

# Lossless Multi-mode Interband Image Compression and its Hardware Architecture

Xiaolin Chen, C. Nishan Canagarajah, Jose L. Nunez-Yanez

Department of Electrical and Electronic Engineering

University of Bristol, UK

Email: {Xiaolin.Chen, Nishan.Canagarajah, J.L.Nunez-Yanez}@bristol.ac.uk

**Abstract**—This paper presents a novel Lossless Multi-Mode Interband image Compression (LMMIC) scheme and its hardware architecture. The approach detects the local features of the image and uses different modes to encode regions with different features. Run-mode is used in homogeneous regions, while ternary-mode and regular-mode are used on edges and other regions, respectively. In regular mode, we propose a simple band shifting technique as inter-band prediction and a gradient-based switching strategy to select between intra-band or inter-band prediction. The advantage of LMMIC is to adaptively switch among modes and predictors in different regions, to avoid complicated calculations in traditional segmentation and inter-band coefficients. This feature enables the hardware amenability. Experimental results show that LMMIC achieves superior compression ratios as well as high throughput. It also provides the benefits of enabling encoding any number of bands and easy access to any band.

## I. INTRODUCTION

The rapid advances of multimedia technology generates huge amount of image data. Most of them are multispectral images, which contain more than one spectral bands. For instance, color images, often displayed as RGB, BMP, or TIF format, have at least three bands. In remote sensing, the LANDSAT 7 images have seven bands, and the AVIRIS hyperspectral images contain 224 bands. These images form the base of the widely used web mapping service, e.g. the Google Earth. In medical imagery, multispectral images also prevail. These images are normally compressed for transmission and storage. As many applications, e.g. remote sensing imaging, medical imaging, pre-press imaging and archiving [1], demand perfect reconstruction of images, lossless compression on multispectral images attracts increasing interests. Also for applications that need to transmit image data instantly after acquisition, real-time compression is desirable. To this end, we aim to design an efficient hardware amenable lossless interband image compression scheme, with the capability of real-time processing.

Unlike gray-scale image, multispectral image has not only spatial but also spectral redundancy. Moreover, the existence of multiple bands suggests two problems worth of concern - to encode any number of bands and to access whichever band. As spatial coding techniques have been extensively studied [2][3][4], recently a lot of research focuses on removing the spectral redundancy. Popular interband coding techniques include vector quantisation [5][6][7], Discrete Cosine

Transform [8], Discrete Wavelet Transform [9], and vector-lifting schemes [10]. These coding techniques are efficient in reducing spectral redundancy, but their high computational complexity and often jointly encoding several bands (e.g. 16 bands in [9]) are obstacles for hardware implementation and real-time processing. On the other hand, predictive coding does not only perform well in removing spatial redundancy but also spectral redundancy. Wu extended Intra-band CALIC [2] to Inter-band CALIC [1], which offers one of the best inter-band compression results in literature but requires complex inter-band correlation coefficients calculation and context formation. SICLIC [11] is a simple and efficient coder based on LOCO-I [3], but its 3-band joint-run mode, while giving good bit rates, constrains it from encoding any number of bands.

To relieve these problems, we propose a Lossless Multi-Mode Interband image Compression (LMMIC) scheme. The proposal of this scheme is inspired by the concept of segmentation. Segmentation, in a general sense, is to partition an image into multiple regions in order to change the representation of an image into something meaningful or easy to analyze. However, traditional segmentation, e.g. statistical model-based methods [12] and graph-based methods [13], is too complex to implement in real-time compression. The novelty of our scheme is to apply the idea of segmentation to group pixels with similar features and use different modes to encode them. We propose a multi-mode strategy, where a new ternary-mode is designed to detect and encode the edges, and a run-length coder [3] is to encode the homogeneous regions. The rest of the image, say the texture regions, is coded by a regular-mode which consists of intra-band and inter-band prediction. We propose a simple band shifting technique for inter-band prediction and adapt the Gradient-Adjusted Prediction (GAP) from CALIC [2] for intra-band prediction. A new gradient-based switch is also designed to select the better predictor. This multi-mode strategy applies to all bands in an image. The proposed scheme does not only offer excellent compression ratios, but also the distinctive feature of the flexibility of encoding any number of bands. The multi-mode strategy and switching method make LMMIC hardware amenable. Note that the proposed scheme in this paper is for general purpose (e.g. space, medical, archiving) images with more than one spectral bands but not specifically geared for hyperspectral images. Since some of the techniques for hyperspectral images make specific use of the structure of these images, for example,

