

# A Search-Aware JPEG-LS Variation for Compressed Image Retrieval

Tao Tao, Amar Mukherjee, Ravi Vijaya Satya

School of Computer Science  
University of Central Florida, U.S.A.

## ABSTRACT

*With increasing amount of image data such as the satellite images being stored in the compressed format, efficient retrieval of the images has become a major concern. For satellite images, one typical problem is, given an image pattern, we need to quickly locate its matches in the image. It is highly desired that the compressed images are not to be decompressed when the matching is being performed. This problem is generally referred as the “Two Dimensional Compressed Pattern Matching” problem. In this paper, we report our recent work on compressed pattern matching in JPEG-LS compressed images. The method proposed in this paper is primarily a two-pass variation of the JPEG-LS compression algorithm and the algorithm is made search-aware. Experimental results show that the two-pass compression algorithm, not like other search-aware compression algorithm, does not sacrifice the compression. The proposed approach can be easily adapted for other compression methods such as lossless JPEG and CALIC.*

## 1. INTRODUCTION

As a result of recent developments in multimedia, the amount of image data needs to be stored has grown substantially. Image compression greatly reduces the cost of storage. A large portion of the images, such as the satellite images, medical images and geometric images, need to be stored losslessly. JPEG-LS [8] is the current internal standard for lossless image compression.

With the images being stored in the compressed format, efficient retrieval of image data has become a major concern. A typical image retrieval problem is, given a small portion of the image, or an image pattern, we need to quickly locate all of its (exact or approximate) matches in the image. It is highly desirable that the images are kept in the compressed format while the matching is being performed because the naïve “decompress-and-search” method does not only require unnecessary storage space but also require extra computational time.

During recent decades, Compressed Pattern Matching (CPM) has emerged for efficient information retrieval from the compressed data. CPM is defined as: *given a data file in compressed format and a pattern, report the occurrence(s) of the pattern in the file with minimal (or no) decompression.* As an important branch, two-dimensional CPM has been developed in recent years [1]. In [2] and [3], 2D CPM algorithms on run-length encoding are reported. As one of the most recent works, in [4], an “inplace” 2D CPM algorithm on LZW compression is reported. The algorithm is inplace because it uses  $O(c(P))$  extra space for searching where  $P$  is the pattern and  $c(P)$  is the result of compressing  $P$ . The algorithm in [4] performs the search in time  $O(n^2)$  and it takes  $O(m^3)$  time to preprocess the pattern, assuming the pattern size is  $m^2$  and the image size is  $n^2$ . However, for image data, the run-length compression and the LZW compression are not efficient in terms of compression ratio and are not generally used for images. Thus, the work from [2-4] will not have strong impact on images. In [5], an image compression algorithm that is similar to lossless JPEG was proposed and the related compressed pattern matching approach was studied. It was proposed in [5] to use Bird's [6] (non-compressed) two dimensional pattern matching algorithm on Huffman encoded symbols and the algorithm works well for satellite images except for the case that parts of the pattern matches while the whole does not. It was suggested in [5] to replace Aho-Corasick algorithm in Bird's algorithm with Commentz-Walter algorithm and it was also suggested to replace Bird's algorithm with that from [7]. The work from [5] is very constructive. However, we doubt if it is worthwhile to propose a new compression algorithm but rather than to work on the existing image compression standard, as the latter is more popular and thus the work will have more impact. In [9], a modified JPEG-LS algorithm was proposed and the patterns are searched after the image is partially decompressed. Comparing with the “decompress-and-search” method, the work in [9] has nearly 30% improvement in searching time for most natural images. However, the searching speed in [9] is achieved with a sacrifice of 5-7% compression ratio. Most importantly, the searching speed could be further improved if no decompression is required at all. We will

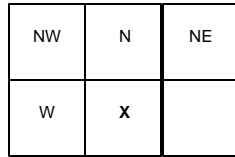
show that the work presented in this paper has addressed the problems in [9].

The final goal of our research is to achieve fully compressed pattern matching based on JPEG-LS. In this paper, we present our initial attempt for this research. We propose a two-pass variation of the JPEG-LS compression algorithm to make the compression search-aware. Unlike the work in [9], we will show that our algorithm does not sacrifice the compression performance.

