Quick sequential search algorithm used to decode high-frequency matrices

SIDDEQ, Mohammed M., RASHEED, Mohammed H., SALIH, Omar M. and RODRIGUES, Marcos <http://orcid.org/0000-0002-6083-1303>

World Academy of Science, Engineering and Technology
International Journal of Electrical and Computer Engineering
Vol:17, No:8, 2023

Quick Sequential Search Algorithm Used to Decode High-Frequency Matrices

Mohammed M. Siddeq, Mohammed H. Rasheed, Omar M. Salih, Marcos A. Rodrigues

***Abstract*—**This research proposes a data encoding and decoding method based on the Matrix Minimization algorithm. This algorithm is applied to high-frequency coefficients for compression/encoding. The algorithm starts by converting every three coefficients to a single value; this is accomplished based on three different keys. The decoding/decompression uses a search method called QSS (Quick Sequential Search) Decoding Algorithm presented in this research based on the sequential search to recover the exact coefficients. In the next step, the decoded data are saved in an auxiliary array. The basic idea behind the auxiliary array is to save all possible decoded coefficients; this is because another algorithm, such as conventional sequential search, could retrieve encoded/compressed data independently from the proposed algorithm. The experimental results showed that our proposed decoding algorithm retrieves original data faster than conventional sequential search algorithms.

***Keywords*—**Matrix Minimization Algorithm, Decoding Sequential Search Algorithm, image compression, Discrete Cosine Transform, Discrete Wavelet Transform.

I. Introduction

THE high-frequency matrices came from discrete transforms (i.e., Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT) and Discrete Fourier Transform (DFT)). The benefits of these transformed matrices are the conversion of image and signals to a special domain of high-frequency data (coefficients), where the transformed matrices can be used for example, for encoding/compressing data, networking media transmission and storage, and image recognition [2], [11] among other applications.

The Matrix Minimization algorithm used to encode high-frequency coefficient matrices used in image compression has been used by Siddeq since 2010 to reduce high frequency coefficients quantity [1], [21]. To recover the original data, where the sequential search algorithm represents the inverse of the matrix minimization algorithm. Previous results showed the capability of the matrix minimization method to compress images up to 90% compression ratios for grey scale images [3], [22].

The Matrix Minimization method is based on three random keys (integer/floating point). This means that if the key values are floating point numbers, then the output will also be floating point. The disadvantage of the sequential search algorithm is decoding complexity, which is based on searching for the missing data in a local range (i.e. the local range between minimum and maximum values in a matrix) [4], [5], [23], [24].

In this paper we will focus on increasing the speed of the sequential search algorithm by using an auxiliary array. The idea behind the auxiliary array is to reduce the sequence of repeated search, once data are decoded. These data will be saved in the auxiliary array to be used later for decoding (i.e., without the need to perform any further sequential search). Fig. 1 shows the layout of our proposed algorithm.

Mohammed Siddeq is with Northern Technical University, Iraq (e-mail: mohammedsiddeq1976@gmail.com).

II. Discrete Transforms

A discrete transform represents a mathematical equation or group of equations that are applied to data as an efficient tool in digital signal processing to transform the data from one domain (e.g., time domain) to another (e.g., frequency domain) [12], [25]. Additionally, discrete transforms can also be usefully applied in digital image compression. Many books and articles [8], [13], [10], [14] discuss the concept and their applications for example: DFT, DCT, DWT and DST [6], [15], [26]. Any kind of images (correlated) can be encoded by transforming it to become de-correlated. The idea behind this step of 'Compression Achievement' is to ascertain that those values obtained from discrete transforms are smaller than the correlated original pixel values [7], [16], [27]. On the other hand, we can increase compression by applying a quantization process on the transformed values, as used in JPEG or JPEG2000 [8], [17], [28].

