2D CURVE AND 3D SURFACE REPRESENTATION USING IMPLICIT POLYNOMIAL AND ITS APPLICATIONS

陰関数を用いた２Ｄ曲線と３Ｄ曲面の多項式表現とその応用

BY

BO ZHENG

A DOCTORAL DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL OF
THE UNIVERSITY OF TOKYO

東京大学
THE UNIVERSITY OF TOKYO

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF INFORMATION SCIENCE AND TECHNOLOGY

JUNE 2008

Committee:

Hiroshi HARASHIMA
Mitsuru ISHIZUKA
Kiyoharu AIZAWA
Yoichi SATO
Toshihiko YAMASAKI

Supervisor:

Katsushi IKEUCHI

ABSTRACT

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 Fourier, B-Splines, Non-Uniform Rational B-Splines (NURBS), Radial Basis Function (RBF), 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.

Despite these excellent characteristics, still IP mainly suffers from the following two issues that not only greatly limit its applicability, but also confuse the users who need to model their target objects with IPs:

- Poor representation for complex objects

  Objects obtained by vision modalities are often very precise and thus complex. Unfortunately, generating IP even with high degrees is nearly impossible to give fine shape representation for these objects due to numerical instability and high computational cost.

- Difficulty in determining a moderate IP degree and achieving global/local stability

  Prior IP fitting methods require that the degree of IP must be determined before handling fitting and therefore they are difficult to determine a moderate degree for a complex object. Furthermore the prior methods suffer from computational 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.

This thesis focuses on IP representation and makes three major contributions on it. The first and second contributions are the developments of methods for addressing the two issues as described above, poor representation for complex object and difficulty in determining a moderate IP degree. The third contribution is a development for a registration method of using IP gradient field.

In order to address the first issue, poor representation for complex object, we propose a 3D segmentation method based on a cut-and-merge approach. Through the method, a complex surface can be divided into segments and each segment is encoded by a low-degree IP. The advantages are that it is capable of 1) finding the appropriate segmentation for various desired accuracies; 2) achieving stable fitting since we avoid using high-degree IPs; and 3) decomposing heavy computation from one high-degree IP fitting into light computations from multiple low-degree IPs. This method maintains inherent algebraic/geometric properties onto the IP-segmented surface. The result also opens up a new vista for the application of non-iterative registration for range image which enable the registration to be fast and independent of position initialization.

For the second issue, difficulty in determining a moderate IP degree and achieving global/local stability, we propose a novel method which has many capabilities over the prior methods. The majors are 1) adaptively determining the moderate degree for fitting that depends on the complexity of objects, in an incremental manner without greatly increasing the computational cost; 2) maintaining global stability while also avoiding excessive loss of local accuracy. Through applying this method into the segmentation method in first contribution, we obtain an adaptive segmentation method that is able to adaptively assign a moderate IP degree to each of the segments.

In addition, since the stable IP representation is guaranteed by the stable fitting method in our second contribution, we exploit a new object registration technique of using IP gradient field. Over the conventional registration method a significant advantage is the computational efficiency, which benefited from 1) the registration is formulated as to minimize an energy functional derived from IP gradient flow, and avoid the cost for calculating point-wise correspondences; 2) with the property of IP having a few coefficients, it makes both IP gradients and its transformation can be calculated in an extremely light manner both on computational complexity and memory; 3) a fast transformation method for IP coefficients is proposed. Furthermore the effectiveness is proved by a new application of a 6-DOF pose estimation method for single Ultrasound image, since the computational efficiency of this method guarantees a real-time achievement in the application.

The research provides some new insights and applicability into the theory and practice of IP representation, and the main contribution can be summarized by the three following points: First, a 3D IP-segment representation method is developed even for robustly representing complex objects. Second, an efficient fitting method has been developed for adaptively estimating the IP of moderate degree for objects with various shapes. Third, a computational efficient and robust object registration method using IP gradient field has been developed.

Acknowledgements

First and foremost, I would like to express my gratitude to my supervisor, Prof. Katsushi Ikeuchi. He gave me a wonderful opportunity to come to the University of Tokyo, and take me into an interesting field, computer vision. Not only has he direct me to pursue my Ph.D., he has offered me an open door and an open mind, encouraging me and providing me with countless ideas through discussions. What most worth appreciating is the time when he spent hours weekly with me discussing my research face to face. I can never forget the effort, many times, he made to revise my paper throughout the night.

I would like to record my gratitude to Prof. Jun Takamatsu for his advice and guidance from the very early stage of this research as well as giving me extraordinary experiences through out the work. Above all and the most needed, he provided me unflinching encouragement and support in various ways. His truly scientist intuition has made him as a constant oasis of ideas and passions in research, which exceptionally inspire and enrich my growth as a student, a researcher and a scientist want to be. I am indebted to him more than he knows.

Many thanks go to the people that I have been working with, the current and former members of the Computer Vision Laboratory at the University of Tokyo. Special thanks go to Dr. Takeshi Oishi, who gave me a many ideas and held important discussions with me throughout my work, to Dr. Daisuke Miyazaki, Dr. Atsuhiko Banno, Dr. Shunsuke Kudoh, Dr. Shintaro Ono, and Dr. Rei Kawakami who instructed me in the earlier years of my study.

I wish to express my deepest gratitude to Canon Inc, since the partial work in this thesis is supported by Canon Inc. Special thanks go to Dr. Ryo Ishikawa in Canon for helping me with experiments and writing a paper, to Dr. Kiyohide Satoh in Canon for sharing their survey results in our seminar.

Many thanks go to the people bring me the knowledges in different research fields. Special thanks to Prof. Shao-Liang Zhang, Nagoya University, for advising me in numerical analysis; to Prof. Hong-En Liao, the University of Tokyo, for providing technical information and sharing their experimental devices on medical image processing.

I would like to thank Dr. Joan Knapp for spending considerable time proofreading my English manuscripts. She kindly improved my writing and gave me many appropriate suggestions.

I would also like to thank my committee members, Prof. Hiroshi Harashima, Prof. Mitsuru Ishizuka, Prof. Kiyoharu Aizawa, Prof. Yoichi Sato, and Prof. Toshihiko Yamasaki, for giving valuable advice on this thesis.

I would like to thank many people in my life. In particular, I am very grateful to all the members of the Chinese Badminton Club in the University of Tokyo, to which I have belonged for 6 years, for their contributions to the enjoyment of my student life. Many thoughts derived from badminton excises and matches have always encouraged me and been used for reference in my study.

Last but not least, I would like to thank my family for always being patient and supporting me during my student life. It has been a long effort but worth doing, and I could not finish it without all of your support. I dedicate this dissertation to my family.

July 2008

Contents

List of Figures

List of Tables

List of Symbols

$A, \cdots, Z$	matrices
$\mathbf{a}, \cdots, \mathbf{z}$	vectors
$\alpha, \beta, \cdots, \omega$	scalars
A^{T}	transpose of A
A^{-1}	matrix inverse
$A^{-\mathrm{T}}$	the inverse of A^{T}
$a_{i,j}$	matrix element
a_i	vector element
u_x, u_{xx}	first, second derivative with respect to x
$\mathbf{x} \cdot \mathbf{y}$	vector dot product (inner product), $\mathbf{x}^{\mathrm{T}}\mathbf{y}$
$\mathrm{diag}(\alpha, \beta, \cdots)$	diagonal matrix constructed from scalars $\alpha, \beta, \cdots$
$\mathrm{span}\{\mathbf{a}, \mathbf{b}, \cdots\}$	spanning space of vector $\mathbf{a}, \mathbf{b}, \cdots$
$\mathcal{R}$	set of real numbers
$\mathcal{R}^n$	real n space
$\parallel \mathbf{x} \parallel$	2-norm of vector $\mathbf{x}$
$f_n^{2D}(\mathbf{x})$	polynomial of degree n with 2D variable $\mathbf{x} = (x, y)^{\mathrm{T}}$
$f_n^{3D}(\mathbf{x})$	polynomial of degree n with 3D variable $\mathbf{x} = (x, y, z)^{\mathrm{T}}$
a_{ijk}	coefficient of monomial $x^i y^j z^k$
$\mathbf{a}$	coefficient vector of polynomial of degree n
$\mathbf{m}_n(\mathbf{x})$	monomial vector of polynomial of degree n, namely $(1\ x\ y\ \ x^2\ \ldots\ y^n)^{\mathrm{T}}$
M	monomial matrix which ith row vector are $\mathbf{m}_n(\mathbf{x}_i)$
M_{3L}	monomial matrix of 3L method
M_{G1}	monomial matrix of Gradient-one method
M_{max}	monomial matrix of Min-Max method
$M^{\mathrm{T}}M$	covariant matrix of monomial matrix M
$\mathbf{b}$	right-hand vector of linear system derived from least square method
$\nabla f(\mathbf{x})$	gradient vector of polynomial f
F_{ijk}	$\frac{F_{ijk}}{i!j!k!}$ is the coefficient corresponding $x^i y^j z^k$

$\boldsymbol{\alpha}$	multi-index of monomial, $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n)$
$\lvert\boldsymbol{\alpha}\rvert$	$\lvert\boldsymbol{\alpha}\rvert = \alpha_1!\alpha_2!\ldots\alpha_n!$
ϕ_r	the rth form of polynomial
$\Phi_{(i,j,k)}$	$\frac{\Phi_{ijk}}{i!j!k!}$ is the coefficient corresponding $x^i y^j z^k$
$\Phi_{[r]}$	the coefficient vector of rth form of polynomial with component $\Phi_{(i,j,k)}$
$\star$	matrix multiplication by symbolic compuation
$\Phi_{[j,k]}$	the covariant matrix, $\Phi_{[j]} \star \Phi_{[k]}$
D_{dist}	Distance metric on Euclidean distance
D_{smooth}	Distance metric on smoothness
$\nabla\mathbf{n}_i^{\mathrm{T}}$	hessian matrix of normal vector $\mathbf{n}_i$
κ_{max}, κ_{min}	maximum/minimum absolute curvature
$dist(\mathbf{x}, f)$	distance from $\mathbf{x}$ to zero set of polynomial f
G_σ	Gaussian filter with standard deviation σ

Chapter 1

Introduction

1.1 Motivation

Shapes are fundamental attributes of visible objects, and reasoning about the shape is a common way of describing and representing real objects in computer science, engineering, mathematics, biology, physics, chemistry, and even in human daily life. Research on shape information is hence multidisciplinary.

Shape representations of a target object have been exploited for a long history for purposes in various fields, such as computer graphics, computer visualization and computer vision. In computer graphics, shape representations require complete models to accurately render a realistic synthetic image of the objects described. For rendering purpose, the key attributes relate to realism (visual fidelity). In computer visualization, the shape representations focus on a visual form enabling the user to observe the information. It is more than pretty images, for human people, successful visualization can reduce the time it takes to understand the underlying data, to find relationships, and to get the information searched.

In computer vision, in contrast, the shape representation often provides feather information for visual objects which is designed for use in the specific vision applications which have been attractive for the wide and various purposes such as industry, medical, culture, traffic system and astronavigation. For example, shape information is extracted for the purpose of supporting a manufacturing process, such as the measurement of position and orientation of targets to be picked up by a robot. Or in medical computer vision, the shape is characterized by information from medical image data for the purpose of making a medical diagnosis of a patient. Furthermore, in the newer application areas, autonomous vehicles include submersibles, land-based vehicles, aerial vehicles, and unmanned aerial vehicles (UAV) capture the shape information of references to localize the position of themselves. Therefore, for vision, shapes are viewed as a feature information which can discriminate the target objects from each other, and thus the shape representation biases to the performance, such as accuracy or computational efficiency, on the discriminability. On the other hand, the visual fidelity may not be a criterion of interest.

This thesis studies the shape representation motivated by solving the tasks of computer vision. Generally the vision application areas described above employ a range of computer vision tasks, such as shape recognition, registration, pose estimation *etc.*, which are often an intermediate but crucial steps inside the computer vision systems. Thus we attempt to provide feature information of shapes for visual objects that can be adopted in accomplishing these tasks efficiently. Therefore the ultimate purpose of this thesis is to exploit or construct a powerful shape representation for computer vision which can bring the higher performance for specific vision applications than the priors.

In the following section, first let us give the overview of modern famous techniques on shape representations, and then we will give a comparison between them.

1.2 Literature Overview of Shape Representations

Without loss of generality, let us discuss the shape representation based on the boundaries of 3D objects which are often shown as the boundaries (surfaces) of 3D objects. 3D surfaces can be represented mathematically in different forms. The most common ones are given in

- Explicit form: $G =< \mathbf{v}, \mathbf{f} >$, where $\mathbf{v}$ explicitly records vertices on surface, and $\mathbf{f}$ is set of polygons consisting of indices of vertices.
- Parametric form: $\mathbf{P} = [x(u, v), y(u, v), z(u, v)]$
- Implicit form: $f(x, y, z) = 0$

Tab. 1.1 gives the simple examples of representation for a sphere surface in three different forms respectively.

1.2.1 Explicit Shape Representation

In an explicit representation one explicitly writes down the points that belong to the surface. A popular explicit representation is polygon mesh which is a collection of vertices, edges and faces that defines the shape of a polyhedral object. The faces usually consist of triangles, quadrilaterals or other simple convex polygons. For example, an object can be defined by a triangular mesh as many triangular surfaces with a shared vertex-list as follow:

$$G =< \mathbf{v}, \mathbf{f} >, \tag{1.1}$$

where $\mathbf{v} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N\}$ is a list of vertices $\mathbf{v}_i = (x_i, y_i, z_i)^{\mathrm{T}}$, and $\mathbf{f} = \{\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_M\}$ is a list of triangular face, each specified as a list of vertex indices: $\mathbf{f}_i = \{< i_1, i_2, i_3 > | i_j = 1, 2, \ldots, N\}$.

Table 1.1: The explicit, parametric and implicit representations for the boundary of a sphere of radius r centered at the origin.

Name	Representation
Explicit	$G = <\mathbf{v}, \mathbf{f}>$ where $\mathbf{v}$: a set of vertices on the sphere $\mathbf{f}$: a list of polygons specified by vertex indices
Parametric	$\mathbf{P} = [x(u,v), y(u,v), z(u,v)]$ where $u \in [-r, r]$ $v \in [0, 2\pi)$ $x(u,v) = \sqrt{r^2 - u^2}\cos(v)$ $y(u,v) = \sqrt{r^2 - u^2}\sin(v)$ $z(u,v) = u$
Implicit	$f(x,y,z) = x^2 + y^2 + z^2 - r^2 = 0$

Explicit surfaces are extensively used in computer graphics, *e.g.*, graphics designers and animators have tended to prefer polygon meshes because with them it is relatively easy to simplify the rendering or control shapes in an intuitively pleasing way. However, they are difficult for fitting and automated modeling purposes [IF03]. Also, it is difficult to confirm if an arbitrary point is on, inside or outside an explicit surface. Since it describes only explicitly where the available data points are, interpolation and computation of surface gradients is problematic. Another disadvantage of explicit representations is that tracking of large deformations and topological changes of the surface is quite difficult [ZOF01]. For instance, since connectivity of the explicit points determines the triangulation of a 3D surface, changes in connectivity due to surface deformations can pose problems for triangulation algorithms.

1.2.2 Parametric Shape Representation

In parametric form, points on the surface can be specified in terms of the parametric variables (u, v) which can be restricted, without loss of generality, to lie on the unit square $[0,1] \times [0,1]$. Generally parametric forms can be used to represent complex object geometries and generate realistic views. The use of control points allows local finite control of the surface which allows accurate modeling of objects, and thus they have the useful ability to identify specific locations on a surface. Complex objects can be modeled using parametric patches that are joined together.

A common parametric forms are the nonuniform rational B-spline (NURBS), the Rational Gaussian (RaG) surface[Gos93], and the Fourier Shape Descriptions. However, fitting of parametric patches of NURBS to arbitrary surface regions is not easy (or impossible). Furthermore, control points determining the shape of a parametric surface are not easily detectable and non-unique. In order to solve the fitting difficulties, Campbell and Flynn [CF01] propose using parametric forms for initial object specification from which a polygonal mesh or other representation can then be generated. Also Rational Gaussian (RaG) surface is proposed by [Gos93], which has a very simple function basis so as to be fitted easily once the parametric parameters are known. The shortcoming of these methods is limited in the condition that an efficient parameterization method is obtained first, and their use for shape recognition purposes has been limited because of the non-uniqueness of their parameters. Fourier Shape representation is designed for the shape recognition with the geometric invariants. But Fourier Shape representation has the limitation for application of non-star shapes, to closed curves, to curves that contain gaps, and to unordered data.

NURBS

The nonuniform rational B-spline (NURBS) is defined as

$$S(u,v) = \sum_i \sum_j B^h_{i,j} N_{i,k}(u) M_{i,l}(v), \tag{1.2}$$

where $N_{i,k}(u)$ and $N_{i,l}(v)$ are rational B-spline functions of order k and l, and $B^h_{i,j}$ are the homogeneous coordinates of control points [Pie91].

Representation defined by NURBS is a very efficient way of surface representation due to their boundedness, continuity, local controllability and affine invariance. It has been used extensively in computer graphics and computer-aided design due to these properties. However, it is difficult to make a surface defined on a parametric rectangle fit an arbitrary region on the surface of an object. Moreover, the homogeneous control points are also not easily detectable nor unique.

RaG surface

In order to solve the fitting problem, Rational Gaussian (RaG) surface [Gos93] is defined as

$$S(u,v) = \sum_i V_i g_i(u,v), \tag{1.3}$$

with

$$g_i = \frac{w_i G_i(u,v)}{\sum_j w_j G_j(u,v)}, \tag{1.4}$$

where $G_i(u, v)$ is the Gaussian kernel function. RaG surface has the simpler form that make it can be easily to fit to a dataset. It also supports the shapes such as torus [Gos93] with specified parameterization, but which is difficult to NURBS.

Fourier Shape representation

A significant advantage of Fourier Shape Descriptor is the applicability to object recognition. The two main types of Fourier Shape Descriptors are: 1) that which represents a shape as a radius as a function of angle, and 2) that which represents shape coordinates as a complex valued function of arc-length. Disadvantages of Fourier Shape representation is the disability to non-star shapes, to closed curves, to curves that contain gaps, and to unordered curve data. There are certain disadvantages to both approaches. (i) One disadvantage is limited to the set of star-shapes; however, many interesting shapes do not fall into this category. (ii) The other disadvantage is that these approaches require the input data sets to be an ordered set of points. Disadvantage (i) can not be directly used for open shapes. Disadvantage (ii) can be used for open curves, but serious difficulties with arc-length normalization arise. Arc-length parameterization is the main drawback of disadvantage (ii) because arc-length can increase significantly if noise is added to the curve. Both have problems with varying data point density and gaps in the data.

1.2.3 Implicit Shape Representation

Implicit representations define objects implicitly as a particular isosurface (usually zero) of some function f. An implicit surface is mathematically defined as the set of points $\mathbf{x} \in \mathcal{R}^3$, satisfying the implicit function $f(\mathbf{x}) = 0$, where $f : \mathcal{R}^3 \longrightarrow \mathcal{R}$. It is also called the zero set or kernel of f and sometimes written as $f^{-1}(0)$. Points on the inside and the outside of the surface are usually chosen to have opposite signs.

Implicit surfaces have two important classes depending on the way the implicit function f is defined: Radial Basis Function (RBF) and algebraic surfaces.

RBF

For RBF surfaces the implicit function f is a sum of radially symmetric functions with Gaussian profiles. The general form is

$$\begin{aligned} f(\mathbf{x}) &= -t + \sum_{i=1}^{n} \lambda_i \Phi(\| \mathbf{x} - \mathbf{x}_i \|) \\ &= 0 \end{aligned} \tag{1.5}$$

where Φ descirbes the radial profile and t represents the isocontour threshoud. Typically, the function Φ is of the form: Gaussian

$$\Phi(\mathbf{x}) = \exp(\| \mathbf{x} \|^2 / \delta_i^2). \tag{1.6}$$

where δ_i is the standard deviation fo the Gaussian. Turk and OBrien [TO99b, TO99a] use Thin-plate splines as radial basis functions:

$$\Phi(\mathbf{x}) = \| \mathbf{x} \|^2 \log(\| \mathbf{x} \|) \tag{1.7}$$

to fit an interpolating surface to a set of irregularly spaced points. The coefficients of the interpolating spline are determined by solving a sparse system of linear equations. However, since the general radial basis functions span over a limited approximation domain, which are used to support to cover the entire domain, a very large and sparse matrix of coefficients is obtained, and thus an efficient solution of the system of equations is often required.

Algebraic surfaces

Implicit polynomial (IP) curves and surfaces, usually called algebraic 3D surfaces. For algebraic surfaces the implicit function f is a polynomial expression in x, y and z described as: (which will be introduced in detail from next section)

$$f(\mathbf{x}) = \sum_{0 \le i,j,k,i+j+k \le n} a_{ijk} x^i y^j z^k. \tag{1.8}$$

And the implicit function of f, namely $f = 0$, represents the surface of target object. For example,

$$f(\mathbf{x}) = -1 + x^2 + y^2 + z^2 = 0 \tag{1.9}$$

represents the surface of a unit sphere. Algebraic surfaces can interpolate between data points in order to handle missing data and they also have a smoothing property for handling noise and perturbations in data [BLC00, TC00a]. Coefficients of polynomials representing algebraic entities encode global shape information about the represented entity [TC92]. However, it suffers from the difficulty on capturing local features.

1.2.4 Comparison on Shape Representations

To summarize, Tab. 1.2 presents a comparison of the different surface representation of explicit, parametric and implicit techniques.

The disadvantages of implicit surfaces include difficulty of visualization and local control. However, the visualization problem is not a minor issue in computer vision, whereas local

Table 1.2: Comparison of surface represnetations

	Explicit	Parametric	Implicit
Data Interpolation	Difficult	Easy	Easiest
Smoothness	NO	Yes	Yes
Compact Representation	NO	Yes	Yes
Visualization	Easy	Easy	difficult
Surface Operations	Bad	Good	Very Good
Topology Preserving Deformation	Difficult	Easy	Easiest
Local Shape Control	Yes	Yes	No
Gradient Computation	Bad	Good	Good

control will be discussed in next subsection. On the other hand, implicit representation has the main advantages attracting the vision applications, including topological representation, compact representation and efficient computation of various surface operations. Moreover, algebraic implicit surfaces are particularly suited for the pose recovery problem. The interested reader is referred to [Blo97] for a nice introduction to implicit surfaces.

Implicit functions make both simple Boolean operations such as surface intersection and union easy to apply. Constructive solid geometry (CSG) reduces to merely looking at the signs of implicit functions without the need to consider geometry or topology [YT99]. With implicit surfaces, surface gradient, normal and curvature as well as volume and surface integrals can be easily defined. Well-developed level set methods [OF03] can be used with implicit surfaces to handle dynamic surfaces with changing topology. Therefore, the simple inside/outside tests associated with implicit surfaces make them ideal for shape transformation, collision detection, CSG, and blending.

In addition, the local feature-based shape representation, such triangulated meshes, NURBS, or RBF, which is more attractive for accurate applications, *e.g.*, it is capable of being applied into fine registration (such as ICP-based registration). But, on the contrary the global (model) feature-based shape representation, such as polynomials, is more attractive for fast cases, *e.g.*, it can easily extract the center point and orientation of objects, and then fast register the objects without an iteration approach (see [TC92, TC00a]).

The locality of an object representation may be of interest in applications of recognition or registration in the presence of occlusion. A representation that explicitly reveals local geometrical structure may be characterized as "occlusion tolerant", hence better suited to such applications as high-quality (accurate) recognition or registration. Unfortunately, representations that are chiefly local are generally verbose. This further motivates the use of multiple representations in applications where requirements conflict; this adds a burden of computation for maintaining consistency between representations.

The global-based representation, by contrast, shows the features of complete object, from simple polyhedra to natural quadrics to various object representations such as superquadrics. The intrinsic and principal information can be extracted fast, so that recognition or registration can be performed in performance. The potential lack of local features like crease edges or linear silhouette edges makes this feature identification task difficult for the partial overlapping objects.

In this thesis, our interest is focused on the representation of using implicit polynomials (IPs), algebraic curve/surface representation, since IP representation has the characteristics, in contrast to other representations, superior to fast fitting, few parameters, algebraic/geometric invariants, and robustness against noise and occlusion (the detail will be given in Section 2.4) and thus it is very suitable and effective for vision applications.

1.3 Summary of Contributions

IP representation have been studied in recent two decades but not sufficiently and the limitations exist so that it is not be widely applied. IP representation mainly suffers from the issues that greatly limit its applicability, and often confuse the users who need to model their target objects with IPs.

In this study, we address these current technical issues for IP, sufficiently take advantages of IP's characteristics, exploit and extend the potentials of IP representation. and bring the new concepts for vision application of using IP.

We make contributions towards the solutions of three computer vision problems. The first one is based on a segmentation method for improvement of IP representation; The second one is the development of an adaptive and robust implicit polynomial curve/surface estimation method which is for general purpose but brings a new extension for our segmentation method in the first contribution; Guaranteed by the result of second contribution, the third contribution is the exploitation of a new registration method of using IP gradient flow. To prove the effectiveness for the above methods, we also open up the vista for the applications for each of them.

In general, our first and second contributions address the two current issues of IPs: poor representation for complex shapes and difficulty of determining a moderate IP degree and achieving global/local stability for fitting as follows.

- Poor representation for complex shapes

  Although IP representation is capable of encoding geometric properties easily to the target objects with desired smoothness, compact set of parameters, algebraic/geometric invariants, and robustness to noisy data, it suffers from the poor representation for local features. Objects obtained by vision modalities are often complex but IP can only coarsely represents the global features for objects.

- Difficulty of determining a moderate IP degree and achieving global/local stability for fitting

  Shapes with different complexities should be encoded with IPs in different degrees, *e.g.*, degree one for a line and degree two for an ellipse, but the prior IP fitting methods are inflexible for deciding the degree of IP. Not only it requires determining the degrees before handling fitting, but also it severely suffers from computational instability that many redundant zero-level sets are generated around the desired one, which makes the fitting result nearly impossible to be interpreted in desired region space.

In order to address the above issues, this thesis presents improvements of IP representation: a segmentation method for improve IP representation for complex objects; and an adaptive and stable fitting method for improving single IP fitting problem; furthermore, we developed a new and fast registration method of using IP gradient flow.

- In order to address the first issue, we propose a 3D segmentation method based on a cut-and-merge approach. Through the method, a complex surface can be divided into segments and each segment is encoded by a low-degree IP. The advantages are that it is capable of 1) finding the appropriate segmentation for various desired accuracies; 2) achieving stable fitting since we avoid using high-degree IPs; and 3) decomposing heavy computation from one high-degree IP fitting into light computations from multiple low-degree IPs. The segmentation result opens up new vista for range image application. We study the applicability of registration method for range image, of which two major advantages over the conventional methods is that 1) the registration is a non-iterative approach and thus fast; 2) no initialization is required.

- For the second issue, we propose an degree-adaptive, stable and accurate fitting method for IP. As a result, the method is capable of: 1) adaptively determining the moderate degree for fitting that depends on the complexity of objects, in an incremental manner without greatly increasing the computational cost; 2) retaining global stability while also avoiding excessive loss of local accuracy. To obtain an extending version of the above segmentation method, this method is well applied , and thus adaptive segmentation result is available.

- In addition, based on the stable IP representation obtained by our second method, we develop a fast registration technique making use of implicit polynomials gradients that has been effectively applied into ultrasound image pose estimation for medical purpose. The efficiency is benefit from three main advantages over the traditional registration method. 1) a fast transformation method for IP coefficients is introduced; 2) the alignment is formulated as to minimize an energy functional derived from IP gradient flow, and thus avoid the cost for calculating point-wise correspondences; 3) with the property of IP having a few coefficients, it makes both IP gradients and its transformation can be calculated in an extremely light manner both on CPU time and memory; 4) applying a real-time US image pose estimation, we demonstrate the capabilities of overcoming the limitations of unconstrained freehand US data, resulting in robust and fast pose estimation.

1.4 Overview of the Thesis

The following chapters are organized as follows.

Chapter 2 introduces the preliminary of implicit polynomials starting with a review of the basic mathematic formulation of IP; then the current IP fitting methods are reviewd, including non-linear methods and linear methods; following the fitting stabilization, the IP's properties are introduced. Also these mathematical preliminary will be needed for later chapters.

Chapter 3 presents the first contribution of this thesis, a 3D segmentation method, followed by an application on non-iterative registration for range images.

Chapter 4 presents the second contribution of this thesis, a adaptive and stable fitting method. Also the result directly brings the extension on an adaptive segmentation method.

Chapter 5 brings the third contribution, by giving a new extension for a fast registration method by using of IP gradient field. And then a medical applications on 6-DOF pose estimation of single ultrasound image is developed.

Chapter 6 summarizes the thesis, clarifies the contributions and discusses a few directions for future research.

Chapter 2

Implicit Polynomial

In this chapter we will present the background of implicit polynomials needed for later chapters. The four main areas introduced in this chapter are

1. Basic mathematic formulation of IP representation,
2. Overview of IP fitting methods,
3. Symbolic computation of IP,
4. Comparison between IP and other function based methods.

This chapter is organized as follows. First Section 2.1 introduces the basic mathematic formulation of IP representation; Then Section 2.2 reviews the state of the art on IP fitting methods, especially for the linear-type methods. Section 2.3 introduces a symbolic computation for IP including the intrinsic properties (principal axes, orientation, Euclidean invariants) computation of IP. Finally Section 2.4 provides a comparison to the other function based representations. All of the preliminaries described in this chapter will be helpful for understanding the content in later chapters.

2.1 Definition of Implicit Polynomial

Implicit Polynomial (IP) is an implicit function defined in a multivariate polynomial Form. Formally, a planar algebraic curve is specified by the zero level set of a 2D polynomial of degree

n given by

$$\begin{aligned} f_n^{2D}(\mathbf{x}) &= \sum_{0\le i,j;i+j\le n} a_{ij}x^iy^j \\ &= a_{00} + a_{10}x + a_{01}y + a_{20}x^2 + a_{11}xy + a_{02}y^2 + \cdots \\ &\quad + a_{n,0}x^n + a_{n-1,1}x^{n-1}y + \cdots + a_{0,n}y^n \\ &= \underbrace{(1\ x\ y\ x^2\ xy\ y^2\ \ldots\ y^n)}_{\mathbf{m}(\mathbf{x})^T}\underbrace{(a_{00}\ a_{10}\ a_{01}\ a_{20}\ a_{11}\ a_{02}\ \ldots\ a_{0n})^T}_{\mathbf{a}}, \end{aligned} \tag{2.1}$$

and a 3D algebraic surface is defined by the zero level set of a 3D IP of degree n denoted by

$$\begin{aligned} f_n^{3D}(\mathbf{x}) &= \sum_{0\le i,j,k;i+j+k\le n} a_{ijk}x^iy^jz^k \\ &= a_{000} + a_{100}x + a_{010}y + a_{001}z + a_{200}x^2 + a_{110}xy + a_{101}xz + a_{020}y^2 + a_{011}yz + a_{002}z^2 \cdots \\ &\quad + a_{n,0,0}x^n + a_{n-1,1,0}x^{n-1}y + a_{n-1,0,1}x^{n-1}z + \cdots + a_{0,n,0}y^n + a_{0,n-1,1}y^{n-1}z + \cdots + a_{0,0,n}z^n \\ &= \underbrace{(1\ x\ y\ z\ x^2\ xy\ xz\ y^2\ yz\ \ldots\ z^n)}_{\mathbf{m}(\mathbf{x})^T}\underbrace{(a_{000}\ a_{100}\ a_{010}\ a_{001}\ a_{200}\ a_{110}\ a_{101}\ a_{020}\ a_{011}\ \ldots\ a_{00n})^T}_{\mathbf{a}}, \end{aligned} \tag{2.2}$$

where variable $\mathbf{x}$ is a 2D/3D point with coordinate $\{x, y\}$ or $\{x, y, z\}$ in f_n^{2D} and f_n^{3D} respectively. An IP can be represented as an inner product between a monomial vector $\mathbf{m}(\mathbf{x})$ and a coefficient vector $\mathbf{a}$ as $\mathbf{m}(\mathbf{x})^T\mathbf{a}$. The well-known algebraic curves can be described by IPs of different degrees, *e.g.*, an IP curve of degree 2 is a conic, degree 3 a cubic, degree 4 a quartic, and so on. In general, the early works, such as [TC92], have proposed a meaningful rule for the order of multi-indices $\{i, j, k\}$ in a degree-increasing manner.

2.1.1 Homogeneous Form of IP

The *homogeneous binary polynomial* h_r of degree r in x, y, and z, $h_r = \sum_{i+j+k=r} a_{ijk}x^iy^jz^k$, is called the r-th degree form of the IP. Thus polynomial of degree n also can be described as:

$$f_n = \sum_{r=0}^{n} h_r. \tag{2.3}$$

For a polynomial which has N variables, in a form of degree r, there are $\begin{pmatrix} N+r-1 \\ N-1 \end{pmatrix}$ monomials (and thus coefficients) and in a polynomial of degree n, there are $\begin{pmatrix} N+n \\ N \end{pmatrix}$. For example, a 2D polynomial f of degree 3 consists of $\begin{pmatrix} 2+3 \\ 2 \end{pmatrix} = 10$ monomials and can be written as:

$$\begin{aligned} f_3^{2D}(x,y) &= a_{00} + a_{10}x + a_{01}y + a_{20}x^2 + a_{11}xy + a_{02}y^2 + a_{30}x^3 + a_{21}x^2y + a_{12}xy^2 + a_{03}y^3. \\ &= \begin{pmatrix} 1 & x & y & x^2 & xy & y^2 & x^3 & x^2y & xy^2 & y^3 \end{pmatrix} \begin{pmatrix} a_{00} \\ a_{10} \\ a_{01} \\ a_{20} \\ a_{11} \\ a_{02} \\ a_{30} \\ a_{21} \\ a_{12} \\ a_{03} \end{pmatrix}. \end{aligned} \tag{2.4}$$

2.1.2 Multi-indices Mapping

Multi-indices of an IP can be linearly ordered in many different ways. In computer vision field, it often uses the (inverse) *lexicographical order* [TC92], but the same results can be obtained using other orders. Without lose of generality, let us explain it using 3D polynomials. Suppose $\boldsymbol{\alpha} = \alpha_1, \alpha_2, \alpha_3$ is a multi-index of a 3D IP (it is corresponding to (i, j, k) in (2.2), and $\boldsymbol{\beta}$ is another multi-index in the same form of $\boldsymbol{\alpha}$, *i.e.*, $(\alpha_1 + \alpha_2 + \alpha_3) = (\beta_1 + \beta_2 + \beta_3)$. We say that $\boldsymbol{\alpha}$ precedes $\boldsymbol{\beta}$, and write $\boldsymbol{\alpha} < \boldsymbol{\beta}$ if, for the first index k such that α_k differs from β_k, we have $\alpha_k < \beta_k$. For example, for multi-indices of form 2 in 3D IP, the lexicographical order is

$$(2,0,0) < (1,1,0) < (1,0,1) < (0,2,0) < (0,1,1) < (0,0,2). \tag{2.5}$$

If $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are multi-indices of different form, and $(\alpha_1 + \alpha_2 + \alpha_3) < (\beta_1 + \beta_2 + \beta_3)$, we also say that $\boldsymbol{\alpha}$ precedes $\boldsymbol{\beta}$, and write $\boldsymbol{\alpha} < \boldsymbol{\beta}$.

If we map these degree-increasing multi-indices (i, j) of 2D or (i, j, k) of 3D to linear indices l, the linear mapping can be shown in Tab. 2.1. Furthermore we can find the relationship between in the multi-indices and linear indices as:

$$l = j + \frac{(i+j+1)(i+j)}{2} + 1 \tag{2.6}$$

for 2D and

$$l = k + \frac{(j+k+1)(j+k)}{2} + \frac{(i+j+k+2)(i+j+k+1)(i+j+k)}{6} + 1 \tag{2.7}$$

for 3D. Tab. 2.1 also shows some 2D/3D forms with notation Fr for the r-th form.

Table 2.1: The relations between l and (i, j) in 2D and (i, j, k) in 3D, where l is the linear index for monomials, and i, j and k are the powers of x, y and z respectively.

l	$[i\ j]$	Form
1	[0 0]	F0 $(i + j = 0)$
2	[1 0]	
3	[0 1]	F1 $(i + j = 1)$
4	[2 0]	
5	[1 1]	
6	[0 2]	F2 $(i + j = 2)$
7	[3 0]	
8	[2 1]	
9	[1 2]	
10	[0 3]	F3 $(i + j = 3)$
	⋮	
m	$[0\ r]$	Fr $(i + j = r)$

(a) Index list for 2D

l	$[i\ j\ k]$	Form
1	[0 0 0]	F0 $(i + j + k = 0)$
2	[1 0 0]	
3	[0 1 0]	
4	[0 0 1]	F1 $(i + j + k = 1)$
5	[2 0 0]	
6	[1 1 0]	
7	[0 2 0]	
8	[1 0 1]	
9	[0 1 1]	
10	[0 0 2]	F2 $(i + j + k = 2)$
	⋮	
m	$[0\ 0\ r]$	Fr $(i + j + k = r)$

(b) Index list for 3D

2.1.3 2D and 3D Examples

A 2D IP example is illustrated in Fig. 2.1. A IP curve is represented by the zero set of a 2D IP $f(\mathbf{x})$, *i.e.*, the set of points (x, y) satisfying the IP equation $f(x, y) = 0$ which is the intersection of the surface defined by an explicit polynomial $z = f(x, y)$ with the plane $z = 0$.

A 3D IP example is shown in Fig. 2.2, where suppose a volume region be defined by an explicit polynomial $v = f(x, y, z)$. Then a IP surface is represented by the set of points (x, y, z) satisfying the IP equation $f(x, y, z)$ which is the intersection of the volume defined by $v = f(x, y, z)$ with the isosurface $v = 0$.