The paper is organized as in the following. Section 2 gives a brief introduction to JPEG-LS. Section 3 introduces our two-pass JPEG-LS algorithm and discusses the pattern matching approach based on it. Section 4 gives the preliminary results. Section 5 concludes the paper.

## 2. JPEG-LS ALGORITHM

Referring the context illustrated in Figure 1, JPEG-LS compression algorithm performs the following operation for each pixel in the scan line:



**Figure 1 JPEG-LS Pixel Prediction**

1. Find the initial prediction  $X^\wedge$

The prediction algorithm is in the following:

```

IF NW  $\geq$  max(W, N)
    X $^\wedge$  = min(W, N)
ELSE
    IF NW  $\leq$  min(W, N)
        X $^\wedge$  = max(W, N)
    ELSE
        X $^\wedge$  = W+N-NW
ENDIF

```

2. Determine the current pixel's context

To be able to predict a pixel's value accurately, in JPEG-LS, each pixel is categorized as in one of 365 contexts, each of which describes a different local characteristic of the pixels. The context is represented by a three-component vector (Q1, Q2, Q3) where  $Q_i$  ( $1 \leq i \leq 3$ ) is determined from  $D_i$  and three pre-define threshold T1, T2 and T3 as shown in the following table:

D $_i$ 's Range	Q $_i$
$D_i \leq -T3$	-4
$-T3 < D_i \leq -T2$	-3
$-T2 < D_i \leq -T1$	-2
$-T1 < D_i \leq 0$	-1
$D_i = 0$	0
$0 < D_i \leq T1$	1

$T1 < D_i \leq T2$	2
$T2 < D_i \leq T3$	3
$D_i > T3$	4

Where  $D1 = NE - N$ ,  $D2 = N - NW$ ,  $D3 = NW - W$ . T1, T2 and T3 are user predefined positive numbers.

3. Refine the prediction by considering the bias of the context in which the current pixel is. Each context maintains a prediction bias value that is used to refine the initial prediction. The bias of a context is updated every time a pixel of that context is being encoded. The prediction is thus adaptive in this sense.

4. Compute the prediction error (residual) and update the bias of the current pixel's context. For a context Q, the following values are updated: N[Q] is increased by 1 each time the given context Q is used; A[Q] is used to track the absolute errors to determine the parameters used in entropy encoding; B[Q] accumulates the pixel errors (positive or negative) and it is used to increment or decrement the bias.

The bias of Q is then updated as:

```

IF B[q]  $\leq$  -N[q]
    B[q] = B[q] + N[q]; C[q] = C[q] - 1;
    IF B[q]  $\leq$  -N[q] B[q] = -N[q] + 1;
ELSE IF B[q] > 0
    B[q] = B[q] - N[q]; C[q] = C[q] + 1;
    IF B[q] > 0 B[q] = 0;
ENDIF

```

When N[Q] exceeds a pre-determined threshold (usually 64), A, B and N are right-shifted by 1 bit to halve their values.

5. Re-map the residuals so the remapped residuals lie in range [0, M-1], provided that the original pixel values lie in that range. The re-mapping is necessary since the entropy encoder – Golomb-Rice encoder requires the input value to be non-negative. The re-mapping is also context-dependent.

6. Finally, entropy-encode the prediction error using Golomb-Rice encoder. For a prediction error to be encoded by Golomb-Rice encoder, a coding parameter is needed and it depends on the context as well.

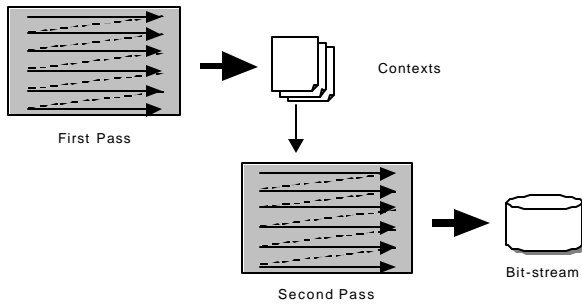
It is obvious that JPEG-LS is an adaptive algorithm, which compresses a pixel based on the previous pixels. Thus, to be able to decode a pixel's value, the previous pixels need to be known. In another word, the image decompression must be started from the beginning of the image. Thus, randomly accessing to the image data is not possible. The JPEG-LS compression algorithm also makes it difficult to perform CPM. If a full CPM is to be performed, the pattern needs to be compressed exactly the same as its matches in

the image. However, without knowing the preceding pixels of its matches in the image, the pattern will have different representation in the compressed domain. The algorithm presented in the next section solves the above problem.

