IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 12, 2016, accepted October 23, 2016, date of publication December 12, 2016, date of current version January 27, 2017.

Digital Object Identifier 10.1109/ACCESS.2016.2638475

VLSI Implementation of a Cost-Efficient Near-Lossless CFA Image Compressor for Wireless Capsule Endoscopy

SHIH-LUN CHEN, (Member, IEEE), TSE-YEN LIU, CHIA-WEI SHEN, AND MIN-CHUN TUAN
Department of Electronic Engineering, Chung Yuan Christian University, Taoyuan 320, Taiwan

Corresponding author: S.-L. Chen (chrischen@cycu.edu.tw)

This work was supported in part by the Ministry of Science and Technology, Taiwan, under Grant MOST-104-2221-E-033-042, Grant MOST-104-2218-E-033-009, Grant MOST-104-2622-E-033-007-CC2, Grant MOST-105-2221-E-033-059, and Grant 105-2622-E-033-001-CC2, and in part by the National Chip Implementation Center, Taiwan.

ABSTRACT In this paper, a novel near-lossless color filter array (CFA) image compression algorithm based on JPEG-LS is proposed for VLSI implementation. It consists of a pixel restoration, a prediction, a run mode, and entropy coding modules. According to the information of the previous research, a context table and row memory consumed more than 81% hardware cost in a JPEG-LS encoder design. Hence, in this paper, a novel context-free and near-lossless image compression algorithm is presented. Since removing the context model causes decreasing of the compression performance, a novel prediction, run mode, and modified Golomb–Rice coding techniques were used to improve the compression efficiency. The VLSI architecture of the proposed image compressor consists of a register bank, a pixel restoration module, a predictor, a run mode module, and an entropy encoder. A pipeline technique was used to improve the performance of this. It contains only 10.9k gate count, and the core area is 30 625 μm^2, synthesized by using a 90-nm CMOS process. Compared with the previous JPEG-LS designs, this paper reduces the gate counts by 44.1% and 41.7%, respectively, for five standard and eight endoscopy testing images in CFA format. It also improves the average PSNR values by 0.96 and 0.43 dB, respectively, for the same test images.

INDEX TERMS Color filter array, context-free, Golomb-Rice coding, JPEG-LS, near-lossless, run mode, VLSI, and wireless capsule endoscopy.

I. INTRODUCTION

Recently, wireless video capsule endoscopy has been offering an efficient way for medical doctors to examine the digestive tracts of patients with gastrointestinal diseases. The wireless video capsule endoscopy includes a CMOS image sensor, microcontroller, microodometer, RF transmitter and image compressor [2]. To reduce power dissipation, causing damage in human bodies, the U.S.A Federal Communications Commission (FCC) limited the frequency of any medical implant wireless communication system to no more than 402 $\sim$ 405 MHz. Hence, to be able to provide higher resolution of images for diagnosis, a high-quality and high-performance image compression algorithm is necessary to develop for wireless video capsule endoscopy systems.

There are many lossy image compression standards, such as JPEG, JPEG 2000 [3], Motion JPEG [4], MPEG-1/2/4 [5], MEPG-7 [6], AVC and HEVC [7]. Although these standard algorithms have benefits of high compression rate, they are unsuitable for developing wireless video capsule endoscopy systems due to their high computational complexities. Moreover, since the volume and energy of wireless capsule endoscopy are limited, the complexity and memory demand of the image compression algorithm are also limited.

For this reason, many low-complexity and high-performance image compression algorithms based on JPEG-LS [8] were proposed. Tao *et al.* [9] proposed a search-aware compression algorithm, in which a two-pass variation technique was used to improve the compression rate of images. A novel usage of regular and run modes to determine the situations in which data hiding occurs was presented in [10]. It is a high-performance near-lossless image compression algorithm. Chen and Chen [10] proposed an efficient algorithm with high compression rate and PSNR value by using a low-pass filter. To be able to increase the compression rate more, a high-performance algorithm composed of JPEG-LS and an inter-frame coding with a

motion estimation technique was proposed in [11]. The JPEG-LS-based algorithms mentioned above are high performance and high-compression rate. However, it is not easy to realize these algorithms by VLSI technique due to the high complexity and high memory requirement.

For this reason, several hardware-oriented low-complexity and high-performance image compression algorithms based on JPEG-LS were proposed. Turcza and Duplaga [12] proposed a hardware-efficient and low-power system for wireless endoscopy. It included an entropy encoder based on an adaptive Golomb-Rice algorithm and a forward error correction encoder for protecting transmission error. Xie *et al.* [13] realized a low-power VLSI architecture including a three-stage clock management design. In [14], a low-complexity JPEG-LS algorithm was implemented by FPGA for real-time applications. A high-performance, fully pipelined architecture of LOCO-I algorithm was realized in [1]. Although the performance in this study was improved greatly by pipelined architecture, the hardware cost and memory demand were not efficient enough. A complete wireless capsule endoscope system was proposed with a low-power controlling ASIC design in [15]. It used low-complexity filter algorithms to process the color filter array (CFA) image and improve the accuracy of prediction. This design had the benefits of low cost, high quality, and high compression rates.

In this paper, a novel low-complexity and high compression rate near-lossless image compression algorithm is proposed for VLSI implementation. It decreases the complexity by reducing the context and merging the run mode with the regular mode. Besides, a novel prediction, a run mode, and entropy coding techniques were created to improve the compression rates. The proposed low-complexity and high compression rate near-lossless image compression algorithm was realized with a four-stage pipeline VLSI architecture. It achieved compressing 30 frames per second with a high-definition (HD) 1920 × 1080 resolution of CFA images in real time for wireless video capsule endoscopic applications.

This paper is organized as follows. In Section II, the near-lossless image compression algorithm is presented. Section III illustrates pipelined VLSI architecture in this study. In Section IV, the simulation results and chip implementation are proposed. Finally, a brief conclusion is presented in Section V.

II. NEAR-LOSSLESS IMAGE COMPRESSION ALGORITHM

A traditional JPEG-LS [15] algorithm consists of five main components: a context, a prediction, a regular, a run mode, and an entropy coding models. The context model is used to predict the value of the current pixel according to the values of adjacent pixels. Moreover, it is also used to determine the pixels to enter the regular or run mode models. By using the run mode model, the number of pixels which have the same value will be counted before the entropy-coding process. Hence, the compression rate can be obviously improved, especially when many similar or continuous parts are within an image. Besides, the pixels which enter the regular model will have to compute the connection between the current pixel and neighboring pixels by using the prediction model. The entropy coding model in JPEG-LS is Golomb-Rice coding, which is used to encode the values of prediction errors efficiently.

