การศึกษาเทคนิคไดเร็กซีเควนสเปดสเป็กตรัม เพื่อการใช้งานทางด้านการบีบอัดข้อมูล

ธำรงรัตน์ อมรรักษา [1]

มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี บางมด ทุ่งครุ กรุงเทพฯ 10140

บทคัดย่อ

งานวิจัยนี้ได้อธิบายถึงวิธีการหนึ่งซึ่งใช้ในการบีบอัดข้อมูลโดยที่ ข้อมูลย่อยขนาด *k* บิต จะถูกแบ่งออกมาจากข้อมูลรวมทั้งหมดขนาด *n* บิต และถูกนำกลับไปใส่ไว้ในส่วนของข้อมูลที่เหลืออยู่ขนาด *n-k* บิต เพื่อที่จะสร้างข้อมูลในรูปแบบที่ถูกบีบอัดเอาไว้ ในกระบวนการขยายข้อมูลตัวข้อมูลย่อยขนาด *k* บิตนี้จะถูกกู้กลับคืนมาก่อนจากข้อมูลที่ถูกบีบอัดไว้ หลังจากนั้นจะถูกนำมาใช้ในการกู้ข้อมูลขนาด *n-k* บิต เพื่อใช้ในการรวมข้อมูลทั้งสองเป็นข้อมูลเดิมขนาด *n* บิต วิธีการนี้สามารถบีบอัดข้อมูลได้ในอัตรา *n/n-k* จากหลักการที่ได้กล่าวมานี้ เทคนิคไดเร็กซีเควนสเปดสเป็กตรัม ซึ่งถูกใช้งานอย่างแพร่หลายในระบบสื่อสารไร้สาย และการทำภาพลายน้ำดิจิตอล ถูกนำมาพิจารณาเพื่อการใช้งานทางด้านการบีบอัดข้อมูล งานวิจัยนี้ได้ทำการศึกษาถึงประสิทธิภาพของเทคนิคดังกล่าวต่อการใช้งานที่เกี่ยวข้องกับการบีบอัดข้อมูลโดยใช้การวิเคราะห์ และการจำลองการทำงานที่ค่าตัวแปรต่างๆ บนเครื่องคอมพิวเตอร์ ผลลัพธ์ที่ได้แสดงให้เห็นถึงความเป็นไปได้ในการประยุกต์ใช้งานเทคนิคไดเร็กซีเควนสเปดสเป็กตรัมสำหรับการบีบอัดข้อมูล

[1] *อาจารย์ ภาควิชาวิศวกรรมคอมพิวเตอร์*

A Study of Direct Sequence Spread Spectrum Technique for Data Compression Purpose

Thumrongrat Amornraksa [1]

King Mongkut's University of Technology Thonburi Bangmod, Toongkru, Bangkok 10140

Abstract

This paper describes a compression method in which k bits, taken from n bits of binary bit stream, are embedded into the remaining $n-k$ bits using watermarking-type techniques to generate a compressed data. In decompression process, the embedded k bits are first extracted from the compressed data. The extracted k bits are used to recover the original $n-k$ bits, and both sets of data are then combined to create the n bits of original binary bit stream. With this method, the compression rate of $n/n-k$ can be achieved. Based on this idea, the Direct Sequence Spread Spectrum (DS-SS) technique that has been widely used in wireless communications and digital watermarking applications is considered for the use in a compression scheme. The study was performed by analytical and simulation method to determine how well it performs in data compression applications. The results from our study showed the possibility of using DS-SS technique for data compression purposes.

[1] *Lecturer, Department of Computer Engineering.*

1. Introduction

Techniques of data compression have been developed since Shannon's and Fano's pioneering work in the late 1940s [1]. It deals exclusively with reversible (or lossless) compression, where the decoder can recover exactly what the transmitter sent. There are many impressive approaches to coding for data compression including Shannon-Fano coding, Huffman coding and numerous elaborations such as efficient methods for adaptive Huffman coding, arithmetic coding and Ziv-Lempel methods [2]-[3]. They are used for text, program and database compression, where errors cannot generally be tolerated. For nonreversible (lossy) compression, it is normally used for coding speech, gray-scale or color pictures, where significantly increased compression can be obtained if one is satisfied with a high-fidelity approximation to the original signal rather than bit-for-bit reproduction [4].

