Computers & Graphics 67 (2017) 14–23

Contents lists available at ScienceDirect

Computers & Graphics

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

Technical Section

Curve and surface fitting by implicit polynomials: Optimum degree finding and heuristic refinement

Ruben Interian[a,*], Juan M. Otero[b], Celso C. Ribeiro[a], Anselmo A. Montenegro[a]

[a] Institute of Computing, Fluminense Federal University, Av. Gen. Milton Tavares de Souza, Niterói, Rio de Janeiro 24210-346, Brazil
[b] Mathematics and Computer Science Faculty, Havana University, San Lazaro and L, Vedado, Havana 10400, Cuba

ARTICLE INFO

Article history:
Received 23 December 2016
Revised 5 May 2017
Accepted 5 May 2017
Available online 17 May 2017

Keywords:
Implicit polynomial
Curve fitting
Surface fitting
Object representations
Optimization

ABSTRACT

Finding an implicit polynomial that fits a set of observations X is the goal of many researches in recent years. However, most existing algorithms assume the knowledge of the degree of the implicit polynomial that best represents the points. This paper presents two main contributions. First, a new distance measure between X and the implicit polynomial is defined. Second, this distance is used to define an algorithm able to find the degree of the polynomial needed for the representation of the data set. The proposed algorithm is based on the idea of gradually increase the degree, while there is an improvement in the smoothness of the solutions. The experiments confirm the validity of the approach for the selected 2D and 3D datasets.

1. Introduction

The problem of 2D and 3D object representation is present in different areas of research as computer graphics and computer vision. Modelling, 3D reconstruction and recognition tasks depend totally on a good representation of the observed objects.

Unfortunately, it is usual to receive noised, discrete or incomplete real world data. The models are obtained from images, videos, 3D scanners or another capture devices [11]. The nature of this devices allow to obtain a finite and discrete amount of data from the original object, commonly as a point set. The process of finding a model that fits better this observation set is a goal of many investigations in the last years [5,15,21,35].

Implicit polynomials (from now on IPs) are proved to be a powerful tool modelling real objects compared to other representation types, like explicit or parametric [34], with surprising variety of forms (Fig. 1, “Cherries” and “The pear”) and good properties where required:

- A compact surface representation, i.e. concise, efficient description of the object, using few parameters [8]
- Algebraic and geometric invariants, for instance, area and volume [30]
- A fast way for classifying points as internal or external to the object
- Robust algorithms in presence of noise and occlusion [34]

1.1. Implicit polynomials

Implicit curve or surface is a zero set of a polynomial function f. For instance, in the case of a surface, the function is expressed as:

$$\begin{aligned} f_a(x,y,z) &= \sum_{0\le i+j+k\le n} a_{ijk}x^iy^jz^k \\ &= \underbrace{(1\ x\ y\ z\ x^2\ \ldots\ )}_{m(\mathbf{x})^T}\underbrace{(a_{000}\ a_{100}\ a_{010}\ a_{001}\ a_{200}\ \ldots\ )^T}_{\mathbf{a}} \\ &= m(\mathbf{x})^T\mathbf{a} \end{aligned}$$

IP is being seen as product between a monomial vector $m(\mathbf{x})^T$ that only depends on the point (x, y, z), and the parameters vector $\mathbf{a}$.

Formally, an implicit polynomial surface is the solution set of the following equation:

$$m(\mathbf{x})^T\mathbf{a} = 0 \tag{1}$$

where $\mathbf{x}$ is the variable. This solution set is denoted as $Z(f_a)$. In both cases - 2D and 3D - usually implicit polynomial term is used.

The main contribution of this research is to develop an IP fitting algorithm *capable of discovering an optimal polynomial degree to be used* during the fitting process.

This paper presents two contributions. A new distance measure between X and the implicit polynomial is defined, and an

[*] Corresponding author.
E-mail addresses: rinterian@ic.uff.br, rubenus@yandex.ru (R. Interian).

http://dx.doi.org/10.1016/j.cag.2017.05.002
0097-8493/© 2017 Elsevier Ltd. All rights reserved.

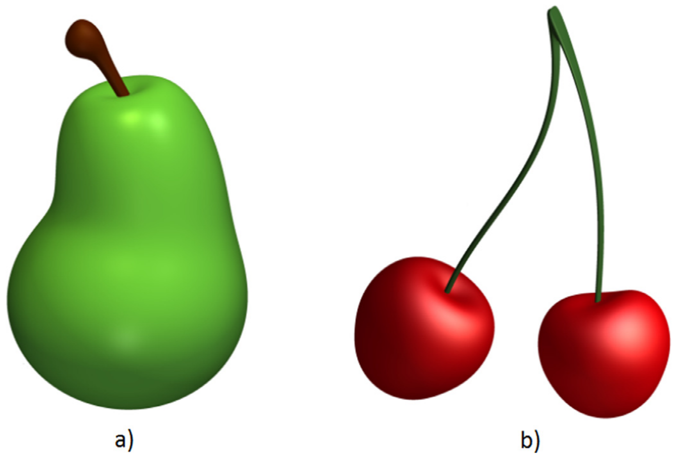


Fig. 1. Implicit Polynomials. (a) "The pear", 6*th* degree IP (b) "The cherries". Taken from [24].

algorithm able to find the optimal degree of the polynomial needed for the representation of the data set is proposed. The goal is to find an implicit polynomial that produces a compact and smooth representation of the point set with the lowest degree as possible, and at the same time minimizes the fitting error. The coefficients of the implicit polynomial can be used then as a descriptor for other tasks, for instance, shape analysis and recognition [7], in which complex and extensive representations are not useful.

1.2. Structure of present work

This work is organized as following. In the Section 2, main IP fitting methods are presented. In the Section 3, a novel distance measure between the point set and an IP is introduced. In the Section 4, fitting algorithm that uses the new measure in order to find IPs with certain desired properties, without previous knowledge of the best degree to be used in the fitting process, is proposed. In the Section 5, experimental results obtained from the proposed method are analyzed.

2. IP fitting methods

Fitting methods are classified as linear or nonlinear according to how the distance $dist(x_i, Z(f_a))$ between a point x_i and the zero set $Z(f_a)$ is defined.

2.1. Nonlinear methods

There is no simple way to find analytically the distance from any point to $Z(f_a)$. Approximate iterative methods are used instead [16,20,33].

The main idea of the nonlinear IP fitting methods is to use the Taubin first order approximation to the exact point-curve or point-surface distance [28,29,31]:

$$dist(\mathbf{x}, Z(f_a)) \approx \frac{|f(\mathbf{x})|}{\|\nabla f(\mathbf{x})\|}$$

This kind of distance is usually named *geometric*, because it uses information from the partial derivatives of f_a.

2.2. Linear methods

The linear methods have been used most [6,36], because they do not require iterative approximations and are faster than nonlinear ones. The most important linear fitting algorithms are classic linear fitting and 3L algorithm.

The linear methods use the algebraic distance:

$$dist(\mathbf{x}, Z(f_a)) \approx f_a(\mathbf{x})$$

The following assumptions are made. By continuity, the value of f_a is close to zero near $Z(f_a)$. It is also assumed that far from $Z(f_a)$, f_a is growing.

In this case, the fitting problem is solved as a overdetermined system $M_{n\times k}\,\mathbf{a}_{k\times 1} = \mathbf{b}_{n\times 1}$ where M is the monomial matrix, which rows are the n monomial vectors of each point in the set X, and $\mathbf{b}$ is (*initially*) a zero vector.

The least squares solution of this overdetermined system [10] is:

$$\mathbf{a} = (M^T M)^{-1} M^T \mathbf{b} = M^{+}\mathbf{b}$$

where M^{+} is the *Moore–Penrose pseudoinverse* [3].

