Pattern Recognition Letters 31 (2010) 1428–1436

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Recursive computation of moments of 2D objects represented by elliptic Fourier descriptors

Octavian Soldea *, Mustafa Unel, Aytul Ercil

Sabanci University, Orhanli, Tuzla 34956, Istanbul, Turkey

ARTICLE INFO

Article history:
Received 23 March 2009
Received in revised form 12 November 2009
Available online 23 February 2010
Communicated by H.H.S. Ip

Keywords:
Elliptic Fourier descriptors
Moments
Superquadrics
B-spline functions
Bernstein–Bézier representations

ABSTRACT

This paper develops a recursive method for computing moments of 2D objects described by elliptic Fourier descriptors (EFD). To this end, Green's theorem is utilized to transform 2D surface integrals into 1D line integrals and EFD description is employed to derive recursions for moments computations. A complexity analysis is provided to demonstrate space and time efficiency of our proposed technique. Accuracy and speed of the recursive computations are analyzed experimentally and comparisons with some existing techniques are also provided.

1. Introduction

Moments of objects (Prokop and Reeves, 1992) are intrinsic to the shape (Meriam, 1966), and therefore efficient computation of moments is a desirable feature for many practical tasks.

Moments of inertia are used in mechanical design and analysis. For example, in the design of aircrafts, ships, and automobiles the moments of inertia are employed to determine the dynamics of the vehicle (Weinstein et al., 2006). In the medical domain, moments are used for automatic diagnosis and prognosis; for example they can be used in computing volumes of healthy and pathologic tissues (Jirapatnakul et al., 2009).

Moment invariants are efficient tools in pattern recognition applications. In (Zhenjiang, 2000), the author presents a moment based pattern recognition application in agronomy and proposes a measure for the analysis of the roundness of rose flowers. Another interesting application of moments can be found, for example, in optical character recognition systems (Cash and Hatamian, 1987). Until recently, it was a common belief that projective moment invariants do not exist; however, their existence was proven in (Suk and Flusser, 2004).

In the past, moments of shapes have usually been computed using all the pixels or voxels that define the analyzed objects. The importance of using modeling tools for boundaries in the context of moments was already understood two decades ago. The work in (Leu, 1991) is among the first which develops a scheme for computing moments of objects employing boundaries, which are polygons.

In (Gonzales-Ochoa et al., 1998), the authors assume that the input is a free-form object modeled by parametric surfaces. They employ the Stokes' theorem (Fikhtengol'ts, 1965) to transform the computation of moments using integrals on volumes to integrals on the surface boundaries, of the input objects. Following Gonzales-Ochoa et al. (1998), in (Soldea et al., 2000), the authors develop two schemes for computing moments of free-form objects. The free-form representations are based on B-Spline blending functions. A particular case is further developed and analyzed in detail in (Sheynin and Tuzikov, 2003).

In this work, we consider moments of another important boundary representation method, namely, the elliptic Fourier descriptors (EFD) (Kuhl and Giardina, 1982; Jeong and Radke, 2007). We first present an overview of EFD and its applications, and then motivate moments of EFD, which highlights our contribution.

1.1. Elliptic Fourier descriptors (EFD)

EFD representation contains shape information in the low frequency components (Kuhl and Giardina, 1982; Lin and Hwang, 1987; Wallace and Mitchell, 1980). This property puts EFD in a central place in medical applications. For example, in (Jeong and Radke, 2007), the authors use EFD to approximate arbitrary axial slices obtained from a volumetric data. They also show that EFD is the key for achieving low computational cost while preserving the accuracy of the approximation, as compared to other existing techniques. EFD also implies various algebraic and geometric

* Corresponding author. Tel.: +90 5457952874; fax: +90 2164839550.
E-mail address: octavian@sabanciuniv.edu (O. Soldea).

0167-8655/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2010.02.009

invariants (Lin and Hwang, 1987; Lestrel, 1997). Moreover, EFD can be converted into implicit polynomial (IP) representations (Yalcin et al., 2003).

The idea of encoding most of the information in low frequency components is re-iterated in (Wang et al., 2006), where the authors employ EFD towards learning and modeling shapes that manifest minimal description lengths. Accordingly, contours of objects are decomposed into a series of elliptical harmonics. In another work (Staib and Duncan, 1992), the authors advocate the use of EFD, among other parametric representations, for segmentation.

1.2. Moments of elliptic Fourier descriptors

Although moments of objects in different forms have been widely studied in the literature, to the best of our knowledge, the moments of the elliptic Fourier descriptors (EFD) have not been explored until now. Since EFD representation is one of the most powerful boundary modeling tools, efficient computation of its moments may prove very useful in several model-based vision and pattern recognition applications. Motivated by this observation, in this work, we develop a computationally efficient recursive scheme for calculating moments of objects represented by elliptic Fourier descriptors. To this end, we utilize Green's theorem to transform 2D surface integrals into 1D line integrals and employ EFD representation to derive recursions for moments computations. A theoretical complexity analysis is developed to demonstrate time and space efficiency of our proposed approach. Comparison of the computational complexity of our method with other techniques is also provided. Several experiments on a large database are performed to quantify the accuracy of our proposed method and compare it with other representations such as Bernstein–Bézier representations (Cohen et al., 2001).

2. Background on elliptic Fourier descriptors

Following Yip et al. (1994), let T be an arbitrary positive real number and let $C(t) : [0..T] \to R^2$,

$$C(t) = (x(t), y(t)) \tag{1}$$

be a planar curve parameterized by t, such that $C(t) \in C^{(2)}$. We can describe the curve in Eq. (1) using elliptic Fourier descriptors as follows:

$$\begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \sum_{i=0}^{\infty} \begin{pmatrix} a_i & b_i \\ c_i & d_i \end{pmatrix} \begin{pmatrix} \cos\left(\frac{2i\pi t}{T}\right) \\ \sin\left(\frac{2i\pi t}{T}\right) \end{pmatrix}, \tag{2}$$

where

$$a_0 = \frac{1}{T}\int_0^T x(t)dt, \quad b_0 = 0, \quad c_0 = 0, \quad d_0 = \frac{1}{T}\int_0^T y(t)dt,$$
$$a_i = \frac{2}{T}\int_0^T x(t)\cos\left(\frac{2\pi it}{T}\right)dt, \quad b_i = \frac{2}{T}\int_0^T x(t)\sin\left(\frac{2\pi it}{T}\right)dt,$$
$$c_i = \frac{2}{T}\int_0^T y(t)\cos\left(\frac{2\pi it}{T}\right)dt, \quad d_i = \frac{2}{T}\int_0^T y(t)\sin\left(\frac{2\pi it}{T}\right)dt,$$

for any $i \in N - \{0\}$. Since $\cos(\cdot)$ and $\sin(\cdot)$ are continuous functions, the integrability of $C(t)$ ensures existence of the above integrals.