Recently, embedding information into multimedia data is a topic that has gained increasing attention. Various digital watermarking algorithms were proposed to embed information bits into images and video signal. Some of them give no increase in transmission bit-rate. When the extra bits are embedded into the video signal, with an appropriate embedding algorithm, the recovery of those extra bits and original video signal may be obtained [5]. Based on this concept, a compression scheme may be constructed by using watermarking-type techniques. Consider a binary bit stream containing n bits, k bits taken from this bit stream can be embedded into the remaining $n-k$ bits, where n and k are integer numbers and $n>k$. This method offers a compression rate of $n/n-k$, provided in the decompression process, both $n-k$ and k bits can be recovered. Although the method gives a small compression rate, since in practice $n>>k$, it might be used along with other existing compression methods, in a cascade manner, to gain a higher total compression rate.

In this paper, we describe a compression method that employs DS-SS technique in the compression process. The technique that is normally used in Spread Spectrum (SS) communications and digital watermarking applications is then studied for the use in data compression. Section 2 reviews the background of the DS-SS technique and its functions while Section 3 explains the methods of how DS-SS technique can be applied in the compression scheme, and the theory behind its operations. Section 4 describes the simulation model used in the experiments, and the results from simulations, analysis and discussions are given in Section 5. Finally, Section 6 provides some concluding remarks.

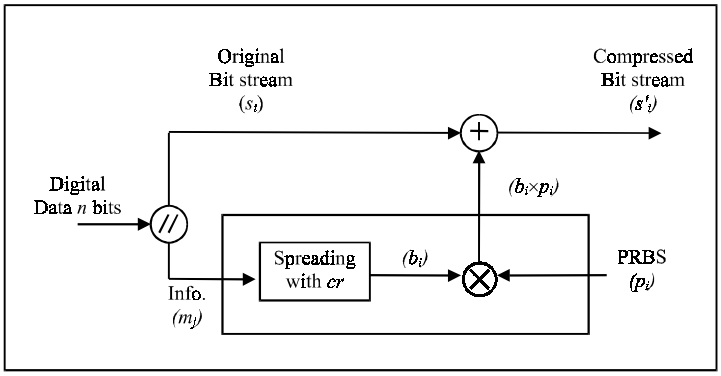
2. Background

In SS communications [6], a low level wideband signal can easily be hidden within the same spectrum as a high power signal where each signal appears to be noise to the other. The heart of these SS systems is a Pseudo-Random Binary Sequence (PRBS). For these DS-SS systems, the original baseband bit stream is multiplied by the PRBS to produce a new bit stream. Only those receivers equipped with correct PRBS can decode the original message. At the receiver, the low level wideband signal will be accompanied by noise, and by using a suitable detector/demodulator with the correct PRBS, this signal can be squeezed back into the original narrow baseband. Because noise is completely random and uncorrelated, the wanted signal can easily be extracted.