2.2 Overview of the IP Fitting Methods

The fitting methods aim at finding a coefficient vector $\mathbf{a}$ by minimizing the distance between the data set and the zero set of IP to give the "best" representation, such as:

$$\mathbf{a}_{min} = \arg\min_{\mathbf{a}}\{\frac{1}{N}\sum_{i=1}^{N} dist(\mathbf{x_i}, \mathcal{Z}(f_n))\} \tag{2.8}$$

where $dist(\mathbf{x_i}, \mathcal{Z}(f_n))$ means a certain distance from the data point $\mathbf{x}_i$ to the zeros set $\mathcal{Z}(f_n) = \{\mathbf{x} : f_n(\mathbf{x}) = 0\}$ of the IP. According to the definition of $dist(\mathbf{x_i}, \mathcal{Z}(f_n))$, fitting methods are classified

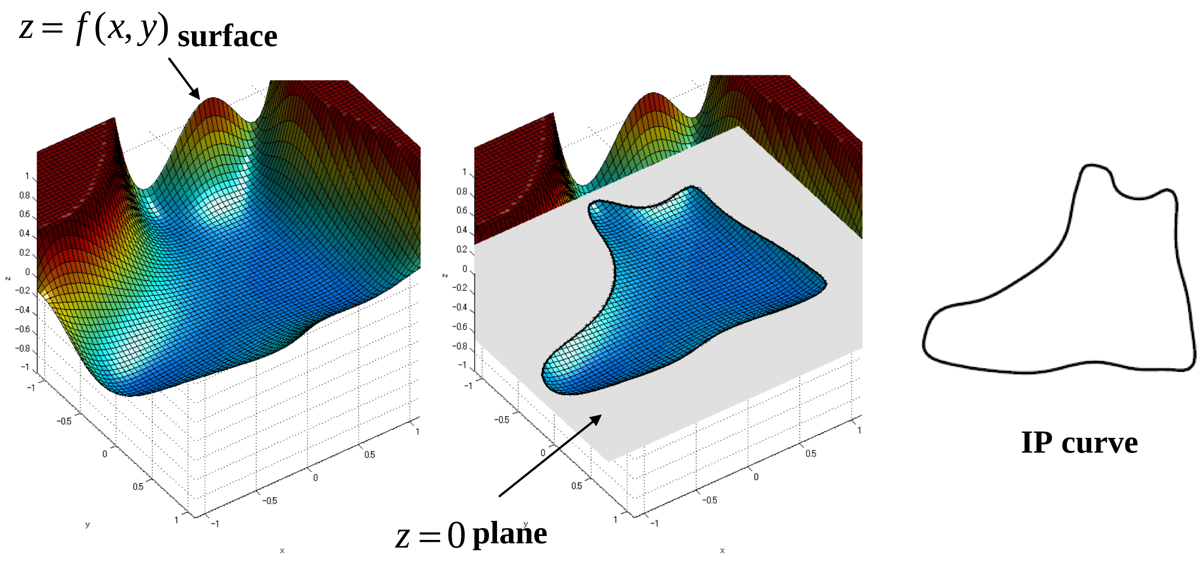


Figure 2.1: IP curve: Intersection of the polynomial $z = f(x, y)$ with the plane $z = 0$, yields the IP curve $f(\mathbf{x}) = 0$ which describes a "boot" shape in the 2D plane.

into two categories:

- the non-linear methods using the geometric distance or Euclidean approximation [Tau91, KC94, Kna94]
- the linear methods based on the algebraic distance [BLC00, TTC00, HBM04, SU05]

Because the linear methods are simpler and much faster than the nonlinear ones, they are receiving attentions.

In a general case, for the minimization of Eq. (2.8) it can be formulated as a linear least square problem of using linear constraints:

$$\begin{aligned} \mathbf{m}(\mathbf{x}_1)^{\mathrm{T}}\mathbf{a} &= 0, \\ \mathbf{m}(\mathbf{x}_2)^{\mathrm{T}}\mathbf{a} &= 0, \\ &\vdots \\ \mathbf{m}(\mathbf{x}_n)^{\mathrm{T}}\mathbf{a} &= 0. \end{aligned}$$

Then it can be formulated as a symmetric linear system of equations as

$$M^{\mathrm{T}}M\mathbf{a} = M^{\mathrm{T}}\mathbf{b}, \tag{2.9}$$

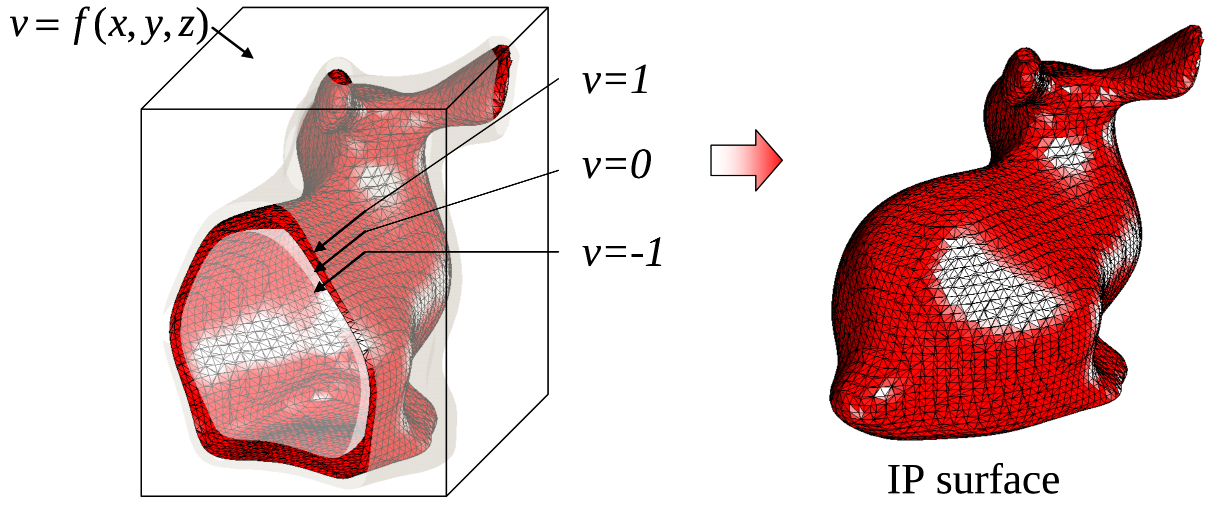


Figure 2.2: IP surface: Intersection of the polynomial $v = f(x, y, z)$ with the isosurface $z = 0$, yields the IP surface $f(\mathbf{x}) = 0$ which describes a "rabbit" shape in the 3D space.

where M is the matrix of monomials with the rows of original one are $\mathbf{m}(\mathbf{x_i})$ as

$$M = \begin{pmatrix} \mathbf{m}(\mathbf{x}_1) \\ \mathbf{m}(\mathbf{x}_2) \\ \vdots \\ \mathbf{m}(\mathbf{x}_n) \end{pmatrix},$$

and the right-hand vector $\mathbf{b}$ is a vector of zeros originally. Note formula (2.9) is just transformed from the least square result, $\mathbf{a} = M^{\dagger}\mathbf{b}$, where $M^{\dagger} = (M^T M)^{-1} M^T$ is the pseudo-inverse matrix.

2.2.1 Non-linear Methods

Actually, we got a classical linear LS problem in (2.9). It can easily be seen that the trivial solution, $\mathbf{a} = \mathbf{0}$ (i.e., $\mathbf{a}$ is a vector of zeros of size $\frac{(r+1)(r+2)}{2}$), is an immediate solution of this problem. In order to avoid this solution, Non-linear method added the following constraint:

$$\| \mathbf{a} \| = c, \tag{2.10}$$

where c is positive constant. The solution of this problem is well-known. It is given by the eigenvector corresponding to the minimal eigenvalue of $M^{\mathrm{T}}M$.

Taubin [TC92] also derives an approximation of the point to implicit entity distance as follows. A closed form expression for $dist(\mathbf{x}, Z(f))$ does exist when f is linear, *i.e.* a first degree polynomial. In that case, the gradient $\nabla f(x)$ is constant and the following identity is satisfied:

$$f(\mathbf{y}) \equiv f(\mathbf{x}) + \nabla f(\mathbf{x}) \cdot (\mathbf{x} - \mathbf{y}) \tag{2.11}$$

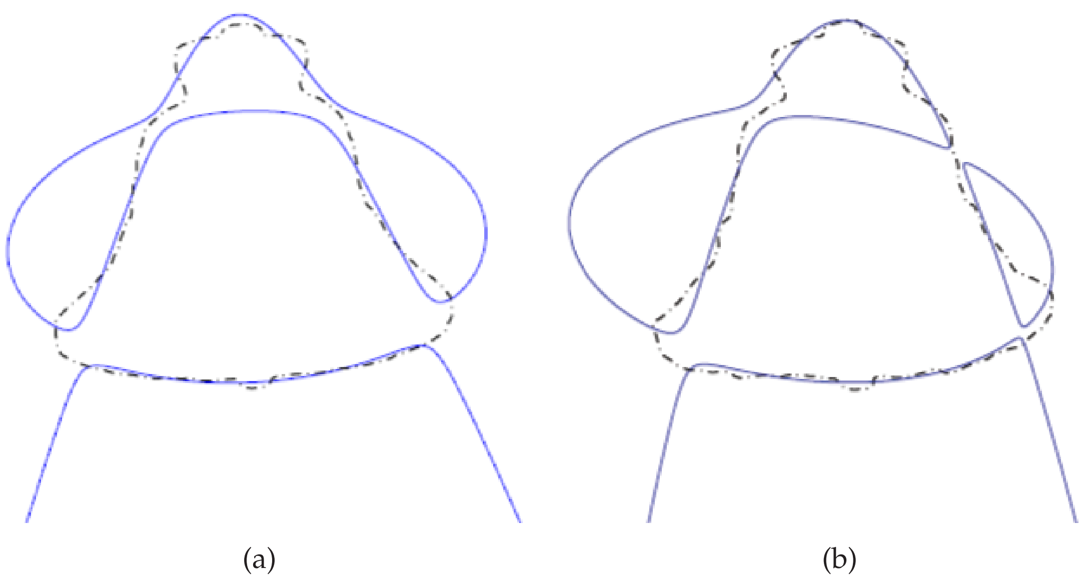


Figure 2.3: Classical least-squares algorithm gives unstable 4th degree IP fits in (a) and (b), when the data is little perturbed with noise. Dash line is the original data set and blue line is the fitting results.

for a point $\mathbf{y} \in Z(f)$ and $\mathbf{x}$, and approximate distance of (2.8) is given by

$$dist(\mathbf{x}, Z(f))^2 \approx \frac{f(\mathbf{x})^2}{\| \nabla f(\mathbf{x}) \|^2}. \tag{2.12}$$

Taubin [Tau91] has shown that the approximate distance function is independent of the representation of $Z(f)$, is invariant to rigid body transformations and is proportional to scale transformations. Thus substitute (2.12) into (2.8) the problem can be formulated as Generalized Eigenvector fitting problem [Tau91] by the following approximation:

$$\frac{1}{N}\sum_{i=1}^{N} \frac{f(\mathbf{x}_i)^2}{\| \nabla f(\mathbf{x}_i) \|^2} \approx \frac{\frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}_i)^2}{\frac{1}{N}\sum_{i=1}^{N} \| \nabla f(\mathbf{x}_i) \|^2}. \tag{2.13}$$

The right hand side can be written as

$$\frac{\frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}_i)^2}{\frac{1}{N}\sum_{i=1}^{N} \| \nabla f(\mathbf{x}_i) \|^2} = \frac{\mathbf{a}^{\mathrm{T}} M^{\mathrm{T}} M \mathbf{a}}{\mathbf{a}^{\mathrm{T}} N^{\mathrm{T}} N \mathbf{a}} \tag{2.14}$$

where N is the matrix of partial difference of monomials, and the each 3 rows of N are $\frac{\partial \mathbf{m}(\mathbf{x_i})}{\partial \mathbf{x}}$. $M^{\mathrm{T}}M$ and $N^{\mathrm{T}}N$ are non-negative definite and symmetric matrices that are functions of only the data points. Eq. (2.14) is minimised by the eigenvector corresponding to the minimum eigenvalue of the pencil $(M^{\mathrm{T}}M - \lambda N^{\mathrm{T}}N)\mathbf{a} = 0$ and hence the minimization problem has been reduced to a generalized eigenvalue problem.

The derived solution has the same drawbacks that the algorithms in [Tau91, KC94] have. These drawbacks are now more intensive to small perturbation because there could be several small eigenvalues, each of them corresponding to different eigenvectors. If this is the case, then a small perturbation of the data may cause small changes in both eigenvectors and eigenvalues, but result in a much different ″minimal-eigenvalue″ eigenvector, and thus in a totally different solution. An example of the above problems is demonstrated in Fig. 2.2.1.

2.2.2 Linear Methods

To improve the linear fitting, some constraints need to add into the computations. For example, 3L method [BLC00] adds two additional layer points into M; Gradient-One method [TTC00] adds the constraints from local normals and tangents into the original scatter matrix; Min-Max method [HBM04] adds the constraints from the approximation of gradients and normals. And the nonzero vector **b** can be derived corresponding to these constraints. Let us give their definition respectively in the followings.

3L method

3L method [BLC00] adds the constraints which restricts the data sets of two imaginary layers (level-sets) Γ^+ and Γ^- (external and internal) around the original data set to belong to the two nonzero level-sets (e.g., $\pm\varepsilon$ sets, where ε is some small positive value) of IP. The additional constraints can be described as:

$$\begin{cases} f(\mathbf{x}_+) = +\varepsilon, & \text{if } \mathbf{x}_+ \in \Gamma^+ \\ f(\mathbf{x}_-) = -\varepsilon, & \text{if } \mathbf{x}_- \in \Gamma^-. \end{cases} \tag{2.15}$$

The distance from these newly created levels-sets to the original data-set is ε. That is, the minimal distance from any point belonging to the level-sets to the original data-set is ε. Note the imaginary layers Γ^+ and Γ^- can be constructed from the points that are in the normal directions to the original data points with distance $\pm\varepsilon$.

Therefore, according to (2.9) the least-square method formulation of 3L method can be written as:

$$M_{3L}^T M_{3L}\mathbf{a} = M_{3L}^T \mathbf{b}_{3L}. \tag{2.16}$$

The monomial matrix M_{3L} and right-hand vector $\mathbf{b}_{3L}$ are written as:

$$M_{3L} = \begin{pmatrix} M \\ M_{int} \\ M_{ext} \end{pmatrix}, \tag{2.17}$$

and

$$\mathbf{b}_{3L} = \begin{pmatrix} \mathbf{0}_{n\times1} \\ -\boldsymbol{\varepsilon}_{n\times1} \\ +\boldsymbol{\varepsilon}_{n\times1} \end{pmatrix}, \tag{2.18}$$

where matrices M_{int} and M_{ext} are the monomial matrix derived from the internal and external layer date sets respectively, and in right-hand vector, $\mathbf{0}_{n\times1}$, $-\boldsymbol{\varepsilon}_{n\times1}$ and $+\boldsymbol{\varepsilon}_{n\times1}$ mean vectors consisting of n elements 0s, $-\varepsilon$s and $+\varepsilon$s respectively.

Instead of demanding positive values inside the data set and negative outside, the 3L algorithm requires specific values ε at specific points in the vicinity of the data-set. Too small ε (too narrow ribbon) might result in global instability and too large ε (too narrow ribbon) might result in local inaccuracy.

Gradient-One method

Instead of requiring the parameter ε of 3L method, *Gradient-One* method [TTC00] exploits properties of the polynomial's gradient at points belonging to the zero set. It is well known that the gradient vector at a point belonging to the zero set is perpendicular to the zero-set tangent at this point. Assuming that the desired zero-set should lie nearby the data-set, the Gradient-One algorithm requires that the gradient of the polynomial at the data points will be perpendicular to the data-set curve tangent. In addition, the algorithm requires that the value of the gradient will be equal to 1. That means, the derived surface's slope has to be the same (and equal to 1) at all the data-set points.

Gradient-One method adds the constraints that restrict the local normals and tangents obtained from the data set to be parallel and orthogonal to the gradient on the IP zero set respectively; The additional constraints can be described as:

$$\begin{cases} \triangledown f(\mathbf{x}_i)^T \mathbf{N}_i = 1 \\ \triangledown f(\mathbf{x}_i)^T \mathbf{T}_i = 0, \end{cases} \tag{2.19}$$

where $\mathbf{N}_i$ is the normal vector and $\mathbf{T}_i$ is the tangent vector at point $\mathbf{x}_i$.

Therefore, according to (2.9) the least-square method formulation of 3L method can be written as:

$$M_{G1}^T M_{G1} \mathbf{a} = M_{G1}^T \mathbf{b}_{G1}. \tag{2.20}$$

The monomial matrix M_{G1} and right-hand vector $\mathbf{b}_{G1}$ are written as:

$$M_{G1} = \begin{pmatrix} M \\ G_N \\ G_T \end{pmatrix}, \tag{2.21}$$

and

$$\mathbf{b}_{G1} = \begin{pmatrix} \mathbf{0}_{n\times 1} \\ \mathbf{1}_{n\times 1} \\ \mathbf{0}_{n\times 1} \end{pmatrix}, \tag{2.22}$$

where matrices G_N and G_T are the monomial matrix derived from the normals and tangents of the data points respectively, and in right-hand vector, $\mathbf{0}_{n\times 1}$ and $\mathbf{1}_{n\times 1}$ mean vectors consisting of n elements of 0s and 1s respectively.

Note, without demanding a specific value of the Gradient-One method, data set is always required to be regularized to keep numerical stability. That is, in advance, the data set should be centered with origin and each data point should be rescaled with the mean distance from points to origin.

Min-Max and Min-Var methods

To further keep the continuity of IP in fitting, *Min-Max* and *Min-Var* methods [HBM04] add the constraints that restrict the local normals obtained from the data set to be equal to a certain approximation of gradients on the IP zero set. The additional constraints can be described as:

$$(\frac{\nabla \mathbf{m}(\mathbf{x}_i)}{\| \nabla \mathbf{m}(\mathbf{x}_i) \|_\alpha})^T \mathbf{a} = \mathbf{N}_i, \tag{2.23}$$

where $\mathbf{N}_i$ is the normal vector, and $\| \cdot \|_\alpha$ is $\mathcal{L}_1$ norm for *Min-Max* and $\mathcal{L}_2$ norm for *Min-Var* method respectively.

Therefore, according to (2.9) the least-square method formulation of Max-Min and Max-Var method can be written as:

$$M_{max}^T M_{max} \mathbf{a} = M_{max}^T \mathbf{b}_{max}. \tag{2.24}$$

The monomial matrix M_{max} and right-hand vector $\mathbf{b}_{max}$ are written as:

$$M_{max} = \begin{pmatrix} M \\ N_x \\ N_y \\ N_z \end{pmatrix}, \tag{2.25}$$

and

$$\mathbf{b}_{max} = \begin{pmatrix} \mathbf{0}_{n\times 1} \\ \mathbf{n}_x \\ \mathbf{n}_y \\ \mathbf{n}_z \end{pmatrix}. \tag{2.26}$$

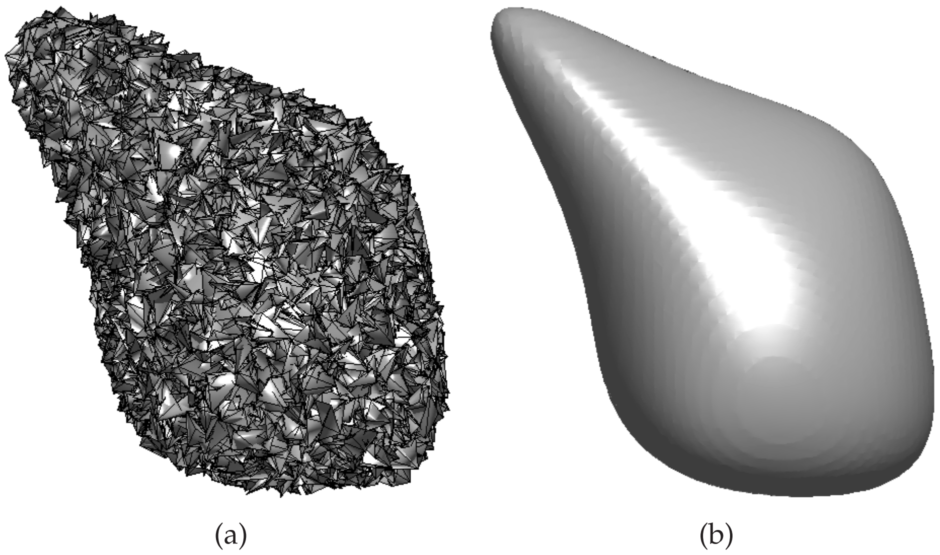


Figure 2.4: Fitting for a noisy data set using 3L method. (a) Original data set. (b) Fitting result of IP of degree 4.

For matrices N_x, N_y and N_z, the ith rows of which can be respectively written as:

$$(\frac{\partial \mathbf{m}(\mathbf{x}_i)/\partial x}{\| \nabla \mathbf{m}(\mathbf{x}_i) \|_\alpha})^T \mathbf{a},\ (\frac{\partial \mathbf{m}(\mathbf{x}_i)/\partial y}{\| \nabla \mathbf{m}(\mathbf{x}_i) \|_\alpha})^T \mathbf{a},\ (\frac{\partial \mathbf{m}(\mathbf{x}_i)/\partial z}{\| \nabla \mathbf{m}(\mathbf{x}_i) \|_\alpha})^T \mathbf{a}.$$

And in right-hand vector, $\mathbf{0}_{n\times 1}$ means vectors consisting of n elements of 0s and vectors $\mathbf{n}_x$, $\mathbf{n}_y$ and $\mathbf{n}_z$ consist of the elements derived from point normals, *e.g.*, the ith element of $\mathbf{n}_x$ is the x-partial element of normal at point $\mathbf{x}_i$.

Solving the linear systems

Now the different symmetric linear systems can be obtained in (2.16,2.20,2.24) according to the different linear methods: 3L method, Gradient-One method and Min-Max and Min-Var methods respectively. Since the coefficient matrix of each of them is symmetric, the linear systems can be solved fast by the solvers such as LU decomposition method, conjugate gradient method, etc.

These linear fitting methods are more robust than the non-linear fitting methods, since they take advantages of more feature information such as normals and tangents. The fitting result is proved more effective and more robust to the noises and occlusions of the data set. Two examples illustrate the robustness to noisy data and data with occlusion shown in Fig. 2.4 and Fig. 2.5 respectively.

Unfortunately, the above linear fitting methods only have the robustness in the ideal cases. That is, fitting for the objects with simple shapes they work very well, but for the objects with

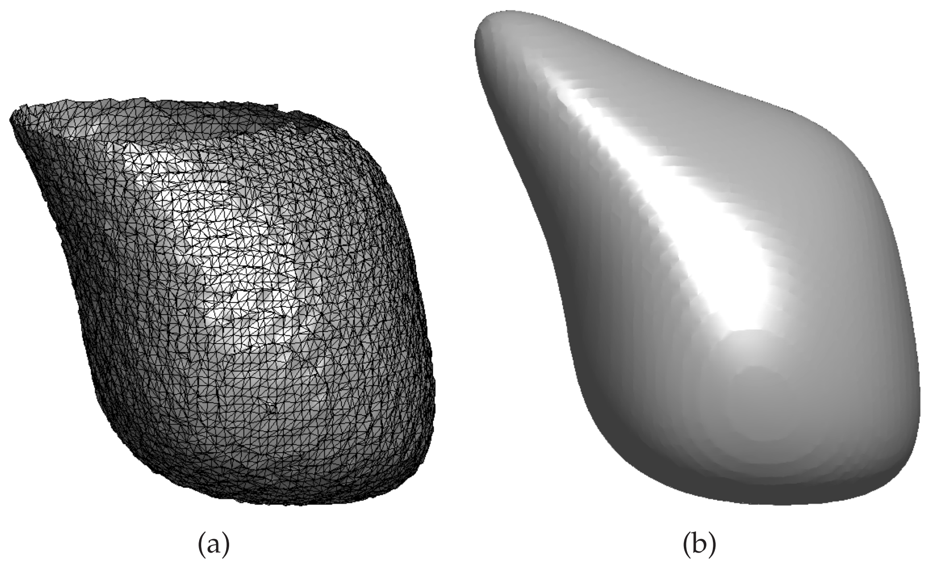


Figure 2.5: Fitting for a data set with occlusion using 3L method. (a) Original data set. (b) Fitting result of IP of degree 4.

complex shapes, they perform not well and furthermore they suffer from the global instability of fitting. Let us show this shorcoming with an example shown in Fig. 2.6. In Fig. 2.6, we can see that due to the instability of fitting for a complex object shown in Fig. 2.6 (a), there are many extra zero level sets generated in the 3D domain around the desired zero set. To overcome this problem, let us introduce a stabilization method in next subsection, which is also significantly related to our work in Section 4.

2.2.3 Stabilization with Rigid Regression

Although the linear methods improve the numerical stability for fitting to some extent, they still suffer from the difficulty in achieving global stability. Especially while modeling a complex shape with a high-degree IP, many extra zero sets are generated, for example the result shown in Fig. 2.6(b), which makes the fitting results nearly impossible to be interpreted. One important reason for global instability is the collinearity of column vectors of matrix M, namely, causing the matrix M^TM to be nearly singular (see [TTC00]) and its condition number to be infinite.

Addressing this issue, Tasdizen *et al.* [TTC00] and Sahin and Unel [SU05] 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 with the same dimension to matrix M^TM. Note that here we just consider the condition number as $\lambda_{max}/\lambda_{min}$, where λ_{max}

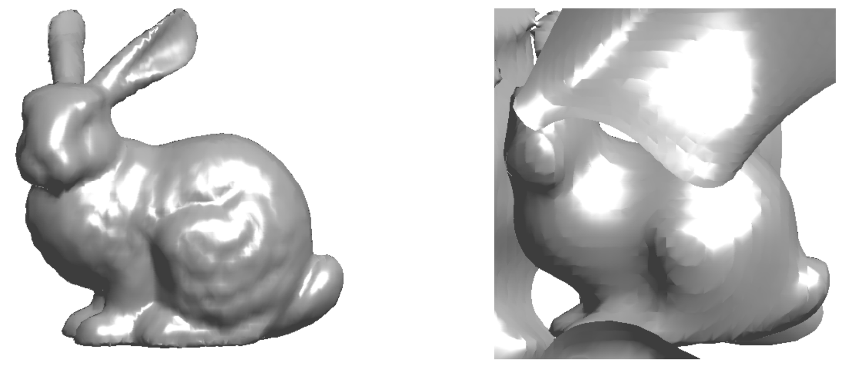

(a) (b)

Figure 2.6: Fitting for a data set with complex shape. (a) Original data set. (b) Fitting result of IP of degree 10.

and λ_{min} are the maximum and minimum eigenvalues respectively. Accordingly Eq. (2.9) can be modified as

$$(M^T M + \kappa D)\mathbf{a} = M^T \mathbf{b}. \tag{2.27}$$

To show the effectiveness for avoiding the ill-condition of $M^T M$, let us show a simple example as follow. Since matrix $M^T M$ is an symetric and positive matrix, it can be decompoed into

$$M^{\mathrm{T}} M = U^{\mathrm{T}} \Lambda U, \tag{2.28}$$

where U is an orthogonal matrix, and Λ is a diagonal matrix which consists of the eigenvalues of $M^{\mathrm{T}} M$. Supposing they are $\{1, 10^{-2}, 10^{-4}, 10^{-6}, 10^{-8}\}$, thus its condition number is $\frac{1}{10^{-8}} = 10^8$; If let D be a identity matrix and $\kappa = 1$, then the matrix $(M^T M + \kappa D)$ can be decomposed into

$$M^T M + \kappa D = U^T (\Lambda + \kappa D) U. \tag{2.29}$$

Now the diagonal matrix $\Lambda + \kappa D$ consists of the eigenvalues of the coefficient matirx of linear system, and they are changed to $\{2, 10^{-2} + 1, 10^{-4} + 1, 10^{-6} + 1, 10^{-8} + 1\}$. Thus the condition number is reduced to $\frac{2}{10^{-8}+1} \approx 2$.

Of course, the example shows an extreme case, but it illustrate the capability of RR method that the condition number can be dramatically decreased by adding the RR term. By this way, parameter κ should be increased from 0 to higher values until a stable closed bounded surface is obtained, and then the small eigenvalues of the coefficient matrix of linear system (4.7) are significantly increased, while the larger eigenvalues that contribute more to the overall shape of the fit are virtually unaffected, hence reducing the condition number to reasonable levels.

In general, D will be chosen to maintain Euclidean invariance. Although the simplest way is to set D to be an identity matrix, the more meaningful choices have been proposed by Tasdizen *et al*. for 2D [TTC00] and Sahin and Unel for 3D [SU05] that the l-th diagonal element of D is written as:

$$d_{ll} = \gamma \hat{t}, \tag{2.30}$$

where γ and $\hat{t}$ are two scalars.

In order to maintain Euclidean invariance for RR regularization, the diagonal matrix D was proposed to be calculated as: $d_{ll} = \gamma \hat{t}$ for the l-th diagonal element [TTC00, SU05]. Scalar γ is a free parameter for the $(i+j)$-th form (2D) or $(i+j+k)$-th form (3D) decided from the data set as follows:

$$\begin{aligned} \gamma_{i+j} &= \sum_{t=1}^{N} \| \mathbf{x}_t \|^{2(i+j)} \qquad (2D), \\ \gamma_{i+j+k} &= \sum_{t=1}^{N} \| \mathbf{x}_t \|^{2(i+j+k)} \qquad (3D), \end{aligned} \tag{2.31}$$

where the relation between l and (i, j, k) can be formulated as:

$$\begin{aligned} l &= j + \frac{(i+j+1)(i+j)}{2} + 1 \text{ for 2D and} \\ l &= k + \frac{(j+k+1)(j+k)}{2} \\ &\quad + \frac{(i+j+k+2)(i+j+k+1)(i+j+k)}{6} + 1 \text{ for 3D.} \end{aligned}$$

Notation $\| \cdot \|$ represents the $\mathcal{L}_2$ norm, and $\mathbf{x}_t$ represents the 2D/3D data point. $\hat{t}$ is designed for maintaining Euclidean invariance. It can be derived as:

$$\begin{aligned} \hat{t} &= \frac{i!j!}{(i+j)!} \qquad (2D), \\ \hat{t} &= \frac{i!j!k!}{(i+j+k)!} \qquad (3D). \end{aligned} \tag{2.32}$$

For example, for a 2D IP of degree 3, D can be described as

$$diag\{\gamma_0, \gamma_1, \gamma_1, \gamma_2, \frac{1}{2}\gamma_2, \gamma_2, \gamma_3, \frac{1}{3}\gamma_3, \frac{1}{3}\gamma_3, \gamma_3\};$$

for a 3D IP of degree 2, D can be described as

$$diag\{\gamma_0, \gamma_1, \gamma_1, \gamma_1, \gamma_2, \frac{1}{2}\gamma_2, \frac{1}{2}\gamma_2, \gamma_2, \frac{1}{2}\gamma_2, \gamma_2\}.$$

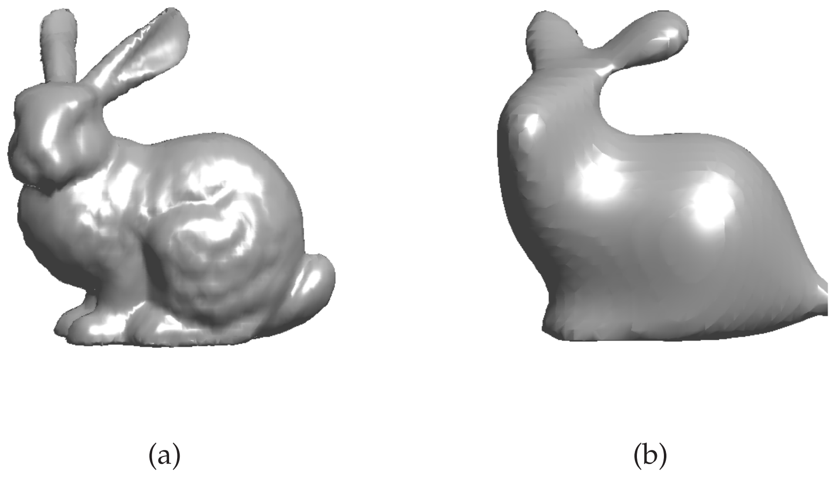


Figure 2.7: Fitting for a data set with complex shape. (a) Original data set. (b) Fitting result of IP of degree 10 improved RR method.

Let us show an example in Fig. 2.7 that improves the stability for the example of Fig. 2.6. From this example, we can see that the extra zero level sets have been removed and thus the fitting result achieves the global stability. However, the local accuracy is also broken due to the RR parameter improve the stability only in a global manner. This is also one of the issues to be addressed in this thesis and the detail will be described in Section 4.

2.3 Symbolic Computation of IP

In this section, let explain some Symbolic computations for IP including IP transformation and partial differential. First we introduce how algebraic entities represented by implicit polynomials can be transformed by manipulating the polynomial coefficients. Note, since this introduction is, according to [TC92], based on the IP's capability of object recognition or non-iterative registration, the true effective transformation was not given out in the literature, instead the existence of the transformation was proved.Second we explain how intrinsic geometry of $Z(f)$ can be extracted from the coefficients and how this leads to a simple method of implicit spatial registration that is a correspondence-free alternative to explicit registration methods such as ICP [BM92]. Third we introduce a simple symbolic computation for calculating the partial differential for IP.

2.3.1 Polynomial and Forms

The zero set $Z(f)$ of a polynomial f can be transformed by manipulating the polynomial coefficients. There are two ways of manipulating polynomial coefficients to affect zero set transformations, the matrix-based approach and the tensor-based approach. In the rest of this section, polynomials will be represented by lowercase letters of the Latin alphabet such as f whereas forms will be represented by Greek symbols such as ϕ.

If we define $F_\alpha = \alpha! a_\alpha$ where $\alpha!$ is a *multi-factorial*, then without loss of generality, a polynomial of degree 3, $f = \phi_0 + \phi_1 + \phi_2 + \phi_3$ can be rewritten as

$$\begin{aligned}
f(\mathbf{x}) &= \underbrace{\frac{F_{00}}{0!0!}}_{\phi_0} + \underbrace{\frac{F_{10}}{1!0!}x_1 + \frac{F_{01}}{0!1!}x_2}_{\phi_1(\mathbf{X})} \\
&\quad \underbrace{+\frac{F_{20}}{2!0!}x_1^2 + \frac{F_{11}}{1!1!}x_1x_2 + \frac{F_{02}}{0!2!}x_2^2}_{\phi_2(\mathbf{X})} \\
&\quad \underbrace{+\frac{F_{30}}{3!0!}x_1^3 + \frac{F_{21}}{2!1!}x_1^2x_2 + \frac{F_{12}}{1!2!}x_1x_2^2 + \frac{F_{03}}{0!3!}x_2^3}_{\phi_3(\mathbf{X})} \\
&= \underbrace{F_{00}}_{\phi_0(\mathbf{X})} + \underbrace{F_{10}x_1 + F_{01}x_2}_{\phi_1(\mathbf{X})} \\
&\quad \underbrace{+\frac{1}{2}F_{20}x_1^2 + F_{10}x_1x_2 + \frac{1}{2}F_{02}x_2^2}_{\phi_2(\mathbf{X})} \\
&\quad \underbrace{+\frac{1}{6}F_{30}x_1^3 + \frac{1}{2}F_{21}x_1^2x_2 + \frac{1}{2}F_{21}x_1^2x_2 + \frac{1}{6}F_{03}x_2^3}_{\phi_3(\mathbf{X})} \\
&= \sum_{r=0}^{3} \phi_r(\mathbf{x}). \qquad (2.33)
\end{aligned}$$

More generally, an n-Dimension polynomial in variables $\mathbf{x} = (x_1, \ldots, x_n)$ can be written as

$$f(x) = \sum_{\alpha} \frac{1}{\alpha!} F_\alpha \mathbf{x}^\alpha \qquad (2.34)$$

where the vector of non-negative integers $\alpha = (\alpha_1, \ldots, \alpha_n)$ is a multi-index of size $|\alpha| = \alpha_1 + \ldots + \alpha_n$,$\alpha! = \alpha_1! \ldots \alpha_n!$ is a multi-index factorial (or multi-factorial), F_α is a coefficient of degree $|\alpha|$, and $\mathbf{x}^\alpha = x_1^{\alpha_1} \ldots x_n^{\alpha_n}$ is a term of degree$|\alpha|$.

A 3D quadratic form can then be written as

$$\begin{aligned}\phi(x_1,x_2,x_3) &= a_{200}x_1^2 + a_{110}x_1x_2 + a_{101}x_1x_3 + a_{020}x_2^2 + a_{011}x_2x_3 + a_{002}x_3^2 \\ &= \frac{\Phi_{(2,0,0)}}{2}x_1^2 + \Phi_{(1,1,0)}x_1x_2 + \Phi_{(1,0,1)}x_1x_3 + \frac{\Phi_{(0,2,0)}}{2}x_2^2 + \Phi_{(0,1,1)}x_2x_3 + \frac{\Phi_{(0,0,2)}}{2}x_3^2 \\ &= \frac{1}{2}\begin{pmatrix} x_1 & x_2 & x_3 \end{pmatrix}\begin{pmatrix} \Phi_{(2,0,0)} & \Phi_{(1,1,0)} & \Phi_{(1,0,1)} \\ \Phi_{(1,1,0)} & \Phi_{(0,2,0)} & \Phi_{(0,1,1)} \\ \Phi_{(1,0,1)} & \Phi_{(0,1,1)} & \Phi_{(0,0,2)} \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}\end{aligned} \tag{2.35}$$

Taubin and Cooper [TC92] represent the 3×3 symmetric matrix as $\Phi_{[1,1]}$. The purpose of constructing $\Phi_{[1,1]}$ is that under a linear transformation $\mathbf{x}' = A\mathbf{x}$, this matrix of coefficients transforms as

$$\Phi'_{[1,1]} = A^{-\mathrm{T}}\Phi_{[1,1]}A^{-1}. \tag{2.36}$$

Hence, the coefficients of the transformed quadratic form can be computed using (2.36). Two questions that arise here concern the construction of the coefficient matrix and the transformation of forms of degree $r > 2$.

