Prediction with Partial Match using Two-Dimensional Approximate Contexts

Ishtiaque Hossain and Mahmoud R. El-Sakka, SM IEEE
Computer Science Department
The University of Western Ontario
London, Ontario N6A 5B7, Canada
Email: ihossai@csd.uwo.ca and elsakka@csd.uwo.ca

***Abstract*—The Prediction with Partial Match (PPM) is a context-based lossless compression scheme developed in the mid 80's. Originally it was targeted towards compressing text that can be viewed as a one-dimensional sequence of symbols. When compressing digital images, PPM usually breaks the two-dimensional data into a one-dimensional raster scan form. This paper extends PPM in order to take full advantage of the two-dimensional nature of digital images. Unlike the traditional two-dimensional raster scan contexts (i.e. concerning upper pixels and pixels to the left), the proposed context is determined using pixels from all directions, including pixels to the right and the lower pixels. Results show that this type of context yields a significant improvement over the traditional raster scan context.**

***Index Terms*—Image encoding; Lossless compression; Context-based schemes; Prediction with Partial Match (PPM); Two-dimensional compression; Approximate context.**

I. Introduction

Image compression refers to removing irrelevant or redundant data from an image to store the information in less space or transmit it in short time. Over the decades, numerous image compression schemes have been developed. These schemes can be classified into two broad categories: lossless schemes and lossy schemes. Lossless compression schemes recover the exact original information after decompressing the compressed file. They do not provide high compression although guarantee the integrity of the information. Lossy schemes, on the other hand, reconstruct an approximated replica of the original information after decompression and provide high compression at the cost of information loss in the original data. This paper focuses only on lossless schemes.

Lossless compression schemes can be further classified into a wide range of categories. One of them is the class of entropy coders, i.e. the Huffman coder and the arithmetic coder. The latter is the one we are interested in. The basic principle behind entropy coders is to use fewer bits to encode frequently occurring pixels (pixels with high probability) and larger number of bits to encode less frequently occurring pixels (pixels with low probability). Entropy coders perform best when the probability distribution across symbols is highly skewed.

In text, we can introduce the notion of context while using entropy coders. For example, if the letters *probabilit* have been encoded so far, the likelihood of the next symbol is to be *y*. In other words, if we look at a particular context and build a list of symbols that appeared in that context, the list will have a skewed probability distribution across symbols. The purpose of the context is to create partitions in the probability distribution to provide even better compression. Based on this observation, Cleary and Witten proposed a compression scheme called Prediction with Partial Match (PPM) in 1984 [1]. PPM can be viewed as an extended arithmetic encoder which estimates the probability of a symbol using contexts from history and encodes it using this probability. Sayood included a detailed description of this scheme in his book [2]. The original algorithm was devised for the purpose of compressing text. The context of a symbol is determined by the symbols preceding it immediately - thus the context is one directional.

There have been a few studies attempting to extend the original PPM scheme. Some of them concentrated on estimating the probability of escape (an event that indicates that a symbol was not found in the current context), also known as the *zero-frequency problem*. Several different approaches like PPMA, PPMB and PPMC have been proposed [2]–[5]. Although a formal proof still does not exist, PPMC is widely believed to be the most efficient among them. Cleary and Teahan reported the effect of unbounded context-order [6]. It has also been observed that there is no significant compression after order 5 or 6 [6].

This notion of context can be applied to images as well because images contain locally homogeneous areas. In other words, pixel values tend to be close to the surrounding pixels. Unlike text, a pixel in an image has its context all around and therefore, the context should be derived using pixels from all directions. This is challenging as the traditional way to traverse an image is the raster scan and with this traversal technique, context-pixels can be found in only two directions at most: from the left and top, since they are the pixels available to both the encoder and the decoder yet. Context from two directions has been attempted in a few studies [7]–[9]. It is worth mentioning that Howard utilized the surrounding pixels from all four directions to better predict the pixel in the middle. However, when it comes to PPM context, only pixels from the left and top were utilized [7].

Although looking for exact matches in context-history produces good results for text, it is not the technique to follow in case of images, as the contexts in images do not repeat themselves in exact form, but in close similarity. If exact contexts are used, the number of escape symbols will be too large to achieve noticeable compression. Some studies have addressed this issue by using PPM with approximated context and have reported promising results [7], [8].