3. Moments of 2D shapes represented by EFD

In this section, we develop a recursive scheme for computing moments of 2D shapes represented by EFD. The recursive scheme provides efficient computation of moments.

We divide the computation of moment $m_{p,q}$, which is defined as a surface integral, into two components that are defined as 1D line integrals. The conversion from surface to line integral is achieved by the utilization of Green's theorem (Fikhtengol'ts, 1965). We outline the fundamental steps in deriving recursions. For details of these derivations, we refer the reader to Soldea et al. (2010).

Consider the 2D shape $D \subseteq R^2$. The standard moment of order p, q of D is given as:

$$\begin{aligned} m_{p,q} &= \int\int_D x^p y^q dxdy = \frac{1}{2}\int\int_D x^p y^q dxdy - \frac{1}{2}\int\int_D -x^p y^q dxdy \\ &= \frac{1}{2}\int_{t=0}^{T} \frac{x(t)^{p+1} y(t)^q y'(t)}{p+1} dt - \frac{1}{2}\int_{t=0}^{T} \frac{x(t)^p y(t)^{q+1} x'(t)}{q+1} dt, \end{aligned} \tag{3}$$

where we employed Green's theorem. Note that Eq. (3) can be rewritten as

$$m_{p,q} = \frac{1}{2}\int_{t=0}^{T} x(t)^p y(t)^q \left[\frac{x(t)y'(t)}{p+1} - \frac{y(t)x'(t)}{q+1}\right] dt. \tag{4}$$

We define the following quantities:

$$\alpha_{i,j,p,q} = j\left(\frac{a_i d_j}{p+1} - \frac{b_j c_i}{q+1}\right), \tag{5}$$

$$\beta_{i,j,p,q} = j\left(\frac{-a_i c_j}{p+1} + \frac{c_i a_j}{q+1}\right) + i\left(\frac{d_i b_j}{p+1} - \frac{b_i d_j}{q+1}\right), \tag{6}$$

$$\gamma_{i,j,p,q} = i\left(\frac{-b_j c_i}{p+1} + \frac{d_j a_i}{q+1}\right), \tag{7}$$

$$m^c_{i,p,q} = \int_{t=0}^{T} x(t)^p y(t)^q \cos\left(\frac{2i\pi t}{T}\right) dt, \tag{8}$$

$$m^s_{i,p,q} = \int_{t=0}^{T} x(t)^p y(t)^q \sin\left(\frac{2i\pi t}{T}\right) dt. \tag{9}$$

Substituting Eq. (2) into (4) and using Eqs. (5)–(9), we get

$$\begin{aligned} m_{p,q} = \frac{\pi}{2T}\sum_{i=0}^{\infty}\sum_{j=0}^{\infty} \Big\{ &\alpha_{i,j,p,q}(m^c_{|i-j|,p,q} + m^c_{i+j,p,q}) \\ &+ \beta_{i,j,p,q}[m^s_{i+j,p,q} - \sigma(i-j)m^s_{|i-j|,p,q}] + \gamma_{i,j,p,q}\left(m^c_{|i-j|,p,q} - m^c_{i+j,p,q}\right)\Big\}. \end{aligned} \tag{10}$$

The computation of $m_{p,q}$ is based on the values $m^c_{i,p,q}$, and $m^s_{i,p,q}$, which will be computed recursively as shown below.

3.1. $i > 0, p > 0$ and $q > 0$:

We have

$$\begin{aligned} m^c_{i,p,q} = \frac{1}{2i}\sum_{k=0}^{\infty} k\Big\{ &pa_k(m^c_{|k-i|,p-1,q} - m^c_{k+i,p-1,q}) \\ &- pb_k\left[m^s_{k+i,p-1,q} - \sigma(k-i)m^s_{|k-i|,p-1,q}\right] \\ &+ qc_k(m^c_{|k-i|,p,q-1} - m^c_{k+i,p,q-1}) \\ &- qd_k\left[m^s_{k+i,p,q-1} - \sigma(k-i)m^s_{|k-i|,p,q-1}\right]\Big\} \end{aligned} \tag{11}$$

and

$$\begin{aligned} m^s_{i,p,q} = \frac{1}{2i}\sum_{k=0}^{\infty} k\Big\{ &-pa_k\left[m^s_{k+i,p-1,q} + \sigma(k-i)m^s_{|k-i|,p-1,q}\right] \\ &+ pb_k\left(m^c_{k+i,p-1,q} + m^c_{|k-i|,p-1,q}\right) \\ &- qc_k\left[m^s_{k+i,p,q-1} + \sigma(k-i)m^s_{|k-i|,p,q-1}\right] \\ &+ qd_k\left(m^c_{k+i,p,q-1} + m^c_{|k-i|,p,q-1}\right)\Big\}. \end{aligned} \tag{12}$$

For details please see Soldea et al. (2010).

3.2. Recursion bases

We next provide several simplified results for particular cases of interest.

- $i = 0,\ p = 0,$ and $q = 0$:

$$m^c_{0,0,0} = T \quad \text{and} \quad m^s_{0,0,0} = 0. \tag{13}$$

- $i = 0,\ p = 0,$ and $q > 0$:

$$m^c_{0,0,q} = \sum_{i=0}^{\infty} \left(c_i m^c_{i,0,q-1} + d_i m^s_{i,0,q-1} \right) \quad \text{and} \quad m^s_{0,0,q} = 0. \tag{14}$$

- $i = 0,\ p > 0,$ and $q = 0$:

$$m^c_{0,p,0} = \sum_{i=0}^{\infty} \left(a_i m^c_{i,p-1,0} + b_i m^s_{i,p-1,0} \right) \quad \text{and} \quad m^s_{0,p,0} = 0. \tag{15}$$

- $i = 0,\ p > 0,$ and $q > 0$:

$$\begin{aligned} m^c_{0,p,q} = \frac{1}{2} \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} [& (a_i c_j + b_i d_j) m^c_{|i-j|,p-1,q-1} \\ & + (a_i c_j - b_i d_j) m^c_{i+j,p-1,q-1} \\ & + (-a_i d_j + b_i c_j) \sigma(i-j) m^s_{|i-j|,p-1,q-1} \\ & + (a_i d_j + b_i c_j) m^s_{i+j,p-1,q-1}] \end{aligned} \tag{16}$$

$$\text{and } m^s_{0,p,q} = 0. \tag{17}$$

- $i > 0,\ p = 0,$ and $q = 0$:

$$m^c_{i,0,0} = m^s_{i,0,0} = 0. \tag{18}$$

- $i > 0,\ p = 0,$ and $q > 0$:

$$\begin{aligned} m^c_{i,0,q} = \frac{q}{2i} \sum_{k=0}^{\infty} k \Big[& c_k \left(m^c_{|k-i|,0,q-1} - m^c_{k+i,0,q-1} \right) \\ & - d_k \left(m^s_{k+i,0,q-1} - \sigma(k-i) m^s_{|k-i|,0,q-1} \right) \Big], \end{aligned} \tag{19}$$

$$\begin{aligned} m^s_{i,0,q} = \frac{q}{2i} \sum_{k=0}^{\infty} k \Big[& -c_k \left(m^s_{k+i,0,q-1} + \sigma(k-i) m^s_{|k-i|,0,q-1} \right) \\ & + d_k \left(m^c_{k+i,0,q-1} + m^c_{|k-i|,0,q-1} \right) \Big]. \end{aligned} \tag{20}$$

