Received: August 31, 2021. Revised: October 25, 2021. 12

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Quantum-dot Cellular Automata Based Lossless CFA Image Compression Using Improved and Extended Golomb-rice Entropy Coder

Mahesh Boddu[1]* **Soumitra Kumar Mandal[2]**

[1]*Department of Electrical Engineering, Maulana Abul Kalam Azad University of Technology, Kolkata, West Bengal, India*
[2]*Department of Electrical Engineering, National Institute of Technical Teacher's Training and Research, Kolkata, West Bengal, India*
* Corresponding author's Email: mboddu567@gmail.com

Abstract: This article presents an efficient hardware architecture of lossless image compression mechanism using colour filter array (CFA) for wireless camera networks. Recently, Quantum-dot cellular automata (QCA) technology has been widely using in the field of VLSI, thus the proposed compressor is developed using QCA technology to mitigate the area, and power as compared to standard CMOS technology. In addition, an improved and extended Golomb-Rice entropy coder (IEGREC) is proposed to reduce the memory requirement, and computational complexity. The IEGREC approach eliminates the requirement of context module, which is a major component in lossless image compressor architectures and its memory to lessen the hardware resource utilization. Further, to maintain the pixel connectivity and higher compression ratio (CR), the proposed architecture utilizes the adaptive Golomb-Rice (GR) parameter prediction and control module. Finally, packer module is used to maintain the flow of output compressed bit stream. The extensive simulation results show that proposed QCA-IEGREC method performs superior as compared to state-of-art approaches in terms of lesser area as 3700 gate count, and reduced power of 1.02 mW, and even image quality statistics such as 18.78dB of peak signal-to-noise ratio (PSNR), 152.3 of figure-of-merit (FOM), and 29.3 of CR.

Keywords: Quantum-dot cellular automata, Image compression, Color filter array, JPEG-LS, Golomb rice-codes, Entropy codes.

1. Introduction

Digital Images are utilized in various applications such as satellites, medicals, computers, mobile phones and numerous other sectors. In any applications, the images need to transmit over the network or channel. If the size of the image is high, then it is difficult to transmit the image, it means the Transmission rate is high. Thus, all these fields of applications need the compression operation to transmit the data into with high speed and reduce the transmission time [1]. The main aim of image compression is to reduce the storage and transmission problems that occur in various fields handling the multimedia contents. Image compression schemes are developed to reduce the high-resolution images to cope up with the low resolution supported devices [2]. The major challenge in the image compression process is preserving the class of the image and sharpness of the image even after decompression. Though many compression schemes have been introduced in the recent past to achieve efficient image compression [3], the flaws in the performance led to image degradation. For instance, the prediction methods employed in the image compression schemes mostly depend on the raster search prediction which is inefficient in the high frequency regions. JPEG [4] is the most used compression method and it is the first international standard image compression method as JPEG reduces the size of the image by not including the redundancy. The JPEG approach mainly used discrete cosine transform (DCT), Run

Length algorithms, Huffman coding and Zigzag reordering, respectively. But all these methods consume the higher power consumption and huge hardware resource utilization. In conventional, various literatures are focused on optimization of JPEG process. In [5] authors developed the JPEG by using the Daubechies coefficients, the coefficients are factorized by using the lifting method based pruned discrete Tchebichef transform (PDT). The updated JPEG standard [6] explains the irreversible PDT. Iterative reconstruction directly calculates the image within a single reconstruction step. The JPEG with fixed IP core was implemented with considerable hardware resources [7-8] and this method is worked on gray scale images only. The main advantage of this method is that it improved the insensitivity and capability of the reconstructed image. Image reconstruction using multi-band wavelet coefficient is used for reducing the hardware complexity during the image processing in the hardware components. Thus, the new JPEG-2000 [9] standard is developed with the entropy encoding scheme using discrete wavelet transform (DWT) and this method gained the good CR compared to JPEG-based image compression approaches. These works are well suitable for greyscale and binary images. But this resulted in reduced image quality for RGB (colour) images. Hence, an improved algorithm of JPEG-2000, which is based on code content prediction is introduced. This algorithm utilized DCT to transform the images to frequency domain and divides them into blocks. It employed sorting transform for reducing the inter-block redundancy which is not dominated by the standard JPEG-2000 compression algorithm. The main proposal is to cluster the coefficients with similar statistics and encode those clusters in the same context adaptive arithmetic coding. The neighborhood properties and corresponding contexts are determined by the sorting transformation. The only disadvantage of this algorithm is the blocks cannot be decoded individually after the coefficient grouping [10]. Therefore, the image compression algorithm using low complexity architectures named as LOCO-I has been implemented to overcome those drawbacks. It is the first standardization of JPEG-LS, but this method has failed to provide the maximum compression standards [11]. Later, pipelined LOCO-I hardware architecture [12] and JPEG-LS (lossless JPEG) [13] with the low-power controlling properties has been developed for image compression. Unlike, the conventional JPEG methods, the JPEG-LS do not contain any transformation method, which results in enhancement of operational speed at the cost of higher computational complexity. Thus, it inhibits the lossless compression standards and can be useful for any kind of applications. Further, the bit plane image compression using the determination of the nearest average of the wavelet coefficients is implemented for wireless endoscopic capsule applications [14]. It utilized stationary probability model with JPEG-LS in image coding systems and additionally the bit-plane coding is jointly utilized with context-adaptive arithmetic coding. This method assumes motionless statistical performance for emitted signs. Probabilities were resolute using the instant neighbors of the present coded coefficient. But the major drawback of these approaches is that it results in low CRs. Thus, various authors performed research on image compression methods without using any transformation methods and they have tried to improve CR as well. In [15], JPEG image compression technique was developed using orthogonal diagonal cross hair search. Later, DWT with tree-seed algorithm (TSA) is designed, where DWT is used for the division of images into 8×8 blocks [16]. From the divided blocks, the TSA adjusts the quantization scaling factors to make sure the image sub-blocks are compressed at higher quality than the image blocks. The blocks with higher activity are scaled with low scaling while the lower activity blocks are scaled with high scaling factors. This approach improved the image compression performance considerably. However, the drawback with this variable quantization is that the introduction of additional overhead bits in the code bit stream. Mukkara et al. proposed a simple VLSI architecture for the applications of image compression using floating point matrix multiplier [17], which is a hybrid unification of pseudo code for matrix multiplication, the algorithm of CSD multiplication, traditional floating point number format and pipelining concept. Then, memory-efficient, and high-speed architecture is proposed using ortho-normalized multi-stage discrete fast stock well transform [18], which performed the operation of lifting with split, predict and the update operations by assuming the odd samples those were left in traditional lifting operations. As mentioned earlier, the discrete transforms have been broadly utilized in the applications of image compression. Thus, discrete Tchebichef transform (DTT)-based image compression [19] is developed as an alternate for the DCT-based compression standards due to its properties of energy compaction. Though, the computational cost has been considerably reduced as compared to DCT, but the compression performance was roughly like DCT performance. Further, the

Daubechies wavelet filters are utilized for image compression, which is applied to DWT to extract the various properties of time and frequency [20]. Usually, the architectures of set partitioning in hierarchical trees (SPHIT) were employed for image/video compression systems, that need higher memory and huge computational time with complicated sorting algorithms. In addition, these SPHIT-based compression architectures also required higher time [21]. For an effective utilization of hardware resources in the applications of image compression, a hybrid compression operation using selective sector concept is designed and implemented [22].

