target application, most transform-based coding methods process the pixel data by

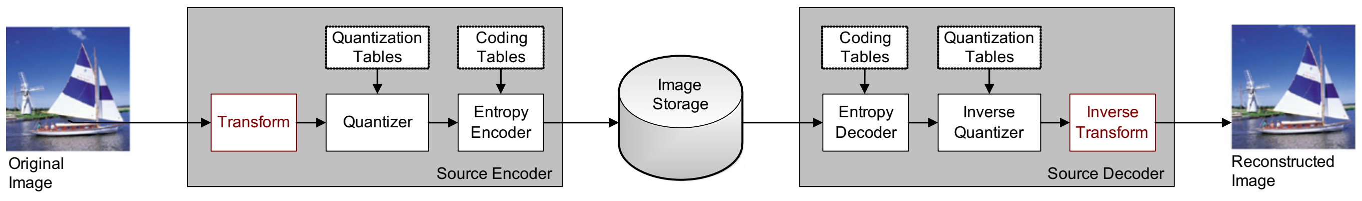


Fig. 1. Block diagram of a general image encoding and decoding system.

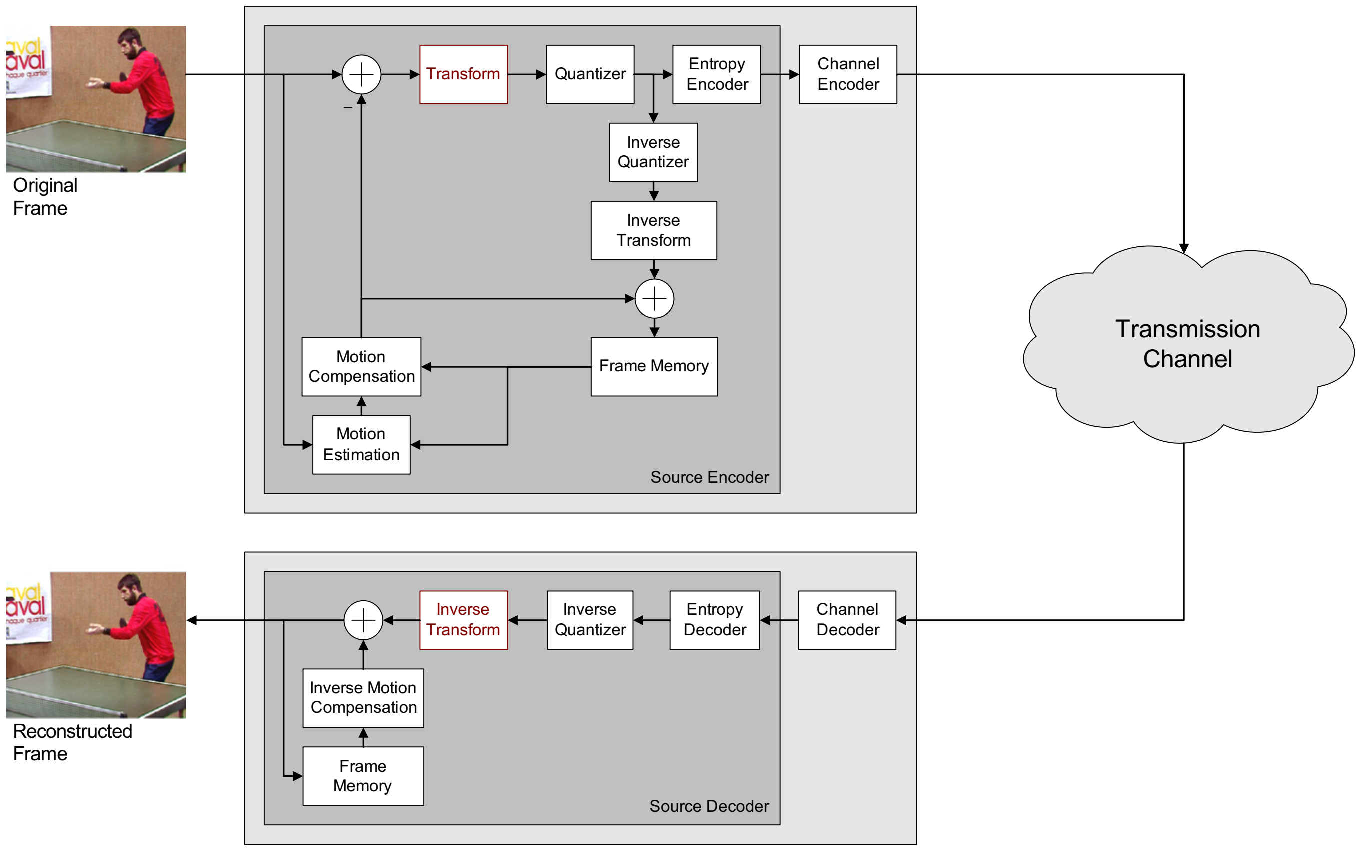


Fig. 2. Block diagram of a general video encoding and transmission system.

grouping the pixels in block structures. These blocks are then transformed and mapped into the frequency domain. By defining two $(N\times N)$ transformation matrices:

$$\mathcal{T}_c=\{\mathcal{T}_c(m,i)\},\quad m,i=0,1,\ldots,N-1 \tag{1}$$

$$\mathcal{T}_r=\{\mathcal{T}_r(n,j)\},\quad n,j=0,1,\ldots,N-1 \tag{2}$$

and by considering a two-dimensional (2-D) image block $\mathbf{x}$, defined in the spatial (pixel) domain, the corresponding transform-domain representation $\mathbf{X}$ is obtained by applying the $(N\times N)$ linear transformation process:

$$\mathbf{X}=\mathcal{T}_c\,\mathbf{x}\,\mathcal{T}_r^{\mathrm{T}} \tag{3}$$

The $\mathcal{T}_c$ and $\mathcal{T}_r$ matrices are usually referred to as the *transformation kernels* or *basis functions*.

As it was previously referred, the main motivation for applying this exchange of the signal domain and compute the transform representation $\mathbf{X}$ of the pixel-domain signal $\mathbf{x}$ is to obtain a more compact representation of the pixel data. Nevertheless, not only should such transformation process present a lossless nature, but also it has to be reversible, so that $\mathbf{x}$ can be reconstructed from $\mathbf{X}$ in the decoder side of the video/image transmission system.

Some of the most commonly used transforms that have been considered for image and video coding are the Karhunen–Loève transform (KLT), the discrete Fourier transform (DFT), the discrete cosine transform (DCT), the discrete sine transform (DST), and the discrete wavelet transform (DWT) [6–10].

The KLT is the most efficient, in terms of compaction efficiency. The basis functions of this transform are obtained from the statistical properties of the pixel data. As a consequence, it gives rise to the optimum performance in terms of energy compaction, thus placing as much energy as possible in as few coefficients as possible. However, since the transform kernel of this transform depends on the image data under processing, it cannot be computed using a fast matrix-multiplication form, based on a separable and pre-computed matrix kernel. It also requires a continuous update of the transform kernel coefficients in the decoder, thus implying a subsequent decrease of the compression rate. Moreover, for block-based coding, the derivation of the basis kernel corresponding to each image block introduces an extra computational effort, which is an important issue in most current image and video coding applications.

As a consequence, other less efficient but image independent transforms have been preferred. Among them, the DFT is characterized by a linear, separable and symmetric definition. Contrary to the KLT it is represented by fixed basis functions and also exhibits good decorrelation and energy compaction characteristics. However, the DFT is defined in the complex-domain and therefore gives rise to both magnitude and phase components for each sample. Furthermore, the implicit periodicity of DFT introduces boundary discontinuities that result in a significant high-frequency content [9].

As a result, discrete transforms characterized by smoother basis functions have been preferred. In particular, the output provided by the DCT usually leads to compaction efficiency levels quite close to the optimum performance provided by the KLT. As a consequence, after it was firstly proposed by Ahmed et al. [11] and latter further described by Wang [12,13], the DCT has been widely adopted in many digital image and video standards, such as the JPEG [14], H.261 [1], H.263 [2], MPEG-1 Video [3], MPEG-2 Video [4] and MPEG-4 Visual [15].

Independently of the adopted standard, the computation of the transform coefficients usually implies the usage of floating-point accuracy. Nevertheless, in many practical implementations, the floating-point DCT and its inverse are usually evaluated with finite precision. This often leads to an accuracy mismatch in the computation of these transforms at both the encoder and decoder ends of the transmission system. These mismatch errors tend to accumulate and often result in a non-negligible distortion component with visible artifacts.

To circumvent this problem, the most recent video standards have adopted alternative transforms. These transforms can be accurately implemented with reduced precision, at the cost of a slight decrease of the provided decorrelation performance. Some examples of these transforms are the Walsh–Hadamard transform (WHT) [16], the slant transform (ST) [17] and the Haar transform (HT) [18]. Another example of such transforms is the Integer Discrete Cosine Transform (IntDCT) [19], that was adopted by the H.264/AVC video standard [20]. It is defined as

$$\mathbf{X}_\mathrm{I} = \mathbf{T}_\mathrm{I}\ \mathbf{x}\ \mathbf{T}_\mathrm{I}^\mathrm{T} \tag{4}$$

where the transform kernel matrix ($\mathbf{T}_\mathrm{I}$) is

$$\mathbf{T}_\mathrm{I} = \begin{pmatrix} 1/2 \\ 1/\sqrt{10} \\ 1/2 \\ 1/\sqrt{10} \end{pmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \tag{5}$$

Since the scaling factors that are associated to this kernel can be absorbed in the quantization process, all arithmetic operations of this transform can be accurately computed with 16-bit integers and using solely additions and shift operations; there is no need to perform any multiplication.

However, despite the orthogonal nature and the computational simplicity that are offered by these transforms, many important mathematical relations already presented for the DCT have not yet been extended to these transforms. One such relation is the *multiplication-convolution property*, whose formal definition, in the DCT-domain, will be presented in Section 5.2. Such limitations restrict the application of some signal processing operations (e.g., transform-domain transcoding [21]) to DCT-based video encoding systems, namely, those based on JPEG [14], H.261 [1], H.263 [2], MPEG-1 Video [3], MPEG-2 Video [4] and MPEG-4 Visual [15] standards. Moreover, the extensive collection of encoded image and video data that is currently stored all over the world will keep the DCT as one of the most important and used transforms, for a significant amount of time. As a consequence, the general designation of image and video standards which henceforward will be extensively adopted in this tutorial, will only accommodate this broad family of DCT-based standards and excludes those based on integer transforms, such as the H.26L [22] and H.264/AVC [20].

In the remaining of this document, it is presented a tutorial description of the discrete cosine transform family. After this introductory section it will be presented, in Section 2, the formal definition and main properties of generic discrete trigonometric transforms (DTTs) and their extensions to multidimensional spaces (Section 3). Section 4 particularizes the previous definitions and presentations to the special case of the even type-II DCT, which has been widely adopted in image and video standards. The definition of the multiplication-convolution property, applied to this specific transform, will be formally presented in Section 5, as well as some applications of this operation conducted in the pixel and transform domains (Section 6). Finally, Section 7 will conclude this tutorial.

From now on, the subscripts i and j will denote coordinates in the spatial (pixel) domain, whereas the subscripts m and n will denote coordinates in the transform (frequency) domain. Likewise, lowercase symbols will denote pixel-domain signal values, whereas uppercase symbols will denote transform-domain values, e.g., $X(m,n) = \mathrm{DCT}(x(i,j))$.

2. Definition

Similarly to what happens with other Fourier-related transforms, the so-called discrete trigonometric transforms (DTTs), such as the DCT and the DST, represent a function or a signal as a sum of trigonometric terms (cosine or sine), with different frequencies and amplitudes. Just like the DFT, the DCT and the DST also operate with a finite number of discrete data samples of a given function. Nevertheless, while the DCT only makes use of cosine functions, the DFT uses both cosines and sines (in the form of complex exponentials) to represent each signal [9]. However, this difference is a direct consequence of a more important characteristic of these transforms. As it will be seen in the following subsections, the DCT and the DST imply different boundary conditions on the sample data than the DFT or other related transforms.

To simplify this description, the presentation that follows will be focused on one-dimensional (1-D) data sequences. Nevertheless, the same definitions can equally be extended to 2-D signals, without any loss of generalization.

2.1. Extension symmetry properties of sampled data beyond original boundaries

Just like any other Fourier-related transform that operates on a given function $f(n)$ over a finite discrete domain, the DFT, the DCT or the DST can be thought of as implicitly defining an infinite extension of that function outside the original domain. Such implicit extension, defined as a sum of trigonometric functions, will then allow the evaluation of that same function at any arbitrary point n, even for points where the original function $f(n)$ was not defined. Nevertheless, while the DFT implies a periodic extension of the original function, the extension properties that are implicit in the DCT and DST imply quite distinct characteristics that provide particularly useful applications in image and video processing.

Since the DCT and the DST operate on finite and discrete sequences, two issues arise concerning the symmetry properties of those extensions that are obtained from the input samples, which do not arise for the continuous cosine transform. Firstly, each boundary of the input data set can be extended *symmetrically* (also known as *even* extension) or *anti-symmetrically* (also known as *odd* extension). Secondly, the symmetry or anti-symmetry point of such extension must be specified. As an example, by considering a symmetric (even) extension of the left boundary of a simple data sequence composed of four equally spaced sampled points *abcd*, two distinct possible solutions arise in terms of the symmetry point: either the data is symmetrically extended about sample a, in which case the even extension is *dcbabcd*; or the data is evenly extended about a hypothetical point, halfway between a and the previous point, in which case the symmetric extension is *dcbaabcd* (a is repeated).

By adopting these different extension setups, infinite sequences can easily be obtained by simply extending the input data samples of a given finite signal. Such infinite extensions are classified according to the types of symmetry that are adopted at each boundary of the original signal. The four possible extension setups are illustrated in Fig. 3 and can be enumerated as follows [23,24]:

- whole-sample symmetry (WS),
- whole-sample anti-symmetry (WA),
- half-sample symmetry (HS) and
- half-sample anti-symmetry (HA),

where the designations *whole-sample* and *half-sample* refer to the position of the point of symmetry; either coincident with one of the original samples or at a theoretical halfway between two samples.

By following this simple procedure, a given finite sequence $f(n)$ can be easily converted into an infinite sequence by symmetrically extending each point of symmetry (POS) using any of the above four possible setups and by continuing that extension indefinitely, in order to obtain a symmetric-periodic sequence (SPS), according to the following rules [23,24]:

$$\mathrm{HSHS}(x_1,\ldots,x_n)=\mathrm{P}(x_1,\ldots,x_n,x_n,\ldots,x_2) \tag{6}$$

$$\mathrm{HAHA}(x_1,\ldots,x_n)=\mathrm{P}(x_1,\ldots,x_n,-x_n,\ldots,-x_2) \tag{7}$$

$$\mathrm{WSWS}(x_1,\ldots,x_n)=\mathrm{P}(x_1,\ldots,x_{n-1},x_n,x_{n-1},\ldots,x_2) \tag{8}$$

$$\mathrm{WAWA}(x_1,\ldots,x_n)=\mathrm{P}(0,x_2,\ldots,x_{n-1},0,-x_{n-1},\ldots,-x_2) \tag{9}$$

where $\mathrm{P}(\varphi)$ denotes the periodic replication of sequence φ. The POSs may be either all of the same type or of two different types. Whenever the adopted types are different, they alternate along the length of the SPS. The obtained extensions are then usually denominated by concatenating the mnemonics of the symmetry types that are used at each of its ends (e.g., WSWS, HAHA, WAHS, etc.).

Independently of the adopted setup, two POS are always associated with the base period: a left point of symmetry (LPOS) and a right point of symmetry (RPOS); between them lie the representative samples. At each POS, one of the four defined types of symmetry is implemented: WS, WA, HS or HA.

Four possible types of extension at each of the two endpoints leads to a total of 16 distinct SPSs. All these different setups characterize a broad set of standard variants of discrete cosine and sine transforms. For each of these two transforms, each of the two data set boundaries can be either symmetrically or anti-symmetrically extended (two possibilities per boundary) and can be extended about a data point or a point halfway between two sample points (two choices per boundary), thus giving rise to a total of $2\times2\times2\times2=16$ different possibilities. Half of these setups, corresponding to those where the left boundary is symmetrically extended, correspond to the eight different types of DCTs, while the remaining half comprises the eight different types of DSTs. In Table 1, it is presented a comprehensive description of the several properties that are implicit to both the input and output extensions of the considered discrete cosine (C) and sine (S) transforms [23,24].

