A FAST ACCURATE IMPLICIT POLYNOMIAL FITTING APPROACH

Mohammad Rouhani and Angel D. Sappa

Computer Vision Center
Edifici O, Campus UAB
08193 Bellaterra, Barcelona, Spain
{*rouhani, asappa*}*@cvc.uab.es*

ABSTRACT

This paper presents a novel hybrid approach that combines state of the art fitting algorithms: *algebraic-based* and *geometric-based*. It consists of two steps; first, the 3L algorithm is used as an initialization and then, the obtained result, is improved through a geometric approach. The adopted geometric approach is based on a distance estimation that avoids costly search for the real orthogonal distance. Experimental results are presented as well as quantitative comparisons.

Index Terms— Implicit Polynomial Fitting, Algebraic and Geometric Approaches, LMA.

1. INTRODUCTION

Implicit polynomial (IP) fitting has been used in the computer vision field for obtaining compact 2D or 3D data representations. These representations have been largely exploited in object description and classification (e.g., [3, 8, 11]).

The fitting problem can be modelled as an optimization problem finding the set of parameters that minimize a distance measurement between the given set of points and the resulting implicit polynomial. The most natural way to define the distance is to measure the deviation of the implicit function values from the expected values (i.e., *level set*) at each given point. In other words, the value of the polynomial should reach zero at the location of the given data points. This measure criterion is referred in the literature as *algebraic distance* [4, 7, 9].

Another distance measure, referred as *orthogonal* or *geometric distance*, is defined as the shortest distance from the given point to the fitting curve/surface. On the contrary to the algebraic distance, this distance has an intuitive geometric meaning, and its final result makes sense as a consequence. Although this definition of the distance is complete and leads us to the best fitting result, it has a nonlinear nature with respect to the model parameter, which sometimes discourages its use. Furthermore, since there is not a closed formula to compute the shortest distance to a general implicit polynomial, iterative approaches should be used to compute this orthogonal distance (e.g., [1, 2]); alternatively, an approximation to that shortest distance could be computed and used as the residual value (e.g., [6, 10]).

Algebraic and geometric distances are two different viewpoints for the fitting problem. Although both of them could be exploited for some optimization models leading to the optimal parameters in their own sense, the frameworks they use are different. Algebraic fitting methods are based on least square approaches giving a non-iterative unique solution, while the geometric ones are based on some non-linear models giving the solution through iterative algorithms.

The current work proposes a novel scheme that combines a well known algebraic distance based approach—the 3L algorithm [4]—with a geometric one based on a novel distance estimation [10]. In other words, the geometric stage is intended to increase the accuracy of the algebraic result by incorporating a geometric criteria. The rest of the paper is organized as follows. Section 2 describes the proposed scheme. Section 3 gives experimental results and comparisons and finally, conclusion and future work are presented in section 4.

2. PROPOSED APPROACH

Let f be an implicit polynomial of degree d represented as:

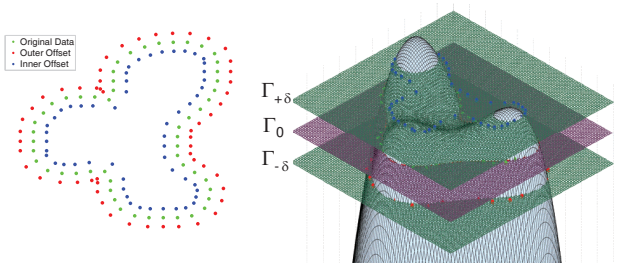
$$f(\mathbf{x}) = \sum_{\substack{(i+j+k)\leq d \\ \{i,j,k\}\geq 0}} a_{i,j,k} \cdot x^i \cdot y^j \cdot z^k = 0, \qquad (1)$$

or, in a vector form:

$$f(\mathbf{x}) = \mathbf{m}^{\mathrm{T}}\mathbf{a} = 0, \qquad (2)$$

where $\mathbf{m}$ is the column vector of monomials and $\mathbf{a}$ is the polynomial coefficient vector. The proposed approach consists of two stages. First, the polynomial coefficients are initialized by using an algebraic fitting algorithm. Then, the obtained result is improved by means of a non-linear optimization geometric approach. The geometric approach is based on an estimation of the shortest distance. The two stages defining the proposed scheme are detailed next.