However, the lossless image compression has been obtained only for the selective sector of input image while all the other sectors of image were lossy compressed. Recent years, the dictionary-based compression technique named as Lempel-Ziv-Welch (LZW) method has been broadly employed in numerous systems of communication and storage applications [23]. However, the computational speed of LZW is bit of slow, but that can be speed up using multiple dictionaries i.e., parallel dictionary based LZW (PDLZW). Be that as it may, the compression gain of PDLZW relies on the address space partitioning (i.e., size of the parallel dictionaries). Thus, it is required to implement the optimal techniques for partitioning of address space. The work addressed in [24] mitigated the memory size of parallel dictionaries by implementing a recursive dictionary structure and a word partitioning technique. However, it suffers from small loss of compression gain due to the increased RAM size of dictionaries. Recently, an image compression for radiographic images was addressed in [25], which utilized region of interest-based hierarchical vector quantization with Huffman coding and obtained an enhanced compression performance over DCT, and JPEG2000 methods. However, above mentioned compression architectures addressed only gray scale image compression applications.

But it is necessary to implement a VLSI architecture for color image compression due to the advancements in digital image processing applications such as remote sensing, dermoscopy, radiology, retinal imaging, aerial imagery, and X-ray imaging etc. Recently, block truncation coding (BTC) with digital halftoning (DHT) is implemented for wireless camera networks [26], which obtained low-complexity, high-performance with lower power consumption and higher CRs. Then, a fuzzy decision-based hardware-oriented VLSI architecture is implemented [27, 28] with the concept of variable-size block partition technique. However, this method majorly focused on random changing of block size and type of block. Specifically, there was no precise finite state machine (FSM) to change the block size, which results in degraded performance. Further, the low-power and high-performance VLSI architectures for colour image compression in wireless camera networks and low-power embedded systems still need to be addressed.

In [29] authors developed the parallel IP-core, which is used to compress the colour images using run length encoding. In [30] authors used multi-wavelet transform with biorthogonal properties for image compression through direct pipeline mapping. These conventional approaches are suffering with the area, delay, and power issues, which needs to be addressed. In addition, recent advancements in VLSI architectures are focusing on QCA-based implementation [31] for image processing applications. Thus, this article proposes advanced QCA-based VLSI implementation for CFA image compression.

The major contribution of this work is as follows:

- Implementation of QCA-based colour image compressor architecture using IEGREC approach.
- In addition, adaptive GR parameter module also used to generate the perfect composition of code for IEGREC, which resulted in improved CR without degrading the visual performance.

Rest of the paper is organized as follows: Section 2 gives the detailed analysis of the QCA technology. Section 3 briefs the description of various modules with the overall architecture of the proposed QCA-IEGREC method with its operation. Section 4 detailed the analysis of obtained results and compared with the literature. Finally, section 5 concludes the work with possible future implementations.

2. QCA-based design

2.1 QCA basics

There are two vigorously irrelevant approaches of two electrons in the QCA cell for an evacuated cell, expected cell polarization $P = +1$ and cell polarization $P = -1$. While cell polarization $P = +1$ alludes to parallel 1 while cell polarization $P = -1$ alludes to relating 0. In expansion, this thought is graphically portrayed in Fig. 1. It is additionally colossal that there is an unpolarized state as well. In an unpolarized state, potential points of confinement

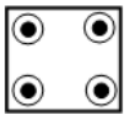

Figure. 1 Unpolarized cell

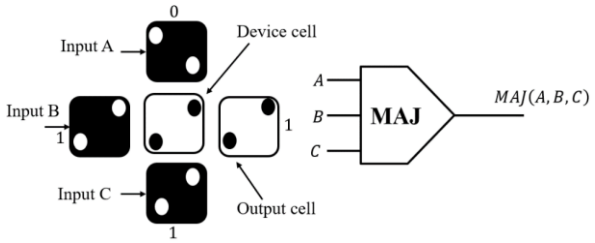

Figure. 2 Majority gate using QCA

between contact are diminished which diminishes the exhibit generally zero polarization and the two electron wave limits have been delocalized over the telephones appeared in Fig. 1. The numbering of specks signified by electrons in the cell goes clockwise beginning from the matrix on the upper appropriate with quantum level $I = 1$, base right cell $I = 2$, base left cell $I = 3$, and upper left cell $I = 4$. The polarization level P in a cell is characterized and here P_i denotes electronic charge at speck current. The polarization estimates the charge design for example the degree to which the electronic charge is appropriated among the four cells.

The basic QCA sensible circuit is the three-input majority gate (MAJ) that shows up in Fig. 2 from which progressively complex circuits can be fabricated. The fundamental MAJ gates are acquired by setting four neighboring cells bordering to a quantum cell, which is in the center. Three of the side cells are utilized as data sources, while the staying one is the yield. The quantum cell will consistently expect the majority polarization is where there will be at least charge between the electrons in the three information cells and the quantum cell. Consider the coulombic interface between cells 1 and 4, cells 2 and 4, cells 3 and 4 to perceive how the contraption cell accomplishes its most minimal imperativeness state (and from now on P=+1 in Fig. 2).

Typically, coulombic association between electrons in cells 1 and 4 would make 4 change its polarization in light of electron stun. (Expecting cell 1 is a data cell). In any case, cells 2 and 3 in like manner sway the polarization of cell 4 and have polarization $P = +1$. Along these lines, in light of the way that the greater part of the cells influencing the contraption cell has polarization 1 P, it additionally will moreover expect this polarization because the forces of Coulombic association are more grounded for it than for 1.

2.2 Proposed adder and subtractor

The proposed method is implemented with only three 3-input majority gates and two inverters and there is no need of five input majority gates. So, the number of quantum cells reduced by using the proposed majority gates. Fig. 3 represent the joint full adder and full subtractor (FAFS) diagram, which is developed using QCA architecture, where the single circuit performs both the operations of addition and subtraction. Majority gate MAJ1 generates the carry out (c_{out}) and Majority gate MAJ2 generates the borrow out (b_{out}). MAJ3 takes c_{out}, b_{out}, and a as inputs to generate both the sum and difference. Fig. 4 represents N-bit adder and subtractor, which is developed by connecting one-bit full adder and subtractor in the series manner.

