electronics

MDPI

Article

The Design of Fast Type-V Discrete Cosine Transform Algorithms for Short-Length Input Sequences

Marina Polyakova [1], Anna Witenberg [2,*] and Aleksandr Cariow [3,*]

[1] Institute of Computer Systems, Odesa Polytechnic National University, Shevchenko ave., 1, 65044 Odesa, Ukraine; polyakova@op.edu.ua
[2] Faculty of Telecommunications, Computer Science and Electrical Engineering, Bydgoszcz University of Science and Technology, Al. prof. S. Kaliskiego 7, 85-796 Bydgoszcz, Poland
[3] Faculty of Computer Science and Information Technology, West Pomeranian University of Technology in Szczecin, Żołnierska 49, 71-210 Szczecin, Poland
* Correspondence: anna.witenberg@pbs.edu.pl (A.W.); atariov@wi.zut.edu.pl (A.C.)

Abstract: Fast algorithms for type-five discrete cosine transform (DCT-V) for sequences of input data of short length in the range of two to eight are elaborated in the paper. A matrix–vector product representation of the DCT-V is the starting point for designing the algorithms. In each specific case, the DCT-V matrices have remarkable structural properties that follow from the localization of identical entries within the matrices. Each matrix of the DCT-V has only a few distinct entries that are repeated at different positions in its structure. Using simple transformations such as permutations of the rows and/or columns of this matrix or its favorable decomposition into two or more matrix components, it is possible to obtain efficient matrix structures that lead to useful factorization schemes. Based on the suitable factorization schemes we obtained, we developed fast algorithms that reduce the number of arithmetic operations when calculating the DCT-V. The correctness of the obtained algorithmic solutions was justified theoretically using a strict mathematical background of each of them. The developed algorithms were then further tested using MATLAB R2023b software to finally confirm their correctness. Finally, an evaluation of the computational complexity for each obtained solution is presented. The evaluation results were compared with the computational complexity of the direct calculation of matrix–vector products. The resulting factorizations of the matrices of the DCT-V reduce the average number of multiplications by 57% but increase the number of additions by 29%.

Keywords: discrete cosine transform; matrix factorization; fast algorithms; computational complexity; digital signal processing

Citation: Polyakova, M.; Witenberg, A.; Cariow, A. The Design of Fast Type-V Discrete Cosine Transform Algorithms for Short-Length Input Sequences. *Electronics* **2024**, *13*, 4165. https://doi.org/10.3390/electronics13214165

Received: 18 September 2024
Revised: 13 October 2024
Accepted: 22 October 2024
Published: 23 October 2024

1. Introduction

Discrete cosine transform (DCT) [1–3] is widely applied in image compression [4,5], image denoising [6–8], healthcare systems [9,10], steganography and image encryption [11,12], image watermarking and image authentication [13,14], data analysis [15], image retrieval [16], video coding [17], audio signal enhancement [18]. The advantages of this transform are real-valued processing of data compared with discrete Fourier transform. This reduces the number of calculations required and simplifies further analysis of the transform results. In the literature, eight types of DCT have been proposed [1–3]. The most popular and researched transform is still DCT types I-IV [19–21]. DCT variant types V-VIII seem to be rarely used in practice [17,22,23]. One reason, perhaps, is that corresponding algorithms are generally more complicated than algorithms for DCT types I-IV. There are, however, several works concerning the efficient implementation of DCT types I-IV [24–26]. This paper is devoted to designing computationally efficient algorithms for small-size DCT-V.

The publication review shows that two approaches to the elaboration of fast algorithms for DCT can be found in the literature, specifically, structural and analytical ones. The first approach is structural [27–30]. It is usually employed to develop fast algorithms for small-size discrete orthogonal transforms. It is based on a deep analysis of the structures of base transform matrices, identifying individual features of the arrangement of identical entries and, if necessary, changing the matrix structures for subsequent use of certain matrix identities that lead to the suitable factorization of these matrices.

The structural approach does not limit the length of the original data sequence, for example, to a power of two or three. However, as the size of data sets increases, fast algorithms become more cumbersome and difficult to implement. The second approach is analytical and is based on the use of the most general patterns of factorization of the basic transformation matrices of arbitrary order [31–33]. It allows the development of efficient algorithms for long-length input sequences. However, in the case of short-length sequences, this approach does not always give good results. The effectiveness of this method in constructing a fast algorithm usually depends on the availability of an analytical representation of the transform's coefficients. For the same reason, the length of the input data sequence is often limited to a power of two. Since short-length input sequences are considered in our research, the structural approach is selected to design computationally efficient algorithms. So, the aim of the article is the elaboration of reduced-complexity algorithms for DCT-V for short-length input sequences of length $N = 2, 3, 4, 5, 6, 7, 8$.

2. Materials and Methods

The DCT is one of the orthogonal transforms used, among other things, to analyze and process audio or other types of signals. DCT-V can be represented by the following expression [2,34]:

$$y_k = \frac{2}{\sqrt{2(N-1)+1}} \sum_{n=0}^{N-1} x_n \epsilon_n \epsilon_k \cos\frac{2n\pi k}{2N-1}, k = 0,\ 1,\ \ldots,\ N-1, \quad (1)$$

where y_k is the output sequence after the DCT-V transform is performed; x_n is the sequence of input data; and N is the number of signal samples,

$$\epsilon_n,\ \epsilon_k = \begin{cases} \frac{1}{\sqrt{2}}, n, k = 0, \\ 1, \text{otherwise.} \end{cases}$$

DCT-V can be represented in matrix notation as follows:

$$\mathbf{Y}_{N\times 1} = \mathbf{C}_N \mathbf{X}_{N\times 1}, \quad (2)$$

where $\mathbf{Y}_{N\times 1} = [y_0, y_1, \ldots, y_{N-1}]^{\mathbf{T}}$, $\mathbf{X}_{N\times 1} = [x_0, x_1, \ldots, x_{N-1}]^{\mathbf{T}}$, $\mathbf{c}_{kl} = \frac{2}{\sqrt{2(N-1)+1}} \epsilon_n \epsilon_k \cos\frac{2n\pi k}{2N-1}$, $k, l = 0, \ldots, N-1$.

In this paper, we use the following markings and signs:

- $\mathbf{I}_N$ is an order N identity matrix;
- $\mathbf{H}_2$ is a 2×2 Hadamard matrix;
- $1_{N\times M}$ is a $N \times M$ matrix of ones (a matrix where every entry is equal to one);
- $\otimes$ is the Kronecker product of two matrices;
- $\oplus$ is the direct sum of two matrices.

Voids in a matrix mean that the entries at those positions have a zero value. The multipliers were marked as $s_m^{(N)}$.

DCT-V in matrix notation is as follows:

$$\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{N-1} \end{bmatrix} = \begin{bmatrix} \mathbf{c}_{0,0} & \mathbf{c}_{0,1} & \cdots & \mathbf{c}_{0,N-1} \\ \mathbf{c}_{1,0} & \mathbf{c}_{1,1} & & \mathbf{c}_{1,N-1} \\ \vdots & & \ddots & \vdots \\ \mathbf{c}_{N-1,\,0} & \mathbf{c}_{N-1,\,1} & \cdots & \mathbf{c}_{N-1,N-1} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{bmatrix} \tag{3}$$

For a two-point DCT-V, the corresponding expression in matrix notation is represented as follows:

$$\mathbf{Y}_{2\times1} = \mathbf{C}_2\mathbf{X}_{2\times1}, \tag{4}$$

where $\mathbf{Y}_{2\times1} = [y_0, y_1]^{\mathbf{T}}$, $\mathbf{X}_{2\times1} = [x_0, x_1]^{\mathbf{T}}$, $\mathbf{C}_2 = \begin{bmatrix} \mathrm{a}_2 & \mathrm{b}_2 \\ \mathrm{b}_2 & -\mathrm{a}_2 \end{bmatrix}$, $a_2 = 0.5774$, $b_2 = 0.8165$.

Now, let us take into account the structural properties of the matrix $\mathbf{C}_2$ [18,27]. Then, the expression for DCT-V for $N = 2$ can be presented as follows:

$$\mathbf{Y}_{2\times1} = \mathbf{W}_{2\times3}\mathbf{D}_3\mathbf{W}_{3\times2}\mathbf{X}_{2\times1}, \tag{5}$$

where $\mathbf{D}_3 = diag\left(s_0^{(2)}, s_1^{(2)}, s_2^{(2)}\right)$, $s_0^{(2)} = a_2 - b_2, s_1^{(2)} = -(a_2 + b_2), s_2^{(2)} = b_2$, $\mathbf{W}_{2\times3} = \begin{bmatrix} 1 & & 1 \\ & 1 & 1 \end{bmatrix}$, $\mathbf{W}_{3\times2} = \begin{bmatrix} 1 & \\ & 1 \\ 1 & 1 \end{bmatrix}$.

Figure 1 shows a data flow graph of the synthesized algorithm for the two-point DCT-V. As can be seen, the number of multiplication operations may be reduced from 4 to 3, but the number of addition operations is increased from 2 to 3.

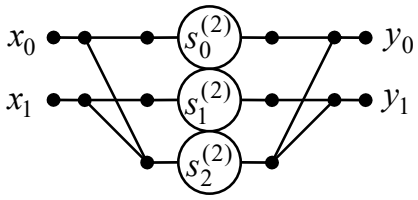

Figure 1. The data flow graph of the proposed algorithm for computation of two-point DCT-V.

Consider the elaboration of the algorithm for three-point DCT-V. The expression for three-point DCT-V is as follows:

$$\mathbf{Y}_{3\times1} = \mathbf{C}_3\mathbf{X}_{3\times1}, \tag{6}$$

where $\mathbf{Y}_{3\times1} = [y_0, y_1, y_2]^{\mathbf{T}}$, $\mathbf{X}_{3\times1} = [x_0, x_1, x_2]^{\mathbf{T}}$, $\mathbf{C}_3 = \begin{bmatrix} b_3 & c_3 & c_3 \\ c_3 & a_3 & -d_3 \\ c_3 & -d_3 & a_3 \end{bmatrix}$ with $a_3 = 0.2764$, $b_3 = 0.4472$, $c_3 = 0.6325$, $d_3 = 0.7236$.

The matrix $\mathbf{C}_3$ is decomposed into two components:

$$\mathbf{C}_3 = \mathbf{C}_3^{(\mathrm{a})} + \mathbf{C}_3^{(\mathrm{b})}, \tag{7}$$

where $\mathbf{C}_3^{(\mathrm{a})} = \begin{bmatrix} b_3 & c_3 & c_3 \\ c_3 & & \\ c_3 & & \end{bmatrix}$, $\mathbf{C}_3^{(\mathrm{b})} = \begin{bmatrix} & & \\ & a_3 & -d_3 \\ & -d_3 & a_3 \end{bmatrix}$.

