Applied Computer Science, vol. 18, no. 4, pp. 53–67
doi: 10.35784/acs-2022-28

Submitted: 2022-09-07 | Revised: 2022-11-10 | Accepted: 2022-12-03

Keywords: singular value decomposition, JPEG2000, JPEG, discrete wavelet transform, discrete cosine transform

Edyta ŁUKASIK [0000-0003-3644-9769]*, *Emilia ŁABUĆ*

ANALYSIS OF THE POSSIBILITY OF USING THE SINGULAR VALUE DECOMPOSITION IN IMAGE COMPRESSION

Abstract

In today's highly computerized world, data compression is a key issue to minimize the costs associated with data storage and transfer. In 2019, more than 70% of the data sent over the network were images. This paper analyses the feasibility of using the SVD algorithm in image compression and shows that it improves the efficiency of JPEG and JPEG2000 compression. Image matrices were decomposed using the SVD algorithm before compression. It has also been shown that as the image dimensions increase, the fraction of eigenvalues that must be used to reconstruct the image in good quality decreases. The study was carried out on a large and diverse set of images, more than 2500 images were examined. The results were analyzed based on criteria typical for the evaluation of numerical algorithms operating on matrices and image compression: compression ratio, size of compressed file, MSE, number of bad pixels, complexity, numerical stability, easiness of implementation..

1. INTRODUCTION

The problem of data compression is a major one these days: as the capabilities of computing machines increase, so does the amount of data that is collected and processed. For example, the images are showcases for websites that will or will not attract a customer to take a look at the offer presented. Not only the good quality of the images displayed, but also the time it takes for them to load must be taken into account. It is therefore natural to be concerned with reducing the size of stored files, which has a significant impact on file transfer times. Many publications have been made on data compression. This topic is repeatedly discussed at scientific conferences (Jackson & Hannah, 1993; Jinchuang, Yan & Wenli, 2009; Nasri et al., 2010; Xiao et al., 2011) and in publications (Gandhi, Patel & Prajapati, 2015; Hoffman, 1997; Pu, 2005; Salomon, Motta & Bryant, 2007; Short, Manohar & Tilton, 1994; Shukla & Prasad, 2011; Wayner, 1999).

* Department of Computer Science, Faculty of Electrical Engineering and Computer Science, Lublin University of Technology, Lublin, Poland, e.lukasik@pollub.pl

Research conducted in 2018 shows that the ratio of images to all data transmitted over the network has been growing rapidly over recent years, with over 70% of data transmitted over the network in 2019 being images (Chen et al., 2020; Karwowski, 2019). A literature survey was conducted to determine the current state of knowledge on the topic of image compression and data compression in general. This has made it possible to identify image compression problems that have not yet been sufficiently researched, implemented or described. Such an issue is, among others, the possibility of using the matrix singular value decomposition (SVD) algorithm for image file compression, hence the author's desire to do research in this field.

There are publications indicating the expediency of such a solution (Cao, 2006; Compton & Ernstberger, 2020; Swathi et al., 2017), however, they are superficial, conducted on single images, without studying the real impact of SVD application on compression ratio, so those publications do not fully cover the subject. The main purpose of image compression is to reduce the space required for image storage so it is possible to minimize the amount of hardware needed to store images, which ultimately means a significant decrease in expenditure on storage of such data (Pratt, Kane & Andrews, 1969; Shih et al., 2012). It is also worth mentioning that thanks to smaller file size, the time needed for its transmission is reduced, meaning less bandwidth used which, of course, also reduces costs while increasing productivity. It is the standard now to store in databases and transmit compressed images. The most popular formats used today for image compression are: JPEG, PNG, GIF and TIFF used to exchange files between applications or computer platforms (Karwowski, 2019). Less commonly used image file compression methods are for example: PIXAR, HEIF, JBIG, PCX, PGF, XPM, EXIF, JPEG XL, JFIF, WEBP or WBMP (Miano, 1999; Murray & VanRyper, 1996).

Data compression is the process of encoding, restructuring or otherwise modifying data to reduce its size. Essentially it involves re-encoding information using fewer bits than its original representation (Hoffman, 2012; Sayood, 2022). An image can be defined as a discrete function that assigns to the coordinates of a pixel its color (and in some image file formats, also an alpha channel, defining transparency). Each pixel can have a different color than all the others. If an exact reproduction of the original image is to be achieved, a certain minimum amount of information has to be stored. Finding this minimum is the task of lossless compression (Arps & Truong, 1994; Gong et al., 2018). Lossless compression is used wherever there is a need for an exact reproduction of the original data. However, there is not always a need for perfect image reconstruction, because man does not have a perfect sight organ. Data that cannot be seen by humans can be omitted from a digital image recording.

Lossy compression is based on this assumption. In the literature (Mammeri, Hadjou & Khoumsi, 2012; Nixon & Aguado, 2019; What's the difference between 'visually lossless' and real lossless and what does this mean for future encodes?, n.d.) it is possible to come across the concept of visually lossless compression, it means de facto lossy compression, in which the differences of the compressed image in relation to the original image are negligible in visual assessment. One of the most significant factors affecting compression is the presence of redundancy in the data (Davies, 2017; Lu & Guo, 2016; Parekh, 2021). Redundancy can be related to coding (using fewer codewords than the optimal number causes coding redundancy), prediction of pixel values based on the values of all neighboring pixels. Visual redundancy, on the other hand, is related to the fact that the human eye cannot process every frequency band. The pixels in the screen are in a grid form, so the image can be represented as a matrix of data.

Singular value decomposition (SVD) allows to approximate a data set with a large number of dimensions by reduced to the minimum number of dimensions (Dumka et al., 2020). By applying the decomposition to the matrix representing the image, redundancy can be used to the maximum extent: only the repeating part can be eliminated, so that the integrity of the image as a whole remains unchanged. The main objective of the paper is a comprehensive analysis of the possibility of using the matrix decomposition algorithm on singular values in image compression. The author has put forward the following research hypotheses:

The application of the SVD algorithm on the image matrix increases the efficiency of its compression by the JPEG2000 method. Applying the SVD algorithm on the image matrix increases the efficiency of its compression by the JPEG method. As the image dimension increases, the fraction of eigenvalues used for good quality image reconstruction decreases.