This work has been supported by the projects TRA2007-62526/AUT and CTP-2008ITT00001 and research programme Consolider-Ingenio 2010: MIPRCV (CSD2007-00018).



(a) *(b)*

Fig. 1. (a) Level sets: original data (Γ_0), outer offset ($\Gamma_{-\delta}$) and inner offset ($\Gamma_{+\delta}$). (b) A 3D illustration of the 3L algorithm.

2.1. COEFFICIENT VECTOR INITIALIZATION

The above coefficient vector $\mathbf{a}$ in (2) is initialized by using the well known 3L fitting algorithm [4]. In short, the 3L algorithm is a linear least squares *explicit* polynomial fitting that consists in generating two additional *level sets*: $\Gamma_{-\delta}$ and $\Gamma_{+\delta}$ from the original data set Γ_0. These two additional data sets are generated so that one is internal and the other is external, and are placed at a distance $\pm\delta$ from the original data along a direction that is locally perpendicular to the given data set (Fig. $1(a)$). Hence, the 3L algorithm incorporates a control for a local continuity resulting in a more stable solution. Figure $1(a)$ depicts the original data with the two additional level sets; the corresponding 3D illustration is presented in Fig. $1(b)$.

Considering the three level sets: $\{\Gamma_{-\delta}, \Gamma_0, \Gamma_{+\delta}\}$ the equation (2) could be represented by using a block matrix $\mathbf{M}_{3L}$ and a block column vector $\mathbf{b}$:

$$\mathbf{M}_{3L} = \begin{bmatrix} \mathbf{M}_{\Gamma_{-\delta}} \\ \mathbf{M}_{\Gamma_0} \\ \mathbf{M}_{\Gamma_{+\delta}} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} -\mathbf{c} \\ \mathbf{0} \\ +\mathbf{c} \end{bmatrix}, \tag{3}$$

where $\mathbf{M}_{\Gamma_0}$, $\mathbf{M}_{\Gamma_{+\delta}}$, $\mathbf{M}_{\Gamma_{-\delta}}$ are matrices of monomials calculated in the original, inner and outer sets respectively; $\pm c$ are the corresponding expected values in the inner and outer level sets. Then, the least squares solution for $\mathbf{a}$ is obtained:

$$\mathbf{a} = \mathbf{M}_{3L}^{\dagger}\mathbf{b} = (\mathbf{M}_{3L}^T\mathbf{M}_{3L})^{-1}\mathbf{M}_{3L}^T\mathbf{b}, \tag{4}$$

where $\mathbf{M}_{3L}^{\dagger}$ denotes the pseudoinverse of $\mathbf{M}_{3L}$.

2.2. COEFFICIENT VECTOR UPDATING

The computation of the polynomial coefficient vector $\mathbf{a}$ in the previous stage lacks of geometric meaning; hence it could happen that this polynomial coefficient vector is not an accurate approximation of the given set of points since the three sets of points ($\Gamma_{-\delta}$, Γ_0 and $\Gamma_{+\delta}$) were equally considered by the least squares solution. In order to overcome this problem, in the current work a geometric fitting approach is used for improving the obtained results.

Geometric fitting approaches rely on the shortest distance —or an approximation of it—between every point and its corresponding one on the curve/surface. Thus, in general case of geometric methods we have the following optimization problem:

$$min_{\mathbf{a}}(\sum_{i=1}^{n} min_{\hat{p}_i} d(p_i, \hat{p}_i)), \tag{5}$$

where each $\hat{p}_i$ is the correspondence of p_i on the curve/surface.

Theoretically, both unknown polynomial coefficients and the correspondences must be found simultaneously, but practically this problem is tackled by first assuming an initial curve/surface (in the proposed scheme the result from Section 2.1 is used for speeding up the process), and then refine it till convergence is reached. So, the fitting problem is split up into two stages: 1) point correspondence search; and 2) surface parameter refinement. The first stage deals with the inner part of (5), while the second one concerns about the outer one.