Taken into account properties of structural matrices [27,28], the computational procedure for the three-point DCT-V is represented by expression

$$\mathbf{Y}_{3\times1} = \mathbf{W}_{3\times5}\mathbf{W}_5\mathbf{D}_5\mathbf{W}_5\mathbf{W}_{5\times6}\mathbf{W}_{6\times3}\mathbf{X}_{3\times1}, \tag{8}$$

where $\mathbf{D}_5 = diag\left(s_0^{(3)}, s_1^{(3)}, s_2^{(3)}, s_3^{(3)}, s_3^{(3)}\right)$, $s_0^{(3)} = b_3$, $s_1^{(3)} = (a_3 - d_3)/2$, $s_1^{(3)} = (a_3 + d_3)/2$, $s_3^{(3)} = c_3$, $\mathbf{W}_5 = 1\mathbf{H}_2\mathbf{I}_2$,

$$\mathbf{W}_{3\times5} = \begin{bmatrix} 1 & & & 1 & \\ & 1 & & & 1 \\ & & 1 & & 1 \end{bmatrix}, \mathbf{W}_{6\times3} = \begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \\ 1 & & \\ & 1 & \\ & & 1 \end{bmatrix}, \mathbf{W}_{5\times6} = \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & & 1 & 1 \\ & & & 1 & & 1 \end{bmatrix}.$$

A data flow graph of the proposed algorithm for the three-point DCT-V is shown in Figure 2. The number of multiplication operations can be reduced from 9 to 5, but the number of addition operations is increased from 6 to 8.

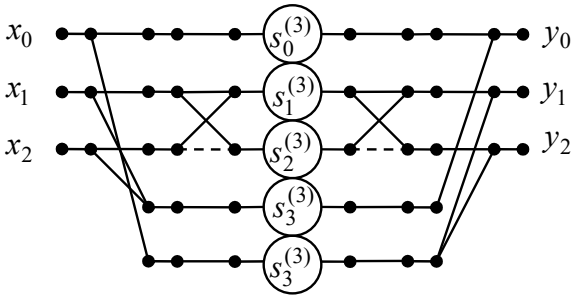

Figure 2. The data flow graph of the algorithm for computation of three-point DCT-V.

Let us design the algorithm for four-point DCT-V. The four-point DCT-V is expressed as follows:

$$\mathbf{Y}_{4\times1} = \mathbf{C}_4\mathbf{X}_{4\times1}, \tag{9}$$

where $\mathbf{Y}_{4\times1} = [y_0, y_1, y_2, y_3]^{\mathbf{T}}$, $\mathbf{X}_{4\times1} = [x_0, x_1, x_2, x_3]^{\mathbf{T}}$, $\mathbf{C}_4 = \begin{bmatrix} b_4 & d_4 & d_4 & d_4 \\ d_4 & c_4 & -a_4 & -e_4 \\ d_4 & -a_4 & -e_4 & c_4 \\ d_4 & -e_4 & c_4 & -a_4 \end{bmatrix}$ with $a_4 = 0.1682$, $b_4 = 0.3780$,$c_4 = 0.4713$, $d_4 = 0.5345$, $e_4 = 0.6811$.

To change the order of columns of $\mathbf{C}_4$, the permutation

$$\pi_4 = \begin{pmatrix} 1 & 23 & 4 \\ 1 & 24 & 3 \end{pmatrix}$$

is defined. After permutation of the columns of $\mathbf{C}_4$ according to π_4, we obtain the matrix $\mathbf{C}_4^{(a)} = \begin{bmatrix} b_4 & d_4 & d_4 & d_4 \\ d_4 & c_4 & -e_4 & -a_4 \\ d_4 & -a_4 & c_4 & -e_4 \\ d_4 & -e_4 & a_4 & -c_4 \end{bmatrix}$ with permutation matrix $\mathbf{P}_4 = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & & 1 \\ & & 1 & \end{bmatrix}$.

The matrix $\mathbf{C}_4^{(a)}$ is decomposed into two components:

$$\mathbf{C}_4^{(a)} = \mathbf{C}_4^{(b)} + \mathbf{C}_4^{(c)}, \tag{10}$$

where $\mathbf{C}_4^{(b)} = \begin{bmatrix} b_4 & d_4 & d_4 & d_4 \\ d_4 & c_4 & -e_4 & -a_4 \\ d_4 & -a_4 & c_4 & -e_4 \\ d_4 & -e_4 & a_4 & -c_4 \end{bmatrix}$, $\mathbf{C}_4^{(c)} = \begin{bmatrix} b_4 & d_4 & d_4 & d_4 \\ d_4 & c_4 & -e_4 & -a_4 \\ d_4 & -a_4 & c_4 & -e_4 \\ d_4 & -e_4 & a_4 & -c_4 \end{bmatrix}$.

To process the matrix $\mathbf{C}_4^{(a)}$, we use the expressions for calculating the entries of a circular convolution matrix $\mathbf{H}_3 = \begin{bmatrix} h_0 & h_2 & h_1 \\ h_1 & h_0 & h_2 \\ h_2 & h_1 & h_0 \end{bmatrix}$ for $N = 3$ [28,29]:

$$\mathbf{H}_3 = \mathbf{T}_3^{(1)} \mathbf{T}_{3\times 4} diag(s_0, s_1, s_2, s_3) \mathbf{T}_{4\times 3} \mathbf{T}_3^{(0)}, \tag{11}$$

where $s_0 = (h_0 + h_1 + h_2)/3, s_1 = h_0 - h_2, s_2 = h_1 - h_2, s_3 = (h_0 + h_1 - 2h_2)/3$; $\mathbf{T}_3^{(1)} = \begin{bmatrix} 1 & 1 & \\ 1 & -1 & -1 \\ 1 & & 1 \end{bmatrix}$, $\mathbf{T}_{3\times 4} = \begin{bmatrix} 1 & & & \\ & 1 & & -1 \\ & & 1 & -1 \end{bmatrix}$, $\mathbf{T}_{4\times 3} = \begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \\ & 1 & 1 \end{bmatrix}$, $\mathbf{T}_3^{(0)} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & & -1 \\ & 1 & -1 \end{bmatrix}$.

Taking into account the properties of structural matrices [28,29], the computational procedure for the four-point DCT-V is represented by the formula

$$\mathbf{Y}_{4\times 1} = \mathbf{W}_{4\times 6} \mathbf{W}_6^{(1)} \mathbf{W}_{6\times 7} \mathbf{D}_7 \mathbf{W}_{7\times 6} \mathbf{W}_6^{(0)} \mathbf{W}_{6\times 8} \mathbf{W}_{8\times 4} \mathrm{P}_4 \mathbf{X}_{4\times 1}, \tag{12}$$

where $\mathbf{W}_6^{(1)} = 1\mathbf{T}_3^{(1)}\mathrm{I}_2, \mathbf{W}_{6\times 7} = 1\mathbf{T}_{3\times 4}\mathbf{I}_2, \mathbf{D}_7 = diag(s_0^{(4)}, s_1^{(4)}, s_2^{(4)}, s_3^{(4)}, s_4^{(4)}, s_5^{(4)}, s_5^{(4)})$, $s_0^{(4)} = b_4$, $s_1^{(4)} = (c_4 - a_4 - e_4)/3$, $s_2^{(4)} = c_4 + e_4$, $s_3^{(4)} = -a_4 + e_4$, $s_4^{(4)} = (c_4 - a_4 + 2e_4)/3$, $s_5^{(4)} = d_4, \mathbf{W}_{7\times 6} = 1\mathbf{T}_{4\times 3}\mathbf{I}_2, \mathbf{W}_6^{(0)} = 1\mathbf{T}_3^{(0)}\mathbf{I}_2, \mathbf{W}_{6\times 8} = \mathbf{I}_4\mathbf{P}_{2\times 4}$, $\mathbf{W}_{4\times 6} = \begin{bmatrix} 1 & & & 1 & & \\ & 1 & & & & 1 \\ & & 1 & & & 1 \\ & & & 1 & & 1 \end{bmatrix}$, $\mathbf{P}_{2\times 4} = \begin{bmatrix} & 1 & 1 & 1 \\ 1 & & & \end{bmatrix}$, $\mathbf{W}_{8\times 4} = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \\ 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{bmatrix}$. A data flow graph of the proposed algorithm for the four-point DCT-V is presented in Figure 3. In particularly, the number of multiplication operations may be reduced from 16 to 7, although the number of addition operations is increased from 12 to 17.

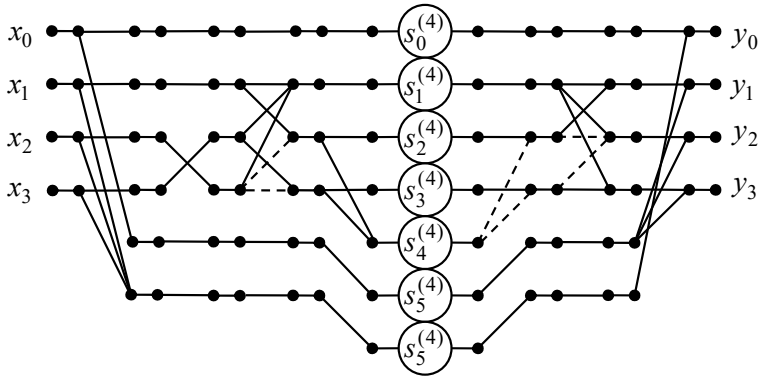

Figure 3. The data flow graph of the algorithm for computation of four-point DCT-V.

Let us obtain the algorithm for five-point DCT-V. The five-point DCT-V is expressed as follows:

$$\mathbf{Y}_{5\times 1} = \mathbf{C}_5 \mathbf{X}_{5\times 1}, \tag{13}$$

where $\mathbf{Y}_{5\times1} = [y_0, y_1, y_2, y_3, y_4]^{\mathbf{T}}$, $\mathbf{X}_{5\times1} = [x_0, x_1, x_2, x_3, x_4]^{\mathbf{T}}$, $a_5 = 0.1158$, $b_5 = 0.3333$, $c_5 = 0.4714$, $d_5 = 0.5107$, $e_5 = 0.6265$, $f_5 = 0.6667$, $\mathbf{C}_5 = \begin{bmatrix} b_5 & c_5 & c_5 & c_5 & c_5 \\ c_5 & d_5 & a_5 & -b_5 & -e_5 \\ c_5 & a_5 & -e_5 & -b_5 & d_5 \\ c_5 & -b_5 & -b_5 & f_5 & -b_5 \\ c_5 & -e_5 & d_5 & -b_5 & a_5 \end{bmatrix}$.

