Vol.2009-CVIM-167 No.4
2009/6/9

Improvements of IP Representation, Fitting and Registration

Bo Zheng,†1 Jun Takamatsu†2 and Katsushi Ikeuchi†1

Among function-based shape representation techniques, representation in implicit polynomial (IPs) focuses attention in the vision community, because IPs are superior especially in the areas of fast fitting, few parameters, algebraic/geometric invariants, robustness against noise and occlusion, *etc.* Despite these excellent characteristics, still IP mainly suffers from three issues: difficulty of fine representation for complex objects, difficulty of determining moderate degree for fitting and difficulty of being used for partial object registration. Addressing these issues, in this paper, first a 3D IP-segment representation method is developed even for robustly representing complex objects. Second, an computationally efficient fitting method is proposed for adaptively estimating the IP of moderate degree to the complexity of object shapes. Third, a computationally efficient and robust object registration method using IP gradient field is presented. With these methods, this paper provides the insights and extendible applicabilities into the theory and practice of IP representation.

1. Introduction

2D curve and 3D surface models are widely used for a variety of purposes in computer vision and graphics. There exist efficient function-based shape representation techniques such as B-spline, NURBS[20], Rational Gaussian[6], radial basis function[31] and Implicit Polynomial (IP). Among these techniques, representation in IPs focuses attention from researchers in the vision community who, in particular, are developing sophisticated techniques for the automatic recognition, registration and matching of 2D/3D objects. In contrast to other representations, IPs are superior especially in the areas of fast fitting, few parameters, algebraic/geometric invariants, robustness against noise and occlusion, etc. Therefore, representing 2D and 3D data sets with implicit polynomials (IPs) is currently attractive for such applications as fast shape registration[2),4),13),21),25),26),28),29], recognition[12),15),17),24),25),33),34], smoothing and denoising[22),28], 3D reconstruction[11], image compression[7], and image boundary estimation[27].

Despite the excellent characteristics, still IP mainly suffers from the following the issues on following three aspects that greatly limit its applicability and urgently require to be addressed:

- **IP representation** – difficulty for modeling complex objects
  Objects obtained by vision modalities are often very precise and thus complex. Unfortunately, the prior methods[2),13),28),29] which used one polynomial for representing a complex object are often restricted, and generating an IP even with high degrees is nearly impossible to give fine shape representation for a complex object due to numerical instability and high computational cost.
- **IP fitting** – difficulty in determining a moderate IP degree and achieving global/local stability
  There have been great improvements concerning IP fitting methods with its increased use during the late 1980s and early 1990s[12),29]; Recently, new robust and consistent fitting methods like 3L fitting[2], gradient-one fitting[28] and Rigid regression[22),28] make them feasible for object recognition tasks. However, unfortunately, these methods require that the degree of IP must be determined before handling fitting and fixed in the fitting procedure, and thus they are difficult to determine a moderate degree for a complex object. Furthermore IP representation suffers from global instability that many redundant zero-level sets are generated around the desired one. This makes the fitting result nearly impossible to be interpreted in the desired region space.
- **IP registration** – difficulty of being used for partial object registration
  The most popular registration approaches are the Iterative Closest Point (ICP) based methods[1),18]. ICP-based methods are effective to the fine registration, but inevitable to require the extra computation for finding the point-wise correspondences. Recently, distance field is considered to be constructed for achieving the registration[10),16] that needs to spend much memory to preserve the distance field for 3D models and the support region for registration is limited in the regions where distance field has been generated.

†1 The University of Tokyo
†2 Nara Institute of Science and Technology

Compare to these method, prior work reported that IP model is more attractive to coarse registration with higher performance derived by a single (non-iterative) computation[26),30)]. However the registration of using IP is only applied into global registration, that is, it is only suitable to deal with the case when two objects are globally modeled but not partially overlapped, which limited thus IP registration only into some specific applications[23)].

This paper focuses on addressing the above three issues. For the first issue, we propose a IP-segment representation method against the inaccuracy when modeling the complex shapes. To this end, we develop a 3D surface segmentation method based on a cut-and-merge approach, through which a complex surface can be divided into segments such that each segment can be encoded by a low-degree IP. The advantages are i) it achieves an appropriate segmentation for any surface model even for a complex one and the segmentation precise is determined by the desired fitting accuracy; ii) each segment successfully avoids to use over-high degree IPs and thus the total computational cost becomes little; and iii) because each segment has been encoded by an IP, it has IP's inherent algebraic/geometric properties which are very useful for shape matching problem. The experimental results even open up a new vista on the application of non-iterative registration for range images, which has potential to fast initial registration for range images.

For the second issue, we propose an incremental fitting scheme that can adaptively determine the moderate degree required to different objects. Our method increases the degree of IP until a satisfactory fitting result is obtained. The incrementability of QR decomposition with Gram-Schmidt orthogonalization gives our method computational efficiency. Furthermore, since the decomposition detects the instability element precisely, our method can selectively apply ridge regression-based constraints to that element only. As a result, our method achieves computational stability while maintaining fitting accuracy and still keeps little computational cost.

Given the stable IP fitting method, we exploit a new object registration technique for solving the third issue that can efficiently achieve 2D-3D/3D-3D registration not only for globally modeled objects but also for the partially overlapped objects. Over the prior methods, the advantages of our method are that: i) unlike the ICP-based methods, it avoids the extra computation for point-wise correspondences; ii) unlike the coarse registration methods, it totally supports partial-overlapping registration; iii) unlike the registration methods of preserving discrete distance field, it only needs a little memory space for preserving a few IP's coefficients, and it can support registration in wider (infinite) region. Furthermore its high performance attracts a new application on 6-DOF pose estimation for single Ultrasound image.

The rest of this paper is organized as follows. Section 2 provides the mathematical background needed for later sections. Section 3 presents our IP-segment representation method. Section 4 describes our incremental scheme for IP fitting. Section 5 provides our registration method with IP models. Each section reports its experimental results and the conclusion is given by Section 6.

2. Mathematical Background

2.1 Definition of IP

IP is the implicit function defined in a multivariate polynomial form. For example, the 3D IP of degree n is denoted by:

$$\begin{aligned} f_n(\mathbf{x}) &= \sum_{0\le i,j,k;i+j+k\le n} a_{ijk}x^iy^jz^k \\ &= \underbrace{(1\ x\ \ldots\ z^n)}_{\mathbf{m}(\mathbf{x})^T}\underbrace{(a_{000}\ a_{100}\ \ldots\ a_{00n})}_{\mathbf{a}}{}^T, \end{aligned} \tag{1}$$

where $\mathbf{x} = (x\ y\ z)$ is one data point in the data set. An IP can be represented as an inner product between the monomial vector and the coefficient vector as $\mathbf{m}(\mathbf{x})^T\mathbf{a}$. The order for monomial indices $\{i, j, k\}$ is a degree-increasing order named *inverse lexicographical order*[30)]. The *homogeneous binary polynomial* of degree r in x, y, and z, $\sum_{i+j+k=r} a_{ijk}x^iy^jz^k$, is called the r-th degree form of the IP.

2.2 Fitting Methods

The objective of IP fitting is to find a polynomial $f(\mathbf{x})$, of which the zero set $\{\mathbf{x}|f_n(\mathbf{x}) = 0\}$ can "best" represent the given data set. This fitting problem can be formulated in least squares optimization manner as:

$$M^TM\mathbf{a} = M^T\mathbf{b}, \tag{2}$$

where M is the matrix of monomials whose i-th row is $\mathbf{m}(\mathbf{x}_i)$ (see Eq. (1)); $\mathbf{a}$

is the unknown coefficient vector; and $\mathbf{b}$ is a zero vector. Note, Eq. (2) is just transformed from the least squares result, $\mathbf{a} = M^{\dagger}\mathbf{b}$, where $M^{\dagger} = (M^TM)^{-1}M^T$ is the so-called pseudo-inverse matrix.

Many efforts have been made for solving the linear system of equations (2) by first overcoming the singularity of M^TM and $\mathbf{b} = \mathbf{0}$. Then, the problem can be solved simply with a linear system solver such as the LU decomposition method, the conjugate gradient (CG) method, singular value decomposition (SVD), or their variations. A common technique for improving singularity of M^TM and making $\mathbf{b} \neq \mathbf{0}$ is to add some additional constraints to the computation, see 2), 8), 28).

In this paper, we employed the *3L method*[2] that takes two additional data layers at a distance $\pm\varepsilon$ outside and inside the original data as the optimization constraint. Thus M is modified by adding the rows of monomials of additional data points and $\mathbf{b}$ is modified by adding the elements of $\pm\varepsilon$ s.

2.3 Ridge Regression (RR) Regularization for Stable Fitting

Although the linear methods improve the numerical stability to some extent, they still suffer from the difficulty of achieving global stability. Especially while modeling a complex shape with a high-degree IP, many extra zero sets are generated. One important reason for global instability is the collinearity of column vectors of matrix M, causing the matrix M^TM to be nearly singular (see 28)).

