Jurnal Teknik Elektro Vol. 13 No. 2
Juli - Desember 2021
P-ISSN 1411 - 0059
E-ISSN 2549 - 1571

Optimized Hybrid DCT-SVD Computation over Extremely Large Images

Iwan Setiawan*, Akbari Indra Basuki, and Didi Rosiyadi

Research Center for Informatics, National Research and Innovation Agency
Jl. Cisitu No.21/154D, Bandung, 40135, Indonesia
Corresponding author. Email: iwan022@brin.go.id

Abstract— High performance computing (HPC) is required for image processing especially for picture element (pixel) with huge size. To avoid dependence to HPC equipment which is very expensive to be provided, the soft approach has been performed in this work. Actually, both hard and soft methods offer similar goal which are to reach time computation as short as possible. The discrete cosine transformation (DCT) and singular values decomposition (SVD) are conventionally performed to original image by consider it as a single matrix. This will result in computational burden for images with huge pixel. To overcome this problem, the second order matrix has been performed as block matrix to be applied on the original image which delivers the DCT-SVD hybrid formula. Hybrid here means the only required parameter shown in formula is intensity of the original pixel as the DCT and SVD formula has been merged in derivation. Result shows that when using Lena as original image, time computation of the singular values using the hybrid formula is almost two seconds faster than the conventional. Instead of pushing hard to provide the equipment, it is possible to overcome computational problem due to the size simply by using the proposed formula.

Keywords— digital image processing, discrete cosine transformation, huge image computing, singular values decomposition

I. Introduction

Inspired by the size of original images available on market today. The largest one could be 320 gig pixel which if printed in normal resolution the image would be 98 meters long and 32 meters high! [1]. On the other hand, for digital image processing purpose, advancement in HPC equipment makes computation for that size of image is possible to be performed but very expensive cost, almost not possible for researchers to provide this advance technology by themselves. Alternatively, the soft approach is a good choice without the need to provide the equipment. The novelty itself is also widely open, for example in decomposing a digital image using conventional approach, computation of singular value would be equal to size of pixel which means more pixel more computation required. For the corresponding 320 gig pixel images as mentioned earlier, the number of singular values would be 320×10^9 calculated by solving the 320 gig-order polynomial equation. This will result in computational burden. Instead of pushing hard to provide the equipment, this research focuses on the soft by looking inside the matrix to solve the computational problem. As a proof, computational problem on digital image processing could be found in many literatures, for example: complexity of decomposition matrix and its time computation [2], the cost of parallel computation for digital image processing [3], [4], and the effort to overcome the computational burden [5].

As solution, this research proposed to perform second-order block matrix to overcome computational problem on the huge image. The expected performance is the proposed method could achieve a better time computation than the conventional, so if using computer with similar specification, the proposed computation will be faster than the conventional with similar accuracy. Actually, using block matrix for SVD computation is already provided by many literatures where number of the block depend on the size of original and watermark image, for example: some literatures used block matrix with size 4×4 [6], [7] while others prefer to use 8×8 block matrix [8] - [11] or the higher one 16×16 [12] - [14]. Instead of the block, it is also possible to exploit low-high frequency band using discrete wavelet transformation to avoid the dependency [15] - [18]. Regarding this issue, the proposed method here is independent from the size although using original image blocking. Furthermore, it is also possible to expect computation with fastest computational time than the conventional because the lowest rank of matrix is second-order. By using this order, computation of singular values will be the fastest than the conventional. Using the second order matrix as a block matrix is a novelty on this work. In addition to the novelty, the second-order block matrix is not only performed to the SVD but also to the DCT. This approach could not be found in any literatures for example in [20]. Before working on numeric, the analytical work has been done with aims to synthesize the hybrid formula to compute the DCT-SVD matrix numerically. On this hybrid formula, the DCT and zigzag function have been included on derivation so the only required parameter shown in the resulted formula is matrix element of the original image. This hybrid formula is our third novelty.