Here, the c_{out}, and b_{out} of the first FAFS block is connected as input to the next stage. If selection line of 2×1 mux is zero, then the proposed circuits act as adder and c_{out} of each module is applied as input to the next stage. If selection line of 2×1 mux is one, then the circuit proposed circuit acts as subtractor and b_{out} of each module is applied as borrow input to the next stage.

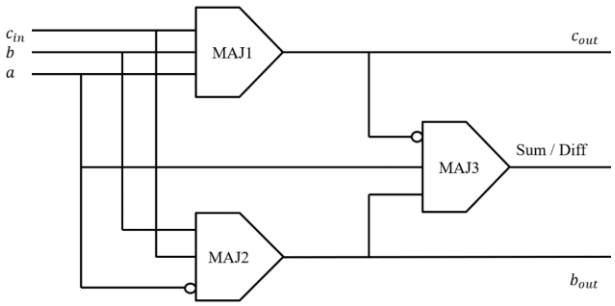

Figure. 3 Proposed full adder and subtractor

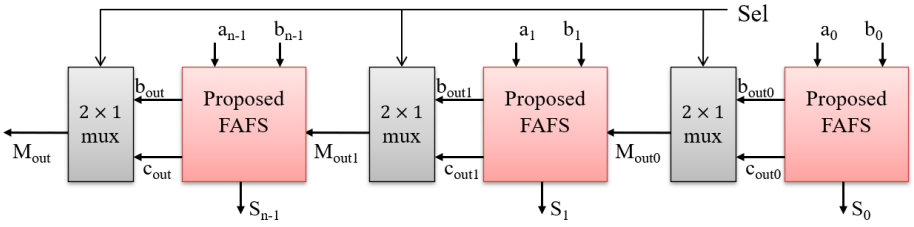

Figure. 4 Proposed N-bit adder and subtractor

3. Proposed methodology

The conventional JPEG-LS compression approach consisting of five models, they are an entropy coding model, run mode model, prediction model and context model. But this method inhibits the higher memory requirement and computational complexity. Thus, the area, and power requirements will be reduced by eliminating the context model and run mode model. Fig. 5 presents the proposed lossless CFA image compression algorithm using QCA-IEGREC, which contains IEGREC coding model, improved Golomb-Rice entropy (IGRE) code, extend Golomb-Rice entropy (EGRE) code, direction detection model, prediction model, and pixel restoration model. All these models are implemented with the hybrid technology. Initially, image is applied to the pixel restoration model; it converts the CFA based input image to RGB format. The main purpose of direction detection model is identifying the direction of pixel continuity. Then, the predictor module is used to predict the accurate compression value and then it applies it to IEGREC encoder. Finally, the IEGREC encoder generates the compressed bit stream.

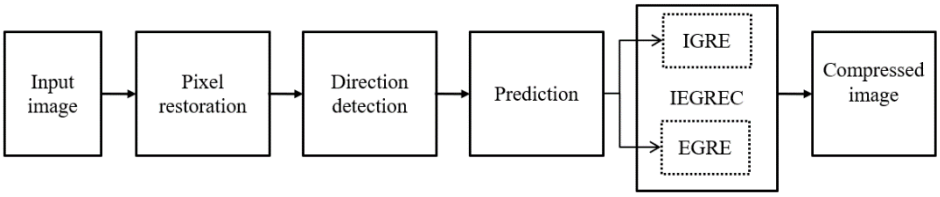

Figure. 5 Proposed QCA-IEGREC block diagram for color image compression

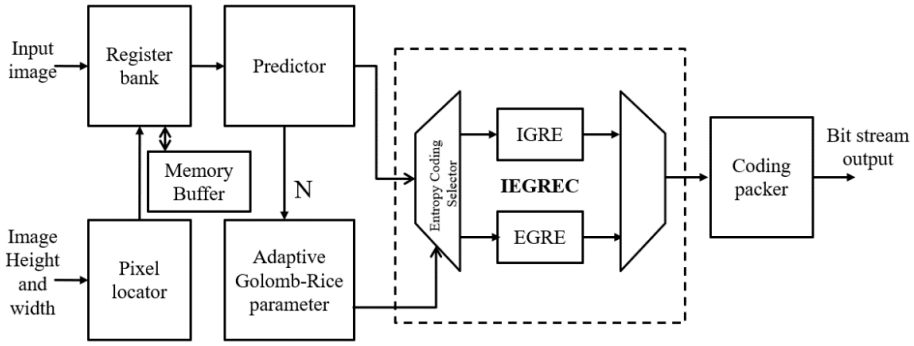

Figure. 6 Block diagram of proposed lossless compression encoder

Table 1. Proposed algorithm for color image compression

Input: Input image pixels Output: Compressed image pixels
Step 1: Apply the input image pixels to the register bank and store them. Step 2: Apply the input image resolutions to the pixel locator, it corrects the CFA pattern and estimates the prediction candidate pixel (X) locations for red, green, and blue colors. Step 3: Now based on X values, the original image pixel locations are updated in the memory buffer of register bank through multiple shifting. Step 4: The predictor is used to identify (predict) the type of compression to be performed and direction of compression to be performed and generates the predicted residual error value (N). Step 5: Parameter N is applied to the adaptive Golomb-Rice Parameter module. It generates the code length variables such as reminder (r) and quotient (q) for various values of predicted parameter (M). Step 6: Entropy coding selector selects the type of encoding based on $r,\ q,\ M,$ and N values. Step 7: If the N is small, then the IGRE is used to encode the data. If N is too large, then the EGRE is used to encode the data. Step 8: IEGREC encoding is used to shorter the code. Here, the IGRE uses short code for the high probability N values. If the code length after IGRE encoding is much longer than the original bit number, then EGRE is uses much short code for encoding the high probability N values. Step 9: Coding packer is used to maintain the constant length of IEGREC code words, such as it reduces the uncertainties. It generates the Compressed image pixels as its final output.

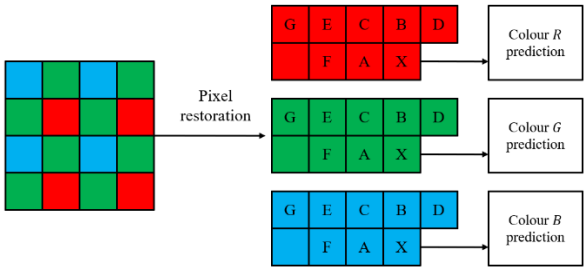

Figure. 7 Pixel restoration from CFA to RGB format

VLSI architecture of proposed lossless CFA image compression approach is presented in the Fig. 6, and it is developed by using QCA based Majority logic gates. Pixel restoration contains pixel locator, it corrects the CFA pattern. Direction detection hardware contains register bank and memory buffer. Prediction hardware contains predictor and adaptive Golomb-Rice parameter modules. IEGREC hardware contains IGRE, EGRE and coding packer, respectively. The detailed operation of proposed algorithm is presented in Table 1.

