ELSEVIER

Pattern Recognition Letters 19 (1998) 461–468

Pattern Recognition Letters

Large encrypting binary images with higher security

Kuo-Liang Chung *, Lung-Chun Chang [1]

Department of Information Management, National Taiwan University of Science and Technology, No. 43, Section 4, Keelung Road, Taipei 10672, Taiwan, ROC

Received 30 January 1997; revised 30 October 1997

Abstract

In this paper, we present a new approach for encrypting binary images. Putting different scan patterns at the same level in the scan tree structure and employing a two-dimensional run-encoding (2DRE) technique, our encryption method can encrypt images with higher security and good compression ratio when compared to the previous results. Detailed security analysis from the combinatorial viewpoint is also given. Some experimentations are carried out to illustrate the good performance of our proposed method.

Keywords: Image encryption; Run-length compression; Security; SCAN language; Spatial data structure

1. Introduction

Encrypting images is an important issue in network communication and pictorial protection. The SCAN language, which is a context-free language to describe and generate a wide range of array accessing algorithms from a set of scan patterns, was first introduced by Bourbakis (1986). Then, Bourbakis et al. (1989) presented a parallel implementation scheme for the SCAN language. Later, Bourbakis and Alexopoulos (1992) presented a new picture data encryption for binary images using the scan patterns. Putting the quadtree compression scheme into the result (Bourbakis and Alexopoulos, 1993), an improved method while preserving the same security was presented by Chang and Liu (1994). Recently, Alexopoulos et al. (1995) presented an encryption method for encrypting 2-D gray scale images by using a larger class of fractals.

This paper presents a new approach for encrypting binary images. Putting different scan patterns at the same level in the scan tree structure and employing a two-dimensional run-encoding (2DRE) technique, our encryption method can encrypt images with higher security and good compression ratio when compared to the previous results (Bourbakis and Alexopoulos, 1992; Chang and Liu, 1994). Detailed security analysis from the combinatorial viewpoint and the formal related algorithms are also given. Some experimentations are carried out to illustrate the good performance of our method.

* Corresponding author. Email: klchung@cs.ntust.edu.tw. Dr. Chung is also with the Graduate Program of Information Engineering.

[1] Email: lcchang@math.thu.edu.tw.

0167-8655/98/$19.00
PII: S0167-8655(98)00017-8

2. The proposed encryption scheme

In this section, we first describe the quadtree spatial data structure (Klinger and Dyer, 1979; Samet, 1990a,b), then we present our modified SCAN language to allow to put different scan patterns at the same level in the scan quadtree structure in order to achieve higher security. At last, we compress the encrypted image using a 2DRE technique in order to achieve good compression ratio. Our proposed encryption and compression scheme as well as the decompression and decryption scheme are shown in Fig. 1. Since the decryption and decompression scheme is the reverse of the encryption and compression, we only focus on the encryption and compression scheme.

Quadtree represents a binary image by using a 4-ary tree structure. It starts from the root node of the quadtree. If the entire image is totally black or white, the image is represented by one node, the root node. Otherwise, the root node is grey and the image is split into four equal-sized subimages, one for each quadrant, that are labeled *nw* (northwest), *ne* (northeast), *sw* (southwest), and *se* (southeast), respectively. This subdivision process is then repeated recursively for each of the four subimages until the subimage is totally black or white. The leaf node in the quadtree is called an external or leaf node; the grey node is called an internal node. Naturally, the quadtree can be implemented by a pointer-type data structure. Here, the storage space required is defined to be the number of nodes needed in the quadtree. Given a binary image with $2^3 \times 2^3$ pixels as shown in Fig. 2(a), the corresponding quadtree is shown in Fig. 2(b), where s_1 denotes the root.

2.1. The modified SCAN language achieving higher security

In this subsection, our modified SCAN language is presented to achieve higher security when compared to the previous results (Bourbakis and Alexopoulos, 1992; Chang and Liu, 1994). We take the example of Fig. 2 to explain how the modified SCAN language works.