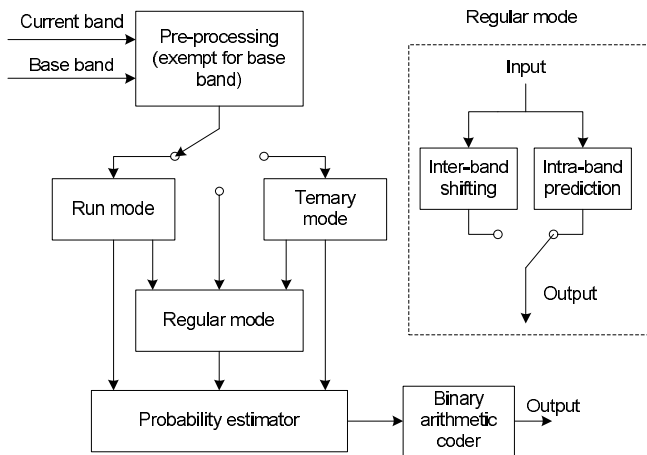


Fig. 1. Schematic of the proposed image compression system

by including a band ordering process [14] or by clustering a number of bands [9], we do not include those highly specialized and not necessarily hardware amenable methods in our comparison study. However, we acknowledge that refining our techniques for specific use with hyperspectral images is an interesting topic for further research, and its findings will be reported elsewhere.

The paper is organized as follows. In Section 2 we present an overview of the proposed scheme. In Section 3 and 4, we explain the details of the multi-mode strategy and the intra/inter-band switching method, followed by the hardware architecture in Section 5. We show the performance comparison with other state-of-the-art schemes in Section 6 and conclude our work in Section 7.

## II. AN OVERVIEW OF LMMIC

An image contains many features, such as smooth regions, edges, texture etc. The complexity of an image is an obstacle for compression, thus segmentation (also referred to as region-based methodology) is a viable approach to help with distinguishing these features. The lossless image compression method TMW [15], which achieves the best gray-scale image compression ratio so far, uses segmentation to analyze the image in the first pass. Shen [16] proposed SLIC to combine the region-growing algorithm in segmentation with lossless compression for medical images. Ratakonda [17] used multiscale segmentation in encoding general images and achieved very good compression results. However, they are both complex and two-pass schemes so cannot well meet the real-time processing requirement. Due to the complexity and the nature of segmentation, we skip the conventional methods but apply its concept, by using a simple online checking of neighboring symbols to detect different image features and using different modes to encode these features. This is the idea that our scheme is based on.

Fig. 1 shows the schematic of LMMIC. Firstly, one band is chosen as base band, for instance band G in RGB images, or the first band received in the sequence. Then a pre-processing

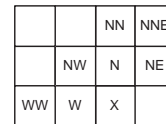


Fig. 2. Neighboring symbols of the current symbol

stage simply subtracts the base band from the current band to get the band difference, as in

$$Band_{diff} = Band_{curr} - Band_{base} \quad (1)$$

Thus each band is only coupled with the base band and no multi-band joint encoding is needed. This allows the flexibility of compressing any number of bands and easy access to any bands. Then a three-way switch enables the system to choose among run-mode, ternary-mode and regular-mode according to the local feature of where the input symbol is. Among these three modes, run-mode and ternary-mode are set to have higher priority and are checked first. Symbols that fail to be encoded in run-mode or ternary-mode are encoded in regular-mode. The diagram in Fig. 1 shows that regular-mode consists of intra-band and inter-band prediction working in parallel, and a switch enables the adaptation. While the base band is directly applied on these three modes, for other bands, only the band difference is applied on run-mode and ternary-mode but both the current band and the band difference are applied on regular-mode, when it is activated. After prediction, higher order redundancy of the image residue is removed by context modeling, which is done in a similar manner with the one in CALIC [2], with some simplification and modification for the intraband and interband case respectively. The output prediction errors and context information are processed by a probability estimator and a binary arithmetic coder [18]. The proposed scheme is a symmetric encoder, as the decoding is just the opposite procedure of encoding. Therefore, both encoding and decoding has the same level of complexity. In the next section, we explain the details of each working mode.

## III. MULTI-MODE STRATEGY

Due to the inter-band correlation in images, subtracting the base band from the current band often generates more homogeneous region in the band difference, which is beneficial for compression. On the other hand, many of the image features, such as edges, tend to still remain in the band difference. Therefore, instead of the original band, we apply the band difference on the run-mode and the ternary-mode, and in areas with more complex texture, we can select between intra-band or inter-band prediction, which will be detailed in the next section. In this section, we present how each mode works and the conditions for entering each mode.

