IRWIN AND JOAN JACOBS
CENTER FOR COMMUNICATION AND INFORMATION TECHNOLOGIES

Recognition of 3D Objects Based on Implicit Polynomials

Hilla Ben-Yaacov, David Malah, and Meir Barzohar

CCIT Report #740
August 2009

DEPARTMENT OF ELECTRICAL ENGINEERING
TECHNION - ISRAEL INSTITUTE OF TECHNOLOGY, HAIFA 32000, ISRAEL

המרכז לטכנולוגיות תקשורת ומידע
הפקולטה להנדסת חשמל
הטכניון - מכון טכנולוגי לישראל, חיפה 32000, ישראל

CCIT REPORT #740 August 2009

Recognition of 3D Objects Based on Implicit Polynomials

Hilla Ben-Yaacov, David Malah, and Meir Barzohar

Abstract

This work deals with 3D objects description and recognition by implicit polynomials (IPs). We explore the description abilities of existing 3D implicit polynomials fitting algorithms, Gradient-One, Min-Max and Min-Var, and suggest a modification for the Min-Max and Min-Var algorithms, so that they will be rotation invariant. We develop a new set of 3D rotation invariants that are linear combinations of the IP coefficients, as well as a set of 3D quadratic and angular rotation invariants, using a tensor representation of the IP. We also obtain closed-form expressions for these invariants. We then present a 3D object recognition method which is based on the Multi Order and Fitting Errors Technique (MOFET) proposed earlier for 2D object recognition. This recognition approach is based on fitting several polynomials to the object surface, each having a different degree, and on their fitting errors. We demonstrate the recognition results on both a rigid objects database and a faces database (acquired in a cooperative situation). Simulation results show that our proposed method outperforms recognition based on IP fitting after pose estimation, as well as on the Shape Spectrum Descriptor (SSD) technique, which was adopted by the MPEG-7 standard.

Index Terms

Implicit polynomials, 3D object recognition, tensor contraction, rotation invariant, 3D object fitting, face recognition.

I. Introduction

IMPLICIT polynomials (IP) are used for efficient representation of 2D curves and 3D surfaces specified by discrete data. The ability to efficiently describe complicated boundaries using the coefficients of IPs is attractive for applications in the fields of object recognition [1], [2], and might also be used as a basis for pose estimation [3].

Over time, different algorithms for fitting IPs to the data were developed. Early fitting algorithms were nonlinear and iterative [4], and often failed to represent or recognize relatively complicated objects. A more recent fitting algorithm is the 3L [5], which solves a Least Squares (LS) problem. However, this algorithm provided coefficients which were not stable enough for object-recognition purposes [6]. Another advanced algorithm [7], [8] used a topological approach in order to obtain IPs which have a simply connected zero-set. State of the art fitting algorithms are Gradient-One [6], Min-Max [9], Min-Var [9] and Ridge-Regression [6] (enhanced for the 3D case over the 3L technique in [10]), all of them apply a linear LS solution to the fitting problem as well. The main difference between them and 3L is that these algorithms have additional requirements on the fitting which improve the stability of the IP coefficients and alleviate the problem of spurious zero sets that typically appear in IP fitting with high order polynomials. Compared with older nonlinear iterative algorithms [4], the Least Squares algorithms have a much better performance in both representation and recognition tasks, with lower complexity. These representation and recognition abilities were previously tested mainly for 2D IPs [9], [11], [12]. As the main research topic of this work is 3D recognition, rather than fitting, we chose to focus on Gradient-One, Min-Max and Min-Var.

Recognition of 3D objects is important in various fields, such as medical imaging, robotics, automatic security systems etc.

Previous approaches usually require a pre-processing stage of pose estimation for the alignment of the new object representation with each of the dictionary objects representations. In many approaches, there is also a need to identify matching key points between 2 object representations in order to achieve good recognition results. These stages have high computational cost, since accurate pose estimation and matching key points are usually iterative (such as the Iterative Closest Point [13] for small angles differences, or the Tensor Voting approach [14]), and each iteration performs calculations for many data-points. Note that finding the correct correspondence between key points is very difficult, and its results are usually not accurate enough. An example for such an approach can be found in [15], which needs

a large number of spin images (2D histograms around objects surface points), from different points of view on the object surface, in order to perform surface matching for recognition. In the case of 3D face recognition, state-of-the-art methods have high complexity, since they are trying to handle a variety of situations, including large changes in face expressions, hair occlusions etc [16], [17].

In this work we first explore the description abilities of existing 3D IP fitting algorithms for both low and high degree fitting. We then use a tensor representation of IPs in order to derive a set of rotation invariants which are linear combinations of the IP coefficients, as well as quadratic and angular rotation invariants.

The IP based 3D rotation invariants are well suited for the Gradient-One fitting algorithm which is rotation invariant (i.e., the relation between IPs fitted to different orientations of the same data is characterized by rotation only). In order to apply them for recognition using other fitting algorithms, we suggest a modification of Min-Max and Min-Var fitting algorithms so that they will be rotation invariant as well. This modification is based on the 2D Rotation-Invariant Min-Max and Rotation-Invariant Min-Var [11], [12].

The recognition processes in [1] and [2] are based on nonlinear low degree (3 or 4) IP fitting algorithms, which are less stable. In addition, their object recognition performance was obtained for relatively small data-sets. In [6], the classifier is based on Gradient-One, which is more stable, but the authors use a single high degree (6 to 10) IP fitting, and results are reported for 2D object recognition only.

In this work we use a tensor representation of IPs in order to derive a set of $\left\lfloor \frac{n}{2} \right\rfloor + 1$ rotation invariants which are linear combinations of the IP coefficients, as well as $\left\lceil \frac{n}{2} \right\rceil$ quadratic and $\frac{\left\lceil \frac{n}{2} \right\rceil \left(\left\lceil \frac{n}{2} \right\rceil - 1 \right)}{2}$ angular rotation invariants, from an IP of degree n. We develop new sets of 3D IP rotation invariants (linear, quadratic and angular), as well as closed-form expressions for these invariants.

Following the 2D IP recognition method Multi Order and Fitting Error Technique (MOFET) [11], [12], we propose a classifier based on these 3D IP rotation invariants, as well as on 3D IP fitting errors, 2D IP rotation invariants and fitting errors (from the most descriptive 2D projections) and the eigenvalues of a Principle Component Analysis (PCA) decomposition. The suggested classifier uses various IP degrees (both for 2D and 3D) in order to utilize both the stability of low degree IPs and the descriptiveness of high degree IPs. Our proposed recognition approach is model based, as the classifier features are combinations of IP coefficients, and the IP is a parametric representation of the object surface. We explore the results of our classifier using Gradient-One, Rotation-Invariant Min-Max and Rotation-Invariant Min-Var fitting

algorithms. We apply the proposed method to both a rigid objects database and a faces database (in a cooperative situation) and compare our recognition results with pose estimation methods followed by IP fitting [3] and with the Shape Spectrum Descriptor (SSD) technique, which was adopted by the MPEG-7 standard for 3D descriptors [18], [19].

This paper is organized as follows: Section II provides background information on IPs fitting algorithms, and presents a rotation invariant modification of the Min-Max and Min-Var algorithms. Section III describes the process of rotation invariants derivation. Section IV deals with feature extraction for 3D object recognition. We describe the objects and faces databases. We discuss the pre-processing stages and the features selection for the classifier. Section V describes the classifier design (learning and testing) for both the objects and the faces databases. Section VI describes our recognition results and compares them with other recognition methods. Section VII summarizes the work and suggests research directions for future study.

II. Background

A. Fitting 3D Implicit Polynomials to Surfaces

In this section, we provide a brief overview of IP fitting and of the Gradient-One, Min-Max and Min-Var fitting algorithms.

The objective of IP fitting is to describe data points by the zero-set of an IP. That is, the value of the IP should be zero at the location of the data points.

The technique assumes that the data points are part of a zero set of an IP of degree n (i.e., $P^n(x,y,z) = 0$, where $P^n(x,y,z)$ is an IP of degree n). Therefore, the first and most intuitive requirement that the IP should satisfy is that its value should be zero at the location of the data points.

The output of the fitting process is a 3D IP $P^n_{\underline{a}}(x,y,z)$:

$$\begin{aligned} P^n_{\underline{a}}(x,y,z) = \; & a_{000} + a_{100}x + a_{010}y + a_{001}z + \\ & + a_{200}x^2 + a_{110}xy + a_{020}y^2 + \\ & + a_{101}xz + a_{011}yz + a_{002}z^2 + ... + \\ & + a_{n00}x^n + ... + a_{0n0}y^n + ... + a_{00n}z^n \end{aligned} \tag{1}$$

where a_{klm} is the coefficient of the monomial with x to the power of k, y to the power of l and z to the power of m, and $k, l, m = 0, 1, ..., n$.

The IP is defined by its coefficients vector $\underline{a}$:

$$\begin{aligned}\underline{a} = \quad & [a_{000}\ a_{100}\ a_{010}\ a_{001}\ a_{200}\ a_{110} \\ & a_{020}\ a_{101}\ a_{011}\ a_{002}\ ...\ a_{n00}\ ...\ a_{0n0}\ ...\ a_{00n}]^T\end{aligned} \tag{2}$$

and its monomials vector $\underline{p}^n(x,y,z)$:

$$\underline{p}^n(x,y,z) = \left[1\ x\ y\ z\ x^2\ xy\ y^2\ xz\ yz\ z^2\ ...\ x^n\ ...\ y^n\ ...\ z^n\right]^T \tag{3}$$

and can be also written as:

$$P^n_{\underline{a}}(x,y,z) = \underline{a}^T \underline{p}^n(x,y,z) \tag{4}$$

The dimension of $\underline{a}$, as well as the dimension of $\underline{p}^n(x,y,z)$, is $\frac{(n+1)(n+2)(n+3)}{2\cdot 3}$.