2.3.2 Construction of Covariant Matrices

For a form of degree j, let the coefficient vector $\Phi_{[j]}$ be defined as

$$\frac{\Phi_\alpha}{\sqrt{\alpha!}} : |\alpha| = j, \tag{2.37}$$

where Φ_α are the sysmbols instead of the values representing the coefficients. Let h_j be the number of monomials in a form of degree j. Then for forms of degree $j + k = r$ the $h_j \times h_k$ coefficient matrix $\Phi_{[j,k]}$ can be constructed from the vectors of lower degree forms $\Phi_{[j]}$ and $\Phi_{[k]}$ using

$$\Phi_{[j,k]} = \Phi_{[j]} \star \Phi_{[k]}^{\mathrm{T}} \tag{2.38}$$

where $\star$ represents the classic matrix multiplication with the diffierence that the individual element-wise multiplications are performed according to the rule $\Phi_\alpha\Phi_\beta = \Phi_{\alpha+\beta}$. Such a symbolic multiplication gives the set of coefficients

$$\{\Phi_{\alpha+\beta} : |\alpha| = j, |\beta| = k\}. \tag{2.39}$$

As an example, to form $\Phi_{[1,1]}$ as above, first the vector $\Phi_{[1]}$ is formed as:

$$\Phi_{[1]} = \begin{pmatrix} \frac{\Phi_{(1,0,0)}}{\sqrt{1!0!0!}} \\ \frac{\Phi_{(0,1,0)}}{\sqrt{0!1!0!}} \\ \frac{\Phi_{(0,0,1)}}{\sqrt{0!0!1!}} \end{pmatrix} = \begin{pmatrix} \Phi_{(1,0,0)} \\ \Phi_{(0,1,0)} \\ \Phi_{(0,0,1)} \end{pmatrix}. \tag{2.40}$$

Then $\Phi_{[1,1]}$ is formed by $\Phi_{[1]} \star \Phi_{[1]}^{rmT}$ with the element-wise multiplications performed as $\Phi_\alpha \Phi_\beta = \Phi_{\alpha+\beta}$:

$$\begin{aligned}\Phi_{[1,1]} &= \Phi_{[1]} \star \Phi_{[1]}^{\mathrm{T}} \\ &= \begin{pmatrix} \Phi_{(1,0,0)} \\ \Phi_{(0,1,0)} \\ \Phi_{(0,0,1)} \end{pmatrix} \begin{pmatrix} \Phi_{(1,0,0)} & \Phi_{(0,1,0)} & \Phi_{(0,0,1)} \end{pmatrix} \\ &= \begin{pmatrix} \Phi_{(2,0,0)} & \Phi_{(1,1,0)} & \Phi_{(1,0,1)} \\ \Phi_{(1,1,0)} & \Phi_{(0,2,0)} & \Phi_{(0,1,1)} \\ \Phi_{(1,0,1)} & \Phi_{(0,1,1)} & \Phi_{(0,0,2)} \end{pmatrix}. \end{aligned} \tag{2.41}$$

2.3.3 Transformation

Using the form representation matrix $\Phi_{[j,k]}$, Taubin and Cooper [TC92] show that under a non-singular coordinate transformation A, the transformed coefficient matrix is given by

$$\Phi'_{[j,k]} = A_{[j]}^{-\mathrm{T}} \Phi_{[j,k]} A_{[k]}^{-1}, \tag{2.42}$$

where $A_{[j]}$ is a non-singular $h_j \times h_j$ matrix and $A_{[k]}$ is a non-singular $h_k \times h_k$ matrix. If the matrix A is orthogonal

$$\Phi'_{[j,k]} = A_{[j]} \Phi_{[j,k]} A_{[k]}^{\mathrm{T}}, \tag{2.43}$$

By using homogeneous coordinates, an IP of n variables in Euclidean space can be described in projective space by a corresponding homogeneous IP of $n+1$ variables. To convert a ternary (*i.e.* 3D) IP of degree d

$$\Psi_{3D}^d(x,y,z) = \sum_{0 \le i,j,k; i+j+k \le d} a_{ijk} x^i y^j z^k \tag{2.44}$$

into its homogeneous representation, a new component $t = 1$ is added to the 3D IP as:

$$\Psi_{4D}^d(x,y,z,t) = \sum_{0 \le i,j,k,l; i+j+k+l=d} a_{ijk} x^i y^j z^k t^l. \tag{2.45}$$

A homogeneous polynomial corresponding to a polynomial of degree d in n variables is a form of degree d in $n+1$ variables. Similar to the coefficient representation matrix $\Phi_{[j,k]}$ for a form, we can make the coefficient representation matrix $F_{[j,k]}$ for a homogeneous polynomial. As an example, for a 2D polynomial of degree 4, the homogeneous form is a 3D polynomial of degree

4 and we can setup the coefficient representation matrix as:

$$\begin{aligned} F_{[2,2]} &= F_{[2]} \star F_{[2]}^{\mathrm{T}} \\ &= \begin{pmatrix} \frac{1}{2}F_{(4,0,0)} & \frac{1}{\sqrt{2}}F_{(3,1,0)} & \frac{1}{\sqrt{2}}F_{(3,0,1)} & \frac{1}{2}F_{(2,2,0)} & \frac{1}{\sqrt{2}}F_{(2,1,1)} & \frac{1}{2}F_{(2,0,2)} \\ \frac{1}{\sqrt{2}}F_{(3,1,0)} & F_{(2,2,0)} & F_{(2,1,1)} & \frac{1}{\sqrt{2}}F_{(1,3,0)} & F_{(1,2,1)} & \frac{1}{\sqrt{2}}F_{(1,1,2)} \\ \frac{1}{\sqrt{2}}F_{(3,0,1)} & F_{(2,1,1)} & F_{(2,0,2)} & \frac{1}{\sqrt{2}}F_{(1,2,1)} & F_{(1,1,2)} & \frac{1}{\sqrt{2}}F_{(1,0,3)} \\ \frac{1}{2}F_{(2,2,0)} & \frac{1}{\sqrt{2}}F_{(1,3,0)} & \frac{1}{\sqrt{2}}F_{(1,2,1)} & \frac{1}{2}F_{(0,4,0)} & \frac{1}{\sqrt{2}}F_{(0,3,1)} & \frac{1}{2}F_{(0,2,2)} \\ \frac{1}{\sqrt{2}}F_{(2,1,1)} & F_{(1,2,1)} & F_{(1,1,2)} & \frac{1}{\sqrt{2}}F_{(0,3,1)} & F_{(0,2,2)} & \frac{1}{\sqrt{2}}F_{(0,1,3)} \\ \frac{1}{2}F_{(2,0,2)} & \frac{1}{\sqrt{2}}F_{(1,1,2)} & \frac{1}{\sqrt{2}}F_{(1,0,3)} & \frac{1}{2}F_{(0,2,2)} & \frac{1}{\sqrt{2}}F_{(0,1,3)} & \frac{1}{2}F_{(0,0,4)} \end{pmatrix}. \end{aligned} \tag{2.46}$$

2.3.4 Partial Differential

To calculate the partial differential of an n-degree polynomial, $f_n = \mathbf{m}_n^{\mathrm{T}}\mathbf{a}$ respect to one variable x_i, it can be also in a simple symbolic manner as follow

$$\frac{\partial f}{\partial x_i} = \mathbf{m}_{n-1}^{\mathrm{T}}\mathbf{a}_{x_i}, \tag{2.47}$$

where $\mathbf{m}_{n-1}$ is the monomial vector of degree $n-1$ (see Eq. (2.1) and (2.2)) and $\mathbf{a}_{x_i}$ is viewed as the partial differential of $\mathbf{a}$ to x_i. Note since $\mathbf{m}_{n-1}$ is the monomial vector of degree $n-1$, it is also a part of the monomial vector of degree n, $\mathbf{m}_n$. Then let $a_{(\alpha_1,\ldots,\alpha_i,\ldots,\alpha_n)}$ be an element of coefficient vector $\mathbf{a}$ corresponding to the monomial $x_1^{\alpha_1}\cdots x_i^{\alpha_i}\cdots x_m^{\alpha_m}$. Then the element $a'_{(\alpha_1,\ldots,\alpha_i,\ldots,\alpha_n)}$ in the partial differential coefficient vector $\mathbf{a}_{x_i}$ can be symbolically calculated as

$$\text{if}\begin{cases} \alpha_i > 0 & a'_{(\alpha_1,\ldots,\alpha_i-1,\ldots,\alpha_n)} \longleftarrow \alpha_i a_{(\alpha_1,\ldots,\alpha_i,\ldots,\alpha_n)} \\ \text{otherwise} & a'_{(\alpha_1,\ldots,\alpha_i-1,\ldots,\alpha_n)} \longleftarrow 0 \end{cases}. \tag{2.48}$$

Therefore, in the elements of vector $\mathbf{a}_{x_i}$ corresponding to the nth form are decreased to correspond to the $(n-1)$-th form.

For example, for a 3D IP f^{3D}, the partial differential $\frac{\partial f^{3D}}{\partial y}$ can be worked out with $\frac{\partial f^{3D}}{\partial y} = \mathbf{m}_{n-1}^{\mathrm{T}}\mathbf{a}_y$, where the element $a'_{i,j,k}$ of $\mathbf{a}_y$ can be calculated as

$$\text{if}\begin{cases} j > 0 & a'_{(i,j-1,k)} \longleftarrow j a_{(i,j,k)} \\ \text{otherwise} & a'_{(i,j-1,k)} \longleftarrow 0 \end{cases}. \tag{2.49}$$

The the element $a'_{i,j,k}$ will be arranged in a coefficient vector of degree $(n-1)$, corresponding to the monomial $x^i y^j z^k$ in $\mathbf{m}_{n-1}$.

Therefore we can see that the computation for IP gradients $\frac{\partial f}{\partial \mathbf{x}} = (\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n})^{\mathrm{T}}$ are also easy to be obtained by the above method.

2.3.5 Intrinsic Geometry of IP

Taubin and Cooper [TC92] show that the Euclidean center **c** of an implicit polynomial $f_d = 0$ of degree d can be computed as

$$\mathbf{c} = -F^{\dagger}_{[d-1,1]} F_{[d-1]} \tag{2.50}$$

where $F_{[d-1,1]}$ is a symmetric matrix constructed from the coefficients of the highest degree form, $F_{[d-1]}$ is a vector constructed from the coefficients of the form of the second highest degree and † implies taking the pseudoinverse which is given as

$$F^{\dagger}_{[d-1,1]} = [F_{[1,d-1]} F_{[d-1,1]}]^{-1} F_{[1,d-1]} \tag{2.51}$$

provided that $F_{[1,d-1]} F_{[d-1,1]}$ is non-singular. For a 2D polynomial of degree 4, (2.50) expands to

$$\begin{aligned} \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} &= -\begin{pmatrix} \frac{1}{\sqrt{6}} F_{[4,0]} & \frac{1}{\sqrt{6}} F_{[3,1]} \\ \frac{1}{\sqrt{2}} F_{[3,1]} & \frac{1}{\sqrt{2}} F_{[2,2]} \\ \frac{1}{\sqrt{2}} F_{[2,2]} & \frac{1}{\sqrt{2}} F_{[1,3]} \\ \frac{1}{\sqrt{6}} F_{[1,3]} & \frac{1}{\sqrt{6}} F_{[0,4]} \end{pmatrix}^{\dagger} \begin{pmatrix} F_{[3,0]} \\ F_{[2,1]} \\ F_{[1,2]} \\ F_{[0,3]} \end{pmatrix} \\ &= -\begin{pmatrix} \frac{4!0!}{\sqrt{6}} a_{40} & \frac{3!1!}{\sqrt{6}} a_{31} \\ \frac{3!1!}{\sqrt{2}} a_{31} & \frac{2!2!}{\sqrt{2}} a_{22} \\ \frac{2!2!}{\sqrt{2}} a_{22} & \frac{1!3!}{\sqrt{2}} a_{13} \\ \frac{1!3!}{\sqrt{6}} a_{13} & \frac{0!4!}{\sqrt{6}} a_{04} \end{pmatrix}^{\dagger} \begin{pmatrix} 3!0! a_{30} \\ 2!1! a_{21} \\ 1!2! a_{12} \\ 0!3! a_{03} \end{pmatrix}. \end{aligned} \tag{2.52}$$

Moreover, the center is a covariant. That is, if the coordinate axes transform by $\mathbf{x}' = A\mathbf{x} + \mathbf{b}$ then the center also transforms by $\mathbf{c}' = A\mathbf{c} + \mathbf{b}$.

Lemma 2.3 *Let $f(\mathbf{x})$ be a polynomial of degree d, $\mathbf{x}' = A\mathbf{x} + \mathbf{b}$ a Euclidean coordinate transformation, $f'(\mathbf{x}') = f(A^{\mathrm{T}}(\mathbf{x}' - \mathbf{b}))$, $\mathbf{c} = -F^{\dagger}_{[d-1,1]} F_{[d-1]}$ and $\mathbf{c}' = -F'_{[d-1,1]} \dagger F'_{[d-1]}$. Then $\mathbf{c}' = A\mathbf{c} + \mathbf{b}$.*

Computation of principal axes

Continuing from the approach for Euclidean center computation, Taubin and Cooper [TC92] show that for an nD polynomial f_d of degree d, the n eigenvectors of the symmetric matrix

$$F^{\mathrm{T}}_{[d-1,1]} F_{[d-1,1]} \tag{2.53}$$

form an orthogonal coordinate system that defines the intrinsic orientation of the implicit polynomial $f_d = 0$ (i.e. $Z(fd)$). Let us denote by V_f the $n \times n$ orthogonal matrix formed by placing the n eigenvectors column-wise. The geometric interpretation of the orthogonal matrix V_f is that it transforms the standard Cartesian axes to the intrinsic axes of the polynomial f_d. Alternatively, V_f^{-1} transforms the polynomial's intrinsic axes into alignment with the standard

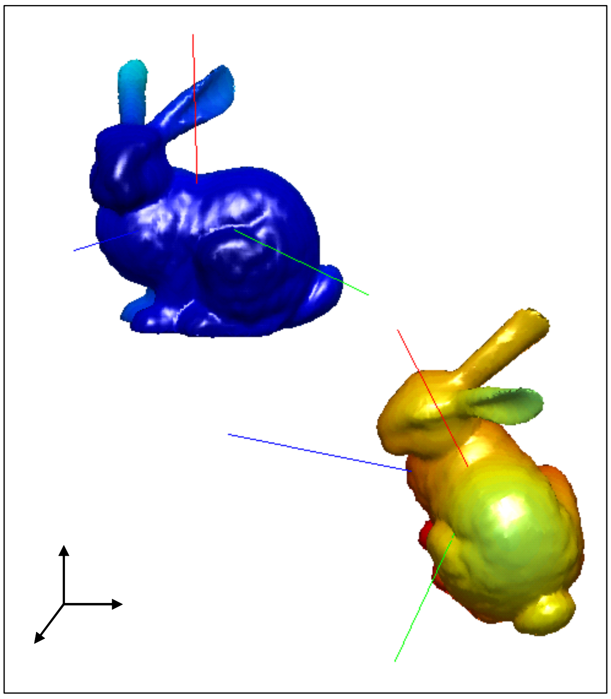


Figure 2.8: Intrinsic axes extracted from coefficients of a 8th-degree algebraic surface representing a bunny object and a rotated one.

Cartesian axes. For a 2D polynomial of degree 4, the symmetric matrix whose 2 eigenvectors define the implicit polynomial's orientation is given by

$$\begin{aligned} F_{[d-1,1]}^{\mathrm{T}} F_{[d-1,1]} &= F_{[3,1]}^{\mathrm{T}} F_{[3,1]} \\ &= \begin{pmatrix} \frac{F_{(4,0)}^2}{6} + \frac{F_{(3,1)}^2}{2} + \frac{F_{(2,2)}^2}{2} + \frac{F_{(1,3)}^2}{6} & A \\ A & \frac{F_{(3,1)}^2}{6} + \frac{F_{(2,2)}^2}{2} + \frac{F_{(1,3)}^2}{2} + \frac{F_{(0,4)}^2}{6} \end{pmatrix} \end{aligned} \tag{2.54}$$

where

$$A = \frac{F_{(4,0)}F_{(3,1)}}{6} + \frac{F_{(3,1)}F_{(2,2)}}{2} + \frac{F_{(2,2)}F_{(1,3)}}{2} + \frac{F_{(1,3)}F_{(0,4)}}{6} \tag{2.55}$$

2.3.6 Algebraic Invariants

The algebraic invariants of IP have been studied for about two decades, we review the recent literatures in Appendix A. However, here we would like to introduce a extended set of invariants based on Taubin and Cooper' theory [TC92].

As shown in last subsection, since the intrinsic axes can be extracted from the leading form of an IP, then the corresponding eigenvalues projected onto the intrinsic axes can be also viewed as a set of invariants. Actually, it does not only limit in the intrinsic axes extraction (extracted from a 3×3 covariant matrix), but also we can extend the result to any dimension. That is, for an nD polynomial f_d of degree d, the n eigenvalues of the symmetric matrix

$$F^{\mathrm{T}}_{[d-i,i]} F_{[d-i,i]} \tag{2.56}$$

can be also considered as the Euclidean invariants of the implicit polynomial $f_d = 0$ (i.e. $Z(fd)$). For example, for a 2D polynomial of degree 4, the symmetric covariant matrix whose eigenvalues define the implicit polynomial's Euclidean invariants is given by

$$F^{\mathrm{T}}_{[d-2,2]} F_{[d-2,2]} = F^{\mathrm{T}}_{[2,2]} F_{[2,2]}. \tag{2.57}$$

2.4 Comparison to Other Shape Representations

After describing the properties of IP and according to the shape representation overview given in Section 1.2, let us give a comparison on IP and other function based representation as follows.

- IP vs. mesh representation

  IP surfaces are harder than explicit polygonal meshes to deform and visualize in real time since that requires the zero set of an implicit function in 3D space. However IP surfaces are smoother and more compact that makes them operated in fast process. Furthermore, it is also non-trivial to determine if an arbitrary point is inside, outside or on the surface.

- IP vs. NURBS or RaG

  As compared to NURBS or RaG surfaces, IP lacks local surface control which makes it less suitable for modeling. However, IP does not require parameterization, and it is easier to fit, easier to blend, and does not suffer from topology problems. It is well-suited for simulating physically based processes and for modeling smooth objects. It can be collided and deformed, with fusions and separations handled automatically [YT99].

- IP vs. Fourier representation

  Implicit polynomials do not suffer from any of the problems listed above: they are directly applicable to non-star shapes, open curves, unordered data sets and are robust to noisy data sets and inhomogeneously spaced data points. The main advantage of Fourier Shape Descriptors over algebraic curves has been their better stability when fit to data. The significance of this is that, in cases where algebraic curves can not be fit to data with high repeatability, good recognition based on algebraic invariants is not possible.

- IP vs. RBF

  With better representation for complex shapes than IP, RBF representation however for representing a large-scale dense data set may have too many parameters as to lead to heavy computation for fitting, gradient computation and visualization, and for different resolution (density) may have non-uniqueness of parameters.

In addition, IP representation has proved very useful in considerable prior literatures, such as registration [FMZ*91, KC94, Tau91, TC00a, TCC98, BLC00, TTC00, Red00], recognition [Ker94, TC00a, SCK96, OEB03, Mar05, JXL*07, UA99, UA00, WU98], smoothing and denoising [TTC00, SU05], 3D reconstruction [KTFC01], image compression [HBZM00], and image boundary estimation [TC00b].

IPs are powerful for shape recognition and single-computation (non-iterative) pose estimation because of their fast fitting and functions of their coefficients which are invariant under Euclidean or Affine transformations. These properties and invariant functions of algebraic curves have been studied extensively. Coefficients of polynomials representing algebraic entities encode global shape information about the represented entity [TC92]. This information includes

- Algebraic invariants that help in classifying and recognising different objects [Ker94, TC00a, SCK96, OEB03, Mar05, JXL*07, UA99, UA00, WU98].
- Intrinsic surface geometry that helps in registration [FMZ*91, KC94, Tau91, TC00a, TCC98, BLC00, TTC00, Red00].

Algebraic implicit surfaces are particularly appealing in the context of coarse (initial) registration because of the second property. Polynomial coeficients of an algebraic entity can be used to estimate its center and orientation vectors which together make up the entity's intrinsic reference frame. This is explained in Section 2.3. Given such intrinsic information, the problem of registering two objects reduces to alignment of their intrinsic reference frames. This obviates the need for iteratively finding correspondences needed by explicit registration approaches such as ICP [BM92, ONKI05]. However, the property of algebraic surfaces that is useful in the projective registration domain is that image outlines of algebraic surfaces completely determine their projective geometry [FMZ*91]. This implies that algebraic curve equations representing the image outlines of projections of algebraic surfaces can be computed given the algebraic surface equation and the camera projection matrix. Given an algebraic surface, elimination theory can be used to compute the surface's tangent cone [PK92] as viewed from a certain point.

On all accounts, in contrast to other representations such as B-spline, Non-Uniform Rational B-Splines (NURBS) [Pie91], Rational Gaussian [Gos93], and radial basis function (RBF) [TO99b], IPs are superior in such computer-vision-attractive areas as fast fitting, few parameters, algebraic/geometric invariants, and robustness against noise and occlusion.

Chapter 3

3D Surface Segmentation and Representation Using IP

In this chapter, we present an 3D surface segmentation of using IP method motivated by the fact that recently 3D models obtained by scanning real objects with range finders are widely applied in computer vision. For example, the progress of the modeling techniques enables us to obtain 3D models of large objects whose dimension is over 100 meters with precision within a few centimeters [IHN*04]. Thus the model data is characterized with large, complex, and of high quality. However, on the contrary, this expansion might cause problems in data processing, *e.g.*, for 3D reconstruction from the 3D scans, it often requires a process that register the scans of different views into a same coordinate, and in other fields, such as rendering data in fine and smooth detail, 3D matching for retrieval objects, or sharing the huge amount of data on the Internet. Therefore there is a need to represent the obtained 3D models by algebraic surfaces for such purposes as obtaining 3D geometric feathers, compression, multi-resolution, and noise elimination.

The representation with algebraic surfaces defined by IPs sounds a good choice, since it can offer the advantages that IP representation is capable of encoding geometric properties easily with smoothness, few parameters, algebraic/geometric invariants, and robustness to noise and missing data. However, unfortunately, there are two nontrivial shortcomings frustrate the IP's application on the efficiency in practical fields. One is the poor representation for complex object, and the other one is that IP only can encode the global feathers with the so-called algebraic representation but is invalid for local feather.

The representability is poor due to 1) the characteristic of IP, namely the disability of fine representation for the small perturbation of data set, no matter how high the degree is (how many parameters are used); 2) the numerical instability arise in the fitting method that might lead the obtained IP representation to be not interpreted (to address this problem it will be discussed in Chapter 4). For the second issue, IP is easy to encode the global features for objects but is unavailable for capturing the local features. Note, this is not a problem in the cases that there are individual objects need to be classified, but for the partially overlapping object, *e.g.*, the registration for the overlapping objects, it is useless.

In this chapter, we propose a solution for solving the above problems is 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. Advantages of our method are that it is capable of 1) encoding the local feathers onto the surface with appropriate segmentation, because each segment is represented with an IP; 2) achieving stable fitting since we avoid using high-degree IPs; and 3) decomposing heavy computation from one high-degree IP fitting into light computations from multiple low-degree IPs. The segmentation result opens up a new vista for the application of a non-iterative registration for range images. Note this method can be applied to general 3D data formats such as polygonal mesh models.

This chapter is organized as follows: Section 3.1 gives a review of prior works on 3D segmentation; Section 3.2 provides a new segmentation method employed IP techniques; Section 3.3 reports experimental results; Section 3.4 provides the practical application on non-iterative registration for range image, followed by summary in section 3.5. Finally, we list the related algorithms in section 3.6.

3.1 Prior Methods for Surface Segmentation

For 3D surface segmentation, there mainly exist two categories of the methods. One is a geometrical method, and the other one is a clustering method. The former segments the surface according to the local geometrical properties of 3D surface, such as Gaussian and mean curvatures. For triangulated meshes, the Gaussian and mean curvature at a vertex is approximated from its adjacent triangles [SK03, YGZS05], or from the locally approximated functions discussed in [KWTM03, dAJ04, JBS06]. The latter segments them by minimizing a certain energy function with topological information and some metric of distance. The minimization is done by some clustering technique, such as the graph cut method.

A deficiency of the geometrical methods is that they are generally vulnerable to data noises and occlusions, since the geometric property associated with each point has to be decided by a differential operation. On the other hand, clustering methods bias segmentation according to topological information. The result might be useful to object partition, such as the fact that a body can be divided into a head, two arms, two legs etc., but it cannot guarantee that each segment is suitable for a low-degree IP. For more detail surveys, we refer to [Pet02, Sha04].

Some other methods are proposed in [OBA*03, OBA05, ABCO*03], where they used low-degree implicit functions for modeling voxel-wise or sphere-wise segments. However, since these methods are mainly for rendering, their segmentation results are not Euclidean invariant and thus not suitable for solving general problems such as object recognition and registration (for example [TC92]).

3.2 Methodology of Segmentation Using IP

As described in last section, the prior methods are difficult to satisfy our purpose that we need an appropriate segmentation method to achieve the resulted segments can be encoded with an IP. Thus we attempt to propose a new method for automatically segmenting 3D shapes with all the segments having a one-to-one correspondence to the low-degree IPs.

Our segmentation method uses a cut-and-merge strategy to simplify the surface into regions satisfying three requirements as follows:

1. Every region can be well fitted by a IP with desired accuracy.
2. The number of the regions is as few as possible;
3. Each boundary is cut as clearly as possible on object edges.

If these three requirements are satisfied, we think it is an appropriate segmentation for the target object surface, and therefore the segmented regions can have a one-to-one correspondence to IPs. Namely, the geometric properties of each region can be well encoded with an IP. A clear boundary of each region also can be extracted easily and automatically.

To do this, first we define the degree of the fitness, that is, the similarity metric between an IP and a point data set. We use this similarity metric to evaluate how well the fitting is performed. Next, we describe our cut-and-merge strategy in detail, which consists of the three following procedures:

- Unfit cutting procedure:

  Iteratively cutting the regions until no region unfit for IPs exists.

- High-curved cutting procedure:

  Iteratively cutting the regions until neither high-curved nor distorted region exists.

- Merging procedure:

  Iteratively merging the regions which are acceptable for the same IP.

3.2.1 Similarity Metric

Before we give the details of our cut-and-merge segmentation strategy, let us define two functions to measure the similarity between the IPs and the regions. As shown in Fig. 3.1, a set of similarity functions can be written as follows:

$$D_{dist} = \frac{1}{N}\sum_{i=1}^{N} e_i \tag{3.1}$$

$$D_{smooth} = \frac{1}{N}\sum_{i=1}^{N} (\mathbf{N}_i \cdot \mathbf{n}_i), \tag{3.2}$$

where

$$e_i = \frac{\mid f(\mathbf{x}_i) \mid}{\| \triangledown f(\mathbf{x}_i)\|}, \quad x_i \in \{data\ set\} \tag{3.3}$$

$$\mathbf{n}_i = \frac{\triangledown f(\mathbf{x}_i)}{\| \triangledown f(\mathbf{x}_i)\|}, \quad x_i \in \{data\ set\}. \tag{3.4}$$

N is the number of vertices of sampled 3D object, and $\mathbf{N}_i$ is the normal vector associated with the point. Although e_i in (3.3) 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 $f(\mathbf{x})$ [TTC00]. $\mathbf{n}_i$ is the normalized gradient vector of f at vertex $\mathbf{x}_i$. Thus it represents the normal vector of $\mathbf{x}_i$ on the zero set.

D_{dist} and D_{smooth} can be considered as two measurements of distance and smoothness between a region and the zero set of an IP. From the definition of the similarity measures, the smaller D_{dist} and the larger D_{smooth} are, the more the region is similar to IP. The perfect case is $D_{dist} = 0 \wedge D_{smooth} = 1$. Accordingly two thresholds $T_1 (T_1 > 0)$ and $T_2 (T_2 < 1)$ are set into the constraint of

$$(D_{dist} < T_1) \ \wedge \ (D_{smooth} > T_2). \tag{3.5}$$

If these two constraints are both satisfied, we then say the current region is IP-representable (*i.e.* it can be described by an IP). Note, we use the term *IP-representable* for this case and the term *IP-unrepresentable* otherwise.

Now let us give a discussion on why we use both the constraints in the stopping criterion (3.5). Can we use either of them, $(D_{dist} < T_1)$ or $(D_{smooth} > T_2)$? Fig. 3.2 shows the two error cases might occur if only use either of them. Fig. 3.2(a) shows the case that, although the data set points are on the IP, that means D_{dist} is very small, the IP might have small fluctuation between the points. In this case, D_{smooth} is not still satisfy the accuracy. On the other hand, Fig. 3.2(b) shows another case that only if the smooth distance D_{smooth} is satisfied to the accuracy, there might still exist the distance between data set and IP. Therefore the combination of them give the more robust distance metric.

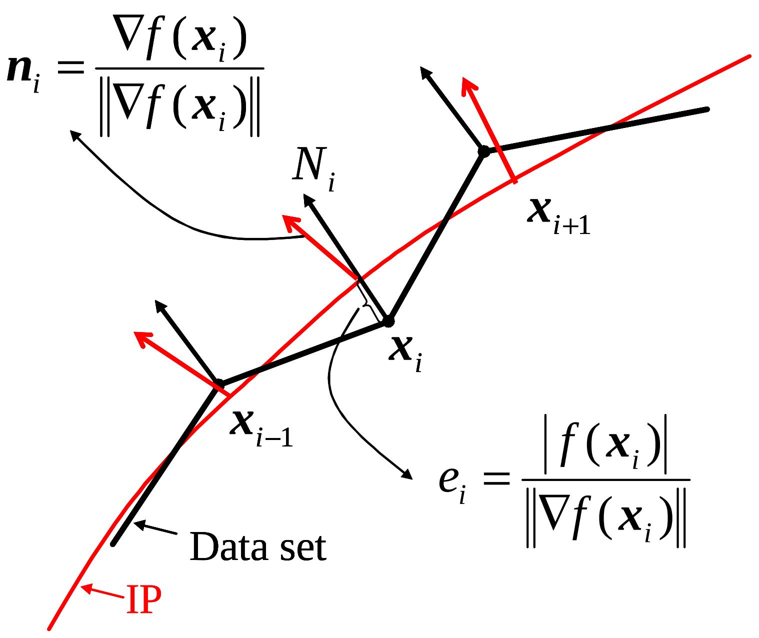

$n_i = \frac{\nabla f(x_i)}{\|\nabla f(x_i)\|}$
N_i
x_{i+1}
x_i
x_{i-1}
$e_i = \frac{|f(x_i)|}{\|\nabla f(x_i)\|}$
Data set
IP

Figure 3.1: 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 difference of normal directions associated with data point and IP zero set.

3.2.2 Unfit Cutting Procedure

The first procedure is named *Unfit cutting procedure*, and its task is to divide the IP-unrepresentable data set into IP-representable segments. First, we use the IP of a certain degree to measure a surface region with constraint (3.5). Second, 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*}) and Outer part ({*OuterRg*}) defined as:

$$\begin{aligned} \{InnerRg\} &:= \{\mathbf{x}_i | f(\mathbf{x}_i) \le 0\}, \\ \{OuterRg\} &:= \{\mathbf{x}_i | f(\mathbf{x}_i) > 0\}; \end{aligned} \tag{3.6}$$

Third, we repeat the above operation until there are no more IP-unrepresentable regions. We also summarize this operation in Algorithm 1 as described in Section 3.7.

Let us show a simple 2D example is shown in Fig. 3.3. In this case, the 1-degree IPs (lines) are used. First, a 1-degree IP is used to fit the whole data set. Second, by measuring the similarity, the data set is divided into two parts, an inner part shown with region I and an outer part shown with regions II and III. Then all of these regions are fitted again with new IPs. Since regions II and III can be fitted well by 1-degree IPs, they will be outputted in the next step. On the other hand, since region I cannot be well fitted by a 1-degree IP, it will be cut again. The

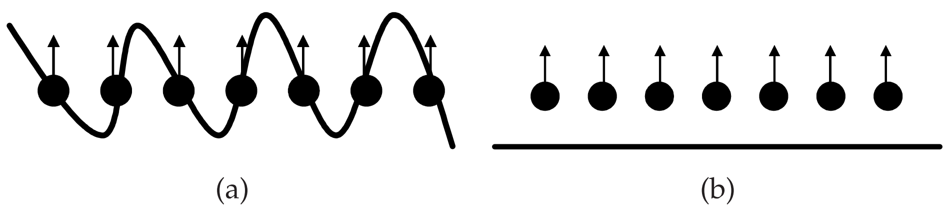


Figure 3.2: Two error cases might arise due to only use one of the constraints (black dots: data set points, black line: IP curve): (a) only use constraint ($D_{dist} < T_1$), (b) only use constraint ($D_{smooth} > T_2$).

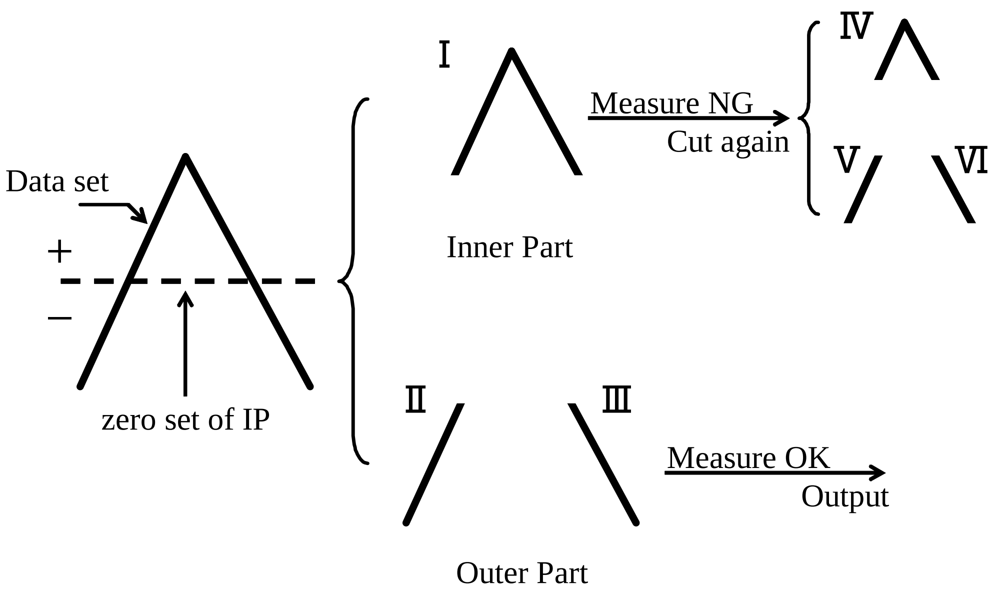


Figure 3.3: Unfit cutting procedure

procedure is done iteratively, until every region is acceptable to a 1-degree IP with constraint (3.5) of two certain thresholds.

3.2.3 High-curved Cutting Procedure

The task of the high-curved cutting procedure is to find the high-curved regions resulting from the unfit cutting procedure, and to divide them into low-curved regions. In the results from the unfit cutting procedure, there may be highly curved or distorted regions, e.g., Region IV in Fig. 3.3. But from the view of segmentation, the two edges of this angle should be divided into two parts, and each part belongs to a different IP. For this reason, we design the high-curved cutting procedure to find the curved regions and cut them again.

How to find the high-curved regions

To find the curved regions, we have to know the geometric characteristics of the surface first. IPs can provide a convenient way to find the surface curvatures quickly and robustly.

As defined in (3.4), the normal vector $\mathbf{n}_i$ at vertex $\mathbf{x}_i$ is represented by the first order partial derivative of f. If $\mathbf{g}_i = \nabla f(\mathbf{x}_i) = (\frac{\partial f(\mathbf{x}_i)}{\partial x}\ \frac{\partial f(\mathbf{x}_i)}{\partial y}\ \frac{\partial f(\mathbf{x}_i)}{\partial z})^T$, then $\mathbf{n}_i = (n_x\ n_y\ n_z)^T = \mathbf{g}_i/||\mathbf{g}_i||$. The second order derivatives in (3.7) contain information about the curvature of isosurfaces of the implicit function.

$$\nabla \mathbf{n}_i^T = \begin{pmatrix} \frac{\partial n_x}{\partial x} & \frac{\partial n_x}{\partial y} & \frac{\partial n_x}{\partial z} \\ \frac{\partial n_y}{\partial x} & \frac{\partial n_y}{\partial y} & \frac{\partial n_y}{\partial z} \\ \frac{\partial n_z}{\partial x} & \frac{\partial n_z}{\partial y} & \frac{\partial n_z}{\partial z} \end{pmatrix}. \tag{3.7}$$

It can be solved as follows (see the derivation in [KWTM03]):

$$\nabla \mathbf{n}_i^T = \frac{1}{||\mathbf{g}_i||}\mathrm{GH}, \tag{3.8}$$

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

$$\kappa_{max} = \max(|k_1|, |k_2|) \tag{3.9}$$

$$\kappa_{min} = \min(|k_1|, |k_2|). \tag{3.10}$$

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. As an example shown in Fig. 3.4 (a), the κ_{max}'s on the edge vertices of "cookie" object have high values and their κ_{min}'s have relatively low values; thus, the edge shown with white points is a ridge.

Cutting the curved region

After finding the high-curved regions, the rest of the task is to cut them into low-curved regions. Following the procedure discussed above, we can extract the points existing on the high-curved place by a function

$$[\kappa_{max}^j,\ \kappa_{min}^j] = c(\mathbf{x}_j, f_i), \tag{3.11}$$

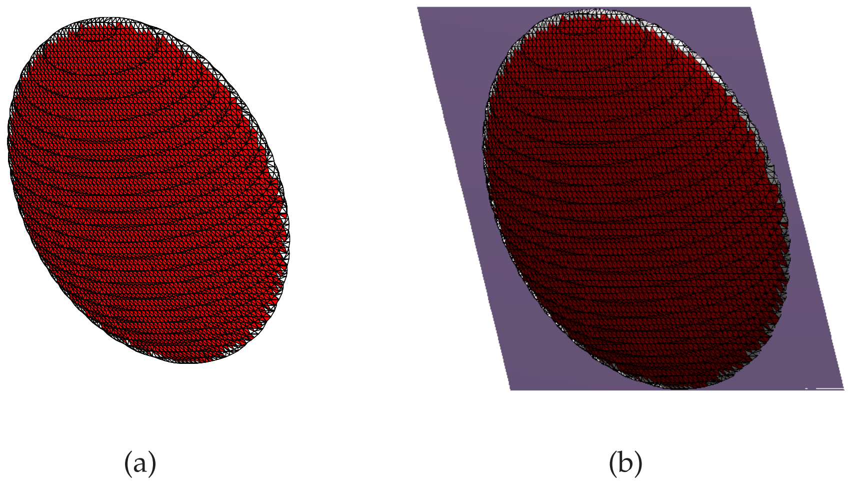


Figure 3.4: (a) White points show the high curvatures of a "cookie" object. (b) "cookie" object is cut along the edge by a 1-degree IP.

where vertex $\mathbf{x}_j \in R_i$; R_i is the found curved region, and f_i is the corresponding IP function of region R_i. The function c returns the maximum/minimum absolute curvatures κ^j_{max}, κ^j_{min} at $\mathbf{x}_j$. We extract the points if they satisfy the constraint:

$$\kappa^j_{max}/\kappa^j_{min} > K, \tag{3.12}$$

where K is a certain threshold for the ratio of maximum and minimum absolute curvatures. Thus we call the extracted points valley or ridge.

From the knowledge of differential geometry, we understand that for an n-degree IP region, there might be continuous valley/ridge curves that can be fitted with an $(n-1)$-degree IP.

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. A simple example is shown in Fig. 3.5. 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 with (3.6).

