Watermarking Electronic Text Documents
Containing Justified Paragraphs and Irregular Line Spacing

Adnan M. Alattar and Osama M. Alattar
Digimarc Corporation, Tualatin, OR 97062

ABSTRACT

In this paper, we propose a new method for watermarking electronic text documents that contain justified paragraphs and irregular line spacing. The proposed method uses a spread-spectrum technique to combat the effects of irregular word or line spacing. It also uses a BCH (Bose-Chaudhuri-Hocquenghem) error coding technique to protect the payload from the noise resulting from the printing and scanning process. Watermark embedding in a justified paragraph is achieved by slightly increasing or decreasing the spaces between words according to the value of the corresponding watermark bit. Similarly, watermark embedding in a text document with variable line spacing is achieved by slightly increasing or decreasing the distance between any two adjacent lines according to the value of the watermark bit. Detecting the watermark is achieved by measuring the spaces between the words or the lines and correlating them with the spreading sequence. In this paper, we present an implementation of the proposed algorithm and discuss its simulation results.

Keywords: Text Watermarking, Document Fingerprinting, Copyright Protection

1. INTRODUCTION

Most companies and government agencies have a dire need for protecting sensitive information. Encryption, access restriction, and locking documents behind firewalls are some common techniques for protecting sensitive information. Encryption is an effective way for preventing an unauthorized person from viewing the content of a sensitive document. Nonetheless, once the document is decrypted for viewing using the secret key, an ill-intentioned authorized person can save, copy, print, or transmit the unencrypted document anywhere he or she wants without any major difficulty.

Restricting access of a document to only a few individuals works well with trustworthy individuals. Unfortunately, it is common to find secret documents circulating outside their trusted rings and even in the public media. Identifying the untrustworthy person is often difficult and unpleasant.

Firewalls are an effective means to ban casual outsiders from accessing an organization's network. Firewalls also make it very difficult for a savvy computer hacker to break in. Unfortunately, firewalls cannot prevent an insider from copying a sensitive document onto a disk or emailing it to an outsider using a third-party Internet service provider to avoid tracking.

Obviously, a comprehensive solution for ensuring the security of sensitive documents cannot rely on a single technique. Instead, effective security must employ all the aforementioned techniques, and furthermore, it must fingerprint the document.

Fingerprinting here refers to the embedding of an indiscernible ID in the document, which identifies the owner or the recipient of the document. Fingerprinting does not mean cryptographically hashing the document into a signature that can be used to identify the document uniquely. The embedded ID can be detected and decoded from a fingerprinted document whenever and wherever the document is encountered, even after DA/AD conversion (print and scan). This process can be achieved through the use of digital watermarking techniques.

Image watermarking techniques can be easily applied to a text document, but they introduce a very noticeable white noise in the text document. This noise is due to the binary (black and white) nature of the text document and its large white background. Moreover, image watermarking techniques are applicable only to image bitmaps, which implies that the text document must first be rasterized.

To avoid the aforementioned problems, several watermarking techniques have been specifically developed for use with text documents [1]-[8]. Mei, et al. [1] developed a method to reduce the watermarking noise by restricting the changes due to the watermarking process to the text boundary. In this technique, edge patterns are first isolated; then, each edge pattern is replaced with one of two predefined, but similar, patterns to indicate the value of the embedded bit (zero or one). Detecting the watermark is achieved easily without referring to the original document by inspecting each of these edge patterns to decide if it corresponds to a zero or one.

Brassil, et al. [2]-[3] developed three methods for watermarking text documents, and Maxemchuk, et al. [4]-[6] analyzed the performance of these methods. Low, et al. [7]-[8] further analyzed the performance of these methods and derived their capacity.

The first method is called line-shift coding, in which each even line is slightly shifted up or down according to the value of a specific bit in the payload. If the bit is one, the corresponding line is shifted up; otherwise, the line is shifted down. The odd lines are considered control lines, hence they remain unmoved. They are used as references for measuring and comparing the distances between lines during the decoding stage. Decoding is achieved by comparing the distances between the bases of the lines, or the distances between the centroids of the lines. The bases of the lines in the original document are normally uniformly spaced; hence, the original document is not needed for the detection process that uses baselines. However, the centroids are not necessarily uniformly spaced; therefore, the original document must be used in the detection process that uses centroids.