2. MATERIALS AND METHODS

2.1. Data

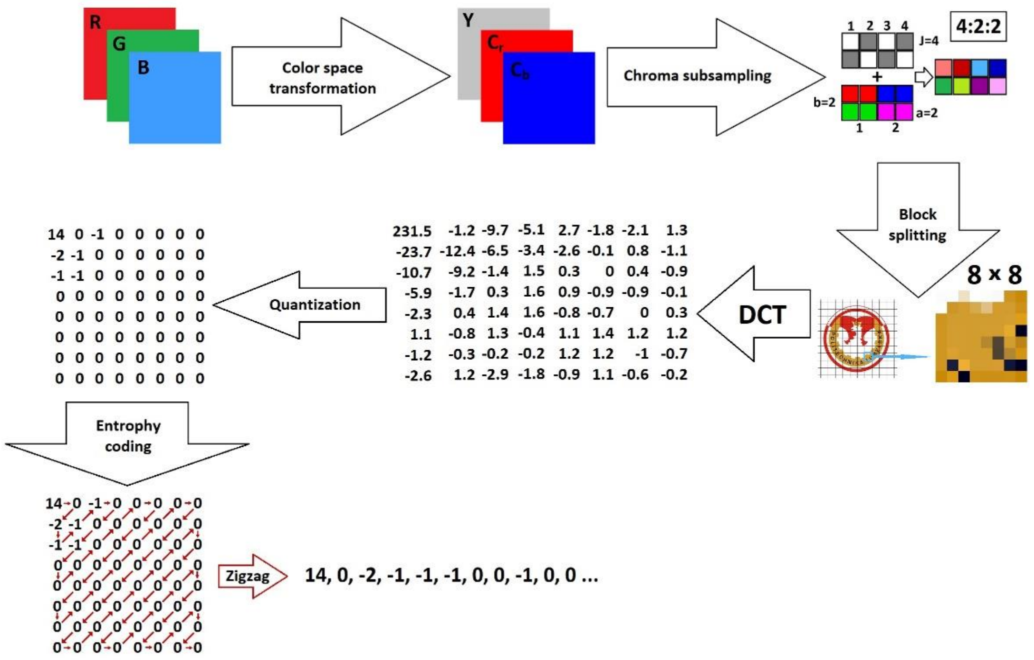
The data used to test the image compression algorithms are 2561 image files in the two most popular formats, specific to data containing image information: .BMP and .JPG. Most of the collected files are images obtained from Wikipedia under the Creative Commons license. Other images used for testing come from the author's private collection and include photos taken with different equipment and of different quality, screenshots, and digital art. Among the collected data there are binary images, monochromatic, as well as color images of different sizes. The color images vary, some are stored in high color palette. Most, however, are stored in true color with 24-bit depth, where each pixel is stored in 3 bytes. This is currently the most commonly used color depth. The smallest image is 100×67 pixels, meaning it contains information about 6700 pixels. Its size in .BMP format is 20154 bytes. The largest image examined has dimensions of 10200× 14039 pixels, containing information on more than 143 million pixels. The size of this image in .BMP format is 429 593 538 bytes. The compressed images are stored in two formats: .JPG and .JP2. The .JP2 extension, now rather rarely used, is a file format that has been compressed using the JPEG2000 method. This method allows both lossy and lossless compression (skipping the quantization step). .JP2 files are therefore larger than .JPG files.

2.2. SVD

Singular Value Decomposition is a factorization of the matrix A into three special matrices such that: $A = U \cdot \Sigma \cdot V^T$ is usually used to reduce the dimension of the original data (Britanak, Yip & Rao, 2007; Jankowska & Jankowski, 1988; Kostrikin, 2004). The SVD can be applied to image and signal processing or robotics. It is useful wherever there is a need to reduce the dimension of the original data: in statistics, in geographic data inversion, or in approximation theory. Theoretically, it can be applied to the calculation of inverse matrices (although there are better algorithms) (Cormen et al., 2005). The SVD decomposition is sometimes used in statistics for factor analysis, more specifically as an alternative to PCA (Principal Component Analysis) (Stewart, 2001). There are publications confirming he possibility of using SVD decomposition in Big Data (Wayner, 1999; Dhawan, 2011). The algorithm for finding SVD involves: computing $A \cdot A^T$, finding the eigenvalues of $A \cdot A^T$, square rooting it and placing the diagonal Σ. The matrices U and V are calculated from the normalized eigenvectors of the matrices $A \cdot A^T$ and $A^T \cdot A$ respectively.

2.3. JPEG

Generally, a JPEG file can be encoded in various ways. The encoding process usually involves several steps as shown in Fig. 1. The most interesting step is the application of the DCT (Discrete Cosine Transform). The individual pixel values are replaced by an average value within the block and an average that determines the frequencies of change within the block, both averages are expressed as floating point numbers, so although the DCT transform is reversible, some information is lost due to a fair amount of rounding. The DCT increases the number of bits needed to store the pixel data because the DCT coefficients are floating point numbers that take up more space in the computer's memory (up to 16 bits instead of the standard 8, the number of bits depends on the accuracy of the DCT calculation). Such a temporary size increase is not a problem in most JPEG implementations, because usually only a small part of the image is stored in the full DCT form. The next step - quantization reduces these values back to 8 bits.



231.5	-1.2	-9.7	-5.1	2.7	-1.8	-2.1	1.3
-23.7	-12.4	-6.5	-3.4	-2.6	-0.1	0.8	-1.1
-10.7	-9.2	-1.4	1.5	0.3	0	0.4	-0.9
-5.9	-1.7	0.3	1.6	0.9	-0.9	-0.9	-0.1
-2.3	0.4	1.4	1.6	-0.8	-0.7	0	0.3
1.1	-0.8	1.3	-0.4	1.1	1.4	1.2	1.2
-1.2	-0.3	-0.2	-0.2	1.2	1.2	-1	-0.7
-2.6	1.2	-2.9	-1.8	-0.9	1.1	-0.6	-0.2

14	0	-1	0	0	0	0	0
-2	-1	0	0	0	0	0	0
-1	-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	0	0
0	0	0	0	0	0	0	0

Fig. 1. JPEG codec example

2.4. JPEG2000

JPEG2000 is an image compression standard and encoding system developed in 2000 by the JPEG committee. Files using JPEG2000 are saved in the .jp2 format. It was originally intended to replace the previously used JPEG standard, but sluggishness of digital camera manufacturers, as well as web application developers (they didn't want to use the standard until it became more widespread) halted the development of JPEG2000 (Tanwar, Ramani & Tyagi, 2018).