Here we choose the renormalization method [Kna94] for fitting the new IP. The reason for not choosing the 3L method is that it is not easy to obtain the information of connectivity between the extracted high-curved points. Thus the tangent at each point cannot be calculated for the generation of the other 2 layers. The Fig. 3.4 (b) shows the cutting result of "cookie". Along the ridge of the "cookie", the new IP shown as a plane in this case, is obtained. And according to the signs (3.6), the "cookie" is cut into two parts in the different sides of the plane.

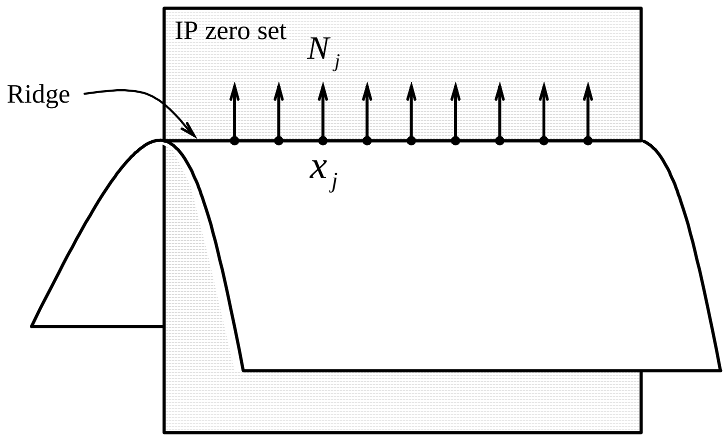


Figure 3.5: An example of cutting the curved region with a new IP

3.2.4 Merging Procedure

The task of the merging procedure is to merge the over-segmented regions to an integral region. The regions resulting from the unfit/high-curved cutting procedure may be over-segmented, as in the simplest case shown in Fig. 3.3. Although the resulting regions, Nos. II, III, V, and VI, were fitted well by 1-degree IPs respectively and outputted as independent regions, No. II and No. V should belong to the same region and served by the same IP. In the regions No. III and No. VI, the situation is the same. For solving this problem, we designed a procedure to merge the over-segmented regions.

This procedure makes use of the region-grow strategy [Pet02] 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 (3.5). The difference is that we use the seed's IP to measure the neighbors. Thus the measurement described in (3.3) and (3.4) should be replaced by:

$$e_i = \frac{|\, f_{seed}(\mathbf{x}_i) \,|}{\| \nabla f_{seed}(\mathbf{x}_i) \|}, \quad x_i \in \{R_{neighbor}\}, \tag{3.13}$$

$$\mathbf{n}_i = \frac{\nabla f_{seed}(\mathbf{x}_i)}{\| \nabla f_{seed}(\mathbf{x}_i) \|}, \quad x_i \in \{R_{neighbor}\}, \tag{3.14}$$

where f_{seed} is the IP function corresponding to the seed region, and $R_{neighbor}$ is a neighbor region of the seed region. The constraint in (3.5) 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. In short, the algorithm is described as Algorithm 2, :

3.3 Experimental results

In this section, we first introduce a simple experiment step by step, and then we report some other results.

3.3.1 A Simple Experiment

To clear the effectiveness of each step of our method, let us show an example of how to segment a noisy model shown in Fig. 3.6 (a) step by step.

Experiments with unfit cutting procedure

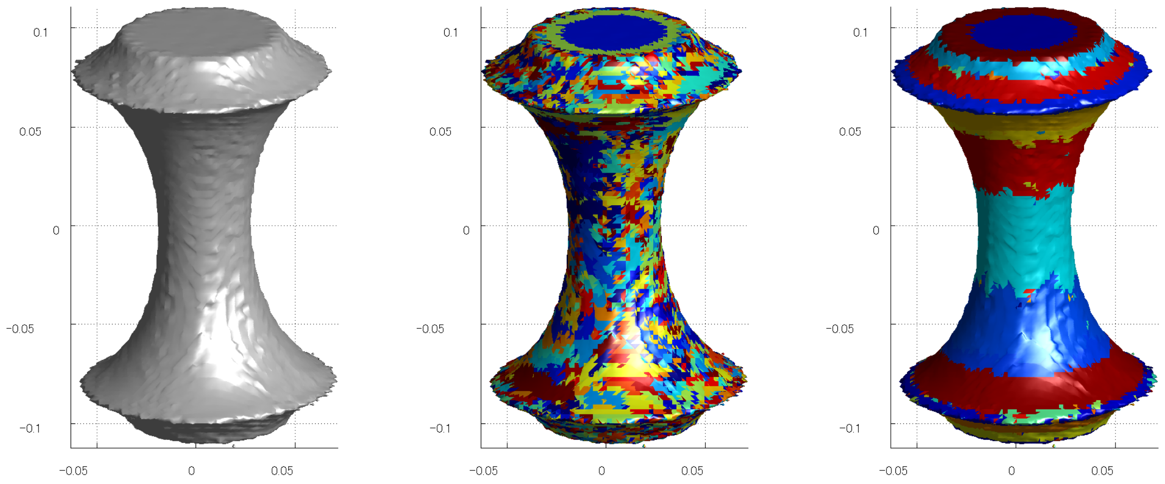
In the first step, unfit cutting procedure (Algorithm 1), the two thresholds T_1 and T_2 are essential for adjusting the accuracy and the number of segments. It is given that the more tolerant T_1 and T_2 are, the more dissimilar the IP is to a region. According to a certain demand, if high accuracy is required, T_1 should be set close to 0 and T_2 close to 1, and vice versa. But note that the immoderate setting may lead to over-segmentation or undesired result. In practice, for choice of the thresholds, should consider the factors according to the applications. For example, for such application as object coarse registration it might not need a fine segmentation result, on the other hand for rendering purpose it might need a fine segmentation result with high similarity to the original object. Also for an automatic system, setup for thresholds the ratio between a scale of object and degree of data noise should be considered.

Let us show an example of differing choices. The target object shown in Fig. 3.6 (a) is segmented with different thresholds. Fig. 3.6 (b) shows the result of 4365 segments, in the case of setting $T_1 = 0.0001$ and $T_2 = 0.9$, while Fig. 3.6 (c) shows the result of 267 segments, in the case of setting $T_1 = 0.001$ and $T_2 = 0.75$. Because of the coarse surface and the strict thresholds, the resulting segments in the former case are cut into small pieces to be suited for locally high accuracy. On the other hand, in the latter case, because of the tolerant thresholds, the resulting segments are larger and more noises are admissible into the regions to be suited for a relatively global accuracy.

Experiments with high-curved cutting procedure

Although the results in Fig. 3.6 (b) and (c) achieve IP fitting with desired accuracy, high-curved regions still exist, as shown in Fig. 3.7. Considering easy visualization of the result, we choose the result shown in Fig. 3.6 (c).

As shown in Fig. 3.7, first, the high-curved cutting procedure checks out the high-curved regions (shown in yellow) by setting $K = 10$ in constraint (3.12). Note the threshold setting refers to the curvature radius estimated from scale of object by experience.



(a) (b) (c)

Figure 3.6: The results of the unfit cutting procedure with different thresholds. (a) Original range data. (b) 4365 segments. (c) 267 segments.

Second, this procedure cuts this high-curved region into two low-curved regions so as to improve the result. By extracting the points with high curvature in the high-curved regions using the constraint (3.12), the IP for separation obtained as shown in the right part of Fig. 3.7. Finally this IP divides the high-curved region into two parts in its different sides respectively. Thus the 267 segments in Fig. 3.6 (c) become 269 segments, that is, two high-curved IPs are divided.

Experiments with merging procedure

The merging procedure is designed to merge similar IPs for further effective representation. Let us show this effectiveness by merging the result of the 269 segments derived from the high-curved cutting procedure.

The example shown in Fig. 3.8 (a) is the case where the thresholds are set the same as the cutting procedure with $T_1 = 0.001$ and $T_2 = 0.75$. Fig. 3.8 (b) shows the result obtained by setting $T_1 = 0.007$ and $T_2 = 0.7$. In these two cases, the 2-degree IPs are used to solve the fitting problem of the $3L$ method. We also give the merging case of 4-degree IPs shown in Fig. 3.8 (c) with settings of $T_1 = 0.001$ and $T_2 = 0.75$. Fig. 3.8 (c) obtains the same result as Fig. 3.8 (b). However, since 4-degree IPs can describe more complex surfaces than 2-degree IPs, merging by 4-degree IPs can be done with higher accuracy but is also more time-consuming than merging by 2-degree IPs. Fig. 3.8 (d) shows the black boundary points of the regions extracted from (b)

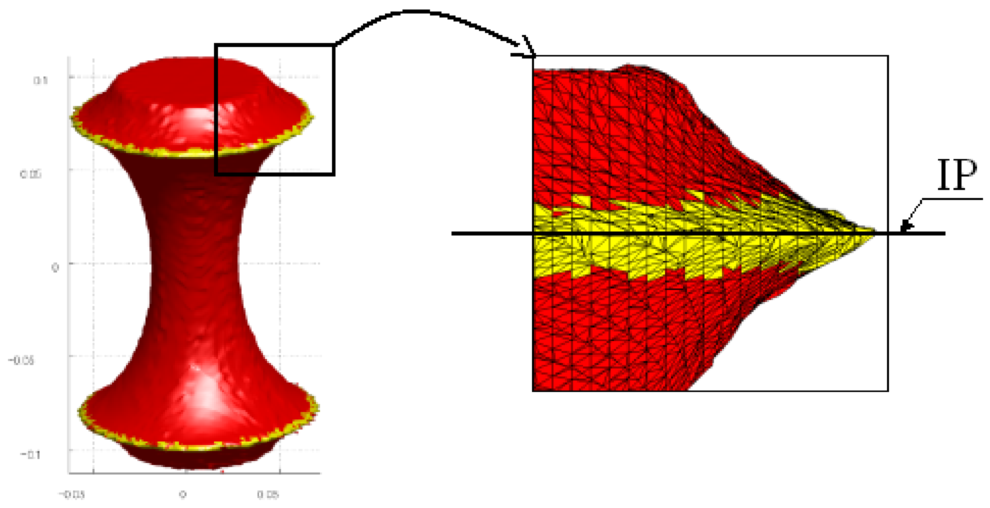


Figure 3.7: Cutting the high-curved regions (yellow regions)

or (c).

Having obtained the segmentation result, segmented regions and their one-to-one corresponding IPs, now let us show the segments with IP zero sets. To simplify the visualization of the IP segmentation result, we choose a simple way that projects the original mesh onto the zero set surfaces of IP segments. But we refer the interested reader to [LC97, dAJ04] for more details about mesh reconstruction for implicit functions.

Fig. 3.9 (a)(b)(c) show the IP surfaces (zero sets) from the segmentation results shown in Fig. 3.8 (a)(b)(c) respectively. Note although Fig. 3.8 (b) shows the same segmentation result as Fig. 3.8 (c), their IP surfaces are rather different. Obviously the 4-degree IPs represent more accurate surfaces than 2-degree IPs.

3.3.2 Discussion

Discussion on definition region

As shown in Fig. 3.8(d), since the boundary points can be easily extracted from each segment, then we can simply define the segment region with boundary by these discrete points with their topology information. Note, for the vision applications such as registration this definition does not need, but for the surface reconstruction from IP segments it is definitely necessary. Given a IP segment with the corresponding boundary points and its IP function, it can be reconstructed to a 3D mesh model. We refer to the interest reader to [LC97, dAJ04] for the relative research.

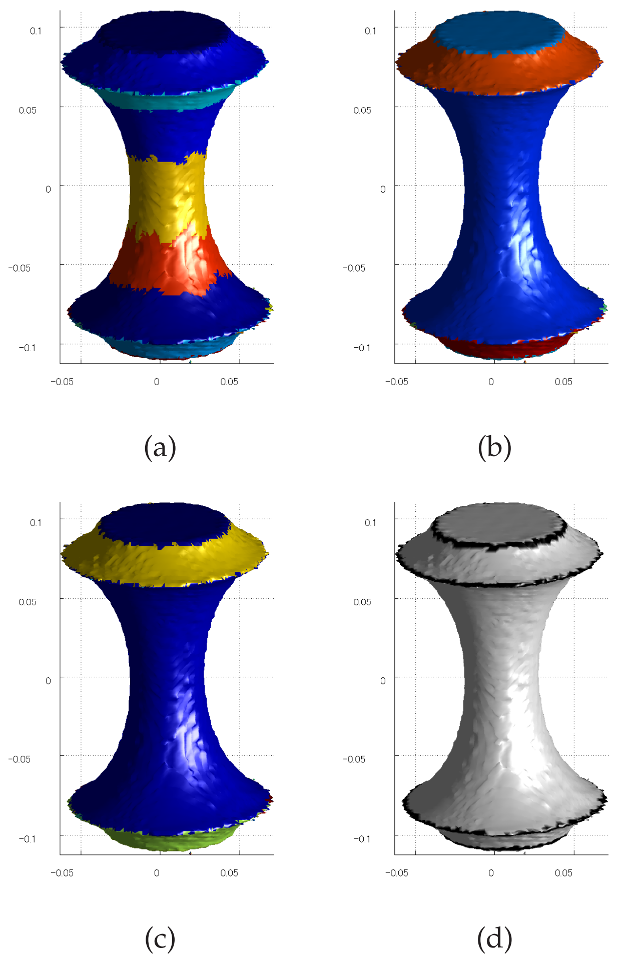


Figure 3.8: The results of the merging procedure with different thresholds. (a) 9 segments. (b) 5 segments with 2-degree IPs. (c) 5 segments with 4-degree IPs. (d) boundaries.

Discussion on parameter setting

In this work, the parameter setting somewhat depends on the scale and coarseness of the target object. In practice, as a simple way, we can consider the threshold T_1 according to a ratio to the scale of target object, and set more tolerant values up to T_1 and T_2 for a coarse model.

On the other hand, parameter setting also depends on the difference of application. For example, for the application of fast matching or coarse registration of range data, we do not need fine segmentation. But for the visualization purpose, that might need a fine segmentation to show the visual fidelity.

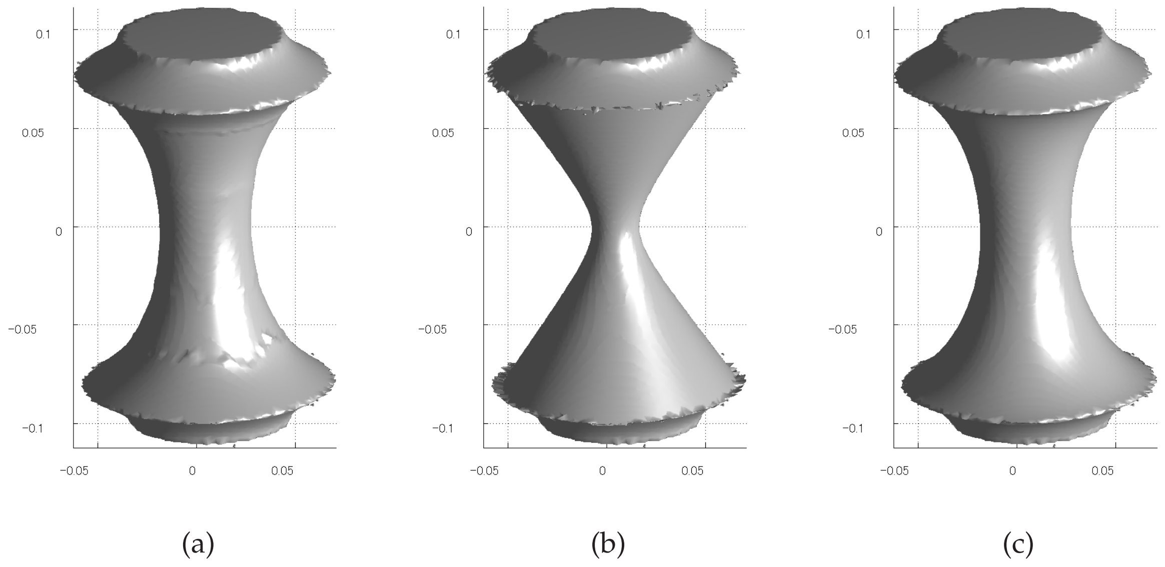


Figure 3.9: IP surfaces representation. (a) 9 segments with 2-degree IPs. (b) 5 segments with 2-degree IPs. (c) 5 segments with 4-degree IPs.

3.3.3 Other Results

In this subsection, let us show some other experimental results to prove the effectiveness of our method. First a simple example of segmented IP surface representation of a cube is shown in Fig. 3.10; the shape of the cube is contaminated by adding noises to one percent of its edge length. Six segmented IP surfaces for the cube are shown in Fig. 3.10 (b), and its corresponding segmentation result is shown in Fig. 3.10 (c). In this result, the cube object is divided into 6 planar segments, and each segment is represented by a 2-degree IP.

Other more complex examples are shown in Figs. 3.11 – 3.13. The original 3D 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 (3.5) and (3.12), 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. In each example, we use the 4-degree IPs to fit the segments and settings with different thresholds obtain the results with different accuracy.

3.4 Application: Non-iterative Registration for Range Images

Recently, 3D range image obtained by scanning large-scale target objects with range finders are widely applied in computer vision. Large objects, such as outdoor cultural heritage objects etc., usually have so complex geometrical shapes that sampling for them is still a difficult issue.

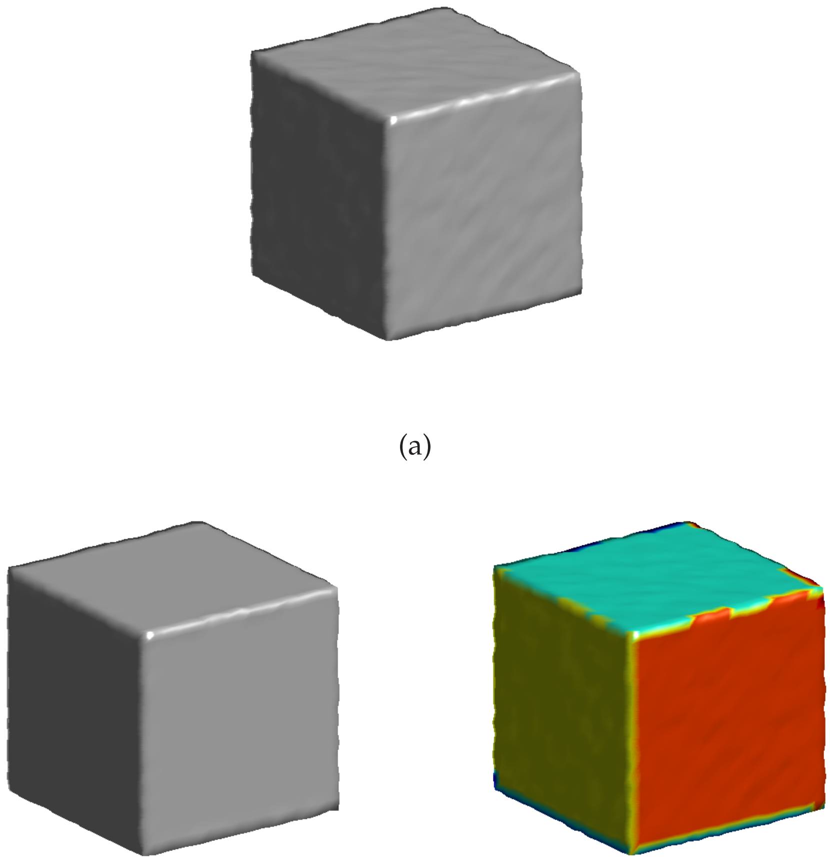


Figure 3.10: (a) Original range data. (b) 6 2-degree IP surfaces. (c) 6 segments with different colors referring to (b).

For example, in our Bayon Digital Archive Project [IHN*04] the Bayon temple located in the center of Angkor Thom in Cambodia has been scanned by various new-developed sensors. The temple is a very large ($160m \times 140m \times 45m$) and structure is complex, and therefore the sampled range images are thousands of pieces obtained in different views. And the finally they are desired to be integrated to a one-piece range data. To achieve that, there often need 3 steps:

- Initial alignment: coarsely aligning the range images into same coordinate.
- Fine alignment: accurately aligning the range images.
- Merging: integrating the aligned range images to one piece.

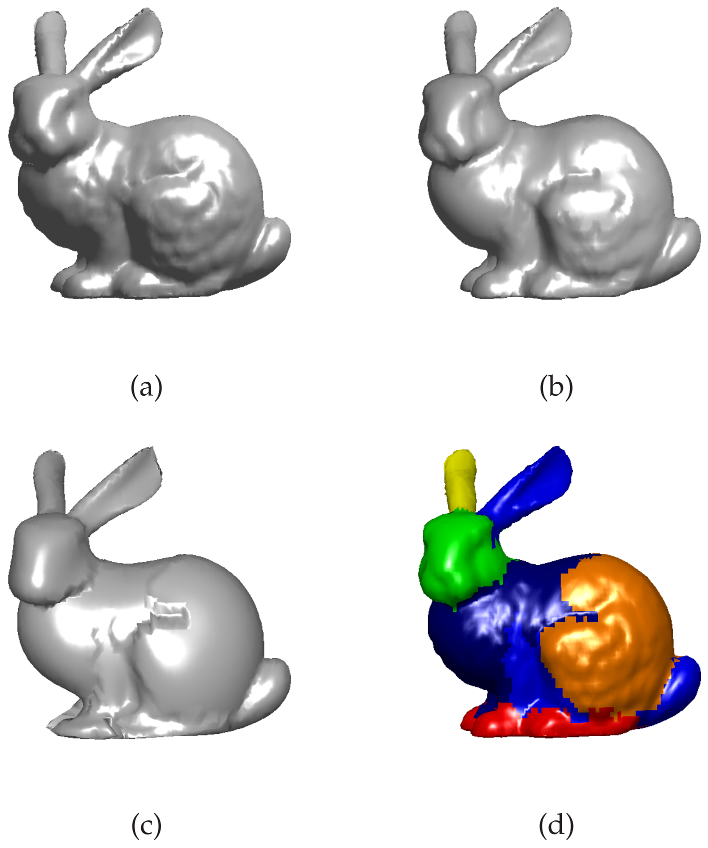
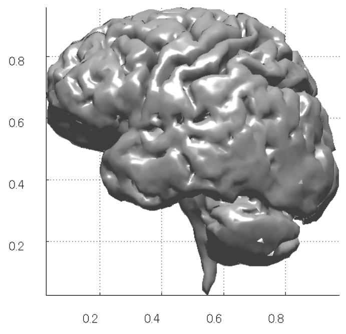


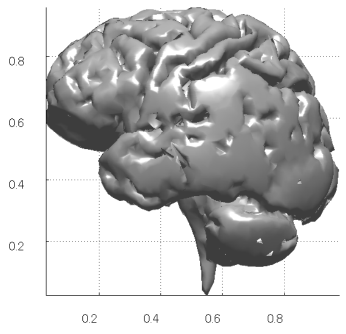
Figure 3.11: (a) Original range data. (b) 100 4-degree IP surfaces. (c) 8 4-degree IP surfaces. (d) 8 segments referring to (c).

In our study, although the problems on the second and the third steps: Fine alignment and Merging, have been solved in the automatic processes, the first step, Initial alignment, is still processed in a manual manner. That is, before handling the fine alignment, we have to manually move the range images into a same coordinate with roughly closed positions via a certain GUI tool, and then by taking these manually aligned position as the initial position, the fine registration is performed. However, in practice, because the large number of pieces of range images need to be registered first, usually the process is too time-consuming to be a non-trivial problem.

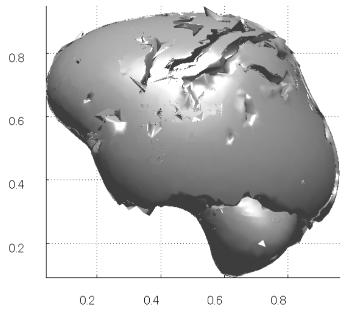
For range data registration, the ICP-based registration scheme is often developed such as [ONKI05]. But these methods require the strict initial positions and then by iteratively minimizing the distance energy function to achieve the high accurate registration. These method are very suitable for a fine registration but not suitable for the initial alignment since they severely depend on the initial positions of the range data. We refer reader to a survey paper [SMFF07].



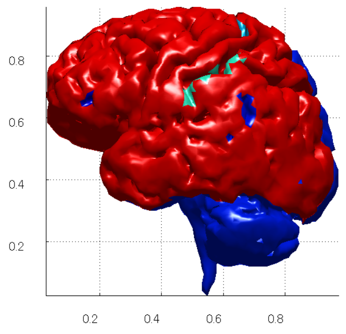
(a)



(b)



(c)



(d)

Figure 3.12: (a) Original range data. (b) 323 4-degree IP surfaces. (c) 19 4-degree IP surfaces. (d) 19 segments referring to (c).

Representing the range images with algebraic surfaces defined by IPs is competent, because of the good property of such as the easy intrinsic center and principal axes extraction (see Section 2.3.5) that enable the local matching and object registration to be in a non-iterative way.

Therefore, we extend the previous segmentation result of the method described in previous sections to a non-registration for rang data. Our non-iterative registration is developed by 2 steps. The first is to solve the matching problem, namely the problem on how to finding the IP-segment correspondences on two different range images. The second step is the calculation for registration using the obtained correspondences.

3.4.1 Finding IP-segment Correspondences

To find the correspondences for range image, many methods and various 3D features have been proposed. A recent survey on 3D recognition is given by [BaKS*05]. For an IP-segmented range image, since each segment has well encoded by robust features, IP coefficients, that makes

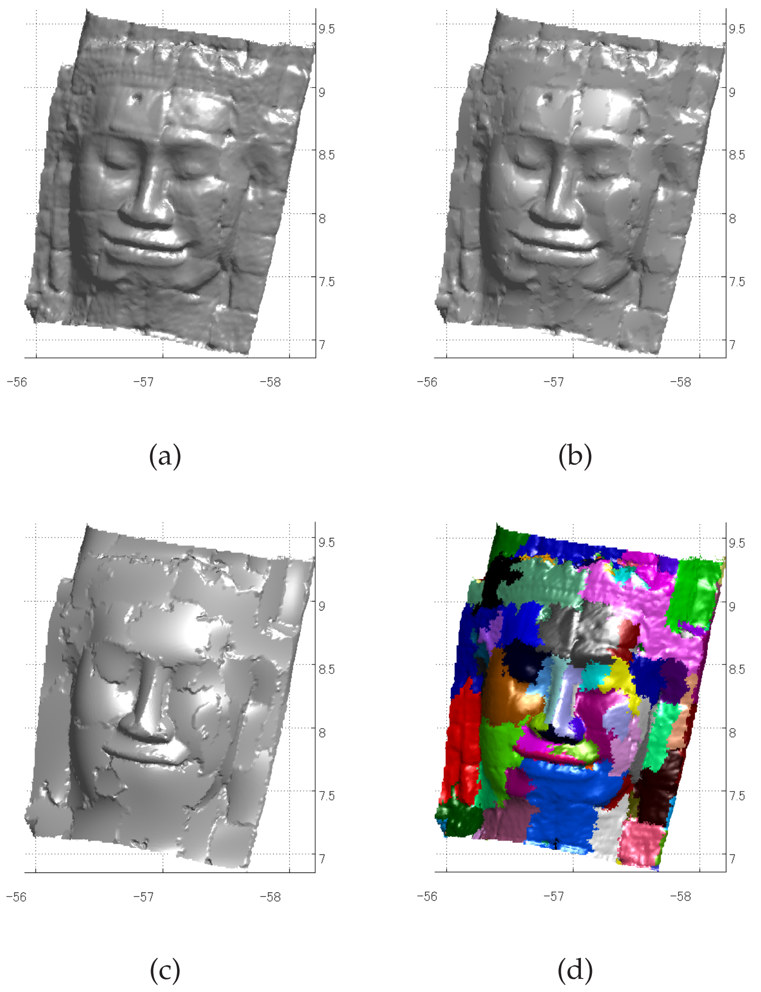


Figure 3.13: (a) Original range data. (b) 763 4-degree IP surfaces. (c) 113 4-degree IP surfaces. (d) 113 segments referring to (e).

the matching problem can be solved in a simple way. That is, because of the good property of IP having algebraic invariants, as introduced in Section 2.3, then we only need to find out the IP correspondences according to these invariants.

From Section 2.3, we have known that some Euclidean invariants can be extracted from the coefficients of the leading form of an IP by the symbolic calculations. The invariants enable us to match two IPs only by simple calculation as follow.

Calculate invariants for each segment

The invariants can be extracted from the covariant matrix derived leading form of an IP, that is, the eigenvalues of covariant matrix are Euclidean invariant. Therefore, for an n-degree

Table 3.1: The calculated invariants of each segment of the two segmented objects.

Segments	Object 1	Object 2
■	1.0e+004 * (0.3009, 0.8786, 2.1953)	1.0e+004 * (0.3009, 0.8786,2.1953)
🟫	1.0e+004 * (0.7127, 2.0522, 6.2591)	1.0e+004 * (0.7127, 2.0522, 6.2591)
🟫	1.0e+007 * (0.0157, 0.0513, 1.7007)	1.0e+007 * (0.0157, 0.0513, 1.7007)
🟫	1.0e+008 * (0.0213, 0.2080, 1.7999)	1.0e+008 * (0.0213, 0.2080, 1.7999)
🟨	1.0e+005 * (0.7364, 1.5098, 8.0651)	1.0e+005 * (0.7364, 1.5098, 8.0651)
▨	1.0e+008 * (0.0180, 0.2021, 2.1230)	1.0e+008 * (0.0180, 0.2021, 2.1230)
□	1.0e+008 * (0.0006, 0.0028, 2.1964)	1.0e+008 * (0.0006, 0.0028, 2.1964)
□	1.0e+009 * (0.0004, 0.0026, 2.7701)	1.0e+009 * (0.0004, 0.0026, 2.7701)

IP coefficient vector **a** whose normalized jth form are $\Phi_{[j]}$ (see Section 2.3). Then for the matrix

$$\Phi_{[j,k]} = \Phi_{[j]} \star \Phi_{[k]}, \tag{3.15}$$

whose covariant matrix will be

$$C = \Phi_{[j,k]}^{\mathrm{T}} \Phi_{[j,k]}. \tag{3.16}$$

Then the eigenvalues of C can be considered as the Euclidean invariants of the implicit polynomial of the corresponding segment. In our work, for convenience, we use the eigenvalues of symmetric covariant matrix of leading form of the IP to define the Euclidean invariants as follow

$$L = \Phi_{[n-1,1]}^{\mathrm{T}} \Phi_{[n-1,1]}. \tag{3.17}$$

Since it is a 3D IP, matrix L is a 3×3 matrix having 3 eigenvalues.

An example is shown in Fig. 3.14. First, we took the Euclidean transformation (rotation and translation) of the original object shown in Fig. 3.11 (a), and the transformed object is shown in Fig. 3.14 (a) marked as *Object 2*. Second, we segmented both objects with the same accurate thresholds as the example shown in Fig. 3.11 (d). Note, since our segmentation method is Euclidean invariant guaranteed by the fitting method, we obtained the same two segmented results: 1) both of them are segmented into 8 segments, and 2) both of their segments are one-to-one responded in different Euclidean coordiantes.

We calculated the Euclidean invariants described above shown in Tab. 3.1. It shows the important property: the segmented regions of the both are robustly one-to-one matched according to the invariants extracted from IP coefficients.

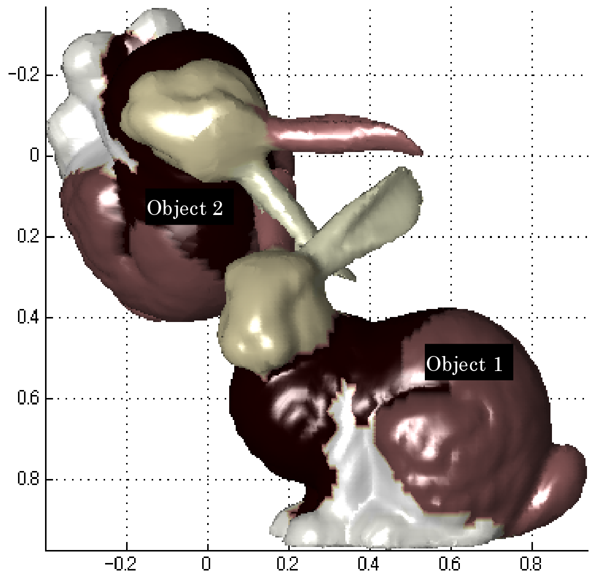


Figure 3.14: Matching two segmented objects: Object 1 is the original and Object 2 is Euclidean transformed.

3.4.2 Non-iterative Registration

Given the segment-wise correspondences, the next task is to register the range images according the correspondences. Taubin and Cooper gave a simple method for calculating IP's Intrinsic orientation and center of mass from the coefficients of IP's leading form (see Section 2.3.5), which made the registration fast and in a non-iterative way. That is the transformation (translation and rotation) information can be calculated in linear-squares method.

First let us introduce how to estimate the translation and rotation information for two corresponding segments. Let $f = 0$ and $g = 0$ be two IPs of a corresponding pair. Then let us explain translation and rotation estimation between them as follow.

Translation Estimation

If let the distance of center of mass of f and g be $\mathbf{t}$, *i.e.*,

$$g(\mathbf{x}) = f(\mathbf{x} + \mathbf{t}), \tag{3.18}$$

where $\mathbf{t}$ is the translation vector, then using (2.50) the centers of both polynomials are given by

$$\begin{aligned}\mathbf{c}_f &= -F^{\dagger}_{[n-1,1]}F_{[n-1]},\\ \mathbf{c}_g &= -G^{\dagger}_{[n-1,1]}G_{[n-1]},\end{aligned} \tag{3.19}$$

Now translation **t** is given as the difference between the centers:

$$\begin{aligned} \mathbf{t} &= \mathbf{c}_g - \mathbf{c}_f \\ &= -G^{\dagger}_{[n-1,1]} G_{[n-1]} + F^{\dagger}_{[n-1,1]} F_{[n-1]}. \end{aligned} \tag{3.20}$$

Note, since translation leaves the leading form unchanged, then usually $G_{[n-1,1]} = F_{[n-1,1]}$ as both matrices are constructed from the respective leading form only. Therefore, in our work we only use the translation estimation as

$$\mathbf{t} = G^{\dagger}_{[n-1,1]}(F_{[d-1]} - G_{[d-1]}). \tag{3.21}$$

Rotation Estimation

For rotation estimation, first let us assume

$$g(\mathbf{x}) = f(R\mathbf{x}) \tag{3.22}$$

where R is the 3×3 rotation matrix. The intrinsic orientations (axes) V_f and V_g can be computed as explained in Section 2.3.5. Since the rotation V_f^{-1} aligns the polynomial f with the world coordinate system and V_g aligns the world coordinate system with the intrinsic axes of polynomial g, the rotation R that transforms the implicit polynomial f to g is given as

$$R = V_g V_f^{-1} = V_g V_f^{\mathrm{T}}. \tag{3.23}$$

It should be noticed, however, that any intrinsic orientation matrix such as V_f does not provide a unique transformation between the intrinsic axes and the standard Cartesian frame. For a 3D polynomial, there are 8 such groups of 3 eignevectors (corresponding to the symmetry of a hyperboloid). A naive method for disambiguating the rotation estimates is to try 8 times for different rotation and find the one with smallest fitting error.

Estimation for whole range image

Actually even if we found only one segment-wise correspond, we can estimate the registration for whole range image. But for more robust estimation we first filter the incorrect estimation. Since each corresponding pair can vote its own estimation, if the current correspondence is not correct, then its estimation must be rather different to the correct ones (the majority ones) and should be filtered. Then we combine the correct estimations for integral estimation as follow.

$$\begin{aligned} \mathbf{t} &= \frac{1}{N}\sum_{i=1}^{N}(\mathbf{c}_f^i - \mathbf{c}_g^i) \\ R &= \frac{1}{N}\sum_{i=1}^{N}(V_g^i {V_f^i}^{\mathrm{T}}) \end{aligned} \tag{3.24}$$

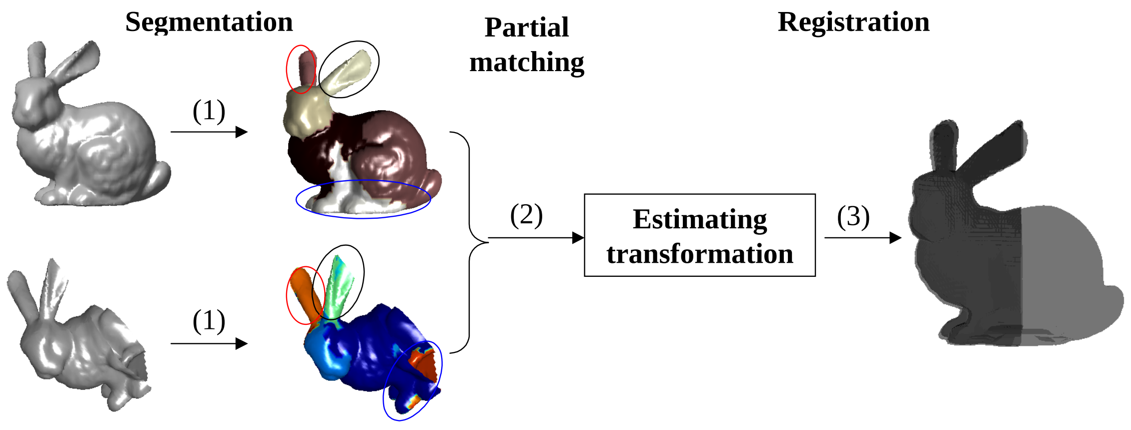


Figure 3.15: Partial 3D registration

Table 3.2: Invariants of Correspondences

Segment	half object	original object
left ear	$10^8(0.0160, 0.4044, 1.8449)$	$10^8(0.0180, 0.2021, 2.1230)$
right ear	$10^8(0.0199, 0.5808, 1.0483)$	$10^8(0.0213, 0.2080, 1.7999)$
bottom	$10^9(0.0003, 0.0025, 3.0211)$	$10^9(0.0004, 0.0026, 2.7701)$

where $\mathbf{c}_f^i$ and $\mathbf{c}_g^i$ are the centers of the two IPs of ith segment pair respectively; V_g^i and V_f^i are the intrinsic matrices of the two IPs of ith segment pair respectively.

3.4.3 Experimental Results

The first example is shown in Fig. 3.15, where we take half of the original "bunny" and transform it to another position using random Euclidean transform, and then test the registration between it and the original one. First we segment each of them with same accuracy. Then we calculated the invariants for each segment, and found three pairs of them with the invariants shown in Tab. 3.2.

The second example is for the real range images which objtained by scanning the stairs in Bayon Temple. The raw data are ploted in Fig. 3.16, where two pieces range images are obtained in different coordinates.