Initially, suppose the size of the input image is $2^n \times 2^n$. The modified SCAN language is defined by $G = \langle V_N, V_T, P, S \rangle$, where V_N $(= \{S, \bigcup_{i=1}^{n} L_i\})$ is the set of nonterminal symbols and L_i denotes the set of different scan patterns at the ith level in the scan quadtree; V_T $(= \{\bigcup_{i=1}^{n} \Omega_i^{4^{i-1}} \mid \Omega_i = \{R_j^i \mid 1 \leqslant j \leqslant 4^{i-1}\}$ and $\Omega_i^{4^{i-1}} = \Omega_i \Omega_i \cdots \Omega_i$ (4^{i-1} times Ω_i)$\})$ is the set of terminal symbols and R_j^i is one of the twenty-four scan patterns which are defined in Fig. 3, where SP_i, $0 \leqslant i \leqslant 23$, denotes the ith scan pattern for the 2×2 pattern window; S is the start symbol; P is the set of production rules and is defined as follows:

$$S \to L_1 L_2 \cdots L_n$$
$$L_i \to R_1^i R_2^i \cdots R_{4^{i-1}}^i \quad \text{for } 1 \leqslant i \leqslant n.$$

Previously, Bourbakis and Alexopoulos (1992) presented a picture data encryption for binary images by using the SCAN language. Their method puts the same scan pattern at the same level. Therefore, the

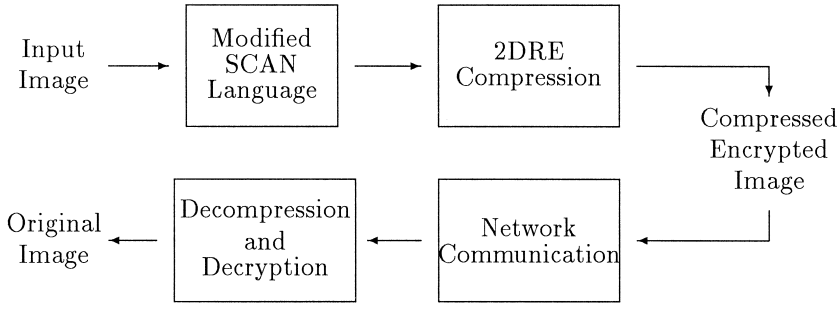


Fig. 1. Block diagram of the proposed encryption and decryption scheme.

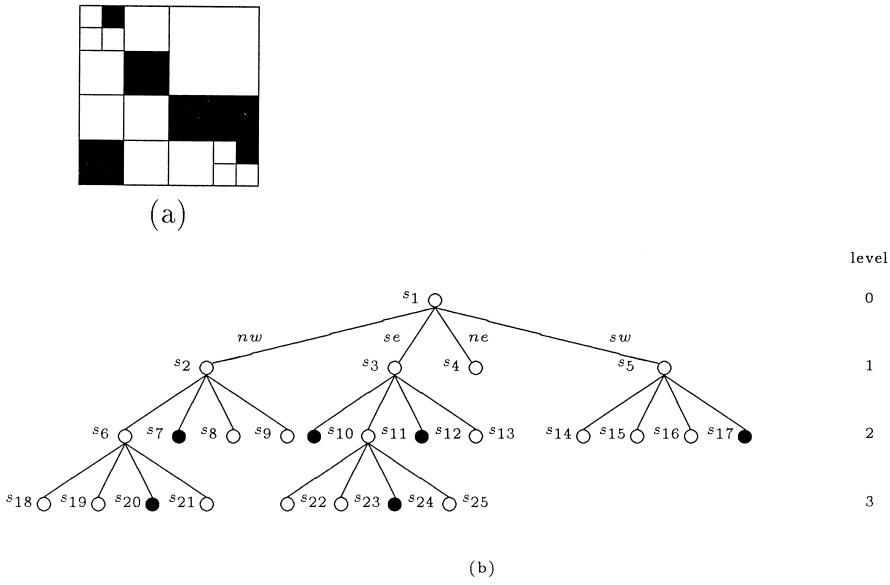


Fig. 2. (a) A $2^3 \times 2^3$ binary image. (b) The corresponding quadtree.