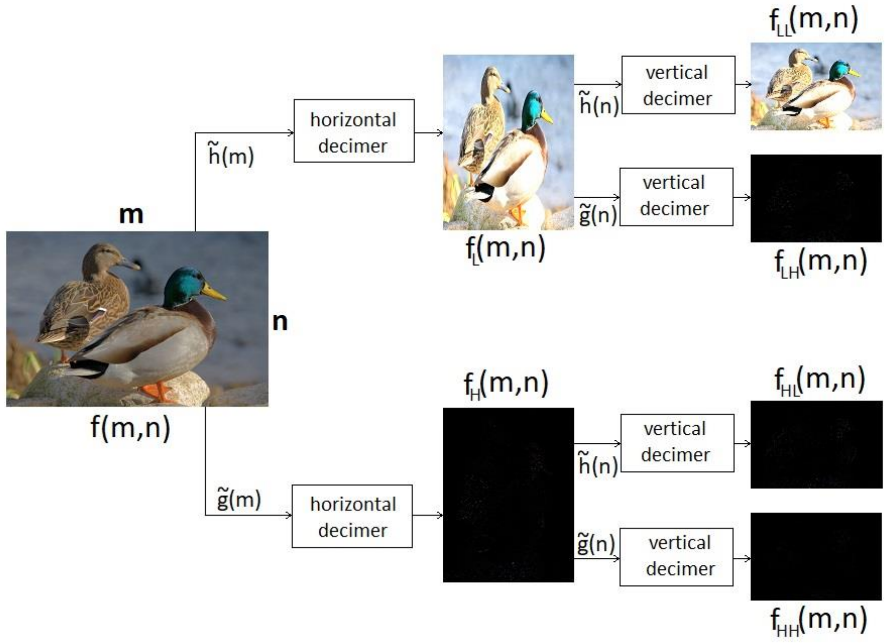


Fig. 2. Figure illustrating the idea of DWT operation on an image matrix

As a result, .JP2 is one of the less popular image formats. JPEG2000 is more flexible than JPEG because it allows both lossy and lossless compression. However, lossless compression is less efficient because it takes up more disk space. The individual compression steps of the JPEG2000 method are very similar to those of the JPEG method. The most significant difference is the use of DWT (Discrete Wavelet Transform) instead of DCT. JPEG2000 uses two types of wavelet transform. For lossless compression it is CDF (Cohen-Daubechies-Feauveau) 5⁄3 wavelet, for lossy it is CDF 9⁄7 wavelet. CDF filters built on the basis of the lifting scheme allow reversible integer transforms to be designed using biorthogonal wavelet coefficients. In this way, the error occurring in the image reconstruction is negligible and the results of the transformations are close to operations performed on a set of real numbers.

The idea of the DWT is well presented in Fig. 2. The original image is described by a binary function: as f(m,n), where m is responsible for the horizontal dimension, and n is for the vertical. Functions $\tilde{f}(m)$ and $\tilde{g}(m)$ are actually low-pass and high-pass filters. Thus, it is apparent that the image function is first filtered, and then followed by decimation, which is the halving of the horizontal dimension. Further on, both components of the image are filtered again and then vertical decimation takes place. The resulting components are in a child-parent relationship with the output image. In the next step, this action is performed on the part containing the most information, that is, the upper left corner. Usually no more than 10 layers are created in this way.

In the images: Fig. 4, Fig. 4, Fig. 6 the effect of applying the discrete wavelet transform to the duck image can be observed. The color transformation was omitted in DWT implementation. The DWT was calculated based on RGB components and the image composed of individual blocks was saved to a new file. The image shows that most of the information about the image is in the top left block. The other blocks are almost black, containing trace amounts of information.

