Int J Comput Vis
DOI 10.1007/s11263-009-0259-4

Implicit Moment Invariants

Jan Flusser · Jaroslav Kautsky · Filip Šroubek

Received: 14 August 2008 / Accepted: 26 May 2009

Abstract The use of traditional moment invariants in object recognition is limited to simple geometric transforms, such as rotation, scaling and affine transformation of the image. This paper introduces so-called implicit moment invariants. Implicit invariants measure the similarity between two images factorized by admissible image deformations. For many types of image deformations traditional invariants do not exist but implicit invariants can be used as features for object recognition. In the paper we present implicit moment invariants with respect to polynomial transform of spatial coordinates, describe their stable and efficient implementation by means of orthogonal moments, and demonstrate their performance in artificial as well as real experiments.

Keywords Invariants · Implicit invariants · Moments · Orthogonal polynomials · Polynomial image deformation

1 Introduction

Recognition of objects and patterns that are deformed in various ways has been a goal of much recent research. There are basically three major approaches to this problem—full search, image normalization, and invariant descriptors. In the full search approach we search the space of all possible image degradations, which means that the training set of each class should contain not only all class representatives but also all their rotated, scaled, and deformed versions. Although this approach can be successful for small deformations (Duda et al. 2001), in other cases it would lead to extreme time complexity and would be practically inapplicable. In the normalization approach, objects are transformed into a certain standard position before they enter the classifier. This could be very efficient in the classification stage but the object normalization usually requires to solve complex inverse problems which are often ill posed. The approach using invariant descriptors appears to be the most promising one and has been used extensively. Its basic idea is to describe the object by a set of features which are not sensitive to particular deformations and which provide enough discrimination power to distinguish among objects belonging to different classes.

In 2-D object recognition, various moment invariants have become classical descriptors during last forty years. No doubt they are one of the most important and most frequently used shape descriptors. Even if they suffer from some intrinsic limitations (the most important of which is their globalness, which prevents them from being used for recognition of occluded objects), they frequently serve as the "first-choice descriptors" and as a reference method for evaluation of the performance of other shape descriptors. Despite a tremendous effort and huge number of published papers, there are still open problems to be resolved.

All moment invariants ever studied may be called *explicit* invariants. From the mathematical point of view, an explicit invariant is a functional (let us denote it as E) acting on the space of image functions which does not change its value if the image f undergoes certain deformation τ from the set of admissible deformations, i.e. which satisfies the condition

J. Flusser · F. Šroubek (✉)
Institute of Information Theory and Automation of the ASCR,
Pod Vodárenskou věží 4, 182 08 Prague 8, Czech Republic
e-mail: sroubekf@utia.cas.cz

J. Flusser
e-mail: flusser@utia.cas.cz

J. Kautsky
The Flinders University of South Australia, Adelaide, Australia
e-mail: jarka@infoeng.flinders.edu.au

Published online: 17 June 2009

$E(f) = E(\tau(f))$ for any image f. Many systems of explicit moment invariants have been described which are invariant with respect to rotation, scaling, affine transform, contrast changes, and linear filtering, more details are given in the literature survey below. However, there are several classes of image deformations which occur frequently in practice but explicit moment invariants with respect to them are not known or have even been proved not to exist. Typical examples are cylindrical and spherical projections, quadratic transform, and other polynomial transforms of the image coordinates.

To overcome this, we propose in this paper so-called *implicit* invariants, which play a role of a distance or similarity measure between two objects independently of their deformations. Implicit invariant I is a functional defined on image pairs such that $I(f, \tau(f)) = I(f, f) = 0$ for any image f and deformation τ and $I(f, g) > 0$ for any f, g such that g is not a τ-deformed version of f. According to this definition, explicit invariants are just particular cases of implicit invariants. Clearly, if explicit invariant E exists, we can set $I(f, g) = |E(f) - E(g)|$. As we show later on in the paper, there are many types of image deformations where explicit moment invariants do not exist while implicit moment invariants do. If for instance the assumed deformation class is not closed with respect to composition (i.e. the deformations do not form a group) then explicit invariants to this class cannot exist in principle. In those cases, implicit invariants can be used as features for object recognition.

Unlike explicit invariants, implicit invariants do not provide description of a single image because they are always defined for a pair of images. This is why they cannot be used for shape encoding and reconstruction. Implicit invariants were designed as a tool for object recognition and matching. We consider $I(f, g)$ to be a "distance measure" (even if it does not exhibit all properties of a metric) between f and g factorized by τ and we can, for each database template g_i, calculate the value of $I(f, g_i)$ and then to classify f according to the minimum. It should be noted that the word "implicit" here has nothing common with implicit curves and implicit polynomials. It just expresses the fact that there is no explicit formula allowing to evaluate this invariant for a given image. Deformation τ may generally influence image graylevels and/or spatial coordinates but in this paper we consider spatial deformations only.

The paper is organized as follows. After a brief survey of moment invariants in Sect. 2 we define basic terms in Sect. 3. In Sect. 4, we put traditional explicit moment invariants into this context. The key idea how to construct implicit moment invariants along with an illustrative example is presented in Sect. 5. In Sects. 6–8, we propose a proper way how to work with polynomials and how to actually implement the invariants in 2-D to ensure numerical stability. The last section demonstrates the performance of the implicit moment invariants in experiments.

2 State-of-the-Art in Brief

The history of moment invariants began many years before the appearance of the first computers, in the 19th century under the framework of the group theory and of the theory of algebraic invariants. The theory of algebraic invariants was thoroughly studied by famous German mathematicians Gordan and Hilbert (Hilbert 1993) and in the 20th century it was further developed in Gurevich (1964) and Schur (1968), among others.

Moment invariants were firstly introduced to the pattern recognition community in 1962, where Hu (1962) employed the results of the theory of algebraic invariants and derived his seven famous invariants to rotation of 2-D objects. Since that time, numerous works have been devoted to various improvements and generalizations of Hu's invariants and also to its use in many application areas.

In Dudani et al. (1977) and Belkasim et al. (1991) an application to aircraft silhouette recognition is described. In Wong and Hall (1978), Goshtasby (1988) and Flusser and Suk (1994b) the authors employed moment invariants in template matching and registration of satellite images. Mukundan and Ramakrishnan (1996), Mukundan and Malik (1993) applied them to estimate the position and the attitude of the object in 3-D space. Additionally in Sluzek (1995) it was proposed to use local moment invariants in industrial quality inspection and many authors used moment invariants for character recognition (Belkasim et al. 1991; El-Khaly and Sid-Ahmed 1990; Tsirikolias and Mertzios 1993; Khotanzad and Hong 1990; Flusser and Suk 1994a). Maitra (1979) and Hupkens and de Clippeleir (1995) made them invariant also to contrast changes, Wang and Healey (1998) proposed illumination invariants particularly suitable for texture classification. Li (1992) and Wong et al. (1995) presented the systems of invariants up to the orders nine and five, respectively. Unfortunately, no one of them paid attention to mutual dependence/independence of the invariants. The invariant sets presented in their papers are algebraically dependent. Most recently, Flusser has proposed a method how to derive independent sets of invariants of any orders for general objects (Flusser 2000, 2002) and for objects having certain types of symmetry (Flusser and Suk 2006).

There is also a group of papers (Khotanzad and Hong 1990; Teague 1980; Wallin and Kubler 1995) that use *Zernike moments* to construct rotation invariants. Their motivation comes from the fact that Zernike polynomials are orthogonal on a unit circle and they can be evaluated by recurrent formulae which prevents them from numerical instability and overflow. However, Teague (1980) showed that Zernike invariants of 2nd and 3rd orders are equivalent to Hu's ones when expressing them in terms of geometric moments. He presented the invariants up to eighth order in explicit form but no general rule how to derive them is given.

Wallin and Kubler (1995) described an algorithm for a formation of moment invariants of any order. Since Teague (1980) as well as Wallin and Kubler (1995) were particularly interested in reconstruction abilities of the invariants, they did not pay much attention to the question of independence.

At the beginning of the 90s, several authors (Flusser and Suk 1993; Reiss 1991; Li 1992) contributed significantly to the theory of moment invariants by correcting the Fundamental Theorem (Hu 1962) and by deriving invariants to general affine transform. Later on, the Fundamental Theorem in N-dimensional form was published by Mamistvalov (1998) and the relationship between 2D affine moment invariants and graphs was discovered by Suk and Flusser (2004a).

Most recently, some authors dealt with moment invariants to projective transform without reaching applicable results. Gool et al. (1995) proved the nonexistence of finite projective moment invariants using the Lie group theory. Their work can be extended to prove the nonexistence of invariants, that would have a form of infinite series with each term equal to a finite product of moments of non-negative indices. Finally, Suk and Flusser (2004b) proved the existence of projective moment invariants in a form of infinite series containing moments with positive as well as negative indices. However, the convergence of these invariants has not been completely proved.

Several papers studied recognitive and reconstruction aspects, noise tolerance, discretization errors and other numerical properties of various kinds of moment invariants, both theoretically and experimentally (Pawlak 2006; Belkasim et al. 1991; Prokop and Reeves 1992; Teh and Chin 1988; Abu-Mostafa and Psaltis 1984; Liao and Pawlak 1996; Pawlak 1992; Bailey and Srinath 1996). Moment invariants were shown to be also a useful tool for geometric normalization of images (Abu-Mostafa and Psaltis 1985; Gruber and Hsu 1997). Large amount of effort has been spent to find effective algorithms for moment calculation (see Yang and Albregtsen 1996 for a survey).

