Available online at www.sciencedirect.com
ScienceDirect
Applied Mathematics and Computation 189 (2007) 1214–1222
APPLIED MATHEMATICS AND COMPUTATION
www.elsevier.com/locate/amc

Exact and fast computation of geometric moments for gray level images

Khalid M. Hosny *

Department of Computer Science, Faculty of Computers and Informatics, Zagazig University, Zagazig, Egypt

Abstract

Geometric moments are widely used in image processing and pattern recognition. While several methods have been proposed, exact geometric moment's computation for gray level images is still unavailable. In this paper exact values of geometric moments are calculated using mathematical integration of the monomial terms over digital image pixels. This method removed the numerical approximation errors involved in conventional methods. A fast algorithm is proposed to accelerate the moment's computations. The method is extended to compute the three-dimensional moments. A comparison with other conventional methods is performed. The obtained results explained the superiority of the proposed method.

Keywords: Geometric moments; Fast algorithm; Exact computation; 3D moments; Gray level images

1. Introduction

Since Hu introduced the moment invariants [1], geometric moments have been widely used in image processing and pattern recognition. Some applications where geometric moments used are: aircraft identification, scene matching, shape analysis, image normalization, character recognition, accurate position detection, color texture recognition, image retrieval and various other image processing tasks. For an overview of the subject see [2].

There are several approaches to calculate geometric moments. The conventional direct method which depends on using zeros-order approximation is time consuming and produces a significant error. Zakaria et al. [3], Dai et al. [4], Li [5], and Flusser [6] proposed various approaches based on the decomposition of the object into rows or row segments. Another 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]. Liao and Pawlak [7] proposed a more accurate approximation formula for computing the 2D geometric moments of a digital image when an analog original image was digitized. Then they used an alternative extended Simpson's rule to numerically calculate a double integral function for a higher order of geometric moments in each pixel.

* Present address: Department of Computer Science, Nejran Community College, Nejran, P.O. Box 1988, Saudi Arabia.
E-mail address: k_hosny@yahoo.com

0096-3003/$ - see front matter
doi:10.1016/j.amc.2006.12.025

Spiliotis and Mertzios [8] proposed a novel method which employs binary image representation by non-overlapping rectangular homogeneous blocks. An image moment is then calculated as a sum of moments of all blocks. Flusser [9] refined this method. Sossa et al. [10] proposed a new algorithm based on a morphologic decomposition of the binary image into a set of closed disks. Sossa and Flusser [11] refined the original work. Recently, Chung and Chen [12] extended the algorithm of Spiliotis and Mertzios [8] to approximately compute the lower order moments for a gray level image using the block representation.

This paper proposes a new method for accurate and efficient computation of geometric Moments for both binary and gray level images. A set of two-dimensional geometric moments are computed exactly by using a mathematical integration of the nominal polynomials, then, a fast algorithm is applied for computational complexity reduction. The proposed method is extended to compute 3D geometric moments. Experimental studies and the complexity analysis clearly show the superiority of this proposed method over the conventional ones.

The rest of the paper is organized as follows: In Section 2, an overview of geometric moments is given. The proposed method is described in Section 3. Section 4 aims to give a comparison between the computational time and some experimental results. Conclusion and concluding remarks are presented in Section 5.

2. Geometric moments

Two-dimensional geometric moments have the form of the projection of the image function $f(x,y)$ onto the nominal $x^p y^q$. The $(p+q)$ order geometric moments M_{pq} are defined as:

$$M_{pq} = \int_{-1}^{1}\int_{-1}^{1} x^p y^q f(x,y)\,\mathrm{d}x\,\mathrm{d}y. \tag{1}$$