Addressing this issue, Tasdizen *et al.* [28] and Sahin and Unel[22] proposed using ridge regression (RR) regularization in the fitting, which improves (that is, decreases) the condition number of M^TM by adding a term κD to the diagonal of M^TM, where κ is a small positive value called the RR parameter, and D is a diagonal matrix. Accordingly Eq. (2) can be modified as

$$(M^TM + \kappa D)\mathbf{a} = M^T\mathbf{b}. \tag{3}$$

In particular, D should be calculated in a specific way to maintain Euclidean invariance indicated by Tasdizen *et al.* for 2D[28] and Sahin and Unel for 3D[22].

2.4 Similarity Measurement

Let us define two functions to measure the similarity for fitting between an IP and data set $\mathcal{S}$ as follows:

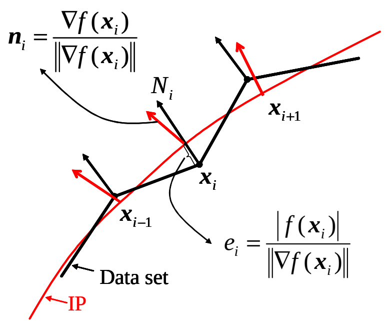
$\boldsymbol{n}_i = \frac{\nabla f(\boldsymbol{x}_i)}{\|\nabla f(\boldsymbol{x}_i)\|}$
N_i
$\boldsymbol{x}_{i+1}$
$\boldsymbol{x}_i$
$\boldsymbol{x}_{i-1}$
$e_i = \frac{|f(\boldsymbol{x}_i)|}{\|\nabla f(\boldsymbol{x}_i)\|}$
Data set
IP

Fig. 1 Definition of two types of our similarity measurement: one is based on the distance between data point and IP zero set, and the other is based on the difference between normal directions associated with data point and IP zero set.

$$D_{dist} = \frac{1}{N}\sum_{i=1}^{N} \frac{|\ f(\mathbf{x}_i)\ |}{\|\ \nabla f(\mathbf{x}_i)\ \|}, \quad x_i \in \mathcal{S} \tag{4}$$

$$D_{smooth} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{N}_i \cdot \frac{\nabla f(\mathbf{x}_i)}{\|\ \nabla f(\mathbf{x}_i)\ \|}), \quad x_i \in \mathcal{S}. \tag{5}$$

where N is the number of vertices, and $\mathbf{N}_i$ is the normal associated with the point $\mathbf{x}_i$, as shown in Fig. 1; ∇f denotes the gradient, *e.g.*, $\nabla f = (\frac{\partial f}{\partial x}\ \frac{\partial f}{\partial y}\ \frac{\partial f}{\partial z})$ in the 3D case. Note, although $\frac{|f|}{\|\nabla f\|}$ in (4) is not a real Euclidean distance, it is proved to be useful for approximating the Euclidean distance from vertex $\mathbf{x}_i$ to the zero set of $f(\mathbf{x})$, see 29).

D_{dist} and D_{smooth} can be considered as two measurements of distance (mean of the perpendicular distances from vertices to IP) and smoothness (bending and twisting) between the point set and the zero set of an IP.

3. IP-segment representation

As described in Section 1, an IP often suffers from the difficulty of representing complex models. The low-degree IP leads to undesired inaccuracy, whereas the high degree leads to global instability. Therefore, in order to accurately describe such a complex model without losing IP's advantages, the method to segment its surfaces and to represent each of them using an IP separately is often used.

In this section, we propose a solution for solving the above problems based on a novel approach on surface segmentation. In this method, a complex surface can be divided into some meaningful segments and each segment can be well represented by a low-degree IP. First, we define the degree of the fitness, that is, the similarity between an IP and a point data set. We use this similarity to decide how well the fitting is performed. Next, we describe our cut-and-merge strategy in detail, which consists of following two procedures:

- Cutting procedure:
  Iteratively cutting the regions until each segment can be represented by an IP and neither high-curved nor distorted region exists.
- Merging procedure:
  Iteratively merging the regions which are acceptable for the same IP.

3.1 Cutting procedure

Accordingly two thresholds $T_1 (T_1 > 0)$ and $T_2 (T_2 < 1)$ are set into the constraint of

$$(D_{dist} < T_1) \ \wedge \ (D_{smooth} > T_2). \tag{6}$$

If this constraint is satisfied, we say the current region is *IP representable*, otherwise *IP unrepresentable*.

The *cutting procedure* can be viewed as a procedure that divides the IP unrepresentable data set into IP representable segments. For achieving that, there are mainly two steps required as: i) we use the IP of a certain degree to measure a surface region with constraint (6). ii) if the constraint is satisfied, this region is regarded as IP representable and will be outputted. Otherwise, this region is regarded as IP unrepresentable and will be divided into two parts: Inner part ($\{InnerRg\} := \{\mathbf{x}_i | f(\mathbf{x}_i) \leq 0\}$) and Outer part ($\{OuterRg\} := \{\mathbf{x}_i | f(\mathbf{x}_i) > 0\}$). We repeat the above operation until there are no more IP unrepresentable regions.

3.2 Removing High-curved segment

Because in the results from the above cutting procedure there might still exist highly curved or distorted regions, an additional cutting procedure is required that can find and divide the high-curved segments into low-curved regions.

To find the curved regions, we take advantage of IP's property that can provide a convenient way to find the surface curvatures quickly and robustly. If the normal vector $\mathbf{n}_i$ at vertex $\mathbf{x}_i$ is approximated by the first order partial derivative of f. And if $\mathbf{g}_i = \bigtriangledown f(\mathbf{x}_i) = (\frac{\partial f(\mathbf{x}_i)}{\partial x} \ \frac{\partial f(\mathbf{x}_i)}{\partial y} \ \frac{\partial f(\mathbf{x}_i)}{\partial z})^T$, then $\mathbf{n}_i = (n_x \ n_y \ n_z)^T = \mathbf{g}_i / |\mathbf{g}_i|$. The second order derivatives in (7) contain information about the curvature of isosurfaces of the implicit function.

$$\bigtriangledown \mathbf{n}_i^T = \begin{pmatrix} \frac{\partial n_x}{\partial x} & \frac{\partial n_x}{\partial y} & \frac{\partial n_x}{\partial z} \\ \frac{\partial n_y}{\partial x} & \frac{\partial n_y}{\partial y} & \frac{\partial n_y}{\partial z} \\ \frac{\partial n_z}{\partial x} & \frac{\partial n_z}{\partial y} & \frac{\partial n_z}{\partial z} \end{pmatrix}. \tag{7}$$

It can be solved as follows (see the derivation in 14)):

$$\bigtriangledown \mathbf{n}_i^T = \frac{1}{|\mathbf{g}_i|} \mathrm{GH}, \tag{8}$$

where $\mathrm{G} = \mathrm{I} - \mathbf{n}_i \mathbf{n}_i^T$; I is the 3×3 identity matrix; and H is the Hessian matrix:

$$\mathrm{H} = \begin{pmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial x \partial z} \\ \frac{\partial^2 f}{\partial x \partial x} & \frac{\partial^2 f}{\partial y^2} & \frac{\partial^2 f}{\partial y \partial z} \\ \frac{\partial^2 f}{\partial x \partial z} & \frac{\partial^2 f}{\partial y \partial z} & \frac{\partial^2 f}{\partial z^2} \end{pmatrix}.$$

From the eigenvalues of the matrix $\bigtriangledown \mathbf{n}_i^T$, the so-called principal curvatures, k_1 and k_2 are obtained. And the maximum/minimum absolute curvatures are defined as

$$\kappa_{max} = \max(|k_1|, |k_2|) \tag{9}$$

$$\kappa_{min} = \min(|k_1|, |k_2|). \tag{10}$$

We extract the points if they satisfy the constraint:

$$\kappa_{max} / \kappa_{min} > K, \tag{11}$$

where K is a certain threshold for the ratio of maximum and minimum absolute curvatures. That is what we are now interested in are the angled regions that contain the *ridge* and *valley*, namely the region where κ_{max} is large and κ_{min} is relatively small. Thus we call the extracted points valley or ridge.

Once the high-curved regions are found, using the information of the vertices and their normals on the valley/ridge curves, we try to fit a new IP, and cut the curved region again with this IP. If we let vertices on the ridge and the points along their normals be the points passed through by the IP zero set, we can fit a new IP whose zero set crosses the ridge curve. Thus, according to the sign of each point measured by the IP function, the region can be cut into two parts, the inner part and the outer part.

3.3 Merging procedure

The task of the merging procedure is to merge the over-segmented regions to an integral region, since the regions resulting from the cutting procedure may be over-segmented.

