



Storage Space Reduction of Biometric Iris Databases by Successive Images Differences and Quadtree Decomposition

Asaad Noori Hashim¹, Marwa Fadhel Jassim², Ashwaq T. Hashim^{2*}

¹ Faculty of Computer Science and Mathematics, University of Kufa, Najaf 54001, Iraq

² Control and Systems Engineering Department, University of Technology-Iraq, Baghdad 10011, Iraq

Corresponding Author Email: ashwaq.t.hashim@uotechnology.edu.iq

<https://doi.org/10.18280/mmep.100518>

ABSTRACT

Received: 16 January 2023

Revised: 15 April 2023

Accepted: 2 May 2023

Available online: 27 October 2023

Keywords:

biometric, iris database, iris compression, quadtree, Huffman coding

In recent years, the rise of biometric applications, particularly those centered around iris-based systems, has been significant. High data volumes inherent in these applications and the potential vulnerability of network links necessitate data compression in certain instances. The advantage of lossless compression methods is twofold: they maintain recognition performance without degradation while necessitating fewer computations for differentiation compared to their lossy counterparts. This study proposes a novel approach for lossless/lossy compression of iris biometric sample data across various public iris databases. Initially, the differences between successive images within each class are calculated, leveraging the strong correlation of images within each class. Subsequently, these differences are compressed using quadtree decomposition. This methodology was tested on six renowned iris databases: CASIA V1, CASIA V3, MMU1, MMU2, and UBIRIS Iris, all of which contain 8-bit grayscale images. The results indicate that the proposed strategy offers superior compression performance across different iris databases in comparison to existing methods. Notably, the results suggest that this method can be effectively integrated into an iris biometric recognition system, providing efficient iris image compression, especially when applied in its lossless form.

1. INTRODUCTION

Each individual's iris, the delicate, ring-shaped portion of the eye uniquely patterned and nestled between the sclera and pupil, functions much like an immutable, organic password. While fingerprints, facial features, and voice patterns have long been utilized as identifiers [1], the iris's pattern offers superior reliability, non-invasiveness, and a higher precision rate in recognition. Remarkably, the iris pattern remains largely unchanged throughout an individual's lifetime, with the left and right eyes each boasting distinct patterns [2]. The high degree of individuality inherent to each eye's iris [3] affords iris recognition advantages over other recognition systems [4].

Among biometric identifiers, iris recognition stands as one of the most precise for human identification, and its application is expanding globally. The rise of portable systems using iris recognition is particularly noticeable in law enforcement applications. Many of these applications necessitate a portable device capable of transmitting an iris image or template over a limited bandwidth communication channel. To ensure accurate recognition results, a full-resolution image (e.g., VGA) is typically required to provide sufficient pixel density across the iris [5].

While Iris Code templates are substantially smaller (600 times) than standard iris images, there is a practical preference for storing, transmitting, and integrating iris data using images rather than templates [6]. Image compression can effectively reduce the file size of an iris image, thereby decreasing the transfer time for large data volumes over narrow bandwidth

communication channels [4].

In 2007, Matschitsch et al. [7] explored the impact of several lossy compression algorithms on the accuracy of iris recognition systems. They found that the PSNR, accurately predicted by JPEG2000 and SPIHT compression algorithms, was aptly suited for iris recognition systems. Fractal compression was found to be least appropriate for the considered recognition system, while PRVQ compression performed remarkably well, delivering the best matching scores in one scenario despite ranking third in PSNR performance.

Later, in 2010, Ives et al. [8] studied the influence of JPEG-2000 image compression on the performance of recognition systems employing Daugman's algorithm. Their findings suggested that recognition performance was minimally impacted despite significant compression.

In 2015, Shaikh and Mukane [9] used a compression scheme combining region-of-interest isolation with JPEG compression at various qualities to compress iris images. Their investigation into the scheme's effect on recognition performance revealed minimal impact when using compressed iris images.