A common denominator of all papers mentioned above is that they did not break the restriction to linear transformation of spatial coordinates of the image. Under general polynomial transformations, the moments change their orders which seemingly makes any construction of invariants impossible. In this paper, we extend the current theory of moment invariants. We study the behavior of moments under polynomial transformations and demonstrate that certain moments can be used for recognition of deformed object when considering implicit instead of explicit invariants.

A question of course arises if some other techniques different from moments can be used for recognition of objects under polynomial transformation. There have been many papers on "elastic matching" and "deformable templates" (see Kybic and Unser 2003 for an example and further references), which are powerful in recognition but usually slow. Most of them essentially perform an exhaustive search in the parameter space of functions approximating the deformation and looking for an extremum of some similarity or dissimilarity measure. Various splines as approximating functions and cross-correlation and mutual information as a similarity measure are popular choices. Even if the authors speed-up the search by pyramidal image representation and/or sophisticated optimization algorithms, these methods still perform relatively slow because they do not use any invariant representation.

In the literature, one can find many papers on "nonlinear" invariants (see Mundy and Zisserman 1992 for a survey and other references). Most of them are limited to projective transformation, which is the simplest nonlinear transform, and their main goal is to find local invariant descriptors allowing recognition of partially occluded objects (see for instance Weiss 1988; Pizlo and Rosenfeld 1992). Since this problem is different from the one we deal with in this paper, those methods cannot be (at least not easily) adopted for our purpose.

3 General Moments

Definition 1 Let $p_0, p_1, \ldots, p_{n-1}$ be some basis functions defined on a bounded $D \subset I\!R^N$ and let f be an image function having a finite integral. By a *general moment* of f we understand the functional

$$\mu_j(f) = \int_D f(x) p_j(x) dx.$$

If $N = 1$ and $p_j(x) = x^j$, we speak about *standard* moments.

Using a matrix notation we can write

$$\boldsymbol{p}(x) = \begin{pmatrix} p_0(x) \\ p_1(x) \\ \vdots \\ p_{n-1}(x) \end{pmatrix} \quad \text{and} \qquad\qquad (1)$$
$$\boldsymbol{\mu}(f) = \begin{pmatrix} \mu_0(f) \\ \mu_1(f) \\ \vdots \\ \mu_{n-1}(f) \end{pmatrix}.$$

Let $r : D \to \tilde{D}$ be a transformation of the domain D into $\tilde{D}$ and let $\tilde{f} : \tilde{D} \to I\!R$ be another image function which satisfies

$$\tilde{f}(r(x)) = f(x) \qquad (2)$$

for $x \in D$ and $f(\tilde{x}) = 0$ for $\tilde{x} \in \tilde{D} \setminus r(D)$. (This means that image $\tilde{f}$ is a spatially deformed version of f.)

We are interested in the relation between the moments $\boldsymbol{\mu}(f)$ and the moments

$$\tilde{\boldsymbol{\mu}}(\tilde{f}) = \int_{r(D)} \tilde{f}(\tilde{x})\tilde{\boldsymbol{p}}(\tilde{x})d\tilde{x} = \int_{\tilde{D}} \tilde{f}(\tilde{x})\tilde{\boldsymbol{p}}(\tilde{x})d\tilde{x}$$

of the transformed function with respect to some other $\tilde{n}$ basis functions

$$\tilde{\boldsymbol{p}}(\tilde{x}) = \begin{pmatrix}\tilde{p}_0(\tilde{x}) & \tilde{p}_1(\tilde{x}) & \dots & \tilde{p}_{\tilde{n}-1}(\tilde{x})\end{pmatrix}^T$$

defined on $\tilde{D}$. By substituting $\tilde{x} = r(x)$ into the definition of $\tilde{\boldsymbol{\mu}}(\tilde{f})$ and by using the composite function integration we obtain the following result.

Theorem 1 *Denote by $\mathcal{J}_r(x)$ the Jacobian of the transform function r. If*

$$\tilde{\boldsymbol{p}}(r(x))|\mathcal{J}_r(x)| = A\boldsymbol{p}(x) \tag{3}$$

for some $\tilde{n} \times n$ matrix A then

$$\tilde{\boldsymbol{\mu}} = A\boldsymbol{\mu}. \tag{4}$$

The power of this theorem depends on our ability to choose the basis functions so that we can, for a given transform r, express the left-hand side of (3) in terms of the basis functions $\boldsymbol{p}$ and thus construct the matrix A. This is always possible for a polynomial r by choosing polynomial bases $\boldsymbol{p}(x)$ and $\tilde{\boldsymbol{p}}(\tilde{x})$.

4 Explicit Moment Invariants

Let us assume that the transformation r depends on a finite number, say m, $m < \tilde{n}$, parameters $a = (a_1, \dots, a_m)$. Traditional explicit moment invariants with respect to r can be obtained in two steps.

1. Eliminate $a = (a_1, \dots, a_m)$ from the system (4). This leaves us $\tilde{n} - m$ equations which depend only on the two sets of general moments (and on the choice of the basis functions, of course). We call it a *reduced system*.
2. Re-write these equations equivalently in the form

$$q_j(\tilde{\boldsymbol{\mu}}) = q_j(\boldsymbol{\mu}), \quad j = 1, \dots, \tilde{n} - m \tag{5}$$

for some functions q_j. Then the explicit moment invariants are $E(f) = q_j(\boldsymbol{\mu}(f))$.

The way of deriving invariants outlined here is very transparent and is called in the literature "normalization approach". Although for various (mostly technical) reasons some authors used different techniques like tensor algebra, Lie groups, complex moments, graph theory, etc. to derive invariants (see refs. in Sect. 2), they all can be equivalently rewritten by means of normalization.

To demonstrate deriving explicit invariants in a simple case, consider a one-dimensional case with a linear transform $r(x) = ax + b$, $a > 0$, and choose the standard powers $p_j(x) = \tilde{p}_j(x) = x^j$, $j = 0, 1, 2, 3$ (here $n = \tilde{n}$ suffices). We have $\mathcal{J}_r(x) = a$ and

$$A = a\begin{pmatrix} 1 & 0 & 0 & 0 \\ b & a & 0 & 0 \\ b^2 & 2ba & a^2 & 0 \\ b^3 & 3b^2a & 3ba^2 & a^3 \end{pmatrix}.$$

Solving the first two equations (4) with this matrix A for a and b gives

$$a = \frac{\tilde{\mu}_0}{\mu_0} \qquad \text{and} \qquad b = \frac{\tilde{\mu}_1\mu_0^2 - \tilde{\mu}_0\mu_1^2}{\tilde{\mu}_0\mu_0^2}$$

and, after substituting these into the remaining two equations and some manipulation, we obtain two equations in the form (5) with

$$q_0(\boldsymbol{\mu}) = \frac{\mu_2\mu_0 - \mu_1^2}{\mu_0^4} \quad \text{and}$$

$$q_1(\boldsymbol{\mu}) = \frac{\mu_3\mu_0^2 - 3\mu_2\mu_1\mu_0 + 2\mu_1^3}{\mu_0^6}.$$

As another example consider, again for one-dimensional standard powers, the transform $r(x) = ax^2$. We now have $\mathcal{J}_r(x) = 2ax$ and (for $\tilde{n} = 2$ we need $n = 4$)

$$A = 2a\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & a \end{pmatrix}.$$

The first of equations (4) gives

$$a = \frac{\tilde{\mu}_0}{2\mu_1}$$

while the second equation rewrites, after substitution, as

$$\frac{\tilde{\mu}_1}{\tilde{\mu}_0^2} = \frac{\mu_3}{2\mu_1^2}$$

which cannot be rewritten in the required form (5). This shows that explicit invariants do not exist for this type of transform (and, similarly, for all other transformations which do not preserve the order of the moments and/or do not form a group).

5 Implicit Invariants—An Introductory Explanation

The second example of the previous section shows that, for some transforms, we may be able to eliminate the transform parameters in the quest of finding moment invariants, while the second step—finding the explicit forms q_j—may not be possible. Introducing implicit invariants can overcome this drawback.

Consider the equations obtained from the system (4) by eliminating the parameters specifying the transformation r. This reduced system is independent of the particular transformation. For classifying of an object, we traditionally compare the values of its descriptors (explicit moment invariants) with those of the database images, that is we look for such database image, that (5) are satisfied. However, it is equivalent to checking for which database image the above reduced system is satisfied.

So we can, in case we are not able to find explicit moment invariants in the form (5), use this system as a set of implicit invariants. More precisely, we arrange each equation of the system in the form having zero on its right-hand side. The magnitude of the left-hand side then represents one implicit invariant; in case of more than one equation the classification is performed according to the ℓ_2 norm of the vector of implicit invariants. In other words, the images are classified according to the minimum error with which the reduced system is satisfied.