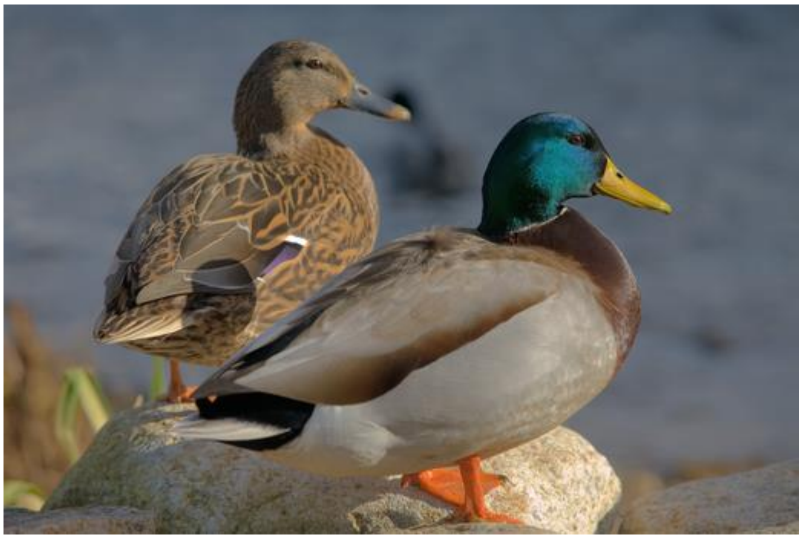


Fig. 4. Original image

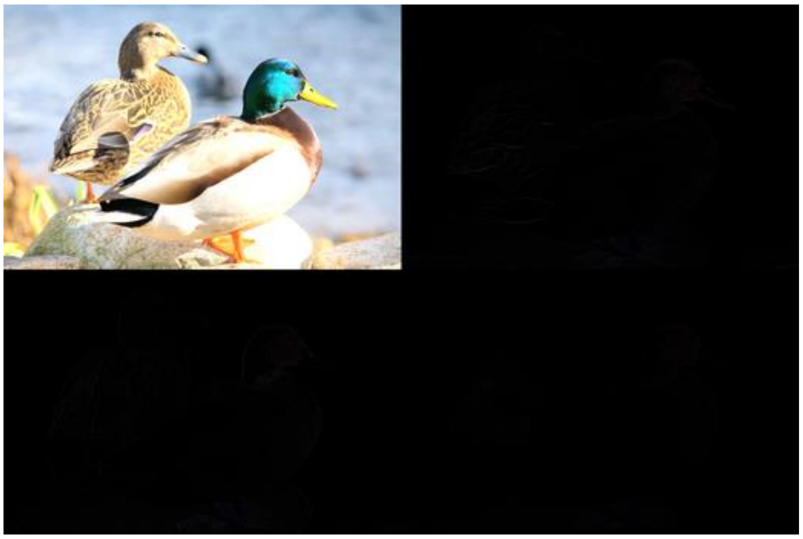


Fig. 5. Image after using first level DWT

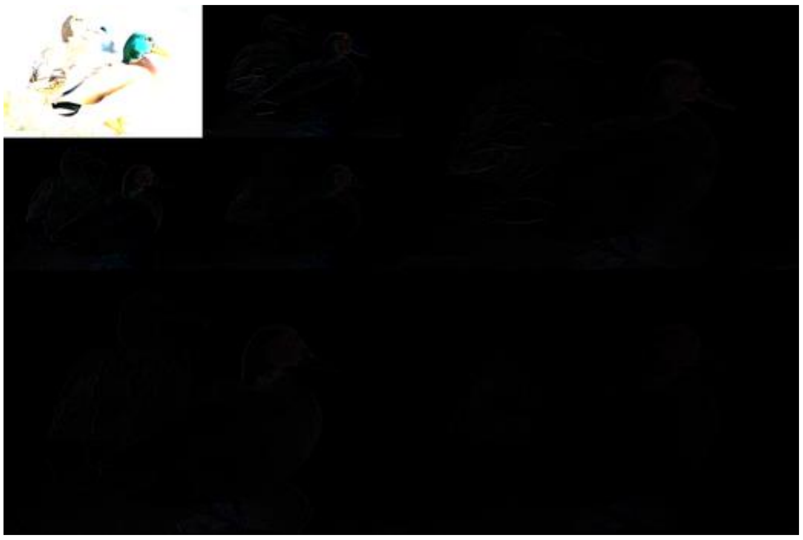


Fig. 6. Image after using second level DWT

Tab. 1. Summary of compression parameters on example images using methods: JPEG, JPEG2000

R_{orig} [B]	Width	Height	R_{JPEG} [B]	k_{JPEG} [%]	R_{JP2} [B]	k_{JP2} [%]	R_{JP2L} [B]	k_{JP2L} [%]
15116598	2592	1944	940098	93.78	8066550	46.64	8605524	43.07
180054	300	200	16278	90.96	121873	32.31	125334	30.39
204534	320	213	25953	87.31	165910	18.88	167198	18.25
24860214	3840	2158	710875	97.14	5180921	79.16	7327314	70.53
720054	600	400	58383	91.89	471683	34.49	486422	32.45
921654	640	480	113024	87.74	733174	20.45	738477	19.87
819894	640	427	95444	88.36	628631	23.33	637605	22.23
1255254	697	600	57546	95.42	500026	60.17	581918	53.64
1279254	800	533	73059	94.29	502030	60.76	591067	53.80
1279254	800	533	147850	88.44	980861	23.33	994869	22.23
499554	500	333	36821	92.63	243678	51.22	275048	44.94
13517454	2600	1733	339097	97.49	3039933	77.51	4120015	69.52
156006	228	228	10778	93.09	75535	51.58	83638	46.39
23887926	2304	3456	329120	98.62	3556418	85.11	5854096	75.49
5992758	1632	1224	191748	96.80	1702784	71.59	2188998	63.47
4316454	1598	900	153640	96.44	1062431	75.39	1341009	68.93

In the literature e.g. (Anutam & Rajni, 2014), as well as on Adobe's (Bovik, 2009) website, one can find the information that lossy JPEG2000 allows compressing the image up to two times better than regular JPEG with the same compressed image parameters. This is not true, as shown in Tab. 1.

2.5. MSE

$$MSE = \frac{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\sum_{k\in\{R,G,B\}}\left[V_O^k(i,j) - V_S^k(i,j)\right]^2}{3\cdot M\cdot N} \tag{1}$$

MSE was calculated according to the formula (1), where *M* is width of image (number of pixels), *N* is height of image (number of pixels), V_O^k the value of the *k*-th color component of the pixel lying at the position with coordinates *(i, j)* in the original image, V_S^k the value of the *k*-th color component of the pixel lying at the position with coordinates *(i, j)* in the compressed image.

