RIFD Fibonacci Zeckendorf Hybrid Encoding and Decoding Algorithm for Medical Image Compression and Reconstruction

Authors	Salem, Nema;Elnaggar, Fathy
DOI	10.1109/MCNA50957.2020.9264295
Publisher	IEEE
Download date	2025-05-13 04:48:02
Link to Item	http://hdl.handle.net/20.500.14131/611

2020 Fourth International Conference on Multimedia Computing, Networking and Applications (MCNA)

RIFD Fibonacci Zeckendorf Hybrid Encoding and Decoding Algorithm for Medical Image Compression and Reconstruction

Nema Salem*, Fathy Elnaggar†
Electrical and Computer Engineering, Effat University
Jeddah, KSA
Email: nsalem@effatuniversity.edu.sa
†*Mechanical Engineering, Alexandria University*
Alexandria, Egypt
Email: Fathyf3@yahoo.com

***Abstract*—Digital medical images are an important source of information that help doctors in diagnosis and treatment. The raw form of a digital image requires a tremendous amount of storage memory and a longer time for transmission from one node to another through a limited bandwidth network. Thus, many algorithms are developed and implemented for image compression that eliminates the redundant information while keeping the essential ones. The decoding algorithms are able to extract this essential information and reconstruct the original images without losing the original image quality. Rounding a pixel's Intensity Followed by a Division process is called $RIFD$ algorithm. It minimizes the information redundancy of the images and maintains the image visual quality with insignificant distortion. The integration of Fibonacci sequence and Zeckendorf's theorem produces a simple and fast variable length code for representing integer data. This article proposes a hybrid encoding and decoding algorithm for medical image compression by merging the $RIFD$ and the suffix variable length second order Fibonacci− Zeckendorf codes. Performance metrics such as number of bits per pixel, compression ratio, memory saving percentage, run time, mean square error, peak signal to noise ratio and similarity structure index are used in evaluating the efficiency of the proposed algorithm on nine medical images. The simulation results shows the efficiency of the proposed algorithm especially in compressing images with non uniform distributed histograms.**

***Index Terms*—medical image compression, Fibonacci codes, Zeckendorf's theorem, gray levels range reduction, performance metrics, $RIDF$**

I. INTRODUCTION

Recent years have seen an explosion in the availability of digital images as a result of the increasing numbers of digital imaging devices such as smart−phones, web−cams, digital−cameras, and scanners. Transmitting large−size image from one network−node to another requires longer transfer time, larger space and higher cost. In the health−care industry, there is an urgent need for storing the images of the patients with reduced expenses, fast accessibility, and the ability of sharing these images between clinicians over the internet without any information loss.

Though the cost of digital memory capacity continues to decrease, and disk drives with high capacities are becoming common, the sizes of digital images continue to increase as well. Thus, the need to efficiently process, store, and transfer images in digital form has motivated the development of many image compression algorithms. The principles of image compression require an understanding of the fundamental concepts of information theory, which were developed by Claude Shannon in the late $1940s$, with later contributions by many authors.

In signal processing, image compression or bit−rate reduction means encoding image using fewer bits than the original representation by minimizing or eliminating at least one of the redundancies which are coding, inter−pixel, and psycho−visual. The researchers developed several compression algorithms for medical images that are broadly categorized into *lossy*, *lossless*, and *hybrid*. The *lossy* compression reduces bits by eliminating the less important information which leads to loss in some information. While the *Lossless* compression reduces bits by eliminating statistical redundancy with no information loss in a way that the reconstructed image is almost similar to the original one. Thus, this type of compression is widely used for medical image compression [1]. The *hybrid* compression algorithms includes more than one compression technique to achieve an effective compression ratio with a high image visual quality.

In [2], the Indian Mathematician Leonardo Fibonacci introduced the recursive sequence of numbers that hold his name. As in coding a one to one mapping is required, thus Fibonacci coding can be considered as a simple and fast coding system with suffix delimiters. Fibonacci coding is the result of merging Fibonacci sequence with the integer Zeckendorf representation. In [3], the authors proposed a technique that increases the similarity among the neighboring pixels of images by rounding the pixels' intensities followed by a division process $RIFD$. Their algorithm gave a higher compression ratio than that has been achieved by Huffman. This work proposes a hybrid compression algorithm for medical images. In the encoding phase, the algorithm starts with reducing the gray−level of the image−pixels followed by applying the second−order Fibonacci−Zeckendorf coding resulting in a

978-1-7281-8373-2/20/$31.00 ©2020 IEEE

stream of bits with a lower size than the original image. The visual and structure quality of the decoded images are tested using many metrics.
The organization of the paper is as follows. Section 2 is about literature survey based on previous and present trends in the field of compression. Section 3 reviews the Round Intensity Followed by Division algorithm $RIFD$, Fibonacci sequence, Fibbinary numbers, Zeckendorf's theorem, and Fibonacci universal coding and decoding system. Section 4 presents different performance metrics. Section 5 exhibits the proposed hybrid image compression approach. Section 6 illustrates and discusses the experimental results while the last section concludes the work.