However, the linear algorithms suffer instability, because small changes in the observations can lead to completely different solutions [6]. Furthermore, since $\mathbf{b} = 0$, there is a trivial solution $\mathbf{a} = 0$ that should be avoid. Thereby this classical linear method is improved by other linear algorithms, like the 3L algorithm [6]. The stability of fitting is increased, and the vector $\mathbf{b}$ is substituted by another nonzero vector.

3. New distance measure to an IP

In order to obtain a good fitting algorithm, a new distance measure between a point and an IP is defined. In this regard, firstly two specific measures (called dissimilarity and smoothness measures) are deduced, with the aim of evaluating the "separation" and the "proximity" of the point set to $Z(f_a)$.

3.1. Dissimilarity measure

The dissimilarity measure evaluates *quantitatively* the real distance between the IP and the points.

First order Taubin approximation [31] is chosen for providing fast and analytically substantiated approximation to the point-IP distance:

$$dissim(\mathbf{x}, Z(f_a)) = \frac{|f(\mathbf{x})|}{\|\nabla f(\mathbf{x})\|}$$

Therefore, for calculating the dissimilarity between a point set $\mathbf{X}$ and the IP, we propose to use the following function by averaging the Taubin approximations in each point:

$$dissim(\mathbf{X}, Z(f_a)) = \frac{1}{N}\sum_{i=1}^{N}\frac{|f(\mathbf{x}_i)|}{\|\nabla f(\mathbf{x}_i)\|} \tag{2}$$

This function is called hereinafter *dissimilarity measure* between $\mathbf{X}$ and $Z(f_a)$.

3.2. Smoothness measure

In some of the latest works [21,27,34,35], the idea of taking control of some geometric characteristics of the IPs over the course of the fitting algorithms, is proposed. These characteristics could be: gradient vector norm near the point set, gradient direction relative to some estimated tangent vector in some points, as well as some others. All of them attempt to avoid the effects of overfitting, promoting those polynomials which have desired geometric properties.

For those reasons, "smoothness" measure is also used in order to provide a geometric proximity criterion between the IP and the dataset [34]. This measure evaluates *qualitatively* any IP as an approximation to the given set of points. The advantage of using this *quality* measure consists in obtain only IPs with certain desired topological properties, as shown below.

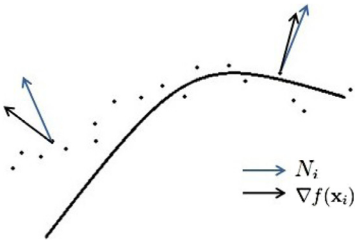


Fig. 2. Smoothness measure. If the directions of the estimated normal vector and the gradient vector are very different, the quality of the fitting is poor.

Given a dataset $\mathbf{X}$, an IP is smooth in $x_i \in \mathbf{X}$ if:

$$\frac{\nabla f(\mathbf{x}_i)}{\|\nabla f(\mathbf{x}_i)\|} N_i \approx 1$$

where N_i is an estimated normalized normal vector in $\mathbf{x}_i$. It can be obtained from physical models or estimated from $\mathbf{X}$.

Care should be taken when generating N_i vectors in a noisy point cloud, since inaccurate generation of these vectors can lead to poor fitting results. There are several algorithms addressing this problem [12,22,23]. In particular, we refer to Sahin method [22,23], in which this issue is tackled. In this work, the existence of a set of N_i vectors computed by any efficient method is assumed.

We use the following expression [35] as the *smoothness measure* of an IP regarding the point set $\mathbf{X}$:

$$smooth(\mathbf{X}, Z(f_a)) = \frac{1}{N}\sum_{i=1}^{N} \frac{\nabla f(\mathbf{x}_i)}{\|\nabla f(\mathbf{x}_i)\|} N_i \tag{3}$$

We note that this function is bounded between -1 and 1 due to the normalization of both, gradient and normal vectors. Nevertheless, in practice, if the directions of the estimated normal vectors are determined accurately, the smoothness measure should never be close to -1. We note that smoothness can be interpreted as the average of cosine values of the angles between the gradient vector and the estimated normal vector, for every point of the dataset. If the cosine value is close to -1, that means the angle is close to 180 degrees. If that happens, it implies inversion of orientation which we should avoid because we are working with orientable surfaces. In the worst case, the gradient can be perpendicular to the estimated normal vector, causing smoothness be zero at this point.

In the Fig. 2, the intuitive idea of the smoothness measure is illustrated.

If the fitting result has good quality, the normalized gradient vector $\frac{\nabla f(\mathbf{x}_i)}{\|\nabla f(\mathbf{x}_i)\|}$ is close to the normal vector N_i in any point $\mathbf{x}_i$.

3.3. Penalization strategy

One of the contributions of this work is to propose a combination of the measures described above as an objective function (from now on OF) used in the fitting process. It is common to use *quantitative* OF in linear and nonlinear fitting methods, i.e. a function that describe an approximation to real distance from some IP to the points. Our approach include an objective function that penalizes non-smooth solutions during the optimization process, promoting higher *quality* IPs.

A strategy based on the idea of penalizing non-smooth IP solutions along the fitting process is described. Given a point set $\mathbf{X}$, we say that an IP is non-smooth if *smooth*($\mathbf{X}$, $Z(f_a)$) ≈ 0, that is:

$$1 - smooth(\mathbf{X}, Z(f_a)) \approx 1$$

Therefore, a new OF that penalizes non-smoothness is defined as the sum of two terms. The first term is the dissimilarity measure. The second term reflects how far the IP is from being smooth.

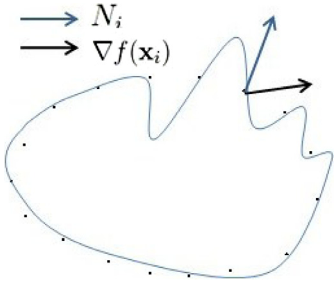


Fig. 3. Overfitting. The implicit polynomial pass close all the points in $\mathbf{X}$; it is well evaluated quantitatively, but has a poor qualitative evaluation, since it can be improved [27] by eliminating the turns it makes when passing through several points.

The positive constant δ has the role of penalty coefficient:

$$dist(\mathbf{X}, Z(f_a)) = dissim(\mathbf{X}, Z(f_a)) + \delta(1 - smooth(\mathbf{X}, Z(f_a)))$$

Substituting the expressions of dissimilarity and smoothness measures, and grouping the terms, we get:

$$dist(\mathbf{X}, Z(f_a)) = \frac{1}{N}\sum_{i=1}^{N} \frac{|f(\mathbf{x}_i)|}{\|\nabla f(\mathbf{x}_i)\|} + \delta\left(1 - \frac{1}{N}\sum_{i=1}^{N} \frac{\nabla f(\mathbf{x}_i)}{\|\nabla f(\mathbf{x}_i)\|} N_i\right)$$

$$dist(\mathbf{X}, Z(f_a)) = \frac{1}{N}\sum_{i=1}^{N} \left(\frac{|f(\mathbf{x}_i)|}{\|\nabla f(\mathbf{x}_i)\|} + \delta\left(1 - \frac{\nabla f(\mathbf{x}_i)}{\|\nabla f(\mathbf{x}_i)\|} N_i\right)\right) \tag{4}$$

This formulation of the distance function attempts to reconcile the two concepts we exposed in past sections: the proximity between the point cloud and the polynomial, and geometrical properties that the polynomial must have. The positive parameter δ indicates how strong is the penalty for not having those desired properties.

The δ parameter acts in a similar way to ridge regularization parameter α [22]. The aim of both parameters is to improve stability. However, there is no equivalence between the two. While in ridge regression the parameter works on global stability [22] (possibly causing local inaccuracy), in our method the value of δ acts locally by fixing gradient directions and improving local stability.