To change the order of columns and rows of $\mathbf{C}_5$, the permutations π_5 and π_6 are defined in the following form:

$$\pi_5 = \begin{pmatrix} 1 & 23 & 4 & 5 \\ 1 & 42 & 5 & 3 \end{pmatrix}, \ \pi_6 = \begin{pmatrix} 1 & 23 & 4 & 5 \\ 1 & 42 & 3 & 5 \end{pmatrix}.$$

The columns of $\mathbf{C}_5$ are permutated according to π_5, and the rows of $\mathbf{C}_5$ are permutated according to π_6. After permutations, the matrix acquires the following structure:

$$\mathbf{C}_5^{(a)} = \begin{bmatrix} b_5 & c_5 & c_5 & c_5 & c_5 \\ c_5 & f_5 & -b_5 & -b_5 & -b_5 \\ c_5 & -b_5 & d_5 & -e_5 & a_5 \\ c_5 & -b_5 & a_5 & d_5 & -e_5 \\ c_5 & -b_5 & -e_5 & a_5 & d_5 \end{bmatrix}$$

Then, the matrix $\mathbf{C}_5^{(a)}$ is decomposed into two components:

$$\mathbf{C}_5^{(a)} = \mathbf{C}_5^{(b)} + \mathbf{C}_5^{(c)}, \tag{14}$$

where $\mathbf{C}_5^{(b)} = \begin{bmatrix} b_5 & c_5 & c_5 & c_5 & c_5 \\ c_5 & f_5 & -b_5 & -b_5 & -b_5 \\ c_5 & -b_5 & & & \\ c_5 & -b_5 & & & \\ c_5 & -b_5 & & & \end{bmatrix}$ and $\mathbf{C}_5^{(c)} = \begin{bmatrix} & & & & \\ & & & & \\ & & d_5 & -e_5 & a_5 \\ & & a_5 & d_5 & -e_5 \\ & & -e_5 & a_5 & d_5 \end{bmatrix}$.

Matrix $\mathbf{C}_5^{(b)}$ has the same entries except on the main diagonal in the first column and first row, and the second column and second row, which allows reducing the number of operations without the need for further transformations. After eliminating the rows and columns containing only zero entries in matrix $\mathbf{C}_5^{(c)}$, we obtain matrix $\mathbf{C}_3^{(d)}$:

$$\mathbf{C}_3^{(d)} = \begin{bmatrix} d_5 & -e_5 & a_5 \\ a_5 & d_5 & -e_5 \\ -e_5 & a_5 & d_5 \end{bmatrix}.$$

To process the matrix $\mathbf{C}_3^{(d)}$ we use the expressions for calculating the entries of a circular convolution matrix $\mathbf{H}_3$ for $N = 3$ (Equation (11)) [28,29]. Taking into account the properties of structural matrices [28,29], the computational procedure for the five-point DCT-V is represented by the matrix–vector procedure

$$\mathbf{Y}_{5\times1} = \mathbf{P}_5^{(1)}\mathbf{W}_{5\times9}\mathbf{W}_9^{(1)}\mathbf{W}_{9\times10}\mathbf{D}_{10}\mathbf{W}_{10\times9}\mathbf{W}_9^{(0)}\mathbf{W}_{9\times10}\mathbf{W}_{10\times5}\mathbf{P}_5^{(0)}\mathbf{X}_{5\times1}, \tag{15}$$

where

$$\mathbf{W}_{9\times10} = \mathbf{I}_5\mathbf{P}_{2\times5}^{(3)}, \ \mathbf{W}_9^{(1)} = \mathbf{I}_2\mathbf{T}_3^{(1)}\mathbf{I}_4, \mathbf{W}_{9\times10} = \mathbf{I}_2\mathbf{T}_{3\times4}\mathbf{I}_4,$$

$$\mathbf{W}_{10\times9} = \mathbf{I}_2\mathbf{T}_{4\times3}\mathbf{I}_4, \ \mathbf{W}_{9\times9}^{(0)} = \mathbf{I}_2\mathbf{T}_3^{(1)}\mathbf{I}_4,$$

$$\mathbf{D}_{10} = diag\left(s_0^{(5)}, s_1^{(5)}, s_2^{(5)}, s_3^{(5)}, s_4^{(5)}, s_5^{(5)}, s_6^{(5)}, s_7^{(5)}, s_8^{(5)}, s_9^{(5)}\right),$$

$$s_0^{(5)} = b_5, s_1^{(5)} = f_5, s_2^{(5)} = \Big(d_5 + a_5 - e_5\Big)/3, s_3^{(5)} = d_5 + e_5,\ s_4^{(5)} = a_5 + e_5,\ s_5^{(5)} = (d_5 + a_5 + 2e_5)/3,$$

$$s_6^{(5)} = c_5, s_7^{(5)} = c_5, s_8^{(5)} = -b_5, s_9^{(5)} = -b_5,$$

$$\mathbf{P}_{2\times5}^{(3)} = \begin{bmatrix} & 1 & 1 & 1 & 1 \\ 1 & & & & \\ & & 1 & 1 & 1 \\ & 1 & & & \end{bmatrix},\ \mathbf{P}_5^{(0)} = \begin{bmatrix} 1 & & & & \\ & & & 1 & \\ & 1 & & & \\ & & & & 1 \\ & & 1 & & \end{bmatrix},\ \mathbf{P}_5^{(1)} = \begin{bmatrix} 1 & & & & \\ & & 1 & & \\ & & & 1 & \\ & 1 & & & \\ & & & & 1 \end{bmatrix},$$

$$\mathbf{W}_{5\times9} = \begin{bmatrix} 1 & & & & & 1 & & & \\ & 1 & & & & & 1 & 1 & \\ & & 1 & & & & 1 & & 1 \\ & & & 1 & & & 1 & & 1 \\ & & & & 1 & & 1 & & 1 \end{bmatrix},\ \mathbf{W}_{10\times5} = \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & 1 & \\ & & & & 1 \\ 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & 1 & \\ & & & & 1 \end{bmatrix}.$$

A data-flow graph of the proposed algorithm for the five-point DCT-V is presented in Figure 4. In particularly, the number of multiplication operations may be reduced from 25 to 10, but the number of addition operations is increased from 20 to 23.

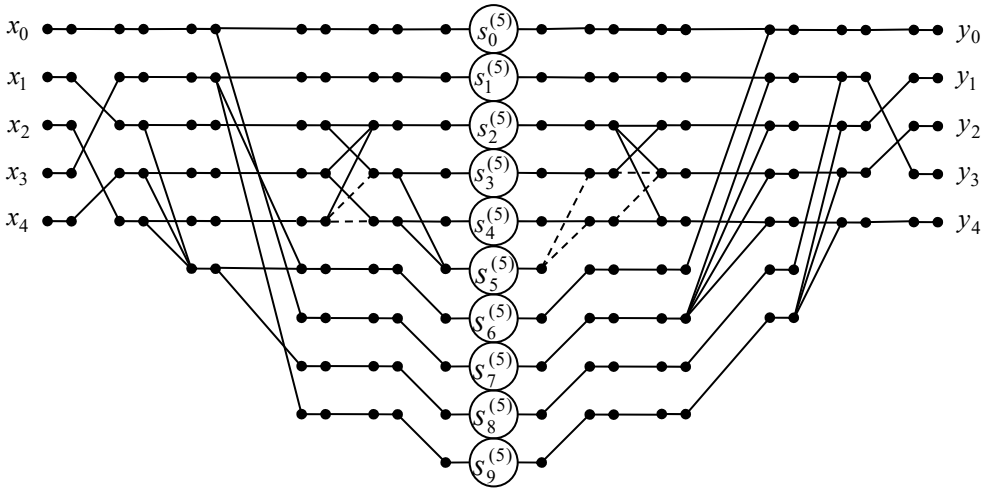

Figure 4. The data-flow graph of the algorithm for computation of five-point DCT-V.

Let us now synthesize an algorithm for a six-point DCT-V, which is expressed as follows:

$$\mathbf{Y}_{6\times1} = \mathbf{C}_6\mathbf{X}_{6\times1}, \tag{16}$$

where $\mathbf{Y}_{6\times1} = [y_0, y_1, y_2, y_3, y_4, y_5]^{\mathbf{T}}$, $\mathbf{X}_{6\times1} = [x_0, x_1, x_2, x_3, x_4, x_5]^{\mathbf{T}}$, $a_6 = 0.0858$, b_6= 0.2505, c_6 = 0.3015, d_6 = 0.3949, e_6 = 0.4264, f_6 = 0.5073, g_6 = 0.5786,

$$\mathbf{C}_6 = \begin{bmatrix} c_6 & e_6 & e_6 & e_6 & e_6 & e_6 \\ e_6 & f_6 & b_6 & -a_6 & -d_6 & -g_6 \\ e_6 & b_6 & -d_6 & -g_6 & -a_6 & f_6 \\ e_6 & -a_6 & -g_6 & b_6 & f_6 & -d_6 \\ e_6 & -d_6 & -a_6 & f_6 & -g_6 & b_6 \\ e_6 & -g_6 & f_6 & -d_6 & b_6 & -a_6 \end{bmatrix}.$$

To change the order of columns and rows of $\mathbf{C}_6$, the permutations π_7 and π_8 are defined in the following form:

$$\pi_7 = \begin{pmatrix} 1 & 23 & 4 & 5 & 6 \\ 1 & 26 & 4 & 5 & 3 \end{pmatrix}, \ \pi_8 = \begin{pmatrix} 1 & 23 & 4 & 5 & 6 \\ 1 & 23 & 5 & 4 & 6 \end{pmatrix}.$$

The columns of $\mathbf{C}_6$ are permutated according to π_7, and the rows of $\mathbf{C}_6$ are permutated according to π_8. After permutations, the matrix acquires the following structure:

$$\mathbf{C}_6^{(a)} = \begin{bmatrix} c_6 & e_6 & e_6 & e_6 & e_6 & e_6 \\ e_6 & f_6 & -g_6 & -a_6 & -d_6 & b_6 \\ e_6 & b_6 & f_6 & -g_6 & -a_6 & -d_6 \\ e_6 & -d_6 & b_6 & f_6 & -g_6 & -a_6 \\ e_6 & -a_6 & -d_6 & b_6 & f_6 & -g_6 \\ e_6 & -g_6 & -a_6 & -d_6 & b_6 & f_6 \end{bmatrix}$$

