LOSSLESS IMAGE COMPRESSION WITH TREE CODING OF MAGNITUDE LEVELS

Hua Cai and Jiang Li

Media Communication Group, Microsoft Research Asia, Beijing, China

ABSTRACT

With the rapid development of digital technology in consumer electronics, the demand to preserve raw image data for further editing or repeated compression is increasing. Traditional lossless image coders usually consist of computationally intensive modeling and entropy coding phases, therefore might not be suitable to mobile devices or scenarios with a strict real-time requirement. This paper presents a new image coding algorithm based on a simple architecture that is easy to model and encode the residual samples. In the proposed algorithm, each residual sample is separated into three parts: (1) a sign value, (2) a magnitude value, and (3) a magnitude level. A tree structure is then used to organize the magnitude levels. By simply coding the tree and the other two parts without any complicated modeling and entropy coding, good performance can be achieved with very low computational cost in the binary-uncoded mode. Moreover, with the aid of context-based arithmetic coding, the magnitude values are further compressed in the arithmetic-coded mode. This gives close performance to JPEG-LS and JPEG2000.

1. INTRODUCTION

With the rapid development of digital technology in consumer electronics, the demand to preserve raw image data is increasing. Lossless image compression is highly desired in applications where the pictures are subject to further processing, intensive editing, or repeated compression/decompression. It is generally the choice also for images obtained at great cost, or in applications where the desired quality of the rendered image is yet unknown. Thus, medical imaging, prepress industry, image archival systems, precious artworks to be preserved, remotely sensed images, and professional digital cameras, are all candidates for lossless compression.

Typically, lossless compression is carried out by means of prediction-based approaches, and the residual samples are then fed to an entropy coder. Since these samples exhibit some residual correlation, state-of-the-art lossless image compression schemes adopt two distinct and independent phases, *modeling* and *coding*, to code them [1]. The modeling phase is aimed at gathering information about the image data, in the form of a probabilistic model, which is used for coding. Entropy coding is then used to compress the data based on the model. Successful modern lossless image coders, such as the Sunset algorithm [2], the JPEG-LS standard [3], the CALIC coder [4], the JPEG2000 coder [5], and some other sophisticated coders [6][7], are all based on the above modeling/coding paradigm. On the other hand, although the modeling/coding architecture can achieve very high coding efficiency, it often results in high computational cost. This is because the analysis of statistical characteristics performed in the modeling phase is computationally intensive. Also, computation is required in the entropy coding phase, especially when an arithmetic coding engine is adopted.

In addition to the compression efficiency that the above algorithms try to improve, computational complexity are also very important for practical applications. For example, in the lossless JPEG standard [8], two different schemes are specified: one using arithmetic coding, and one for Huffman coding. Even though the compression gap between the Huffman and the arithmetic coding version is significant, the latter did not achieve widespread use, probably due to its higher complexity requirements [1]. Although the above-mentioned schemes can be simplified to reduce the coding complexity, the resultant complexity might be still too high for some application scenarios such as doing compression in mobile devices or realtime compression of image sequence.

This paper presents a new image coding algorithm based on a simple architecture that is easy to model and encode the residual samples. In the proposed algorithm, each residual sample is separated into three parts: (1) *a sign value*, (2) *a magnitude value*, and (3) *a magnitude level*. As a result, most of the correlation is moved to the magnitude level part, which is then organized with a hierarchical tree structure. By simply coding the tree and the other two parts without any complicated modeling and entropy coding, good performance can be achieved with very low computational cost in the *binary-uncoded* mode. Moreover, with the aid of context-based arithmetic coding, the magnitude values are further compressed in the *arithmetic-coded* mode. This gives close performance to the JPEG-LS and JPEG2000 standards. The proposed algorithm is also used for compressing the Bayer color image sequence [9] with very low computational cost.

The rest of the paper is organized as follows. Section 2 presents our proposed algorithm, including an overall introduction followed by the descriptions of the three main modules. Some experimental results are reported and discussed in Section 3. Some conclusions are drawn in Section 4.