Similarly, in 2015, Ramadan [10] introduced an automated 3D iris compression and recognition system using spherical wavelet coefficients rooted in effective 3D iris representation. The results demonstrated that spherical wavelet coefficients yield impressive compression abilities with minimal features.

Čorić et al. in 2014 [11] applied Classified Vector Quantization (CVQ) and ordinary Vector Quantization (V.Q.)

to compress grayscale iris images sourced from a public iris image database. Their findings confirmed that, given the uniformity and low contrast levels of iris images, both compression methods are markedly more efficient when applied to them than to standard images from everyday environments.

Shaikh and Mukane in 2015 [12], utilized JPEG compression for iris images and studied its impact on recognition performance.

In 2017, Kapoor and Rawat [13] proposed a scheme that combines ROI (region-of-interest) isolation with JPEG 2000 compression at varying levels. They concluded that JPEG 2000 compression yields superior results with iris images normalized using the Biomechanical model, with minimal impact on recognition performance.

Al-Khafaji and Ali [14] introduced a method in 2019 for compressing biometric iris grayscale images, combining the DWT base's multiresolution scheme and the zipper transformation base techniques. Further, in 2022, Jalilian et al. [15] examined the feasibility of a deep learning-based compression model for iris data compression.

Lossy image compression can significantly reduce the space and bandwidth requirements for image storage and transmission, a feature highly sought after by developers of iris recognition systems. While deep learning techniques, with their various methods, are rapidly emerging as the preferred tool for general image compression tasks, their application for iris image compression demands meeting specific critical quality criteria. These include high perceptual quality, spatial accuracy of the images, and storage space efficiency.

In this paper, we examine and evaluate the effectiveness of a method grounded in successive image differencing, quadtree decomposition, and post-compression Huffman coding. When compared to its performance with familiar and broadly used lossy compression methods, this approach could offer efficient compression without substantial data loss. We conduct a comprehensive comparison and analysis of compression and recognition results to pinpoint the optimal compression algorithm for iris recognition systems.

The remainder of this paper is structured as follows: Section 2 will delve into the concept of quadtree decomposition. Section 3 will outline the steps of the proposed system. Section 4 will detail the experimental framework, testing, and analysis. Finally, Section 5 will conclude the paper.

2. QUADTREE

Using recursion, the quadtree approach divides an image into blocks or regions [16]. The blocks that have been partitioned are organized using a hierarchal tree structure. The subdivided blocks are known as the child blocks, while the root blocks are known as the parent blocks. Each of the four (quad) equal-sized subblocks that make up the parent block is exposed for testing. If a block passes the homogeneity condition, no further division takes place, the node is left undivided, and it is referred to as a leaf node. Divide the block into four subblocks or regions and reapply the test criteria if the criterion is not met. This process is repeated up until each sub-block is subject to the conditions, this process is repeated.

Consequently, each node/parent has no children or four children. The last block size is variable and relies on the application's requirements. The decomposing last until the quadtree reaches its minimum size [17]. decomposition. The

root node represents the entire image, and if it fails to meet the homogeneity criteria, it is divided into four equal-sized sub-blocks. The leaf node indicates that a block meets the homogeneity requirements [18], Figure 1 depicts a tree structure as well as a quadtree, and Figure 2 shows some examples from utilised dataset.

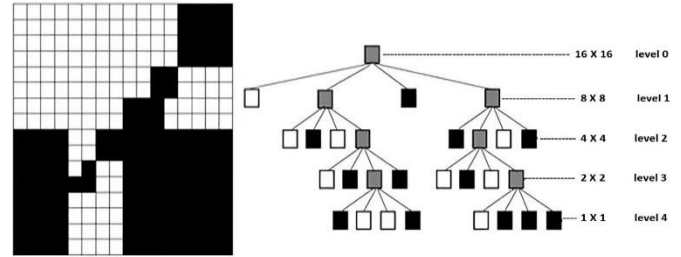


Figure 1. Quadtree: Sample image and Quadtree decomposition structure [18]