Then, the matrix $\mathbf{C}_6^{(a)}$ is decomposed into two components:

$$\mathbf{C}_6^{(a)} = \mathbf{C}_6^{(b)} + \mathbf{C}_6^{(c)}, \tag{17}$$

where

$$\mathbf{C}_6^{(b)} = \begin{bmatrix} c_6 & e_6 & e_6 & e_6 & e_6 & e_6 \\ e_6 & & & & & \\ e_6 & & & & & \\ e_6 & & & & & \\ e_6 & & & & & \\ e_6 & & & & & \end{bmatrix}, \ \mathbf{C}_6^{(c)} = \begin{bmatrix} & & & & & \\ & f_6 & -g_6 & -a_6 & -d_6 & b_6 \\ & b_6 & f_6 & -g_6 & -a_6 & -d_6 \\ & -d_6 & b_6 & f_6 & -g_6 & -a_6 \\ & -a_6 & -d_6 & b_6 & f_6 & -g_6 \\ & -g_6 & -a_6 & -d_6 & b_6 & f_6 \end{bmatrix}.$$

Matrix $\mathbf{C}_6^{(b)}$ has the same entries except on the main diagonal in the first column and first row, which allows reducing the number of operations without the need for further transformations. After eliminating the rows and columns containing only zero entries in matrix $\mathbf{C}_6^{(c)}$, we obtain matrix $\mathbf{C}_5^{(d)}$:

$$\mathbf{C}_5^{(d)} = \begin{bmatrix} f_6 & -g_6 & -a_6 & -d_6 & b_6 \\ b_6 & f_6 & -g_6 & -a_6 & -d_6 \\ -d_6 & b_6 & f_6 & -g_6 & -a_6 \\ -a_6 & -d_6 & b_6 & f_6 & -g_6 \\ -g_6 & -a_6 & -d_6 & b_6 & f_6 \end{bmatrix}.$$

To process the matrix $\mathbf{C}_5^{(d)}$, we use the expressions for calculating the entries of a circular convolution matrix $\mathbf{H}_5 = \begin{bmatrix} h_0 & h_4 & h_3 & h_2 & h_1 \\ h_1 & h_0 & h_4 & h_3 & h_2 \\ h_2 & h_1 & h_0 & h_4 & h_3 \\ h_3 & h_2 & h_1 & h_0 & h_4 \\ h_4 & h_3 & h_2 & h_1 & h_0 \end{bmatrix}$ for $N = 5$ [29]. Then,

$$h_0 = f_6, h_1 = b_6, h_2 = -d_6, h_3 = -a_6, h_4 = -g_6,$$

and the computational procedure for the six-point DCT-V is expressed by formula

$$\mathbf{Y}_{6\times1} = \mathbf{P}_6^{(1)}\mathbf{W}_{6\times8}\mathbf{W}_8^{(0)}\mathbf{W}_{8\times10}\mathbf{W}_{10\times13}\mathbf{D}_{13\times13}\mathbf{W}_{13\times10}\mathbf{W}_{10\times8}\mathbf{W}_8^{(0)}\mathbf{W}_8^{(1)}\mathbf{W}_{8\times12}\mathbf{W}_{12\times6}\mathbf{P}_6^{(0)}\mathbf{X}_{6\times1}, \tag{18}$$

where

$$\mathbf{P}_6^{(0)} = \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & & & & 1 \\ & & & 1 & & \\ & & & & 1 & \\ & & 1 & & & \end{bmatrix}, \mathbf{P}_6^{(1)} = \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & & 1 & \\ & & & 1 & & \\ & & & & & 1 \end{bmatrix},$$

$$\mathbf{W}_{8\times 10} = \begin{bmatrix} 1 & & & & & & & & & \\ & 1 & & & & & & & & \\ & & 1 & & & & 1 & & & \\ & & & 1 & & & & 1 & & \\ & & & & 1 & & 1 & & & \\ & & & & & 1 & & 1 & & \\ & & & & & & & & 1 & \\ & & & & & & & & & 1 \end{bmatrix},$$

$$\mathbf{W}_{10\times 8} = \begin{bmatrix} 1 & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & 1 & & 1 & & & \\ & & & 1 & & 1 & & \\ & & & & & & 1 & \\ & & & & & & & 1 \end{bmatrix}, \mathbf{W}_8^{(0)} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & & \\ & 1 & -1 & & & & & \\ & 1 & & -1 & & & & \\ & 1 & & & -1 & & & \\ & 1 & & & & -1 & & \\ & & & & & & 1 & \\ & & & & & & & 1 \end{bmatrix},$$

$$\mathbf{W}_8^{(1)} = \begin{bmatrix} 1 & & & & & & & \\ & 1 & & & & & & \\ & & & & 1 & & & \\ & & & 1 & & & & \\ & & 1 & & & & & \\ & & & & & 1 & & \\ & & & & & & 1 & \\ & & & & & & & 1 \end{bmatrix},$$

$$\mathbf{W}_{12\times 6} = \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & 1 \\ 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{bmatrix}, \mathbf{W}_{6\times 8} = \begin{bmatrix} 1 & & & & & & 1 & \\ & 1 & & & & & & 1 \\ & & 1 & & & & & 1 \\ & & & 1 & & & & 1 \\ & & & & 1 & & & 1 \\ & & & & & 1 & & 1 \end{bmatrix},$$

$$\mathbf{W}_{10\times 13} = \mathbf{I}_2\mathbf{T}_{2\times 3}\mathbf{T}_{2\times 3}\mathbf{T}_{2\times 3}\mathbf{I}_2, \mathbf{W}_{13\times 10} = \mathbf{I}_2\mathbf{T}_{3\times 2}\mathbf{T}_{3\times 2}\mathbf{T}_{3\times 2}\mathbf{I}_2, \mathbf{T}_{2\times 3} = \begin{bmatrix} 1 & & 1 \\ & 1 & 1 \end{bmatrix},$$

$$\mathbf{T}_{3\times 2} = \begin{bmatrix} 1 & \\ & 1 \\ 1 & 1 \end{bmatrix}, \mathbf{W}_{8\times 12} = \mathbf{I}_6\mathbf{P}_{2\times 6}^{(2)}, \mathbf{P}_{2\times 6}^{(2)} = \begin{bmatrix} & 1 & 1 & 1 & 1 & 1 \\ 1 & & & & & \end{bmatrix},$$

$$\mathbf{D}_{13} = diag\left(s_0^{(6)}, s_1^{(6)}, s_2^{(6)}, s_3^{(6)}, s_4^{(6)}, s_5^{(6)}, s_6^{(6)}, s_7^{(6)}, s_8^{(6)}, s_9^{(6)}, s_{10}^{(6)}, s_{11}^{(6)}, s_{11}^{(6)}\right),$$

$$s_0^{(6)} = c_6, s_1^{(6)} = (h_0 + h_1 + h_2 + h_3 + h_4)/5, s_2^{(6)} = h_2 - h_4 + h_0 - h_3, s_3^{(6)} = -h_1 + h_4 + h_0 - h_3, s_4^{(6)} = h_3 - h_0,$$
$$s_5^{(6)} = h_1 - h_4 + h_0 - h_2, s_6^{(6)} = h_3 - h_1 + h_0 - h_2, s_7^{(6)} = h_2 - h_0, s_8^{(6)} = h_4 - h_0, s_9^{(6)} = h_1 - h_0, s_{10}^{(6)} = h_0 - s_0^{(6)},$$
$$s_{11}^{(6)} = e_6.$$

A data-flow graph of the proposed algorithm of the six-point DCT-V is presented in Figure 5. We are able to reduce the number of multiplication operations from 36 to 13, but the number of addition operations is increased from 30 to 41.

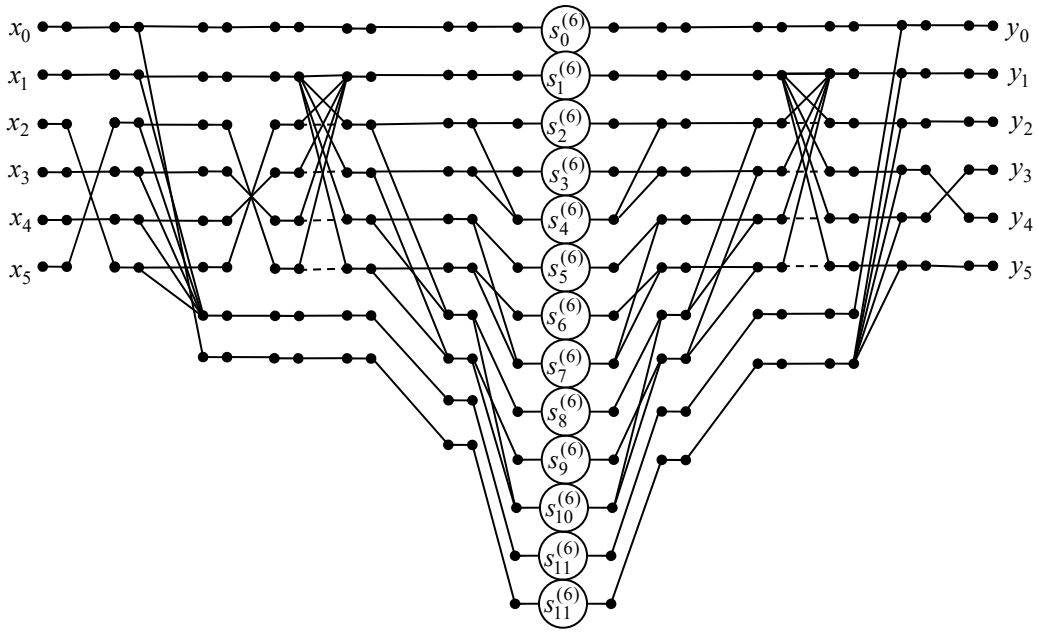

Figure 5. The data-flow graph of the algorithm for computation of six-point DCT-V.