3.1 Pixel locator

The test CFA image resolutions are applied to the pixel locator. Generally, the neighboring pixels correlation is exceptionally low in input image. The color intensity values of the original image does not changed by the pixel locator, it just rearranges the pixel positions. Thus, the pixel locator extracts the red, green, and blue color components from the input image as presented in Fig. 7, respectively. Since, the colors of the reference pixels A, B, C, D, E, F and G are all not the same as the color of the prediction candidate pixel X, a pixel locator is used to estimate the accurate reference pixels in memory. Thus, the correlation between each colour with neighboring pixels gets improved and results in enhanced prediction performance.

3.2 Register bank

The prediction candidate pixel X is used to estimate the accurate reference pixels in memory generated from pixel locator. The architecture of register bank is presented in Fig. 8, it is connected to memory buffer to generate the pixel connectivity of four neighboring pixels in sequence. Register bank contains two-line-buffer memory for identifying the value of pixel X. The applied test image values are stored into register bank in sequence. Then, the register values are pushed into line buffer. Then, boundary detector presented in the line buffer is used to detect the edges, region of each colour and it also avoids the wrong pixels and wrong colors. The

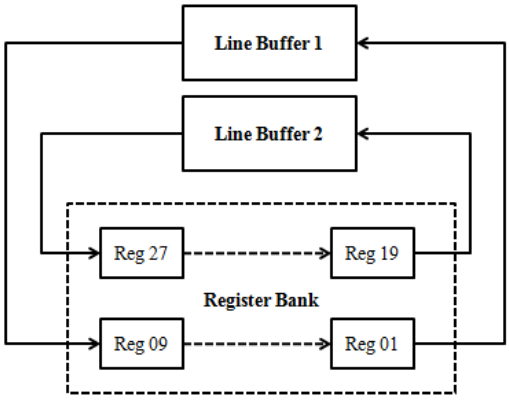

Figure. 8 Architecture of the register bank

continuous shifting operations are performed between the registers and resulted in the updated positions of $A-G$ and X according to the Bayer CFA pattern, respectively.

3.3 Predictor

The pixel values from register bank are applied as input to the predictor model. This model is used to identify type of compression for pixel values $A-G$ and reference pixel value X. The CFA corrected pixel values are grouped into seven segments with various prediction directions as shown in Fig. 9. The proposed predictor develops the four possible prediction types based on four directions; they are Type 4 (diagonal down left –DDL), Type 3 (diagonal down right -DDR), Type 2 (vertical-V) and Type 1 (horizontal-H). These directions are used to identify the various edges based on the difference between neighboring pixels.

Here, only A, B, C and, D pixel values are considered for the median values. Because they contain the perfect continuity as compared to the E, F *and* G, respectively. By using the above directions, the new reference pixel value (X_{med}) is calculated for each prediction type as follows:

Type 1 (H) : $X_{med} = A$
Type 2 (V) : $X_{med} = B$
Type 3 (DDR) : $X_{med} = C$
Type 4 (DDL) : $X_{med} = D$

Additionally, the difference is calculated and generates the predicted residual error value (N) as formulated below.

$$N = X_{med} - X \quad (1)$$

Fig. 10 presents the hardware architecture of

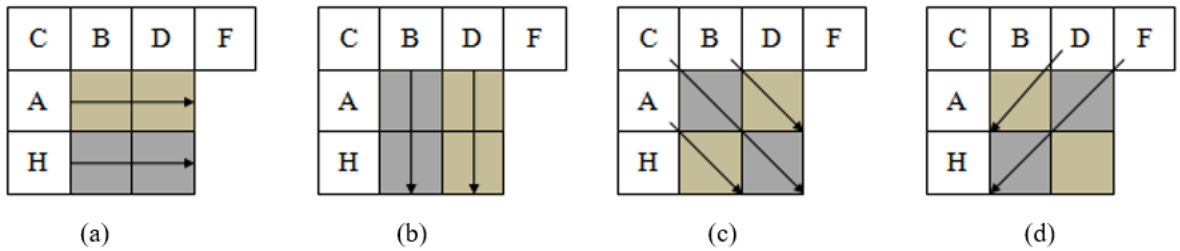

Figure. 9 Four types of pixel directions: (a) Type 1, (b) Type 2, (c) Type 3, and (d) Type 4

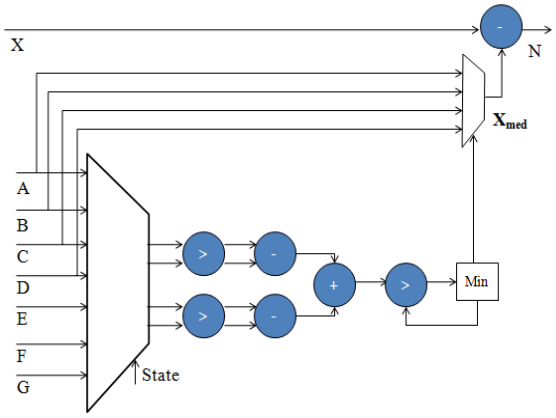

Figure. 10 Hardware architecture of predictor

proposed predictor, and it is developed with low computational complexity as it needs only four cycles to generate the final residual error value. The predicted pixel values $A, B, C, D, E, F,$ and G are applied to a FSM. Here, FSM compares the direction of input pixels with the standard directions with four types and results the matched direction with its type as output.

Here, the state is used to change the direction of input pixels. Like this, the four types of directions similarities are calculated. Then, they each of two are further compared $(>)$. So, highest, and lowest values are identified and subtracted. Then, the resultant subtracted values are added together and applied to the minimum value identifier. Through the iteration process, the minimum value generates the control selection line for 4 to 1 multiplexer. Then, the multiplexer selects the new reference pixel value (X_{med}) from inputs $A\ to\ D$ based on its selection type. so, the pixels continuity is perfectly established, it is used for higher CR with less educed image quality. This $A\ to\ D$ continuity also reduces the final error rate. Finally, the residual error value (N) is calculated by subtracting the X and X_{med}.

3.4 Adaptive GR parameter module

Golomb rice coding (GRC) is the best entropy coding, and it consumes less hardware resources as compared to BTC. Adaptive GR parameter module is utilized to choose the best value of tunable parameter M by considering the various conditions and generates the optimal combination of quotient q and reminder r for IEGREC code generation, respectively.