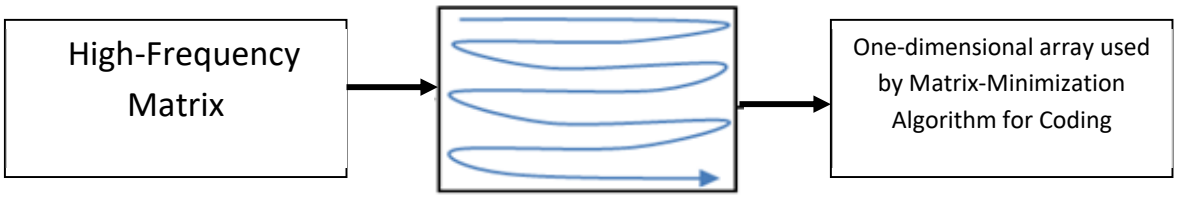
In methods using discrete transforms, the decompression/ decoder reads transformed values from the compressed stream and re-constructs the approximately original correlated pixel values by using the inverse transform. Each of the aforementioned discrete transforms has their own inverse transform [9], [18], [29].

The meaning of de-correlated values is that the transformed values are independent from each other. Besides this, most of these values are insignificant, this means that we can remove such values or replace it by 'zeros' [19], [20]. Of course, this simple step helps to encode independently and makes it simpler to organize a statistical model for image compression. Fig. 2 shows the DCT and DWT discrete transform values as an 8x8 block [10], [30].

III. Matrix Minimization Algorithm

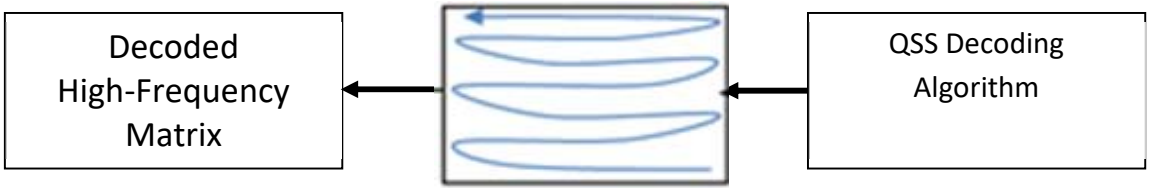
This algorithm was proposed by Siddeq in 2010 [1], and developed by Siddeq and Rodrigues at Sheffield Hallam University since 2014 [4] to shrink high frequency data to another set of encoded data. This operation depends on the *Random-Weight-Values* (generated by user or by a random

generator number – these values represented as a key for encoding/encryption). Each random weight value is multiplied with each adjacent coefficients (the number of random weights and coefficients are 3), yielding a new set of data (i.e., an encoded value). Fig. 3 (List-A) describes the steps in the Matrix Minimization Algorithm.



Row-by-Row scans to convert matrix to one-dimensional array

(a) Matrix Minimization algorithm is applied to encode High-Frequency matrix



One-dimensional array converted to matrix by relocate using row-by-row scan

(b) The QSS Algorithm is applied to decode the High-Frequency matrix

Fig. 1 Layout of encoding and decoding of high-frequency matrix

20	5	−3	1	3	−2	1	0
−3	−2	1	2	1	0	0	0
−1	−1	1	1	1	0	0	0
−1	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

(a) Block 8x8 (DCT)

63	-34	49	10	7	13	-12	7
-31	23	14	-13	3	4	6	-1
15	14	3	-12	5	-7	3	9
-9	-7	-14	8	4	-2	3	2
-5	9	-1	47	4	6	-2	2
3	0	-3	2	3	-2	0	4
2	-3	6	-4	3	6	3	6
5	11	5	6	0	3	-4	4

(b) Block 8x8 (DWT)

Fig. 2 8x8 block

List-A

```
Let K=3
  W=generate_random_number (K) %% generate three different random values
   p=1;
For i=1 to column size
    For  j=1 to row size
        Array [p]=high_Frequency_Matrix[i,j]  %% Scan row-by-row to convert matrix into 1D array
        p++;
   End
End
     j=1; p=1;
          While (j<row size*column size)
```

$$M(p) = \sum_{i=1}^{K} W(i) \times Array(i + j - 1)$$

```
          j=j+k;
          p++;
          End
```

Fig. 3 Matrix Minimization Algorithm

In the List-A the weight values are in the range {0…1} multiplied by an array (i.e., each weight represents three coefficients from the array) to produce an encoded array called "M".