- $i > 0,\ p > 0,$ and $q = 0$.

$$\begin{aligned} m^c_{i,p,0} = \frac{p}{2i} \sum_{k=0}^{\infty} k \Big[& a_k \left(m^c_{|k-i|,p-1,0} - m^c_{k+i,p-1,0} \right) \\ & - b_k \left(m^s_{k+i,p-1,0} - \sigma(k-i) m^s_{|k-i|,p-1,0} \right) \Big], \end{aligned} \tag{21}$$

$$\begin{aligned} m^s_{i,p,0} = \frac{p}{2i} \sum_{k=0}^{\infty} k \Big[& -a_k \left(m^s_{k+i,p-1,0} + \sigma(k-i) m^s_{|k-i|,p-1,0} \right) \\ & + b_k \left(m^c_{|k-i|,p-1,0} + m^c_{k+i,p-1,0} \right) \Big]. \end{aligned} \tag{22}$$

3.3. Complexity analysis of the proposed approach

If schemes of computing moments can be converted into recursive algorithms (Soldea et al., 2000), one can take advantage of *amortized times* of computations. In this context, usually, moments of higher degree depend on lower degrees.

Let $M_r \triangleq \{m_{p,q} | p+q = r\}$ be the set of moments of order r. Moreover, let $\mathscr{M}_r \triangleq \bigcup_{i=0}^{r} M_i$ be the complete set of moments up to r. There are plenty of scenarios where complete sets of moments up to a certain degree may be required. For example, consider the problem of characterization of the position of an object (Meriam, 1966). In this case, we need to compute the mass center as well as the orientation of the analyzed object. While the mass center is given by the zero and first order moments (M_0 and M_1), the orientation is provided by the principal axes of inertia, which depend on the second order moments (M_2). Therefore, we need to compute $\mathscr{M}_2$. If the directions of the principal axes are required, the third order moments (M_3) are required as well.

Assume that additions, subtractions, multiplications, and divisions require $O(1)$ time. In addition, assume that the sign function $\sigma(\cdot)$ (see Eqs. (10)–(12)) requires $O(1)$ time as well. Furthermore, suppose that the order of the EFD used is L, i.e. Eq. (2) can be approximated as

$$\begin{pmatrix} x(t) \\ y(t) \end{pmatrix} \approx \sum_{i=0}^{L} \begin{pmatrix} a_i & b_i \\ c_i & d_i \end{pmatrix} \begin{pmatrix} \cos\left(\frac{2i\pi t}{T}\right) \\ \sin\left(\frac{2i\pi t}{T}\right) \end{pmatrix}.$$

Let T_x be the time required for computing the coefficient x, where x can be $\alpha_{i,j,p,q}, \beta_{i,j,p,q}, \gamma_{i,j,p,q}, m_{p,q}, m^c_{i,p,q}$ or $m^c_{i,p,q}$. Then, in light of Eqs. (5)–(7), we have $T_\alpha = T_\beta = T_\gamma = O(1)$. Moreover, in light of Eq. (10), we have

$$T_{m_{p,q}} = O\left(L^2\right) + Amortized(p,q), \tag{23}$$

where $Amortized(p,q)$ is the amortized time required to evaluate $m^c_{i,p,q}$ and $m^s_{i,p,q}$ coefficients that are required in the computation of M_{p+q} as compared to $\mathscr{M}_{p+q-1}$, to be described below. In light of Eqs. (11) and (12), $T_{m^c_{i,p,q}} = T_{m^s_{i,p,q}} = O(L)$. Next, we analyze the amortized time. Let

$$M^c_{r,a,b} \triangleq \left\{ m^c_{i,p,q} | p+q = r, i \in \{a, a+1, \ldots, b\} \right\},$$
$$M^s_{r,a,b} \triangleq \left\{ m^s_{i,p,q} | p+q = r, i \in \{a, a+1, \ldots, b\} \right\},$$

and

$$M_{r,a,b} \triangleq M^c_{r,a,b} \bigcup M^s_{r,a,b}.$$

The set of moments M_{r+1} as compared to $\mathscr{M}_r$, requires the computation of $A_{r+1} = \bigcup_{s=0}^{r+1} M_{s,0,2L}, A_r = \bigcup_{s=0}^{r} M_{s,2L+1,3L}, \ldots, A_0 = M_{0,(r+1)L+1,(r+2)L}$. The computation of $M_{r+1,0,2L}$ is done using Eq. (10). Following Eqs. (11) and (12), a worst case upper bound for the cardinalities of A_k is $O(L(p+q)^2)$, for any $k \in \{0, 1, \ldots, r+1\}$. Bearing in mind that the cardinality of $M_{p,q}$ is $p+q+1$, we have $Amortized(p,q) = O\left(\frac{(p+q+1)L(p+q)^2}{p+q+1}\right) = O(L(p+q)^2)$. Therefore, the amortized time for each moment is

$$O(L(p+q)^2). \tag{24}$$

Substituting Eq. (24) into (23), we have

$$T_{m_{p,q}} = O(L^2 + L(p+q)^2). \tag{25}$$

In the theoretical case when $L, p,$ and q are considered constants, we have $T_{m_{p,q}} = O(1)$. However, we recommend that, in practice, only p and q be considered constants for small values. In our case the inherent constants in the complexity order evaluations are relatively high; therefore, whenever one needs to compute higher order moments, these constant should be regarded as polynomial (quadratic) time consumption factors.

On the other hand, consider the method of computing moments in (Leu, 1991). Each object under analysis is received at input as a polygon, and each edge defines a triangle that has one of its corners at the origin of the coordinate axes. Each such triangle defines a trapezoid and two triangles for integration purposes. The computation of moments is driven via such decomposition for each edge. For example, $m_{0,0}$, which is the area of the input object, is decomposed into trapezoids and triangles whose areas are summed or subtracted according to their orientation. In this scheme, each moment $m_{p,q}$ is subject to computing all possible combinations $\binom{n}{m} \forall n \in \{0, \ldots, p+q\}, \forall m \in \{0, \ldots, n\}$. These values are required for coefficient multiplications. In addition, for each corner point in the input polygon (x,y), the values $x^2, x^3, \ldots, x^{p+q+1}$ are also required. Suppose that the input polygon has L corners. Then, the

amortized time for computing the $m_{p,q}$ moment is $O(qL)$. If q is a constant, then this time resumes to $O(L)$.