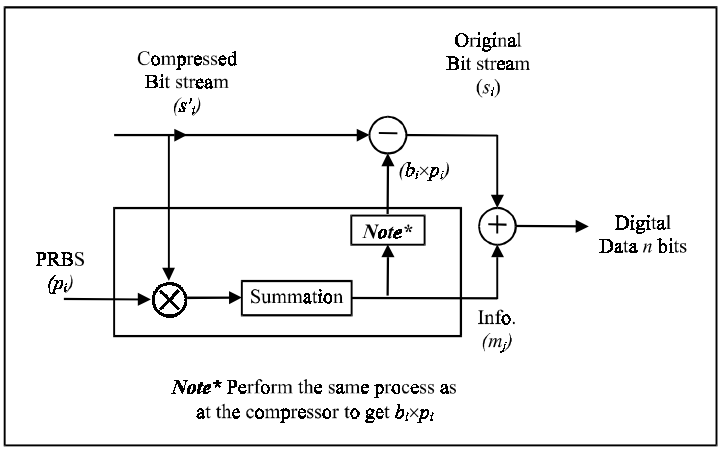
The same principle can be applied in various digital watermarking applications with small modifications, such as those proposed in [7]-[9]. By spreading the information bits and modulating them with a PRBS, the SS signal can be obtained. This signal is then embedded in the video signal below the threshold of perception. However, in the watermarking applications, the information bits will be embedded into the video signal in such a way that the contained watermark signal should be very difficult to be noticed, destroyed, modified by an unauthorized person. The recovery of the embedded SS signal can be accomplished by correlating the modified video signal with the same PRBS that was used in the process of constructing the SS signal. Correlation here is demodulation followed by summation over the width of the chip-rate (the number of blocks over which each information bit is spread). If the peak of the correlation is positive (or respectively, negative), the recovered information bit is a +1 (or -1).

3. Method Description

To compress a binary digital data, see Fig. 1 (a), the data will be divided into two parts. The first part is considered as the baseband bit stream to be hidden in the second part. That is, the k bits of so-called information bits will be added to the $n-k$ bits of so-called original bit stream to obtain a compressed data. Given a key to reproduce the same PRBS in the decompression process, see Fig. 1 (b), the information bits can be extracted. Then the original bit stream can be recovered by subtracting the information bits from the compressed data. At this step, the decompressed original binary data is finally obtained by combining the $n-k$ bits of original bit stream and the k bits of information. The operation of the compression scheme is shown in Fig. 1.



(a) Compression Process



(b) Decompression Process

Fig. 1 The operation of the compression scheme

We now describe the basic steps of adding the information bits to the original bit stream, using a similar technique to that proposed in [8]. We denote a sequence of information bits we want to add to in the original bit stream by (m_j), $m_j \in \{-1, 1\}$. These binary bits are spread by a large factor *cr*, the chip-rate, to obtain the spread sequence (b_i) as follows:

$$b_i = m_j,\ j \bullet cr \leq i < (j+1) \bullet cr$$

The spread sequence (b_i) is then modulated with a PRBS (p_i), $p_i \in \{-1, 1\}$ and then added to the original bit stream s_i, each s_i block contains x bits, yielding the following output of compressed data:

$$s'_i = s_i + p_i \bullet b_i \qquad (1)$$

In decompression process, the recovery of the added information is easily accomplished by multiplying the compressed data with the same PRBS (p_i) that was used in the compressor. The summation over the correlation window i.e. *cr* is as follows:

$$r_j = \sum_{i=j\bullet cr}^{(j+1)\bullet cr-1} p_i \bullet s'_i = \sum_{i=j\bullet cr}^{(j+1)\bullet cr-1} p_i \bullet s_i + \sum_{i=j\bullet cr}^{(j+1)\bullet cr-1} p_i^2 \bullet b_i \qquad (2)$$

The first term on the right-hand side of (2) vanishes if p_i and s_i are uncorrelated and $\sum_{i=j\bullet cr}^{(j+1)\bullet cr-1} p_i = 0$. However, we account for a different number of -1's and 1's in p_i over the interval [$j\bullet cr$, $(j+1)\bullet cr$] by including the term

$$\Delta = \left(\sum_{i=j\bullet cr}^{(j+1)\bullet cr-1} p_i\right) \bullet (\bar{s'_i}) \qquad (3)$$

Then r_j ideally becomes

$$r'_j = \sum_{i=j\bullet cr}^{(j+1)\bullet cr-1} p_i \bullet s'_i - \Delta \approx cr \bullet m_j \qquad (4)$$

and the recovered information bit m'_j = sign (r'_j). This information bits will be spread again with the same chip-rate before being subtracted from the compressed data. The original bit stream can then be reconstructed.

As an example, let the size of the original bit stream be 10 Mb/s, the chip-rate *cr* = 200 and let the block size *x* be 4 bits. Then, the amount at which information bits can be added is approximately 12.5 kb/s. To increase the number of information bits, the chip-rate and the block size should be reduced. However, a smaller block size implies a greater likelihood that subtracting the information bits from the compressed data will not give the original bit stream. In addition, a smaller chip-rate implies a greater likelihood of error in extracting the information bits [6].

