International Journal of Innovative Computing, Information and Control
Volume **8**, Number **9**, September **2012**

ICIC International ©2012 ISSN 1349-4198
pp. 6123–6140

A NOVEL SYMMETRY-BASED METHOD FOR EFFICIENT COMPUTATION OF 2D AND 3D GEOMETRIC MOMENTS

Khalid Mohamed Hosny[1,2]

[1]Department of Computer Science
Najran Community College
Najran University
P. O. Box 1988, Najran, Saudi Arabia

[2]Department of Information Technology
Faculty of Computers and Informatics
Zagazig University
Zagazig 55914, Egypt
k_hosny@yahoo.com

Received June 2011; revised December 2011

Abstract. *Moment invariants are widely used in reconstruction, recognition and discrimination of 2D and 3D images. Accurate and efficient computation of these moment invariants is a big challenge to the community of image processing and pattern recognition. Since the wide class of moment invariants of digital images is generally expressed as a combination of geometric moments, highly accurate and efficient computation of 2D and 3D geometric moments is a very desirable target. In this work, a novel highly efficient symmetry-based method is proposed for exact computation of 2D and 3D geometric moments. Exact values of 2D and 3D geometric moments are calculated by using pixel- and voxel-wise integration of the monomial terms over digital image pixels/voxels. Three types of symmetry are applied to reduce the computational complexity of 2D geometric moments by 87%. The proposed method is extended to compute 3D geometric moments, where the computational complexity is reduced by more than 93%. The proposed method is adapted to compute different families of continuous and discrete orthogonal moments. A comparison with other existing methods is performed where the numerical experiments and the memory storage analysis ensure the efficiency of the proposed method.*

Keywords: Geometric moments, Symmetry property, Fast algorithm, Exact computation, 3D moments, Gray level images, Discrete orthogonal moments

1. Introduction.

Hu in his famous work [1] introduced the concept of geometric moment and derived a set of two-dimensional (2D) geometric moments that are invariants with respect to translation, scaling and rotation. Sadjadi and Hall [2] extended the work of Hu and derived a set of 3D geometric moment invariants. Teague [3] introduced the concept of orthogonal moments. Multidimensional geometric moments and orthogonal moments with their invariants have been widely used in the different applications such as optimum portfolio management [4], content-based retrieval [5], object identification and positioning [6], medical image processing [7-11], face recognition [12], fingerprint matching and verification [13,14].

Reconstruction, recognition and discrimination of 3D images/objects gained more interest during the last decade. The 3D moment invariants are used as shape descriptors. Novotni and Klein [15] are used 3D Zernike descriptors for content-based shape retrieval where the set of 3D Zernike descriptors is computed according to their relation with the 3D geometric moments. Recently, 3D moment invariants play a crucial role in the shape analysis and understanding protein structure-function relationships [16-18]. Generally, 3D moment invariants are promising descriptors in the field of electronic microscopic imaging where 3D descriptors could be used to build up 3D views of biological entities such as proteins, nucleic acids, cells, tumors, tissues, and whole organs or organisms.

Computational process of geometric moments and moment invariants encounters two challenging problems. The first one is the accuracy where approximate computation of geometric moments results in a set of inaccurate moments. The degraded accuracy negatively affects the performance of the geometric moment invariants especially in constructing discrimination classifiers. The second problem is concerned with the highly computational requirements especially for images of big sizes. For 3D moments, the computational problem becomes more critical.

The conventional direct method which depends on using zeros-order approximation (ZOA) is time-consuming and produces significant errors. Several methods and algorithms are proposed to overcome these challenging problems. Spiliotis and Mertzios [19] proposed a method which employs a binary image representation by non-overlapping rectangular homogeneous blocks, and then the image moments are calculated as the sum of the moments of all blocks. Recently, Papakostas et al. [20] extended the method of Spiliotis and Mertzios to compute geometric moments for gray level images. These methods attempt to improve the accuracy of the computed 2D geometric moments while the computational problem is still encountered. Unfortunately, these methods cannot be extended to compute the 3D geometric moments of volumetric images.

Liao and Pawlak [21] used another methodology. They proposed a formula for computing the 2D geometric moments of a digital image. They numerically integrate the monomial functions over digital image pixels by using Simpson's integration rule. Hosny [22] modified the method of Liao and Pawlak where he evaluates the double integration analytically and completely removes the numerical approximation error. Wee et al. [23] applied a symmetry property which was based on 1D monomial to reduce the computational process of gray level images. These methods are accurate. On the other side, their performance in reducing the computational process is limited. More reduction in computational process is desirable especially in the case of big images.

Only a few works are interested in the computation of 3D geometric moments. Yang et al. [24] used discrete divergence theorem for fast computation of 3D geometric moments of binary images. The method of Yang cannot be applied to gray level images. However, to the best of the authors' knowledge, no previously published papers presented the idea of symmetry-based efficient computation of 3D geometric moments for volumetric images.

The limitations in the existing methods for 2D geometric moment computation and the lack of efficient and accurate methods for 3D geometric moment computation have motivated the author to derive a novel symmetry-based method for highly efficient and accurate computation of 2D and 3D geometric moments. The wide class of image moments such as Zernike, pseudo Zernike, Legendre, Gaussian-Hermite, Tchebichef, Krawtchouk, Hahn, Racah, complex, and radial moments and their invariants could be expressed in terms of geometric moments of the same order or less [25]. This is an additional motivation where highly efficient and accurate computation of 2D and 3D geometric moment offers an excellent way to construct a library for highly efficient, accurate and easily programmable image moments.

This paper proposes a novel-symmetry based method for fast, memory-efficient and accurate computation of geometric moments for both 2D and 3D images/objects. The set of 2D geometric moments is computed exactly by using pixel-wise integration of the 2D monomials, and then, three kinds of symmetry properties are applied for computational complexity reduction. In the proposed method, the derived symmetry-based method has achieved 87% reduction for gray level images.

The proposed method is extended to compute 3D geometric moments where four kinds of symmetry properties are applied. A 93% reduction is achieved for gray level volumetric images. Conducted numerical experiments and complexity analysis clearly show the efficiency of the proposed method in comparison with the existing methods.

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 is devoted to discuss experimental results. Conclusion and concluding remarks are presented in Section 5.

2. **Approximate Geometric Moments.**

Geometric moments are defined as the projection of the image intensity function $f(x, y)$ onto the monomial $x^p y^q$. The geometric moments of order $(p + q)$, G_{pq}, are defined as:

$$G_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x^p y^q f(x, y) dx dy \tag{1}$$

For the discrete-space version of the image, Equation (1) was approximated where the integration is replaced by summation as follows:

$$\tilde{G}_{pq} = \sum_{i=1}^{N} \sum_{j=1}^{N} x_i^p y_j^q f(x_i, y_j) \Delta x \Delta y \tag{2}$$

Both Δx and Δy are the pixel width in x- and y-direction respectively. This process produces what is called approximation error. This error increases as the moment order increases. The accumulation of this error produces numerical instabilities which degrade the accuracy of the computed moments. Liao and Pawlak [21] show that, Equation (2) is not an accurate approximation of Equation (1). They attempt to improve the accuracy of the computed geometric moments by using the following form:

$$G_{pq} = \sum_{i=1}^{M} \sum_{j=1}^{N} f(x_i, y_j) \int_{x_i - \frac{\Delta x_i}{2}}^{x_i + \frac{\Delta x_i}{2}} \int_{y_j - \frac{\Delta y_j}{2}}^{y_j + \frac{\Delta y_j}{2}} x^p y^q dx dy, \tag{3}$$

Liao and Pawlak proposed an alternative extended Simpson's rule to evaluate the double integral in Equation (3), then used to calculate the geometric moments. In fact, their proposed method marginally improved the accuracy.

3. **The Proposed Method.**

All aspects of the proposed method are presented in this section. Mapping of 2D images and 3D objects is discussed in the first subsection. Detailed discussions of symmetry property for the 2D and 3D geometric moments are presented in the second and third subsections where the concept of augmented image/object intensity functions is discussed. The fourth subsection is devoted to discuss the efficient computation of exact 2D and 3D geometric moments. Extension to exactly compute 2D and 3D continuous and discrete orthogonal moments is briefly discussed in the fifth subsection.