The problem treated in (Leu, 1991) is a particular case of an object represented by B-spline functions of degree one, i.e. straight lines, see also (Soldea et al., 2000; Sheynin and Tuzikov, 2003). Following Theorem 3 in (Sheynin and Tuzikov, 2003), the time requirement for computing the moments in M_2 is $O(s^3L)$, where s is the degree of the B-spline used and L is the number of points. For objects represented by B-spline functions, the time requirements for moments computation increases with the degree and the number of points. While the number of points seems to have a linear effect, the degree implies a polynomial dependency. However, bearing in mind that the modeling community usually employs cubic B-spline functions, the degree can be considered constant.

The theoretical time requirements of these schemes are low and comparable. The reason for the efficiency relies on the fact that all of them compute moments of objects represented by boundaries. In terms of memory consumption, our scheme needs $L^2(p+q+1)^4$ memory locations. For today's computers, this is a very small space requirement. All other schemes imply similar low memory requirements. Note that the vast majority of schemes of moments computations can be formulated in terms of lower space consumption, provided more time computation is required. Higher order moments can be computed based on recomputing lower order moments several times, but not memorizing their values. Hence, the memory consumption in eventual recursive computations is very small. We deep further in our time analysis in Section 4.4.

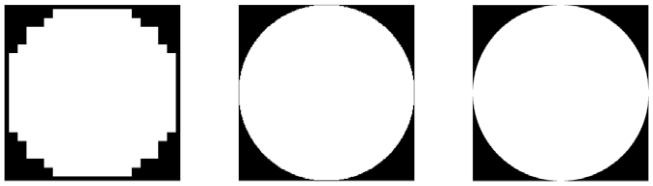

Fig. 1. Circles with different radii are scaled for visual comparison: (a) radius $R = 10$ and scaling factor of 100; (b) radius $R = 100$ and scaling factor of 10. (c) radius $R = 1000$ and scaling factor of 1.

4. Experimental results

We present experimental results related to the accuracy of our proposed computational scheme. We test our method on different shapes and provide several comparisons with other methods.

4.1. A comparison of accuracy on circles and superquadrics

We picked several shapes for which moments can easily be computed analytically. First, we use discrete moments computations method. Second, we use our proposed method. Third, we use an analytical method where the surface integrals are transformed into line integrals and computed with Maple (Geddes et al., 2002).

We selected three circles (disks) with radii $R = 10,\ 100,$ and 1000, and centered at $(R, 0)$, see Fig. 1. Table 1 shows a comparison of moment computations.

Next, we analyze the accuracy of moment computations on the superquadric $x(t) = \cos(t)^5$ and $y(t) = \sin(t)^5$, which is parameterized over $[0..2\pi)$. By a reparameterization over $[0..T]$, we have

$$x(t) = \cos\left(\frac{2\pi t}{T}\right)^5 \quad \text{and} \quad y(t) = \sin\left(\frac{2\pi t}{T}\right)^5, \tag{26}$$

Consider the following trigonometric identities:

$$\cos^5(t) = \frac{10\cos(t) + 5\cos(3t) + \cos(5t)}{16} \tag{27}$$

and

$$\sin^5(t) = \frac{10\sin(t) - 5\sin(3t) + \sin(5t)}{16}. \tag{28}$$

Substituting these into (26), we obtain the following EFD representation

$$\begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} \cos^5\left(\frac{2\pi t}{T}\right) \\ \sin^5\left(\frac{2\pi t}{T}\right) \end{pmatrix} = \frac{1}{16}\begin{pmatrix} 10\cos\left(\frac{2\pi t}{T}\right) + 5\cos\left(\frac{3\cdot 2\pi t}{T}\right) + \cos\left(\frac{5\cdot 2\pi t}{T}\right) \\ 10\sin\left(\frac{2\pi t}{T}\right) - 5\sin\left(\frac{3\cdot 2\pi t}{T}\right) + \sin\left(\frac{5\cdot 2\pi t}{T}\right) \end{pmatrix}. \tag{29}$$

In light of Eq. (2), the non-zero coefficients are $a_1 = 0.625 = \frac{10}{16}$, $d_1 = 0.625 = \frac{10}{16}$, $a_3 = 0.3125 = \frac{5}{16}$, $d_3 = -0.3125 = -\frac{5}{16}$, $a_5 = 0.0625 = \frac{1}{16}$, and $d_5 = 0.0625 = \frac{1}{16}$. Fig. 2 depicts the superquadric $x(t) = \cos(t)^5$ and $y(t) = \sin(t)^5$. Table 2 shows accuracy comparisons for this superquadric.

Table 1
Moments for the circles with radii $R = 10,\ 100,$ and 1000, and centered at $(R, 0)$.

Moments-circle $R = 10$	Discrete	EFD	Analytic
$m_{0,0}$	305	314.159265358979	314.159265358979
$m_{1,0}$	3050	3141.59265358979	3141.59265358979
$m_{0,1}$	0	0	0
$m_{2,0}$	37,906	39269.9081698724	39269.9081698724
$m_{1,1}$	0	0	0
$m_{0,2}$	7406	7853.98163397448	7853.98163397448
Moments-circle $R = 100$	Discrete	EFD	Analytic
$m_{0,0}$	31,397	31415.9265358979	31415.9265358979
$m_{1,0}$	3,139,700	3141592.65358979	3141592.65358979
$m_{0,1}$	0	0	0
$m_{2,0}$	392,415,340	392699081.698724	392699081.698724
$m_{1,1}$	0	0	0
$m_{0,2}$	78,445,340	78539816.3397448	78539816.3397448
Moments-circle $R = 1000$	Discrete	EFD	Analytic
$m_{0,0}$	3,141,521	3141592.65358979	3141592.65358979
$m_{1,0}$	3,141,521,000	3141592653.58979	3141592653.58979
$m_{0,1}$	0	0	0
$m_{2,0}$	3,926,883,347,836	3926990816987.24	3926990816987.24
$m_{1,1}$	0	0	0
$m_{0,2}$	785,362,347,836	785398163397.448	785398163397.448

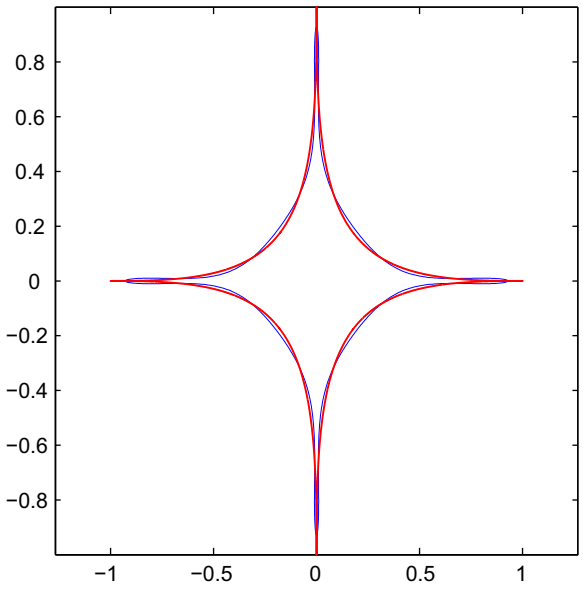

Fig. 2. A superquadric and its superimposed representation by EFD. The superquadric $x(t) = \cos^5(t)$ and $y(t) = \sin^5(t)$ is the red curve. The number of harmonics used for EFD representation is 11 i.e. $n = 0, \ldots, 10$. The EFD representation is the blue curve. (For interpretation of the colors in this figure, the reader is referred to the online version of this article.)