II. LITERATURE REVIEW

There is a big foundation of researchers who developed many compression algorithms such as Shannon [4], [5], Huffman [6], Lempel and Ziv [7], and Welch [8]. In [9], the authors compressed medical images using Discrete Cosine Transform *DCT*, Singular Value Decomposition *SVD*, Discrete Hadamard Transform *DHT*, Slant Transform *ST*, and Discrete Haar Transform *DHT* while maintaining the *PSNR* in the range of $(25 - 30dB)$.Their results showed that *DCT* keeps the best energy compaction among the used the transforms while Hadamard gives better *SSIM* for the same *PSNR*. Haar is the fastest transform and SVD has best energy packing efficiency for any given image and its performance varies according to the input images. Moreover, SSIM is more reliable than MSE in terms of measuring the quality of reconstructed image from human perspective. In [10], the authors used Huffman; Discrete Wavelet Transform (DWT) coding and fractal algorithm in compressing images. Their analysis showed that Huffman coding can reduce the redundant data, while DWT is able to improve the quality of compressed image. Their results showed that Fractal algorithm provides better Compression ratio (CR) and Peak Signal to noise ratio (PSNR). In [11], the authors recommended the use of hybrid compression techniques to improve the compression efficiency of medical images. For brain MRI images, they encoded the ROI by a lossless arithmetic coding while the they heavily compressed the background region by the lossy SPIHT technique. The authors in [12] updated the LZW technique for compressing the medical images. In [13], the authors proposed a fast fractal-based compression algorithm for compressing MRI images. Their algorithm starts by converting the (3D) MRI images into a (2D) image sequence. They speed-up their algorithm by using self-similarity in a way that the number of blocks in the matching pool is reduced. Their experimental results showed improvement in the compression speed, and the PSNR.Finally, a residual compensation mechanism is introduced to achieve compression of MRI images with high decompression quality. Their experimental results showed improvement in the compression speed, and the PSNR. In [14], the authors proposed the image averaging and transform coding for the compression of images. They generated the residual images by taking the difference between the mean of all the images and original images. Then, they encoded the wavelet coefficients of the DWT transformed mean and the residual images by Huffman. Their results showed improvement in bit saving. In [15], the author implemented the write compression in flash memory using Zeckendorf two-round rewriting codes. In [16], the authors investigated the possibility of compressing XML labels by different prefix-encoding methods. They implemented Lucas coding, Elias-Fibonacci of Order ≥ 2, and Elias-Delta coding. Their results showed that the structure of an XML tree representation affects the performance of the compression methods but not the XML document size. In addition, the Elias-Delta achieved the fastest encoding time on average whilst Fibonacci of order 2 had the best decoding time and Fibonacci of order 3 produced the most compressed codes. Finally, they recommended the use of Fibonacci coding for encoding XML labels. In [17], the authors used the FibLSS encoding scheme that exploits the properties of the Fibonacci sequence to provide variable$-$length node labels of appropriate size in XML. Their results indicated that the exponential increase of the number of nodes to be labeled is associated by a linear growth of the size of labels which makes it suitable for big databases. To enhance the energy efficiency in low-powered smart meters and to reduce the RF communication time, the article [18] investigated an experimental comparative study of data compression. the authors investigated the performance of the lossless run$-$length binary encoding algorithm, Huffman coding, even$-$Rodeh, Exponential$-$Golomb, LZW, Fibonacci coding, and the hybrid Bzip2 algorithm. The performance of each compression algorithms has been verified experimentally with real metering datasets from industrial and domestic cases. In [19], the authors introduced a new numeration system which is based on variable$-$r meta$-$Fibonacci sequences and it is a generalization of the Zeckendorf numeration system. They used this new numeration system to construct binary, prefix-free, uniquely decodable universal codes called meta-Fibonacci codes.

III. REVIEW OF $RIFD$ AND FIBONACCI

A. Round Intensity Followed by Division, $RIFD$

The $RIFD$ algorithm focuses on the correlation between the neighboring pixels and the limitation of the HVS of perceiving intensities. It rounds the intensity values of the adjacent pixels to the same value in a way that the HVS can't notice the updated intensity values. For a gray image with 256 gray$-$levels from 0 to 255, the algorithm starts with rounding the pixels' intensities to a multiple of 10 that lies between 0 and 250. Therefore, the pixels of intensities $100, 101, 102, 103, 104$ and 105 are rounded to a new value of 100. While, the pixels with intensities $106, 107, 108, 109$ and 110 are rounded to a new value of 110. This step results in 26 values: $0, 10, 20, 30, \ldots$ and 250. The second step of the algorithm narrows the range of pixels' intensities between 0 and 25 that can be represented by 5 binary bits rather than 8 binary ones. This minimization of the bit$-$depth of the pixels' values leads to a high compression ratio.

B. Fibonacci Sequence and Fibbinary Numbers

The Fibonacci sequence is defined as follows.

$$\begin{aligned} &F_0 = 0; \qquad F_1 = 1 \\ &F_n = F_{n-1} + F_{n-2} \qquad n \geq 2 \end{aligned} \tag{1}$$

This Fibonacci recurrence equation is a second−degree homogeneous linear difference equation with constant coefficients. Solving it yields to two eigenvalues: λ_1 = the irrational number $(1+\sqrt{5})/2 \approx 1.618$, defined as the golden number Φ and λ_2 = the irrational number $(1-\sqrt{5})/2 \approx -0.618$, defined as the golden number conjugate $-\varphi$. Noting the property that $\varphi = \Phi - 1 = 1/\Phi$. The general solution is $C_1\Phi^n + C_2(\varphi)^n$. Substituting by the initial conditions: $F_0 = 0$ and $F_1 = 1$, leads to $C_1 = -C_2 = 1/\sqrt{5}$. The general Fibonacci recursive equation and the n^{th} Fibonacci term are expressed in equation 2 while table I presents the calculation of the first Fibonacci numbers.

$$\begin{aligned} F_n &= \frac{(\Phi^n - (-\varphi)^n)}{\sqrt{5}} \\ F_n &= \|a\Phi^n\| = \|\frac{3\sqrt{5}+5}{10} \quad \Phi^n\| \end{aligned} \tag{2}$$

TABLE I: Fibonacci Sequence Calculation

n	0	1	2	3	4	5	6
$a\Phi^n$	1.17	1.89	3.07	4.96	8.02	12.98	21.01
$\|a\Phi^n\|$	1	2	3	5	8	13	21

Fibbinary Numbers are an alternative way of representing the integer numbers by using Fibonacci numbers as bases instead of powers of two as in binary number system. Table II gives some examples for Fibbinary number presentation and shows Fibbinary representation of integer numbers is not always unique.