set of production rules in their SCAN language is defined to be

$$S \to L_1 L_2 \cdots L_n$$
$$L_i \to R^i \quad \text{for } 1 \leqslant i \leqslant n,$$

where $R^i \in \{SP_i \mid 0 \leqslant i \leqslant 23\}$. The advantage of our modified SCAN language is that it allows to put different scan patterns at the same level. Thus, we can increase the number of combinations of different scan patterns at each level easily. Consequently, we increase the security in the encrypted images significantly.

Based on our modified SCAN language, applying the production rules as shown in Fig. 4 to the scan quadtree in Fig. 2(b), Fig. 5 is the scanned encrypted image displayed by using the raster scanning method.

We now analyze the security from the combinatorial viewpoint. Given a $2^n \times 2^n$ image and its corresponding complete quadtree, since there is only one node at level 0, the root node, in the scan quadtree structure, there is only one combination for the scan patterns. At level 1 in the same quadtree structure, there are four nodes, so the number of combinations

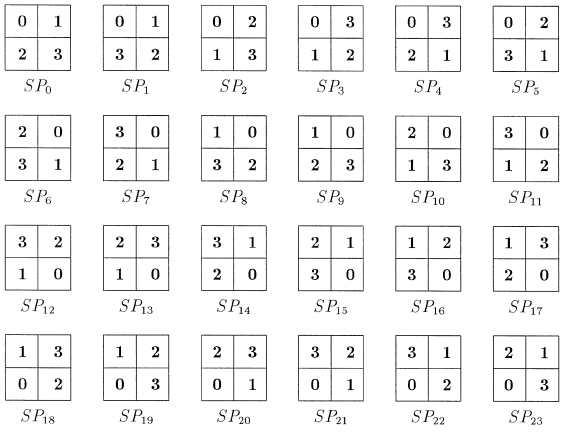


Fig. 3. The twenty-four scan patterns.

$$S \to L_1 L_2 L_3$$
$$L_1 \to R_1^1$$
$$L_2 \to R_1^2 R_2^2 R_3^2 R_4^2$$
$$L_3 \to R_1^3 R_2^3 R_3^3 R_4^3 R_5^3 R_6^3 R_7^3 R_8^3 R_9^3 R_{10}^3 R_{11}^3 R_{12}^3 R_{13}^3 R_{14}^3 R_{15}^3 R_{16}^3$$
$$R_1^1 = SP_1.$$
$$R_1^2 = SP_{23},\ R_2^2 = SP_2,\ R_3^2 = SP_4,\ R_4^2 = SP_7$$
$$R_1^3 = SP_1,\ R_2^3 = SP_{11},\ R_3^3 = SP_{13},\ R_4^3 = SP_1$$
$$R_5^3 = SP_4,\ R_6^3 = SP_1,\ R_7^3 = SP_0,\ R_8^3 = SP_7$$
$$R_9^3 = SP_1,\ R_{10}^3 = SP_{10},\ R_{11}^3 = SP_1,\ R_{12}^3 = SP_{21}$$
$$R_{13}^3 = SP_{11},\ R_{14}^3 = SP_1,\ R_{15}^3 = SP_{13},\ R_{16}^3 = SP_{15}$$

Fig. 4. The production rules.