This procedure makes use of the region-grow strategy[19] where the seed region is selected and then is merged with its neighbors if possible. The criterion for judging whether a neighbor should be merged into the seed region is the same as the constraint (6). The difference is that we use the seed's IP to measure the neighbors. Thus the measurement described in (4) and (5) should be replaced by:

$$e_i = \frac{\mid f_{seed}(\mathbf{x}_i) \mid}{\parallel \bigtriangledown f_{seed}(\mathbf{x}_i) \parallel}, \quad x_i \in \mathcal{N}, \tag{12}$$

$$\mathbf{n}_i = \frac{\bigtriangledown f_{seed}(\mathbf{x}_i)}{\parallel \bigtriangledown f_{seed}(\mathbf{x}_i) \parallel}, \quad x_i \in \mathcal{N}, \tag{13}$$

where f_{seed} is the IP function corresponding to the seed region, and $\mathcal{N}$ is a neighbor region of the seed region. The constraint in (6) is still used as the measurement of the similarity between a seed IP and its neighbor. In this procedure, the larger region is first to be chosen as a seed region.

3.4 Experimental Results

In this subsection, we show some other experimental results to prove the effectiveness of our method. A simple example of segmented IP surface representation of a synthetic data is shown in Fig. 2; the shape has been contaminated by adding noises to one percent of its height. Five segmented IP surfaces for the object are shown in Fig. 2 (b), and its corresponding segmentation result is shown in Fig. 2 (c).

Other complex examples are shown in Figs. 3 and 4. The original range data are shown in column (a) respectively. Then each of these images is represented with IP surfaces in different segmentation levels, by changing the thresholds of (6), shown in columns (b) and (c) of each figure. The figures in column (d) show the segmentation results corresponding to the IP surfaces shown in column (c) respectively.

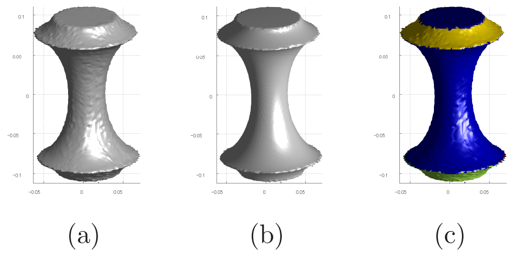
Fig. 2 (a) Original range data. (b) 5 4-degree IP surfaces. (c) 5 segments with different colors corresponding to (b).

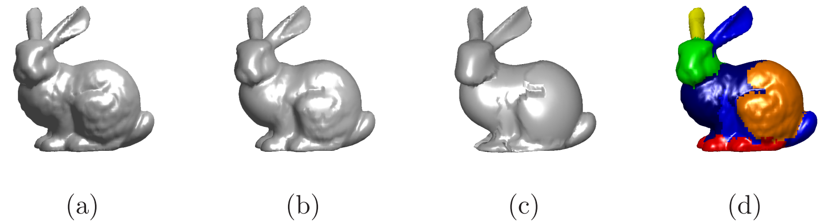
Fig. 3 (a) Original range data. (b) 100 4-degree IP surfaces. (c) 8 4-degree IP surfaces. (d) 8 segments referring to (c).

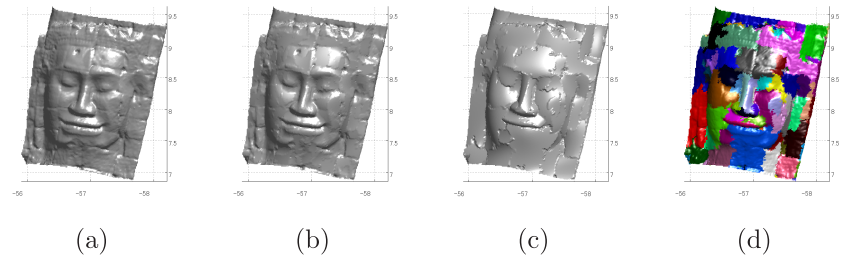
Fig. 4 (a) Original range data. (b) 763 4-degree IP surfaces. (c) 113 4-degree IP surfaces. (d) 113 segments referring to (e).

3.5 Application to Non-iterative Registration for Range Images

We extend the previous result to the application of 3D registration (alignment). 3D registration is one of the important sub-steps for large-scale 3D modeling. Most of these registrations need to manually align the corresponding scans for

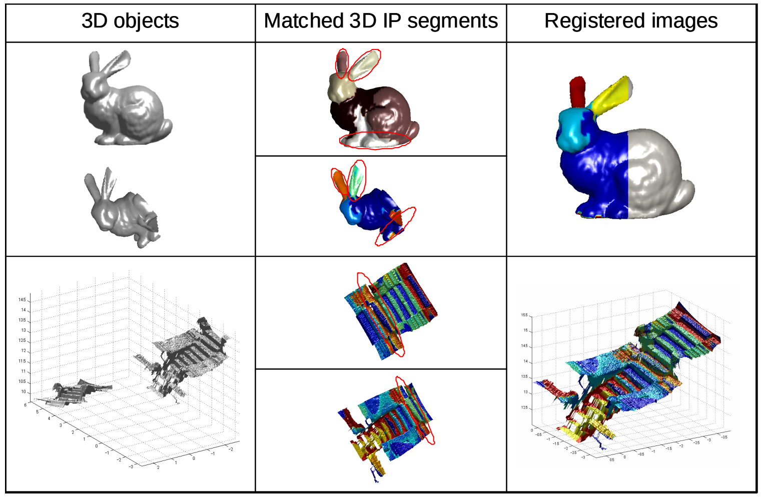

Fig. 5 Two examples on registration for range images

the initial process, and then use an ICP-based algorithm such as 18) to achieve further accuracy in alignment. We refer this survey to 23).

Taubin and Cooper gave a simple method for calculating IP's orientation and center of mass from the coefficients of IP's leading form 30), which made the registration (or pose estimation) fast and simple. Fortunately, this method can be applied to the results of our segmentation method.

Fig. 5 shows two simple examples of this application. In the first example, we took half of the original "bunny" and transform it to another position using random Euclidean transform; the second example is for the real range images obtained by scanning the stairs in Bayon Temple[9]. The raw data are plotted in same coordinate but different position, see Fig. 5. Then we tested the registration. The process can be described in three steps: i) **Segmentation**: segmenting the objects with the same thresholds; ii) **Matching**: calculating invariants and finding the matching pairs in the same way as in the recognition application[30]; iii) **Registration**: calculating the orientation and center of the matched segments by Taubin's method[30] and then estimating the transformation.

Although there are some errors in the registration results, since our segmentation method cannot give exactly the same segments between the original "bunny" and the partial "bunny." But this result is acceptable as the result of the initial alignment process (coarse registration) to be succeeded by a fine registration such as the ICP process[18]. The advantages of this method are that it is fully automatic, fast, simple and free to any point-to-point corresponding procedures.

4. An Adaptive and Stable Method for IP Fitting

In this section, we present an incremental scheme using QR decomposition for the fitting methods, which allows the IP degree to increase during the fitting procedure until a moderate fitting result is obtained. The computational cost is also saved because each step can completely reuse the calculation results of the previous step.

4.1 Incremental Fitting

In this subsection, first we describe the method for fitting an IP with the QR decomposition method. Next, we show the incrementability of Gram-Schmidt QR decomposition. After that, we clarify the amount of calculation needed to increase the IP degree.

Without solving the linear system (2) directly, we first carry out QR decomposition on matrix M as: $M = Q_{N\times m}R_{m\times m}$, where Q satisfies: $Q^TQ = I$ (I is an $m \times m$ identity matrix), and R is an invertible upper triangular matrix.

Then, substituting $M = QR$ into Eq. (2), we obtain:

$$R^TQ^TQR\mathbf{a} = R^TQ^T\mathbf{b} \rightarrow R^TR\mathbf{a} = R^TQ^T\mathbf{b} \rightarrow R\mathbf{a} = Q^T\mathbf{b} \rightarrow R\mathbf{a} = \widetilde{\mathbf{b}}. \quad (14)$$

After upper triangular matrix R and vector $\widetilde{\mathbf{b}}$ are calculated, the upper triangular linear system can be solved quickly in $\mathcal{O}(m^2)$.

4.1.1 Gram-Schmidt QR Decomposition

Our incremental algorithm depends on the QR-decomposition process. We found that the *Gram-Schmidt* QR Decomposition is very suitable and powerful for saving the computational cost for our algorithm (discussed in the next subsection).