At this point, it is worth recalling that any discontinuity of the considered function potentially reduces the rate of convergence of its Fourier series, so that more sine or cosine terms are needed to represent it with a given accuracy level. As a consequence, these different boundary conditions lead to different but useful properties that distinguish the several DCT and DST variants. In fact, these different characteristics significantly influence the actual usefulness of each particular DTT for signal compression: the smoother an extension is, the fewer terms

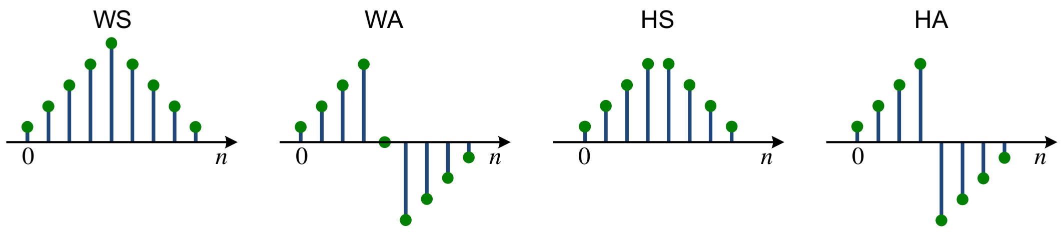


Fig. 3. Symmetric-periodic extensions of a finite sequence [23].

Table 1
Properties of the implicit input and output extensions of the considered discrete sine and cosine transforms [10].

Transform	Input extension properties					$\Longrightarrow$ Output extension properties					
	SPS	Length	Index Range	LPOS	RPOS	SPS	Length	Index Range	LPOS	RPOS	
C_{1e}	WSWS	N	$0 \to N-1$	0	$N-1$	WSWS	N	$0 \to N-1$	0	$N-1$	${C_{1e}}^{-1}$
C_{2e}	HSHS	N	$0 \to N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	WSWA	N	$0 \to N-1$	0	N^*	${C_{3e}}^{-1}$
C_{3e}	WSWA	N	$0 \to N-1$	0	N^*	HSHS	N	$0 \to N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	$C_{2e}-1$
C_{4e}	HSHA	N	$0 \to N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	HSHA	N	$0 \to N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	${C_{4e}}^{-1}$
C_{1o}	WSHS	N	$0 \to N-1$	0	$N-\frac{1}{2}$	WSHS	N	$0 \to N-1$	0	$N-\frac{1}{2}$	${C_{1o}}^{-1}$
C_{2o}	HSWS	N	$0 \to N-1$	$-\frac{1}{2}$	$N-1$	WSHA	N	$0 \to N-1$	0	$N-\frac{1}{2}$	${C_{3o}}^{-1}$
C_{3o}	WSHA	N	$0 \to N-1$	0	$N-\frac{1}{2}$	HSWS	N	$0 \to N-1$	$-\frac{1}{2}$	$N-1$	${C_{2o}}^{-1}$
C_{4o}	HSWA	N	$0 \to N-1$	$-\frac{1}{2}$	N^*	HSWA	N	$0 \to N-1$	$-\frac{1}{2}$	N^*	${C_{4o}}^{-1}$
S_{1e}	WAWA	$N-1$	$1 \to N-1$	0^*	N^*	WAWA	$N-1$	$1 \to N-1$	0^*	N^*	${S_{1e}}^{-1}$
S_{2e}	HAHA	N	$0 \to N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	WAWS	N	$0 \to N-1$	-1^*	$N-1$	${S_{3e}}^{-1}$
S_{3e}	WAWS	N	$0 \to N-1$	-1^*	$N-1$	HAHA	N	$0 \to N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	${S_{2e}}^{-1}$
S_{4e}	HAHS	N	$0 \to N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	HAHS	N	$0 \to N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	${S_{4e}}^{-1}$
S_{1o}	WAHA	$N-1$	$1 \to N-1$	0^*	$N-\frac{1}{2}$	WAHA	$N-1$	$1 \to N-1$	0^*	$N-\frac{1}{2}$	${S_{1o}}^{-1}$
S_{2o}	HAWA	$N-1$	$0 \to N-2$	$-\frac{1}{2}$	$(N-1)^*$	WAHS	$N-1$	$0 \to N-2$	-1^*	$N-\frac{3}{2}$	${S_{3o}}^{-1}$
S_{3o}	WAHS	$N-1$	$0 \to N-2$	-1^*	$N-\frac{3}{2}$	HAWA	$N-1$	$0 \to N-2$	$-\frac{1}{2}$	$(N-1)^*$	${S_{2o}}^{-1}$
S_{4o}	HAWS	N	$0 \to N-1$	$-\frac{1}{2}$	$N-1$	HAWS	N	$0 \to N-1$	$-\frac{1}{2}$	$N-1$	${S_{4o}}^{-1}$
	SPS	Length	Index Range	LPOS	RPOS	SPS	Length	Index Range	LPOS	RPOS	Transform
	Output extension properties					$\Longleftarrow$ Input extension properties					

are required to accurately represent a given function using the DFT, DCT or DST, and the more it can be compressed. However, the implicit periodicity of the input sequence that characterizes each DTT often implies considerable discontinuities at the signal boundaries, since any random segment of a given signal is unlikely to have the same pattern at both the left and right boundaries. In contrast, when both boundaries of a given signal are symmetrically extended, it naturally yields a continuous and smooth extension at the boundaries. This is why some types of DCT that have two symmetrically extended boundaries (in particular, DCTs of types I, II, V, and VI, as will be defined in the following subsections) generally perform better for signal compression than the other DTTs.

2.2. Discrete cosine transforms

From a pure and rather simplistic mathematical point of view, each DCT can be defined as a linear and invertible trigonometric function $\mathcal{F}:\mathbb{R}^N \to \mathbb{R}^N$. However, by recalling what was referred in the previous subsection, there are several different variants of the DCT, presenting distinct formal definitions and characteristics. Nevertheless, all of them share a common important property: they all process and output data sequences that are characterized by symmetric extensions at their left boundary. Hence, eight different variants of the DCT are available, corresponding to all possible symmetry extension combinations in both boundaries of the output sequence that comply with this specific characteristic. A comprehensive list of extension alternatives and their corresponding variants of the DCT is presented in Table 1.

Among these transforms, those that also present the same characteristic in terms of the symmetry point in both boundaries (half-sample or whole-sample symmetry) are often denoted by *even* DCTs. Such transforms are usually denominated by DCT I, II, III and IV. On the other hand, those transforms with distinct characteristics in terms of the adopted point of symmetry in the two boundaries are usually denoted by *odd* DCTs and are denominated by DCT V, VI, VII and VIII. Hence, while one of the boundaries presents a symmetry/anti-symmetry characteristic around an original data point, the other is extended around an implicit halfway point between two data samples. These odd transforms, however, have been rarely used in practical image and video processing and coding applications.

Independently of the specific type of DCT, the mathematical definition of each of these transforms is represented as a sum of product terms that multiply a cosine function $\mathrm{C}(m,i)$ with the input sequence $x(i)$:

$$X(m)=\sum_i \mathrm{C}(m,i)x(i) \tag{10}$$

According to the previously defined nomenclature, the subscript i denotes the coordinate in the spatial (pixel) domain, whereas the subscript m represents the coordinate in the transform-domain.

Equivalent matrix definitions are often adopted in the literature, which represent the DCT computation as a simple matrix multiplication of a kernel matrix $\mathbf{C}$ by an input data vector **x**, in the form:

$$\mathbf{X} = \mathbf{C}\,\mathbf{x} \tag{11}$$

In such matrix formulations, an *e* or *o* subscript is often appended to the kernel matrix definition (**C**), to denote *even* or *odd* transforms (e.g., $\mathbf{C_e},\mathbf{C_o}$).

Independently of the adopted formalization, the term $\mathsf{C}(m,i)$ can be defined by the product:

$$\mathsf{C}(m,i) = Aw_1(m)w_2(i)t(m,i) \tag{12}$$

where $t(m,i) = \cos(f(m,i))$ is the corresponding transform kernel; the term $w_1(m)$ is a weighting function that is adopted by some transforms to make the column vectors orthogonal to each other; the weighting function $w_2(i)$ is required by some transforms to make the row vectors orthogonal; and the scalar A is a final multiplier that normalizes the rows and columns in order to produce an orthonormal matrix. Hence, the mutual contribution of all these weighting functions leads to an *orthonormal* definition of each considered transform. These properties will be further described in Section 2.5.

In Table 2, it is presented a comprehensive list of the several orthogonal DCT kernel matrix definitions [10]. The terms $w_1(m)$ and $w_2(i)$ are implemented by the orthogonalization functions $\xi(p)$ and $\zeta(p)$, defined in Eqs. (13) and (14), respectively.

$$\xi(p) = \begin{cases} \sqrt{\frac{1}{2}} & \text{for } p=0 \text{ or } p=N \\ 1 & \text{for } p=1,2,\ldots,N-1 \end{cases} \tag{13}$$

$$\zeta(p) = \begin{cases} 1 & \text{for } p=0,1,\ldots,N-2 \\ \sqrt{\frac{1}{2}} & \text{for } p=N-1 \end{cases} \tag{14}$$

An interesting observation is that the denominator in each kernel definition in Table 2 agrees with the distance between the corresponding LPOS and RPOS of Table 1. It can also be observed that the definitions of the odd DCTs are quite similar to the corresponding even definitions, where the denominators of the cosine arguments are replaced by the value $N \pm \frac{1}{2}$.

2.3. Discrete sine transforms

Similarly to the previously defined DCTs, each DST can be defined as a linear and invertible trigonometric function $\mathcal{F}:\mathbb{R}^N \to \mathbb{R}^N$. Likewise, the several variants of the DST also share a common and important property, since they all process and output data sequences that are characterized by anti-symmetric extensions at their left boundary. Consequently, eight different variants of the DST are also available, corresponding to all possible extension combinations in both boundaries of the output sequence that comply with this specific characteristic. A comprehensive list of extension alternatives and the corresponding variants of the DST is also presented in Table 1.

Just like the cosine transforms, DSTs are also divided in *even* and *odd* transforms, whenever they present the same characteristic in terms of the symmetry point in both boundaries (half-sample or whole-sample symmetry), or when they have distinct characteristics in terms of the adopted point of symmetry in the two boundaries, respectively. Hence, *even* DSTs are also denoted by DST I, II, III and IV, while *odd* DSTs are usually referred to as DST V, VI, VII and VIII.

Similarly to the DCTs, the mathematical definition of each DST is represented as a sum of product terms that multiply a sine function $\mathsf{S}(m,i)$ with the input sequence $x(i)$:

$$X(m) = \sum_i \mathsf{S}(m,i)x(i) \tag{15}$$

where the term $\mathsf{S}(m,i)$ is defined by the product:

$$\mathsf{S}(m,i) = Aw_1(m)w_2(i)t(m,i) \tag{16}$$

and $t(m,i) = \sin(f(m,i))$ is the corresponding transform kernel.

The equivalent matrix definitions represent the computation of each DST as a matrix multiplication of a kernel matrix **S** by an input data vector **x**, in the form:

$$\mathbf{X} = \mathbf{S}\,\mathbf{x} \tag{17}$$

A comprehensive list of the several orthogonal DST kernel matrix definitions is also presented in Table 2 [10]. Similarly to the DCT definitions, the transform kernel of each odd DST are quite similar to the corresponding even definition, where the denominators of the sine arguments are replaced by the value $N \pm \frac{1}{2}$.

2.4. Inverse transforms

The following expressions represent the relations between each inverse kernel matrix and the respective forward kernel matrix:

$$\mathsf{C}_1^{-1} = \mathsf{C}_1 \tag{18}$$

$$\mathsf{C}_2^{-1} = \mathsf{C}_3 \tag{19}$$

$$\mathsf{C}_3^{-1} = \mathsf{C}_2 \tag{20}$$

$$\mathsf{C}_4^{-1} = \mathsf{C}_4 \tag{21}$$

$$\mathsf{S}_1^{-1} = \mathsf{S}_1 \tag{22}$$

$$\mathsf{S}_2^{-1} = \mathsf{S}_3 \tag{23}$$

$$\mathsf{S}_3^{-1} = \mathsf{S}_2 \tag{24}$$

$$\mathsf{S}_4^{-1} = \mathsf{S}_4 \tag{25}$$

The designations for *even* or *odd* transforms were omitted from these equations, since the same relation holds for both the *even* and *odd* cases.

2.5. Main mathematical properties

Several useful properties can be derived from the previously defined sine and cosine transforms. This section presents a brief overview of the most important and useful properties of these transforms applied for image and video coding [5,7,8,10,25]. In such presentation, a generic discrete cosine transform will be used for the sake of illustration. Nevertheless, the presented properties are

Table 2
Definition of the orthogonal DCT and DST kernel matrices, as defined by [10].

DTT	Definition	Length	Index range
DCT-I DCT-1e	$[\mathrm{C}_{1e}]_{\mathrm{m,i}}=\sqrt{\frac{2}{N-1}}\xi(m)\zeta(m)\xi(i)\zeta(i)\cos\left(\frac{mi\pi}{N-1}\right)$	N	$m,i=0,1,\ldots,(N-1)$
DCT-II DCT-2e	$[\mathrm{C}_{2e}]_{\mathrm{m,i}}=\sqrt{\frac{2}{N}}\xi(m)\cos\left(\frac{m\left(i+\frac{1}{2}\right)\pi}{N}\right)$	N	$m,i=0,1,\ldots,(N-1)$
DCT-III DCT-3e	$[\mathrm{C}_{3e}]_{\mathrm{m,i}}=\sqrt{\frac{2}{N}}\xi(i)\cos\left(\frac{\left(m+\frac{1}{2}\right)i\pi}{N}\right)$	N	$m,i=0,1,\ldots,(N-1)$
DCT-IV DCT-4e	$[\mathrm{C}_{4e}]_{\mathrm{m,i}}=\sqrt{\frac{2}{N}}\cos\left(\frac{\left(m+\frac{1}{2}\right)\left(i+\frac{1}{2}\right)\pi}{N}\right)$	N	$m,i=0,1,\ldots,(N-1)$
DCT-V DCT-1o	$[\mathrm{C}_{1o}]_{\mathrm{m,i}}=\sqrt{\frac{2}{N-\frac{1}{2}}}\xi(m)\xi(i)\cos\left(\frac{mi\pi}{N-\frac{1}{2}}\right)$	N	$m,i=0,1,\ldots,(N-1)$
DCT-VI DCT-2o	$[\mathrm{C}_{2o}]_{\mathrm{m,i}}=\sqrt{\frac{2}{N-\frac{1}{2}}}\xi(m)\zeta(i)\cos\left(\frac{m\left(i+\frac{1}{2}\right)\pi}{N-\frac{1}{2}}\right)$	N	$m,i=0,1,\ldots,(N-1)$
DCT-VII DCT-3o	$[\mathrm{C}_{3o}]_{\mathrm{m,i}}=\sqrt{\frac{2}{N-\frac{1}{2}}}\zeta(m)\xi(i)\cos\left(\frac{\left(m+\frac{1}{2}\right)i\pi}{N-\frac{1}{2}}\right)$	N	$m,i=0,1,\ldots,(N-1)$
DCT-VIII DCT-4o	$[\mathrm{C}_{4o}]_{\mathrm{m,i}}=\sqrt{\frac{2}{N+\frac{1}{2}}}\cos\left(\frac{\left(m+\frac{1}{2}\right)\left(i+\frac{1}{2}\right)\pi}{N+\frac{1}{2}}\right)$	N	$m,i=0,1,\ldots,(N-1)$
DST-I DST-1e	$[\mathrm{S}_{1e}]_{\mathrm{m,i}}=\sqrt{\frac{2}{N}}\sin\left(\frac{mi\pi}{N}\right)$	$(N-1)$	$m,i=1,2,\ldots,(N-1)$
DST-II DST-2e	$[\mathrm{S}_{2e}]_{\mathrm{m,i}}=\sqrt{\frac{2}{N}}\zeta(m)\sin\left(\frac{(m+1)\left(i+\frac{1}{2}\right)\pi}{N}\right)$	N	$m,i=0,1,\ldots,(N-1)$
DST-III DST-3e	$[\mathrm{S}_{3e}]_{\mathrm{m,i}}=\sqrt{\frac{2}{N}}\zeta(i)\sin\left(\frac{\left(m+\frac{1}{2}\right)(i+1)\pi}{N}\right)$	N	$m,i=0,1,\ldots,(N-1)$
DST-IV DST-4e	$[\mathrm{S}_{4e}]_{\mathrm{m,i}}=\sqrt{\frac{2}{N}}\sin\left(\frac{\left(m+\frac{1}{2}\right)\left(i+\frac{1}{2}\right)\pi}{N}\right)$	N	$m,i=0,1,\ldots,(N-1)$
DST-V DST-1o	$[\mathrm{S}_{1o}]_{\mathrm{m,i}}=\sqrt{\frac{2}{N-\frac{1}{2}}}\sin\left(\frac{(m+1)(i+1)\pi}{N-\frac{1}{2}}\right)$	$(N-1)$	$m,i=1,2,\ldots,(N-1)$
DST-VI DST-2o	$[\mathrm{S}_{2o}]_{\mathrm{m,i}}=\sqrt{\frac{2}{N-\frac{1}{2}}}\sin\left(\frac{(m+1)\left(i+\frac{1}{2}\right)\pi}{N-\frac{1}{2}}\right)$	$(N-1)$	$m,i=0,1,\ldots,(N-2)$
DST-VII DST-3o	$[\mathrm{S}_{3o}]_{\mathrm{m,i}}=\sqrt{\frac{2}{N-\frac{1}{2}}}\sin\left(\frac{\left(m+\frac{1}{2}\right)(i+1)\pi}{N-\frac{1}{2}}\right)$	$(N-1)$	$m,i=0,1,\ldots,(N-2)$
DST-VIII DST-4o	$[\mathrm{S}_{4o}]_{\mathrm{m,i}}=\sqrt{\frac{2}{N-\frac{1}{2}}}\zeta(m)\zeta(i)\sin\left(\frac{\left(m+\frac{1}{2}\right)\left(i+\frac{1}{2}\right)\pi}{N-\frac{1}{2}}\right)$	N	$m,i=0,1,\ldots,(N-1)$

equally valid for the whole family of trigonometric transforms defined in the previous subsections.