In summary, the existing variants of the PPM scheme are not well-suited for digital images since they don't form contexts using pixels from all directions and contexts should not be compared exactly, but their closeness should be measured. This paper attempts to make improvements over the original PPM scheme in order to eliminate these shortcomings. The context is derived by looking at pixels from four directions which would yield better context. Also a degree of approximation is used while determining the context by quantizing the range of pixel values. These issues has been addressed in Section II. The experiments performed in this paper are described in Section III and the findings are summarized in Section IV.

II. Improvements

A. Two-dimensional context

To travel through pixels, instead of the traditional raster scan, we followed a pyramid-like fashion as proposed by Sloan and Tanimoto [10]. A thorough description of pyramid-based image representation can be found at a survey conducted by Tzou [11]. In this representation, an image is divided into a tree of sub-sampled images. For an odd numbered level in the tree, sub-sampling is done by removing every even numbered pixel from every odd numbered row and every odd numbered pixel from every even numbered row. For an even numbered level, sub-sampling is done by removing every even numbered row. This process can be repeated continuously for the sub-sampled image resulting a number of levels of sub-sampled images. Figure 1 shows the sub-sampling process for the first two levels. The removed pixels in the sub-sampling process always have four pixels from the remaining ones surrounding it from four directions (except for the ones at the image borders that require special treatment).

The encoder starts with the coarsest sub-sampled image and encodes the pixels removed in that level by treating the four surrounding pixels as the context for the pixel to be encoded. On completion, the encoder goes to one level above and continues until the topmost level is encoded. Figure 2 shows the location of context-pixels for the first two levels. The decoder, in the same fashion, starts from the coarsest level, reconstructs the image one level above by decoding the removed pixels and continues until the entire image is reconstructed.

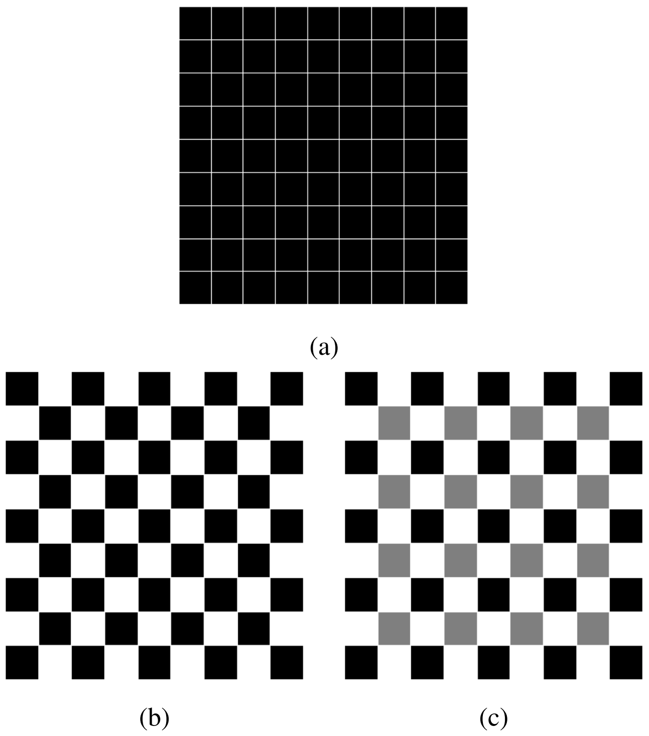

Fig. 1. Sub-sampling in different levels: (a) initial arrangement of pixels; (b) white pixels are removed in the first level; and (c) gray pixels are removed in the second level

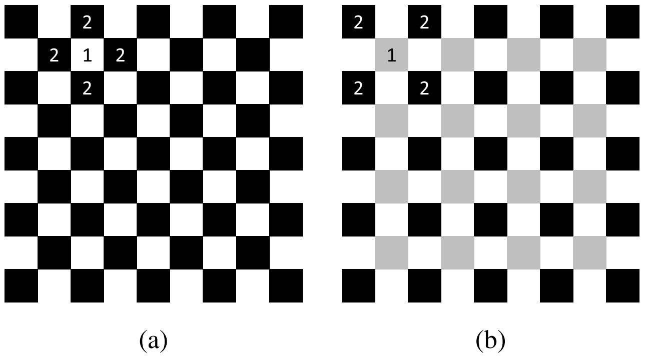

Fig. 2. Location of context pixels in: (a) an odd numbered level; and (b) an even numbered level. The pixels to be encoded are labeled 1 and the pixels labeled 2 are used for encoding them.

