Research Article

Image Data Compression Based on Two Hybrids Algorithms

Adil I. Khalil[1*], Ahmed K. Abas[2], Maad M. Khalil[3]

[1]Computer science department, College of education for pure science, University of Diyala, Diyala-Iraq
[2]Computer science department, College of education for pure science, University of Diyala, Diyala-Iraq
[3]Computer science department, College of education for pure science, University of Diyala, Diyala-Iraq
adil.khalil.uod@gmail.com[1], purecomp.adeel.khalil@uodiyala.edu.iq

Article History: Received: 10 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 20 April 2021

Abstract. Image compression is a way to reduce the storage space of images. Therefore, image compression process increases the performance of such as the data transmission process. This paper aims to present a new technique to compress digital image data. In this technique, two-hybrid algorithms were used to compress the image data. The first system consists of one-dimensional discrete cosine transform (DCT), differential pulse code modulation (DPCM), and Huffman code for difference signal. The second hybrid system utilizes an expert system called Learning Automata (LA) to code the difference signal obtained from the first system. A compression ratio of about (10.8:1) was obtained from the first system. The second system provides a (20.6:1) compression ratio with non-noticeable impairment. The information loss is caused by a hybrid (DCT/DPCM) system, not by the LA system. The conclusion drawn is that using two-hybrid systems to compress the image data provides a high compression ratio. Furthermore, learning automata is preferable since it removes all the redundancy in the row data. However, in learning automata, a Huffman code is determined pixel by pixel which takes a long time.

Keywords: Digital image compression, learning automata, discrete cosine transform, differential pulse code modulation.

1. Introduction

Data compression is a process to minimize the size of data (number of bits) needed to represent a specific volume of information. Compression is a special case from a more general technique known as encoding [1]. Encoding can be described as the process that takes an input string of symbols, assigns codes for each input symbol, and gives an output string of code [2, 3]. A compression algorithm and its corresponding decompression algorithm are collectively referred to as "coding". The applications of data compression are varied, including for example image transmission, video conferencing, photographic facsimile distribution, and geophysical images transmission obtained from satellites. Furthermore, data coding could be implemented to store data in a database such as archiving medical images, multispectral images, fingerprints, and drawing [4, 5, 6]. Images are highly data-intensive, include a great amount of redundancy. Therefore, image data compression facilities the transmission process and reduces the memory area needed for storage. Image data could be compressed by using some kind of transformation completed. To recover the image data pixels, a reversible linear step is used [7, 8].

Learning may be defined as those changes in the behavior that result from experience. Intelligent algorithms are identified by the ability to improve performance over time. In other words, the Intelligent algorithms aim toward an ultimate goal. The learning purpose can be thoroughly defined due to a lack of adequate knowledge [9].

The goal of the learning system (viewed in a purely mathematical sense) is to optimize a function that is not clearly understood and minimize the error. The learning automata chooses an operation at any time depending on a probability distribution across a specified and finite set of decisions [10]. The response obtained from the environment is often used to modify the probability of action using a training framework or learning system. Via this cycle of interacting with the environment, the automaton learns to select the appropriate solution with the highest probability. In addition to their utility informal language and complexity theory, it has been shown that finite automata could be applied to the digital image to compress the image data [11, 12, 13]. By exploiting self-similarity within image data, learning automata are capable of significantly reducing the amount of memory needed to encode the image data. The concept and theory behind learning automata have been covered by several contributions from books and papers (see Refs. [14, 15, 9]).

Image data compression was the objective of several studies. A predictive image compression scheme was implemented by Das and S. Chande [16]. Their method used a new suboptimal adaptive DPCM coder. This study was focused on the lossless compression scheme, which integrates three new concepts: a simple two-step predicted structure, a simple pixel- by -pixel adaptation scheme that guarantees the stability of the predictive coder, and a simple contextual entropy coder for the residuals. In this study, it has been shown that the proposed scheme produces a significantly high compression ratio than traditional DPCM techniques.

A lossless compression method for 4-D medical images was presented based on advanced video coding (H.264/AVC) [17]. This method significantly reduces information redundancies by recursively applying for multi-frame motion compensation in all four dimensions. Evaluations of results on actual 4-D medical images of different modes provided high compression efficiency compared to that of other traditional methods.

George Prince [4] explained the amount of text data has reached 40% of the total Internet traffic volume, it is imperative to compress textual information. To this end, many algorithms have been introduced and used including Huffman encoding, arithmetic encoding, the Ziv-Lempel family, Dynamic Markov Compression, and Burrow-Wheeler Transform. They used an algorithm to compress text images. The algorithm consists of two parts: a fixed–to–variable codebook, and row and column.

Chrysa, Christos [2], Addressed compression Wavelet picture with poor memory. Although it has been shown that wavelet or subband coding of images is superior to more traditional transform coding techniques, little attention has been paid to the important question of whether both wavelet transforms and subsequent coding can be applied in low memory without significant failure at the output. They present a complete system for coding images with low memory wavelets. They used a line-based in that the images are read line by line and only the minimum number of lines needed is retained in memory.

Lin, Yuzhanget et al. [18] suggested a task-based image compression metric that preserves the data that is most important to a given task. For each sub-band, the task-specific information (TSI) is calculated as the shared information in the wavelet domain between data (X) and class tag (C). To evaluate the quantization step-size for each sub-band, a heuristic TSI-based rate allocation algorithm is built to reduce the total file size while retaining as much TSI as possible. The proposed method generates a compressed code-stream compliant with JPEG 2000 that can be decoded by any decoder compliant with JPEG 2000.