3.1. **Image and object mapping.**

In the literature of digital image processing, the origin of a 2D digital image of size $N \times N$ is located on the upper left of the image and its indices, i and j increase from left to right and from top to bottom, respectively, i.e., $i, j = 1, 2, \ldots, N$ as shown in Figure 1(a). A kind of transformation is applied to the input image where the transformed image is defined in the square $[-1,1]\times[-1,1]$ as shown in Figure 1(b). This transformation could be done by using the following equations:

$$x_i = \frac{-N+2i-1}{N}, \quad y_j = -\frac{-N+2j-1}{N} \tag{4}$$

With $i, j = 1, 2, 3, \ldots, N$. The mapped digital image has the same size of the input image. This image is $N \times N$ array of pixels, where centers of these pixels are the points (x_i, y_j). The image intensity function is defined for this set of discrete points $(x_i, y_j) \in [-1,1]\times[-1,1]$ as shown in Figure 1(b). The sampling intervals in the x- and y-directions are $\Delta x_i = x_{i+1} - x_i$, $\Delta y_j = y_{j+1} - y_j$ respectively. In the literature of digital image processing, the intervals Δx_i and Δy_j are fixed at the constant value $\Delta x_i = \Delta y_j = 2/N$. It is clear that, the centre of input image is coinciding with the center of the square $[-1,1]\times[-1,1]$.

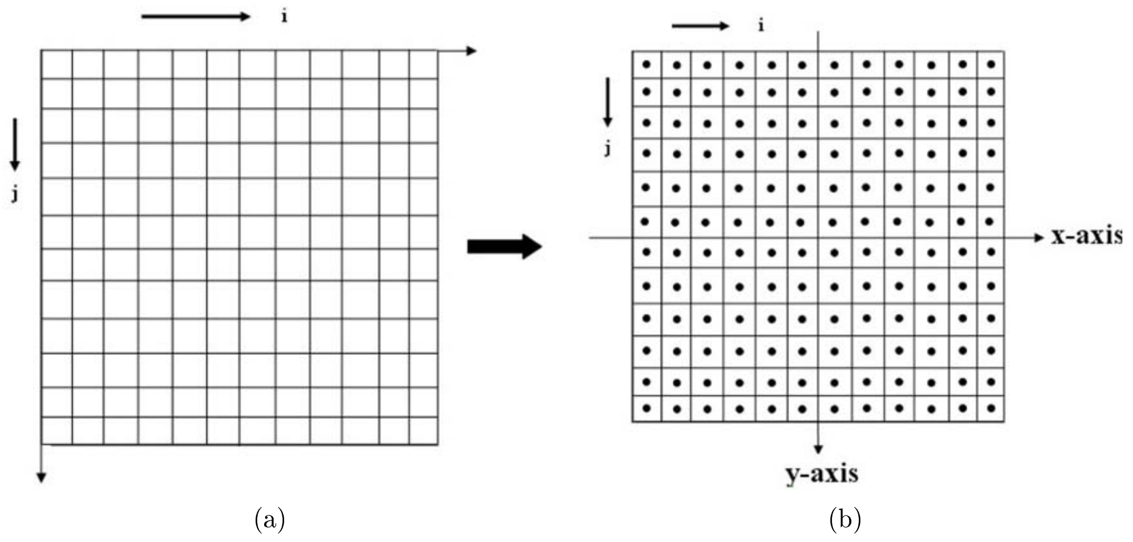


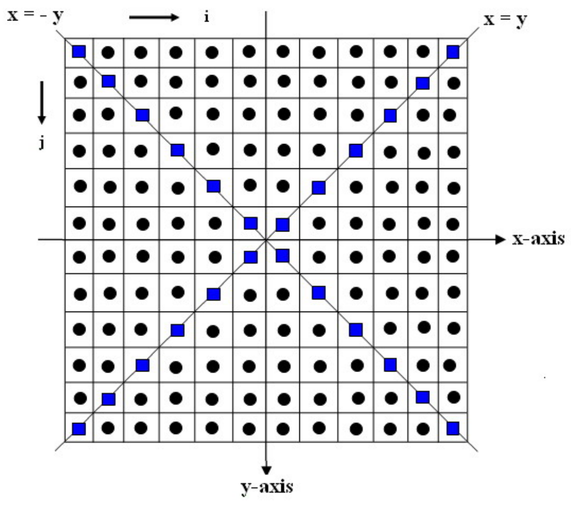
FIGURE 1. Image mapping: (a) original image, (b) mapped image

Similar to the 2D case, a 3D digital image/object of size $N \times N \times N$ is mapped into the cube $[-1,1]\times[-1,1]\times[-1,1]$ by using the following equation:

$$x_i = \frac{-N+2i-1}{N}, \quad y_j = \frac{-N+2j-1}{N}, \quad z_k = \frac{-N+2k-1}{N} \tag{5}$$

With $i, j, k = 1, 2, 3, \ldots, N$. This mapped object is $N \times N \times N$ array of voxels, where the centers of these voxels are the points (x_i, y_j, z_k). The object intensity function is defined for this set of points (x_i, y_j, z_k) where $\Delta x_i = x_{i+1} - x_i$, $\Delta y_j = y_{j+1} - y_j$, $\Delta z_k = z_{k+1} - z_k$ are sampling intervals in the x-, y- and z-direction respectively. The intervals Δx_i, Δy_j and Δz_k have the constant values, $\Delta x_i = \Delta y_j = \Delta z_k = 2/N$.

3.2. **Symmetry for 2D geometric moments.** The x- and y-axis in addition to the two lines $x = y$ and $x = -y$ divided the transformed image into eight octants as shown in Figure 2(a). Three types of symmetry could be observed and explored. In the first type, each point P_1 in the first octant with the Cartesian coordinates (x_i, y_j) has three similar points in the other three octants as shown in Figure 2(b). These points are P_4, P_5 and P_8 where the index associated with the point symbol referring to the octant number. The second type of symmetry is concerned with the rest of the eight points where the subscripts i and j are interchanged.



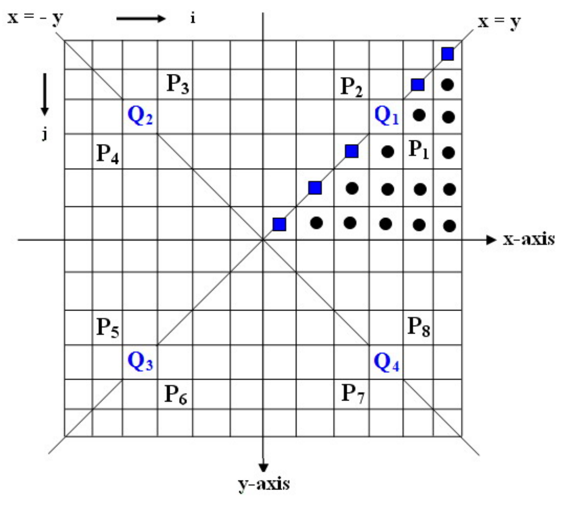


FIGURE 2. (a) The symmetry axes divide the transformed image into eight octants; (b) similar points of the different octants

The point $P_2(x_j, y_i)$ in the second octant has three sibling points P_3, P_6 and P_7. It must be noted that all of these eight points have the same radial distance to the origin point [26]. The third type of symmetry is concerned with the points which lie on the symmetrical lines, $x = y$ and $x = -y$. These points are $Q_1(x_i, x_i)$, Q_2, Q_3 and Q_4. All points of the three symmetrical types and their coordinates are shown in Table 1.

TABLE 1. Symmetry points and their coordinates: 2D case

First type	Second type	Third type
$P_1(x_i, y_j)$	$P_2(x_j, y_i)$	$Q_1(x_i, x_i)$
$P_4(x_{N-i+1}, y_j)$	$P_3(x_{N-j+1}, y_i)$	$Q_2(x_{N-i+1}, x_i)$
$P_5(x_{N-i+1}, y_{N-j+1})$	$P_6(x_{N-j+1}, y_{N-i+1})$	$Q_3(x_{N-i+1}, x_{N-i+1})$
$P_8(x_i, y_{N-j+1})$	$P_7(x_j, y_{N-i+1})$	$Q_4(x_i, x_{N-i+1})$

Since the points P_1, P_2, P_3, P_4, P_5, P_6, P_7 and P_8 has the same radial distance to the coordinate origin, then; the numerical value of $x^p y^q$ will be dependent on whatever p and q are even or odd. An illustrative example is presented to ensure this fact. Assume a small image of size 8×8. The point $P_1(x_7, y_3)$ in the first octant has the coordinates $P_1 = (5/8, 3/8)$. Consequently, the coordinates of the other corresponding seven points that refer to the octants ranging from 2 to 8 are: $P_2(3/8, 5/8)$, $P_3(-3/8, 5/8)$, $P_4(-5/8, 3/8)$, $P_5(-5/8, -3/8)$, $P_6(-3/8, -5/8)$, $P_7(3/8, -5/8)$ and $P_8(5/8, -3/8)$. Numerical values of $x^p y^q$ for the points P_1, P_4, P_5 and P_8; and $x^q y^p$ for the points P_2, P_3, P_6 and P_7 with different possibilities of exponent indices p and q are listed in Table 2. The values for the third type of symmetry are listed in Table 3. The computational process of 2D geometric moments requires only one octant.