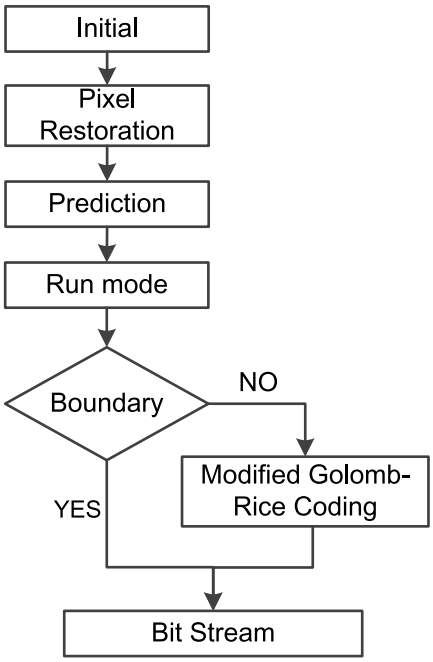

FIGURE 1. Flow chart of the proposed near-lossless image compression algorithm.

According to the experience in study [1], a context model consumed the most hardware cost in a JPEG-LS image encoder design. Hence, a novel context-free image compression algorithm is proposed in this paper. Fig. 1 shows the flow chart of the proposed near-lossless image compression algorithm based on JPEG-LS. It includes a pixel restoration, a prediction, a run mode, a modified Golomb-Rice coding, and a bitstream models. The details of each part are described as following.

A. PIXEL RESTORATION

Generally, each pixel in a color image is composed of three colors: red, green, and blue (RGB). However, a CCD or CMOS image sensor captures images by a color filter array (CFA) technique, in which each pixel in a captured image contains only one color, as shown in the left picture of Fig. 2. Hence, the number of pixels in a CFA image is only one-third of a general full RGB color image. It is helpful to decrease the bit rate for image transmission and storage. However, the dependence between two contiguous pixels is missed due to the fact that they are captured in different colors. For this reason, a pixel restoration module was developed to arrange the pixels in CFA image to a color continuous format, in which the image in CFA format was restored to a red, blue and green line buffers as shown in Fig. 2. Each pixel in the CFA image is restored to red, blue

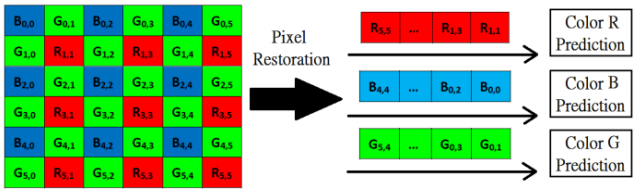

FIGURE 2. Restoring image from the CFA format to RGB line buffers.

and green line buffers with the lengths of 1/4, 1/4 and 1/2 width of the CFA image, respectively. Hence, it only needs a line buffer as long as the width of the CFA image to predict the three colors.

B. PREDICTION

In a traditional JPEG-LS algorithm, if a pixel is selected to the run mode by the context module, it would not pass through the median edge detector model. This process decreases the compression rate because the median edge detector is useless for the pixels that are selected by the run mode. For this reason, a novel prediction methodology is proposed in this paper. As shown in Fig. 1, the prediction model was moved to the front of the run mode model, which avoids the compression process against wasting too many bits when the pixels enter the run mode without prediction.

Moreover, the median edge detector model in JPEG-LS standard uses the surrounding pixels to predict the current pixel. If the value of the current pixel has high correlation with the values of the surrounding pixels, the difference between the values of the current and predicted pixels would be closed to zero. It can improve the compression rate cooperating with an entropy coding model. However, if the correlation of the surrounding pixels is very small, the compression rate would be obviously decreased. The situation of small correlation occurs frequently, especially when the surrounding pixels located in a region have huge variation.

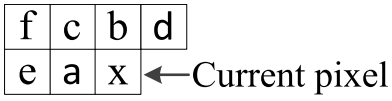

FIGURE 3. Relationship between current and neighboring restored color continuous pixels.

Hence, in order to improve the performance of the proposed compression algorithm, a novel prediction methodology is proposed in this paper. In the proposed prediction model, all red and blue pixels pass the median edge detection module as shown in equation (1). Also, the relationship between the current and neighboring pixels is shown in Fig. 3. If the value of red or blue neighboring pixel c is much greater or smaller than the values of pixels a and b, the predicted value of the current red or blue pixel x ($xmed_{(R,B)}$) will be affected more by the value of neighboring pixel d. Otherwise, the value of the neighboring pixel d will influence the predicted value of current x less with a smaller weighting coefficient. Since there are twice as many green pixels as red or blue pixels, the weighting coefficients of pixels neighboring a, b, and d for the green color are closer to the average than those of the red and blue colors. Hence the predicted value of the current green pixel x ($xmed_{(G)}$) would be obtained by an average filter as shown in equation (2).

$$\begin{cases} xmed_{(R,B)} = \dfrac{2^{*}a + b + d}{4}, \\ \qquad c \geqq max\,(a, b) \ \text{ or } c < min(a, b) \\ xmed_{(R,B)} = \dfrac{4^{*}a + 3^{*}b + d}{8}, \\ \qquad \text{others} \end{cases} \tag{1}$$

$$xmed_{(G)} = \frac{3^{*}a + 3^{*}b + 2^{*}d}{8} \tag{2}$$

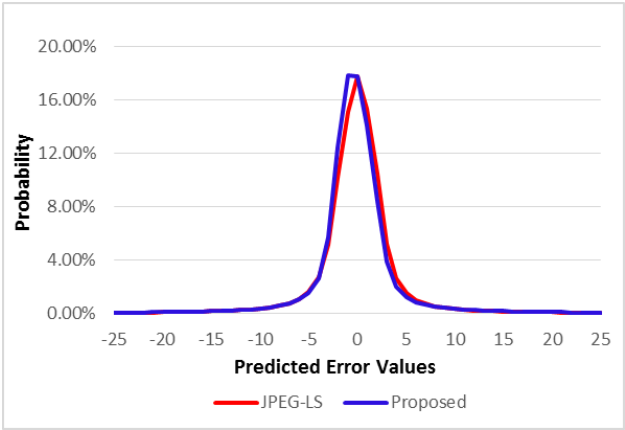

FIGURE 4. Probability distributions by using JPEG-LS and the proposed prediction methods in CFA format by video_1 in the Hamlyn endoscopic video dataset [16].

In order to compare the performance of prediction methods in JPEG-LS standards and the proposed video_1 in the Hamlyn Centre Laparoscopic / Endoscopic Video Dataset [16], which includes 811 frames of 640 × 480 in CFA format, was used as a testing dataset. Fig. 4 shows probability distributions by using prediction methods of JPEG-LS and proposed CFA format for video_1 in the Hamlyn endoscopic video dataset [16]. The probability distribution of the proposed novel prediction model is probably suitable for lossy compression methods and provides lower mean squared error (MSE) values than JPEG-LS. In addition, Table 1 compares the compression rates of the JPEG-LS and the proposed prediction methods, in both cases using the same modified Golomb-Rice coding algorithm, on 5 standard images, 8 endoscopy images [19], 101 endoscopy frames [20] and 811 endoscopy frames [16]. The results show that the proposed novel prediction method had better compression performance than the JPEG-LS prediction method in this case.