TABLE II: Fibbinaries are not Unique Representation.

Integer Number	**Binary 8 4 2 1**	**Fibbinary 1 2 3 5 8 13**
5	101	011 0001
6	110	1001 111
10	1010	01001 0111

C. Zeckendorf's Theorem

Zeckendorf's theorem [20], [21] states that every positive integer number N can be represented uniquely as the sum of one or more distinct Fibonacci numbers F_i in a way that the sum does not include any two consecutive Fibonacci numbers. Thus, for the integer numbers in table II, 5 ⇒ 0001, 6 ⇒ 1001, and 10 ⇒ 01001. Thus, any positive number can uniquely be represented as in equation 3 and the N^{th} Fibbinary number, *fib* (N), is defined in equation 4.

$$\begin{aligned} &N = F_{n1} + F_{n2} + \cdots + F_{nk} \\ &where \quad n_i - n_{i+1} \geq 2 \quad for \quad i = 1, 2, \cdots, k-1. \end{aligned} \tag{3}$$

$$fib(N) = \sum_{m=1}^{k} 2^{n_m - 2}. \tag{4}$$

D. Fibonacci Universal Coding Decoding System

By considering Zeckendorf's theorem, any integer number can be presented in Fibonacci bases and encoded as a string of $zeros$ and $ones$ in a reverse order together with a suffix bit of logic one. This way of number presentation makes each codeword to be ended with the pattern of two consecutive ones, 11, while not containing any other instances of this pattern before the end. Merging the simple classical second−order Fibonacci with Zeckendorf's theorem leads to a universal variable−length self−delimited Fibonacci code that doesn't need any lookup table in the decoding phase. Table III shows the first Fibonacci codes associated with their implied probability, the value for each number that has a minimum-size code in Fibonacci coding.

TABLE III: Fibonacci Codeword and Implied Probability.

Integer Number	**Fibonacci Presentation**	**Fibonacci Codeword**	**Implied Probability**
1	$F_2 \rightarrow 2^{2-2} = 2^0$	11	$1/4$
2	$F_3 \rightarrow 2^{3-2} = 2^1$	011	$1/8$
3	$F_4 \rightarrow 2^{4-2} = 2^2$	0011	$1/16$
4	$F_2 + F_4 \rightarrow 2^{2-2} + 2^{4-2} = 2^0 + 2^2$	1011	$1/16$
5	$F_5 \rightarrow 2^{5-2} = 2^3$	00011	$1/32$
6	$F_2 + F_5 \rightarrow 2^{2-2} + 2^{5-2} = 2^0 + 2^3$	10011	$1/32$

The coding algorithm of a positive integer N is as follows.

1) Initialize: $F(N) = 0$.
2) Find the i^{th} index of the largest fibonacci number such that $F_i \leq N$.
3) If $F_i \leq N$ then update N with $N - F_i$ and push the *1-bit* into stack. Otherwise, push a *0-bit*.
4) Update the index i by $i - 1$. If $i \geq 0$ then go to step 3.
5) While stack is not empty: remove a bit from the stack and put bit at the end of $F(N)$.
6) Attach the *1-bit* at the end of $F(N)$.

The decoding phase is to reconstruct the data back that's opposite to the encoding phase of work. The Fibonacci decoding algorithm is as follows.

1) Initialize: N, *Previous*, and i to $zero$.
2) Read the current bit from the input stream of data.
3) If both current bit and *Previous* are 1−bit then *END*.
4) If the current bit is 1−bit, then add F_i number to N.
5) Update i by $i + 1$ and *Previous* to the current bit. Go to step 2.

IV. PERFORMANCE METRICS

There are many evaluation parameters for a compression algorithm. The Compression Ratio CR is the ratio of the number of bits required to represent the image before compression to the number of bits required to represent the image after compression, equation 5. The Bits Per Pixel BPP is the average number of bits required to represent one pixel, equation 6. The memory−saving percentage is calculated as in equation 7.

$$CR = \frac{\text{Original Image Size}}{\text{Compressed Image Size}} \tag{5}$$

$$BPP|_{\textit{Compressed Image}} = \frac{\text{Number of Bits}}{\text{Number of Pixels}} \tag{6}$$

$$Saving\% = \frac{\text{Size before - size after}}{\text{Size before compression}} \times 100 \tag{7}$$

The Mean Square Error MSE is the mostly used metric in image processing field to represent the average pixel difference between two images. It can be easily calculated as in equation 8. The Peak Signal to Noise Ratio $PSNR$ is a measure of visual difference of two images. It represents the dissimilarity between the compressed and the original images as a ratio between the peak square gray level value 255 of the original image and the MSE. A higher value of $PSNR$ indicates better quality of reconstructed image. it can be calculated as in equation 9.