- *Linearity*: According to the several definitions summarized in Table 2, any trigonometric transform can be regarded as a linear combination of linear functions (sine (S) or cosine (C) functions), which are added together using the input signal samples as weighting factors:

$$X(m)=\sum_i x(i)\mathrm{C}(m,i) \tag{26}$$

As a consequence, by denoting by $X(m)$ and $Y(m)$ the cosine transforms of the input samples $x(i)$ and $y(i)$, the following statement defines the linearity property of this transform for any scalar α and $\beta \in \mathbb{R}$:

$$\mathrm{DCT}[\alpha x(i)+\beta y(i)] = \alpha X(m)+\beta Y(m) \tag{27}$$

This particular property has provided several important and interesting applications, by allowing the processing and combination of pre-encoded image/video data directly in the DCT-domain (e.g., transform-domain transcoding [21]).

- *Orthogonality*: The row and column vectors that compose each discrete sine and cosine transform kernel matrix define a set of orthogonal basis functions. Let $\mathbf{C}$ denote the $(n \times m)$ kernel matrix of a given cosine transform:

$$\mathbf{C} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{bmatrix} \tag{28}$$

This matrix is said to be orthogonal because all column vectors $\mathbf{c}_i = [a_{1,i}\ a_{2,i}\ \dots\ a_{n,i}]^{\mathrm{T}}$ fulfill the following relation, denoted by $\mathbf{c}_i \perp \mathbf{c}_j$:

$$\mathbf{c}_1^{\mathrm{T}}\mathbf{c}_2 = \mathbf{c}_1^{\mathrm{T}}\mathbf{c}_3 = \cdots = \mathbf{c}_i^{\mathrm{T}}\mathbf{c}_j = 0, \quad \forall i \neq j \tag{29}$$

with $1 \le i,j \le m$. As a consequence, these relations also imply that

$$\mathbf{C}^{\mathrm{T}} = \mathbf{C}^{-1} \tag{30}$$

Entirely similar relations also apply for the row vectors $\mathbf{r}_i$.

- *Normalization*: Each column vector $\mathbf{c}_i$ of any discrete sine and cosine transform kernel matrix presented in Table 2 also fulfills the following property:

$$\|\mathbf{c}_i\| = 1, \quad \forall i : 1 \le i \le m \tag{31}$$

As a consequence, the kernel matrices defined in Table 2 are also said to be normalized.

- *Orthonormality*: Considering that each column or row vector of the presented discrete sine or cosine transform kernel matrices are both orthogonal and normalized, these matrices are also said to be orthonormal, thus presenting the following important properties:

$$\mathbf{C}^{\mathrm{T}} = \mathbf{C}^{-1}, \quad \mathbf{C}^{\mathrm{T}}\mathbf{C} = \mathbf{I}, \quad \mathbf{C}\mathbf{C}^{\mathrm{T}} = \mathbf{I} \tag{32}$$

These relations lead to a quite important consequence, since the matrix inversion operation is reduced to a simple matrix transpose, resulting in a significant computational cost reduction.

- *Energy conservation (Parseval's theorem)*: Another important property of these discrete sine and cosine transform kernel matrices is related to the conservation of the signal energy after the computation of its transform. This property (also denoted by Parseval's theorem) can be formulated by the following expression:

$$\sum_{m=0}^{N-1} |X(m)|^2 = \|\mathbf{X}\|^2 = \mathbf{X}^{\mathrm{T}}\mathbf{X} = \mathbf{x}^{\mathrm{T}}\mathbf{C}^{\mathrm{T}}\mathbf{C}\mathbf{x} = \mathbf{x}^{\mathrm{T}}\mathbf{x} = \|\mathbf{x}\|^2 = \sum_{i=0}^{N-1} |x(i)|^2 \tag{33}$$

Besides this, an important aspect about these transforms is related to their capability to pack most of the signal energy into the lower order coefficients. One consequence of such characteristic is that the drop of some high-order coefficients (through quantization and truncation) usually leads to a marginal loss of the signal energy, resulting in a minimal distortion level [9].

- *Scaling*: Since the DTTs deal with discrete sampled points, a scaling in the pixel-domain has no direct effect in the transform-domain, except for a change in the frequency unit [10]. Hence, as the sampling interval δi changes to $\alpha\delta i$, the frequency unit δm changes to $\delta m/\alpha$, provided that the sequence length (N) remains constant. As an example, in the particular case of the DCT, this property can be stated as

$$\mathrm{DCT}[x(\alpha i)] = X\left(\frac{m}{\alpha}\right) \quad \text{for } \alpha > 0 \tag{34}$$

- *Shift*: Let $\mathbf{x} = [x(0),\dots,x(N-1)]^{\mathrm{T}}$ and $\mathbf{x}_+ = [x(1),\dots,x(N)]^{\mathrm{T}}$ be two length-N sequences, where $\mathbf{x}_+$ denotes the sequence $\mathbf{x}$ shifted by one sample point. Britanak et al. presented a detailed formulation of the relation between the corresponding transform-domain sequences for the whole family of DTTs [10]. As an example, for the particular case of the even type-II DCT, they have shown that

$$X_+^{\mathbf{C_{2e}}}(m) = \cos\left(\frac{m\pi}{N}\right)X^{\mathbf{C_{2e}}}(m) + \sin\left(\frac{m\pi}{N}\right)X^{\mathbf{S_{2e}}}(m-1) + \sqrt{\frac{2}{N}}\xi(m)\cos\left(\frac{m\pi}{2N}\right)[(-1)^m x(N) - x(0)] \tag{35}$$

where $\xi(m)$ is given by Eq. (13) and $\mathbf{X}^{\mathbf{C_{2e}}} = \mathbf{C_{2e}}\mathbf{x}$ and $\mathbf{X}_+^{\mathbf{C_{2e}}} = \mathbf{C_{2e}}\mathbf{x}_+$ are the even type-II DCT vectors of the pixel sequence $\mathbf{x}$ and of its shifted version $\mathbf{x}_+$, respectively. Likewise, $\mathbf{X}^{\mathbf{S_{2e}}}$ is used to denote the even type-II DST, as defined in Table 2.

3. Multidimensional transforms

The 2-D nature of image data leads to the usual representation of pixel data along the two spatial orthogonal directions. Likewise, the representation of image data in the frequency domain makes use of multidimensional extensions of the several previously described sine and cosine transforms, defined along the two corresponding spatial frequencies. Such extensions can be straightforwardly defined by considering separable decompositions along each dimension. In fact, it can be easily shown that a 2-D transform can be regarded as the application of the same 1-D transform, performed firstly along the rows and then along the columns (see Fig. 4), or vice-versa. As an example, by considering the generic 1-D definition of the cosine transform, the extension to two dimensions can be defined as

$$X(m,n) = \sum_i \sum_j \mathrm{C}(m,i)\mathrm{C}(n,j)x(i,j) \tag{36}$$

By re-arranging this equation, one can obtain

$$X(m,n) = \sum_i \mathrm{C}(m,i) \sum_j \mathrm{C}(n,j)x(i,j) \tag{37}$$

$$X(m,n) = \sum_i \mathrm{C}(m,i)\tilde{X}^{1D}(i,n) \tag{38}$$

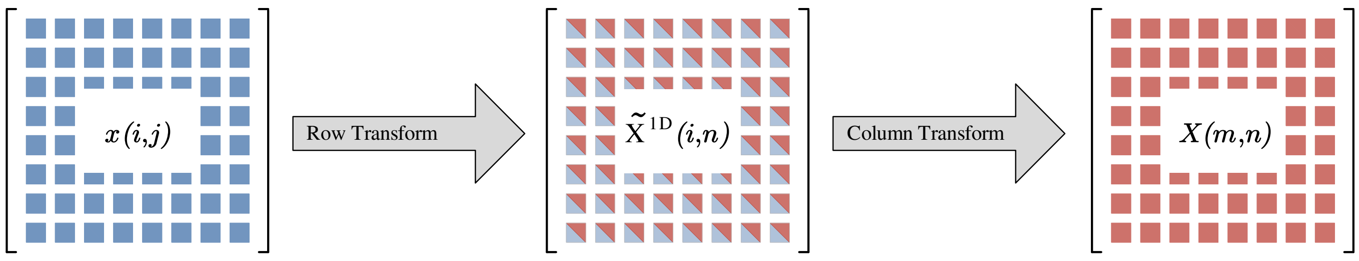


Fig. 4. Row–column decomposition of a 2-D transform.

where $\tilde{X}^{1D}(i,n) = [\mathbf{C}\,\mathbf{x}^{\mathrm{T}}]^{\mathrm{T}} = \mathbf{x}\,\mathbf{C}^{\mathrm{T}}$ is a matrix whose lines are the 1-D discrete cosine transform of the lines of $x(i,j)$.

Hence, the 2-D discrete cosine transform of the input matrix $\mathbf{x} = [\mathbf{x}]_{i,j}$ can be represented in a matrix-product form as follows:

$$\mathbf{X} = [\mathbf{X}]_{m,n} = \mathbf{C}\,\mathbf{x}\,\mathbf{C}^{\mathrm{T}} \tag{39}$$

The same formulation could equally be applied to any other type of sine or cosine transform, as well as to their 2-D inverse transforms.

The corresponding inverse transform can then be derived as

$$\mathbf{x} = [\mathbf{x}]_{i,j} = (\mathbf{C})^{-1}\mathbf{X}(\mathbf{C}^{\mathrm{T}})^{-1} \tag{40}$$

By taking into account the orthogonality property of the kernel matrices ($\mathbf{C}^{-1} = \mathbf{C}^{\mathrm{T}}$), defined in Subsection 2.5, the following expression can be easily derived for the inverse transform:

$$\mathbf{x} = [\mathbf{x}]_{i,j} = \mathbf{C}^{\mathrm{T}}\mathbf{X}\mathbf{C} \tag{41}$$

This row-column decomposition defines an additional property that is often referred to as *separability property*. Such relation may provide a quite important computational advantage: $X(m,n)$ can be computed in two steps by applying successive 1-D operations on the rows and columns of the image data. From a dedicated implementation point of view, this row–column decomposition may significantly simplify the hardware requirements, at the expense of a slight increase in the overall operation-count.

4. Application of the DCT to image and video encoding

Contrasting to the other DTTs, the even type-II discrete cosine transform (DCT-II) has been widely adopted in image and video processing applications and is currently the basis of many image and video standards (e.g., JPEG [14], H.263 [2], MPEG-2 video [4], etc.). As it was previously referred, such fact is mainly owed to its particularly well suited characteristics to exploit the spatial irrelevancies of a given pixels area, by concentrating most of the pixels energy in a restricted set of DCT coefficients [26]. Hence, most image and video standards transform each $(N \times N)$ pixels block from the spatial-domain into a $(N \times N)$ matrix of DCT-domain coefficients, where N is typically set to eight pixels. The selection of this particular block size is historically related to several reasons. In what concerns the hardware and software implementation point of view, an (8×8) block size does not impose significant memory requirements and its DCT computation is easily manageable in most computing platforms. On the other hand, in what concerns the compaction efficiency point of view, it has been observed that block sizes larger than (8×8) pixels do not offer any significantly better compression levels [26].

4.1. 1-D discrete cosine transform

According to the above definitions, the 1-D DCT usually adopted in image and video coding can be formulated as [7,8,10–12,27]

$$X(m) = \sqrt{\frac{2}{N}}\xi(m)\sum_{i=0}^{N-1}\cos\left(\frac{m(i+\frac{1}{2})\pi}{N}\right)x(i) \Longleftrightarrow \mathbf{X} = \mathbf{T}\mathbf{x} \tag{42}$$

$$x(i) = \sqrt{\frac{2}{N}}\sum_{m=0}^{N-1}\xi(m)\cos\left(\frac{m(i+\frac{1}{2})\pi}{N}\right)X(m) \Longleftrightarrow \mathbf{x} = \mathbf{T}^{\mathrm{T}}\mathbf{X} \tag{43}$$

with $m,i = 0,1,\ldots,(N-1)$ and $\xi(m)$ as defined in Eq. (13). The corresponding (8×8) kernel matrix $\mathbf{T}$ is defined as

$$[\mathbf{T}(m,i)] \triangleq [\mathrm{C}_{2e}(m,i)] = \sqrt{\frac{2}{N}}\xi(m)\cos\left(\frac{m(i+\frac{1}{2})\pi}{N}\right) \tag{44}$$

From a careful analysis of Eqs. (42) and (43), it can be observed that the computations of the forward and inverse DCT are nearly the same. Thus, from a dedicated implementation point of view, the same computational unit can be used for both the forward and the inverse DCTs.

It can also be observed, from Eq. (42), that the first transform coefficient ($X(0)$) represents the average value of the input sequence. As a consequence, this value is often referred to as the *DC coefficient*, in analogy to what happens with the circuit analysis theory in electrical engineering. In accordance, all other transform coefficients are often denoted by *AC coefficients*.