We will demonstrate the above idea of implicit moment invariants on a 1-D example. Consider the transform

$$r(x) = x + ax^2,$$

where $a \in (0, 1/2]$, which maps interval $D = [-1, 1]$ on interval $\tilde{D} = [a-1, a+1]$. As $m = 1$ and we want to show two implicit invariants we need $\tilde{n} = 3$ and $n = 6$. The Jacobian is $\mathcal{J}_r(x) = 1 + 2ax$ and for standard powers for both $\boldsymbol{p}$ and $\tilde{\boldsymbol{p}}$ we would get

$$A = \begin{pmatrix} 1 & 2a & 0 & 0 & 0 & 0 \\ 0 & 1 & 3a & 2a^2 & 0 & 0 \\ 0 & 0 & 1 & 4a & 5a^2 & 2a^3 \end{pmatrix}.$$

However, now we have to evaluate the moments of the transformed signal over the domain $\tilde{D}$ which depends on the unknown parameter a. To resolve this problem, one can imagine a shifted power basis

$$\tilde{p}_j(\tilde{x}) = p_j(\tilde{x} - a), \quad j = 0, 1, \ldots, \tilde{n} - 1.$$

Then we have, after the shift of variable $\tilde{x} = \hat{x} + a$,

$$\tilde{\mu}_j(\tilde{f}) = \int_{a-1}^{a+1} \tilde{f}(\tilde{x}) \tilde{p}_j(\tilde{x}) d\tilde{x} = \int_{-1}^{1} \tilde{f}(\hat{x} + a) p_j(\hat{x}) d\hat{x}$$

which is now independent of a as $\hat{f}(\hat{x}) = \tilde{f}(\hat{x} + a)$ has domain $[-1, 1]$. Note that in order to calculate moments of transformed image $\tilde{f}$ we do not have to consider the shifted basis $\tilde{\boldsymbol{p}}$. For $\boldsymbol{p}$ as standard powers and $\tilde{\boldsymbol{p}}$ as defined above, we obtain a different transform matrix

$$A = \begin{pmatrix} 1 & 2a & 0 & 0 & 0 & 0 \\ -a & 1 - 2a^2 & 3a & 2a^2 & 0 & 0 \\ a^2 & 2a(a^2 - 1) & 1 - 6a^2 & 4a(1 - a^2) & 5a^2 & 2a^3 \end{pmatrix}.$$

Another way to resolve this problem without imagining different power basis is by assuming such $r(x)$, which maps interval D on itself. Both approaches are equivalent and they do not restrict applicability of implicit invariants. To better understand this, consider classification tasks in practice. We usually segment objects we want to classify from images. Without loss of generality one can assume that the segmented objects are defined on the same domain, which implies that the transform inscribes $\tilde{D}$ in D.

The first of equations (4) gives

$$a = \frac{\tilde{\mu}_0 - \mu_0}{2\mu_1}$$

while the two reduced equations rewrite, after substitution, as

$$\begin{aligned} 2\mu_1^2(\tilde{\mu}_1 - \mu_1) &= \mu_1(3\mu_2 - \tilde{\mu}_0)(\tilde{\mu}_0 - \mu_0) \\ &\quad + \mu_3(\tilde{\mu}_0 - \mu_0)^2, \\ 4\mu_1^3(\tilde{\mu}_2 - \mu_2) &= 4\mu_1^2(2\mu_3 - \mu_1)(\tilde{\mu}_0 - \mu_0) \\ &\quad + \mu_1(5\mu_4 + \tilde{\mu}_0 - 6\mu_2)(\tilde{\mu}_0 - \mu_0)^2 \\ &\quad + (\mu_5 - 2\mu_3)(\tilde{\mu}_0 - \mu_0)^3. \end{aligned} \tag{6}$$

Note that parameter a was eliminated and is not present in both implicit invariants. Thanks to this, the method does not require its knowledge.

We want to show how these two implicit invariants can identify a 1-D signal after such quadratic transformation. As testing data (database templates) we use the rows of the 256 × 256 Lena image scaled to $[-1, 1]$. As the query signal we use the row No. 135, which we have transformed by $r(x) = x + ax^2$ for three different values of the parameter ($a = 0.125$, $a = 0.175$, and $a = 0.225$, respectively). In Table 1 we list the moments of selected rows of the image. In Table 2 we show the moments of three modifications of the query signal.

We now substitute each of the three sets in Table 2 and each of the 11 sets in Table 1 into (6) and look which combinations conform to these equations best. As a measure we take the relative difference between the left and right hand sides, that is

$$\rho_1(\tilde{\boldsymbol{\mu}}, \boldsymbol{\mu}) = 1 - \frac{\mu_1(3\mu_2 - \tilde{\mu}_0)(\tilde{\mu}_0 - \mu_0) + \mu_3(\tilde{\mu}_0 - \mu_0)^2}{2\mu_1^2(\tilde{\mu}_1 - \mu_1)},$$

$$\rho_2(\tilde{\boldsymbol{\mu}}, \boldsymbol{\mu}) = 1 - \frac{(\mu_5 - 2\mu_3)(\tilde{\mu}_0 - \mu_0)^3}{4\mu_1^3(\tilde{\mu}_2 - \mu_2)}$$

Table 1 Moments of selected rows of the original Lena image

Row number	Moments using standard powers on interval $[-1, 1]$					
	μ_0	μ_1	μ_2	μ_3	μ_4	μ_5
104	194.224	11.156	67.909	7.494	42.759	6.611
120	194.561	16.232	66.559	8.044	41.957	6.293
128	181.298	12.879	66.678	8.329	42.255	6.585
132	167.522	10.819	64.851	8.379	40.973	6.768
134	169.773	11.710	64.118	7.768	40.382	6.050
135	170.886	12.324	63.965	7.729	40.340	5.923
136	175.122	13.120	64.515	8.074	40.741	6.209
138	186.388	14.541	67.174	9.485	42.595	7.663
142	190.122	18.626	69.841	12.645	45.045	10.365
150	193.918	28.174	71.122	18.305	46.221	14.612
166	196.741	32.371	76.960	22.021	50.030	17.326

$$- \frac{4\mu_1(2\mu_3 - \mu_1)(\tilde{\mu}_0 - \mu_0)}{4\mu_1^2(\tilde{\mu}_2 - \mu_2)} + \frac{(5\mu_4 + \tilde{\mu}_0 - 6\mu_2)(\tilde{\mu}_0 - \mu_0)^2}{4\mu_1^2(\tilde{\mu}_2 - \mu_2)}.$$

The results in the form of relative differences in the first and second implicit moment invariants of the transformed query row against selected row in the data set are summarized in Table 3. We have deliberately chosen both close and distant rows in the image. We observe that each of the implicit invariants, even on its own, perfectly identifies the query row independently of the steepness of the quadratic transformation.

Once the query signal has been recognized, we can—just for illustration, this is not required for recognition—estimate the deformation parameter as

$$\hat{a} = \frac{\tilde{\mu}_0 - \mu_0}{2\mu_1}$$

which gives $\hat{a} = (0.1242, 0.1761, 0.2310)$. When calculating the relative errors $|\hat{a} - a|/a$, we get $(0.0064, 0.0063, 0.0267)$, which proves a reasonable accuracy.

The aim of this illustrative experiment was to show how to construct implicit moment invariants and how they can be used for signal identification. Moreover, we showed that even in such a simple case of the transform and the basis functions, certain care of the signal support has to be taken.

Table 2 Moments of the query signal after quadratic transformation with three different values of coefficient a

a	$\tilde{\mu}_0$	$\tilde{\mu}_1$	$\tilde{\mu}_2$
0.125	173.947	14.768	64.592
0.175	175.227	15.762	64.724
0.225	176.580	16.529	64.941

6 Constructing Matrix A in 1-D

In the examples above it was straightforward to derive the transform matrix A for simple transformations r and small number of moments. For reasons of programming complexity and numerical stability, this intuitive approach cannot be used for higher-order polynomial transform r and/or for more invariants. In this section we present an algorithm to obtain matrix A for any polynomial transform r and any polynomial bases $\boldsymbol{p}$ and $\tilde{\boldsymbol{p}}$.

To get numerically stable method it is important to use suitable polynomial bases, such as orthogonal polynomials, without using their expansions into standard (monomial) powers. Our implementation is based on the representation of polynomial bases by matrices with special structure (see Golub and Kautsky 1983; Kautsky and Golub 1983 for details). We present only the main points of the process here.

Theorem 2 *Polynomials $p_j(t)$ of exact degree j, $j = 0, 1, \dots, n$, satisfy*

$$t\,\boldsymbol{p}(t) = J\,\boldsymbol{p}(t) + \beta_n p_n(t)\boldsymbol{e}_n, \tag{7}$$

where $\boldsymbol{e}_n$ is the n-th column of the identity matrix, $\beta_n \neq 0$ and $J = \{\alpha_{j,k}\}$ is an $n \times n$ matrix such that $\alpha_{j,j+1} \neq 0$ and $\alpha_{j,k} = 0$ for $k > j + 1$; such matrix is called a proper lower Hessenberg matrix. On the other hand, given such a matrix, non-zero p_0 and β_n, the polynomial bases is fully determined by (7).