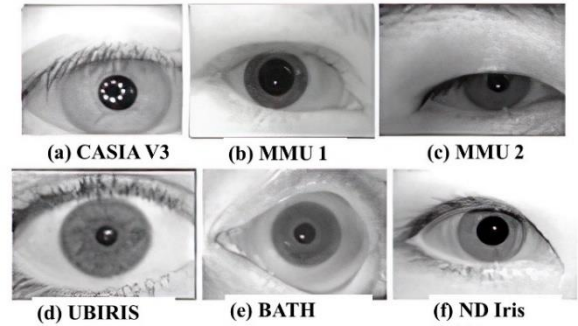


Figure 2. Samples of iris images from different databases

3. PROPOSED METHOD

At first, the different images are found for each class in the iris dataset by subtracting the successive images of a person from the first image in the same class. Then the quadtree decomposition is employed to compress these different images due to the close correlation of images belonging to each class. The Huffman encoding is used for extra compression to compress the quadtree data result.

3.1 Image differences

Image differencing is used in image processing to detect differences between images. The difference between two images is calculated by subtracting the pixels in each image and then producing an image based on the result.

The following are the differences between the two images:

$$ImgDiff(i, j, k) = firstimg(i, j, 1) - img_k(i, j, k) \quad (1)$$

for $i = 1, \dots, w$, for $j = 1, \dots, h$, $k=2, \dots, 5$

where, *firstimg* is the first image belonging to each class in the iris dataset, *w* and *h* are the dimensions of the iris image, and *k* is the number of iris images in each class.

Figure 3 shows the five images for 'chualsr' from the MMU1 database. Figure 4 shows the differences between the first and the remaining four images.



Figure 3. Class from MMU1 database [1]

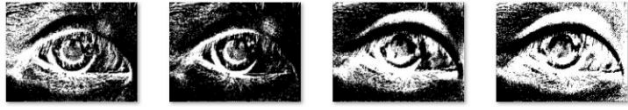


Figure 4. Successive differences: (a) The first and second images difference; (b) The first and the second image difference; (c) The first and the third image; (d) The first and fourth image difference

3.2 A negative/positive map

By this step, the complication resulting from negative values is removed. These negative values are continually modified to positive by applying the following mapping equation:

$$X_i = \begin{cases} 2X_i & \text{if } X_i \geq 0 \\ -2X_i - 1 & \text{if } X_i < 0 \end{cases} \quad (2)$$

where, X_i is a negative number, by the above equation, the negative values are changed to be odd while all positive values become even in the difference image *ImgDiff*.

To return the numbers (X_i) to their original signs, perform the following inverse equation:

$$X_i = \begin{cases} X_i \text{ div } 2 & \text{if } X_i \text{ is even} \\ -(X_i + 1) \text{ div } 2 & \text{if } X_i \text{ is odd} \end{cases} \quad (3)$$

where, X_i is the i^{th} element, the above equation converts all odd values to negative while converting even values to positive.

3.3 Quadtree decomposition

The quadtree decomposition is done with the Mean of each block in the image as a measure. The Mean of a block such as the following:

$$\mu(X) = \frac{1}{n} \sum_{i=1}^n X_i \quad (4)$$

where, X_i is the i^{th} element and n is the number of elements.

At first, the $\mu(x)$ of the parent block is computed first, and the max value in each block is found. Then the breakup children's blocks are specified separately. Now find the maximum value of the separate block; if it is more than its Mean block μ , further resolve the child block. Or else, if there is no further division, the block/node is left as a leaf—the division results in decomposing the image with unequal size partitioned blocks. Figure 5 shows the homogeneous and inhomogeneous quadrants of a decomposed quadtree image, and the minimum block size is 4×4 and $th=0$ (i.e., lossless compression). Because non-homogeneous quadrants have more than one luminance value, individually saving each pixel (total of 16 pixels in a minimum size quadrant) is not valuable. As we note, there are a lot of regions with equal or close intensity values, and with quadtree decomposition, we can

exhaust to compress. The result after Huffman coding for extra compression is shown in Figure 6.