For instance, if N is huge and it is divided by smaller tunable parameter $M = 2$, then the code length of q becomes higher. Similarly, if N is divided by larger parameter $M = 16$, then the code length of q becomes smaller. Thus, it is mandatory to maintain the parameter M level accurately. Hence, this work proposed new method of adjusting the value of parameter M for each N value. This module is used to predict the perfect value of M based on the previous M values. The optimized code length of quotient q will be acquired by using predicted N and M values, respectively. Thus, this prediction trend makes the neighboring pixels are remarkably close. The current pixel quotient q is used to predict the next pixel parameter M, so the perfect compression with high quality is achieved. The value of N is same for each colour and results in the high correlation. Thus, the CR is increased by adjusting the parameter M of GRC.

The hardware architecture of adaptive GR parameter module is presented in Fig. 11, which contains three registers, a shifter, a subtractor, a comparator, and a multiplexer, respectively. Initially, the value of N is stored into a register and various values of M ranged from 2 to 16 are applied to the multiplexer. The conventional mechanisms are using the dividing the M value by N times, such as a divider. The computational complexity is increasing due to this division. Thus, the proposed module replaces the division by shifting operation, such as multiplexer selected M value is shifted by the N times and resulted in the quotient q. The multiplexer selected M value is subtracted by N and results in the reminder r. The q value is stores into a register and feedback to a comparator. The comparator compares the present quotient with the previous quotient value; if the previous value is high then the comparator results outcome as one else comparator generates the outcomes as zero. The comparator outcome is applied as input to the multiplexer

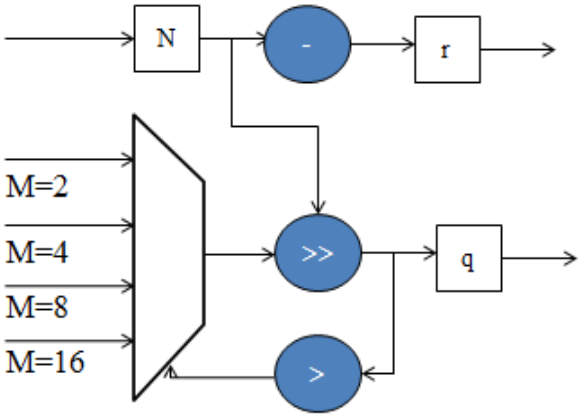

Figure. 11 Architecture of adaptive GR parameter module

selection line. Like this, the process will repeat and present M value is selected and M is utilized to divide the next N value. The process repeats and perfect reminder and quotient values will generate.

3.5 IEGREC approach

The proposed IEGREC utilizes tunable parameter M, which is used to divide the input x into two parts namely quotient q and reminder r, respectively. Further, the parts are derived as follows:

$$q = \left[\frac{x}{M}\right] \tag{2}$$

$$r = x - qM \tag{3}$$

Golomb used the input x to represent the run length of a Bernoulli process with a geometric distribution starting at 0. The optimal value of M is determined by the matching Bernoulli process, which is parameterized by using Adaptive GR parameter module. M is either the distribution's median or the probabilistic median $+/1$. The optimal M value is defined as follows:

$$M = \left[-\frac{\log(2-p)}{\log(1-p)}\right] \tag{4}$$

Here, p is the probability to generate optimal M value, whose range is $(1-p)^M + (1-p)^{M+1} \leq 1 < (1-p)^{M-1} + (1-p)^M$, respectively.

3.5.1. Improved GR entropy (IGRE) coder

The conventional GRC uses two dissimilar codes for two numbers, which contains the dissimilar sign but same value. For instance, the codes of conventional GRC are “101” and “110” for the inputs -1 and $+1$, respectively. Thus, it is mandatory to prepare the codebook with dissimilar numbers for these codes. The hamming distance between the numbers is 2. So, it requires more memory. Thus, IGRE is developed to overcome this problem by reducing hamming distance. It means there is only one bit difference among codes, which have the dissimilar sign but same value. For instance, the codes of IGRE coding are “1101” and “1001” for the inputs -1 and $+1$, respectively for $M = 4$. In this example, k will be set as 2, because LSB 2 bits are same in both codes, so r is treated as LSB 01 bits and MSB 1 is treated as q. The dissimilar bit is treated as the sign bit. As the hamming distance is reduced, the size of the IGRE codebook is reduced to half as compared to the conventional GRC. Saving in the codebook will reduces the resource blocks and reduces the hardware cost. Table 2 illustrates the four examples of IGRE coding for various M values.

Table 2. IGRE codes of proposed method

N	M	Codes
0	2	100
	4	1000
	8	10000
	16	100000
1	2	101
	4	1001
	8	10001
	16	100001
-5	2	00111
	4	01101
	8	11101
	16	110101
17	2	00000000101
	4	00001001
	8	001001
	16	010001

The residual error value (N) generated in the prediction process may positive or negative. Thus, a novel IGRE coding was used to encoder the positive and negative number effectively. This is achieved by adding the extra sig bit to the data. The input data is divided into two parts as shown in Fig. 12 and extra sign bit is added across the lower part LSB. This also reduces the size of IGRE code, respectively. As mentioned in previous sub-section, the parameter calculator block generates the M and divides the N into two parts q and r, respectively. Here, q represents the MSB part and acts as quotient of N. Similarly, r represents the LSB part and acts as reminder of N.

3.5.2. Extend GR entropy (EGRE) coder

The conventional GRC inhibits the properties of variable length coding, thus each quotient value needs a unique code. So, the value of N increases resulted in increment in length of conventional GRC coding, which is usually more than 8-bit in original pixel value. The increment in the length results in the loss of pixel values. Thus, EGRE is developed to reduce the length of conventional GRC codes by simply adding an extra sign bit to it. For an instance, $N = 0$ and $M = 4$ parameters are denoted as 1000. Here, q is zero, r is zero and sign is positive. The negative zero code 1100 is not useful in this scenario. Thus, to avoid the code wastages (non useful codes), the negative zero is treated as an extend code by EGRE codes. Some more examples of EGRE codes are 110 if $M = 2$, 1100 if $M = 4$, 11000 if $M = 8$, and 110000 if $M = 16$.

Table 3 presents the entropy codes of proposed EGRE. Every input number N contains the four types of EGRE codes for each M value. The length of the extended value and extended code of EGRE codes is very lesser than conventional GRC. Thus, the proposed EGRE codes can significantly improves the CRs with reduced hardware. If the N is small, then the IGRE is used to encode the data. If N is too large, then the EGRE is used to encode the data. The encoding region comparison of proposed IGRE and EGRE codes are presented in Table 4 for various values of M.