C. RUN MODE MODULE

The run mode module is constructed by a run length table and an encoder. An important parameter, "*NEAR,*" is used to set the quality and compression rate in the run mode model.

TABLE 1. Comparisons of compression rates (bit per pixel, BPP) for JPEG-LS and the proposed prediction methods with the same modified Golomb-Rice coding algorithm in CFA format by seven datasets.

	Frames	Frame Size	JPEG-LS Prediction	Proposed Prediction
Standard Images	5	512×512	4.128	4.014
Endoscopy Images [19]	8	352×240	3.536	3.467
Oesophagus [20]	30	1200×945 1200×900	2.554	2.405
Duodenum [20]	26	1280×1008 1200×945 891×704	3.087	2.986
Ileum Terminal [20]	25	1280×1008 1200×945	3.352	3.252
Caecum [20]	20	1280×1008 1200×945	3.467	3.369
Video1 [16]	811	640×480	1.929	1.912

Equation (4) shows the definition of *NEAR* in the proposed algorithm, in which there are three conditions to determine the value of *NEAR* according to the relationship of predicted error value (*Errval*) and *NEAR*. The value of *Errval* can be obtained according to the value of *NEAR* which is determined by the user. The value of *Errval* can be obtained by

$$Errval \begin{cases} \dfrac{Errval + NEAR}{2{*}NEAR} & if\ Errval \geqq NEAR + 1 \\ -\dfrac{NEAR - Errval}{2{*}NEAR} & else\ if\ Errval \leqq -NEAR - 1 \\ Run\ mode\ Processing & else \end{cases} \tag{3}$$

where *Errval* is a predicted value and the run mode processing will be introduced later. After determining the value of *NEAR*, a bitstream will be produced. For example, if the parameter *NEAR* is equal to 2, the run mode module will set the values of $-2, -1, 0, 1$, and 2 as 0, and then count the occurring number of continuous 0s. In order to promote the performance of the run mode model, a novel J table is created to encode the counting number of the run mode model.

$$J = \{0, 0, 0, 0, 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 4, 4, 5, 5, 6, 6\}; \tag{4}$$

Equation (4) shows the proposed novel J table. For example, if the counting number is equal to 8, the counting number would be encoded as "11111100" by a coding process. First, the original number 8 is compared with 2^0, in which 0 is the first element in the J table. Since 8 is greater than 2^0, the first bit is coded as "1" and the counting number is modified by $8 - 2^0 = 7$. The process will continuously encode the bit pattern as "111111" until the final counting number 0 is less than 2^1, in which 1 is the seventh element in the J table. Finally, a boundary information bit "0" is added to show the end of looking up J table and then add the remainder to the bit pattern. So, the final bit pattern to repress the counting number 8 is "11111100." In this case, the run mode module only uses 8 bits to encode eight values of pixels. Otherwise, if these eight pixels were encoded by a traditional Golomb-Rice algorithm, it is necessary to use more than 24 bits to express the values of these eight pixels. Hence, the proposed run mode module is very efficient to improve the compression rate, especially when the counting number is huge.

D. MODIFIED Golomb-Rice CODING

The principle of Golomb-Rice algorithms [17], [18] is that by using an adaptive technique, a coding parameter k is adjusted according to the previous information in the context table. Since the proposed image compression algorithm is a context-free methodology, the parameter k is fixed by a constant value 2. For example, if a prediction error value is eleven, "1011" in a binary representation, first it is shifted right by $k = 2$ and then encoded as "0010." Second, the quotient 2 was encoded by the run-length coding by putting two bits of "0" in front of "1" as "001." Third, a sign bit "0" is added to the bitstream to represent the value is positive. Finally, the remainder "11" is added to the bitstream and the value eleven can be encoded by the Golomb-Rice algorithm as "001011".

TABLE 2. Difference between Golomb-Rice and modified Golomb-Rice codes.

Number	Golomb-Rice Codes	Modified Golomb-Rice Codes
0	100	1000
Extend Code	X	1100
-1	101	1101
1	110	1001
-2	111	1110
2	0100	1010
-3	0101	1111
3	0110	1011
-4	0111	01100
	:	
15	0000000110	0001011
-16	0000000111	0001111
16	00000000100	00001000
-17	00000000101	00001100

Table 2 lists the proposed modified Golomb-Rice coding algorithm in which the characteristic of this algorithm is each code including a sign bit. The bit after the first "1" bit from the left side is a sign bit. "1" represents positive values and "0" represents negative values. It is easy to find that when the entropy table is longer, the modified Golomb-Rice coding is more efficient than the traditional Golomb-Rice algorithm. For example, in a traditional Golomb-Rice algorithm, when the number is 15 in decimal representation, the encoded bit pattern is "0000000110," which demands 10 bits. However, by using the proposed modified Golomb-Rice coding algorithm, the encoded bit pattern is "0001011," which only demands 7 bits. Although in this case the length of the bit pattern encoded by the proposed modified Golomb-Rice algorithm is 30% less than the traditional Golomb-Rice algorithm, the performance of the variable length code (VLC) depends on the probability distribution. Hence, the performance of entropy coding

algorithms depends on the average product of VLC length and probability for the whole dataset.

Table 2 lists the difference between the Golomb-Rice code and the proposed modified Golomb-Rice code. The Golomb-Rice coding was modified in this study because the proposed near-lossless compression algorithm was developed for compressing CFA images rather than full RGB images. Since the continuity of each color in a CFA image is worse than that in a full RGB image, the probability distribution of predicted error values is more distributed in CFA images than in full RGB images. In addition, in order to improve the image restoration quality, the proposed near-lossless compression algorithm used the modified Golomb-Rice coding without quantization when the pixel is located in boundary regions. Since there is no negative zero (−0), the code ''1100'' was used to express the ''extending code'' as shown in Table 2. The short extending code ''1100'' not only improved the compression performance when the pixels were located in boundary regions but also reduced the size of the entropy buffer for hardware implementation.

TABLE 3. Comparisons of Compression Rates (Bit per Pixel, BPP) for Traditional and Modified Golomb-Rice Coding Algorithms in CFA Format for the five standard Testing Dataset.