Through the calculation of invariants, we find the corresponding segment pairs as shown in Fig. 3.17(a). And then according to these correspondences we estimate the translation and rotation parameters with the method described above.

Note, in the examples, although our segmentation cannot give the exact segmentation results for the overlapping segments, the IPs to some extend can overcome the problem, since IP has good robustness to missing data. On the other hand, the relative larger errors still exist in the registration results, since it cannot give a registration estimation as accurate as ICP method done. 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 [ONKI05].

3.5 Summary

In this work, we proposed a 3D surface segmentation method to the field of vision for improving the IP representation. First, we propose a three-step cut-and-merge approach for obtaining a low-degree IP segment combination. Segmenting a whole object into small parts and then fitting them to moderate lower-degree IPs avoids the time-consuming computation of fitting a high-degree IP to a complex object. As a result, the IPs assigned to segments provide the geometric properties such as curvatures and Euclidean invariants. By adjusting the thresholds using different tolerance levels, IP surfaces with different accuracy and smoothness are obtained. Degrees of accuracy and smoothness can be freely selected so as to be suitable for the requirements for various vision applications.

To prove the effectiveness of the method, directly utilizing the segmentation results, we derived a method for the application on registration of range images. Over the traditional registration method, our method achieves automatic, non-iterative and initialization-independent registration. As a result, it is expected to contribute to the initial registration in 3D reconstruction for range images.

Unfortunately, however, this IP segmentation method might suffer from the over fitting problem. That is we use the over-high degree IP to present a simple region. As the example shown in Fig. 3.10, since each of segments is a 3D planar, actually it should have be not encoded with 2-degree IPs, but the 1st degree IP is sufficient. For solving this problem, we described a new adaptive IP fitting method at Chapter 4.

3.6 Algorithms

Finally, let us give the algorithms which are needed in previous sections.

Algorithm 1 presents the description of Unfit cutting procedure, where the function **3LFitting** uses the 3L method [BLC00] to fit a region and return an IP as a result; the function **Measuring** is used to measure the similarity between a region and an IP referring to (3.1, 3.2); and function **BiSegment** divides a region into two parts, Inner part and Outer part defined in (3.6).

Algorithm 2 describes the Merging procedure, where **FindNeighbors** is a function for finding the neighbors of a certain region.

Algorithm 1: Unfit cutting procedure

```
Input $I$: point data set, $T_1$, $T_2$: thresholds
Output $\{\tilde{R}\}$: regions, $\{\tilde{IP}\}$:polynomials
Working Variables $IsAnyCut$: flag. $\begin{cases} true, & \text{cut operation occured in loop,} \\ false, & \text{otherwise} \end{cases}$
Step 1 Initialization:
  $\{\mathcal{R}\} \leftarrow \{whole\ data\ set\}$;
  Initialize $\{InnerRg\}$ and $\{OuterRg\}$ with two empty stacks;
Step 2 Main loop:
  $IsAnyCut \leftarrow false$;
  for $R_i \in \{\mathcal{R}\}, i = 1, 2, \ldots$ do:
    $[IP_i] \leftarrow$ 3LFitting($R_i$);
    $[Dd, Ds] \leftarrow$ Measuring($R_i, IP_i$);
    if Dd < $T_1$ and Ds > $T_2$
      push $R_i$ into $\{\tilde{R}\}$;
      push $IP_i$ into $\{\tilde{IP}\}$;
    else
      [InnerPart, OuterPart] $\leftarrow$ BiSegment($R_i, IP_i$);
      add InnerPart into $\{InnerRg\}$;
      add OuterPart into $\{OuterRg\}$;
      $IsAnyCut \leftarrow true$;
    end
  end
Step 3 Checking end:
  if $IsAnyCut$
    $\{\mathcal{R}\} \leftarrow \{InnerRg\} \cup \{OuterRg\}$;
    Clear $\{InnerRg\}$ and $\{OuterRg\}$;
    GOTO Step 2;
  else
    Output $\{\tilde{R}\}$, $\{\tilde{IP}\}$;
    Return;
  end
```

Algorithm 2: Merging procedure

```
Input $\{\mathcal{R}\}$: segmented regions; $\{IP\}$: IPs corresponding to the regions; $T_1/T_2$: thresholds
Output $\{\mathcal{R}\}$, $\{IP\}$
Working Variables $mergedFlag$: a flag checking whether merging occurred in the loop;
                  $\{\hat{R}\}$: a stack preserving the regions
Initialization
  $\{\hat{R}\} \leftarrow \{\mathcal{R}\}$;
Main Loop
  while $\{\hat{R}\}$ is not empty do:
    Find the largest region $R_{max} \in \{\hat{R}\}$ and its corresponding IP $IP_{max}$;
    $\{R_{neighbor}\} \leftarrow$ FindNeighbors($R_{max}$);
    $mergedFlag \leftarrow false$;
    for $R_i \in \{R_{neighbor}\}, i = 1, 2, \ldots$ do:
      $[Dd, Ds] \leftarrow$ Measuring($R_i, IP_{max}$);
      if $Dd < T_1$ and $Ds > T_2$
        $[R_{max}] \leftarrow merge(R_i, R_{max})$;
        $[IP_{max}] \leftarrow$ 3LFitting($R_{max}$);
        $mergedFlag \leftarrow true$;
        Updating $\mathcal{R}$ with $R_{max}$ and removing $R_i$ from $\mathcal{R}$
      end
    end
    if $mergedFlag$ is false
      pop $R_{max}$ from $\{\hat{R}\}$;
    else
      $\{\hat{R}\} \leftarrow \{\mathcal{R}\}$;
    end
  end
```

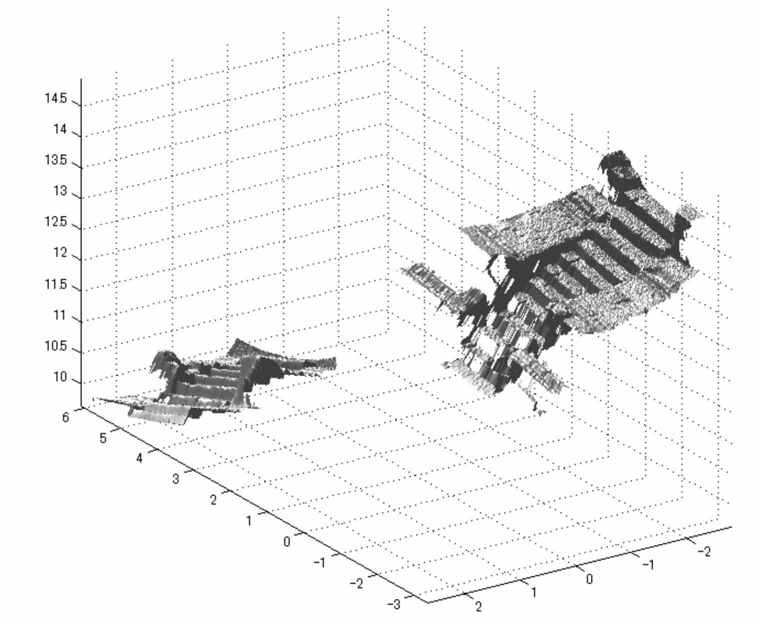


Figure 3.16: Two scans of the stairs in Bayon Temple

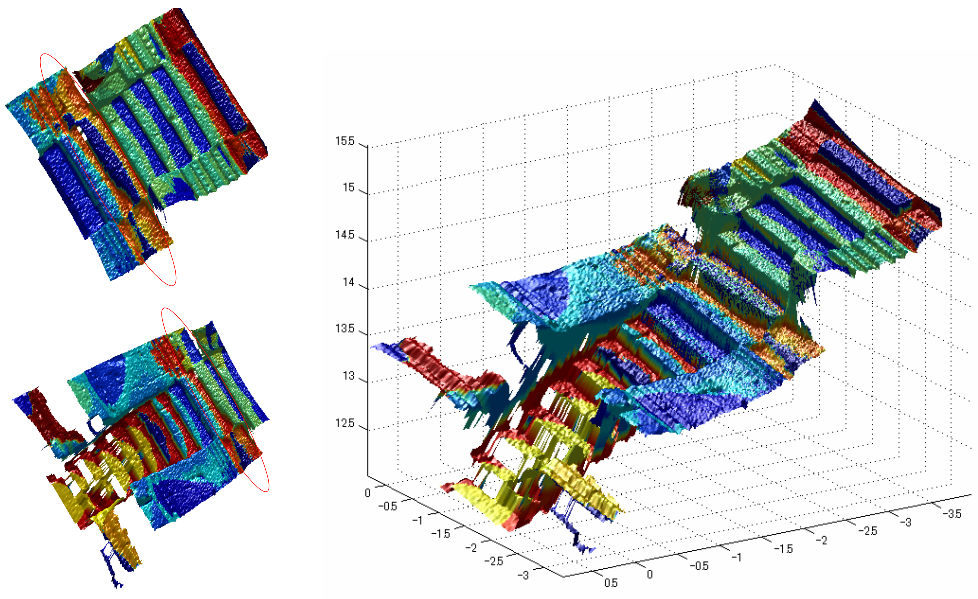


Figure 3.17: Registration for range data sample in Bayon Temple. (a) segment matching (b) result of registration

Chapter 4

An Adaptive and Stable Method for IP Fitting

In this chapter, we will present a new fitting method for implicit polynomial (IP) which is capable of adaptively determining a moderate degree for an IP in fitting procedure and simultaneously improving the numerical stability for the fitting.

As reviewed in Section 2.2, in order to achieve a stable IP representation for a single object, various fitting methods have been developed. Non-linear algorithms [Tau91, KC94, Kna94] maintain the Affine invariant, but often failed a reliable representation, and due to numerical instability and high computational cost, these algorithms are able to fit the data by polynomials of relatively low degree only. Linear fitting algorithms: 3L [BLC00], Gradient-one [TTC00], Min-Max and Min-Var [HBM04], apply a linear LS (Least Squares) solution to the fitting problem and thus have a lower computational cost and appear to have better performance in representation. All of the prior fitting methods seem developed for the same purpose, achieving numerical stability. Over time, the authors have done many efforts, by changing the constraints for fitting (see Section Sec:FittingOverwiew), to find the stable behavior. However, they neglected a crucial problem and issues which can summarized as follows

1. **Degree-fixed fitting procedure**
   That means, before handling a fitting procedure, the degree of IP first must be determined according to the complexity of the object, and the degree will be fixed during the procedure. This is no problem when fitting a simple object; for example, it is easy to determine a 2D IP of degree 2 to fit a simple 2D shape that looks like an ellipsoid. However, the determination confuses a researcher who needs to fit a complex object such as the bunny in Fig. 1(a); determining an over-low degree leads to undesired inaccuracy (Fig. 1(b)), whereas determining an over-high degree leads to the over-fitting problem (Fig. 1(d)). Therefore, the researcher is forced to waste much time finding a moderate degree by trying different degrees several times and selecting the best one from the

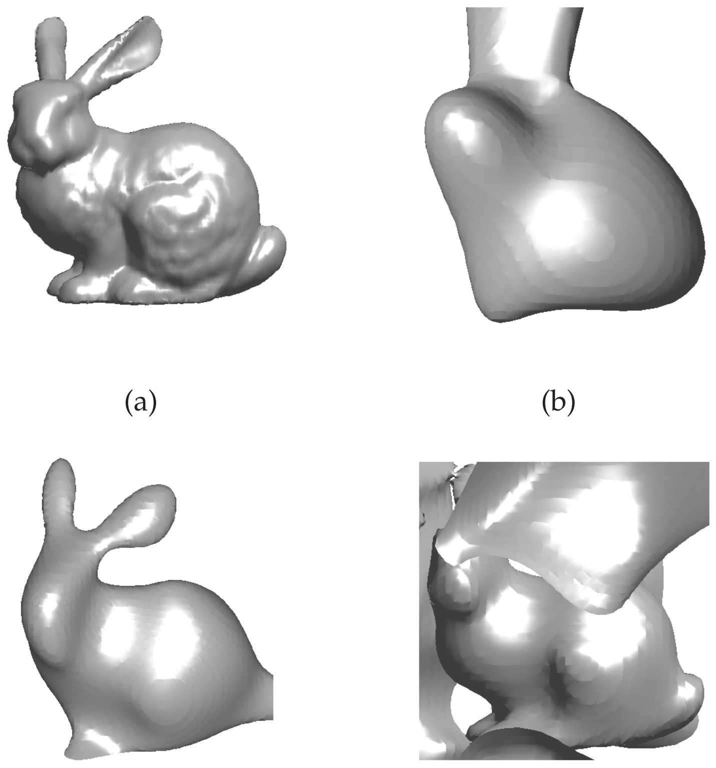


Figure 4.1: IP fitting results by conventional methods: (a) Original data set; (b) 4-degree IP surface; (c) 6-degree IP surface; (d) 10-degree IP surface. After reviewing all the results, the moderate 6-degree IP surface can be selected. But it is difficult to determine it from the bunny shape alone, and therefore much time is wasted in adjusting the degree.

results (Fig. 1(b)–(d)).

Also this problem might limit IP fitting method to be difficultly applied into a automatic application system which faces to deal with various object shapes with different complexities. For exmaple, for the segmentation method described in last chapter, we have to fix the same degree IP for each segments, no matter how complex the segments are.

2. **Global fitting instability**
   In common sense, despite the over-fitting problem, there is no doubt that the higher degree IP is very convenient for a complex object. Fortunately, several research groups have already proposed the method, *ridge regression (RR) regularization*, to overcome over-fitting problems (see the overview in Section 2.2.3), that is, to remove the extra zero sets by improving the numerical stability of the fitting procedure. Fig. 4(a) shows an example

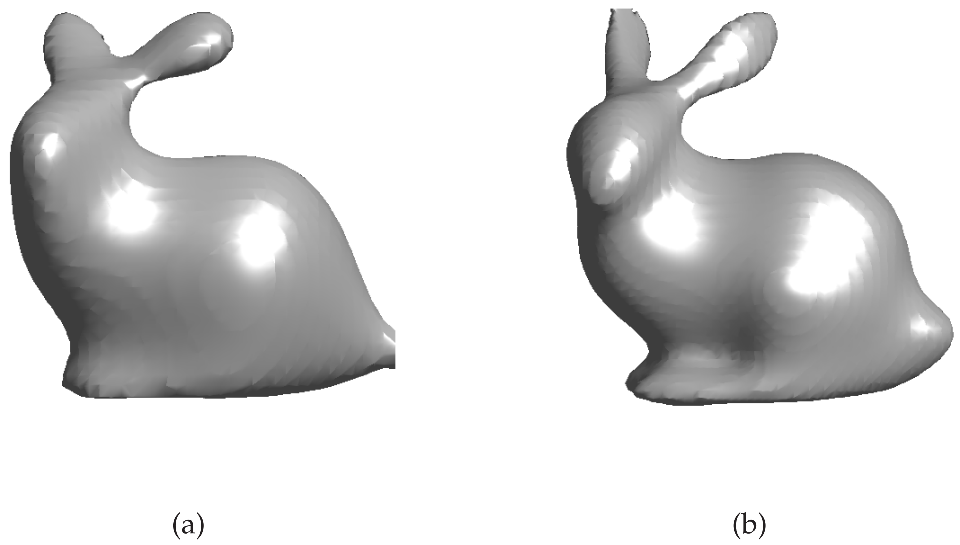


Figure 4.2: Comparison between conventional RR regularization and our RR regularization: (a) 10-degree IP surface globally stabilized by conventional RR regularization. (b) The result of our method.

where the RR regularization was used to improve the global stability of the 10-degree IP shown in Fig. 1(d). The method is obviously useful for over-fitting, but, on the other hand, the method produces deterioration in local accuracy.

These shortcomings greatly limit the applicability of IP. For example, the indetermination of the IP degree leads to the disapplicable for an automatic vision system; and the fitting instability leads to the bad performance such inaccuracy for the application such as object recognition.

In this chapter, we will solve above two issues by presenting a novel method based on QR-decomposition with Gram-Schmidt orthogonalization and a novel RR regularization. Over the traditional fitting method, it is capable of: 1) adaptively determining the moderate degree for fitting that depends on the complexity of objects, in an incremental manner without greatly increasing the computational cost; 2) maintaining global stability while also avoiding excessive loss of local accuracy. As an example, shown in Fig. 4(b), our method adaptively determines an IP of moderate degree for fitting the *bunny* object, and the result shows better global and local accuracy than the prior methods.

The first advantage of this method is its computational efficiency because the incrementability of Gram-Schmidt QR decomposition dramatically reduces the burden of the incremental fitting by reusing the calculation results of the previous step. The second advantage is that the method can successfully avoid global instability, and furthermore maintain local accuracy, because constraints derived from the RR regularization are selectively utilized in our incremental

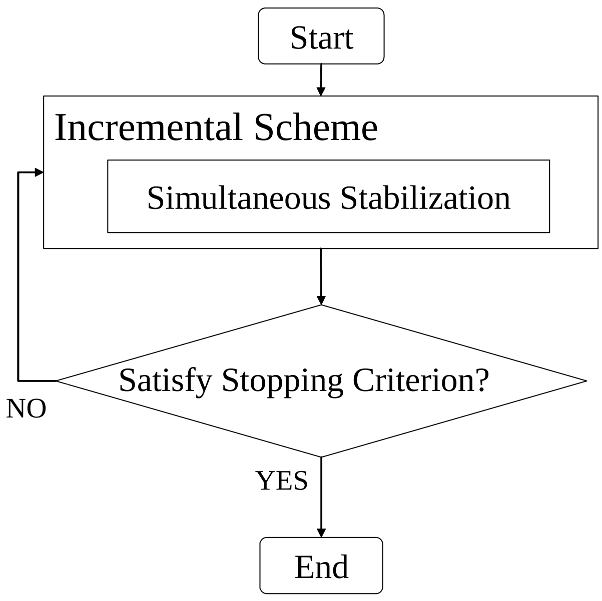


Figure 4.3: Flowchart of the algorithm.

scheme. Finally, a set of stopping criteria can successfully stop the incremental scheme at the step where the moderate degree is achieved.

This chapter is organized as follows. After clearing the problem definition of IP fitting in Section 4.1, we first present key components of our algorithm in Sections 4.2, 4.3, and 4.4 that are reflected in the flowchart shown in Fig. 4. In Section 4.2, we first propose an incremental scheme based on Gram-Schmidt QR decomposition that allows IP coefficients to be solved in incremental manner. Second, in Section 4.3, we propose a simultaneous procedure that enables us to automatically eliminate the numerical instability elements in the covariant matrix derived from least squares method, and in time we selectively adopt the RR constraints extracted from RR regularization. Third, in Section 4.4, a stopping criterion is designed so that the IP degree can gradually increase until an appropriate fitting result is achieved. The appropriateness is automatically determined by using our defined similarity functions between IPs and data sets. Section 4.5 reports experimental results followed by discussion in Sections 4.6. Section 4.7 extends the result to an application of adaptive segmentation based on the segmentation method provided in last chapter, and finally the summary is given in Section 4.8.

4.1 Formulations for IP Fitting and Stabilization

As described in section 2.3, 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. Thus the fitting methods aim at finding the coefficient vector $\mathbf{a}$ by minimizing the distance between the data set and the zero set of IP, such as:

$$\mathbf{a}_{min} = \arg\min_{\mathbf{a}}\{\frac{1}{N}\sum_{i=1}^{N} dist(\mathbf{x}_i, \mathcal{Z}(f_n))\}, \tag{4.1}$$

where $dist(\mathbf{x}_i, \mathcal{Z}(f_n))$ means a certain distance from the data point $\mathbf{x}_i$ to the zero set $\mathcal{Z}(f_n) = \{\mathbf{x} : f_n(\mathbf{x}) = 0\}$ of the IP. The simplest way is to directly take the $f_n(\mathbf{x}) = \mathbf{m}(\mathbf{x})^T\mathbf{a}$ as the distance metric and evaluate similarity errors in the least squares manner as

$$\varepsilon^2 = \| M\mathbf{a} - \mathbf{b} \|^2, \tag{4.2}$$

where M is the matrix of monomials of the IP, namely the i-th row of M is $\mathbf{m}(\mathbf{x}_i)$ (see Eq. (2.1)); $\mathbf{a}$ is the unknown coefficient vector of IP; and $\mathbf{b}$ is a zero vector. Following the least squares minimization manner, the problem can be formulated as:

$$M^T M\mathbf{a} = M^T\mathbf{b}. \tag{4.3}$$

Eq. (4.3) is just transformed from the least squares result, $\mathbf{a} = M^\dagger\mathbf{b}$, where $M^\dagger = (M^TM)^{-1}M^T$ is called a pseudo-inverse matrix. Note that in general, the number of data points is greatly larger than the number of IP coefficients.

Since M^TM is nearly singular and $\mathbf{b} = \mathbf{0}$, the non-linear methods solve the problem (4.3) by finding the eigenvector corresponding to the smallest eigenvalue [Tau91, Kna94]. But they suffer some major problems like local inconsistency with the continuity of the data set, local oversensitivity to small data perturbations, and instability of the coefficients due to excessive degrees.

In contrast, linear methods solve the problem (4.3) by first overcoming the singularity of M^TM and $\mathbf{b} = \mathbf{0}$. Thus, the problem can be solved more simply with linear least squares methods such as the LU decomposition method, the conjugate gradient (CG) method, singular value decomposition (SVD), and their variations. Many efforts for improving singularity of M^TM and making $\mathbf{b} \neq \mathbf{0}$ have been made in some recent research methods [BLC00, TTC00, HBM04] (see the overview in Section 2.2).

For example, *3L* method [BLC00] adds the linear constraints which restricts the data sets of two imaginary layers Γ^+ and Γ^- (external and internal) around the original data set to belong to the two nonzero sets (e.g., $\pm\varepsilon$ sets, where ε is some small positive value) of IP. The additional constraints can be described as:

$$\begin{cases} f(\mathbf{x}_+) = +\varepsilon, & \text{if } \mathbf{x}_+ \in \Gamma^+ \\ f(\mathbf{x}_-) = -\varepsilon, & \text{if } \mathbf{x}_- \in \Gamma^-. \end{cases} \tag{4.4}$$

The imaginary layers Γ^+ and Γ^- can be constructed from the points that are in the normal directions to the original data points with distance $\pm\varepsilon$. Therefore, according to (4.3) the least-square method formulation of 3L method can be written as:

$$M_{3L}^T M_{3L}\mathbf{a} = M_{3L}^T\mathbf{b}_{3L}. \tag{4.5}$$

The monomial matrix M_{3L} and right-hand vector $\mathbf{b}_{3L}$ are written as:

$$M_{3L} = \begin{pmatrix} M \\ M_{int} \\ M_{ext} \end{pmatrix}, \quad \mathbf{b}_{3L} = \begin{pmatrix} \mathbf{0}_{n\times 1} \\ -\boldsymbol{\varepsilon}_{n\times 1} \\ +\boldsymbol{\varepsilon}_{n\times 1} \end{pmatrix}, \tag{4.6}$$

where matrices M_{int} and M_{ext} are the monomial matrix derived from the internal and external layer date sets respectively, and in right-hand vector, $\mathbf{0}_{n\times 1}$, $-\boldsymbol{\varepsilon}_{n\times 1}$ and $+\boldsymbol{\varepsilon}_{n\times 1}$ mean vectors consisting of n elements 0s, $-\varepsilon$s and $+\varepsilon$s respectively.

Then 3L method successfully overcomes the singularity of M^TM with $M_{3L}^T M_{3L}$ and making $\mathbf{b}$ be $\mathbf{b}_{3L} \neq \mathbf{0}$. Since the linear methods like 3L method are simpler, numerically more stable and faster than the non-linear methods, they are receiving more attention. In this chapter, we focus on the linear methods.

Although the linear methods improve the numerical stability for fitting to some extent, they still suffer from the difficulty in achieving global stability. Especially while modeling a complex shape with a high-degree IP (with more coefficients), many extra zero sets are generated, which makes the fitting results nearly impossible to be interpreted. One important reason for global instability is the collinearity of column vectors of matrix M, namely, causing the matrix M^TM to be nearly singular (see [TTC00]).

Addressing this issue, Tasdizen *et al.* [TTC00] and Sahin and Unel [SU05] 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 with the same dimension to matrix M^TM. Note that here we just consider the condition number as $\lambda_{max}/\lambda_{min}$, where λ_{max} and λ_{min} are the maximum and minimum eigenvalues respectively. Accordingly Eq. (4.3) can be modified as

$$(M^TM + \kappa D)\mathbf{a} = M^T\mathbf{b}. \tag{4.7}$$

In general, D will be chosen to maintain Euclidean invariance. Although the simplest way is to set D to be an identity matrix, the more meaningful choices have been proposed by Tasdizen *et al.* for 2D [TTC00] and Sahin and Unel for 3D [SU05] that the l-th diagonal element of D is written as:

$$d_{ll} = \gamma \hat{t}, \tag{4.8}$$

where γ and $\hat{t}$ are two scalars, the details of which are shown in Section 2.2.3.

As mentioned above, the first shortcoming of the prior methods is that none of them allow changing the degree during the fitting procedure. Because of this, for solving the linear system (4.3) derived from the least squares method, the fixed IP degree must be assigned for calculating the inverse matrix of M^TM. The second shortcoming is related to the local inaccuracy resulting from the conventional RR regularization proposed in [TTC00, SU05]. Since the methods cannot rule out the ill condition for the covariant matrix $M^T M$ in time, they have to employ the RR term κD to operate the whole matrix $M^T M$, which leads to losing much local accuracy while global stability is improved. In this chapter, we will present a method to solve both these problems.

4.2 Incremental Fitting

In this section, we present an incremental scheme for the fitting methods, which allows the IP degree to increase in 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.

In this section, 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 in order to increase the IP degree.

4.2.1 Fitting using QR Decomposition

Without solving the linear system (4.3) directly, our method first carries out QR decomposition on matrix M as: $M = Q_{N\times m}R_{m\times m}$, where Q satisfies: $Q^T Q = I$ (I is an $m \times m$ identity matrix), and R is an invertible upper triangular matrix.

Then substituting $M = QR$ into Eq. (4.3), we obtain:

$$\begin{aligned} & R^T Q^T Q R \mathbf{a} = R^T Q^T \mathbf{b} \\ & \rightarrow R^T R \mathbf{a} = R^T Q^T \mathbf{b} \\ & \rightarrow R \mathbf{a} = Q^T \mathbf{b} \\ & \rightarrow R \mathbf{a} = \widetilde{\mathbf{b}}. \end{aligned} \tag{4.9}$$

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

4.2.2 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 describe how to decompose matrix M into QR with the Gram-Schmidt orthogonalization. Assume that matrix M consisting of columns

$$\{\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_m\} \tag{4.10}$$

is known. The Gram-Schmidt algorithm orthogonalizes the columns of M into the orthonormal vectors

$$\{\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_m\} \tag{4.11}$$

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

$$\mathbf{q}_1 = \mathbf{c}_1 / \parallel \mathbf{c}_1 \parallel, \;\; r_{1,1} = \parallel \mathbf{c}_1 \parallel \tag{4.12}$$

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:

$$\begin{aligned} r_{j,i+1} &= \mathbf{q}_j^T \mathbf{c}_{i+1}, \;\; \text{for } j \leq i, \\ \mathbf{q}_{i+1} &= \mathbf{c}_{i+1} - \sum_{j=1}^{i} r_{j,i+1} \mathbf{q}_j, \\ r_{i+1,i+1} &= \parallel \mathbf{q}_{i+1} \parallel, \\ \mathbf{q}_{i+1} &= \mathbf{q}_{i+1} / \parallel \mathbf{q}_{i+1} \parallel . \end{aligned} \tag{4.13}$$

With the Gram-Schmidt algorithm, matrix M is successfully QR-decomposed as $M = QR$, and thus the problem of solving Eq. (4.3) can be transformed to solving a linear system with an upper triangular coefficient matrix. Note if the upper triangular linear system of equations (4.9) is constructed, it can be much faster solved than solving a square linear system.

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.

4.2.3 Incremental Scheme

The idea of our incremental scheme is to continuously solve upper triangular linear systems (4.9) in different dimensions where the QR Decomposition with Gram-Schmidt orthogonalization (4.13) is utilized. This process is illustrated in Fig. 4.4, where the dimension of the upper triangular linear system increases, and thus the coefficient vectors of different degrees can be solved.

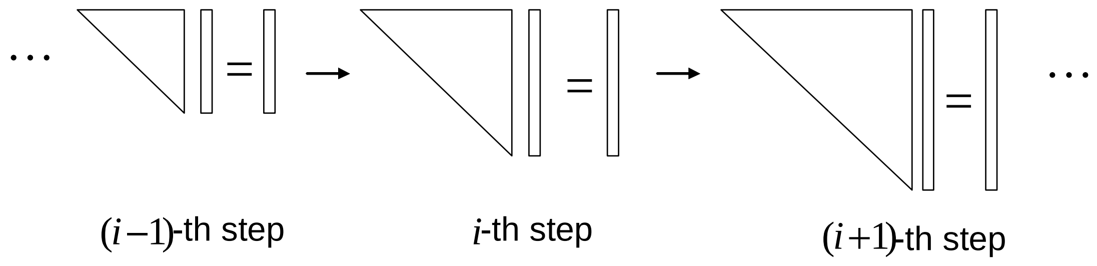


Figure 4.4: The incremental scheme that iteratively keeps solving an upper triangular linear system.

But the problem is that, if we do QR decomposition at each iteration that decomposes each square matrix into QR matrices at every iteration, it is obviously too time-consuming. For solving this problem, we find convenient way in what we construct each upper triangular linear system only using what has been done at the previous step.

Therefore 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. 4.5 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 gray blocks in Fig. 4.5.

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

1) How to calculate the upper triangular matrix R_{i+1},
2) 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. (4.13). 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 calculated 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. (4.13).

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 [BLC00], that iteratively calls the linear method at each step for obtaining the coefficient vectors of different degrees. It is obvious

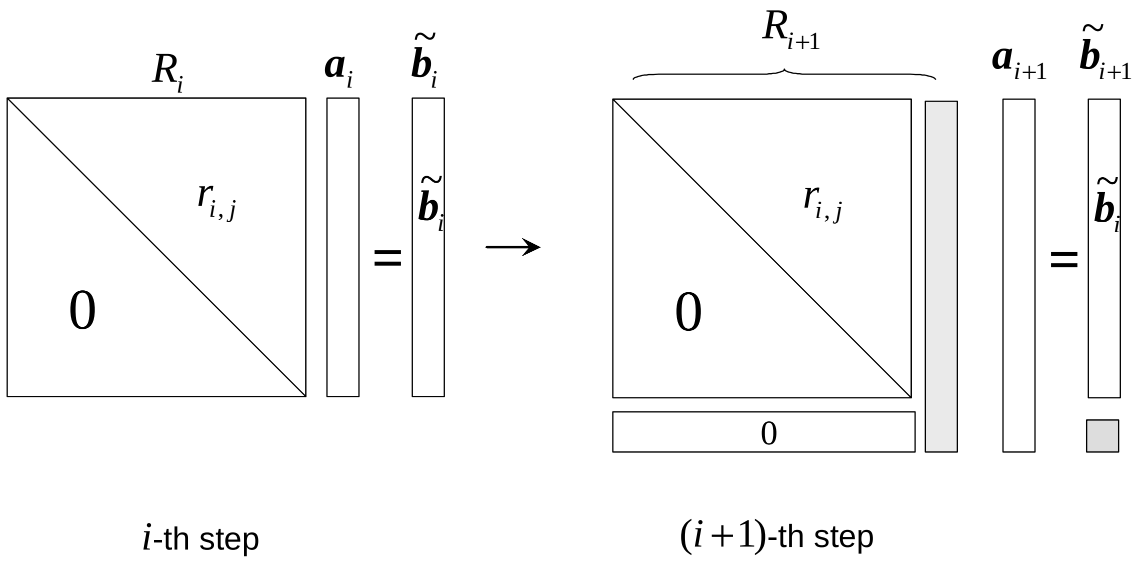


Figure 4.5: In order that the triangular linear system grows from the *i*-th step to the (*i* + 1)-th, only the calculation shown in light gray is required; other calculation is omitted by reuse at the *i*-th step.

that, for solving coefficient **a** at the *i*-th step, our method needs *i* inner-product operations for constructing the upper triangular linear system (see (4.13)), and $O(i^2)$ for solving this linear system; whereas the latter method needs i^2 inner-product operations for constructing linear system (4.3), and $O(i^3)$ for solving Eq. (4.3).

We would like to show a simple example in Fig. 4.6 to compare the actual calculation time between our method described above and two other methods that solve the linear system (4.3) independently at each step with two famous linear system solvers: 1) *LU* decomposition and 2) 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 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^TM in linear system (4.3) is often a dense matrix, and therefore ICCG cannot perform with good quality.

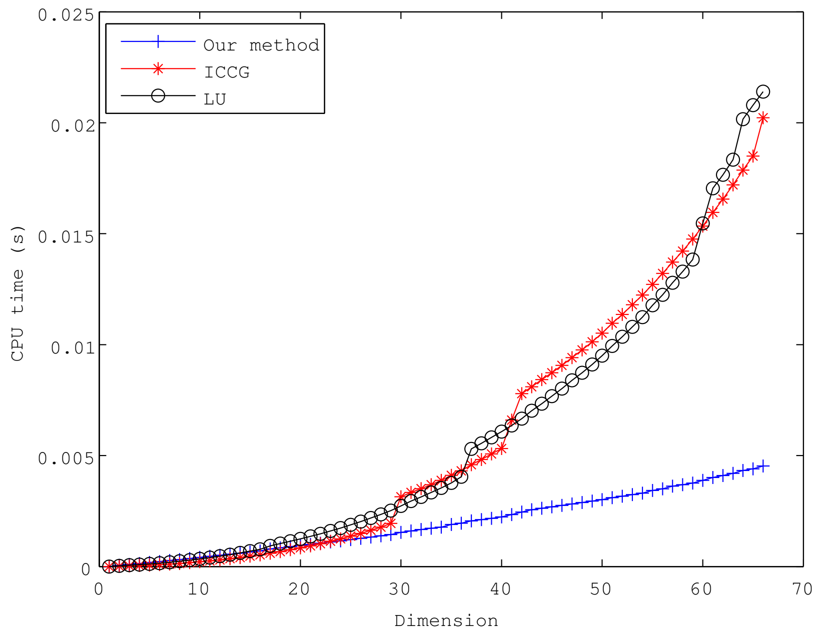


Figure 4.6: A comparison between our method and the assumed methods with respect to calculation time.

4.3 Global/Local Stabilization

As described in section 4.1, linear fitting methods usually suffer from the difficulty of achieving global stability. Although Ridge regression (RR) regularization can be adopted to achieve global stability [TTC00, SU05], local accuracy might deteriorate too much. First, we clarify how the regularization achieves global stability and why it sacrifices local accuracy. By considering this clarification, we propose a new RR regularization-based stabilization technique: our regularization can concentrate on the improvement of global stability by partially applying RR regularization through our incremental scheme.

4.3.1 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^TM to be nearly singular. The collinear columns of M are degenerated to contribute very little to the overall shape of the fit (see [TTC00]). 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 [GO00, TTC00, SU05] to be some individual constraints that we call RR constriants hereafter. According to the conventional definition in [GO00], the formula of RR regularization shown in Eq. (4.7) can be equivalently transformed as

$$\hat{M}^T\hat{M}\mathbf{a} = \hat{M}^T\hat{\mathbf{b}}, \tag{4.14}$$

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 as follows:

$$\hat{M} = \begin{pmatrix} & & M & \\ \sqrt{\kappa d_{11}} & & & \\ & \sqrt{\kappa d_{22}} & & \\ & & \ddots & \\ & & & \sqrt{\kappa d_{ll}} \end{pmatrix} \text{and } \hat{\mathbf{b}} = \begin{pmatrix} \mathbf{b} \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

where d_{ll} is the l-th diagonal element of D. Eq. (4.14) is similar to Eq. (4.3). Considering the difference between them, we can observe that Eq. (4.14) uses more n additional constraints:

$$\begin{aligned} \sqrt{\kappa d_{11}} \cdot a_1 &= 0, \\ \sqrt{\kappa d_{22}} \cdot a_2 &= 0, \\ &\vdots \\ \sqrt{\kappa d_{ll}} \cdot a_l &= 0, \end{aligned} \tag{4.15}$$

where a_l is the l-th element of coefficient vector $\mathbf{a}$. We call constraints in Eq. (4.15) RR constraints. If we see these constraints as Lagrange multiplier equations and $\sqrt{\kappa d_{ll}}$ as the corresponding weight, then all of these constraints tend to penalize the coefficients to be zero. As a result, the divergence of the coefficient value that causes global instability can be prevented.

However, over-penalizing all the coefficients is often inefficient, because such a brute-force constraint manner causes all the IP zero set to be penalized to its origin. Thus, local accuracy deteriorates (see [TTC00]). Our claim is that if we can apply the RR constraint only to the necessary part such as the part corresponding to the small eigenvalue, the RR regularization can resist deterioration of local accuracy.

4.3.2 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 $M^TM = R^TR$, and thus instability of M^TM can be determined from the eigenvalues of R. We can easily evaluate the singularity of R by only observing the diagonal values at each step, since upper triangular matrix R's eigenvalues always lay 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, $\sqrt{\kappa d_{ii}} \cdot a_i = 0$, only to this part to concentrate on solving this collinearity.

Suppose matrix $\hat{M}$ denotes the matrix M after having added the i-th RR constraint in Eq. (4.15) as:

$$\hat{M} = \begin{pmatrix} & & M & \\ & & & \\ 0 & \dots & 0 & \sqrt{\kappa d_{ii}} \end{pmatrix}.$$

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, that is, effective. From Eq. 4.13, 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:

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

From the derivation,

$$\begin{aligned} \hat{b}_i &= \hat{\mathbf{q}}_i^T \begin{pmatrix} \widetilde{\mathbf{b}} \\ 0 \end{pmatrix} \\ &= \frac{r_{ii}}{\sqrt{r_{ii}^2 + \kappa d_{ii}}} \mathbf{q}_i^T \widetilde{\mathbf{b}} = \frac{r_{ii}}{\hat{r}_{ii}} \widetilde{b_i}. \end{aligned} \tag{4.16}$$

We can see obviously that the i-th eigenvalue r_{ii} is improved to be a larger one $\hat{r}_{ii}$ by adding the RR constraint, and that this is adjusted according to the RR parameter κd_{ii}. On the other hand, the last element b_i of right-hand vector $\mathbf{b}_i$ is adjusted to be a smaller one. Thus, if we solve the i-th updated coefficient $\hat{a}_i$ at this step, then

$$\hat{a}_i = \frac{\hat{b}_i}{\hat{r}_{ii}} = \frac{r_{ii}}{r_{ii}^2 + \kappa d_{ii}} \widetilde{b_i} < \frac{\widetilde{b_i}}{r_{ii}} = a_i. \tag{4.17}$$

As a result, a_i is penalized to be a smaller one after adding the i-th RR constraint.