Regarding to the first stage, we need to find the correspondence for each data point. For this purpose, two different strategies have been proposed in the literature: (a) finding the shortest distance by solving a non-linear system (e.g., [2, 1]); and (b) computing an estimation of the shortest distance (e.g., [6, 10]). In the current work, costly iterations are avoided by computing an efficient estimation of the geometric distance [10], which despite other approaches, is not based on a single direction. It is briefly explained in the next section.

2.2.1. DISTANCE ESTIMATION

First a *simplex* is constructed through each point and its intersections along the coordinate axis. Figure $2(a)$ and Fig. $2(b)$ show the simplex in 2D and 3D cases respectively. In the 3D case, having constructed the tetrahedron, its height segment is considered as an approximation of the geometric distance. Precisely speaking, this tetrahedron is defined by the given point and three intersections satisfying $f_{\mathbf{a}}(x, y_p, z_p) = 0$, $f_{\mathbf{a}}(x_p, y, z_p) = 0$ and $f_{\mathbf{a}}(x_p, y_p, z) = 0$, where (x_p, y_p, z_p) is the given point.

In the particular case tackled in this work, since the fitted curve/surface is defined by the implicit equation (1), the intersection points are found by finding the root of a one dimensional function close to the data point. In order to estimate a root an ordinary algorithm like the Newton's method could be used.

A direct formula to describe the estimated distance can be obtained by using a mathematical trick. Without loss of generality the 3D case is considered here. Let r, s and t be the three intersections with the current surface, which create a triangular planar patch (see Fig. $2(b)$). Since the volume of the tetrahedron is defined as the product of the area of each base

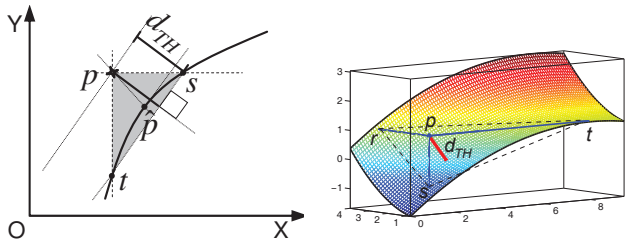


Fig. 2. Illustration of the simplex used for estimating the geometric distance: (a) 2D case; (b) 3D case.

by its corresponding height, three sets of expressions lead us to the same value. Hence, the height of the tetrahedron, d_{TH}, could easily be computed from the following relationship:

$$d_{TH} = (|pr|.|ps|.|pt|)/|\vec{rs} \times \vec{rt}|, \quad (6)$$

where $\times$ refers to the cross product operator between two vectors. Similar relationship can be obtained in the 2D case, but by using the area of the triangle instead of the volume. More details can be found in [10].

2.2.2. *NON-LINEAR OPTIMIZATION*

As a result from the previous stage the total distance of the data set to the current curve/surface has been found. Hence, now an optimization framework could be used to refine the curve/surface parameters. LevenbergMarquardt algorithm (LMA) is a well-known method in non-linear optimization [5], which in some sense interpolates between the Gauss–Newton algorithm and the gradient descent.

The geometric distance (6) provides a straightforward method to approximate the orthogonal distance. Moreover, its Jacobian matrix could be directly derived through the differentiation rules. The Jacobian matrix shows the sensitivity of each d_i with respect to the parameter vector, and could be calculated from (6):

$$D_j|d_{TH}| = (|\vec{rs} \times \vec{rt}|.D_j\upsilon - \upsilon.D_j|\vec{rs} \times \vec{rt}|)/|\vec{rs} \times \vec{rt}|^2, \quad (7)$$

where υ shows the volume of the tetrahedron, and $D_j = \partial/\partial c_j$ is the differentiation operator respect to the parameter vector. In order to calculate each term of the (7) the implicit differentiation rule must be used for each intersection point.