Before applying the Matrix Minimization algorithm, the encoding algorithm computes the minimum and maximum value from the original data matrix. These two values are needed later in the decoding process. For example, in Fig. 2 (a) Min = -3 and Max = 20. The Matrix Minimization algorithm is applied to the example in Fig. 2 (a): Array = [20 -3 -1 -1 0 0 0 0 5 -2 -1 0 0 0 0 0 -3 1 1 0 0 0 0 0 1 2 1 1 0 0 0 0 3 1 1 0 0 0 0 0 -2 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0] to produce M = [12.66, -0.8147, 4.56, -2.53, 0, -1.803, 0.8147, 0, 3.53, 0.8147, 2.74, 1.72, 0, -1.81, 0, 0, 0.8147, 0, 0, 0, 0, 0] where W(1) = 0.8147, W(2) = 0.9058, and W(3) = 0.9134.

IV. The Proposed QSS Decoding Algorithm

From the code illustrated in List-A, the output of Matrix Minimization algorithm is the encoded/compressed data and the "Min_Max" values generated from the original matrix (as illustrated in the previous section). The main use of the "Min_Max" values is that these two values constrain the search area, starting at "Min" and finishing at "Max".

Siddeq and Rodrigues in 2014 [6] used a sequential search algorithm to decode the data stream by using the same random weight and same Min_Max search area. However, this decoding algorithm has some drawbacks:

1- If the length between Min and Max is more than 256, this means a long execution time for the decoding process.
2- The decoding algorithm is based on S1, S2 and S3 values (representing the 3 encoded values multiplied by their keys) to find the actual original data. Because the algorithm performs a blind search, it starts again from the beginning for each new value, increasing overall complexity and

execution time of the algorithm.

For this reason, we added auxiliary array to the search algorithm to speed up the outcome. Proposed decoding algorithm is based on the following steps:

1- Starting to pick up encoded data from the encoded array, and the use S1, S2 and S3 to find the original data by using [10]:

$$Estimated_{output} = S1 * W(1) + S2 * W(2) + S3 * W(3) \quad (1)$$

2- The search performed on S1, S2 and S3 works respectively as a clock. Firstly, S1, S2 and S3 are increment by one respectively then (1) computes the estimated output. In case the estimated output matches the selected encoded data (already available in auxiliary array), this means that S1, S2 and S3 represent the original three (i.e. no need for further search next time it is needed).

3- Otherwise, if the estimated output from (1) did not match, in this case S1, S2 and S3 continue incrementing by one respectively. At each iteration, (1) is tested with the selected encoded data until the estimated output is found.

4- We select the next encoded data, if it is available in the Auxiliary Array, the data are decoded immediately without needing to search S1, S2 and S3 again by the sequential search algorithm.

5- We repeat steps (1), (2), (3) and (4) until all encoded data are decoded.

6- At the end, we delete the Auxiliary Array as it is not needed anymore.

To illustrate our proposed decoding algorithm, we assume the encoded array "M" (Example from Section 3) are: M = {12.66, -0.8147, 4.56, -2.53, 0, -1.803, 0.8147, 0, 3.53, 0.8147, 2.74, 1.72, 0, -1.81, 0, 0, 0.8147, 0, 0, 0, 0, 0}

We need the following information for decoding:

1- W(1) = 0.8147, W(2) = 0.9058, W(3) = 0.9134.

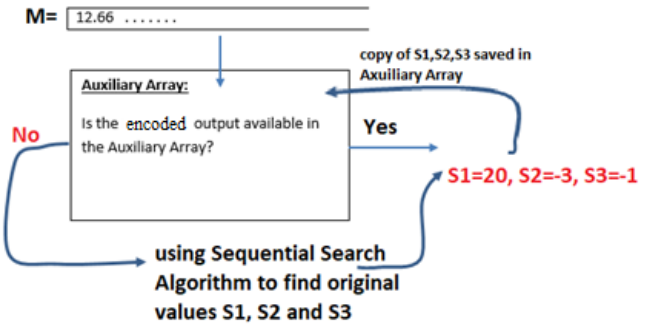
2- Min_Max = [-3, 20].