	Traditional Golomb-Rice	Modified Golomb-Rice
Airplane	3.089	3.285
Lena	3.748	3.831
House	3.852	3.817
Peppers	3.791	3.822
Baboon	6.107	5.318
Average	4.117	4.014

In order to compare the performance of the traditional Golomb-Rice coding algorithm with the proposed novel modified Golomb-Rice coding algorithm, 5 standard images, 8 endoscopy images [19], 101 endoscopy frames [20] and 811 endoscopy frames [16] were selected as testing datasets in CFA format. First, each pixel in the testing CFA images was predicted by the proposed novel prediction methodology and obtained the values of prediction errors. Second, the values of prediction error were encoded by the traditional and modified Golomb-Rice coding algorithms. Finally, the values of bit per pixel (BPP) could be calculated for each testing image by the two different Golomb- Rice coding algorithms. Tables 3, 4 and 5 show the comparisons of compression rates for the traditional and modified Golomb- Rice coding algorithms for 5 standard images, 8 endoscopy images [19], and 101 endoscopy [20] and 811 endoscopy [16] frames in CFA format. The results show that the proposed modified Golomb-Rice coding algorithm improved the average compression rates 2.5%, 2.25%, 1.28% and 0.88% over the traditional Golomb- Rice coding algorithm for 5 standard images, 8 endoscopy images [19], 101 endoscopy frames [20] and 811 endoscopy frames [16], respectively.

TABLE 4. Comparisons of compression rates (Bit per Pixel, BPP) for traditional and modified Golomb-Rice coding algorithms in CFA format for the eight video endoscopy images dataset [20].

	Traditional Golomb-Rice	Modified Golomb-Rice
Hereditary_Telangiectasia2 080	3.608	3.574
Hereditary_Telangiectasia2 221	3.651	3.506
Hereditary_Telangiectasia2 300	3.188	3.143
Hereditary_Telangiectasia2 071	4.239	4.030
Hereditary_Telangiectasia2 298	3.344	3.319
Hereditary_Telangiectasia2 06	3.360	3.304
Hereditary_Telangiectasia2 043	3.610	3.540
Hereditary_Telangiectasia3 06	3.371	3.317
Average	3.547	3.467

TABLE 5. Comparisons of compression rates (bit per pixel, BPP) for traditional and modified Golomb-Rice coding algorithms in CFA format for the gastrolab [20] and hamlyn endoscopic video dataset [16].

	Frames	Frame Size	Traditional Golomb-Rice	Modified Golomb-Rice
Oesophagus [20]	30	1200×945 1200×900	2.406	2.405
Duodenum [20]	26	1280×1008 1200×945 891×704	3.024	2.986
Ileum Terminal [20]	25	1280×1008 1200×945	3.318	3.252
Caecum [20]	20	1280×1008 1200×945	3.420	3.369
Video1 [16]	811	640×480	1.929	1.912

TABLE 6. Three coding methodologies for entropy coding.

Bitstream	Mode A + Mode B + Mode C		
Characteristic	Continuous	End of Line	Interrupted
Encoding Module	Run Mode	Boundary Mode	Modified Golomb-Rice

E. ENTROPY CODING PROCESS

There are three entropy modes used in the proposed near-lossless image compression algorithm. It includes a run mode, a boundary, and modified Golomb-Rice coding modules as shown in Table 6. The predicted error values are encoded by the run mode module first and then encoded by a boundary or a modified Golomb-Rice coding module according to the boundary information as shown in Fig. 1. In this section, an example is used to explain the entropy coding process by using three entropy modes to compress the predicted values efficiently.

As shown in Table 6, the bitstream can be separated into three kinds of modes, Mode A, Mode B, and Mode C. Mode A is designed by the run mode module which is used to compress the predicted error values when it is located in a continuous region. The run mode can be used to decrease the bit rate very efficiently when the predicted error values are appearing in the range of −2 to 2 continuously when the parameter *NEAR* is set to 2. The concept of the run mode module is to count the times of continuous appearances of 0 in the predicted error values. For example, if the counting

number in run mode is 5, Mode A encodes the bitstream as "111101." By looking up the J table as shown in Equation (3), the counting number 5 is equal to $\{2^0 + 2^0 + 2^0 + 2^0 + 1\}$. In this case, the position coding of the J table is 5 and its code is "11110." The location 5 in the J table is 1, which means one bit remainder number is necessary to be added. According to the J table, the number of $\{2^0 + 2^0 + 2^0 + 2^0\}$ is 4 and the code includes a remainder bit. Hence, the counting number 5 can be encoded by adding a remainder bit "1" to "11110" as "111101." In another case, if the counting number in run mode is 0, Mode A is encoded by only one bit as "0."

After encoding by the run mode, there are two modes selected according to the location of pixels. For example, if the pixel is located at the end of line, the boundary mode is terminating the encoding process in the current line and then starting the entropy coding process for the next line. In contrast, if the location of pixels is not at the end of line, the modified Golomb-Rice module is used to encode the predicted error value. After coding by the modified Golomb-Rice module, the next predicted error value will be sent into the run mode module and restart the entropy encoding process again.

F. DECODING PROCESS

After receiving the encoded bitstream by the proposed near-lossless compression algorithm, it is necessary to decode the bitstream to refine each value of each pixel and then restore the refined pixels to an image in CFA format. There are five main components in the decoding process: a run mode decoder,

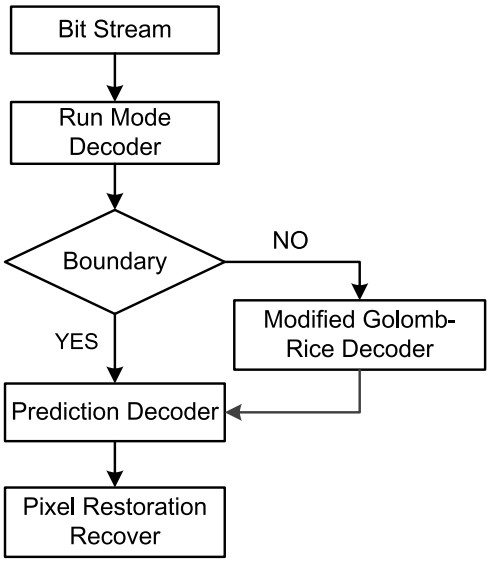

FIGURE 5. Flowchart of the proposed decoder.

a boundary module, a modified Golomb-Rice decoder, a prediction decoder, and a pixel restoration recover. Fig. 5 illustrates the flowchart of the corresponding decoder.

a) Run Mode Decoder

To decode the bitstream, the first step is decoding the run mode information. First, the bits of the bitstream are read one by one until finding the first "0" and then getting the remainder bit/bits according to the J table as shown in Equation (3). The counting number of "1" indicates the position in the J table. For example, if the bitstream is "111101," in which four "1" bits are appeared and then connected with a "0." It expresses the fourth location of the J table, which means the counting number is $\{2^0 + 2^0 + 2^0 + 2^0\}$, added a remainder 1, equal to 5. The decoded counting number expressing the number of the predicted error value "0" appeared continuously. In this case, since the decoded counting number is 5, the first five predicted error values are $\{0, 0, 0, 0, 0\}$. Hence, all the "0" values of the predicted error values can be obtained by the run mode decoder.