The fitting problem definition is therefore to find a coefficients vector $\underline{a}$ that defines the IP which best fits the data under certain criteria.

An example of a 3D object and an IP describing it is shown in Fig. 1.

1) Overview of the Gradient-One Algorithm: The Gradient-One fitting algorithm is presented in [6].

The first fitting criterion is the demand that the zero set of the IP will fit the data-set.

The Gradient-One algorithm requires one more fitting criterion: the IP gradient at the location of each data point needs to be in the same direction as the normal vector of the data set in the same location. In addition, the algorithm requires that the value of the gradient will be the same (and w.l.o.g. equal to 1) at all the data-set points.

Using the same notation as in [9], we can formulate the fitting criteria as:

$$M\underline{a} = \underline{b}. \tag{5}$$

where M and b define the fitting requirements (see [9] for more details). If we formulate it as a minimization problem of the l_2 squared norm of the fitting error vector we get:

$$\min_{\underline{a}} \|M\underline{a} - \underline{b}\|^2 \tag{6}$$

which is a linear least squares problem with a known solution when $M^T M$ is not singular:

$$\underline{a}_{LS} = \left(M^T M\right)^{-1} M^T \underline{b} \tag{7}$$

2) Overview of the Min-Max and Min-Var Algorithms: Although Gradient-One has good fitting properties, it still suffers from zero-set stability problems. An IP may have several zero-sets, and only one of them is describing the data-set. The zero-sets which do not describe the data-set are called *spurious zero-sets*. When using the Gradient-One fitting algorithm, the spurious zero-sets are sometimes very close to the desired zero-set (which describes the data-set). Consequently, if the data points are perturbed, these external zero-sets might become too close, and in extreme cases - even intersect the desired zero-set.

The Min-Max and Min-Var fitting algorithms are presented in [9], [20].

These improved algorithms are based on analysis of the sensitivity of zero-set points to small changes in the IP coefficients.

The Min-Max fitting algorithm has three fitting requirements:

1) The zero set of the IP should fit the data-set.
2) The IP gradient at the location of each data point should be in the same direction as the normal vector of the data set in the same location.
3) Minimize the maximum error.

Similarly, the Min-Var fitting algorithm has three fitting requirements: the first two are the same as in Min-Max, and the third is to minimize the variance of the error (instead of the maximum error as in Min-Max).

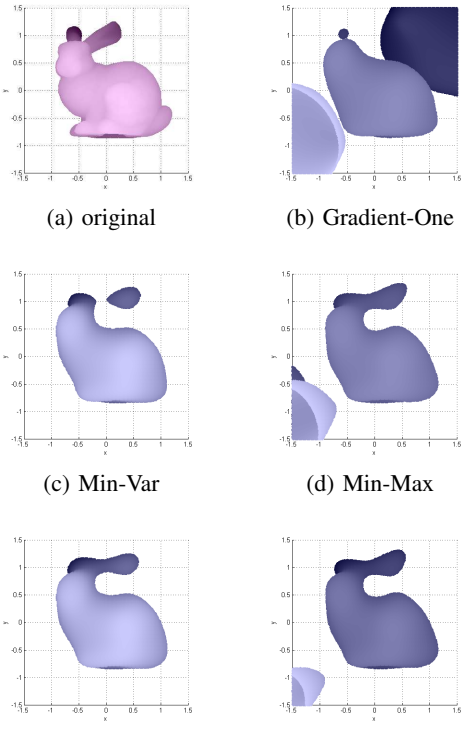
This third requirement for the Min-Max/Min-Var modifies the least squares matrix M (see Table 2 in [9] for more details), but we can still formulate the fitting problem as in (6), and get the least squares solution of the same form as in (7).

A comparison between Gradient-One, Min-Max and Min-Var fitting is shown in Fig. 1.

Based on these observations we can state that compared with Gradient-One, Min-Max and Min-Var spurious zero-sets appear farther away from the desired zero-set.

B. Rotation Invariant Fitting Algorithms

For the application of object recognition, we would like that the relation between IPs fitted to the original and the rotated data-sets will be rotation only. In other words, if we fit an IP to 2 different orientations of the same object, we would like that each fitting requirement won't be affected by the rotation angles. Gradient-One is already rotation invariant [6], [22] and therefore it does not need any adjustment. As Min-Max and Min-Var have an advantage of better fitting, it is of interest to examine their



(a) original (b) Gradient-One

(c) Min-Var (d) Min-Max

(e) RI-Min-Var (f) RI-Min-Max

Fig. 1. Fit of a 6^{th} degree IP to "The Stanford Bunny" (taken from "The Stanford 3D Scanning Repository" [21]): (a) original data points (b) Gradient-One fitting (c) Min-Var fitting (d) Min-Max fitting (e) RI-Min-Var fitting (f) RI-Min-Max fitting. Note that several zero-sets appear in the image, the central one describes the object and the rest are spurious.

recognition properties as well. Therefore, we would like to adjust the fitting requirements of Min-Max and Min-Var, so that for each data point they will not be affected by rotation.

Similarly to what was done for the 2D case in [11], [12], the only requirement that needs adjustment is the requirement regarding the gradient size. The full explanation can be found in [23], here we present only the adjusted Min-Max and Min-Var. For Min-Max, similarly to the 2D case [11], [12], we transform into spherical coordinates:

$(x, y, z) \rightarrow (rcos\theta sin\varphi, rsin\theta sin\varphi, rcos\varphi)$,

and replace the gradient value requirement with the following requirement:

$$\left\|\nabla P_{\underline{a}}^{n}\left(rcos\theta sin\varphi, rsin\theta sin\varphi, rcos\varphi\right)\right\|_{l_2} = \\ = E_{\theta,\varphi}\left\{\left\|\underline{p}^{n}\left(rcos\theta sin\varphi, rsin\theta sin\varphi, rcos\varphi\right)\right\|_{l_1}\right\}$$

where $0 \leq \theta < 2\pi$ and $0 \leq \varphi < \pi$, and $E_{\theta,\varphi}\{\cdot\}$ is the expectation with respect to θ and φ.

Now the sum of monomials is averaged over all the possible values of coordinates (θ, φ) and the gradient value requirement is invariant to rotation.

For Min-Var, similarly to the 2D case [11], [12], we replace each monomial $x^k y^l z^m$ with $\sqrt{\frac{n!}{k!l!m!}} x^k y^l z^m$.

In other words, the adjustment we do is the following:

$$\begin{aligned} P_{\underline{a}}^{n}(x,y,z) &= \sum_{0\leq k,l,m,k+l+m\leq n} a_{klm}x^k y^l z^m = \\ &= \sum_{0\leq k,l,m,k+l+m\leq n} \underbrace{\frac{a_{klm}}{\sqrt{\frac{n!}{k!l!m!}}}}_{a'_{klm}} \sqrt{\tfrac{n!}{k!l!m!}}x^k y^l z^m = \\ &= \sum_{0\leq k,l,m,k+l+m\leq n} a'_{klm}\sqrt{\tfrac{n!}{k!l!m!}}x^k y^l z^m \end{aligned} \tag{8}$$

Actually we create a new factorized monomials vector, and solve the least squares problem for $\underline{a}'$. After we have $\underline{a}'$, we can transform back to our original coefficients vector $\underline{a}$.

The use of the average value for Min-Max or the factorized monomials vector for Min-Var hardly affects the Min-Max/Min-Var properties, respectively. A comparison of the fitting performance appears in Fig. 1. As it can be seen, the results of Min-Max and the Rotation-Invariant Min-Max (RI-Min-Max) are quite similar, as well as the results of Min-Var and the rotation invariant Min-Var (RI-Min-Var).

III. Rotation Invariant Expressions for 3D Implicit Polynomials

In order to avoid pose estimation in the recognition process, we need to use expressions that will be invariant to rotation. Such IP based expressions have been developed for the 2D case, both analytically [24], and using symbolic computation [1], [25]. Additional IP based geometric invariants were developed for the 2D case in [26], [27]. However, for the 3D case, the IP based rotation invariants were developed by using symbolic computation [1] only. These invariants contain sum of high degree products of IP coefficients. These products may cause instability due to their high degree, and therefore we prefer the analytically derived invariants which we will develop in this section for the 3D case, extending those developed for the 2D case in [24].

A. Separation of an IP into Forms

A 3D IP:

$$f_n(x,y,z) = \sum_{0\leq k,l,m,k+l+m\leq n} a_{klm}x^k y^l z^m \tag{9}$$

can be separated into homogeneous parts in the following way:

$$\begin{aligned} f_n(x,y,z) \;=\; & \\ & + \underbrace{a_{000}}_{H_0} + \\ & + \underbrace{a_{100}x + a_{010}y + a_{001}z}_{H_1(x,y,z)} + \\ & + ... + \\ & + \underbrace{a_{n00}x^n + ... + a_{0n0}y^n + ... + a_{00n}z^n}_{H_n(x,y,z)} \end{aligned} \tag{10}$$

where $H_r(x,y,z)$ denotes a homogeneous ternary IP of degree r, also called a *'form'* of degree r:

$$H_r(x,y,z) = \sum_{k+l+m=r} a_{klm} x^k y^l z^m \tag{11}$$

$H_n(x,y,z)$ is called *'the leading form'* of an IP of degree *n*. Note that translations leave the leading form unaffected, but affect the rest of the forms [3].