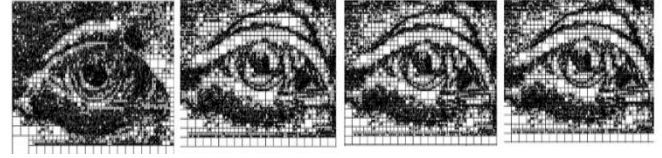


Figure 5. Quadtree decomposition and the minimum block size is 4×4 for Figure 3 (a-d)

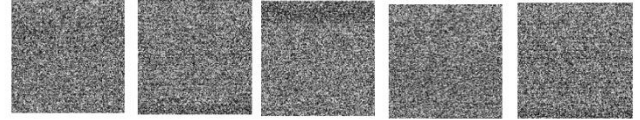


Figure 6. Huffman coding for Figure 5

Figure 7 depicts the proposed compression flowchart based on quadtree decomposition and Huffman coding.

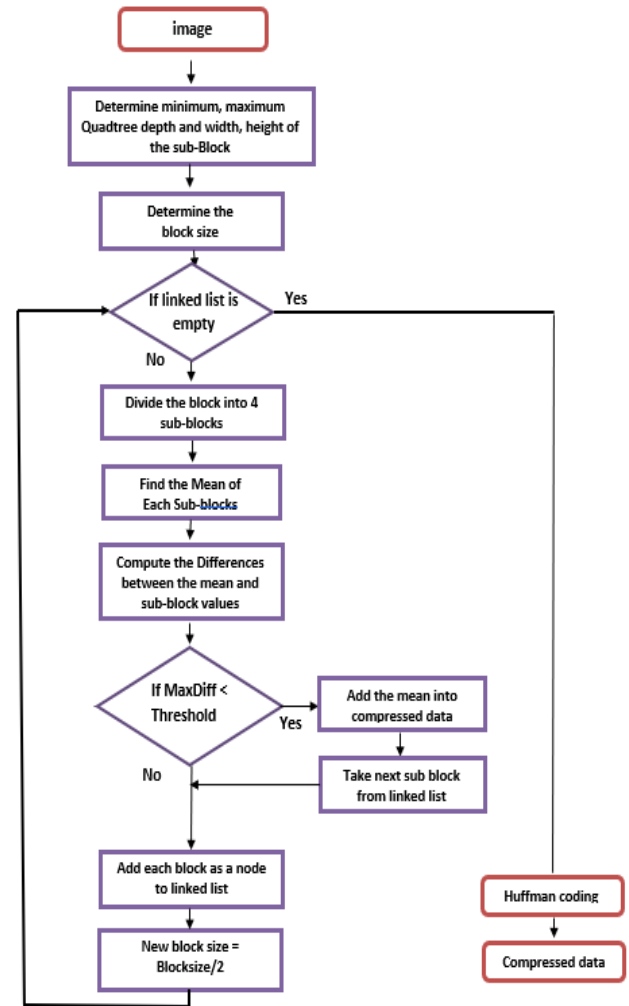


Figure 7. The flowchart of the proposed compression method

In the suggested system, a regular quadrant is a segment in which all values are different by a minimal amount (max difference $\text{MaxDiff} \leq 1, 2$) if the quadrant is lossy or the same if it is lossless ($\text{MaxDiff}=0$). The output from the quadtree is a list of block sizes, (x, y) coordinates, and the means of quadrants. After that, the list is compressed using Huffman

coding. The details of all steps of the proposed method are explained in the algorithm 1.

Algorithm 1: Encoding

Input

ImgDiff, // The difference between two images
 w, h , // width and height of *ImgDiff*

Output

COMP // Compressed data

Step 1: If the size of the sub-image *ImgDiff* is not square, then the borders of *ImgDiff* are padded with 0's. Assume that *J* is the padded image with size $w' \times h'$.

Step 2: Determine the minimum block size *BkSize* quadtree decomposition's threshold. Then if *BkSize*=4, the minimum block size became 4×4 .

Step 3: Specify a threshold for divided blocks *th*.