The DCT decomposes each signal into a series of cosine waveforms (basis functions), each one with a particular frequency. In Fig. 5(a) it is depicted the set of eight basis functions corresponding to the discrete cosine transform, with $N=8$. These waveforms actually correspond to the set of functions defined by the sum $\sum_{i=0}^{N-1}\cos(m(i+\frac{1}{2})\pi/N)$, with $N=8$ and m varying from $m=0$ to $N-1$. In accordance with the previous paragraph, the bottom waveform ($m=0$) renders a constant (DC) value, whereas all other waveforms ($m=1,2,\ldots,7$) represent different basis for progressively increasing frequencies. All these basis functions are

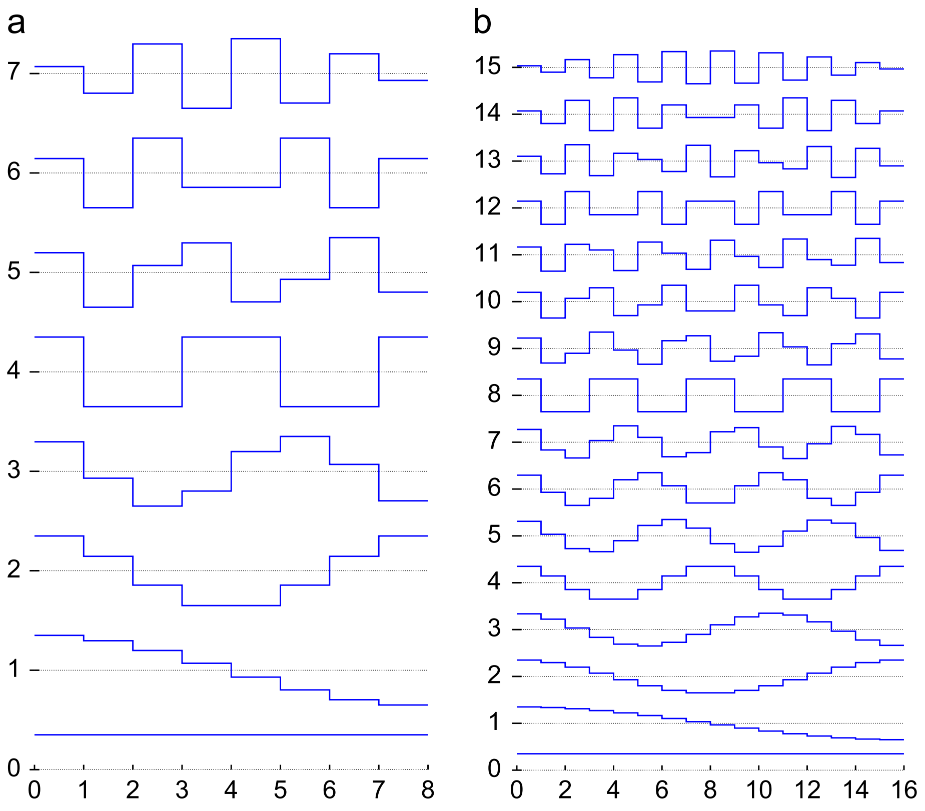


Fig. 5. 1-D DCT basis functions. (a) $N=8$. (b) $N=16$.

orthogonal. Hence, the multiplication between any pair of these waveforms followed by a summation over all sample points yields a zero (scalar) value, whereas the multiplication of any of these waveforms with itself followed by a summation operation yields a constant (scalar) value. As a consequence, according to the orthogonal definition, these waveforms are said to be independent: none of the basis functions can be represented as a combination of the other basis functions. Hence, the computation of the DCT can be regarded as the process of finding the weight sequence $X(m)$, corresponding to each waveform shown in Fig. 5(a), so that the sum of the eight waveforms, scaled by the corresponding weights $X(m)$, yields the reconstructed version of the original eight-point vector $x(i)$.

For illustration purposes, it is also presented in Fig. 5(b) the set of basis functions corresponding to a $N=16$-point DCT.

4.2. 2-D discrete cosine transform

The extension of the above defined transform to a bidimensional space is straightforward [7,8,10,11,27]:

$$X(m,n)=\frac{2}{N}\xi(m)\xi(n)\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}x(i,j)\cos\left(\frac{m\left(i+\frac{1}{2}\right)\pi}{N}\right)\times\cos\left(\frac{n\left(j+\frac{1}{2}\right)\pi}{N}\right) \tag{45}$$

$$\Leftrightarrow \mathbf{X}=\mathbf{T}\mathbf{x}\mathbf{T}^{\mathrm{T}} \tag{46}$$

$$x(i,j)=\frac{2}{N}\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}\xi(m)\xi(n)X(m,n)\times\cos\left(\frac{m\left(i+\frac{1}{2}\right)\pi}{N}\right)\cos\left(\frac{n\left(j+\frac{1}{2}\right)\pi}{N}\right) \tag{47}$$

$$\Leftrightarrow \mathbf{x}=\mathbf{T}^{\mathrm{T}}\mathbf{X}\mathbf{T} \tag{48}$$

with $m,n=0,1,\ldots,(N-1)$; $i,j=0,1,\ldots,(N-1)$; and $\xi(m)$ and $\xi(n)$ defined in Eq. (13).

Just like the definition of the 1-D DCT, the transform coefficient $X(0,0)$ represents the average value of the input sequence and is denoted by *DC coefficient*, while all other transform coefficients are denoted by *AC coefficients*. In Fig. 6, it is depicted the set of 64 basis functions corresponding to the 2-D discrete cosine transform, with $N=8$. These 2-D basis functions can be generated by multiplying the horizontally oriented 1-D basis functions (shown in Fig. 5(a)) with a vertically oriented set of the same functions. As it was observed for the 1-D case, these basis functions exhibit a progressive increase of their frequency component, both in the vertical and horizontal directions. A particular case occurs with the top-left basis function (*DC coefficient*), which results from the multiplication of two constant vectors, corresponding to the DC component in Fig. 5(a). Hence, the computation of each 2-D DCT can be regarded as the process of finding the weight sequence $X(m,n)$, corresponding to each waveform shown in Fig. 6, so that the sum of the 64 waveforms, scaled by the corresponding weights $X(m,n)$, yields the reconstructed version of the original (8×8) pixels matrix $x(i,j)$.

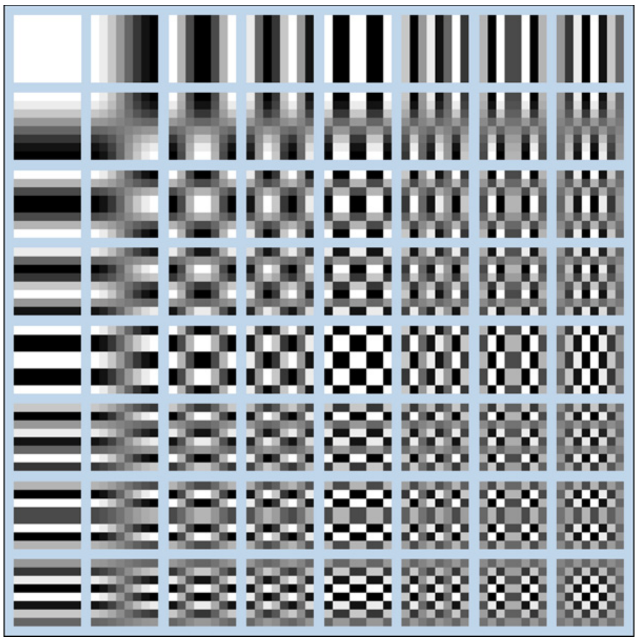


Fig. 6. 2-D DCT basis functions.

4.3. Fast computation of the discrete cosine transform

A significant effort has been devised, since the first applications of the DCT, in order to obtain efficient 1-D and 2-D implementations. In fact, it can be easily observed that a direct computation of an 1-D eight-point DCT (see Eq. (42)) requires up to 64 multiplications and 63 additions, while a 2-D eight-point DCT (see Eq. (45)) requires up to 4096 multiplications and 409 additions. Although such computational cost can be trivially reduced to 1024 multiplications and 1024 additions by using the separability property, it still represents a significant computational burden for most practical applications and frequently compromises real-time requisites.

As such, a significant number of fast algorithms for the computation of the DCT have been developed in the last decades. Most of these processing approaches either extensively exploit the properties of the transform kernel matrix **T** or the final precision of the transform result, in order to reduce the number of involved operations. A possible classification of these algorithms is the following (an extensive bibliography overview of these and other efficient algorithms can be found in [10]):

- *Factorization* of the kernel matrix, in order to decompose it into a set of sparse matrices; since only the non-null terms of each of these matrices need to be considered, a significant reduction of the involved operations is achieved.
- *Multiplication-free* implementations, by expressing the kernel matrix as a product of a diagonal matrix and a matrix with very small integer weights, in order to replace the multiplications by simple shift and add operations; such implementations are particularly attractive for low-cost processors that do not incorporate a multiplier unit, but introduce a small degradation at the attained precision.
- Partial *scaling* of the kernel matrix terms with the quantization matrix weights, in order to further reduce the number of non-null terms of the sparse decomposition of the matrix kernel; however, such technique assumes the use of a fixed quantization matrix, which is not always the case (e.g., video coding).
- *Pruning* approximations, by only computing some low-frequency DCT coefficients; such implementations are particularly adopted in low bit-rate image/video coding, where only a small subset of the DCT coefficients located at the top-left corner are evaluated.
- Use of *MAC arithmetic units*, by expressing the several involved operations as a compound (multiply-and-accumulate) form $a=bc+d$; such operations are efficiently implemented in most digital signal processor (DSPs) using a single clock cycle.

Either with the application of a single or of several of these methods, a significant reduction of the computational cost can be achieved, when compared with the direct implementations of the DCT.

Nevertheless, the algorithms based on *factorization* of the kernel matrix have had a particular attention over the last years, not only due to the universality of their application, but also because they do not impose any loss of the resulting precision. In Table 3, it is presented the computational cost of some of such algorithms (see further details in [10]). As an example, Feig and Winograd established a theoretical lower bound on the multiplication complexity (#mult) of both the 1-D and 2-D N-point DCT (with $N=2^n$), leading to an absolute minimum of 11 multiplications for the eight-point 1-D implementations and 88 multiplications for the (8×8)-point 2-D DCT [26,10,28]:

$$\#\text{mult}^{1\text{-}D}=2^{n+1}-n-2 \tag{49}$$

$$\#\text{mult}^{2\text{-}D}=2^{n}(2^{n+1}-n-2) \tag{50}$$

Meanwhile, it was demonstrated that such optimal performance can be achieved by applying Loeffler's 1-D eight-point algorithm together with Cho's 2-D (8×8)-point processing scheme [28]. Nevertheless, it is worth noting that most of these rather optimized algorithms

Table 3
Computational complexity of several fast DCT algorithms based on *factorization* of the kernel matrix.

Algorithm (N = 8)	**1-D**		**2-D**	
	Mults.	**Sums**	**Mults.**	**Sums**
Direct implementation	64	64	4096	4096
Chen (1977)	16	26	256	416
Lee (1984)	12	29	192	464
Leoffler (1989)	11	29	176	464
Chan (1991)			144	464
Kamangar (1982)			128	430
Cho (1991)			96	466
Feig (1992)			94	454
Wu (1998)			88	466

may also impose some important disadvantages, from a strict implementation point of view [26]:

- Data addressing is highly irregular, leading to an additional overhead for address calculations (which is not included in the performance metrics presented in Table 3), as well as important cache miss penalties.
- Significant data storage requirements, to accommodate temporary results.

As such, many practical implementations usually adopt simpler but still efficient row-column decompositions.

Meanwhile, with the recent advent of modern graphics processing units (GPUs), the general-purpose computation on GPUs (GPGPU) paradigm has gradually been applied in order to obtain highly parallel implementations of the DCT [29]. With this programming model, a significant number of parallel threads are concurrently executed, in order to simultaneously process several blocks of pixels of the image/frame under processing. Such implementations usually exploit the separability property and the symmetry characteristics of the DCT kernel matrix, in order to implement single instruction multiple data (SIMD) vectorization schemes that make extensive use of the highly efficient multiplication, addition and MAC arithmetic units available at these platforms.

5. Multiplication-convolution property: definition and applications

Frequently, signal processing functions that directly operate with the DCT coefficients of an encoded image or video stream require the application of other more complex properties, besides those presented in Section 2.5. One of such properties concerns the relation between the convolution operation and the corresponding point-by-point multiplication.

The formulation of this type of relations is quite common in other transform domains. As an example, the DFT domain component-wise multiplication is widely used to compute the spatial (or time) domain convolution, required by many feature extraction and filtering operations. Nevertheless, a similar relation, but implemented in the DCT-domain, is also required by several other applications (e.g., linear filtering [30], static image/video composition [21], etc.). However, although the DCT is closely related to the DFT, the multiplication-convolution theorem for the DCT was formulated much after the corresponding relationship for the DFT. In fact, despite the several attempts to establish this relation [31], a complete and more consistent formalization was only presented relatively recently [23,24,32]. In particular, Martucci [23] presented a formalized and detailed description of the convolution operation for the entire family of discrete sine and cosine transforms. In his presentation, the DCT and the DST are regarded as special cases of the so called generalized discrete Fourier transform (GDFT), which operates on strictly periodic or antiperiodic (see Section 2.1) infinite sequences, with period N. Martucci denoted such definition as *symmetric convolution*, since it proved to be specially suited to convolve symmetrically extended sequences. In practice, such type of convolution can be regarded as a conventional convolution sum, that has been suitably modified to incorporate the implicit symmetric extensions of both operands.

In this section, the formulation of the DCT-domain multiplication-convolution property, in the particular scope of encoded images and video processing will be presented. Such property provides the means to implement several processing functions directly in the encoded domain, by replacing the spatial-domain pixel-wise multiplication by a DCT-domain symmetric convolution operation. A fast computational method to compute such convolution will be presented in Section 5.3.

5.1. Generic discrete trigonometric transform

The definition of the discrete cosine transform that was presented in the previous sections corresponds to the first formulation of the DCT, reported by Ahmed et al. in 1974 [11] and latter categorized by Wang [12,13]. It is also the most commonly used in image and video standards. As it was referred in Section 2, several other DTTs have also been proposed, such as the eight types of DCT and the eight types of DST defined in Tables 1 and 2 [10,23]. Since the kernel matrices of all these transforms are orthogonal and invertible, the kernel matrices of their inverses can be easily obtained by transposing them.

Meanwhile, Martucci [23] proposed a new formulation for these DTTs, denoted by *convolution form*. In this formulation, the orthogonality property of the kernel matrices was waived for most DTT types. Such formulation was shown to be more appropriate for applying the *convolution-multiplication property* than the above *orthogonal form*, derived by Wang [12], since it avoids the need for adding any scaling factors or other weighting functions to the convolution-multiplication formula. The definition and formulation of each of these alternative kernel matrices for the DCT ($\mathcal{C}$) and the DST ($\mathcal{S}$) is presented in Tables 4 and 5 [23,30]. Nevertheless, direct relations between the orthogonal and convolution forms of each DTT can be easily established. As an example, the kernel matrices $\mathbf{C_{2e}}$ and $\mathcal{C}_{\mathbf{2e}}$ can be related by a $(N \times N)$ diagonal matrix $\mathbf{R}$ as $\mathbf{C_{2e}} = \mathbf{R}\mathcal{C}_{\mathbf{2e}}$, where $R(0,0) = 2\sqrt{N}$ and $R(m,m) = \sqrt{2N}$, for $m = 1, \ldots, (N-1)$ [30]. Since the kernel functions of some of these transforms evaluate to zero for some values of the indices m and i, Martucci rearranged some of their index ranges in order to avoid null values. As a consequence, contrasting with the previous definitions presented in Tables 1 and 2, the index ranges of m and i for some of these transforms are no longer the same. Nevertheless, with this alternative formulation there is a direct link between all DTTs and the GDFT.

The relations between each inverse kernel matrix and its own (or another) forward kernel matrix, entirely similar to those that were presented in Eqs. (18)–(25) for the orthogonal DCT and DST definitions, can also be formulated for these convolution-form transforms. Nevertheless, it should be noted that the matrix inversion operation can no longer be implemented through a matrix transposition, since these definitions are not orthogonal any more.

Table 4
Properties of the implicit input and output extensions of the convolution forms of the considered discrete sine and cosine transforms.