2.6. Compression methods tested and verification of the result

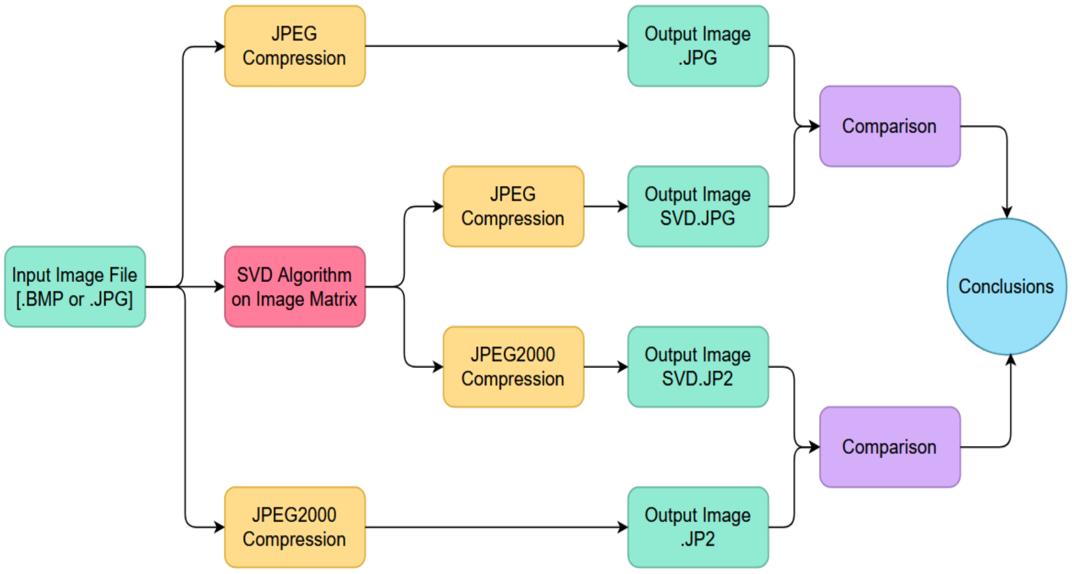


Fig. 7. Diagram showing the idea of proceeding with the verification of results

The Fig. 6 diagram illustrates the results verification procedure. Each bitmap has been compressed using JPEG and JPEG 2000 methods with and without SVD algorithm.

The comparison shown in the diagram refers to an image which was SVD processed prior to compression and one for which no SVD was used. The size of the JPEG-compressed image was compared with that of the previously SVD-applied image (same for JPEG2000 and SVD+JPEG2000).

The statistical analysis summary of the comparisons for the JPEG and SVD+JPEG methods is included in section 3.1, for JPEG2000 and SVD+JPEG2000 in section 3.2. Conclusions are provided in chapter 4.

2.7. Compression evaluation criteria

Tab. 2. Compression evaluation criteria

	JPEG	**SVD+JPEG**	**JPEG2000**	**SVD+JPEG 2000**
Lossless compression	No	No	Yes	No
Lossy compression	Yes	Yes	Yes	Yes
Flexibility	Low	High	High	Very high
Pessimistic time complexity	$O(n^2)$	$O(n^3)$	$O(n^2)$	$O(n^3)$
Expected time complexity	$O(n \log(n))$	$O(n^3)$	$O(n \log(n))$	$O(n^3)$
Memory complexity	$O(n \log(n))$	$O(n^2)$	$O(n \log(n))$	$O(n^2)$
Ease of implementation(Python)	Easy	Medium	Easy	Medium

It is not necessary to run the program to determine the above criteria. They result directly from the properties of the individual algorithms. Detailed analyses have been carried out to measure file size, compression ratio or to determine the MSE. The results will be presented in the subsequent section.

2.8. Hardware & Software

Tab. 3. Parameters of the equipment used to conduct the tests

Processor	AMD Ryzen 7 4800H with Radeon Graphics
Clock signal	2.90 GHz
Number of cores	8
Logic processors	16
Installed RAM	32.0 GB
GPU 1	NVIDIA GeForce GTX 1650
GPU 2	AMD Radeon (TM) Graphics
Operating system type	64-bit operating system, processor x64
Version of Operating System	Windows 11 Home 21H2

The important factor affecting the time of the calculations is the hardware parameters on which they are performed. In this case it was a Lenovo Legion 5 15ARH05 laptop with the parameters as shown in Tab. 3.

The second important factor is the version of the programming language used to create the image compression application: Python version 3.10.

2.9. Statistical methods used to test the hypotheses

To test the hypothesis whether the use of the SVD algorithm significantly affects the efficiency of the JPEG method, a group of 2561 different images were compressed using the methods: JPEG and SVD+JPEG. For each image, the size of the file after compression was measured for both methods. Two equinumerous series of data were obtained, which were naturally combined into pairs. Then, for each of them, the one-sided Wilcoxon test for pairs of observations with alternative hypothesis “less” was carried out at a fixed significance level of $\alpha = 0.05$. The demonstration of the impact of the SVD algorithm on the efficiency of the JPEG2000 method was carried out in a similar way. The tests were performed in the RStudio environment (R version 4.2.0 – Vigorous Calisthenics).

In order to check whether the fraction of eigenvalues required to reconstruct the image in good quality decreases as the image dimension increases, the image matrix was decomposed using the SVD algorithm. The image was then reconstructed a hundred times using successively 1%, 2%, ..., 100% of the found eigenvalues of the image matrix. The desired level of image quality was determined based on the MSE. For each compressed image, the MSE value was calculated for each fraction of the eigenvalues. The image with the MSE parameter closest to the determined MSE <= 0.5 was selected. In this way, coordinates of the points were obtained: fraction of specific eigenvalues used for reconstruction, minimum of width and height of the image, which were plotted on a graph. This study was performed on a group of 50 randomly selected images in .BMP format. The graph was generated in a Python program using matplotlib library.

3. RESULTS

The following images were created by compressing the original image using different eigenvalue fractions. The fraction of eigenvalues used to reconstruct them and their sizes after compression are shown above the individual compressed images.

Fig. 8 presents result of compression small image, 100×67 pixels. Only a small part of the results of the tests performed are included in Tab. 4.

Fig. 9 shows the result of compression performed on a medium image. Fig. 9 presented MSE depending on the value of the image’s compression ratio.

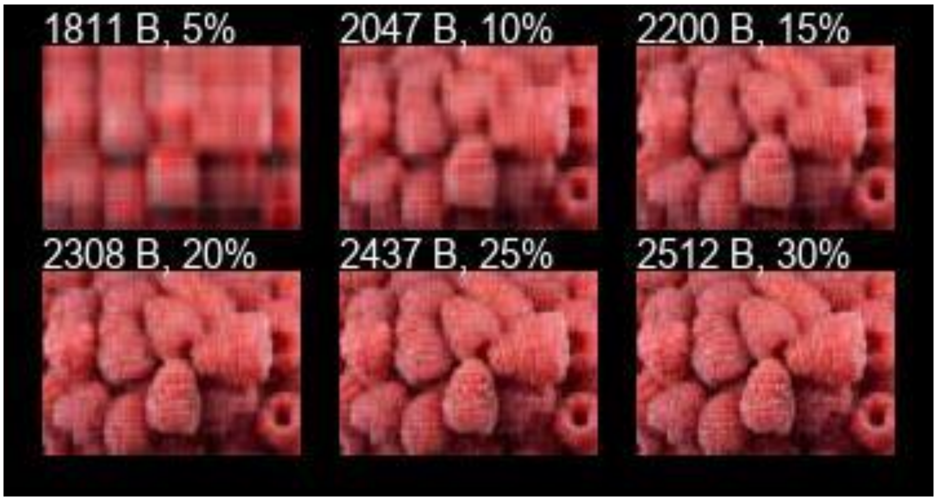


Fig. 8. Image compression using eigenvalue fractions: 5–30%. Information about the original image: dimension: 100×67, file size .BMP: 20 154 B