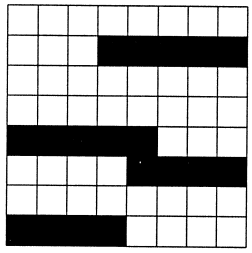


Fig. 5. The scanned encrypted image by the raster method.

for the scan patterns is 24 (see Fig. 3). At level 2 in the same quadtree structure, there are sixteen nodes (four 2×2 pattern windows). Every 2×2 pattern window has 24 combinations, so the total number of combinations is 24^4 at level 2. Similarly, there are $24^{4^{i-1}}$ combinations for the scan patterns at level i, $1 \leqslant i \leqslant n$, in the scan quadtree structure when we put different scan patterns into this level. Using our proposed method, the security of that $2^n \times 2^n$ image, say $N_1(n)$, is equal to the total number of combinations and yields

$$N_1(n) = 1 \times 24 \times 24^4 \times \cdots \times 24^{4^{n-1}} = 24^{(4^n-1)/3}.$$

In the previous results (Bourbakis and Alexopoulos, 1992; Chang and Liu, 1994), there are only 24 combinations for each level i, $1 \leqslant i \leqslant n$, since each quadrant adopts the same scan pattern at the same level. Thus, the security in those results (Bourbakis and Alexopoulos, 1992; Chang and Liu, 1994) for a $2^n \times 2^n$ image, say $N_2(n)$, is equal to

$$N_2(n) = 1 \times 24 \times 24 \times \cdots \times 24 \,(n \text{ times } 24) = 24^n.$$

In fact, under the 2×2 scan pattern window, the encryption scheme in (Bourbakis and Alexopoulos, 1992; Chang and Liu, 1994) is a special case of our proposed scheme. Fig. 6 illustrates the number of combinations at each level in the scan quadtree structure among the previous results (Bourbakis and Alexopoulos, 1992; Chang and Liu, 1994) and ours.

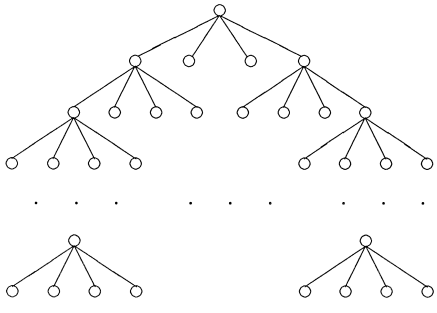
Table 1
Security comparison of the results in (Bourbakis and Alexopoulos, 1992; Chang and Liu, 1994) and ours

n	Previous results		This paper	
	$N_2(n)$	$T_2(n)$	$N_1(n)$	$T_1(n)$
1	24	$7.610 \cdot 10^{-13}$	24	$7.610 \cdot 10^{-13}$
2	$5.760 \cdot 10^{2}$	$1.826 \cdot 10^{-11}$	$7.963 \cdot 10^{6}$	$2.525 \cdot 10^{-7}$
3	$1.382 \cdot 10^{4}$	$4.384 \cdot 10^{-10}$	$9.648 \cdot 10^{28}$	$3.059 \cdot 10^{15}$
4	$3.318 \cdot 10^{5}$	$1.052 \cdot 10^{-8}$	$2.079 \cdot 10^{117}$	$6.594 \cdot 10^{103}$
5	$7.963 \cdot 10^{6}$	$2.525 \cdot 10^{-7}$	$4.488 \cdot 10^{470}$	$1.423 \cdot 10^{457}$
6	$1.911 \cdot 10^{8}$	$6.060 \cdot 10^{-6}$	$9.735 \cdot 10^{1883}$	$3.087 \cdot 10^{1870}$
7	$4.586 \cdot 10^{9}$	$1.454 \cdot 10^{-4}$	$2.156 \cdot 10^{7537}$	$6.836 \cdot 10^{7523}$
8	$1.101 \cdot 10^{11}$	$3.490 \cdot 10^{-3}$	$5.183 \cdot 10^{30150}$	$1.643 \cdot 10^{30137}$
9	$2.642 \cdot 10^{12}$	$8.377 \cdot 10^{-2}$	$1.732 \cdot 10^{120604}$	$5.492 \cdot 10^{120590}$