The distance function (4) is taken as the new OF that should be minimized during the fitting process:

$$\min_{f_a} \frac{1}{N}\sum_{i=1}^{N} \left(\frac{|f(\mathbf{x}_i)|}{\|\nabla f(\mathbf{x}_i)\|} + \delta\left(1 - \frac{\nabla f(\mathbf{x}_i)}{\|\nabla f(\mathbf{x}_i)\|} N_i\right)\right) \tag{5}$$

Note that the IP can pass through all the points in $\mathbf{X}$ having positive OF value, for example, as in Fig. 3. We define now the fitting algorithm based on the distance measure (4).

4. An adaptive fitting algorithm

The main contribution of this work is to provide an algorithm capable of finding the degree of the polynomial needed for the representation of the points, without previous knowledge of the complexity of the dataset. This kind of algorithm is called hereinafter “adaptive fitting algorithm”.

Firstly, a fixed degree fitting algorithm is discussed. Subsequently, this algorithm is generalized to the variable degree case.

4.1. Fixed degree fitting

The minimization of the objective function defined above is proposed:

$$\min_{f_a} \frac{1}{N}\sum_{i=1}^{N} \left(\frac{|f(\mathbf{x}_i)|}{\|\nabla f(\mathbf{x}_i)\|} + \delta\left(1 - \frac{\nabla f(\mathbf{x}_i)}{\|\nabla f(\mathbf{x}_i)\|} N_i\right)\right)$$

Besides having not derivable nominator, it is well known that curve and surface fitting problems commonly lead to multimodal functional dependencies of the solution from data, even those associated to linear methods, as explained very well in [6]. This means that the objective function may have several local optimums. Exact methods are not efficient in this kind of problems, converging naturally to closest of them.

Therefore, metaheuristic algorithms are used in order to realize a better exploration of the search space and most important, *refine* several solutions provided by classical fitting methods, improving the quality of those. Those that were designed specifically for continuous problems are chosen, such as Particle Swarm Optimization and Differential Evolution [13,25]. Both have been previously used in curve fitting problems [18].

4.1.1. Optimization by PSO metaheuristic

The Particle Swarm Optimization metaheuristic (PSO) keep a set (swarm) of N particles traveling in a d-dimensional space [13]. Every particle represents a solution, and has associated a position and a speed vector, this last indicating the direction and the step of the movement. Each time some particle moves, it readjust its speed using information from the swarm, such that the search process is directed to promising search space regions. The parameter χ, called restriction factor, allows to bound speed values. In this way, the success of some particles affects the behavior of the others.

This algorithm has only two parameters: χ, the restriction factor, and N, the size of the swarm.

4.1.2. Optimization by DE metaheuristic

The Differential Evolution metaheuristic (DE) is an Evolutionary Algorithm and can be seen as a variation of the Genetic Algorithm [25]. Like any genetic algorithm, DE uses a population of N individuals represented by d-dimensional vectors. In summary, the algorithm generate new individuals using sum and difference operations over vectors in the population. In recent years this metaheuristic has been widely used in many practical problems, specially continuous ones [19].

Besides the population size N, DE has two parameters: weighting factor F, which controls the amplification of the variation obtained from the vector difference; and crossover constant CR, which indicates the percentage of the new individual vector components taken into account in the vectors of the next generation.

4.1.3. Seeded elements

Small number of seeded elements is included into the initial population of the iterative algorithms. Seeded elements typically have good values of the objective function, and is assumed that they increase the fitness of the population.

Exact algorithms solutions can be used as seeded elements. In particular, we use solutions of linear classic and 3L algorithms. The solutions of these linear methods can be obtained very quickly. The proposed number of seeded elements is $\frac{N}{10}$, where N is the total size of the population. The role of the metaheuristic is to improve the quality of the exact method solutions if they have good fitness respect to the distance measure defined above.

4.2. Adaptive fitting

The following question arises: what criteria should be followed for finding the optimum IP degree that best represents some dataset **X**? In order to find an answer, several aspects of this problem are discussed below.

Table 1
Number of IP coefficients (problem dimension) from polynomial degree.

Degree	Coefficients, 2D	Coefficients, 3D
1	3	4
2	6	10
4	15	35
6	28	84
8	45	165
10	66	286
12	91	455
14	120	680
16	153	969

4.2.1. Maximum IP degree selection

Some practical considerations on IP degrees utilization are exposed.

In most research works addressing IP fitting problem, only even degrees are used in experiments or practical examples [6,14,22,34,35].

Taubin's work [32] has the explanation. Zero set of any *odd degree* polynomial is always unbounded[1]. Even degree polynomial zero sets can be bounded or unbounded. Therefore, are only considered even degree polynomials, due to finite nature of the observed datasets.

Furthermore, on the same researches, the most used IP degrees are in the range from 2 to 10. The use of superior degrees (12, 14, 16 and 18) is exceptional. In fact, the advantage of having a compact representation of the object is lost when these degrees are utilized (see the Table 1).

4.2.2. Smoothness measure behaviour analysis increasing IP degree

The key to determining the optimum IP degree is the behavior of the smoothness measure when this degree is increasing. The Fig. 4 shows the evolution of the above mentioned measure of the solutions of two fitting algorithms, for two distinct datasets.

In both cases, the smoothness measure increases, and then stabilizes its growth. In one case, the measure reaches the maximum, and then drops slightly. In the second case, asymptotic behavior is observed. The 3L algorithm solutions have higher quality than the solutions of the classical fitting algorithm.

These considerations lead us to use the change of the smoothness measure as a criterion of closeness to optimum IP degree. If the increment of the smoothness is very small or negative (the smoothness decreases) the “best” degree is reached.

4.2.3. Summary of the adaptive fitting algorithm

Taking into consideration all the above, the following fitting algorithm is proposed:

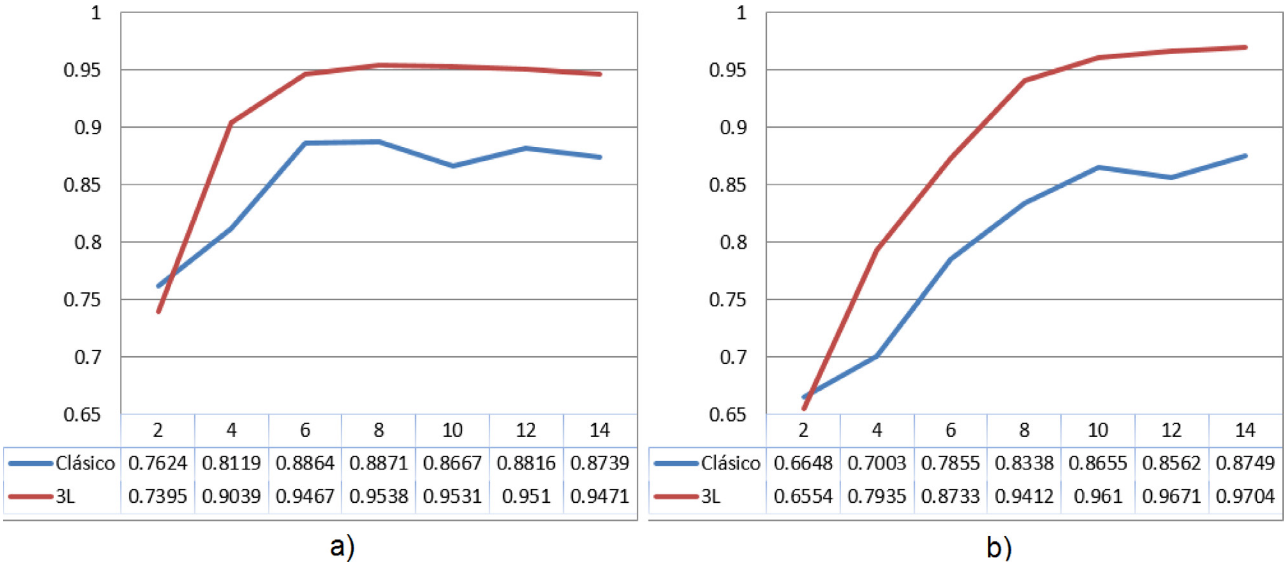
The stopping criterion of this algorithm is the non-increment of the smoothness measure in some ϵ. If $\epsilon = 0$, this is equivalent to a decrement of the smoothness.

