ISSN: 2348 9510
International Journal Of Core Engineering & Management (IJCEM)
Volume 2, Issue 1, April 2015

Image Compression for Mobile Devices using Prediction and Direct Coding Approach

Joshua Rajah Devadason
M.E. scholar,
CIT
Coimbatore, India

Mr. T. Ramraj
Assistant Professor,
CIT
Coimbatore, India

Abstract

Images are used in various applications. With the rapid development of digital technology in consumer electronics, the demand to preserve raw image data for further editing is increasing. Domains such as medical image processing, military, image printing, and land terrain analysis require the complete original image to be present. Traditional lossless image coders usually consist of computationally intensive modelling and entropy coding phases, therefore might not be suitable to mobile devices or scenarios with a strict real-time requirement. A new image coding algorithm is developed based on a simple architecture that is easy to model and encode the residual samples.

In the proposed algorithm, the image is first split into blocks of equal sizes. Then a prediction method is applied for the block. The available prediction methods are horizontal-wrap, vertical-wrap, random and diagonal. After the prediction, the blocks are encoded using one of the available schemes. The encoding schemes are truncate eight, difference and direct code. In truncate eight, if the encoded pixel length exceeds eight bits, it is discarded. Instead a code of '10' is added to indicate that the pixel can be directly encoded. In difference mode, both positive difference and negative difference are considered when encoding. Results show that the algorithm produces a compression efficiency of 6.4 bits/pixel.

Index Terms—Compression, prediction, encoding

I. Introduction

Image compression is representing the image in a smaller size. This is done to store the image in a more efficient manner to save space. Also the processing of the image is quicker because of the smaller size. This is very important in the case of mobile devices. Mobile devices have limited memory and processing power. Hence a compressed image performs better and uses lesser resources when compared to an uncompressed image.

There are two types of image compression. The first is lossy image compression; the next is lossless image compression. Lossy image compression is used in most frequent applications such as digital cameras and the images on the internet. Example of lossy image compression is the JPEG image compression. Lossless image compression is highly desired in applications where the pictures are subject to further processing. The image is compressed with significant information loss from the original image. The original image cannot be reconstructed from the compressed image.

Lossy image compression is not desired in applications such as archival image management, medical image processing, geographical image processing. In applications such as these, the entire detail of the image is required as further processing is done on the original image. Hence a lossless image compression scheme is developed. This paper deals with developing a technique that performs lossless image compression on the image. Using this technique, the complete original image can be reconstructed from the original image.

Traditional lossless image coders usually consist of computationally intensive modelling and entropy coding phases, therefore might not be suitable to mobile devices or scenarios with a strict real-time requirement. 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, modelling and coding, to code them.

The new technique consists of two steps. The first step is the prediction phase. The next phase is the encoding phase. In the prediction phase, three prediction modes can be used; horizontal-wrap, vertical-wrap, and random. In the horizontal-wrap mode, the prediction is done based on the neighboring horizontal pixel, additionally the edge pixels is also taken into consideration when performing the prediction. In the random mode, the order of the prediction is performed with a predefined random pixel. The next pixel is not necessarily the neighboring pixel.

After the prediction is performed, the residual sample is fed into the encoding phase. Here three modes are available; the truncate eight, difference and direct code. In truncate eight, if the encoded pixel length exceeds eight bits, it is discarded. Instead a code of ‘10’ is added to indicate that the pixel can be directly encoded. In difference mode, both positive difference and negative difference are considered when encoding. A code of ‘01’ indicates that the encoded difference is negative. In direct coding, the magnitude level of the pixel is encoded directly, however if the magnitude level is greater than eight, it is discarded and the original magnitude level is encoded.

II. Related Work

Said and Pearlman [1] developed S+P Transform based compression. The algorithm is suited for both lossless and lossy compression. This transform requires only integer addition and bit shift operation. By using careful scaling and truncation, the number of bits used to represent the transformed image is kept low. The S transformation truncation removes the redundancy in the least significant bit. Instead of performing the prediction as a post transform operation, Said and Pearlman developed a scheme which predicts the high frequency components from a combination of computed low and high frequency components. This transformation yields better results than JPEG LS.

