Proceedings of the Fifth IASTED International Conference
Signal Processing, Pattern Recognition, and Applications
February 13–15, 2008, Innsbruck, Austria
ISBN Hardcopy: 978-0-88986-717-8 / CD: 978-0-88986-718-5

FAST AND REAL-TIME MOMENT COMPUTATION METHODS OF GRAY IMAGES USING IMAGE BLOCK REPRESENTATION

Iraklis M. Spiliotis and Yiannis S. Boutalis
Department of Electrical and Computer Engineering
Democritus University of Thrace
GR-67100, Xanthi, Greece
e-mails: spiliot@ee.duth.gr, ybout@ee.duth.gr

ABSTRACT
Moments constitute a well known tool in the field of image analysis and pattern recognition but they suffer from the drawback of high computational cost. Efforts for the reduction of the required computational complexity have been reported, but are mainly focused on binary images. Image Block Representation (IBR) has been introduced some years ago and is applicable only on binary images. The block represented binary image is well suited for fast implementation of various processing and analysis algorithms and a real-time algorithm for the computation of moments has been presented in the past. In this paper an application of the image block representation on gray images and two fast algorithms for the computation of moments are presented. The first algorithm provides the exact moment values and is fast implemented. The second algorithm derives from the first, computes approximated moment values with an error of 2-3% from the exact values and operates in real-time.

KEY WORDS
Image analysis, Moments, Image Block Representation

1. Introduction

Moments and functions of moments have been successfully and extensively used in pattern recognition, image classification and scene analysis [1]. In pattern recognition applications a small set of lower order moments are able to discriminate among different patterns [2]. Since the introduction of moment invariants by Hu [3], which are invariant to translation, rotation and scale change the geometric moments and their derivatives (i.e. central, normalized central and invariant moments) have been extensively used in these applications.
The geometric moment of order *(p,q)* of a 2-D function *g(u,v)* is defined as:

$$m_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(u,v)\, u^p v^q \, du dv \qquad (1)$$

Given a $N_1 \times N_2$ discrete image $g(x,y)$, the moment of order (p,q) is defined as:

$$m_{pq} = \sum_{x=0}^{N_1-1} \sum_{y=0}^{N_2-1} x^p y^q \; g(x,y), \; p,q \in \mathbf{Z} \qquad (2)$$

The computation of moments up to the order $(L-1, L-1)$, directly from (2) requires $2N^2L^2$ power calculations, $2N^2L^2$ multiplications and N^2L^2 additions. Several researchers have proposed approaches for the reduction of the required computational complexity. A group of methods is based on Green's theorem, which evaluates the double integral over the object by means of single integration along the object boundary [2]. Zakaria et al. [4], Dai et al. [5], Li [6], and Flusser [7] proposed various approaches based on the decomposition of the object into rows or row segments. Spiliotis and Mertzios [8] proposed a method for the real-time moment computation of binary images representing the image by non-overlapping homogenuous orthogonal blocks and performing moment calculations using the coordinates of the upper left and lower right corner of each orthogonal. Chung and Chen [9], extended the idea of [8] to gray images by separating gray images to "homogeneous" blocks of image intensities and performing moment calculation using the blocks. The definition of homogeneity introduces an error in the moment calculation requiring a compromise between the accuracy and the number of homogeneous blocks making the error smaller when a large number of homogeneous blocks are used.
In this paper the gray level image is decomposed in a number of binary images without any loss of information. The IBR scheme is applied to the binary images that constitute the gray-level image. In the sequel the algorithm of Spiliotis and Mertzios [8] is applied for the fast and exact computation of the moments of each binary image. The moments of the gray images are derived easily from the moments of the binary images.
A second algorithm omits some of the lower order binary images from the computation of the moments, with significant savings to the computational time. However the number of the omitted binary images is small, in order to keep the approximation error at accepted levels.
Moreover the substitution of the omitted binary images with chessboard images, reduces drastically the

approximation error. This method permits the computation of the moments with only 3 higher order binary images and keeps the approximation error as low as 2-3%, while the proposed algorithm operates in real-time.