Tab. 4. Example results of compressing a .BMP image into .JPG format using SVD and different eigenvalue fractions

k	n	Time of compression [μs]	Size of compressed file [B]	Avg number of bad pixels	Max number of bad pixels	MSE	Compression [%]
0.01	1	39126.7000	1498	72.1179	300.0582	916.2527	92.57
0.05	3	31761.8000	1811	54.0602	242.9183	533.8225	91.01
0.10	7	28343.7000	2047	36.3769	215.8520	243.0560	89.84
0.15	10	30138.7000	2200	29.3871	177.6175	158.9510	89.08
0.20	13	28359.3000	2308	24.2515	152.8387	107.5304	88.55
0.25	17	26359.9000	2437	19.4205	120.2190	69.0755	87.91
0.30	20	26305.8000	2512	16.8861	100.7209	52.1632	87.54
0.35	23	27013.5000	2561	14.7831	78.0716	39.5877	87.29
0.40	27	26235.9000	2625	12.2134	64.1442	27.0916	86.98
0.50	34	26323.1000	2721	8.5490	46.4939	13.2923	86.50
0.60	40	33556.9000	2773	5.8081	35.4037	6.2488	86.24
0.70	47	27329.6000	2804	3.4723	23.3793	2.2468	86.09
0.80	54	32545.1000	2810	1.6939	11.7402	0.5417	86.06
0.90	60	27339.0000	2806	0.6080	4.1410	0.0787	86.08
0.95	64	33509.5000	2811	0.1766	1.7116	0.0092	86.05
1.00	67	33555.9000	2809	0.0000	0.0000	0.0000	86.06

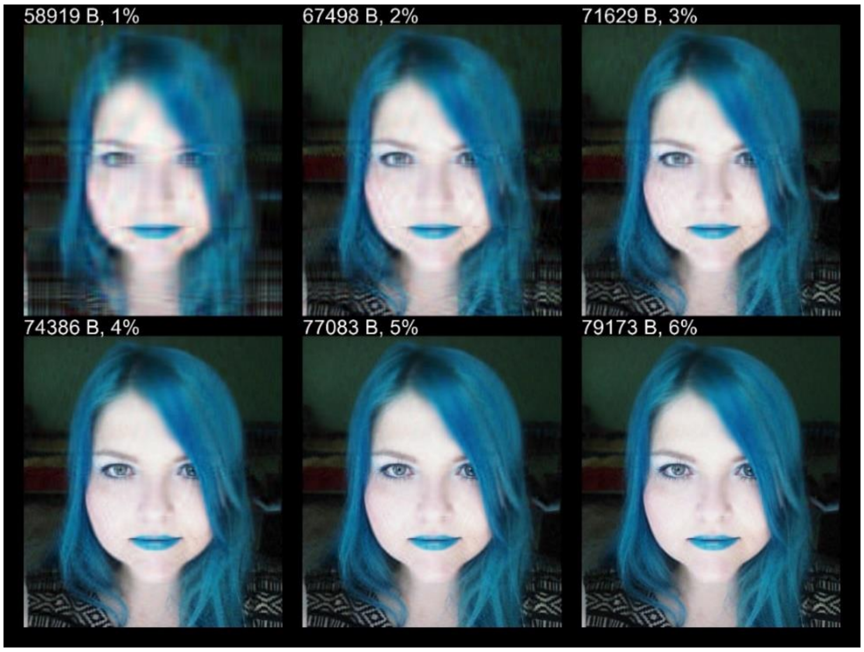


Fig. 9. Image compression using eigenvalue fractions: 1-6%. Information about the original image: dimension: 960×1081, file size .JPG: 243 728 B

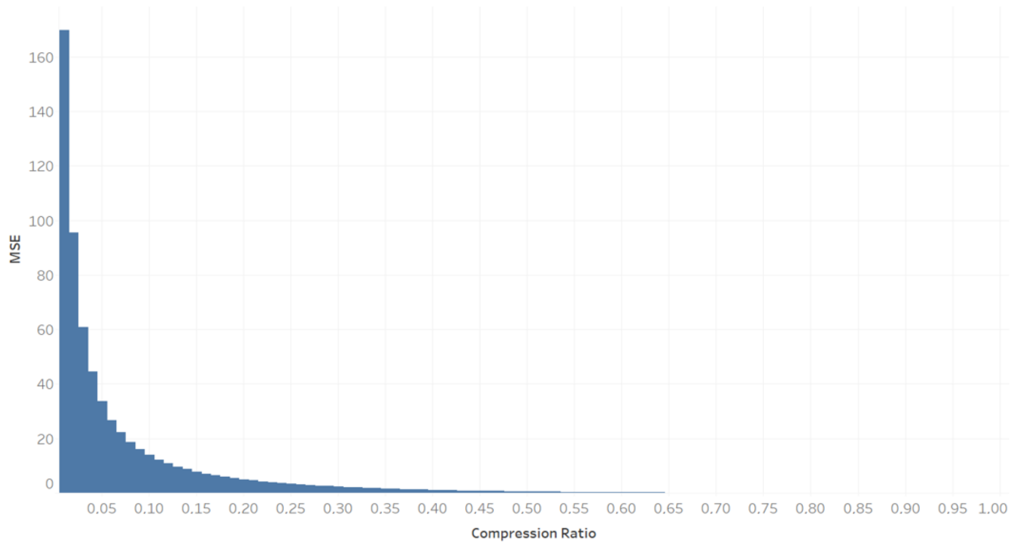


Fig. 10. MSE depending on the value of the compression ratio using SVD+JPEG for image showed on figure 8

3.1. Applying the SVD algorithm on the image matrix increases the efficiency of its compression by the JPEG method

Null hypothesis: The size of the SVD+JPEG compressed file is larger than the size of the same JPEG compressed file.