### A. Run-mode

Run-length coding is simple and efficient in grouping identical symbols [3]. It encodes the occurrence, also called *run*, of a symbol if it appears continuously. We use run-length coding to encode the homogeneous regions of the image. Fig. 2 shows

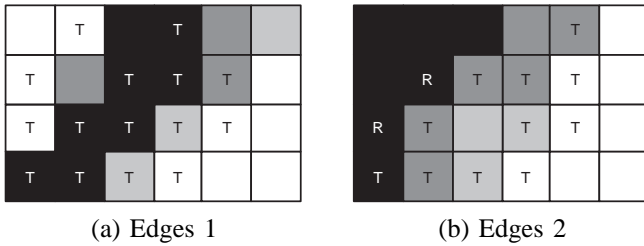


Fig. 3. Areas where ternary-mode is performed

the neighboring symbols of the current symbol  $X$  according to their geographical positions. When  $W = N = NW = NE$ , the current symbol is assumed to be in a homogeneous region and is tried to be encoded in run-mode. If  $X$  is identical to  $W$ , the run-length increases by one; otherwise “run” stops and the current run-length is encoded. The latter case means that encoding symbol in run-mode is unsuccessful, so the symbol is encoded in regular-mode.

### B. Ternary-mode

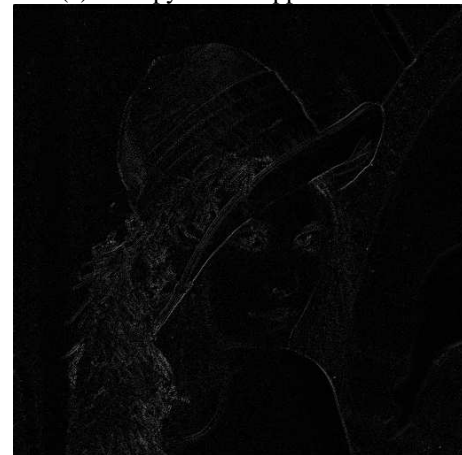
Ternary-mode is inspired by the binary-mode in CALIC, which works on the binary area where there are only two different symbols in the neighborhood, e.g. black and white texts. However, unlike CALIC, our ternary-mode is based on the idea of edge detection. In areas where a sharp edge occurs, shown in Fig. 3 (a), symbol values at the two sides of the edge are usually different; also, where a less sharp edge occurs, as in Fig. 3 (b), symbol values tend to be changing gradually. In both cases, we assume that within a small neighborhood of the current symbol, say the seven neighboring symbols in Fig. 2, there are no more than three distinctive symbols and the ternary-mode is triggered. Fig. 3 shows the areas that the ternary-mode is performed.  $T$  indicates the symbols encoded by ternary-mode, while  $R$  indicates run-mode, and the color is the gray-level of the symbols. The figure tells that edge areas can be largely covered by this mode. In ternary-mode, symbol  $W$  is represented as  $s_1$ , while the second and third distinctive symbols are represented as  $s_2$  and  $s_3$  respectively. In other word, the current symbol can be denoted by their order of appearance given the condition that the checking of the context is always conducted in the same order. The current symbol is encoded by

$$T = \begin{cases} 0, & \text{if } x = s_1; \\ 1, & \text{if } x = s_2; \\ 2, & \text{if } x = s_3; \\ \text{escape}, & \text{otherwise.} \end{cases} \quad (2)$$

“Escape” happens when encoding in this mode fails. It is a way of switching among modes. The alphabet size for encoding in this mode is only four so lower entropy is obtained. Ternary-mode also works as a “backup” of the run-mode in smooth but not exactly homogeneous regions. Fig. 4 are the residue images of “lenagrey” after GAP and LMMIC respectively. The zero order entropy of the residue image after GAP is 4.39bpp while it is 4.31bpp after LMMIC. Fig. 5 (b) shows the regions



(a) entropy = 4.39bpp after GAP



(b) entropy = 4.31bpp after LMMIC

Fig. 4. Sample residue images after GAP (a) and LMMIC (b)

where different modes perform, comparing with the original image (a). Run-mode works on the grey homogeneous regions; ternary-mode works on the white regions, which often lie on the edge of the homogeneous regions, some smooth regions or clear edges; regular-mode works on the dark regions, which are mostly texture or noisy areas.

### C. Regular-mode

Regular mode is triggered, either when the entry conditions for run-mode and ternary-mode cannot be met, or when encoding in other modes fails. Regular mode consists of intra-band and inter-band prediction, which is selected according to the local features adaptively. Details of the inter-band prediction and switching strategy are discussed in next section.

## IV. GRADIENT BASED SWITCHED INTRA/INTER-BAND PREDICTION AND CONTEXT MODELING

We design a simple band shifting technique for inter-band prediction, and adapt the GAP from CALIC for intra-band prediction. However, the performance of inter-band prediction depends on the band correlation. In the case of strong band



(a) original image “bike3”



(b) different modes indication on image “bike3”, where grey area indicates run-mode; white dots indicate ternary-mode; dark area indicates regular-mode.