4. Simulation Model

Simulations were carried out using C programming language. The block size x was varied from 2-7 bits to represent up to 128 values. The chip-rate was varied from 0 to a value that gives no error in the extracted information. According to eq. (1), the addition between s_i and $p_i \bullet b_i$ can be performed in five different methods, yielding five operations, as follows:

i. $s'_i = si + pi \bullet bi$

ii. $s'_i = s_i$, if $s_i = 0$ and $p_i \bullet b_i = -1$, or $s_i = (2^x - 1)$ and $p_i \bullet b_i = 1$,
Otherwise $s'_i = s_i + p_i \bullet b_i$

iii. $s'_i = s_i$, if $s_i = 0$ and $p_i \bullet b_i = -1$, otherwise $s'_i = (s_i + p_i \bullet b_i) \bmod 2^x$

iv. $s'_i = s_i$, if $s_i = (2^x - 1)$ and $p_i \bullet b_i = 1$, otherwise $s'_i = (s_i + p_i \bullet b_i) \bmod 2^x$

v. $s'_i = (s_i + p_i \bullet b_i) \bmod 2^x$

Table 1 shows five possibilities of s'_i resulting from different adding methods (*i*-*v*), which can be used in the compression scheme. Since each method gives different levels of performance in the decompression processes, they will be investigated to determine a suitable one to be used in practice.

Table 1 : Possible values result from the different adding methods in equation (1) for block size x =2

	$s_i + p_i \bullet b_i$							
s_i	0	0	1	1	2	2	3	3
$p_i \bullet b_i$	-1	1	-1	1	-1	1	-1	1
i.) s'_i	-1	1	0	2	1	3	2	4
ii.) s'_i	0	1	0	2	1	3	2	3
iii.) s'_i	0	1	0	2	1	3	2	0
iv.) s'_i	3	1	0	2	1	3	2	3
v.) s'_i	3	1	0	2	1	3	2	0

As Table 1 indicates, the method i produces some results that are out of the range of the values that the original bit stream can represent, e.g. the value of 4 cannot be represented by 2-bit number, and thus this method will not be used in the simulation. For the remaining methods, the different values of s'_i exist when performing the addition between $s_i = 0$ and $p_i \bullet b_i = -1$, or $s_i = (2^x - 1)$ and $p_i \bullet b_i = 1$. In the experiments, the methods *ii*-*v* were used in the simulations, with the aim of demonstrating how a compression scheme may be constructed as well as how well it performs. The differences when applying each method were then analyzed, based on the simulation results obtained.

The data used in our experiments was pseudo-randomly generated by a function *rand( )* in C language, and then input into the constructed compression scheme as described in Fig. 1 We rather chose a pseudo-random-like data than other types such as text or image since it contains smaller amount of redundancy compared to the others', and therefore is suited for data compression tests.

5. Simulation Results

From the simulation results, the smallest chip-rate with no errors after the extraction process using the adding methods ii and iii are shown in Table 2.

Table 2 : Values of the chip-rate with no errors after the extracting process, at different block sizes

	Block Size *x*					
Chip-rate *cr*	2	3	4	5	6	7
Method *ii*	46	110	455	1100	4150	12000
Method *iii*	190	400	1450	5100	15200	45000
Method *iv*	210	410	1400	4500	16000	43500
Method *v*	α	α	α	α	α	α

It is clear from Table 2 that the adding method *ii* gave the best performance, i.e., needs the smallest value of the chip-rate, especially in the larger block sizes, compared to other methods, and therefore should be used in practice. At this step, the adding methods *ii* and *iii* are chosen for the simulations to measure the performance of the compression scheme at various block sizes. For these block sizes, other values of the chip-rate considered resulted in different values of BER in the extracted information bits, and these values and the underlying line are shown in Fig. 2 and 3, for the adding methods *ii* and *iii* respectively.