If N is divided by the bigger parameter $M = 16$, then the code length becomes smaller. To control the code length, Adaptive Golomb-Rice Parameter block is used to generate the adjustable M for each N values. Thus, it is concluded that the IGRE uses short code for the high probability N values. If the code length after IGRE encoding is much longer than the original bit number, then EGRE uses much short code for encoding the high probability N values.

Table 3. EGRE codes of proposed method

N	M	Codes
1	2	101
	4	1001
	8	10001
	16	100001
-1	2	111
	4	1101
	8	11001
	16	110001
63	2	11001001111
	4	110001001111
	8	000000010111
	16	000101111
-87	2	11001010001
	4	110001010001
	8	1100001010001
	16	00000110001

Table 4. Encode region comparison of proposed IGRE and EGRE

M	IGRE encode region	EGRE encode region
2	-17 to +17	- 255 to -17 and 17 to 255
4	-35 to +35	- 255 to -35 and 35 to 255
8	-71 to +71	- 255 to -71 and 71 to 255
16	-143 to +143	255 to -143 and 143 to 255

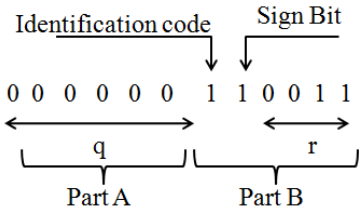

Figure. 12 Composition of code

3.5.3. Hardware architecture

In this, IEGRE codes are divided into two types with four levels as shown in Fig. 12. The code of IEGREC encoder is consisting of a sign bit "1", symbol bit "1", q bits "0", and a P bits of r. The code length is sum of M, 2 (a sign bit "1", a symbol bit "1") and q. Part A contains sign bit and reminder bits. Part B contains the quotient q and identification code, respectively.

Consequently, each encoded pixel code length is change from other encoded pixel. Theoretically, the minimum code length is 3 and maximum code length is 15 bits, respectively. In hardware environment, the size of the register is fixed and cannot change randomly. The hardware architecture of IEGREC approach is presented in Fig. 13.

Initially, the code length is generated by using addition of q, constant 2 and M shifted value. The IGRE code is generated by the multistage additions. Initially, constant 2 is added with the sign bit and outcome is shifted twice. Then, the shifted value is added with the reminder r and generates the IGRE coder, respectively. The EGRE code is generated by using multiplexer, shifter, and adder. Initially, the P is applied as the selection line to multiplexer, so P length of r bits are selected. Then, the r bits are shifted twice, and resultant is added with the value

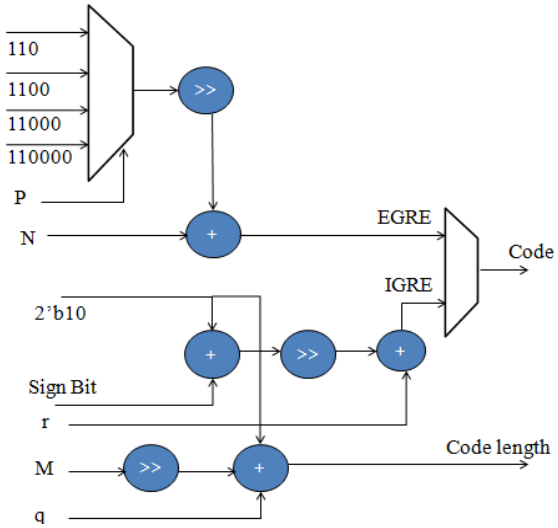

Figure. 13 Proposed hardware architecture of the IEGREC approach

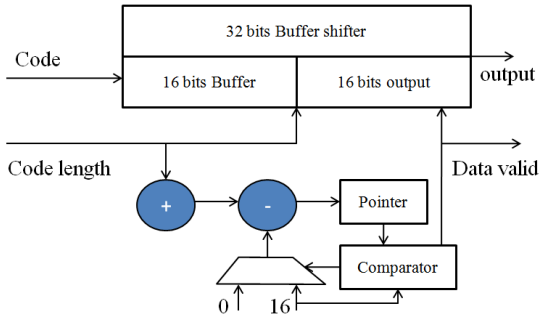

Figure. 14 Architecture of the coding packer

N and generates the outcome as EGRE code. Finally, EGRE and IGRE codes are applied as input to the 2to1 multiplexer and selected based on *M* value.

3.6 Coding packer

The coding packer is the important block in the compression operation. Each pixel has its own residual *N* value with a dissimilar length. The packer maintains the IEGREC code length as 16-bit. Fig. 14 presents the detailed architecture of packer module. The code string is applied as input to the 32-bit buffer shift register.

It stores the present the present value into one 16 bits buffer and generates the shifted output in another 16-bits register. The code length is used to control the shifting operations such as used as shift control signal, thus the input and output shifting values changed based on the code length. Here, 2to1 multiplexer is used to select the input 0 and 16, and then the selected value is subtracted from the code length. Then, the pointer is used to select the number of ones and maximum numbers of data values are selected. The pointer output is compared with the constant 16 and generates the outcome as valid control signal. If the pointer value matches with 16, then the Data valid becomes 1 else Data value becomes 0. Same data valid signal is applied as selection input to 2to1 multiplexer. If data valid is 1, then the multiplexer selects 16 else multiplexer selects zero. If data valid is 1, then the output buffered data is considered as the final output.

4. Results and discussion

This section describes the results and discussion on proposed hardware architecture, which is tested using Xilinx ISE software. In addition, image quality and compression analysis are calculated using MATLAB environment. Initially, the test image is read using MATLAB tool and corresponding pixels-based text files are fed as input to Xilinx environment, then the proposed hardware environment performed its compression operation and results in the compressed bit-stream. This obtained bit-stream is again read using MATLAB and results in production of reconstructed image. All the architectures discussed in above sections are developed using Verilog programming language, then both simulation and synthesis are performed using Xilinx ISE. For quality evaluation, the area, and power metrics are observed. In addition, various image quality metrics like PSNR, CR, and FOM also calculated to disclose the effectiveness of proposed colour image compressor using QCA-IEGREC. Generally, the CR and PSNR are inversely eruptional to each other, as CR increases; the number of pixels reduces, and results in the reduction of PSNR, respectively. Using the FOM, the relationship between image compression and hardware cost can be maintained. Thus, FOM depends on the obtained values of PSNR, CR and gate count as formulated below.

$$FOM = \frac{PSNR \times CR}{Gate\ Count} \tag{5}$$

$$PSNR = 10\log_{10}\left(\frac{255^2}{MSE}\right) \tag{6}$$

$$CR = (1-p).\left(b + \left(1 - p^{2^b}\right)^{-1}\right) \tag{7}$$

Here, b is the number of output bits presented in IEGREC code.