The initial set values: S1 = -3, S2 = -3 and S3 = -3. Because the minimum value is "-3", upper bound for S1, S2 and S3 is "20" (i.e., maximum value) [2]. Table I shows the status of S1, S2 and S3 until the estimated output matches the selected encoded data.

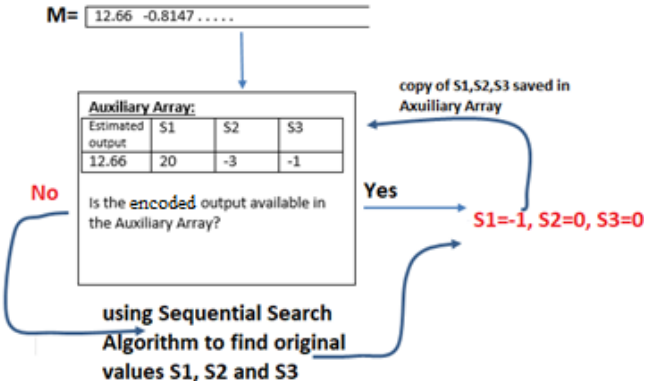
TABLE I
STATUS OF S1, S2 AND S3 IN THE SEQUENTIAL SEARCH ALGORITHM TO FIND FIRST ENCODED DATA

S1	S2	S3	Estimated output	Encoded data	Iteration	Note
-3	-3	-3	-7.9017	12.66	1	S1 (increment)
-2	-3	-3	-7.087	12.66	2	S1 (increment)
-1	-3	-3	-6.2723	12.66	3	S1 (increment)
0	-3	-3	-5.4576	12.66	3	S1 (increment)
1	-3	-3	-4.6429	12.66	4	S1 (increment)
			Continue searching….			
20	-3	-3	10.8364	12.66	24	S1 (increment)
-3	-2	-3	-6.9959	12.66	25	S1(reset to minimum), S2 (increment)
-2	-2	-3	-6.1812	12.66	26	Continue: S1 (increment)
			Continue searching….			
20	**-3**	**-1**	12.66	12.66	1176	Matched (**Stop process**)

From Table I values, S1, S2 and S3 represent the original coefficients of the matrix after 1176 iterations for just the first encoded data. This is not efficient, as if we had hundreds or thousands more encoded data, certainly takes more execution time and many iterations. For this reason, our proposed decoding algorithm saves every time S1, S2 and S3 after it has found the matched encoded data. The decoding steps are shown in Fig. 4.



(a) 1st encoded output "M" was not found in the auxiliary array, using sequential search algorithm after 1176 iterations



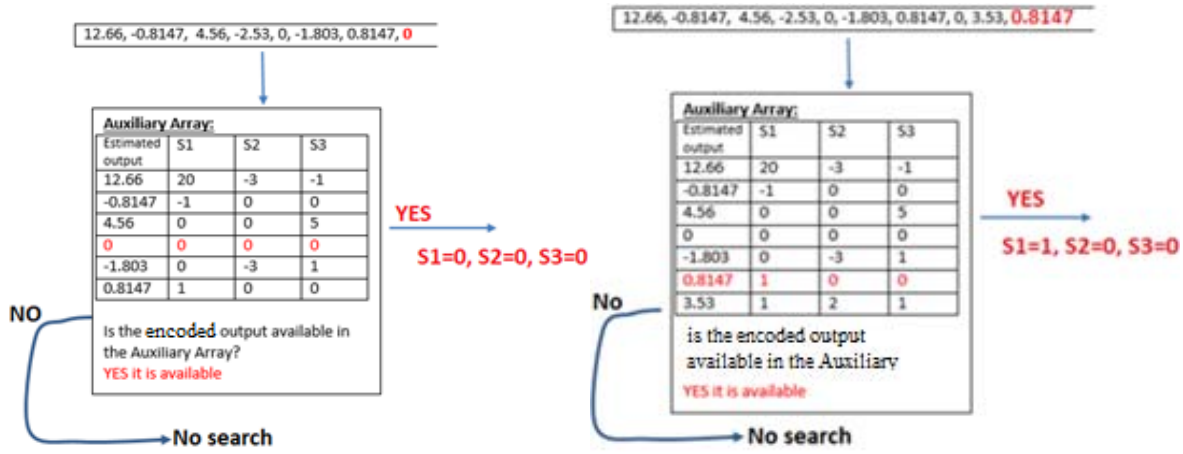
(b) 2nd encoded output "M" was not found in the auxiliary array, using sequential search algorithm after 1803 iterations