Alternative hypothesis: The size of the file compressed by SVD+JPEG method is smaller than the size of the same file compressed by JPEG method.

The result of Wilcoxon signed rank test with continuity correction:

data: size$SVDJPG and size$JPG, V = 111090, p-value < 2.2e-16.

The p-value is less than the set significance level, so the null hypothesis should be rejected in favour of the alternative hypothesis: The size of the file compressed with the SVD+JPEG method is smaller than the size of the same file compressed with the JPEG method.

3.2. The application of the SVD algorithm on the image matrix increases the efficiency of its compression by the JPEG2000 method

Null hypothesis: The file size compressed by the SVD+JPEG2000 method is larger than the size of the same file compressed by the JPEG2000 method.

Alternative hypothesis: The size of the file compressed by the SVD+JPEG2000 method is smaller than the size of the same file compressed by the JPEG2000 method.

The result of Wilcoxon signed rank test with continuity correction:

data: size$SVDJPG2000 and size$JPG2000, V = 81092, p-value < 2.2e-16.

The p-value is less than the established significance level, so the null hypothesis should be rejected in favour of the alternative hypothesis: The size of the file compressed by SVD+JPEG2000 is smaller than the size of the same file compressed by JPEG2000.

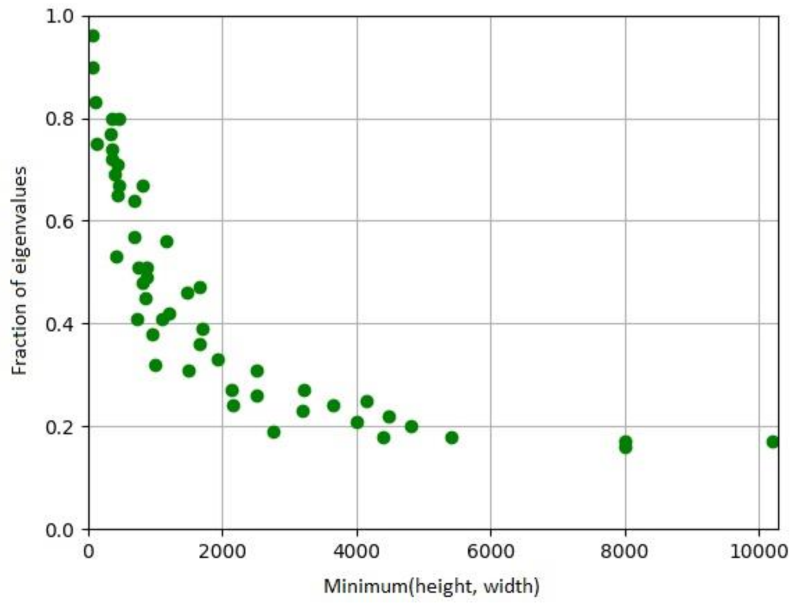


Fig. 11. Fraction of eigenvalues to be used for image reconstruction based on minimum of height and width, with fixed MSE <= 0.5

As the image dimension increases, the fraction of eigenvalues to be used for good quality image reconstruction decreases.

4. CONCLUSIONS

In conclusion, this paper analyses the possibility of using SVD matrix decomposition in image compression and discusses the performance, strengths and weaknesses of the JPEG and JPEG2000 algorithms. The analysis performed confirmed the accuracy of the hypotheses. It can therefore be concluded that the use of the SVD algorithm before compression with the JPEG and JPEG2000 methods significantly reduces the size of the output file. As presented on Fig. 8, Fig. 9 and Fig. 11, the fraction of eigenvalues to be used for image reconstruction, based on image size, decreases with the increase of dimension of image. Minimum of height and width was taken to show that relation, because it determines total number of eigenvalues, which is the same as total number of singular values of image matrix. Summarizing, JPEG is based on DCT, while JPEG2000 on DWT. In the authors opinion, the full potential of using the SVD algorithm in image compression would only be explored when creating a new standard for image file compression, in which SVD decomposition of image matrices would be one of the compression steps. In order to investigate the full power of SVD it would be necessary to write a custom image compressor.

REFERENCES

Anutam, & Rajni. (2014). Comparative Analysis of Filters and Wavelet Based Thresholding Methods for Image Denoising. *Computer Science & Amp; Information Technology (CS &Amp; IT)* (pp. 137–148). https://doi.org/10.5121/csit.2014.4515

Arps, R., & Truong, T. (1994). Comparison of international standards for lossless still image compression. *Proceedings of the IEEE, 82*(6), 889–899. https://doi.org/10.1109/5.286193

Bovik, A. C. (2009). *The Essential Guide to Image Processing* (1st ed.). Academic Press.

Britanak, V., Yip, P. C., & Rao, K. (2007). CHAPTER 4 – Fast DCT/DST Algorithms. *Discrete Cosine and Sine Transforms. General Properties, Fast Algorithms and Integer Approximations* (pp. 73–140). Academic Press. https://doi.org/10.1016/b978-012373624-6/50006-0

Cao, L. (2006). *SVD applied to digital image processing*. Division of Computing Studies, Arizona State University Polytechnic Campus.

Chen, Y., Mukherjee, D., Han, J., Grange, A., Xu, Y., Parker, S., Chen, C., Su, H., Joshi, U., Chiang, C. H., Wang, Y., Wilkins, P., Bankoski, J., Trudeau, L., Egge, N., Valin, J. M., Davies, T., Midtskogen, S., Norkin, A., de Rivaz, P., Design, A., & Liu, Z. (2020). An Overview of Coding Tools in AV1: the First Video Codec from the Alliance for Open Media. *APSIPA Transactions on Signal and Information Processing, 9*(1), e6. https://doi.org/10.1017/atsip.2020.2

Compton, E. A., & Ernstberger, S. L. (2020). Singular Value Decomposition: Applications to Image Processing. Lagrange College. *Journal of Undergraduate Research, 17,* 99–105.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2005). *Wprowadzenie do algorytmów*. Wydawnictwo Naukowe PWN.