II. Method

For figuring what conventional methods do in computing the DCT and SVD formula, this section is started with delivering the algorithm, continued by stating the problem

Received 2 September 2021, Accepted 6 November 2021, Published 20 December 2021.
DOI: https://doi.org/10.15294/jte.v13i2.31879

and proposing solution. Conventionally, the original or host image is divided into square block matrix. Let, we have an original image with $m \times m$ square matrix. Matrix representation of the image is shown in Figure 1.

$$\begin{bmatrix} I_{(1,1)} & I_{(1,2)} & \cdots & I_{(1,m-1)} & I_{(1,m)} \\ I_{(2,1)} & I_{(2,2)} & & I_{(2,m-1)} & I_{(2,m)} \\ \vdots & & \ddots & \vdots & \\ I_{(m-1,1)} & I_{(m-1,2)} & \cdots & I_{(m-1,m-1)} & I_{(m-1,m)} \\ I_{(m,1)} & I_{(m,2)} & & I_{(m,m-1)} & I_{(m,m)} \end{bmatrix}_{m\times m}$$

Figure 1. Matrix representation of image with size $m \times m$

Refers to [20], the DCT and SVD are performed one by one which means the method is not a hybrid. The algorithm is as follows:

1) Applying DCT to the whole $m \times m$ original image with result is called DCT-transformed host image or I_d.

$$I_d = \begin{bmatrix} I_{d(1,1)} & I_{d(1,2)} & \cdots & I_{d(1,m-1)} & I_{d(1,m)} \\ I_{d(2,1)} & I_{d(2,2)} & & I_{d(2,m-1)} & I_{d(2,m)} \\ \vdots & & \ddots & \vdots & \\ I_{d(m-1,1)} & I_{d(m-1,2)} & \cdots & I_{d(m-1,m-1)} & I_{d(m-1,m)} \\ I_{d(m,1)} & I_{d(m,2)} & & I_{d(m,m-1)} & I_{d(m,m)} \end{bmatrix} \quad (1)$$

2) Taking zigzag manner to the DCT-transformed host image as follows:

$$z = zigzag(I_d) \quad (2)$$

The result is called I_{dz} as follows:

$$I_{dz} = \begin{bmatrix} I_{d(1,1)} & I_{d(1,2)} & \cdots & \cdots & \cdots \\ I_{d(2,1)} & \boldsymbol{I_{d(3,1)}} & & \cdots & \cdots \\ \vdots & & \ddots & \vdots & \\ \cdots & \cdots & & \boldsymbol{I_{d(m-2,m)}} & I_{d(m-1,m)} \\ \cdots & \cdots & \cdots & I_{d(m,m-1)} & I_{d(m,m)} \end{bmatrix} \quad (3)$$

The bold shows the changing due to the zigzag.

3) Performing the SVD operation on I_{dz} with result as follows:

$$U, S, V = SVD(I_{dz}) \quad (4)$$

4) Performing validation formula as follows:

$$I = U \times S \times V^T \quad (5)$$

A. Problem

Matrix representation of the huge image is also huge. The huge matrix implies that computation of singular values becomes huge as number of singular values will be equal to polynomial-order. For example, let I is a matrix with size $n \times n$, representing original image with n^2 pixels.

$$\mathrm{I_{image}} = \begin{bmatrix} \mathrm{I}_{(1,1)} & \mathrm{I}_{(1,2)} & \cdots & \mathrm{I}_{(1,n-1)} & \mathrm{I}_{(1,n)} \\ \mathrm{I}_{(2,1)} & \mathrm{I}_{(2,2)} & & \mathrm{I}_{(2,n-1)} & \mathrm{I}_{(2,n)} \\ \vdots & & \ddots & \vdots & \\ \mathrm{I}_{(n-1,1)} & \mathrm{I}_{(n-1,2)} & \cdots & \mathrm{I}_{(n-1,n-1)} & \mathrm{I}_{(n-1,n)} \\ \mathrm{I}_{(n,1)} & \mathrm{I}_{(n,2)} & & \mathrm{I}_{(n,n-1)} & \mathrm{I}_{(n,n)} \end{bmatrix} \quad (6)$$

To determine singular (σ) or Eigen (λ) values, determinant of matrix is required to equal to zero as follows:

$$det\left(\begin{bmatrix} I_{(1,1)} & \cdots & I_{(1,n)} \\ \vdots & \ddots & \vdots \\ I_{(n,1)} & \cdots & I_{(n,n)} \end{bmatrix}\begin{bmatrix} I_{(1,1)} & \cdots & I_{(1,n)} \\ \vdots & \ddots & \vdots \\ I_{(n,1)} & \cdots & I_{(n,n)} \end{bmatrix}^T - \lambda I_{identity}\right) = 0, \quad (7)$$

where the identity matrix ($I_{identity}$) is

$$I_{identity} = \begin{bmatrix} 1 & 0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & 0 & 1 & & & \vdots & \vdots \\ \vdots & \vdots & & \ddots & & \vdots & \vdots \\ \vdots & \vdots & & & 1 & 0 & 0 \\ 0 & 0 & \cdots & & 0 & 1 & 0 \\ 0 & 0 & \cdots & & 0 & 0 & 1 \end{bmatrix} \quad (8)$$

then (7) will result in polynomial equation as:

$$\alpha_0\lambda^n + \alpha_1\lambda^{n-1} + \cdots + \alpha_{n-1}\lambda + \alpha_n = 0 \quad (9)$$

Equation (9) implies that computation will be performed to the n^{th}-order polynomial to determine n numbers of singular values. For image with huge pixel, n is limit to infinite ($n \to \infty$), and the numbers of singular values as well. The impact is for original image with huge pixels, determining the singular values leads to computational burden, so special treatment should be carried out for example by reducing the order. This work proposes to exploit the simply second-order polynomial by performing the second-order block matrix to the original image.