In this paper, we present a new method of image data compression. In this method, a learning Automata system was used to code the difference signal obtained from another image data compression system called (DCT/DPCM). The (DCT/DPCM) consists of one-dimensional discrete cosine transform (DCT), differential pulse code modulation (DPCM), and Huffman code for difference signal. The remainder of the paper is organized as follows: material and method are presented in section 2. Section 3 includes the results and discussion of the proposed system. Finally, the conclusion is shown in section 4.

2. Materials and Method

2.1. The Hybrid (DCT-DPCM) System with Variable Length Code Design

To compress data, the first hybrid method is based on transformation using DCT combined with the DPCM and known as (DCT/DPCM) system. The difference signal resulting from the DPCM stage is quantized using scalar quantizer and encoded using the variable-length code of (4-bit) quantizer.

The following steps were used for coding image data using hybrid (DCT/DPCM):

Step1: Each row from the input image is divided into several blocks of equal size [19].
Step2: One-dimensional discrete cosine transform (DCT) is applied to each block.
Step3: Mask is used on each block to discard the high-frequency coefficient.
Step4: Apply The (DPCM) coding to (DCT) coefficients using (1-D) DPCM with the first-order predictor.
Step5: A quantizer is used to convert the data into a discrete form that called the quantized difference.
Step6: Coding the probabilities distribution of gray level using the entropy coding (Huffman code).

The flowchart of (DCT/DPCM) system is presented in Figure 1.

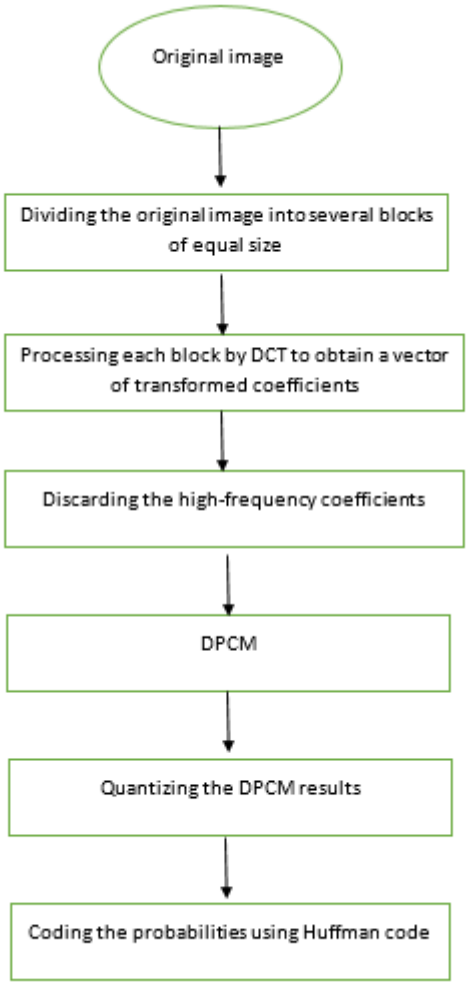

Figure 1: DCT/DPCM flowchart.

2.2. Learning automata theory

A learning automaton (LA) consists of two parts [20]: the environment (E) and the automaton (A) [21]. The term environment is related to the collection of all external circumstances and factors that influence the life and development of an organism. The environment is quantitatively identified by three variables {α, c,β} as shown in Fig. 2, where α ={ α 1, α 2,…., α r} is a finite input set, β ={ β1, β2 } is a binary output set, and c ={ c1,c2,…,cr } is a set of penalty probabilities. Each element ci represents one input action αi. The output β(n) of the environment belongs to β and can take one of two values β1 and β2 [17]. For mathematical convenience β1, and β2 are chosen to be (0 and 1) respectively. An output β(n)=1 is identified with a failure or an unfavorable response and β(n)=0 with a success or favorable response of the environment at time instant n (n=0,1,2….) , the element ci of c which characterizes the environment may then be defined by Eq. 1[14].

$$P_r(\beta(n)=1 \mid \alpha(\mathrm{n})=\alpha_i)=ci, \qquad (i=1,2,.....,r) \mid \qquad (1)$$

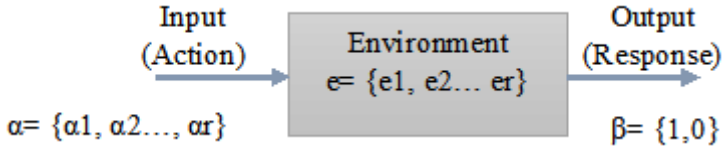

Figure 2: The environment scheme

ci represents the probability of failure where αi is the input set. The complexity of the allowed actions and the corresponding favorable and unfavorable responses determine the nature of the environment [22]. If the penalty probabilities (ci), which describe the environment, are constants, then the environment is called a stationary environment. In contrast, it is called non- stationary environment when the penalty probabilities (ci) corresponding

to the varicose actions vary with time. Typically ci(n) may vary from instant to instant by small increments or alternately ci(n) may be a constant over an interval (n, n+T-1) and switch to a new value at a time (n+T). Different types of non-stationary environments include those in which ci(n) are (i) periodic, (ii) slowly varying, and (iii) random variables that depend on the actions of the automaton [17]. The automaton is defined by the quintuple {Φ,β, α,p, F, H }as shown in Fig. 3. Where Φ={Φ1, Φ2,…., Φs}, is the internal states, β= {0,1} is the set of input response , α ={ α1 , α2,…., αr} with $r \leq s$ is the output or actions set and p=[p1,p2,…,pr] is the action probability vector where:

$$P_i(n) = P_r[\alpha(n) = \alpha_i] \quad (2)$$

$$\sum_{i=1}^{r} P_i(n) = 1, \quad for \ \ all \ n \quad (3)$$

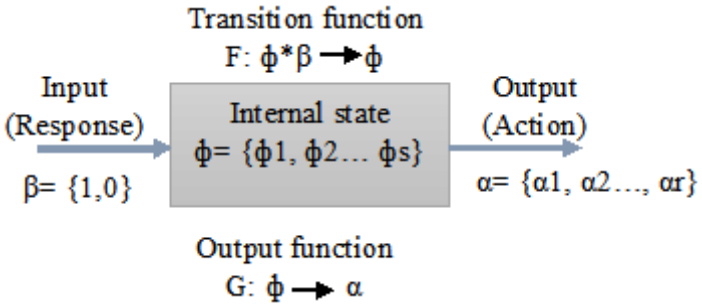

Figure 3: The Automata scheme

$F: \Phi \times \beta \rightarrow \Phi$ is a function that maps the current state and current input into the next stage, and it is called the transition function [23]. $H: \Phi * \beta \rightarrow \alpha$ is a function that maps the current state and input into the current output. If the current output depends only on the current state, the automaton is referred to as a state - output automaton. In such a case the function H in the definition above is replaced by output function $G: \Phi \rightarrow \alpha$ and it is called output function. The function F and G can be either deterministic or stochastic.

The objective of the automaton is to increase the probability of the automaton being in the state that corresponds to the smallest ci value and decrease all others' values [30]. The automaton takes in a sequence of input and puts out a sequence of actions.

2.2.1. Feed Back Connection of Automaton and Environment

The feedback structure that connects the environment and automaton is shown in Fig. 4, which forms the learning automata, where the output of the environment β(n) forms the input to the automaton and the action of the automaton α(n) provides the input to the environment. Starting from an initial state Φ (0), the automaton generates the corresponding action α(0). The response of the environment to this input is β (0), which in turn changes the state of automaton to Φ (1). This sequence of operations is then repeated to result in a sequence of states, actions, and responses of the environment [1, 0]. When the function (F and G) are assumed constant, i.e. independent on (n) and input sequence, the stochastic automaton is referred to as a fixed structure stochastic automaton. In the case when the transition probabilities are to be updated at each (n) based on the input at that instant, the automaton is called a variable structure stochastic automaton [22].

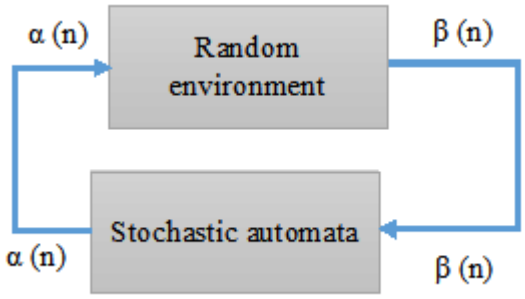

Figure 4: Feedback connection of automaton and environment

2.2.2. Reinforcement Scheme

The reinforcement schemes represent the important factor that affects the performance of the learning automata. A reinforcement scheme is defined as shown in Eq. 4

$$P(n+1) = T[P(n), \alpha(n), \beta(n)] \quad (4)$$

where T is the mapping function. The action probability at stage (n+1) is updated based on its previous value, the action α (n) at the instant (n), and the input β (n). The basic idea behind a reinforcement scheme is simple. If the automaton selects an action αi at an instant (n) and a favorable input (β (n) =0) result, the actin probability Pi (n) is increased and all the other components of P (n) are decreased. For an unfavorable input (β (n)=1), Pi(n) is decreased and all the other components are increased. These changes in Pi (n) are known as reward and penalty respectively. The action probabilities may be retained at their previous value and this case is known as inaction [22, 4]. In general reinforcement scheme performed by updating the action, is defined as shown below: $\alpha(n) = \alpha_i, \quad i = 1,2.....,r$

$$P_i(n+1) = P_i(n) - g_i(P(n)), \qquad When\ \beta(n) = 0 \quad (5a)$$

$$P_i(n+1) = P_i(n) - h_j(P(n)), \qquad When\ \beta(n) = 1 \quad (5b)$$

For all $j \neq i$

For preserving probability measure we have $\sum_{j=1} P_j(n) = 1$

$$P_i(n+1) = P_i(n) + \sum_{j=1 \& j \neq i}^{r} g_i(P(n)),\ \beta(n) = 0 \quad (6a)$$

$$P_i(n+1) = P_i(n) - \sum_{j=1 \& j \neq i}^{r} h_i(P(n)),\ \beta(n) = 1 \quad (6b)$$

2.2.3. Schemes of Learning Automata

The learning automata scheme is generally classified based on the nature of the function T in Eq. (4), as linear, nonlinear, and hybrid. The scheme is said to be linear if P(n+1) is a linear function of components of P(n), otherwise, it is nonlinear. Sometimes it is called hybrid when the action probabilities are updated according to a certain scheme depending on the intervals in which the values of the action probabilities lie [24].

Linear schemes are classified as a linear reward-reward scheme (LR-R), linear reward-penalty scheme (LR-P), linear reward-ε-penalty scheme (LR- εP), and the linear reward – inaction scheme (LR-I). These schemes exhibit interesting and significantly different characteristics and at present are prototypes for distinct types of behavior observed in all learning automata [17].

2.2.4. Convergence Time

An important factor that affects the results obtained by the schemes is the convergence time, which decides the systems speed to learn. It is defined as the number of reward iterations needed for an initial probability to reach a given maximum value. When the probability of a gray level is rewarded, the value that will be added to this probability and will be decided by the convergence factor. The convergence time of this system must not be

very fast because the probability will fluctuate and it must not be too slow because the system will not catch the variation of the gray levels [17].

