Available online at www.sciencedirect.com

SCIENCE DIRECT®

Pattern Recognition 38 (2005) 2578–2586

PATTERN RECOGNITION

THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

An efficient algorithm for computing moments on a block representation of a grey-scale image[☆]

Kuo-Liang Chung[*], Ping-Chin Chen

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

Received 15 January 2004; received in revised form 11 April 2005; accepted 11 April 2005

Abstract

Computing lower order moments is important in image processing. Suppose the input grey image with size $N \times N$ has been compressed into the block representation where the number of blocks is K, commonly $K < N^2$ due to the compression effect. This correspondence presents an efficient algorithm for computing lower order moments on the block representation directly. Our proposed algorithm takes $O(K)$ time which is proportional to the number of blocks. Experimental results reveal the computational advantage of our proposed algorithm. In addition, the results of this paper can be viewed as a generalization of the previous result by Spiliotis and Mertzios for computing lower order moments from the binary image domain to the grey image domain.

© 2005 Pattern Recognition Society. Published by Elsevier Ltd. All rights reserved.

Keywords: Block representation; Grey image; Lower order moments; Moment computation

1. Introduction

Computing moments on a whole image is important in image processing [1,2]. Given an $N \times N$ image, let $f(x, y)$ denote the grey level of the pixel at location (x, y) for $0 \leqslant x, y \leqslant N-1$. The $(p+q)$th order moment [1,2] is defined as

$$m_{pq} = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} x^p y^q f(x, y). \tag{1}$$

Among these different orders of calculated moments in Eq. (1), lower order moments for $0 \leqslant p+q \leqslant 3$ are especially useful in several applications, such as determining motion parameters of a deformable object [3], deskewing rotationally symmetric shapes [4], determining the centroid and the major axis of an object [1,2], moment-preserving thresholding [5], and recognizing visual patterns by moment invariants [6]. Due to these applications, throughout this paper, we only focus on computing lower order moments. Computing lower order moments on the original grey image domain can be done in $O(N^2)$ time where the time complexity notation $O(N^2)$ denotes the required time complexity $f(N)$ such that there exist two constant terms N_0 and c satisfying $f(N) \leqslant cN^2$ for $N \geqslant N_0$ [7].

Suppose the input binary image of size $N \times N$ has been partitioned into a set of K rectangular blocks where each block is totally black or totally white. Under this block representation of a binary image, Spiliotis and Mertzios [8] presented an efficient algorithm for computing general order moments on the block representation directly. Some elegant formulas are derived in Ref. [8] to speed up the computation of moments. Considering the computation of lower order moments, their algorithm can be done in $O(K)$ time.

[☆] Supported by Contracts NSC91-2213-E011-022 and NSC92-2213-E011-080.

[*] Corresponding author. Tel.: +886 2 27 376 771; fax: +886 2 27 301 081.

E-mail address: klchung@cs.ntust.edu.tw (K.-L. Chung).

Later, Flusser [9] presented an improved way to modify the result by Spiliotis and Mertzios. Note that once a binary image has been partitioned into a set of blocks, then these partitioned blocks can be reused for different kinds of block-based image algorithms and it can lead to computation-saving due to the compression effect, i.e. $K < N^2$, since we only need to consider the relevant blocks instead of considering the whole original image.

In real applications, we face grey images more often than binary images. Based on the bintree partition principle and the linear interpolation method, a grey image can be partitioned into a set of blocks [10] which can be viewed as a promising spatial data structure that extends the previous spatial data structures [11] from the binary image domain to the grey image domain. Under this block representation, an $O(N\sqrt{K})$-time algorithm [12] was presented to compute lower order moments. In Ref. [12], based on the slice integration technique, the local moment of each block B_i, $1 \leqslant i \leqslant K$, can be computed in $O(L_i)$ time where L_i denotes the perimeter of the block B_i. By the Cauchy–Schwarz inequality, it has been shown that the total time required in computing lower order moments of all the blocks is bounded by $O(N\sqrt{K})$. How to reduce the time complexity for computing lower order moments from $O(N\sqrt{K})$ to $O(K)$ has been an open problem since the lower bound is $\Omega(K)$ for computing lower order moments on the block representation [10] directly where the time complexity notation $\Omega(K)$ denotes the lower bound time complexity $g(N)$ such that there exist two constant terms N_0 and c satisfying $g(N) \geqslant cK$ for $N \geqslant N_0$ [7]. The motivation of this research is to present a new algorithm, which takes $O(K)$ time, for computing lower order moments in order to meet the lower bound complexity.

Suppose the input grey image with size $N \times N$ has been compressed into the block representation where the number of blocks used is K. For each block B_i, instead of computing the local moment of that block B_i in $O(L_i)$ time [12], this correspondence first derives some useful closed forms for computing it in $O(1)$ time. According to these closed forms, we further present a new $O(K)$-time algorithm for computing moments on the whole block representation. The time complexity required in our proposed algorithm meets the lower bound $\Omega(K)$. For computing lower order moments, our proposed algorithm generalizes the previous result by Spiliotis and Mertzios [8] from the binary image domain to the grey image domain. Finally, experimental results reveal a significant computational advantage of our proposed algorithm.