This Theorem expresses in matrix form the consequence of the simple fact that $tp_j(t)$, being a polynomial of degree $j + 1$, is a linear combination of polynomials $p_0, \dots, p_{j+1}$. The construction of the polynomial bases from p_0 and J is by the recurrence

Table 3 Relative differences in the first and second implicit moment invariants of the transformed query row against selected row in the data set

Row	Query row: 135, x^2 coefficient:					
number	$a = 0.125$		$a = 0.175$		$a = 0.225$	
	ρ_1	ρ_2	ρ_1	ρ_2	ρ_1	ρ_2
104	1.239	0.114	1.142	0.098	1.057	0.085
120	0.568	−0.013	0.532	−0.016	0.493	−0.017
128	0.540	0.013	0.495	0.005	0.447	−0.001
132	−0.247	−0.036	−0.258	−0.036	−0.287	−0.032
134	−0.048	−0.008	−0.049	−0.008	−0.065	−0.006
135	−0.002	0.000	0.001	−0.000	−0.010	0.001
136	0.169	0.006	0.163	0.003	0.146	0.002
138	0.579	0.037	0.540	0.029	0.499	0.022
142	0.462	0.029	0.432	0.020	0.395	0.012
150	−0.273	0.003	−0.240	−0.003	−0.229	−0.007
166	−0.204	−0.028	−0.189	−0.037	−0.194	−0.046

$$p_j(t) = \frac{1}{\alpha_{j,j+1}}\left(t p_{j-1}(t) - \sum_{k=1}^{j} \alpha_{j,k} p_{k-1}(t)\right),$$

$$j = 1, \ldots, n-1$$

and similarly for $p_n(t)$. Alternatively, we can solve the system

$$(tI - J)\boldsymbol{x} = \boldsymbol{e}_n$$

from which $\boldsymbol{p}(t) = p_0 \boldsymbol{x}/x_1$ which may be numerically more stable, particularly if t is not close to any eigenvalue of J. Once the values of $\boldsymbol{p}(t)$ are available it is straightforward to approximate the generalized moments $\boldsymbol{\mu}(f)$ by a suitable quadrature formula.

We use matrix J to represent our polynomial base; in fact, all computations with polynomials can be done efficiently and stably using this matrix only. We can, for example, calculate the values of $\boldsymbol{p}(t)$ for any given t by forward substitution in (7) using the known value of p_0.

There are many interesting properties which can be derived from (7). For the purpose of this paper it is important to mention just the following generalization of (7).

Theorem 3 *Let $s(t) = \sum_{k=0}^{d} \sigma_k t^k$ be a polynomial of degree d. Then*

$$s(t)\boldsymbol{p}(t) = s(J)\boldsymbol{p}(t) + \beta_n p_n(t) \sum_{k=1}^{d} s_k(t) J^{k-1} \boldsymbol{e}_n, \tag{8}$$

where we denoted

$$s_k(t) = \sum_{j=k}^{d} \sigma_j t^{j-k}.$$

For monomial $s(t) = t^d$ the proof is by induction from which the general case follows by combining the relevant terms.

The simplest example of a polynomial base of increasing exact degree is that of standard powers $p_j(t) = t^j$ when $J = S = \{\delta_{j,k-1}\}$, the shift matrix.

For numerical stability we prefer to work with orthogonal polynomials. In that case matrix J is tri-diagonal; a useful example is $J = S + S^T$ which is the recurrence matrix for 2nd Chebyshev polynomials on the interval $[-2, 2]$ (orthogonal with respect to the weight function $\sqrt{4 - t^2}$).

Matrix A, as introduced in the Theorem 1, represents the relation between the moments of the original and transformed signals. We will now assume that we use polynomial bases $\boldsymbol{p}$ and $\tilde{\boldsymbol{p}}$ of degrees $n-1$ and $\tilde{n}-1$ determined by recurrence matrices J and $\tilde{J}$, respectively. Thus, besides (7), we also have

$$\tilde{t}\tilde{\boldsymbol{p}}(\tilde{t}) = \tilde{J}\tilde{\boldsymbol{p}}(\tilde{t}) + \tilde{\beta}_{\tilde{n}} \tilde{p}_{\tilde{n}}(\tilde{t}) \boldsymbol{e}_{\tilde{n}}. \tag{9}$$

It is important to show that matrix A can be constructed from the available information, that is from the transform r and from the matrices J and $\tilde{J}$ only. Theorem 3 is applied to $r(t)$ and also to $r'(t)$ to obtain relations for matrices A and B, satisfying $\tilde{p}(r(x)) = Bp(x)$, from which the conclusions follow by comparing terms with equal powers.

Theorem 4 *Let r be a polynomial of degree d mapping interval $D \subset I\!R$ onto interval $\tilde{D}$. Let p_0, J, β_n and $\tilde{p}_0$, $\tilde{J}$, $\tilde{\beta}_{\tilde{n}}$ represent polynomial bases on D and on $\tilde{D}$, respectively, where the degrees satisfy*

$$n = \tilde{n}d. \tag{10}$$

Then the matrix A for which $\tilde{\boldsymbol{p}}(r(x))r'(x) = A\boldsymbol{p}(x)$ (see (3) in Theorem 1) is given by

$$A = Br'(J) \tag{11}$$

where the matrix B satisfies

$$\begin{aligned} \boldsymbol{e}_1^T B &= \frac{\tilde{p}_0}{p_0} \boldsymbol{e}_1^T, \\ B r(J) &= \tilde{J} B + \boldsymbol{e}_{\tilde{n}} \boldsymbol{b}^T \end{aligned} \tag{12}$$

from which the rows of matrix B, and, if desired, the vector $\boldsymbol{b}$, can be calculated by a recurrence.

We have thus given an explicit construction of the matrix A in terms of a polynomial transform function $r(t)$ and of the matrices J and $\tilde{J}$ representing chosen polynomial bases for the original and transformed domains. We have also shown how to calculate moments of the original and transformed function with respect to such bases, again using only the matrices representing the polynomial bases. Using symmetric matrices J and $\tilde{J}$ with spectra covering the domains greatly enhances numerical stability of all calculations.

For proofs of Theorems 2, 3, and 4 see Appendix.

7 Constructing Matrix A in 2-D

The approach described in the previous section for 1-D domains can be extended to the 2-D cases. For brevity, we will point out only the main steps of one such extension.

We will assume that the two-dimensional domain D is a rectangle and we use product polynomials

$$p_{j,k}(x, y) = p_j(x) q_k(y)$$

where p_j and q_k are univariate polynomials of exact degree j and k, respectively.

We want to order the two dimensional polynomial basis by increasing degree

$$\boldsymbol{p}(x, y) = \begin{pmatrix} \boldsymbol{\pi}_0(x, y) & \boldsymbol{\pi}_1(x, y) & \dots & \boldsymbol{\pi}_\lambda(x, y) \end{pmatrix}^T$$

where $\boldsymbol{\pi}_l(x, y)$ is a vector of $p_{j,k}(x, y)$ of the same degree l. Two popular choices what to consider as degree of a multivariate polynomial are $l = j + k$ or $l = \max(j, k)$ and there are good reasons to use either of them; we choose the former so that, ordering the product polynomials by decreasing degree in x,

$$\boldsymbol{\pi}_l(x, y) = \begin{pmatrix} p_{l,0}(x, y) & p_{l-1,1}(x, y) & \dots & p_{0,l}(x, y) \end{pmatrix}^T.$$

The length of vector $\boldsymbol{p}(x, y)$ is

$$n = 1 + 2 + \cdots + \lambda - 1 = \frac{1}{2} \lambda(\lambda + 1).$$

As in one dimension there exist $n \times n$ matrices J_x and J_y such that

$$\begin{aligned} x \boldsymbol{p}(x, y) &= J_x \boldsymbol{p}(x, y) + \mathcal{T}, \\ y \boldsymbol{p}(x, y) &= J_y \boldsymbol{p}(x, y) + \mathcal{T}, \end{aligned} \tag{13}$$

where we use (and will use) $\mathcal{T}$ to denote a generic "tail", that is a matrix with non-zero elements involving higher degree polynomials only in last few rows.

Using (13) we derive, as in (8),

$$s(x, y) \boldsymbol{p}(x, y) = s(J_x, J_y) \boldsymbol{p}(x, y) + \mathcal{T} \tag{14}$$

for a (low degree) 2-D polynomial function $s(x, y)$. Note that terms containing both x and y in the polynomial $s(x, y)$ give an ambiguity in defining $s(J_x, J_y)$; due to the above mentioned structure of J_x and J_y the difference due to commuting these matrices in products can, however, be hidden in the tail $\mathcal{T}$.

We now want to apply this methodology to transformed 2-D images. As in Sect. 6 we need polynomial bases for both domains D ($\boldsymbol{p}(x, y)$) and $\tilde{D}$ ($\tilde{\boldsymbol{p}}(\tilde{x}, \tilde{y})$).

The transform is now given by a $D \to I\!R^2$ function

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} = \boldsymbol{r}(x, y) = \begin{pmatrix} r_x(x, y) \\ r_y(x, y) \end{pmatrix}$$

with (see Theorem 1)

$$\mathcal{J}_r(x, y) = \det \begin{pmatrix} \frac{\partial r_x(x,y)}{\partial x} & \frac{\partial r_x(x,y)}{\partial y} \\ \frac{\partial r_y(x,y)}{\partial x} & \frac{\partial r_y(x,y)}{\partial y} \end{pmatrix}.$$

Applying (14) to $s(x, y) = \mathcal{J}_r(x, y)$ shows that matrix A satisfying (3) will have the form

$$A = B \mathcal{J}_r(J_x, J_y)$$

where, using (14) again for both components of $\boldsymbol{r}(x, y)$, B must satisfy

$$\begin{aligned} B r_x(J_x, J_y) \boldsymbol{p}(x, y) &= \tilde{J}_x B \boldsymbol{p}(x, y) + \mathcal{T}, \\ B r_y(J_x, J_y) \boldsymbol{p}(x, y) &= \tilde{J}_y B \boldsymbol{p}(x, y) + \mathcal{T} \end{aligned}$$

from which we conclude that

$$\begin{aligned} B r_x(J_x, J_y) \ &= \tilde{J}_x B, \\ B r_y(J_x, J_y)) &= \tilde{J}_y B. \end{aligned} \tag{15}$$

This is again sufficient to determine B row by row.