One advantage of using this stopping criterion in the algorithm is that it has a clear geometric interpretation. The smoothness is the average of cosine values of the angles between the gradient vector and the estimated normal vector, for every point of the dataset. For instance, setting the value $\epsilon = 0.01$ in the stopping criterion represents a reduction of those angles in approximately 1°.

The following properties of the adaptive fitting algorithm can be exploited:

- It is able to find the degree of the IP that is necessary to get a good fitting result *without any previous knowledge of the complexity of the dataset*.

[1] For instance, any line in the plane, or any plane in the space, are zero sets of one-degree polynomials, and also unbounded.



a)	2	4	6	8	10	12	14
Clásico	0.7624	0.8119	0.8864	0.8871	0.8667	0.8816	0.8739
3L	0.7395	0.9039	0.9467	0.9538	0.9531	0.951	0.9471

b)	2	4	6	8	10	12	14
Clásico	0.6648	0.7003	0.7855	0.8338	0.8655	0.8562	0.8749
3L	0.6554	0.7935	0.8733	0.9412	0.961	0.9671	0.9704

Fig. 4. Smoothness measure behaviour increasing IP degree, datasets: (a) “Boot”, (b) “Horse” (*Large Geometric Models Archive*, *Georgia Institute of Technology* [1]).

- It introduces variability of the solutions, allowing to generate a set of them as output of the algorithm. With this we capitalize the multimodal nature of the IP fitting problem[2], which is an advantage over exact algorithms.
- It utilizes and improves the quality of the IPs that are solutions of the classic algorithms. This can be seen as a post-optimization process. The metaheuristics can generate new IPs that are unrelated to classic solutions, if they have desired geometric properties.

Algorithm 1 Adaptive fitting algorithm.

```
$G \leftarrow$ Maximum degree to be used for the current dataset
$g \leftarrow 2$
while $g \le G$ do
    $best\_elements_g \leftarrow FixedDegree(g)$
    if $g > 2$ and $smooth(best\_elements_g) - smooth(best\_elements_{g-1}) \le \epsilon$ then
        return $best\_elements_{g-1}$
    else
        $g \leftarrow g + 2$
    end if
end while
return $best\_elements_g$
```

5. Experimental results

Several issues concerning the practical implementation of the proposed algorithms are clarified.

One aspect that should be considered when applying fitting algorithms is the need of centering the dataset **X** at the coordinate origin. Furthermore, the dataset must be scaled (for example, dividing the points by their average distance from the coordinate origin). These transformations avoid numerical problems in fitting algorithms.

The experiments evaluate the performance of the adaptive fitting method in real 2D and 3D scenarios. Our main goal is to evaluate the quality of the solutions. The algorithm should generate curves and surfaces that are interpretable and similar to the original 2D and 3D objects.

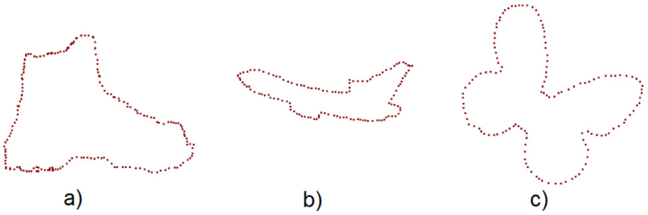


Fig. 5. 2D datasets. (a) Boot, (b) Airplane, (c) Butterfly.

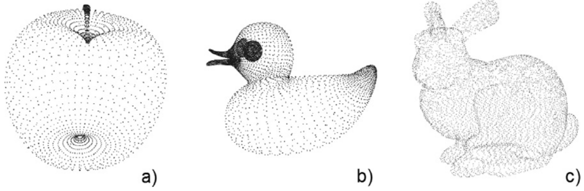


Fig. 6. 3D datasets. (a) Apple, (b) Rubber duck, (c) Stanford Bunny.

5.1. Datasets

The 2D and 3D datasets are selected as follows.

Point clouds are taken from widely known repositories, such as Stanford 3D Scanning Repository [2] and others (Fig. 6). In particular, we take the following models:

- Apple
- Rubber duck
- Stanford Bunny

In the case of 2D models, plane figures that appear frequently in implicit polynomial fitting research [6,27,35] are chosen (Fig. 5). There is not neither consensus nor repositories containing the ideal datasets for two-dimensional fitting. Moreover, 2D models are easily reproducible by researchers in the area.

5.2. Parameters of the algorithms

The following parameters were used in the experiments.

[2] A multimodal problem has many local optima, which may have close values of the objective function.

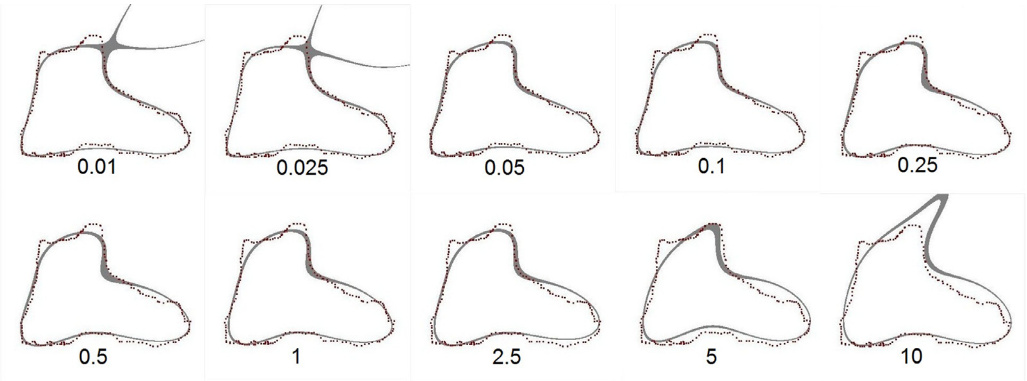


Fig. 7. Dataset “Boot”: setting parameter δ, 4th degree. Below each image is shown the value of δ.