$$MSE = \frac{1}{R \times C} \sum_{i=1}^{R} \sum_{j=1}^{C} \left\{ f(i,j) - \widehat{f}(i,j) \right\}^2 \tag{8}$$

$$PSNR = 10 \ \log_{10} \left(\frac{255^2}{MSE} \right) \ dB \tag{9}$$

where $R \times C$ is the size of the image; and $f(i,j)$ and $\widehat{f}(i,j)$ represent the pixel value of the original and compressed images, respectively. The Mean Structural SIMilarity $MSSIM$ measures the image structure and information contents [22]. It includes the fact of the HVS that scans an image from the Fovea outwards and concludes about the luminance and contrast in the image. This metric mimics the fovea input by using a (11×11) or $(13 \times \times 13)$ pixel Gaussian window. Running this window pixel by pixel across the original image and compressed image, the metric calculates a weighted approximation of the difference between luminance and contrast kernels in the images at each pixel. The kernel difference is calculated as in equation 10.

$$\begin{aligned} MSSIM(x,y) &= \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{\left(\mu_x^2 + \mu_y^2 + C_1\right)\left(\sigma_X^2 + \sigma_y^2 + C_2\right)} \\ \mu_x &= \frac{1}{N}\sum_{i=1}^{N} x_i \\ \sigma_x &= \left(\frac{1}{N-1} \sum_{i=1}^{N} (x_i - \mu_i)^2 \right)^{1/2} \\ C_1 &= (K_1 \times L)^2 \\ C_2 &= (K_2 \times L)^2 \end{aligned} \tag{10}$$

where x and y are kernels from the original and compressed images. The parameters μ, σ_x, and σ_{xy} are the mean, standard deviation, the square root of variance, and the covariance of each kernel; respectively. Noting that, μ is estimation of the mean intensity, σ_x is estimation of the contrast, and the function of μ_x and μ_y is estimation of the luminance. The intensity and contract are normalized so that the two images being compared have unit standard deviation. The constants C_1 and C_2 are included to avoid instability. L is the dynamic range of the pixel values 255 for $8\,bit$ gray−scale images, and $K_1 \lll 1$, $K_2 \lll 1$ are small constants. The $SSIM$ metric results in a structural map of the differences in the images as the kernel is passed across the images and it collapses the map into a single measure of quality using the mean value.

V. PROPOSED CODING AND DECODING ALGORITHM

The raw−form of a medical image occupies a large size besides being a colored one. The proposed algorithm starts with extracting only the intensity part and resize it to a smaller size as 128×128 or 256×256. The second step of the algorithm is to reduce the intensity values from $0 - 255$ to $0 - 25$ by utilizing the $RIFD$ algorithm. The third step of the algorithm is the encoding process by applying Fibonacci−Zeckendorf algorithm. This last step produces a bit−stream compressed file that can be saved in a less space than the original one. The second phase of the proposed system is the decoding phase in which the algorithm reconstruct the image from the bit−stream one. This can achieved as opposite process to the decoding phase. The flowchart in Fig. 1 concludes the procedures of the encoding and decoding phases of the proposed algorithm.

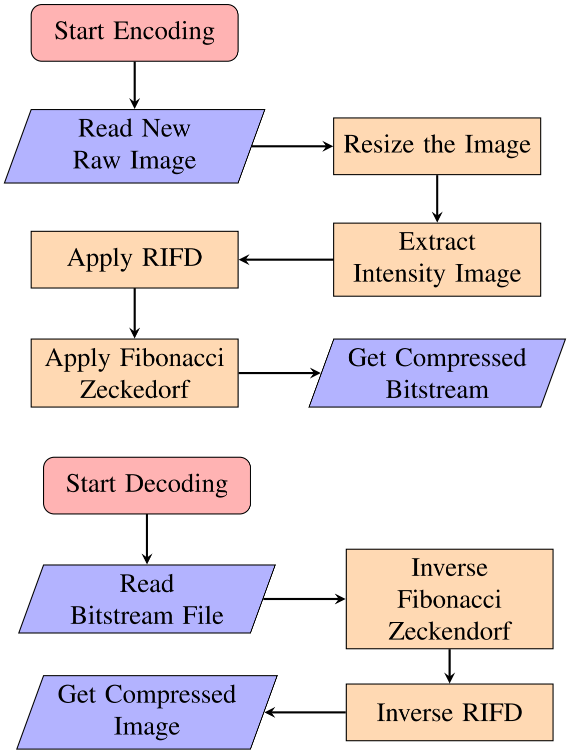


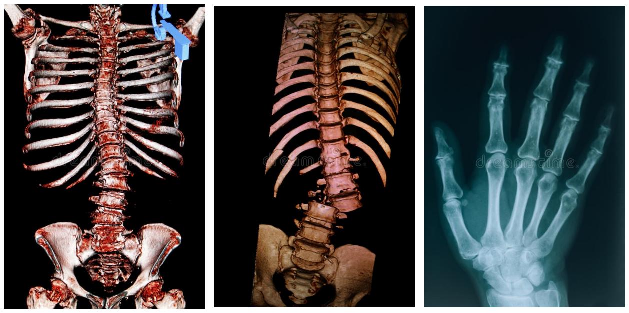
Fig. 1: Proposed Encoding/Decoding Algorithm Flowchart.

VI. EXPERIMENTAL RESULTS

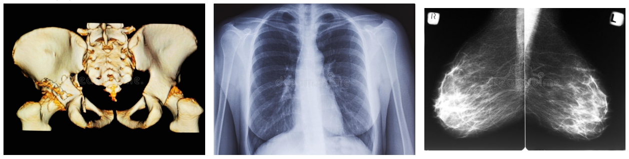
The proposed encoding/decoding algorithm is programmed and implemented in *MATLAB R2019a* on a PC with Intel (R) Core(TM) i7$-$8550U CPU $@1.8GHz$, $8GB$ RAM, 64$-$bit Operating System, X64$-$based processor, and windows 10 Pro.