The remainder of the paper is organized as follows: Section 2 gives the basic definitions for the block representation of binary images and presents the mathematical formulae for the real-time computation of moments on binary images. Section 3 presents the decomposition of a gray image to a set of binary images. In Section 4 the computation of the moments is given, while section 5 provides experimental results and comparisons. Finally, section 6 provides some concluding remarks.

2. IBR and moments computation on binary images

Suppose that in a binary image the object pixels are assigned to level 1. Suppose also that the object pixels are represented by a set of non-overlapping rectangles with edges parallel to the axes containing integer number of pixels, in such way that every object pixel belongs to only one block. These rectangles are called blocks. It is always feasible to represent a binary image with a set of all the nonoverlapping blocks with object level and this representation is called *Image Block Representation* (IBR) [10].

The IBR concept leads to a simple and fast algorithm, which requires just one pass of the image and simple bookkeeping process. As a result of the application of the above algorithm, we obtain a set of all the rectangular areas with level 1 that form the object. A block represented image is denoted as $f(x,y)=\{b_i : i=0,1,...,k-1\}$, where k is the number of the blocks. Each block is described by four integers, the coordinates of the upper left and down right corner in vertical and horizontal axes, $b_i=(x_{1,b_i}, x_{2,b_i}, y_{1,b_i}, y_{2,b_i})$.

The block extraction process is implemented easily with low computational complexity, since it is a pixel checking process without numerical operations. The block extraction process requires a pass from each line y of the image. In this pass all object level intervals are extracted and compared with the previous extracted blocks. In the following, an IBR algorithm is given.

Algorithm 1: *Image Block Representation.*

Step 1: Consider each line y of the image f and find the object level intervals in line y.

Step 2: Compare intervals and blocks that have pixels in line y-1.

Step 3: If an interval does not match with any block, this is the beginning of a new block.

Step 4: If a block matches with an interval, the end of the block is in the line y.

Given a specific binary image, different sets of different blocks can be formed. Actually, the nonunique block representation does not have any implications on the implementation of any operation on a block represented image. The optimum representation is characterized by the minimum possible number of blocks.

Since the background level is 0, only the pixels with level 1 are taken into account for the computation of the moments. Thus, the 2-D geometrical moments of order *(p,q)* of the image *f(x,y)* are defined by the relation:

$$m_{pq}=\sum_x \sum_y x^p y^q \qquad \forall x,y:\ f(x,y)=1 \tag{3}$$

Since the image is represented by blocks, all the object pixels belong to the k blocks and taking into account the rectangular form of the block, (3) may be rewritten [8] as:

$$m_{pq}=\sum_{i=0}^{k-1} m_{pq}^{b_i}=\sum_{i=0}^{k-1}\sum_{x=x_{1,b_i}}^{x_{2,b_i}}\sum_{y=y_{1,b_i}}^{y_{2,b_i}} x^p y^q$$
$$=\sum_{i=0}^{k-1}\left(\sum_{x=x_{1b}}^{x_{2b}} x^p\right)\left(\sum_{y=y_{1b}}^{y_{2b}} y^q\right)=\sum_{i=0}^{k-1} S^p_{x_{1b_i},x_{2b_i}} S^p_{y_{1b_i},y_{2b_i}} \tag{4}$$

where

$$S^1_{1,n}=\frac{n(n+1)}{2},\ S^2_{1,n}=\frac{n(n+1)(2n+1)}{6},\ S^3_{1,n}=\frac{n^2(n+1)^2}{4},$$
$$S^4_{1,n}=\frac{n(n+1)(2n+1)(3n^2+3n+1)}{30} \tag{5}$$
$$\sum_{j=1}^{m}\binom{m+1}{j} S^j_{1,n}=(n+1)^{m+1}-(n+1)$$

Note that using (4) the computational complexity is reduced to $O(N)$ instead of $O(N^2)$ using (2). Moreover, the summations of x^p and y^q may be computed using the above known formulae (5) resulting to further speed-up of the calculations.

Spiliotis and Mertzios at [8], have show that their method has a computational complexity of *O(k)*, where k is the number of the blocks and the method operates in real-time. They also have show that other sets of statistical methods as central moments, normalized central methods and Hu's moment invariants are computed in real-time since are based on geometric moments.