B. Context quantization

To reduce the number of escape symbols while encoding, multiple contexts are merged into groups and contexts within a group share a common frequency table for symbols. This is done by quantizing the context pixels into groups. Four pixels from the context are sorted in ascending order. The range of pixel values is quantized and a number of contexts are merged into several groups. For example, if pixel values are quantized into two levels (0 to 127 and 128 to 255), then contexts from (0, 0, 0, 0) to (127, 127, 127, 127) will be mapped into the same group and contexts from (0, 0, 0, 128) to (127, 127, 127, 255) will be mapped into another. In this example, there will be exactly 16 different groups of contexts at the 4th order,

TABLE I
AVERAGE BITS-PER-PIXEL AT VARIOUS QUANTIZATION LEVELS FOR PPMA, PPMB AND PPMC

quantization levels	bpp		
	PPMA	PPMB	PPMC
2	5.83	5.84	5.83
4	5.40	5.27	5.25
8	5.26	4.89	4.82
16	5.35	4.75	4.59
32	5.55	4.83	**4.57**
64	5.75	5.00	4.71
128	5.93	5.20	4.90
256	6.07	5.36	5.07

where all the contexts in a single group will share a common frequency table.

If too many quantization levels are used, the number of escape symbols increase proportionally. On the other hand, if only a few quantization levels are used, the number of different contexts will be too small to yield compression. The optimum level of quantization lies between these two extremes.

III. EXPERIMENTS AND RESULTS

Both the encoder and the decoder are implemented in C. A practical implementation of the original PPM is made available by Mark Nelson and has been used in this research [12].

We tried various quantization levels on a set of 27 training images where quantization levels are in the form $2^n, n \in [1,8]$. The corresponding weighted average bits-per-pixels (bpp) for different quantization levels are recorded. This process is repeated for PPMA, PPMB and PPMC. Table I reflects the result from this experiment. Figure 3 summarizes this result in a plot where the X axis represents different quantization levels and the corresponding bpp is placed along the Y axis. All of the plots show that bpp is comparatively higher at the beginning, reaches an optimum level as the number of quantization levels is increased and then reaches the extreme as the the number of quantization levels is further increased. PPMA performed the best with 8 quantization levels, PPMB with 16 levels and PPMC with 32 levels, revealing the fact that among these three methods, PPMC performed the best (with 32 levels). For this reason, this particular combination is used in the rest of the experiments in this research.

The original PPM code and proposed method are tested on 18 standard test images, as shown in Figure 4. The set of training images used for determining the optimum quantization level is not included in the set of test images. For both methods, test images are encoded into compressed files and then decoded into images again. Each original image and decoded image are compared to each other to ensure that the integrity of the information was preserved. The maximum order is set to 4 for both methods since the number of immediate surrounding pixels can be 4 at most. Both methods use

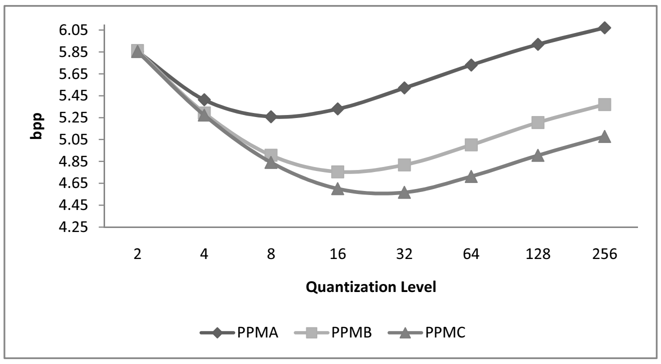

Fig. 3. Average bits-per-pixel at various quantization levels for PPMA, PPMB and PPMC