Based on the first type of symmetry property, the image intensity function in different octants could be represented by only one augmented function. This function is a combination of the image intensity functions in the first, fourth, fifth and eighth octants respectively. The augmented function is defined as follows:

Case 1: p = Even, q = Even:

$$f_{k1}(x_i, y_j) = f_1(x_i, y_j) + f_4(x_i, y_j) + f_5(x_i, y_j) + f_8(x_i, y_j), \tag{6.1}$$

TABLE 2. First and second symmetry: values of the monomials $x^p y^q$ are dependent on whatever p and q are even or odd

p	q	$x^p y^q$				$x^q y^p$			
		P_1	P_4	P_5	P_8	P_2	P_3	P_6	P_7
E = 4	E = 2	+0.02145	+0.02145	+0.02145	+0.02145	+0.02145	+0.02145	+0.02145	+0.02145
E = 2	O = 1	+0.14648	+0.14648	−0.14648	−0.14648	+0.14648	−0.14648	−0.14648	+0.14648
O = 3	E = 2	+0.03433	−0.03433	−0.03433	+0.03433	+0.03433	+0.03433	−0.03433	−0.03433
O = 3	O = 1	+0.09155	−0.09155	+0.09155	−0.09155	+0.09155	−0.09155	+0.09155	−0.09155

TABLE 3. Third symmetry: values of the monomials $x^p y^q$ are dependent on whatever p and q are even or odd

p	Q	$x^p y^q$			
		Q_1	Q_2	Q_3	Q_4
E = 4	E = 2	+0.002780	+0.002780	+0.002780	+0.002780
E = 2	O = 1	−0.052734	−0.052734	+0.052734	+0.052734
O = 3	E = 2	+0.007415	−0.007415	−0.007415	+0.007415
O = 3	O = 1	−0.019775	+0.019775	−0.019775	+0.019775

Case 2: $p =$ Even, $q =$ Odd:

$$f_{k1}(x_i, y_j) = f_1(x_i, y_j) + f_4(x_i, y_j) - f_5(x_i, y_j) - f_8(x_i, y_j), \tag{6.2}$$

Case 3: $p =$ Odd, $q =$ Even:

$$f_{k1}(x_i, y_j) = f_1(x_i, y_j) - f_4(x_i, y_j) - f_5(x_i, y_j) + f_8(x_i, y_j), \tag{6.3}$$

Case 4: $p =$ Odd, $q =$ Odd:

$$f_{k1}(x_i y_j) = f_1(x_i, y_j) - f_4(x_i, y_j) + f_5(x_i, y_j) - f_8(x_i, y_j). \tag{6.4}$$

Similar to the first type of symmetry, the second augmented intensity function could be represented by a combination of the image intensity functions in the second, third, sixth and seventh octants respectively and defined as follows:
Case 1: $p =$ Even, $q =$ Even:

$$f_{k2}(x_i, y_j) = f_2(x_j, y_i) + f_3(x_j, y_i) + f_6(x_j, y_i) + f_7(x_j, y_i), \tag{7.1}$$

Case 2: $p =$ Even, $q =$ Odd:

$$f_{k2}(x_i, y_j) = f_2(x_j, y_i) - f_3(x_j, y_i) - f_6(x_j, y_i) + f_7(x_j, y_i), \tag{7.2}$$

Case 3: $p =$ Odd, $q =$ Even:

$$f_{k2}(x_i, y_j) = f_2(x_j, y_i) + f_3(x_j, y_i) - f_6(x_j, y_i) - f_7(x_j, y_i), \tag{7.3}$$

Case 4: $p =$ Odd, $q =$ Odd:

$$f_{k2}(x_i, y_j) = f_2(x_j, y_i) - f_3(x_j, y_i) + f_6(x_j, y_i) - f_7(x_j, y_i). \tag{7.4}$$

Similarly, the augmented intensity function of the third type of symmetry is represented as a combination of the image intensity functions for the points that lie on the symmetrical lines $x = y$ and $x = -y$ as follows:
Case 1: $p =$ Even, $q =$ Even:

$$f_{k3}(x_i, x_i) = Q_1(x_i, x_i) + Q_2(x_i, x_i) + Q_3(x_i, x_i) + Q_4(x_i, x_i), \tag{8.1}$$

Case 2: $p =$ Even, $q =$ Odd:

$$f_{k3}(x_i, x_i) = Q_1(x_i, x_i) + Q_2(x_i, x_i) - Q_3(x_i, x_i) - Q_4(x_i, x_i), \tag{8.2}$$

Case 3: $p =$ Odd, $q =$ Even:

$$f_{k3}(x_i, x_i) = Q_1(x_i, x_i) - Q_2(x_i, x_i) - Q_3(x_i, x_i) + Q_4(x_i, x_i), \tag{8.3}$$

Case 4: $p =$ Odd, $q =$ Odd:

$$f_{k3}(x_i, x_i) = Q_1(x_i, x_i) - Q_2(x_i, x_i) + Q_3(x_i, x_i) - Q_4(x_i, x_i). \tag{8.4}$$

3.3. **Symmetry for 3D geometric moments.** The main axes, x-, y- and z-axis, divide the cube $[-1, 1] \times [-1, 1] \times [-1, 1]$ into eight small cubes as shown in Figure 3(a). Each one of these small cubes could be divided into two prisms. Therefore, the original cube is divided into 16 small prisms. Similar to the 2D case, there are different types of symmetry that could be observed and explored. Four types of symmetry are observed. In the first type; each point P_1 in the first prism with the Cartesian coordinates (x_i, y_j, z_k) has eight similar points in the other eight prisms. These points are P_4, P_5, P_8, P_9, P_{12}, P_{13} and P_{16}. In the second type of symmetry, the other eight points, $P_2(x_j, y_i, z_k)$, P_3, P_6, P_7, P_9, P_{11}, P_{14} and P_{15}, are considered where the subscripts i and j are interchanged. All of these sixteen points have the same radial distance to the origin point. In the third type of symmetry, all points $Q_1(x_i, x_i, z_k)$, Q_2, Q_3, Q_4, Q_5, Q_6, Q_7 and Q_8 that have equal x- and y-coordinates with different z-coordinate. The final type of symmetry points represents the points that lie on the body diagonals of the whole cube. In other words, these points have equal Cartesian coordinates. These points are $D_1(x_i, x_i, x_i)$, D_2, D_3, D_4, D_5, D_6, D_7 and D_8. All points of the four symmetrical types and their Cartesian coordinates are shown in Table 4. The computed numerical values of $x^p y^q x^r$ for the points P_1, P_4, P_5, P_8, P_9, P_{12}, P_{13} and P_{16}; and $x^q y^p x^r$ for the points P_2, P_3, P_6, P_7, P_{10}, P_{11}, P_{14} and P_{15} with different possibilities of exponent indices p, q and r confirm the proposed idea. The entire computational process of 3D geometric moments could be done by using only one prism from the 3D object space as in Figure 3(b).

TABLE 4. Four types of symmetry points and their coordinates: 3D case

First type	Second type	Third type	Fourth type
$P_1(x_i, y_j, z_k)$	$P_2(x_j, y_i, z_k)$	$Q_1(x_i, x_i, z_k)$	$D_1(x_i, x_i, x_i)$
$P_4(x_{N-i+1}, y_j, z_k)$	$P_3(x_{N-j+1}, y_i, z_k)$	$Q_2(x_{i-N+1}, x_i, z_k)$	$D_2(x_{i-N+1}, x_i, x_i)$
$P_5(x_{N-i+1}, y_{N-j+1}, z_k)$	$P_6(x_{N-j+1}, y_{N-i+1}, z_k)$	$Q_3(x_i, x_{N-i+1}, z_k)$	$D_3(x_i, x_{N-i+1}, x_i)$
$P_8(x_i, y_{N-j+1}, z_k)$	$P_7(x_j, y_{N-i+1}, z_k)$	$Q_4(x_{N-i+1}, x_{N-i+1}, z_k)$	$D_4(x_{N-i+1}, x_{N-i+1}, x_i)$
$P_9(x_i, y_j, z_{N-k+1})$	$P_{10}(x_j, y_i, z_{N-k+1})$	$Q_5(x_i, x_i, z_{N-k+1})$	$D_5(x_i, x_i, x_{N-i+1})$
$P_{12}(x_{N-i+1}, y_j, z_{N-k+1})$	$P_{11}(x_{N-j+1}, y_i, z_{N-k+1})$	$Q_6(x_{i-N+1}, x_i, z_{N-k+1})$	$D_6(x_{i-N+1}, x_i, x_{N-i+1})$
$P_{13}(x_{N-i+1}, y_{N-j+1}, z_{N-k+1})$	$P_{14}(x_{N-j+1}, y_{N-i+1}, z_{N-k+1})$	$Q_7(x_i, x_{N-i+1}, z_{N-k+1})$	$D_7(x_i, x_{N-i+1}, x_{N-i+1})$
$P_{16}(x_i, y_{N-j+1}, z_{N-k+1})$	$P_{15}(x_j, y_{N-i+1}, z_{N-k+1})$	$Q_8(x_{N-i+1}, x_{N-i+1}, z_{N-k+1})$	$D_8(x_{N-i+1}, x_{N-i+1}, x_{N-i+1})$