8 Implementation of the Implicit Invariants

Depending on r, the elimination of the m parameters of the transformation function from the $\tilde{n}$ equations of (4) to obtain a parameter-free reduced system may require numerical solving of nonlinear equations. This may be undesirable or impossible. Even the simple transform r used in the experimental section would lead to cubic equations in terms of its parameters. Obtaining a neat reduced system may be very difficult. Furthermore, even if successful, we create an unbalanced method—we have demanded some of the equations in (4) to hold exactly and use the accuracy in the resulting system as a matching criterion to find the transformed image. We therefore propose another implementation of the implicit invariants. Instead of eliminating the parameters, we calculate the "uniform best fit" from all equations in (4). For a given set of values of the moments $\boldsymbol{\mu}$ and $\tilde{\boldsymbol{\mu}}$, we find values of the m parameters to satisfy (4) as best as possible in ℓ_2 norm; the error of this fit then becomes the value of the respective implicit invariant.

Our actual implementation of the recognition by implicit invariants can be described as follows.

1. Given is a library (database) of images $g_j(x,y)$, $j = 1,\dots,L$, and a deformed image $\tilde{f}(\tilde{x},\tilde{y})$ which is assumed to have been obtained by a transform of a known polynomial form $r(x,y,\boldsymbol{a})$ with unknown values of m parameters $\boldsymbol{a}$.
2. Choose the appropriate domains, polynomial bases $\boldsymbol{p}$ and $\tilde{\boldsymbol{p}}$, and the recurrence matrices J_x, J_y, $\tilde{J}_x$ and $\tilde{J}_y$ for evaluation of the polynomials.
3. Derive a program to evaluate the matrix $A(\boldsymbol{a})$. This critical error-prone step is performed by a symbolic algorithmic procedure which produces the program used then in numerical calculations. This step is performed only once for the given task. (It has to be repeated only if we change the polynomial bases or the form of transform $r(x,y,\boldsymbol{a})$, which basically means only if we move to another application.)
4. Calculate the moments $\boldsymbol{\mu}(g_j)$ of all library images $g_j(x,y)$.
5. Calculate the moments $\tilde{\boldsymbol{\mu}}(\tilde{f})$ of the deformed image $\tilde{f}(\tilde{x},\tilde{y})$.
6. For all $j = 1,\dots,L$ calculate, using an optimizer, the values of the implicit invariant

   $$I(\tilde{f},g_j) = \min_{\boldsymbol{a}} \|\tilde{\boldsymbol{\mu}}(\tilde{f}) - A(\boldsymbol{a})\boldsymbol{\mu}(g_j)\| \tag{16}$$

   and denote

   $$M = \min_j I(\tilde{f},g_j).$$

   The norm used here should be weighted, for example relatively to the components corresponding to the same degree.
7. The identified image is g_k for which $I(\tilde{f},g_k) = M$. The ratios

   $$\frac{I(\tilde{f},g_j)}{I(\tilde{f},g_k)}, \qquad j \neq k,$$

   where $I(\tilde{f},g_j)$ is the second minimum, may be used as confidence measures of the identification. We may accept only decisions the confidence of which exceeds some threshold.

9 Numerical Experiments

As we have shown earlier, the implicit moment invariants can be constructed for a very broad class of image transforms including all polynomial transforms. Here we will demonstrate the implementation and the power of the method on images transformed by the following function

$$\begin{pmatrix}\tilde{x}\\ \tilde{y}\end{pmatrix} = \boldsymbol{r}(x,y) = \begin{pmatrix} ax+by+c(ax+by)^2 \\ -bx+ay\end{pmatrix}, \tag{17}$$

that is, a rotation with scaling (parameters a and b) followed by a quadratic deformation in the $\tilde{x}$ direction (parameter c). We have chosen this particular transform for our tests for the following reasons:

- It is general enough to approximate many real-life situations, for instance deformations caused by the fact that the photographed object was drawn/printed on a cylindrical or conical surface like a bottle, a can or a cornet.
- It is sometimes used by web designers to warp images in order to reach desirable visual effect. Very often this is an unauthorized act violating the copyright. It is important for the copyright owners to have a tool for identifying such images.
- Explicit invariants to this kind of transforms cannot exist because they do not preserve the moment orders and do not form a group.

This transformation maps $D = [-2,2]^2$ into $\tilde{D} = [-\sigma(1-c\sigma),\sigma(1+c\sigma)] \times [-\sigma,\sigma]$ where we denoted $\sigma = 2(|a|+|b|)$ the scaling factor due to the rotation. We need to restrict $|c| \le \frac{1}{2\sigma}$ to have a one-to-one transform.

To illustrate the performance of the method, we carried out the following three sets of experiments. In the first set, we evaluated robustness of the implicit invariants with respect to noise and proved experimentally their invariance to image rotation and quadratic warping. This experiment was done on artificial data. The second experimental set was aimed to demonstrate that the implicit invariants can be used as a shape similarity measure of real objects and that this technique can be used for recognition of distorted images. This test was done on a standard benchmark database of Amsterdam Library of Object Images (ALOI) (Geusebroek et al. 2005). The last experimental set was done on real images taken in a lab. It illustrates good performance and high recognition power of the implicit invariants even in the case where theoretical assumptions about the degradation are not fulfilled.

9.1 Invariance and Robustness

In order to demonstrate numerical behavior of the proposed method, we conducted an experiment on standard Lena image, which was artificially deformed and corrupted by additive noise.

First, we applied the geometric transform in (17) on original Lena. We did not introduce any scaling, so the transform then became a two-parameter one fully determined by rotation angle (parameters a and b in the original transform constrained by $a^2+b^2=1$) and quadratic deformation (parameter c). To have one-to-one mapping, c must be restricted to the interval $|c| \leq 1/(2\sigma)$, where $\sigma = 2(|a|+|b|)$. For the sake of simplicity, we used a normalized parameter $q = 2\sigma c$, that lies always in the interval $(-1, 1)$, in the following discussion. The range of parameter values we used in the test was from $-40°$ to $40°$ with a 4° step for rotation angle and from -1 to 1 with a 0.1 step for q (see Fig. 1 for an example of a deformed image). For each deformed image and the original Lena, we calculated the values of six implicit invariants according to (16). This was implemented as a numeric optimization in the parameter space. In Fig. 2(a), we plotted ℓ_2 norm of the vector of implicit invariants. One can observe that it is more or less constant and very close to zero, irrespective of the degree of geometrical deformation. (The norm is slightly bigger when the rotation angle approaches ±40° because of resampling errors.) This illustrates a perfect invariance w.r.t. both rotation and quadratic deformation.

Then we performed a similar experiment. We fixed the rotation angle to zero and added white noise n to the transformed images $\tilde{f}$, where the noise level was measured in dB as a signal to noise ratio $\mathrm{SNR} = 10\log(\mathrm{var}(\tilde{f})/\mathrm{var}(n))$. For each deformed image and each noise level, ten realizations of noise were generated. Figure 2(b) shows again the ℓ_2 norm of the vector of implicit invariants for quadratic deformation q between ±1 and SNR from 50 (low noise) to 0 (high noise) dB. Like in the previous case, the graph is very flat. Up to 10 dB no disturbing slope is visible. Only for severely corrupted data with SNR around 0 dB (the noise variance is equal to the image variance) the norm starts to increase. This experiment proved high noise robustness of the implicit invariants, which is a direct consequence of the robustness of moments. It should be, however, noted that the vulnerability of the moments increases slowly as the moment order increases and the same is true for implicit invariants as well.

9.2 Classification Experiment

The aim of the second experiment was to test the discriminative power of the implicit invariants. We took 100 images from the commonly used benchmark database ALOI and deformed each of them by the warping model (17) (see Fig. 3 for some examples). The coefficients of the deformations were generated randomly; q from the interval $(-1, 1)$ and the rotation angle from $(-40°, 40°)$, both with uniform distribution. Each deformed image was then classified against the undistorted database by two different methods: by implicit invariants according to minimal norm and by affine moment invariants (AMI) proposed by Suk and Flusser (2004a) using the minimum-distance rule. In both cases, six invariants were used. We picked the AMI's for a comparison because they are similar to the new technique in their nature (both of them are based on moments) and because they are a traditional, well-established reference method in pattern recognition.