A digital image of size $M \times N$ is an array of pixels. Centers of these pixels are the points (x_i, y_j), where the image intensity function is defined only for this discrete set of points $(x_i, y_j) \in [-1,1] \times [-1,1]$. $\Delta x_i = x_{i+1} - x_i$, $\Delta y_j = y_{j+1} - y_j$ are sampling intervals in the x- and y-directions, respectively. In the literature of digital image processing, the intervals Δx_i and Δy_j are fixed at constant values $\Delta x_i = 2/M$, and $\Delta y_j = 2/N$ respectively. Therefore, the set of points (x_i, y_j) will be defined as follows:

$$x_i = -1 + \left(i - \frac{1}{2}\right)\Delta x, \tag{2.1}$$

$$y_j = -1 + \left(j - \frac{1}{2}\right)\Delta y, \tag{2.2}$$

with $i = 1,2,3,\ldots,M$, and $j = 1,2,3,\ldots,N$. For the discrete-space version of the image, Eq. (1) is usually approximated as:

$$\tilde{M}_{pq} = \sum_{i=1}^{M}\sum_{j=1}^{N} x_i^p y_j^q f(x_i, y_j)\Delta x \Delta y. \tag{3}$$

Eq. (3) is so-called direct method for geometric moment's computations, which is the approximated version using zeroth-order approximation (ZOA). As indicated by Liao and Pawlak [7], Eq. (3) is not a very accurate approximation of Eq. (1). To improve the accuracy, they proposed to use the approximated form:

$$M_{pq} = \sum_{i=1}^{M}\sum_{j=1}^{N} h_{pq}(x_i, y_j) f(x_i, y_j), \tag{4}$$

where

$$h_{pq}(x_i, y_j) = \int_{x_i - \frac{\Delta x_i}{2}}^{x_i + \frac{\Delta x_i}{2}} \int_{y_j - \frac{\Delta y_j}{2}}^{y_j + \frac{\Delta y_j}{2}} x^p y^q\,\mathrm{d}x\,\mathrm{d}y. \tag{5}$$

Liao and Pawlak proposed an alternative extended Simpson's rule to evaluate the double integral defined by Eq. (5), then used to calculate the geometric moments defined by Eq. (4).

3. The proposed method

The approximation of the integral terms in Eq. (5) is responsible for the approximation error of geometric moments. These integrals need to be evaluated exactly to remove the approximation error. To achieve this, a new accurate and fast method will be discussed for exact geometric moment's computation.

3.1. Exact computation of geometric moments

Eq. (5) can be written as following:

$$h_{pq}(x_i, y_j) = I_p(i)I_q(j), \tag{6}$$

where

$$I_p(i) = \int_{x_i-\frac{\Delta x_i}{2}}^{x_i+\frac{\Delta x_i}{2}} x^p \,\mathrm{d}x = \frac{1}{p+1}[U_{i+1}^{p+1} - U_i^{p+1}], \tag{7.1}$$

$$I_q(j) = \int_{y_j-\frac{\Delta y_j}{2}}^{y_j+\frac{\Delta y_j}{2}} y^q \,\mathrm{d}y = \frac{1}{q+1}[V_{j+1}^{q+1} - V_j^{q+1}]. \tag{7.2}$$

The upper and lower limits of the integration in Eq. (7) have the values:

$$U_{i+1} = x_i + \frac{\Delta x_i}{2} = -1 + i\Delta x_i, \tag{8.1}$$

$$U_i = x_i - \frac{\Delta x_i}{2} = -1 + (i-1)\Delta x_i. \tag{8.2}$$

Similarly,

$$V_{j+1} = y_j + \frac{\Delta y_j}{2} = -1 + j\Delta y_j, \tag{9.1}$$

$$V_j = y_j - \frac{\Delta y_j}{2} = -1 + (j-1)\Delta y_j. \tag{9.2}$$

Substituting Eqs. (7.1) and (7.2) into (5), the set of geometric moments can thus be computed exactly by:

$$\hat{M}_{pq} = \sum_{i=1}^{M}\sum_{j=1}^{N} I_p(i)I_q(j)f(x_i, y_j). \tag{10}$$

The moment kernel of exact 2D geometric moments is defined by Eq. (7). This kernel is independent of image. Therefore, this kernel can be pre-computed, stored, recalled whenever it is needed to avoid repetitive computation.