Based on the four symmetry types, the object intensity function in different prisms could be represented by only one augmented function. The augmented function is defined as follows:

Case 1: $p =$ Even, $q =$ Even, $r =$ Even;

$$f_{A1}(x_i, y_j, z_k) = f_1 + f_4 + f_5 + f_8 + f_9 + f_{12} + f_{13} + f_{16}, \tag{9.1}$$

Case 2: $p =$ Even, $q =$ Even, $r =$ Odd;

$$f_{A1}(x_i, y_j, z_k) = f_1 + f_4 + f_5 + f_8 - f_9 - f_{12} - f_{13} - f_{16}, \tag{9.2}$$

Case 3: $p =$ Even, $q =$ Odd, $r =$ Even;

$$f_{A1}(x_i, y_j, z_k) = f_1 - f_4 - f_5 + f_8 + f_9 - f_{12} - f_{13} + f_{16}, \tag{9.3}$$

Case 4: $p =$ Even, $q =$ Odd, $r =$ Odd;

$$f_{A1}(x_i, y_j, z_k) = f_1 - f_4 - f_5 + f_8 - f_9 + f_{12} + f_{13} - f_{16}, \tag{9.4}$$

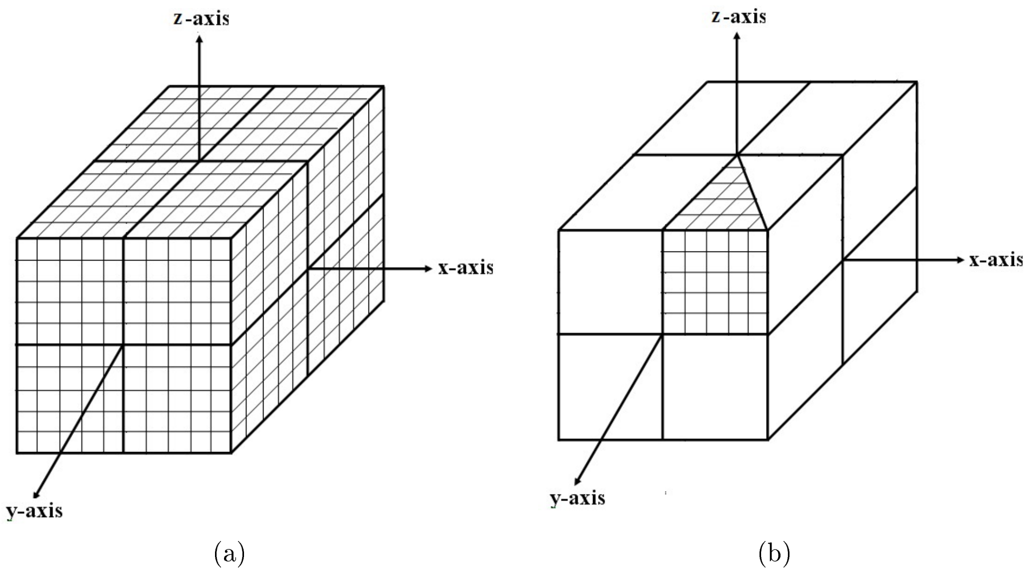


FIGURE 3. (a) A unit cube represents the 3D object space; (b) computational process of 3D GMs is done by using one prism

Case 5: p = Odd, q = Even, r = Even;

$$f_{A1}(x_i, y_j, z_k) = f_1 + f_4 - f_5 - f_8 + f_9 + f_{12} - f_{13} - f_{16}, \tag{9.5}$$

Case 6: p = Odd, q = Even, r = Odd;

$$f_{A1}(x_i, y_j, z_k) = f_1 + f_4 - f_5 - f_8 - f_9 - f_{12} + f_{13} + f_{16}, \tag{9.6}$$

Case 7: p = Odd, q = Odd, r = Even;

$$f_{A1}(x_i, y_j, z_k) = f_1 - f_4 + f_5 - f_8 + f_9 - f_{12} + f_{13} - f_{16}, \tag{9.7}$$

Case 8: p = Odd, q = Odd, r = Odd;

$$f_{A1}(x_i, y_j, z_k) = f_1 - f_4 + f_5 - f_8 - f_9 + f_{12} - f_{13} + f_{16}, \tag{9.8}$$

Similar to the first type of symmetry, the augmented intensity function of the second symmetry type is defined as follows:

Case 1: p = Even, q = Even, r = Even;

$$f_{A2}(x_i, y_j, z_k) = f_2 + f_3 + f_6 + f_7 + f_{10} + f_{11} + f_{14} + f_{15}, \tag{10.1}$$

Case 2: p = Even, q = Even, r = Odd;

$$f_{A2}(x_i, y_j, z_k) = f_2 + f_3 + f_6 + f_7 - f_{10} - f_{11} - f_{14} - f_{15}, \tag{10.2}$$

Case 3: p = Even, q = Odd, r = Even;

$$f_{A2}(x_i, y_j, z_k) = f_2 + f_3 - f_6 - f_7 + f_{10} + f_{11} - f_{14} - f_{15}, \tag{10.3}$$

Case 4: p = Even, q = Odd, r = Odd;

$$f_{A2}(x_i, y_j, z_k) = f_2 + f_3 - f_6 - f_7 - f_{10} - f_{11} + f_{14} + f_{15}, \tag{10.4}$$

Case 5: p = Odd, q = Even, r = Even;

$$f_{A2}(x_i, y_j, z_k) = f_2 - f_3 - f_6 + f_7 + f_{10} - f_{11} - f_{14} + f_{15}, \tag{10.5}$$

Case 6: p = Odd, q = Even, r = Odd;

$$f_{A2}(x_i, y_j, z_k) = f_2 - f_3 - f_6 + f_7 - f_{10} + f_{11} + f_{14} - f_{15}, \tag{10.6}$$

Case 7: $p =$ Odd, $q =$ Odd, $r =$ Even;

$$f_{A2}(x_i, y_j, z_k) = f_2 - f_3 + f_6 - f_7 + f_{10} - f_{11} + f_{14} - f_{15}, \tag{10.7}$$

Case 8: $p =$ Odd, $q =$ Odd, $r =$ Odd;

$$f_{A2}(x_i, y_j, z_k) = f_2 - f_3 + f_6 - f_7 - f_{10} + f_{11} - f_{14} + f_{15}, \tag{10.8}$$

For the third type of symmetry, the augmented function is represented as a combination of the object intensity functions as follows:

Case 1: $p =$ Even, $q =$ Even, $r =$ Even;

$$Q_{A3}(x_i, x_i, z_k) = Q_1 + Q_2 + Q_3 + Q_4 + Q_5 + Q_6 + Q_7 + Q_8, \tag{11.1}$$

Case 2: $p =$ Even, $q =$ Even, $r =$ Odd;

$$Q_{A3}(x_i, x_i, z_k) = Q_1 + Q_2 + Q_3 + Q_4 - Q_5 - Q_6 - Q_7 - Q_8, \tag{11.2}$$

Case 3: $p =$ Even, $q =$ Odd, $r =$ Even;

$$Q_{A3}(x_i, x_i, z_k) = Q_1 + Q_2 - Q_3 - Q_4 + Q_5 + Q_6 - Q_7 - Q_8, \tag{11.3}$$

Case 4: $p =$ Even, $q =$ Odd, $r =$ Odd;

$$Q_{A3}(x_i, x_i, z_k) = Q_1 + Q_2 - Q_3 - Q_4 - Q_5 - Q_6 + Q_7 + Q_8, \tag{11.4}$$

Case 5: $p =$ Odd, $q =$ Even, $r =$ Even;

$$Q_{A3}(x_i, x_i, z_k) = Q_1 - Q_2 - Q_3 + Q_4 + Q_5 - Q_6 - Q_7 + Q_8, \tag{11.5}$$

Case 6: $p =$ Odd, $q =$ Even, $r =$ Odd

$$Q_{A3}(x_i, x_i, z_k) = Q_1 - Q_2 - Q_3 + Q_4 - Q_5 + Q_6 + Q_7 - Q_8, \tag{11.6}$$

Case 7: $p =$ Odd, $q =$ Odd, $r =$ Even;

$$Q_{A3}(x_i, x_i, z_k) = Q_1 - Q_2 + Q_3 - Q_4 + Q_5 - Q_6 + Q_7 - Q_8, \tag{11.7}$$

Case 8: $p =$ Odd, $q =$ Odd, $r =$ Odd;

$$Q_{A3}(x_i, x_i, z_k) = Q_1 - Q_2 + Q_3 - Q_4 - Q_5 + Q_6 - Q_7 + Q_8. \tag{11.8}$$