For one encrypted image, the probability that we want to decrypt it successfully depends on the total number of combinations employed in the encrypted image. Suppose the computer used has 100 MIPS (Million Instructions Per Second) computational capability. Here, we assume one instruction can perform one combination for putting one specific scan pattern. Therefore, for the $2^n \times 2^n$ encrypted image, the security of the encrypted image is define to be

$$T_i(n) = \frac{N_i(n)}{1000000 \times 60 \times 60 \times 24 \times 365} \text{ (years)}, \quad 1 \leqslant i \leqslant 2.$$

Table 1 illustrates the security comparison of the previous results (Bourbakis and Alexopoulos, 1992;



level	previous results	this paper
0	1	1
1	24	24
2	24	24^4
3	24	24^{4^2}
⋮	⋮	⋮
$n-1$	24	$24^{4^{n-2}}$
n	24	$24^{4^{n-1}}$

Fig. 6. The number of combinations at each level.

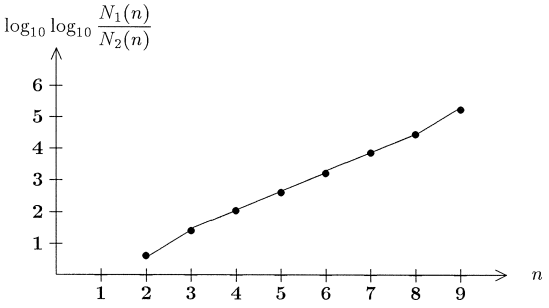


Fig. 7. Security performance.

Chang and Liu, 1994) and ours. Fig. 7 illustrates the related performance, where the function ''$\log_{10}\log_{10}$'' is a monotonically increasing function. Since the security of our method over those in (Bourbakis and Alexopoulos, 1992; Chang and Liu, 1994) is equal to

$$\frac{N_1(n)}{N_2(n)} = 24^{(4^n-3n-1)/3} \geqslant 1$$

for $n \geqslant 1$, our security is always higher than those in (Bourbakis and Alexopoulos, 1992; Chang and Liu, 1994).

Following the above description, our encryption algorithm is listed below.

Algorithm: Encryption
Input:
scan_pattern[24][2][2] /* twenty-four scan patterns, SP_i's, $0 \leqslant i \leqslant 23$ */
image[M][M] /* binary image with size $M \times M$, where $M = 2^N$ */
scan[] /* array saving the sequence of scan patterns */
Output:
encrypted[M][M] /* the encrypted image with size $M \times M$ */
begin
for $x := 0$ to $M-1$ do /* x denotes the row index */
for $y := 0$ to $M-1$ do /* y denotes the column index */
for $i := 0$ to 1 do
for $j := 0$ to 1 do
$r[i][j] = scan\_pattern[scan[0][i][j]$
/* assign scan pattern to level 1 */
end for
end for
$b[x][y] = 0$ /* matrix b saves the scan order for *image*[x][y] */
index $= 0$ /* calculate the position of *scan*[] for (x,y) coordinate */
call *get_combination_number*(x,y,M) /* calculate the value of $b[x][y]$ */
$[quot, rem] = divide(b[x][y], M)$ /* as a quotient, *quot* denotes the row number; as a remainder, *rem* denotes the column number */
encrypted[*quot*][*rem*] = *image*[x][y] /* record the encrypted data */
end for
end for
end
function *get_combination_number*(xx, yy, n) /* calculate the value of $b[x][y]$ */
begin
$p = xx/(n/2)$ /* find relative (p,q) coordinate by using scan window */
$q = yy/(n/2)$ /* when (x,y) coordinate at level $(\log M - \log n + 1)$ */
$b[x][y] = b[x][y] + n^2 \times r[p][q]/4$
if xx is greater than or equal to $n/2$
$xx = xx - n/2$
end if
if yy is greater than or equal to $n/2$
$yy = yy - n/2$
end if
$n = n/2$
if n is greater than 1
index $= (4 \times index + 1) + r[p][q]$
for $s := 0$ to 1 do
for $t := 0$ to 1 do
$r[s][t]$
$= scan\_pattern[scan[index]][s][t]$
/* change the scan pattern window */
end for
end for
call *get_combination_number*(xx, yy, n)
end if
end