Calderbank et al. [2], developed the wavelet transform which is very popular for lossy image compression. In wavelet transformation based compression, the input sequence is split into even and odd samples. It is followed by a repeated alternate application of lifting and dual lifting steps. Lifting steps involves applying a filter to the odd samples and subtracting the result from even samples. Lifting makes it possible to obtain an integer transform simply by using truncation without sacrificing invertibility. The dual lifting step does the opposite, it involves applying filter to the even samples and subtracting the result from the odd samples. After N such application of dual and primal lifting operations, the even and odd samples produce the low and high pass coefficients respectively. The inverse transform can be obtained by simply reversing the operation and flipping the signs. The lifting scheme implementation of the wavelet transform has numerous advantages over the traditional wavelet transform.

Wu and Memon [3] developed CALIC (Context Based Adaptive Lossless Image Coder). Basically, it is a lossy+residual approach. In the first stage, the algorithm uses a gradient-based non-linear prediction to get a lossy image and a residual image. In the second stage, it uses arithmetic coding to encode the residuals based on the conditional probability of the symbols in different contexts. CALIC also has a mechanism to automatically trigger a binary mode which is used to code either uniform or binary subimages or both. CALIC uses a nonlinear predictor that adapts itself to the image gradient near the predicted pixel. This predictor improves the precision of traditional DPCM predictors, particularly in areas of sharp edges.

The JPEG standard specifies coding schemes with Huffman and arithmetic encoding [4]. The JPEG lossless mode uses a predictive coding technique. The prediction is formed using previously encoded pixels in the current row of the image. The algorithm is efficiently mapped to the hardware because of the simplicity of the algorithm. There exist several methods for prediction. It uses a predictive scheme based on the three nearest (causal) neighbours (upper, left, and upper-left), and entropy coding is used on the prediction error.

In LOCO-I [5], context definition is based on the local image gradients allowing one to capture the level of activity surrounding a pixel. Lempel and Ziv and Welch [6] proposed the LZW coding. The LZW coding uses an adaptive coding method that does not require the data that is to be compressed to be available at the start. Rather, their technique generates codes as it examines the source file from beginning to end. In Huffman coding, each symbol of the uncompressed data is replaced by a code [7]. The symbol codes are of variable length and symbols that occur more frequently in the uncompressed data have codes of smaller length that symbols that occur less frequently.