The second watermarking method proposed by Brassil et al. is called word-shift coding. In this method, each line is first divided into groups of words. Each group has a sufficient number of characters. Then, each even group is shifted to the left or the right according to the value of a specific bit in the payload. The odd groups are used as references for measuring and comparing the distances between the groups during the decoding stage. A correlation and centroid-based method have been suggested for detecting the watermark. Both of these methods require the use of the original document, especially when the inter-word spacing is variable.

The third method proposed by Brassil et al. is called feature coding. In this method, certain text features (e.g., vertical end lines) are altered in a specific way to encode the zeros and ones of the payload. Watermark detection is achieved by comparing the original document with the watermarked document.

Huang and Yan [9] developed a text watermarking technique based on the average inter-word distance in each line. They readjust these distances and the length of each word so that the average inter-word distance in each line represents a sample of a sine wave of specific phase and frequency. The ID (payload) is represented by the phase and the frequency of the sine wave. A correlation detector is used to decode the payload from the watermarked document without referring to the original document.

In this paper, we present a new method for watermarking electronic text documents which is similar to the word-shift coding and line-shift coding methods described above. However, unlike those methods, our method does not require the original document for detecting the watermark and it works with documents that contain aligned left, centered, aligned right, or justified paragraphs as well as regular or irregular line spacing. Justified paragraphs are very common in electronic documents. To force the ends of the last word of each line to lie precisely on the right margin, the most widely used word processors automatically and systematically spread the words within each line. Irregular line spacing happens naturally as a result of insertion of mathematical symbols, super- or sub-script characters, and other objects. To accommodate the tallest object in each line, the word processor automatically adjusts the spacing between the lines as necessary.

The details of the proposed embedding algorithm are presented in the next section, and the details of the detection algorithm are presented in Section 3. The simulation of these algorithms and the empirical results are presented in Section 4. Our conclusions are presented in the last section.

2. THE EMBEDDING ALGORITHM

2.1. The General Algorithm

We use spread-spectrum and BCH error coding techniques to combat the effects of irregular word or line spacing commonly found in text documents. The technique we are proposing for irregular line spacing is very similar to that for irregular word spacing. Therefore, to avoid undue redundancy, our discussion in this paper will be limited to word spacing.

Figure 1 depicts the embedding process of the proposed algorithm. The embedding process starts by using a BCH error coding technique to protect the payload from noise. Next, a unique m-sequence is used to spread each of the encoded bits to form the watermark. The spread bits (chips) are embedded into the text document by slightly increasing or decreasing the spaces between words. The document is scanned from top to bottom and each space is slightly increased or decreased by a small delta according to the value of its corresponding watermark bit. If the bit is zero, the space is decreased; if the bit is one, the space is increased.

Delta is the gain factor, and increasing its value increases the strength of the watermark. However, the delta should be kept small enough so that when a space between two words is increased or decreased, the words are reasonably separated from each other.
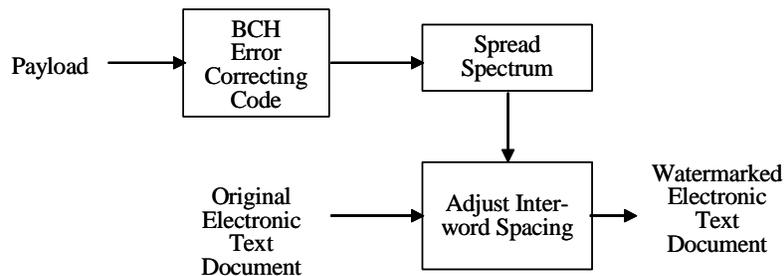
Fig. 1. The electronic text document embedder.

A typical double-spaced text, written in an 11 points Times New Roman font on an 8 ½ x 11 inch page with 1" margins all around, contains about 25 lines. Each line contains an average of 13 words. Therefore, each line contains an average of 12 spaces between words, resulting in about 300 spaces between words per page.

If a 16-bit spreading sequence is used to spread each bit, then an 18-bit payload can be hidden in each page. This payload is enough for 262,144 different IDs, but the document will be very susceptible to printing and scanning noise. If a BCH code is used to guard against noise, then the number of allowed IDs will decrease significantly.

Table 1gives the size of the allowed payload and the number of errors that can be corrected by different BCH codes. Theoretically, the (15,5) BCH code provides the most protection, but this code provides the smallest payload size. The (15,5) BCH code allows the embedding of only 32 IDs, but can correct up to three errors.