Step 4: Compute the max value of the block elements *MaxDiff*.

If *MaxDiff* > *th* then

ImgDiff image is split into four blocks.

Else

Store in *H.Q.* the following:

- (i) The quadrant size is because the dimensions of a quadrant (i.e., height and width) are all the time equal, so only one value is desired to store.
- (ii) (x, y) coordinates of the quadrant's upper left corner.
- (iii) Mean of the quadrant.

The threshold *th* represents the bare minimum (0 for lossless encoding). Because the quadrants have many equivalent sizes, a list of sizes is created to ensure adequate storage capacity for each list item, regarding (ii) and (iii) for all quadrants whose size is equal to the corresponding component in the list. It is not necessary to store the data of any quadrant in *J* that is entirely outside of the original image *I*. It is simple to determine by comparing the coordinates of the quadrant's upper left corner with the values of *w* and *h*.

Step 5: Huffman encodes *H.Q.* to obtain *COMP*.

The decoding by Huffman is applied to *COMP* for retrieving the homogeneous quadrants *H.Q.*, and then the inverse of mapping is performed by the following equation:

3.4 Decoding phase

The inverse of the encoding phase is used in the decoding stage. To retrieve the difference images, the inverse of quadtree decomposition is used. Algorithm 2 illustrates the steps of the encoding phase.

Algorithm 2: Decoding

Input

H.Q., w', h' .

Output:

ImgDiff'.

Step 1: Inverse the quadtree decomposition on the *H.Q.* by performing the sub-steps such as:

Step 1.1: Form an image of zeros called *ImgDiff*' of size $w' \times h'$. The size of *ImgDiff*' is equivalent to the padded image *J*.

Step 1.2: Fill the homogeneous quadrants of *ImgDiff*' using the Means of each

homogeneous quadrant block saved in step 4 of the encoding process in *H.Q.*, besides its size and coordinates.

Step 2: Resize the *ImgDiff*' to size $w \times h$, i.e., equal to the size of the original sub-image of *ImgDiff* and then generate the decoded image.

3.5 Image addition

All the successive differences images are added with the first image *firtimg* for each class to reveal the original images. The following is the image addition:

$$img_k = firtimg(i, j) + ImgDiff^{(i, j)} \quad (5)$$

for $i = 1, \dots, w$, for $j = 1, \dots, h$, $k = 2, \dots, 5$

4. EXPERIMENTAL RESULTS

In all experiments, 8-bit grayscale images are used in BMP format. The RGB images were converted to the YUV format. Channel Y is used as a grayscale image.

4.1 Dataset

Images of irises utilized in this research are accessible from the National Institute of Standards and Technology (NIST). Iris image databases are used:

- The **CASIA V1** dataset [19] contained 756 8-bit grayscale images with a 320×280 pixels resolution. All were in .bmp, and 756 polar iris images were extracted.
- The **CASIA V3** Interval dataset contained 2639 8-bit grayscale images with a 320×280 pixels resolution. All were in .jpeg format, and 2638 polar iris images were extracted.
- The **MMU** dataset [20] contained 457 24-bit grayscale images with a 320×240 pixels resolution. All were in .bmp format, and 439 polar iris images were extracted.
- The **MMU2** dataset contained 996 24-bit grayscale images with 320×238 pixels resolution. All were in .bmp format, and 981 polar iris images were extracted.
- The **UBIRIS** dataset [21] contained 1876 24-bit grayscale images with 200×150 pixels resolution. All were in jpeg format, and 614 polar iris images were extracted.

Table 1 illustrates the description of the iris datasets which are tested.

Table 1. Details of the iris dataset

Database Name	Number of Images	Resolution
CASIA V1	756	320×280
CASIA V3 Interval	2639	320×280
MMU1	457	320×240
MMU2	996	320×238
UBIRIS	1876	200×150

Table 2 shows the simulation results of the proposed method for sample images for 'chualsr' from the MMU1 dataset regarding Compression Ratio (C.R.) and PSNR. The proposed method is applied with the Lossless version with $th=0$ and with $th=10$ and $th=30$ for the lossy version.