2. PROPOSED ALGORITHM

2.1. System-Level Description

Fig. 1 illustrates the system architecture of our proposed algorithm, which consists of three main modules: a predictor, a tree constructor, and three bitstream coders. In the predictor module, each of the two-dimensional image samples, $X_{(x,y)}$, where (x, y) denotes the coordinates of a certain sample, is first decorrelated to remove the spatial redundancy. Next, each prediction residual sample, $C_{(x,y)}$, is separated into three parts: a sign value $S_{(x,y)}$, a magnitude value $|C_{(x,y)}|$, and a magnitude level $L_{(x,y)}$ that satisfies $2^{L_{(x,y)}-1} < |C_{(x,y)}| \leq 2^{L_{(x,y)}}$. In this way, most of the statistical correlation of $C_{(x,y)}$ is moved to $L_{(x,y)}$, while the remaining $S_{(x,y)}$ and $|C_{(x,y)}|$ are less correlated. This clearly differentiates our method from some other schemes that model the statistical correlation of $C_{(x,y)}$ or $|C_{(x,y)}|$ directly.

0-7803-9332-5/05/$20.00 ©2005 IEEE

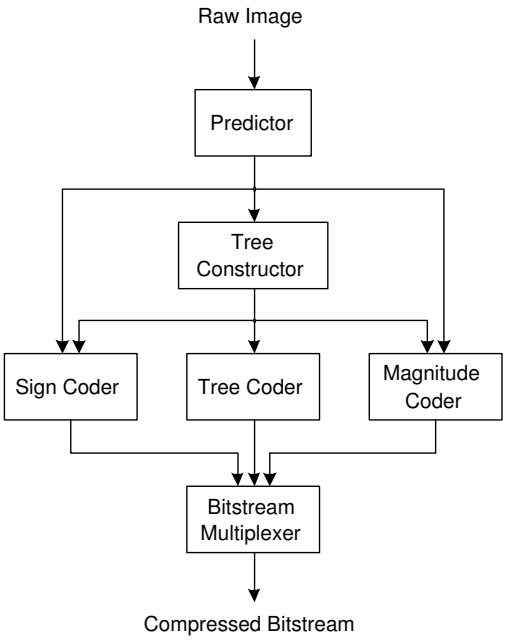

Figure 1: System architecture of the proposed coder.

The tree constructor module is then used to represent $L_{(x,y)}$ using a hierarchical structure. Since $C_{(x,y)}$ and $|C_{(x,y)}|$ are spatially correlated, similar correlation can be found in $L_{(x,y)}$ as well: the surrounding magnitude levels of a small $L_{(x,y)}$ are usually small, whereas the surrounding magnitude levels of a large $L_{(x,y)}$ have higher probability to be large. Therefore, a tree structure that hierarchically represents $L_{(x,y)}$ can efficiently compress $L_{(x,y)}$ even without entropy coding. The constructed tree is then truncated to a certain depth and then coded. Note that because of the tree truncation, the actual coded magnitude levels $\hat{L}_{(x,y)}$ might be different from $L_{(x,y)}$: $\hat{L}_{(x,y)} \geq L_{(x,y)}$.

Based on the tree coding results $\hat{L}_{(x,y)}$, $S_{(x,y)}$ and $|C_{(x,y)}|$ are then coded. To adapt to different complexity requirements, two coding modes are used in this paper when generating the compressed bitstream. In the *binary-uncoded* mode, all of the bits are coded in a binary manner without any entropy coding. Thus very fast encoding speed can be achieved. In the *arithmetic-coded* mode, $|C_{(x,y)}|$ is coded with context-based arithmetic coding. Thus higher coding efficiency can be attained with a cost of increased computation.

In the rest of this section, we will present the predictor, the tree constructor and coder, and the sign and magnitude value coders.

2.2. Prediction

Predictor plays an important role in a lossless image compression system. Although sophisticated prediction methods could significantly improve the compression efficiency [7], they often require high computation. Therefore, due to the complexity consideration, only simple prediction method is chosen in this paper.

As shown in Fig. 2, three neighboring samples, $X_{(x-1,y)}$, $X_{(x,y-1)}$ and $X_{(x-1,y-1)}$, are used for predicting the current sample $X_{(x,y)}$. To adapt to different texture features, the image samples are first partitioned into 8×8 sample blocks, each sample

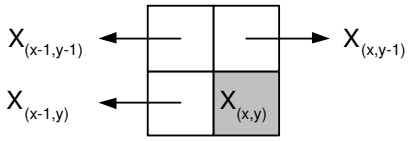

Figure 2: Neighboring pixels used for prediction.