Transform	Input extension properties				$\Longrightarrow$	Output extension properties					
	SPS	Length	Index Range	LPOS	RPOS	SPS	Length	Index Range	LPOS	RPOS	
$\mathcal{C}_{1e}$	WSWS	$N+1$	$0 \rightarrow N$	0	N	WSWS	$N+1$	$0 \rightarrow N$	0	N	$\mathcal{C}_{1e}^{-1}$
$\mathcal{C}_{2e}$	HSHS	N	$0 \rightarrow N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	WSWA	N	$0 \rightarrow N-1$	0	N^*	$\mathcal{C}_{3e}^{-1}$
$\mathcal{C}_{3e}$	WSWA	N	$0 \rightarrow N-1$	0	N^*	HSHS	N	$0 \rightarrow N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	$\mathcal{C}_{2e}^{-1}$
$\mathcal{C}_{4e}$	HSHA	N	$0 \rightarrow N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	HSHA	N	$0 \rightarrow N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	$\mathcal{C}_{4e}^{-1}$
$\mathcal{C}_{1o}$	WSHS	N	$0 \rightarrow N-1$	0	$N-\frac{1}{2}$	WSHS	N	$0 \rightarrow N-1$	0	$N-\frac{1}{2}$	$\mathcal{C}_{1o}^{-1}$
$\mathcal{C}_{2o}$	HSWS	N	$0 \rightarrow N-1$	$-\frac{1}{2}$	$N-1$	WSHA	N	$0 \rightarrow N-1$	0	$N-\frac{1}{2}$	$\mathcal{C}_{3o}^{-1}$
$\mathcal{C}_{3o}$	WSHA	N	$0 \rightarrow N-1$	0	$N-\frac{1}{2}$	HSWS	N	$0 \rightarrow N-1$	$-\frac{1}{2}$	$N-1$	$\mathcal{C}_{2o}^{-1}$
$\mathcal{C}_{4o}$	HSWA	$N-1$	$0 \rightarrow N-2$	$-\frac{1}{2}$	$N-1^*$	HSWA	$N-1$	$0 \rightarrow N-2$	$-\frac{1}{2}$	$N-1^*$	$\mathcal{C}_{4o}^{-1}$
$\mathcal{S}_{1e}$	WAWA	$N-1$	$1 \rightarrow N-1$	0^*	N^*	WAWA	$N-1$	$1 \rightarrow N-1$	0^*	N^*	$\mathcal{S}_{1e}^{-1}$
$\mathcal{S}_{2e}$	HAHA	N	$0 \rightarrow N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	WAWS	N	$1 \rightarrow N$	0^*	N	$\mathcal{S}_{3e}^{-1}$
$\mathcal{S}_{3e}$	WAWS	N	$1 \rightarrow N$	0^*	N	HAHA	N	$0 \rightarrow N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	$\mathcal{S}_{2e}^{-1}$
$\mathcal{S}_{4e}$	HAHS	N	$0 \rightarrow N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	HAHS	N	$0 \rightarrow N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	$\mathcal{S}_{4e}^{-1}$
$\mathcal{S}_{1o}$	WAHA	$N-1$	$1 \rightarrow N-1$	0^*	$N-\frac{1}{2}$	WAHA	$N-1$	$1 \rightarrow N-1$	0^*	$N-\frac{1}{2}$	$\mathcal{S}_{1o}^{-1}$
$\mathcal{S}_{2o}$	HAWA	$N-1$	$0 \rightarrow N-2$	$-\frac{1}{2}$	$N-1^*$	WAHS	$N-1$	$1 \rightarrow N-1$	0^*	$N-\frac{1}{2}$	$\mathcal{S}_{3o}^{-1}$
$\mathcal{S}_{3o}$	WAHS	$N-1$	$1 \rightarrow N-1$	0^*	$N-\frac{1}{2}$	HAWA	$N-1$	$0 \rightarrow N-2$	$-\frac{1}{2}$	$N-1^*$	$\mathcal{S}_{2o}^{-1}$
$\mathcal{S}_{4o}$	HAWS	N	$0 \rightarrow N-1$	$-\frac{1}{2}$	$N-1$	HAWS	N	$0 \rightarrow N-1$	$-\frac{1}{2}$	$N-1$	$\mathcal{S}_{4o}^{-1}$
	SPS	Length	Index Range	LPOS	RPOS	SPS	Length	Index Range	LPOS	RPOS	Transform
	Output extension properties				$\Longleftarrow$	Input extension properties					

Moreover, a scaling factor, given by $1/M$, should be applied to each relation, where $M=2N$ for *even* transforms and $M=2N-1$ for *odd* transforms (e.g., $\mathcal{C}_{\mathbf{2e}}^{-1}=(1/M)\mathcal{C}_{\mathbf{3e}}$) [23].

5.2. Definition of the multiplication-convolution property

The multiplication-convolution property relates the convolution operation of two periodic sequences, defined in a given trigonometric domain, with the element-by-element multiplication operation of the corresponding sequences' coefficients, defined in their inverse transformed trigonometric domains.

Martucci presented the formulation of this property for all considered DTTs by stating that the inverse transform, after element-by-element multiplication, gives the same result as the convolution of the original sequences [23]:

$$\mathbf{\Omega}_n = \varepsilon_a\{\mathbf{\Phi}_n\} \circledast \varepsilon_b\{\mathbf{\Psi}_n\} = \tau_c^{-1}\{\tau_a\{\mathbf{\Phi}_n\} \odot \tau_b\{\mathbf{\Psi}_n\}\} \qquad (51)$$

where $\mathbf{\Phi}_n$ and $\mathbf{\Psi}_n$ are two input sequences of finite length and $\mathbf{\Omega}_n$ is the output convolved sequence. In this expression, ε_a and ε_b are two generic symmetric or anti-symmetric extension operators, as defined in Section 2.1, and $\odot$ denotes the element-by-element multiplication. The symbol $\circledast$ denotes the symmetric convolution operation, defined in terms of a conventional convolution sum that has been suitably modified to incorporate the implicit symmetric extensions of both operands. The operators τ_a, τ_b and τ_c^{-1} define three invertible convolution-form DTTs, as defined in Table 5, that transform from one trigonometric domain to another.

Martucci grouped such relations in several families of three or four DTTs, whose input and output data sequences share the same type of symmetry [23]. However, considering that most image and video standards make use of the even type-II DCT (DCT-IIe), characterized by transforming HSHS symmetric sequences into WSWA sequences (see Table 1), two particular classes of convolution formulations are specially relevant, in order to allow the processing of encoded image/video blocks directly in the DCT-domain:

Class 1: HSHS convolution output; at least one HSHS input sequence: $\mathbf{\Omega}^{\text{HSHS}}=f(\mathbf{\Phi}^{\text{HSHS}},\mathbf{\Psi})$.

Class 2: WSWA convolution output; at least one WSWA input sequence: $\mathbf{\Omega}^{\text{WSWA}}=f(\mathbf{\Phi}^{\text{WSWA}},\mathbf{\Psi})$.

The main characteristics of such specific relations are depicted in Table 6, where Ⓒ and Ⓢ denote the *circular* and the *skew-circular* convolution operations, respectively. Such particular forms of the symmetric convolution of two length-M sequences $\alpha(n)$ and $\beta(n)$, with $n=0,1,\ldots,M-1$ are defined as

$$\alpha(n) \text{ Ⓒ } \beta(n) = \sum_{k=0}^{n} \alpha(k)\beta(n-k) + \sum_{k=n+1}^{M-1} \alpha(k)\beta(n-k+M) \qquad (52)$$

$$\alpha(n) \text{Ⓢ} \beta(n) = \sum_{k=0}^{n} \alpha(k)\beta(n-k) - \sum_{k=n+1}^{M-1} \alpha(k)\beta(n-k+M) \qquad (53)$$

Accordingly, the circular and the skew-circular convolution operations of two length-M sequences define an output sequence that is equivalent to the corresponding period of a periodic convolution of symmetric and anti-symmetric sequences, respectively, with period M.

Table 5
Definition of the convolution forms of the DCT and DST kernel matrices [23].

DTT[a]	Definition	Length	Index range
cDCT-I cDCT-1e	$[\mathcal{C}_{1e}]_{\mathrm{m,i}}=2\xi^2(i)\cos\left(\frac{mi\pi}{N}\right)$	$(N+1)$	$m,\ i=0,1,\ldots,N$
cDCT-II cDCT-2e	$[\mathcal{C}_{2e}]_{\mathrm{m,i}}=2\cos\left(\frac{m(i+\frac{1}{2})\pi}{N}\right)$	N	$m,\ i=0,1,\ldots,\ (N-1)$
cDCT-III cDCT-3e	$[\mathcal{C}_{3e}]_{\mathrm{m,i}}=2\xi^2(i)\cos\left(\frac{m(i+\frac{1}{2})i\pi}{N}\right)$	N	$m,\ i=0,1,\ldots,\ (N-1)$
cDCT-IV cDCT-4e	$[\mathcal{C}_{4e}]_{\mathrm{m,i}}=2\cos\left(\frac{(m+\frac{1}{2})(i+\frac{1}{2})\pi}{N}\right)$	N	$m,\ i=0,1,\ldots,\ (N-1)$
cDCT- V cDCT-1o	$[\mathcal{C}_{1o}]_{\mathrm{m,i}}=2\xi^2(i)\cos\left(\frac{\mathrm{min}}{N-\frac{1}{2}}\right)$	N	$m,\ i=0,1,\ldots,\ (N-1)$
cDCT-VI cDCT-2o	$[\mathcal{C}_{2o}]_{\mathrm{m,i}}=2\zeta^2(i)\cos\left(\frac{m(i+\frac{1}{2})\pi}{N-\frac{1}{2}}\right)$	N	$m,\ i=0,1,\ldots,\ (N-1)$
cDCT-VII cDCT-3o	$[\mathcal{C}_{3o}]_{\mathrm{m,i}}=2\xi^2(i)\cos\left(\frac{m(m+\frac{1}{2})i\pi}{N-\frac{1}{2}}\right)$	N	$m,\ i=0,1,\ldots,\ (N-1)$
cDCT-VIII cDCT-4o	$[\mathcal{C}_{4o}]_{\mathrm{m,i}}=2\cos\left(\frac{(m+\frac{1}{2})(i+\frac{1}{2})\pi}{N-\frac{1}{2}}\right)$	$(N-1)$	$m,\ i=0,1,\ldots,\ (N-2)$
cDST-I cDST-1e	$[\mathcal{S}_{1e}]_{\mathrm{m,i}}=2\sin\left(\frac{mi\pi}{N}\right)$	$(N-1)$	$m,\ i=1,2,\ldots,\ (N-1)$
cDST-II cDST-2e	$[\mathcal{S}_{2e}]_{\mathrm{m,i}}=2\sin\left(\frac{m(i+\frac{1}{2})\pi}{N}\right)$	N	$m=1,2,\ldots,N,\ i=0,1,\ldots,\ (N-1)$
cDST-III cDST-3e	$[\mathcal{S}_{3e}]_{\mathrm{m,i}}=2\xi^2(i)\sin\left(\frac{m(i+\frac{1}{2})i\pi}{N}\right)$	N	$m,\ i=0,1,\ldots,\ (N-1),\ i=1,2,\ldots,N$
cDST-IV cDST-4e	$[\mathcal{S}_{4e}]_{\mathrm{m,i}}=2\sin\left(\frac{(m+\frac{1}{2})(i+\frac{1}{2})\pi}{N}\right)$	N	$m,\ i=0,1,\ldots,\ (N-1)$
cDST-V cDST-1o	$[\mathcal{S}_{1o}]_{\mathrm{m,i}}=2\sin\left(\frac{mi\pi}{N-\frac{1}{2}}\right)$	$(N-1)$	$m,\ i=1,2,\ldots,\ (N-1)$
cDST-VI cDST-2o	$[\mathcal{S}_{2o}]_{\mathrm{m,i}}=2\sin\left(\frac{m(i+\frac{1}{2})\pi}{N-\frac{1}{2}}\right)$	$(N-1)$	$m=1,2,\ldots,\ (N-1),\ i=0,1,\ldots,\ (N-2)$
cDST-VII cDST-3o	$[\mathcal{S}_{3o}]_{\mathrm{m,i}}=2\sin\left(\frac{m(i+\frac{1}{2})i\pi}{N-\frac{1}{2}}\right)$	$(N-1)$	$m,\ i=0,1,\ldots,\ (N-2),\ i=1,2,\ldots,\ (N-1)$
cDST-VIII cDST-4o	$[\mathcal{S}_{4o}]_{\mathrm{m,i}}=2\zeta^2(i)\sin\left(\frac{(m+\frac{1}{2})(i+\frac{1}{2})\pi}{N-\frac{1}{2}}\right)$	N	$m,\ i=0,1,\ldots,\ (N-1)$

[a] A prefix ‘c’ has been appended to the denomination of each transform to distinguish these *convolution* definitions from the *orthogonal* definitions defined in Tables 1 and 2.

The relation stated by the formulation of *Class* 1 is particularly useful. By considering, as one of its inputs, a HSHS symmetric extension to produce another HSHS sequence, it can be shown that it provides the means to compute the result of a symmetric convolution operation of two pixel-domain input sequences by simply element-by-element multiplying the corresponding DCT-domain coefficients. By applying such property to the formulation stated in Eq. (51), the following relation is obtained:

$$\mathbf{\Omega}^{\mathrm{HSHS}}=\mathbf{\Phi}^{\mathrm{HSHS}} \circledS \mathbf{\Psi}^{\mathrm{WSWS}}=\mathcal{C}_{\mathbf{2e}}^{-1}\{\mathcal{C}_{\mathbf{2e}}\{\mathbf{\Phi}\}\odot\mathcal{C}_{\mathbf{1e}}\{\mathbf{\Psi}\}\} \tag{54}$$

Table 6
Multiplication-convolution properties of DTT extensions applied in encoded image and video processing.

ε_a	ε_b	Output extension	$\circledast$	Input index ranges		Output index range	τ_a	τ_b	τ_c
				Φ_n	Ψ_n				
HSHS	WSWS	HSHS	©	$0\rightarrow N-1$	$0\rightarrow N$	$0\rightarrow N-1$	$\mathcal{C}_{2e}$	$\mathcal{C}_{1e}$	$\mathcal{C}_{2e}$
WSWA	WSWA	WSWA	Ⓢ	$0\rightarrow N-1$	$0\rightarrow N-1$	$0\rightarrow N-1$	$\mathcal{C}_{3e}$	$\mathcal{C}_{3e}$	$\mathcal{C}_{3e}$

If we denote the generic data sequences $\mathbf{\Omega}$ and $\mathbf{\Phi}$ by the half-sample symmetrically extended sequences of pixels $h(i)$ and $f(i)$ and the data sequence $\mathbf{\Psi}$ by the whole-sample symmetrically extended sequence of pixels $g(i)$, the type-II DCT coefficients $H(m)$ (corresponding to the pixels vector $h(i)$) that are obtained by the symmetric convolution operation of sequences $f(i)$ and $g(i)$, can be computed by the element-by-element multiplication of the corresponding convolution form of the even type-II DCT coefficients $F(m)$ and the even type-I DCT coefficients $G(m)$ of sequences $f(i)$ and $g(i)$, respectively:

$$\begin{aligned} H(m)^{\mathrm{WSWA}} &= \mathcal{C}_{\mathbf{2e}}\{h(i)^{\mathrm{HSHS}}\} = \mathcal{C}_{\mathbf{2e}}\{f^{\mathrm{HSHS}}(i) \circledcirc g^{\mathrm{WSWS}}(m)\} \\ &= \mathcal{C}_{\mathbf{2e}}\{f(i)\} \odot \mathcal{C}_{\mathbf{1e}}\{g(i)\} \\ &= F(m) \odot G(m) \end{aligned} \tag{55}$$

As it will be shown in section 6, this property allows the implementation of linear filtering directly in the transform-domain, by considering a pre-computed matrix filter given by $G(m) = \mathcal{C}_{\mathbf{1e}}\{g(i)\}$.

The relation stated by the formulation of *Class* 2 is also particularly useful. By relating two input WSWA symmetric extensions to produce one WSWA extension, it can be shown that it provides the means to implement an element-by-element multiplication of two pixel-domain sequences by performing a symmetric convolution of the corresponding DCT-domain coefficients. In fact, by applying such property to the formulation stated in Eq. (51), the following relation is obtained:

$$\mathbf{\Omega}^{\mathrm{WSWA}} = \mathbf{\Phi}^{\mathrm{WSWA}} \circledS \mathbf{\Psi}^{\mathrm{WSWA}} = \mathcal{C}_{\mathbf{3e}}^{-1}\{\mathcal{C}_{\mathbf{3e}}\{\mathbf{\Phi}\} \odot \mathcal{C}_{\mathbf{3e}}\{\mathbf{\Psi}\}\} \tag{56}$$

By recalling the relation between the forward and inverse definitions of the $\mathcal{C}_{\mathbf{2e}}$ and the $\mathcal{C}_{\mathbf{3e}}$ transform kernels, stated in Eqs. (19) and (20), the above equation comes as follows:

$$\mathbf{\Omega}^{\mathrm{WSWA}} = \mathbf{\Phi}^{\mathrm{WSWA}} \circledS \mathbf{\Psi}^{\mathrm{WSWA}} = \mathcal{C}_{\mathbf{2e}}\{\mathcal{C}_{\mathbf{2e}}^{-1}\{\mathbf{\Phi}\} \odot \mathcal{C}_{\mathbf{2e}}^{-1}\{\mathbf{\Psi}\}\} \tag{57}$$

Under this assumption, if we denote the generic WSWA data sequences $\mathbf{\Phi}$, $\mathbf{\Psi}$ and $\mathbf{\Omega}$ by the corresponding convolution form of the even type-II DCT coefficients $F(m)$, $G(m)$ and $H(m)$, it comes

$$H(m)^{\mathrm{WSWA}} = F^{\mathrm{WSWA}}(m) \circledS G^{\mathrm{WSWA}}(m) = \mathcal{C}_{\mathbf{2e}}\{f(i) \odot g(i)\} \tag{58}$$

According to Eq. (58), the DCT coefficients $H(m)$ (corresponding to the pixels vector $h(i)$) that are obtained by the element-by-element multiplication of the pixels sequences $f(i)$ and $g(i)$, can be computed with a skew-circular convolution operation of the DCT coefficients $F(m)$ and $G(m)$, corresponding to the pixel sequences $f(i)$ and $g(i)$, respectively.