Finally, the augmented function of the fourth type of symmetry is represented as a combination of the object intensity functions which lie on the body diagonals of the whole cube as follows:

Case 1: $p =$Even, $q =$ Even, $r =$ Even

$$D_{A4}(x_i, x_i, x_i) = D_1 + D_2 + D_3 + D_4 + D_5 + D_6 + D_7 + D_8, \tag{12.1}$$

Case 2: $p =$ Even, $q =$ Even, $r =$ Odd;

$$D_{A4}(x_i, x_i, x_i) = D_1 + D_2 + D_3 + D_4 - D_5 - D_6 - D_7 - D_8, \tag{12.2}$$

Case 3: $p =$ Even, $q =$ Odd, $r =$ Even;

$$D_{A4}(x_i, x_i, x_i) = D_1 - D_2 - D_3 + D_4 + D_5 - D_6 - D_7 + D_8, \tag{12.3}$$

Case 4: $p =$ Even, $q =$ Odd, $r =$ Odd;

$$D_{A4}(x_i, x_i, x_i) = D_1 - D_2 - D_3 + D_4 - D_5 + D_6 + D_7 - D_8, \tag{12.4}$$

Case 5: $p =$ Odd, $q =$ Even, $r =$ Even;

$$D_{A4}(x_i, x_i, x_i) = D_1 + D_2 - D_3 - D_4 + D_5 + D_6 - D_7 - D_8, \tag{12.5}$$

Case 6: $p =$ Odd, $q =$ Even, $r =$ Odd;

$$D_{A4}(x_i, x_i, x_i) = D_1 + D_2 - D_3 - D_4 - D_5 - D_6 + D_7 + D_8, \tag{12.6}$$

Case 7: $p = \text{Odd}$, $q = \text{Odd}$, $r = \text{Even}$;

$$D_{A4}(x_i, x_i, x_i) = D_1 - D_2 + D_3 - D_4 + D_5 - D_6 + D_7 - D_8, \tag{12.7}$$

Case 8: $p = \text{Odd}$, $q = \text{Odd}$, $r = \text{Odd}$;

$$D_{A4}(x_i, x_i, x_i) = D_1 - D_2 + D_3 - D_4 - D_5 + D_6 - D_7 + D_8. \tag{12.8}$$

3.4. **Exact computation of 2D and 3D geometric moments.** The approximation of the integral terms in Equation (3) is responsible for the approximation error of geometric moments. These integrals need to be evaluated exactly to remove the approximation error.

$$I_p(i) = \int_{x_i - \frac{\Delta x_i}{2}}^{x_i + \frac{\Delta x_i}{2}} x^p dx = \frac{1}{p+1}\left[U_{i+1}^{p+1} - U_i^{p+1}\right] \tag{13}$$

$$I_q(j) = \int_{y_j - \frac{\Delta y_j}{2}}^{y_j + \frac{\Delta y_j}{2}} y^q dy = \frac{1}{q+1}\left[V_{j+1}^{q+1} - V_j^{q+1}\right] \tag{14}$$

The upper and lower limits of the integration in Equations (13) and (14) are defined according [27]. Substituting Equations (13) and (14) into (3) yields a set of exact geometric moments. The computational complexity could be significantly reduced through the computation of the first octant only by applying the three types of symmetry as follows:

$$\begin{aligned}\hat{G}_{pq} = & \sum_{i=2}^{\lfloor \frac{N}{2} \rfloor} \sum_{j=1}^{i-1} I_p(i) I_q(j) f_{k1}(x_i, y_j) + \sum_{i=2}^{\lfloor \frac{N}{2} \rfloor} \sum_{j=1}^{i-1} I_q(i) I_p(j) f_{k2}(x_i, y_j) \\ & + \sum_{i=1}^{\lfloor \frac{N}{2} \rfloor} I_p(i) I_q(i) f_{k3}(x_i, x_i)\end{aligned} \tag{15}$$

where $\lfloor N/2 \rfloor$ equal $(N-1)/2$ for odd values of N and equal to $N/2$ otherwise. The augmented intensity functions $f_{k1}(x_i, y_j)$, $f_{k2}(x_i, y_j)$ and $f_{k3}(x_i, x_i)$ are defined by Equations (6), (7) and (8), respectively. The moment kernel of exact 2D geometric moments is defined by Equations (13) and (14). These kernels are image-independent. Therefore, these kernels could be pre-computed, stored, recalled whenever needed to avoid repetitive computation. The proposed method is extended to the 3D case, where 3D geometric moments are defined as:

$$G_{pqr} = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} x^p y^q z^r f(x, y, z) dx dy dz \tag{16}$$

For 3D digital objects of size $N \times N \times N$, the set of the 3D geometric moments is computed exactly by using the following equation:

$$\hat{G}_{pqr} = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N} I_p(i) I_q(j) I_r(k) f(x_i, y_j, z_k) \tag{17}$$

where $I_p(i)$, $I_q(j)$ are defined by Equations (13) and (14), while $I_r(k)$ is defined as:

$$I_r(k) = \int_{z_k - \frac{\Delta z_k}{2}}^{z_k + \frac{\Delta z_k}{2}} z^r dz = \frac{1}{r+1}\left[W_{k+1}^{r+1} - W_k^{r+1}\right] \tag{18}$$

By applying the symmetry property of the 3D case, the computational complexity could be significantly reduced through the computation of only one prism of the total sixteen prisms. Equation (17) is rewritten as follows:

$$\begin{aligned}\hat{G}_{pqr} = & \sum_{i=2}^{\lfloor\frac{N}{2}\rfloor}\sum_{j=1}^{i-1}\sum_{k=1}^{\lfloor\frac{N}{2}\rfloor} I_p(i) I_q(j) I_r(k) f_{A1}(x_i, y_j, z_k) \\ & + \sum_{i=2}^{\lfloor\frac{N}{2}\rfloor}\sum_{j=1}^{i-1}\sum_{k=1}^{\lfloor\frac{N}{2}\rfloor} I_q(i) I_p(j) I_r(k) f_{A2}(x_i, y_j, z_k) \\ & + \sum_{i=2}^{\lfloor\frac{N}{2}\rfloor}\sum_{k=1}^{\lfloor\frac{N}{2}\rfloor} I_p(i) I_q(i) I_r(k) f_{A3}(x_i, y_j, z_k) \\ & + \sum_{i=1}^{\lfloor\frac{N}{2}\rfloor} I_p(i) I_q(i) I_r(i) f_{A4}(x_i, y_j, z_k)\end{aligned} \tag{19}$$

3.5. **Extension to 2D/3D continuous and discrete orthogonal moments.** The wide class of image moments could be classified into orthogonal and non-orthogonal moments. Orthogonal moments are divided into continuous and discrete moments. Zernike, pseudo Zernike, Legendre, Gaussian-Hermite and Gegenbauer moments are examples of the orthogonal continuous moments while Tchebichef, Krawtchouk, Hahn and Racah moments are examples of the discrete orthogonal moments. On the other side, complex and radial moments are examples of non-orthogonal moments. As previously stated, all of these mentioned moments and their invariants could be expressed in terms of geometric moments of the same order or less [25]. This subsection presents a concise description of the implementation of the proposed in computation of both 2D and 3D continuous, discrete orthogonal and non-orthogonal moments.

The 2D continuous orthogonal moments such as Legendre, Gaussian-Hermite and Gegenbauer moments are defined over the square $[-1,1]\times[-1,1]$. The proposed method is applied to exactly compute the aforementioned sets of moments according to their relations with the 2D geometric moments without any modifications. The implementation to exactly 2D orthogonal Legendre moments is discussed as an example through the next subsections. For 3D objects, this family of continuous orthogonal moments is defined in the cube $[-1,1]\times[-1,1]\times[-1,1]$. Therefore, the implementation of the proposed method in 3D case is straightforward.

The 2D circular continuous orthogonal moments are defined over a unit disk. Zernike, pseudo Zernike, Fourier-Mellin are examples of these moments. The proposed method is slightly modified where the input image is mapped inside the unit circle and the mapped image is defined over the square $\lfloor -1/\sqrt{2}, 1/\sqrt{2}\rfloor \times \lfloor -1/\sqrt{2}, 1/\sqrt{2}\rfloor$. All other steps of the proposed method are applied without any modification. For 3D objects, the family of spherical continuous orthogonal moments is defined inside a unit ball. The input object is mapped to be defined in the cube $\lfloor -1/\sqrt{3}, 1/\sqrt{3}\rfloor \times \lfloor -1/\sqrt{3}, 1/\sqrt{3}\rfloor \times \lfloor -1/\sqrt{3}, 1/\sqrt{3}\rfloor$. Then, the implementation of the proposed method in 3D is straightforward.