Fig. 4 QSS algorithm, also the saved and recovered coefficients data in the auxiliary array

Fig. 4 shows 1st and 2nd items from encoded data "M" decoded by our proposed QSS Algorithm. All encoded data from "M" are decoded in the same way. But if we take a look to the encoded data "M" at 8th position and 10th position these data can be quickly decoded without using the sequential search. This is so because previously those two encoded data are decoded (see location 5th location = "0" and 7th location = "0.8147"). Fig. 5 shows the quick reconstruction for S1, S2 and S3 without search.

According to example in Fig. 2 (a), our proposed method decodes the matrix much faster than a conventional sequential search algorithm. This is because the auxiliary array plays a main role in fast recovery. Also, the type of the block 8x8 is very useful to speed up the decoding algorithm (i.e., the high frequency coefficients are decoded much faster than the low frequency coefficients). On the other hand, the 8x8 block in Fig. 2 (b) has very low frequency coefficients. Equation (1) is

applied yielding the encoded data: Estimated$_{(output)}$ = [36.94, -9.12, -24.897, 25.025, 4.592, 66.133, -11.1505, 7.557, -14.589, 50.917, 8.569, 10.626, 8.71, 15.49, -2.034, 6.545, -1.60, 0.632, 5.214, 9.164, 10.73, 3.258] (here we used the same keys values as before). This type of coefficients makes our proposed decoding algorithm not suitable for quick decoding. The auxiliary array is not useful in this case and the proposed decoding algorithm takes a long time to decode as shown in Table II.



(a) 8th encoded data are decoded (b) 10th encoded data are decoded

Fig. 5 Quick reconstruction

TABLE II
COMPARISON BETWEEN TWO DECODING METHODS ACCORDING TO TIME EXECUTION

Sample	Conventional Sequential Search Decoding Algorithm	The QSS Decoding algorithm
Fig. 2 (a)	Decode time = 8.7 sec.	Decode time=2.9 sec.
Fig. 2 (b)	Decode time = 1.85 min.	Decode time = 1.85 min.

V. EXPERIMENTAL RESULTS AND COMPARISON

The encoding and decoding algorithms are applied to different types of greyscale images as shown in Fig. 6. Our proposed algorithm has been developed in MATLAB, and the computer specification is CPU AMD Ryzen7-4800H speed 3.9 MHz, and RAM DDR4 16Gbytes. In this paper we divided the results in two parts:

- Firstly, the images are divided into 8x8 blocks and then transformed by DCT followed by uniform quantization value (i.e., same quantization applied to all blocks in same image). After that, the Matrix Minimization algorithm is applied for encoding [1].
- Secondly, the same images are transformed by DWT using three level scale transformation, each level is quantized independently by a uniform quantization value (i.e., same uniform quantization value applied to each level). Each level is encoded independently by the Matrix Minimization algorithm [3].

The results show the ability of the encoding algorithm to compress images without using compression tools (Huffman or arithmetic coding) [9]. Because the Matrix Minimization algorithm converts each three coefficients to a single floating-point value which represents data compression [1], [2] (see (1)), Tables III and IV show the encoding/compression by the Matrix Minimization algorithm. Also, Figs. 7 and 8 show the decoded images by our proposed QSS Decoding Algorithm.

Fig. 6 Different size of images: Lena image size 1 Mbyte, Apple image size 1.37 Mbytes, Woman image size 257 Kbytes

Our proposed algorithm is compared with conventional Sequential Search Algorithm, to show the performance on decoding. As we know firstly, the DCT and quantization process is applied, then Matrix Minimization algorithm is applied. Secondly, Conventional Sequential Search Algorithm is applied to decode images. Table V shows the average time execution for conventional search algorithm applied.