Fig. 5. Original image (a) and image indicating where different modes applied (b)

correlation, inter-band prediction is preferred, otherwise intra-band prediction is selected. A gradient based switching method is proposed for the selection.

#### A. Band shifting for inter-band prediction

In the regions where bands are strongly correlated, symbol changes in one band often happen in another band. Fig. 6 shows the plots of one line in band G and band B in the image “peppers”. It is clear that the dot plot from band G has a similar trend with the dash plot from band B. We intend to shift the reference band G to a position that is close to the current band B so that only a small difference between the current band and the shifted reference band needs to be encoded. There are a lot of possible ways to predict the value

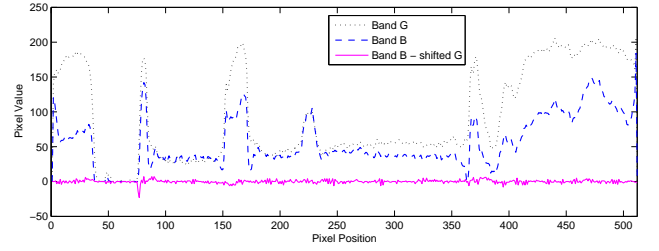


Fig. 6. Plots of one line in band G and band B and their difference after shifting

for band shifting. Since this band shifting is only used when band correlation is strong, the band difference tend to be very small. Considering the level-set preserving property of the simple median predictor, we use it to predict the band shifting. We denote the band difference at position  $W$ ,  $N$ ,  $NW$  as  $W\_diff$ ,  $N\_diff$ ,  $NW\_diff$ , and calculate the band shifting by

```

if NW_diff >= max(N_diff, W_diff)
    shift_band = min(N_diff, W_diff);
else if NW_diff <= min(N_diff, W_diff)
    shift_band = max(N_diff, W_diff);
else
    shift_band = N_diff + W_diff - NW_diff;
end

```

The solid plot in Fig. 6 shows that this prediction method successfully generate a zero-mean band difference between the current band and the shifted reference band.

#### B. Gradient-based switching

As the performance of the two predictors varies in different regions of an image depending on the spatial and spectral correlation, it is critical to design a suitable selecting strategy to decide which one to use. As we aim at designing a hardware amenable scheme, complex calculation of inter-band correlation coefficients has to be avoided. We propose a simple switching method according to the local horizontal and vertical gradients, which are calculated by

$$\begin{aligned} dh &= |W - WW| + |N - NW| + |N - NE| \\ dv &= |W - NW| + |N - NN| + |NE - NNE| \end{aligned} \quad (3)$$

where  $dv$  is the vertical gradients and  $dh$  is the horizontal one. When calculating the inter-band gradients,  $W$ ,  $N$ ,  $NW$ ,  $NE$ ,  $NN$ ,  $WW$ ,  $NNE$  are substituted by the inter-band difference at the same positions. Inter-band gradients indicate how closely the two bands change in the same way. In addition to the gradients, the previous prediction error is taken into account to evaluate how well the predictor performs in the local area. Therefore, for both intra-band and inter-band prediction, we calculate

$$S = dv + dh + |E_w| \quad (4)$$

where  $E_w$  means the prediction error at position  $W$ . The predictor that gives smaller  $S$  is selected to encode the current symbol. Experiments on the test images used in the results section show that this simple method makes 70% or above right decisions.

### C. Context Modeling

Context modeling is to group the prediction residue based on local contexts in order to obtain a lower conditional entropy. In the proposed scheme, context is formed in a similar manner with CALIC [2] but is simplified to reduce the memory usage. We take 6 context symbols ( $W, N, NW, NE, NN, WW$ ) to compare with the predicted value  $\hat{X}$  to obtain a texture pattern  $t$ , representing the local texture feature. Also, to indicate the activity of errors in a context, a coding context is generated with the local gradients  $dv, dh$  and a previous prediction error  $e$  of  $W$ . The coding context is then quantized into 8 levels to form a coding context index. Combining the texture pattern and the coding context, a set of 512 compound contexts are formed with 6 bits texture pattern  $t$  and 3 bits coding context index. In the case of interband context formation, context symbols are replaced by the band difference at the same position. These contexts are also used to generate an bias cancellation of the predictor, which will be discussed in the next section. The 8 coding contexts are used to calculate the occurrence probability of symbols in the probability estimator presented in Section V-B.