The remainder of this paper is organized as follows. Section 2 introduces the block representation used for representing grey images. Our proposed new optimal algorithm for computing lower order moments on the block representation directly is presented in Section 3. Some related experiments are illustrated in Section 4. Finally, some concluding remarks are addressed in Section 5.

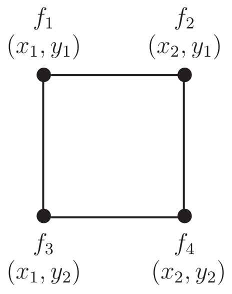

Fig. 1. Homogeneous block *B*.

2. The block representation

In Ref. [10], the original grey image is first partitioned into some homogeneous blocks based on the bintree decomposition principle; then two simple arrays, namely, the linear-tree table and the color table, are used to represent these blocks. During the bintree partition processing, the bilinear interpolation [13,14], is used to control the image quality under the specified error tolerance. Under the same image quality, the block representation presented in Ref. [10] has a better encoding-time performance when compared to the triangular approach [15]. In Section 4, some experimental results are illustrated to show the reasonable compression performance in terms of bits per pixel (BPP) and the quality performance in terms of peak-signal-to-noise ratios (PSNR) of our adopted block representation.

According to the preorder traversal technique [7], the bintree decompression is based on recursively dividing the image into two equal-sized subimages. At each division step, the partition is alternated between the *x*- and *y*-axes. If a subimage is not a homogeneous block, it is subdivided into two equal-sized subimages again until all the blocks are homogeneous.

We now give a quantified definition for the homogeneous block, say block *B*, as shown in Fig. 1 where f_1, f_2, f_3, and f_4 are grey-levels of the four corners. Using the linear interpolation method [13,14], the estimated grey-level of the pixel at (x, y) in the block B is calculated by

$$f_{est}(x, y) = f_5 + \frac{f_6 - f_5}{y_2 - y_1}(y - y_1), \tag{2}$$

where $f_5 = f_1 + ((f_2 - f_1)/(x_2 - x_1))(x - x_1)$ and $f_6 = f_3 + ((f_4 - f_3)/(x_2 - x_1))(x - x_1)$. Given a specified error tolerance ε, by Eq. (2), if the condition $|f(x, y) - f_{est}(x, y)| \leqslant \varepsilon$ holds for all the pixels in the block, $x_1 \leqslant x \leqslant x_2$ and $y_1 \leqslant y \leqslant y_2$, where $f(x, y)$ is the grey-level at (x, y), then we say that the block B is a homogeneous block.

For example, one grey image has been partitioned into seven homogeneous blocks as shown in Fig. 2(a) according to the above partition principle. The corresponding binary tree, i.e. bintree, representation is illustrated in Fig. 2(b). In

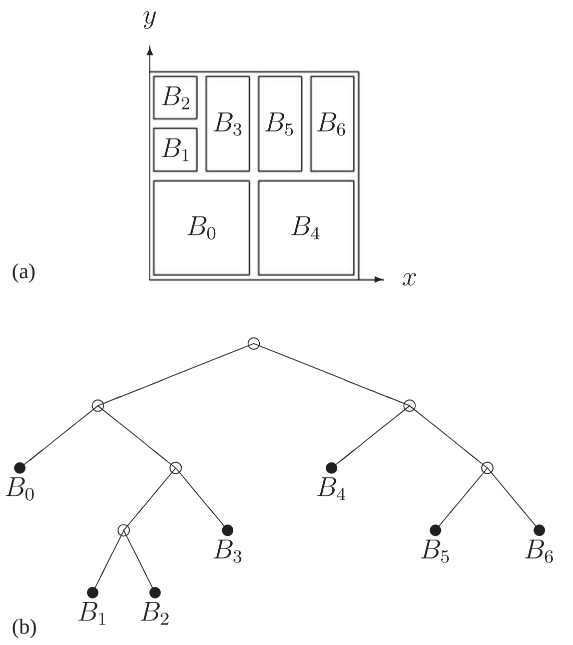

Fig. 2. An example: (a) The partitioned homogeneous blocks. (b) The bintree representation.

Fig. 2(a), the seven partitioned homogeneous blocks are denoted by B_0, B_1, B_2, B_3, B_4, B_5, and B_6, which correspond to the seven leaves in Fig. 2(b).

Traversing the bintree of Fig. 2(b) via preorder traversal manner, at each iteration, we save the bit '0' into the linear-tree table when an internal node is encountered; save the bit '1' into the linear-tree table and save the four grey-levels of the four corners in the related homogeneous block, say (f_1, f_2, f_3, f_4), into the color table when a leaf node is encountered. The contents of the linear-tree table and the color table for Fig. 2 are listed below:

linear-tree table:	0 0 1 0 0 1 1 1 0 1 0 1 1
color table:	$(f_{1,0}, f_{2,0}, f_{3,0}, f_{4,0})$,
	$(f_{1,1}, f_{2,1}, f_{3,1}, f_{4,1})$,
	$(f_{1,2}, f_{2,2}, f_{3,2}, f_{4,2})$,
	$(f_{1,3}, f_{2,3}, f_{3,3}, f_{4,3})$,
	$(f_{1,4}, f_{2,4}, f_{3,4}, f_{4,4})$,
	$(f_{1,5}, f_{2,5}, f_{3,5}, f_{4,5})$,
	$(f_{1,6}, f_{2,6}, f_{3,6}, f_{4,6})$.