For performing the various experiments, the real-time Kodak dataset is considered for the testing of proposed color image compressor using QCA-IEGREC. The simulation results are presented for the test images Lena, peppers, and Airplane. Fig. 17(a) presents the input test images, the compressed

output images of fuzzy-BTC with GR [27] is shown in Fig. 17(b) and the compressed output of proposed QCA-IEGREC is shown in Fig. 17(c) respectively. By comparing that, the proposed QCA-IEGREC gives the enhanced image quality with highest CR, as compared to conventional fuzzy-BTC with GR [27] method.

Table 5 demonstrates the obtained values of PSNR, and CR with size of block for existing JPEG, fixed-size BTC, GR, and proposed QCA-IEGREC for the test image Lena, where LOCO-I + JPEG-LS [11] and fixed-size BTC [26] methods utilize the fixed block size such as 4×4, and 8×8, respectively and results in reduced values of PSNR while the block size increases, and there by enhancement in the value of CR. However, in practical scenarios, the block-size cannot be increased more than 16×16, as the computational complexity gets increased with the size of block. Thus, to overcome these drawbacks, the proposed QCA-IEGREC method is developed with the variable block size. All the resultant blocks in the sub-image consisting of the size as 4×4 to 8×8 combinations only. Due to probability-based selection of various block size, the proposed QCA-IEGREC method gives the better PSNR with improved CR compared to the LOCO-I + JPEG-LS [11], BTC-DHT [26], and fuzzy-BTC with GR [27], respectively.

Figure. 17 (a) Original images, (b) fuzzy-BTC with GR results [27], and (c) Proposed QCA-IEGREC results

Table 5. Comparison of obtained PSNR, and CR with size of block using existing LOCO-I + JPEG-LS, fixed-size BTC, fuzzy-BTC with GR, and proposed QCA-IEGREC for Lena

Method	Block size	PSNR	CR
LOCO-I + JPEG-LS [11]	4×4	36.74	5.2
	8×8	34.25	19.2
BTC-DHT [26]	4×4	31.66	7.1
	8×8	28.89	14.2
	16×16	24.41	20.6
Fuzzy-BTC with GR [27]	Variable block	29.0	14.58
Proposed QCA-IEGREC	Variable block	18.78	29.30

Table 6 demonstrates the comparison of hardware utilization and compression metrics for proposed QCA-IEGREC method in comparison with the various existing approaches such as JPEG [7], pipelined JPEG [8], LOCO-I [12], JPEG-LS [14], BTC-DHT [26], and fuzzy-BTC with GR [27], respectively. Due to the presence of transformation techniques like DCT, and DWT, JPEG [7], pipelined JPEG [8], LOCO-I [12], and JPEG-LS [14] techniques utilized higher gate count, which results in more computational complexity. In addition, the size of block is fixed in these techniques and even in BTC-DHT [26], while the architectures such as fuzzy-BTC with GR [27], JPEG-LS with IGR [28]

Table 6. Comparison of hardware utilization and compression metrics using existing and proposed QCA-IEGREC methods for Lena

Algorithm	JEPG [7]	Pipelined JPEG [8]	LOCO-I [12]	JPEG-LS [14]	BTC-DHT [26]	Fuzzy-BTC with GR [27]	JPEG-LS with IGR [28]	Proposed QCA-IEGREC
Block size	Fixed	Fixed	Fixed	Fixed	Fixed	Variable	Variable	Variable
PSNR	35.53	35.53	45.53	45.53	31.02	28.33	36.3627	**18.78**
CR	12.41	12.41	1.94	1.94	7.14	14.58	16.58	**29.30**
Gate count (RAM excluded)	33.1 k	56.6 k	36 k	90 k	8.1 k	6.4 k	4.8k	**3.7k**
Power (mW)	310	260	N/A	8.2	2.91	3.11	4.5	**1.02**
Memory	1 frame	1 frame	1 frame	1 frame	1 line	1 line	1 line	**1 line**
FOM	13.31	7.79	2.45	0.98	27.3	64.54	125.59	**152.3**

considered variable size of block. However, due to the random selection of block size, the hardware resource utilization is bit high in terms of gate count, power consumption, and memory as demonstrated in Table 6, where the proposed QCA-IEGREC obtained effective hardware resource utilization with lesser gate count, and lower power consumption with improved performance of CR and FOM.

5. Conclusions

This article proposed efficient hardware architecture of lossless colour image compression mechanism using CFA for wireless camera networks with QCA technology to mitigate the area, and power as compared to standard CMOS technology. In addition, IEGREC is proposed to reduce the memory requirement, computational complexity in terms of gate count and eliminated the requirement of context module to lessen the hardware resource utilization. Further, adaptive GR parameter prediction and control module is employed for maintaining the pixel connectivity and higher CR. Finally, packer module is used to maintain the flow of output compressed bit stream. The extensive simulation results disclosed that proposed QCA-IEGREC method performed superior as compared to state-of-art approaches with less area i.e., 3700 gate count, and reduced power about 1.02 mW. It also outperformed the existing approaches with image quality metrics such as 18.78dB of PSNR, 152.3 of FOM, and 29.3 of CR. This work can be extended to implement the joint blur detection and compression operations using IEGREC coding. The blur detection can prevent the attacks generated in the compression process.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions (Mandatory)

“Conceptualization, Mahesh Boddu; methodology, Mahesh Boddu; software, Mahesh Boddu; validation, Mahesh Boddu; formal analysis, Mahesh Boddu; investigation, Mahesh Boddu; writing—original draft preparation, Mahesh Boddu; writing—review and editing, Mahesh Boddu, Soumitra Kumar Mandal;

References

[1] S. Chen, T. Liu, C. Shen, and M. Tuan, “VLSI implementation of a cost-efficient near-lossless CFA image compressor for wireless capsule endoscopy”, *IEEE Access*, Vol. 4, pp. 10235-10245, 2016.

[2] S. T. Mrudula, K. E. S. Murthy, and M. N. G. Prasad, “M-ABRC (Adaptive Binary Range Coder) using Virtual Sliding Window technique and its VLSI implementation”, *Microprocessors and Microsystems*, Vol. 71, 102901, 2019.

[3] M. S. A. Ani, “Hardware Implementation of a Real Time Image Compression”, *Architecture*, Vol. 28, No. 29, 2017.

[4] W. Pennebakerand and J. Mitchell, “JPEG still image data compression standard”, 1992.

[5] A. M. Butt and R. A. Sattar, “On image compression using curve fitting”, *Master Thesis, School of Computing Blekinge Institute of Technology*, 2010. [Online]. Available: http://www.divaportal.org/smash/get/diva2:830444/FULLTEXT01.pdf. Accessed: Oct. 9, 2016.

[6] R. K. Senapati, “Development of novel image compression Algorithms for portable multimedia applications”, *Department of Electronics and Communication Engineering*, 2012. [Online]. Available: http://ethesis.nitrkl.ac.in/5485/. Accessed: Oct. 9, 2016.

