Abstract

Pattern matching is the most time consuming process in printed text image compression with JBIG2 and wavelet transform. In this paper we propose two techniques to pattern matching process for compression. By limiting the search range for matching symbols in the dictionary and by making early decisions about the pattern matching outcome, the first technique saves another 52% of encoding time with no coding loss. The second technique looks at enhanced prescreening using additional symbol features (extract using wavelet transform) besides symbol size from the selected wavelet coefficients of transform symbol. Using certain topological features, enhanced prescreening can save up to 92% of encoding time.

1. Introduction

The state-of-the art in bi-level image compression is the JBIG international standard. The basic approach of the non-hierarchical version which is a one-pass
algorithm its context model is the neighborhood template (pixel values in the causal plane, i.e., already decoded) whose bit values, concatenated together, form an address to the binary statistic for each particular context. Coding is done using adaptive binary arithmetic coding.

In JBIG, the coder is a 10-pixel neighborhood, and the adaptive binary arithmetic code is called the QM-coder. The QM-coder uses the adapter-coder technique of the Q-coder. This technique uses the renormalization inherent in the arithmetic coding process to drive the adaptation algorithm.

The JBIG2 standard [1, 2, and 3] is the new international standard for bi-level image compression. Bi-level images have only one bit-plane, where each pixel takes one of two possible colors. A typical JBIG2 encoder first segments an image into different regions [4] and then uses different coding mechanisms for text and for halftones. In this paper, we are concerned with compressing decomposed printed text images using JBIG2. Text images consist mainly of repeated printed text characters and possibly some general graphics. The input image will be transfer to wavelet domain using the wavelet transform to reduce: 1. the encoding operation time, 2. the error in encoding operation, and 3. the size of image (decompose image into 3\textsuperscript{rd} level will decrease the image into 1/8 from the original size).

In JBIG2, the coding of printed text is based on pattern matching techniques [2, 3]. JBIG2 defines two modes for text compression: pattern matching and substitution (PM-S) [5] and soft pattern matching (SPM) [6].

On a typical page of text, there are many repeated characters. The bitmap of a character instance on the
page is called a "symbol." We can extract symbols from the input image using connected component analysis (preprocessing operations will be applied into the decomposed image) [7]. Rather than coding all the pixels (coefficients) of all the symbols on the page, we code the bitmaps of a representative subset and put them into the symbol dictionary. Then, each symbol on the page is coded by giving its position on the page; the index of its best matching symbol in the dictionary, and, in the SPM mode, possibly its actual bitmap which is refinement coded using its matching dictionary symbol [1, 2]. This type of bitmap coding, called refinement coding, is done by context-based arithmetic coding using a context drawn from both the best match bitmap from the dictionary, and the already coded part of the current bitmap [8, 6].

General graphic data not identified as text is encoded at the end using a basic bitmap coder such as specified by JBIG1 [9].

A JBIG2 coding system for printed text images consists of several components: symbol extraction, pattern matching, arithmetic/Huffman integer/bitmap coding. To speed up arithmetic bitmap coding, JBIG2 allows typical prediction (TP) as specified in JBIG1 [9] and typical prediction for residue (TPR) as proposed in [11]. In this paper, we focus on reducing the encoding time spent on pattern matching by using two speedup encoding techniques and the wavelet transform. In our work we use the Hamming distance based matching criterion. We measure the percentage of different pixels between two symbols. For SPM-based JBIG2, even using our simple matching criterion, the time spent on
pattern matching accounts for as much as 92% of the total encoding time.

In this paper we develop two categories of newest techniques that can significantly reduce the amount of pattern matching time while causing only a very small loss in coding efficiency.

This paper is organized as follows. In Section 2 we explain the wavelet transform theory. In Section 3 we propose the two techniques for pattern matching. In Section 4 we show experimental results on coding time saved and bit rate penalty incurred from using these techniques. We conclude our paper in Section 5.