4.2. Comparison with Bernstein–Bézier curves approximations

In this section, we compare the accuracy of moment computations for objects modeled by EFD and Bernstein–Bézier curves. We begin with a quick overview of several basic facts.

Let $i, n \in N$. The Bernstein–Bézier ith basis function of order n is given as:

$$\theta_{i,n}(t) = \binom{n}{i}(1-t)^{n-i}t^i.$$

Consider the following Bernstein–Bézier curve (Cohen et al., 2001) $C(t) = \sum_{i=0}^{n} P_i\theta_{i,n}(t)$, where $\theta_{i,n}(t) = \binom{n}{i}(1-t)^{n-i}t^i$, and $P_i = (x_i, y_i)$ are 2D points in a plane.

In our experiments, we employ interpolation to approximate several point sets (Cohen et al., 2001). While efficient algorithms for computing moments of objects represented by Bernstein–Bézier curves can be found in (Gonzales-Ochoa et al., 1998; Soldea et al., 2000; Sheynin and Tuzikov, 2003), our goal is to compare the accuracy of the moments computations for the Bernstein–Bézier and EFD representations. We evaluate Eq. (4) in Maple, symbolically, with a precision of 100 digits.

Consider the circle in Fig. 1c. This circle can be described exactly by EFD, but not necessarily with Bernstein–Bézier boundary curves. Since a circle can be approximated by zeroth and first harmonics, i.e. $n = 0, 1$ using EFD, i.e. six coefficients, we approximated the circle with a Bernstein–Bézier of degree five, which has six control points. The exact EFD representation of a circle implies higher accuracy when computing moments. In addition, we also present moments computed for a circle approximated with a Bernstein–Bézier of degree seven, which has eight control points. Table 3 illustrates accuracy of moments computations.

Consider the case where free-form objects represented by the Bernstein–Bézier curves are approximated by EFD. Clearly, the moments computations will be more accurate for Bernstein–Bézier approximations than EFD.

4.3. Empirical accuracy performance analysis

In this section, we compare the accuracy and feasibility of EFD representations versus Bernstein–Bézier ones. We have employed a collection of 2D contours from the TOSCA repository, (Bronstein et al., 2008b and Bronstein et al., 2008a). In addition to these images, we have also used several contours generated by ourselves. We begin by resampling each contour with one thousand of points. We model the input contours with EFD contours, see Figs. 3 and 4.

The experiment consists of two parts. In the first part we compute accuracy of EFD representations. In the second one, we

Table 2
Moments for the superquadric with $n = 5$, $m = 5$, and its representation by EFD with harmonics 0,...,10.

Moments	EFD representation with 11 harmonics approximation	Exact EFD representation	Analytic equation (29)
$m_{0,0}$	0.372271912342849	0.368155389092554	0.368155389092554
$m_{1,0}$	$-1.27673732795806 \times 10^{-11}$	0	0
$m_{0,1}$	$-2.53978925869032 \times 10^{-11}$	0	0
$m_{2,0}$	0.019359500946643	0.01713744161466	0.0171374416146599
$m_{1,1}$	$-5.74767927721875 \times 10^{-13}$	0	0
$m_{0,2}$	0.019360168316938	0.01713744161466	0.0171374416146599

Table 3
Moments for the circle with radius $R = 1000$. The circle is represented in Fig. 1 (c).

Moments	Bernstein–Bézier		EFD	Analytic
	Degree 5	Degree 7		
$m_{0,0}$	3206617.96941187	3137516.28455599	3141592.65358979	3141592.65358979
$m_{1,0}$	3267858859.38165	3133557489.98957	3141592653.58979	3141592653.58979
$m_{0,1}$	0	0	0	0
$m_{2,0}$	4157670991486.19	3911259368408.53	3926990816987.24	3926990816987.24
$m_{1,1}$	0	0	0	0
$m_{0,2}$	809820556032.588	785073802696.762	785398163397.448	785398163397.448
$m_{3,0}$	$5.93227968076246 \times 10^{15}$	$5.46725703525929 \times 10^{15}$	$5.49778714378214 \times 10^{15}$	$5.49778714378214 \times 10^{15}$
$m_{2,1}$	0	0	0	0
$m_{1,2}$	816818307826222	784493267922204	785398163397448	785398163397448
$m_{0,3}$	0	0	0	0