A. Database Description

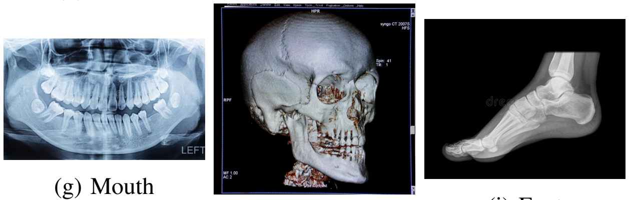
Nine medical images, shown in Fig. 2, are used to evaluate the proposed algorithm. Their specifications including their dimensions and the required storage memory capacity for each are given in table IV with a total required memory capacity of 934.4 kB. The pixel$-$intensity distribution of each raw$-$image is tested through the histogram given in Fig. 3. The images are selected to be different in the histogram distribution. It is clear that images as chest and mouth have uniform distribution while the images mammo and hand are almost with biased distribution. The rest of the images have poor distribution.



(a) Skelton (b) Spine (c) Hand



(d) Pelvis (e) Chest (f) Mammo

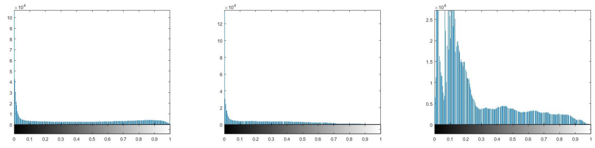


(g) Mouth (h) Skull (i) Foot

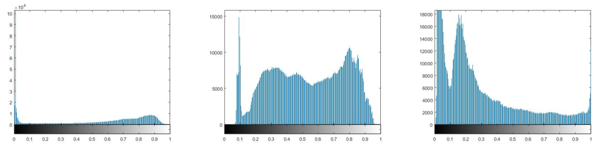
Fig. 2: The Nine Tested Medical Images.

TABLE IV: Database Specifications.

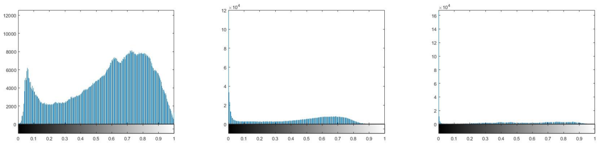
Raw Colored Image	**Size Rows × Columns**	**Memory Capacity in kB**
Chest	599 × 2400	89.1
Mouth	482 × 2400	93
Skelton	900 × 1785	207
Skull	759 × 2400	161
Foot	636 × 2400	43.8
Hands	900 × 1800	58.8
Spine	900 × 1800	123
Pelvis	600 × 2400	81.6
Mammo	563 × 2400	77.1



(a) Skelton (b) Spine (c) Hand



(d) Pelvis (e) Chest (f) Mammo



(g) Mouth (h) Skull (i) Foot

Fig. 3: Histogram of the Original Tested Images.

B. Performance Evaluations

At first, the proposed algorithm extracts two versions of each colored raw$-$image with sizes of 128×128 and 256×256, followed by extracting the intensity image of each version. Then, the algorithm applies the $RIFD$ followed by $Fibonacci\text{-}Zeckendorf$ on each version. These steps generate eighteen bit$-$stream text$-$ files which are the compressed versions of the images. The storage capacity, the percentage memory save, and the the number of bits of each compressed file are determined and shown in table V. A comparison of the storage capacity of the original and the corresponding two compressed versions is plotted in Fig. 4. The algorithm has been run 5 times and the average run time for each image is calculated and given in table V.

The second stage of the proposed algorithm is the decoding process which generates the reconstructed images shown in Fig. 5 and Fig. 6. In addition, the evaluation metrics: BPP, CR, MSE, $PSNR$, and $MSSIM$ are calculated and plotted in Fig. 7, 8, 9, 10 and 11.

VII. CONCLUSION

An image can be compressed because of the presence of redundancies. In this paper, a hybrid approach is proposed to compress the medical images. The intensity image of each 128×128 and 256×256 is extracted from the original color image. The approach starts by $RIFD$ algorithm to reduce the pixels intensities and to be in the rang $(0 - 25)$ instead of the range $(0 - 255)$. The next step of the approach is to apply the Fibonacci$-$Zeckondorf compression algorithm on each image. The last step is evaluating the approach by metric parameters as BPP, MSE, PSNR and MSSIM.