B. Solution

To overcome the problem, performing block matrix with size 2×2 is taken. In this case, the matrix order will reduce from the original to the block size of matrix. The matrix order which is originally as function of $n \to \infty$ will move to only the second-order which means very easy to compute. For images with huge pixel, the following scheme is proposed. Let, I represent the original image with size $n \times n$. Performing the second-order block matrix to the original image matrix is simply as shown in Figure 2.

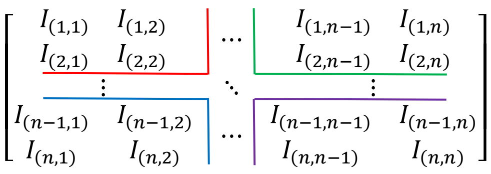


Figure 2. Performing the second-order block to the original

To formulate the hybrid DCT-SVD with second-order block matrix, the following algorithm is performed refers to flow diagram as shown in Figure 4.

1) Transform the original image from spatial to frequency domain using the DCT with formula as follows:

$$I_{dct}(x,y) = \frac{1}{\sqrt{2N}}\alpha(u)\alpha(v)\sum_{x,y=0}^{N-1} f(x,y)\cos\left(\frac{(2x+1)u\pi}{2N}\right)\cos\left(\frac{(2y+1)v\pi}{2N}\right)$$

$$\text{where: } \alpha(k) = \begin{cases} \frac{1}{\sqrt{2}}, \text{ for } k = 1 \\ 1, \text{ others} \end{cases} \quad (10)$$

This formula is applied to each 2×2 block matrix as shown in Figure 3.

$$\begin{bmatrix} \begin{bmatrix} I_{(1,1)} & I_{(1,2)} \\ I_{(2,1)} & I_{(2,2)} \end{bmatrix}_{dct} & \cdots & \begin{bmatrix} I_{(1,n-1)} & I_{(1,n)} \\ I_{(2,n-1)} & I_{(2,n)} \end{bmatrix}_{dct} \\ \vdots & \ddots & \vdots \\ \begin{bmatrix} I_{(n-1,1)} & I_{(n-1,2)} \\ I_{(n,1)} & I_{(n,2)} \end{bmatrix}_{dct} & \cdots & \begin{bmatrix} I_{(n-1,n-1)} & I_{(n-1,n)} \\ I_{(n,n-1)} & I_{(n,n)} \end{bmatrix}_{dct} \end{bmatrix}$$