On the other hand, the (15,11) BCH code provides the largest payload size but the least protection. This code can correct only one error, but allows the embedding of 2048 different IDs. The (7,4) BCH code strikes a balance between protection and payload size. This code can correct one error in four bits. Hence, when two (7,4) BCH codes are concatenated to protect eight bits, together they can correct two errors, while allowing the embedding of 256 unique IDs.

A single-spaced page contains about double the number of spaces that a double-spaced page contains. Hence, a single-spaced page allows the embedding of about double the number of bits, which substantially increases the number of allowed IDs.

The ID space can also be increased by allowing the embedding process to use the spaces from multiple consecutive pages to embed one ID. However, this enhancement requires crossing the page boundary which would complicate the decoding process. After all, a large ID space may not be necessary for fingerprinting sensitive documents, which are

usually distributed to a small number of people. Also, the same IDs can be used with different documents, since different documents need not have unique IDs.

BCH Code	Length	Payload Size	Parity Bits	Number of Correctible Errors	Number of IDs
(7,4)	7	4	3	1	16
(15,11)	15	11	4	1	2048
(15,7)	15	7	8	2	128
(15,5)	15	5	10	3	32

Table 1. Payload size, number of allowed IDs, and number of bits that can be corrected for different BCH codes.

The 16-bit spread-spectrum code can be generated using the 4-bit shift register shown in Fig. 2. This code generates an m-sequence of period 16, which we denote by $m(n)$. The m-sequence has the desired correlation properties for use with a correlation-based detector, since its autocorrelation is a delta function. Now, the spreading code, $c(n)$, can be generated from $m(n)$ as follows:

$$c(n) = 2m(n) - 1 \tag{1}$$

This equation converts the range of the m-sequence from $\{0,1\}$ to $\{-1,1\}$. If we denote each bit of the encoded payload bits by $b\varepsilon\{-1,1\}$, then applying the spread spectrum technique to the encoded payload can be expressed by

$$w(n) = b \times c(n), \tag{2}$$

where $w(n)$ is a 16-bit spread spectrum sequence representing the bit b.
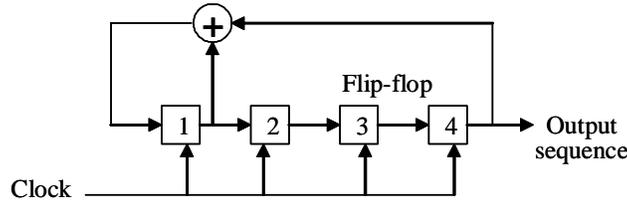

Fig. 2. Generating 16-bit m-sequence.

2.2. Embedding Electronic Document

Adjusting the spacing between the words is as simple as detecting the white spaces and increasing or decreasing their widths. If this space adjustment is done in real time and if the document is justified, then oftentimes the word processor readjusts the inter-word spacing to maintain justification. Consequently, this automatic readjustment may cause a ripple effect in which words will be moved from one line to the next. It is possible to avoid this ripple effect by slightly adjusting the distance between the characters of each word in every line in a way similar to that described in [9]. If $ows_{i,j}$ represents the width of the jth original white spaces in the ith line, and if $nws_{i,j}$ represents its new width after watermarking, then the sum of these widths before and after watermarking is given by $osw_i = \sum_{j=1}^{N_i} ows_{i,j}$ and

$nsw_i = \sum_{j=1}^{N_i} nws_{i,j}$, respectively. If $wl_{i,j}$ represents the width of the jth word in the ith line, then the sum of the widths

of all the words in that line is given by $swl_i = \sum_{j=1}^{N_i+1} wl_{i,j}$. To compensate for the difference between nsw_i and osw_i,

the width of each word in a line must be adjusted to $\hat{w}l_{i,j} = wl_{i,j} + \Delta_{i,j}$, where $\Delta_{i,j}$ is a very small number calculated according to the following formula:

$$\Delta_{i,j} = \begin{cases} \left\lceil \left(nsw_i - osw_i\right)\dfrac{wl_{i,j}}{swl_i} \right\rceil & if \ \left(nsw_i - osw_i\right)_i \geq 0 \\[3mm] \left\lfloor \left(nsw_i - osw_i\right)\dfrac{wl_{i,j}}{swl_i} \right\rfloor & if \ \left(nsw_i - osw_i\right)_i < 0 \end{cases} \qquad (3)$$

The embedding process, described above, can also be implemented covertly at the level of the printer driver of a post script printer. In this case, the printer driver produces a postscript file that contains instructions describing the page. The printer interprets these instructions and correctly prints the page. By modifying the instructions in the postscript file to induce the effect described above, the printer will print a watermarked document.