Chang and Messerschmitt [32] presented an equivalent formulation of this definition that directly operates with the orthogonal form of the even type-II DCT (DCT-IIe) of the input signals. According to such formulation, the DCT of $h(i)$ can be computed by applying the symmetric convolution operation to the DCT coefficients of the two length-N WSWA sequences $\mathbf{F} = \text{DCT-IIe}(\mathbf{f})$ and $\mathbf{G} = \text{DCT-IIe}(\mathbf{g})$, as defined as follows:

$$H(m) = F(m) \circledast G(m) = W_N(m)(\tilde{F}(m) \circledS \tilde{G}(m)) \tag{59}$$

The vectors $\tilde{F}(m)$ and $\tilde{G}(m)$ correspond to symmetric length-$2N$ WSWA extended sequences of $F(m)$ and $G(m)$, defined as

$$\tilde{X}(m) = \begin{cases} 0, & m = 0 \\ \hat{X}(N-m), & m = 1\ldots(N-1) \\ \hat{X}(m-N), & m = N\ldots(2N-1) \end{cases} \tag{60}$$

where $\hat{X}(m) = X(m)/\xi(m)$, with $X(m) = \text{DCT-IIe}[x(i)]$ and $\xi(m)$ as defined in Eq. (13). The skew-circular convolution Ⓢ, defined in Eq. (53), is computed as follows:

$$\tilde{F}(m) \circledS \tilde{G}(m) = \frac{1}{\sqrt{2N}}\xi(m)\left[\sum_{n=0}^{m} \tilde{F}(n)\tilde{G}(m-n) - \sum_{n=m+1}^{2N-1} \tilde{F}(n)\tilde{G}(m-n+2N)\right] \tag{61}$$

$$\tilde{F}(m) \circledS \tilde{G}(m) = \frac{1}{\sqrt{2N}}\xi(m)\left[\sum_{n=0}^{2N-1} \tilde{F}(n)\tilde{G}(\mathrm{mod}_{2N}(m-n))S(m-n)\right] \tag{62}$$

where

$$S(m-n) = \begin{cases} 1, & (m-n) \in [0,\ (2N-1)] \\ -1, & \text{otherwise} \end{cases} \tag{63}$$

and $W_N(m)$ is a length-N rectangular window, which is used to extract the representative samples out of the base period of the convolution result.

The extension of the above definition to the 2-D domain can be easily formulated as shown in Eq. (64), where $\xi(m)$ and $S(m)$ were defined in Eq. (13) and (63), respectively, and $\tilde{X}(m_1,m_2)$ is a $(2N \times 2N)$ symmetric WSWA extended sequence defined as shown in Eq. (65).

$$\begin{aligned} F(m_1,m_2) \circledS G(m_1,m_2) &= \frac{1}{2N}\xi(m_1)\xi(m_2) \\ &\times \left[\sum_{n_1=0}^{2N-1}\sum_{n_2=0}^{2N-1} \tilde{F}(n_1,n_2)\tilde{G}(\mathrm{mod}_{2N}(m_1-n_1),\right. \\ &\left. \mathrm{mod}_{2N}(m_2-n_2))S(m_1-n_1)S(m_2-n_2)\right] \end{aligned} \tag{64}$$

$$\tilde{X}(m_1,m_2) = \begin{cases} 0, & m_1=0 \text{ or } m_2=0 \\ \hat{X}(N-m_1,N-m_2), & m_1=1\ldots(N-1),\ m_2=1\ldots(N-1) \\ \hat{X}(m_1-N,N-m_2), & m_1=N\ldots(2N-1),\ m_2=1\ldots(N-1) \\ \hat{X}(N-m_1,m_2-N), & m_1=1\ldots(N-1),\ m_2=N\ldots(2N-1) \\ \hat{X}(m_1-N,m_2-N), & m_1,\ m_2=N\ldots(2N-1) \end{cases} \tag{65}$$

5.3. *Fast computation of the convolution operation in the DCT-domain*

By analyzing the definition of the convolution operation, expressed in Eq. (59)– (63), it can be observed that the total number of multiplications required to perform the convolution between two length-N sequences is proportional to $(2N)^2 = 4N^2$. By extending this analysis to the 2-D domain, one can conclude that $[(2N)^2]^2 = 16N^4$ multiplications are required to evaluate the convolution between two $(N \times N)$ bidimensional sequences (see Eq. (64)).

To avoid such computational burden, Shen et al. [33] proposed a different approach in order to compute the DCT-domain convolution operation, by exploiting its symmetry and orthogonality properties. They started their formulation from the length-N 1-D DCT-II transform definition, as follows:

$$X(m) = \sum_{i=0}^{N-1} C_{2e}(m,i)x(i) \tag{66}$$

where $C_{2e}(m,i) = \sqrt{(2/N)}\xi(m)\cos(m(i+\frac{1}{2})\pi/N)$, defined in Table 2, with $\xi(m)$ defined in Eq. (13). Each element of the pixel-domain sequence $x(i)$ can be reconstructed using the inverse discrete cosine transform (IDCT) as follows:

$$x(i) = \sum_{m=0}^{N-1} C_{2e}(m,i)X(m) \tag{67}$$

Hence, each $h(i)$ value, obtained by the element-by-element multiplication between $f(i)$ and $g(i)$, can be expressed as

$$h(i) = f(i) \odot g(i) = \sum_{m_1=0}^{N-1}\sum_{m_2=0}^{N-1} C_{2e}(m_1,i)C_{2e}(m_2,i)F(m_1)G(m_2) \tag{68}$$

where vectors **F** and **G** are the discrete cosine transforms of the pixel-domain sequences **f** and **g**, respectively. Since only these transform data sequences are actually available from an image or video compressed bitstream, there is a considerable interest in computing **H** directly from **F** and **G**. The discrete cosine transform of $h(i)$, expressed as $H(m)$, is stated from Eq. (66) as follows:

$$H(m) = \sum_{i=0}^{N-1} C_{2e}(m,i)h(i) \tag{69}$$

$$H(m) = \sum_{i=0}^{N-1} C_{2e}(m,i)\left(\sum_{m_1=0}^{N-1}\sum_{m_2=0}^{N-1} C_{2e}(m_1,i)C_{2e}(m_2,i)F(m_1)G(m_2)\right) \tag{70}$$

By performing some simple manipulations, Eq. (70) can be expressed as

$$H(m) = \sum_{m_1=0}^{N-1}\sum_{m_2=0}^{N-1} W(m,m_1,m_2)F(m_1)G(m_2) \tag{71}$$

where

$$W(m,m_1,m_2) = \sum_{i=0}^{N-1} C_{2e}(m,i)C_{2e}(m_1,i)C_{2e}(m_2,i) \tag{72}$$

Hence, Eq. (71) expresses the DCT-domain convolution operation corresponding to the spatial-domain pixel-wise multiplication. Nevertheless, by comparing these two approaches in what concerns the computational cost, the pixel-domain approach seems to require significantly less operations ($\mathcal{O}(N)$) than the DCT-domain counterpart, which implies a computational cost proportional to $\mathcal{O}(N^3)$ (disregarding the cost of performing two IDCTs and one direct DCT in the pixel-domain approach). However, by considering that many high-frequency DCT coefficients of most compressed image and video streams are zero, the DCT-domain convolution between **F** and **G** requires, in practice, only $N_F \times N_G \times N$ multiplications, where N_F and N_G represent the number of nonzero coefficients in **F** and **G**, respectively. Consequently, since only the nonzero DCT coefficients actually need to be used in the convolution process, the convolution operation in the DCT-domain may have, in practice, a lower computational cost than the spatial-domain approach.

Besides these observations, Shen et al. [33] have also shown that the computation cost inherent to this operation can still be significantly reduced if the sparse nature of the $W(m,m_1,m_2)$ matrix is taken into account. They proved that each cross product between any two arbitrary elements of **F** and **G** (e.g., $F(m_1)$ and $G(m_2)$), contributes to no more than two different elements $H(a)$ and $H(b)$ of the resulting DCT output vector **H**, where these indexes a and b are given by

$$a = \begin{cases} m_1+m_2 & \text{if } m_1+m_2 < N \\ 2N-(m_1+m_2) & \text{if } m_1+m_2 > N \\ \emptyset \text{ (empty)} & \text{if } m_1+m_2 = N \end{cases} \tag{73}$$

and

$$b = |m_1-m_2| \tag{74}$$

The proof of this statement comes from the orthogonality property of the DCT, formulated as

$$\sum_{m=0}^{N-1} C_{2e}(m,i)\cdot C_{2e}(m,j) = \begin{cases} 1, & i=j \\ 0, & i\neq j \end{cases} \tag{75}$$

In fact, from Eq. (72),

$$W(m,m_1,m_2) = \sum_{i=0}^{N-1} C_{2e}(m,i)[C_{2e}(m_1,i)C_{2e}(m_2,i)] \tag{76}$$

By using simple trigonometric relations, it can be shown that the term $C_{2e}(m_1,i)C_{2e}(m_2,i)$ of the previous equation can be formulated as follows:

$$C_{2e}(m_1,i)C_{2e}(m_2,i) = \sqrt{\frac{2}{N}}\xi(m_1)\cos\left(\frac{m_1(i+\frac{1}{2})\pi}{N}\right) \times \sqrt{\frac{2}{N}}\xi(m_2)\cos\left(\frac{m_2(i+\frac{1}{2})\pi}{N}\right) \tag{77}$$

$$\mathrm{C}_{2e}(m_1,i)\mathrm{C}_{2e}(m_2,i) = \frac{\xi(m_1)\xi(m_2)}{N}\cos\left(\frac{(m_1+m_2)(i+\frac{1}{2})\pi}{N}\right) + \frac{\xi(m_1)\xi(m_2)}{N}\cos\left(\frac{(m_1-m_2)(i+\frac{1}{2})\pi}{N}\right) \tag{78}$$

$$\mathrm{C}_{2e}(m_1,i)\mathrm{C}_{2e}(m_2,i) = \frac{\xi(m_1)\xi(m_2)}{N}\left[\frac{\mathrm{C}_{2e}(m_1+m_2,i)}{\sqrt{\frac{2}{N}}\xi(m_1+m_2)} + \frac{\mathrm{C}_{2e}(m_1-m_2,i)}{\sqrt{\frac{2}{N}}\xi(m_1-m_2)}\right] \tag{79}$$

$$\mathrm{C}_{2e}(m_1,i)\mathrm{C}_{2e}(m_2,i) = \sqrt{\frac{1}{2N}}\frac{\xi(m_1)\xi(m_2)}{\xi(m_1+m_2)}\mathrm{C}_{2e}(m_1+m_2,i) + \sqrt{\frac{1}{2N}}\frac{\xi(m_1)\xi(m_2)}{\xi(m_1-m_2)}\mathrm{C}_{2e}(m_1-m_2,i) \tag{80}$$

Hence, Eq. (76) can be re-written as

$$W(m,m_1,m_2) = \sqrt{\frac{1}{2N}}\frac{\xi(m_1)\xi(m_2)}{\xi(m_1+m_2)}\sum_{i=0}^{N-1}\mathrm{C}_{2e}(m,i)\mathrm{C}_{2e}(m_1+m_2,i) + \sqrt{\frac{1}{2N}}\frac{\xi(m_1)\xi(m_2)}{\xi(m_1-m_2)}\sum_{i=0}^{N-1}\mathrm{C}_{2e}(m,i)\mathrm{C}_{2e}(m_1-m_2,i) \tag{81}$$

By taking Eq. (75) into account, one concludes that the first term on the right side of Eq. (81) is non-null only when $m=m_1+m_2$. Similarly, the second term will be nonzero when $m=m_1-m_2$. However, since $\cos(m_1-m_2)=\cos(m_2-m_1)$, one realizes that this term will be nonzero whenever $m=|m_1-m_2|$. A detailed proof of this formulation can be found in [33].

Shen et al. also proposed a rather efficient algorithm for computing $H(m)$, by observing that the entries in $W(m,m_1,m_2)$ take only three possible nonzero values: $\sqrt{1/N}$ and $\pm\sqrt{1/2N}$.

(1) If $m_1=0$ *and* $m_2=0$ then $m=(m_1+m_2)=|m_1-m_2|=0$. From Eqs. (75) and (81), it comes that

$$W(m,m_1,m_2) = \frac{1}{\sqrt{2N}}\;\frac{\sqrt{1/2}\cdot\sqrt{1/2}}{\sqrt{1/2}}\cdot 1 + \frac{1}{\sqrt{2N}}\;\frac{\sqrt{1/2}\cdot\sqrt{1/2}}{\sqrt{1/2}}\cdot 1 \tag{82}$$

$$W(m,m_1,m_2) = 2\frac{1}{\sqrt{2N}}\sqrt{1/2} = \sqrt{\frac{1}{N}} \tag{83}$$

(2) If $m_1=0$ *or* $m_2=0$, it comes that $m=(m_1+m_2)=|m_1-m_2|\neq 0$ and

$$W(m,m_1,m_2) = \frac{1}{\sqrt{2N}}\;\frac{1\cdot\sqrt{1/2}}{1}\cdot 1 + \frac{1}{\sqrt{2N}}\;\frac{1\cdot\sqrt{1/2}}{1}\cdot 1 = \sqrt{\frac{1}{N}} \tag{84}$$

(3) If $m_1\neq 0$ *and* $m_2\neq 0$, it comes that $(m_1+m_2)\neq|m_1-m_2|$. Hence, two different cases should be taken into account: $m=a$ or $m=b$ (see Eqs. (73) and (74)):

$\underline{m=a:}$

- If $m=(m_1+m_2)<N$, then

$$W(m,m_1,m_2) = \frac{1}{\sqrt{2N}}\;\frac{1.1}{1}\cdot 1 + 0 = \sqrt{\frac{1}{2N}} \tag{85}$$

- If $(m_1+m_2)>N$, then $m=2N-(m_1+m_2)$; considering that $\mathrm{C}_{2e}(m_1+m_2,i)=-\mathrm{C}_{2e}(2N-(m_1+m_2),i)$, it comes

$$W(m,m_1,m_2) = \frac{1}{\sqrt{2N}}\;\frac{1.1}{1}\cdot(-1)+0 = -\sqrt{\frac{1}{2N}} \tag{86}$$

$\underline{m=b:}$

- If $m=|m_1-m_2|\neq 0$, then

$$W(m,m_1,m_2) = 0+\frac{1}{\sqrt{2N}}\;\frac{1.1}{1}\cdot 1 = \sqrt{\frac{1}{2N}} \tag{87}$$

- If $m=|m_1-m_2|=0$, then

$$W(m,m_1,m_2) = 0+\frac{1}{\sqrt{2N}}\;\frac{1.1}{\sqrt{1/2}}\cdot 1 = \sqrt{\frac{1}{2N}} \tag{88}$$

The weighting factors $W(a,m_1,m_2)$ and $W(b,m_1,m_2)$ for each cross product $F(m_1)G(m_2)$, applied in the computation of the two elements $H(a)$ and $H(b)$ of the resulting DCT output vector $H(m)$ (see Eqs. (73) and (74)) are shown in Table 7, with $\alpha=\sqrt{1/N}$ and $\beta=\sqrt{1/(2N)}$. As it was shown from the previous description, when either m_1 or m_2 are zero, the product $F(m_1)G(m_2)$ contributes to exactly one output component. Otherwise, at most two components are required in the evaluation.