Figure 3. Performing DCT on each 2×2 block matrix

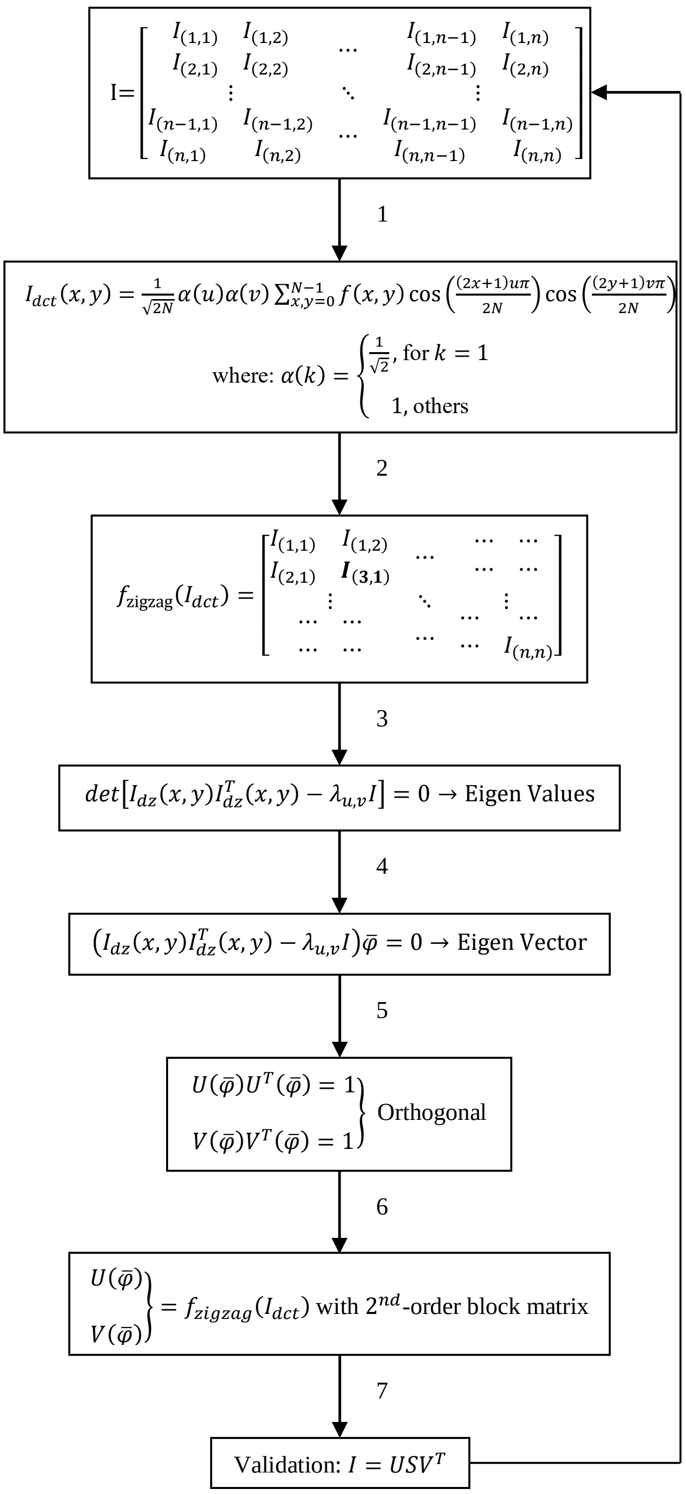

I=
1
$I_{dct}(x,y) = \frac{1}{\sqrt{2N}}\alpha(u)\alpha(v)\sum_{x,y=0}^{N-1} f(x,y)\cos\left(\frac{(2x+1)u\pi}{2N}\right)\cos\left(\frac{(2y+1)v\pi}{2N}\right)$
where: $\alpha(k) = \begin{cases} \frac{1}{\sqrt{2}}, \text{ for } k = 1 \\ 1, \text{ others} \end{cases}$
2
$f_{\text{zigzag}}(I_{dct}) =$
3
$det[I_{dz}(x,y)I_{dz}^T(x,y) - \lambda_{u,v}I] = 0 \rightarrow$ Eigen Values
4
$(I_{dz}(x,y)I_{dz}^T(x,y) - \lambda_{u,v}I)\bar{\varphi} = 0 \rightarrow$ Eigen Vector
5
$U(\bar{\varphi})U^T(\bar{\varphi}) = 1$
$V(\bar{\varphi})V^T(\bar{\varphi}) = 1$
Orthogonal
6
$U(\bar{\varphi})$, $V(\bar{\varphi})$ $= f_{zigzag}(I_{dct})$ with 2^{nd}-order block matrix
7
Validation: $I = USV^T$

Figure 4. The proposed algorithm

2) Perform zigzag function to all element of DCT matrix. As example, the zigzag operation for 4 × 4 matrix is shown in Figure 5.

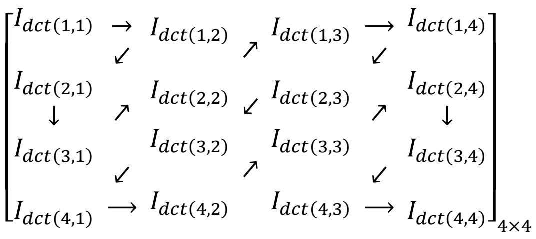


Figure 5. Example of the zigzag operation

The matrix will have new elements as follow:

$$I_{dz} = \begin{bmatrix} I_{dct(1,1)} & I_{dct(1,2)} & \mathbf{I_{dct(2,2)}} & \mathbf{I_{dct(1,3)}} \\ I_{dct(2,1)} & \mathbf{I_{dct(3,1)}} & \mathbf{I_{dct(1,4)}} & \mathbf{I_{dct(2,3)}} \\ \mathbf{I_{dct(3,2)}} & \mathbf{I_{dct(4,1)}} & \mathbf{I_{dct(2,4)}} & I_{dct(3,4)} \\ \mathbf{I_{dct(4,2)}} & \mathbf{I_{dct(3,3)}} & I_{dct(4,3)} & I_{dct(4,4)} \end{bmatrix} \tag{11}$$

Where $I_{dz} = f_{zigzag}(I_{dct})$, while the bold shows the changing in matrix element.