III. VLSI ARCHITECTURE

Based on the algorithm mentioned above, the architecture of the proposed near-lossless image encoder design is shown in Fig. 6. It is composed of four main parts: a pixel restoration module, a predictor, an entropy coder, and a barrel shifter. In addition, a register bank was added to provide four neighboring pixels—Ra, Rb, Rc, and Rd—for the predictor to calculate the predicted value of Rx. Since the register bank needs the information from two lines, it is connected with a two-line-buffer memory. In order to improve the performance, this design was realized with a four-stage pipeline architecture. Moreover, a register bank is added to provide the values of the required pixels for compression, and a finite statement machine (FSM) is used to realize a controller. The barrel shifter is used to packet the variable length codes and then output bitstream in a fixed length. The details of the four main parts are described as following.

A. PIXEL RESTORATION CIRCUIT

The pixel restoration circuit is a module designed for producing memory addresses and then reading the values of target pixels stored in the memory. It is used to read the values of the pixels in the CFA image and then construct an integrated image for prediction as shown in Fig. 2. Moreover, the pixel restoration circuit is also including a boundary detector that is designed to find the boundary information, which can avoid getting the values of the pixel in the wrong color when it locates them in the boundary regions. For example, the green color has two different arranging orders in the CFA format. If it is unable to distinguish between odd and even rows, it is impossible to produce the right address to get the right value of the pixel.

B. PREDICTOR AND RUN MODE CIRCUIT

The predictor is used to predict the value of the current pixel according to the neighboring pixels. Fig. 7 shows the architecture of the proposed predictor and the run mode circuit designs. It consists of two prediction circuits, a reconstructed pixel module and a run counter. Since the two novel prediction methodologies, as shown in equations (1) and (2), were developed to advance the compression performance, the two prediction circuits prediction 1 and prediction 2 were realized

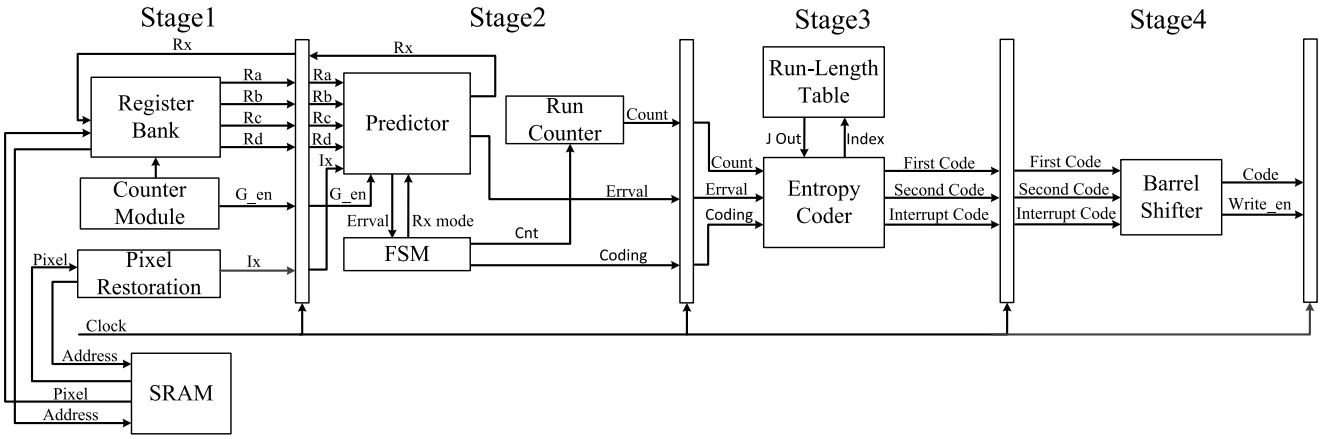

FIGURE 6. Block diagram of four-stage pipeline architecture for the proposed near-lossless image encoder design.

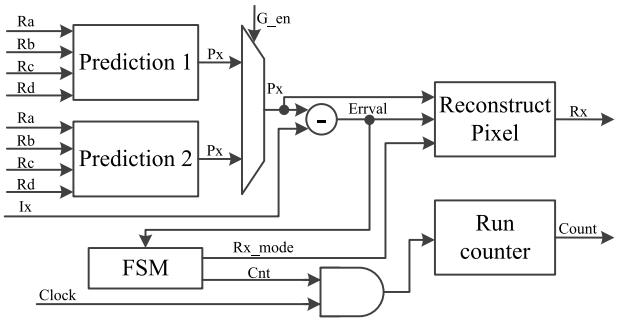

FIGURE 7. Architecture of the predictor and run mode circuit.

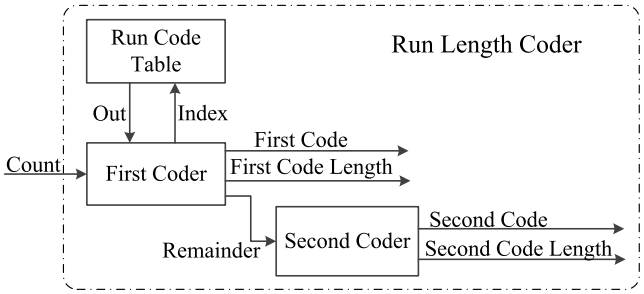

FIGURE 8. Architecture of the run length coder.

according to equations (1) and (2), which are used to predict R/B and G pixels, respectively.

The run counter is designed to count the number of *Errval*s entering the run mode module, which means that the values of *Errval*s are ranged −2 to 2. The finite state machine (FSM) produces a control signal *Rx_mode* to select one prediction error *Errval* and run count values to send to the entropy encoder. If the value of *Errval* ranges in −2 to 2, the signal *cnt* is 1 and the coding is 0. In this situation, the run counter module starts to count the number of *Errval*s in the run mode and then sends the counted value to the entropy coder. Otherwise, if the value of Errval is not ranged in −2 to 2, the signal cnt is 0 and the coding is "1". The designs of the predictor and run mode circuit can efficiently improve the compression performance for the proposed near-lossless image encoder circuit.

C. ENTROPY CODER

The entropy coding circuit is composed of a run length coder and a modified Golomb-Rice coder. The run length coder is used when the values of *Errval*s are selected by the run mode module. Fig. 8 illustrates the architecture of the run length coder, which includes a run code table, a first coder, and a second coder. When the value of count is received from the run mode circuit, the first coder encodes the count value by looking up run code table. After the first coder obtains the first code, first code length, and remainder, a second coder is used to produce the second code and second code length.

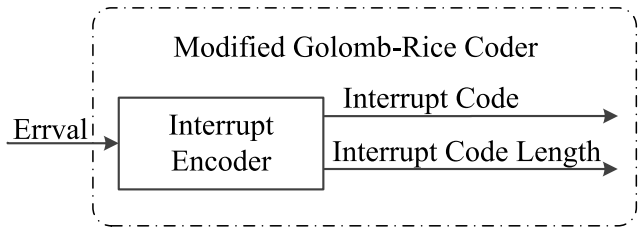

FIGURE 9. Architecture of the modified Golomb-Rice coder.