2.3. Hybrid System with Learning Automata

In this system, the quantized difference is coded by Learning Automata instead of using the Huffman coder. The scheme used in this system is the Linear Reward – Reward (LR-R) scheme. The probabilities are updated pixel by pixel so the coder matched to local statistics the input data (in this case the probabilities of the quantized differences). The automata determine the updating procedure of the probabilities. The input to the environment is the quantized difference (QD). According to the input value of the environment, the Huffman code determines a distinct Huffman sequence from the probabilities distribution. The code corresponding to the probability of the input value is stored or transmitted. At the same time, the automaton according to some rule updates the probabilities and feeds them back to the environment. The next value is input to the environment and the procedure is repeated. The updating is based on increasing the probability belonging to the input value and decreasing the remaining probabilities. When a sequence of an adjacent pixel of equal value exists, then the probability of that value will be increased and its code becomes short which results in a compression of an image file. Probability increasing operation is called a reward while the decreasing operation is called a punishment.

The linear reward- reward (LR-R) scheme has this name. Because the probability of the gray level of the picked pixel is always rewarded whatever the value of that pixel is. This is the motivation behind using this scheme in the present proposed systems. The model of this scheme is a P-model in which the output of the environment takes one of two values (0 or 1), and the penalty probabilities (ci) are constant that means the environment is stationary.

The following procedure was performed to compute the Hybrid System with Learning Automata:

Step1: Initialization: The probability array of the environment is first Initialized by assigning an equal value of probability to each gray level.

Step2: Picking a pixel: The first pixel from the quantized difference to be compressed is picked.

Step3: Coding the probability: The Huffman code is determined using the probability distribution of the gray level.

Step4: Rewarding the probabilities: The function that rewards the probabilities is derived from Eq. (6a) in which only the first part of this equation is used because of only one response received from the environment, which is the success response.

From eq. (6a) we let

$$g_j(P(n)) = \alpha P_j(n) \tag{7}$$

Therefore, the updating algorithm of eq. (6a) will be written as:

$$P_i(n+1) = P_i(n) + \alpha * [1 - P_i(n)] \tag{8}$$

Where (α) is the reward parameter that defined as:

$$\alpha = kx\,|(\cos(\theta)| \tag{9}$$

Step5: Penalty the other probabilities: The other probabilities are decreased (penalty) according to the equation

$$P_{i \neq j}(n+1) = P_i(n) - \alpha * [1 - P_i(n)]/(L-1) \tag{10}$$

Where L is the gray level.

The motivation behind the use of this function is to punish the other probabilities with the same amount.

Step6: Normalization: Because the summation of the updating probabilities will not equal one therefore a normalization function is applied to these probabilities so that the summation is equal to one.

The above steps will be repeated until the end of the image is reached. : The flowchart of hybrid system with learning automata is shown in Fig. 5

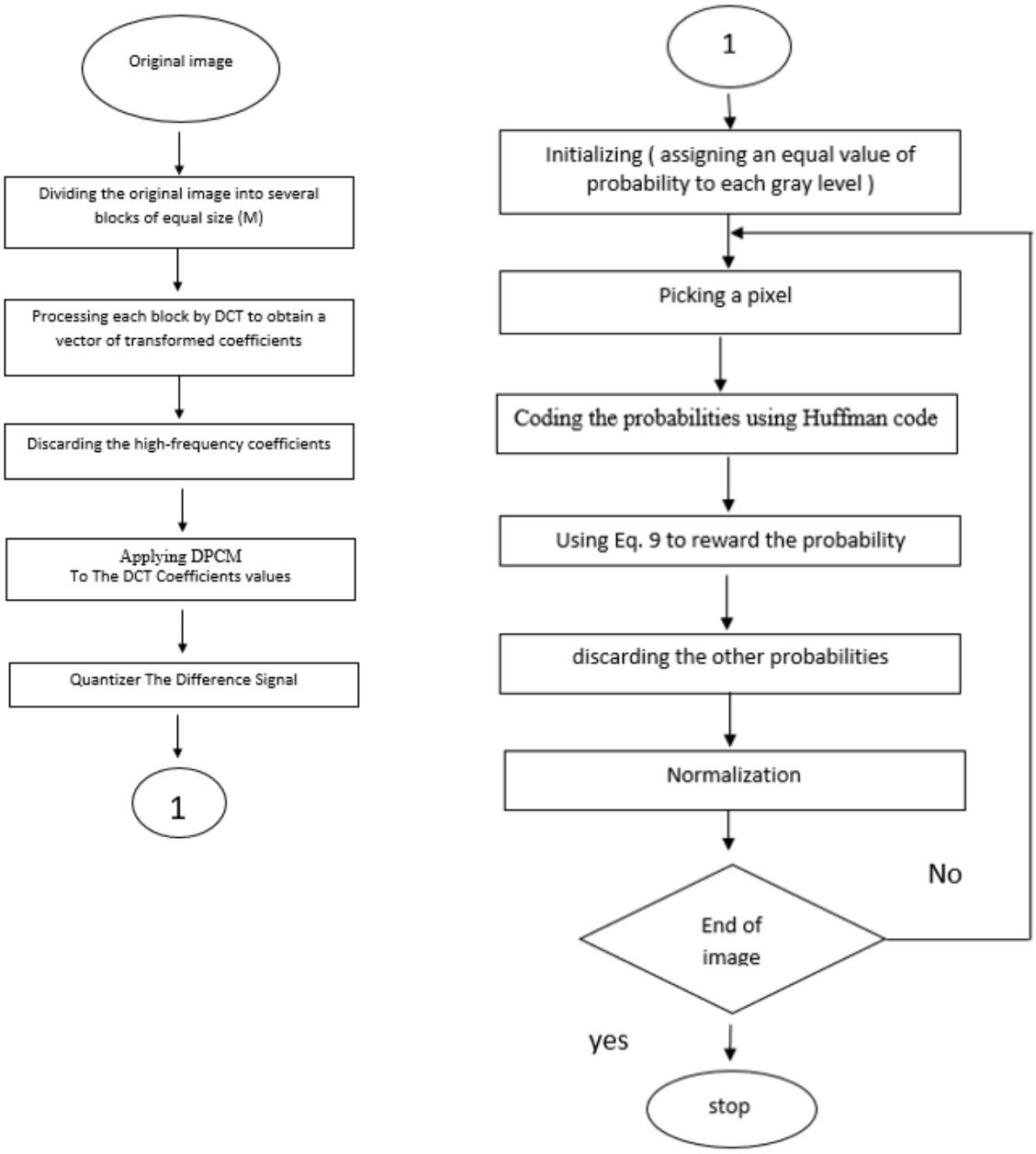

Figure 5: The flowchart of hybrid system with learning automata.