3) Before calculating singular values, the block matrix is performed to the DCT-transformed with zigzag matrix. For the 4 × 4 matrix, the result is shown in Figure 6 while for $n \times n$ matrix shown in Figure 7.

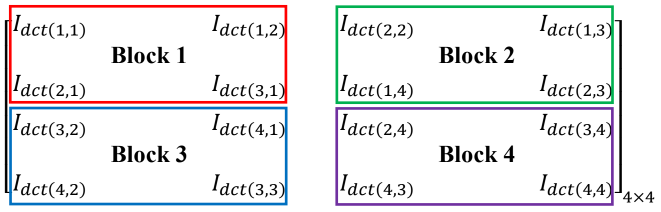


Figure 6. Performing the block to the zigzag matrix with size 4 × 4

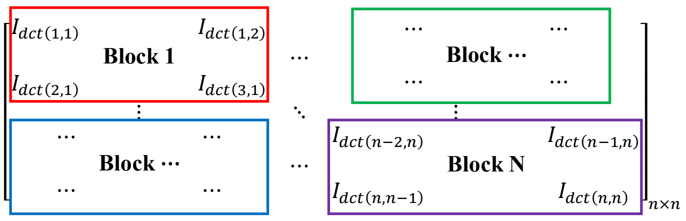


Figure 7. Performing the block to the zigzag matrix with size $n \times n$

The singular values are calculated using the following formula:

$$det[I_{dz}(x,y)I_{dz}^T(x,y) - \lambda_{u,v}I] = 0$$
$$where\ I_{dz} = f_{zigzag}(I_{dct}) \tag{12}$$

Singular values for each block are determined by solving second-order polynomial as follows:

$$\alpha_0\lambda^2 + \alpha_1\lambda + \alpha_2 = 0 \tag{13}$$

4) To determine vector U and V, solve the following Eigen vector formula:

$$(I_{dz}(x,y)I_{dz}^T(x,y) - \lambda_{u,v}I)\bar{\varphi} = 0 \tag{14}$$

5) The taken vectors U and V from (14) are dependent variables. In order to obtain independent variable, it is required to performed the orthogonal formula as follows:

$$U(\bar{\varphi})U^T(\bar{\varphi}) = 1$$
$$V(\bar{\varphi})V^T(\bar{\varphi}) = 1 \tag{15}$$

6) By solving (15), the result is obtained.

7) Validate the result by performing the following formula.

$$I = USV^T \tag{16}$$

Matrices U and V refer to decomposition matrix, while matrices S and I are singular and original matrix respectively. As summary for this section, Table I is presented.

TABLE I. COMPARISON OF THE METHOD

No	Comparison of the Method	
	Conventional SVD	**Proposed SVD**
1.	Solving the n^{th} order polynomial equation to obtain n singular values	Solving the 2^{nd} order polynomial equation to obtain n singular values
2.	Calculate Eigen vector for the huge matrix $n \times n$	Calculate Eigen vector for corresponding 2×2 matrix.
3.	Computational burden	Light computation

III. Results and Discussion

This section is started with delivering the hybrid formula as result of the analytical works, continued by validating the formula numerically using the original or host digital image. Its performance then being compared to the conventional in term of time computation. The following formula is delivered by performing the step-by-step works which is already explained in previous section.

$$U = \begin{bmatrix} U_{(1,1)} & U_{(1,2)} & \cdots & U_{(1,n-1)} & U_{(1,n)} \\ U_{(2,1)} & U_{(2,2)} & & U_{(2,n-1)} & U_{(2,n)} \\ \vdots & & \ddots & & \vdots \\ U_{(n-1,1)} & U_{(n-1,2)} & \cdots & U_{(n-1,n-1)} & U_{(n-1,n)} \\ U_{(n,1)} & U_{(n,2)} & & U_{(n,n-1)} & U_{(n,n)} \end{bmatrix} \quad (17)$$

$$V = \begin{bmatrix} V_{(1,1)} & V_{(1,2)} & \cdots & V_{(1,n-1)} & V_{(1,n)} \\ V_{(2,1)} & V_{(2,2)} & & V_{(2,n-1)} & V_{(2,n)} \\ \vdots & & \ddots & & \vdots \\ V_{(n-1,1)} & V_{(n-1,2)} & \cdots & V_{(n-1,n-1)} & V_{(n-1,n)} \\ V_{(n,1)} & V_{(n,2)} & & V_{(n,n-1)} & V_{(n,n)} \end{bmatrix} \quad (18)$$