The seven-point DCT-V transform in matrix notation can be represented by the following expression:

$$\mathbf{Y}_{7\times1} = \mathbf{C}_7\mathbf{X}_{7\times1}, \tag{19}$$

where $\mathbf{Y}_{7\times1} = [y_0, y_1, y_2, y_3, y_4, y_5, y_6]^{\mathbf{T}}$, $\mathbf{X}_{7\times1} = [x_0, x_1, x_2, x_3, x_4, x_5, x_6]^{\mathbf{T}}$, $a_7 = 0.0669$, $b_7 = 0.1967$, $c_7 = 0.2774$, $d_7 = 0.3151$, $e_7 = 0.3922$, $f_7 = 0.4152$, $g_7 = 0.4912$, $h_7 = 0.5386$,

$$\mathbf{C}_7 = \begin{bmatrix} c_7 & e_7 & e_7 & e_7 & e_7 & e_7 & e_7 \\ e_7 & g_7 & d_7 & a_7 & -b_7 & -f_7 & -h_7 \\ e_7 & d_7 & -b_7 & -h_7 & -f_7 & a_7 & g_7 \\ e_7 & a_7 & -h_7 & -b_7 & g_7 & d_7 & -f_7 \\ e_7 & -b_7 & -f_7 & g_7 & a_7 & -h_7 & d_7 \\ e_7 & -f_7 & a_7 & d_7 & -h_7 & g_7 & -b_7 \\ e_7 & -h_7 & g_7 & -f_7 & d_7 & -b_7 & a_7 \end{bmatrix}.$$

Now, we need to change the order of columns and rows. Let us define the permutations π_9, π_{10} in the following form:

$$\pi_9 = \begin{pmatrix} 1 & 23 & 4 & 5 & 6 & 7 \\ 1 & 27 & 4 & 6 & 5 & 3 \end{pmatrix}, \ \pi_{10} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 2 & 3 & 5 & 6 & 4 & 7 \end{pmatrix}.$$

Permute columns and rows of $\mathbf{C}_7$ according to π_9, π_{10}. After permutations, the matrix acquires the following structure:

$$\mathbf{C}_7^{(a)} = \begin{bmatrix} c_7 & e_7 & e_7 & e_7 & e_7 & e_7 & e_7 \\ e_7 & g_7 & -h_7 & a_7 & -f_7 & -b_7 & d_7 \\ e_7 & d_7 & g_7 & -h_7 & a_7 & -f_7 & -b_7 \\ e_7 & -b_7 & d_7 & g_7 & -h_7 & a_7 & -f_7 \\ e_7 & -f_7 & -b_7 & d_7 & g_7 & -h_7 & a_7 \\ e_7 & a_7 & -f_7 & -b_7 & d_7 & g_7 & -h_7 \\ e_7 & -h_7 & a_7 & -f_7 & -b_7 & d_7 & g_7 \end{bmatrix}.$$

Then, the matrix $\mathbf{C}_7^{(a)}$ is decomposed into two components:

$$\mathbf{C}_7^{(a)} = \mathbf{C}_7^{(b)} + \mathbf{C}_7^{(c)}, \tag{20}$$

where

$$\mathbf{C}_7^{(b)} = \begin{bmatrix} c_7 & e_7 & e_7 & e_7 & e_7 & e_7 & e_7 \\ e_7 & & & & & & \\ e_7 & & & & & & \\ e_7 & & & & & & \\ e_7 & & & & & & \\ e_7 & & & & & & \\ e_7 & & & & & & \end{bmatrix}, \mathbf{C}_7^{(c)} = \begin{bmatrix} & & & & & \\ g_7 & -h_7 & a_7 & -f_7 & -b_7 & d_7 \\ d_7 & g_7 & -h_7 & a_7 & -f_7 & -b_7 \\ -b_7 & d_7 & g_7 & -h_7 & a_7 & -f_7 \\ -f_7 & -b_7 & d_7 & g_7 & -h_7 & a_7 \\ a_7 & -f_7 & -b_7 & d_7 & g_7 & -h_7 \\ -h_7 & a_7 & -f_7 & -b_7 & d_7 & g_7 \end{bmatrix}.$$

Matrix $\mathbf{C}_7^{(b)}$ has the same entries in the first column and first row, which allows us to reduce the number of operations without the need for further transformations. After eliminating the rows and columns containing only zero entries in matrix $\mathbf{C}_7^{(c)}$, we obtain matrix $\mathbf{C}_7^{(d)}$:

$$\mathbf{C}_6^{(d)} = \begin{bmatrix} g_7 & -h_7 & a_7 & -f_7 & -b_7 & d_7 \\ d_7 & g_7 & -h_7 & a_7 & -f_7 & -b_7 \\ -b_7 & d_7 & g_7 & -h_7 & a_7 & -f_7 \\ -f_7 & -b_7 & d_7 & g_7 & -h_7 & a_7 \\ a_7 & -f_7 & -b_7 & d_7 & g_7 & -h_7 \\ -h_7 & a_7 & -f_7 & -b_7 & d_7 & g_7 \end{bmatrix}.$$

The obtained matrix acquires the structure of the expressions for calculating the entries of a circular convolution matrix $\mathbf{H}_6 = \begin{bmatrix} h_0 & h_5 & h_4 & h_3 & h_2 & h_1 \\ h_1 & h_0 & h_5 & h_4 & h_3 & h_2 \\ h_2 & h_1 & h_0 & h_5 & h_4 & h_3 \\ h_3 & h_2 & h_1 & h_0 & h_5 & h_4 \\ h_4 & h_3 & h_2 & h_1 & h_0 & h_5 \\ h_5 & h_4 & h_3 & h_2 & h_1 & h_0 \end{bmatrix}$ for $N = 6$ [29].

Then, denote

$$s_0^{(7)} = c_7; s_1^{(7)} = \left(g_7 + b_7 - f_7 + h_7\right)/6; s_2^{(7)} = (d_7 + b_7 + a_7 + h_7)/6;$$

$$s_3^{(7)} = \left(d_7 + a_7 - g_7 + f_7\right)/6; \; s_4^{(7)} = \left(g_7 + b_7 + f_7 - h_7\right)/6; \; s_5^{(7)} = (d_7 - b_7 - a_7 + h_7)/6;$$

$$s_6^{(7)} = \left(g_7 + f_7 - a_7 + d_7\right)/6; \; s_7^{(7)} = \left(g_7 - d_7 - b_7 + f_7 + a_7 + h_7\right)/6; \; s_8^{(7)} = \left(g_7 + d_7 - b_7 - f_7 + a_7 - h_7\right)/6; \; s_9^{(7)} = e_7.$$

Based on properties of structural matrices [28,29], the computational procedure for the seven-point DCT-V is represented by expression

$$\mathbf{Y}_{7\times1} = \mathbf{P}_7^{(1)}\mathbf{W}_{7\times9}\mathbf{W}_{9\times11}^{(1)}\mathbf{W}_{11}^{(0)}\mathbf{W}_{11}^{(1)}\mathbf{D}_{11}\mathbf{W}_{11\times9}^{(1)}\mathbf{W}_{9\times11}^{(0)}\mathbf{W}_{11\times9}^{(0)}\mathbf{W}_{9\times14}\mathbf{W}_{14\times7}\mathbf{P}_7^{(0)}\mathbf{X}_{7\times1}, \tag{21}$$

where

$$\mathbf{D}_{11} = diag\left(s_0^{(7)}, s_1^{(7)}, s_2^{(7)}, s_3^{(7)}, s_4^{(7)}, s_5^{(7)}, s_6^{(7)}, s_7^{(7)}, s_8^{(7)}, s_9^{(7)}, s_9^{(7)}\right),$$

$$\mathbf{W}_{9\times11}^{(0)} = 1\mathbf{T}_{6\times8}^{(0)}\mathbf{I}_2,\ \mathbf{W}_{9\times11}^{(1)} = 1\mathbf{T}_{6\times8}^{(1)}\mathbf{I}_2,\ \mathbf{W}_{11\times9}^{(0)} = 1\mathbf{T}_{8\times6}\mathbf{I}_2,\ \mathbf{W}_{11}^{(0)} = 1\mathbf{T}_3^{(2)}\mathbf{T}_3^{(2)}\mathbf{I}_4,$$

$$\mathbf{W}_{11}^{(1)} = 1\mathbf{T}_3^{(3)}\mathbf{T}_3^{(4)}\mathbf{H}_2\mathbf{I}_2,\ \mathbf{W}_{11\times9}^{(1)} = 1\mathbf{T}_{6\times4}\mathbf{I}_4,\ \mathbf{W}_{9\times14} = \mathbf{I}_7\mathbf{T}_{2\times7},$$

$$\mathbf{P}_7^{(0)} = \begin{bmatrix} 1 & & & & & & \\ & 1 & & & & & \\ & & & & & & 1 \\ & & & 1 & & & \\ & & & & 1 & & \\ & & & & & 1 & \\ & & 1 & & & & \end{bmatrix}, \mathbf{P}_7^{(1)} = \begin{bmatrix} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & & & 1 & \\ & & & 1 & & & \\ & & & & 1 & & \\ & & & & & & 1 \end{bmatrix},$$

$$\mathbf{T}_{6\times8}^{(0)} = \begin{bmatrix} & 1 & & 1 & & \\ & & 1 & & & 1 \\ & 1 & & -1 & & \\ 1 & & & & & -1 \\ 1 & & -1 & & 1 & \\ 1 & & 1 & & & 1 \end{bmatrix}, \mathbf{T}_{8\times6} = \begin{bmatrix} 1 & & 1 & & & \\ & 1 & 1 & & & \\ 1 & & -1 & & & \\ & 1 & -1 & & & \\ & 1 & & 1 & & \\ & & & 1 & & -1 \\ & & & & 1 & -1 \\ & & & & 1 & 1 \end{bmatrix},$$

$$\mathbf{T}_{6\times8}^{(1)} = \begin{bmatrix} 1 & & 1 & & & 1 \\ & 1 & & -1 & & -1 \\ & & -1 & & -1 & 1 \\ 1 & & & -1 & & -1 \\ & 1 & & & 1 & \\ & & -1 & & 1 & -1 \end{bmatrix}, \mathbf{T}_{6\times4} = \begin{bmatrix} 1 & & & \\ & 1 & & \\ 1 & -1 & & \\ & & 1 & \\ & & & 1 \\ & & 1 & 1 \end{bmatrix},$$

$$\mathbf{T}_3^{(2)} = \begin{bmatrix} 1 & & -1 \\ & 1 & 1 \\ 1 & 1 & \end{bmatrix}, \mathbf{T}_3^{(3)} = \begin{bmatrix} 1 & -1 & \\ 1 & & 1 \\ & 1 & 1 \end{bmatrix}, \mathbf{T}_3^{(4)} = \begin{bmatrix} 1 & -1 & \\ 1 & & -1 \\ & 1 & -1 \end{bmatrix},$$

$$\mathbf{T}_{2\times7} = \begin{bmatrix} & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & & & & & & \end{bmatrix},$$

$$\mathbf{W}_{14\times7} = \begin{bmatrix} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ & & & & 1 & & \\ & & & & & 1 & \\ & & & & & & 1 \\ 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ & & & & 1 & & \\ & & & & & 1 & \\ & & & & & & 1 \end{bmatrix}, \mathbf{W}_{7\times9} = \begin{bmatrix} 1 & & & & & & & & 1 \\ & 1 & & & & & & & 1 \\ & & 1 & & & & & & 1 \\ & & & 1 & & & & & 1 \\ & & & & 1 & & & & 1 \\ & & & & & 1 & & & 1 \\ & & & & & & 1 & & 1 \\ & & & & & & & 1 & 1 \end{bmatrix}.$$

Figure 6 shows a data flow graph of the synthesized algorithm for the seven-point DCT-V. As can be seen, we were able to reduce the number of multiplications from 49 to 11, although the number of additions increased from 42 to 55.