Now, first let us briefly describe the QR decomposition based on the Gram-Schmidt orthogonalization. Assume that matrix M consisting of columns $\{\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_m\}$ is known. The Gram-Schmidt algorithm orthogonalizes $\{\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_m\}$ into the orthonormal vectors $\{\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_m\}$ that are the columns of matrix Q, and simultaneously calculates the corresponding upper triangular matrix R consisting of elements $r_{i,j}$. The algorithm is written in an inductive manner. Initially let $\mathbf{q}_1 = \mathbf{c}_1 / \parallel \mathbf{c}_1 \parallel$ and $r_{1,1} = \parallel \mathbf{c}_1 \parallel$. If $\{\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_i\}$ have been computed at the i-th step, then the $(i+1)$-th step for orthonormalizing vector $\mathbf{c}_{i+1}$ is

$$\begin{aligned} &r_{j,i+1} = \mathbf{q}_j^T \mathbf{c}_{i+1}, \quad \text{for } j \le i, \\ &\mathbf{q}_{i+1} = \mathbf{c}_{i+1} - \sum_{j=1}^{i} r_{j,i+1} \mathbf{q}_j, \\ &r_{i+1,i+1} = \parallel \mathbf{q}_{i+1} \parallel, \\ &\mathbf{q}_{i+1} = \mathbf{q}_{i+1} / \parallel \mathbf{q}_{i+1} \parallel . \end{aligned} \tag{15}$$

With the Gram-Schmidt algorithm, matrix M is successfully QR-decomposed as $M = QR$, and thus the problem of solving Eq. (2) can be transformed to solve a linear system with an upper triangular coefficient matrix.

Furthermore, from the Gram-Schmidt algorithm, we can see that Gram-Schmidt orthogonalization can be carried out in an incremental manner, which orthogonalizes the columns of M one by one.

4.1.2 Incremental Scheme

The idea of our incremental scheme is to continuously solve upper triangular linear systems (14) in different dimensions where the QR Decomposition with Gram-Schmidt orthogonalization (15) is utilized. This process is illustrated in Fig. 6, where the dimension of the upper triangular linear system increases, and then the coefficient vectors of different degrees can be solved.

We designed this incremental scheme not only because, at each step, solving an upper triangular linear system is much faster than solving a square one, but also because the calculation for dimension increment between two successive steps is computationally efficient. Fig. 6 illustrates this efficiency by clarifying necessary calculation from the i-th step to the $(i+1)$-th step in our incremental process. For this calculation, in fact, it is only necessary to calculate the parts that are illustrated with dark gray blocks in Fig. 6.

For constructing the $(i+1)$-th upper triangular linear system from the i-th one, we are concerned with two types of calculation:

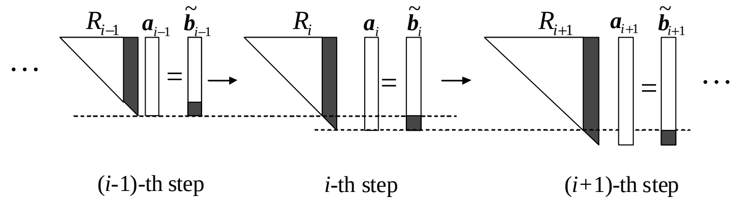

Fig. 6 The incremental scheme that iteratively keeps solving an upper triangular linear system. In order that the triangular linear system grows from the i-th to $(i+1)$-th step, only the calculation shown in dark gray is required; other calculation is omitted by reuse at the i-th step.

i) How to calculate the upper triangular matrix R_{i+1},
ii) How to calculate the right-hand vector $\widetilde{\mathbf{b}}_{i+1}$.

Taking advantage of Gram-Schmidt QR Decomposition, we find that, for the first calculation, that is, growing from R_i to R_{i+1}, we only need to calculate the rightmost column of R_{i+1}, and the other elements of R_{i+1} are not changed from R_i; for the second calculation, that is, growing from $\widetilde{\mathbf{b}}_i$ to $\widetilde{\mathbf{b}}_{i+1}$, we only need to calculate the bottom element of $\widetilde{\mathbf{b}}_{i+1}$, and the other elements of $\widetilde{\mathbf{b}}_{i+1}$ are not changed from $\widetilde{\mathbf{b}}_i$.

For the first calculation, the result can be simply obtained from Gram-Schmidt orthogonalization in Eq. (15). For the second calculation, assuming $\widetilde{b}_{i+1}$ is the bottom element of vector $\widetilde{\mathbf{b}}_{i+1}$, the calculation of $\widetilde{b}_{i+1}$ can be stated as $\widetilde{b}_{i+1} = \mathbf{q}_{i+1}^T \mathbf{b}$ after carrying out the $(i+1)$-th step of Gram-Schmidt orthogonalization in Eq. (15).

In order to clarify the computational efficiency, let us assume a comparison between our method and a brute-force method, such as the 3L method[2)], that iteratively calls the linear method at each step for obtaining the coefficient vectors of different degrees. It is obvious that, for solving coefficient $\mathbf{a}$ at the i-th step, our method needs i inner-product operations for constructing the upper triangular linear system (see (15)), and $\mathcal{O}(i^2)$ for solving this linear system; whereas the latter method needs i^2 inner-product operations for constructing linear system (2), and $\mathcal{O}(i^3)$ for solving Eq. (2).

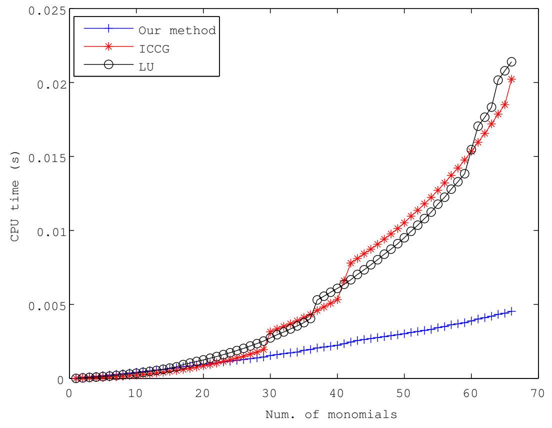

Fig. 7 A comparison between our method and the prior methods with respect to calculation time.

We show a simple example in Fig. 7 to compare the actual calculation time between our method described above and two other methods that solve the linear system (2) independently at each step with two famous linear system solvers: i) LU decomposition and ii) incomplete Cholesky conjugate gradient (ICCG). The result was taken from the mean of $10,000$ calculations for solving the same 2D fitting problem that incrementally fits an IP to a certain data set until achieving the 10th degree; practically the IPs of about the 10th degree are often required to represent complex shapes. As shown in this figure, our incremental scheme performs much faster than the other two methods during the dimension-growing process. Note that, although the ICCG algorithm is very efficient for solving a large-scale sparse linear system, matrix $M^T M$ in linear system (2) is often a dense matrix, and therefore ICCG cannot result in good performance.

4.2 Global/Local Stabilization

As described in Sections 1 and 2, linear fitting methods usually suffer from the difficulty of achieving global stability. Although ridge regression (RR) regularization can be adopted to achieve global stability[22),28)], local accuracy might deteriorate too much. Addressing this problem, in this section we present a computationally simple procedure to detect whether a specific monomial will contribute to the accuracy of IP representation during the incremental steps. If it will not, RR is used to stabilize the fitting at the current step.

To explain the procedure, we first clarify how RR regularization achieves global stability and why it sacrifices local accuracy. Then we propose a new RR regularization-based stabilization technique: our regularization can concentrate on the improvement of stability by selectively applying RR regularization to our incremental scheme.

4.2.1 RR Constraints

First let us analyze why the numerical instability occurs. In fact, an important reason is the collinearity of matrix M, which causes its covariance matrix $M^T M$ to be nearly singular. The collinear columns of M are degenerated to contribute very little to the overall shape of the fit (see 28)). But this little contribution may result in the sensitivity for a high-degree IP, e.g., divergence of some coefficient values. As a result, there are extra undesired solutions generated.

Now let us interpret RR regularization in[5),22),28)] to be some individual constraints that we call RR constraints hereafter. According to the conventional definition in[5)], the formula of RR regularization shown in Eq. (3) can be equivalently transformed as

$$\hat{M}^T \hat{M}\mathbf{a} = \hat{M}^T \hat{\mathbf{b}}, \tag{16}$$

where $\hat{M}$ is the matrix combining matrix M and the square roots of diagonal elements of D, and vector $\hat{\mathbf{b}}$ is from the extension of $\mathbf{b}$ with zeros. Eq. (16) is similar to Eq. (2). The only difference between them is that there are some additional row vectors at the bottom of matrix $\hat{M}$, and actually these additional row vectors act as the linear constraints in fitting. Let us call the constraints RR constraints.

These RR constraints overcome the singularity of matrix $\hat{M}^T \hat{M}$ and thus keep the IP presentation globally stable, but an excessive brute-force constraint manner causes all the zero set to be closed to its origin (see 28)), and thus local accuracy deteriorates. Our claim is that if we can apply the RR constraints only to the necessary part, such as the part corresponding to the monomial that weakly contributes to the whole IP representation, the RR regularization can resist deterioration of local accuracy.

4.2.2 Proposed RR Regularization

First, it is necessary to detect global instability. Fortunately, in our incremental process, since matrix M has been QR-decomposed as $M = QR$, we can observe that $M^T M = R^T R$, and thus instability of $M^T M$ can be determined from the eigenvalues of R. We can easily evaluate the singularity of R by observing only the diagonal values at each step, since upper triangular matrix R's eigenvalues always lie on its main diagonal.