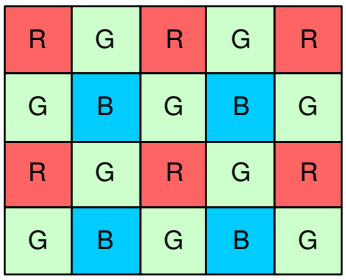

Figure 3: Bayer patterned color filter array.

block can then choose one of the following four prediction modes:

$$\begin{aligned}
&\text{mode 1: } C_{(x,y)} = X_{(x,y)} - X_{(x-1,y)} \\
&\text{mode 2: } C_{(x,y)} = X_{(x,y)} - X_{(x,y-1)} \\
&\text{mode 3: } C_{(x,y)} = X_{(x,y)} - X_{(x-1,y-1)} \\
&\text{mode 4: } C_{(x,y)} = X_{(x,y)} - 0.5 \times (X_{(x-1,y)} + X_{(x,y-1)}).
\end{aligned} \tag{1}$$

And two bits are used to signal the prediction mode of each block, which result in an additional 0.03125 bit/pixel overhead. Moreover, when determining the prediction mode for each sample block, two different criteria are used for the binary-uncoded mode and the arithmetic-coded mode, respectively. In the binary-uncoded mode, the best prediction mode is defined as the one that minimizes the following cost function [1]:

$$Cost = \sum_{i=0}^{3}\sum_{j=0}^{3} \max\{L_{(2i,2j)}, L_{(2i+1,2j)}, L_{(2i,2j+1)}, L_{(2i+1,2j+1)}\}. \tag{2}$$

And in the arithmetic-coded mode, we define the best prediction mode as the one that minimizes the sum of the absolute difference (SAD):

$$SAD = \sum_{i=0}^{7}\sum_{j=0}^{7} |C_{(i,j)}|. \tag{3}$$

The above prediction method is used for grayscale images and color images that have separated Y, U, and V components. As for the Bayer color images [9], which are the outputs of many consumer digital color cameras that use a single sensor with a color filter array (CFA) to capture images (shown in Fig. 3), a different prediction method is used:

$$\begin{aligned}
&\text{R: } C_{(x,y)} = X_{(x,y)} - 0.5 \times (X_{(x-2,y)} + X_{(x,y-2)}) \\
&\text{G: } C_{(x,y)} = X_{(x,y)} - 0.5 \times (X_{(x-1,y-1)} + X_{(x+1,y-1)}) \\
&\text{B: } C_{(x,y)} = X_{(x,y)} - 0.5 \times (X_{(x-2,y)} + X_{(x,y-2)}).
\end{aligned} \tag{4}$$

[1] To simplify representation, we use (i, j), where $i \in [0, 7]$ and $j \in [0, 7]$, to represent the actual coordinates of a certain pixel in a sample block hereinafter this subsection.

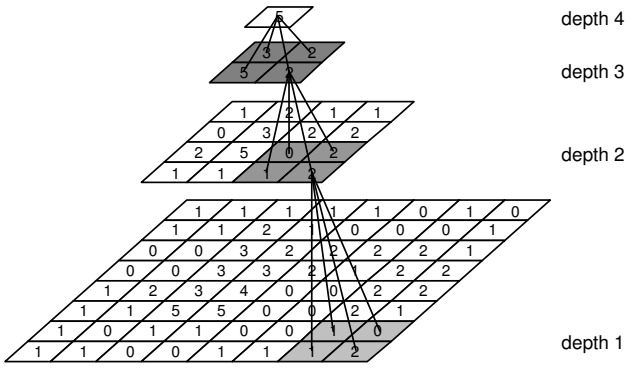

Figure 4: An example of the tree construction.

And no sample partitioning and prediction mode selection will be applied for the Bayer color images in our implementation.

2.3. Tree Construction and Coding

Since $L_{(x,y)}$ is spatially correlated, a hierarchical tree is proposed to remove its redundancy. As shown in Fig. 4, the tree is constructed in a stepwise manner, from the bottom to the top. When the depth increases by one, both the horizontal and vertical resolutions will be reduced by half. A coefficient in the lowest depth, say $T^1_{(x,y)}$, equals to the corresponding magnitude level $L_{(x,y)}$. For other coefficients in the i^{th} ($i > 1$) depth, they can be derived from their children coefficients:

$$T^i_{(x,y)} = \max\{T^{i-1}_{(2x,2y)}, T^{i-1}_{(2x+1,2y)}, T^{i-1}_{(2x,2y+1)}, T^{i-1}_{(2x+1,2y+1)}\}. \quad (5)$$

Note that there is a tradeoff between the tree coding and the $S_{(x,y)}$ and $|C_{(x,y)}|$ coding. When more bits are used in the tree coding, fewer bits will be required in the $S_{(x,y)}$ and $|C_{(x,y)}|$ coding. Therefore, before coding the above hierarchical tree, it is necessary to determine to which depth the tree is coded. In this paper, two different truncation depths are used for different coding modes. In the binary-uncoded mode, the tree is truncated to the second depth; whereas in the arithmetic-coded mode, the tree is truncated to the third depth.