Davies, E. R. (2017). *Computer Vision: Principles, Algorithms, Applications, Learning*. Elsevier Gezondheidszorg.

Dhawan, S. (2011). *A Review of Image Compression and Comparison of its Algorithms*. https://www.semanticscholar.org/paper/A-Review-of-Image-Compression-and-Comparison-of-its-Dhawan/034dcf50d99bbd9870c5c2e67201f6d792f96a5f

Dumka, A., Ashok, A., Verma, P., & Verma, P. (2020). *Advanced Digital Image Processing and Its Applications in Big Data* (1st ed.). CRC Press.

Gandhi, T., Patel, H., & Prajapati, D. (2015). *Image Compression Using Fractal: Image compression based upon the self-similarities present in the image*. LAP LAMBERT Academic Publishing.

Gong, L., Deng, C., Pan, S., & Zhou, N. (2018). Image compression-encryption algorithms by combining hyper-chaotic system with discrete fractional random transform. *Optics &Amp; Laser Technology, 103,* 48–58. https://doi.org/10.1016/j.optlastec.2018.01.007

Hoffman, R. (1997). *Data Compression in Digital Systems*. Springer Publishing.

Hoffman, R. (2012). *Data Compression in Digital Systems*. Springer Publishing.

Jackson, D., & Hannah, S. (1993). Comparative analysis of image compression techniques. *1993 (25th) Southeastern Symposium on System Theory* (pp. 513–517). IEEE. https://doi.org/10.1109/ssst.1993.522833

Jankowska, J., & Jankowski, M. (1988). *Przegląd metod i algorytmów numerycznych*. Wydawnictwa Naukowo-Techniczne.

Jinchuang, Z., Yan, T., & Wenli, F. (2009). Research of image compression based on Wireless visual sensor networks. *4th International Conference on Computer Science & Education* (pp. 353–356). IEEE. https://doi.org/10.1109/iccse.2009.5228430

Karwowski, D. (2019). *Zrozumieć kompresję obrazu: podstawy technik kodowania stratnego oraz bezstratnego obrazów*. Damian Karwowski.

Kostrikin, A. I. (2004). *Wstęp do algebry cz. 1 i cz. 2 Podstawy algebry*. Wydawnictwo Naukowe PWN.

Lu, Z., & Guo, S. (2016). *Lossless Information Hiding in Images* (1st ed.). Syngress.

Mammeri, A., Hadjou, B., & Khoumsi, A. (2012). A Survey of Image Compression Algorithms for Visual Sensor Networks. *International Scholarly Research Notices, 2012*, 760320. https://doi.org/10.5402/2012/760320

Miano, J. (1999). *Compressed Image File Formats: JPEG, PNG, GIF, XBM, BMP*. Addison-Wesley Professional.

Murray, J. D., & VanRyper, W. (1996). *Encyclopedia of Graphics File Formats*. O'Reilly & Associates.

Nasri, M., Helali, A., Sghaier, H., & Maaref, H. (2010). Energy-efficient wavelet image compression in Wireless Sensor Network. *2010 International Conference on Wireless and Ubiquitous Systems* (pp. 1–7). IEEE. https://doi.org/10.1109/icwus.2010.5670430

Nixon, M., & Aguado, A. (2019). *Feature Extraction and Image Processing for Computer Vision* (4th ed.). Academic Press.

Parekh, D. (2021, April 25). *Image Compression Standards | Digital Image Processing* [Video file]. YouTube. https://www.youtube.com/watch?v=6IuKH7IGspU

Pratt, W., Kane, J., & Andrews, H. (1969). Hadamard transform image coding. *Proceedings of the IEEE*, *57*(1), 58–68. https://doi.org/10.1109/proc.1969.6869

Pu, I. M. (2005). *Fundamental Data Compression* (1st ed.). Butterworth-Heinemann.

Salomon, D., Motta, G., & Bryant, D. (2007). *Data Compression: The Complete Reference*. Springer Publishing.

Sayood, K. (2002). *Lossless Compression Handbook*. Elsevier Gezondheidszorg.

Shih, C. W., Chu, H. C., Chen, Y. M., & Wen, C. C. (2012). The effectiveness of image features based on fractal image coding for image annotation. *Expert Systems With Applications, 39*(17), 12897–12904. https://doi.org/10.1016/j.eswa.2012.05.003

Short, M. N., Manohar, M., & Tilton, J. C. (1994). Planning/Scheduling Techniques for VQ-Based Image Compression. Science Information Management and Data Compression Workshop. *1994 Science Information Management and Data Compression Workshop* (pp. 95–104). US Government.

Shukla, K. K., & Prasad, M. V. (2011). *Lossy Image Compression: Domain Decomposition-Based Algorithm*s. Springer Publishing.

Stewart, G. W. (2001). *Matrix Algorithms: Volume 2, Eigensystems* (1st ed.). SIAM: Society for Industrial and Applied Mathematics.

Swathi, H. R., Sohini, S., Surbhi, & Gopichand, G. (2017). Image compression using singular value decomposition. *IOP Conference Series: Materials Science and Engineering, 263,* 042082. https://doi.org/10.1088/1757-899x/263/4/042082

Tanwar, S., Ramani, T., & Tyagi, S. (2018). Dimensionality Reduction Using PCA and SVD in Big Data: A Comparative Case Study. *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering* (pp. 116–125). Springer. https://doi.org/10.1007/978-3-319-73712-6_12

Wayner, P. (1999). *Compression Algorithms for Real Programmers*. Elsevier Gezondheidszorg.

What’s the difference between ‘visually lossless’ and real lossless and what does this mean for future encodes? (2019, May 18). *Video Production Stack Exchange*. Retrieved May 2022 from https://video.stackexchange.com/questions/27656/whats-the-difference-between-visually-lossless-and-real-lossless-and-what-doe

Xiao, F., Zhang, P., Sun, L. J., Wang, J., & Wang, R. C. (2011). Research on image compression and transmission mechanism for wireless multimedia sensor networks. *2011 International Conference on Electrical and Control Engineering* (pp. 788–791). IEEE. https://doi.org/10.1109/iceceng.2011.6057601