Otherwise, if the values of *Errval*s are over the range of −2 to 2, the values of *Errval*s are encoded by the modified Golomb-Rice coder. As shown in Fig. 9, the interrupted coder receives the values of *Errval*s from the predictor and then produces an interrupted code and length. Since the first, second, and interrupted codes are all variable-length codes (VLC), the lengths of the first, second, and interrupted codes are necessary to be sent to the barrel shifter with the first, second, and interrupt codes.

D. BARREL SHIFTER

After receiving the first, second, and interrupt codes and lengths from the entropy coder, the barrel shifter collects the codes according to various lengths to arrange an output bitstream with a fixed length of bits. Fig. 10 shows the

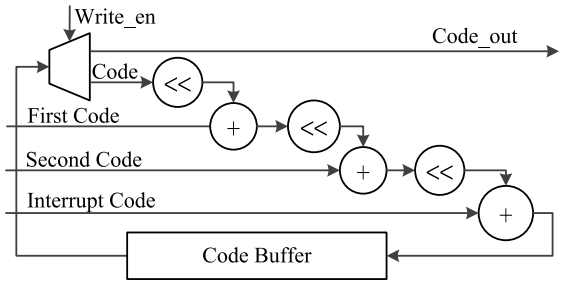

FIGURE 10. Architecture of the barrel shifter.

architecture of the barrel shifter, in which the code buffer is composed of a 40-bit register. The barrel shifter only consists of three shifters, three adders, registers, and a multiplexer. It is an efficient design to arrange the VLC into a fixed length of bit pattern.

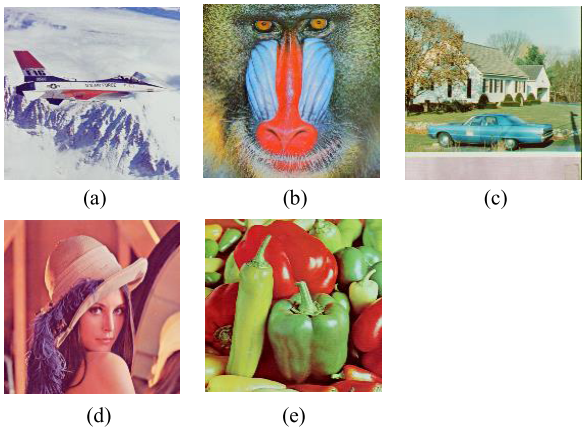

FIGURE 11. Five standard images for testing. (a) Airplane. (b) Baboon. (c) House. (d) Lenna. (f) Peppers.

IV. SIMULATION RESULTS AND CHIP IMPLEMENTATION

In order to evaluate the performance of the proposed near-lossless image compression algorithm, a MATLAB tool was used to simulate the previous near-lossless algorithms and this work. Five standard testing images of 512 × 512 pixels were selected as the first dataset as shown in Fig. 11 and eight video endoscopy images from [19] with the size of 352 × 240 pixels were selected as the second dataset as shown in Fig. 12. The original color format of the two testing sets was full RGB format. Each original image was a red, green and blue image of 512 × 512 pixels and 352 × 240 pixels in the five standard images dataset and eight endoscopy images dataset, respectively.

In order to produce golden CFA testing images, all images in these two datasets were mapped to the Bayer CFA format by a down-sampling technique from three-color images that included red, green and blue to a single CFA format image, as shown on the left side of Fig. 2. Each pixel in the golden CFA testing image was selected from one of three pixels in the red, green and blue original testing images according to the position of the target pixel in the CFA format. After this conversion, the two golden testing image sets were

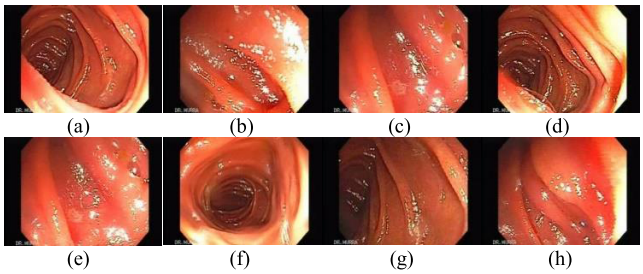

FIGURE 12. Testing video endoscopy images from Gastro Gastroenterologist Hospital [20]. (a) Image of Hereditary_Telangiectasia2 080. (b) Image of Hereditary_Telangiectasia2 221. (c) Image of Hereditary_Telangiectasia2 300. (d) Image of Hereditary_Telangiectasia2 071. (e) Image of Hereditary_Telangiectasia2 298. (f) Image of Hereditary_Telangiectasia2 06. (g) Image of Hereditary_Telangiectasia2 043. (h) Image of Hereditary_Telangiectasia3 06.

produced in the CFA format. Next, these golden CFA testing images were compressed by JPEG-LS, Filter [15], and proposed near-lossless compression algorithm to obtain the compression rates for each compression algorithm. And then each compressed bitstream was decoded and restored to CFA images. Finally,

each decoded CFA image and related golden image were used to obtain the PSNR and mean squared error (MSE) values for comparing image quality.

TABLE 7. Comparisons of compression rates (Bit per Pixel, BPP) and average PSNR (dB) for various near-lossless CFA image compression algorithms.

	Airplane	Baboon	House	Lenna	Peppers	Average	PSNR
JPEG-LS	3.578	6.606	4.146	4.366	4.105	4.560	45.468
Filter[15]	2.983	4.912	3.474	3.464	3.527	3.672	46.428
This Work	3.285	5.318	3.817	3.831	3.822	4.014	46.433

TABLE 8. Compression rates (bit per pixel, BPP), PSNR, and MSE for JPEG-LS and this work for the five standard images dataset in CFA format.

	JPEG-LS			This Work		
	BPP	PSNR	MSE	BPP	PSNR	MSE
Airplane	3.578	45.456	1.8513	3.285	46.4191	1.4831
Lenna	4.366	45.459	1.85	3.831	46.3818	1.4959
House	4.146	45.373	1.8869	3.817	46.5315	1.4452
Peppers	4.105	45.628	1.7793	3.822	46.4291	1.4797
Baboon	6.606	45.422	1.8661	5.318	46.4016	1.4891
Average	4.560	45.468	1.84672	4.0146	46.4332	1.4786

Table 7 lists the compression rates and average PSNR values for the JPEG-LS, the filter [15], and this work for the five standard CFA images testing dataset. In order to compare objectively, the parameters "*Near*" used in JPEG-LS and the proposed near-lossless compression were both set to 2. In the case of JPEG-LS, when the absolute values of prediction errors are less than or equal to "*Near*", the encoder enters the run mode, otherwise enters the regular mode [21]. In the proposed near-lossless algorithm, when the absolute values

TABLE 9. Comparisons of compression rates (bit per pixel, BPP), PSNR (dB) and MSE for JPEG-LS and this work for the eight video endoscopy dataset in CFA format.