As shown by Liao and Pawlak in [11], eq. (2) is not a very exact approximation of (1), due to the zero-order approximation of the double integral of (1) with the summations of (2) and due to the numeric integration of $x^p y^q$ over each pixel and proposed a correction for the moments computation at the discrete domain.

In accordance, Flusser in [12] proposed a correction to [8], that concludes that:

$$S^p_{x_1,x_2}=\frac{(x_{2,b_i}+0.5)^{p+1}-(x_{1,b_i}-0.5)^{p+1}}{(p+1)} \tag{6}$$

where also proposed that parts of (6) can be precalculated, something which may also be done to quantities of eq. (5).

3. Decomposition of the gray image

Consider the gray image $g(x,y)$, with 2^n gray levels and dimensions $N_1 \times N_2$. The gray image can be decomposed in a set of *n* binary images, so that each binary image is a bit plane of the original gray image. This way, the first binary image b_{n-1} contains the most significant bits (MSB*)* of the binary representation of the image intensities *g(x,y),* the second binary image b_{n-2} contains the second MSB and so on until the binary image b_0, containing the least significant bits of the binary representation of the gray image intensities. The relation between the gray image *g(x,y)* and the *n* binary images b_{n-1}, .., b_1, b_0 is given by:

$$g(x,y) = 2^{n-1} b_{n-1}(x,y) + \ldots + 2^0 b_0(x,y) = \sum_{i=1}^{n-1} 2^i b_i(x,y) \quad (7)$$

Fig. 1, illustrates an image with 256 gray levels and the 8 corresponding binary images that result from the decomposition of the original image. It can be observed that lower order binary images are quite noisy. This feature is exploited below for the reduction of the computational cost of moment calculations.

4. Computation of moments

4.1 Method 1. The exact - fast method

Applying eq. (7) to eq. (2) for the computation of moments, one obtains the following formula:

$$m_{pq} = \sum_{x=0}^{N_1-1} \sum_{y=0}^{N_2-1} x^p y^q \, g(x,y) = \sum_{x=0}^{N_1-1} \sum_{y=0}^{N_2-1} x^p y^q \Big[\sum_{i=1}^{n-1} 2^i b_i(x,y) \Big] = \sum_{i=0}^{n-1} 2^i mb_{ipq} \quad (8)$$

where $mb_{i,pq}$ is the (p,q)-th order moment of the binary image b_i that derives from the decomposition of the gray image. The binary images are represented by blocks and their moments are fast computed using eq. (4) and Flusser's exact formula [12] of eq. (6) with pre-calculations.

4.2 Method 2. The approximate - real-time method

As eq.(8) implies, the moments of lower order bitplanes don't participate equally to the computation of the gray image moments, because of the weight factors of 2^i. If a number of these bitplanes are omitted from the calculation of (8) this results to a calculation error and computational savings.

As already mentioned and illustrated in Fig. (1) the lower order bitplanes look like noise with transitions from white to black. Condider a $N_1 \times N_2$ binary chessboard image $c(x,y)$, where each pixel is different from its 4-neighbors. The substitution of the *m* omitted binary images with the chessboard image, results to:

$$m'_{pq} = \sum_{i=m}^{n-1} 2^i mb_{ipq} + \sum_{i=0}^{m-1} 2^i mc_{pq} = \sum_{i=m}^{n-1} 2^i mb_{ipq} + mc_{pq} \sum_{i=0}^{m-1} 2^i \quad (9)$$

where mc_{pq} are the moments of the chessboard image.

The percentage error of the approximation, between the exact and the approximated moment values, is expressed as:

$$\varepsilon_{pq} = \left| \frac{m_{pq} - m'_{pq}}{m_{pq}} \right| \times 100 \quad (10)$$

The approximation error in the case where some bitplanes are omitted is much bigger in comparison with the case of substitution of the omitted bitplanes with the chessboard image. This is due to the fact that the contribution of the lower order bitplanes to the moments of the gray image is quite similar to the contribution of the chessboard planes to the moments of the gray image.