2. Wavelet Transforms Theory
Wavelets are functions of limited duration and having average value of zero. These are generated from the single function by dilations and translations

$$\psi_{a,b}(t) = |a|^{1/2} \psi\left(\frac{t-b}{a}\right)$$

The definition of wavelets as dilates of one function means that high frequency wavelets correspond to $a<1$ or narrow width, while low frequency wavelets have $a>1$ or wider width. Daubechies [5] was the first to discover that the discrete time filters or QMFs can be iterated and under certain regularity conditions will lead to continuous-time wavelets. Since digital image is a discrete signal, so wavelet based image decomposition can be implemented using FIR discrete time filters.
There are number of ways, wavelet transform may be used to decompose a signal into various sub bands such as uniform decomposition, octave-band decomposition, adaptive or wavelet packet decomposition etc. This paper uses the octave-band decomposition to decomposed the printed text image into regions to decreased the encoding operation time and reduce the error rate.

In the octave-band decomposition, we first pass each row of image through analysis filter bank \((h_0, g_0)\) and down sampled to get the transformed image which contains the average value and detail coefficients along each row. Next we treat these transformed rows as if they were themselves an image and apply the same process to each column. This transformation process results in 4-band \((LL, LH, HL, HH)\) decomposition of an image [6].

To further decompose the resulting image, we repeat this process recursively on the LL-sub band containing averages in both directions. This decomposition provides sub images corresponding to different resolution levels (fig. 1). At the decoder (fig 2), the sub band signals are decoded, up sampled and passed through a bank of synthesis filters \((h_1, g_1)\) and properly summed up to yield the reconstructed image.
<table>
<thead>
<tr>
<th>m ≥ 2</th>
<th>m = 2</th>
<th>m = 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low resolution sub images (LL2)</td>
<td>Horizontal orientation sub image at m = 2 (LH2)</td>
<td>Resolution m=1 Horizontal Orientation Sub-image (LH1)</td>
</tr>
<tr>
<td>Vertical orientation sub image at m=2 (HL2)</td>
<td>Diagonal orientation sub image at m=2 (HH2)</td>
<td>Resolution m=1 Diagonal Orientation Sub-image (HH1)</td>
</tr>
</tbody>
</table>

Figure 1 the decomposed image using wavelet transform.

(a) The one stage in multiscale image decomposition.
(b) The one stage in multiscale image reconstruction.

Figure 2 the wavelet decomposing Operation

The main property of wavelet transform is that regions of little variation in original data manifest themselves as small or zero elements in the wavelet-transformed version. Hence, the wavelet Transform of the image contains a large number of detail coefficients, which are very small in magnitude. By fixing a nonnegative threshold, we can reset these small coefficients to zero resulting in a very sparse matrix [8]. Very sparse matrices are easier to store and transmit than ordinary matrices of the same size. Moreover, the image constructed from the threshold data gives visually acceptable results.

A no uniform shareholding technique is used in this paper split the affect image coefficient (like the hole and connected coefficient explain in 2.2). In this paper, the wavelet threshold value calculated using the following equations:
\[ T_N = \frac{\beta \hat{\sigma}^2}{\hat{\sigma}_y} \]

Where, the scale parameter \( \beta \) is computed once for each scale using the following equation:

\[ \beta = \sqrt{\log\left(\frac{L_k}{J}\right)} \]

\( L_k \) is the length of the sub band at \( k_{th} \) scale. \( \sigma^2 \) is the noise variance, which is estimated from the sub band LL1, using the formula:

\[ \sigma^2 = \begin{cases} \text{median}(Y_{ij})^2 \\ 0.6745 \end{cases} \]

\( Y_{ij} \in \text{sub band of LL1} \)

And \( \sigma_y \) is the standard deviation of the sub band under consideration.