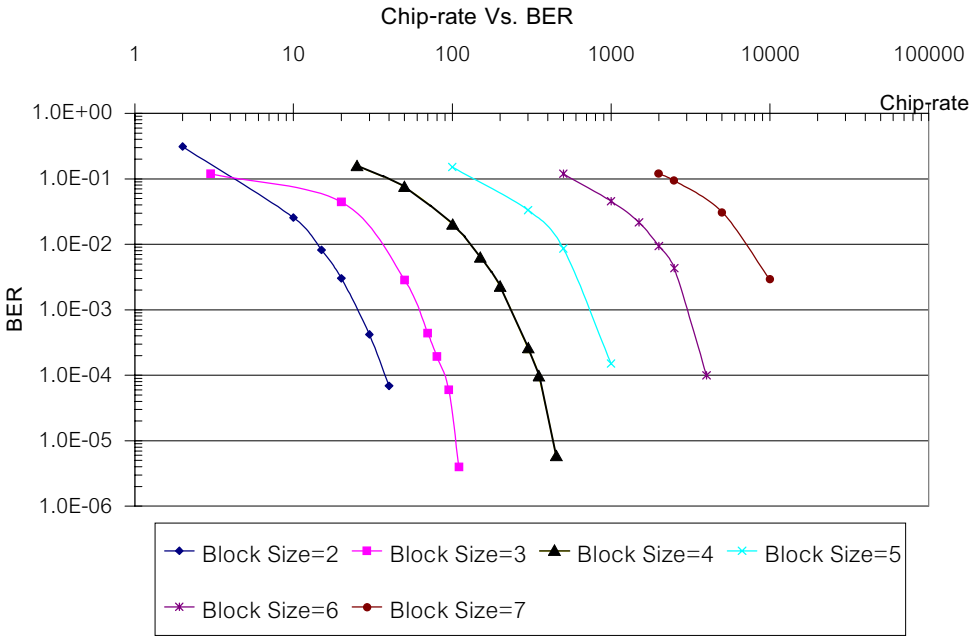


Fig. 2 Bit error rate of the extracted information bits at different block sizes using the adding method *ii*

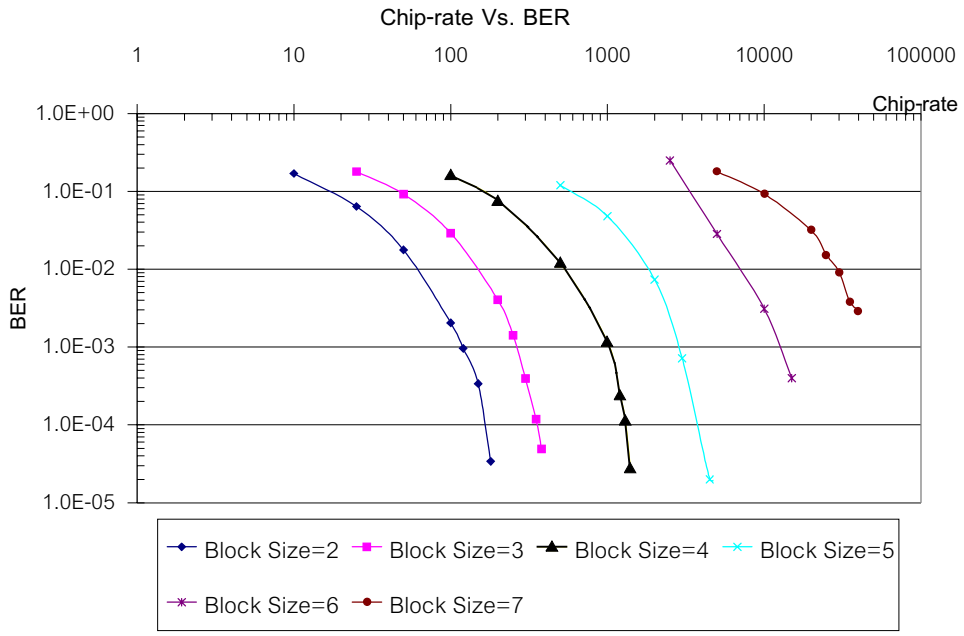


Fig. 3 Bit error rate of the extracted information bits at different block sizes using the adding method *iii*