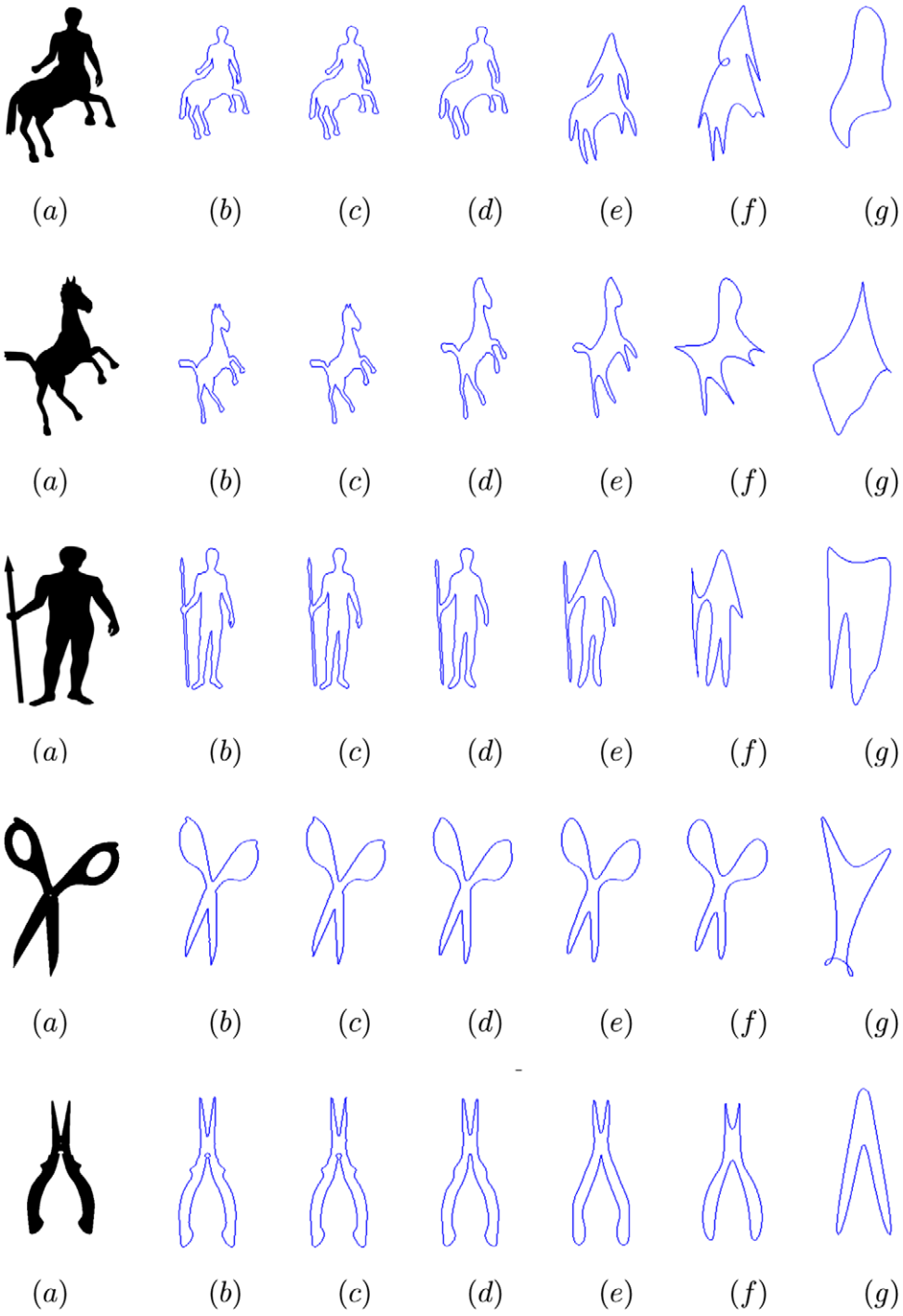

Fig. 3. Contours from TOSCA benchmark database, see (Bronstein et al., 2008b and Bronstein et al., 2008a). The images in column (a) represent originals. The (b), (c), (d), (e), (f), and (g) columns represent approximations of the original shape with 128, 64, 32, 16, 8, and 4 harmonics respectively.

evaluate the accuracy of Bernstein–Bézier representation. For clarity, we have

1. The input resampled curves are approximated with 128, 64, 32, 16, 8, and 4 harmonics, respectively, see also Fig. 3. This corresponds to a total of 130, 66, 34, 18, 10 and 8 coefficients, respectively. We refer to the moments of the most refined curve as ground truth and evaluate relative errors on moments evaluations while successively decreasing the number of harmonics used.
2. The input resampled curves are approximated with Bernstein–Bézier curves of degrees 65, 33, 17, and 9, respectively, see also Fig. 5. We refer to the moments of the most refined curve as ground truth and evaluate relative errors on moments evaluations while successively decreasing the degrees, i.e. the number of coefficients used.

We average these relative errors over all the contours and show the results in Fig. 6a and b. Fig. 6a and b show that for the range of parameters considered, the relative errors in moments computa-

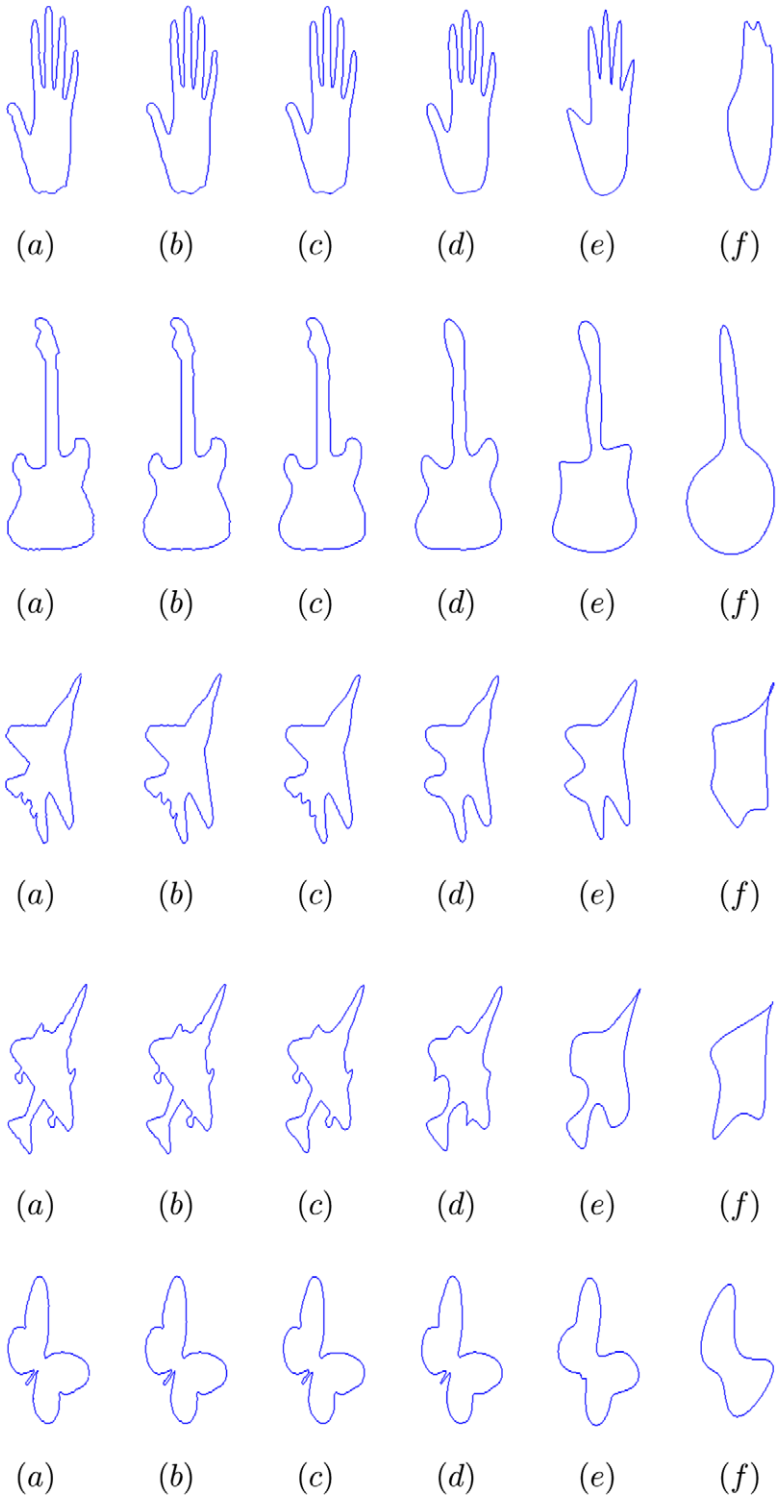

Fig. 4. Contours of several shapes that we generated. Columns (a), (b), (c), (d), (e), and (f) represent EFD contours with 128, 64, 32, 16, 8, and 4 harmonics, respectively.

tions in EFD increases a lot with harmonics reduction. However, EFD provides better overall accuracy than Bernstein–Bézier curves. Note also that a Bernstein–Bézier curve with degree 65, 33, 17, and 9, has 66, 34, 18, and 10, coefficients, respectively. These many coefficients correspond to the four coarsest EFD based representations considered here.