By adopting this method, the computational cost is now reduced to $N_F\times N_G\times N_W$ multiplications, with N_W representing the number of nonzero $W(m,m_1,m_2)$ elements for each fixed m (1 or 2). Actually, this computational cost can be further reduced by exploiting the symmetry property of the DCT operation (in particular, the symmetry of the $W(m,m_1,m_2)$ matrix) in the mapping of m_1, m_2 to m, in order to reduce the number of multiplications required to compute Eq. (71). In fact, Eq. (71) can be computed as

Table 7
Weighting factors $W(a,m_1,m_2)$ and $W(b,m_1,m_2)$, with $N=8$, a and b given by Eqs. (73) and (74) and $\alpha=\sqrt{1/N}$ and $\beta=\sqrt{1/(2N)}$.

m_1	m_2							
	0	1	2	3	4	5	6	7
0	α, α	α, α	α, α	α, α	α, α	α, α	α, α	α, α
1	α, α	β, α	β, β	β, β	β, β	β, β	β, β	$-, \beta$
2	α, α	β, β	β, α	β, β	β, β	β, β	$-, \beta$	$-\beta, \alpha$
3	α, α	β, β	β, β	β, α	β, β	$-, \beta$	$-\beta, \alpha$	$-\beta, \alpha$
4	α, α	β, β	β, β	β, β	$-, \alpha$	$-\beta, \alpha$	$-\beta, \alpha$	$-\beta, \alpha$
5	α, α	β, β	β, β	$-, \beta$	$-\beta, \alpha$	$-\beta, \alpha$	$-\beta, \alpha$	$-\beta, \alpha$
6	α, α	β, β	$-, \beta$	$-\beta, \alpha$	$-\beta, \alpha$	$-\beta, \alpha$	$-\beta, \alpha$	$-\beta, \alpha$
7	α, α	$-, \beta$	$-\beta, \alpha$	$-\beta, \alpha$	$-\beta, \alpha$	$-\beta, \alpha$	$-\beta, \alpha$	$-\beta, \alpha$

follows [33]:

$$H(m) = \sum_{m_1=0}^{N-2} \sum_{m_2=m_1+1}^{N-1} W(m,m_1,m_2)[F(m_1)\,G(m_2) + F(m_2)\,G(m_1)] + \sum_{m_1=0}^{N-1} W(m,m_1,m_1)F(m_1)G(m_1) \tag{89}$$

Hence, if $F(m_1)G(m_2)$ and $F(m_2)G(m_1)$ are both nonzero, their multiplications with the weighting factor $W(m,m_1,m_2)$ can still be merged, thus reducing the amount of required multiplications.

By adopting an entirely similar procedure, the formulation described above to fasten the computation of the convolution operation in the DCT-domain can be easily extended to the 2-D case. It can be shown [33] that, in such a case, each element of the **W** matrix takes on one of five possible nonzero values: $1/N$, $\pm 1/(2N)$ and $\pm\sqrt{2}/(2N)$. Each cross product contributes to no more than four output values. Just like the 1-D case, this fast algorithm provides the means to significantly reduce the computational load; instead of $16N^4$ multiplications, only about $4N^4$ operations are now required.

6. Example applications of the DCT properties for image/video transcoding

Some of the most common applications of the several DCT properties that were formulated in the previous sections are frequently found in the implementation of transcoding architectures, in order to process pre-encoded image or video data directly in the compressed DCT-domain. With such schemes, not only is avoided (i) the calculation of the inverse DCT of each block, (ii) the implementation of the intended processing (in the pixel-domain), and (iii) the calculation of the transform operation over the resulting pixels block back to the DCT-domain; but they also frequently take advantage of the presence of a large number of null quantized DCT coefficients to heavily reduce the data manipulation rate. Furthermore, a quality improvement is often achieved mainly due to the absence of arithmetic round-off errors introduced by the DCT and IDCT computational blocks, leading to average peak signal to noise ratio (PSNR) gains as high as 1–2 dB.

In this section, three particular applications of the above stated properties of the even type-II DCT will be illustrated: *linear filtering*, *image masking/segmentation* and *video composition* in the transform domain.

6.1. Linear filtering

The multiplication-convolution property of the even type-II DCT provides a systematic way to convolve linear phase symmetric FIR filters with symmetrically extended pixel data [23]. According to Eq. (55), the element-by-element multiplication of the DCT coefficients $X(m)$ of a given image/video block by an appropriate filter matrix $H(m)$ is equivalent to the computation of a symmetric (circular) convolution operation between a half-sample symmetrically extended sequence $x^{\mathrm{HSHS}}(i)$ of the N-pixel data sequence $x(i)$ and a whole-sample symmetrically extended sequence $h^{\mathrm{WSWS}}(i)$ of the L-tap filter sequence $h(i)$ (see Fig. 7(a)):

$$Y(m) = X(m) \odot H(m) = \mathcal{C}_{\mathbf{2e}}\{x^{\mathrm{HSHS}}(i) \circledS h^{\mathrm{WSWS}}(i)\} \tag{90}$$

where $X(m)$ and $Y(m)$ are the coefficients of the convolution form type-II DCT (e.g., $X(m) = \mathcal{C}_{\mathbf{2e}}[x(i)]$). The length L of

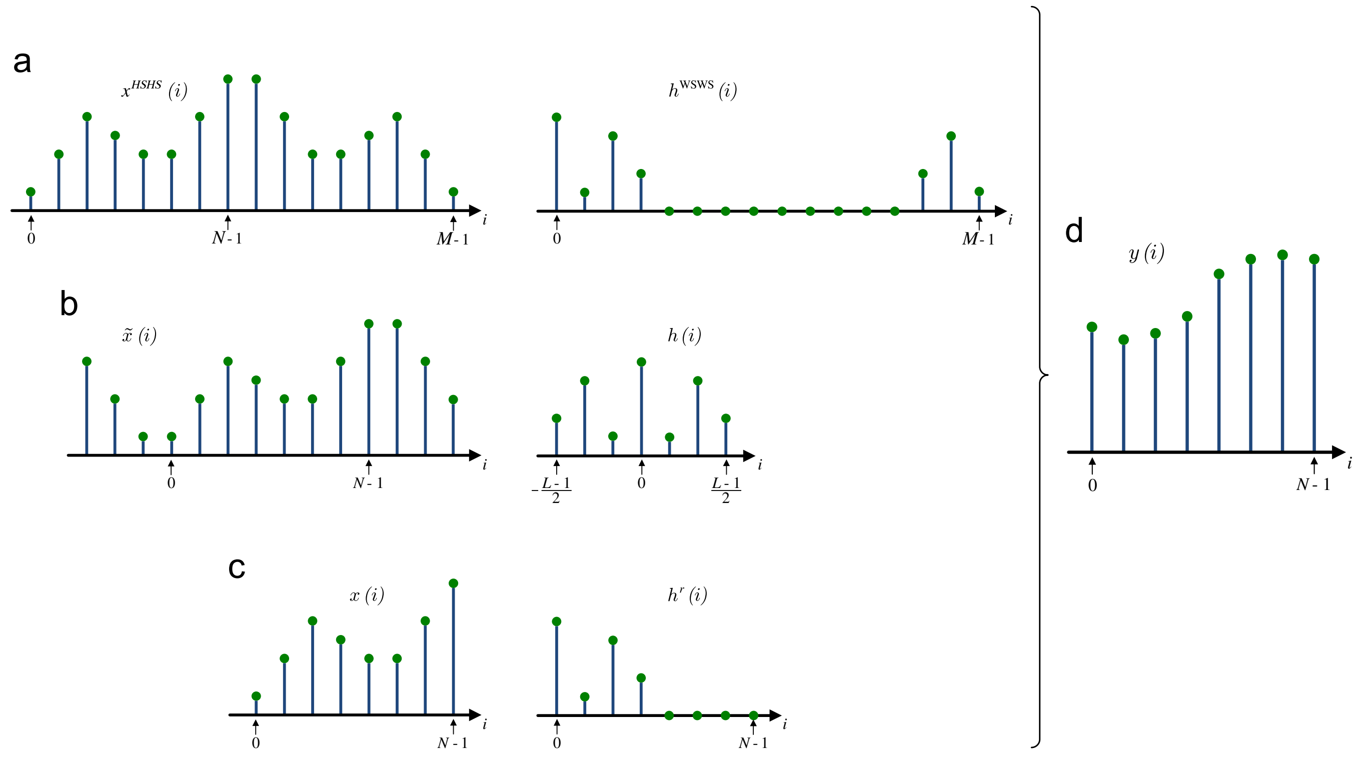


Fig. 7. Application of the multiplication–convolution property in the implementation of linear filtering in the transform-domain.

the effective filter must satisfy $L \leq M$, where $M = 2N$ is the period of the sequence being convolved $x^{\mathrm{HSHS}}(i)$, which means that $L/2 \leq N$.

According to this formulation [23], $h(i)$ can be regarded as an L-tap zero-phase FIR filter (spanning the interval $i = -(L-1)/2, \ldots, (L-1)/2$), which is convolved with $\tilde{x}(i)$, a HSHS symmetric extention of $x(i)$ performed at both of its ends, to the extent needed to perform the summation (see Fig. 7(b)):

$$y(i) = h(i) * \tilde{x}(i) = \sum_{k=-(L-1)/2}^{(L-1)/2} h(k)\tilde{x}(i-k) \tag{91}$$

with $i = 0, 1, \ldots, N-1$.

In practice, the matrix filter $H(m)$, used in the transform-domain element-by-element multiplication of Eq. (90), is obtained by computing the convolution form of the even type-I DCT of sequence $h^r(i)$, defined as the right-half of the filter sequence $h(i)$ (see Fig. 7(c)):

$$h^r(i) = \begin{cases} h(i), & i = 0, 1, \ldots, (L-1)/2 \\ 0, & i = (L+1)/2, \ldots, N-1 \end{cases} \tag{92}$$

Nevertheless, a particular attention should be taken to the involved index ranges. Since the input and output index ranges of the N-point $\mathcal{C}_{\mathbf{1e}}$ are both $i, m = 0, \ldots, N$ (see Tables 5 and 6), an appropriate adjustment of its index ranges to the interval $0, \ldots, (N-1)$ should be considered in the implementation of the transform-domain operation:

$$Y(m) = X(m) \odot [\widehat{\mathcal{C}_{\mathbf{1e}}}\ \mathbf{h}^r] \tag{93}$$

where $\widehat{\mathcal{C}_{\mathbf{1e}}}$ is the truncation of the matrix kernel $\mathcal{C}_{\mathbf{1e}}$ to index ranges $m, i = 0, \ldots, (N-1)$. It is also worth noting that although the $\mathcal{C}_{\mathbf{1e}}$ transform is not commonly adopted by current image and video standards, $H(m)$ corresponds to a fixed filter matrix, which can be pre-computed and stored in memory.

Fig. 7 exemplifies the application of this method in the 1-D domain, considering $N=8$ and $L=7$. In Fig. 8, it is illustrated the usage of this multiplication-convolution property in the scope of image processing in the compressed domain. In particular, it is depicted the implementation of a low-pass FIR filtering scheme (with impulse response defined by $h(i,j)$) as a simple element-by-element multiplication of the DCT encoded blocks and the corresponding filter DCT coefficients ($H^r(m,n)$).

Kresch and Merhav [30] presented an extensive study about the application of this formulation by using matrix diagonalization properties that may be applied not only for symmetric and anti-symmetric filtering, but also for the implementation of non-symmetric filters (see [30] for more details).

6.2. *Image masking and segmentation*

Another application of the multiplication-convolution property is found in the implementation of masking schemes directly in the transform-domain, applied in transcoding structures for the insertion of non-regular shaped visible objects (such as logos and subtitles) in pre-encoded image and video sequences [21].

When stated in the pixel-domain, object insertion can be performed by combining the pixels of the background scene $b(i,j)$ with the object $\ell(i,j)$ to obtain the output image $f(i,j)$. This operation is usually expressed as a linear combination of the form

$$f(i,j) = [\alpha\theta(i,j)] \odot \ell(i,j) + [1-\alpha\theta(i,j)] \odot b(i,j) \tag{94}$$

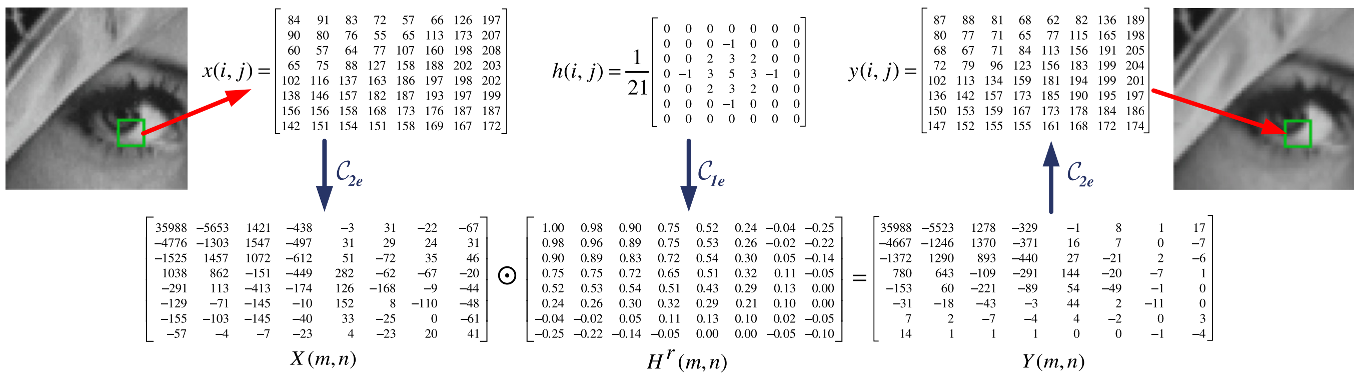
where α denotes the transparency level, $\odot$ represents the element-by-element multiplication and $\theta(i,j)$ is a segmentation mask, required to isolate the pixels corresponding to the background scene from the pixels of the object to be inserted, and defined as $\theta(i,j) = 1$ for (i,j) in the foreground object area and set to zero otherwise. Hence, Eq. (94) becomes

$$f(i,j) = \phi(i,j) + \psi(i,j) \odot b(i,j) \tag{95}$$

In this equation, $\phi(i,j)$ and $\psi(i,j)$ represent constant matrices that are solely dependent on the considered foreground object. Consequently, they can be pre-computed and stored in memory.

This pixel-domain insertion algorithm can be directly applied in the compressed domain by using the previously stated linearity and multiplication–convolution properties of the DCT. In fact, since $[\alpha\theta(i,j)]$ and $[1-\alpha\theta(i,j)]$ of Eq. (94) are not scalars, the element-by-element multiplications will have to be replaced by skew-circular convolutions in the DCT-domain. Consequently, the application of Eq. (95) in the compressed DCT-domain is stated as follows:

$$F(m,n) = \Phi(m,n) + \Psi(m,n) \circledS B(m,n) \tag{96}$$



$$x(i,j) = \begin{bmatrix} 84 & 91 & 83 & 72 & 57 & 66 & 126 & 197 \\ 90 & 80 & 76 & 55 & 65 & 113 & 173 & 207 \\ 60 & 57 & 64 & 77 & 107 & 160 & 198 & 208 \\ 65 & 75 & 88 & 127 & 158 & 188 & 202 & 203 \\ 102 & 116 & 137 & 163 & 186 & 197 & 198 & 202 \\ 138 & 146 & 157 & 182 & 187 & 193 & 197 & 199 \\ 156 & 156 & 158 & 168 & 173 & 176 & 187 & 187 \\ 142 & 151 & 154 & 151 & 158 & 169 & 167 & 172 \end{bmatrix} \quad h(i,j) = \frac{1}{21}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 3 & 2 & 0 & 0 \\ 0 & -1 & 3 & 5 & 3 & -1 & 0 \\ 0 & 0 & 2 & 3 & 2 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad y(i,j) = \begin{bmatrix} 87 & 88 & 81 & 68 & 62 & 82 & 136 & 189 \\ 80 & 77 & 71 & 65 & 77 & 115 & 165 & 198 \\ 68 & 67 & 71 & 84 & 113 & 156 & 191 & 205 \\ 72 & 79 & 96 & 123 & 156 & 183 & 199 & 204 \\ 102 & 113 & 134 & 159 & 181 & 194 & 199 & 201 \\ 136 & 142 & 157 & 173 & 185 & 190 & 195 & 197 \\ 150 & 153 & 159 & 167 & 173 & 178 & 184 & 186 \\ 147 & 152 & 155 & 155 & 161 & 168 & 172 & 174 \end{bmatrix}$$

$$\begin{tikzcd} x(i,j) \arrow[d, "\mathcal{C}_{2e}"] & h(i,j) \arrow[d, "\mathcal{C}_{1e}"] & y(i,j) \\ X(m,n) & H^r(m,n) & Y(m,n) \arrow[u, "\mathcal{C}_{2e}"] \end{tikzcd}$$

$$\underbrace{\begin{bmatrix} 35988 & -5653 & 1421 & -438 & -3 & 31 & -22 & -67 \\ -4776 & -1303 & 1547 & -497 & 31 & 29 & 24 & 31 \\ -1525 & 1457 & 1072 & -612 & 51 & -72 & 35 & 46 \\ 1038 & 862 & -151 & -449 & 282 & -62 & -67 & -20 \\ -291 & 113 & -413 & -174 & 126 & -168 & -9 & -44 \\ -129 & -71 & -145 & -10 & 152 & 8 & -110 & -48 \\ -155 & -103 & -145 & -40 & 33 & -25 & 0 & -61 \\ -57 & -4 & -7 & -23 & 4 & -23 & 20 & 41 \end{bmatrix}}_{X(m,n)} \odot \underbrace{\begin{bmatrix} 1.00 & 0.98 & 0.90 & 0.75 & 0.52 & 0.24 & -0.04 & -0.25 \\ 0.98 & 0.96 & 0.89 & 0.75 & 0.53 & 0.26 & -0.02 & -0.22 \\ 0.90 & 0.89 & 0.83 & 0.72 & 0.54 & 0.30 & 0.05 & -0.14 \\ 0.75 & 0.75 & 0.72 & 0.65 & 0.51 & 0.32 & 0.11 & -0.05 \\ 0.52 & 0.53 & 0.54 & 0.51 & 0.43 & 0.29 & 0.13 & 0.00 \\ 0.24 & 0.26 & 0.30 & 0.32 & 0.29 & 0.21 & 0.10 & 0.00 \\ -0.04 & -0.02 & 0.05 & 0.11 & 0.13 & 0.10 & 0.02 & -0.05 \\ -0.25 & -0.22 & -0.14 & -0.05 & 0.00 & 0.00 & -0.05 & -0.10 \end{bmatrix}}_{H^r(m,n)} = \underbrace{\begin{bmatrix} 35988 & -5523 & 1278 & -329 & -1 & 8 & 1 & 17 \\ -4667 & -1246 & 1370 & -371 & 16 & 7 & 0 & -7 \\ -1372 & 1290 & 893 & -440 & 27 & -21 & 2 & -6 \\ 780 & 643 & -109 & -291 & 144 & -20 & -7 & 1 \\ -153 & 60 & -221 & -89 & 54 & -49 & -1 & 0 \\ -31 & -18 & -43 & -3 & 44 & 2 & -11 & 0 \\ 7 & 2 & -7 & -4 & 4 & -2 & 0 & 3 \\ 14 & 1 & 1 & 1 & 0 & 0 & -1 & -4 \end{bmatrix}}_{Y(m,n)}$$

Fig. 8. Application of the multiplication–convolution property to implement a compressed DCT-domain low-pass FIR filtering scheme.