From both Figures, it can be seen that a larger block size needs a bigger chip-rate to retain the same BER. The adding method *iii* also requires a bigger chip-rate to obtain the same performance as in the method *ii*'s. In addition, since one single bit error in the extracted extra information causes error propagation in the original signal, any value other than a large chip-rate will result in a large BER. Since the adding method *ii* used in the simulation gave the best performance i.e. achieved better compression rate compared to the others, its compression rates at different block sizes are shown below.

Table 3 : Compression rates that the proposed method achieved at different block sizes, using the adding method *ii*

Compression rate	Block Size x				
	2	3	4	5	6
Method *ii*	1.01098	1.00304	1.00056	1.00018	1.00004

Table 3 shows the advantage of using the smaller block sizes which give the better compression rates, and it can be seen that the maximum rate we can achieve from the experiments was approximately 1.01098. That means 1.01098 information bit can be compressed to 1 output bit. However, after extracting the k bits of information, we still need to recover the $n-k$ bits of original bit stream by subtracting the extracted k bits from the compressed data. At this step, assuming the adding method *ii* is used, the problem occurs when performing the subtraction between $s'_i = 0$ and $p_i \bullet b_i = -1$, or $s'_i = (2^x - 1)$ and $p_i \bullet b_i = 1$. The analysis in this circumstance is given below.

Analysis and Discussions

First of all, note that the method *v*, according to Table 1, provides reversible compression, i.e. the original bit stream can be correctly recovered. By using the knowledge of $p_i \bullet b_i$, every value of s'_i can be referred back to s_i, in the same way as one-to-one mapping, while the remaining methods cause some errors in the decompression process. However, the results from Table 2 showed that no matter how large the chip-rate is, when method *v* is used, the embedded information will not be correctly extracted. The reason is because of inaccurate results from the summation over the correlation window, shown in eq. (4), in the decompression process; that is, the decompressor will give a wrong sign of r'_j, and translate to a wrong value of m_j. This event can

be noticed by considering the original bit stream as a random sequence and observing whether its distribution is flat or not. If so, it is likely that the summation results from eq. 4 will lead to a wrong value of m_j. A good example that indicates this notification is shown in Table 1, where the original bit stream is equally distributed; i.e., each value (sample) has the same probability of occurrence. It can now be seen that the summation term of all possible values of $s'_i \times p_i \bullet b_i$ in each adding method is 8, 6, 3, 3 and 0, respectively. For example, in method *ii*, the summation term can be calculated as follows: $(0\times-1) + (1\times1) + (0\times-1) + (2\times1) + (1\times-1) + (3\times1) + (2\times-1) + (3\times1) = 6$. It is obvious that the larger the value of the summation term, the smaller the chip-rate needed to correctly extract the embedded information bits. This analytical observation can be proven by the simulation results from Figures 2 and 3. In contrary, a smaller value of the summation term results in more incorrectly recovered bit stream. The explanation is given below. However, method *v* should not be practically used in the compression scheme.

As mentioned earlier, although the embedded information is correctly obtained when a proper adding methods is used, the recovered original bit stream, after subtracting the information bits from the compressed data, still contains some errors. The reason for this is implicitly shown in Table 1. That is, for example in the adding method *ii*, when $s'_i = 0$ and $p_i \bullet b_i = -1$, the decoder will not be able to determine whether s_i is 0 or 1, and this gives the possibility of making a wrong decision up to 50%. If the block size x is used, the errors occurring in the recovered original signal will be approximately $1/(2^x)$ %. However we can reduce this error rate by using different adding methods e.g. method *iii*. According to Table 1, the remaining errors will be approximately $1/(2^{x+1})$ %. Nevertheless, when the adding method *iii* is used, the chip-rate needs to be increased in order to prevent any error in the process of extracting information bits. The simulation results in Table 2 already verified this fact. The same explanation can also be applied to the adding method *iv*.