## V. HARDWARE ARCHITECTURE

Hardware amenability is one of the priorities in the design of the proposed scheme. Therefore, as previously described, the whole procedure, including the prediction and mode switching, only involves a limited number of addition and shift, and only need to buffer three lines in two bands. In this section we propose the suitable hardware architecture to support the proposed compression scheme, which includes the architecture of the prediction and context modeling module, probability estimator and binary arithmetic coder. As the binary arithmetic coder is adopted from [18], we will not explain in detail here.

### A. Prediction and Context Modeling Module

To further optimize the hardware architecture, the data flow of the prediction and context modeling module is achieved with two pipelines running in parallel, as shown in Fig. 7. Line 1, indicated by the flow on the left, operates on the current symbol and yields the symbol “runs” or symbol order or prediction error with the selected mode for the probability estimator; Line 2, indicated by the flow on the right, updates the data from Line 1, calculates the prediction value and the context index for the next symbol under the selected mode. The advantage of dividing the procedure into two parallel pipelines is, while not violating the sequential constraint, to halve the execution time and hence obtain higher throughput. As the arithmetic calculation part is introduced in previous section, we will explain the rest, which is managing the context

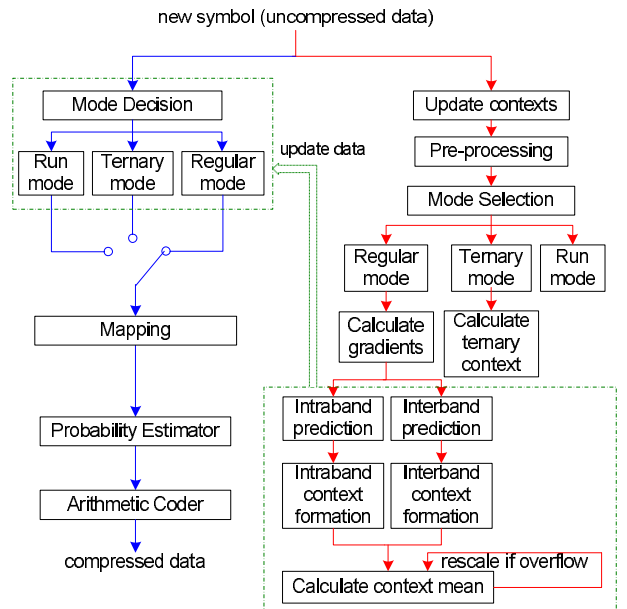


Fig. 7. Data Flow of the prediction and context modeling module

memory and the error feedback technique for bias cancellation, below in this subsection.

In the compression process, we need to store 3 lines of image pixel values in memory as context. To avoid reading in 3 lines every time when a line finishes, we use 3 lines’ memory to store the pixel values and 3 pointers to indicate symbol locations in each line. At the end of processing each line, the 3 pointers to each line rotate in a certain order so that the oldest line is discarded and the newly formed line is saved.

An error feedback technique is used in the prediction step to adjust the bias in prediction in certain context. As prediction can not be accurate, the mean of errors  $\bar{e}$  is assumed to be the most probable prediction offset error in each context, and hence is a good observation of the bias of the predictor. We improve the prediction by adding on this term. It is calculated by

$$\bar{e} = \text{sum} / \text{count} \quad (5)$$

where  $\text{sum}$  and  $\text{count}$  are the sum and occurrence count of errors in the context, respectively. The calculation of mean requires arbitrary division, which is not desirable in hardware implementation. To solve the problem, we store the  $\text{count}$  with only 5 bits ( $2^5 - 1 = 31$ ). When the  $\text{count}$  reaches its maximal value 31, it is halved by right-shifting one bit; meanwhile  $\text{sum}$  is halved so as to maintain the mean  $\bar{e}$ . Thus we only need 13 bits ( $2^5 \times 2^8 = 2^{13}$ , assuming the error range is 0 to  $2^8 - 1$ ) plus one sign bit to store the sum of errors safely. Experimental results prove that this rescaling technique slightly improves the compression ratio by “aging” the observed data. However, division is always a difficult problem in hardware, especially when the dividend can be as large as 13 bits. To make this division practical, we bound the dividend  $\text{sum}$  by 10 bits for two reasons: firstly,