3 Speedup Techniques for JBIG2 Encoding
3.1 Early jump-out based on previous best match

When matching one symbol with another, we save the previous lowest mismatch score; the pattern matcher compares on-the-fly the current accumulated mismatch score against the previous lowest one. If the current mismatch is already above the previous lowest, then we terminate the current matching process. Computing the Hamming distance between two symbols coefficients is fast because it only requires the exclusive-OR (XOR) operation and incrementing the mismatch score accordingly. On the other hand, comparing the two integer mismatch scores also takes time. Therefore, we do the real number comparison of mismatch scores only once for each row of pixels coefficients in the bitmap. At
the end of each row, the current accumulated mismatch is checked; if it exceeds the previous lowest, the pattern matching process terminates.

### 3.2 Enhanced Prescreening

Before matching a pair of symbols, it is advantageous to prescreen them by certain features. There is no need to apply pattern matching to two symbols that are obviously dissimilar. For example, symbols that differ greatly in size (e.g. a capital “D” and a comma “,”) obviously do not match. The original SPM system will prescreens symbols using size; only symbols with similar sizes (defined as not more than 2 pixels different in either dimension) are given to the pattern matcher which computes their mismatch score. Prescreening is intended to reduce the number of unnecessary pattern matching calls that will not return a match. At the same time, prescreening should not rule out potentially good matches. Otherwise it will incur a high bit rate penalty. Therefore, the ideal pre-screening rules out all “unmatchable” symbols and passes on all “matchable” symbols to the more expensive pattern matching subroutine.

Other features can be used in prescreening besides symbol size. One such example is to use symbol area and/or perimeter [7, 14]. However, these two features are not particularly helpful for two reasons: they are correlated with symbol size, and they are usually sensitive to scanning noise and digitization parameters such as contrast [7]. According to our experiments, in the English language, using the Hamming distance based matching criterion, letter pairs that are among the most
easily confused include "b" and "h," "c" and "e," and "i" and "l." In this paper we propose two topological features for prescreening: number of holes and number of connected coefficients of components (for example the hole coefficient that have zero's values and the connected have a value greater than the wavelet threshold).

Prescreening by these two features can effectively prevent these symbol pairs from being handed over to the pattern matcher (as shown in Figure 3).

Another useful feature for prescreening is introduced in [7] and developed as the following. We call it the coefficient quadrant centroid distance. It is calculated as follows. We divide each symbol coefficients into four quadrants and calculate the centroid for each coefficients quadrant. To prescreen two symbols, we calculate the distance between each coefficient pair of corresponding coefficients quadrant centroids, sum the four distances and compare the total to a threshold, which is preset to 4 coefficients in our implementation. A small total distance means that the two symbols have similar mass distribution in all four quadrants; only such symbol coefficients pairs are passed on to pattern matching to be further examined.

4. Experimental results
In this section we show experimental results on the two proposed techniques, the early jump-out (EJMOT), and enhanced prescreening (ENHPRE). We consider two figures of merit, the encoding time saved and the bit rate penalty incurred.

Our experiments use a set of twelve test images, two
Figure 3 Examples of similar bitmaps that have different features. Bitmaps “b” and “h” differ in the number of holes; and Bitmaps “i” and “l” differ in the number of connected components.

Standard Bitmap (BMP) images (resolution 200 dpi), and ten images (resolution 300 dpi). The computer system used is a Pentium4 2.8GHz, running under Windows XP, with 512MB physical memory. We measure encoding time (in sec). Our code was simples as possible and optimized for speedup calculation and written by using Delphi 6 language. All coding results given are averaged over all test images.