	JPEG-LS			This Work		
	BPS	PSNR	MSE	BPS	PSNR	MSE
Hereditary_Telangiectasia2 080	3.8224	46.0308	1.6218	3.5735	46.2805	1.5312
Hereditary_Telangiectasia2 221	4.0261	46.0937	1.5985	3.5055	46.455	1.4709
Hereditary_Telangiectasia2 300	3.4436	45.8643	1.6852	3.1432	46.3766	1.4977
Hereditary_Telangiectasia2 071	4.6951	45.6361	1.7761	4.0302	46.3141	1.5194
Hereditary_Telangiectasia2 298	3.6484	46.1379	1.5823	3.3192	46.4299	1.4794
Hereditary_Telangiectasia2 06	3.4765	45.8230	1.7013	3.3041	46.1993	1.5601
Hereditary_Telangiectasia2 043	3.7819	45.9689	1.6451	3.5401	46.4358	1.4774
Hereditary_Telangiectasia3 06	3.4876	45.9347	1.6581	3.3166	46.4109	1.4859
Average	3.7977	45.9361	1.6585	3.46655	46.3627	1.5027

TABLE 10. Comparisons of compression rates (bit per pixel, BPP), PSNR, and MSE for JPEG-LS and this work for the Gastrolab [20] and Hamlyn endoscopic video datasets [16] in CFA format.

	Frames	Frame Size	JPEG-LS			This Work		
			BPS	PSNR	MSE	BPS	PSNR	MSE
Oesophagus [20]	30	1200×945 1200×900	2.465	46.095	1.598	2.405	46.234	1.549
Duodenum [20]	26	1280×1008 1200×945 891×704	3.154	46.062	1.61	2.986	46.303	1.523
Ileum Terminal [20]	25	1280×1008 1200×945	3.401	46.068	1.608	3.252	46.329	1.514
Caecum [20]	20	1280×1008 1200×945	3.480	46.037	1.62	3.369	46.332	1.513
Video1 [16]	811	640×480	2.017	46.111	1.592	1.912	46.398	1.49

of prediction errors are less than or equal to "*Near*", the encoder enters the run mode, otherwise enters the the modified Golomb-Rice coding module.

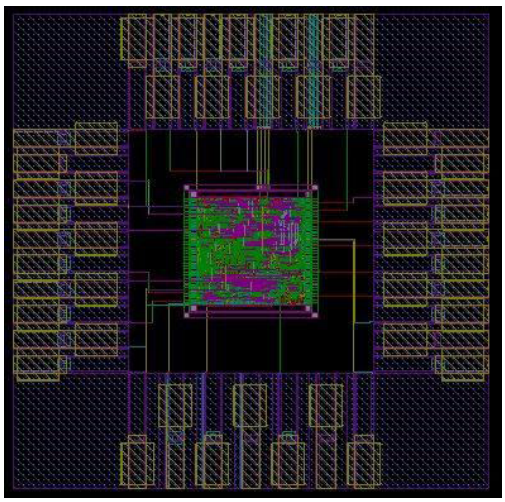

FIGURE 13. Chip photomicrograph by 90-nm CMOS process.

Compared with the high-performance filter algorithm in [15], although the average compression rate is a little less than the previous work [15], the average PSNR value in this work is 0.08 dB better than the previous work [15]. The BPP, PSNR and MSE values for each image in CFA format for the 5 standard images, 8 endoscopy images [19], 101 endoscopy frames [20] and 811 endoscopy frames [16] datasets by JPEG-LS and this work are listed in Table 8, 9 and 10. Compared with the JPEG-LS, this work had better compression rates, higher PSNR and lower MSE values in these four datasets in CFA format.

TABLE 11. Specifications of the proposed near-lossless image compressor design.

Process	TSMC 0.18-μm	TSMC 90-nm
Chip Area	370×372 μm^2	175×175 μm^2
Gate Count	12.4 k	10.9 k
Max Frequency	80 MHz	240 MHz
Supply Voltage	1.8 V	1 V
Power Consumption	4.354 mW @ 80 MHz	4.704 mW @ 240 MHz

The proposed near-lossless image compression algorithm was realized by VLSI technique. The VLSI architecture of this work was implemented by using the hardware description language Verilog and synthesized by the electronic design automation (EDA) tool Design Vision. This design was also compiled by Quartus II and evaluated using a FPGA emulation board with Altera FPGA EP2C70F896C6 core. Table 11 lists the specifications in this design, which were synthesized by a 0.18-μm and 90-nm TSMC CMOS processes. According to the synthesized results, this work contains 10.9-k gate count, and the chip area is 30,625 um^2 synthesized by a 90-nm TSMC CMOS process. It consumes 4.704 mW when operating at 240 MHz, operation frequency with 1 V supply voltage. Fig. 13 shows the chip photomicrograph by a 90-nm TSMC CMOS process.

Table 12 lists the process, operating frequency, image resolution, gate count, and memory requirement in the

TABLE 12. Comparisons of this work with previous low-complexity designs.

	JPEG-LS [15]	JPEG-LS [14]	JPEG-LS [1]	Filter [16]	This Work	
					A	B
Process	N/A	0.18-μm	FPGA	0.18-μm	0.18-μm	90-nm
Operating Frequency	66 M	40 M	21 M	24 M	80 M	240 M
Power Consumption	59.07 mW	N/A	N/A	1.3 mW	3.1 mW @80-MHz, 0.9 mW @ 25-MHz	4.7 mW @240-MHz, 0.51 mW @ 25-MHz
Resolution	640	640	N/A	640	640	640
Gate Count	49.4 k	28.1 k	36 k	19.5 k	12.4 k	10.9 k
Memory (bits)	32.7 k	20.8 k	1.01 M	17.5 k	10.2 k	10.2 k
Normalized Area	4.53	2.58	3.3	1.79	1.14	1

Note: The normalized core area is normalized by gate counts where gate counts denote the NAND-equivalent gate counts

previous works and this work. Compared with the previous low-complexity designs, this work reduces gate counts by at least 77.9%, 61.2%, 69.7%, and 44.1% more than [1], [13], and [14], respectively. The size of each line buffer is 640-byte, which provides 640 pixels for each line. Hence, the size of the images are up to 640×480. The memory requirement in this work is only 10.2 k bits, which is successful saving 68.8%, 51%, 99%, and 41.7% of memory requirements in [14], [13], [1], and [15], respectively. Since this work is a fully pipelined and context-free design, the throughput of this design up to 80 and 240 mega pixels per second by using a 0.18-μm and 90-nm CMOS processes, respectively.

The throughput in this work achieves 240 mega pixel per second compression of CFA images in real time for video endoscopy applications. According to Table 12, the proposed design reduces gate count by at least 44.1% and memory requirement by 41.7% more than previous low-complexity lossless encoder designs. To summarize, a low-cost, low-memory-requirement, high-performance, and high-quality near-lossless CFA image compressor VLSI design was proposed in this study for wireless video capsule endoscopy.