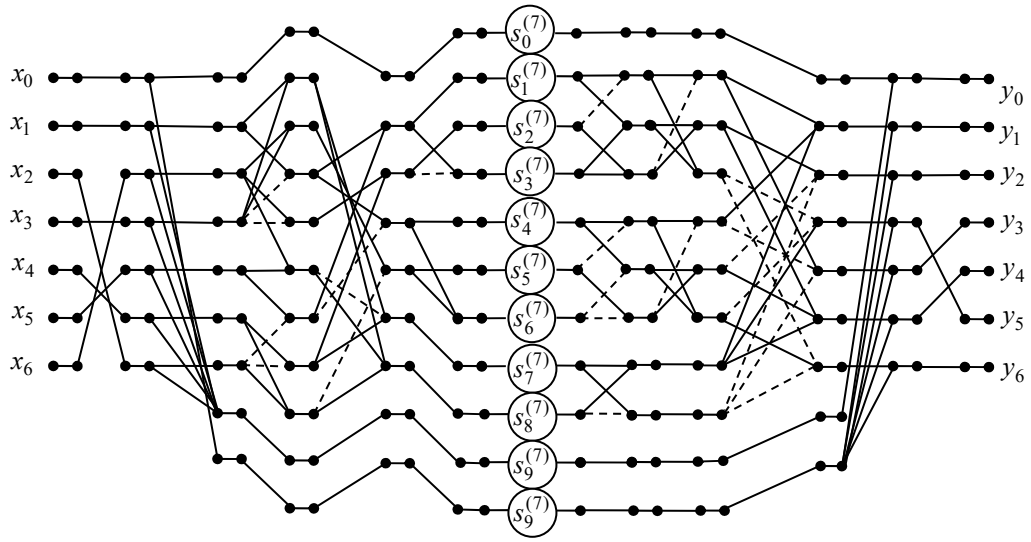

Figure 6. The data-flow graph of the algorithm for computation of seven-point DCT-V.

Let us design the algorithm for eight-point DCT-V. The eight-point DCT-V is expressed as follows:

$$\mathbf{Y}_{8\times1} = \mathbf{C}_8\mathbf{X}_{8\times1}, \tag{22}$$

where $\mathbf{Y}_{8\times1} = [y_0, y_1, y_2, y_3, y_4, y_5, y_6, y_7]^{\mathbf{T}}$, $\mathbf{X}_{8\times1} = [x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7]^{\mathbf{T}}$,

$$\mathbf{C}_8 = \begin{bmatrix} c_8 & e_8 & e_8 & e_8 & e_8 & e_8 & e_8 & e_8 \\ e_8 & g_8 & d_8 & b_8 & -a_8 & -c_8 & -f_8 & -h_8 \\ e_8 & d_8 & -a_8 & -f_8 & -h_8 & -c_8 & b_8 & g_8 \\ e_8 & b_8 & -f_8 & -f_8 & b_8 & q_8 & b_8 & -f_8 \\ e_8 & -a_8 & -h_8 & b_8 & g_8 & -c_8 & -f_8 & d_8 \\ e_8 & -c_8 & -c_8 & q_8 & -c_8 & -c_8 & q_8 & -c_8 \\ e_8 & -f_8 & b_8 & b_8 & -f_8 & q_8 & -f_8 & b_8 \\ e_8 & -h_8 & g_8 & -f_8 & d_8 & -c_8 & b_8 & -a_8 \end{bmatrix}$$

with $a_8 = 0.0540$, $b_8 = 0.1596$, $c_8 = 0.2582$, $d_8 = 0.3455$, $e_8 = 0.3651$, $f_8 = 0.4178$, $g_8 = 0.4718$, $h_8 = 0.5051$, $q_8 = 0.5164$.

Let us define the permutations

$$\pi_{11} = \begin{pmatrix} 1 & 23 & 4 & 5 & 6 & 7 & 8 \\ 1 & 64 & 7 & 2 & 8 & 5 & 3 \end{pmatrix}, \ \pi_{12} = \begin{pmatrix} 1 & 23 & 4 & 5 & 6 & 7 & 8 \\ 1 & 67 & 4 & 2 & 3 & 5 & 8 \end{pmatrix}$$

to change the order of columns and rows, respectively. As a result of the permutations, the matrix $\mathbf{C}_8^{(a)}$ is obtained:

$$\mathbf{C}_8^{(a)} = \begin{bmatrix} c_8 & e_8 & e_8 & e_8 & e_8 & e_8 & e_8 & e_8 \\ e_8 & -c_8 & q_8 & q_8 & -c_8 & -c_8 & -c_8 & -c_8 \\ e_8 & q_8 & b_8 & -f_8 & -f_8 & b_8 & -f_8 & b_8 \\ e_8 & q_8 & -f_8 & b_8 & b_8 & -f_8 & b_8 & -f_8 \\ e_8 & -c_8 & b_8 & -f_8 & g_8 & -h_8 & -a_8 & d_8 \\ e_8 & -c_8 & -f_8 & b_8 & d_8 & g_8 & -h_8 & -a_8 \\ e_8 & -c_8 & b_8 & -f_8 & -a_8 & d_8 & g_8 & -h_8 \\ e_8 & -c_8 & -f_8 & b_8 & -h_8 & -a_8 & d_8 & g_8 \end{bmatrix}.$$

Further, the matrix $\mathbf{C}_8^{(a)}$ is decomposed into two components:

$$\mathbf{C}_8^{(a)} = \mathbf{C}_8^{(b)} + \mathbf{C}_8^{(c)}, \tag{23}$$

where

$$\mathbf{C}_8^{(b)} = \begin{bmatrix} c_8 & e_8 & e_8 & e_8 & e_8 & e_8 & e_8 & e_8 \\ e_8 & -c_8 & q_8 & q_8 & -c_8 & -c_8 & -c_8 & -c_8 \\ e_8 & q_8 & b_8 & -f_8 & -f_8 & b_8 & -f_8 & b_8 \\ e_8 & q_8 & -f_8 & b_8 & b_8 & -f_8 & b_8 & -f_8 \\ e_8 & -c_8 & b_8 & -f_8 & & & & \\ e_8 & -c_8 & -f_8 & b_8 & & & & \\ e_8 & -c_8 & b_8 & -f_8 & & & & \\ e_8 & -c_8 & -f_8 & b_8 & & & & \end{bmatrix}, \mathbf{C}_8^{(c)} = \begin{bmatrix} & & & & g_8 & -h_8 & -a_8 & d_8 \\ & & & & d_8 & g_8 & -h_8 & -a_8 \\ & & & & -a_8 & d_8 & g_8 & -h_8 \\ & & & & -h_8 & -a_8 & d_8 & g_8 \end{bmatrix}.$$

Matrix $\mathbf{C}_8^{(b)}$ has the repeated entries in the first, second, third, and fourth columns and rows, which allows us to reduce the number of operations without the need for further transformations. After eliminating the rows and columns containing only zero entries in matrix $\mathbf{C}_8^{(c)}$, we obtain matrix $\mathbf{C}_4^{(e)}$:

$$\mathbf{C}_4^{(e)} = \begin{bmatrix} g_8 & -h_8 & -a_8 & d_8 \\ d_8 & g_8 & -h_8 & -a_8 \\ -a_8 & d_8 & g_8 & -h_8 \\ -h_8 & -a_8 & d_8 & g_8 \end{bmatrix}.$$

To process the matrix $\mathbf{C}_4^{(e)}$, we use the expressions for calculating the entries of a circular convolution matrix $\mathbf{H}_4 = \begin{bmatrix} h_0 & h_3 & h_2 & h_1 \\ h_1 & h_0 & h_3 & h_2 \\ h_2 & h_1 & h_0 & h_3 \\ h_1 & h_2 & h_1 & h_0 \end{bmatrix}$ for $N = 4$ [29]:

$$\mathbf{H}_3 = \mathbf{T}_4^{(1)}\mathbf{T}_{4\times5}diag\left(s_0^{(0)}, s_1^{(0)}, s_2^{(0)}, s_3^{(0)}, s_4^{(0)}\right)\mathbf{T}_{5\times4}\mathbf{T}_4^{(0)}, \tag{24}$$

where

$$s_0^{(0)} = (g_8 + d_8 - a_8 - h_8)/4, s_1^{(0)} = (g_8 - d_8 - a_8 + h_8)/4, s_2^{(0)} = (g_8 + a_8)/2, s_3^{(0)} = (g_8 - d_8 + a_8 - h_8)/2;$$

$$s_4^{(0)} = (g_8 + d_8 + a_8 + h_8)/2;$$

$$\mathbf{T}_4^{(1)} = \begin{bmatrix} 1 & & & 1 \\ & 1 & 1 & \\ 1 & & & -1 \\ & 1 & -1 & \end{bmatrix}, \mathbf{T}_4^{(0)} = \begin{bmatrix} 1 & & 1 & \\ & 1 & & 1 \\ 1 & & -1 & \\ & 1 & & -1 \end{bmatrix}, \mathbf{T}_{4\times5} = \begin{bmatrix} 1 & 1 & & & \\ 1 & -1 & & & \\ & & 1 & -1 & \\ & & 1 & & -1 \end{bmatrix}, \mathbf{T}_{5\times4} = \begin{bmatrix} 1 & 1 & & \\ 1 & -1 & & \\ & & 1 & 1 \\ & & 1 & \\ & & & 1 \end{bmatrix}.$$