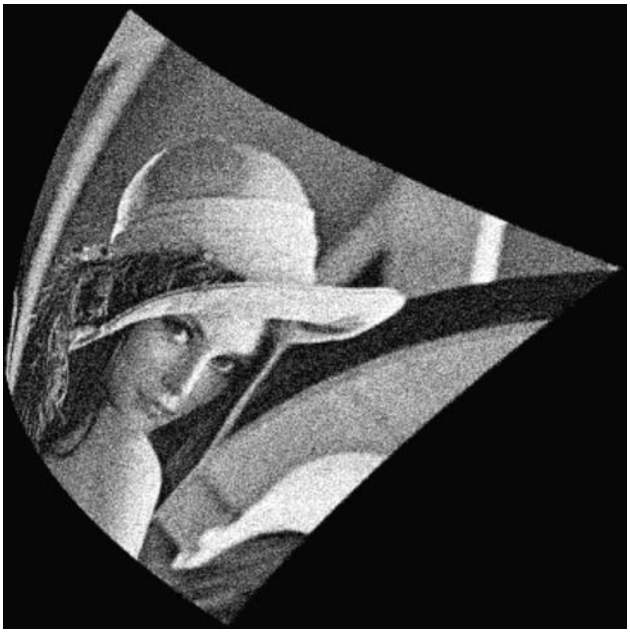

Fig. 1 One of the deformed and noisy images used in the experiment

We run the whole experiment several times with different deformation parameters. In each run the recognition rate we achieved was 99 or 100% for the implicit invariants and from 34 to 40% for the AMI's. These results illustrate two important facts. First, the implicit invariants can serve as an efficient tool for object recognition in case when the object deformation corresponds to the assumed model. Secondly, in case of nonlinear distortions the implicit invariants significantly outperform the affine moment invariants, which corresponds to our theoretical expectation—if q is close to zero, the nonlinear term is negligible, the distortion can be approximated by affine transform and the AMI's have a reasonable chance to recognize the object correctly. However, for larger q such an approximation is very rough and the AMI's naturally fail.

In case when only rotation is present, both methods should be equivalent. To prove this, we run the experiment once again but we fixed $q = 0$. Then the recognition rate of both methods was 100% as expected.

9.3 Character Recognition on Real Images

The last two experiments illustrate the applicability of the implicit invariants to classification of real images in situa-

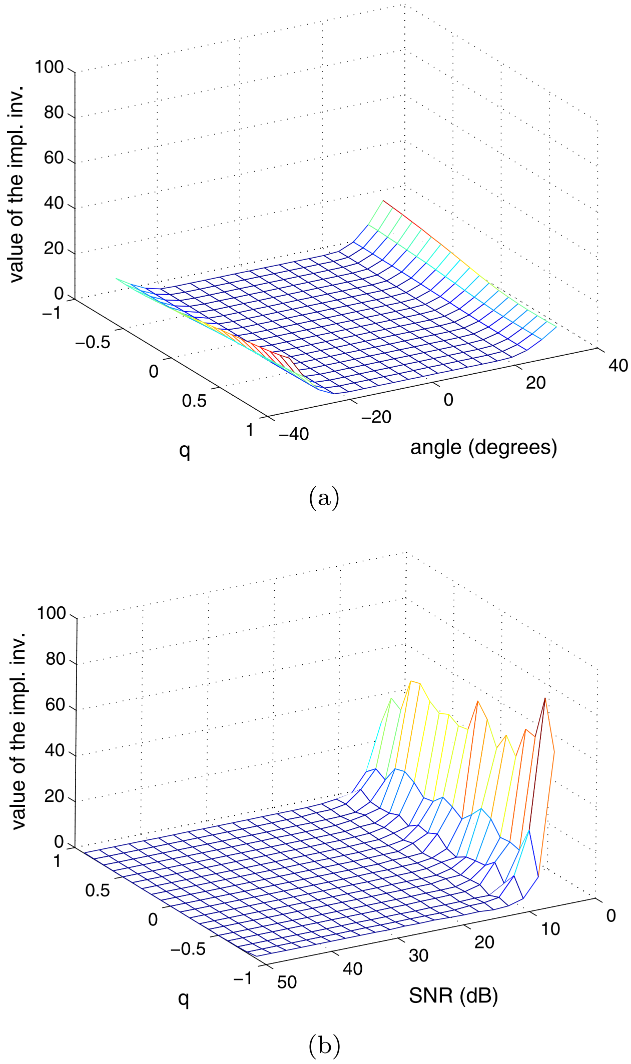

Fig. 2 Properties of the implicit invariants: (**a**) dependence of the implicit invariants on rotation angle and quadratic warping coefficient q; (**b**) dependence of the implicit invariants on the noise level (SNR) and quadratic warping coefficient q. Note that both graphs are almost constant and close to zero

tions where the image deformation is not known and, moreover, is not exactly polynomial. Such a situation occurs in practice when we want to recognize for instance letters or digits which are painted/printed on curved surfaces like balls, cans, bottles, etc.

With a standard digital camera (Olympus C-5050), we took a photo of letters printed on a label which was glued to a bottle, see Fig. 4 (left). The letters were organized in a 4 × 3 mesh with "A"s, "B"s, "V"s and "X"s each printed three times in a row. After a simple segmentation, the letters were labeled from left to right A_1, A_2, A_3, $B_1, \dots, V_1, \dots, X_1$, X_2, and X_3. Due to the curvature of the bottle surface, the letters appear distorted in the horizontal direction and the distortion grows to the right. A_1 does not exhibit any visible distortion while A_3 is the most distorted one and likewise for the other three letters. The task was to recognize (classify) these letters against a database containing the full English alphabet (26 undistorted letters of the same font).

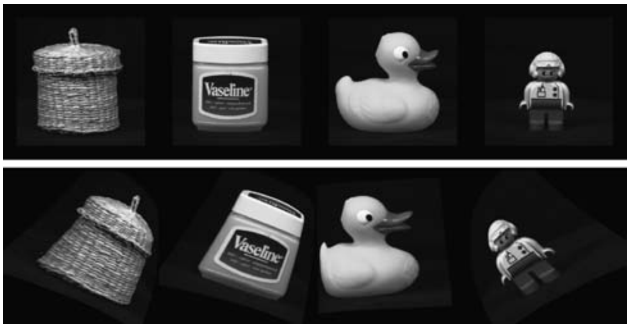

Fig. 3 The original images from the ALOI database (*top*) and their deformed versions (*bottom*)

In an "ideal" case when the camera is in infinity the image distortion can be described by orthogonal projection of the cylinder onto a plane, i.e.

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} = \begin{pmatrix} R\sin(\frac{x}{R}) \\ y \end{pmatrix},$$

where R is the bottle radius, x, y are the coordinates on the bottle surface and $\tilde{x}, \tilde{y}$ are the coordinates on the acquired images. In our case the object-to-camera distance was finite, so small perspective effect also appears. Although in this case we would be able to measure the camera-to-object distance and the radius R, we did not do that. We intentionally approached this experiment as a blind one, without employing any model/parameter knowledge.

We assumed the actual (unknown) image deformation can be approximated by a quadratic polynomial in x direction. Although it is clear that such approximation cannot be very accurate, we demonstrate that it is sufficient for our purpose.

We classified all deformed letters by means of the implicit invariants in the same way as in the previous experiment. Table 4 summarizes the classification results. Implicit invariants provided a perfect recognition rate as all deformed letters were classified correctly with high confidence (see the definition of the confidence in (g), Sect. 8). This may be a bit surprising with respect to the rough approximation of the transformation model we have used. It indicates some degree of robustness of the implicit invariants to the type of the image deformation.

The set-up of the last experiment was similar to the previous one. Again, we photographed a bottle with printed capital letters "M", "N", "E" and "K" but this time the label was intentionally tilted by a random degree of rotation (Fig. 4 right). We repeated this for different positions of the label which resulted in six differently deformed instances of each letter; see Fig. 5. One can see that, apart from the deformation induced by the cylindrical shape of the bottle, rotation

Table 4 Classification of four letters (each having three different degrees of distortion) by implicit invariants (first row) with a confidence measure in the second row

	A_1	A_2	A_3	B_1	B_2	B_3	V_1	V_2	V_3	X_1	X_2	X_3
Classified as	A	A	A	B	B	B	V	V	V	X	X	X
Confidence	65	29	15	64	186	69	96	62	80	35	98	19

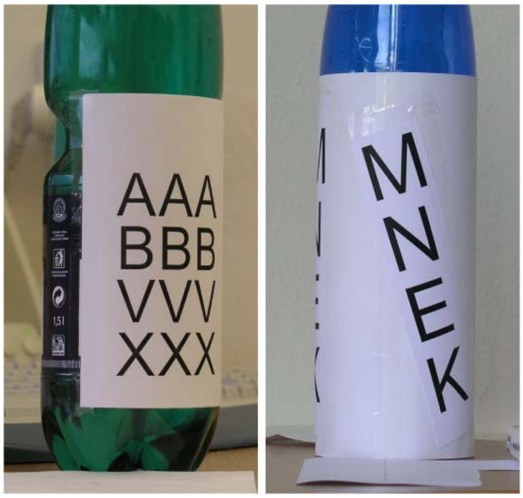

Fig. 4 The bottle images used in the experiment: the letters without rotation (*left*) and with rotation (*right*, one sample)

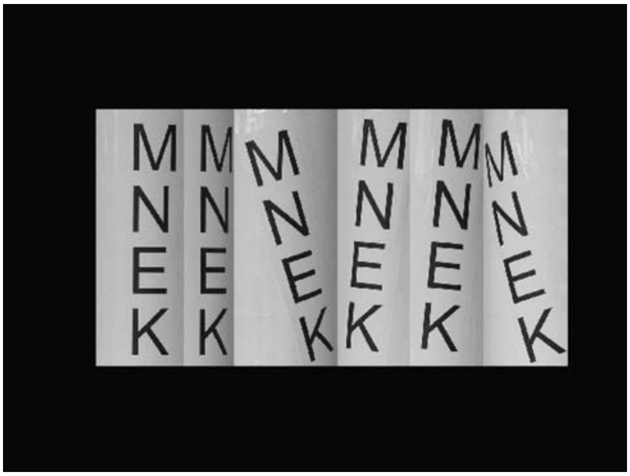

Fig. 5 The letters on the bottle exhibit distortion due to tilting and the cylindrical shape of the bottle. Images were cropped and stacked

and slight perspective is present. Scaling must be also considered, since the size of the captured letters and database templates may not match.