TABLE II
PERFORMANCE OF BOTH METHODS ON STANDARD TEST IMAGES

Image	Width	Height	bpp		Improvement	
			PPM	Proposed	bpp	%
baboon	512	512	6.86	**6.77**	0.09	1.31
barbara	720	580	6.35	**5.44**	0.91	14.33
boats	720	576	5.23	**4.54**	0.69	13.19
bridge	512	512	4.49	**4.06**	0.43	9.58
camera	256	256	5.38	**5.05**	0.33	6.13
columbia	480	480	4.71	**4.48**	0.23	4.88
couple	512	512	5.60	**5.04**	0.56	10.00
crowd	512	512	5.02	**4.52**	0.50	9.96
goldhill	720	576	5.68	**4.93**	0.75	13.20
lake	512	512	6.19	**5.49**	0.70	11.31
lax	512	512	6.81	**6.16**	0.65	9.54
lena	512	512	5.40	**4.38**	1.02	18.89
man	512	512	5.78	**4.91**	0.87	15.05
milkdrop	512	512	4.77	**3.97**	0.80	16.77
peppers	512	512	5.86	**4.90**	0.96	16.38
plane	512	512	4.95	**4.40**	0.55	11.11
woman1	512	512	5.30	**4.41**	0.89	16.79
woman2	512	512	4.48	**3.72**	0.76	16.96
Average			5.53	**4.85**	0.68	12.30

PPMC for estimating the escape probability and the number of quantization levels is set to 32 for the proposed method; see Table I. Table II shows the compression performance of both methods on the 18 standard test images. The proposed method performs better than the traditional PPM for each image out of the 18 test images. The weighted average bits-per-pixels for both methods show that the average compression ratio of the proposed method is better than that of the PPM method by 0.68 bits-per-pixel which is over 12% improvement in terms of compression performance.

IV. CONCLUSION

The proposed PPM with two-dimensional approximate context method outperforms the original one-dimensional PPM in all test cases. Our method should work with any PPM method to further improve it.