B. Derivation of Rotation Invariants Based on Tensor Representation

Our proposed method for 3D rotation invariants derivation, is based on the pose estimation method of [3]. In this section, we use the notation (x_1, x_2, x_3), instead of (x, y, z), for convenience.

We start by briefly reviewing the tensor representation suggested in [3] for pose estimation, and we will then use this method for developing 3D rotation invariants.

For pose estimation purposes [3], it is more convenient to use a tensor representation for each form:

$$H_n(x_1,x_2,x_3) = \sum_{i_1=1}^{3}\sum_{i_2=1}^{3}...\sum_{i_n=1}^{3} s^{i_1 i_2 ... i_n} x_{i_1} x_{i_2} ... x_{i_n} \tag{12}$$

where $(s^{i_1 i_2 ... i_n})_{1 \le i_1, i_2, ..., i_n \le 3}$ is a symmetric tensor of order *n*, denoted S_n. It can also be considered as an *n*-dimensional array with 3^n entries. Each entry $s^{i_1 i_2 ... i_n}$ can be expressed by the form coefficients:

$$s^{i_1 i_2 ... i_n} = \frac{a_{klm}}{\frac{n!}{k!l!m!}} \tag{13}$$

where *k* is the number of tensor indices $(i_1, i_2, ..., i_n)$ that are equal to 1, *l* is the number of tensor indices that are equal to 2, and *m* is number of tensor indices that are equal to 3.

Given a tensor, a contraction with respect to 2 indices (e.g., i_1 and i_2), is defined as a new tensor of

order $n-2$:

$$s'^{i_3 i_4 \ldots i_n} = \sum_{i_1=1}^{3} s^{i_1 i_1 i_3 i_4 \ldots i_n} \tag{14}$$

In other words, we set $i_2 = i_1$, and sum over i_1. A total contraction of an even degree tensor gives a zero order tensor (a scalar) which is an invariant.

For example, for a symmetric 3x3 matrix (tensor of order 2), the tensor contraction gives the trace of the matrix, which is known to be an invariant under Euclidean transformations (such as rotation).

The IP based pose estimation suggested in [3] fits an even degree IP with $n = 2p$ and extracts only the leading form ($H_n(x, y, z)$) since this form is invariant to translation. In order to find the intrinsic orientation of the IP, it performs $(p-1)$ tensor contractions on the leading form, resulting in a 3x3 matrix denoted by $V(H_n)$, whose elements are linear combinations of the leading form coefficients.

Now, after the brief review of [3], we explain how to develop a novel set of closed-form 3D rotation invariant expressions. We suggest to add a p^{th} tensor contraction (i.e., a total of p contractions for an IP of degree $n = 2p$). This will result in a tensor of order 0 (a scalar) which is the trace of $V(H_n)$.

For example, when using a 2^{nd} degree form we get:

$$\begin{aligned} H_2(x_1, x_2, x_3) &= a_{200}x_1^2 + a_{020}x_2^2 + a_{002}x_3^2 + \\ &+ a_{110}x_1x_2 + a_{101}x_1x_3 + a_{011}x_2x_3 \end{aligned} \tag{15}$$

and:

$$V(H_2) = \begin{bmatrix} a_{200} & \frac{a_{110}}{2} & \frac{a_{101}}{2} \\ \frac{a_{110}}{2} & a_{020} & \frac{a_{011}}{2} \\ \frac{a_{101}}{2} & \frac{a_{011}}{2} & a_{002} \end{bmatrix} \tag{16}$$

We denote the 3D linear invariants by $L_{3D,k}$, where k is the form degree. The rotation invariant we get is

$$L_{3D,2} = trace\left[V(H_2)\right] = a_{200} + a_{020} + a_{002}, \tag{17}$$

The advantage of this method, is the simplicity in which we can derive an invariant from any even degree form, even for high degrees. For example, for a 4^{th} degree form, the rotation invariant we get is

$$\begin{aligned} L_{3D,4} &= trace\left[V(H_4)\right] = \\ &= a_{400} + a_{040} + a_{004} + \tfrac{1}{3}\left(a_{220} + a_{202} + a_{022}\right) \end{aligned} \tag{18}$$

In the general case, of a form of degree $n = 2p$, we developed a general expression for the linear invariant obtained by p tensor contractions:

$$\begin{aligned} &L_{3D,n} = trace\left[V\left(H_n\right)\right] = \\ &= \sum_{i_{n-1}=1}^{3} \cdots \sum_{i_3=1}^{3} \sum_{i_1=1}^{3} s^{i_1 i_1 i_3 i_3 \cdots i_{n-1} i_{n-1}} \end{aligned} \tag{19}$$

and using (13)) we get:

$$\begin{aligned} &L_{3D,n} = \\ &= \sum_{k,l,m\ even, k+l+m=n} \frac{k!l!m!}{n!} \cdot \frac{(n/2)!}{(k/2)!(l/2)!(m/2)!} \cdot a_{klm} \end{aligned} \tag{20}$$

where $\frac{(n/2)!}{(k/2)!(l/2)!(m/2)!}$ is the number of times each tensor element participates in the tensor contractions (the indexes are all divided by 2 since each pair of indexes is set to be equal in the contraction process). Using the tensor approach, we can derive the complete set of $\left\lfloor \frac{n}{2} \right\rfloor + 1$ linear invariants for an IP of degree n, one linear invariant for each even degree form.

We now examine odd degree forms. Each odd degree form $n = 2p + 1$, when represented as a tensor, can be contracted p times until we get a 1^{st} degree tensor (a vector). The squared magnitude of this vector is invariant under rotation. In the same way, the relative angles between these vectors (derived from different forms of the same IP) are also invariant under rotation.

For a 1^{st} degree form:

$$H_1\left(x_1, x_2, x_3\right) = a_{100}x_1 + a_{010}x_2 + a_{001}x_3 \tag{21}$$

we get the tensor representation (no contraction is needed, $p = 0$):

$$V\left(H_1\right) = \left[a_{100}\ a_{010}\ a_{001}\right]^T \tag{22}$$

and the squared magnitude of this vector is:

$$Q_{3D,1} = \left\|V\left(H_1\right)\right\|_{l_2}^2 = a_{100}^2 + a_{010}^2 + a_{001}^2 \tag{23}$$

We denote the 3D quadratic invariants by $Q_{3D,k}$, where k is the form degree. For a 3^{rd} degree form we

get:

$$\begin{aligned} Q_{3D,3} &= \|V(H_3)\|_{l_2}^2 = \\ &= \left[a_{300} + \tfrac{a_{120}}{3} + \tfrac{a_{102}}{3}\right]^2 + \\ &+ \left[a_{030} + \tfrac{a_{210}}{3} + \tfrac{a_{012}}{3}\right]^2 + \\ &+ \left[a_{003} + \tfrac{a_{201}}{3} + \tfrac{a_{021}}{3}\right]^2 \end{aligned} \tag{24}$$

And the angle between a 1^{st} degree form and a 3^{rd} degree form will be:

$$V(H_1) \cdot V(H_3) = \|V(H_1)\|_{l_2} \|V(H_3)\|_{l_2} cos\theta_{13} \tag{25}$$

$$\theta_{13} = cos^{-1}\left[\frac{V(H_1) \cdot V(H_3)}{\|V(H_1)\|_{l_2} \|V(H_3)\|_{l_2}}\right] \tag{26}$$

where $\{\cdot\}$ is the dot product between 2 vectors.

For an n^{th} degree form ($n = 2p+1$), we get after p contractions:

$$\begin{aligned} &V(H_n) = \\ &= \begin{bmatrix} \sum\limits_{k,l,m\ even,\ k+1+l+m=n} \frac{a_{k+1,l,m}}{\frac{n!}{(k+1)!l!m!}} \frac{((n-1)/2)!}{(k/2)!(l/2)!(m/2)!} \\ \sum\limits_{k,l,m\ even,\ k+l+1+m=n} \frac{a_{k,l+1,m}}{\frac{n!}{k!(l+1)!m!}} \frac{((n-1)/2)!}{(k/2)!(l/2)!(m/2)!} \\ \sum\limits_{k,l,m\ even,\ k+l+m+1=n} \frac{a_{k,l,m+1}}{\frac{n!}{k!l!(m+1)!}} \frac{((n-1)/2)!}{(k/2)!(l/2)!(m/2)!} \end{bmatrix} \end{aligned} \tag{27}$$

and:

$$Q_{3D,n} = \|V(H_n)\|_{l_2}^2 \tag{28}$$

Using tensor representation we get $\left\lceil \frac{n}{2} \right\rceil$ quadratic invariants from an n^{th} degree IP, one for each odd degree form. Therefore, we can also derive $\frac{\left\lceil \frac{n}{2} \right\rceil \left(\left\lceil \frac{n}{2} \right\rceil - 1 \right)}{2}$ angular invariants (one between each possible pair of odd degree forms).