Having estimated the geometric distance (6) and its Jacobian matrix through (7) it is easy to handle LMA in order to refine the curve/surface parameters:

$$\vec{\mathbf{a}}^{t+1} = \vec{\mathbf{a}}^t + \beta\triangle\vec{\mathbf{a}},$$
$$(J^TJ + \lambda diag(J^TJ))\triangle\vec{\mathbf{a}} = J^TD, \quad (8)$$

where β is the refinement step; $diag(J^TJ)$ is the diagonal matrix containing the elements of (J^TJ); $\triangle\mathbf{a}$ represents the refinement vector showing the amount of change in the current polynomial coefficients; λ is the damping parameter in LMA which shall be adjusted in each iteration; and the vector $D = (d_1(\mathbf{a}^t), ..., d_n(\mathbf{a}^t))^T$ corresponds to the distances (computed from (6)). Parameter refinement (8) must be repeated till convergence happens.

3. EXPERIMENTAL RESULTS

This section presents experimental results obtained with the proposed hybrid approach as well as comparisons with the 3L algorithm [4] and an orthogonal distance based fitting approach [1]. The results are provided for both 2D and 3D cloud of points.

Figure 3 shows 2D contours fitted by fifth and sixth degree IPs (based on its shape complexity) using the 3L algorithm (Fig. $3(a)$), the proposed approach (Fig. $3(b)$) and a non-linear orthogonal distance based approach [1] (Fig. $3(c)$). The fitting error, computed through the whole set of points with [1], is used as a quantitative criterion for comparison; it is referred as AFE: Accumulated Fitting Error. In all the cases the accuracy obtained with the proposed approach considerably improves the one obtained with the 3L algorithm; moreover, it is comparable (in one case even better, see the last row) to the results obtained when the non-linear approach is used. Although out of the scope of the current work, it should be mentioned that the proposed approach is about ten times faster than [1].

Figure 4 shows two examples with synthetic 3D data sets. In both cases fifth degree IPs are used. The proposed approach converges to almost the same result as [1], but twice faster. The same number of iterations was performed with [1] and with the proposed approach (i.e., 50 iterations during the LMA optimization step). The surface presented in Fig. $4(top)$ is the result obtained with the proposed approach (AFE=0.68) when a 3D data set containing 360 points is used. This result is similar to the one obtained with [1] (AFE=0.69), while is considerable better than 3L (AFE=1.89). Figure $4(bottom)$ shows the result of the proposed approach (AFE=1.03) from a cloud of 438 points. Again, it is comparable with [1] (AFE=1.28) and better than 3L (AFE=2.67).

4. CONCLUSIONS

This paper presents a novel scheme for exploiting the speed of an algebraic based approach together with the accuracy of a geometric distance based approach. The adopted geometric approach avoids the time consuming task to find the real orthogonal distance by using an efficient distance estimation, which despite being proposed for quadrics, in the current work is used for higher degree polynomials. Experimental results show improvements both in accuracy and CPU time. As future work, the use of the proposed scheme will be considered for object recognition.