It seems that the CPU time required in the above encryption algorithm is more than those of the previous ones (Bourbakis and Alexopoulos, 1992; Alexopoulos et al., 1995) because we need some extra time to process the different scan patterns at the

same level. However, the proposed encryption algorithm increases the total number of combinations and it makes the breaking work harder,

2.2. Employing 2DRE into encrypted images

The well-known 2DRE technique is used to compress the binary encrypted image in terms of segmented black or white pixel-strings based on the scan order. We take an example to illustrate how it works.

Initially, we record the binary value of the first scanning pixel and save the length of the segmented black or white pixel-strings successively based on the scan order. Suppose each length is represented by a fixed-length binary representation with l bits. Then, the total bits required is equal to $(1 + s) \times l$, where s denotes the number of segmented pixel-strings. The compression ratio is defined by

$$\text{Compression ratio} = \frac{R_o}{R_c},$$

where R_o is the total number of bits used in the image before compression and R_c is the total number of bits used in the image after compression.

Return to the scanned encrypted image of Fig. 5. Throughout the remainder of this paper, the scanned encrypted image and the encrypted image are used interchangeability. Based on the row-major scanning order, the bit-string of the scanned encrypted image is shown below:

0000000000011111
0000000000000000
1111100000001111
0000000011110000

Using the 2DRE technique, the compressed representation is

0 11 5 16 5 7 4 8 4 4.

Suppose each element is represented by 5 bits, then the compression ratio is 1.28 ($= 64/50$) since saving Fig. 5 needs 64 ($= 8 \times 8$) bits.

If we use a quadtree encryption scheme (Chang and Liu, 1994) (QES for short), it starts from the lowest level of the quadtree. At the lowest level, if the four nodes have the same parent node and the color of each node is white (black), these four nodes are compressed to the parent node and the representation of the parent node is $0x$ ($1x$). Similarly, at the upper level, if the four nodes with the same parent node have the same representations $0x$'s ($1x$'s), they are compressed to a node and the representation is $0xx$ ($1xx$). This merging process is then repeated recursively until it cannot be merged. The compressed representation of Fig. 2(a) by using the 2×2 raster scan pattern, SP_0 of Fig. 3, is shown below:

$01000x0x1x0xx0x0x1x0x1x1x0x0100.$

Since 2 bits can represent 0, 1, and x, the compressed representation using the QES method requires 62 bits and the compression ratio is 1.032 ($= 64/62$). In the QES method, if we use another different 2×2 scan pattern at the same level, the compressed ratio is not changed. Since the image structure of the encrypted image is very messy, the 2DRE approach will have better compression performance when compared to the quadtree approach in most cases.

Following the above description, our compression algorithm is listed below.

Algorithm: Compression
Input:
  encrypted[M][M] /* the encrypted image with size $M \times M$, where $M = 2^N$ */
Output:
  run[] /* array saving the sequence of run-encoding */
  total_bits /* the total bits required in the compressed image */
begin
  $i = j = 1$
  length = 1
  maximum_length = 0 /* calculate the maximum element in *run*[] */
  run[0] = *encrypted*[0][0] /* record the first scanned value */
  fixed = *encrypted*[0][0]
  while i is less than M^2 do
  /* record the length of each pixel-string */
    [*quot*, *rem*] = *divide*(i, M)
    if fixed is equal to *encrypted*[*quot*][*rem*]
      /* test whether two consecutive pixels are equal */
      length = *length* + 1
    else
      $j = j + 1$