The percentage contribution Cb_i of the bitplane of the *i*-th plane and the percentage contribution of the chessboard image at the *i*-th plane are calculated according to:

$$Cb_i = \frac{2^i mb_{ipq}}{m_{pq}} \times 100 \quad (11.1)$$

$$Cc_i = \frac{2^i mc_{pq}}{m_{pq}} \times 100 \quad (11.2)$$

As implied from eq. (11.2) $Cc_i = 2Cc_{i-1}$.

In pattern recognition applications a small set of low order moments is adequate to discriminate among different patterns. Hu's moment invariants that have been used extensively for such tasks, are low order moments. The approximate method is particularly suitable for applications using low order moments because the approximation error tends to increase for high moment order.

4.3 Other moment sets

The two proposed methods are also suitable for the computation of other sets of moments. Specifically, the central moments, the normalized central moments and the Hu's moment invariants can also be computed using the fast and exact method 1, or the approximate and real-time method 2.

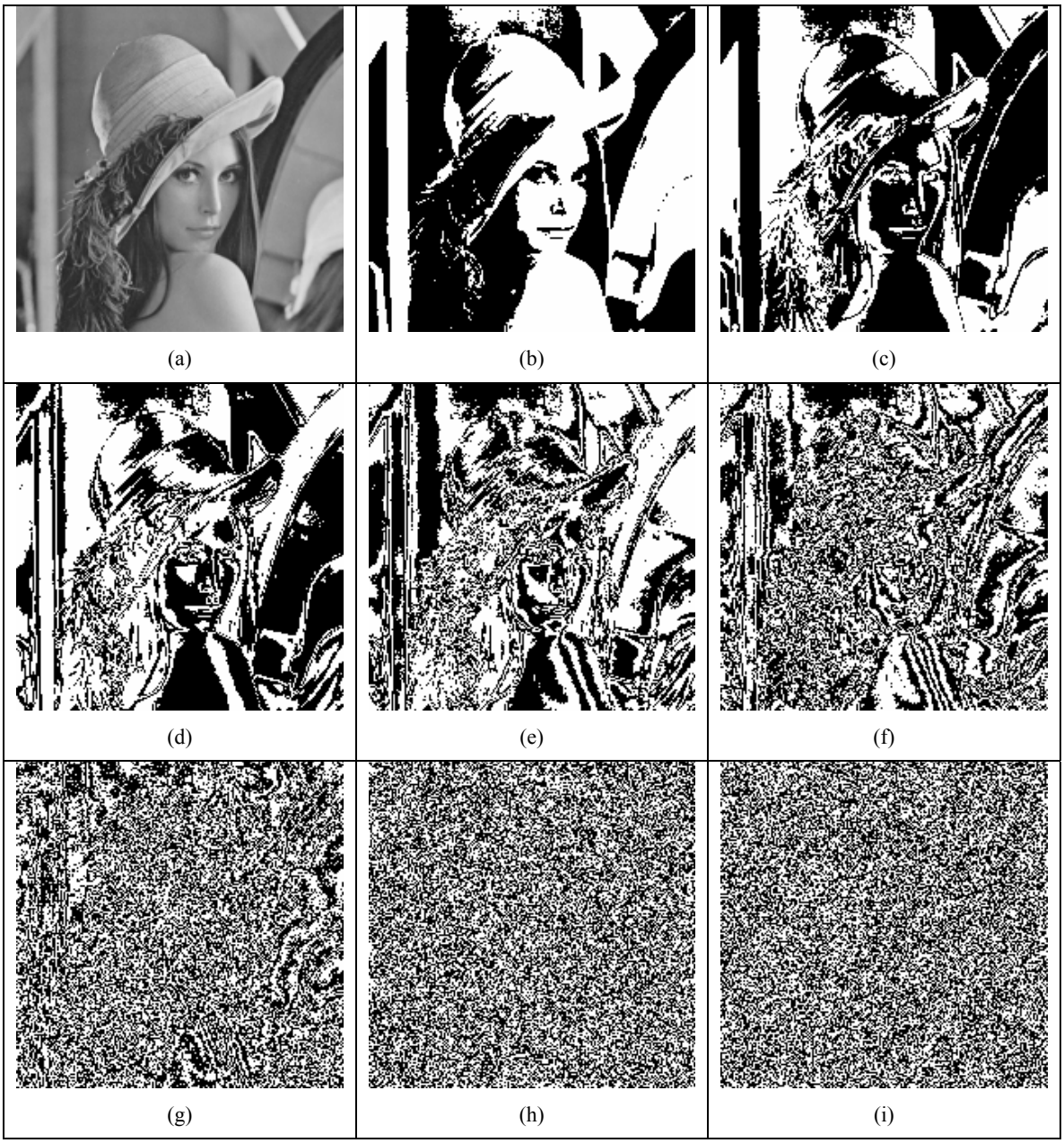

Figure 1 (a). The original gray image with 200x200 pixels and 256 gray levels. (b) – (i). the decomposition of the original image results to these eight binary images b_7 to b_0, where b_7 at (b) derives from the most significant bits and b_0 at (i) derives from the least significant bits, of the pixel values of (a)

5. Experimental results

Lets first consider the exact computation of moments with method 1 of the previous section. For the image of Fig. 1 the required time for the computation of moments up to the order (14,14), with proposed method 1, is 0,672sec, while the computation of these moments using the pixel values and eq. (2) requires 4,511sec, therefore the speedup value is 6,71. Another 0,051 sec are required for the IBR process of the 8 binary images.

The percentage contributions of the i-th binary image and the percentage contribution of the chessboard image at the i-th plane, Cc_i, Cb_i respectively, for i=0,1,..,5, to the moments of the gray image of Fig. 1, are computed using eq. (11) and their values are given at Table 1. As shown in Table 1, the contributions Cc_i, Cb_i for the first 5 lower order planes, are quite close. For the sixth plane i=5, the differences of the contributions become larger. The same holds for planes 6 and 7 which are not given in Table 1. Fig. (2), illustrates the same values in a graphical way.