2.3. Embedding Printed Document

Fingerprinting a printed document is to some extent harder than fingerprinting an electronic document. The process is very similar to the process described in the next section for detecting the watermark in a printed document. In this process, a printed document must first be scanned then operated on by an image processor to identify the lines and the spaces between the words. Once these spaces are identified, each word can be extracted and slightly shifted. The same process can be used to identify and modify the spaces between the characters of each word to compress or expand the word and maintain paragraph justification. During the process, extra care must be applied to avoid leaving any artifacts.

3. THE DETECTION ALGORITHM

3.1. Detecting a Watermark in an Electronic Document

Detecting the watermark in an electronic document is a straightforward process. The detector starts by measuring and recording the distances, $nws_{i,j}$, between every two consecutive words. Before watermarking, $ows_{i,j}$ in each line is almost the same. The word processor usually adjusts these distances slightly to justify the line. Therefore, the average, $\overline{nws}_{i,j}$, of $nws_{i,j-1}$ and $nws_{i,j+1}$ is a good prediction of $ows_{i,j}$. Hence, $\overline{nws}_{i,j}$ is computed and subtracted from each of the recorded distances to estimate the nth sample, $w(n)$, of the watermark signal. The resulting watermark estimates, $\hat{w}(n)$, are then segmented into segments of 16 samples each. Each of these segments is correlated with the original m-sequence to retrieve a payload bit. Finally, the BCH decoding is performed on the payload bits to correct any errors.

The estimate, $\hat{w}(n)$, of the nth sample of the watermark signal can be written as

$$\hat{w}(n) = w(n) - (w(n+1) + w(n-1))/2 + \phi(n) \qquad (4)$$

where $\phi(n)$ denotes a random noise. For electronic documents, $\phi(n)$ is purely due to the irregular spacing between words before watermarking. Figure 3 shows the probability distribution of $\phi(n)$ for a justified text written in 11 point Times New Roman typeface. The figure indicates that the noise $\phi(n)$ is a zero mean, Gaussian noise with a variance of 0.23. It should be noted here that $\phi(n)$ is assumed zero for unjustified text. For a scanned document, $\phi(n)$ also includes printing and scanning noise.
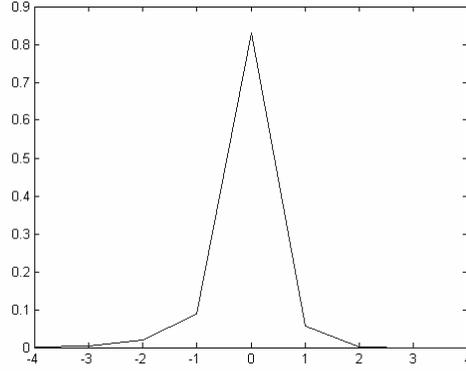
Fig. 3. The probability distribution of the noise, $\phi(n)$, due to line justification.

The term $(w(n+1)+w(n-1))/2$ is another source of noise that should have virtually no effect on the detector. Substituting $w(n)$ from equation (2) into equation (4) results in the following equation:

$$\hat{w}(n) = bc(n) - (bc(n+1) + bc(n-1))/2 + \phi(n) \tag{5}$$

Applying the correlation detector to equation (5) results in

$$\sum_{n=1}^{N} \hat{w}(n)c(n) = b\sum_{n=1}^{N} c(n)c(n) - \frac{1}{2}\sum_{n=1}^{N} bc(n+1)c(n) + \frac{1}{2}\sum_{n=1}^{N} bc(n-1)c(n) + \sum_{n=1}^{N} \phi(n)c(n) \tag{6}$$

where N is the length of the spreading code $c(n)$. Since $c(n)$ is an m-sequence, the second and third terms on the right-hand side of equation (6) are zeros. This result is because the autocorrelation of the m-sequence is a delta function. The last term on the right-hand side of equation (6) represents a low magnitude noise, $\eta(n)$, because multiplying the spreading sequence with a constant has the effect of spreading the power of the constant over a much wider bandwidth. Hence, equation (6) can be simplified to

$$\sum_{n=1}^{N} \hat{w}(n)c(n) = b + \eta(n). \tag{7}$$

Now, the value b of the watermark bit can be obtained by applying the following threshold to the correlation results:

$$b = \begin{cases} 1 & \sum_{n=1}^{N} \hat{w}(n)c(n) \geq 0 \\ -1 & \sum_{n=1}^{N} \hat{w}(n)c(n) < 0 \end{cases}. \tag{8}$$

3.2. Detecting a Watermark in a Printed Document

The decoding of the watermark from a printed document is a bit more challenging. The process can be summarized as follows:

1. Scan the printed document at a reasonable quality and a reasonable resolution. The higher the quality and the resolution, the better the detection results.