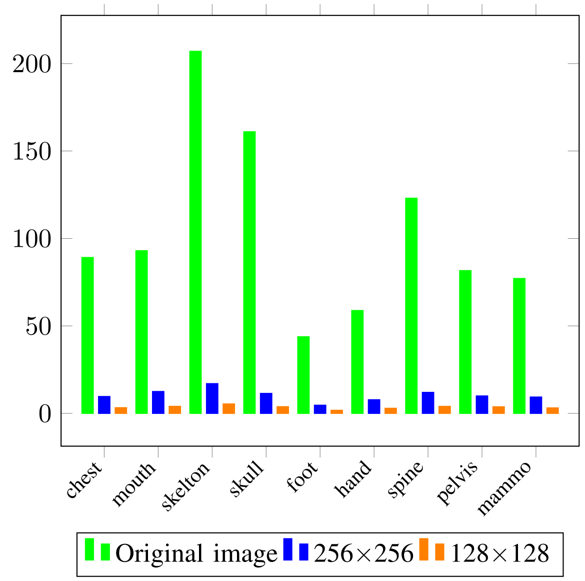
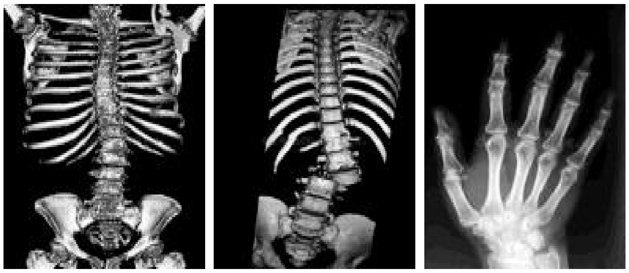
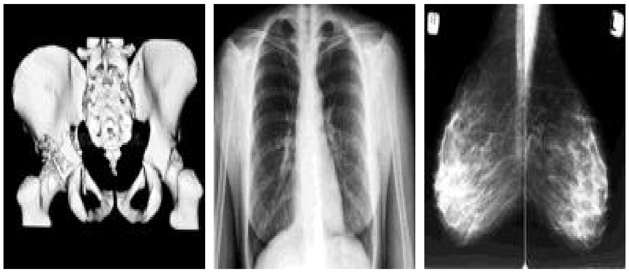


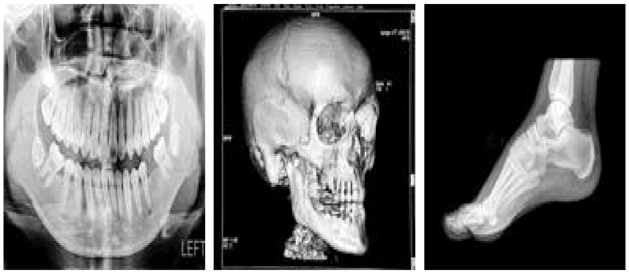
Fig. 4: Comparison of the Memory capacity in KB.



(a) Skelton (b) Spine (c) Hand



(d) Pelvis (e) Chest (f) Mammo

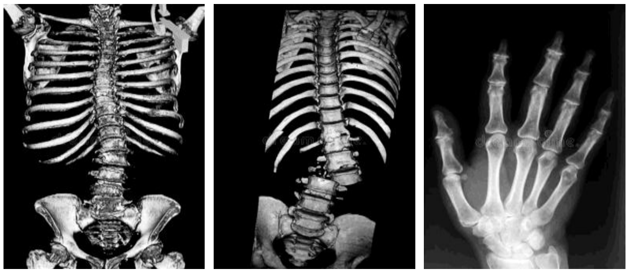


(g) Mouth (h) Skull (i) Foot

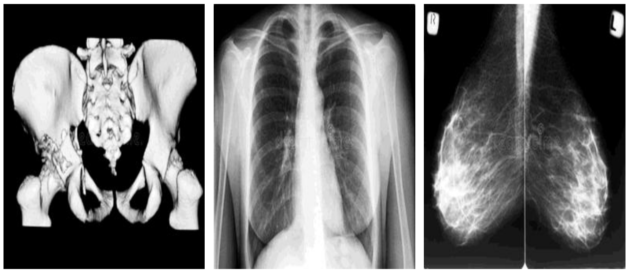
Fig. 5: Decoded Medical Nine Images: 128×128 Each.

TABLE V: The Encoder Outputs.

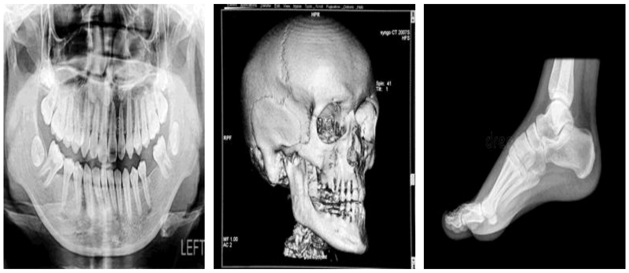
Image Version	**Memory in kB**	**Memory Save%**	**Ave. Run time, sec**	**Compressed Length, bits**
Chest128	3.04	96.6%	5.9	120,049
Chest256	8.64	90.3%	46.7	481,620
Mouth128	3.84	95.9%	6.1	123,801
Mouth256	11.7	87.4%	50.9	491,969
Skelton128	5.21	97.5%	5.4	87,650
Skelton256	16.5	92.0%	38.1	343,791
Skull128	3.68	97.7%	5.4	92,379
Skull256	10.8	93.3%	39.1	363,537
Foot128	1.73	96.1%	3.4	71,350
Foot256	4.47	89.8%	24.4	284,706
Hands128	2.75	95.3%	4.8	87,474
Hands256	7.12	87.9%	33.3	348,710
Spine128	3.87	96.9%	5.0	78,491
Spine256	11.6	90.6%	28.4	307,411
Pelvis128	3.63	95.6%	5.3	92,748
Pelvis256	9.5	88.4%	34.0	367,662
Mammo128	2.96	96.2%	4.9	93,525
Mammo256	8.64	88.8%	36.4	372,340



(a) Skelton (b) Spine (c) Hand



(d) Pelvis (e) Chest (f) Mammo



(g) Mouth (h) Skull (i) Foot

Fig. 6: Decoded Medical Nine Images: 256×256 Each.