The above block representation consists of two arrays. In the color table, each entry contains four grey-levels. The binary string 0010 011101011 in the linear-tree table is used to keep the geometrical relationship among these homogeneous blocks. After scanning the linear-tree table, all the (x, y)-coordinates of the four corners of each homogeneous block can be calculated in $O(K)$ time, where K denotes the number of blocks. For example, upon scanning the three bits '001' in the linear-tree table, the coordinates of the four corners of block B_0 (see Fig. 2) can be derived and the four corresponding grey levels $(f_{1,0}, f_{2,0}, f_{3,0}, f_{4,0})$ can be obtained. Upon obtaining the coordinates and the relevant grey levels of all homogeneous blocks, our proposed block-based algorithm for computing estimated lower order moments is presented in next section.

3. The proposed block-based algorithm for computing lower order moments

Instead of computing local lower order moments of each block B_i in $O(L_i)$ time [12], where L_i denotes the perimeter of the block B_i, this section first presents a new method to compute local lower order moments of each block in $O(1)$ time. Then we present an $O(K)$-time algorithm to compute lower order moments for all the blocks.

For clarity, let the width and height of block B_i be denoted by w_i $(=x_{2,i} - x_{1,i} + 1)$ and h_i $(=y_{2,i} - y_{1,i} + 1)$, respectively. By Eq. (2), the estimated grey-level at position $(x_{1,i} + x, y_{1,i} + y)$, $0 \leqslant x \leqslant w_i - 1$ and $0 \leqslant y \leqslant h_i - 1$, is equal to

$$\begin{aligned} f_{est}(x_{1,i} + x, y_{1,i} + y) &= f_{est}(x_{1,i} + x, y_{1,i}) + y \\ &\quad \times \frac{f_{est}(x_{1,i} + x, y_{2,i}) - f_{est}(x_{1,i} + x, y_{1,i})}{h_i - 1} \\ &= f_{1,i} + x \times \frac{f_{2,i} - f_{1,i}}{w_i - 1} \\ &\quad + \frac{y}{h_i - 1}\left[\left(f_{3,i} + x \times \frac{f_{4,i} - f_{3,i}}{w_i - 1}\right)\right. \\ &\quad \left. - \left(f_{1,i} + x \times \frac{f_{2,i} - f_{1,i}}{w_i - 1}\right)\right] \\ &= f_{1,i} + x \times \frac{f_{2,i} - f_{1,i}}{w_i - 1} + y \times \frac{f_{3,i} - f_{1,i}}{h_i - 1} \\ &\quad + xy \times \frac{f_{1,i} - f_{2,i} - f_{3,i} + f_{4,i}}{(w_i - 1)(h_i - 1)}. \end{aligned} \tag{3}$$

Let $\triangle f_{s_t,i} = (f_{2,i} - f_{1,i})/(w_i - 1)$, $\triangle f_{s_l,i} = (f_{3,i} - f_{1,i})/(h_i - 1)$, and $D_{1,i} = (f_{1,i} - f_{2,i} - f_{3,i} + f_{4,i})/((w_i - 1)(h_i - 1))$. Then Eq. (3) can be written as

$$\begin{aligned} f_{est}(x_{1,i} + x, y_{1,i} + y) \\ = f_{1,i} + x\triangle f_{s_t,i} + y\triangle f_{s_l,i} + xyD_{1,i}. \end{aligned} \tag{4}$$

By Eq. (4), Eq. (1) is represented by

$$\begin{aligned} m_{pq} &= \sum_{i=0}^{K-1} m_{pq,i} \\ &= \sum_{i=0}^{K-1} \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} (x_{1,i} + x)^p (y_{1,i} + y)^q \\ &\quad \times f_{est}(x_{1,i} + x, y_{1,i} + y). \end{aligned} \tag{5}$$