Table 2. The C.R. and PSNR were achieved for iris images from the MMU1 dataset

th	Image	PSNR	CR
0	chualsr1	Inf	1.0694
	Difference between fionar1 and	Inf	2.9712
	Difference between fionar1 and	Inf	3.3700
	Difference between fionar1 and	Inf	3.1207
	Difference between fionar1 and	Inf	2.9546
20	chualsr1	36.016	2.0099
	Difference between fionar1 and	35.738	4.4117
	Difference between fionar1 and	36.502	4.3251
	Difference between fionar1 and	37.178	4.7813
	Difference between fionar1 and	37.848	3.9854
30	chualsr1	33.301	4.919
	Difference between fionar1 and	30.388	5.3473
	Difference between fionar1 and	30.356	5.6932
	Difference between fionar1 and	31.295	5.8243
	Difference between fionar1 and	32.769	4.8912

Table 3 shows the C.R. of the lossless version of the proposed system applied to all iris images in the selected iris dataset, while Table 4 shows the C.R. and PSNR for the lossy version of the proposed approach.

Table 3. Compression ratio of the proposed system for the lossless version of the iris dataset

Database Name	CR
CASIA V1	2.9716
CASIA V3 Interval	3.1265
MMU1	3.0136
MMU2	3.9872
UBIRIS	2.8391

Table 4. Compression ratio and PSNR of the proposed system for the lossy version of the iris datasets

Database Name	C.R. th=20	PSNR	CR th=30	PSNR
CASIA V1	3.1265	40.05	5.7321	33.54
CASIA V3 Interval	2.9716	41.87	4.8569	34.65
MMU1	3.0136	42.42	5.3283	33.82
MMU2	3.9872	42.40	5.3217	33.61
UBIRIS	2.8391	40.56	4.8192	33.12

4.2 Setting and compression method

To practically assessed the proposed lossy version method, it must be first compared its performance with well-known and popular lossless compression methods, including:

- JPEG-LS IrfanView used JPEG-LS relied on Median edge detection, subsequent predictive, and Golumb encoding (in two modes: on and normal) [22].
- PNG is also used by the XN-View application, with an LZSS encoding variant and a compression strength of 6.
- JPEG2000 Image magick used JPEG2000 Part 1, a lossy-to-lossless transform coder based on wavelets. [23].

Tables 5 and 3 present the results of compression quality in terms of C.R. measure belonging to each database (averaged of the entire images) applying the various lossless compression algorithms. As noticed, the proposed method displays superior performance across all other algorithms.

Table 5. Lossless compression algorithms for each database with corresponding achieved C.R.

Method	Database	CR
JPEG-LS	CASIA V1	1.75
	CASIA V3 Interval	2.25
	MMU1	1.85
	MMU2	3.40
	UBIRIS	1.50
JPEG2000	CASIA V1	1.72
	CASIA V3 Interval	2.23
	MMU1	1.82
	MMU2	2.24
	UBIRIS	1.50
PNG	CASIA V1	1.75
	CASIA V3 Interval	1.85
	MMU1	1.50
	MMU2	1.65
	UBIRIS	1.20

The performance of the proposed lossy version is further compared with some of the most common and state-of-the-art lossy compression methods, including:

- SPIHT lossy-to-lossless Zerotree-based wavelet transform codec5 [24].
- JPEG XR FuturixImager6 applied this most current ISO still image coding standard based on the Microsoft H.D. format [25]. Table 6 shows the C.R. for compression methods for each used dataset.

Table 6. Lossy compression algorithms for each database with corresponding achieved C.R.