Both of the representations are $C^{(\infty)}$ continuous. Each one has its own advantages and drawbacks, however. The fitting of Bernstein–Bézier is a problem of modeling with polynomials. While we use control points defined from the contours, other methods such as interpolation can be employed. Polynomial interpolation, however, especially when high degree are involved in, is error prone due to Runge–Kutta phenomena, see Dahlquist and Björck (1974). On the other hand, polynomials are convenient and relatively simple tools for modeling. Unlike Bernstein–Bézier approximations, EFD contours seem to be more efficient in terms of coefficients description.

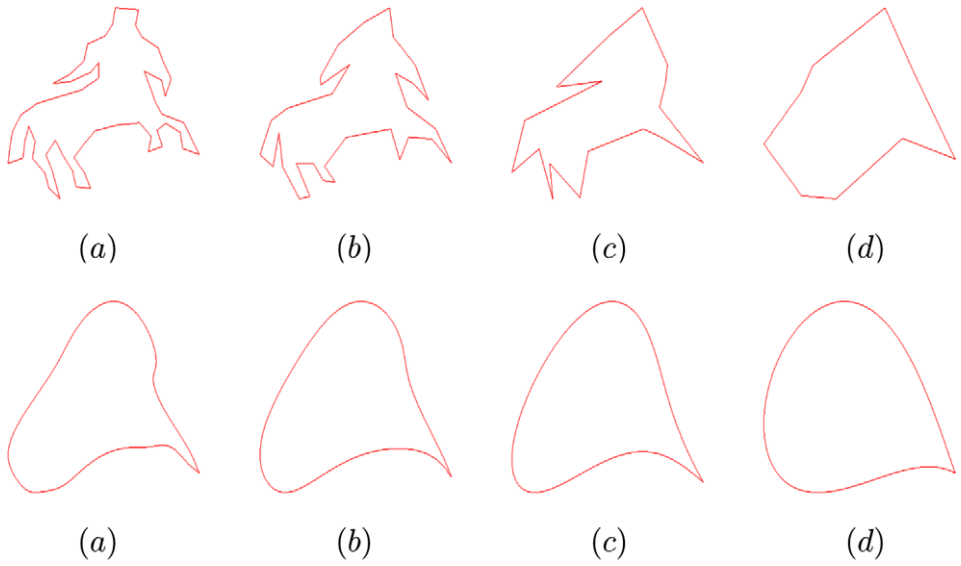

Fig. 5. Bernstein–Bézier approximations of the centaur image in Fig. 3 (a). The (a), (b), (c), and (d) columns represent Bernstein–Bézier models with 65, 33, 17, and 9 coefficients, respectively. The top line represents control points, while the bottom one shows the Bernstein–Bézier contour itself.

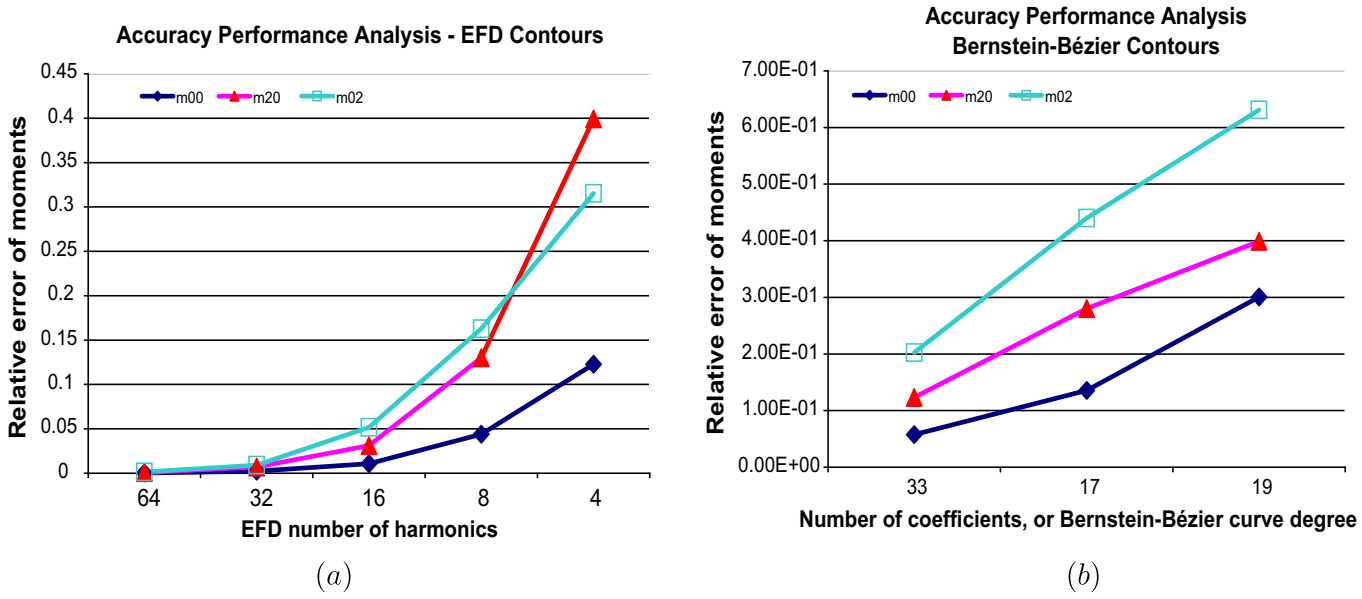

Fig. 6. Accuracy performance analysis. In (a), we show average relative errors of three moments as a function of the harmonics number of EFD representations. In (b), we show relative errors of the same moments when using Bernstein–Bézier approximations as a function of the coefficients, or equivalently, the degrees of the analyzed curves. In both of these two cases, the averages are performed over the entire set of contours (TOSCA and our).

EFD contours, as compared to Bernstein–Bézier approximations, at the same number of coefficients, provide better approximations to the shape, see for example the centaur in Figs. 3 and 5.

4.4. Empirical time performance analysis

In this section, we experimentally validate the quadratic time consumption developed in Eq. (25). We present average time consumptions when computing moments of elliptic Fourier curves that have various degrees and number of harmonics. The average is on a number of runs. We show the parabolic behaviour of Eq. (25) in Fig. 7a. We also show that the dependence on the degree of moments is relatively reduced in the range of low values, see Fig. 7(b). We finally remark that, in general, low order moments are more used than higher order ones.