4.3.3 Normalization for the Linear System

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 (4.3).

Normalizing the linear system (4.3) is described as follows:

$$M'^T M' \mathbf{a}' = M'^T \mathbf{b}', \tag{4.18}$$

where M' is the normalized matrix of M and $\mathbf{b}'$ is the normalized vector of $\mathbf{b}$ in Eq. (4.3) described as:

$$\begin{aligned} M' &= \left\{ \frac{\mathbf{c}_1}{\| \mathbf{c}_1 \|}, \frac{\mathbf{c}_2}{\| \mathbf{c}_2 \|}, \ldots, \frac{\mathbf{c}_n}{\| \mathbf{c}_n \|} \right\}, \\ \mathbf{b}' &= \frac{\mathbf{b}}{\| \mathbf{b} \|}, \end{aligned} \tag{4.19}$$

assuming $M = \{\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_n\}$. Therefore the original coefficients can be obtained as $\mathbf{a} = \{\frac{\|\mathbf{b}\|}{\|\mathbf{c}_1\|} a'_1, \frac{\|\mathbf{b}\|}{\|\mathbf{c}_2\|} a'_2, \ldots, \frac{\|\mathbf{b}\|}{\|\mathbf{c}_n\|} a'_n\}$, assuming $\mathbf{a}' = (a'_1, a'_2, \ldots, a'_n)$.

Consequently the RR regularization is formulated as:

$$(M'^T M' + \kappa D')\mathbf{a}' = M'^T \mathbf{b}'. \tag{4.20}$$

Here we choose the same selection method for determining RR matrix D' as Tasdizen's method described in Eq. (4.8) in Section 4.1. Namely, each diagonal element of D' is chosen as $d_{ll} = \frac{\gamma \hat{t}}{\|\mathbf{c}_l\|^2}$, where the detail for calculating $\gamma \hat{t}$ refers to Eq. (2.31) and (2.32) in Appendix.

An important property of the normalization process is that, if M' is QR-decomposed as $M' = Q'R'$, then all the main diagonal elements of R' vary between 0 and 1, and the elements of vector $Q'^T \mathbf{b}'$ vary between -1 and 1. Its result is guaranteed by Gram-Schmidt orthogonalization (4.13), in which the result of mutual inner-product operation between the columns of M' only varies from -1 to 1. This beneficial property helps us to make a fair judgment to check out the existence of collinearity in the incremental process, by just checking whether r_{ii} is relatively too small in the range: [0 1].

4.3.4 Achieving Euclidean Invariant Method

Unfortunately, the above method is not Euclidean invariant, although it effectively improves the stability for fittings. That might restrain further serving for the vision applications such as object recognition and pose estimation. For keeping the fitting method Euclidean invariant, the diagonal matrix D in Eq. (4.7) is designed to satisfy $VDV^T = D$, where V is a block transformation matrix for IP transformation as: $\mathbf{a}' = V\mathbf{a}$ (see [TTC00] and [SU05]).

According to [TTC00, SU05], if we let $D' = KD$, where K is a diagonal matrix and its diagonal elements vary the same as the blocks of V (e.g. $K = diag\{k_0\ k_1\ k_1\ k_2\ k_2\ k_2\ \ldots\ k_i \ldots\ k_n\}$ in 2D), then D' is also Euclidean invariant. The important point is that k_is correspond to elements in the i-th form. Therefore, to extend our method to be a Euclidean invariant version, the incremental fitting procedure can just be modified to increase one-form by one-form, and once the unstable situation is eliminated, then we can improve the eigenvalues corresponding to the whole form.

4.4 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?

4.4.1 Stopping Criterion

For this problem, we use the same similarity evaluation described in section 3.3.1 for measuring the distance between the IP and the data set. Then, based on this similarity metric, we use the same stopping criterion in (3.5)

$$(D_{dist} < T_1) \ \wedge \ (D_{smooth} > T_2). \tag{4.21}$$

for terminating the incremental procedure. Once this stopping criterion is satisfied, we consider the desired accuracy is reached and the procedure will be terminated.

4.4.2 Algorithm for Finding the Moderate IPs

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

1) Constructing the upper triangular linear system with i column vectors of M with the method described in Section 4.2 and Section 4.3;

2) Solving this linear system to obtain coefficient vector $\mathbf{a}$;

3) Measuring the similarity for the obtained IP;

4) Stopping the algorithm if the stopping criterion (4.21) is satisfied; otherwise going back to 1) 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. 1) As a matter of convenience, we employ the constraints of the 3L method [BLC00] (The layer distance of 3L method in (4.4) is set as $\varepsilon = 0.05$). 2) 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 points to origin, as done in [TTC00]; 3) We choose T_1 and T_2 in Eq. (4.21) with about 0.01 and 0.95 respectively. 4) The RR parameter κ in Eq. (4.16) is empirically set to increase the original diagonal element r_{ii} about 10%. Note: we also refer the interested reader to see the discussion on setting κ in [TTC00].

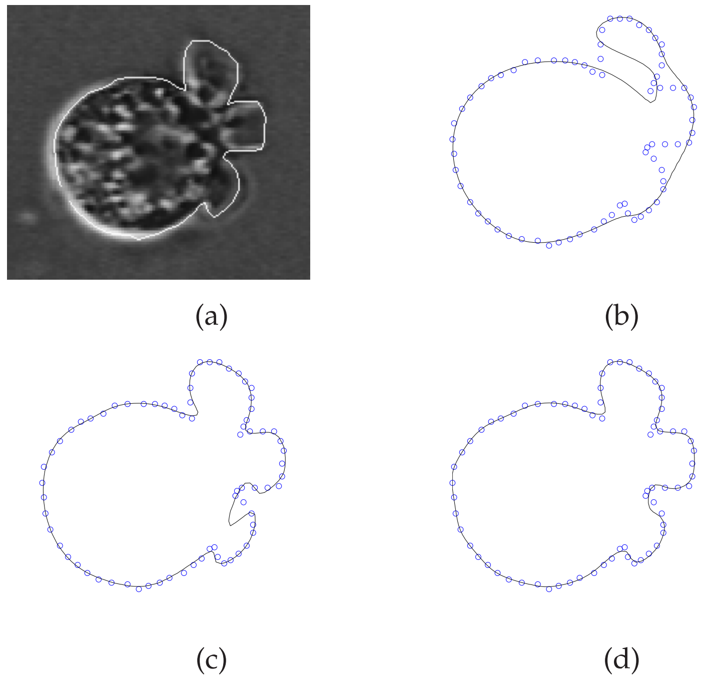


Figure 4.7: IP fitting results: (a) Original image. (b) $\alpha = 28$ (six-degree). (c) $\alpha = 54$ (nine-degree). (d) $\alpha = 92$ (thirteen-degree). Note "o" symbols represent the boundary points extracted from the image, and solid lines represent the IP zero set in (b)-(d).

4.5.1 Numerical Example

In this experiment, we fit an IP to the boundary of a "cell" object shown in Fig. 4.8(a). The moderate IP is found out automatically based on our stopping criterion (see Fig. 4.8(d)). To give a comparison, we also show some fits before the desired accuracy is reached (see Fig. 4.8(b) and (c)). We also see that, in the incremental fitting procedure, the result at each step is globally stabilized (without any extra zero set) by partially using RR constraints described in Section 4.3.

We also track the convergence of D_{dist} and D_{smooth} shown in Fig. 4.8(e). Although there are some little fluctuations on the graph, D_{dist} and D_{smooth} are convergent to 0 and 1 respectively, which also proves the stopping criterion in Eq. (4.21) can effectively measure the similarity between IPs and data sets.

4.5.2 2-D and 3-D Examples

Some 2-D and 3-D experiments are shown in Fig. 4.9 and Fig. 4.10 respectively where the fittings are set with the same parameters as the first example. The result shows our method's adaptivity for different complexities of shapes.

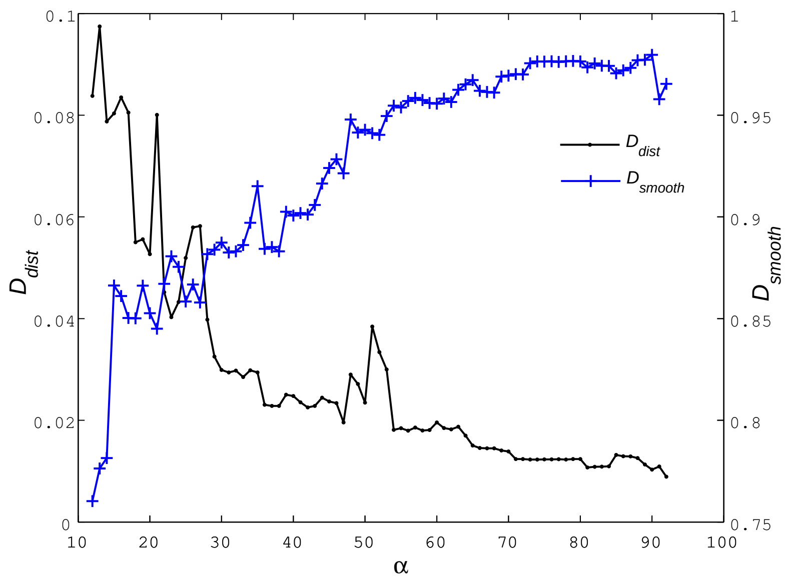


Figure 4.8: IP fitting results: Convergence of D_{dist}. and D_{smooth}.

4.5.3 Degree-fixed Fitting vs Adaptive Fitting

Fig. 4.11 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 the over-fitting nor the 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 in Section III-C, our method saves much computational time despite its determination of the moderate degree by the incremental process.

4.5.4 Comparison on Fitting Stability

Let us show a comparison in Fig. 4.13 between our method and the prior methods such as 1) 3L method [BLC00], 2) RR regularizations [TTC00] and [SU05]. Also Fig. 4.14 shows that it is difficult, maybe impossible, for the conventional methods to solve the local accuracy problem by adjusting κ.

Compare to the conventional RR method, Fig. 4.15 shows an example where we select the same RR parameter κ for both methods the conventional RR method [TTC00] and our method. Fig. 4.15 shows the stabilized results by changing κ in three different extends: $\kappa = 10^{-8}$, $\kappa = 10^{-6}$ and $\kappa = 10^{-4}$. From the results, we can see that, although with same RR parameter, stabilization

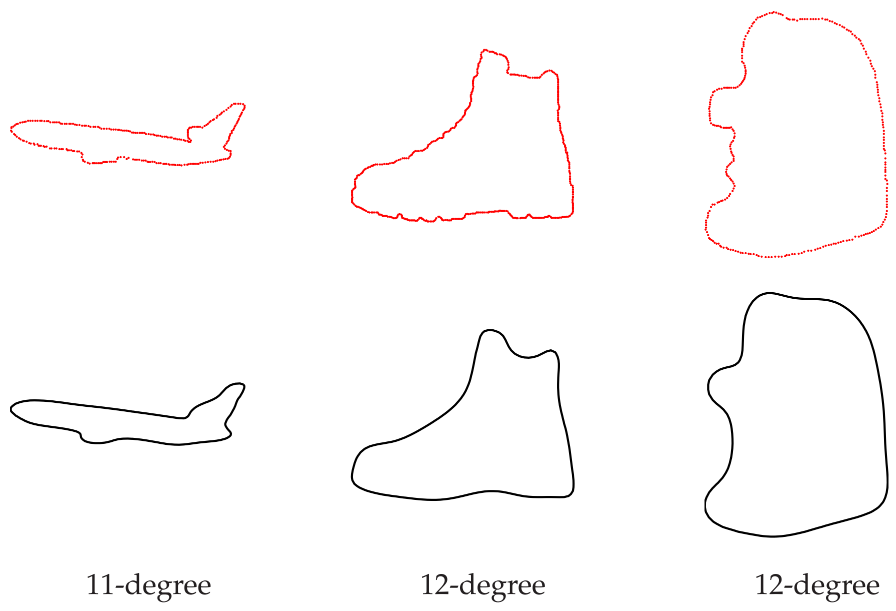


Figure 4.9: IP fitting results in 2D. First row: Original objects; Second row: IP fits.

levels are rather different that both of the methods trend to remove the extra zero sets, but our method prefers to maintain the local difference with higher accuracy.

In Fig. 4.16, we plot the eigenvalues (diagonal elements) of the upper triangular matrix R' decomposed from Eq.(4.20) in the fittings both for three cases: 1) 3L method (without using RR method), 2) using conventional RR method of $\kappa = 10^{-4}$, and 3) our method with selectively using RR constraints of $\kappa = 10^{-4}$. Since R' is derived from the normalized form of Eq.(4.20), the eigenvalues decreases from 1 to be closed to 0. We can see that our method more selectively modify the eigenvalues of R' than the conventional RR method.

As a result, our method shows better performance than the others, on both global stability and local accuracy.

4.6 Discussion

4.6.1 QR Decomposition Methods

Other famous algorithms of QR decomposition are Householder and Givens [HJ85]. They have proved more stable than the conventional Gram-Schmidt method in several severe cases. But in this chapter, since our target is only the regularized data set, we ignore the minor influence of rounding errors. Here we would just like to take advantage of the properties of QR decomposition that orthogonalize vectors one by one, which is the indispensable aspect to achieve our proposed adaptive-degree fitting algorithm.

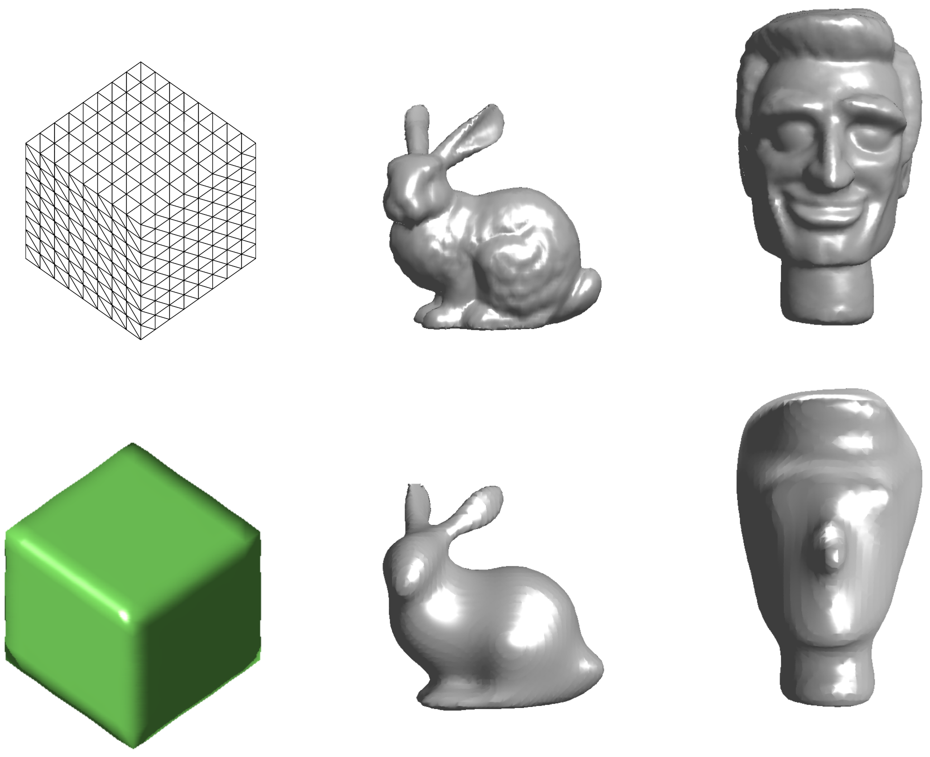


Figure 4.10: IP fitting results in 3D. First row: Original objects; Second row: IP fits.

4.6.2 Setting the Parameters T_1 and T_2

Since our stopping criteria can be approximately seen as a kind of Euclidean metric, it is intuitive to control the appropriate fitting accuracy. For example, a naive method is to let T_1 and T_2 be respected to degrees of data noises in statistics. Further discussion of this topic is beyond the scope of this work. In this work, we let T_1 and T_2 be close to zero and one respectively for a fine model and more tolerant values for a coarse one, since the 2D/3D Euclidean distance can be easily observed from coordinates.

4.7 Application: Adaptive Segmentation

The proposed adaptive fitting method gives the extendibility for the segmentation method which has been described in Chapter 3. The significant advantage brought to the segmentation method is that it can avoid the over fitting problem for the simple segments, and thus it results the adaptive segmentation. That is, the surface segments are represented with appropriate degree of IP respectively, the simple ones represented with low-degree IPs and the complex ones represented with higher-degree IPs.

To achieve this, we simply substitute the adaptive fitting method into the merging procedure described in Section 3.2.4. That means we adaptively merge the segmented result obtained by the cutting procedure with moderate-degree IPs.

Let us show a simple example in Fig.4.12. For segmenting the same target object given in Fig.3.8, the adaptive segmentation obtain the same segmentation result compare to the result in Fig.3.8(c) (both of them obtain 5 segments), but the adaptive segmentation method is more appropriate that it assigns the IPs of different degrees the segments of different complexities.

4.8 Summary

This chapter brings two major sub-contributions. First, it proposes a degree-incremental IP fitting method. The incremental procedure is very fast since the computational cost is saved by the process in which we keep solving upper triangular linear systems of different degrees, and effectively employ the incrementability of the Gram-Schmidt QR decomposition in the implementation. Thus the method is able to adaptively determine the moderate degree for various shapes.

Second, based on the beneficial property of the upper triangular matrix R in QR decomposition that eigenvalues are the elements on its main diagonal, the fitting instability can be checked out easily and quickly from the diagonal values. By selectively utilizing the RR constraints, our method not only improves global stability, but also maintains local accuracy. Furthermore, our method is easily modified as a Euclidean invariant method if the RR constraints are applied one form by one form, but not one by one.

By applying the adaptive fitting result into the 3D surface segmentation method described in Chapter 3, we obtained an extended version of the segmentation which is capable of adaptively segmenting 3D surface into IP segments of appropriate degrees, and accordingly avoiding the over fitting problem in previous method.

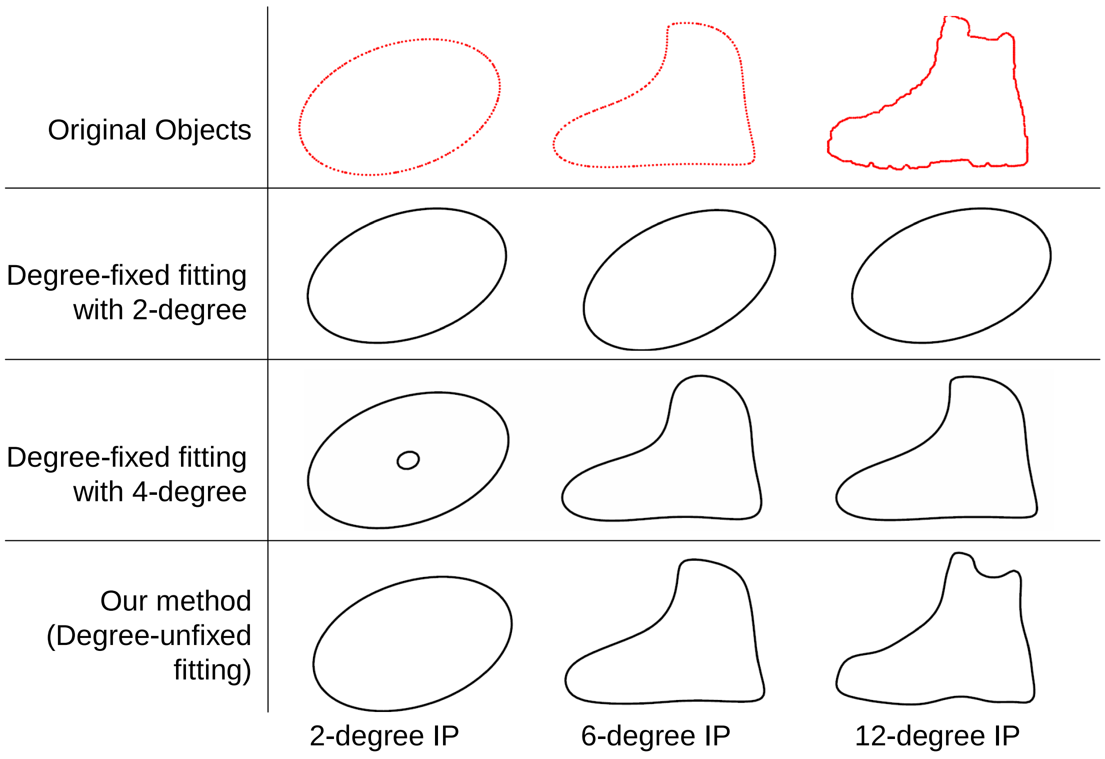

Original Objects
Degree-fixed fitting with 2-degree
Degree-fixed fitting with 4-degree
Our method (Degree-unfixed fitting)
2-degree IP
6-degree IP
12-degree IP

Figure 4.11: Comparison between degree-fixed fitting and adaptive fitting. First row: Original objects. Second row: IP fits resulted from degree-fixed fitting with two-degree. Third row: IP fits resulted from degree-fixed fitting with four-degree. Fourth row: IP fits in different degrees resulted from adaptive fitting with setting parameters as $T_1 = 0.01$ and $T_2 = 0.95$.

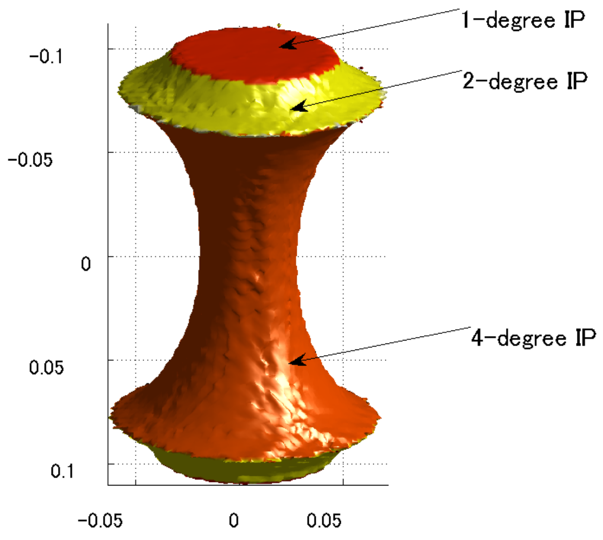


Figure 4.12: Adaptive segmentation results for the same object shown in Fig. 3.8.

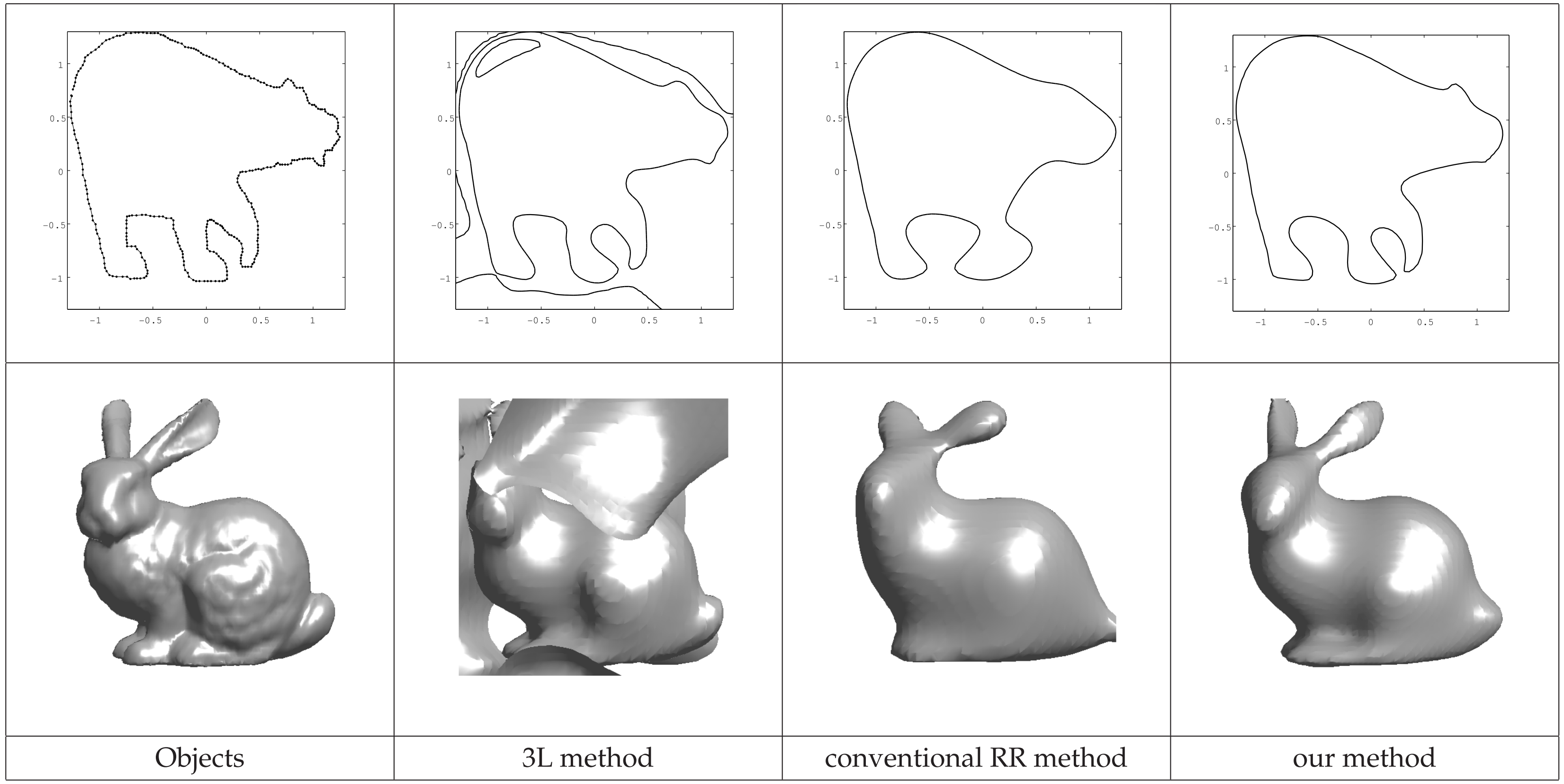


Figure 4.13: Comparison of fitting results: first column: original 2D/3D objects, second column: results obtained by 3L method [BLC00], third column: results obtained by RR method [TTC00, SU05], last column: results obtained by our method

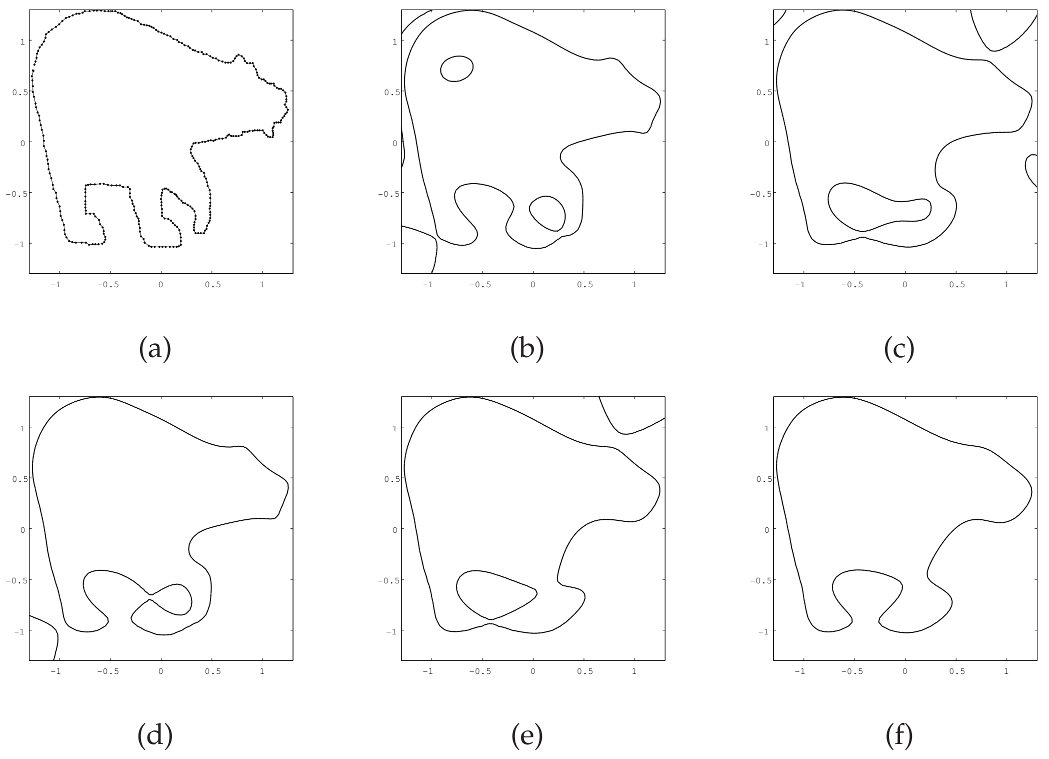


Figure 4.14: The relationship between κ and fitting results for conventional RR method is shown. For fitting the "bear" data set shown in Fig. 4.13, the shown results are obtained by setting different parameters: (a) original object, (b) $\kappa = 10^{-8}$, (c) $\kappa = 10^{-7}$, (d) $\kappa = 10^{-6}$, (e) $\kappa = 10^{-5}$ and (f) $\kappa = 10^{-4}$. As shown in this case, in order to achieve global stability completely, the conventional RR method cannot help but give up local accuracy.

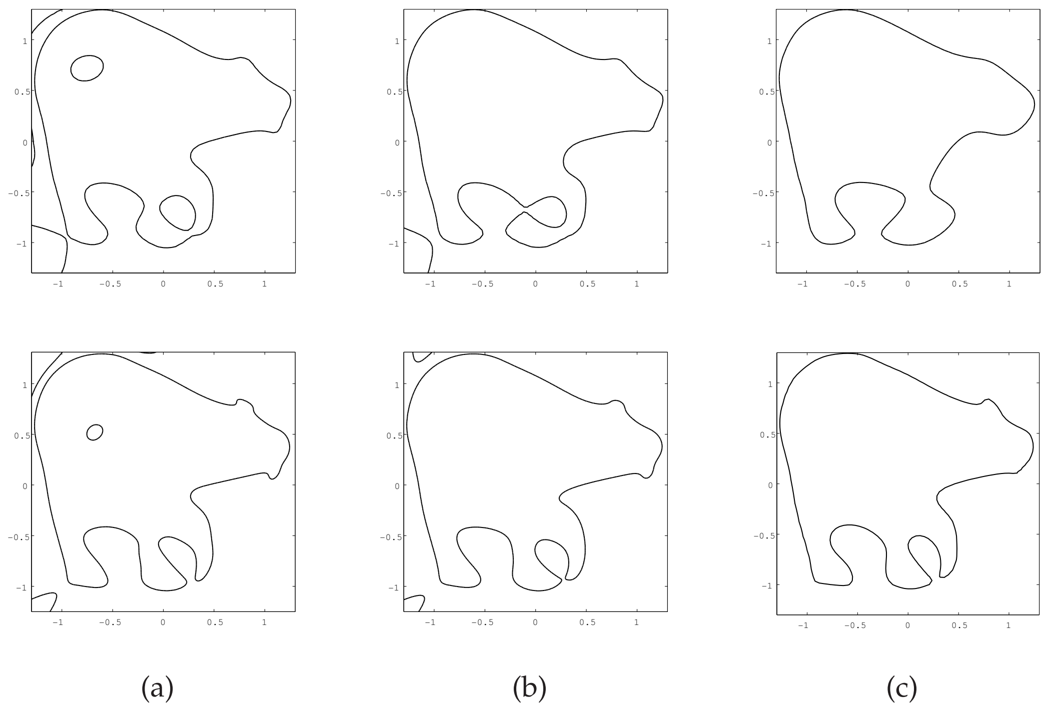


Figure 4.15: Comparison between the conventional RR method [TTC00] and our method by changing RR parameter κ. Top row: results of conventional RR method; Bottom row: results of our method. Column (a) $\kappa = 10^{-8}$, (b) $\kappa = 10^{-6}$, and (c) $\kappa = 10^{-4}$.

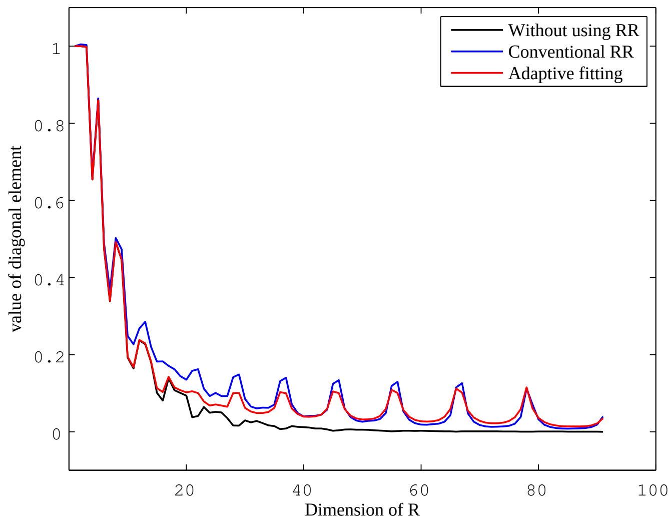


Figure 4.16: Eigenvalues of the upper triangular matrix R' resulted in three cases: 1) without using RR method, 2) conventional RR method of using $\kappa = 10^{-4}$, and 3) our method with selectively using RR constraints of $\kappa = 10^{-4}$.

Chapter 5

A Fast Registration Method of Using IP Gradient Field

In this chapter, we present a fast registration method by taking advantages of IP gradient field. That is, the registration views the gradient flow generated by IP as the descent direction for convergence of registration. It would be mentioned that this achievement needs two requirements: 1) the stable gradient field generation of IP and 2) the fast generation of gradient field of IP. The former one is fortunately guaranteed by our stable fitting method described in Chapter 4, and the second one is guaranteed by the symbolic computation described in Section 2.3. The prior fitting methods cannot generate a stable gradient field due to the fitting instability; that is why the prior works neglected the importance of the usage of IP's gradient field.

The task of rigid shape registration (alignment) aims to build a transformation relationship between a given shape (source) boundary and a target shape boundary. The process is often an intermediate but crucial step inside the whole computer vision complexity. An effective registration method benefits many vision applications such as surface reconstruction, texture mapping, view pose estimation, *etc*. Therefore, considerable work has been done on the registration problems.

In this chapter, the method registers an IP encoding the target object to a source object. In registration process, our method drives the motion of IP toward the desired 2D/3D image features, such as object boundaries, or inversely. The resulting evolution is the gradient flow derived from IP function that minimizes the proposed energy functional. The efficiency benefits from three main advantages over the traditional method as follows:

- With the property of IP having a few coefficients, it makes both IP gradients and its transformation to be calculated in an extremely light manner both on CPU time and memory.

- The registration is formulated as to minimize an energy functional derived from IP gradient flow, and thus it is efficiently superior to the traditional registration by eliminating the cost for calculating point-wise correspondences.
- A fast transformation method for IP coefficients is developed that achieves the acceleration for large data set through the inverse registration, that is, it transforms an IP to the large data set rather than transforming a large data set to an IP.

5.1 Related work

Considerable work has been done on rigid registration problems. One of the most popular approaches for spatial registration is the Iterative Closest Point (ICP) method introduced by several groups [BM92, RL01, CM94, CLSB92, MYL92, Zha92, ONKI05]. In its original form, ICP iteratively finds the closest point pairs between the data sets and tries to find the transformation that minimizes the distances between the closest point pairs. The obtained transformation is then applied to the source data set and new closest point pairs are found which are in turn used to find a new distance minimizing transformation. This process continues until the registration error between source and target falls below some quality threshold or the solution converges.

The transformation space of rotations and translations might contain many transformations that minimize the point pair distances locally. Such transformations are called local minima. In contrast, the global minimum is the transformation that minimizes the distances globally. The problem with ICP is that it can get stuck in local minima. In order to converge to the global minimum, an initial transformation is required that is close to the true value. ICP also suffers from the problem of slow convergence [RL01]. Moreover, one data set has to be a proper subset of the other and the method is sensitive to errors in feature extraction and occlusion. There are numerous ICP variants that address these problems.

Zhang [Zha92] improves upon the original ICP by using k-D trees to speed up closest point search and by using a dynamic distance threshold derived through distance histogram statistics. Points further apart than this adaptive threshold are not used for motion computation.

Using Euclidean distance in ICP restricts tangential sliding along aligned lines which causes slow convergence and multiple weak minima near the global minimum. To remedy that problem, Nguyen et al. [NNS99] replace the Euclidean distance by the "normal distance" between patches.

Rusinkiewicz and Levoy [RL01] breakup the ICP algorithm into 6 main stages and examine the effect of choices at each stage on convergence speed. They conclude that projection based correspondence finding and a point-to-plane error metric are most useful for increasing convergence speed. They construct a high-speed ICP by combining different variants and propose the use of variant switching algorithms that switch between appropriate variants depending on local error landscape or the probable presence of local minima.

Stewart [Ste02] proposes using the error covariance matrices of the data to get more accurate estimates of registration since estimates depend on error in data. He formulates registration as a statistical optimisation problem whereby point matching is based on the Mahalanobis distance instead of Euclidean distance and registration therefore involves minimizing the combined Mahalanobis distances of all data points.

Boughorbel et al. [BKAA04] replace the Euclidean distance of ICP by a Gaussian measure of distance and similarity. Minimizing an energy function of the Gaussian measures implies maximization of both overlap and similarity.

Oishi et al. [ONKI05] proposed a simultaneous registration approach of multiple range images using index images. It leads to faster convergence and better registration accuracy, noise tolerance and surface coverage are achieved by using replacing the Euclidean distance by the distance along the view direction for rendering.

Standard ICP is sensitive to outliers, *i.e.* a single bad correspondence will affect the sum of squared errors that is to be minimized and hence the transformation estimate is not robust. Standard ICP uses a closed form solution for the transformation estimate (in the minimization step). The problem is that a robust estimate in closed-form is not known. So, researchers have tried rob justifying ICP through 1) non-linear or RANSAC-based estimation in the minimization step, or 2) excluding bad correspondences computed during the matching step. Both approaches deal with outliers after they have already affected the error/distance values.

Chetverikov et al. [CSSK02] introduced the Trimmed ICP (TrICP) that excludes bad correspondences. It attempts to reject/trim wrong correspondences by using only those correspondences that have the smallest distances. Correspondences are sorted by squared distance and only a fixed fraction of them are used (Least Trimmed Squares). TrICP is robust to rotations and noisy data and has a proved convergence.