Method	Database	CR
SPIHT	CASIA V1	0.02
	CASIA V3 Interval	2.25
	MMU1	1.85
	MMU2	3.40
	UBIRIS	1.50
JPEG XR	CASIA V1	1.72
	CASIA V3 Interval	2.23
	MMU1	1.82
	MMU2	2.24
	UBIRIS	1.50
JPEG 2000	CASIA V1	0.27
	CASIA V3 Interval	1.85
	MMU1	1.50
	MMU2	1.65
	UBIRIS	1.20

Tables 6 and 4 show the correlated outcomes of each database (averaged over all images) regarding C.R. after performing lossy compression methods. The outstanding fulfilment of the proposed system, more than other algorithms, is also shown in these results.

5. CONCLUSIONS

The technique proposed depends on successive image differences, quadtree decomposition, and Huffman coding post-compression. We examined and evaluated the practicability of the proposed method. Comparing the performance of the proposed approach to that of well-known and popular lossy compression methods of large databases, such as Iris-based identification databases presently used for personal identification, the proposed method demonstrated effective reduction without significant data loss. The quadtree

decomposition effectively divides an image into quadrants that are evaluated based on the correlation between their data. We use the difference to determine the abrupt data change. If MaxDiff is high, the block is uncorrelated and must be decomposed; otherwise, it is correlated, and the mean is sufficient for storage. The quadtree output is further compressed using Huffman encoding. On an open iris image database, various lossless and lossy image algorithms are tested to compare the efficiency for iris images that can be anticipated from the most popular and recent lossless image coding algorithms for still images. The proposed system's results depend on the database, which is generally regarded as the most effective technique.