where:

$$U_{(n-1,n-1)} = \frac{-\left(I_{dz(n-1,n-1)}I_{dz(n,n-1)} + I_{dz(n-1,n)}I_{dz(n,n)}\right)}{\sqrt{\left(I^2_{dz(n-1,n-1)} + I^2_{dz(n-1,n)} - \sigma_1^2\right)\left(\sigma_2^2 - \sigma_1^2\right)}} \quad (19)$$

$$U_{(n-1,n)} = \sqrt{\frac{\sigma_1^2 - I^2_{dz(n-1,n-1)} - I^2_{dz(n-1,n)}}{\sigma_1^2 - \sigma_2^2}} \quad (20)$$

$$U_{(n,n-1)} = \frac{I_{dz(n-1,n-1)}I_{dz(n,n-1)} + I_{dz(n-1,n)}I_{dz(n,n)}}{\sqrt{\left(\sigma_1^2 - I^2_{dz(n-1,n-1)} - I^2_{dz(n-1,n)}\right)\left(\sigma_1^2 - \sigma_2^2\right)}} \quad (21)$$

$$U_{(n,n)} = \sqrt{\frac{I^2_{dz(n-1,n-1)} + I^2_{dz(n-1,n)} - \sigma_1^2}{\sigma_2^2 - \sigma_1^2}} \quad (22)$$

$$V_{(n-1,n-1)} = \frac{-\left(I_{dz(n-1,n-1)}I_{dz(n-1,n)} + I_{dz(n,n-1)}I_{dz(n,n)}\right)}{\sqrt{\left(I^2_{dz(n-1,n-1)} + I^2_{dz(n,n-1)} - \sigma_2^2\right)\left(\sigma_1^2 - \sigma_2^2\right)}} \quad (23)$$

$$V_{(n-1,n)} = \sqrt{\frac{I^2_{dz(n-1,n-1)} + I^2_{dz(n,n-1)} - \sigma_2^2}{\sigma_1^2 - \sigma_2^2}} \quad (24)$$

$$V_{(n,n-1)} = \frac{I_{dz(n-1,n-1)}I_{dz(n-1,n)} + I_{dz(n,n-1)}I_{dz(n,n)}}{\sqrt{\left(\sigma_1^2 - I^2_{dz(n-1,n-1)} - I^2_{dz(n,n-1)}\right)\left(\sigma_1^2 - \sigma_2^2\right)}} \quad (25)$$

$$V_{(n,n)} = \sqrt{\frac{\sigma_1^2 - I^2_{dz(n-1,n-1)} - I^2_{dz(n,n-1)}}{\sigma_1^2 - \sigma_2^2}} \quad (26)$$

A. Validation

For numerical validation purpose, the following original Lena image has been used.

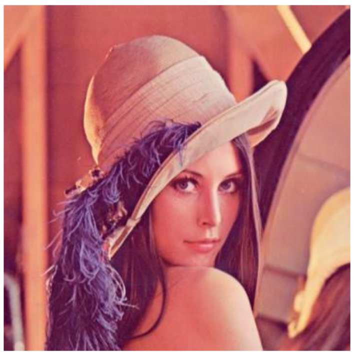


Figure 8. The original Lena image

Converting the RGB to the greyscale image will change Figure 8 to Figure 9 with matrix representation shown in (27).

Figure 9. Greyscale of the image

$$I_{image} = \begin{bmatrix} 161 & 162 & \cdots & 155 & 127 \\ 162 & 163 & & 157 & 128 \\ \vdots & & \ddots & & \vdots \\ 43 & 47 & \cdots & 106 & 111 \\ 43 & 48 & & 102 & 110 \end{bmatrix}_{512\times512} \quad (27)$$

The following pseudo-codes shown in Figure 10 have been performed to the original image. By running the codes, multiplication of $U \times S \times V^T$ is achieved as follows:

$$U \times S \times V^T = \begin{bmatrix} 161 & 162 & \cdots & 155 & 127 \\ 162 & 163 & & 157 & 128 \\ \vdots & & \ddots & & \vdots \\ 43 & 47 & \cdots & 106 & 111 \\ 43 & 48 & & 102 & 110 \end{bmatrix}_{512\times512} \quad (28)$$