In what follows, we want to present a novel approach for computing $m_{pq,i}$ in $O(1)$ time. Instead of using the slice integration technique [12], according to the binomial expansion technique, the term $m_{pq,i}$ in Eq. (5) can be written as

$$
\begin{aligned}
m_{pq,i} = {} & \sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1}\left[x_{1,i}^p + \binom{p}{1}x_{1,i}^{p-1}x + \cdots + x^p\right] \\
& \times \left[y_{1,i}^q + \binom{q}{1}y_{1,i}^{q-1}y + \cdots + y^q\right] \\
& \times f_{est}(x_{1,i}+x, y_{1,i}+y) \\
= {} & x_{1,i}^p y_{1,i}^q \sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1} f_{est}(x_{1,i}+x, y_{1,i}+y) \\
& + \binom{q}{1} x_{1,i}^p y_{1,i}^{q-1} \sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1} y \\
& \times f_{est}(x_{1,i}+x, y_{1,i}+y) \\
& + \cdots + x_{1,i}^p \sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1} y^q \\
& \times f_{est}(x_{1,i}+x, y_{1,i}+y) \\
& + \binom{p}{1} x_{1,i}^{p-1} y_{1,i}^q \sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1} x \\
& \times f_{est}(x_{1,i}+x, y_{1,i}+y) \\
& + \binom{p}{1}\binom{q}{1} x_{1,i}^{p-1} y_{1,i}^{q-1} \sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1} xy \\
& \times f_{est}(x_{1,i}+x, y_{1,i}+y) \\
& + \cdots + \binom{p}{1} x_{1,i}^{p-1} \sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1} xy^q \\
& \times f_{est}(x_{1,i}+x, y_{1,i}+y) \\
& + \cdots \\
& + \cdots \\
& + y_{1,i}^q \sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1} x^p f_{est}(x_{1,i}+x, y_{1,i}+y) \\
& + \binom{q}{1} y_{1,i}^{q-1} \sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1} x^p y \\
& \times f_{est}(x_{1,i}+x, y_{1,i}+y) \\
& + \cdots + \sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1} x^p y^q \\
& \times f_{est}(x_{1,i}+x, y_{1,i}+y). \qquad (6)
\end{aligned}
$$

Looking at each double summation term at the right side of Eq. (6), the kernel computation involved in Eq. (6) is

$$
\begin{aligned}
S_{pq,i} = {} & \sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1} x^p y^q f_{est}(x_{1,i}+x, y_{1,i}+y) \\
= {} & \sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1} x^p y^q [f_{1,i} + x \times \triangle f_{s_t,i} + y \\
& \times \triangle f_{s_l,i} + xy \times D_{1,i}] \\
= {} & f_{1,i} \sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1} x^p y^q + \triangle f_{s_t,i} \sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1} x^{p+1} y^q \\
& + \triangle f_{s_l,i} \sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1} x^p y^{q+1} \\
& + D_{1,i} \sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1} x^{p+1} y^{q+1}. \qquad (7)
\end{aligned}
$$

In Eq. (7), the summation term $s_{0,w_i-1}^p = \sum_{x=0}^{w_i-1} x^p$ for a specific p, $0 \leqslant p \leqslant 4$, can be computed in $O(1)$ time by using the following five closed forms:

$$
\begin{aligned}
s_{0,w_i-1}^0 &= \sum_{x=0}^{w_i-1} x^0 = w_i, \\
s_{0,w_i-1}^1 &= \sum_{x=0}^{w_i-1} x^1 = \frac{w_i(w_i-1)}{2}, \\
s_{0,w_i-1}^2 &= \sum_{x=0}^{w_i-1} x^2 = \frac{w_i(w_i-1)(2w_i-1)}{6}, \\
s_{0,w_i-1}^3 &= \sum_{x=0}^{w_i-1} x^3 = \frac{w_i^2(w_i-1)^2}{4}, \\
s_{0,w_i-1}^4 &= \sum_{x=0}^{w_i-1} x^4 \\
&= \frac{w_i(w_i-1)(2w_i-1)(3w_i^2-3w_i-1)}{30}. \qquad (8)
\end{aligned}
$$

By the same arguments as in Eq. (8), the summation term $s_{0,h_i-1}^q = \sum_{y=0}^{h_i-1} y^q$, $0 \leqslant q \leqslant 4$, can be computed in $O(1)$ time. Thus, Eq. (7) can be written as

$$
\begin{aligned}
S_{pq,i} = {} & f_{1,i} s_{0,w_i-1}^p s_{0,h_i-1}^q + \triangle f_{s_t,i} s_{0,w_i-1}^{p+1} s_{0,h_i-1}^q \\
& + \triangle f_{s_l,i} s_{0,w_i-1}^p s_{0,h_i-1}^{q+1} \\
& + D_{1,i} s_{0,w_i-1}^{p+1} s_{0,h_i-1}^{q+1}. \qquad (9)
\end{aligned}
$$

Eq. (9) implies that the computation of $S_{pq,i}$ can be done in $O(1)$ time. Further, we have the following result.

Theorem 1. *The estimated lower order moments of the ith block, i.e.* $m_{pq,i}$, $0 \leqslant p+q \leqslant 3$, *can be calculated in* $O(1)$ *time*.