3.2. Fast algorithm

Computation of exact geometric moments using Eq. (10) is similar to the direct method, which is very time consuming. Similar to the method of Fourier transform, the principle advantage of separability property is that: the 2D $(p+q)$-order geometric moment can be obtained in two steps by successive computation of the 1D qth order moments for each row. A fast method for exact geometric moment's computation will be proposed. Eq. (10) will be rewritten in a separable form as follows:

$$\hat{M}_{pq} = \sum_{i=1}^{M} I_p(i)Y_{iq}, \tag{11}$$

where

$$Y_{iq} = \sum_{j=1}^{N} I_q(j)f(x_i, y_j). \tag{12}$$

Y_{iq} in Eq. (12) is the qth order moment of row i. Since,

$$I_0(i) = 2/M. \tag{13}$$

Substitute Eq. (13) into Eq. (7), yields;

$$\hat{M}_{0q} = \frac{2}{M}\sum_{i=1}^{M} Y_{iq}. \tag{14}$$

3.3. Three-dimensional geometric moments

The proposed method is extended easily to cover the 3D case, where the 3D geometric moments are defined as:

$$M_{pqr} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q z^r f(x,y,z)\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z. \tag{15}$$

A 3D digital image or shape of size $M \times N \times K$ is a multidimensional array of voxels. Centers of these pixels are the points (x_i, y_j, z_ℓ), where the image intensity function is defined only for this discrete set of points $(x_i, y_j, z_\ell) \in [x_1, x_2] \times [y_1, y_2] \times [z_1, z_2]$; where $\Delta x_i = x_{i+1} - x_i$,$\Delta y_j = y_{j+1} - y_j$, $\Delta z_\ell = z_{\ell+1} - z_\ell$ are sampling intervals in the x-, y- and z-direction respectively. In the literature of digital image processing, the intervals Δx_i, Δy_j and Δz_ℓ are fixed at constant values $\Delta x_i = (x_2 - x_1)/M$, $\Delta y_j = (y_2 - y_1)/N$ and $\Delta z_\ell = (z_2 - z_1)/K$ respectively. The set of the 3D geometric moments computed exactly by:

$$\hat{M}_{pqr} = \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{\ell=1}^{K} I_p(i) I_q(j) I_r(\ell) f(x_i, y_j, z_\ell), \tag{16}$$

where $I_p(i)$, $I_q(j)$ are defined by Eq. (7), while $I_r(\ell)$ is defined as:

$$I_r(\ell) = \int_{z_\ell - \frac{\Delta z_\ell}{2}}^{z_\ell + \frac{\Delta z_\ell}{2}} z^r dz = \frac{1}{r+1}[W_{\ell+1}^{r+1} - W_\ell^{r+1}]. \tag{17}$$

The fast algorithm for exact 3D geometric moments (Eq. (16)) is summarized as follows:

$$\hat{M}_{pqr} = \sum_{\ell=1}^{K} I_r(\ell) L_\ell. \tag{18}$$

L_ℓ in Eq. (18) represents the 2D geometric moments for the plate of order ℓ.

4. Experimental results

In this section, the validity proof of the proposed method will be presented, where the computed values are compared with theoretical ones. The theoretical values are computed for relatively small artificial images so that hand calculations can be employed. The performance for the proposed method is evaluated and compared with the ZOA method. Finally, a comparison between the computation times of the proposed and the direct method is presented.

4.1. First image

As mentioned above, artificial images are used to prove validity of the proposed methods. A special image whose function $f(x,y)$ has the same constant value 1 for all (x,y) is considered. In such a case, theoretical geometric moment's values of this image are calculated by the following equation:

$$M_{pq} = \int_{-1}^{1}\int_{-1}^{1} x^p y^q\,\mathrm{d}x\,\mathrm{d}y = \left(\frac{(1)^{p+1} - (-1)^{p+1}}{p+1}\right)\left(\frac{(1)^{q+1} - (-1)^{q+1}}{q+1}\right). \tag{19}$$

Table 1
Comparison between moment values of the first image $f(x, y) = 1$

Theoretical, M_{pq}					
4.0000	0	1.3333	0	0.8000	0
0	0	0	0	0	0
1.3333	0	0.4444	0	0.2667	0
0	0	0	0	0	0
0.8000	0	0.2667	0	0.1600	0
0	0	0	0	0	0
ZOA, $\tilde{M}_{pq}$					
4.0000	0	1.2500	0	0.6406	0
0	0	0	0	0	0
1.2500	0	0.3906	0	0.2002	0
0	0	0	0	0	0
0.6406	0	0.2002	0	0.1026	0
0	0	0	0	0	0
Exact, $\hat{M}_{pq}$					
4.0000	0	1.3333	0	0.8000	0
0	0	0	0	0	0
1.3333	0	0.4444	0	0.2667	0
0	0	0	0	0	0
0.8000	0	0.2667	0	0.1600	0
0	0	0	0	0	0

The results are shown in Table 1. It is obvious that the exact values ($\hat{M}_{pq}$, (7)–(9)) match the theoretical values (M_{pq}, (19)) while that of ZOA ($\tilde{M}_{pq}$, (3)) deviates from the theoretical values.