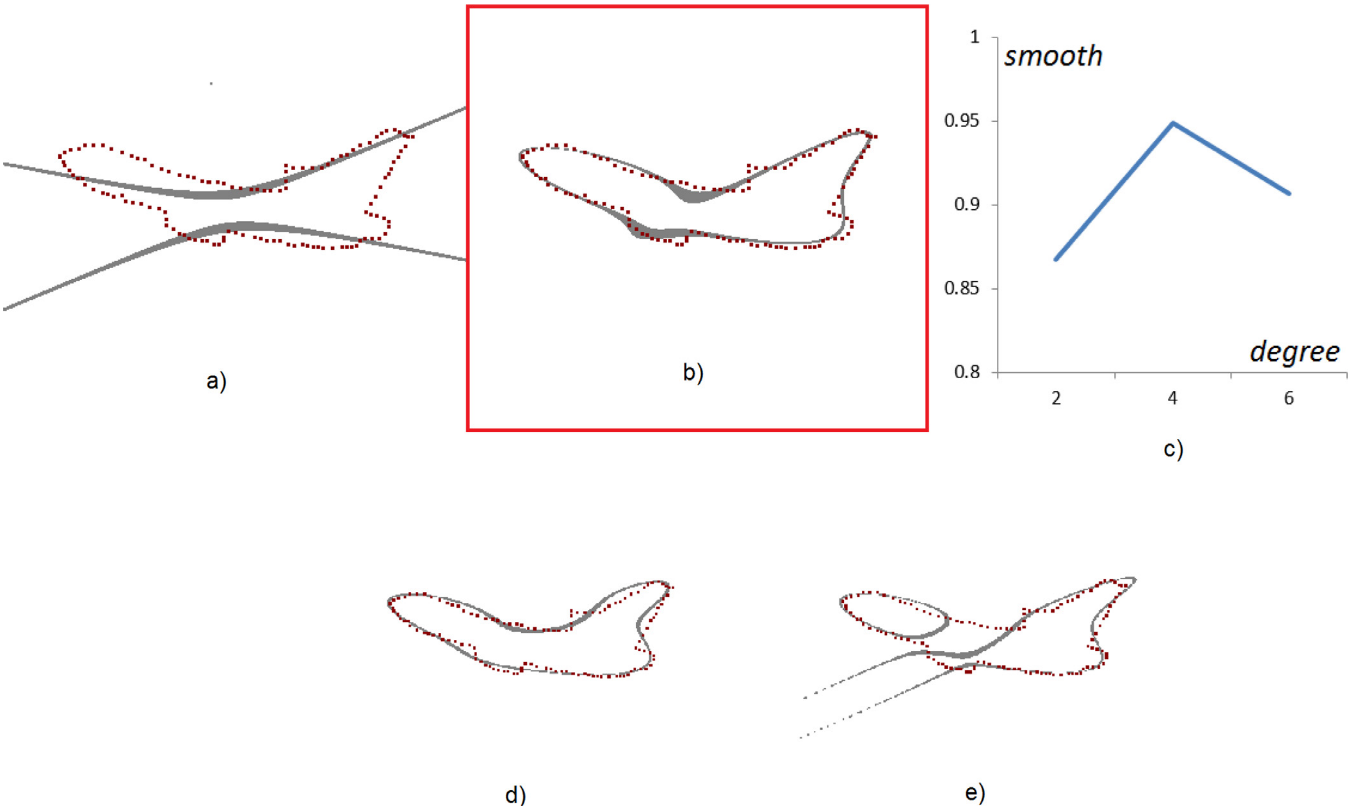


Fig. 8. Dataset “Airplane”, adaptive fitting run: (a) 2nd degree fit, (b) 4th degree fit (optimal degree), (c) evolution of the smoothness measure during the execution, (d) 3L fit for the 4th degree, (e) lineal classic fit for the 4th degree.

In DE and PSO metaheuristics, the population (swarm) size is set to four times the dimension of the problem [17], that is, four times the dimension of the parameters vector **a**. As stopping condition in both metaheuristics, the criterion of reaching 500 iterations is used. The run times of both algorithms are very similar.

In PSO metaheuristic, value $\chi = 0.729$ is taken, as suggested in [9].

In DE metaheuristic, values for weighting factor $F = 0.7$ and crossover constant $CR = 0.9$ are assumed, as recommended in [26].

The penalization parameter δ from the objective function (5) is estimated in Section 5.3.

Finally, for 3D models, fractions of the total number of points in the original dataset are taken, since the cardinality of these sets is very large, becoming 35,947 points in the dataset Stanford Bunny. This is because these models are often used in research in the field of computer graphics, where local accuracy is critical.

5.3. Setting parameter δ

The parameter δ introduced in the fitting algorithm, determines how strong is the penalty for having too low values of the smoothness measure.

Setting this parameter correctly is essential to obtain results that can be used and interpreted correctly, because a low value can eliminate the necessary effect of the penalty, and a high value can cause the implicit polynomial to stay away from the original set of observations.

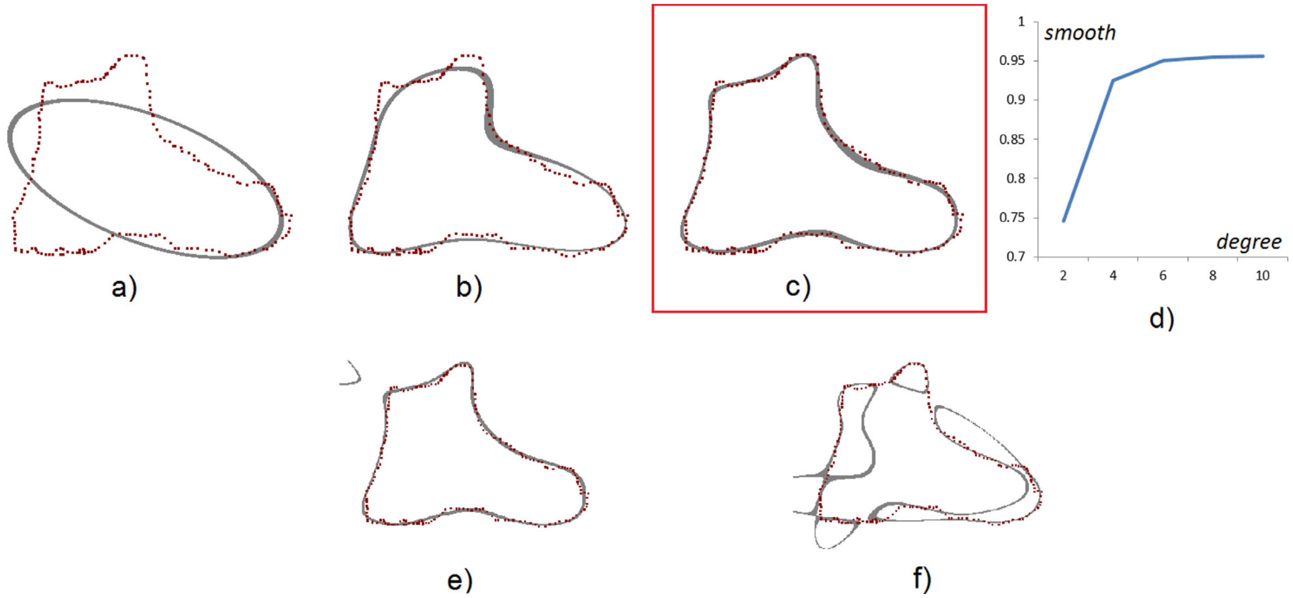


Fig. 9. Dataset “Boot”, adaptive fitting run: (a) 2nd degree fit, (b) 4th degree fit, (c) 6th degree fit (optimal degree for $\epsilon = 0.01$), (d) evolution of the smoothness measure during the execution, (e) 3L fit for the 6th degree, (f) lineal classic fit for the 6th degree.

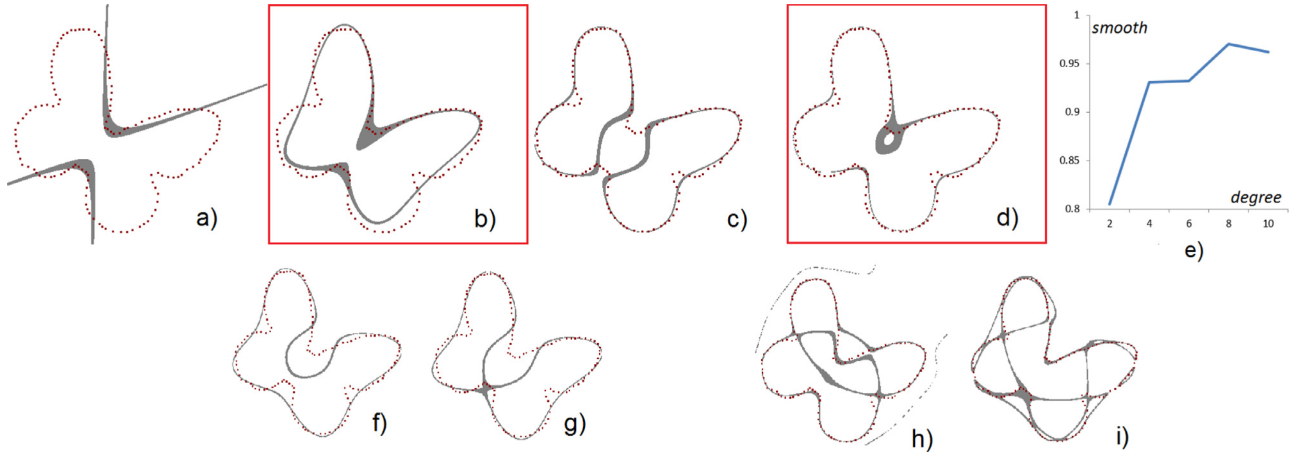


Fig. 10. Dataset “Butterfly”, adaptive fitting run: (a) 2nd degree fit, (b) 4th degree fit (optimal degree for $\epsilon = 0.01$), (c) 6th degree fit, (d) 8th degree fit (optimal degree for $\epsilon = 0$), (e) evolution of the smoothness measure during the execution, (f) 3L fit for the 4th degree, (g) lineal classic fit for the 4th degree, (h) 3L fit for the 8th degree, (i) lineal classic fit for the 8th degree.