### 3. TWO-PASS JPEG-LS VARIATION

#### Compression and Decompression

We propose a search-aware two-pass JPEG-LS compression/decompression scheme. The first pass scans the image in the raster scan order and computes the context data (bias of the contexts and other context-dependent information). No compression is done in this pass. The second pass scans the image again in the same order and takes the pre-computed contexts to compress the pixels. Figure 2 illustrates the two-pass compression scheme.



**Figure 2 Two-pass Compression Scheme**

We now give detailed description of the two-pass algorithm:

**Pass 1:** Pre-processing of the image and computation of the contexts.

- Compute the initial prediction  $X^\wedge$ .
- Determine the current pixel's context  $Q$ .
- Refine the prediction by considering the bias of  $Q$ .
- Compute the prediction error (residual) and update  $A[Q]$ ,  $B[Q]$ ,  $N[Q]$  and the bias of  $Q$ . Unlike in the original algorithm,  $A[Q]$ ,  $B[Q]$  and  $N[Q]$  are not right-shifted.

The operations in Pass 1 are identical to the first four steps in the original JPEG-LS compression algorithm except that  $A[Q]$ ,  $B[Q]$  and  $N[Q]$  are not right-shifted when  $N[Q]$  exceeds the threshold. By halving these values, the pixels that are spatially close to the current pixel will have more impact on the prediction of the current pixel, thus that the original algorithm takes the advantage of the local characteristics of the image and should have better prediction.

**Pass 2:** Image Compression.

- Compute the initial prediction  $X^\wedge$ .
- Determine the current pixel's context  $Q$ .
- Refine the prediction by considering the final bias of  $Q$ . The final bias of  $Q$  is the final bias of  $Q$  after Pass 1 has finished.
- Compute the parameter for entropy encoding from  $N[Q]$  and  $A[Q]$  and entropy-encode the prediction error. Note that these two values are also the final values after Pass 1 has finished. The encoding parameter  $k$  of Golomb-Rice coder is the least number of bits needed to represent the cumulative errors for  $Q$  and it is computed as in Equation 1. Note that in the original algorithm, the equation takes the cumulative errors adaptively. While in the two-pass algorithm, the above equation takes the overall cumulative errors. Thus,  $k$  is optimal in the two-pass algorithm and it should give better compression performance than the original algorithm.

$$k = \min( k' | 2^{k'} \bullet N(Q) \geq A(Q) ) \quad (1)$$

**Output the contexts:** the context data is also output as part of the bit-stream. The decoder needs the same context data to decode and the pattern-matching algorithm needs the same context data to compress the patterns so that the patterns will have the identical representation as their matches in the image.

In conclusion, the bias computation in Pass 1 may cause worse prediction because it is not adapted to the local characteristics of the image; however, the entropy encoding parameter computed in Pass 2 should be able to somewhat compensate the loss of compression performance since the encoding parameter is computed taking into account the entire image. The experimental results reported in Section 4 verified our analysis.

The decompression is almost identical to the original decompression algorithm except that no context information needs to be computed because they are pre-loaded before the decompression is started.

#### Pattern Matching

The two-pass algorithm proposed above is search-aware and it is possible to perform even a full CPM based on it. The basic idea is: given a pattern, we simply compress the pattern using the same contexts used for compressing the image (Keep in mind that the contexts are part of the bit stream and can be loaded for pattern compression). The compressed pattern, or pattern bit stream, will have the same binary representation as its match (if there is any) in

the image bit stream except for the first row and the first column of the pattern.