On the other side, the 2D discrete orthogonal moments could be expressed in terms of geometric moments. Recently, Papakostas et al. [28] derived a set of translation, scaling and rotation Krawtchouk moment invariants by using 2D geometric moment invariants. The discrete orthogonal moments are defined over the square $[0, N-1]\times[0, N-1]$. In order to apply the proposed method with discrete orthogonal moments, the input squared image is divided into eight equal octants by using horizontal, vertical and diagonal lines and then the proposed method is applied where only one octant is used to compute the entire set of 2D moments.

For 3D image/object, the family of discrete orthogonal moments is defined over a cub $[0, N-1] \times [0, N-1] \times [0, N-1]$. This cube is divided into 16 similar parts. Each of these parts is a collection of voxels and could be represented as a prism. The rest of the steps are the same where only one prism is used to compute the entire set of 3D moments.

4. **Experimental Results.**

The conducted numerical experiments in this section concentrate on the accuracy and efficiency of the proposed method in both 2D and 3D cases. Image reconstruction and robustness against image noise are used to rate the accuracy of the proposed method. The computed geometric moments are used to compute a set of orthogonal moments where the later are used in the image reconstruction process. Mean square error (MSE) is used as a measure of the accuracy of the reconstruction process. Elapsed CPU times of the conducted numerical experiments and theoretical complexity analysis are used to judge the efficiency of the proposed method. The performance of the proposed method is compared with the performance of the existing methods.

4.1. **Accuracy.**

The accuracy of the proposed method could be tested by reconstructing the input image by using the computed moments and compare the reconstructed image with the original one. The proposed method is accurate if the Mean-Square Error (MSE) approaches zero as the moment order increases. At a certain moment order, both original and reconstructed images are almost identical. Image reconstruction by using non-orthogonal geometric moments is very difficult and impractical when higher orders are required [3]. This process could be easily achieved by using the orthogonal moments. Legendre moments are very popular orthogonal moments where this set of moments could be represented as a linear combination of geometric moments of the same order or less. Image reconstruction by using Legendre moments is used to rate the accuracy of the proposed method. For digital image of size $N \times N$, the MSE is defined as:

$$MSE = \frac{1}{N \times N} \sum_{i=1}^{N} \sum_{j=1}^{N} \left(\hat{f}_{Max}\left(x_i, y_j\right) - f\left(x_i, y_j\right) \right)^2 \tag{20}$$

where $\hat{f}_{Max}(x_i, y_j)$ and $f(x_i, y_j)$ are the intensity functions of the reconstructed and the original images respectively. The subscript Max refers to the maximum order used in the reconstruction process. The 2D Legendre moments of image function $f(x, y)$ are:

$$L_{pq} = \frac{(2p+1)(2q+1)}{4} \int_{-1}^{1} \int_{-1}^{1} P_p(x) P_q(y) f(x, y) dx dy, \tag{21}$$

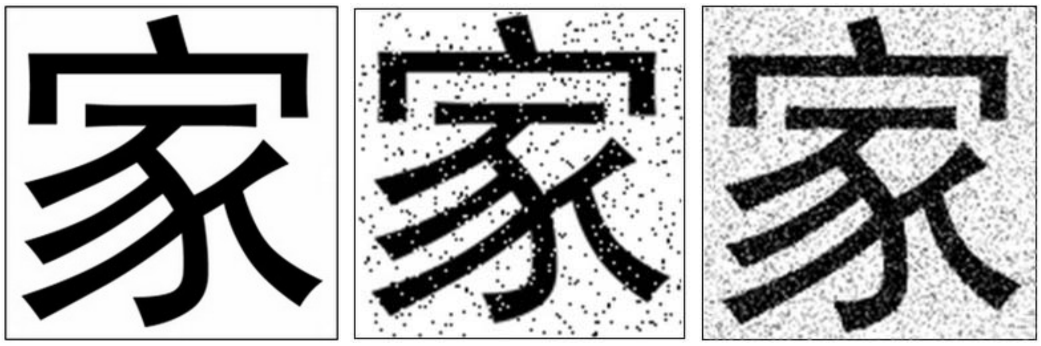
Legendre moments are expressed as a combination of geometric moments as follows:

$$L_{pq} = \frac{(2p+1)(2q+1)}{4} \sum_{i=0}^{\lfloor \frac{p}{2} \rfloor} \sum_{j=0}^{\lfloor \frac{q}{2} \rfloor} B_{p,i} B_{q,j} G_{p-2i,q-2j} \tag{22}$$

Legendre polynomial coefficients are defined in [25]. Geometric moments are computed by using the proposed method, and then used in Equation (22) to compute the corresponding Legendre moments. Fortunately, both sets of geometric and Legendre moments have the same number of independent moments.

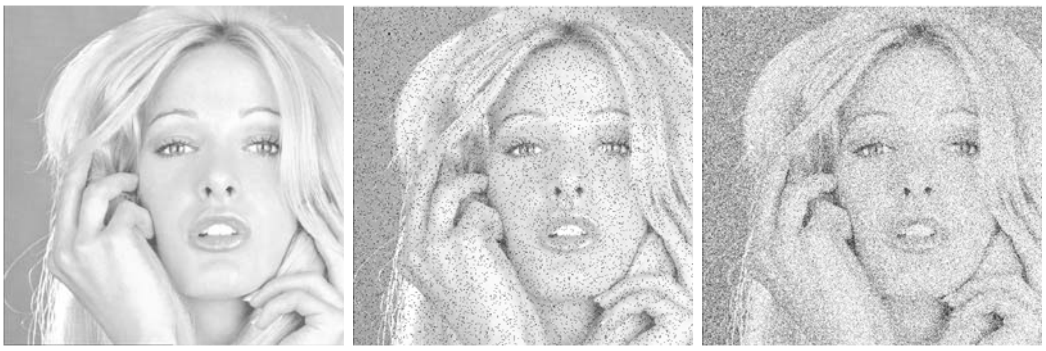
The inverse Legendre transform is used to reconstruct the input images by using the computed Legendre moments up to the maximum moment order Max. Then, the MSE is computed by using Equation (20) for the input images where the computed values are plotted against the moment order. In order to address the accuracy of the proposed method, two numerical experiments are conducted with real world binary and gray level images. In the first one, a binary image of Chinese letter of size 128×128 as displayed in Figure 4(a) is used. The standard gray level image of 'Blonde Woman' of size 128×128 as displayed in Figure 5(a) is used in the second numerical experiment.

The MSE for these two images are computed by using the proposed and the conventional ZOA method where the computed values are plotted against the moment order. Figures 6(a) and 7(a) display the MSE plotted curves of noise-free image of the binary Chinese letter and the noise-free image of 'Blonde Woman' respectively. These figures clearly show that, the MSE of the proposed method is decreases as the moment order increases and approaches the zero value while the MSE of the conventional ZOA method strongly increases as the moment order increases. The results of these numerical experiments ensure the accuracy of the proposed method.



(a) (b) (c)

Figure 4. Binary image: (a) noise-free image; (b) first noisy image; (c) second noisy image



(a) (b) (c)

Figure 5. Gray image: (a) noise-free image; (b) first noisy image; (c) second noisy image

Robustness against different kinds of image noise is another way to ensure the accuracy of the proposed method. The original images of the Chinese letter and the 'Blonde Woman' are contaminated with two kinds of popular image noise. The first one is the "salt & Peppers" noise while the white Gaussian noise represents the second kind. The "salt & Peppers" noise is added to the images by using the Matlab statement; A = imnoise (A, 'salt & pepper', S) where S = 0.08. The white Gaussian noise is added to images by using the Matlab statement; A = imnoise (A, 'gaussian', m, v) where m and v are the mean and the variance respectively.