Finally, the truncated tree is coded simply by travelling the coefficients one by one from the top depth all the way down to the truncated depth. Coefficients are compared with their corresponding parent coefficients of higher depth, and binary values are output to form the final bitstream without any entropy coding. The method of coding a D-depth tree, which is truncated to the d_0^{th} depth, is summarized below:

```
d = D
while d ≥ d_0 {
    for all T^d_(x,y) {
        while T^d_(x,y) < T^{d+1}_(x,y) {
            output 1;
            T^{d+1}_(x,y) = T^{d+1}_(x,y) − 1;
        }
        output 0;
    }
    d = d − 1;
}
```

2.4. Sign and Magnitude Coding

2.4.1. Binary-Uncoded Mode

In the binary-uncoded mode, $S_{(x,y)}$ and $|C_{(x,y)}|$ are simply coded based on $\hat{L}_{(x,y)}$ using the following method:

```
If L̂_(x,y) > 0 {
    Output the least significant bit of S_(x,y);
    Output the L̂_(x,y) number of less significant bits of
    |C_(x,y)|;
}
```

And no entropy coding will be applied on the output bits. This makes the entire algorithm very fast.

2.4.2. Arithmetic-Coded Mode

In the arithmetic-coded mode, context-based arithmetic coding is utilized to further improve the coding efficiency. Although most of correlation is moved to $L_{(x,y)}$ after separating the residual coefficients, $|C_{(x,y)}|$ still contains a certain degree of correlation since only a truncated version of $L_{(x,y)}$ is coded. Thus, $|C_{(x,y)}|$ can be compressed more efficiently with the aid of entropy coding. On the other hand, little correlation is found in $S_{(x,y)}$. Hence, $S_{(x,y)}$ is coded with the same method as that in the binary-uncoded mode.

When encoding $|C_{(x,y)}|$, the $\hat{L}_{(x,y)}$ number of less significant bits of $|C_{(x,y)}|$ are first represented by its bit-plane representation, bit in each bit-plane is then modelled by a context template that includes: (1) the level of the current bit-plane; (2) the significance of $|C_{(x-1,y)}|$ and $|C_{(x,y-1)}|$ with respect to the current bit-plane; and (3) the significance of $|C_{(x,y)}|$ with respect to the previous bit-plane. The found context is then fed to a QM coding engine together with the corresponding binary bit of $|C_{(x,y)}|$ in the coded bit-plane. Context-based arithmetic coding is thus achieved.

3. EXPERIMENTAL RESULTS AND DISCUSSIONS

To test comprehensively the performance of our proposed coder, a rich test set including twelve images (monochrome, 8 bits/pixel) of a wide variety of features (shown in Fig. 5) are used in our experiments. The LOCO-I coder [10], which is the core of the JPEG-LS standard, and the JPEG2000 coder [11], are used for comparison.

In Table 1, we study the compression performance of our proposed algorithm, in comparison with LOCO-I and JPEG2000. For some images, our arithmetic-coded bitstream achieves better performance than LOCO-I and JPEG2000, while for other images, our performance is slightly worse than them. The average performance of the arithmetic-coded bitstreams slightly outperforms the above two coders. Moreover, the performance degradation of the binary-uncoded bitstream is limited to 0.4 bit/pixel on average.

The proposed coder is also used for compressing a 300-frames Bayer color image sequence captured by a single CCD camera. Each frame is encoded independently as a single image using the binary-uncoded mode. The image resolution is 640×480, and each pixel has 8 bits to represent its color value. As can be seen from Fig. 6, the 100^{th} frame of the test sequence, the test sequence includes a wall, a color pattern, and two people, which makes the prediction and coding difficult. When it is coded on a PC of a 2.8GHz Pentium CPU with 512M RAM, over 50 frames/sec is achieved. The ultimate bit rate is 4.572 bits/pixel on average.