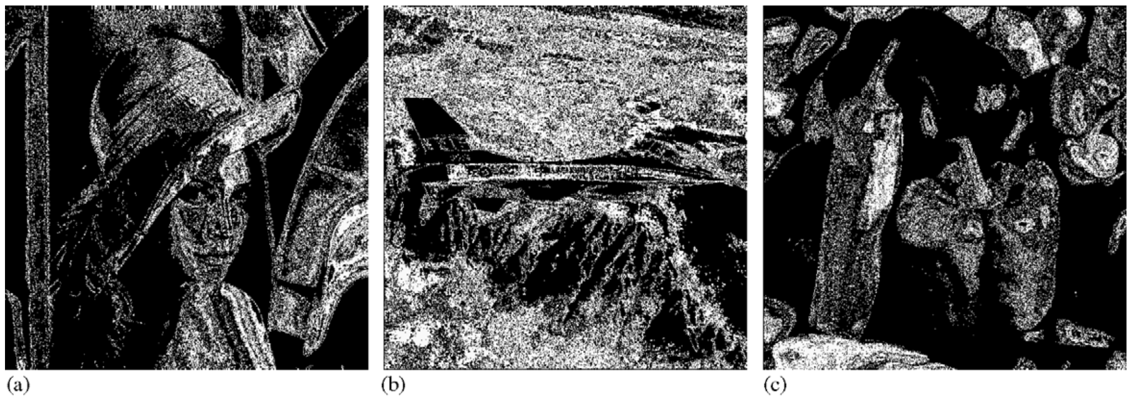

Fig. 3. Three original images: (a) Lena; (b) F16; (c) Pepper.

Proof. (see Appendix A). □

The main result follows from Theorem 1 and Eq. (5).

Theorem 2. *Supposing the given $N \times N$ grey image has been represented by the block representation with K blocks mentioned in Section* 2, *our proposed new algorithm can compute the estimated lower order moments in $O(K)$ time*.

By Theorem 2, note that the error in m_{pq} is bounded by $\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} x^p y^q \varepsilon$, where ε is the specified error tolerance.

4. Experimental results

In this section, first some experimental results are illustrated to show the reasonable compression performance in terms of BPP and the quality performance in terms of PSNR of our block representation. As shown in Fig. 3, three real images, namely the Lena, the F16, and the Pepper, are used in the experiments. Each image is of size 512 × 512 and each pixel requires 8 bits. All experiments are performed on the IBM Pentium III microprocessor with 667 MHz and 128 MB RAM. The operation system is MS-Windows 2000 and the program developing environment is Borland C++ Builder 5.0. Following the same performance evaluation used in Refs. [11,8,12], we ignore the decomposition time in our proposed block-based method since once a grey image has been compressed into the block representation, the block representation can be reused for many block-based image algorithms. There are three possible ways to compute lower order moments, namely, (1) the conventional method which runs on the original $N \times N$ grey image, (2) our proposed method on the compressed image in terms of block representation directly, or (3) the indirect method (first decompressing the compressed image, then computing moments on the decompressed image). Therefore we compare the computational performance among the three methods.

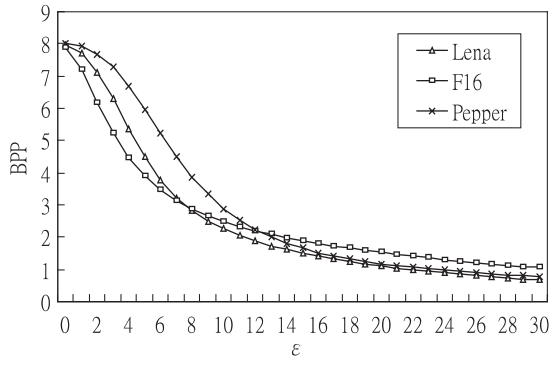

Fig. 4. The BPP performance of the adopted block representation.

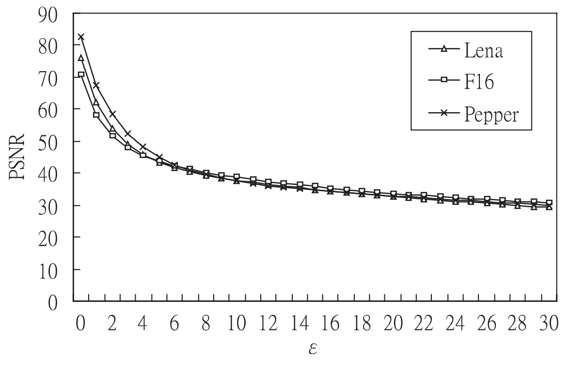

Fig. 5. The PSNR performance of the adopted block representation.

Fig. 4 (Fig. 5) shows the average BPP (PSNR) performance for different ε's. It is observed that when $\varepsilon = 20$, one pixel of the block representation only needs 1.266 bits, while it needs 8 bits to save one pixel in the original image. It is encouraging that under the same ε, i.e. $\varepsilon = 20$, the PSNR of the decompressed image based on our block representation