4.2. Second image

Consider the artificial image $f(x_i, y_j)$. The image is represented by the matrix $A = [3,2,1,5;6,1,7,3;2,8,4,6;5,1,4,2]$. Theoretical values for digital gray level images can be calculated using the following form:

$$M_{pq} = \sum_{i=1}^{M}\sum_{j=1}^{N} f(x_i, y_j) \times \left(\frac{\left(x_i + \frac{\Delta x_i}{2}\right)^{p+1} - \left(x_i - \frac{\Delta x_i}{2}\right)^{p+1}}{p+1}\right)\left(\frac{\left(y_j + \frac{\Delta y_j}{2}\right)^{q+1} - \left(y_j - \frac{\Delta y_j}{2}\right)^{q+1}}{q+1}\right). \tag{20}$$

The results are shown in Table 2. As shown in the previous test case, exact values ($\hat{M}_{pq}$, (7)–(9)) match the theoretical values (M_{pq}, (20)) while that of ZOA ($\tilde{M}_{pq}$, (3)) deviates from the theoretical values especially when the order increases. These results ensure the validity of proposed method.

4.3. Error analysis

To analyze the computational errors a comparison between the obtained results will be performed. To make such comparison we will construct a 1D array of moments from the computed 2D array. The following algorithm is designed to perform the conversion process:

```
for p = 0 to Max
   for q = p to 0
      k = 0.5 * (p + 1) * (p + 2) - q - 1;
      I(k) = M(q, p - q);
   endfor
endfor
```

Table 2
Comparison between moment values of the second artificial image

Theoretical, M_{pq}					
15.0000	0.2500	5.2500	0.0313	3.1875	0.0052
0.3750	−0.3438	−0.0938	−0.2305	−0.0891	−0.1634
4.1250	−0.0104	1.6146	−0.0482	1.0047	−0.0393
0.1406	−0.3008	−0.0508	−0.2368	−0.0451	−0.1723
2.3438	−0.0203	0.9484	−0.0377	0.5941	−0.0297
0.0859	−0.2220	−0.0345	−0.1781	−0.0302	−0.1300
ZOA, $\tilde{M}_{pq}$					
15.0000	0.2500	4.9375	0.0156	2.5586	0.0010
0.3750	−0.3438	−0.1016	−0.2090	−0.0767	−0.1185
3.8125	−0.0156	1.4258	−0.0479	0.7571	−0.0294
0.1172	−0.2793	−0.0474	−0.2050	−0.0337	−0.1183
1.8555	−0.0186	0.7175	−0.0305	0.3832	−0.0184
0.0601	−0.1625	−0.0260	−0.1208	−0.0184	−0.0698
Exact, $\hat{M}_{pq}$					
15.0000	0.2500	5.2500	0.0313	3.1875	0.0052
0.3750	−0.3438	−0.0938	−0.2305	−0.0891	−0.1634
4.1250	−0.0104	1.6146	−0.0482	1.0047	−0.0393
0.1406	−0.3008	−0.0508	−0.2368	−0.0451	−0.1723
2.3438	−0.0203	0.9484	−0.0377	0.5941	−0.0297
0.0859	−0.2220	−0.0345	−0.1781	−0.0302	−0.1300

Table 3
The conversion of a 2D array to a 1D vector

p	q	k
0	0	0
1	1	1
1	0	2
2	2	3
2	1	4
2	0	5
3	3	6
3	2	7
3	1	8
3	0	9

Table 3 shows the conversion process, where **I** refers to the 1D array and **M** to the 2D array. **Max** is the maximum order of moments $(p+q)$. For each element of the vector **I**, we compute the relative error:

$$\text{Relative Error} = \left|\frac{\text{Theoretical Value} - \text{Calculated Value}}{\text{Theoretical Value}}\right|. \tag{21}$$

For the first test image, we plot the relative errors of the proposed method and the direct ZOA method for each element of the moment vector in Fig. 1. Similarly, Fig. 2 presents a comparison of relative errors of the second test image. It is clear that, the ZOA relative error increases as the order of the moment increase while the relative error of the proposed method equal zero for all moments.