We assume the deformation can be approximated by linear polynomials in x and y directions, which model the similarity transform, and a quadratic polynomial in x direction approximates cylinder-to-plane projection. This leads to a class of geometric transforms of the shape (17) for which we have built implicit invariants.

As in the previous experiment, after segmentation of the deformed letters we used six implicit invariants to classify them against the undistorted alphabet of 26 letters. The recognition was quite successful—only the most deformed letter "N" (in Fig. 5, 2nd row, 6th column) was misclassified as "H", all other letters were recognized correctly.

For comparison, we also tried to recognize the letters by two other methods. First, we employ the affine moment invariants similarly as in the experiment with the ALOI images with a similar result—the AMI's failed in many cases because they are not designed to handle nonlinear deformations. Then we employed spline-based elastic matching proposed in Kybic and Unser (2003). This method searches the space of parameters of cubic B-spline deformation of the templates, calculates the mean square error as a measure of dissimilarity between the images and finds the minimum of this error function using hierarchical optimization. This method was designed in medical imaging for recognition of templates undergoing elastic deformations and was proved to be very powerful (Kybic and Unser 2003). We used the original implementation of the method kindly provided by J. Kybic. The recognition rate was 100%, which is not surprising—the method performs in fact exhaustive search in the space of deformations while working with full uncompressed representation of each template.

It is interesting to compare the methods in terms of their speed and confidence. The computational cost of both methods is of vastly different magnitude. On a computer with the Intel Pentium Core Duo T2400 processor, the time required to classify all 24 deformed letters against 26 letters in the database was in the case of elastic matching over one hour. On the other hand, it took less then one minute when using implicit invariants. Such a significant difference was caused mainly by the fact that elastic matching works with full image representation while implicit invariants use a big compression into few moments. The efficiency of the implicit invariants could be further improved by using fast algorithms for moment calculation, particularly in case of binary images.

Surprisingly, the use of different amount of data did not lead to different confidences. The confidence rates of both methods are not very high but similar; see Table 5. It is interesting to notice that extremely low confidence (close to 1) is frequent for letter "M" often having the second closest match "N". The explanation is that quadratic deformation of

Table 5 Classification confidence of four letters (each having six different degrees of distortion) using the implicit invariants (top) and the spline-based elastic matching (bottom). Arrangement of the confidence measures in the table is the same as the arrangement of the letters in Fig. 5 (MC means misclassified letter)

	Imp-inv. confidence					
M	9.4	1.3	2.3	6.4	1.1	1.6
N	28.5	11.9	1.3	9.7	10.9	(MC)
E	5.8	2.2	10.4	5.5	3.3	3.0
K	12.1	10.0	2.1	5.4	6.4	2.6
	Elastic reg. confidence					
M	11.5	1.7	1.9	9.8	10.5	1.4
N	8.3	3.9	2.5	9.3	6.8	2.5
E	6.4	3.1	2.2	5.3	3.8	2.0
K	10.8	3.0	1.9	5.0	3.4	2.2

"M" can merge the right diagonal line with the right vertical line and thus creating an impression of letter "N".

10 Conclusion

In this paper we introduced a new method for recognition of objects undergoing unknown elastic deformation. The method is based on so-called implicit invariants which are quantities that do not change under certain class of spatial transformations. The implicit invariants described in the paper are functions of image moments (preferably but not necessarily of orthogonal moments). We demonstrated how to derive implicit moment invariants with respect to general polynomial transformation of the spatial coordinates. It should be emphasized that this idea is much more general, it is not restricted to invariants from moments and to spatial transformations. Implicit invariants are general tools which can help in solving many classification tasks where explicit invariants do not exist. They are particularly useful when the transformations in question do not form a group.

We explained that any implicit invariant can be viewed as a distance measure between two images factorized by admissible transformation. However, implicit invariants need not be a metric in a strict sense. For polynomial transform r and for moment invariants defined in this paper, only some properties of a metric are fulfilled. For any f and g we have $I(f,g) \geq 0$ and $I(f,f) = I(f,r(f)) = 0$. If $I(f,g) = 0$ for invariants of all moments, then $g = r(f)$. This property follows from the uniqueness theorem of moments of compactly supported functions and holds in ideal infinite case only. It is not guaranteed in practice, when always working with a (small) finite set of moments. Since nonlinear polynomials are not invertible on the set of polynomials, I is not symmetric, i.e. $I(f,g) \neq I(g,f)$. This is why in practical applications we have to consider carefully the "direction" of the transform and construct the invariants accordingly. Triangular inequality $I(f,g) \leq I(f,h) + I(h,g)$ is also not guaranteed.

All moment invariants—explicit as well as implicit—suffers from high vulnerability to occlusion. This is an intrinsic property because they are calculated by integration over the whole image domain and small local image change/occlusion may significantly influence the moment values. It has been proven many times that moment invariants cannot be directly used for recognition of occluded objects. Our implicit invariants are not an exception. Some authors have made explicit moment invariants "local" in the following way. They divide the object into stable parts (most often this division was based on inflection points or vertices of the boundary) and describe each part separately by moment invariants. The whole object is then characterized by a string of vectors of invariants and recognition under occlusion is performed by maximum substring matching. This approach could be in principle adopted in case of implicit invariants too but due to the non-linear deformations it would be more difficult to find stable regions.

In the process of deriving implicit moment invariants with respect to general polynomial transforms we have exhibited an explicit algorithmic method for employing orthogonal polynomials without loosing their superior numerical stability. Finally, there is a wide variety of potential applications—apart form the above mentioned images printed on curved surfaces we envisage applications in omnidirectional vision (using this technique, objects might be classified directly without any re-projection into a plane) and in recognition of images taken by fish-eye lens camera.

Appendix

In this Appendix we present the proofs of Theorems 2, 3, and 4.

Proof of Theorem 2 As $tp_{j-1}(t)$ is a polynomial of degree j we can expand it in terms of our basis polynomials as

$$tp_{j-1}(t)=\sum_{k=0}^{j}\alpha_{j,k+1}p_k(t) \quad j=1,2,\ldots,n \tag{18}$$

where $\alpha_{j,j+1}\neq 0$ and $\alpha_{j,k}$ are some constants. This, with $\beta_n=\alpha_{n,n+1}$, establishes the matrix identity

$$t\boldsymbol{p}(t)=J\boldsymbol{p}(t)+\beta_n p_n(t)\boldsymbol{e}_n \tag{19}$$

where J is a lower Hessenberg matrix with elements $\alpha_{j,k}$. On the other hand, given J and p_0, we can establish $p_j(t)$, $j=1,2,\ldots,$ from a recurrence obtained by eliminating $p_j(t)$ from (18). This can be also expressed in matrix form as solving the system

$$(L-tS^T)\boldsymbol{p}(t)=\beta_0 p_0\boldsymbol{e}_1 \tag{20}$$

where $\boldsymbol{e}_1^T L=\beta_0\boldsymbol{e}_1^T$ for some $\beta_0\neq 0$ (choose $\beta_0=1$ or $\beta_0=\frac{1}{p_0}$) and the rest of matrix L comprises the first $n-1$ rows of J. Matrix L is thus a lower triangular matrix and the recurrence determining the values $\boldsymbol{p}(t)$ is a forward substitution on (20). □

Proof of Theorem 3 We first prove by induction that

$$t^k\boldsymbol{p}(t)=J^k\boldsymbol{p}(t)+\beta_n p_n(t)\sum_{j=1}^{k}t^{k-j}J^{j-1}\boldsymbol{e}_n. \tag{21}$$

For $k=0$ we just have $\boldsymbol{p}(t)=\boldsymbol{p}(t)$. Assume it holds for $k\geq 0$ and calculate, using (18):

$$\begin{aligned}
t^{k+1}\boldsymbol{p}(t)&=t(t^k\boldsymbol{p}(t))\\
&=t\left(J^k\boldsymbol{p}(t)+\beta_n p_n(t)\sum_{j=1}^{k}t^{k-j}J^{j-1}\boldsymbol{e}_n\right)\\
&=J^k(J\boldsymbol{p}(t)+\beta_n p_n(t)\boldsymbol{e}_n)\\
&\quad+\beta_n p_n(t)\sum_{j=1}^{k}t^{k-j+1}J^{j-1}\boldsymbol{e}_n\\
&=J^{k+1}\boldsymbol{p}(t)+\beta_n p_n(t)\sum_{j=1}^{k+1}t^{k+1-j}J^{j-1}\boldsymbol{e}_n
\end{aligned}$$

as required for the induction step.

Equation (8) is now a linear combination of (21) for $s(t)=\sum_{k=0}^{d}\sigma_k t^k$. □

The shifted polynomials s_k can be obtained from s recurrently by

$$s_0=s \quad \text{and} \quad s_{k+1}(t)=\frac{s_k(t)-s_k(0)}{t}.$$

Note that $s_d=\sigma_d$.