This fact is illustrated in the Fig. 7. Unstable behavior of the algorithm for small values of δ, as well as imprecise results for the large values, is observed. There is also a wide range of acceptable values for δ. We use the value $\delta = 0.25$ in the experiments described below.

5.4. Analysis of adaptive fitting results

The results of adaptive fitting algorithm for the different datasets are presented.

5.4.1. Fitting 2D curves

Adaptive fitting algorithm runs for 2D datasets are shown (Figs. 8–10), comparing the obtained results with classic linear and 3L algorithm results.

As observed, the adaptive algorithm can determine the degree of implicit polynomial needed to obtain an interpretable fitting.

5.4.2. Fitting 3D surfaces

Just as in 2D, experimental runs of the adaptive fitting algorithm for 3D datasets are executed. The Figs. 11–13 show the obtained results as well as the fits obtained by 3L and classic linear algorithms for the same datasets.

It is remarked that the algorithm is able to represent complex objects by relatively low degree implicit polynomials (degrees 2, 4, 6, 8).

5.4.3. Discussion of the graphical results

The results we presented in Figs. 8–13 can be divided into two groups. The first one contains datasets in which the smooth function has a strict maximum (Figs. 8, 11 and 12). In this cases, any nonnegative value for ϵ leads to easily identifiable optimum degree, in particular, $\epsilon = 0$ or $\epsilon = 0.01$.

The second group contains point clouds where the smoothness is a strictly increasing function (like Figs. 9 and 13). In order to reach correct results, the value of ϵ must be strictly positive, for example $\epsilon = 0.01$.

There is also a “Butterfly” dataset (Fig. 10), an example of a smoothness function with a strict maximum that has some unusual behavior. In this case, any of the aforementioned values of ϵ (0 and 0.01), leads to different but interpretable results.

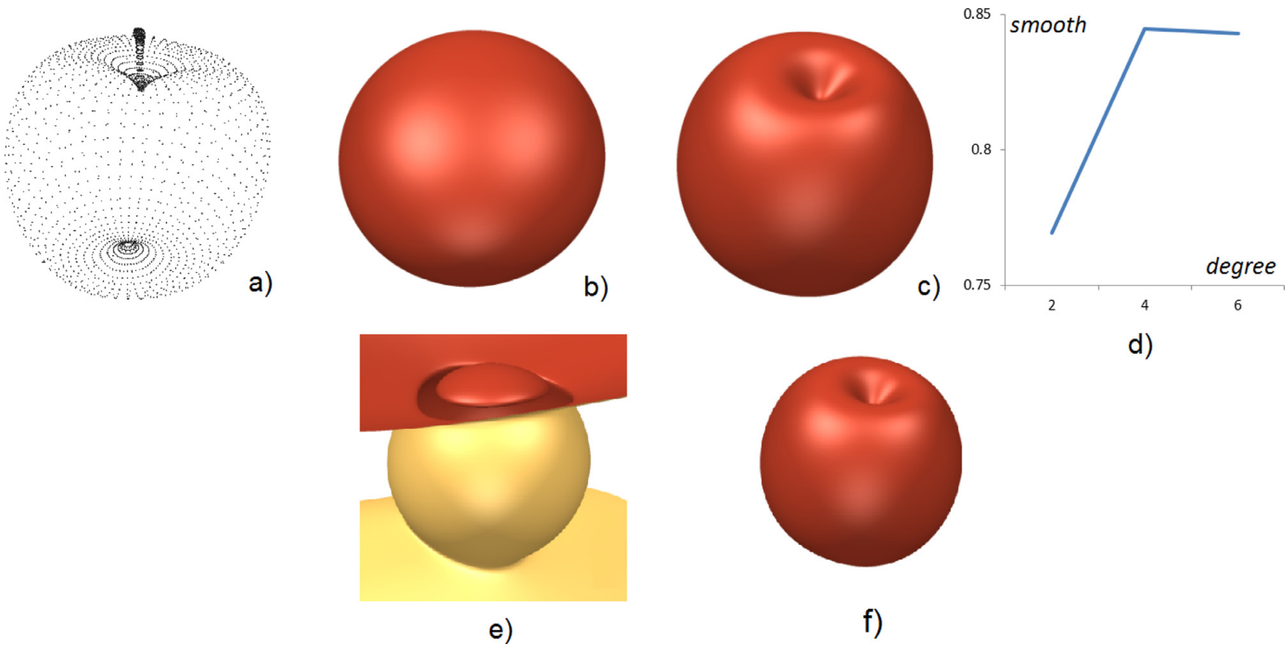


Fig. 11. Dataset "Apple", adaptive fitting run: (a) original point cloud, (b) 2nd degree fit, (c) 4th degree fit (optimal degree), (d) evolution of the smoothness measure during the execution, (e) lineal classic fit for the 4th degree, (f) 3L fit for the 4th degree.

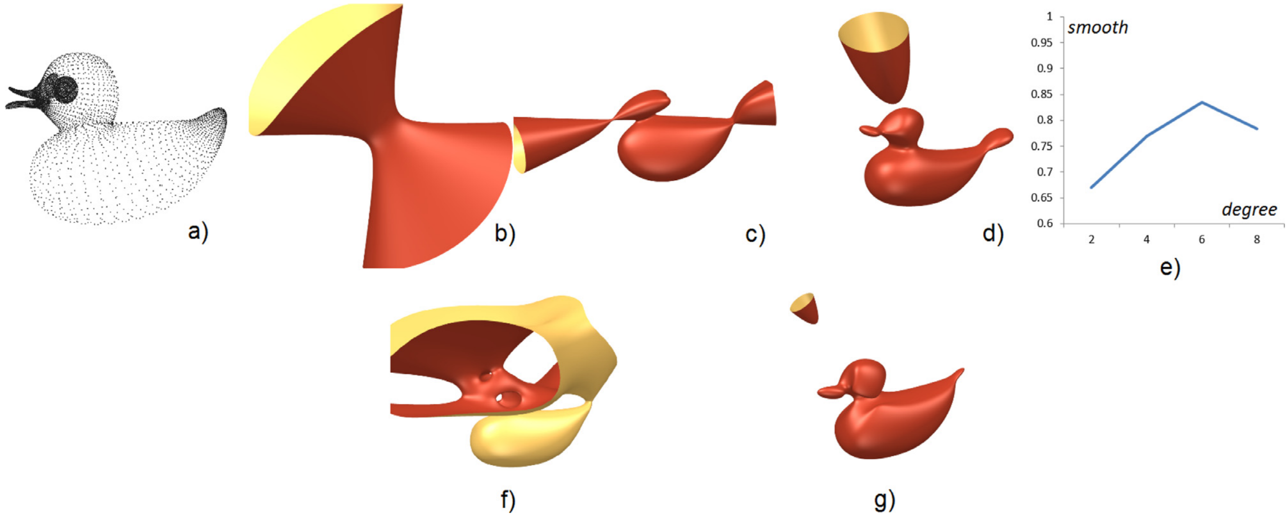


Fig. 12. Dataset "Rubber duck", adaptive fitting run: (a) original point cloud, (b) 2nd degree fit, (c) 4th degree fit, (d) 6th degree fit (optimal degree), (e) evolution of the smoothness measure during the execution, (f) lineal classic fit for the 6th degree, (g) 3L fit for the 6th degree.