Finally, from the analytical results, when implementing the proposed method, we need to consider the characteristics of the input data so that a proper adding method will be used to obtain the optimum compression rate. Since the DS-SS technique is also used in digital watermarking applications, similar to the proposed method, the original video signal will not be able to be recovered from the watermarked video signal. This is because there will be some remaining errors in the video signal caused by the need to ensure that the output bit rate uses the same bandwidth for its symbols, as does the original video signal. However, for this kind of application, the drawback can be ignored since only embedded information is required to be recovered for identification purpose, not the original video signal.

6. Conclusions

It was shown experimentally and analytically in this paper that the DS-SS techniques may be used for data compression purposes, as long as the probability of occurrence of values 0 and 2^x-1 in the original bit stream approaches zero. It was also shown that different adding methods used in the compression scheme not only gave different values of chip-rate that enables the embedded information bits to be correctly extracted, but also gave different compression rates.

Another point that is worth considering is that when all parameters are properly selected, the proposed method can be fitted with any application that employs channel coding. It is shown in [10] that any errors which occur at the receiver's side both communication channel errors and any resulting from the decompression process will be detected and corrected by the channel decoder. One application in access control systems for broadcasting networks which implements the similar method can be found in [11]. Another application that may be well suited to the scheme is digital TV broadcasting, where large number of extra bits can be embedded into the signal before being broadcast.

7. Acknowledgment

The author thanks the KMUTT Research Fund for financial support throughout this work.

References

1. Lelewer, D. A. and Hirschberg, D. S., 1987, "Data Compression", *ACM Computing Surveys*, Vol. 19, No. 3, September, pp. 261-296.

2. Witten, I. H., Neal, R. M., and Cleary, J. G., 1987, "Arithmetic Coding for Data Compression", *Communication ACM*, Vol. 30, No. 6, June, pp. 520-540.

3. Ziv, J. and Lempel, A., 1977, "A Universal Algorithm for Sequential Data Compression", *IEEE Transaction on Information Theory*, IT-23, No. 3, May, pp.337-343.

4. Ghanbari, M., 1999, "*Video Coding: An Introduction to Standard Codecs*", IEE Telecommunications Series 42, The Institution of Electrical Engineers, London, United Kingdom.

5. Hartung, F. and Girod, B., 1997, "Digital Watermarking of MPEG-2 Coded Video in the Bitstream Domain", *Proceeding of ICASSP 97*, Vol. 4, Munich, Germany, April, pp. 2621-2624.

6. Pickholtz, R., Schilling, D., and Millstein, L., 1982, "Theory of Spread Spectrum Communications A Tutorial", *IEEE Transaction on Communication*, Vol. COMM-30, pp. 855-884.

7. Cox, I., Kilian, J., Leighton, T., and Shamoon, T., 1997, "Secure Spread Spectrum Watermarking for Multimedia", *IEEE Trans. on Image Processing*, Vol. 6, No. 12, December, pp. 1673-1687.

8. Hartung, F. and Girod, B., 1998, "Watermarking of Uncompressed and Compressed Video", *Signal Processing*, Vol. 66, No. 3 (Special issue on Watermarking), May, pp. 283-301.

9. George, M., Chouinard, J-V., and Georganas, N., 1999, "Digital Watermarking of Images and Video using Direct Sequence Spread Spectrum Techniques", *Proceeding of the 1999 IEEE Canadian Conference on Electrical and Computer Engineering*, Vol. 1, pp. 116-121.

10. Amornraksa, T., 2000, "Transmitting Extra Information Bit using Direct-sequence Spread Spectrum", *Proceedings of the Third International Symposium on Wireless Personal Multimedia Communications*, Thailand, November 12-15, pp. 76-80.

11. Amornraksa, T., Burgess, D. R. B., and Sweeney, P., 1999, "An Encoding Scheme for Dual Level Access to Broadcasting Networks", *Proceeding of the Seventh IMA International Conference on Cryptography and Coding, Cirencester*, UK, December, LNCS 1746, pp. 114-118.