V. CONCLUSION

In this paper, a novel context-free and near-lossless image compression algorithm is proposed for VLSI implementation. The compression performance of the proposed algorithm can be promoted obviously by a novel prediction, run mode, and modified Golomb-Rice coding techniques. After comparing with previous designs, the VLSI architecture of this work owns the benefits of low cost, low memory demand, high performance, and high quality. It is very suitable for developing wireless video capsule endoscopy systems.

REFERENCES

[1] P. Merlino and A. Abramo, "A fully pipelined architecture for the LOCO-I compression algorithm," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 17, no. 7, pp. 967–971, Jul. 2009.

[2] A. Karargyirs and A. Koulaouzidis, "OdoCapsule: Next-generation wireless capsule endoscopy with accurate lesion localization and video stabilization capabilities," *IEEE Trans. Biomed. Eng.*, vol. 62, no. 1, pp. 352–360, Jan. 2015.

[3] K. Sarawadekar and S. Banerjee, "An efficient pass-parallel architecture for embedded block coder in JPEG 2000," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 6, pp. 825–836, Jun. 2011.

[4] D. T. Vo and T. Q. Nguyen, "Quality enhancement for motion JPEG using temporal redundancies," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 5, pp. 609–619, May 2008.

[5] C.-P. Fan, C.-W. Chang, and S.-J. Hsu, "Cost-effective hardware-sharing design of fast algorithm based multiple forward and inverse transforms for H.264/AVC, MPEG-1/2/4, AVS, and VC-1 video encoding and decoding applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 4, pp. 714–720, Apr. 2014.

[6] M. J. Coimbra and J. O. S. Cunha, "MPEG-7 visual descriptors—Contributions for automated feature extraction in capsule endoscopy," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 5, pp. 628–637, May 2006.

[7] J. Vanne, M. Viitanen, T. D. Hamalainen, and A. Hallapuro, "Comparative rate-distortion-complexity analysis of HEVC and AVC video codecs," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1885–1898, Dec. 2012.

[8] H. Niu, Y. Shang, X. Yang, D. Xu, B. Han, and C. Chen, "Design and research on the JPEG-LS image compression algorithm," in *Proc. IEEE 2nd Int. Conf. Commun.*, Jul. 2010, pp. 232–234.

[9] T. Tao, A. Mukherjee, and R. V. Satya, "A search-aware JPEG-LS variation for compressed image retrieval," in *Proc. Int. Symp. Intell. Multimedia, Video Speech Process.*, Oct. 2004, pp. 169–172.

[10] J. Chen and T.-S. Chen, "Content-based rate controlled data hiding in JPEG-LS," in *Proc. 3rd Int. Conf. Intell. Inf. Hiding Multimedia Signal Process.*, vol. 1. Nov. 2007, pp. 477–480.

[11] S. G. Miaou, F. S. Ke, and S. C. Chen, "A lossless compression method for medical image sequences using JPEG-LS and interframe coding," *IEEE Trans. Inf. Technol. Biomed.*, vol. 13, no. 5, pp. 818–821, Sep. 2009.

[12] P. Turcza and M. Duplaga, "Hardware-efficient low-power image processing system for wireless capsule endoscopy," *IEEE Trans. Biomed. Health Inf.*, vol. 17, no. 6, pp. 1046–1056, Nov. 2013.

[13] X. Xie, G. Li, X. Chen, X. Li, and Z. Wang, "A low-power digital IC design inside the wireless endoscopic capsule," *IEEE J. Solid-State Circuits*, vol. 41, no. 11, pp. 2390–2400, Nov. 2006.

[14] A. Savakis and M. Piorun, "Benchmarking and hardware implementation of JPEG-LS," in *Proc. ICIP*, Rochester, NY, USA, Sep. 2002, pp. II-949–II-952.

[15] X. Chen *et al.*, "A wireless capsule endoscope system with low-power controlling and processing ASIC," *IEEE Trans. Biomed. Circuits Syst.*, vol. 3, no. 1, pp. 11–22, Feb. 2009.

[16] *Hamlyn Centre Laparoscopic/Endoscopic Video Datasets*, accessed on Oct. 6, 2015. [Online]. Available: http://hamlyn.doc.ic.ac.uk/vision/

[17] H. S. Kim, J. Lee, H. Kim, S. Kang, and W. C. Park, "A lossless color image compression architecture using a parallel Golomb-Rice hardware CODEC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 11, pp. 1581–1587, Nov. 2011.

[18] T. H. Khan and K. A. Wahid, "Low power and low complexity compressor for video capsule endoscopy," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 10, pp. 1534–1546, Oct. 2011.

[19] *Gastroenterologist Hospital. El Salvador Atlas of Gastrointestinal VideoEndoscopy*, accessed on Oct. 6, 2015. [Online]. Available: http://www.gastrointestinalatlas.com

[20] *Gastrollab—The Gastrointestinal Site*, accessed on Oct. 6, 2015. [Online]. Available: http://www.gastrolab.net/ni.htm

[21] *Information Technology-Lossless and Near-Lossless Compression of Continuous-Tone Images-Baseline*, documnet ISO/IEC 14495-1, International Telecommunication Union (ITU-T Rec. T.87), 1998.

SHIH-LUN CHEN (M'12) received the B.S., M.S. degree, and Ph.D. degrees from National Cheng Kung University, Tainan, Taiwan, in 2002, 2004, and 2011, respectively, all in electrical engineering. He was an Assistant Professor with the Department of Electronic Engineering, Chung Yuan Christian University, Taiwan, from 2011 to 2014, where he has been an Associate Professor with the Department of Electronic Engineering, since 2014. His current research interests include VLSI chip design, image processor design, image processing, data compression, fuzzy logic control, wireless body sensor network, bio-medical signal processing, and reconfigurable architecture.

TSE-YEN LIU received the B.S. degrees in electronic engineering from Chung Yuan Christian University, Taoyuan, Taiwan, in 2015. His current research interests include VLSI chip design and image processing.

CHIA-WEI SHEN received the B.S. degree in electrical engineering from Chung Yuan Christian University, Taoyuan, Taiwan, in 2015. He was involved in algorithm analysis and software programming. He was a Student with S.-L. Chen, to research JPEG-LS system architecture and image compression code in a group, from 2013 to 2014. His current research interests in image processing and data compression.

MIN-CHUN TUAN received the B.S. degree and applied through a Dual bachelor's-Master's degree from the Department of Electronic Engineering, Chung Yuan Christian University, in 2013 and 2014, respectively, where he is currently pursuing the Ph.D. degree with the Biomedical/Multimedia IC and System Design Laboratory. His research fields include image processing, digital chip design, bio-medical signal processing, and FPGA verification.