5. Conclusions

We have outlined a computationally efficient scheme for calculating moments of objects represented by elliptic Fourier descriptors (EFD). The method is recursive and therefore implies fast computation of moments. To the best of our knowledge, this is the first work on moments of EFD which might trigger further works in the field. We have experimented our scheme on 2D shapes modeled by EFD and provided measurement of accuracy of our method along with comparisons with some other tech-

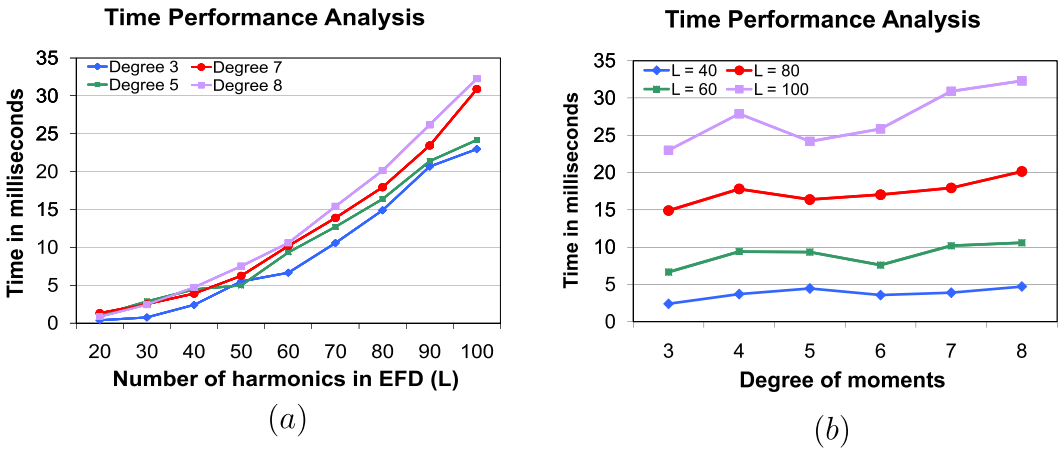

Fig. 7. Average time performance analysis. In (a), while the order of the computed moments is kept constant, the number of harmonics, i.e. $(L+1)$, increases gradually. In (b), unlike in (a), L is constant and the order of the computed moments increases. The average is measured over several computations.

niques. The method is described for 2D shapes. Extension of these ideas to 3D will be considered as a future work.

References

Bronstein, A.M., Bronstein, M.M., Bruckstein, A.M., Kimmel, R., 2008a. Analysis of two-dimensional non-rigid shapes. Internat. J. Comput. Vision 78 (1), 67–88.

Bronstein, A.M., Bronstein, M.M., Kimmel, R., 2008b. Numerical Geometry of Non-Rigid Shapes. Springer.

Cash, L.G., Hatamian, M., 1987. Optical character recognition by the method of moments. Comput. Vision Graphics Image Process. 39, 291–310.

Cohen, E., Riesenfeld, F.R., Elber, G., 2001. Geometric Modeling with Splines, An Introduction. A.K. Peters.

Dahlquist, G., Björck, Å., 1974. Numerical Methods. Prentice-Hall, Englewood Cliffs, NJ.

Fikhtengol'ts, M.G., 1965. The Fundamentals of Mathematical Analysis. Pergamon Press, New York.

Geddes, K.O., Heal, K.M., Labahn, G., Vorkoetter, S.M., McCarron, J., DeMarco, P., Monagan, M.B., 2002. Maple 8 Advanced Programming Guide. Waterloo Maple, Inc., ISBN-10: 189451128X, ISBN-13: 978-1894511285.

Gonzales-Ochoa, C., McCamnon, S., Peters, J., 1998. Computing moments of objects enclosed by piecewise polynomial surfaces. ACM Trans. Graphics 17 (3), 143–157.

Jeong, Y., Radke, J.R., 2007. Reslicing axially sampled 3d shapes using elliptic fourier descriptors. Med. Image Anal. 11 (2), 197–206.

Jirapatnakul, A.C., Reeves, A.P., Biancardi, A.M., Yankelevitz, D.F., Henschke, C.I., February 2009. Identification of asymmetric pulmonary nodule growth using a moment-based algorithm. In: Proc. SPIE, Medical Imaging, Computer-Aided Diagnosis 7260, 72602W–72602W–8.

Kuhl, P.F., Giardina, R.C., 1982. Elliptic fourier features of a closed contour. Comput. Graphics Image Process. 18 (3), 236–258.

Lestrel, E.P., 1997. Fourier Descriptors and their Applications in Biology. Cambrigde University Press.

Leu, J.-G., 1991. Computing a shape's moments from its boundary. Pattern Recognition 24 (10), 949–957.

Lin, S.C., Hwang, L.C., 1987. New forms of shape invariants from elliptic fourier descriptors. Pattern Recognition 20 (5), 535–545.

Meriam, L.J., 1966. Dynamics. John Wiley & Sons Inc., New York.

Prokop, J.R., Reeves, P.A., 1992. A survey of moment-based techniques for unoccluded object representation and recognition. CVGIP: Graphical Models Image Process. 54 (5), 438–460.

Sheynin, A.S., Tuzikov, V.A., 2003. Moment computation for objects with spline curve boundary. IEEE Trans. Pattern Anal. Machine Intell. 25 (10), 1317–1322.

Soldea, O., Elber, G., Rivlin, E., 2000. Exact and efficient computation of moments of free-form surface and trivariate based geometry. Computer-Aided Design 34 (7), 529–539.

Soldea, O., Unel, M., Ercil, A., 2010. Available from <http://www.cs.technion.ac.il/~octavian/PUBLICATIONS/PRL_App.pdf>.

Staib, H.L., Duncan, J., 1992. Boundary finding with parametrically deformable models. IEEE Trans. Pattern Anal. Machine Intell. 14 (10), 1061–1075.

Suk, T., Flusser, J., 2004. Projective moment invariants. IEEE Trans. Pattern Anal. Machine Intell. 26 (10), 1364–1367.

Wallace, P.T., Mitchell, R.O., 1980. Analysis of three-dimensional movement using fourier descriptors. IEEE Trans. Pattern Anal. Machine Intell. 2 (6), 583–588.

Wang, S., Qi, F., Li, H., 2006. Minimum description length shape model based on elliptic fourier descriptors. In: Lecture Notes in Computer Science, Advances in Neural Networks – The Third Internat. Symposium on Neural Networks, vol. 3972, pp. 646–651.

Weinstein, R., Teran, J., Fedkiw, R., 2006. Dynamic simulation of articulated rigid bodies with contact and collision. IEEE Trans. Visualization Comput. Graphics 12 (3), 265–374.

Yalcin, H., Unel, M., Wolovich, W., 2003. Implicitization of parametric curves by matrix annihilation. Internat. J. Comput. Vision 54 (1/2/3), 105–111.

Yip, K.K.R., Tam, K.P., Leung, K.N.D., 1994. Application of elliptic fourier descriptors to symmetry detection under parallel projection. IEEE Trans. Pattern Anal. Machine Intell. 16 (3), 277–286.

Zhenjiang, M., 2000. Zernike moment-based image shape analysis and its application. Pattern Recognition Lett. 21 (2), 169–177.