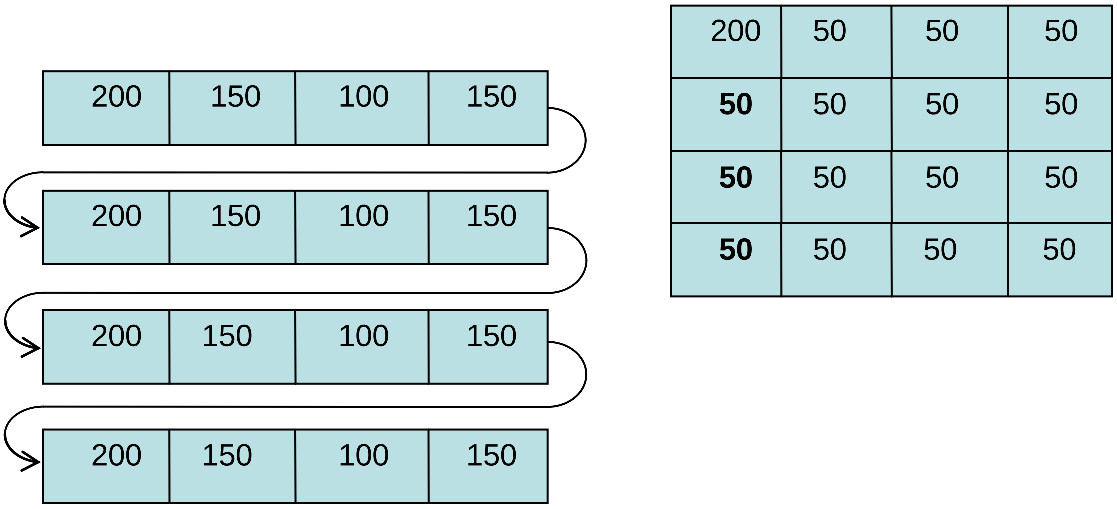
III. Proposed Algorithm

The proposed algorithm consists of two phases; the prediction phase and encoding phase. The prediction phase reads the input image. It converts the image into a local two dimensional array data structure. The image is then split into 8x8 image blocks. Four prediction modes are applied on the split blocks and the best prediction is chosen and passed onto the encoding phase. In the encoding phase, the magnitude level is encoded as a binary bit stream. The method encodes the value as a bit 1's when the value is less than eight.

A. Prediction Phase

In the prediction phase, four prediction modes can be used; horizontal-wrap, vertical-wrap, and random. To adapt to different texture features, the image samples are first partitioned into 8 X 8 sample blocks, each sample block can then choose one of the following four prediction modes. The sum of absolute difference is found for each of the predicted block and the lowest is chosen as the ideal prediction.

For the horizontal-warp prediction, the prediction is based on the neighbouring pixel. When the end pixel is reached, the pixel in the next row is used for performing the prediction.



200	150	100	150
200	150	100	150
200	150	100	150
200	150	100	150

200	50	50	50
50	50	50	50
50	50	50	50
50	50	50	50

Fig 1: Diagram showing the horizontal wrap around

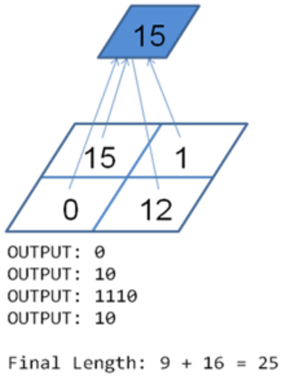
The vertical prediction is performed similarly to the horizontal prediction, except that the vertical neighbor is considered when predicting the image. When the edge pixels are reached the pixel in the next vertical column is used for performing the prediction. To perform the random prediction, the neighbor pixels are not used for performing the prediction. Instead, the pixels are traversed in a predetermined random manner, this type of prediction is ideal for block which has a noise like texture.

After the prediction is done, the ideal residual sample along with a two bit scheme to denote the prediction method used is stored. The residual sample is passed along to the encoding module where the residual magnitude levels are encoded.

B. Encoding Phase

Several methods were developed for the encoding phase. A single method can be chosen when performing the compression. The methods are truncate eight, direct code, and other methods. In the truncate eight method, the highest among the block is found, and if the difference is lesser than eight, it is encoded. In the direct code method, the magnitude levels are encoded directly without finding the highest. As with the truncate eight, if the value of the magnitude is greater than eight, it is discarded and the original value is encoded. In other methods, the difference sign method finds the difference between two pixels and the difference is encoded.

In truncate eight method, first the magnitude levels are split into blocks of four. Next, the highest value among the four is found. The difference between the highest and the magnitude levels is found one by one. The difference is encoded as a sequence of 1’s. However, if the code for the pixel exceeds a length of eight bits. The new code is discarded. Instead a code of ’10’ is added to the output stream to denote that it is discarded. Later the original magnitude level is encoded directly in the bitstream.

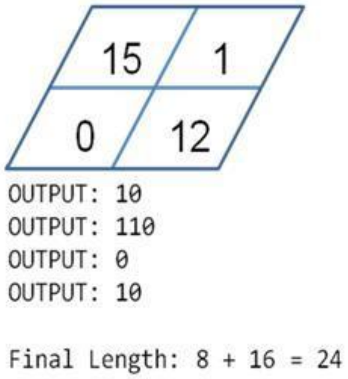


```
OUTPUT: 0
OUTPUT: 10
OUTPUT: 1110
OUTPUT: 10

Final Length: 9 + 16 = 25
```