Multiplication of SVD matrix in (28) is absolutely the same with the original image as shown in (27), this means the required validation has been achieved and (17) to (26) is valid numerically, also prove that the idea of using second-order matrix as block matrix is also valid.

```
1. Input: the original image
   ≫ Read the $n \times n$ matrix element of the image
2. Loop: to compute the DCT, to zigzag the DCT values,
   and to compute singular values and SVD matrix in a
   single loop.
   ≫ For first to $n^{th}$ matrix element of original matrix
   do:
           ≫ Set a blank cell with $2 \times 2$ block matrix
           ≫ For each block do:
                   ≫ Address 1 to 2 rows every time
                   ≫ Address 1 to 2 Columns every time
                   ≫ Read element matrix for each block
                   ≫ Compute DCT values
                   ≫ Apply zigzag function
               END FOR
           ≫ Get the DCT-transformed zigzag matrix
           ≫ For the DCT-transformed zigzag matrix do:
                   ≫ Set a $2 \times 2$ block matrix
                   ≫ Compute $2^{nd}$-order singular values
                   ≫ Compute matrix $U$ and $V$ using the
                     formula
               END FOR
       END FOR
3. Output: to get matrix $U$, $S$, and $V$, including validation
   ≫ Get the SVD matrix
   ≫ Calculate $I = U * S * V^T$ for validation
```

Figure 10. The pseudo-codes

B. Performance Analysis

After validating the formula analytically and numerically as describe in previous section, the next step is measuring its performance which is measured by calculating time computation for the hybrid and for the conventional. The comparison result is reported in Table II.

TABLE II. COMPARISON OF TIME COMPUTATION

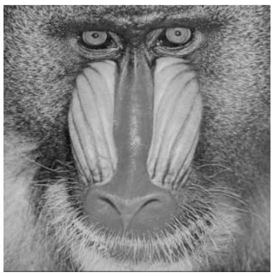
Measured Parameter	Time Computation (seconds)	
	Conventional	Proposed
Singular Values (*S*)	2.100781	0.206700
Decomposition Matrix (*U*, *V*)	2.547371	2.145800

The result shows that time computation for DCT-SVD using the hybrid formula to determine decomposition matrix (*U*, *V*) is faster than the conventional and the singular values as well. The explanation is as follow: comparison of DCT-SVD computation between the conventional and the proposed method means comparing the conventional algorithm presented by (1) to (5) to the proposed algorithm as shown in Figure 4. In this context, we would like to know which one is faster, is it computation of a single matrix with n^{th}-order or n times loop of the 2^{nd}-order block matrix, and the answer is the last one. For the singular values, the conventional computes (9) while the proposed method computes (13) as reported in Table III. It makes sense that computing second-order polynomial equation is faster than the n-order. In addition to the results, Table IV shows performance of the optimized hybrid DCT-SVD computation over different original images.

TABLE III. COMPARISON OF THE EQUATION

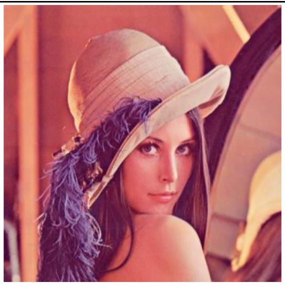
Equation of the singular values	
The Conventional	**The Proposed**
$\alpha_0\lambda^n + \alpha_1\lambda^{n-1} + \cdots + \alpha_{n-1}\lambda + \alpha_n = 0$	$\alpha_0\lambda^2 + \alpha_1\lambda + \alpha_2 = 0$

TABLE IV. TIME COMPUTATION FOR DIFFERENT IMAGES

No	Original image with size 512×512	Time computation (seconds)	
		Conventional	Proposed
1.	Zelda	2.550163	2.158441
2.	Boat	2.523884	2.126979
3.	Barbara	2.477470	2.094580
4.	 Baboon	2.459073	2.072076

While for images with higher resolution, the performance can be found in Table V. By comparing its time computation, difference between the conventional and proposed method applied for the picture with size 512×512, 1024×1024, and 2048×2048 are approximately 0.4 seconds, 2 seconds, and 10 seconds respectively. This means the time computation will continue to rise with 5 times slower when the resolution increased two times. So we can predict that the difference could be 50 seconds for the image with 4096×4096 pixel, 250 seconds for the image with 8192×8192, and so on. This means for the extremely large image, the proposed method will be extremely faster than the conventional.