4.4. Computational time

The computation time is a crucial issue. The CPU time required to compute geometric moments by using three different methods will be compared. These methods are ZOA represented by Eq. (3), the direct exact method represented by Eq. (19) and the proposed method represented by the set of Eqs. (11)–(14). The

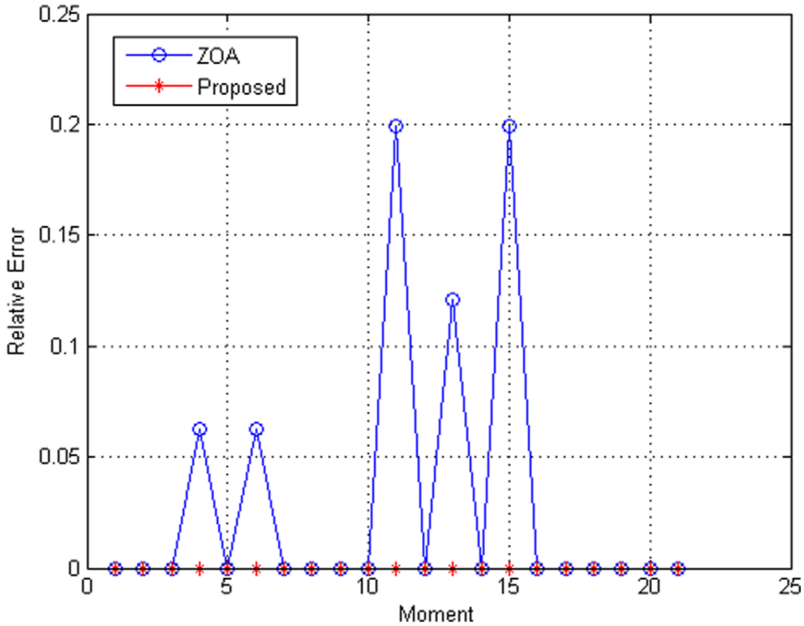

Fig. 1. Relative errors of the first test image.

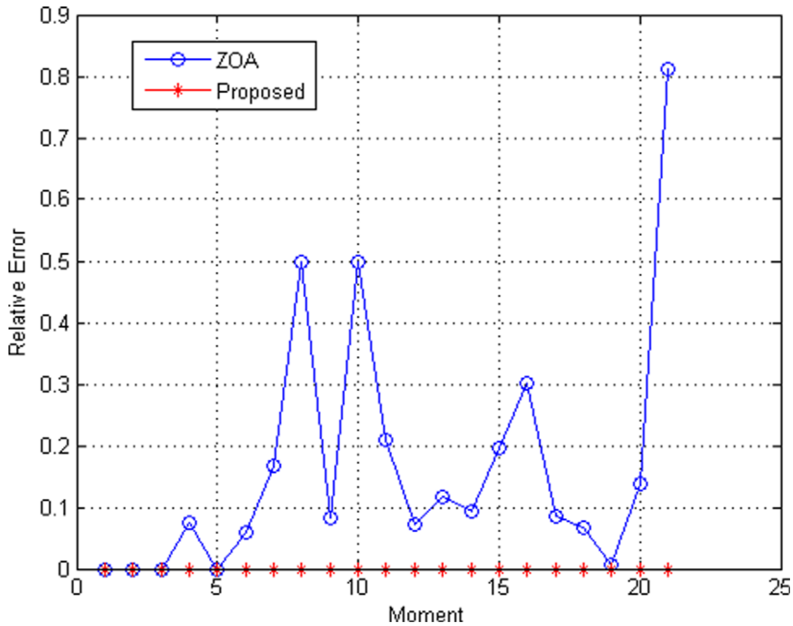

Fig. 2. Relative errors of the second test image.