Note that these are explicit expressions for the derivation of 3D invariants, unlike the recursive derivation method given in [24] for 2D invariants. We can extend our explicit 3D invariants expression for the 2D invariants computation as well, in the following way:

$$L_{2D,n} = \sum_{k,l\ even, k+l=n} \frac{k!l!}{n!} \cdot \frac{(n/2)!}{(k/2)!\,(l/2)!} \cdot a_{kl} \tag{29}$$

and in a similar way for the quadratic and angular invariants. In [24], the suggested recursive scheme

includes the calculations of the same binomial coefficients as in our proposed method, but in addition, in [24], it is also needed to invert a matrix of $\frac{(n+1)(n+2)}{2}$ on $\frac{(n+1)(n+2)}{2}$ for the derivation of the invariants of a form of degree n. Thus, we need less computations than the scheme given in [24] for the 2D linear invariants. Note however, that if we apply this method in the 2D case, we get the same set of linear invariants, but only part of the quadratic/angular sets described in [24].

C. Derivation of Rotation Invariants Based on Trigonometric Expressions

In addition to the previous derivation approach, we now present another derivation method, useful mainly when dealing with low degree IPs.

Every 3D rotation can be considered as 3 rotations, one around each axis (*x*, *y* and *z*) with angles α, β and γ respectively [28].

One of the 2D invariants derivation methods described in [24] uses the 2D rotation matrix. Following the same approach in the 3D case, will require to prove that the expressions are invariant under each of the above rotations.

For example, for a 2^{nd} degree IP the original polynomial is:

$$\begin{aligned}
&f_2(x,y,z) = \\
&= a_{000} + a_{100}x + a_{010}y + a_{001}z + a_{200}x^2 + \\
&+ a_{110}xy + a_{101}xz + a_{020}y^2 + a_{011}yz + a_{002}z^2
\end{aligned}$$

After substituting the relations between x, y, z and x', y', z' for a rotation around z axis, and rearranging, we get:

$$\begin{aligned}
&f_2(x',y',z') = \\
&= \underbrace{a_{000}}_{b_{000}} + \underbrace{\left(ca_{100} - sa_{010}\right)}_{b_{100}}x' + \underbrace{\left(sa_{100} + ca_{010}\right)}_{b_{010}}y' + \underbrace{a_{001}}_{b_{001}}z' + \\
&+ \underbrace{\left(c^2a_{200} - csa_{110} + s^2a_{020}\right)}_{b_{200}}x'^2 + \\
&+ \underbrace{\left(s^2a_{200} + csa_{110} + c^2a_{020}\right)}_{b_{020}}y'^2 + \\
&+ \underbrace{\left(2csa_{200} - 2csa_{020} + \left(c^2 - s^2\right)a_{110}\right)}_{b_{110}}x'y' + \\
&+ \underbrace{\left(ca_{101} - sa_{011}\right)}_{b_{101}}x'z' + \underbrace{\left(sa_{101} + ca_{011}\right)}_{b_{011}}y'z' + \underbrace{a_{002}}_{b_{002}}z'^2
\end{aligned}$$

where $c \triangleq \cos\gamma$ and $s \triangleq \sin\gamma$. Obviously, we have the following invariant:

$$L_{3D,0} = b_{000} = a_{000} \tag{30}$$

From examination of the new IP coefficients b_{klm} and some trigonometric identities, we find that after z axis rotation we also get that:

$$L_{3D,2} = b_{200} + b_{020} + b_{002} = a_{200} + a_{020} + a_{002} \tag{31}$$

If we follow the same procedure for each of the rotation matrices, we'll see that (30) and (31) are still invariants. this is the same linear invariant we got using the previous suggested derivation method in (17). The complexity of the trigonometric expressions grows considerably with the IP degree, and therefore the rotation matrix together with the trigonometric identities are useful only for low degree IPs. We used this method for deriving 2 more quadratic invariants, the first appears in (23), and the second is (for the full proof see Appendix D of [23]):

$$\begin{aligned} Q_{3D,2} &= b_{200}^2 + b_{020}^2 + b_{002}^2 - 2b_{200}b_{020} - 2b_{200}b_{002} - \\ &-2b_{020}b_{002} + b_{110}^2 + b_{101}^2 + b_{011}^2 = \\ &= a_{200}^2 + a_{020}^2 + a_{002}^2 - 2a_{200}a_{020} - 2a_{200}a_{002} - \\ &-2a_{020}a_{002} + a_{110}^2 + a_{101}^2 + a_{011}^2 \end{aligned} \tag{32}$$

Note that the invariant in (32) cannot be obtained using the tensor method.

IV. Feature Extraction

A. Databases

The databases which we found available for 3D object recognition performance evaluation are either synthetic databases (i.e., computer graphic models, such as [29]) or small real databases (only a relatively small number of different objects, such as [30] or [31]). Therefore, we created a new objects database, which includes real acquisitions of 40 objects, each in 9 different viewing angles.

1) Rigid Objects Database: This database was acquired using the equipment of the Geometric Image Processing (GIP) lab at the Computer Science Faculty of the Technion.

The system is based on the 'structured light' technique [32]. The projector's hardware is manipulated so that it triggers the camera. The camera frame rate is 30Hz, and each 3D image is produced from a sequence

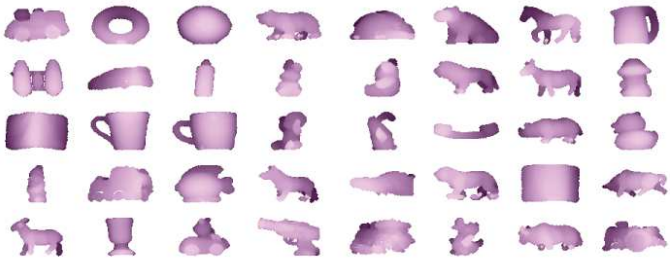


Fig. 2. one position of each of the 40 rigid objects in the database

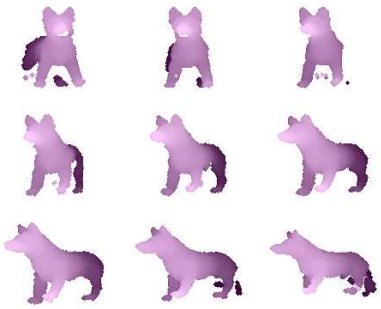


Fig. 3. 9 different positions of the object 'fox'

of 12 consecutive camera frames of 12 different projected patterns. Therefore, the system acquires 2.5 images per second. For the acquisition, the object is placed on a stand, with a black background behind it. After the acquisition, each 3D frame is re-sampled on a uniform *xy* grid of 320x240 pixels, then filtered in order to smooth noise artifacts, and cropped so that it won't contain background elements. The final number of data-points per object in our database is $\sim 12,000$ on average. Each pixel has 3 coordinates (x, y, z) and represents a point in the 3D space.

We acquired 40 different objects, each was placed on the rotatable stand and acquired in 9 consecutive positions. The difference between 2 consecutive positions is around $10^o - 15^o$ degrees, and we acquired 5 frames in each position of each object. A possible application for such a setup is a factory production line that contains various products which need automatic sorting. The database objects appear in Fig. 2. 9 different positions of the object 'fox' are shown in Fig. 3.

2) Faces Database: This database is also from the Geometric Image Processing (GIP) lab. The GIP lab gathered acquisitions of many human volunteers using the same system described in the previous section (same frame rate and resolution). The subjects are generally told to look straight at the camera and hold still, since precision is much higher for stationary objects. The result for each subject is a video of $\sim 50-70$ 3D frames, during which the subject usually moves his head a little and changes his expression

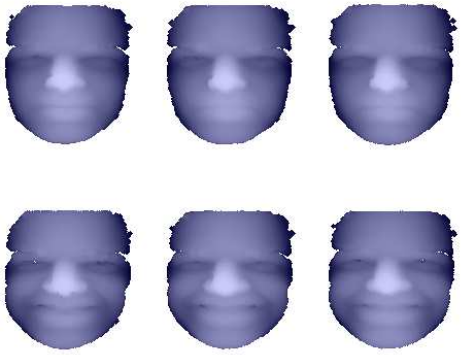


Fig. 4. 6 different frames of the same face

from neutral to happy. Since the expressions are usually moderate, and the subject is looking directly at the camera, this is a cooperative situation. In other words - the subject would like the system to correctly identify him/her. This application is interesting, for example, for identification at an entrance to restricted facilities. An example for 6 different frames of the same face is shown in Fig. 4.

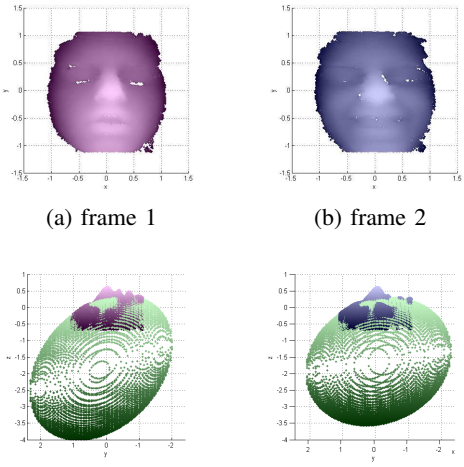
B. Pre-Processing

In order to create a feature vector for the classifier, we have to apply some pre-processing to each object/face frames. The input to this stage is a cloud of data-points $\{(x_i, y_i, z_i)\}|_{i=1,...,N}$.