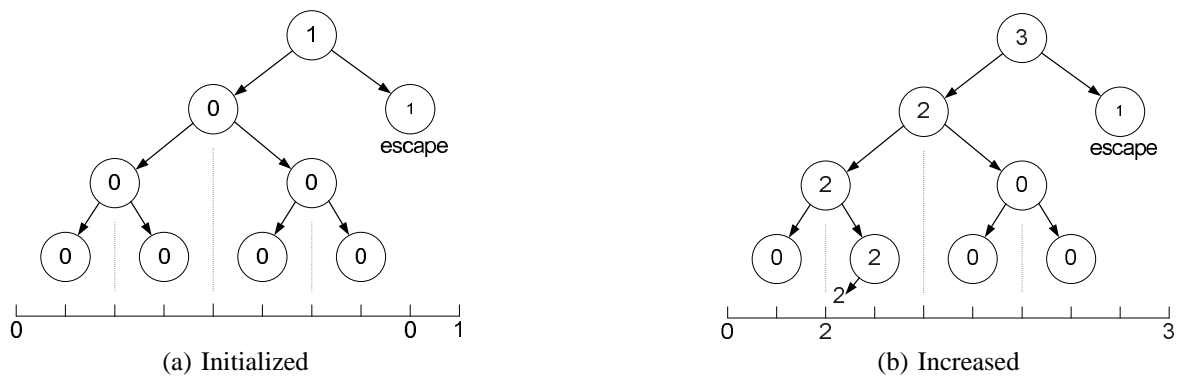


Fig. 8. Simplified tree structure of the probability estimator

experiments on our image test set show that the chance of the *sum* being larger than 1023 is less than 0.001%; secondly, extraordinary large errors tend not to reflect the true behavior of the context because prediction errors should be moderately small given an adequate predictor. Therefore, we use the most significant bits of the divisor *count* in the division, with the dividend being rescaled accordingly to maintain the same result. Consequently, we only need a lookup table of 1KByte ( $2 \times 512 = 1024$ ) to complete fast division. Although the result of division is only an approximation, it does not affect the compression performance in our experiments.

### B. Probability Estimator

As the prediction residue generated by the prediction and context modeling module has a multiple-symbol alphabet (e.g. for 8 bits per pixel, there are  $2^8 = 256$  symbols), they cannot be processed in a binary arithmetic coder directly. We propose a probability estimator to adaptively calculate the probability of symbol occurrence in each context in a SRAM, and to decompose the data into bit level. Thus it enables the application of a simple and efficient binary arithmetic coder and hence results in full pipelining and high throughput.

The main part of the probability estimator is a tree structure model stored in a SRAM. Each context is represented by a balanced binary tree with  $2^n$  ( $n$  is the bits per pixel) nodes associated with each symbol in the alphabet. A number of bits are used to store the symbol frequency count in each node. Initially, all the symbols in the alphabet are assigned an equal probability, and the whole range of the probability is 1. When one symbol is received, the value of the corresponding tree node increases to reflect the probability distribution of symbol occurrence. We demonstrate the working theory of the model with a simplified tree structure, as shown in Fig. 8, where  $n = 3$ . Firstly, in Fig. 8(a), each tree node is assigned to 0, and the whole range of the probability is assigned to “*escape*”. “*Escape*” here means coding is not successful. Then in Fig. 8(b) a symbol “2” (“010” in binary) arrives, so the counts for symbol “2” is increased by 2, or any specified value, and the probability of symbol “2” becomes  $2/3$ . This is reflected by the value of the tree nodes over the total value of the tree root. In this way, to encode the symbol “2”, we only need to

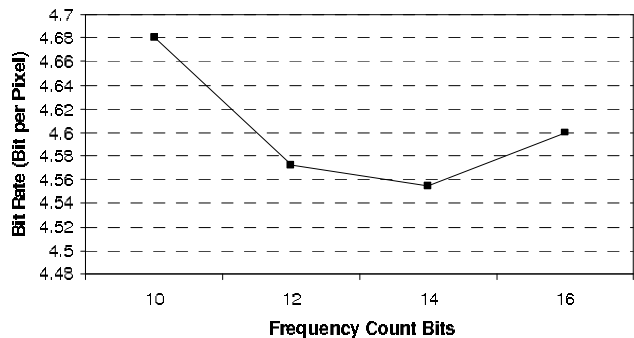


Fig. 9. Average Bit Rates under Different Probability Precision

encode the left or right decision when the symbol comes down through the tree.

As mentioned in previous section, there are 8 coding contexts, corresponding to 8 “dynamic” trees and one “static” tree for coding the *escape* symbols. *Escape* happens when a valid probability of a symbol cannot be found, e.g. when its probability is 0, in which case the symbol is “escaped” to the “static” tree and is sent as it is. *Escape* is undesirable as it does not achieve any compression. It takes place when some symbol frequency counts reach the maximal frequency count, e.g. 14 bits for  $(2^{14} - 1)$ , in which case all the symbol frequency counts in the tree will be halved. Consequently, the counts of symbols that have not been seen before will be rescaled to 0, resulting in *escape* when those symbols occur later for the first time. Therefore, the frequency count bits have to be carefully chosen. Experimental results of average compression bit rates under different frequency count bits are shown in Fig. 9. When the maximal frequency count is too small, more *escapes* are likely to happen; when the maximal frequency count is too big, the “aging” effect of observed data is less. Therefore, we choose 14 bits according to the result of Fig. 9.