If the value of the diagonal element r_{ii} at the i-th step is relatively too small, we can assume the current column $\mathbf{c}_i$ of M might be nearly collinear to the previously generated orthogonal space of $\{\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_{i-1}\}$ so as to have little contribution to the overall shape of fit. Thus, we apply the i-th RR constraint only to this part to concentrate on solving this collinearity. The action is as follows.

Let us suppose matrix $\hat{M}$ denotes the matrix M after having added the i-th row vector of RR constraint: $(0, \ldots, 0, \sqrt{\kappa d_{ii}})$, where d_{ii} is the i-th diagonal element of D. We assume its QR decomposition is $\hat{M} = \hat{Q}\hat{R}$ and each element after the constraint is denoted with $\hat{}$. The difference between M and $\hat{M}$ is only the last row whose last element is only non-zero. From Eq. 15, this effect propagates only r_{ii}, $\mathbf{q}_i$, and $\widetilde{b}_i$. The norm of $\hat{\mathbf{q}}_i$ before normalizing is $\sqrt{r_{ii}^2 + \kappa d_{ii}}$, where r_{ii} is the norm of $\mathbf{q}_i$. Therefore, $\hat{r}_{ii}$ is derived as follows:

$$\hat{r}_{ii} = \sqrt{r_{ii}^2 + \kappa d_{ii}}. \tag{17}$$

From the derivation,

$$\begin{aligned}\hat{b}_i &= \hat{\mathbf{q}}_i^T \begin{pmatrix} \widetilde{\mathbf{b}} \\ 0 \end{pmatrix} \\ &= \frac{r_{ii}}{\sqrt{r_{ii}^2 + \kappa d_{ii}}} \mathbf{q}_i^T \widetilde{\mathbf{b}} = \frac{r_{ii}}{\hat{r}_{ii}} \widetilde{b}_i. \end{aligned} \tag{18}$$

From Eq. (17) we can see obviously that once the i-th eigenvalue r_{ii} is relatively too small, it can be improved to be a larger one as $\hat{r}_{ii}$ by adding the ith RR constraint. Following this stability improvement, coefficient $\hat{a}_i$ at this step is thus calculated as

$$\hat{a}_i = \frac{\hat{b}_i}{\hat{r}_{ii}} = \frac{r_{ii}}{r_{ii}^2 + \kappa d_{ii}} \widetilde{b}_i \ (< \frac{\widetilde{b}_i}{r_{ii}} = a_i). \tag{19}$$

Therefore we can see that the divergence of a_i is restrained by adding the i-th RR constraint.

In short, at an incremental step, once a column vector $\mathbf{c}_i$ is detected to be linearly dependent on the preceding columns, the computation in Eqs. (17), (18) and (19) should be done.

However, checking the value of r_{ii} might be unfair for judging the collinearity. In the result obtained from Gram-Schmidt orthogonalization, r_{ii} might be related not only to the degree of the collinearity but also to the norm of the corresponding column of M. In order to remove only the effect of the norm, it is necessary to normalize the linear system (2). The normalizing operation can be described as follows

$$M := \left\{ \frac{\mathbf{c}_1}{\| \mathbf{c}_1 \|}, \frac{\mathbf{c}_2}{\| \mathbf{c}_2 \|}, \ldots, \frac{\mathbf{c}_n}{\| \mathbf{c}_n \|} \right\}, \ \mathbf{b} := \frac{\mathbf{b}}{\| \mathbf{b} \|}, \tag{20}$$

assuming originally $M = \{\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_n\}$. Therefore the finally obtained coefficients need the transformation as: $\mathbf{a} = \{\frac{\|\mathbf{b}\|}{\|\mathbf{c}_1\|} a_1, \frac{\|\mathbf{b}\|}{\|\mathbf{c}_2\|} a_2, \ldots, \frac{\|\mathbf{b}\|}{\|\mathbf{c}_n\|} a_n\}$.

4.3 Finding the Moderate Degree

Now the coefficients of various degrees can be worked out stably by the incremental method described above. The remaining problem is how to measure the moderation of degrees for these resolved coefficients. In other words, when should we stop the incremental procedure?

4.3.1 Stopping Criterion

A naive method is to define a stopping criteria such as constraint Eq. (6) that let T_1 and T_2 correspond to data noise levels in statistics. Then we can let T_1 and T_2 be close to zero and one respectively for smooth models and more tolerant values for coarse ones. But for the data set with variation of noise level or whose noise condition cannot be observed, an alternative way is considered as

$$(R_{dist} < T_1) \ \wedge \ (R_{smooth} > T_2). \tag{21}$$

where R_{dist} and R_{smooth} are residuals for errors of distance and smoothness respectively, so that we only see the difference between the errors of current and previous steps. An example of using this stopping criterion is shown in Section 4.5.

4.4 Algorithm for Finding the Moderate IPs

Given the above conditions, our algorithm can be simply described as follows:

i) Constructing the upper triangular linear system with i column vectors of M with the method described in Section 4.1;

ii) Stabilization with the method described in Section 4.2;

iii) Solving current linear system to obtain coefficient vector $\mathbf{a}$;

iv) Measuring the similarity for the obtained IP;

v) Stopping the algorithm if the stopping criterion (6) or (21) is satisfied; otherwise going back to i) and increasing the dimension by adding the $(i+1)$-th column of M.

4.5 Experimental Results

Our experiments are set in some pre-conditions. i) As a matter of convenience, we employ the constraints of the 3L method[2)][⋆1]. ii) All the data sets are regularized by centering the data-set center of mass at the origin of the coordinate system and scaling it by dividing each point by the average length from point to origin, as done in[28)]; iii) We choose T_1 and T_2 in Eq. (6) with about 0.01 and 0.95 respectively, excepted for the experiment in Section 4.5. iv) The RR parameter κ in Eq. (17) is empirically set to increase the original diagonal element r_{ii} about 10%. Note: we also refer the interested reader to the discussion on setting κ in[28)].

Some 2-D and 3-D experiments are shown in Fig. 8. The result shows our method's adaptivity for different complexities of shapes.

4.5.1 Degree-fixed Fitting vs. Adaptive Fitting

Fig. 9 shows some comparisons between the degree-fixed fitting methods and our adaptive fitting method. Compared with degree-fixed methods, in the results of our method, there is neither over-fitting nor insufficient fitting. This shows that our method is more meaningful than the degree-fixed methods, since it fulfills the requirement that the degrees should be subject to the complexities of object shapes. To clarify again, as described above, our method saves much computational time despite its determination of the moderate degree by an incremental process.

⋆1 The layer distance of the 3L method in[2)] is set as 0.05.

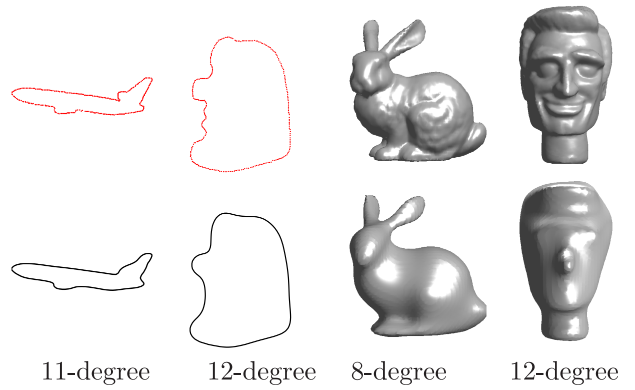

Fig. 8 Adaptive IP fitting results in 2D/3D. First row: original objects; Second row: IP fits.

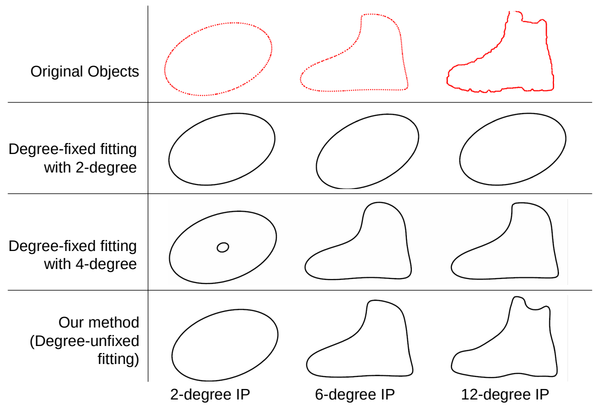
Original Objects
Degree-fixed fitting with 2-degree
Degree-fixed fitting with 4-degree
Our method (Degree-unfixed fitting)
2-degree IP
6-degree IP
12-degree IP

Fig. 9 Comparison between degree-fixed fitting and adaptive fitting. First row: original objects. Second row: IP fits resulting from two-degree degree-fixed fitting. Third row: IP fits resulting from four-degree degree-fixed fitting. Fourth row: IP fits in different degrees resulting from adaptive fitting setting parameters as $T_1 = 0.01$ and $T_2 = 0.95$.