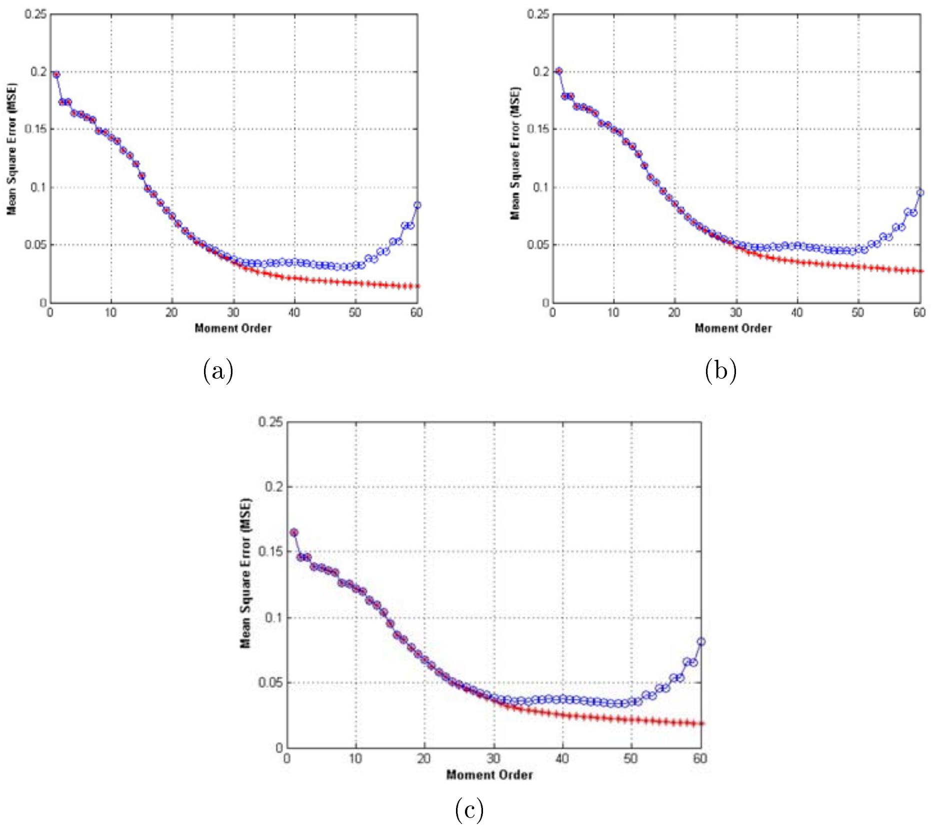


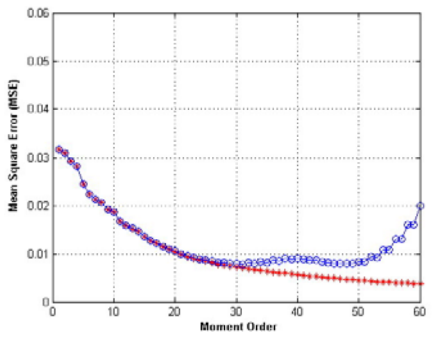
FIGURE 6. MSE of Binary image: (a) noise-free image; (b) first noisy image; (c) second noisy image

In the conducted numerical experiment, these parameters are $m = 0$ and $v = 0.05$. The contaminated images of the Chinese letter are displayed in Figures 4(b) and 4(c) while the contaminated images of the gray level image are displayed in Figures 5(b) and 5(c).

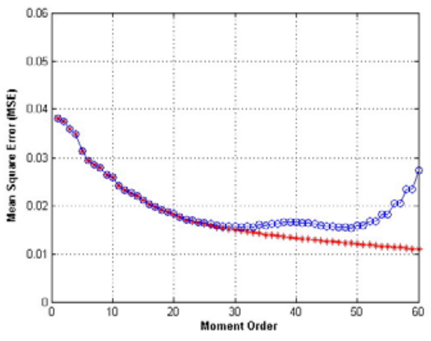
The Figures 6(b), 6(c), 7(b) and 7(c) display the MSE plotted the curves of contaminated binary Chinese letter image and the gray level image of 'Blonde Woman' respectively. These figures clearly show that, the MSE decreases as the moment order increases. Despite the fact that the gray-level images are more sensitive to the Gaussian white noise than the binary images, the values of MSE approach zero value as the moment order increases. This is an evidence of the accuracy of the proposed method.

4.2. **Complexity analysis.** The performance of the proposed method is compared with the other existing methods. The efficiency is an essential issue that must be addressed and proved. Theoretical complexity analysis is a very attractive aspect in the area of moment computation. This analysis addresses the theoretical proof for the efficiency of the proposed method. Such proof is not affected by any circumstances of conducting numerical experiments. In this section, a discussion of the complexity analysis is presented. The reduction in the computational complexity is the result of applying symmetry properties.

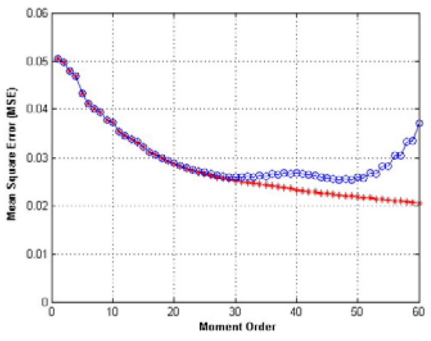
Based on the symmetry property, one octant will be used to compute the full set of 2D geometric moments. The computation process requires only two points. The first one is fall inside the first octant and has a similar seven points in the other seven octants. The second one is defined according to the third type of symmetry and has three similar points. Consequently, the total number of points in the first octant is equal to $[1 + 2 + 3 + \ldots + (N/2 - 1)] + N/2$. Therefore, the reduction in the computed points



(a)



(b)



(c)

FIGURE 7. MSE of gray level image: (a) noise-free image; (b) first noisy image; (c) second noisy image

TABLE 5. Reduction percentage (RP) of the proposed method: 2D case

2D Image size	Direct Method	Wee's Method [23]	Proposed Method	RP
64×64	4096	1024	528	87.1 %
128×128	16384	4096	2080	87.3 %
256×256	65536	16384	8256	87.4 %
512×512	262144	65536	32896	87.5 %
1024×1024	1048576	262144	131328	87.5 %

according to the symmetry property is defined as follows:

$$RP = \left(1 - \frac{N(N+2)}{8}\right) \times 100 \tag{23}$$

An explicit comparison is presented where the theoretical complexity analysis is performed for the proposed method and a recent existing method (Wee et al. [23]). The results of this theoretical complexity analysis are listed in Table 5. The obtained results clearly show the superiority of the proposed method over the other existing methods. The difference between the proposed method in 2D case and the method of Wee et al. [23] could be easily noticed. Wee and his co-authors achieved 75% reduction for squared images and 50% reduction for rectangular images. On the other side, the proposed method achieved 87% reduction for square image and 75% reduction for rectangular images.

TABLE 6. Reduction percentage (RP) of the proposed method: 3D case

3D Image size	Direct Method	Proposed Method	RP
$64 \times 64 \times 64$	262,144	16,896	93.55 %
$128 \times 128 \times 128$	2,097,152	133,120	93.65 %
$256 \times 256 \times 256$	16,777,216	1,056,768	93.70 %
$512 \times 512 \times 512$	134,217,728	8,421,376	93.72 %
$1024 \times 1024 \times 1024$	1,073,741,824	67,239,936	93.74 %

A similar theoretical complexity analysis is performed for 3D geometric moments. The reduction percentage based on the four types of symmetry is defined as follows:

$$RP = \left(1 - \frac{N^2(N+2)}{16}\right) \times 100 \tag{24}$$

To the best of the authors' knowledge, no previously published papers presented the idea of symmetry-based method in 3D case. Consequently, Table 6 shows an explicit comparison between the proposed method and the direct method. The obtained results clearly show that, the computational complexity of the proposed method is tremendously reduced where the percentage of the reduction exceeds 93%.

4.3. **Computational time.** Reduction in the computational time is an essential target especially with big size 2D and 3D images. In addition to theoretical complexity analysis, the elapsed CPU times of the conducted numerical experiments are used to measure the efficiency of the proposed method. Two numerical experiments are conducted. The first one is concerned with the 2D geometric moments while the second is concerned with 3D geometric moments. In the first experiment, a set of standard gray level images of size 512×512 are used. These images are displayed in Figure 8. In the second experiment, a 3D dataset of protein structure is used. All computational processes are performed by using a code designed with Matlab8 and operated on a Lenovo R400 Laptop.

The full set of 2D geometric moments is computed by using the proposed method and a group of the existing methods. These methods are the direct ZOA method represented by Equation (2) and the method of Wee et al. [23]. The elapsed CPU times are computed ten times for each gray level image and the averages elapsed CPU times are compared and plotted in Figure 9(a).