2. Convert the image into a binary image by properly thresholding it. The value of the threshold can be easily determined from the image histogram, which is a bimodal. Assign 1 to any value above the threshold and 0 to any value below the threshold. Hence, the text will be assigned the zero value.
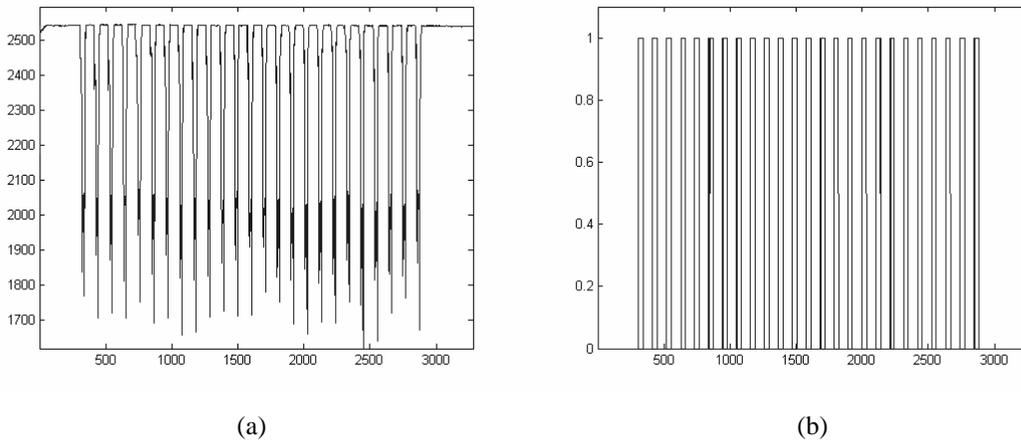

(a)   (b)

Fig. 4. (a) Vertical signature of a typical text document, (b) Locations of extracted lines indicated by 1.

3. Correct any discrepancy between the orientation of the scanned document and that of the electronic document. For this purpose, the detector can use the direction of the lines in the scanned document as a reference. Normally, in the intended application, the user would carefully and properly position the printed document on the scanner. This would cause only slight discrepancies in the orientation, which would not be hard to correct.

Line Number	Y-Coordinate	Width	Line Number	Y-Coordinate	Width
1	310	35	14	1690	27
2	415	36	15	1787	35
3	521	36	16	1891	36
4	629	33	17	1997	36
5	733	35	18	2103	36
6	847	28	19	2217	27
7	952	27	20	2315	34
8	1055	30	21	2419	36
9	1154	37	22	2526	35
10	1260	36	23	2632	36
11	1366	35	24	2739	35
12	1471	35	25	2854	27
13	1576	36			

Table 2. Y-coordinates and widths of the lines detected in a typical text page.

4. Extract the lines of the scanned document. This can be achieved by computing the vertical signature, where the vertical signature, $v(i)$, of the image $I(i, j)$ is

$$v(i) = \sum_{j=1}^{W} I(i, j) \qquad (9)$$

where W denotes the width of the image $I(i, j)$. Figure 4a shows the vertical signature of a typical text document scanned at 300 DPI then converted to a black-and-white image. Figure 4b shows the locations of the extracted lines from the same document, and Table 2 lists the y-coordinates and widths of all detected lines. These locations are determined by thresholding the signature $v(i)$, and then recording the locations of the valleys.

5. Detect and extract the spaces between every two consecutive words. This step can be achieved by computing the horizontal signature, $h(j)$, of a small image strip $S(i, j)$ around each line as follows:

$$h(j) = \sum_{i=1}^{H} S(i, j) \qquad (10)$$

where H denotes the height of the image strip $S(i, j)$. Figure 5a shows the image segment around a line in a typical text document. Figure 5b shows the horizontal signature of the image segment that is shown in Fig. 5a. The distances between the words can be detected from the horizontal signature by detecting the mountains in Fig. 5b.