4.5.2 Comparison of Fitting Stability

Fig. 10 shows a comparison between three methods: i) 3L method[2)], ii) 3L

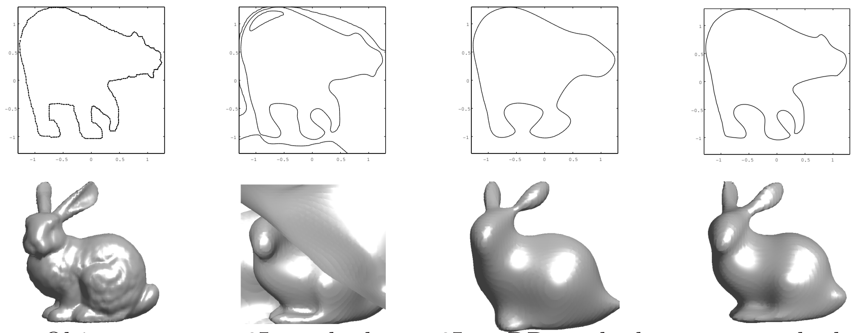

Fig. 10 Comparison of fitting results: first column: original 2D/3D objects, second column: results obtained by 3L method[2], third column: results obtained by 3L method + RR method[22),28] (for 2D and 3D respectively), last column: results obtained by our method. Results in top row are 12th degree and in bottom row are 8th degree.

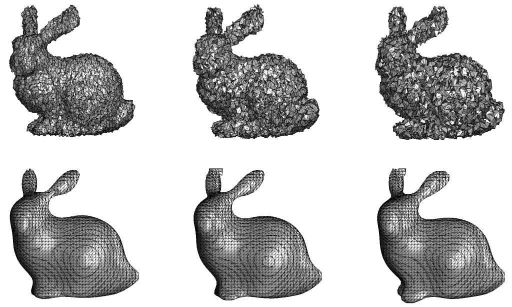

Fig. 11 Top row: Noisy data sets made by adding Gaussian noise to the original model with standard deviations: from left to right 0.1, 0.15, 0.2; Bottom row: the corresponding fitting results.

method[2] + 2D RR regularization[28] and 3L method[2] + 3D RR regularization[22] and iii) our method. As a result, our method shows better performance than the others, in both global stability and local accuracy.

4.5.3 An Example for Noisy Data Fitting

We generated some synthetic noisy data sets shown in the top row of Fig. 11. For overcoming the effect on the variation of noise, we adopted the second stopping criterion (21) where parameters with respect to residuals are set as $T_1 = 0.001$ and $T_2 = 0.01$.[⋆1] Then we obtained the fitting results as shown in the bottom row of Fig. 11.

5. A Registration Method using IP Gradient Field

The task of rigid shape registration aims to build a transformation relationship between a source object and a target object. In this section, we present a fast registration method by using IP gradient field which supposes that the target object has been modeled by an IP beforehand.

Given the IP model representing the target object, first let us consider how to solve the registration problem between the IP and the source object of discrete model. To achieve that, we first define the registration problem as an energy minimization problem, and then by minimizing this energy function the motion of source object can be driven to the target model (IP model).

5.1 Energy functional

The objective of registration is to find a transformation, through which the zero set $\{\mathbf{x}|f_n(\mathbf{x}) = 0\}$ can "best" register to the discrete points (source object). To achieve this, we first define an energy functional E which will be minimized to find the proper transformation, described as follow:

$$\mathbf{p} = \arg\min_{\mathbf{p}} E(\mathbf{p}). \tag{22}$$

where $\mathbf{p}$ is the transformation parameters ($\mathbf{p} \in \mathcal{R}^6$ for rigid transformation). In general, the energy functional E evaluates the registration by minimizing the distance between the data set and IP, defined as:

$$E = \sum_i dist(T(\mathbf{p}, \mathbf{x}_i), f_n), \forall \mathbf{x}_i \in \Omega, \tag{23}$$

where $T(\mathbf{p}, \mathbf{x}_i)$ is a function: $\mathcal{R}^3 \to \mathcal{R}^3$, that returns the transformed point of $\mathbf{x}_i$ by the rigid-transform operation respected to parameter $\mathbf{p}$; $dist(\mathbf{x}, f_n)$ means a certain distance from the data point $\mathbf{x}$ to the zero set of f_n; and Ω represents the 3D region of the source model.

⋆1 Since the normals often vary more strongly than vertices, we set the smoothness threshold T_2 to be more tolerant than the distance threshold T_1

There are multiple choices for the distance $dist(\mathbf{x}, f_n)$ in (23), and the common one is to use $\mathcal{L}_2$ norm. Thus it becomes possible to form least-squares regression with $dist(\mathbf{x}, f_n) = f_n(\mathbf{x})^2$. However, in this work, we choose more meaningful approximation for the distance representation of IP as (see 29)):

$$dist(\mathbf{x}, f_n) = \frac{f_n(\mathbf{x})^2}{\parallel \bigtriangledown f_n(\mathbf{x}) \parallel^2}, \forall \mathbf{x} \in \Omega, \tag{24}$$

where $\bigtriangledown$ denotes the gradient of IP function.

5.2 Minimizing energy functional

To minimize the energy functional (23), we employ the following two steps: for accelerating the convergence, first, it minimizes the function without any constraint in the transformation. This means every point can move freely towards IP along their gradients during the first minimization. Next, it determines the transformation parameters to maintain the Euclidean transformation. These two steps are repeated alternately until convergence. The efficiency of this minimization benefits from that it can successfully avoid the time-consuming computation for finding the point-wise correspondence.

5.2.1 First step: free deformation

At the first step, by calculus of variations[3], the Gateaux derivative (first variation) of the functional E in (23) to point $\mathbf{x}$ can be approximately formulated as

$$\frac{\partial E}{\partial \mathbf{x}} = \frac{\partial dist(\mathbf{x}, f)}{\partial \mathbf{x}} \approx 2f(\mathbf{x})\frac{\bigtriangledown f_n(\mathbf{x})}{\parallel \bigtriangledown f_n(\mathbf{x}) \parallel^2} = 2G(\mathbf{x}), \tag{25}$$

if we assume $\parallel \bigtriangledown f_n(\mathbf{x}) \parallel$ is a constant and let $G(\mathbf{x}) = f(\mathbf{x})\frac{\bigtriangledown f_n(\mathbf{x})}{\parallel \bigtriangledown f_n(\mathbf{x}) \parallel^2}$.

Therefore, we need to minimize this functional to satisfy the Euler-Lagrange equation $\frac{\partial E}{\partial \mathbf{X}} = 0$. Thus the steepest descent process is executed in the following gradient flow for each point $\mathbf{x}$:

$$\frac{\partial \mathbf{x}}{\partial t} \approx -2G(\mathbf{x}), \tag{26}$$

where t denotes the time step. From the view of implementation, each point can be updated as:

$$\mathbf{x}^{k+1} = \mathbf{x}^k - 2G(\mathbf{x}^k), \tag{27}$$

where $\mathbf{x}^k$ denotes the point at k-th time step.

Similarly, if let $X \in \mathcal{R}^{N\times 3}$ represent the integral data set consisting of each point $\mathbf{x}$ at each row vector, and $\widetilde{G} \in \mathcal{R}^{N\times 3}$ be the integral gradients for integral data set X, then X^k at the k-th time step can be updated as the following recurrence:

$$X^{k+1} = X^k - 2\widetilde{G}(X). \tag{28}$$

This shows the fact that, under this operation, every points in the discrete data set of source object will be moved towards the zero set of polynomial along their gradients.

5.2.2 Second step: rigid deformation

In the first step, although every point in source model can move closed to IP surface, the transformation dose not maintain the rigid transformation form, but rigid transformation is necessary for the estimation of (22).

Therefore, in the second step, we fix the first step to be rigid transformation. For achieving this, the integral evolution of the data set is need to be analyzed first. Since from the first step's transformation in Eq. (28) we can obtain data sets represented by two matrices X^k and X^{k+1} in two continuous steps. Then, the covariant matrix of these two matrices can be calculated as:

$$A = (X^k - \bar{X}^k)^{\mathrm{T}}(X^{k+1} - \bar{X}^{k+1}), \tag{29}$$

where $\bar{X}$ is the matrix each row consisting of the mean value (center point) of X. If A $(\in \mathcal{R}^{3\times 3})$ is decomposed as $A = USV^{\mathrm{T}}$ by using the singular value decomposition (SVD), where S is the diagonal matrix and U and V are the unitary matrices, then the rigid transformation is obtained by:

$$\begin{aligned} R &= UV^{\mathrm{T}}, \\ \mathbf{t} &= \bar{X}^k - \bar{X}^{k+1}R^{\mathrm{T}}, \end{aligned} \tag{30}$$

where R and $\mathbf{t}$ are rotation and translation parameters respectively[32]. Thus, with R and $\mathbf{t}$, we can fix the transformation in first step shown in Eq. (28) to be rigid transformation as:

$$X^{k+1} = X^k R^{\mathrm{T}} + \mathbf{t}. \tag{31}$$

Therefore, the above two steps can be alternately repeated until the discrete data set of source model is moved near to IP with desired accuracy.