1) Translation: Consider that we have a data-set with a center-of-mass point at (x_{cm}, y_{cm}, z_{cm}), and that the maximum distance between (x_{cm}, y_{cm}, z_{cm}) and the data-points is d. If the coordinates (x_{cm}, y_{cm}, z_{cm}) are much larger than d, then the monomials belonging to the different points of the data-set wouldn't differ much. Thus, the corresponding rows of the matrix M would be almost the same, and it will be close to singular. Therefore, a reasonable choice of the center-of-mass of the data-set location would be the origin.

Thus, for translation invariance, we locate the center of mass of the data-points at the origin. Note, that the center-of-mass point is invariant to scaling and rotation around the origin, and therefore it can be handled independently of these stages.

2) Scaling: In the 2D case [11], [12] it was shown that the scaling factor affects the stability of the IP coefficients. If the object is very small, so that the coordinates of the data-set points are very close to zero, then the higher degree monomials will be much smaller than the lower degree ones. If the object is very big, so that the coordinates of the data-set points are much larger than 1, then the higher degree



(a) frame 1 (b) frame 2

(c) IP fit, frame 1 (d) IP fit, frame 2

Fig. 5. An example for different 2^{nd} degree IP fitting (using Gradient-One) of 2 different scans of the same face: (a) neutral expression (b) smiling expression (c) a 2^{nd} degree IP fit to the neutral face (side view, in green) (d) a 2^{nd} degree IP fit to the smiling face (side view, in green)

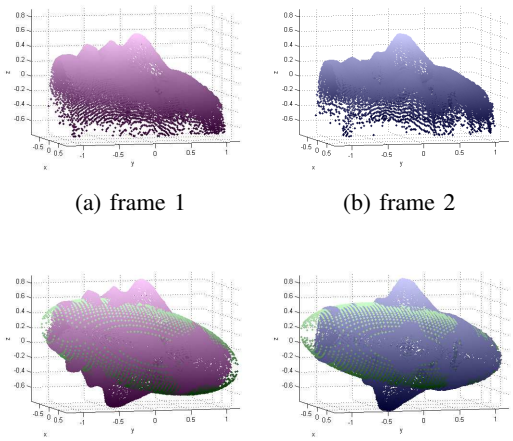
monomials will be much larger than the lower degree ones. The 2D analysis of these cases shows that such scaling factors causes instability in the IP coefficients. Thus, scaling the data-set so that its coordinates will be close to ± 1 may provide an IP with more stable coefficients in the presence of the data-set noise.

The 2D conclusion is applicable also in the 3D case, and so we would like our surface to have coordinates close to ± 1. Applying scaling so that *all* the points will have coordinates in the range $[-1, 1]$ is too sensitive to outliers. As in [11], [12], we chose our scaling factor to be the 75^{th} percentile of the distances of the data-points from the origin. In other words, we find the distances of all the data-points from the origin, sort them and choose the element that is larger than 75% of the other elements as the scaling factor $S_{75\%}$. We then scale each coordinate using this scaling factor.

3) Stabilization of IP Fitting Using Mirroring: Since a 3D face is an open surface, the IP that we fit to it is unconstrained in areas in which there are no data-points. An example is shown in Fig. 5, where a 2^{nd} degree IP fit to a face results in an ellipsoid much larger than the face. The second instance of the face (the smiling expression) has a different ellipsoid fit even though the difference between the 2 acquisitions is only a slight deformation.

In order to get a stable IP fitting, we suggest to mirror each face in the following way:

We fit a 1^{st} degree IP to the face and place the resulting plane so that it passes through the origin (the center of mass of the face). Then we lower it further in the z direction by an empiric distance (-0.25 for faces after the scaling stage). We dispose of the points below the plane and mirror the rest of the points with respect to the plane using some simple linear operations [23].



(a) frame 1 (b) frame 2

(c) Mirroring, frame 1 (d) Mirroring, frame 2

Fig. 6. An example for the similar 2^{nd} degree IP fitting (using Gradient-One) of 2 different scans of the same face when using mirroring: (a) neutral expression (b) smiling expression (c) mirroring of the neutral face and its 2^{nd} degree IP fit (in green) (d) mirroring of the smiling face and its 2^{nd} degree IP fit (in green)

The mirroring results are shown in Fig. 6. It can be seen that the ellipsoid is now very similar between the 2 occurrences of the same face, despite the smile deformation. In addition, after the mirroring, the IP is better constrained, and the ellipsoid size corresponds to the face size, unlike the large ellipsoid we got before the mirroring (Fig. 5). Thus, we get a very similar IP fit for both face scans. A similar IP fit means similar invariants, and therefore better recognition.

Although the rigid objects may suffer from the same instability, we don't apply mirroring in their case. The reason is that they are a heterogeneous group, and the desired location of the mirroring plane varies considerably from one object to another.

4) 2D Projections: In order to obtain more features using IPs, we used also 2D projections. We used only projections on the main planes (xy, xz and yz) in order to avoid high complexity calculations. If we examine 2D projections that are obtained from several different orientations of a 3D object (see for example Fig. 7), they are connected by a non-linear projective transform. However, since the viewing angle differences between consecutive orientations are relatively small, there are two approaches that can be used: The first, to consider the projective transform as an affine transform, and transform each projection into its "mother-shape", as described in [11], [12]. The disadvantage of this approach is that when we deal with very similar objects, like faces, along with affine invariance we also lose some of the object properties that distinguishes it from other similar objects. The second approach is to simply consider this transform as some additional model error that was added to the contour, and perform the IP fitting and invariants calculation without transforming it into its "mother-shape". In our experiments, the

Fig. 7. xy projections of 3 different positions of the object 'fox'

Fig. 8. 3 projections, from left to right, on xy, xz and yz, respectively, of a mirrored face. The xy projection was excluded from our face recognition scheme due to its high similarity between different faces.

second approach has shown better results, and so these are the results which we demonstrate in Section VI.

For the rigid objects, the most descriptive projection is the projection on xy. The two other projections (xz and yz) were not very informative, and had very noisy contours. After the 2D projection, we used morphological operators for holes filling and then extracted the 2D contour. An example for xy projection of 3 different positions of the object 'fox' is shown in Fig. 7.

In the case of faces, the *xy* projection is very similar for different faces, and the more descriptive projections are the projections on *xz* and *yz*. We perform the 2D projections after the mirroring of the faces, so that they won't have noisy contours as in the objects database. After each 2D projection, we used morphological operators for holes filling and then extracted the 2D contour. An example for the 3 projections of a face is shown in Fig. 8.

C. Selected Features

1) Linear Invariants: After all the pre-processing stages, we fit IPs of degrees 2, 4, 6 and 8 to the 3D objects and faces, and to their 2D projections (*xy* projection for the objects, and *xz* and *yz* projections for the faces). The reason for the multi degree IP fitting is that different IP degrees describe different features in the object. A simple and smooth object can be well described by a low degree IP, while higher IP degrees fitting will be less stable. However, a more complicated object with many details needs a relatively high degree IP for description, and a low degree IP won't contain enough information about it to be able to classify it correctly. By this approach we follow the MOFET technique, introduced for 2D contours recognition using IPs [11], [12].

Using the IP representation, we separate the IP into forms, and from each even degree form we compute

the linear rotation invariants described in Section III using the explicit expressions we developed in (20) and in (29).

In both 3D and 2D cases, we have $\left\lfloor \frac{n}{2} \right\rfloor + 1$ linear invariants for an IP of degree n.

2) Quadratic Invariants: For the 3D case, using the same IP representation, we also derived the first 2 quadratic rotation invariants (from the 1^{st} and 2^{nd} degree forms) described in Section III (see (23) and (32)). The rest of the 3D quadratic invariants, and all of the 3D angular invariants, are not used as features, as they aren't stable enough. The quadratic and angular invariants in the 2D case [24] are also not stable enough [11], [12], and therefore were not used as features.

The reason that we fit *even* degree IPs, to both the 3D objects and their 2D projections, is that our linear invariants are derived only from even degree forms, and the quadratic invariants - only from the 1^{st} and 2^{nd} degree forms. Thus, if we fit an IP of an odd degree, we won't use its leading form for invariants derivation, and so we could use a lower (even) degree IP for the same number of invariants derivation.

3) Implicit Polynomial Fitting Errors as Features: Following the MOFET technique [11], [12], we also suggest to use the IP fitting error as a feature. When comparing 2 different objects that have a similar shape, but one is smooth while the other has fine details, the first object will have a lower IP fitting error than the second one, since its smooth IP description will describe it well. Obviously, this feature is an inherent property of the object, and is also invariant to rotation.

Therefore, for each IP fitting degree, after solving the least squares problem for the IP fitting, we calculate the fitting error for each data-point and use the 75^{th} percentile of the errors vector as a feature describing the fitting error.

4) Eigenvalues of 3D PCA: We obtain 3 more features using principal component analysis on the original data points. The PCA eigenvalues are an inherent property of the object, since they describe its data-points scatter on its principal axes, and therefore - they are also invariant to rotation.

In the 3D case, we perform an eigenvalue decomposition of the 3x3 data scatter matrix. We sort the eigenvalues in decreasing order according to their magnitude, and use them as additional features.