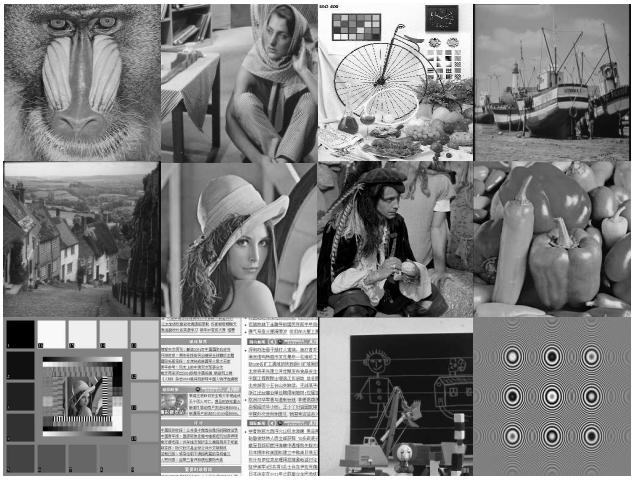

Figure 5: Test image set, the name of the images are (from left to right and from top to bottom): baboon (512×512), barbara (512×512), bike (2048×2560), boats (720×576), goldhill (720×576), lena (512×512), man (768×768), peppers (512×512), testpat (768×768), text (512×512), toys (512×512) and zone-plate (1024×768).

Moreover, as the binary-uncoded mode of the proposed coder abandons the modeling and entropy coding phases that are widely adopted by other coders, the entire algorithm becomes very simple. Therefore, we believe that, a faster encoding speed is achievable by proper source code optimization.

Table 1: Compression results on the test set (in bits/pixel).

Image	binary -uncoded	arithmetic -coded	LOCO-I	JPEG 2000
baboon	6.190	6.112	5.970	6.034
barbara	5.171	5.061	4.864	4.785
bike	4.692	4.585	4.356	4.528
boats	4.439	4.302	3.933	4.066
goldhill	4.860	4.732	4.477	4.603
lena	4.652	4.517	4.245	4.315
man	5.267	4.514	4.749	5.229
peppers	4.890	4.765	4.438	4.530
testpat	1.923	1.139	1.968	1.938
text	3.238	1.773	2.596	4.296
toys	4.412	4.253	3.951	4.094
zone-plate	7.662	7.188	7.542	5.930
Average	4.783	4.412	4.424	4.529

4. CONCLUSIONS

This paper presents a new image coding algorithm based on a simple architecture that is easy to model and encode the residual samples. The key idea is to separate each residual sample into three parts, and then encode the magnitude levels by a hierarchical tree structure. Moreover, two coding modes are proposed to adapt to different complexity requirements. Close performance to the JPEG-LS and JPEG2000 standards is achieved on a rich test set.

Figure 6: The 100^{th} frame of the test Bayer color image sequence.

5. REFERENCES

[1] B. Carpentieri, M. J. Weinberger, and G. Seroussi, "Lossless compression of continuous-tone images," *Proceedings of the IEEE*, vol. 88, pp. 1797–1809, Nov. 2000.

[2] S. Todd, G. G. Langdon Jr., and J. Rissanen, "Parameter reduction and context selection for compression of the gray-scale images," *IBM J. Res. Develop.*, vol. 29, pp. 188–193, Mar. 1985.

[3] ISO/IEC 14495-1, "Information technology–Lossless and near-lossless compression of continuous-tone still images," *ITU Recommendation T.87*, 1999.

[4] X. Wu and N. D. Memon, "Context-based, adaptive, lossless image coding," *IEEE Transactions on Communications*, vol. 45, pp. 437–444, Apr. 1997.

[5] ISO/IEC 15444-1, "Information technology–JPEG2000–Image coding system–Part 1: Core coding system," 2000.

[6] X. Wu, "Lossless compression of continuous-tone images via context selection, quantization, and modeling," *IEEE Transactions on Image Processing*, vol. 6, pp. 656–664, May 1997.

[7] X. Li and M. T. Orchard, "Edge-directed prediction for lossless compression of natural images," *IEEE Transactions on Image Processing*, vol. 10, pp. 813–817, June 2001.

[8] ISO/IEC 10918-1, "Digital compression and coding of continuous tone still images - Requirements and guildlines," *ITU Recommendation T.81*, Sept. 1993.

[9] Bryce E. Bayer, "Color imaging array," 1976.

[10] M. J. Weinberger, G. Seroussi, and G. Sapiro, "LOCO-I: A low complexity, context-based, lossless image compression algorithm," *Proc. 1996 Data Compression Conference*, pp. 140–149, Mar. 1996.

[11] ISO/IEC JTC1/SC29/WG1, "JPEG2000 Verification Model 7.2," May 2000.