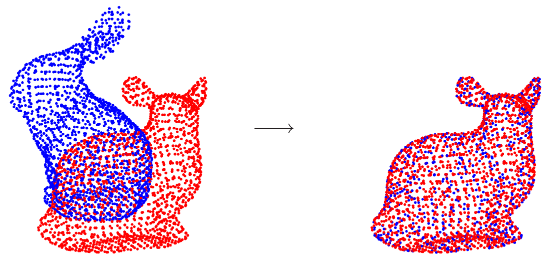

Fig. 12 Illustration for registration. Blue points for target object and red points for source object

5.3 An acceleration method

Let us present an acceleration method which makes the registration more efficient in the case that the number of data points in the discrete model is much larger than the number of coefficients of the IP. Since in this case moving the large-scale data sets to IP model is relatively more expensive than moving inversely the IP towards the discrete model.

To drive this motion of IP, it requires a transformation method for IP through which IP's coefficients are transformed to satisfy that IP model can maintain a rigid transformation. Therefore the acceleration method can described as that, instead of updating discrete data set in Eq. (31), we update the coefficients of IP at each step:

$$\mathbf{a}^{k+1} = V(R, \mathbf{t})\mathbf{a}^k, \tag{32}$$

where we suppose that $\mathbf{a}^k$ is the coefficient vector at the k-th time step, and V is a square transformation matrix corresponding to the Euclidean transformation of rotation R and translation $\mathbf{t}$. The remained problem is how to construct the transformation matrix V in (32). For solving V, we refer the interested reader to[36].

5.4 Experimental Results

For comparing our method to the ICP-based methods, we solved a registration problem shown in Fig. 12, where the required initial settings are: i) both the target and source model were extracted from same object; ii) the target model was rotated and translated to the position as the blue points shown in Fig. 12; iii) the registration was tested by different methods, and all the methods will stopped until satisfy the same accuracy and computational time and memory were used to evaluate the performance.

Table 1 Computational cost comparison for registering a 10k-points data set.

	ICP	ICP with KD tree	IP
CPU Time (sec)	123.01	35.12	1.07
Memory	20K points	20K points + KD tree	10K points + 165 coef.

We tested three methods: standard ICP method[1], ICP method with KD tree[35] and our method. Since they are set with the same stopping criterion (same threshold for registration accuracy), we only compared the computational time and memory in Tab. 1 which show the registration task of 10k points both for target and source data sets. Since the implementations were done in Matlab, it was necessary to take care to eliminate the effect of Matlab being an interpreted language.

5.5 Application to Pose Estimation for US Image

Ultrasound (US) imaging is widely used for assistance with surgical operations and real-time diagnosis. The relative pose estimation for US image to another modality, such as MRI or CT, is desired and will be helpful to diagnosis guidance. But it suffers the difficulties due to poor image quality with speckle noises, low signal-to-noise ratio, and uniform brightness, and only cross-sectional images obtained. Fortunately, considering the characteristics of our method described above, we confront this registration problem by making use of IP.

5.5.1 Pose Estimation for Duck Toy

Fig. 14 shows the result of the US image pose estimation that the images are obtained by scanning a duck toy made of rubber in the cistern shown in Fig. 13 left, and we modeled the duck object with an 8-degree IP shown in Fig. 13 right. Fig. 14 shows the relative position and the cross-section contour at each iteration resulting from the registration. In this case the total consumed CPU time is about 180 ms excluding the rendering time.

5.5.2 Pose Estimation and Tracking for CT data

The third example is tested for registration between a real CT data and US image. To do this, we first segmented the CT data to obtain the desired organ

Vol.2009-CVIM-167 No.4
2009/6/9

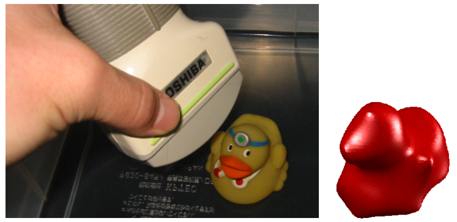

Fig. 13 Left: photograph of measuring a duck toy. Right: IP model of duck object

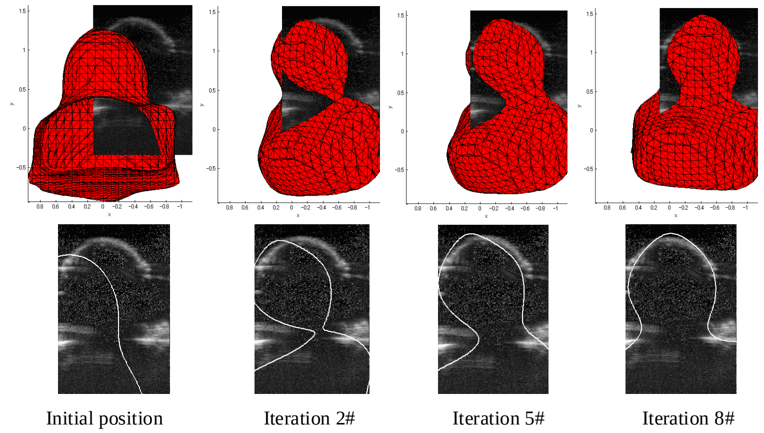

Fig. 14 Pose estimation for duck toy. First row: relative pose of IP and US image at each iteration. Second row: cross-section contour at each iteration.

object by modern segmentation methods such as Graph Cut (see Fig. 15). Then we model the organ object by a 4-degree IP as shown in Fig. 15 right. Therefore for US images to find the relative pose to CT data becomes to find the relative pose to the IP.

The first example is pose estimation for a single US image shown in Fig. 16 where we show the registration process with relative pose and cross-section contour at each selected iteration.

We also experimented the US image sequence by tracking the position of the kidney object with the above IP model. Results are shown in Fig. 17 where we show the cross-section contours of IP for selected frames. The only difference to single frame registration is that we set the initial position of each frame with the resulted position of the previous frame.

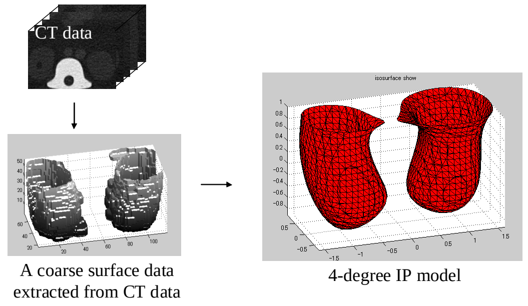

Fig. 15 Segmenting kidney data from the CT scans and modeling with a 4-degree IP.

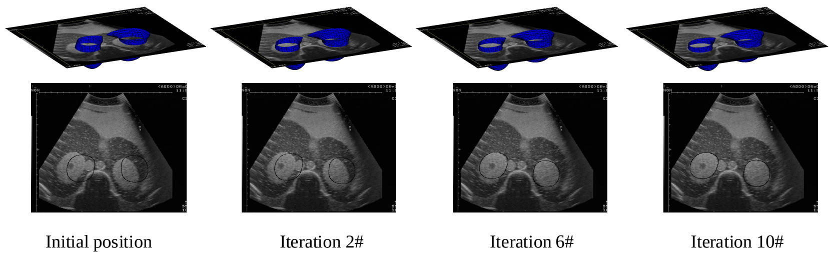

Fig. 16 US pose estimation for CT data with 4-degree IP.

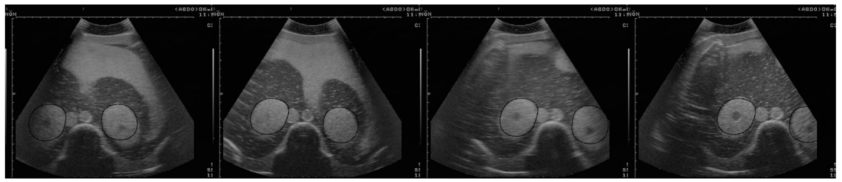

Fig. 17 Tracking for US image sequence. Cross-section contours of frames are shown in black points.

6. Conclusions

We have explored the problems on IP representation, fitting and registration, and provided the three solutions: i) a segmentation method for achieving IP-segment representation. ii) a globally/locally stable fitting method that can adaptively determine moderate degrees for fitting. iii) a fast registration method using IP gradient field that supports partially overlapping object registration.

With the first method, IPs can be assigned to surface segments not only for providing good representation, but also for assigning the geometric/algebraic properties to the surface, such as the curvatures and Euclidean invariants encoded by IP. The results have the potentials for opening up the new vista for 3D object recognition and registration.