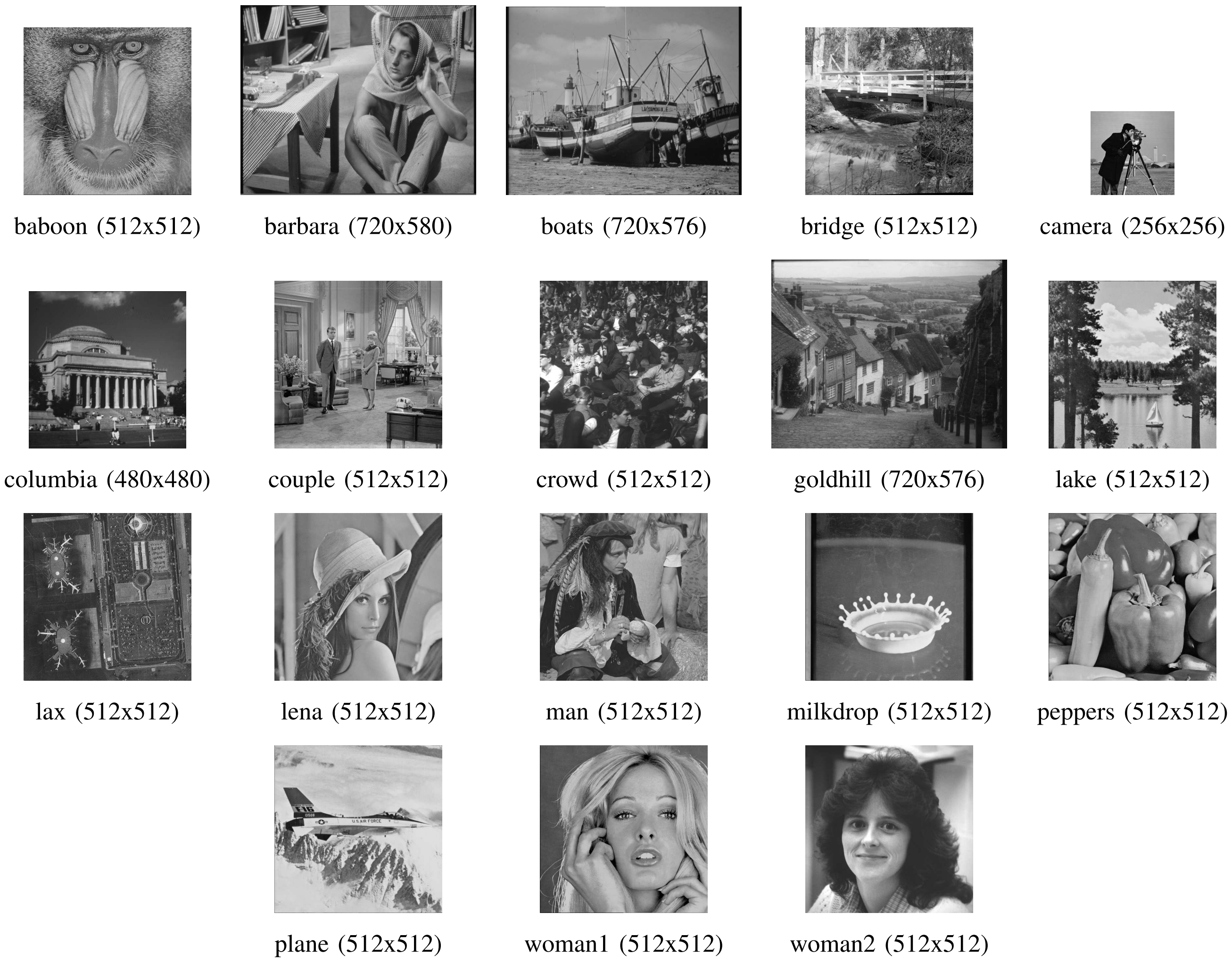

Fig. 4. Standard test images

ACKNOWLEDGMENT

The authors would like to acknowledge Mark Nelson (http://marknelson.us/about/) for preparing an excellent implementation of the original PPM scheme and making it publicly available on the internet.

REFERENCES

[1] J. Cleary and I. Witten, "Data compression using adaptive coding and partial string matching," *Communications, IEEE Transactions on*, vol. 32, no. 4, pp. 396–402, 1984.

[2] K. Sayood, *Introduction to data compression*. Elsevier, 2006. [Online]. Available: http://books.google.ca/books?id=044wLaqZ8twC

[3] F. Choong, M. Reaz, T. Chin, and F. Mohd-Yasin, "Design and implementation of a data compression scheme: A partial matching approach," in *Computer Graphics, Imaging and Visualisation, 2006 International Conference on*, 2006, pp. 150–155.

[4] A. Moffat, "Implementing the PPM data compression scheme," *Communications, IEEE Transactions on*, vol. 38, no. 11, pp. 1917–1921, 1990.

[5] I. Witten and T. Bell, "The zero-frequency problem: estimating the probabilities of novel events in adaptive text compression," *Information Theory, IEEE Transactions on*, vol. 37, no. 4, pp. 1085–1094, 1991.

[6] J. G. Cleary and W. J. Teahan, "Unbounded length contexts for PPM," *The Computer Journal*, vol. 40, no. 2 and 3, pp. 67–75, 1997. [Online]. Available: http://comjnl.oxfordjournals.org/content/40/2_and_3/67.abstract

[7] P. G. Howard, "The design and analysis of efficient lossless data compression systems," Ph.D. dissertation, Providence, RI, USA, 1993, uMI Order No. GAX94-06956.

[8] S. Forchhammer and J. Salinas, "Progressive coding of palette images and digital maps," in *Data Compression Conference, 2002. Proceedings. DCC 2002*, 2002, pp. 362–371.

[9] Y. Zhang and D. Adjeroh, "Prediction by partial approximate matching for lossless image compression," *Image Processing, IEEE Transactions on*, vol. 17, no. 6, pp. 924–935, 2008.

[10] K. Sloan and S. Tanimoto, "Progressive refinement of raster images," *Computers, IEEE Transactions on*, vol. C-28, no. 11, pp. 871–874, 1979.

[11] K. Tzou, *Progressive image transmission: a review and comparison of techniques*, ser. Optical engineering. Bellingham, WA, ETATS-UNIS: Society of Photo-Optical Instrumentation Engineers, 1987, vol. 26, undefined Anglais.

[12] (1991, Feb.) Arithmetic coding + statistical modeling = data compression. [Online]. Available: http://marknelson.us/1991/02/01/arithmetic-coding-statistical-modeling-data-compression/