Avoiding the computation for searching point-wise correspondences, the methods like in [TC92, TCC98] are very efficient for coarse registration (initial registration). They achieve the registration by linearly (without any iteration) minimizing global distance, such as the distance according to center points and orientations of the objects, which can be extracted from some algebraic/geometric invariants. However these methods cannot deal with registration in the case of partially overlapping the target objects [SMFF07].

Recently, distance field is considered to be constructed for achieving the registration, such as the work in [MF07, IKK*07]. These works often need to take large memory to preserve the discrete distance field in advance, and then register the objects according to the potential distances in the distance field. Munim *et al.* [MF07] use closest-point distance to generate the distance field. Iwashita *et al.* [IKK*07] use the Fast Marching algorithm to calculate distance field for a 2D target object. The shortcomings of discrete distance field can be considered as: 1) taking too much memory space especially for 3D objects, 2) only finite region of distance field can be generated, and thus the registration is limited in the regions.

5.2 Methodology for Registration of Using IP

In this work, we propose a method to register the source data set modeled by implicit polynomial (IP) to 2D/3D target data set such as boundary information, or in an inverse way, which is independent of the types of modalities. We formulate an energy functional derived from IP gradient field, and iteratively minimize the functional for achieving the registration.

Over the prior methods, our method has the advantages: 1) unlike the ICP-based methods, our method does not require any computation for point-wise correspondences; 2) unlike the coarse registration methods, our method totally supports partial-overlapping registration; 3) unlike the registration methods using discrete distance field, our method needs a little memory space for only saving a few IP coefficients, and IP can generate infinite region of distance field to support registration in wide space. Additionally, since IP representation needs a few parameters and is based on the implicit form, both the calculation of its transformation and gradient flow for the minimization are extremely fast.

5.2.1 Energy Functional

The objective of registration is to find a transformation, through which the zero set $\{\mathbf{x}|f_n^{3D}(\mathbf{x}) = 0\}$ can "best" match the given points (source model). To achieve this, generally an energy functional E is required to be defined and then a external force will be generated to drive the motion that satisfies minimizing this potential energy.

Assume we minimize the following minimization

$$\mathbf{p} = \arg\min_{\mathbf{p}} E(\mathbf{p}). \tag{5.1}$$

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, such as:

$$E = \sum_{i=1}^{N} dist^2(T(\mathbf{p}, \mathbf{x}_i), f_n^{3D}), \forall \mathbf{x_i} \in \Omega \tag{5.2}$$

where the rigid transformation for point $\mathbf{x}_i$ respect to rotation R and translation $\mathbf{t}$ is defined as

$$\begin{aligned} T(\mathbf{p}, \mathbf{x}_i) &= R\mathbf{x}_i + \mathbf{t}, \\ \mathbf{p} &= \{R, t\}, \end{aligned} \tag{5.3}$$

$dist(\mathbf{x}_i, f_n^{3D})$ means a certain distance from the data point $\mathbf{x}_i$ to the zero set of f_n^{3D} of the IP; and Ω represents the 3D region of the source model.

5.2.2 Distance Metric

In general cases, in order to achieve the rigid registration, we often define the distance in Euclidean space, *e.g.*, the ICP-based registration defines the distance between the target object and source object by searching the nearest correspondences. Therefore it is unavoidable to the heavy computation for the nearest correspondence searching. For example the distance can be defined as:

$$dist(\mathbf{x}, Y) = \min_{\mathbf{y} \in Y} \| \mathbf{x} - \mathbf{y} \|_2, \tag{5.4}$$

where Y is the dataset of target model.

After the target object is modeled by an IP, we can give a distance metric for IP model, the simplest method is to use the algebraic distance, that is, directly use the value of $f(\mathbf{x})$ to define the distance between $\mathbf{x}$ and zero set of f. However, it cannot grantee a regular distance map that might cause the energy minimized unstably.

In this work, we choose the signed distance manner defined as:

$$dist(\mathbf{x}, f_n^{3D}) = \frac{f_n^{3D}(\mathbf{x})}{\| \triangledown f_n^{3D}(\mathbf{x}) \|}, \forall \mathbf{x} \in \Omega, \tag{5.5}$$

where $\triangledown$ denotes the gradient of IP function. The absolute distance $|dist|$ is the approximation of Euclidean distance (see [Tau91]) and it can give a relatively regular distance map. In Fig. 5.1, we show different distance maps for two shapes, a cross shape and a tooth shape. Fig. 5.1(a) gives the Euclidean distance map for each shape by searching the nearest points to the discrete model. Fig. 5.1(b) and (c) show the maps of algebraic distance and our distance respectively. As a result, the distance maps shown in Fig. 5.1(c) are approximate to the Euclidean map.

Note we choose this distance, also because it has three advantages: 1) the distance can be easily and fast calculated by IP function f_n^{3D} (the gradient of IP $\triangledown f_n^{3D}$ is calculated with symbolic method described in Section 2.3.4); 2) it successfully avoid the heavy computation for the nearest correspondence searching; 3) for calculating the distance, it requires small memory storage only for preserving the IP coefficients (note, the coefficients are few).

5.3 Registration

Given the metric of distance between a certain data set and an IP, the next problem is how to do the minimization for (5.1) to estimate the transformation parameters.

5.3.1 Minimization Using the Optimization Methods

Generally the minimization of the given criterion can be handled by the non-linear optimization tools, such as gradient descent method, Newton method, non-linear conjugate gradient

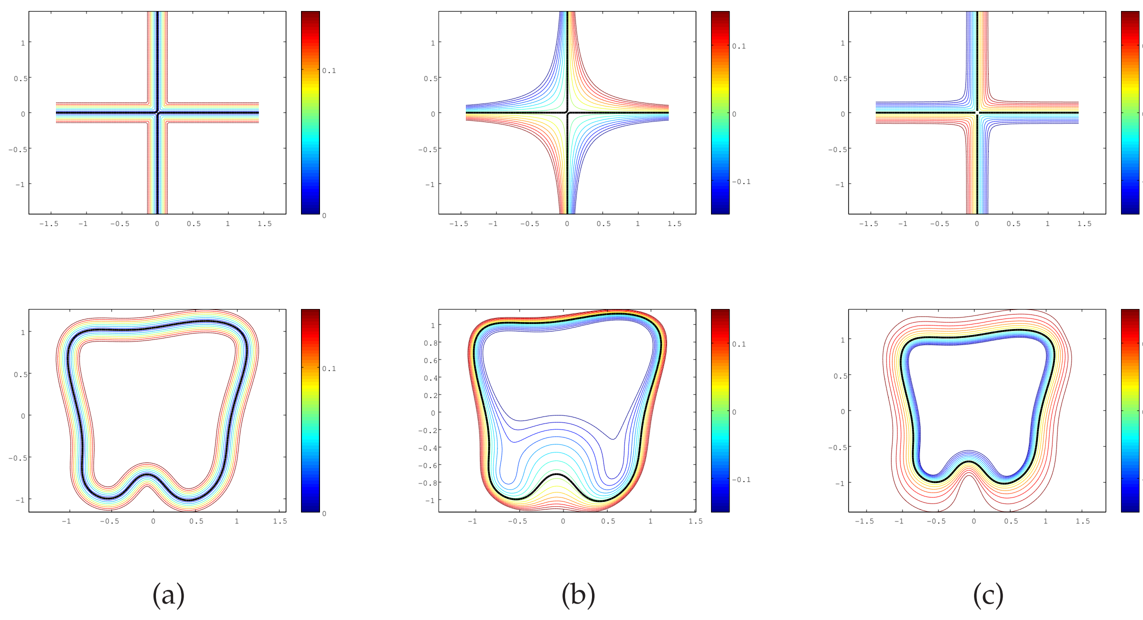


Figure 5.1: IP distance fields of a cross shape and a tooth shape shown in black line. Distance contours are shown from −0.15 to 0.15: (a) Euclidean distance to the shape contour calculated by searching nearest correspondences. (b) Algebraic distance calculated by polynomial f. (c) Algebraic distance calculated by polynomial $\frac{f}{\|\nabla f\|}$.

method by giving out the integral energy gradients:

$$\begin{aligned} \frac{\partial E}{\partial \mathbf{p}} &= \sum_{i=1}^{N} \frac{\partial dist^2(T(\mathbf{p},\mathbf{x}_i), f^{3D})}{\partial \mathbf{p}} \\ &= \sum_{i=1}^{N} \frac{\partial dist^2(T(\mathbf{p},\mathbf{x}_i), f^{3D})}{\partial T}\frac{\partial T}{\partial \mathbf{p}}. \end{aligned} \tag{5.6}$$

For simplifying the compuation, if we see $\| \nabla f_n^{3D}(\mathbf{x}) \|$ as a constant, then gradient function $G(\mathbf{x})$ ($\mathcal{R}^3 \rightarrow \mathcal{R}^3$) representing the partial differential of $dist(\mathbf{x}, f_n^{3D})$ respect to $\mathbf{x}$ can be simply calculated as

$$\begin{aligned} G(\mathbf{x}) &= \frac{\partial dist(\mathbf{x}, f_n^{3D})}{\partial \mathbf{x}} \\ &\approx dist(\mathbf{x}, f_n^{3D})\frac{\nabla f_n^{3D}(\mathbf{x})}{\| \nabla f_n^{3D}(\mathbf{x}) \|}. \end{aligned} \tag{5.7}$$

Then, by substituting (5.7) into (5.6), the calculation of the partial differential (5.6) can be derived as

$$\begin{aligned} \frac{\partial E}{\partial \mathbf{p}} &= \sum 2G(T)\frac{\partial T}{\partial \mathbf{x}_i} \\ &= \sum \begin{pmatrix} 2G(\mathbf{x}_i) \\ 4\mathbf{x}_i \times G(\mathbf{x}_i) \end{pmatrix}. \end{aligned} \tag{5.8}$$

Here we just use the quaternion transformation, for the detail on the quaternion transformation and the partial differential derivation, we refer the interest reader to Appendix C.

Therefore the minimization of (5.2) can be handled by the non-linear optimization tools, such as gradient descent method, Newton method, non-linear conjugate gradient, by using the gradient (5.8). However the optimizations suffer from the heavy computation of line searching (see [Mas06]), that is, at each iteration in such as Quasi-Newton method or non-linear conjugate gradient method, there generally need a line searching method to find the optimal direction for descent.

5.3.2 Our Minimization

In our method, the minimization is done through 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 during the first minimization. Next, it determines the transformation parameters to better describe the first minimization result. These two steps are repeated until convergence.

First step At the first step, by calculus of variations [Eva98], the Gateaux derivative (first variation) of the functional E in (5.2) to point $\mathbf{x}$ can be approximately formulated as

$$\frac{\partial E}{\partial \mathbf{x}} \approx 2G(\mathbf{x}), \tag{5.9}$$

if we consider ∇f_{3D} as a constant value for the computational convenience.

Therefore, we need 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 $\mathbf{x}$:

$$\frac{\partial \mathbf{x}}{\partial t} \approx -2G(\mathbf{x}). \tag{5.10}$$

But the optimization of the given criterion is not handled using the continuous rigid transformation. That is, every data point in the first step is considered to be gravitated towards the IP model regardless the rigid-transformation criterion.

Second step In the second step, from the view of computational efficiency, the integral forms are calculated as approximation by using summing operations; in the case where the target region Ω is represented in continuous manner, the calculation is done by approximately converting continuous one into discrete one.

Let $X \in \mathcal{R}^{N\times 3}$ preserve N 3D data points. Then the approximation of (5.10) by the above difference scheme can be simply written as:

$$\frac{\partial X}{\partial t} = X^{k+1} - X^{k}. \tag{5.11}$$

It is the approximation of discrete data set transformation by the above spatial difference scheme in (5.10). Then, let

$$A = (X^{k} - \bar{X}^{k})^{\mathrm{T}}(X^{k+1} - \bar{X}^{k+1}), \tag{5.12}$$

where $\bar{X}$ is the matrix each row consisting of the mean value (center point) of X. If A is decomposed with singular value decomposition (SVD) algorithm as $A = USV^{\mathrm{T}}$, then transformation is given by:

$$\begin{aligned} R &= UV^{\mathrm{T}}, \\ \mathbf{t} &= \bar{X} - \bar{X}'R^{\mathrm{T}}, \end{aligned} \tag{5.13}$$

where R and $\mathbf{t}$ are rotation and translation parameters respectively.

5.4 Inverse Registration

Inverse registration is the inverse process for the registration, that means we register an IP to the source model instead of registering a source model to an IP. Now let consider the case that the number of points required to be transformed is too large, each point $\mathbf{x}$ will be updated as $\mathbf{x}^{k+1} = R\mathbf{x}^{k} + \mathbf{t}$ at each iteration, and that is time-consuming for large-scale problem. Therefore, in this section, let us propose an acceleration method, an inverse registration. For achieving that, therefore, we first require a fast transformation method for an IP, that is the transformation just changes the coefficients of an IP, and then the IP is driven in the space like a rigid object.

5.4.1 IP Transformation

As described in Section 2.3.3, when a Euclidean transformation is applied to the IP model $f(\mathbf{x}) = \mathbf{m}(\mathbf{x})^{\mathrm{T}}\mathbf{a}$, vector of monomial $\mathbf{m}$ is transformed as $\mathbf{m}' = V(\mathbf{p})\mathbf{m}$, where we suppose that vector $\mathbf{p}$ consists of the parameters for affine transformation, and the square matrix V is transformation for monomial deriving from the affine transformation uniquely [TC92]. The

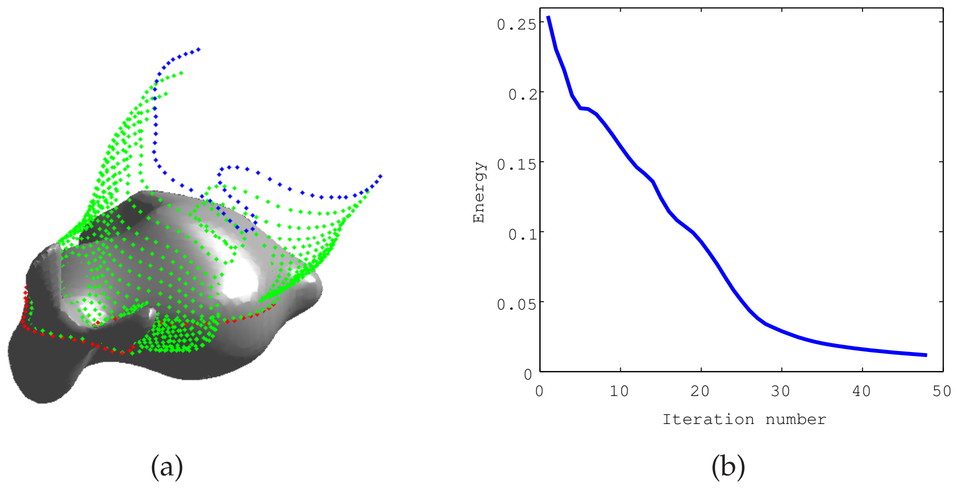


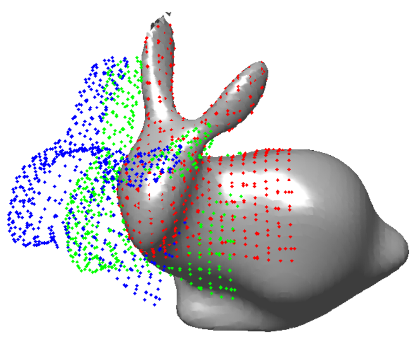
Figure 5.2: IP registration with planar 3D points: (a) registration process between "bunny" IP model and points (blue points: initial position; green points: selected mid-step positions; red points: final position). (b) the mean absolute distance $\frac{1}{N}\sum|dist(\mathbf{x}, f)|$ versus iteration number.

zero set of the polynomial is defined by $\mathbf{m}^{\mathrm{T}}\mathbf{a} = 0$. Thus after substituting $\mathbf{m} = V^{-1}\mathbf{m}'$, the transformed coefficients are $\mathbf{a}' = V^{-\mathrm{T}}\mathbf{a}$.

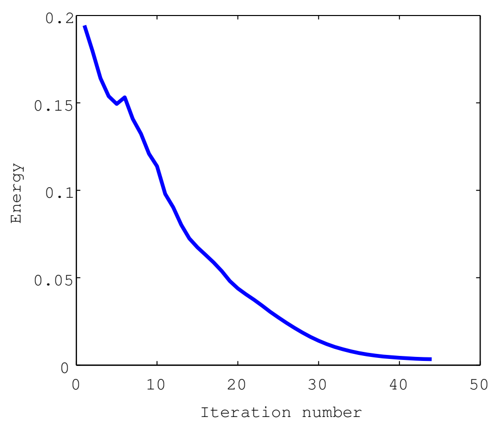
Taubin and Cooper [TC92] prove the existence of IP transformation matrix V for extracting the invariants from IP coefficients. However unfortunately computational implementation of IP transformation is not explicitly descried. Tarel *et al.* [TCC98] show a tensor-based transformation that represents 3D IP in 4D homogeneous form, and then transform the tensor elements. The method suffers from too time-consuming computation for tensor transformation, since for transforming a r-degree 3D IP, it need to transform 4^d tensor elements.

5.4.2 Proposed IP Transformation Method

This section presents our symbolic computational rule based on the Taubin and Cooper's theory [TC92]. Our method is a class of incremental calculation beginning from the transformation for first-degree monomial, *i.e.*, just Affine transformation, until obtaining one for n-th degree monomial. Thus, IP model transformation is computationally efficient, in the case that the size of modeled data set is greatly larger than the number of IP coefficients, since obviously transforming a few of IP coefficients is superior to transforming whole data points with respect to computational cost.



(a)



(b)

Figure 5.3: IP registration with surface 3D points: (a) registration process between "bunny" IP model and points (blue points: initial position; green points: selected mid-step positions; red points: final position). (b) the mean absolute distance $\frac{1}{N}\sum |dist(\mathbf{x}, f)|$ versus iteration number.

5.4.3 Pure Rotation

First let us explain some notations for polynomial operation which is first introduced by Taubin and Cooper [TC92]. Let a coefficient a_{ijk} of an IP in Eq. (2.1) be presented as $\frac{\Phi_{ijk}}{i!j!k!}$, and a vector $\Phi_{[l]} = (\frac{\Phi_{l00}}{\sqrt{l!0!0!}} \; \frac{\Phi_{l-1,1,0}}{\sqrt{(l-1)!1!0!}} \; \cdots \; \frac{\Phi_{00l}}{\sqrt{0!0!l!}})^{\mathrm{T}}$ is according to the l-th form of IP. And an operation on this vector is defined as: $\Phi_{[l,m]} = \Phi_{[l]} \star \Phi_{[m]}^{\mathrm{T}}$, where $\star$ represents the classic matrix multiplication with the difference that the individual element-wise multiplications are performed according to the rule $\frac{\Phi_{ijk}}{\sqrt{i!j!k!}} \star \frac{\Phi_{abc}}{\sqrt{a!b!c!}} = \frac{\Phi_{i+a,j+b,k+c}}{\sqrt{i!j!k!}\sqrt{a!b!c!}}$. For example,

$$\Phi_{[1]} = (\frac{\Phi_{100}}{\sqrt{1!0!0!}} \; \frac{\Phi_{010}}{\sqrt{0!1!0!}} \; \frac{\Phi_{001}}{\sqrt{0!0!1!}})^{\mathrm{T}}, \tag{5.14}$$

and

$$\Phi_{[1,1]} = \Phi_{[1]} \star \Phi_{[1]}^{\mathrm{T}} = \begin{pmatrix} \Phi_{200} & \Phi_{110} & \Phi_{101} \\ \Phi_{110} & \Phi_{020} & \Phi_{011} \\ \Phi_{101} & \Phi_{011} & \Phi_{002} \end{pmatrix}. \tag{5.15}$$

In [TC92], it was pointed out that under a non-singular coordinate transformation A, *e.g.*, $\Phi'_{[l]} = A_{[l]}\Phi_{[l]}$, the transformed coefficient matrix is given by

$$\Phi'_{[l,m]} = A_{[l]}^{-\mathrm{T}} \Phi_{[l,m]} A_{[m]}^{-1}, \tag{5.16}$$

where $A_{[l]}$ is a non-singular $h_l \times h_l$ transformation matrix (assuming h_l be the number of monomial in the l-th form), and $A_{[m]}$ is in the same way. From this equation, we know that if $A_{[l]}$ and

Table 5.1: 1k points

	ICP	ICP with KD tree	IP
CPU Time (sec)	11.9522	0.9128	0.4338
Memory	2K points	2K points + KD tree	1K points + 165 coef.

Table 5.2: 2.5k points

	ICP	ICP with KD tree	IP
CPU Time (sec)	30.5396	6.5570	0.5071
Memory	5K points	5K points + KD tree	2.5K points + 165 coef.

Table 5.3: 10k points

	ICP	ICP with KD tree	IP
CPU Time (sec)	123.0137	35.1214	1.0728
Memory	20K points	20K points + KD tree	10K points + 165 coef.

$A_{[m]}$ are given, then a linear relationship between $\Phi'_{[l,m]}$ and $\Phi_{[l,m]}$ can be found out; that is, the element-wise correspondence can be linearly expressed as

$$\Phi'_{ijk} = \sum_{\beta;i,j,k\leq l+m} a_{\alpha\beta}\Phi_{ijk} = \mathbf{a}_\alpha^{\mathrm{T}}\Phi_{[l+m]}. \tag{5.17}$$

Then since all of the elements of $\Phi_{[l+m]}$ are contained in $\Phi_{[l,m]}$, and all of the elements of $\Phi'_{[l+m]}$ are contained in $\Phi'_{[l,m]}$, a new linear correspondence can be constructed between $\Phi_{[l+m]}$ and $\Phi'_{[l+m]}$ by arranging the necessary elements in the right order into $\Phi'_{[l+m]}$ as:

$$\begin{aligned} \Phi'_{[l+m]} &= (\Phi'_{l+m,0,0}, \Phi'_{l+m-1,1,0}, \Phi'_{l+m-2,2,0}, \ldots)^{\mathrm{T}} \\ &= (\mathbf{a}_1^{\mathrm{T}}\Phi_{[l+m]}, \mathbf{a}_2^{\mathrm{T}}\Phi_{[l+m]}, \mathbf{a}_3^{\mathrm{T}}\Phi_{[l+m]}, \ldots)^{\mathrm{T}}. \\ &= A_{[l+m]}\Phi_{[l+m]}, \end{aligned} \tag{5.18}$$

if the $(l+m)$-th transformation matrix is constructed as: $A_{[l+m]} = [\mathbf{a}_1^{\mathrm{T}}, \mathbf{a}_2^{\mathrm{T}}, \mathbf{a}_3^{\mathrm{T}}, \ldots]^{\mathrm{T}}$.

As a simple example, the elements of $\Phi_{[2]}$ ($= (\frac{\Phi_{200}}{2}\ \Phi_{110}\ \Phi_{101}\ \frac{\Phi_{020}}{2}\ \Phi_{011}\frac{\Phi_{002}}{2})^{\mathrm{T}}$) are contained in $\Phi_{[1,1]}$ shown in (5.15), and the elements of $\Phi'_{[2]}$ are contained in $\Phi'_{[1,1]}$. From the equation $\Phi'_{[1,1]} = R_{[1]}\Phi_{[1,1]}R_{[1]}^{\mathrm{T}}$, (supposing $R_{[1]}$ is a pure rotation matrix) we can find out a relationship of a linear expansion for the first element of $\Phi'_{[1,1]}$ as: $\Phi'_{200} = \mathbf{r}_1\Phi_{[1,1]}\mathbf{r}_1^{\mathrm{T}}$, where $\mathbf{r}_1$ is the first row of $R_{[1]}$. Then since all the elements in $\Phi_{[1,1]}$ is contained in $\Phi_{[2]}$, we can find a linear relationship $\mathbf{r}_1\Phi_{[1,1]}\mathbf{r}_1^{\mathrm{T}} = \mathbf{a}_1^{\mathrm{T}}\Phi_{[2]}$. Thus $\mathbf{a}_1$ can be viewed as the first row of $A_{[2]}$.

Therefore, transformation matrix for different forms can be calculated in an incremental manner as:

```
Algorithm
1)  Initialization: given $A_{[1]}$;
2)  $\Phi'_{[1,1]} = A_{[1]}^{-\mathrm{T}}\Phi_{[1,1]}A_{[1]}^{-1} \longrightarrow A_{[2]}$;
3)  $\Phi'_{[1,2]} = A_{[1]}^{-\mathrm{T}}\Phi_{[1,2]}A_{[2]}^{-1} \longrightarrow A_{[3]}$;
        $\vdots$
n)  $\Phi'_{[1,n-1]} = A_{[1]}^{-\mathrm{T}}\Phi_{[1,n-1]}A_{[n-1]}^{-1} \longrightarrow A_{[n]}$.
```

Here, $\longrightarrow$ represents constructing the transformation matrix for next step, after finding out the linear mapping relationship between $\Phi'_{[1,l]}$ and $\Phi_{[1,l]}$. Note if the initial argument $A_{[1]}$ represents the pure rotation, then $A_{[l]}$ is an orthogonal matrix (see [TC92]), and thus Eq. (5.16) degenerate to

$$\Phi'_{[l,m]} = A_{[l]}\Phi_{[l,m]}A_{[m]}^{\mathrm{T}}.$$

5.4.4 Affine Transformation

By using homogeneous coordinates, an IP of n variables in Euclidean space can be described in projective space by a corresponding homogeneous IP of $n+1$ variables. To convert a ternary (*i.e.* 3D) IP of degree d

$$f_{3D}^{d}(x,y,z) = \sum_{0\le i,j,k;i+j+k\le d} a_{ijk}x^i y^j z^k \tag{5.19}$$

into its homogeneous representation, a new component $t = 1$ is added to the 3D IP as:

$$f_{4D}^{d}(x,y,z,t) = \sum_{0\le i,j,k,l;i+j+k+l=d} a_{ijk}x^i y^j z^k t^l. \tag{5.20}$$

Therefore a homogeneous polynomial corresponding to a 3D IP of degree d is a form of degree d in a 4D IP. Thus the procedure mentioned in the last section can be used to transform the homogeneous IP (4D IP). The different is that the incremental algorithm starts from a 4×4 affine transformation matrix $A_{[1]}$, *e.g.*, in the Euclidean case

$$A_{[1]} = \begin{pmatrix} R_{3\times3} & \mathbf{t} \\ \mathbf{0}_{1\times3} & 1 \end{pmatrix},$$

where $R_{3\times3}$ and $\mathbf{t}$ are pure 3D rotation matrix and translation vector. As a result, a 3D IP of degree d can affine-transformed only by $A_{[d]}$, once it is worked out. Furthermore, the incremental scheme can be modified for saving the computational cost, *e.g.*, for calculating $A_{[9]}$ we can select the incremental order as:
$A_{[1]} \to A_{[2]} \to A_{[4]} \to A_{[8]} \to A_{[9]}$.

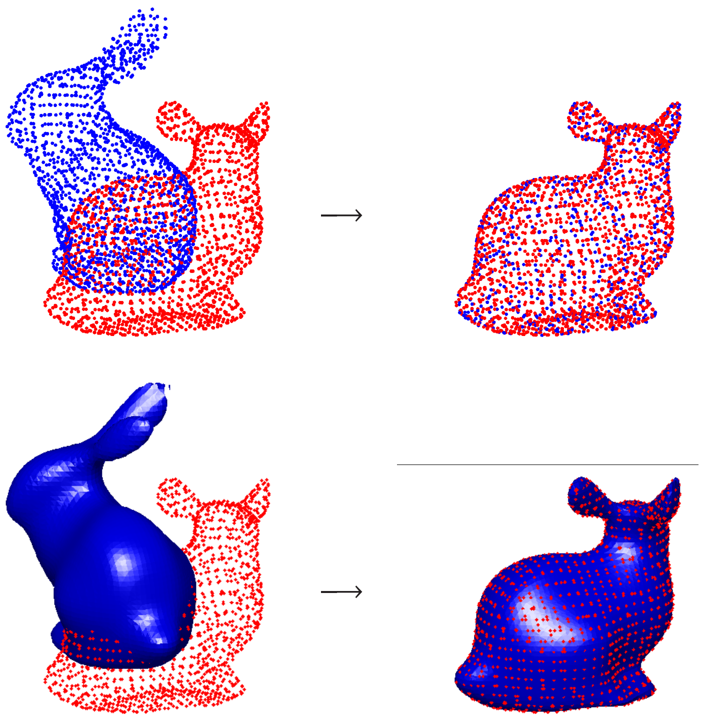


Figure 5.4: Illustration for registration. top row: register a point set to model point set by ICP method; bottom row: register an IP to the model point set by our method

5.5 Experimental Results

The proposed registration method has been applied to a variety of synthetic and real data sets. First, for all the experimental results shown in this section, let us set some pre-conditions as: 1) the time step t is simply set with 1; 2) the IP models are scaled according to the target object; 3) Registration is performed with a modern PC as: Intel core 2 CPU with 2.4 GHz and 2 GB memory, and the algorithm is implemented with Matlab 7.

5.5.1 Registration from Partial Dataset to IP

Fig. 5.2 and Fig. 5.3 show the registrations in general case that registers two synthetic data sets to "bunny" object. The synthetic data sets are selected from the "bunny" slice (Fig. 5.2 (a)) and surface (Fig. 5.3 (b)). The initial positions are shown with blue points in each figure. By carrying out the registration algorithm described in Section 3, they are successfully registered to "bunny" IP model with the final positions shown with red points. The mean absolute distance of Eq. (5.5) of the points for each registration versus the iteration number is plotted in (b) of each figure. The consumed CPU time are 10.3 *ms* and 23.1 *ms* respectively.

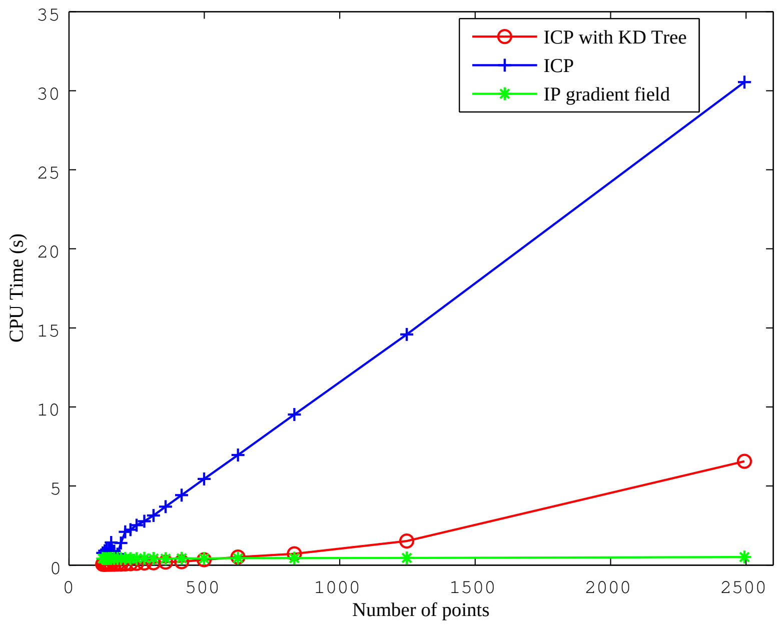


Figure 5.5: Comparison on CPU time of ICP method, ICP with KD tree and our method.

5.5.2 Comparison between ICP Method and Our Method

Let us show the effectiveness of our method by comparing the computational performance with three methods: standard ICP method [BM92], ICP method with KD tree [Zha92] and our method. Let us setup experimental assumptions as: 1) two datasets, the target and the source, with same number of points are extracted from same object; 2) the target has been rotated 90 degrees around y-axis and translated with length of 10% of the scale of the object along y-axis, as shown in Fig. 5.4; 3) the computations for KD tree and IP modeling are considered into the performance evaluation; 4)The same registration task will be done, and all the methods will stopped until satisfy the same accuracy.

As illustrated in Fig. 5.4, ICP method and ICP with KD tree register the discrete target model to the discrete source model, in the top of Fig. 5.4; our method registers the IP of the target to the source model, in the bottom of Fig. 5.4.

We evaluate the performance on computational times versus the increasing data points, as shown in Fig. 5.5. Also we compare the compuational time and memory in Tab. 5.1, Tab. 5.2 and Tab. 5.3 which show the registration tasks of 1k points, 2.5k points and 10k points both for target and source datasets.

5.6 Application: 6-DOF Pose Estimation for a Single US Image

The second application is 6-DOF pose estimation of a single Ultrasound (US) image for medical purpose. In this application, we extend our IP registration method using gradient field as presented in Chapter 5 to a 2D-3D registration version. The different is that the planar feature points are extracted from US images.

5.6.1 Background

Real-time pose estimation of a free-hand Ultrasound (US) image without any position sensor is desirable for diagnostics and image guidance, but it suffers from poor image quality as a result of processing. The work confronts this problem by proposing a new 6-DOF pose estimation method based on a fast registration process making use of 3D implicit polynomials (IP). The proposed registration method has some main advantages over the traditional methods. First, our formulation is based on minimization of energy functional derived from IP gradient flow, and thus it is more efficient than traditional registration because it eliminates the cost for calculating point-wise correspondences. Second, the efficient calculation benefits from the property of IP having few coefficients, which means that both the gradient field and its transformation can be calculated in an extremely light manner. Third, applying a real-time US image pose estimation, we demonstrate the capabilities of overcoming the limitations of unconstrained freehand US data, resulting in robust and fast pose estimation.

To support medical diagnosis, various imaging modalities, such as CT scan, MRI, PET, and ultrasound (US), are widely used in clinics. Among these modalities, US has beneficial characteristics such as free-hand manner, non-invasiveness, compactness, low cost, and synchronization of operations and imaging. Thus US is attractive for assistance with surgical operations and real-time diagnosis of problems with the circulatory system, abdomen, breast, prostate gland, etc.

However, US images are notorious for the poor image quality, due to speckle noises, low signal-to-noise ratio, occlusions, and uniform brightness. And field of view (FOV) in US imaging is very limited; in severe cases, only 2D cross-sectional images are obtained. These aspects confuse the doctors in making the right decisions for diagnosis.

In order to solve these issues, some recent literature advocates the fusion of modality techniques. For example, before the surgical operation, 3D models of target parts are obtained by rich but time-consuming modalities such as CT, MRI, and PET. By superimposing US images obtained during the operation on these 3D models, the result will provide rich information to help a doctor's diagnosis. To achieve this, the key for superimposing is to estimate the pose of US images related to the images derived from multiple modalities.

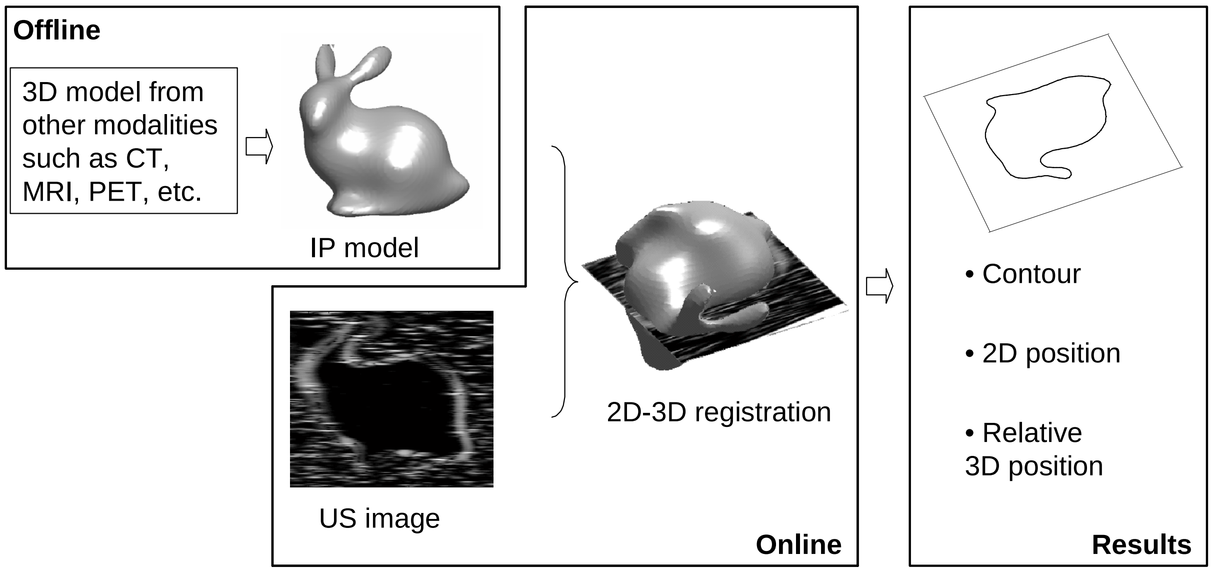


Figure 5.6: Method outline: offline, we model the boundary of the target object, which is extracted from 3D imaging techniques such as CT, MRI, and PET, with an implicit polynomial (IP). The online process registers the 3D IP model and 2D US image using gradient flows of the IP. Then, as results, the image contour, the 2D position, and the relative 3D position of the IP model are extracted.

To do this, the methods in [PME*99, TRSB94] bind the optical position sensors to a US probe, and measure the relative US position to 3D models; the methods in [LMPT04, RPMA01, WRN05] estimate the relative positions according to the image features of US images and preoperative 3D models. For enhancing robustness, the methods in [BPKH05, PBC*06, X, 07] combine the information from position sensors and image features. Although each of the methods has its effectiveness, they suffer from heavy computation and thus cannot work in real time.

To overcome the time-consuming issue, we present a method to register the 3D model and 2D US image using boundary information, which is independent of the types of modalities. As illustrated in Fig. 5.6, our method represents 3D model with an implicit polynomial (IP) in an advance offline process. In an online process, for example, during a surgical operation, it aligns a 3D IP model to a 2D US image based on the energy minimization manner. Then the desired information regarding the 2D contour, the 2D position, and the relative 3D pose of the US probe is obtained.

Our method has three significant advantages: 1) Because our method is based on the registration technique, no special equipment, such as position sensor, is required. 2) Differentiability of IPs enables the application of gradient-based minimization technique. Thus our method achieves fast registration. 3) Since IP representation needs few parameters and is based on the implicit form, both the calculation of gradient flow and its transformation for the minimization are extremely fast.