CPU elapsed times of geometric moment computation for a digital gray scale image of size 512×512 (image of baboon) are displayed in Table 4. All computations are performed using Matlab7 on a 1.8 P4 processor with RAM 512 MB.

According to the big different scale of the CPU elapsed time required by the three different methods, Fig. 3 shows only the ZOA and the proposed method and ignores the third method. It is clear that, both ZOA and the direct exact method are impractical, while the proposed method tremendously reduced the computational time. This comparison ensures the superiority of our method.

Table 4
CPU elapsed time required for geometric moment's computation (s)

	ZOA	Direct exact method	Proposed method
N = 512, Max = 10	84.750	182.719	0.4380
N = 512, Max = 20	500.313	1090.1	0.8280
N = 512, Max = 30	1257	2650.8	1.870

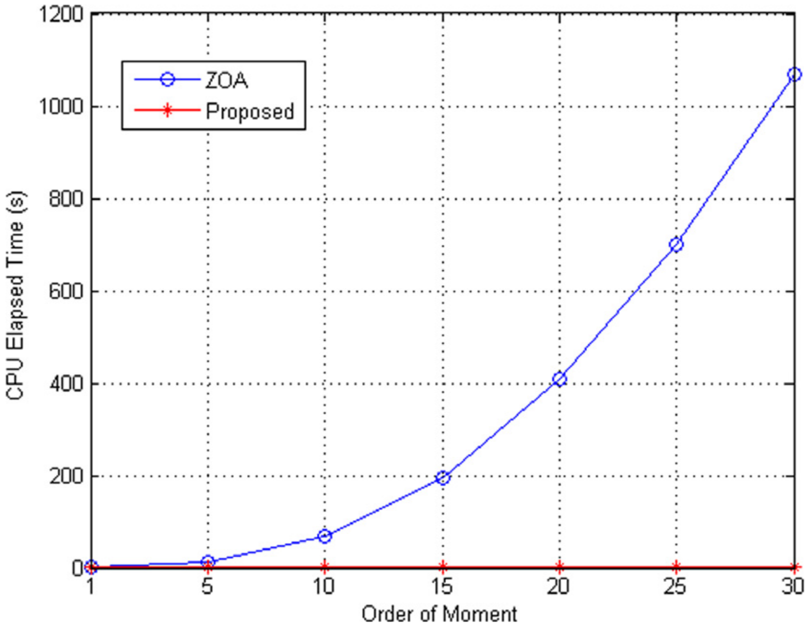

Fig. 3. CPU elapsed time in seconds.

5. Conclusion

This paper proposes a new exact and fast method for computing 2D, and 3D geometric moments for gray level images. The geometric moment values calculated by using the approximated method are deviated from those theoretical values, where the error steadily increases as the moment order increases. On the other hand, geometric moments calculated using the proposed method are identical to those obtained by theoretical calculations. The computation time of the proposed method is extremely smaller than that of the approximated method and the direct method. It is obvious that, the proposed method is more outperformed than all available methods for geometric moment computations.

References

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

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

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

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

[5] B.C. Li, A new computation of geometric moments, Pattern Recognit. 26 (1) (1993) 109–113.

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

[7] S.X. Liao, M. Pawlak, On image analysis by moments, IEEE Trans. Pattern Anal. Mach. Intell. 18 (3) (1996) 254–266.

[8] I.M. Spiliotis, B.G. Mertzios, Real-time computation of two-dimensional moments on binary images using image block representation, IEEE Trans. Image Process. 7 (1998) 1609–1615.
[9] J. Flusser, Refined moment calculation using image block representation, IEEE Trans. Image Process. 9 (11) (2000).
[10] H. Sosa, C. Yañez, J.L. Díaz, Computing geometric moments using morphological erosions, Pattern Recognit. 34 (2) (2001) 271–276.
[11] J. Flusser, H. Sossa, Refined method for the fast and exact computation of moment invariants, in: A. Sanfeliu et al. (Ed.), CIARP 2004, LNCS, vol. 3287, 2004, pp. 487–494.
[12] Kuo-Liang Chung, Ping-Chin Chen, An efficient algorithm for computing moments on a block representation of a gray-scale image, Pattern Recognit. 38 (12) (2005) 2578–2586.