Table I summarizes the features that were chosen for each application. The exact features were chosen for each application (object/face recognition) based on their robustness and informativeness in each case. For example, the IP fitting error is very similar between different faces and therefore were not used as a feature for the face recognition application. Another example is the IP degrees that were chosen: in the objects case, where the objects are quite different from one another, IP degrees of 2,4 and 6 were found

TABLE I
CHOSEN FEATURES FOR THE OBJECTS DATABASE AND THE FACES DATABASE

Features	Object Recognition	Face Recognition
3D linear invariants	IP degrees 2,4,6 (2+3+4=9 features)	IP degrees 2,4,6,8 (2+3+4+5=14 features)
3D IP fitting error	IP degrees 2,4,6 (1+1+1=3 features)	—
two 3D quadratic invariants $Q_{3D,1}$, $Q_{3D,2}$	IP degrees 2,4 (2+2=4 features)	IP degrees 2,4 (2+2=4 features)
2D linear invariants	*xy* - IP degrees 2,4,6 (2+3+4=9 features)	*xz*IP degrees 2,4,6 (2+3+4=9 features) *yz* - IP degrees 2,4 (2+3=5 features)
2D IP fitting error	*xy* - IP degrees 2,4,6 (1+1+1=3 features)	—
3D PCA eigenvalues	3 eigenvalues (3 features)	3 eigenvalues (3 features)
Total number of features	31 features	35 features

to be enough for the recognition. However, since the faces are very similar, and we need the features to represent the fine details that differentiate them - we used an IP of degree 8 as well. The total number of features was 31 in the object recognition case and 35 in the face recognition case.

V. RECOGNITION

A. Classifier Design

We chose to use a classifier that is based on the probability density functions (PDFs) of feature vectors. We assume that the feature vector has a Gaussian distribution. Each multivariate Gaussian PDF is estimated from feature vectors belonging to one or more different views of an object from the dictionary. Thus, each dictionary object is represented by one or more PDFs. In all the simulations we examined the 3 fitting algorithms: Gradient-One, RI-Min-Max and RI-Min-Var.

B. Objects Database

We divide the objects database into learning and testing data sets in the following way: we have 9 different positions for each object ($\sim 10^o - 15^o$ difference between consecutive positions, we denote these positions by #1-#9), and for each position we have 5 different consecutive frames. We use even positions for learning and odd positions for testing.

Note that this is a worst case scenario: we learned each object from positions with a difference of $\sim 25^o$ between them (#2,#4,#6 and #8), and then we tested our performance using the most different viewing

positions we could acquire - in the middle between the angles of the learning positions (#1,#3,#5,#7 and #9).

1) Learning Positions: For each learning position of each object we do the following: each of the 5 frames is perturbed 10 times by adding colored Gaussian noise (which, according to our analysis [23], is the appropriate acquisition noise model). We used an averaging filter of 11x11 (normalized so that the sum of its square coefficients is equal to 1) to filter white Gaussian noise with standard deviation of 0.01. The parameters were chosen empirically according to our noise model analysis. Thus, we have $40 \cdot 4 \cdot 5 = 800$ real frames in the learning data set, synthetically increased 10 times to obtain $800 \cdot 10 = 8000$ frames. For each of the 8000 perturbed instances we perform all the pre-processing stages from the previous section (except mirroring) and calculate the feature vector $\underline{v}$.

Let us denote each object by O_k , $k = 1, ..., 40$, and each learning position (view) by V_n, $n = 2, 4, 6, 8$. We examined several approaches for the classifier design [23]. The best approach, according to our experimental results, is estimating the parameters of a PDF from learning position pairs of each object.

Specifically, for each object we used the $50 \cdot 2 = 100$ feature vectors of positions #2 and #4 for the estimation of one PDF, the 100 feature vectors of positions #4 and #6 for the estimation of a second PDF, and the 100 feature vectors of positions #6 and #8 for the estimation of a third PDF. We used the feature vectors of each learning positions pair in order to compute a vector of means $\underline{\mu}$, and a covariance matrix Σ for construction of their PDF:

$$\begin{aligned} &P\left(\underline{v}/O_k, V_n\right) = \\ &= \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} exp\left(-\frac{1}{2}\left(\underline{v}-\underline{\mu}\right)^T \Sigma^{-1}\left(\underline{v}-\underline{\mu}\right)\right) \end{aligned} \tag{33}$$

where $k = 1, ..., 40$, $n = 2\&4, 4\&6, 6\&8$, and in case we use the entire feature vector we have $d = 31$.

In this approach we have $40 \cdot 3 = 120$ different PDFs for the description of 40 objects (3 PDFs per object).

This approach assumes that positions with very different viewing angles of the same object have very little in common, and therefore it will be better to learn similar positions together, but to separate the learning of different positions.

2) Testing Positions: We calculate the same feature vector for each of the 5 frames of each object testing position, resulting in $40 \cdot 5 \cdot 5 = 1000$ real frames in the testing data set. We then calculated the probability of it originating from each of the objects. The object which resulted in the highest probability

was chosen as the best recognition for the test vector.

We want to compare the probabilities of each object O_k in each position combination V_n, given the observation $\underline{v}$:

$$P\left(O_k, V_n/\underline{v}\right),\ k = 1, 2, ..., 40,\ n = 2\&4, 4\&6, 6\&8 \tag{34}$$

Using Bayes rule, we get:

$$\begin{aligned} &P\left(O_k, V_n/\underline{v}\right) = \frac{P(\underline{v}/O_k, V_n)P(O_k, V_n)}{P(\underline{v})} = \\ &= \frac{P(\underline{v}/O_k, V_n)P(V_n/O_k)P(O_k)}{P(\underline{v})} \end{aligned} \tag{35}$$

$P\left(\underline{v}\right)$ is the same for every k, and we assume that $P\left(O_k\right) = \frac{1}{40}$, and $P\left(V_n/O_k\right) = \frac{1}{3}$. Therefore, we can ignore these expressions in the probabilities comparison and compare only $P\left(\underline{v}/O_k, V_n\right)$, $k = 1, ..., 40$ and $n = 2\&4, 4\&6, 6\&8$.

C. Faces Database

We divide the faces database into learning and testing databases in the following way: we have about ~ 50 different frames for each face (acquired at a rate of 2.5 frame/sec). We use the first 5 frames as learning frames and 5 other frames are used as test frames (the test frames are linearly spaced along the rest of the movie).

1) Learning Frames: In a similar way to what was described in Section V-B1, we do the following for each face: each of the 5 learning frames is perturbed 10 times by adding colored Gaussian noise with standard deviation of 0.01. Thus, we have $41 \cdot 5 = 205$ real frames in the learning data set, synthetically enhanced 10 times to obtain $205 \cdot 10 = 2050$ frames. For each of the 2050 perturbed instances we perform all the pre-processing stages from the previous section and calculate the feature vector $\underline{v}$.

Let us denote each face by O_k , $k = 1, ..., 41$.

For each face we used the 50 feature vectors of the learning frames in order to compute a vector of means $\underline{\mu}$, and a covariance matrix Σ for the construction of their PDF $P\left(\underline{v}/O_k\right)$, in the same way as we did for the objects (see (33)).

In this case, the dimension of the feature vector $\underline{v}$ (if we use the entire feature vector) is $d = 35$, and $k = 1, ..., 41$.

In this approach we have 41 different PDFs for the description of 41 faces (one PDF per face).

2) Testing Frames: We calculate the same feature vector for each of the 5 testing frames of each face, resulting in $41 \cdot 5 = 205$ frames in the testing data set. We then calculated the probability of it originating from each of the faces. The face which resulted in the highest probability was chosen as the best recognition for the test vector.

Following the same method we introduced for the rigid object recognition, (only now, $k = 1, ..., 41$ since we have 41 faces), we use Bayes rule (again, ignoring $P(\underline{v})$). Since we use one PDF to describe each face, we only need to compare the nominators of the probabilities (i.e., compare $P(\underline{v}/O_k) P(O_k)$, $k = 1, ..., 41$). We assume that all faces have the same probability (i.e., $P(O_k) = \frac{1}{41}$), and so we can ignore $P(O_k)$ as well. Thus, for the comparison of the probabilities $P(O_k/\underline{v})$, we need to compare only $P(\underline{v}/O_k)$, $k = 1, ..., 41$.

D. Summary of the Recognition Process

Feature Vector Determination

For each 3D frame:

1) In case of the faces database only - perform mirroring as described in Section IV-B3.
2) Perform translation and scaling as described in Sections IV-B1 and .
3) Perform 2D projections on the main plains (xy for the objects, xz and yz for the faces) and extract the 2D contours, as described in Section IV-B4.
4) Fit 3D IPs of various degrees (See Table I) to the 3D object/face.
5) Fit 2D IPs of various degrees (See Table I) to the 2D contours.
6) Calculate the 3D/2D rotation invariant expressions and fitting errors from the IPs (See Table I).
7) Perform Principle Component Analysis on the data-points of the 3D object/face for the calculation of the 3D Eigen Values.
8) Construct the feature vector from the invariants, fitting errors and Eigen Values.

Learning Stage

Use the obtained feature vectors for estimation of multivariate Gaussian PDFs as described in Sections V-B1 and V-C1.

Testing Stage

Calculate the probabilities of each feature vector using the multivariate Gaussian PDFs learned during the learning stage, as described in Section V-B2 and V-C2. Compare these probabilities and select the object/face with the largest probability as the recognition test outcome of the tested object/face.

VI. Experimental Results

A. Synthetic Simulations

As we mentioned in Section V-B, we have a difference of about 12^o between consecutive positions (i.e., in each testing position, around $12^o/360^o \simeq 3\%$ of the closest learning positions' data-points are missing), and an acquisition noise model of colored Gaussian noise with a standard deviation of about 0.01.