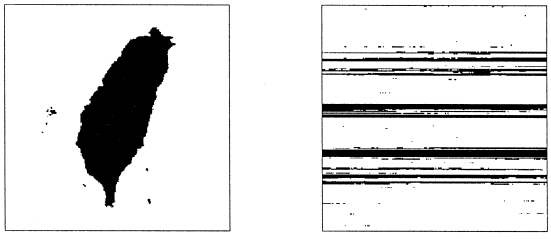
```
        fixed = encrypted[quot][rem]
        length = 1
      end if
      run[j] = length
      i = i + 1
      if maximum_length is less than run[j]
      /* find the maximum length */
        maximum_length = run[j]
      end if
    end while
    maximum_bits = ⌈log2 maximum_length⌉
    total_bits = maximum_bits × (j + 1)
end
```

Their proposed methods (Bourbakis and Alexopoulos, 1992; Alexopoulos et al., 1995) did not involve the compression technique. However, we spend some extra time to perform the above compression algorithm. Thus, the proposed compression algorithm reduces the space and time when transporting the compressed image in network communication.

After performing our encryption and compression scheme, we thus have an encrypted and compressed image. In fact, it is rather straightforward to decompress and decrypt the encrypted and compressed image in a reverse manner.

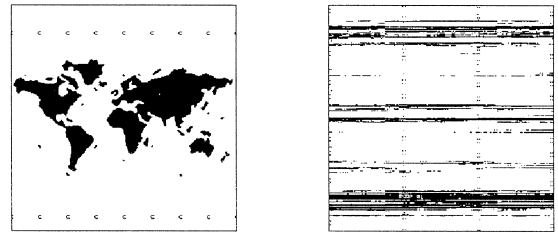
3. Experimentations

We take two real binary images, say the Taiwan map and the world map, as shown in Fig. 8(a) and Fig. 9(a), respectively, to evaluate the performance.



(a) (b)

Fig. 8. (a) The Taiwan map; (b) the encrypted image of the Taiwan map.



(a) (b)

Fig. 9. (a) The world map; (b) the encrypted image of the world map.

The size of both images are 256×256 ($= 2^8 \times 2^8$) and the total number of combinations of any one image is equal to $5.183 \cdot 10^{30150}$ according to Table 1. That is, an illegal receiver may take $1.643 \cdot 10^{30137}$ years to decrypt the image unless the receiver knows our modified SCAN language and the scan patterns in advance.

We have shown that 4^{i-1} different scan patterns can be assigned at level i, $1 \leqslant i \leqslant 8$. For simplicity, the scan patterns used in the Taiwan map at level i, $1 \leqslant i \leqslant 8$, are defined to be $SP_{(j \bmod 3)}$ for $0 \leqslant j < 4^{i-1}$. The encrypted Taiwan image is shown in Fig. 8(b). The original Taiwan map requires 65536 ($= 2^{16}$) bits. When we employ the 2DRE technique in the encrypted Taiwan map, the compressed image requires 6446 bits. We find that the compression ratio is 10.167 ($= 65536/6446$). The scan patterns used in the world map at level i, $1 \leqslant i \leqslant n$, are all defined to be SP_4's. The encrypted world image is shown in Fig. 9(b). Using the 2DRE technique, the compressed world image requires 17440 bits and the compression ratio is 3.758 ($= 65536/17440$). For saving the space of context, we omit the detailed source codes.

Table 2
Compression ratio comparison

	Taiwan map		World map	
	QES	ours	QES	ours
R_o	65536	65536	65536	65536
R_c	7458	6576	18454	17700
compression ratio	8.787	9.966	3.551	3.702

For evaluating the compression ratio performance between ours and the one of Chang and Liu (1994), we use the same scan pattern, say SP_{19}, at each level in the Taiwan map; we use the same scan pattern, say SP_8, at each level in the world map. The compression ratio performance is shown in Table 2, where R_o, R_c, and the compression ratio have been defined previously. In this table, it is observed that our compression ratios for these two real images are better than those of Chang and Liu (1994).