Based on properties of structural matrices [29], the computational procedure for the eight-point DCT-V is represented by expression

$$\mathbf{Y}_{8\times1} = \mathbf{P}_8^{(1)}\mathbf{W}_{8\times18}\mathbf{W}_{18}^{(1)}\mathbf{W}_{18\times19}\mathbf{D}_{19}\mathbf{W}_{19\times18}\mathbf{W}_{18}^{(0)}\mathbf{W}_{18\times16}\mathbf{W}_{16\times8}\mathbf{P}_8^{(0)}\mathbf{X}_{8\times1}, \tag{25}$$

where

$$\mathbf{W}_{18\times16} = \mathbf{I}_8\mathbf{T}_{10\times8}, \mathbf{W}_{18}^{(1)} = \mathbf{I}_2\mathbf{H}_2\mathbf{T}_4^{(1)}\mathbf{H}_2\mathbf{H}_2\mathbf{I}_6, \mathbf{W}_{18}^{(0)} = \mathbf{I}_2\mathbf{H}_2\mathbf{T}_4^{(0)}\mathbf{H}_2\mathbf{H}_2\mathbf{I}_6, \mathbf{W}_{19\times18} = \mathbf{I}_4\mathbf{T}_{5\times4}\mathbf{I}_{10}, \mathbf{W}_{18\times19} = \mathbf{I}_4\mathbf{T}_{4\times5}\mathbf{I}_{10},$$

$$\mathbf{D}_{19} = diag\left(s_0^{(8)}, s_1^{(8)}, s_2^{(8)}, s_3^{(8)}, s_4^{(8)}, s_5^{(8)}, s_6^{(8)}, s_7^{(8)}, s_8^{(8)}, s_2^{(8)}, s_9^{(8)}, s_2^{(8)}, s_3^{(8)}, s_{10}^{(8)}, s_{10}^{(8)}, s_{11}^{(8)}, s_{11}^{(8)}, s_1^{(8)}, s_1^{(8)}\right),$$

$$s_0^{(8)} = c_8, s_1^{(8)} = -c_8, s_2^{(8)} = (b_8 - f_8)/2, s_3^{(8)} = (b_8 + f_8)/2, s_4^{(8)} = s_0^{(0)},$$

$$s_5^{(8)} = s_1^{(0)}, s_6^{(8)} = s_2^{(0)}, s_7^{(8)} = s_3^{(0)}, s_8^{(8)} = s_4^{(0)}, s_9^{(8)} = -(b_8 + f_8)/2, s_{10}^{(8)} = e_8, s_{11}^{(8)} = q_8.$$

$$\mathbf{P}_8^{(0)} = \begin{bmatrix} 1 & & & & & & & \\ & & & 1 & & & & \\ & & 1 & & & & & \\ & & & & & & 1 & \\ & 1 & & & & & & \\ & & & & & & & 1 \\ & & & & 1 & & & \\ & & & 1 & & & & \end{bmatrix}, \mathbf{P}_8^{(1)} = \begin{bmatrix} 1 & & & & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & & 1 & & & & \\ & & & & & & 1 & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & & & & & 1 \end{bmatrix},$$

$$\mathbf{W}_{8\times 18} = \begin{bmatrix} 1 & & & & & & & & & & 1 & & & & & & \\ & 1 & & & & & & & & & 1 & 1 & & 1 & & \\ & & 1 & & & & & & 1 & & 1 & & 1 & & & \\ & & & 1 & & & & & & 1 & 1 & & 1 & & & \\ & & & & 1 & & & & & & 1 & & & & 1 & \\ & & & & & 1 & & & & & 1 & & & & 1 & \\ & & & & & & 1 & & & & 1 & & & & 1 & \\ & & & & & & & 1 & & & 1 & & & & 1 & \end{bmatrix},$$

$$\mathbf{T}_{10\times 8} = \begin{bmatrix} & & & 1 & & 1 & & \\ & & & & 1 & & 1 & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & & & & & & & \\ & & 1 & 1 & & & & \\ & 1 & & & & & & \\ & & & & 1 & 1 & 1 & 1 \\ & 1 & & & & & & \end{bmatrix}, \mathbf{W}_{16\times 8} = \begin{bmatrix} 1 & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & & & 1 & \\ & & & & & & & 1 \\ 1 & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & & & 1 & \\ & & & & & & & 1 \end{bmatrix},$$

A data-flow graph of the proposed algorithm of the eight-point DCT-V is presented in Figure 7. Based on this algorithm, the number of multiplications can be reduced from 64 to 19, but the number of additions is increased from 56 to 61.

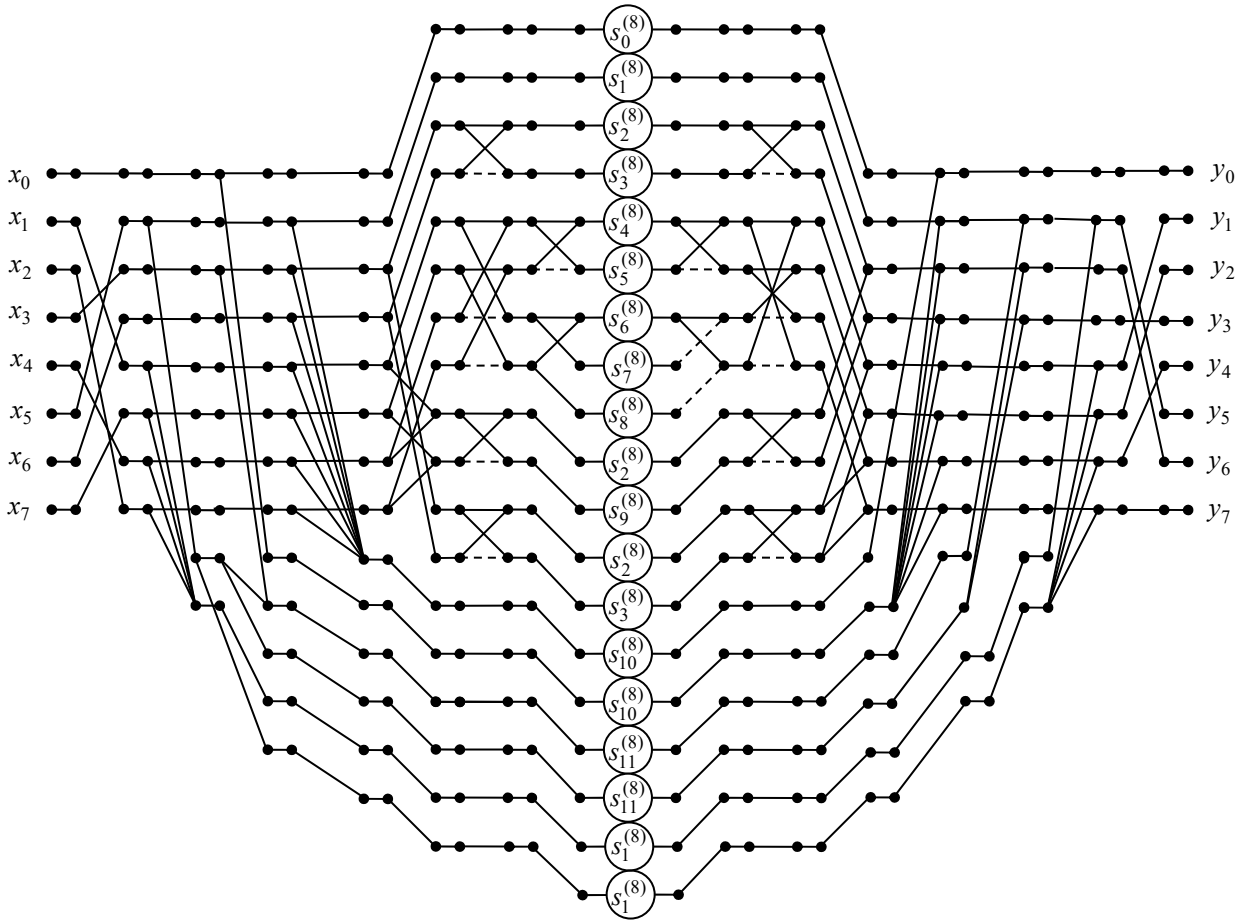

Figure 7. The data-flow graph of the algorithm for computation of eight-point DCT-V.

3. Discussion of Computational Complexity

The computational complexity of the proposed algorithms is evaluated in this section. The proposed algorithms significantly reduce the number of multiplications in DCT-V algorithms for a 1D signal with the number of samples in the range from two to eight. At the same time, the number of additions was increased by an average of 29%. The number of multiplications was reduced by an average of 57% in the range of number of samples from two to eight. The results obtained based on the elaborated algorithms are shown in Table 1. In parentheses, the percentage difference in the number of operations is indicated: plus means that the number of operations has risen compared to the direct method, and minus means that it has reduced.

The reduction in multiplications significantly contributes to speeding up the signal processing since the multiplications are more expensive to use than additions. As a result, the amount of resources used on the signal processor is significantly reduced, while allowing for easier operation in real time. The correctness of each proposed algorithm is verified by implementation in the MATLAB environment.

Analyzing the obtained results, we note the following. First, it is necessary to emphasize that the number of arithmetic operations in proposed algorithms depends on how successfully the transform matrix can be factorized. And success closely depends on the structural properties of this matrix. That is why often, in such algorithms, the reduction in the number of multiplications is achieved by increasing the number of additions. For example, in the DCT-V algorithms proposed by the authors, the number of additions increases, and not always delicately. However, if the structure of the transform matrix is "suitable", we obtain a reduction in both multiplications and additions. For example, this is the case with the developed in [28] fast DST-II algorithms.

Table 1. The number of additions and multiplications of the direct method against the proposed algorithms.

N	Direct Method		Proposed Algorithms	
	Additions	**Multiplications**	**Additions**	**Multiplications**
2	2	4	3 (+33%)	3 (−25%)
3	6	9	8 (+33%)	5 (−44%)
4	12	16	17 (+42%)	7 (−56%)
5	20	25	23 (+15%)	10 (−60%)
6	30	36	41 (+37%)	13 (−64%)
7	42	49	55 (+31%)	11 (−78%)
8	56	64	61 (+9%)	19 (70%)

Secondly, designers usually try to reduce the number of multiplications as much as possible, even at the expense of a small increase in the number of additions. The problem of reducing the number of multiplications is especially relevant when designing parallel VLSI processors. This is because, compared to a hardware adder, a hardware multiplier occupies a significantly larger area in VLSI, consumes more energy, emits more heat, and creates a very large delay.

Thirdly, this paper considers DCT-V algorithms for short-length input sequences. Processing short-length blocks of signals requires less memory to save temporary data, which is critical for mobile phones and other resource-constrained devices [35]. Fast DCT algorithms for short-length block handling allow detailed signal processing and reduce overall system complexity and delay, which is especially useful for real-time applications such as voice control systems and video conferencing [35].

The development of fast algorithms based on structural properties of transform matrices may be extended to larger input sequences processing in two ways. The first one is applying fast algorithms for short-length sequences as typical modules in synthesizing more complex algorithms [1]. Once constructed, the fast algorithms for short-length sequences can be successfully applied in various projects to unify the process of developing the final algorithm. The second way is designing the intelligent system to obtain the "suitable" matrix structure, for example, with a genetic algorithm applied [36].

4. Conclusions

In this paper, fast type-V DCT algorithms designed for processing short input data sequences were developed and investigated. Fast algorithms for short input data sequences are of particular interest because they are subsequently used as building blocks for developing fast algorithms of large-sized discrete transforms.