In order to analyze the sensitivity of the entire recognition process, we performed the following two synthetic simulations:

The first: In the learning stage we used colored Gaussian noise as described in Section V-B1, each time with a different standard deviation (0.01-0.05). Then, in the testing stage, we added to each testing frame, of each object, colored Gaussian noise with the same standard deviation as in the learning stage. The recognition results appear in Fig. 9(a). Note that the noise was added *in addition* to the acquisition noise which already exists in the testing frames.

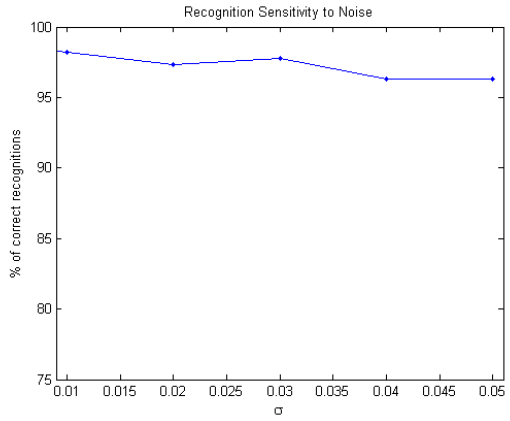
The second: The learning stage was the same as described in Section V-B1. Then, in the testing stage, we chose various percentage values of points (1%-10%) to eliminate from each testing frame of each object (a central point was randomly selected, then an appropriate number of closest points were eliminated). The results appear in Fig. 9(b). Note that these data-points were eliminated *in addition* to the occlusion which already exists in the testing frames.

It can be seen that as the standard deviation of the noise or the percentage of missing points becomes larger, the recognition process performance is reduced relatively slowly.

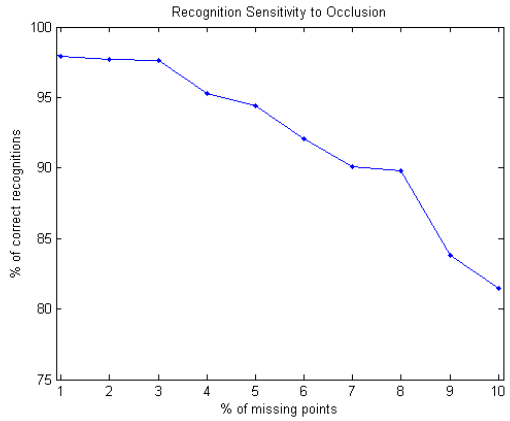
B. Rigid Object Recognition Results

Using the entire feature vector ($d = 31$) the results appear on the first row of Table II. The different fitting approaches have similar performance, all of them classify correctly over 98% of the instances, but the Gradient-One fitting results are slightly better than those of the other fitting methods.

The analysis of the contribution of features based on IPs only (3D/2D invariants and fitting errors, $d = 28$), of features based on 3D IPs only (invariants and fitting errors, $d = 16$) and of 3D PCA features only ($d = 3$) to the recognition appears in also in Table II. It can be seen that without the additional PCA features, the results are less good, but we still manage to classify correctly over 95% of the instances.



(a) Sensitivity to noise



(b) Sensitivity to occlusion

Fig. 9. Analysis of the recognition process sensitivity to (a) noise and (b) occlusion.

TABLE II
RESULTS OF OBJECT RECOGNITION USING THE ENTIRE FEATURE VECTOR ($d = 31$), USING IP BASED FEATURES ONLY ($d = 28$), USING 3D FEATURES ONLY ($d = 16$) AND USING 3D PCA EIGENVALUES ONLY ($d = 3$)

	Gradient-One	RI-Min-Max	RI-Min-Var
$d = 31$	**98.8%**	98.1%	98.0%
$d = 28$	**96.9%**	96.4%	95.6%
$d = 16$	**96.1%**	94.8%	93.2%
$d = 3$	74.2%		

Note that even without the 2D/PCA based features we classify correctly over 93% of the instances. Gradient-One fitting results are better than those of the other fitting methods in this case as well.

1) Comparison with Other Methods: We compared our proposed method results with pose estimation recognition results. The pose estimation methods we examined are PCA and IP based tensor approach [3]. In both cases, after the rotation of the data points, we fit an IP and use the IP coefficients as our features.

We then use an l_2 distance between feature vectors for classification. The IP degree we chose is 4

TABLE III
COMPARISON BETWEEN THE PROPOSED METHOD, POSE ESTIMATION AND THE SSD METHOD FOR OBJECT RECOGNITION

Method	Results
Proposed method (Gradient-One, $d = 31$)	**98.8%**
PCA based pose estimation (4^{th} degree IP, $d = 35$)	91.6%
IP based pose estimation (4^{th} degree IP, $d = 35$)	90.5%
SSD ($d = 27$)	90.1%

($d = 35$), since in this case we use a single degree IP, and higher degree IP coefficients were less stable and showed poor recognition performance. Note that our proposed invariants are *combinations* of IP coefficients, and they were found to be useful even for IPs with degree larger than 4. We used IP based pose estimation with a 2^{nd} degree IP (i.e., the 2^{nd} degree IP was used for pose estimation and then a 4^{th} degree IP was fitted and its coefficients were used as features).

The pose estimation results using Gradient-One appear in Table III. Using a different IP fitting algorithm (RI-Min-Max or RI-Min-Var) didn't affect the results. It can be seen that the PCA has a similar performance to the 2^{nd} degree IP pose estimation.

It can also be seen that our proposed method results using Gradient-One with the $d = 31$ (98.8%) are better than the PCA/IP based pose estimation methods.

We also compared the performance of the proposed method with the Shape Spectrum Descriptor (SSD) technique [18]. This technique was adopted by the MPEG-7 standard for 3D descriptors and has a relatively low complexity. We used a histogram of 25 bins for the descriptor (the default for MPEG-7 is 100 bins, but its results were found to be a little worse) and we also used the singular and planar descriptors (i.e., we had the 25 bins plus the 2 descriptors, a total of 27 features). We used the l_1 norm on the difference between the feature vectors for classification. We implemented the SSD technique in Matlab®. Our implementation is based on [18] and on the freely available MPEG-7 reference software [33].

The results of this comparison also appear in Table III. It can be seen that our proposed method has better performance. The computational complexity of the methods is considered in the next section.

2) Comparison of Computational Complexity: All 3 methods, our proposed method, pose estimation based techniques, and the SSD technique, have *similar computational complexity*. If we denote the number of object data-points by N, then each of the stages of each method has a complexity of $O(N)$ (The SSD stages are actually dependent on the number of triangles, which in our databases is around $2N$ for an object with N data-points). The constant that multiplies N is large in some stages (such as the Least-Squares IP fit in our case and in the pose estimation case, or the 2^{nd} degree explicit polynomial fit at

TABLE IV
AVERAGE RUNNING TIMES OF RECOGNITION OF A SINGLE OBJECT (MATLAB® IMPLEMENTATION)

proposed method (Gradient-One, $d = 31$)	pose estimation+IP coef. (PCA/4^{th}degree IP, $d = 35$)	SSD ($d = 27$)
13 Sec	15 Sec	55 Sec

TABLE V
RESULTS OF FACE RECOGNITION USING THE ENTIRE FEATURE VECTOR ($d = 35$), USING IP BASED FEATURES ONLY ($d = 32$), USING 3D FEATURES ONLY ($d = 18$) AND USING 3D PCA EIGENVALUES ONLY ($d = 3$)

	Gradient-One	RI-Min-Max	RI-Min-Var
$d = 35$	**97.1%**	96.6%	93.7%
$d = 32$	**90.2%**	89.3%	90.7%
$d = 18$	89.3%	**91.7%**	**91.7%**
$d = 3$	65.8%		

each point in the SSD case), but is difficult to estimate. Average running times for the recognition of a single object, using a Matlab implementation, appear in Table IV. The proposed method appear to have a commensurate running time with the other two methods, while presenting better results on the examined databases.

C. Face Recognition Results

Using the entire feature vector ($d = 35$) the results appear on the first row of Table V. It can be seen that in this case as well, the Gradient-One fitting based features have the best recognition performance.

The analysis of the contribution of features based on IPs only (3D/2D invariants, $d = 32$), of features based on 3D IPs only (invariants, $d = 18$) and of 3D PCA features only ($d = 3$) to the recognition appears also in Table V. It can be seen that without the additional 2D/PCA features, the results are less good, but we still manage to classify correctly over 91% of the instances. Note that Gradient-One fitting results are now a little worse than those of the other fitting methods. We conclude that for this database, the 3D features are more informative in the cases of RI-Min-Max and RI-Min-Var, compared with Gradient-One.

1) Comparison with Other Methods: We compared our results with pose estimation recognition results. The pose estimation methods we examined are PCA and IP based tensor approach in the same way described in Section VI-B1.

The pose estimation results using Gradient-One appear in Table VI. Using a different IP fitting algorithm (RI-Min-Max or RI-Min-Var) didn't affect the results. It can be seen that again, the PCA method has a similar performance to the 2^{nd} degree IP pose estimation method.

TABLE VI
COMPARISON BETWEEN THE PROPOSED METHOD, POSE ESTIMATION AND THE SSD METHOD FOR FACE RECOGNITION

Method	Results
Proposed method (Gradient-One, $d = 35$)	**97.1%**
PCA based pose estimation (4^{th} degree IP, $d = 35$)	93.2%
IP based pose estimation (4^{th} degree IP, $d = 35$)	92.7%
SSD ($d = 27$)	72.2%

It can also be seen that our proposed method results using Gradient-One with the $d = 35$ (97.1%) are better than the PCA/IP based pose estimation methods that appear in the table.