TABLE III
DCT APPLIED TO EACH IMAGE FOLLOWED BY MATRIX MINIMIZATION AND QSS DECODING ALGORITHM

Image name	Image Dimension	Image size	Quantization Value	Encoded size	Average iterations	QSS Decoding Average time	PSNR
Lena	1024x1024	1 Mbytes	10	275 Kbytes	98921	38.9 Sec.	45.09
			25	151 Kbytes	46928	20.8 Sec.	43.1
			45	104 Kbytes	30575	9.4 Sec.	41.6
Apple	1200x1200	1.37 Mbytes	10	267 Kbytes	89062	18.7 Sec.	46.7
			25	157 Kbytes	47666	11 Sec.	44.3
			50	103 Kbytes	29641	7.2 sec.	42.4
Woman	512x512	257 Kbytes	10	179 Kbytes	56384	12.2 Sec.	43.8
			25	104 Kbytes	31413	7.3 sec.	41.0
			45	71 Kbytes	20528	3.1 sec.	39.2

Quantization = 10 Quantization = 25 Quantization = 45

(a) QSS Decoding Algorithm applied for decode Lena's image, according to quantization value

Quantization = 10 Quantization = 25 Quantization = 50

(b) QSS Decoding Algorithm applied for decode Apple's image, according to quantization value

Quantization = 10 Quantization = 25 Quantization = 45

(c) QSS Decoding Algorithm applied for decode woman's image, according to quantization value

Fig. 7 Propsoed QSS Decoding Algorithm applied to decode images

TABLE IV
THREE LEVELS DWT APPLIED TO EACH IMAGE FOLLOWED BY MATRIX MINIMIZATION AND QSS DECODING ALGORITHM

Image name	Image Dimension	Image size	Quantization Value	Image Encode size	Average iterations	Our proposed Decoding Average time	PSNR
Lena	1024x1024	1 Mbytes	10	230 Kbytes	89234	19.4 sec.	45.6
			25	105 Kbytes	26402	10.2 sec.	42.98
			45	65.4 Kbytes	9778	5.8 sec.	41.6
Apple	1200x1200	1.37 Mbytes	10	195 Kbytes	90212	16.9 sec.	46.8
			25	99.5 Kbytes	32670	7.4 sec.	44.5
			50	62 Kbytes	7532	4.3 sec.	42.8
Woman	512x512	257 Kbytes	10	177 Kbytes	32093	9.3 sec.	43.7
			25	94.5 Kbytes	12831	6.1 sec.	40.7
			45	53.6 Kbytes	8912	3.2 sec.	38.6

(a) Quantization values = 10, 25 and 45 respectively

(b) Quantization values = 10, 25 and 50 respectively

(c) Quantization values = 10, 25 and 45 respectively

Fig. 8 Proposed QSS Decoding Algorithm applied to decode images

Also, we will apply the same steps as explained in Table V, but this time we will replace DCT with three level DWT. Table VI shows average time decoding for conventional sequential search algorithm.

TABLE V
DCT APPLIED TO EACH IMAGE FOLLOWED BY MATRIX MINIMIZATION AND CONVENTIONAL SEQUENTIAL SEARCH ALGORITHM

Image name	Quantization value	Average number of iterations	Average time
Lena	10	770123	7.3 min.
	25	556134	5.8 min.
	45	321012	3.1 min.
Apple	10	659012	10.9 min.
	25	416012	6.3 min.
	50	129301	3.6 min.
Woman	10	231983	3.4 min.
	25	102671	1.8 min.
	45	31023	46 sec.

TABLE VI
THREE LEVELS DWT APPLIED TO EACH IMAGE FOLLOWED BY MINIMIZE-MATRIX-SIZE AND CONVENTIONAL SEQUENTIAL SEARCH ALGORITHM

Image name	Quantization value	Average number of iterations	Average time
Lena	10	170123	5.8 min.
	25	96134	3.7 min.
	45	21012	2.8 min.
Apple	10	659012	6.5 min.
	25	416012	4.8 min.
	50	129301	2.9 min.
Woman	10	231983	3.1 min.
	25	102671	1.1 min.
	45	31023	36 sec.