The percentage approximation moment computation error for m=4,5,6 is given at Tables 2,3,4 respectively, where m is the number of chessboard images that used for the computation of moments for the image of Fig. 1. As indicated from Tables 2,3,4 the percentage error is quite low for m=4, for m=5 is still at accepted levels and for m=6 is unacceptable. Note also that the error becomes larger as the moment order increases.

Table 1 The percentage contributions of the binary images and of the chessboard image at different plane positions to the moments of the gray image, of Fig. 1

	Cb_0	Cc_0	Cb_1	Cc_1	Cb_2	Cc_2	Cb_3	Cc_3	Cb_4	Cc_4	Cb_5	Cc_5
m_{00}	0,4035	0,4044	0,8076	0,8087	1,6026	1,6175	3,2155	3,2349	6,8956	6,4699	12,7042	12,9397
m_{01}	0,3858	0,3886	0,7771	0,7771	1,5304	1,5543	3,1364	3,1086	7,1211	6,2172	10,5664	12,4344
m_{02}	0,3778	0,3817	0,7620	0,7633	1,4990	1,5267	3,1137	3,0534	7,2752	6,1068	9,5462	12,2135
m_{03}	0,3752	0,3795	0,7561	0,7591	1,4879	1,5181	3,1148	3,0362	7,4343	6,0725	9,0647	12,1449
m_{10}	0,4163	0,4173	0,8351	0,8347	1,6566	1,6694	3,3012	3,3387	7,0232	6,6775	12,7210	13,3550
m_{11}	0,3901	0,3933	0,7900	0,7866	1,5515	1,5731	3,1350	3,1463	6,8390	6,2925	10,0361	12,5851
m_{12}	0,3766	0,3810	0,7650	0,7621	1,5000	1,5241	3,0439	3,0483	6,7275	6,0965	8,7415	12,1931
m_{13}	0,3705	0,3755	0,7532	0,7510	1,4775	1,5019	2,9974	3,0039	6,7185	6,0077	8,0878	12,0154
m_{20}	0,4237	0,4244	0,8502	0,8488	1,6794	1,6976	3,3621	3,3952	7,1036	6,7904	12,6997	13,5807
m_{21}	0,3945	0,3976	0,8004	0,7952	1,5646	1,5904	3,1674	3,1807	6,7039	6,3615	9,9214	12,7230
m_{22}	0,3794	0,3841	0,7729	0,7683	1,5107	1,5365	3,0589	3,0731	6,4954	6,1462	8,6736	12,2923
m_{23}	0,3729	0,3785	0,7609	0,7571	1,4904	1,5141	3,0040	3,0282	6,4439	6,0565	8,1125	12,1130
m_{30}	0,4279	0,4284	0,8588	0,8567	1,6902	1,7134	3,4027	3,4269	7,1444	6,8537	12,7319	13,7075
m_{31}	0,3980	0,4011	0,8084	0,8021	1,5734	1,6043	3,2054	3,2085	6,6277	6,4171	9,9650	12,8341
m_{32}	0,3831	0,3881	0,7818	0,7762	1,5231	1,5524	3,0987	3,1048	6,3884	6,2096	8,8226	12,4193
m_{33}	0,3775	0,3837	0,7720	0,7673	1,5090	1,5347	3,0499	3,0694	6,3404	6,1388	8,3913	12,2776