From Figure 1 and the first two steps in Section 2, it can be seen that the initial prediction depends on the current pixel X's three immediate neighbors N, W and NW while the context computation depends on four immediate neighbors N, W, NW and NE. For pixels on the first row and the first column of the pattern, these neighboring pixels are not available. Thus, the initial prediction and the context determination cannot be done for the first row and the first column of the pattern. However, for the rest of the pixels in a pattern, this problem does not exist. Figure 3 illustrates the above situation.

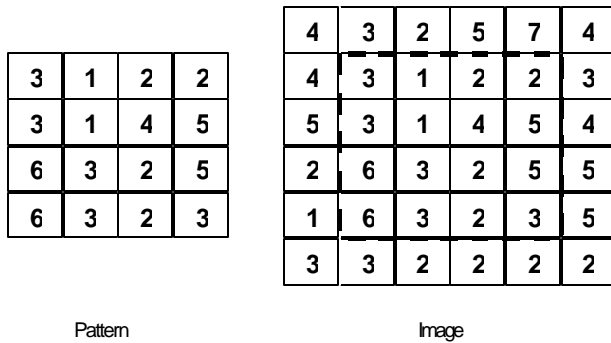


Figure 3 Pattern and Its Match in Image

#### 4. RESULTS

The 27 test images are from the DOI-10M - National Imagery and Mapping Agency. The images cover the region from 35 N, 34 E (northwest corner) to 29 N, 40 E (southeast corner). Most images cover an area that is one degree east-west and one-half degree north-south. The total image size is about 1.35 gigabytes.

The compression performance, including the compression ratio, the encoding and decoding time are given in Table I.

It can be seen that the compression ratio of the two-pass algorithm is equal to the original algorithm. This is very impressive comparing with the work done in [9], where the compression ratio of the search-aware algorithm is 6-7% worse than the original JPEG-LS compression. The encoding time, as shown in the table, is slower than the original algorithm in average because our algorithm takes two passes while the original algorithm takes only one pass to compress the images. This is a cost to pay for being search-aware and this is actually the only cost our algorithm has to pay. The decoding time, on average, is faster than the original algorithm.

	JPEGLS	Two-pass JPEGLS
Average bpp	3.61	3.61
Average encode time(s)	16.58	23.85
Average decode time(s)	16.85	15.95

Table I Compression Performance

#### 5. CONCLUSION AND FUTURE WORK

In this paper, we proposed a two-pass variation of the JPEG-LS compression standard and the new algorithm is search-aware. Our preliminary results showed that the two-pass algorithm, even with the search-awareness, still has the same compression ratio with the original JPEG-LS algorithm. Since the compression is search-aware, a full compressed pattern-matching algorithm will be pursued in our future works.

#### 6. REFERENCES

- [1] A. Amir, G. Benson: "Efficient two-dimensional compressed matching", Snow Bird, Utah, Data Compression Conference 1992.
- [2] A. Amir, G.M. Landau, and M. Farach, "Optimal two-dimensional compressed matching", Journal of Algorithms, 24(2): 354-379, August 1997.
- [3] A. Amir, G. M. Landau, D. Sokol, "Inplace run-length 2D compressed search", Theory of Computer Science, 2000
- [4] A. Amir, G. M. Landau, D. Sokol, "Inplace 2D Matching in compressed image", Theory of Computer Science, 2003.
- [5] Renato B. Pajarola, "Access to large scale terrain and image databases in geo-information system", Ph.D Dissertation, Swis Federal Instrtute of Technology, Switzerland 1998.
- [6] Bird, R.S., "Two dimensional pattern matching", Information Processing Letters, Vol.6, No.5, p.168-70, October 1977
- [7] Ricardo Baeza-Yates, Mireille Regnier, "Fast two-dimensional pattern matching", Information Processing Letters, Vol.1 No. 45, p.51-7, January 1993.
- [8] M. Weinberger, g. Seroussi, and G. Sapiro, "The LOCO-I lossless image compression algorithm: principles and standardization into JPEG-LS". IEEE Transactions on Image Processing, 9(8), 2000.
- [9] Tao Tao and Amar Mukherjee, "Compressed pattern matching for predictive lossless image encoding", Proceeding of Distributed Multimedia Systems, Miami, September 2003.