VI. CONCLUSION

This paper has proposed and demonstrated a method for QSS where values are kept in an auxiliary array for later reference, speeding up the decoding of high frequency coefficients. The method is suitable for data that contain many repeated values, such as high frequency coefficients. From the test results of the proposed system, the following points are noted:

1- Used two types of discrete transformations (DWT and DCT), these two transforms are applied separately to each tested image. Then, a uniform quantization value has been applied to each transformed image separately (i.e., for tested images by DCT each block 8x8 divided by a integer number for transformed image by DWT each sub-band divided a number).
2- The Matrix Minimization algorithm has been applied to encode/compress the final high frequency coefficients matrix.
3- The proposed decoding algorithm (QSS) has been used for decoding high-frequency matrices. This decoding algorithm has been demonstrated to recover original coefficients very efficiently as shown in Table III and IV.
4- The decoding algorithm is compared with Conventional Sequential Search algorithm according to time execution. Our proposed algorithm is much faster than previous Sequential Search algorithms.
5- The main differences between the previous Sequential Search algorithm and QSS Decoding algorithm is the auxiliary array which plays a main role making our proposed algorithm faster than previous ones.

A limitation of the research is concerned with low frequency coefficients. For matrices that contain large numbers of low frequency coefficients the speed of the decoding algorithm decreases. And in some cases, the speed of the QSS Decoding algorithm is approximately equivalent to the conventional sequential search algorithm (see Table II). Research is under way on methods to improve decoding speed of low frequency coefficients and results will be reported in the near future.

REFERENCES

[1] Mohammed Mustafa Siddeq, (2010), JPEG and Sequential Search Algorithm applied on Low-Frequency Sub-Band, *Journal of Information and Computing Science. 5(3). p:161-240.*
[2] M. Siddeq, "Using Two Levels DWT with Limited Sequential Search Algorithm for Image Compression," Journal of Signal and Information Processing, Vol. 3 No. 1, 2012, pp. 51-62. doi: 10.4236/jsip.2012.31008.
[3] M. M. Siddeq, G. Al-Khafaji, (2013) Applied Minimize-Matrix-Size Algorithm on the Transformed images by DCT and DWT used for image Compression, International Journal of Computer Applications, Vol.70, No. 15.
[4] M. M. Siddeq, M. A. Rodrigues (2014) A Novel Image Compression Algorithm for high resolution 3D Reconstruction, 3D Research. Springer Vol. 5 No.2.DOI 10.1007/s13319-014-0007-6.
[5] M. M. Siddeq and Rodrigues, Marcos (2015). Applied sequential-search algorithm for compression-encryption of high-resolution structured light 3D data. In: Blashki, Katherine and XIAO, Yingcai, (eds.) MCCSIS: Multi-conference on Computer Science and Information Systems 2015. IADIS Press, 195-202.
[6] Siddeq, M. M., Rodrigues, M. A. A Novel Image Compression Algorithm for High Resolution 3D Reconstruction. 3D Res 5, 7 (2014). https://doi.org/10.1007/s13319-014-0007-6
[7] Siddeq, M.M., Rodrigues, M.A. A novel high-frequency encoding algorithm for image compression. EURASIP Journal of Advanced and Signal Process. 2017, 26 (2017). https://doi.org/10.1186/s13634-017-0461-4.
[8] Rasheed, Mohammed H, Salih, Omar M, Siddeq, Mohammed M and Rodrigues, Marcos (2020). Image compression based on 2D Discrete Fourier Transform and matrix minimization algorithm. *Array*, 6 (100024).
[9] Abdullah A. Hussain, Ghadah K. AL-Khafaji and Mohammed M. Siddeq. Developed JPEG Algorithm Applied in Image Compression. IOP Conference Series: Materials Science and Engineering, Volume 928, 2nd International Scientific Conference of Al-Ayen University (ISCAU-2020) 15-16 July 2020.
[10] Siddeq, M. M., Rodrigues, M. A. DCT and DST Based Image Compression for 3D Reconstruction. 3D Res 5, 8 (2017). https://doi.org/10.1007/s13319-017-0116-0.
[11] AL-Hadithy, S.S., "Adaptive 1-d polynomial coding of c621 base for image compression", Turkish Journal of Computer and Mathematics Education (TURCOMAT), Vol. 12, No. 13, (2021), 5720-5731. https://www.turcomat.org/index.php/turkbilmat/article/view/9823
[12] Setyaningsih, E. and Harjoko, A., "Survey of hybrid image compression techniques", International Journal of Electrical and Computer Engineering, Vol. 7, No. 4, (2017), 2206. doi: 10.11591/ijece.v7i4.pp2206-2214.
[13] Kotha, H.D., Tummanapally, M. and Upadhyay, V.K., "Review on lossless compression techniques", in Journal of physics: conference series, IOP Publishing. Vol. 1228, (2019), 012007.
[14] Al-Khafaji[1], G. and Bassim, M., "Color image compression of inter-prediction base", International Journal of Computer Science and Mobile Computing, Vol. 8, No. 11, (2019), 65-70.
[15] Garg, Garima and Kumar, Raman, Analysis of Different Image Compression Techniques: A Review (February 10, 2022). Available at SSRN: https://ssrn.com/abstract=4031725 or http://dx.doi.org/10.2139/ssrn.4031725
[16] DeVore, R. A., Jawerth, B., & Lucier, B. J. (1992). Image compression through wavelet transform coding. IEEE Transactions on information theory, 38(2), 719-746.
[17] Nan, Sx., Feng, Xf., Wu, Yf. et al. Remote sensing image compression and encryption based on block compressive sensing and 2D-LCCCM. Nonlinear Dyn 108, 2705–2729 (2022). https://doi.org/10.1007/s11071-022-07335-4.
[18] Liu, H., Zhao, B., Huang, L.: A remote-sensing image encryption scheme