Proof of Theorem 4 The composite polynomial $\tilde p_j(r(t))$ has degree jd and the degree of $\tilde p_{\tilde n-1}(r(t))r'(t)$ is $(\tilde n-1)d+d-1=n-1$ which determines (10). There is a $\tilde n\times n$ matrix B such that

$$\tilde{\boldsymbol{p}}(r(t))=B\boldsymbol{p}(t). \tag{22}$$

Substitute $\tilde t=r(t)$ into (9) and, using (22) and also

$$\tilde\beta_{\tilde n}\tilde p_{\tilde n}(r(t))=\boldsymbol{b}^T\boldsymbol{p}(t)$$

for some vector $\boldsymbol{b}$, we obtain

$$r(t)B\boldsymbol{p}(t)=(\tilde J B+\boldsymbol{e}_{\tilde n}\boldsymbol{b}^T)\boldsymbol{p}(t). \tag{23}$$

On the other hand, use (8) with $s(t)$ replaced by $r(t)$ and pre-multiply it by B to get

$$r(t)B\boldsymbol{p}(t)=Br(J)\boldsymbol{p}(t)+\beta_n p_n(t)\sum_{k=1}^{d}r_k(t)BJ^{k-1}\boldsymbol{e}_n.$$

The last term must vanish as there are no other terms of degree at least n so that, comparing with (23), we obtain the second equation in (12). The first one is obvious as $\tilde p_0(r(t))=\tilde p_0$ is still a constant. Using a similar notation for the elements of $\tilde J$ as we did for J the j-th row of (23) gives the recurrence

$$\tilde\beta_j\boldsymbol{e}_{j+1}^T B=\boldsymbol{e}_j^T Br(J)-\sum_{k=1}^{j}\tilde\alpha_{j,k}\boldsymbol{e}_k^T B$$

for $j=1,2,\ldots,\tilde n-1$ (if required, vector $\boldsymbol{b}$ can also be obtained by using $j=\tilde n$). To complete the construction use (8) again, but with $s(t)=r'(t)$, to obtain, using (3), (22) and (8),

$$\begin{aligned}
A\boldsymbol{p}(t)&=s(t)B\boldsymbol{p}(t)=Bs(t)\boldsymbol{p}(t)\\
&=Bs(J)\boldsymbol{p}(t)+\beta_n p_n(t)\sum_{k=1}^{d-1}s_k(t)BJ^{k-1}\boldsymbol{e}_n
\end{aligned}$$

from which (11) follows by comparing terms with the same powers of t. □

References

Abu-Mostafa, Y. S., & Psaltis, D. (1984). Recognitive aspects of moment invariants. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *6*, 698–706.

Abu-Mostafa, Y. S., & Psaltis, D. (1985). Image normalization by complex moments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *7*, 46–55.

Bailey, R. R., & Srinath, M. (1996). Orthogonal moment features for use with parametric and non-parametric classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *18*, 389–398.

Belkasim, S. O., Shridhar, M., & Ahmadi, M. (1991). Pattern recognition with moment invariants: a comparative study and new results. *Pattern Recognition*, *24*, 1117–1138.

Duda, R. O., Hart, P. E., & Stork, D. (2001). *Pattern classification* (2nd ed.). New York: Wiley Interscience.

Dudani, S. A., Breeding, K. J., & McGhee, R. B. (1977). Aircraft identification by moment invariants. *IEEE Transactions on Computers*, *26*, 39–45.

El-Khaly, F., & Sid-Ahmed, M. A. (1990). Machine recognition of optically captured machine printed arabic text. *Pattern Recognition*, *23*, 1207–1214.

Flusser, J. (2000). On the independence of rotation moment invariants. *Pattern Recognition*, *33*(9), 1405–1410.

Flusser, J. (2002). On the inverse problem of rotation moment invariants. *Pattern Recognition*, *35*, 3015–3017.

Flusser, J., & Suk, T. (1993). Pattern recognition by affine moment invariants. *Pattern Recognition*, *26*, 167–174.

Flusser, J., & Suk, T. (1994a). Affine moment invariants: A new tool for character recognition. *Pattern Recognition Letters*, *15*, 433–436.

Flusser, J., & Suk, T. (1994b). A moment-based approach to registration of images with affine geometric distortion. *IEEE Transactions on Geoscience and Remote Sensing*, *32*, 382–387.

Flusser, J., & Suk, T. (2006). Rotation moment invariants for recognition of symmetric objects. *IEEE Transactions on Image Processing*, *15*, 3784–3790.

Geusebroek, J. M., Burghouts, G. J., & Smeulders, A. W. M. (2005). The Amsterdam library of object images. *International Journal of Computer Vision*, *61*(1), 103–112.

Golub, G., & Kautsky, J. (1983). Calculation of Gauss quadratures with multiple free and fixed knots. *Numerische Mathematik*, *41*, 147–163.

Gool, L. V., Moons, T., Pauwels, E., & Oosterlinck, A. (1995). Vision and Lie's approach to invariance. *Image and Vision Computing*, *13*, 259–277.

Goshtasby, A. (1988). Registration of images with geometric distortions. *IEEE Transactions on Geoscience and Remote Sensing*, *26*, 60–64.

Gruber, M., & Hsu, K. Y. (1997). Moment-based image normalization with high noise-tolerance. *Pattern Recognition*, *19*, 136–139.

Gurevich, G. B. (1964). *Foundations of the theory of algebraic invariants*. Noordhoff: Groningen.

Hilbert, D. (1993). *Theory of algebraic invariants*. Cambridge: Cambridge University Press.

Hu, M. K. (1962). Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, *8*, 179–187.

Hupkens, T. M., & de Clippeleir, J. (1995). Noise and intensity invariant moments. *Pattern Recognition*, *16*, 371–376.

Kautsky, J., & Golub, G. (1983). On the calculation of Jacobi matrices. *Linear Algebra Applications*, *52/53*, 439–455.

Khotanzad, A., & Hong, Y. H. (1990). Invariant image recognition by Zernike moments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *12*, 489–497.

Kybic, J., & Unser, M. (2003). Fast parametric elastic image registration. *IEEE Transactions on Image Processing*, *12*(11), 1427–1442.

Li, Y. (1992). Reforming the theory of invariant moments for pattern recognition. *Pattern Recognition*, *25*, 723–730.

Liao, S. X., & Pawlak, M. (1996). On image analysis by moments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *18*, 254–266.

Maitra, S. (1979). Moment invariants. *Proceedings of the IEEE*, *67*, 697–699.

Mamistvalov, A. G. (1998). N-dimensional moment invariants and conceptual mathematical theory of recognition n-dimensional solids. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *20*, 819–831.

Mukundan, R., & Malik, N. K. (1993). Attitude estimation using moment invariants. *Pattern Recognition Letters*, *14*, 199–205.

Mukundan, R., & Ramakrishnan, K. R. (1996). An iterative solution for object pose parameters using image moments. *Pattern Recognition Letters*, *17*, 1279–1284.

Mundy, J., & Zisserman, A. (1992). *Geometric invariance in computer vision*. Cambridge: MIT Press.

Pawlak, M. (1992). On the reconstruction aspects of moment descriptors. *IEEE Transactions on Information Theory*, *38*, 1698–1708.

Pawlak, M. (2006). *Image analysis by moments: reconstruction and computational aspects*. Wroclaw: Wroclaw University of Technology Press.

Pizlo, Z., & Rosenfeld, A. (1992). Recognition of planar shapes from perspective images using contour-based invariants. *CVGIP: Image Understanding*, *56*(3), 330–350.

Prokop, R. J., & Reeves, A. P. (1992). A survey of moment-based techniques for unoccluded object representation and recognition. *CVGIP: Graphical Models and Image Processing*, *54*, 438–460.

Reiss, T. H. (1991). The revised fundamental theorem of moment invariants. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *13*, 830–834.

Schur, I. (1968). *Vorlesungen uber Invariantentheorie*. Berlin: Springer.

Sluzek, A. (1995). Identification and inspection of 2-D objects using new moment-based shape descriptors. *Pattern Recognition Letters*, *16*, 687–697.

Suk, T., & Flusser, J. (2004a). Graph method for generating affine moment invariants. In *Proceedings of the 17th int. conf. pattern recognition ICPR'04* (Vol. 2, pp. 192–195). Cambridge, UK.

Suk, T., & Flusser, J. (2004b). Projective moment invariants. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*, 1364–1367.

Teague, M. R. (1980). Image analysis via the general theory of moments. *Journal of Optical Society of America*, *70*, 920–930.

Teh, C. H., & Chin, R. T. (1988). On image analysis by the method of moments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *10*, 496–513.

Tsirikolias, K., & Mertzios, B. G. (1993). Statistical pattern recognition using efficient two-dimensional moments with applications to character recognition. *Pattern Recognition*, *26*, 877–882.

Wallin, A., & Kubler, O. (1995). Complete sets of complex Zernike moment invariants and the role of the pseudoinvariants. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *17*, 1106–1110.

Wang, L., & Healey, G. (1998). Using Zernike moments for the illumination and geometry invariant classification of multispectral texture. *IEEE Transactions on Image Processing*, *7*, 196–203.

Weiss, I. (1988). Projective invariants of shapes. In *Proc. image understanding workshop* (pp. 1125–1134).

Wong, R. Y., & Hall, E. L. (1978). Scene matching with invariant moments. *Computer Graphics and Image Processing*, *8*, 16–24.

Wong, W. H., Siu, W. C., & Lam, K. M. (1995). Generation of moment invariants and their uses for character recognition. *Pattern Recognition Letters*, *16*, 115–123.

Yang, L., & Albregtsen, F. (1996). Fast and exact computation of Cartesian geometric moments using discrete Green's theorem. *Pattern Recognition*, *29*, 1061–1073.