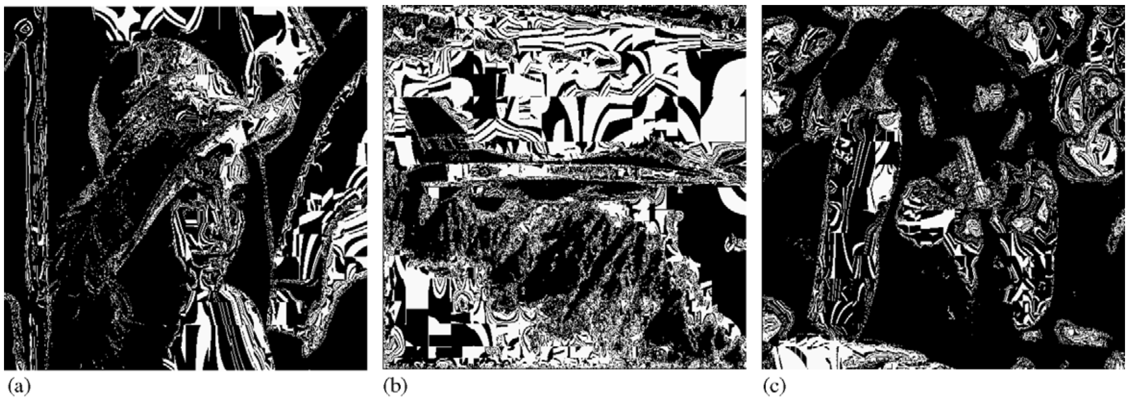
Fig. 6. Three decompressed images: (a) Lena; (b) F16; (c) Pepper.

is still higher than 33.052. Here, the PSNR is defined by

$$\text{PSNR} = 10\log_{10}\frac{255^2}{1/N^2\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}(f(x,y)-f_{est}(x,y))^2}.$$

For the error tolerance $\varepsilon = 20$, Fig. 6 depicts the three decompressed images. Comparing the three original images as shown in Fig. 3 with the three decompressed images, both relevant images, e.g. the original Lena image and the decompressed Lena image, are quite similar from our visual inspection.

After demonstrating the BPP and PSNR performance of our block representation, we now evaluate the execution time performance among the three methods. Fig. 7(a)–(c) illustrate the execution time in terms of seconds required in our proposed method, the indirect method, and the conventional method, respectively. From Fig. 7, it is observed that our proposed optimal algorithm for computing lower order moments is faster than the indirect method and the conventional method. When $\varepsilon = 20$, the average executing time improvement ratio of our proposed method over the indirect method is 80% = (Time(Indirect Method) − Time(Proposed Method))/Time(Indirect Method)=(0.087−0.017)/0.087 and the average executing time improvement ratio of our proposed method over the conventional method is 76% = (0.07 − 0.017)/0.07.

Table 1 illustrates the accuracy of calculated values of the relevant moments when $\varepsilon = 20$. In Table 1, the moments calculated by using the conventional method are the exact moments and the moments calculated by using our proposed method are the estimated moments. It is observed that the estimated moments calculated by our proposed method are encouraging when compared to the conventional method. For example, the calculated value of m_{10} is 8.60×10^9 using the proposed method on the compressed image and the calculated value of m_{10} is 8.56×10^9 using the conventional method on Lena.

5. Conclusions

The computation of lower order moments for grey images is very important in many applications in the image processing field. Under the block representation, we have presented a new algorithm for computing lower order moments. The time complexity of our proposed algorithm meets the lower bound. Experimental results reveal that the average execution time improvement ratio of our proposed method over the indirect method is 80% and the average execution time improvement ratio of our proposed method over the conventional method is 76%.

For lower order moments only, the closed forms in Eq. (8) are quite elegant. However, for arbitrary p, Eq. (8) can be extended to arbitrary order by employing the following recursive summation formulas used in Ref. [8]:

$$s^p_{0,w_i-1} = \sum_{x=0}^{w_i-1} x^p = \frac{w_i^{p+1} - w_i - \binom{p+1}{1}s^1_{0,w_i-1} - \binom{p+1}{2}s^2_{0,w_i-1} - \cdots - \binom{p+1}{p-1}s^{p-1}_{0,w_i-1}}{p+1},$$

where $\binom{i}{j} = \frac{i!}{j!(i-j)!}$. Recursive summation formulas similar to the above, s^q_{0,h_i-1} also can be extended to arbitrary q.

Consequently, the computation of $m_{pq,i}$ as shown in Eq. (7) can be rewritten as

$$\begin{aligned}m_{pq,i} = &\sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1}(x_{1,i}+x)^p(y_{1,i}+y)^q\\ &\times f_{est}(x_{1,i}+x, y_{1,i}+y)\\ = &\, x_{1,i}^p y_{1,i}^q S_{00,i} + \binom{q}{1} x_{1,i}^p y_{1,i}^{q-1} S_{01,i}\\ &+\cdots + x_{1,i}^p S_{0q,i}\\ &+\binom{p}{1} x_{1,i}^{p-1} y_{1,i}^q S_{10,i}\\ &+\binom{p}{1}\binom{q}{1} x_{1,i}^{p-1} y_{1,i}^{q-1} S_{11,i}\\ &+\cdots+\binom{p}{1} x_{1,i}^{p-1} S_{1q,i}\\ &+\cdots\\ &+\cdots\\ &+ y_{1,i}^q S_{p0,i} + \binom{q}{1} y_{1,i}^{q-1} S_{p1,i} + \binom{q}{2} y_{1,i}^{q-2} S_{p2,i}\\ &+\cdots + S_{pq,i}.\end{aligned}$$

Based on the above identity, it is not hard to verify that it takes $O(r^2)$ time to compute all moments up to certain order r for $p, q \leqslant r$ since there are $O(pq)$ terms to be calculated in the above identity.