4. Conclusions and discussions

The significance of image encryption is due to its use in image protection and channel communication. The major contribution of this paper is that we have presented our encryption method with higher security and good compression ratio when compared to the previous results in (Bourbakis and Alexopoulos, 1992; Chang and Liu, 1994). In addition, the security analysis from the combinatorial viewpoint and some experimentations have been carried out to demonstrate the advantages of the proposed method.

For the gray level image with 256 gray levels, we employ the bit-plane slicing method (Gonzalez and Woods, 1992). Since each pixel of the gray level image is represented by 8 bits, this image can be viewed as eight binary planes. For example, plane 0 contains all the highest order bits of all the pixels in the image; plane 7 contains all the lowest order bits, and so on. The proposed encryption and compression methods presented in this paper could be applied to each binary plane to achieve the same advantages. Further, combining these eight encrypted and compressed representations leads to the encryption and compression scheme for gray level images.

Bourbakis et al. (1995) presented a hardware design and implementation for the SCAN scheme. Recently, Bourbakis (1997a,b,c) presented a generalization of the SCAN language and its applications. How to employ this generalization of SCAN language and the hardware design scheme into our result is our future research topic. Plugging the analytic cryptographic techniques (Schneier, 1994) into the results of this paper is another research topic.

Acknowledgements

The authors thank the two referees, Professor E. Backer, and F.D. Mesman for their constructive comments and criticism that improve the presentation and quality of this paper. Specifically, we would like to thank Dr. N. Bourbakis for providing us many related references. This research was supported in part by the National Science Council of R.O.C. under contracts NSC87-2213-E011-001 and NSC87-2213-E011-003.

References

Alexopoulos, C., Bourbakis, N., Ioannou, N., 1995. Image encryption method using a class of fractals. J. Electronic Imaging 4 (3), 251–259.

Bourbakis, N., 1986. A language for effective accessing of a 2D array. In: Proc. IEEE Workshop on LFA, Singapore, pp. 52–58.

Bourbakis, N., 1997a. Image data encryption and compression using G-SCAN. In: IEEE Conf. on Systems, Man and Cybernetics, Orlando, FL, vol. 2, pp. 1117–1120.

Bourbakis, N., 1997b. Image compression using G-SCAN. Pattern Recognition, to appear.

Bourbakis, N., 1997c. The family of SCAN languages and applications. Tech. Rept. TR-1993.

Bourbakis, N., Alexopoulos, C., Klinger, A., 1989. A parallel implementation of the scan language. Internat. J. Comput. Languages 14 (4), 239–254.

Bourbakis, N., Alexopoulos, C., 1992. Picture data encryption using scan patterns. Pattern Recognition 25 (6), 567–581.

Bourbakis, N., Alexopoulos, C., 1993. Image data compression using SCAN patterns. In: Proc. SPIE – The Internat. Society for Optical Engineering, vol. 1903, pp. 255–265.

Bourbakis, N., Brause, R., Alexopoulos, C., 1995. SCAN image compression/encryption hardware system. In: Proc. SPIE – The Internat. Society for Optical Engineering, vol. 2419, pp. 419–428.

Chang, K.C., Liu, J.L., 1994. An image encryption scheme based on quadtree compression scheme. In: Proc. 1994 Internat. Comput. Symp., Taiwan, pp. 230–237.

Gonzalez, R.C., Woods, R.E., 1992. Digital Image Processing. Addison-Wesley, New York, pp. 166-171.

Klinger, A., Dyer, C.R., 1979. Experiments in picture representation using regular decomposition. Comput. Graphics Image Process. 5 (1), 68–105.

Samet, H., 1990. The Design and Analysis of Spatial Data Structures, Addison-Wesley, New York.

Samet, H., 1990. Applications of Spatial Data Structures, Addison-Wesley, New York.

Schneier, B., 1994. Applied Cryptography, John Wiley & Sons, New York.