Similarly, the full of 3D geometric moments is computed by the proposed method where the elapsed CPU times are computed. The experiment is executed ten times where average CPU times are compared with the direct ZOA method. The results are graphically plotted

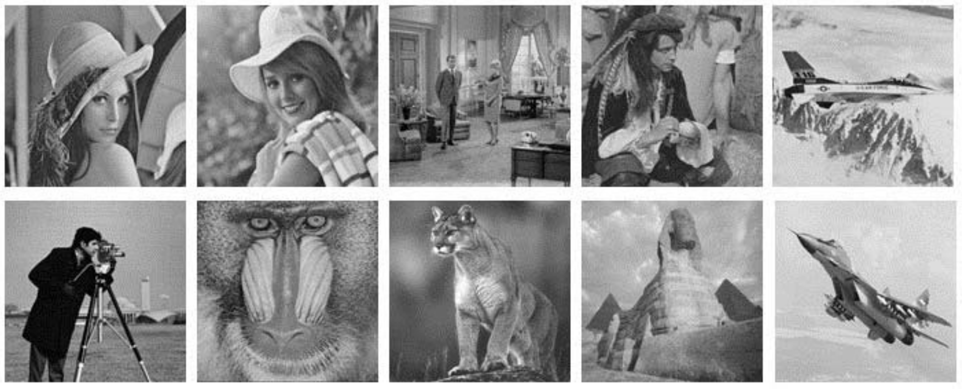


FIGURE 8. A set of standard gray images

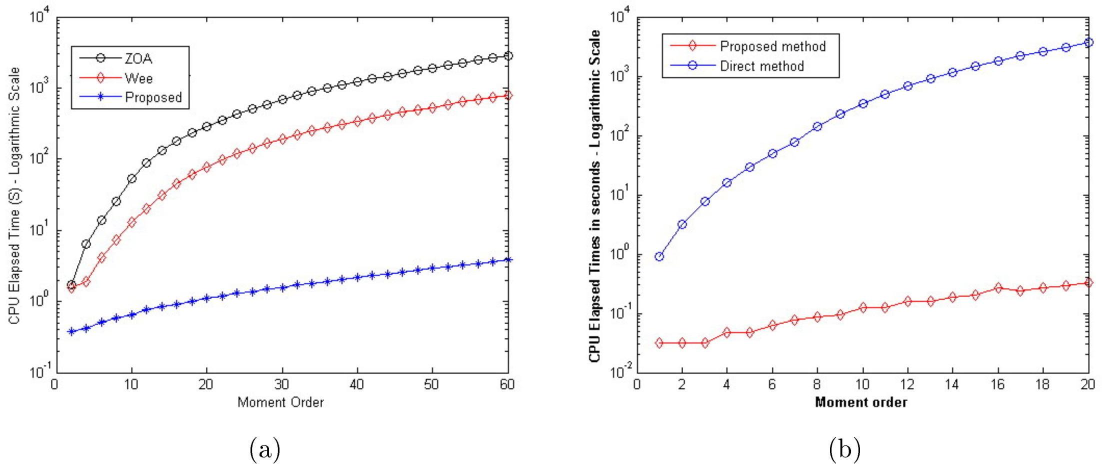


FIGURE 9. Elapsed CPU times: (a) 2D GMs; (b) 3D GMs

in Figure 9(b). It is clear that, the direct method in 2D and 3D cases is impractical. The proposed method outperformed the method of Wee [23] in 2D case. Generally, we could see that, the proposed method tremendously reduced the computational time which candidate it for highly efficient computation of big size images and the implementation in real time applications. The comparison ensures the superiority of the proposed method.

5. Conclusion.

This paper proposes a novel symmetry-based method for highly efficient computation of accurate 2D and 3D geometric moments for binary and gray level images/objects. In 2D case, three types of symmetry are applied where 87% of the computational demands are removed. The reduction percentage increased to 93% in the case of 3D geometric moments. The calculated values of geometric moments are very accurate where the double and triple integrations are analytically evaluated without any kind of approximation. Implementation of the proposed method to the families of continuous and discrete 2D and 3D orthogonal moments is promising and leads to the construction of very efficient and easily programmable generic library of functions. The conducted numerical experiments and theoretical complexity analysis confirm the efficiency of the proposed method.

Acknowledgment. The author acknowledge the financial support of the deanship of scientific research, Najran University (Grant No. NU24/2010) and the assistance of Prof. Richard Morris from the Computational and Systems Biology department, John Innes Centre, Norwich, UK. He provided the data set of protein structure used in the second numerical experiment.

REFERENCES

[1] M. K. Hu, Visual pattern recognition by moment invariants, *IRE Trans. Inf. Theory*, vol.8, no.1, pp.179-187, 1962.

[2] F. Sadjadi and E. L. Hall, Three-dimensional moment invariants, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol.2, no.2, pp.127-136, 1980.

[3] M. R. Teague, Image analysis via the general theory of moments, *J. Opt. Soc. Am.*, vol.70, no.8, pp.920-930, 1980.

[4] G. A. Anastassiou, Applications of geometric moment theory related to optimal portfolio management, *Computers & Mathematics with Applications*, vol.51, no.9-10, pp.1405-1430, 2006.

[5] M. Elad, A. Tal and S. Ar, Content based retrieval of VRML objects – An iterative and interactive approach, *Eurographics Multimedia Workshop*, pp.97-108, 2001.

[6] C.-H. Lo and H.-S. Don, 3-D moments forms: Their construction and application to object identification and positioning, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol.11, no.10, pp.1053-1064, 1989.

[7] J.-F. Mangin, F. Poupon, E. Duchesnay et al., Brain morphometry using 3-D moment invariants, *Medical Image Analysis*, vol.8, pp.187-196, 2004.

[8] R. D. Millán, L. Dempere-Marco, J. M. Pozo, J. R. Cebral and A. F. Frangi, Morphological characterization of intracranial aneurysms using 3-D moment invariants, *IEEE Trans. on Medical Imaging*, vol.26, no.9, pp.1270-1282, 2007.

[9] B. Ng, R. Abugharbieh, X. Huang and M. J. McKeown, Spatial characterization of fMRI activation maps using invariant 3-D moment descriptors, *IEEE Trans. on Medical Imaging*, vol.28, no.2, pp.261-268, 2009.

[10] B. Ng, R. Abugharbieh, X. Huang and M. J. McKeown, Characterizing fMRI activations within regions of interest (ROIs) using 3-D moment invariants, *Proc. of IEEE Workshop Mathematical Methods Biomed. Image Anal.*, New York, USA, 2006.

[11] M. Pan, J. Tang and X. Yang, An algorithm for medical image tilt correction using B-spline and moment invariants, *ICIC Express Letters*, vol.4, no.1, pp.57-63, 2010.

[12] S. D. Lin, J.-H. Lin and C.-C. Chiang, Combining scale invariant feature transform with principal component analysis in face recognition, *ICIC Express Letters*, vol.3, no.4, pp.927-932, 2009.

[13] J. Yang, Fingerprint matching using multiple sets of invariant moments, *ICIC Express Letters*, vol.5, no.1, pp.243-248, 2011.

[14] J. C. Yang and D. S. Park, Fingerprint verification based on invariant moment features and nonlinear BPNN, *International Journal of Control, Automation and Systems*, vol.6, no.6, pp.800-808, 2008.

[15] M. Novotni and R. Klein, Shape retrieval using 3D Zernike descriptors, *Computer-Aided Design*, vol.36, pp.1047-1062, 2004.

[16] R. Gurunathan, B. V. Emden, S. Panchanathan and S. Kumar, Identifying spatially similar gene expression patterns in early stage fruit fly embryo images: Binary feature versus invariant moment digital representations, *BMC Bioinformatics*, vol.5, pp.202-214, 2004.

[17] V. Vishwesh, R. C. Padmasini and K. Daisuke, Application of 3D Zernike descriptors to shape-based ligand similarity searching, *Journal of Cheminformatics*, vol.1, no.19, 2009.

[18] V. Vishwesh, D. Y. Yifeng, S. Lee and K. Daisuke, Protein-protein docking using region-based 3D Zernike descriptors, *BMC Bioinformatics*, vol.10, pp.407-428, 2009.

[19] I. M. Spiliotis and B. G. Mertzios, Real-time computation of two-dimensional moments on binary images using image block representation, *IEEE Trans. Image Processing*, vol.7, no.11, pp.1609-1615, 1998.

[20] G. A. Papakostas, E. G. Karakasis and D. E. Koulouriotis, Efficient and accurate computation of geometric moments on gray-scale images, *Pattern Recognition*, vol.41, no.6, pp.1895-1904, 2008.

[21] S. X. Liao and M. Pawlak, On image analysis by moments, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol.18, no.3, pp.254-266, 1996.

[22] K. M. Hosny, Exact and fast computation of geometric moments for gray level images, *Applied Mathematics and Computation*, vol.189, no.2, pp.1214-1222, 2007.

[23] C.-Y. Wee, R. Paramesran and R. Mukundan, Fast computation of geometric moments using a symmetric kernel, *Pattern Recognition*, vol.41, no.7, pp.2369-2380, 2008.

[24] L. Yang, F. Albregtsen and T. Taxt, Fast computation of 3-D geometric moments using a discrete divergence theorem and a generalization to higher dimensions, *Graph Models Image Processing*, vol.59, no.2, pp.97-108, 1997.

[25] R. Mukundan and K. R. Ramakrishnan, Moment functions in image analysis theory and applications, *World Scientific*, 1998.

[26] K. M. Hosny, Fast and accurate method for radial moment's computation, *Pattern Recognition Letters*, vol.31, no.2, pp.143-150, 2010.

[27] K. M. Hosny, Exact Legendre moment computation for gray level images, *Pattern Recognition*, vol.40, no.12, pp.3597-3605, 2007.

[28] G. A. Papakostas, E. G. Karakasis and D. E. Koulouriotis, Novel moment invariants for improved classification performance in computer vision applications, *Pattern Recognition*, vol.43, no.1, pp.58-68, 2010.