TABLE V. TIME COMPUTATION FOR HIGH RESOLUTION IMAGES

No	Original Image	Time computation (seconds)	
		Conventional	Proposed
1.	 Size: 512×512	2.547371	2.145800
2.	Size: 1024×1024	12.918915	10.865629
3.	Size: 2048×2048	57.942025	47.151266

IV. CONCLUSION

The hybrid DCT-SVD formula has been carried out by performing the second-order block matrix to the DCT-SVD, and validated numerically using digital image. Its performance is determined by comparing time computation between the conventional and proposed method. With this reduced size of matrix, time computation of the decomposition matrix is comparable but with faster computation of singular values. This means the research objective which is to reach time computation as short as possible for the singular values has been achieved. For further work, improving time computation of decomposition matrix using this hybrid formula is open to be researched.

REFERENCES

[1] J. Martin, “BT Tower Test Gigapixel Panorama,” 360cities.net, para. 2012.

[2] D. Hartman, M. Hladik, and D. Riha, “Computing the spectral decomposition of interval matrices and a study on interval matrix powers,” *Applied Mathematics and Computation*, vol. 403, Aug 2021.

[3] Y. K. Kim, Y. Kim, and C. S. Jeong, "RIDE: real-time massive image processing platform on distributed environment,” *Eurasip Journal on Image and Video Processing*, no. 1, Jun 2018.

[4] S. Saxena, S. Sharma, and N. Sharma, “Parallel image processing techniques, benefits and limitations,” *Research Journal of Applied Sciences, Engineering and Technology*, vol. 2, pp. 223-238, Jan 2016.

[5] Q. Guo, C. Zhang, Y. Zhang, and H. Liu, “An Efficient SVD-Based Method for Image Denoising,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 5, May 2016.

[6] M. Du, T. Luo, L. Li, H. Xu, and Y. Song, “T-SVD-based robust color image watermarking,” *IEEE Access*, vol. 7, Nov 2019.

[7] F. Ernawan, M. N. Kabir, “A block-based RDWT-SVD image watermarking method using human visual system characteristics,” *Visual Computer*, vol. 36, no. 1, pp. 19–37, Jun 2020.

[8] L. Zhang, D. Wei, “Dual DCT-DWT-SVD digital watermarking algorithm based on particle swarm optimization,” *Multimedia Tools and Applications*, vol. 78, no. 19, Jul 2019.

[9] H. S. Shihab, S. Shafie, A. R. Ramli, and F. Ahmad, “Enhancement of satellite image compression using a hybrid (DWT–DCT) algorithm,” *Sens Imaging*, vol. 18, no. 1, Nov 2017.

[10] S. K. Ahmed, “A Comparison of the methods used for selecting singular values in image compression using SVD,” *International Journal of Computer Applications*, vol. 181, no. 1, Jul 2018.

[11] M. Al-qdah, “Secure watermarking technique for medical images with visual evaluation, *Signal and Image Processing: An International Journal*, vol. 9, no. 1, Feb 2018.

[12] E. Gul, S. Ozturk, ”A novel triple recovery information embedding approach for self-embedded digital image watermarking,” *Multimedia Tools and Applications*, vol. 79, Aug 2020.

[13] T. K. Araghi, A. A. Manaf, “An enhanced hybrid image watermarking scheme for security of medical and non-medical images based on DWT and 2-D SVD,” *Future Generation Computer Systems*, vol. 101, Dec 2019.

[14] X. Wang, D. Ma, K. Hu, J. Hu, and L. Du, “Mapping based residual convolution neural network for non-embedding and blind image watermarking,” *Journal of Information Security and Applications*, vol. 59, Jun 2021.

[15] A. Zear, A. K. Singh, and P. Kumar, “A proposed secure multiple watermarking technique based on DWT, DCT and SVD for application in medicine,” *Multimedia Tools and Applications*, vol. 77, no. 4, Feb 2018.

[16] P. Khare, V. K. Srivastava, “A secured and robust medical image watermarking approach for protecting integrity of medical images,” *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 2, Mar 2021.

[17] A. K. Singh, M. Dave, and A. Mohan, “Hybrid technique for robust and imperceptible multiple watermarking using medical images,” *Multimedia Tools and Applications*, vol. 75, no. 14, Jul 2016.

[18] A. K. Singh, “Improved hybrid algorithm for robust and imperceptible multiple watermarking using digital images,” *Multimedia Tools and Applications*, vol. 76, no. 6, Apr 2017.

[19] D. Singh, S. K. Singh, “DWT-SVD and DCT based robust and blind watermarking scheme for copyright protection,” *Multimedia Tools and Applications*, vol. 76, no. 11, Jul 2017.

[20] A. R. Yuliani, and D. Rosiyadi, "Copyright protection for color images based on transform domain and luminance component," in *Proc International Conference on Information Technology Systems and Innovation (ICITSI)*, 2016, pp. 1-4.