A number of real images have been used in experiments, in order to determine the optimum number of binary images that have to be used in moment computation for keeping the approximation error low.

Specifically the maximum error for m=4 is 0,57% for moment m_{33}, for the image of Fig. 3(a). The maximum error for m=5 is 2,32% for moment m_{33}, for the same image of Fig. 3(a). And the maximum error for m=6 is 4,14% for moment m_{00}, for the image of Fig. 3(b).

Therefore, from the computation of the percentage errors for a number of test images and for accepted moment approximation error less than 3% the optimum number of chessboard images is determined to be 5.

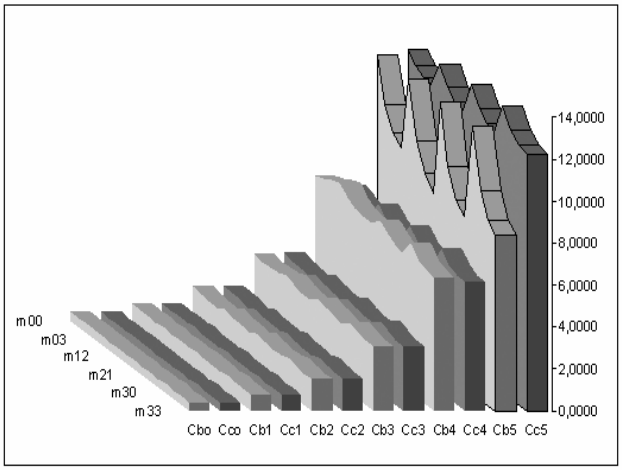

Figure 2 The percentage contributions of the binary images and of the chessboard image at different plane positions to the moments of the gray image

Table 2 The percentage approximation moment computation error for m=4 for the image of Fig. 1

ε_{pq}	**p=0**	**p=1**	**p=2**	**p=3**
q=0	0,0363	0,0510	0,0506	0,0458
q=1	0,0011	0,0326	0,0371	0,0308
q=2	0,0273	0,0299	0,0401	0,0348
q=3	0,0411	0,0337	0,0497	0,0467

Table 3 The percentage approximation moment computation error for m=5 for the image of Fig. 1

ε_{pq}	**p=0**	**p=1**	**p=2**	**p=3**
q=0	0,3894	0,2948	0,2626	0,2448
q=1	0,9049	0,5139	0,3054	0,1798
q=2	1,1957	0,6011	0,3092	0,1439
q=3	1,4030	0,6771	0,3377	0,1550

Table 4 The percentage approximation moment computation error for m=6 for the image of Fig. 1

ε_{pq}	**p=0**	**p=1**	**p=2**	**p=3**
q=0	0,1539	0,3392	0,6185	0,7307
q=1	0,9630	2,0351	2,4962	2,6893
q=2	1,4716	2,8505	3,3095	3,4528
q=3	1,6772	3,2505	3,6628	3,7313

(a) (b)

Figure 3 Sample of test images. (a). Horses with 256x256 pixels. (b). F16 with 200x200 pixels

In order to have significant computational savings, the moments of the chessboard image are precalculated and stored and their use requires only one addition per moment order as indicated by eq. (9). It is worth noting, that at lower bitplanes the number of the blocks is much bigger, therefore the savings in IBR time from the omission of a lower bitplane is greater than the savings from the omission of a higher bitplane.