Fig. 10 shows a simplified diagram of the probability estimator. There is a SRAM memory where the probability of symbols in each coding context is stored, and a SRAM to store the total root value of each context tree. The incoming symbol, which is the prediction error, is shifted and compared with

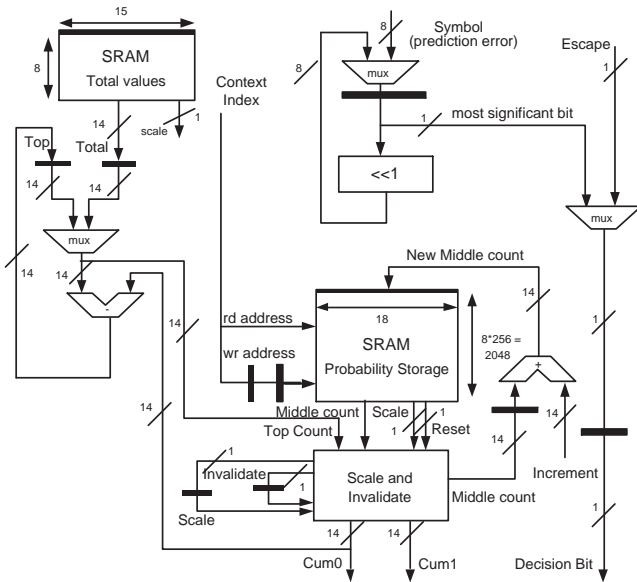


Fig. 10. Architecture of the probability estimator

the value “1” to get the decision bit (0 or 1, as left or right). Meanwhile, the context index is read in the corresponding context tree, where the *middlecount* represents the current symbol and the *totalcount* represents the total value in the context. So the probability is calculated as

$$p = \frac{\text{Middle Count}}{\text{Total Count}} \quad (6)$$

The probability is output as *Cum0* and *Cum1* as a binary sequence. This module maps the probability data into a set of binary decisions from the root to the leaves through each context tree. The binary arithmetic coder is driven by these decision bits and the probability data. It is multiplication-free, resulting in an improved clock ratio of the system. One decision bit is processed per clock cycle, and hence 8 cycles are needed for encoding one byte.

## VI. PERFORMANCE COMPARISON

The experimental result in terms of compression ratio is presented in this section. We choose a set of standard 3-band RGB images, a 4-band CMYK image “park” and a 7-band LANDSAT 7 image “coastal” from CCSDC (the Consultative Committee for Space Data Systems). The RGB images include continuous-tone images (“cats”, “water”, “lena” and “peppers”), compound images (“cmpnd1” and “cmpnd2”) and synthesized images (“bike3”). We compare the proposed compression scheme with JPEG2000 [19], intra-band CALIC [2], IB-CALIC [1] and SICLIC [11]. The results of IB-CALIC and SICLIC are extracted from [11]. Some results are absent due to the unavailability of the programs. JPEG2000 is the current standard for image compression. The results of IB-CALIC are one of the best in literature in terms of general inter-band image compression, but not hyperspectral image compression, which is not the scope of our proposed method either. And SICLIC is a good trade-off between compression

ratio and complexity. Table. I shows that LMMIC outperforms JPEG2000 and intra-band CALIC by 10% and 14%, respectively. It is superior than SICLIC on average, though slightly inferior than IB-CALIC which has higher computational complexity. Since the inter-band coding in LMMIC only couples two bands, it has the flexibility of compressing images with any number of bands and easy access to any bands. The proposed hardware architecture together with the binary arithmetic coder enables the system to process one bit per clock cycle, which translates into a throughput of around 123Mbits/s on a Xilinx Virtex 4 FPGA.

## VII. CONCLUSIONS

An original Lossless Multi-Mode Interband image Compression (LMMIC) scheme is proposed. The concept of segmentation is well ingrained in this scheme to deal with different regions in the image adaptively. The simple and efficient band shifting technique and the switching strategy successfully remove the inter-band redundancy. Experiments show that LMMIC achieves highly competitive compression ratios and provides the flexibility of compressing any number of bands as well as easy access to any bands, which are not offered by many other schemes. The complexity of the scheme is strictly controlled and hardware amenability is maintained. The proposed corresponding hardware architecture proves the achievement of high throughput.

## ACKNOWLEDGMENT

The authors would like to thank the support from EPSRC under grant EP/D011639/1.