[7] C. J. Lian, L. G. Chen, H. C. Chang, and Y. C. Chang, "Design and implementation of JPEG encoder IP core", In: *Proc. of the ASP-DAC 2001. Asia and South Pacific Design Automation Conference 2001*, 2001.

[8] Z. Qihui, C. Jianghua, Z. Shaohui, and M. Nan, "A VLSI implementation of pipelined JPEG encoder for grayscale images", *2009 International Symposium on Signals, Circuits and Systems*, 2009.

[9] A. Skodras, C. Christopoulos, and T. Ebrahimi, "The JPEG 2000 still image compression standard", *IEEE Signal Processing Magazine*, Vol. 18, No. 5, pp. 36-58, 2001.

[10] F. Liang, J. Xing, and X. Li, "A rate control algorithm in JPEG2000 based on code content prediction", *TELKOMNIKA Indonesian Journal of Electrical Engineering*, Vol. 11, No. 11, 2013.

[11] M. J. Weinberger, G. Seroussi, and G. Sapiro, "The LOCO-I lossless image compression algorithm: Principles and standardization into JPEGLS", *IEEE Transactions on Image Processing*, Vol. 9, No. 8, pp. 1309-1324, 2000.

[12] P. Merlino and A. Abramo, "A fully pipelined architecture for the LOCO-I compression algorithm", *IEEE Transactions on Very Large-Scale Integration (VLSI) Systems*, Vol. 17, No. 7, pp. 967-971, 2009.

[13] X. Chen, "A wireless capsule Endoscope system with low-power controlling and processing ASIC", *IEEE Transactions on Biomedical Circuits and Systems*, Vol. 3, No. 1, pp. 11-22, 2009.

[14] X. Xie, G. Li, X. Chen, X. Li, and Z. Wang, "A low-power digital IC design inside the wireless endoscopic capsule", *IEEE Journal of Solid-State Circuits*, Vol. 41, No. 11, pp. 2390-2400, 2006.

[15] R. Krishnaswamy, "VLSI based orthogonal diagonal cross hair search (ODCHS) algorithm implementation for efficient image compression", *Microprocessors and Microsystems*, Vol. 76, 103114, 2020.

[16] P. Nagaraj, "VLSI Implementation of Image Compression using TSA Optimized Discrete Wavelet Transform Techniques", In: *Proc. of 2020 Third International Conference on Smart Systems and Inventive Technology*, 2020.

[17] L. K. Mukkara and K. V. Ramanaiah, "A simple novel floating point matrix multiplier VLSI architecture for digital image compression applications", In: *Proc. of 2nd International Conf. on Inventive Communication and Computational Technologies*, 2018.

[18] G. Kiranmayi and S. Tadisetty, "A novel ortho normalized multi-stage discrete fast Stockwell transform based memory-aware high-speed VLSI implementation for image compression", *Multimedia Tools and Applications*, Vol. 78, No. 13, pp. 17673-17699, 2019.

[19] V. A. Coutinho, R. J. Contra, F. M. Bayer, P. A. M. Oliveira, R. S. Oliveira, and A. Madanayake, "Pruned discrete Tchebichef transform approximation for image compression", *Circuits, Systems, and Signal Processing*, Vol. 37, No. 10, pp. 4363-4383, 2018.

[20] K. Thilagavathi and S. Sivanantham. "Two-stage low power test data compression for digital VLSI circuits", *Computers and Electrical Engineering*, Vol. 71, pp. 309-320, 2018.

[21] J. H. Hsieh, M. J. Shih, and X. H. Huang, "Algorithm and VLSI architecture design of low-power SPIHT decoder for mHealth applications", *IEEE Transactions on Biomedical Circuits and Systems*, Vol. 12, No. 6, pp. 1450-1457, 2018.

[22] D. R. Premachand and U. Eranna, "Sector-selective hybrid scheme facilitating hardware supportability over image compression", In: *Proc. of Silhavy R. (eds) Intelligent Algorithms in Software Engineering, CSOC 2020, Advances in Intelligent Systems and Computing*, Vol. 1224, 2020.

[23] M. A. Alam, F. Ahsan, A. Soobhee, M. Subhani, F. M. F. Hossain, M. S. Islam, and K. N. Ruma, "Faster Image Compression Technique Based on LZW Algorithm Using GPU Parallel Processing", In: *Proc. of Joint 7th International Conference on Informatics, Electronics and Vision and 2nd International Conference on Imaging, Vision and Pattern Recognition*, pp. 272-275, 2018.

[24] M. Safieh and J. Freudenberger, "Efficient VLSI architecture for the parallel dictionary LZW data compression algorithm", *IET Circuits, Devices and Systems*, Vol. 13, No. 5, pp. 576-583, 2019.

[25] H. Kasban and S. Hashima, "Adaptive radiographic image compression technique using hierarchical vector quantization and Huffman encoding", *Journal of Ambient Intelligence and Humanized Computing*, Vol. 10, No. 7, pp. 2855-2867, 2019.

[26] S. L. Chen, J. Nie, T. L. Lin, R. L. Chung, C. H. Hsia, T. Y. Liu, S. Y. Lin, and H. X. Wu, "VLSI implementation of an ultra-low-cost and

low power image compressor for wireless camera networks", *Journal of Real Time Image Processing*, Vol. 14, pp. 803-812, 2018.

[27] S. L. Chen and G. S. Wu, "A cost and power efficient image compressor VLSI design with fuzzy decision and block partition for wireless sensor networks", *IEEE Sensors Journal*, Vol. 17, No. 15, pp. 4999-5007, 2017.

[28] C. A. Chen, S. L. Chen, C. H. Lioa, and P. A. R. Abu, "Lossless CFA Image Compression Chip Design for Wireless Capsule Endoscopy", *IEEE Access*, Vol. 7, pp. 107047-107057, 2019.

[29] A. A. Rezk, A. H. Madian, A. G. Radwan, and A. M. Soliman, "On-the-Fly Parallel Processing IP-Core for Image Blur Detection, Compression, and Chaotic Encryption Based on FPGA", *IEEE Access*, Vol. 9, pp. 82726-82746, 2021.

[30] T. K. Devi, E. B. Priyanka, and P. Sakthivel, "FPGA Implementation of Balanced Biorthogonal Multiwavelet Using Direct Pipelined Mapping Method for Image Compression Applications", *Sensing and Imaging*, Vol. 22, No. 1, pp. 1-19, 2021.

[31] G. U. Bhargava and S. V. Gangadharan, "FPGA Implementation of Modified Recursive Box Filter-Based Fast Bilateral Filter for Image Denoising", *Circuits Systems and Signal Processing*, Vol. 40, pp. 1438-1457, 2021.