Besides the moments used in this paper (see Eq. (1)), there are other different kinds of moments, such as Zernike moments [16] and geometric moments [17]. The applications of the results of this paper to the computation of these different moments on the adopted block representation for grey images is an interesting research issue. In addition, it is also of interest to apply the results of this paper and the window query approach [18] to compute the moments of grey images in small windows for certain applications.

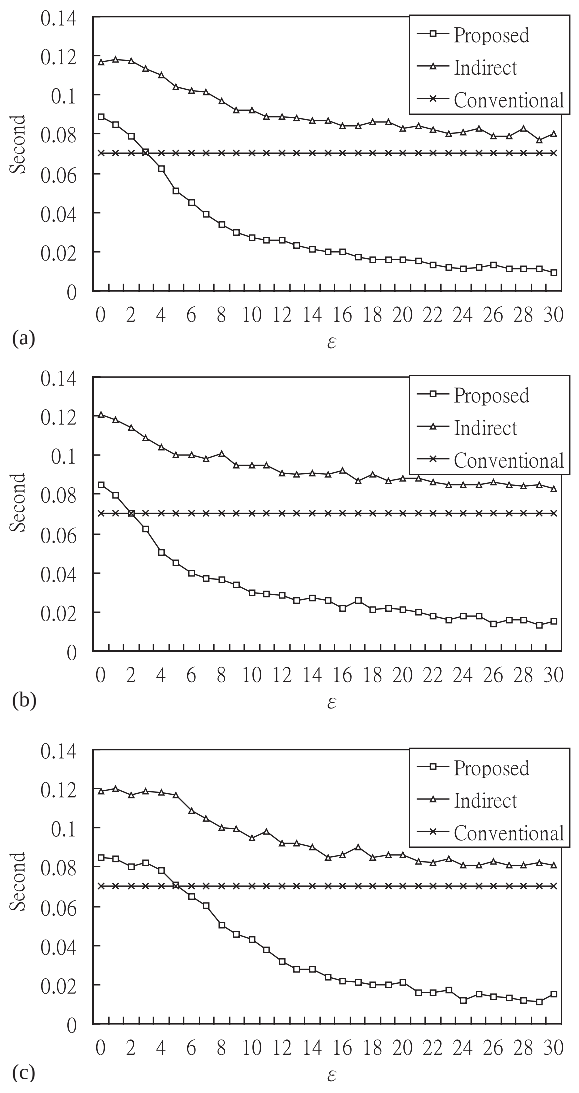

Fig. 7. The execution time performance among our proposed method, the indirect method, and the conventional method: (a) execution time performance for Lena; (b) execution time performance for F16; (c) execution time performance for Pepper.

Appendix A. The proof of Theorem 1

We only prove the validity for $m_{11,i}$, $m_{20,i}$, and $m_{03,i}$ since the proof technique is also valid for the remaining cases. From Eqs. (8) and (9), we have

$$\begin{aligned}m_{11,i} = &\sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1}(x_{1,i}+x)^1(y_{1,i}+y)^1\\ &\times f_{est}(x_{1,i}+x, y_{1,i}+y)\\ = &\sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1}(x_{1,i}y_{1,i}+x_{1,i}y+y_{1,i}x+xy)\\ &\times f_{est}(x_{1,i}+x, y_{1,i}+y)\\ = &\, x_{1,i}y_{1,i}S_{00,i}+x_{1,i}S_{01,i}+y_{1,i}S_{10,i}+S_{11,i},\end{aligned}$$

$$\begin{aligned}m_{20,i} = &\sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1}(x_{1,i}+x)^2(y_{1,i}+y)^0\\ &\times f_{est}(x_{1,i}+x, y_{1,i}+y)\\ = &\sum_{x=0}^{w_i-1}\sum_{y=0}^{h_i-1}(x_{1,i}^2+2x_{1,i}x+x^2)\\ &\times f_{est}(x_{1,i}+x, y_{1,i}+y)\\ = &\, x_{1,i}^2 S_{00,i}+2x_{1,i}S_{10,i}+S_{20,i},\end{aligned}$$