3. Results and discussion

3.1. Results of Hybrid (DCT/DPCM) System

The images are classified according to their information contents as shown in Fig. 6. From this figure, one can identify that the image "plan" includes smooth regions. Therefore, this image has poor details. Furthermore, the image "solider" has moderate information because it contains some smooth area and some texture while the image "monkey" has the highest information. The average information (entropy) of each image was determined to support this argument. This measure gives us a theoretical minimum for the average number of bits per pixel capable of encoding the image. The number of bits per pixel is theoretically optimal and could be used as a metric for judging the success of a coding scheme.

(Plan) (Soldier) (Monkey)

Figure 6: Three images to be compressed by the systems.

Table1 shows that the entropy of Plan is the lowest one, and then of Soldier and the entropy of Monkey is the highest one.

Table 1: The entropy values of images shown in Figure 6.

Image	Entropy
Plan image	5.881
Soldier image	7.536
Monkey image	7.622

The first system was applied to the original images shown in Fig. 6. Each row is subdivided into blocks of equal size. Each block is considered a row vector. The selected block size must not be long to avoid the blocking effect because the inter-pixel correlation becomes weak. Simulation results prove that the best block size is 128 pixels.

A one-dimensional discrete cosine transform (1-D DCT) was applied to each block to obtain the transformed coefficients decorrelated from each block. The DCT arranges the energy values into a few numbers of low frequencies-associated transferred coefficients. The high coefficients values are replaced by zero coefficients with little loss in information and without perceptible degradation in the reconstructed image. The process of removing high-frequency values provides a trade-off between compression ratio and image quality (peak signal-to-noise ratio, PSNR) [25].

The DCT coefficients with the same horizontal frequency are highly correlated. Therefore, a first-order one-dimensional differential pulse code modulation (DPCM) is applied to column-wise to the transform coefficient to build a difference signal. The difference signal between coefficients of the same frequency will be very small. The difference signal has been quantized with a non –uniform quantizer with normalized levels distribution. The actual levels are obtained by multiplying the normalized level with the standard deviation of the quantizer input, which was found to be 50.

Figures (7a to 7d) show the original image with (8 bit/pixel) and the reconstructed images using only 45 transform coefficients and 2-bit, 3-bit, and 4-bit quantizer which gives an average bits/pixel equal to (45×2)/128 = 0.7 bit/pixel, 1.05 bit/pixel and 1.4 bit/pixel respectively. The reconstructed image (b) with a 2-bit quantized shows a little noticeable noise. Image (c) with a 3-bit quantizer has reasonable quality, whereas image (d) with a 4-bit quantizer has an acceptable look with little smearing in some part of the image.

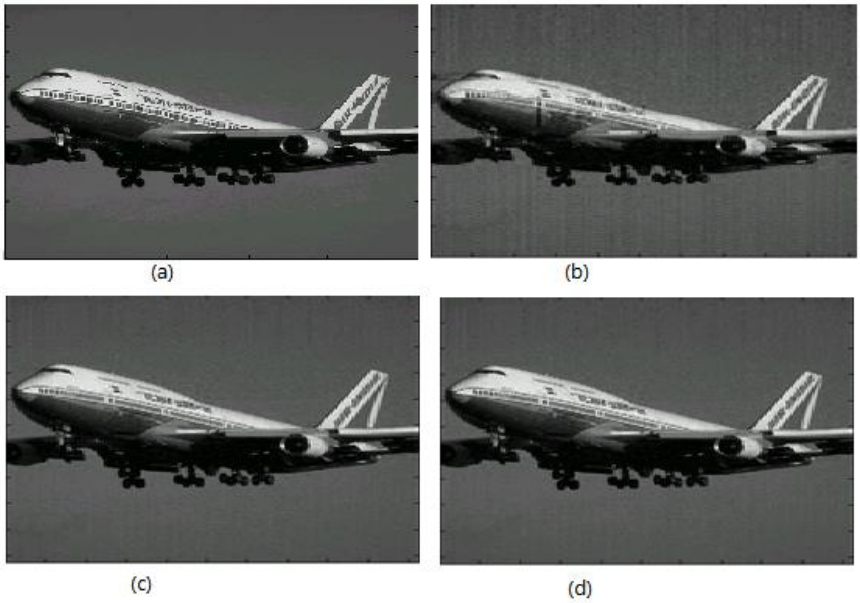

Figure 7: (a:) Original image of 8 bpp, (b): reconstructed image using (2 bit) quantizer, ave. = 0.7 bpp. (c): the reconstructed image using (3 bit) quantizer, ave. = 1.05 bpp, and (d): the reconstructed image using (4 bit) quantizer, ave. = 1.4 bpp.

Figure 8a shows the original plan, including 256 levels (8 bit/pixel), whereas Fig. (8b to 8d) shows the reconstructed image using 45, 40, and 35 transformed coefficient, respectively. For all images, a 4-bit quantizer has been used. Without a doubt, image (d) has noticeable smearing, especially in the edges. Image (c) has hardly recognizable noise and image (b) is exactly as the original. The coefficients (45) are selected from each block because larger coefficients than 45 lead to a reduction in image quality and compression ratio.