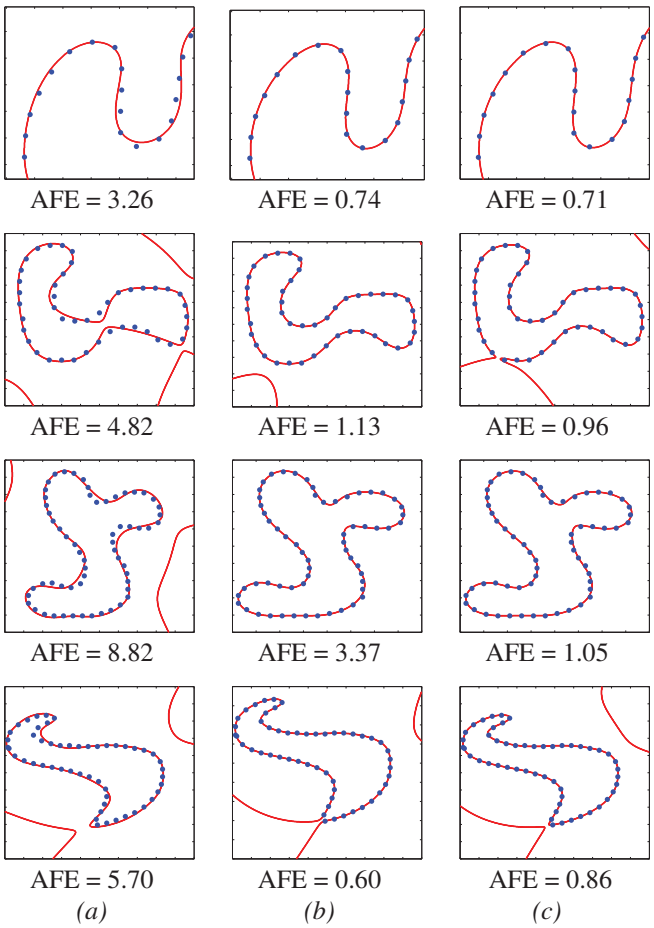


Fig. 3. 2D contours fitted by fifth (1st row) and sixth (2nd, 3rd and 4th rows) degree IPs: (a) Results from the 3L algorithm; (b) the proposed approach; (c) results from [1], which is used as ground truths. AFE shows the accumulated fitting error respectively. The last row shows a case where [1] stops due to the maximum iteration criterion, and cannot reach the error by the proposed approach.

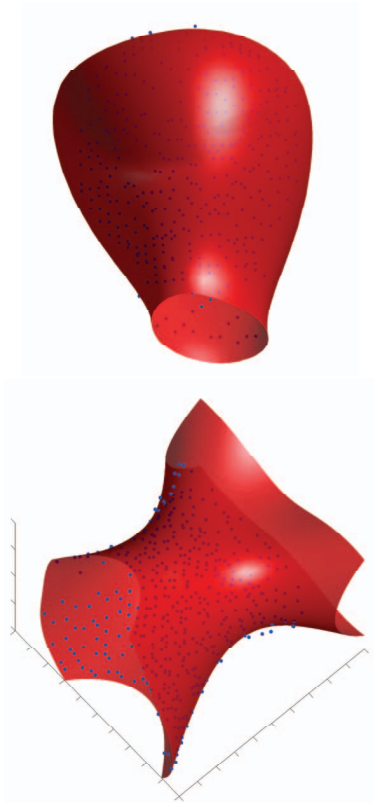


Fig. 4. Surfaces obtained with the proposed approach (fifth degree IPs) fitting synthetic 3D data sets.

5. REFERENCES

[1] S. Ahn, W. Rauh, H. Cho, and H. Warnecke. Orthogonal distance fitting of implicit curves and surfaces. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(5):620–638, May 2002.

[2] M. Aigner and B. Jutler. Gauss-newton-type technique for robustly fitting implicit defined curves and surfaces to unorganized data points. *IEEE International Conference on Shape Modelling and Application*, 1:121–130, 2008.

[3] H. Ben-Yaacov, D. Malah, and M. Barzohar. Recognition of 3d objects based on implicit polynomials. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, in press.

[4] M. Blane, Z. Lei, H. Civil, and D. Cooper. The 3l algorithm for fitting implicit polynomials curves and surface to data. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(3):298–313, March 2000.

[5] R. Fletcher. *Practical Methods of Optimization*. New York: Wiley, 1990.

[6] P. Gotardo, O. Bellon, K. Boyer, and L. Silva. Range image segmentation into planar and quadric surfaces using an improved robust estimator and genetic algorithm. *IEEE Trans. on Systems, Man, and CyberneticsPart B: Cybernetics*, 34(6):2303–2316, 2004.

[7] A. Helzer, M. Barzohar, and D. Malah. Stable fitting of 2d curves and 3d surfaces by implicit polynomials. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(10):p 1283–1294, October 2004.

[8] D. Keren, D. Cooper, and J. Subrahmonia. Describing complicated objects by implicit polynomials. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 16(1):38–53, January 1994.

[9] M. Rouhani and A. D. Sappa. Relaxing the 3L algorithm for an accurate implicit polynomial fitting. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, San Francisco, USA, June 2010.

[10] A. Sappa and M. Rouhani. Efficient distance estimation for fitting implicit quadric surfaces. In *Proc. IEEE Int. Conf. on Image Processing*, Cairo, Egypt, 2009.

[11] B. Zheng, J. Takamatsu, and K. Ikeuchi. An adaptive and stable method for fitting implicit polynomial curves and surface. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, in press.