Tables 1 and 2 give the total encoding time; time spent on pattern matching, and coded file size for each individual technique and several combinations of them, for SPM-and PM&S-based JBIG2, respectively. In SPM (as shown in Table 1), the encoding time spent on pattern matching accounts for up to 92% of the total
encoding time. The rest of the encoding time (spent on symbol extraction, arithmetic bitmap and integer coding, etc.) is a fixed value of 5.2 seconds. For the PM&S mode (as shown in Table 2), the time spent on pattern matching accounts for up to 88% of the total encoding time. The rest of the encoding time is a bigger fixed value of 6.1 seconds. Compared to the first row (NONE) where no proposed techniques are used and prescreening is only by symbol size, early jump-out technique saves 52% 40% of the pattern matching time in SPM and PM&S, respectively, while incurring no bit rate penalty at all. For the ENHPRE rows, adding the coefficients quadrant centroid distance (S+Q) to prescreening saves almost 3/4 of the pattern matching time, while incurring a bit rate penalty of around 1%. Using the numbers of holes and connected coefficient components together (S+H+C) saves 90% of the pattern matching time, which is less efficient than the Q feature. However, the bit rate penalty incurred is only half as big (0.5%). Combining all these proposed techniques together (the ALL rows in the two tables) saves 92% and 87% of the pattern matching time in SPM and PM&S, respectively.

The bit rate penalty incurred is relatively small, 1.4% for SPM and 1.1% for PM&S.

Over a channel of fixed bandwidth, transmission of a bigger file takes longer time. Some applications, e.g., sending an international fax, favor the shortest channel time possible. For such applications, achieving the best compression is the most important, even if it takes some extra encoding time. Other applications, especially real-time applications, can only tolerate a small delay between the sender and receiver. For these applications,
the goal is to achieve the best compression within a short encoding time. However, better compression usually requires longer encoding time. The SPM system is similar observations are made for the PM&S system.

5. Conclusion
In this paper we propose two techniques to pattern attaching process in printed text image compression with JBIG2 and wavelet transform. Experiments show that the limited dictionary symbol search technique and the early jump-out technique can each bring about 52% of savings in encoding time without loss in coding efficiency. Depending on the specific features used, the enhanced prescreening technique can save up to 92% of encoding time while only suffering a small bit rate penalty of at most 1.4%. These speedup techniques are effective for both SPM-based and PM&S-based JBIG2.

Also, from the results its clear to show the effect of the wavelet transform in the encoding operation by reduce the time of encoding and the size of the compressed image.

Table 1 using the two proposed techniques in SPM JBIG2

<table>
<thead>
<tr>
<th>Technique</th>
<th>Total time</th>
<th>Match time</th>
<th>Coded size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sec</td>
<td>% gain</td>
<td>sec</td>
</tr>
<tr>
<td>NONE</td>
<td>85.10</td>
<td>-</td>
<td>70.00</td>
</tr>
<tr>
<td>EJO</td>
<td>40.84</td>
<td>52</td>
<td>36.47</td>
</tr>
<tr>
<td>PRESCRN</td>
<td>S+Q</td>
<td>21.27</td>
<td>75</td>
</tr>
<tr>
<td>S+H+C</td>
<td>8.1</td>
<td>90</td>
<td>8.40</td>
</tr>
<tr>
<td>S+Q+H+C</td>
<td>6.80</td>
<td>92</td>
<td>7.0</td>
</tr>
</tbody>
</table>
Table 2 using the two proposed techniques in PM&S JBIG2

<table>
<thead>
<tr>
<th>Technique</th>
<th>Total Time (sec)</th>
<th>Match Time (sec)</th>
<th>Coded Size (bytes)</th>
<th>Gain (%)</th>
<th>Loss (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>20.00</td>
<td>10.24</td>
<td>41,404</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>EJO</td>
<td>12.00</td>
<td>5.93</td>
<td>41,404</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>PRESCRN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S+Q</td>
<td>6.00</td>
<td>2.76</td>
<td>41,730</td>
<td>70</td>
<td>0.75</td>
</tr>
<tr>
<td>S+H+C</td>
<td>5.60</td>
<td>2.86</td>
<td>41,566</td>
<td>72</td>
<td>0.5</td>
</tr>
<tr>
<td>S+Q+H+C</td>
<td>4.40</td>
<td>2.25</td>
<td>41,925</td>
<td>78</td>
<td>1.4</td>
</tr>
</tbody>
</table>

6. References


1974.