## REFERENCES

- [1] X. Wu and N. Memon, “Context-based lossless interband compression – extending CALIC,” *IEEE Trans. Image Processing*, vol. 9, no. 6, pp. 994–1001, June 2000.
- [2] X. Wu and N. Memon, “Context-based, adaptive, lossless image coding,” *IEEE Trans. Commun.*, vol. 45, no. 4, pp. 437–444, Apr. 1997.
- [3] M. J. Weinberger, G. Seroussi, and G. Sapiro, “LOCO-I: A low complexity, context-based, lossless image compression algorithm,” in *Proc. Data Compression Conference*, Mar. 1996, pp. 140–149.
- [4] X. Li and M. T. Orchard, “Edge-directed prediction for lossless compression of natural images,” *IEEE Trans. Image Processing*, vol. 10, no. 6, pp. 813–817, June 2001.
- [5] M. J. Ryan and J. F. Arnold, “The lossless compression of aviris images by vector quantization,” *IEEE Trans. Geosci. Remote Sens.*, vol. 35, no. 3, pp. 546–550, May 1997.
- [6] J. Mielikainen and P. Toivanen, “Improved vector quantization for lossless compression of AVIRIS images,” in *Proc. XI Eur. Signal Process. Conf.*, Sept. 2002.
- [7] G. Motta, F. Rizzo, and J. A. Storer, “Compression of hyperspectral imagery,” in *Proc. Data Compression Conference, DCC’03*, Mar. 2003, pp. 333–342.
- [8] P.L. Dragotti, G. Poggi, and A.R.P. Ragozini, “Compression of multi-spectral images by three-dimensional SPIHT algorithm,” *IEEE Trans. Geosci. Remote Sens.*, vol. 38, no. 1, pp. 416–428, Jan. 2000.
- [9] X. Tang, W. A. Pearlman, and J. W. Modestino, “Hyperspectral image compression using three-dimensional wavelet coding,” in *Proc. SPIE*, Jan. 2003, vol. 5022, pp. 1037–1047.
- [10] A. Benazza-Benyahia, J. C. Pesquet, and M. Hamdi, “Vector-lifting schemes for lossless coding and progressive archival of multispectral images,” *IEEE Trans. Geosci. Remote Sens.*, vol. 40, no. 9, pp. 2011–2024, Sept. 2002.

TABLE I  
BIT RATES (BITS PER PIXEL PER BAND) COMPARISON ON SELECTED SCHEMES

image	JPEG2000	CALIC	IB-CALIC	SICLIC	LMMIC
cmpnd1	1.44	1.21	1.02	1.12	1.05
cmpnd2	1.30	1.22	0.92	0.97	0.97
cats	1.75	2.49	1.81	1.85	1.82
water	1.41	1.74	1.51	1.45	1.44
lena	4.53	4.40		4.46	4.23
peppers	3.41	4.62		3.25	3.35
bike3	5.17	4.21		4.41	4.29
<b>average</b>	<b>2.72</b>	<b>2.84</b>		<b>2.50</b>	<b>2.45</b>
park	5.72	5.39			5.30
coastal	2.89	2.68			2.62

- [11] R. Barequet and M. Feder, "SICLIC: a simple inter-color lossless image coder," in *Proc. Data Compression Conference, DCC'99*, Mar. 1999, pp. 501–510.
- [12] J. Kim, J. W. Fisher, A. Yezzi, M. Cetin, and A. S. Willsky, "A non-parametric statistical method for image segmentation using information theory and curve evolution," *IEEE Trans. Image Processing*, vol. 14, no. 10, pp. 1486–1502, Oct. 2005.
- [13] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *Internation Journal of Computer Vision*, vol. 59, no. 2, pp. 167–181, Sept. 2004.
- [14] S. R. Tate, "Band ordering in lossless compression of multispectral images," *IEEE Trans. Comput.*, vol. 46, no. 4, pp. 477–483, Apr. 1997.
- [15] B. Meyer and P. E. Tischer, "TMW - a new method for lossless image compression," in *Proc. Int Picture Coding Symp.*, Oct. 1997.
- [16] L. Shen and R. M. Rangayyan, "A segmentation based lossless image coding methods for high-resolution medical image compression," *IEEE Trans. Med. Imag.*, vol. 16, no. 3, pp. 301–307, June 1997.
- [17] K. Ratakonda and N. Ahuja, "Lossless image compression with multiscale segmentation," *IEEE Trans. Image Processing*, vol. 11, no. 11, pp. 1228–1237, Nov. 2002.
- [18] J. Nunez-Yanez and V. Chouliaras, "A configurable statistical lossless compression core based on variable order Markov modeling and arithmetic coding," *IEEE Trans. Comput.*, vol. 54, no. 11, pp. 1345–1359, Nov. 2005.
- [19] D. S. Taubman and M. W. Marcellin, *JPEG2000 Image Compression Fundamentals, Standards and Practice*, Norwell, MA: Kluwer, 2002.