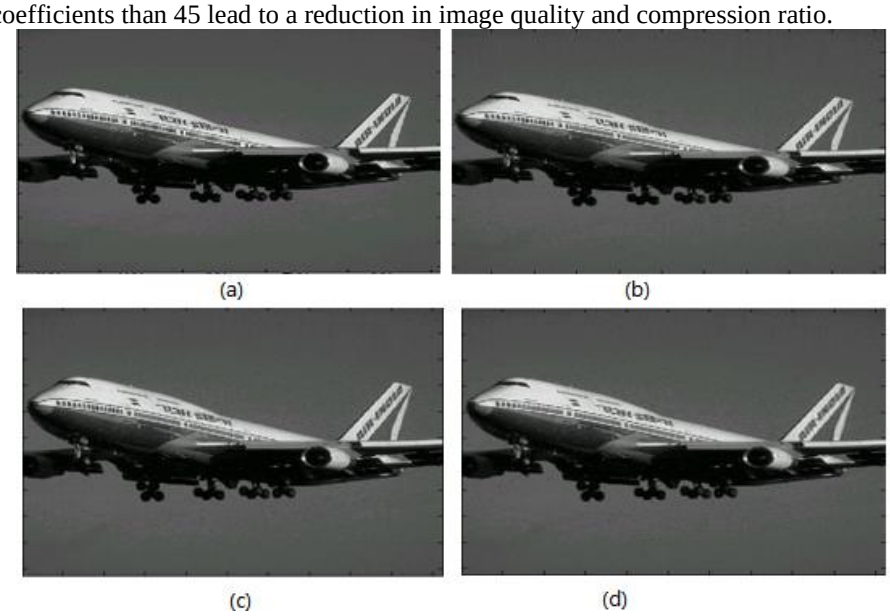

Figure 8: 6. (a). The original image with (8 bpp), (b).The reconstructed image with 45 column/block (1.406 bpp), (c): the reconstructed image with 40 column/block (1.25 bpp) and (d): the reconstructed image with 35 column/block (1.094 bpp).

The peak signal to noise ratio (PSNR) is computed for all images with 45, 40, and 35 coefficients and a 4-bits quantizer. Table 2 provides the compression ratio (CR), average bit per pixel, and PSNR for the latter cases. The average bits per pixel is calculated using the following formula.

$$ave.bitperpixel = \frac{No.ofretainedcofficients \times quantizerbit}{blocksize} \qquad (11)$$

Table 2: Results for hybrid DCT/DPCM of Plan image using (4-bit) quantizer

No. of selected Column/ vector	Compression Ratio	Ave. (bpp)	PSNR (dB)

45 column	5.689	1.406	32.594
40 column	6.4	1.25	31.801
35 column	7.314	1.094	30.709

The same system was applied to the images "Soldier" and "Monkey" as shown in Figures (6b to 6c) respectively. The reconstructed images are shown in figures (9 and 10) and the subjective test containing CR, Ave. and PSNR are listed in tables (3 and 4). Table (5) shows the entropy, Huffman code for remaining pixel, overall average, and compression ratio for quantized difference obtained by applied hybrid (DCT/DPM) system with 4-bit quantized.

The overall average and total compression ratio calculated using the following formulas:

$$OverallAve.(bpp) = \frac{No.oftheremainingcofficient \times HC(bit/pixel)}{block \quad size} \tag{12}$$

$$compression\ ratio(CR) = \frac{uncompressed\ \ file\ size}{compressed\ \ file\ siz} = \frac{8(bpp)}{overall\ \ ave.(bpp)} \tag{13}$$

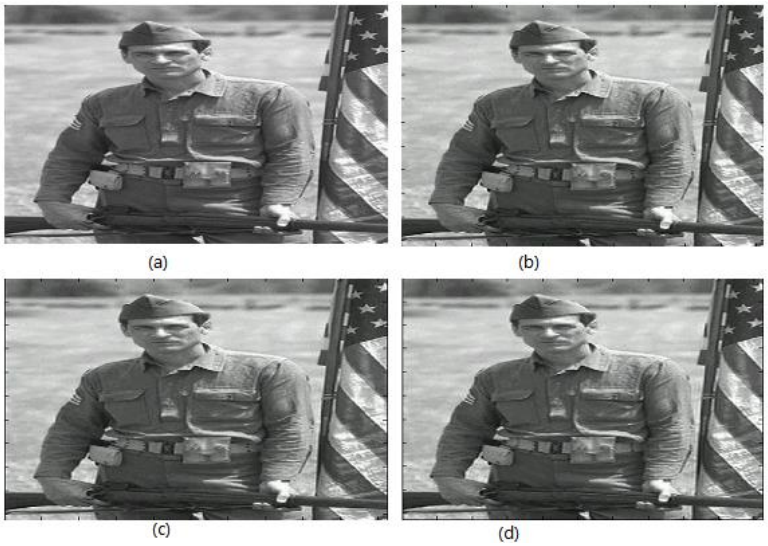

Figure 9: . (a).The original image with (8 bpp), (b).The reconstructed image with 45 column/block (1.406 bpp), (c): the reconstructed image with 40 column/block (1.25 bpp), (d): the reconstructed image with 35 column/block (1.094 bpp).