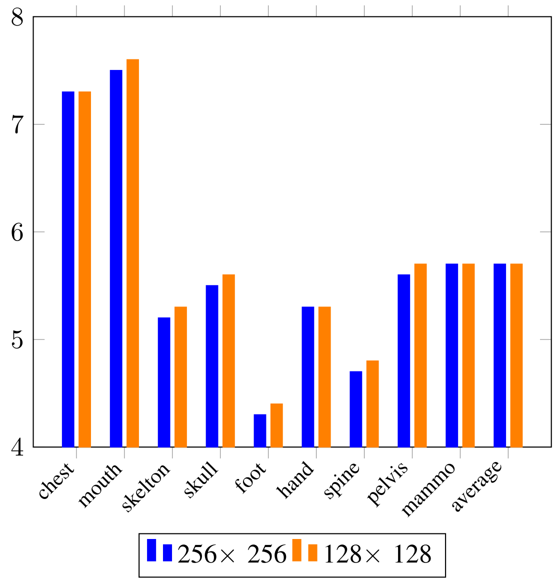


Fig. 7: Bits Per Pixel, BPP.

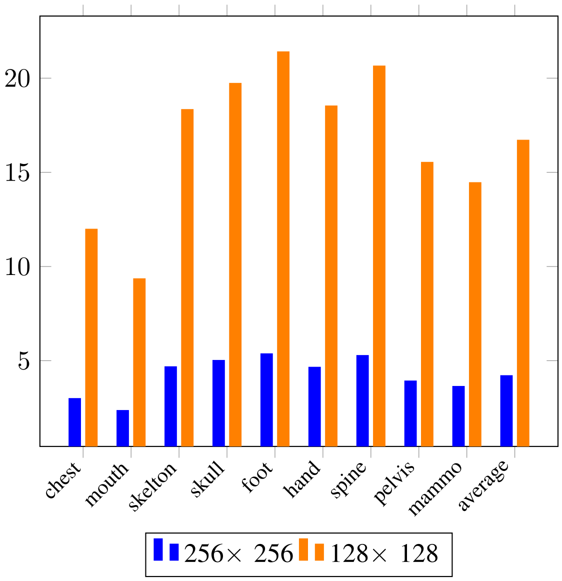


Fig. 8: Compression Ratio, CR.

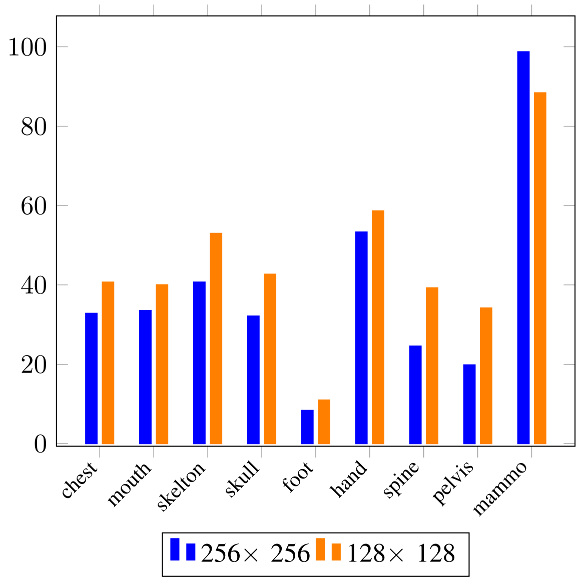


Fig. 9: Mean Square Error, MSE.

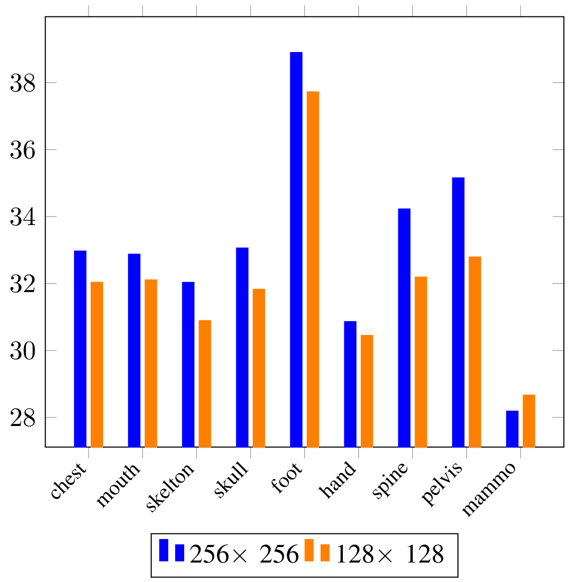


Fig. 10: Peak Signal to Noise Ratio, $PSNR$.

The experimental results proved the strength of the approach from the simplicity in implementation, robustness against errors, less needed memory space, but the execution time. The average run time for compressing images with size of 128×128 is about $5sec$, which is much less than the run time for compressing images with size 256×256, ($\simeq 35sec$). The number of bits per pixel ranges from 4 to 7, with high values in images with uniform intensity distribution. The compression ratio is in the range: $(3 - 22)$. The smallest CR is for the chest256×256 while the highest CR is for the foot128×128 images. The original nine images need 934.4 kB storage memory, while the nine 256×256 versions need 94.28 KB and the nine 128×128 versions need only 32.03 kB. The $PSNR$ range is $(28 - 39)\,dB$ which is in acceptable range in image processing field, indicating less loss of information. The $MSSIM$ range is $(0.4192 - 0.9954)$. The lowest values are for the mammogram (0.4192) and hand (0.74) images while the other images have values closer to $unity$ indicating almost no loss in the internal structure of the reconstructed images. The proposed hybrid approach gives a good performance for images with almost uniform probability distribution and a solid performance for the images with stacked, and nonuniform probability distribution. All the reconstructed images have been compressed with high visual quality as proved by $PSNR$ and almost no loss in the internal structure except for two images out of eighteen (11%), as proved by $MSSIM$. Future work employs a more powerful technique to improve