We compared the performance of the proposed method with the Shape Spectrum Descriptor (SSD) technique [18]. We used the same features and norm as in Section VI-B1.

The results of this comparison also appear in Table VI. It can be seen that our proposed method has better performance for the examined database.

VII. SUMMARY AND CONCLUSIONS

In this work we examined the description and recognition abilities of 3D IPs using existing fitting algorithms (Gradient-One, Min-Max and Min-Var). We introduced new sets of 3D rotation invariants (linear, quadratic and angular) which are based on IPs and their tensor representation and obtained closed-form expressions for these invariants for every IP degree. We also developed one more quadratic invariant based on IP properties and trigonometric identities. The pre-processing we apply to the data has a considerable influence on the recognition performance. Our pre-processing includes translation, scaling, robust normal direction calculation, and the novel approaches of mirroring with respect to an appropriate plane (in case of the faces database) and preparing 2D projections for obtaining extra features. Following the 2D IP recognition method known as MOFET [11], [12] we suggested the use of a feature vector that contains 3D IP rotation invariants and fitting errors, 2D IP rotation invariants and fitting errors (from the most descriptive 2D projections) and the eigenvalues of a PCA decomposition. The feature vector included invariants of several IP degrees (both for 2D and 3D) in order to utilize both the stability of low degree IPs and the descriptiveness of high degree IPs.

We designed a PDF based classifier and showed that the 3D IP invariants and fitting error approach has better performance for objects and face recognition compared with pose estimation methods followed by IP fitting [3]. We also found in our tests that our proposed method outperforms the Shape Spectrum Descriptor (SSD) technique [18], which was adopted by the MPEG-7 standard for 3D descriptors.

Future work could consider combining the regularization technique (Ridge-Regression) of [6], [10], with the Min-Max/Min-Var algorithms [9], and explore the description and recognition abilities of the obtained fitting algorithms with the newly derived invariants. Another interesting direction for future work, is exploring the use of quaternions [34] for the derivation of a full set of quadratic rotation invariants, in a similar way to the use of complex representation in the 2D case [24].

ACKNOWLEDGMENT

The authors would like to thank the Geometric Image Processing (GIP) lab at the Computer Science Faculty of the Technion, for the faces database and for using their equipment to acquire the rigid objects database. The authors would also like to thank Zoya Landa, for the contribution of the Matlab implementation of the 2D MOFET approach.

REFERENCES

[1] J. Subrahmonia, D. B. Cooper, and D. Keren, "Practical reliable bayesian recognition of 2D and 3D objects using implicit polynomials and algebraic invariants," *IEEE Trans. on PAMI*, vol. 18, no. 5, pp. 505–519, May 1996.

[2] M. Barzohar, D. Keren, and D. Cooper, "Recognizing groups of curves based on new affine mutual geometric invariants, with applications to recognizing intersecting roads in aerial images," *IAPR Int'l Conf. Pattern Recognition*, vol. 1, pp. 205–209, October 1994.

[3] J. Tarel, H. Civi, and D. Cooper, "Estimation of free-form 3D objects without point matching using algebraic surface models," *Proc. IEEE Workshop Model-Based 3-D Image Analysis, Mumbai, India*, pp. 13–21, 1998.

[4] G. Taubin, "Estimation of planar curves, surfaces and nonplanar space curves defined by implicit equations, with applications to edge and range image segmentation," *IEEE Trans. on PAMI*, vol. 13, no. 11, pp. 1115–1138, November 1991.

[5] M. M. Blane, Z. Lei, H. Civi, and D. B. Cooper, "The 3L algorithm for fitting implicit polynomial curves and surfaces to data," *IEEE Trans. on PAMI*, vol. 22, no. 3, pp. 298–313, March 2000.

[6] T. Tasdizen, J. Tarel, and D. Cooper, "Improving the stability of algebraic curves for applications," *IEEE Trans. on Image Proc.*, vol. 9, no. 3, pp. 405–416, March 2000.

[7] D. Keren and C. Gotsman, "Fitting curves and surfaces with constrained implicit polynomials," *IEEE Trans. on PAMI*, vol. 21, no. 1, pp. 31–41, January 1999.

[8] D. Keren, "Topologically faithful fitting of simple closed curves," *IEEE Trans. on PAMI*, vol. 26, no. 1, pp. 118–123, January 2004.

[9] A. Helzer, M. Barzohar, and D. Malah, "Stable fitting of 2D curves and 3d surfaces by implicit polynomials," *IEEE Trans. on PAMI*, vol. 26, no. 10, pp. 1283–1294, October 2004.

[10] T. Sahin and M. Unel, "Stable algebraic surfaces for 3D object representation," *Journal of Mathematical Imaging and Vision*, vol. 32, no. 2, pp. 127–137, October 2008.

[11] Z. Landa, *2D Object Description and Classification Based on Contour Matching by Implicit Polynomials*. M.Sc. Thesis, The Technion - Israel Institute of Technology, August 2006. [Online]. Available: http://sipl.technion.ac.il/siglib/FP/Zoya_Landa.pdf

[12] Z. Landa, D. Malah, and M. Barzohar, "2D object description and recognition based on contour matching by implicit polynomials," *submitted to IEEE Trans. on PAMI*, 2008.

[13] P. Besl and N. McKay, "A method for registration of 3-D shapes," *IEEE Trans. on PAMI*, vol. 14, no. 2, pp. 239–256, February 1992.

[14] P. Mordohai and G. Medioni, "Dimensionality estimation and manifold learning using tensor voting." [Online]. Available: http://iris.usc.edu/ medioni/download/ndmanual.htm

[15] A. E. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3D scenes," *IEEE Trans. on PAMI*, vol. 21, no. 5, pp. 433–449, May 1999.

[16] C. Zhong, Z. Sun, T. Tan, and Z. He, "Robust 3D face recognition in uncontrolled environments," *IEEE Conference on Computer Vision and Pattern Recognition 2008*, pp. 1–8, June 2008.

[17] A. Bronstein, M. Bronstein, and R. Kimmel, "Expression-invariant representations of faces," *IEEE Trans. Image Processing*, vol. 16, no. 1, pp. 188–197, January 2007.

[18] T. Zaharia and F. Preteux, "3d shape-based retrieval within the mpeg-7 framework," *Proc. SPIE Conf. on Nonlinear Image Processing and Pattern Analysis XII, San Jose, Etats-Unis*, vol. 4304, pp. 133–145, January 2001.

[19] C. Dorai and A. K. Jain, "Shape spectrum based view grouping and matching of 3D free-form objects," *IEEE Trans. on PAMI*, vol. 19, no. 10, pp. 1139–1145, October 1997.

[20] A. Helzer, *Robust Fitting of Implicit Polynomials with Application to Contour Coding*. M.Sc. Thesis, The Technion - Israel Institute of Technology, June 2000.

[21] Stanford University Computer Graphics Laboratory, "The stanford 3D scanning repository." [Online]. Available: http://www-graphics.stanford.edu/data/3Dscanrep/

[22] T. Tasdizen, *Robust and Repeatable Fitting of Implicit Polynomial Curves to Point Data Sets and to Intensity Images*. Ph.D. Thesis, Brown University, September 2000.

[23] H. Ben-Yaacov, *3D Object Description and Classification by Implicit Polynomials*. M.Sc. Thesis, Submitted to The Technion - Israel Institute of Technology, September 2008. [Online]. Available: http://sipl.technion.ac.il/siglib/FP/Hilla-Ben-Yaacov.pdf

[24] J. Tarel and D. Cooper, "The complex representation of algebraic curves and its simple exploitation for pose estimation and invariant recognition," *IEEE Trans. on PAMI*, vol. 22, no. 7, pp. 663–674, July 2000.

[25] D. Keren, "Using symbolic computation to find algebraic invariants," *IEEE Trans. on PAMI*, vol. 16, no. 11, pp. 1143–1149, November 1994.

[26] M. Unel and W. A. Wolovich, "A new representation for quartic curves and complete sets of geometric invariants," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 13, no. 8, pp. 1137–1149, December 1999.

[27] M. Unel and W. Wolovich, "On the construction of complete sets of geometric invariants for algebraic curves," *Advances in Applied Mathematics*, vol. 24, no. 1, pp. 65–87, January 2000.

[28] G. B. Arfken and H. J. Weber, *Mathematical Methods For Physicists*. Elsevier, 2005.

[29] "Stuttgart range image database." [Online]. Available: http://range.informatik.uni-stuttgart.de/htdocs/html/

[30] S. Rusinkiewicz, D. DeCarlo, A. Finkelstein, and A. Santella, "Suggestive contour gallery." [Online]. Available: http://www.cs.princeton.edu/gfx/proj/sugcon/models/

[31] "The digital michelangelo project." [Online]. Available: www.graphics.stanford.edu/data/dmich-public/

[32] D. Scharstein and R. Szeliski, "High-accuracy stereo depth maps using structured light," *Proceedings. 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. I–195–I–202, June 2003.

[33] MPEG-7 Implementation Studies Group, "Technology - multimedia content description interface - part 6: Reference software, ISO/IEC FCD 15938-6 / N4006, MPEG-7," March 2001.

[34] B. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Journal of the Optical Society of America*, vol. 4, pp. 629–642, April 1987.