Table 1
Accuracy comparison

	Conventional method			Proposed method		
	Lena	F16	Pepper	Lena	F16	Pepper
m_{00}	3.22E07	4.68E07	3.14E07	3.21E07	4.67E07	3.13E07
m_{10}	8.60E09	1.22E10	8.05E09	8.56E09	1.22E10	8.03E09
m_{01}	7.98E09	1.18E10	7.76E09	7.95E09	1.17E10	7.73E09
m_{11}	2.17E12	3.05E12	1.93E12	2.16E12	3.04E12	1.92E12
m_{20}	2.99E12	4.24E12	2.75E12	2.97E12	4.23E12	2.74E12
m_{02}	2.67E12	4.02E12	2.61E12	2.67E12	4.01E12	2.60E12
m_{21}	7.65E14	1.06E15	6.44E14	7.62E14	1.06E15	6.43E14
m_{12}	7.32E14	1.03E15	6.32E14	7.29E14	1.02E15	6.30E14
m_{30}	1.15E15	1.65E15	1.06E15	1.15E15	1.56E15	1.05E15
m_{03}	1.02E15	1.55E15	9.97E14	1.01E15	1.55E15	9.93E14

$$\begin{aligned} m_{03,i} &= \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} (x_{1,i}+x)^0 (y_{1,i}+y)^3 \\ &\quad \times f_{est}(x_{1,i}+x, y_{1,i}+y) \\ &= \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} (y_{1,i}^3 + 3y_{1,i}^2 y + 3y_{1,i} y^2 + y^3) \\ &\quad \times f_{est}(x_{1,i}+x, y_{1,i}+y) \\ &= y_{1,i}^3 S_{00,i} + 3y_{1,i}^2 S_{01,i} + 3y_{1,i} S_{02,i} + S_{03,i}. \qquad (10) \end{aligned}$$

Since each $S_{pq,i}$ (see Eq. (9)) can be done in $O(1)$ time, it follows that each lower order moment $m_{pq,i}$ can also be computed in $O(1)$ time. This completes the proof.

References

[1] R.C. Gonzalez, R.E. Woods, Digital Image Processing, second ed., Prentice-Hall, New York, 2002.
[2] M. Sonka, V. Hlavac, R. Boyle, Image Processing, Analysis, and Machine Vision, second ed., PWS, New York, 1998.
[3] S.C. Pei, L.G. Liou, Using moments to acquire the motion parameters of a deformable object without correspondence, Image Vision Comput. 12 (1994) 475–485.
[4] S.C. Pei, J.H. Horng, A moment-based approach for deskewing rotationally symmetric shapes, IEEE Trans. Image Process. 8 (1999) 1831–1834.
[5] W.H. Tsai, Moment-preserving thresholding: a new approach, Computer Vision, Graphics, and Image Processing 29 (1985) 377–393.
[6] M.K. Hu, Visual pattern recognition by moment invariants, IRE Trans. Information Theory 8 (1962) 179–187.
[7] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, second ed., McGraw-Hill, New York, 2001.
[8] I.M. Spiliotis, B.G. Mertzios, Real-time computation of two-dimensional moments on binary images using image block representation, IEEE Trans. on Image Processing 7 (1998) 1609–1615.
[9] J. Flusser, Refined moment calculation using image block representation, IEEE Trans. on Image Processing 9 (2000) 1977–1978.
[10] K.L. Chung, J.G. Wu, Improved image compression using s-tree and shading approach, IEEE Trans. Commun. 48 (2000) 748–751.
[11] H. Samet, The Design and Analysis of Spatial Data Structures, Addison-Wesley, New York, 1990.
[12] K.L. Chung, W.M. Yan, Z.H. Liao, Fast computation of moments on compressed grey images using block representation, Real-Time Imaging 8 (2002) 137–144.
[13] G. Farin, Curves and Surfaces for Computer Aided Geometric Design, Chapter 16, second ed., Academic Press, New York, 1990.
[14] J.D. Foley, A.V. Dam, S.K. Feiner, J.F. Hughes, Computer Graphics, Principle, and Practice, second ed., Addision-Wesley, Reading, MA, 1990.
[15] R. Distasi, M. Nappi, S. Vitulano, Image compression by S-tree triangular coding, IEEE Trans. Commun. 45 (1997) 1095–1100.
[16] L. Wang, G. Healey, Using Zernike moments for the illumination and geometry invariant classification of multispectral texture, IEEE Trans. on Image Processing 7 (1998) 196–203.
[17] J. Martinez, F. Thomas, Efficient computation of local geometric moments, IEEE Trans. on Image Processing 11 (2002) 1102–1111.
[18] K.L. Chung, Y.H. Tsai, F.C. Hu, Space-filling approach for fast window query on compressed images, IEEE Trans. on Image Processing 9 (2000) 2109–2116.

About the Author—KUO-LIANG CHUNG received the B.S., M.S., and Ph.D. degrees in computer science and information engineering from National Taiwan University in 1982, 1984, and 1990, respectively. From 1984 to 1986, he completed the military service. From 1986 to 1987, he was a research assistant in the Institute of Information Science, Academic Sinica. Since 1995, he has been a Professor in the

Department of Computer Science and Information Engineering at National Taiwan University of Science and Technology. He is now the chair of the same department. He has been an IEEE senior member since November 2001. Prof. Chung received the Distinguished Professor Award from the Chinese Institute of Engineers in May 2001. He received the distinguished research award (2004–2007) from the National Science Council, Taiwan. His research interests include image compression, image processing, video compression, coding theory, and algorithms.

About the Author—PING-CHIN CHEN received the B.S. and M.S. degrees in information management from National Taiwan University of Science and Technology in 2001 and 2003, respectively. Her research interests include image compression, image processing, and algorithms.