All these examples suggest that that the value of ϵ, however small, must be strictly positive. The value 0.01 seems to be a good strategy in all 6 cases we analyzed.

5.5. Comparison with 3L method plus ridge regression regularization

Ridge regression regularization (RRR) is a method that improves global stability of other linear methods, like 3L algorithm. It consists in adding the term kD to the M^TM matrix used in these methods (see Section 2 and [22] for further details). The variable k is the ridge regression parameter that should be small enough to preserve properties of the original matrix, but large enough to achieve global stability.

We now compare the performance of our method against 3L with RRR (3L+RRR) using some selected values of k parameter. The results are shown in Figs. 14 and 15. We use 3L+RRR with the same degree founded by our algorithm. Unlike the 3L method with ridge regression regularization, our method preserves important shape features, like a pronounced corner in the tail of the airplane, or the central part of the butterfly. The ridge regression is unable to achieve this kind of results for these datasets, since it only cares about global stability. Increasing k parameter, 3L+RRR fitting results tend to be more "circular", eliminating corners and protrusions.

5.6. Fitting in presence of noisy data

To analyze the behavior of the algorithm when the data is noisy, we generated Gaussian noise in the original point clouds, with different standard deviation values. The results are presented in Fig. 16. Normal vectors N_i are generated using a simple algorithm presented in [12]. Our fitting algorithm always stopped in degree 4. The presented empirical evidence indicates that the method is

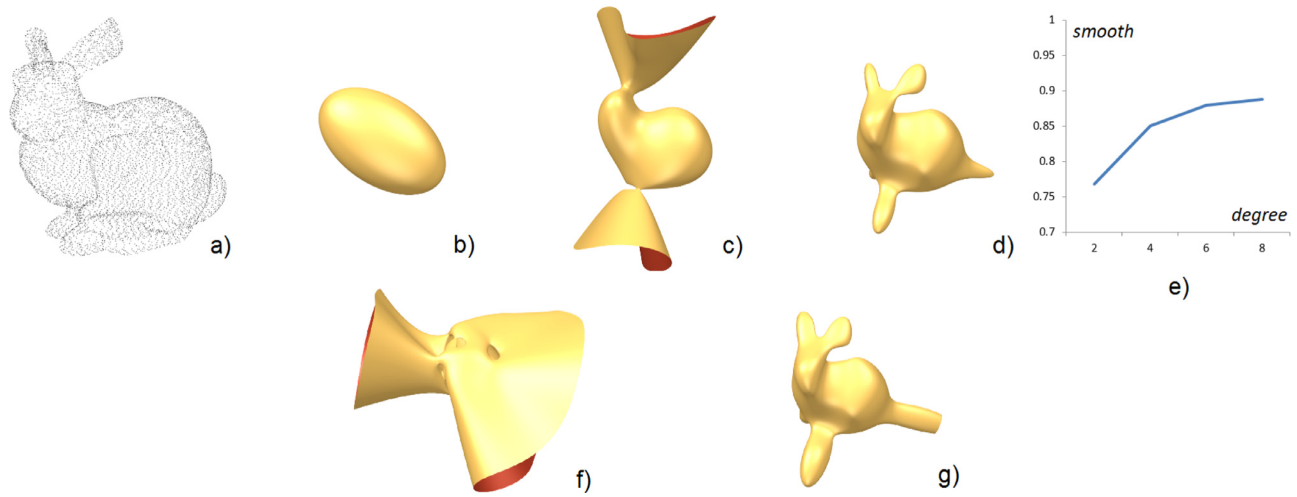


Fig. 13. Dataset “Stanford Bunny”, adaptive fitting run: (a) original point cloud, (b) 2nd degree fit, (c) 4th degree fit, (d) 6th degree fit (optimal degree for $\epsilon = 0.01$), (e) evolution of the smoothness measure during the execution, (f) lineal classic fit for the 6th degree, (g) 3L fit for the 6th degree.

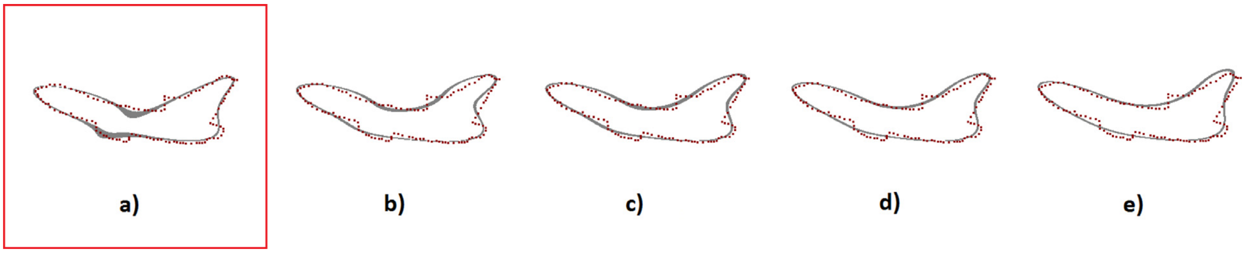


Fig. 14. Dataset “Airplane”, comparing with 3L+RRR method: (a) Our method, (b) 3L method without RRR ($k = 0$), (c) 3L+RRR, $k = 10^{-8}$, (d) $k = 3 * 10^{-8}$, e) $k = 10^{-7}$.

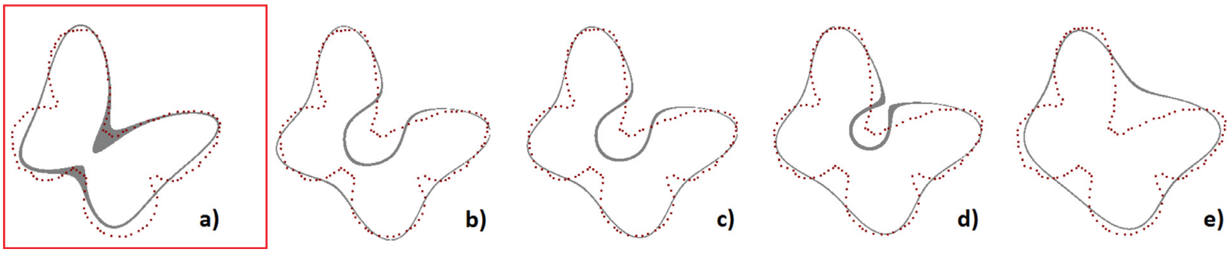


Fig. 15. Dataset “Butterfly”, comparing with 3L+RRR method: (a) Our method, (b) 3L method without RRR ($k = 0$), (c) 3L+RRR, $k = 10^{-5}$, (d) $k = 3 * 10^{-5}$, (e) $k = 10^{-4}$.