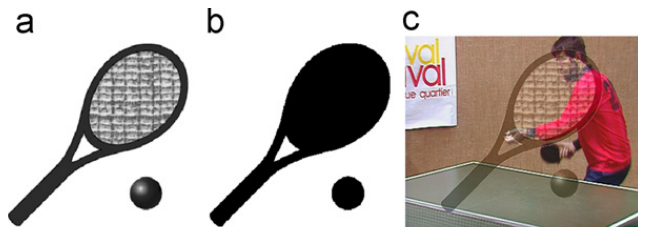


Fig. 9. Transform-domain object insertion. (a) Foreground object $\ell(i,j)$. (b) Segmentation mask $\theta(i,j)$. (c) Output scene.

where $F = \mathrm{DCT}(f)$, $\Phi = \mathrm{DCT}(\phi)$, $\Psi = \mathrm{DCT}(\psi)$, $B = \mathrm{DCT}(b)$ and the convolution operation Ⓢ is implemented as stated in Eq. (64). In Fig. 9(c), it is illustrated the result of the application of this method, considering the foreground object illustrated in Fig. 9(a) using $\alpha = 0.5$.

6.3. Video composition

Picture-in-picture (PIP) and picture-and-picture (PAP) video composition schemes, where two or more sequences obtained from multiple video sources are combined into a single scene, either at the same scale, side-by-side (PAP), or by scaling all but one of the sequences (foreground scenes) and inserting them over the background scene (PIP), are often required by surveillance systems, multipoint videoconferencing and interactive network video. In particular, the PAP compositing layout can be regarded as a particular case of the PIP layout (with unitary scaling factor). These manipulations can be implemented either at the client side or at the server side. Nevertheless, specialized network transcoding systems at the server side not only provide significant advantages in terms of bandwidth, but also allow the implementation of much simpler and cost effective receiver terminal devices. As a consequence, several different transcoding approaches, either in the pixel-domain or in the DCT-domain, have been proposed [34]. One possible architecture of a DCT-domain PIP/PAP compositing transcoder is presented in Fig. 10.

The composition of one or more foreground sequences with the background video sequence at arbitrary positions may lead to mismatches of the corresponding encoding block grids [32], as it is illustrated in Fig. 11. In such a situation, each $(N \times N)$ DCT coefficients block of the involved foreground scenes ($\mathbf{Y_i}$) has to be re-segmented and translated with respect to the block structure of the background scene ($\mathbf{X}$). The output block $\mathbf{B}$ will then contain the contributions from each original scene:

$$\mathbf{B} = \mathbf{X} - \sum_i \mathbf{X_{seg_i}} + \sum_i \mathbf{Y_{seg_i}} \tag{97}$$

The mathematical model to obtain the DCT coefficients of a given extracted and translated sub-block ($\mathbf{Y_{seg_i}}$) can be defined with the linearity properties described in Section 2.5 [32,34]. According to the example shown in Fig. 11(a):

$$\mathbf{X_{seg}} = \mathbf{H_1^x} \cdot \mathbf{X} \cdot \mathbf{H_2^x}, \quad \mathbf{Y_{seg}} = \mathbf{H_1^y} \cdot \mathbf{Y} \cdot \mathbf{H_2^y} \tag{98}$$

where $\mathbf{H} = \mathrm{DCT}(\mathbf{h})$ are pre-computed and stored matrices, with $\mathbf{h_1^x} = \begin{bmatrix} 0 & 0 \\ 0 & I_h \end{bmatrix}$ $\mathbf{h_2^x} = \begin{bmatrix} 0 & 0 \\ 0 & I_w \end{bmatrix}$, $\mathbf{h_1^y} = \begin{bmatrix} 0 & 0 \\ I_h & 0 \end{bmatrix}$, $\mathbf{h_2^y} = \begin{bmatrix} 0 & I_w \\ 0 & 0 \end{bmatrix}$. I_h and I_w are $(h \times h)$ and $(w \times w)$ identity matrices, where h and w are the number of rows and columns to be extracted, respectively.

In Fig. 11(b), it is presented the result of the application of this method by considering a single foreground video sequence that is scaled by a factor of 3 and positioned over a CIF format background sequence at coordinates $(l,c) = (11,223)$. This particular setup corresponds to a common layout used by many television applications [34].

7. Conclusion

In this tutorial it was presented a general overview about discrete trigonometric transforms, with a special emphasis to the presentation of the main properties and characteristics of the even type-II discrete cosine transform. This transform has been particularly adopted in image and video standards, due to its high suitability to exploit inter-pixel redundancies, rendering excellent decorrelation for most image data. Moreover, since this particular DCT efficiently packs the energy content in a reduced number of low-frequency coefficients, it also provides the capability to discard some high-frequency coefficients without significantly degrading the output

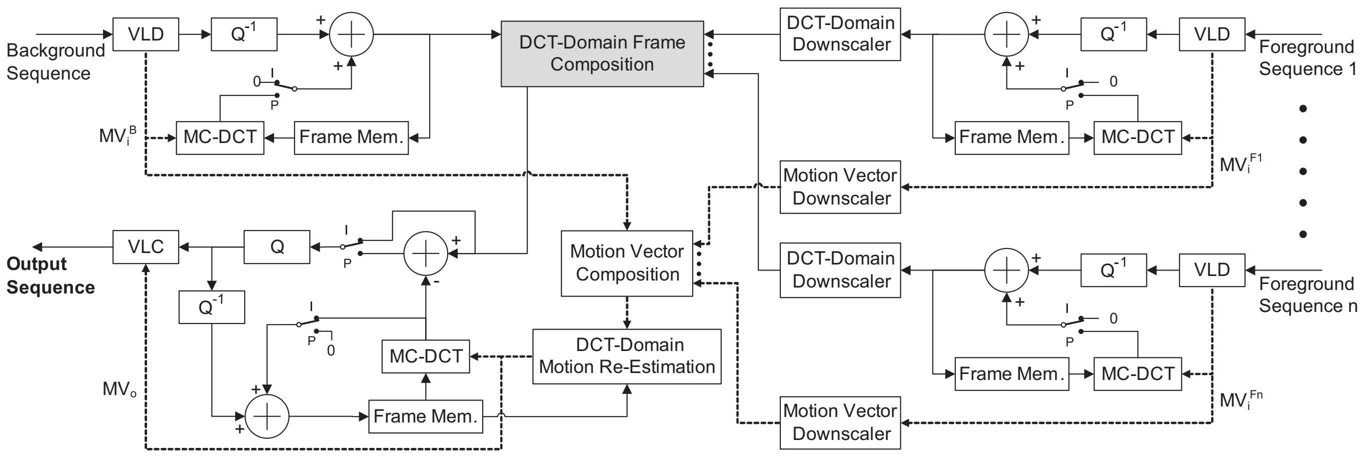
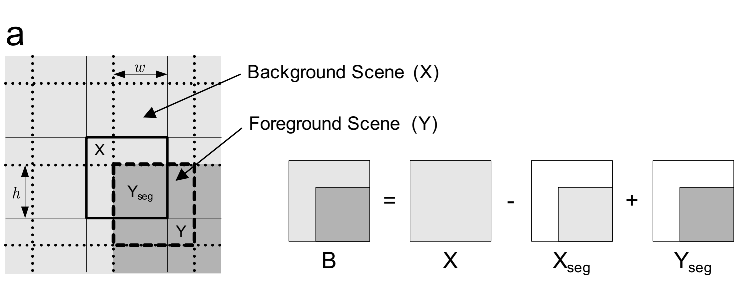


Fig. 10. DCT-domain PIP/PAP compositing transcoder architecture.



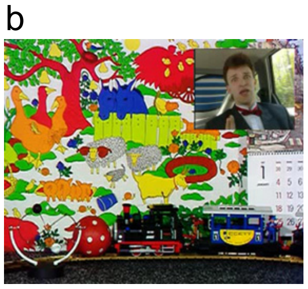


Fig. 11. DCT-domain PIP/PAP compositing. (a) Segmentation and translation layout. (b) PIP composition example.

image quality. With this implicit coarse quantization scheme, a significant reduction of the resulting signal entropy is usually provided, as well as a consequent reduction of the average number of required bits to encode each pixel.

By following the context usually adopted in the scope of encoded image and video processing, the formulation of many of the presented properties was based on the boundary characteristics of the data sequences under processing and on the implicit symmetric extensions of those sequences outside the original domain. Among the several properties that have been presented, a special attention was devoted to the formalization of the multiplication-convolution property. This property is often used by linear filtering [30] and static composition applications [21] and concerns the relation between the symmetric convolution operation, implemented in one trigonometric domain, and the corresponding element-by-element multiplication, implemented in the corresponding inverse trigonometric domain.

Acknowledgment

This work was supported by FCT (INESC-ID multi-annual funding) through the PIDDAC Program funds.

References

[1] H.261, ITU-T Recommendation H.261, Video codec for audiovisual services at $p \times 64$ Kbit/s, ITU-T, 1993.
[2] H.263, ITU-T Recommendation H.263, Video coding for low bitrate communication, ITU-T, 1998.
[3] MPEG-1, MPEG-1: ISO/IEC JTC1 CD 11172, Coding of moving pictures and associated audio for digital storage media up to 1.5 Mbit/s—part 2: video, ISO, 1992.
[4] MPEG-2, MPEG-2: ISO/IEC JTC1 CD 13818, Generic coding of moving pictures and associated audio—part 2: video, ISO, 1994.
[5] S.A. Khayam, The discrete cosine transform (DCT): theory and application, Technical Report WAVES-TR-ECE802.602, Wireless and Video Communications (WAVES) Lab at Michigan State University, 2003.
[6] W.K. Pratt, Digital Image Processing, John Wiley & Sons Inc, 1978.
[7] A.K. Jain, Fundamentals of Digital Image Processing, Prentice-Hall, 1989.
[8] J.S. Lim, Two-Dimensional Signal and Image Processing, Prentice-Hall, 1990.
[9] J.F. Blinn, What's the deal with the DCT?, IEEE Computer Graphics and Applications 13 (1993) 78–83.
[10] V. Britanak, P.C. Yip, K.R. Rao, Discrete Cosine and Sine Transforms: General Properties, Fast Algorithms and Integer Approximations, Academic Press, 2007.
[11] N. Ahmed, T. Natarajan, K.R. Rao, Discrete cosine transform, IEEE Transactions on Computers C-23 (1974) 90–93.
[12] Z. Wang, Fast algorithms for the discrete W transform and for the discrete Fourier transform, IEEE Transactions on Acoustics, Speech, and Signal Processing 32 (1984) 803–816.
[13] Z. Wang, B. Hunt, The discrete W transform, Applied Mathematics and Computation 16 (1985) 19–48.
[14] JPEG, JPEG: ITU-T Recommendation T.81, Digital compression and coding of continuous-tone still images, ITU-T, 1993.
[15] MPEG-4, MPEG-4: ISO/IEC 14496-2:2004. Information technology—coding of audio–visual objects—part 2: visual, ISO, 2004.
[16] W.K. Pratt, J. Kane, H.C. Andrews, Hadamard transform image coding, Proceedings of the IEEE 57 (1969) 58–68.
[17] W.K. Pratt, W.-H. Chen, L.R. Welch, Slant transform image coding, IEEE Transactions on Communications 22 (1974) 1075–1093.
[18] A. Haar, Zur theorie der orthogonalen funktionen-systeme [on the theory of orthogonal function systems], Mathematische Annalen (1910) 331–371.
[19] W.-K. Cham, Family of order-4 four-level orthogonal transforms, IEE Electronic Letters 19 (1983) 869–871.
[20] H.264, ITU-T Recommendation H.264, Advanced video coding for generic audiovisual services, ITU-T, 2003.
[21] N. Roma, L. Sousa, Fast transcoding architectures for insertion of non-regular shaped objects in the compressed DCT-domain, Signal Processing: Image Communication 18 (2003) 659–683.
[22] H.26L, ITU-T/SG16/VCEG (Q.6), H.26L test model long-term number 8 (TML -8), ITU-T, Video Coding Experts Group (VCEG), 2001.
[23] S.A. Martucci, Symmetric convolution and discrete sine and cosine transforms, IEEE Transactions on Signal Processing SP-42 (1994) 1038–1051.
[24] G. Strang, The discrete cosine transform, Society for Industrial and Applied Mathematic (SIAM Review) 41 (1999) 135–147.
[25] H. Hedberg, P. Nilsson, A survey of various discrete transforms used in digital image compression algorithms, in: Proceedings of the Swedish System-On-Chip Conference (SSoCC'04), Bastad—Sweden, pp. 1–5.
[26] V. Bhaskaran, K. Konstantinides, Image and Video Compression Standards: Algorithms and Architectures, second ed., Kluwer Academic Publishers, 1997.
[27] B. Porat, A Course in Digital Signal Processing, John Wiley & Sons Inc, 1997.
[28] H.R. Wu, Z. Man, Comments on "Fast algorithms and implementation of 2-D discrete cosine transform", IEEE Transactions on Circuits and Systems for Video Technology 8 (1998) 128–129.
[29] A. Obukhov, A. Kharlamov, Discrete cosine transform for 8×8 blocks with CUDA, Technical Report, nVIDIA, 2008.
[30] R. Kresch, N. Merhav, Fast DCT domain filtering using the DCT and the DST, IEEE Transactions on Image Processing 8 (1999) 821–833.
[31] B.C. Smith, L.A. Rowe, Algorithms for manipulating compressed images, IEEE Computer Graphics and Applications (1993) 34–42.
[32] S.-F. Chang, D.G. Messerschmitt, Manipulation and compositing of MC-DCT compressed video, IEEE Journal on Selected Areas in Communications 13 (1995) 1–11.
[33] B. Shen, I.K. Sethi, V. Bhaskaran, DCT convolution and its application in compressed domain, IEEE Transactions on Circuits and Systems for Video Technology 8 (1998) 947–952.
[34] N. Roma, L. Sousa, Fully compressed domain transcoder for PIP/PAP video composition, in: Proceedings of the Picture Coding Symposium (PCS'2007), 2007.