using DNA bases probability and two-dimensional logistic map. IEEE Access 7, 65450–65459 (2019).
[19] Huang, W., Jiang, D., An, Y., Liu, L., Wang, X.: A novel double-image encryption algorithm based on Rossler hyperchaotic system and compressive sensing. IEEE Access 9, 41704–41716 (2021).
[20] Wang, X.Q., Zhang, H., Sun, Y.J., Wang, X.Y.: A Plaintext-related image encryption algorithm based on compressive sensing and a novel hyperchaotic system. Int. J. Bifurc. Chaos 31(2), 5128–5143 (2021).
[21] M. Burrows and D.J. Wheeler, A Block-Sorting Lossless Data Compression Algorithm, Technical report 124, Digital Equipment Corporation, Palo Alto CA, 1994.
[22] T.M. Cover, Enumerative source coding, *IEEE Trans. Inf. Theory*, IT-19, 73–77, 1973.
[23] V. Dai and A. Zakhor, Lossless layout compression for maskless lithography systems, Emerging *Lithographic Technologies IV, Proceedings of the SPIE*, Vol. 3997, 2000, pp. 467–477.
[24] J. Ziv and A. Lempel,A universal algorithm for sequential data compression, IEEE Trans. Inf. Theory, IT-23, 337–343, 1977.
[25] M.J. Weinberger, G. Seroussi, and G. Sapiro, The LOCO-I lossless image compression algorithm: principles and standardization into JPEG-LS, IEEE Trans. Image Process., 9, 1309–1324, 2000.
[26] Francisco, A.P., Gagie, T., Köppl, D. et al. Correction to: Graph Compression for Adjacency-Matrix Multiplication. SN COMPUT.SCI.3,228(2022). ttps://doi.org/10.1007/s42979-022-01141-w
[27] Elgohary A, Boehm M, Haas PJ, Reiss FR, Reinwald B. Compressed linear algebra for declarative large-scale machine learning. Commun ACM. 2019;62(5):83–91.
[28] Gagie T, Gawrychowski P, Puglisi SJ. Approximate pattern matching in LZ77-compressed texts. J Discrete Algorithms. 2015;32:64–8.
[29] Lohrey M. Algorithmics on SLP-compressed strings: a survey. Groups Complex Cryptol. 2012;4(2):241–99.
[30] Bayazit U. Adaptive spectral transform for wavelet-based color image compression. IEEE Trans Circuits Syst Video Technol. 2011;21(7):983–92.