As a result of the literature review, two main approaches were identified for designing fast algorithms for DCT. The structural approach analyzes and considers the compositions of identical entries, using heuristic tricks to find successful transformation base matrix factorization schemes [27]. The search for successful factorization schemes largely depends on the intuition and experience of the designer of such algorithms. The analytical approach is based on searching and identifying general principles for factorizing the base transform matrix for arbitrary lengths of the input data sequences. To obtain the fast algorithms for short-length data processing, we use a structural approach. With its help, we analyzed the structural properties of the type-V DCT coefficient matrix and found algorithmic solutions that allow us to reduce the computational complexity of implementing the DCT-V transform. The calculations we carried out, as well as the comparison of the obtained solutions with naive calculation methods, prove the usefulness and cost-effectiveness of the developed algorithms. The resulting factorizations of the DCT-V matrix reduce the

number of multiplications by 57% but increase the number of additions by 29% on average in the range of signal sample numbers from two to eight.

As the future development of the research might be the implementation of the designed algorithms to elaboration of time–frequency dictionaries of functions [37,38]. Such dictionaries allow obtaining a compact representation of signals and images to increase the performance of their processing [39,40].

Author Contributions: Conceptualization, A.C.; methodology, A.C. and M.P.; software, M.P.; validation, A.C., M.P. and A.W.; formal analysis, A.C., M.P. and A.W.; investigation, M.P.; writing—original draft preparation, M.P. and A.C.; writing—review and editing, A.C. and M.P.; supervision, A.C. All authors have read and agreed to the published version of the manuscript.All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Bi, G.; Zeng, Y. *Transforms and Fast Algorithms for Signal Analysis and Representations*, 1st ed.; Birkhäuser: Boston, MA, USA, 2004.
2. Britanak, V.; Yip, P.; Rao, K. *Discrete Cosine and Sine Transforms: General Properties, Fast Algorithms and Integer Approximations*, 1st ed.; Elsevier Science: Amsterdam, The Netherlands, 2007.
3. Ramamohan Rao, K.; Yip, P. *Discrete Cosine Transform: Algorithms, Advantages, Applications*, 1st ed.; Academic Press: Cambridge, MA, USA, 2014.
4. Abramova, V.V.; Lukin, V.V.; Abramov, S.K.; Kryvenko, S.S.; Lech, P.; Okarma, K. A fast and accurate prediction of distortions in DCT-based lossy image compression. *Electronics* **2023**, *12*, 2347. [CrossRef]
5. Hnativ, L.O. Discrete cosine-sine type VII transform and fast integer transforms for intra prediction of images and video coding. *Cybern. Syst. Anal.* **2021**, *57*, 827–835. [CrossRef]
6. Lukin, V.V.; Abramova, V.V.; Abramov, S.K.; Grigelionis, K. A terahertz imaging system using adaptive DCT-based image denoising. In Proceedings of the IEEE 2nd Ukrainian Microwave Week Conference (UkrMW), Kharkiv, Ukraine, 14–18 November 2022.
7. Pogrebnyak, O.B.; Lukin, V.V. Wiener discrete cosine transform based image filtering. *J. Electron. Imaging* **2012**, *21*, 043020. [CrossRef]
8. Sugimoto, K.; Kamata, S. Fast Gaussian filter with second-order shift property of DCT-5. In Proceedings of the IEEE International Conference on Image Processing, Melbourne, Australia, 15–18 September 2013.
9. Yamamoto, K.; Toyoda, K.; Ohtsuki, T. Non-contact heartbeat detection by MUSIC with discrete cosine transform-based parameter adjustment. In Proceedings of the IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 9–13 December 2018.
10. Park, J.; Ham, J.-W.; Park, S.; Kim, D.-H.; Park, S.-J.; Kang, H.; Park, S.-O. Polyphase-basis discrete cosine transform for real-time measurement of heart rate with CW Doppler radar. *IEEE Trans. Microw. Theory Tech.* **2017**, *66*, 1644–1659. [CrossRef]
11. Tauhid, A.; Tasnim, M.; Noor, S.A.; Faruqui, N.; Yousuf, M.A. A secure image steganography using advanced encryption standard and discrete cosine transform. *J. Inf. Secur.* **2019**, *10*, 117–129. [CrossRef]
12. Krikor, L.; Baba, S.; Alnasiri, T.; Shaaban, Z. Image encryption using DCT and stream cipher. *Eur. J. Sci. Res.* **2009**, *32*, 48–58.
13. Lipinski, P.; Puchala, D. Digital image watermarking using fast parametric transforms. *Bull. Pol. Acad. Sci. Tech. Sci.* **2019**, *67*, 463–477. [CrossRef]
14. Armas Vega, E.A.; Sandoval Orozco, A.L.; García Villalba, L.J.; Hernandez-Castro, J. Digital images authentication technique based on DWT, DCT and local binary patterns. *Sensors* **2018**, *18*, 3372. [CrossRef]
15. Boukhechba, K.; Wu, H.; Bazine, R. DCT-based preprocessing approach for ICA in hyperspectral data analysis. *Sensors* **2018**, *18*, 1138. [CrossRef]
16. Ramaputra, M.G.; Irianto, S.Y.; Karnila, S. Content based image retrieval method with discrete cosine feature extraction in natural images. *J. Technol. Accept. Model* **2021**, *12*, 171. [CrossRef]
17. Zhang, Z.; Zhao, X.; Li, X.; Li, L.; Luo, Y.; Liu, S.; Li, Z. Fast DST-VII/DCT-VIII with dual implementation support for versatile video coding. *IEEE Trans. Circuits Syst. Video Technol.* **2021**, *31*, 355–371. [CrossRef]
18. Koizumi, Y.; Harada, N.; Haneda, Y.; Hioka, Y.; Kobayashi, K. End-to-end sound source enhancement using deep neural network in the modified discrete cosine transform domain. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, Canada, 15–20 April 2018.
19. Chiper, D.F.; Dobrea, D.M. A novel low-complexity and parallel algorithm for DCT IV transform and its GPU implementation. *Appl. Sci.* **2024**, *14*, 7491. [CrossRef]

20. Cariow, A.; Lesiecki, Ł. Small-size algorithms for type-IV discrete cosine transform with reduced multiplicative complexity. *Radioelectron. Commun. Syst.* **2020**, *63*, 465–487. [CrossRef]
21. Kolenderski, M.; Cariow, A. Small-size algorithms for the type-I discrete cosine transform with reduced complexity. *Electronics* **2022**, *11*, 2411. [CrossRef]
22. Chivukula, R.K.; Reznik, Y.A. Fast computing of discrete cosine and sine transforms of types VI and VII. In Proceedings of the SPIE 8135, Applications of Digital Image Processing XXXIV, San Diego, CA, USA, 19 September 2011.
23. Park, W.; Lee, B.; Kim, M. Fast computation of integer DCT-V, DCT-VIII, and DST-VII for video coding. *IEEE Trans. Image Process.* **2019**, *28*, 5839–5851. [CrossRef] [PubMed]
24. Chiper, D.F.; Cotorobai, L.T. A new approach for a unified architecture for type IV DCT/DST with an efficient incorporation of obfuscation technique. *Electronics* **2021**, *10*, 1656. [CrossRef]
25. Chiper, D.F.; Cotorobai, L.T. A novel VLSI algorithm for a low complexity VLSI implementation of DCT based on pseudo circular correlation structures. In Proceedings of the International Symposium on Electronics and Telecommunications (ISETC), Timisoara, Romania, 5–6 November 2020.
26. Chiper, D.F.; Cracan, A. An area-efficient unified VLSI architecture for type IV DCT/DST having an efficient hardware security with low overheads. *Electronics* **2023**, *12*, 4471. [CrossRef]
27. Cariow, A. Strategies for the synthesis of fast algorithms for the computation of the matrix-vector product. *J. Signal Process. Theory Appl.* **2014**, *3*, 1–19. [CrossRef]
28. Bielak, K.; Cariow, A.; Raciborski, M. The development of fast DST-II algorithms for short-length input sequences. *Electronics* **2024**, *13*, 2301. [CrossRef]
29. Cariow, A.; Paplinski, J. Algorithmic structures for realizing short-length circular convolutions with reduced complexity. *Electronics* **2021**, *10*, 2800. [CrossRef]
30. Cariow, A.; Makowska, M.; Strzelec, P. Small-size FDCT/IDCT algorithms with reduced multiplicative complexity. *Radioelectron. Commun. Syst.* **2019**, *62*, 559–576. [CrossRef]
31. Chen, X.; Dai, O.; Li, C. A fast algorithm for computing multidimensional DCT on certain small sizes. *IEEE Trans. Signal Process.* **2023**, *51*, 213–220. [CrossRef]
32. Plonka, G.; Potts, D.; Steifdl, G.; Tasche, M. Chebyshev methods and fast DCT algorithms. In *Book Numerical Fourier Analysis. Applied and Numerical Harmonic Analysis*, 1st ed.; Plonka, G., Potts, D., Steidl, G., Tasche, M., Eds.; Birkhäuser: Boston, MA, USA, 2018; pp. 339–411.
33. Radunz, A.P.; Portella, L.; Oliveira, R.S.; Bayer, F.M.; Cintra, R.J. Extensions on low-complexity DCT approximations for larger blocklengths based on minimal angle similarity. *J. Signal Process. Syst.* **2023**, *95*, 495–516. [CrossRef]
34. Korohoda, R.; Dąbrowski, A. Generalized convolution as a tool for multi-dimensional filtering tasks. *Multidimens. Syst. Signal Process.* **2008**, *19*, 361–377. [CrossRef]
35. Brysin, P.; Lukin, V. DCT-based denoising of speech signals. *Her. Khmelnytskyi Natl. Univ.* **2024**, *4*, 301–309.
36. Chen, Y.; Elliot, M.J.; Sakshaug, J.W. A genetic algorithm approach to synthetic data production. In Proceedings of the 1st International Workshop on AI for Privacy and Security, Hague, The Netherlands, 29–30 August 2016.
37. Průša, Z.; Holighaus, N.; Balazs, P. Fast matching pursuit with multi-Gabor dictionaries. *ACM Trans. Math. Softw. (TOMS)* **2022**, *47*, 24. [CrossRef]
38. Průša, Z.; Holighaus, N.; Balazs, P. Accelerating matching pursuit for multiple time-frequency dictionaries. In Proceedings of the 23rd International Conference on Digital Audio Effects (DAFx-20), Vienna, Austria, 8–12 September 2020.
39. Polyakova, M.; Ishchenko, A.; Volkova, N.; Pavlov, O. Combined method for scanned documents images segmentation using sequential extraction of regions. *East. Eur. J. Enterp. Technol.* **2018**, *5*, 6–15. [CrossRef]
40. Ishchenko, A.; Polyakova, M.; Nesteryuk, A. The technique of extraction text areas on scanned document image using linear filtration. *Appl. Asp. Inf. Technol.* **2019**, *2*, 206–215.