Very wide and very narrow mountains should be ignored. Very wide mountains are attributed to the page margins, and very narrow mountains are attributed to spaces between characters within the words. Figure 5c shows the location of the detected spaces between the words in the line shown in Fig 5a. Table 3 lists the x-coordinates and widths of all the detected spaces.

(a)
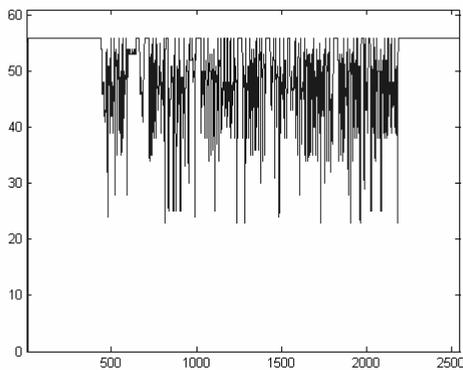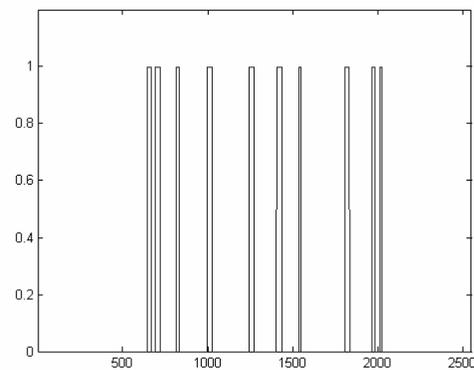
(b)   (c)

Fig. 5. (a) Image segment around a line in a typical text document,
(b) Horizontal signature of the image segment shown in (a),
(c) Locations of extracted spaces indicated by 1.

Space Number	1	2	3	4	5	6	7	8	9	10
X-Coordinate	645	692	816	997	1242	1405	1535	1806	1967	2011
Space Width	20	28	15	27	28	30	15	27	16	12

Table 3. X-coordinates and widths of all detected spaces in the line shown in Fig. 5b.

6. Concatenate all the measured spaces from all the lines and proceed in the same way as in the case of the electronic document; i.e, correlate every 16 segments with the spreading sequence to retrieve the watermark bit; then decode the watermark bits with a BCH decoder to correct any error and retrieve the payload bits.

4. EXPERIMENTAL RESULTS

We have implemented the algorithm described in Sections 2 and 3 for fingerprinting electronic text documents including aligned and justified documents by varying the spacing between the words. We used several configurations for the length of the spreading sequence and the use of the BCH code.

In one configuration, we used a 32-bit m-sequence and an 8-bit payload (256 ID space), but we did not use error-correcting codes. The 8-bit payload is sufficient for tracking a document and identifying its original recipient for an organization with up to 256 employees. The m-sequence was generated with a shift register similar to that shown in Fig. 2, but with 5 flip-flops and two feedback branches, one branch after flip-flop #1 and the other after flip-flop #5. We initialized the shift register with 16.

We tested our algorithm with the above configuration by embedding a text document written in 11 point Times New Roman typeface with 256 different IDs. We used the detection algorithm to detect the IDs from each of the unaltered digital files of the marked text documents, and the detection rate was 98.8%. For each of the miss-detected IDs, the number of error bits was one. Table 4 lists the cases of the miss-detected IDs, and it compares the embedded ID with the detected ID. The table illustrates the error bit in each case. Since the maximum bit error was one bit, any of the BCH error correction codes listed in Table 1 should be sufficient to correct the bit errors.

Detected ID		Embedded ID	
Decimal	Binary	Decimal	Binary
9	00001001	41	00101001
25	00011001	57	00111001
145	10010001	209	11010001

Table 4 Miss-detected IDs and their originally embedded values.

A similar detection rate was obtained when the algorithm was tested with slightly reformatted text. Such reformatting includes changing the type size and typeface; replacing words; center-, left-, or right-aligning the text; and changing the left and right margins of the page.

Text editing operations such as deleting and inserting a word produced diverse results. In many cases, detection was successful because the spread-spectrum technique is robust to local errors, especially when such errors occur near the ends of the spreading sequence. In this case, a large portion of the spread sequence is left intact, which allows the correlator to recover the embedded bit. However, when the error is near the center, neither of the resulting portions is long enough to recover the embedded bit successfully.