Fig 2: Encoding example for truncate eight method

The direct code method produces the best performance amongst the different methods tried. Here instead of finding the highest and encoding the difference, the pixel is encoded directly. If the encoded pixel length is greater than eight, it is discarded and the pixel value is saved without any encoding. In the example below, there exists a 2x2 predicted block with pixel values 15, 1, 0, 12. The first predicted value to be encoded is 15. Since 15 is greater than eight, encoding it using a sequence of 1s is costly. It is more effective if it is represented as a byte value of 15, hence a value of ‘10’ is encoded to denote deferred byte encoding. Next, the predicted value of 1 is read for encoding. 1 is less than eight, hence a code of ‘110’ is printed at the output stream. Similarly, the rest of the predicted pixels are encoded.



```
OUTPUT: 10
OUTPUT: 110
OUTPUT: 0
OUTPUT: 10

Final Length: 8 + 16 = 24
```

Fig 3: Example coding for direct coding method

Several other methods for encoding were tried. However, the methods did not improve the overall efficiency of the compression process. The method difference sign is a method which finds the difference between the pixels and then encodes the difference. If the difference is greater than eight, it is skipped and a value of zero is entered in the bit-stream. The problem arises because the difference can be negative in value, in this case extra pixels are included in the encoded bitstream to indicate the sign value. The next method is Highest retain. This mode is similar to the truncate eight method. Here instead of resetting the highest pixel between pixel encoding, the highest is retained.

IV. Performance Evaluation

To test the performance of the image compression algorithm, it is run against a number of similar algorithms in its domain. The JPEG-LS and the LOCO-I coder is used for comparison. The images that are used for image compression and comparison are shown in the figure below. The image that are used for compression are pirate (512x512 pixels), house (512x512 pixels), lena (512x512 pixels), and walkbridge (512x512 pixels). The images are given as input one by one to the image compression algorithm, the result is obtained as a file with a bit-stream representation of the input image. The compressed image bit-stream produced by the algorithm is expected to be lesser in size when compared to the original image.

Fig 4: Images used for comparison

To obtain the compression result, the ratio of the compression is obtained. To calculate the compression, the size of the original image is divided by the size of the compressed image. The ratio is then displayed as the efficiency of the image compression.

A. Block sizes parameter change compression

The parameter of block sizes for prediction where changed and tested. When the block sizes changes an improvement in performance is expected. However, if the block size is too small it is expected that the performance would be bad. The reason being that the number of bytes used to indicate the prediction mode will increase. It takes two bits to represent the prediction mode for each block. If the block size is too big, then too it is expected that the performance be poor. When the block size increases, the ability of the prediction to identify the texture property decreases, hence performance decreases.

Table 1: Results evaluation table for different block sizes

Block Size	**8x8**	**16x16**	**256x256**	**4x4**
Compression Percentage	86 %	**83 %**	85 %	95 %
Comments		Increasing the block size results in better compression	Too big block sizes results in degrading performance	Smaller block sizes also results in poor performance

Results show that lower block size and too high block size results in poor performance. The ideal block size is shown as the 16x16 block size.

B. Performance comparison for different prediction modes

The new prediction modes developed are horizontal-wrap, vertical rap and random prediction schemes. A total of four prediction modes are allowed to be tested at a single time. The first column shows a combination of the prediction modes horizontal, vertical, diagonal and method IV. The combination provides a compression ratio of 86 percent. The next test case includes horizontal-wrap, this results in a performance gain of two percent. Test case three includes a prediction combination of vertical, vertical-wrap, diagonal and method IV. This case, like case two before it also produces the same performance gain of two percent over the first test case. The final prediction test case includes a combination of horizontal-wrap, vertical-wrap, random, and diagonal. This results in the best results of 80% compression ratio.

Table 2 : Results evaluation table different prediction modes

Prediction Modes	**Hor, ver, diag, IV**	**Hor, hor-wrap, ver, diag**	**Ver, ver-wrap, diag, IV**	**Hor-wrap, ver-wrap, random, diag**
Compression Percentage	86 %	84 %	84 %	**80 %**
Comments		Including horizontal wrap results in 2% gain.	Including vertical wrap mode saves 2% again.	Including both modes results in best performance.

The test results conclude that the prediction modes horizontal-wrap, vertical-wrap, random and diagonal produces the best compression percentage of 80 percent.