The second method solves the fitting problems on how to determine the moderate degree and brings numerical stability. First, its computational efficiency benefits from keeping incrementally solving upper-triangular linear systems by taking advantage of QR decomposition. Second, its numerical stability benefits from the easy and quick instability detection by checking the diagonal elements of upper triangular matrix and modifying selectively the unstable elements.

The third method has high computational efficiency, because i) the avoidance of point-wise correspondence by using IP gradient flows; ii) a fast two-step minimization adopted; iii) acceleration by IP transformation to large-scale problem.

As the result, this research provided some new insights into the theory and practice for IP representation. It extends IP's applicability and makes it have potentials for wider applications in computer vision.

References

1) Besl, P. and McKay, N.: A Method for Registration of 3-D Shapes, *IEEE Trans. on Patt. Anal. Mach. Intell.*, Vol.14, No.2, pp.239–256 (1992).
2) Blane, M., Lei, Z.B. and Cooper, D.B.: The 3L Algorithm for Fitting Implicit Polynomial Curves and Surfaces to Data, *IEEE Trans. on Patt. Anal. Mach. Intell.*, Vol.22, No.3, pp.298–313 (2000).
3) Evans, L.: *Partial Differential Equations*, Providence: American Mathematical Society (1998).
4) Forsyth, D., Mundy, J., Zisserman, A., Coelho, C., Heller, A. and Rothwell, C.: Invariant Descriptors for 3D Object Recognition and Pose, *IEEE Trans. on Patt. Anal. Mach. Intell.*, Vol.13, No.10, pp.971–992 (1991).
5) Golub, G. and O'Leary, D.P.: Tikhonov Regularization and Total Least Squares, *SIAM J. Matrix Anal. Appl.*, Vol.21, pp.185–194 (2000).
6) Goshtasby, A.: Design and Recovery of 2-D and 3-D Shapes Using Rational Gaussian Curves and Surfaces, *Int. J. Comp. Visi.*, Vol.10, No.3, pp.233–256 (1993).
7) Helzer, A., Bar-Zohar, M. and Malah, D.: Using Implicit Polynomials for Image Compression, *Proc. 21st IEEE Convention of the Electrical and Electronic Eng*, pp. 384–388 (2000).
8) Helzer, A., Barzohar, M. and Malah, D.: Stable Fitting of 2D Curves and 3D Surfaces by Implicit Polynomials, *IEEE Trans. on Patt. Anal. Mach. Intell.*, Vol.26, No.10, pp.1283–1294 (2004).
9) Ikeuchi, K., Hasegawa, K., Nakazawa, A., Takamatsu, J., Oishi, T. and Masuda, T.: Bayon Digital Archival Project, *Proceedings of the Tenth International Conference on Virtual Systems and Multimedia*, pp.334–343 (2004).
10) Iwashita, Y., Kurazume, R., Konishi, K., Nakamoto, M., Aburaya, N., Sato, Y., Hashizume, M. and Hasegawa, T.: Fast Model-Image Registration using 2D Distance Map for Surgical Navigation System, *Advanced Robotics*, Vol.21, No.7, pp. 751–770 (2007).
11) Kang, K., Tarel, J.-P., F., R. and Cooper, D.: A Linear Dual-Space Approach to 3D Surface Reconstruction from Occluding Contours using Algebraic Surfaces, *Proc. IEEE Conf. Int. Conf. on Comp. Visi.*, Vol.1, pp.198–204 (2001).
12) Keren, D.: Using symbolic computation to find algebraic invariants, *IEEE Trans. on Patt. Anal. Mach. Intell.*, Vol.16, No.11, pp.1143–1149 (1994).
13) Keren, D., Cooper, D. and Subrahmonia, J.: Describing complicated objects by implicit polynomials, *IEEE Trans. on Patt. Anal. Mach. Intell.*, Vol.16, No.1, pp. 38–53 (1994).
14) Kindlmann, G., Whitaker, R., Tasdizen, T. and Moller, T.: Curvature-based transfer functions for direct volume rendering: methods and applications, *Visualization, 2003. VIS 2003. IEEE*, pp.513– 520 (2003).
15) Marola, G.: A Technique for Finding the Symmetry Axes of Implicit Polynomial Curves under Perspective Projection, *IEEE Trans. on Patt. Anal. Mach. Intell.*, Vol.27, No.3 (2005).
16) Munim, H.E. and Farag, A.: Shape Representation and Registration using Vector Distance Functions, *Proc. IEEE Conf. Comp. Visi. and Patt. Rec.*, pp.Y1–Y8 (2007).
17) Oden, C., Ercil, A. and Buke, B.: Combining implicit polynomials and geometric features for hand recognition, *Pattern Recognition Letters*, Vol.24, No.13, pp.2145–2152 (2003).
18) Oishi, T., Nakazawa, A., Kurazume, R. and Ikeuchi, K.: Fast Simultaneous Alignment of Multiple Range Images using Index Images, *Proc. The 5th International Conference on 3-D Digital Imaging and Modeling (3DIM 2005)*, pp.476–483 (2005).
19) Petitjean, S.: A Survey of Methods for Recovering Quadrics in Triangle Meshes, *ACM Computing Surveys*, Vol.34, No.2, pp.211–262 (2002).
20) Piegl, L.: On NURBS: a survey, *Computer Graphics and Applications*, Vol.11,

No.1, pp.55–71 (1991).

21) Redding, N. J.: Implicit Polynomials, Orthogonal Distance Regression, and the Closest Point on a Curve, *IEEE Trans. on Patt. Anal. Mach. Intell.*, Vol.22, No.2, pp.191–199 (2000).

22) Sahin, T. and Unel, M.: Fitting Globally Stabilized Algebraic Surfaces to Range Data, *Proc. IEEE Conf. Int. Conf. on Comp. Visi.*, Vol.2, pp.1083–1088 (2005).

23) Salvi, J., Matabosch, C., Fofi, D. and Forest, J.: A review of recent range image registration methods with accuracy evaluation, *Image and Vision Computing*, Vol. 25, No.5, pp.578–596 (2007).

24) Subrahmonia, J., Cooper, D. and Keren, D.: Practical reliable bayesian recognition of 2D and 3D objects using implicit polynomials and algebraic invariants, *IEEE Trans. on Patt. Anal. Mach. Intell.*, Vol.18, No.5, pp.505–519 (1996).

25) Tarel, J. and Cooper, D.: The Complex Representation of Algebraic Curves and Its Simple Exploitation for Pose Estimation and Invariant Recognition, *IEEE Trans. on Patt. Anal. Mach. Intell.*, Vol.22, No.7, pp.663–674 (2000).

26) Tarel, J.-P., Civi, H. and Cooper, D.B.: Pose Estimation of Free-Form 3D Objects without Point Matching using Algebraic Surface Models, *In Proceedings of IEEE Workshop Model Based 3D Image Analysis*, pp.13–21 (1998).

27) Tasdizen, T. and Cooper, D.: Boundary Estimation from Intensity/Color Images with Algebraic Curve Models, *Int. Conf. Pattern Recognition*, pp.1225–1228 (2000).

28) Tasdizen, T., Tarel, J.-P. and Cooper, D.B.: Improving the Stability of Algebraic Curves for Applications, *IEEE Trans. on Imag. Proc.*, Vol. 9, No. 3, pp. 405–416 (2000).

29) Taubin, G.: Estimation of Planar Curves, Surfaces and Nonplanar Space Curves Defined by Implicit Equations with Applications to Edge and Range Image Segmentation, *IEEE Trans. on Patt. Anal. Mach. Intell.*, Vol.13, No.11, pp.1115–1138 (1991).

30) Taubin, G. and Cooper, D.: *Symbolic and Numerical Computation for Artificial Intelligence, chapter 6*, Computational Mathematics and Applications. Academic Press (1992).

31) Turk, G. and OBrieni, J.F.: Variational Implicit Surfaces, *Technical Report GIT-GVU-99-15, Graph., Visual., and Useab. Cen.* (1999).

32) Umeyama, S.: Least-Squares Estimation of Transformation Parameters Between Two Point Patterns, *IEEE Trans. on Patt. Anal. Mach. Intell.*, Vol.13, No.4, pp. 376–380 (1991).

33) Unel, M. and A.Wolovich, W.: On the Construction of Complete Sets of Geometric Invariants for Algebraic Curves, *Advances In Applied Mathematics*, Vol.24, pp.65–87 (2000).

34) Wolovich, W.A. and Unel, M.: The Determination of Implicit Polynomial Canonical Curves, *IEEE Trans. on Patt. Anal. Mach. Intell.*, Vol.20, No.10, pp.1080–1090 (1998).

35) Zhang, Z.: Iterative point matching for registration of free-form curves and surfaces, *Int. J. Comp. Visi.*, Vol.13, No.2, pp.119–152 (1992).

36) Zheng, B.: 2D Curve and 3D Surface Representation of using IP and Its Applications, *Ph.D Thesis in Information Science and Technology, The University of Tokyo* (2008).