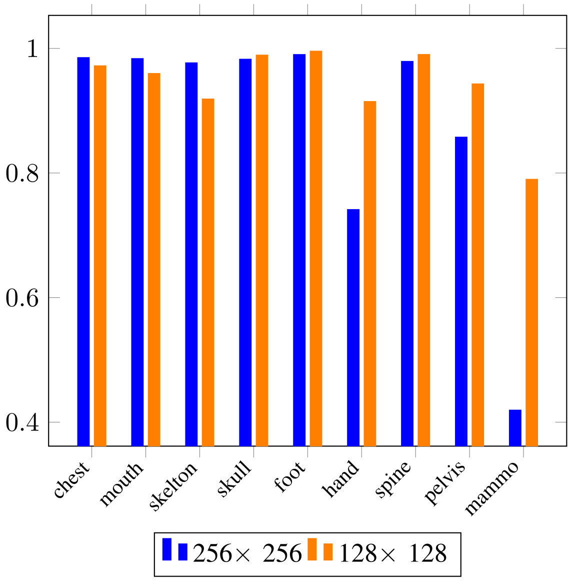


Fig. 11: Mean Structure Similarity, $MSSIM$.

the compression rate using higher degree of Fibonacci coding identifying the conditions at which the representation is unique. A fast implementation should be considered to achieve lower bit rate in less executing time.

REFERENCES

[1] P. Kumar and A. Parmar, “Versatile approaches for medical image compression: A review,” *Procedia Computer Science*, vol. 167, pp. 1380–1389, 2020.
[2] L. Sigler, *Fibonacci’s Liber Abaci: a translation into modern English of Leonardo Pisano’s book of calculation*. Springer Science & Business Media, 2003.
[3] M. Otair and F. Shehadeh, “Lossy image compression by rounding the intensity followed by dividing (rifd),” *Research Journal of Applied Sciences, Engineering and Technology*, vol. 12, no. 6, pp. 680–685, 2016.
[4] C. E. Shannon, “A mathematical theory of communication,” *Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.
[5] R. M. Fano, *The transmission of information*. Massachusetts Institute of Technology, Research Laboratory of Electronics, 1949.
[6] D. A. Huffman, “A method for the construction of minimum-redundancy codes,” *Proceedings of the IRE*, vol. 40, no. 9, pp. 1098–1101, 1952.
[7] J. Ziv and A. Lempel, “Compression of individual sequences via variable rate coding,” *IEEE transactions on Information Theory*, vol. 24, no. 5, pp. 530–536, 1978.
[8] T. A. Welch, “A technique for high-performance data compression,” *Computer*, no. 6, pp. 8–19, 1984.
[9] K. Shruthi, B. Shashank, Y. S. Saketh, H. Prasantha, and S. Sandya, “Comparison analysis of a biomedical image for compression using various transform coding techniques,” in *2016 IEEE 6th International Conference on Advanced Computing (IACC)*. IEEE, 2016, pp. 297–303.
[10] R. P. Jasmi, B. Perumal, and M. P. Rajasekaran, “Comparison of image compression techniques using huffman coding, dwt and fractal algorithm,” in *2015 International Conference on Computer Communication and Informatics (ICCCI)*. IEEE, 2015, pp. 1–5.
[11] P. V. Joshi and C. Rawat, “Region based hybrid compression for medical images,” in *2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES)*. IEEE, 2016, pp. 1212–1217.
[12] S. Singh and P. Pandey, “Enhanced lzw technique for medical image compression,” in *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*. IEEE, 2016, pp. 1080–1084.
[13] S. Liu, W. Bai, N. Zeng, and S. Wang, “A fast fractal based compression for mri images,” *IEEE Access*, vol. 7, pp. 62 412–62 420, 2019.
[14] P. M. Latha and A. A. Fathima, “Collective compression of images using averaging and transform coding,” *Measurement*, vol. 135, pp. 795–805, 2019.
[15] V. T. Druschke, “Implementing write compression in flash memory using zeckendorf two-round rewriting codes,” 2018.
[16] H. Zadjali and S. North, “Xml labels compression using prefix-encodings,” in *Proceedings of the 12th International Conference on Web Information Systems and Technologies*. SCITEPRESS, Science and Technology Publications, 2016, pp. 69–75.
[17] E. Khanjari and L. Gaeini, “A new effective method for labeling dynamic xml data,” *Journal of Big Data*, vol. 5, no. 1, pp. 1–17, 2018.
[18] J. Spiegel, P. Wira, and G. Hermann, “A comparative experimental study of lossless compression algorithms for enhancing energy efficiency in smart meters,” in *2018 IEEE 16th International Conference on Industrial Informatics (INDIN)*. IEEE, 2018, pp. 447–452.
[19] B. T. Ávila and R. M. C. de Souza, “Meta-fibonacci codes: Efficient universal coding of natural numbers,” *IEEE Transactions on Information Theory*, vol. 63, no. 4, pp. 2357–2375, 2017.
[20] P. Demontigny, T. Do, A. Kulkarni, S. J. Miller, D. Moon, and U. Varma, “Generalizing zeckendorf’s theorem to f-decompositions,” *Journal of Number Theory*, vol. 141, pp. 136–158, 2014.
[21] L. Lindroos, “Integer compositions, gray code, and the fibonacci sequence,” 2012.
[22] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error measurement to structural similarity,” *IEEE transactions on image processing*, vol. 13, no. 1, 2004.