The required times for the moments computation up to the order (3,3) of the 3 images are presented in Table 5. Column BMC refers to the time taken by the proposed approximated method 2, while column IMC refers to the time taken by using image pixels and eq. (2). Note that the whole time with IBR is less than 0.033sec, therefore the proposed method operates in real-time with more than 30 images/sec.

Table 5 Required times(sec) for the computation of approximated moments up to the order (3,3) with m=5

Image	IBR	BMC	IMC
Lena	0,007	0,007	0,293
Horses	0,012	0,011	0,481
F16	0,006	0,006	0,315

5.1 Comparisons

Chung's method [9], which mentioned at the introduction, uses an error parameter ε for controlling the process of the formation of the blocks. When $\varepsilon = 0$, then the method is lossless and the exact reconstruction of the gray image from its blocks is possible. The comparison of the two proposed methods with Chung's method is for $\varepsilon = 0$. For all the test images, the 2 proposed methods outperform Chung's method. For the image of Fig.1 and the computation of moments up to the order (3,3) the required times are: 0,007 sec for the proposed method 2 with the use of 5 chessboard images, 0,049 sec for the exact proposed method 1, 0,076 sec for the method of Chung and 0,293 sec for the computation on the pixel values using eq. (2).

All computations are performed using the C programming language and a 1.5GHz Athlon PC with 1GB RAM.

6. Conclusion

In this paper two methods for the computation of gray image moments are proposed. The first method provides the exact moment values and operates much faster than the conventional moment calculation and faster than other fast moment calculation technique. The second method provides approximated moment values and operates in real-time.

The two methods are suitable for pattern recognition, image retrieval, stereo matching and all the applications where moments are used.

References

[1] R. J. Prokop and A. P. Reeves, "A survey of moment-based techniques for unoccluded object representation and recognition", *Graphical Models and Image Processing, 54*(5), 1992, 438-460.

[2] L. Yang, F. Albregtsen, "Fast and exact computation of cartesian geometric moments using discrete Green's theorem", *Pattern Recognition, 29*(7), 1996, 1061–1073.

[3] M.K. Hu, "Visual pattern recognition by moment invariants", *IRE Trans Inf Theory*, *8*(1), 1962, 179–187.

[4] M.F. Zakaria, L.J. Vroomen, P. Zsombor-Murray, J.M. van Kessel, "Fast algorithm for the computation of moment invariants", *Pattern Recognition, 20*(6), 1987, 639–643.

[5] M. Dai, P. Baylou, M. Najim, "An efficient algorithm for computation of shape moments from run-length codes or chain codes", *Pattern Recognition*, *25*(10), 1992, 1119–1128.

[6] B.C.Li, "A new computation of geometric moments", *Pattern Recognition*, *26*(1), 1993, 109–113.

[7] J. Flusser, Fast calculation of geometric moments of binary images, *Proc. 22nd OAGM'98 Workshop Pattern Recognition.Medical Computer Vision,* Illmitz, Austria, 1998, 265–274.

[8] I.M. Spiliotis, B.G. Mertzios, "Real-time computation of two-dimensional moments on binary images using image block representation", *IEEE Trans Image Processing, 7*(11), 1998, 1609–1615.

[9] K. Chung, P. Chen, "An efficient algorithm for computing moments on a block representation of a gray-scale image", *Pattern Recognition*, *38*(12), 2005, 2578–2586.

[10] I.M. Spiliotis and B.G. Mertzios, "Fast algorithms for basic processing and analysis operations on block represented binary images", *Pattern Recognition Letters, 17*(14), 1996, 1437-1450.

[11] S. Liao, M. Pawlak "On image analysis by moments", *IEEE Trans PAMI*, *18*(3), 1996, 254-266.

[12] J. Flusser, "Refined moments calculation using image block representation", *IEEE Trans Image Processing*, *9*(11), 2000, 1977-1978.