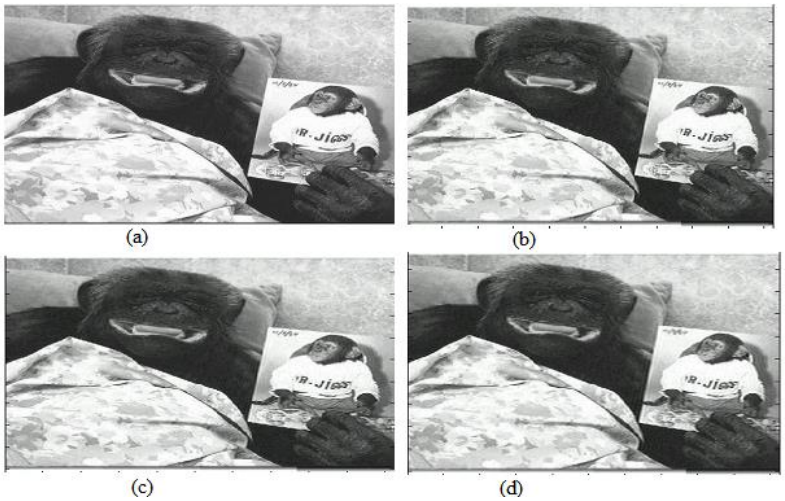

Figure 10: (a): the original image with (8 bpp),(b): the reconstructed image with 45 column/block (1.406 bpp), (c): the reconstructed image with 40 column/block (1.25 bpp), (d): the reconstructed image with 35 column/block (1.094 bpp).

Table 3: hybrid DCT/DPCM results of Soldier image using (4-bit) quantizer.

No. of selected Column/ vector	Compression ratio	Ave. (bpp)	PSNR (dB)
45column	5.689	1.406	37.327
40 column	6.4	1.25	36.667
35 column	7.314	1.094	35.001

Table 4: hybrid DCT/DPCM results of Monkey image using (4-bit) quantizer.

No. of selected Column/ vector	Compression ratio	Ave. (bpp)	PSNR (dB)
45 column	5.689	1.406	25.668
40column	6.4	1.25	25.497
35 column	7.314	1.094	24.9221

Table 5: The Entropy, Huffman code, overall Ave. and comp. the ratio obtained by using a hybrid (DCT/DPCM) system with a 4-bit quantizer.

Image	Entropy Bit/pixel	HC For remaining pixel Pit/pixel	Overall Ave. Bit/pixel	Comp. Ratio.
Plan	1.9985	2.0918	0.735	10.884
soldier	2.3758	2.4015	0.844	9.479
monkey	2.5636	2.6559	0.93	8.602

From table 5 it is noticeable that the entropy values of quantized difference probability for remaining coefficients result by hybrid (DCT/DPCM) system is less than the entropy of gray level probability for original images, those shown in table 3. The mean number of bits used to represent the quantized difference is less than the number of bits used to represent the gray level probability. The overall average for reaming coefficients for (DCT/DPCM) with variable length code is less than the average bits per pixel for reaming coefficients with fixed-length code, that shown in tables 2 to 4. Therefore, the hybrid (DCT/DPCM) system provides a good compression ratio (CR).

3.2. Results of Hybrid (DCT-DPCM) With (LA) System

In this section, we present the application of (LR-R) system to quantize the signals obtained from the hybrid (DCT/DPCM) system. The performance of the linear reward – reward (LR-R) scheme is strongly influenced by the convergence time. Hence one of the most important steps to optimize the performance is the setting of the convergence time. The convergence time depends on the reward parameter defined in Eq. (9).

The convergence time is chosen as the number of iteration to increase the probability from a minimum value to a predefined maximum probability. The minimum value was chosen to be the initial probability, which in this case is equal to (1/16 = 0.0625). The maximum probability is chosen to have a one-bit code when the probabilities are Huffman code. Therefore, the value of maximum probability is selected as (P_{max} = 0.55). This satisfies the Huffman code for the quantized difference probability to become one-bit code when or before the quantized difference probability arrives at (P_{max}). The run is considered to be (10 iterations) which is a compromise. The value of θ was found to be 0.95 degrees and k = 0.055.

Table 6 shows the average bits rate for the remaining coefficients, overall average, and compression ratio obtained by applying the linear reward–reward (LR-R) scheme to the quantized difference with a 4-bit quantizer. The overall average and total compression ratio were calculated. From table (6), one can notice that the average bits per pixel for remaining coefficients hybrid (DCT/DPCM) with (LA) is less than the entropy and average bits per pixel for Huffman code for remaining coefficients that obtained by hybrid (DCT/DPCM) without (LA). The overall average bit per pixel for hybrid (DCT/DPCM) with (LA) is approximately half of the overall average bit per pixel obtained by hybrid (DCT/DPCM) without (LA), therefor the compression ratio obtained by hybrid (DCT/DPCM) with (LA) is approximately double of compression ratio obtained by hybrid (DCT/DPCM) without (LA). Finally, table 7, illustrates the comparison between the overall average bit rate and compression ratio obtained when compressing the image plan, soldier and monkey by using a hybrid (DCT/DPCM) system and by using hybrid (DCT/DPCM) with (LA) system

Table 6: The average bit rates and compression ratio when using the (LR-R) scheme on the quantized difference for images plan, soldier and monkey, and quantizer(4 bit).

Image	LR_R For remaining coefficient bit/pixel	Overall Ave. Bit/pixel	Comp. Ratio.
Plan	1.1034	0.388	20.618
soldier	1.1213	0.395	20.235
monkey	1.1719	0.42	19.047

Table 7: Comparison between the overall average bit rate and compression ratio obtained when compressing the image.plan, soldier and monkey by using a hybrid (DCT/DPCM) system and by using hybrid (DCT/DPCM) with (LA) system

Image	Hybrid (DCT/DPCM) system		Hybrid (DCT/DPCM) with (LA) system	
	Overall Ave. Bit/pixel	Comp. Ratio	Overall Ave. Bit/pixel	Comp. ratio
Plan	0.735	10.884	0.388	20.618
Soldier	0.844	9.479	0.395	20.235
Monkey	0.93	8.602	0.42	19.047

Conclusion

The row-wise DCT removes the horizontal inter-pixel redundancy. Using DPCM to the DCT coefficient removes the vertical inter-pixel redundancy. Large blocks size will decrease the compression ratio without improvement in image quality. Small blocks size will increase the compression ratio but will result in a blocking effect. Using the Huffman code directly to the quantizer difference needs overhead information. Since the direct application of the Huffman code matches the overall statistic of the signal, it does not remove the local redundancy. Learning Automata is preferable since it removes all the redundancy in the row data. However, in learning automata, a Huffman code is determined pixel by pixel which takes a long time.