REFERENCES

- [1] Hafeez, H., Zafar, M.N., Abbas, C.A., Elahi, H., Ali, M.O. (2022). Real-time human authentication system based on iris recognition. *Eng.* 3(4): 693-708. <https://doi.org/10.3390/eng3040047>
- [2] Daugman, J., Downing, C. (2001). Epigenetic randomness, complexity and singularity of human iris patterns. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 268(1477): 1737-1740. <https://doi.org/10.1098/rspb.2001.1696>
- [3] Jeong, D.S., Hwang, J.W., Kang, B.J., Park, K.R., Won, C.S., Park, D.K., Kim, J. (2010). A new iris segmentation method for non-ideal iris images. *Image and Vision Computing*, 28(2): 254-260. <https://doi.org/10.1016/j.imavis.2009.04.001>
- [4] Szwedczyk, R., Grabowski, K., Napieralska, M., Sankowski, W., Zubert, M., Napieralski, A. (2012). A reliable iris recognition algorithm based on reverse biorthogonal wavelet transform. *Pattern Recognition Letters*, 33(8): 1019-1026. <https://doi.org/10.1016/j.patrec.2011.08.018>
- [5] Sangeetha, M., Betty, P., Kumar, G.N. (2017). A biometric iris image compression using LZW and hybrid LZW coding algorithm. In 2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), Coimbatore, India, pp. 1-6. <https://doi.org/10.1109/ICIIECS.2017.8275906>
- [6] Daugman, J., Downing, C. (2008). Effect of severe image compression on iris recognition performance. *IEEE Transactions on Information Forensics and Security*, 3(1): 52-61. <https://doi.org/10.1109/TIFS.2007.916009>
- [7] Matschitsch, S., Tschinder, M., Uhl, A. (2007). Comparison of Compression Algorithms' Impact on Iris Recognition Accuracy. In: Lee, S.W., Li, S.Z. (eds) *Advances in Biometrics. ICB 2007. Lecture Notes in Computer Science*, vol. 4642. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-74549-5_25
- [8] Ives, R.W., Bishop, D.A., Du, Y., Belcher, C. (2010). Iris recognition: The consequences of image compression. *EURASIP Journal on Advances in Signal Processing*, 2010: 1-9. <https://doi.org/10.1155/2010/680845>
- [9] Shaikh, I.J., Mukane S.M. (2015). Iris image compression using JPEG & its effect on recognition performance. *International Journal of Scientific & Engineering Research*, 6(2).
- [10] Ramadan, R.M. (2015). Iris compression and recognition using spherical geometry image. *International Journal of Advanced Research in Artificial Intelligence (IJARAI)*, 4(6). <https://doi.org/10.14569/IJARAI.2015.040605>
- [11] Čorić, M., Lušić, Z., Gudelj, A. (2014). Classified vector quantization and its application on compression of iris images in the safety of marine systems. *Promet–Traffic & Transportation*, 28(2): 125-131. <https://doi.org/10.7307/ptt.v28i2.1707>
- [12] Paul, A., Khan, T.Z., Podder, P., Ahmed, R., Rahman, M.M., Khan, M.H. (2015). Iris image compression using wavelets transform coding. In *Proceedings of the 2nd International Conference on Signal Processing and Integrated Networks*, Noida, India, pp. 544-548. <https://doi.org/10.1109/SPIN.2015.7095407>
- [13] Kapoor, P., Rawat, P. (2017). Comparison of detection accuracy and effect of JPEG2000 compression on iris recognition. *International Journal of Computer Applications*, 162(4): 37-42. <https://doi.org/10.5120/ijca2017913281>
- [14] Al-Khafaji, G., Ali, N. (2019). Fourier transform coding-based techniques for lossless iris image compression. *Iraqi Journal of Science*, 60(11): 2506-2511. <https://doi.org/10.24996/ijcs.2019.60.11.23>
- [15] Jalilian, E., Hofbauer, H., Uhl, A. (2022). Iris image compression using deep convolutional neural networks. *Sensors*, 22(7): 2698. <https://doi.org/10.3390/s22072698>
- [16] Jagadeesh, P., Nagabhushan, P., Kumar, R.P. (2013). A novel image scrambling technique based on information entropy and quad tree decomposition. *International Journal of Computer Science Issues (IJCSI)*, 10(2 Part 1): 285-294.
- [17] Marquez, G.R.C., Escalante, H.J., Sucar, L.E. (2010). Simplified quadtree image segmentation for image annotation. *Proceedings of the 1st Automatic Image Annotation and Retrieval Workshop*, 1(1): 24-34.
- [18] Revanna, C.R., Keshavamurthy, C. (2020). A new partial image encryption method for document images using variance based quad tree decomposition. *International Journal of Electrical and Computer Engineering*, 10(1): 786-800. <http://doi.org/10.11591/ijece.v10i1.pp786-800>
- [19] BIT. CASIA Iris Image Database. <http://biometrics.idealtest.org/>, accessed on Dec. 20, 2016.
- [20] Multimedia-University. MMU Database. <http://pesona.mmu.edu.my/~ccteo>, accessed on Dec. 20, 2016.
- [21] Proença, H., Alexandre, L.A. (2005). UBIRIS: A Noisy Iris Image Database. In: Roli, F., Vitulano, S. (eds) *Image Analysis and Processing – ICIAP 2005. ICIAP 2005. Lecture Notes in Computer Science*, vol. 3617. Springer, Berlin, Heidelberg. https://doi.org/10.1007/11553595_119
- [22] Weinberger, M.J., Seroussi, G., Sapiro, G. (2000). The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS. *IEEE Transactions on Image Processing*, 9(8): 1309-1324. <https://doi.org/10.1109/83.855427>
- [23] Taubman, D.S., Marcellin, M.W., Rabbani, M. (2002). JPEG2000: Image compression fundamentals, standards and practice. *Journal of Electronic Imaging*, 11(2): 286-287. <https://doi.org/10.1117/1.1469618>
- [24] Krishnan, S., Sathe, P.M. (2013). Comparison of IRIS image compression using JPEG 2000 and SPIHT algorithm. *IOSR Journal of Electronics and*

Communication Engineering (IOSR-JECE), 4(4): 5-9.
[25] Horvath, K., Stögner, H., Uhl, A., Weinhandel, G. (2011).
Lossless compression of polar iris image data. In: Vitrià,
J., Sanches, J.M., Hernández, M. (eds) Pattern

Recognition and Image Analysis. IbPRIA 2011. Lecture
Notes in Computer Science, vol. 6669. Springer, Berlin,
Heidelberg. https://doi.org/10.1007/978-3-642-21257-4_41