In another configuration, we used the (15,7) BCH code to protect the document from double errors. This configuration required reducing the spreading sequence from 32 bits to 16 bits, to fit the 15- bit code in a double-spaced page. This step is not necessary for a single-spaced page, since it contains enough spaces. The 7-bit payload is sufficient for tracking a document and identifying its original recipient for an organization with up to 128 employees. We generated the m-sequence using the generator shown in Fig. 2 with an initial state of 8. The detection rate was 100% when the algorithm was tested with an unaltered, marked word processing file. However, close inspection of the results indicated that 40% of the IDs had some errors that were corrected by the BCH code. Most of these errors were single errors, but a few were double errors. This result indicates that reducing the spreading sequence from 32 to 16 has a negative effect on error rate.

The spreading sequence and the payload can be improved by replacing the BCH with a concatenate (7,4) BCH code. The concatenated BCH code will allow for 8-bit payload and will allow for increasing the spreading sequence to 20 bits for the same size document. The 8-bit payload allows the embedding of 256 different IDs. From the above experiment, the maximum error was a 2-bit error and the concatenated BCH code will be sufficient for correcting 2-bit errors. The increase in the length of the spreading sequence should reduce the probability of error before the error correction is applied.

We also used Matlab to implement the procedure described in Section 3.2 for detecting the lines and measuring the spaces between the words of each line. Our preliminary test results indicated that the algorithm was able to correctly detect and measure all the spaces in the printed documents. However, fine tuning is still needed to make the algorithm more reliable for detecting the watermark from a printed document.

5. CONCLUSIONS

In this paper, we presented a new algorithm for fingerprinting a sensitive document to identify its original recipient whenever the document is found. The algorithm is based on adjusting the distances between the words or the distances between the lines of the text. Our algorithm is more suitable for centered, left- and right-aligned text, and justified text as well as text with irregular spacing between lines. Unlike the algorithms proposed in [2], our algorithm uses spread-spectrum and BCH error correction techniques, and it does not need the original document for detection. The spread-spectrum technique guards against noise due to paragraph justification or irregular line spacing. The BCH error correction technique guards against printing and scanning noise. Simulation results showed that the algorithm is resilient to some document reformatting, such as changing the typeface and changing the margins. Also, our preliminary results of detecting and measuring the distances between words and between lines from a printed document are promising. Fine tuning and additional investigation of the algorithm is underway to detect the watermark from a printed document.

REFERENCES

1. Q. Mei, E. K. Wong, and N. Memon, "Data Hiding in Binary Text Documents," *Proceedings of the SPIE, Security and Watermarking of Multimedia Contents III*, vol. 4314, San Jose, California, Jan. 2001, pp. 369-375.

2. J. T. Brassil, S. Low, N. F. Maxemchuk, and L. O'Gorman, "Electronic Marking and Identification Techniques to Discourage Document Copying," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 8, October 1995, pp. 1495-1504.

3. J. T. Brassil, S. Low, and N. F. Maxemchuk, "Copyright Protection for the Electronic Distribution of Text Documents," *Proceedings of the IEEE*, vol. 87, no. 7, July 1999, pp.1181-1196.

4. N. F. Maxemchuk, S. H. Low, "Performance Comparison of Two Text Marking Methods," *IEEE Journal of Selected Areas in Communications (JSAC)*, May 1998. vol. 16 no. 4 1998. pp. 561-572.

5. N. F. Maxemchuk, "Electronic Document Distribution," *AT&T Technical Journal, September 1994*, pp. 73-80.

6. N. F. Maxemchuk and S. Low, "Marking Text Documents," *Proceedings of the IEEE International Conference on Image Processing*, Washington, DC, Oct. 26-29, 1997, pp. 13-16.

7. S. H. Low, N. F. Maxemchuk, and A. M. Lapone, "Document Identification for Copyright Protection Using Centroid Detection," *IEEE Transactions on Communications*, Mar. 1998, vol. 46, no.3, pp 372-381.

8. S. H. Low and N. F. Maxemchuk, "Capacity of Text Marking Channel," *IEEE Signal Processing Letters*, vol. 7, no. 12 , Dec. 2000, pp. 345 -347.

9. Ding Huang, Hong Yan, "Interword Distance Changes Represented by Sine Waves for Watermarking Text Images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 12, pp. 1237 -1245, Dec 2001.