C. Performance comparison for different encoding modes

The modes tested for the performance comparison are the truncate eight, difference and direct code. The truncate eight mode produces an average compression efficiency of 94 percent. The mode encodes magnitude levels that is only of a length lesser than eight. The next method tested is the difference mode. The difference mode produces a compression efficiency of 90 percent. The final mode tested which produces the best compression ratio is the direct code. The direct code produces an efficiency of 86 percent. In this mode the magnitude levels are encoded directly.

Table 3: Results evaluation table different encoding modes

Tree Mode	**Truncate Eight**	**Difference Mode**	**Direct Code**
Compression Percentage	94 %	90 %	**86 %**
Comments	Mode discards code which has length greater than eight.	Performs better than previous. Mode encodes difference.	Predicted pixels are directly encoded.

D. Performance comparison of the proposed algorithm with existing standards

Table 4: Image compression comparison table (in bits/pixel)

Image	**JPEG-LS (bits/pixel)**	**LOCO-I (bits/pixel)**	**Proposed algorithm (bits/pixel)**
Walkbridge	6.19	6.11	7.60
House	3.48	3.42	4.40
Man	5.22	4.70	5.58
Lena	4.30	4.20	6.08

The above table gives a comparison of the proposed algorithm against two other algorithms, the JPEG-LS and LOCO-1. The JPEG-LS performs better than the proposed algorithm for the house image, a value of 3.48 is obtained against a value of 4.40. The JPEG-LS and LOCO-I algorithm performs better than the proposed algorithm for the man image. As for the other image, JPEG-LS and LOCO-I generally outperforms the proposed algorithm. However, JPEG-LS and LOCO-I uses the complex entropy encoding step which is not present in the proposed algorithm. Without the entropy encoding, the proposed algorithm will take up lesser processing time and resources when compared to the existing algorithms. On summary, the proposed algorithm generally perform 15% to 20% worse than existing methods without performing any computationally expensive entropy encoding.

V. Conclusion and scope for future work

- A method for lossless image compression has been proposed by this project work. This enables quick compression of images for mobile devices.
- The algorithm performs 15–20 percent worse than existing standards for image compression without performing complex entropy coding.
- The algorithm uses two modules, the prediction module and the encoding module. In the prediction module, the neighbouring pixels are used for predicting the value of the current pixel. The output of the prediction module is fed to the encoding module. In the encoding module, a mode of direct coding encodes the magnitude level.
- The algorithm was tested with different block sizes, and the ideal block size was found to be a size of 16 pixels.
- Future work includes further improving the compression efficiency, so that the battery life of the mobile device can be enhanced. Also a module for altering the computing efficiency with the capability of the mobile device is to be developed.

REFERENCES

[1] A. Said and W. A. Pearlman (1996), “An Image Multiresolution Representation for Lossless and Lossy Compression”, IEEE Transaction Image Process., vol. 5, no. 9, Sept. 1996.

[2] A. R. Calderbank, I. Daubechies, W. M. Sweldens, and Book-Lock Yeo, “Wavelet transforms that map integers to integers”, Department of Mathematics, Princeton University, August, 1996.

[3] X. Wu, N. Menon, “CALIC- A context based adaptive lossless image codec”, Proceedings of 1996 International Conference on Acoustics, Speech and Signal Processing, pp. 1890-1893, 1996.

[4] V. Bhaskaran and K. Konstantinides, “Image and Video Compression Standards, Algorithms and Architectures”, pp. 47-49, 1995, Kluwer Academic Publishers.

[5] M. J. Weinberger, G. Seroussi, and G. Sapiro, “LOCO-I: A low complexity, context-based lossless image compression algorithm”, Proceedings of 1996 Data Compression Conference, pp. 140-149, 1996

[6] J. Ziv and A. Lempel, “A Universal Algorithm for Sequential Data Compression”, IEEE Transaction on Information Theory, Vol. 24, No. 5, pp. 530-536, 1978.

[7] D. Huffman, “A Method for the Construction of Minimum Redundancy Codes”, Proc. IRE, Vol. 40, pp 1098-1101, 1952