In the first hybrid (DCT/DPCM) system the quantized difference must be computed, the probabilities calculated, and the Huffman code for the calculated probabilities distribution must be designed. The Huffman code therefore must be designed and stored or transmitted for each image as overhead information, whereas the system using LR-R does not need to transmit overhead since it calculates the Huffman code pixel by pixel depending on the update probabilities. Moreover using Huffman code direct to the data does not consider the local variation of the signal. Learning Automata takes into account the local variation.

References

A. M. a. M. H. Rahman, “Lossless Image Compression Techniques: A State-of-the-Art Survey,” Symmetry, vol. 11, no. 10, pp. 1-22, 2019.

B. C. a. O. A. Chrysafis, “Line-based, reduced memory, wavelet image compression,” IEEE Transactions on Image processing, vol. 9, no. 3, pp. 378-389, 2000.

C. M. K. I. R. K. a. M. P. Mihcak, “Low-complexity image denoising based on statistical modeling of wavelet coefficients,” IEEE Signal Processing Letters, vol. 6, no. 12, pp. 300-303, 1999.

D. Saif alZahir, “A fast lossless compression algorithm for Arabic textual images,” in IEEE International Conference on Signal and Image Processing Applications, 2011.

E. R. M. a. A. K. Thanki, Hybrid and Advanced Compression Techniques for Medical Images, springer, 2019, pp. 1-15.

F. P. a. S. T. Hurtik, “A review on the application of fuzzy transform in data and image compression,” Soft Computing, vol. 23, no. 23, pp. 12641-12653., 2019.

G. S. Y. B. a. V. M. Chang, "Adaptive wavelet thresholding for image denoising and compression," IEEE transactions on image processing, vol. 9, no. 9, pp. 1532-1546., 2000.

H. S. K. F. a. C. S. Miaou, " A lossless compression method for medical image sequences using JPEG-LS and interframe coding," IEEE transactions on information technology in biomedicine, , vol. 13, no. 5, pp. 818-821, 2009.

I. K. S. a. M. A. T. Narendra, Learning automata: an introduction, Courier corporation, 2012.

J. A. e. a. Rezvanian, Learning automata approach for social networks, springer, 2019..

K. Y. a. Y. H. Lin, "An ω-automata approach to the compression of bi-level images," Electronic Notes in Theoretical Computer Science, vol. 31, pp. 170-184, 2000.

L. K. a. K. J. Culik II, " Image compression using weighted finite automata," Computers & Graphics, vol. 17, no. 3, pp. 305-313, 1993.

M. I. K. a. K. J. Culik, " Finite automata computing real functions," SIAM Journal on Computing, vol. 23, no. 4, pp. 789-814, 1994.

N. K. S. a. M. A. T. Narendra, " Learning automata-a survey," IEEE Transactions on systems , man, and cybernetics, vol. 4, pp. 323-334, 1974.

O. A. A. S. A. a. P. M. Hashim, "Application of learning automata to image data compression," Adaptive and learning systems, pp. 229-234., 1986.

P. M. a. C. S. Das, "Efficient lossless image compression using a simple adaptive DPCM model," in In Proceedings of the 44th IEEE 2001 Midwest Symposium on Circuits and Systems. MWSCAS, 2001.

Q. K. a. O. B. Hashem, "Using Learning Automata to Model the Behavior of a Teacher in a Tutorial-like," in IEEE International Conference on Systems, Man and Cybernetics System, 2007.

R. Y. A. A. M. M. a. B. A. Lin, "Task-based JPEG 2000 image compression: an information-theoretic approach," in 2018 Data Compression Conference, 2018.

S. A. J. A. A.-F. a. N. R. Hussain, "Image compression techniques: A survey in lossless and lossy algorithms," Neurocomputing , vol. 300, pp. 44-69., 2018.

T. V. P. N. a. R. A. Sanchez, "Efficient lossless compression of 4-D medical images based on the advanced video coding scheme," IEEE Transactions on Information Technology in Biomedicine , vol. 12, no. 4, pp. 442-446., 2008.

U. X. e. a. Ruan, "Dynamic cellular learning automata for evacuation simulation," IEEE Intelligent Transportation Systems Magazine, vol. 11, no. 3, pp. 129-142., 2019.

V. J. a. M. M. Torkestani, "A learning automata-based heuristic algorithm for solving the minimum spanning tree problem in stochastic graphs," The Journal of Supercomputing, vol. 59, no. 2, pp. 1035-1054, 2012.

W. M. R. a. M. R. M. Mirsaleh, "A learning automata-based memetic algorithm," Genetic Programming and Evolvable Machines , vol. 16, no. 4, pp. 399-453, 2015.

X. J. X. L. M. J. a. Z. M. Zhang, " A learning automata-based particle swarm optimization algorithm for noisy environment," in In 2015 IEEE Congress on Evolutionary Computation (CEC) , 2015.

Y. K. K. S. S. S. a. M. U. B. Hasan, "Reserved Discrete Cosine Transform Coefficients Effects on Image Compression," IOP Conference Series: Materials Science and Engineering. Vol. 454. No. 1. IOP Publishing, 2018., vol. 454, no. 1, pp. 1-7, 2018.