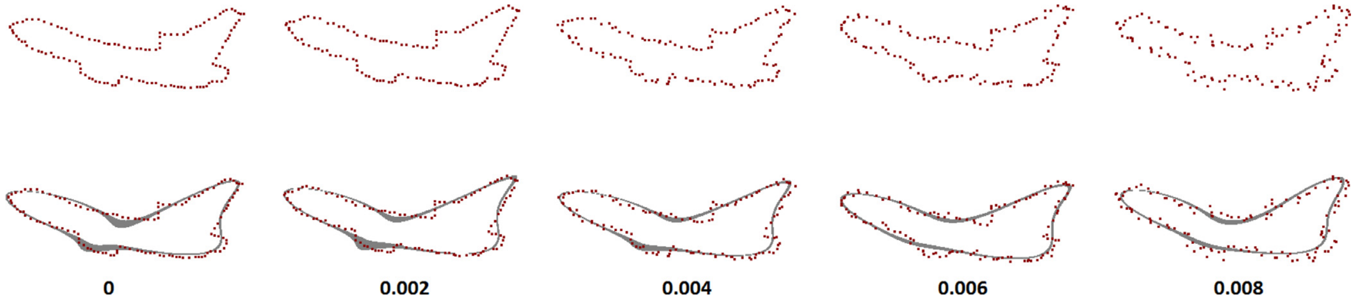


Fig. 16. Fitting in presence of noisy data. Below each image is shown the standard deviation value of Gaussian noise.

little sensitive to noise, which allows it to be used for recognition tasks.

6. Conclusions

Although the implicit polynomials are not the most locally precise representation scheme, they are very useful for applications requiring a compact registration of data from a complex real-world object, in order to perform a process of recognition [4] of the same, or other objects that correspond to the pattern. With the aim of finding an IP that properly represents some dataset, a fitting process is performed. The vast majority of fitting algorithms requires knowledge of the IP degree that best represents the points. This work presents an alternative to these fitting algorithms.

Firstly, an objective function to be used during the fitting process is defined. This function is characterized by including a penalty of undesired geometric properties (non-smoothness) in the polynomial.

Then, heuristic adaptive fitting algorithm is proposed, which is able to find the degree of the implicit polynomial that is necessary to represent the dataset. The algorithm is based on the idea of gradually increasing the degree of the IP, while there is an improvement in the smoothness of the solutions.

This algorithm is beneficial in comparison to others for three reasons: it can automatically find the required degree of the implicit polynomial, it can offer a set of solutions in contexts where variability (options) is required, and it can post optimize solutions of known fitting algorithms (classical linear and 3L), in order to obtain better quality solutions.

The experiments confirm the validity of the approach for the selected 2D and 3D datasets, since the fits made by the proposed algorithm are interpretable.

Acknowledgments

Ruben Interian has been sponsored by a CAPES scholarship. The work of Celso C. Ribeiro was partially supported by CNPq and FAPERJ research grants. Anselmo A. Montenegro is grateful for the fund from FAPERJ.

References

[1] Greg Turk and Brendan Mullins. Large geometric models archive, Georgia Institute of Technology. http://www.cc.gatech.edu/projects/large_models/ (accessed May 2017).
[2] The Stanford 3D Scanning Repository, Stanford Computer Graphics Laboratory. http://graphics.stanford.edu/data/3Dscanrep/ (accessed May 2017).
[3] Ben-Israel A. Generalized inverses: theory and applications. second. Springer; 2001.
[4] Ben-Yaacov H. Recognition of 3D objects based on implicit polynomials. Irwin and Joan Jacobs Center for Communication and Information Technologies; 2009.
[5] Berger M, Levine J, Nonato LG, Taubin G, Silva CT. An end-to-end framework for evaluating surface reconstruction. Sci Comput Imag Inst 2011.
[6] Blane MM. The 3l algorithm for fitting implicit polynomial curves and surfaces to data. IEEE Trans Pattern Anal Mach Intell 2000;22:298–313.
[7] Bronstein AM, Bronstein MM, Kimmel R. Numerical geometry of non-rigid shapes. Monographs in Computer Science. 1st. New York: Springer-Verlag; 2009.
[8] Campbell RJ, Flynn PJ. A survey of free-form object representation and recognition techniques. Comput Vis Image Underst 2001;81(2):166–210. http://dx.doi.org/10.1006/cviu.2000.0889.
[9] Eberhart R, Shi Y. Comparing inertia weights and constriction factors in particle swarm optimization. Proc Congr Evol Comput 2000;1:84–8.
[10] Golub G. Matrix computations. third. The Johns Hopkins University Press; 1996.
[11] Gomes AJP, Voiculescu I, Jorge J, Wyvill B, Galbraith C. Implicit curves and surfaces: mathematics, data structures and algorithms. 1st. London: Springer-Verlag; 2009.
[12] Hoppe H, DeRose T, Duchamp T, McDonald J, Stuetzle W. Surface reconstruction from unorganized points. SIGGRAPH Comput Graph 1992;26(2):71–8.
[13] Kennedy J, Eberhart R. Particle swarm optimization. Proceedings of the IEEE international conference on neural networks, 4; 1995.
[14] Landa Z. 2D object description and recognition based on contour matching by implicit polynomials. In: The 18th European Signal Processing Conference Proceedings; 2006. p. 1796–800.
[15] Liu T, Liu W, Qiao L, Luo T, Peng X. Point set registration based on implicit surface fitting with equivalent distance. Proceedings of the IEEE international conference on image processing; 2015.
[16] Mederos V, Estrada J. A new algorithm to compute the euclidean distance from a point to a conic. Invest Oper 2002;23.
[17] M. Pedersen, Good parameters for particle swarm optimization (2010), technical report no. HL1001 of Hvass Laboratories.
[18] Polo-Corpa M. Curve fitting using heuristics and bio-inspired optimization algorithms for experimental data processing in chemistry. Chemom Intell Lab Syst 2009;96:34–42.
[19] Price K, Storn R. Differential evolution - a practical approach to global optimization. Springer; 2006.
[20] Rouhani M. Implicit polynomial representation through a fast fitting error estimation. IEEE Trans Image Process 2012;21:2089–98.
[21] Rouhani M. The richer representation the better registration. IEEE Trans Image Process 2013;22:5036–49.
[22] Sahin T. Fitting globally stabilized algebraic surfaces to range data. Proceedings of the IEEE international conference of computer vision, 2; 2005.
[23] Sahin T, Unel M. Stable algebraic surfaces for 3D object representation. J Math Imag Vis 2008;32:127–37.
[24] Schenzel P. On the interactive visualization of implicit surfaces. Martin-Luther University Halle Press; 2012.
[25] Storn R, Price K. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 1997;11:341–59.
[26] Talbi E. Metaheuristics from design to implementation. John Wiley & Sons, Inc.; 2009.
[27] Tasdizen T. Improving the stability of algebraic curves for applications. IEEE Trans Image Process 2000;9:405–16.
[28] Taubin G. Nonplanar curve and surface estimation in 3-space. Proceedings of the IEEE international conference on robotics and automation,; 1988. p. 644–5.
[29] Taubin G. Estimation of planar curves, surfaces, and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation. Pattern Anal Mach Intell IEEE 1991;13:1115–38.
[30] Taubin G. Object recognition based on moment (or algebraic) invariants. Geometric invariance in computer vision. MIT Press; 1992. p. 375–97.
[31] Taubin G. Distance approximations for rasterizing implicit curves. ACM Trans Graph 1994;13.
[32] Taubin G. Parameterized families of polynomials for bounded algebraic curve and surface fitting. IEEE Trans Pattern Anal Mach Intell 1994;16:287–303.
[33] Taubin G, Cooper D. Symbolic and numerical computation for artificial intelligence. Academic Press; 1992.
[34] Zheng B. 2D curve and 3D surface representation using implicit polynomial and its applications. University of Tokyo; 2008. Ph.D. thesis.
[35] Zheng B. An adaptive and stable method for fitting implicit polynomial curves and surfaces. IEEE Trans Pattern Anal Mach Intell 2010;32:561–8.
[36] Zheng B. Breast MR image fusion by deformable implicit polynomial (DIP). IPSJ Trans Comput Vis Appl 2013;5:99–103.