5.6.2 Modeling

Since volume data are widely used in the medical imaging and for robust modeling, in this work we propose to use volumic linear constraints (different to the constraints described in chapter 2). Suppose the model $\mathcal{M}$ represents a closed surface that defines two 3D regions in the 3D space $\mathcal{R}_\Omega$, namely the inner region $\mathcal{R}_\mathcal{M}$ enclosed by the model $\mathcal{M}$ and the outer region $\mathcal{R}_\Omega \setminus (\mathcal{R}_\mathcal{M} \bigcup \mathcal{M})$. The model is represented implicitly by its distance transform, *i.e.*, the signed distance function,

$$\Phi(\mathbf{x}) = \begin{cases} 0, & \mathbf{x} \in \mathcal{M} \\ +d(\mathbf{x}, \mathcal{M}), & \mathbf{x} \in \mathcal{R}_\mathcal{M} \\ -d(\mathbf{x}, \mathcal{M}), & \mathbf{x} \in \mathcal{R}_\Omega \setminus (\mathcal{R}_\mathcal{M} \bigcup \mathcal{M}) \end{cases}, \tag{5.21}$$

where $\mathbf{x} = (x\ y\ z)$ is the location of one data point in Cartesian coordinates, and $d(\mathbf{x}, \mathcal{M}) = \min_{\mathbf{x}_\mathcal{M} \in \mathcal{M}} \|\mathbf{x} - \mathbf{x}_\mathcal{M}\|$ is the Euclidean distance from $\mathbf{x}$ to the model consisting of the points $\mathbf{x}_\mathcal{M}$. In this work, we adopt an implicit polynomial (IP) to approximately satisfy Φ.

5.6.3 Registration for US Images

For the task of pose estimation of the US probe, the US images from the 2D probe that image the organ slice boundaries are usually expected to be matched with the corresponding planar intersection of the 3D model.

Furthermore, US images are heavily noise-contaminated. Since precise extraction of the organ's boundary is therefore very difficult, maybe impossible, simple boundary-based registration described in the general case is not sufficient for US images. But fortunately, the organ's inside region is clearly seen in US images as shown in Fig. 1. These two points are key for improving the general method for US images.

For solving the first point, we use a Gaussian smoothed edge indicator defined by [CLMC92]. Let $I(\mathbf{x})$ be a 2D US image function in 3D space, and pixel position $\mathbf{x}$ ($\in \mathcal{R}^3$) be in the 3D plane Ω. Let $g(\mathbf{x})$ be the indicator defined as follows:

$$g = \frac{1}{(1 + |\nabla G_\sigma * I|/k)^2}, \tag{5.22}$$

where G_σ is a Gaussian filter with standard deviation σ, and $*$ denotes convolution.

For making use of the second point, we compose the energy function with combining a boundary constraint $\mathcal{L}(\mathbf{x})$ and an inside constraint $\mathcal{A}(\mathbf{x})$, such as

$$\mathcal{E}(\mathbf{x}) = \alpha\mathcal{L}(\mathbf{x}) - \beta\mathcal{A}(\mathbf{x}), \tag{5.23}$$

where α and β are constants, and the terms $\mathcal{L}(\mathbf{x})$ and $\mathcal{A}(\mathbf{x})$ are defined by

$$\mathcal{L}(\mathbf{x}) = \int_{\Omega} \delta(dist(\mathbf{x}, \Psi(\mathbf{x})) \cdot g(\mathbf{x})) dist^2(\mathbf{x}, \Psi(\mathbf{x})) d\Omega, \tag{5.24}$$

and

$$\mathcal{A}(\mathbf{x}) = \int_{\Omega} H(-dist(\mathbf{x}, \Psi(\mathbf{x})) \cdot g(\mathbf{x})) dist^2(\mathbf{x}, \Psi(\mathbf{x})) d\Omega, \tag{5.25}$$

respectively, where δ is the univariate Dirac function, H is the Heaviside function, and Ω corresponds to the image plane.

Let us explain the meaning of these terms. Minimize the energy functional $\mathcal{E}$ in (5.23), which is equivalent to minimizing $\mathcal{L}$ in (5.24), and maximize $\mathcal{A}$ in (5.25). First, the energy function with respect to the boundary $\mathcal{L}$ is equivalent to the integral of edge indicator values along the intersected curve between IP and US image plane because the IP model should be converged around an edge-like part in US images. Second, the energy function with respect to the inside region $\mathcal{A}$ is the integral of edge indicator values over the inner region surrounded by the intersected curve. It is expected to be as large as possible. To summarize, minimizing $\mathcal{L}$ and $-\mathcal{A}$ achieves the registration.

By calculus of variations, the Gateaux derivative (first variation) of the functional $\mathcal{E}$ in (5.23) can be approximately written as

$$\frac{\partial \mathcal{E}}{\partial \mathbf{x}} \approx (\alpha\delta(dist \cdot g) - \beta H(-dist \cdot g))G. \tag{5.26}$$

The steepest descent process for minimization of the functional $\mathcal{E}$ is the following gradient flow:

$$\frac{\partial \mathbf{x}}{\partial t} = -(\alpha\delta(dist \cdot g) - \beta H(-dist \cdot g))G. \tag{5.27}$$

The Dirac function δ in (5.24) and the Heaviside function H in (5.25) are simplely defined as:

$$\delta(x) = \begin{cases} 0, & |x| > \kappa \\ \frac{1}{2\kappa}[1 + \cos(\frac{\pi x}{\kappa})], & |x| \le \kappa \end{cases} \tag{5.28}$$

and

$$H(x) = \begin{cases} 0, & x < 0 \\ x + x^2/2, & x \ge 0 \end{cases} \tag{5.29}$$

respectively, where κ is constant and practically set with $\kappa = 2$ in our experiments.

5.6.4 Experimental Results

First, for all the experimental results shown in this section, let us set the pre-conditions as: 1) the time step t is simply set with 1; 2) the IP models are scaled according to the US

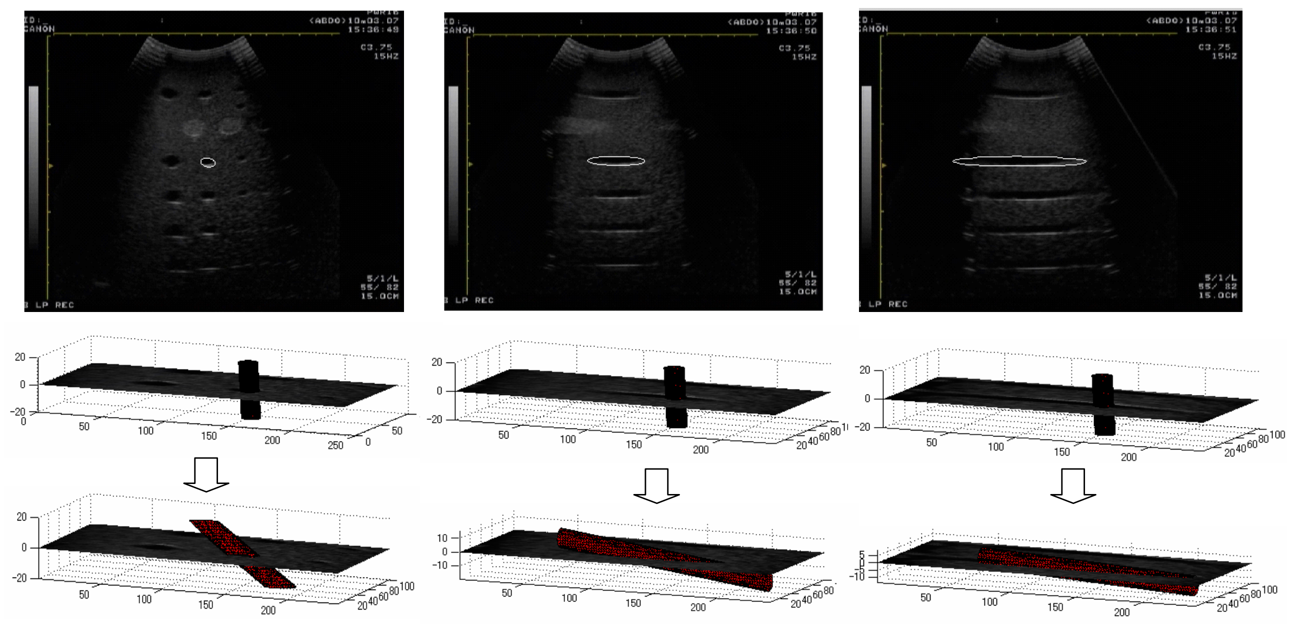


Figure 5.7: Pose Estimation for phantom ATS514. First row: intersection curve of IP and US image (white points); Second row: initial positions of IP; Third row: final positions of IP.

images; 3) α and β in (5.23) are set with about 0.9 and 0.1 respectively for US image case; 4) US images (640×480 pixel image) are measured by TOSHIBA SONOLAYER SSA-260A with two types of 2D probe: TOSHIBA PLF-703NT (7.5MHz) and TOSHIBA PVF-375MT (3.75MHz); 5) Registration is performed with a modern PC such as Intel core 2 CPU with 2.4 GHz and 2 GB memory, and the algorithm is implemented with Matlab 7.

Fig. 5.7 shows a US image case, where the US images are obtained by measuring phantom ATS514 [Pha]. Then we use a cylinder-like IP model (2-degree IP), according to the shape of the phantom, to register the US images. The initial position for each case is set as follows: the cylinder-like IP model cross the US image plane with an angle of 85°, and it is placed near to one of the hole-like areas of the images, as shown in the second row of Fig. 5.7. The registration result of each case is shown at each column, respectively; the first row shows the intersection curve between the IP and the US plane at the final iteration, and the second and third row show the initial and final position of each case. For each case, the consumed CPU time is within 30 *ms*.

In Fig. 5.8, we show the iterations in detail for dealing with one single US image, frame 21#. For each iteration, we plot the cross section of IP and the US image and the IP motions for the registration. In Fig. 5.12(a), we plot the convergence of the energy function versus iteration number.

Fig. 5.9 shows another result of the US images that are obtained by measuring a duck toy made of rubber in the cistern shown in Fig. 5.9 left. We modeled the duck object with an 8-degree

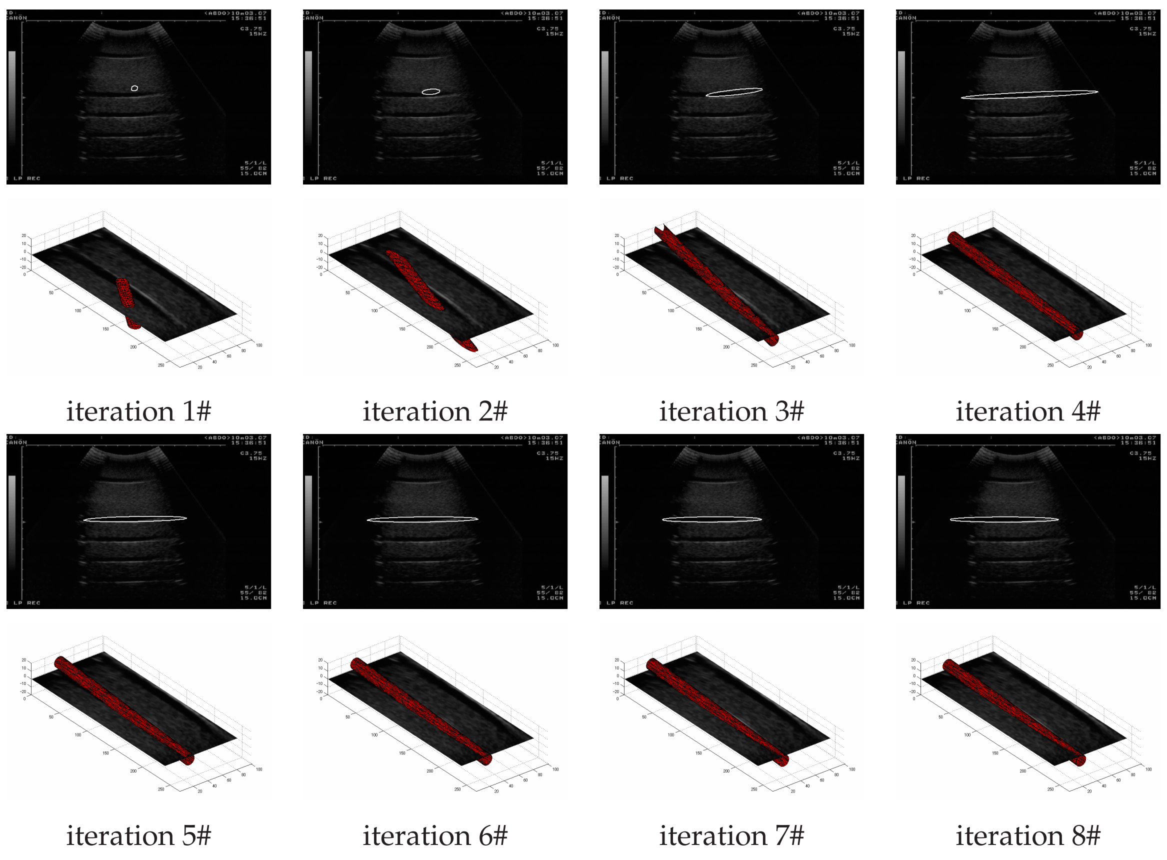


Figure 5.8: Pose Estimation for one single US image of frame 21#.

IP shown in Fig. 5.9 right. The initial position for each case is set near to the object boundary. The registration result of a case is shown in Fig. 5.10, where the cross section contours for each iteration are plotted. The motion of IPs for the registration are shown in Fig. 5.11. Finally Fig. 5.12(b) shows the energy convergence. For this case, the consumed CPU time is within 180 *ms*.

5.7 Summary

An registration method has been developed, whose efficiency benefits from: 1) Thanks to IP gradient flows which approximately defines the Euclidean distance for the registration and can be fast calculated due to the characteristics of IP; 2) two-step minimization, non-rigid descent-based minimization and rigid ICP-based minimization is adopted for fast registration; 3) an efficient transformation of IP coefficients is proposed for large-scale problem. As result, IP registration performs much more efficiently than the conventional method such as ICP registration, since it successfully avoid the time-consuming calculation of point-wise correspondence.

This method has the limitation that it is only appropriate for the target object with simple

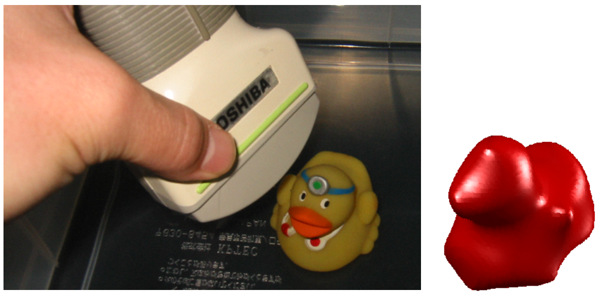


Figure 5.9: Left: photograph of measuring a duck toy. Right: IP model of duck object

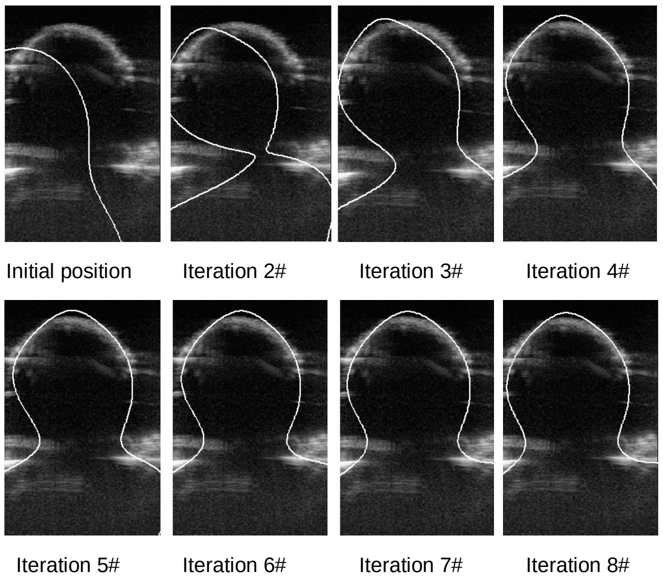


Figure 5.10: Contour crosssection between IP and US image of 188#.

shape which can be represented by an IP accurately or it would be suit for the coarse registration for the complex target object, since IP can not model a complex shape with fine representation. This registration method will be robust in noisy condition or the condition with occlusions because of IP's characteristic but not suitable to fine registration. The method is remarkable especially for solving the large-scale problem.

The proposed method is extend to a novel method for the application of pose estimation method for US images. US image feature information is considered of the acceleration of the registration. We demonstrated the performance of the proposed algorithm using real US images, and in particular we showed its robustness against the presence of weak boundaries and strong noise. Our algorithm works in real time (at most 180 *ms* in our conducted experiments), and therefore has the potential for various real-time US diagnosis and image guidance.

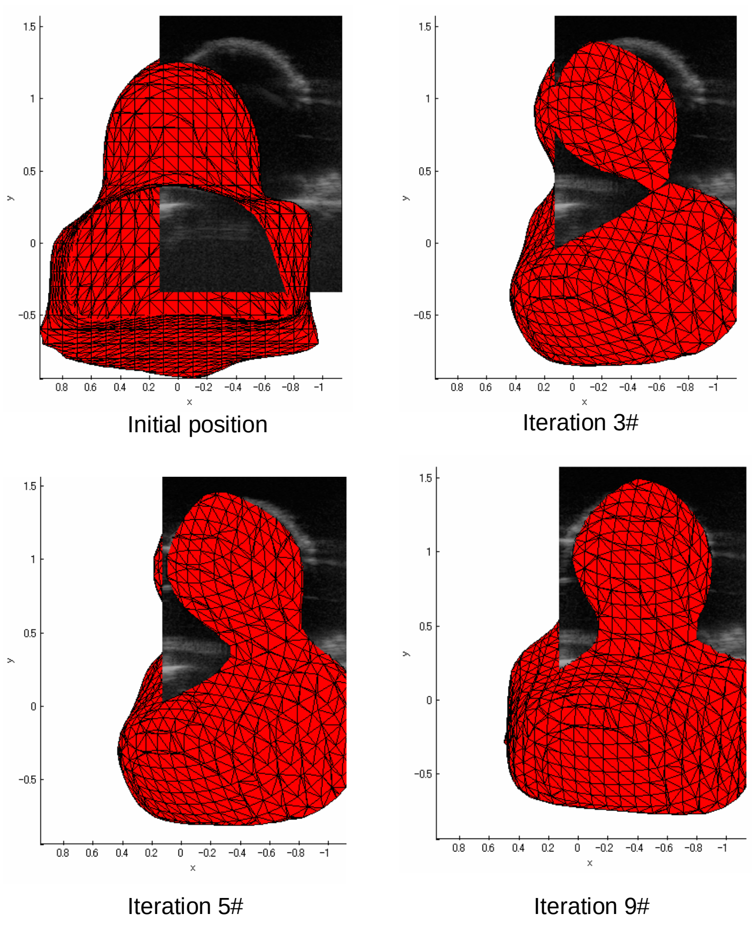


Figure 5.11: Registration vs iteration number for estimating 188# frame.

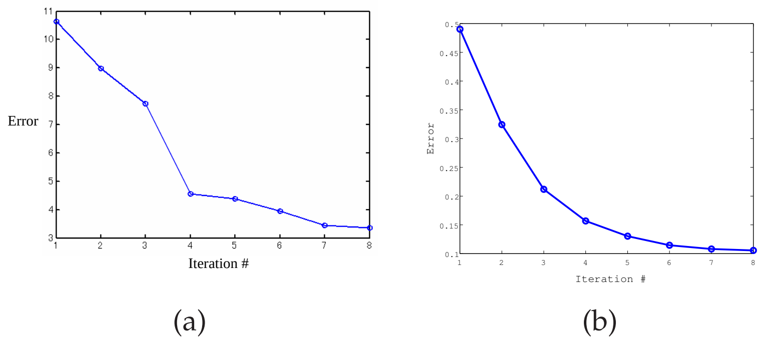


Figure 5.12: Energy Error vs iteration number (a) 21# frame for phantom ATS514 (b)188# frame for duck toy.

Chapter 6

Conclusion

6.1 Summary

The ultimate purpose of this dissertation is to provide a powerful shape representation for the vision applications. This purpose is strongly motivated by the requirement on high performance in specific computer vision system. To accomplish this, one of the most suitable shape representation, the algebraic representation of implicit polynomials, has been studied and also enhanced. We developed three methods of using implicit polynomials to extend the IP's capabilities towards vision applications and thus IP's potential is rediscovered.

A 3D surface segmentation method

The first method is designed for segmenting 3D surface, as described in Chapter 3, to achieve a IP-segment representation for 3D models. The method is based on a cut-and-merge approach. Two cutting procedures adopt low-degree IPs to divide and fit the surface segments simultaneously, while avoiding generating high-curved segments. A merging procedure merges the similar adjacent segments to avoid over-segmentation. To prove the effectiveness of this segmentation method, we open up a new vista for registration of range images.

An adaptive and stable method for IP fittings

The second method is a novel method to solve the present problems on IP fitting methods, as described in Chapter 4. It can adaptively determine the moderate degree for IP in an incremental fitting procedure, and simultaneously improve the numerical stabilities in fitting. To achieve this, we developed the incremental process in which it keeps solving upper triangular linear systems of different degrees, and effectively employ the incrementability of the Gram-Schmidt

QR decomposition in the implementation. Simultaneously based on the beneficial property of the upper triangular matrix R in QR decomposition that eigenvalues are the elements on its main diagonal, the fitting instability can be checked out easily and quickly from the diagonal values. By selectively utilizing the RR constraints, our method not only improves global stability, but also maintains local accuracy.

A registration method of using IP gradient field

Based on the IP gradient field can be generated stably, the third method has been developed towards registration problem for the computer vision system in noisy condition, as described in Chapter 5. To enhance the registration performance, we use IP gradient flows to replace the Euclidean distance for the registration and thus successfully avoid the time-consuming calculation of point-wise correspondence. For further acceleration, we adopted two-step minimization, non-rigid descent-based minimization and rigid ICP-based minimization, and an efficient transformation of IP coefficients is proposed for large-scale problem. The effectiveness has been proved by a proposed application method for real-time estimation for ultrasound images.

6.2 Contributions

We have explored the problem on IP representation and developed the novel applications with our new presentation methods. There are three Contributions of the thesis which towards the solutions of three computer vision problems.

1. Development of a segmentation method for encoding the IP features onto a segmented object surface.
2. Development of an adaptive and stable fitting method.
3. Development of a registration of using IP gradient field.

In the first contribution, the most important is that the segmentation assigns IPs to surface segments to provide the geometric/algebraic properties such as curvatures and Euclidean invariants. Furthermore, by adjusting the thresholds using different tolerance levels, IP surfaces with different accuracy and smoothness are obtained. Desired accuracy and smoothness can be freely selected so as to be suitable for the requirements for various vision applications in the future. We opens up the new vista for such applications as 3D object recognition, matching or registration. In particular, it makes contribution in the field of range image registration to achieve that registration approach for range images is non-iterative and independent to position initialization.

In the second contribution, it mainly solve the problems 1) the difficulty of determining the moderate degree for IP fitting, 2) numerical instability of fitting. The novel approach is based on an incremental procedure in which the computational cost is eliminated by the process in which we keep solving upper triangular linear systems in different degrees and effectively employ the incrementability of the Gram-Schmidt QR decomposition in the implementation. Second, based on the beneficial property of the upper triangular matrix R in QR decomposition that eigenvalues are the elements on its main diagonal, the fitting instability can be checked out easily and quickly from the diagonal values. That also gives the capability of having the selectivity for utilizing the RR constraints that improves not all but only the "bad" factors. As a result, the method achieves to adaptively determining the moderate degree according to complexity of various shapes, and not only improves global stability, but also maintains local accuracy. It brought a direct application that makes our segmentation can perform in an adaptive manner.

Finally, in the third contribution, a fast registration method of using IP gradient field has been proposed. Its high computational efficiency benefits from: 1) Thanks to IP gradient flows which approximately defines the Euclidean distance for the registration and can be fast calculated due to the characteristics of IP; 2) two-step minimization, non-rigid descent-based minimization and rigid ICP-based minimization is adopted for fast registration; 3) an efficient transformation of IP coefficients is proposed for large-scale problem. As result, IP registration performs much more efficiently than the conventional method such as ICP registration, since it successfully avoid the time-consuming calculation of point-wise correspondence. The effectiveness has been proved by an application of a novel pose estimation method for US images using gradient flows of IPs. In particular we showed its efficiency in the real-time performance and robustness against the presence of weak boundaries and strong noise. Therefore it has the potential for helping various real-time US diagnosis and image guidance.

6.3 Future Directions

We conclude this dissertation with a frank discussion of open problems and future improvements in our proposed method which we are interested in pursuing and would like to see pursued by other researchers.

Surface segmentation using curvature clustering

In our segmentation method, for example, for segmenting two partial overlapping objects in different coordinates, it cannot guarantee the segmentation result is Euclidean invariant, althought this shortcoming can be overcame, to some extend, by the IP's robustness to occlusion and missing data. As a improvement, we are considering that the segmentation can be conducted by combining the curvature information (curvature is easy and stable to be extracted by IP). That is, if we adopt a clustering method to cluster the low-curvature surface, we can obtain local Euclidean invariant segments even in the overlapping between two models.

Combining the topology information for range image registration

This is a future consideration for range image registration that, for more robust registration, we are considering to use the topology information to the matching process in registration. That is, for example, if we can construct the topology information for segmentation in a graph based structure, such as tree structure (each node denotes a segment), then we have a robust indexing for all segments and use this indexing we may obtain the robust correspondences between two range images.

Promoting the adaptive IP fitting method to a common method for parameter estimation

Another future direction is on our adaptive and stable method for IP fitting. Our IP fitting method has the generality for common linear least-squares problems that should be not only suited to estimate IP parameters. We are considering to promote the method to a common numerical method contributing to the field of numeric computation, since we believe that there are needs for an estimation method which can adaptively estimating the parameters with appropriate number using linear least-squares method.

Consideration on US image pose estimation

The final future direction is on the application of 6-DOF pose estimation for single ultrasound image. As the most pose estimation methods, the efficiency of our method depends on the initial position too, because of the ambiguity between a single US image and a 3D model. That is, in some cases, it has not an unique or robust correspondence between the single US image and 3D model. For solving this problem, we are considering to combine the pose information of using multi images derived from the continuous frames. In addition, we are currently developing to extract the rich feature information, such as the gradient information, from US image to match with IP gradients for enhancing the registration robustness.

Appendix A. Invariants of IP

Implicit polynomials are the most suitable modeling tools due to their global representation property. The Euclidean invariant of IP is one of the most important properties that have been attractive for the applications on object recognition and the matching problem in registration or pose estimation. Here let us briefly introduce the prior works and on how to calculate the invariants in Taubin's theory.

Taubin and Cooper [TC92] remark the previous works on invariant theory and give the methods invariant under Euclidean coordinate transformations. They also include some extensions to the affine and projective cases. Unel and Wolovich [UA98] introduced a pose estimation and object matching method based on complex representation of 2D implicit polynomials. Tarel and Cooper [TC00a] introduced a decomposition method for 2D implicit polynomials to extract invariants. They also summarize previous work on pose estimation. Tarel et al. [TWC03] introduced a decomposition method such that a quartic polynomial (in 2D only) can be written as the product of two conics plus a third conic. This decomposition leads to a pose estimation method. Unsalan [Uns07] introduces a set of invariants based on the 2D implicit polynomials under 6th degree, considering the Affine transformation.

Regardless various variations have been proposed for algebraic invariants, the basic and simplest method is the one proposed by Taubin and Cooper [TC92]. Let us briefly introduce their method here. Note we use the notations in Section 2.3.

A.1 Taubin and Cooper's Invariants

Based on the introduction of Euler's Theorem in Eq. (2.42), Taubin and Cooper first involve the computation of linear invariants, namely the computation of determinants.

Lemma A.1 *Let ϕ be a form of degree of even degree $d = 2k$. Then, the determinant $\Phi_{[k,k]}$ is linear invariant of weight* $w = 2\begin{pmatrix} n+d-1 \\ n \end{pmatrix}$

The next method is a consequence one. It applies to two form of the same degree under nonsingular linear transformations.

Corollary A.1 *Let ϕ and ψ be two forms of degree of even degree. Then, the coefficients of the homogeneous polynomial of two variables $\varepsilon(\theta_1, \theta_2) = |\theta_1 \Phi_{[k,k]} + \theta_2 \Psi_{[k,k]}|$, are combined invariants. Equivalently, the generalized eigenvalues of the pair of square matrices are absolute invariants of the pair of forms.*

This result can be extended, with basically the same manner, to combine absolute invariants of many forms. That means $\varepsilon(\theta_1, \ldots, \theta_r) = |\theta_1\Phi_1[k, k] + \ldots + \theta_r\Phi_r[k, k]|$ are rational invariants.

Note, if ϕ is not form of even degree, we can apply the construction of corollary A.1 to their squares ϕ^2. Therefore, even for forms of even degree, because the number of invariants obtained is equal to the number of coefficients. Then if ϕ is a form of degree d_1, and ψ is a form of $d_2 \neq d_1$, we can apply corollary A.1 to $\phi^{(d/d_1)}$ and $\psi^{(d/d_2)}$, where d is equal to twice the minimum common multiple of d_1 and d_2. Also this method can be applied to the cases of many forms.

Taubin and Cooper also give the computation of orthogonal invariants according to Eq. (2.43).

Corollary A.2 *Let ϕ be a form of even degree $d = 2k$. Then, the coefficients of the characteristic polynomial $|\Phi_{[k,k]} - \theta I|$ are orthogonal invariants, or equivalently, the eigenvalues of the square matrix $\Phi_{[k,k]}$ are orthogonal invariants.*

In the orthogonal case they also define a norm in the vector space of forms of degree d which is invariant under orthogonal transformation.

Corollary A.3 *Let $\phi(\mathbf{x}) = \Phi^t_{[d]}X_{[d]}(\mathbf{x})$ be a form of degree d. Then,*

$$\| \phi \|^2 = \Phi^t_{[d]}\Phi_{[d]} = \sum_{\alpha} \frac{1}{\alpha!}\Phi^2_\alpha \tag{6.1}$$

is an orthogonal invariant of ϕ.

With respect to join otrhogonal invariants of two or more forms, they also proposed a stronger result.

Corollary A.4 *Let ϕ and ψ be two forms of degree $d = j + k$, then the eigenvalues of the square matrix $\Phi_{[k,j]}\Psi_{[j,k]}$ are join orthogonal invariants of the pair.*

The proofs for the above corollaries can be found in [TC92]. Also they give some simple examples to prove the effectiveness for the invariants.

A.2 Another Derived Invariant

According to the Taubin and Cooper's theory, we also derived another invariant for the general cases (not only for even degrees) and with more robustness.

Corollary A.5 *Let ϕ be a form of degree $d = j + k$, then the eigenvalues of the square matrix $\Phi^T_{[k,j]}\Phi_{[k,j]}$ are join orthogonal invariants of the pair.*

Appendix B. An improvement for rendering IP-segmented surface

Although the 3D surface segmentation method described in Chapter 3 is not mainly designed for computer graphics (CG). It might be attractive for the researchers in CG who has interests in rendering a IP-segmented surface, since each IP segment provides a smooth surface.

The problem frustrating our results of the application for rendering is the errors existing on the boundaries between the segments. Especially when the threshold T_1 and T_2 are too tolerant, the fitting accuracy is accordingly decreased, and thus the boundaries connecting two neighbor segments become more coarse (see Fig. 3.11 (c)). To solve this problem, we propose a method based on smoothing the segment boundaries. First we define a metric from a point to the boundary of the segment which the point belongs to as:

$$d(\mathbf{x}) = \min_{\mathbf{b}_i \in Bdry} \| \mathbf{x} - \mathbf{b}_i \|, \tag{6.2}$$

where $\mathbf{x}$ is the point in a segment and $\mathbf{b}_i$ is the point at the boundary of the corresponding segment. Then by defining two weights w_1 and w_2 using the metrix $d(\mathbf{x})$, we can combine the current IP and its neighbor IP to one implicit function as

$$F(\mathbf{x}) = \frac{w_1 * f + w_2 * f_{neighbor}}{w_1 + w_2}, \tag{6.3}$$

where f is the IP function of the current segment, $f_{neighbor}$ is the IP function of a neighbor segment connected by the nearest boundary point. The weights w_1 and w_2 can be seen as two Gaussian-like functions respective to $d(\mathbf{x})$, *e.g.*,

$$\begin{aligned} w_1(d(\mathbf{x})) &= 0.5(1 - G(d(\mathbf{x}))) + 0.5, \\ w_2(d(\mathbf{x})) &= 0.5(G(d(\mathbf{x}))), \end{aligned} \tag{6.4}$$

where G is a Gaussian function with standard deviation σ simply defined as

$$G = e^{-d(\mathbf{x})/2\sigma^2}. \tag{6.5}$$

Fig. 6.1 gives the illustration for weights.

We show an example in Fig. 6.2. Fig. 6.2 (a) shows a segment result the same as that in Fig. 3.11 (d). Fig. 6.2 (b) shows the weight calculated by (6.3) according to the boundaries. Fig. 6.2 (c) shows the rendering result by directly rendering the segments without using (6.3). And Fig. 6.2 (d) shows the rendering result by using the weighted function in (6.3).

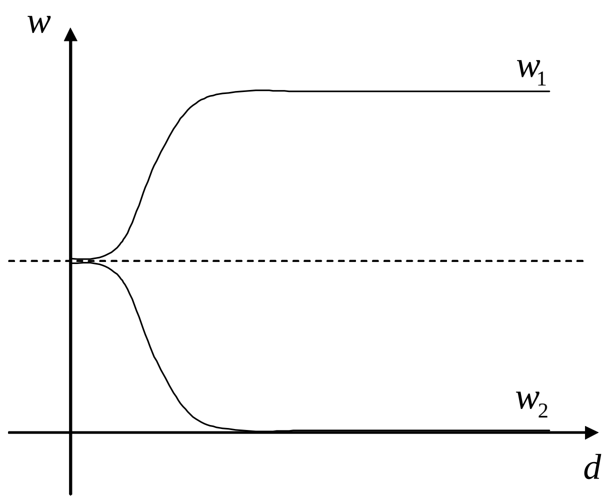


Figure 6.1: Weight functions

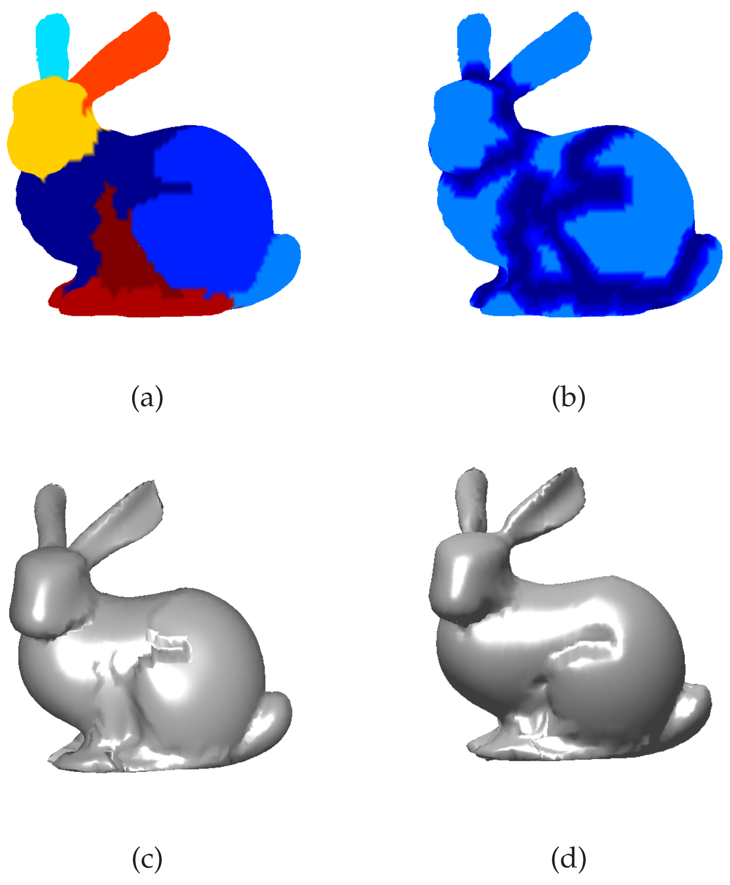


Figure 6.2: Rendering example by our improved method. (a) Segment result. (b) Weight calculation according to boundary. (c) Rendering without the weight function. (d) Rendering with the weight function.

Appendix C. Energy minimization for registration of using IP gradient flow

C.1 Quaternion Compuation for Derivative

In Chapter 5, for minimizing the energy function with conventional optimization tool, we have to calculate the derivative using the quaternion presentation. Let us first explain the quaternion expression for the rotation matrix.

Generally a rotation matrix can be represented with the parameters:

$$\mathbf{q} = (s, u, v, w). \tag{6.6}$$

Geometrically, when an object is rotated by θ around axis $\mathbf{m}$, there parameters have the following meaning.

$$s = \cos\frac{\theta}{2} \tag{6.7}$$

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = \sin\frac{\theta}{2}\mathbf{m} \tag{6.8}$$

therefore

$$s^2 + u^2 + v^2 + w^2 = \cos^2\frac{\theta}{2} + \sin^2\frac{\theta}{2} \parallel \mathbf{m} \parallel^2 = \cos^2\frac{\theta}{2} + \sin^2\frac{\theta}{2} = 1. \tag{6.9}$$

Thus, quaternion is interpreted as a combination of a scalar and a 3D vector. Here, we explain basis opterations of a quaternion. Suppose 2 quaternion as $\mathbf{p} = (a, \mathbf{u}^t)^t$ and $\mathbf{q} = (b, \mathbf{v}^t)^t$. The plus (or subtraction) operation of quaternions is defined as

$$\mathbf{p} + \mathbf{q} = \begin{pmatrix} a + b \\ \mathbf{u} + \mathbf{v} \end{pmatrix} \tag{6.10}$$

And the product operation is defined as

$$\mathbf{pq} = \begin{pmatrix} ab - \mathbf{u} \cdot \mathbf{v} \\ a\mathbf{v} + b\mathbf{u} + \mathbf{u} \times \mathbf{u} \end{pmatrix}. \tag{6.11}$$

Finally the norm of a quaternion is defined as

$$|\mathbf{p}| = \sqrt{a^2 + \mathbf{u}^t\mathbf{u}} \tag{6.12}$$

Conjugate and inverse quaternion of **q** is defined as follows, respectively

$$\mathbf{p}^* = \begin{pmatrix} a \\ -\mathbf{u} \end{pmatrix} \tag{6.13}$$

$$\mathbf{p}^{-1} = \frac{\mathbf{p}^*}{|\mathbf{p}|^2} \tag{6.14}$$

In the case of the rotation of angle θ around axis **u**, the rotated vector of **x** is generally described by using the quaternion **p** as

$$\begin{pmatrix} 0 \\ \mathbf{x}' \end{pmatrix} = \mathbf{p}\begin{pmatrix} 0 \\ \mathbf{x} \end{pmatrix}\mathbf{p}^{-1} \tag{6.15}$$

where **x**′ is the destination vector of **x**.

In the case of the same rotation $\mathbf{x}' = R\mathbf{x}$, the rotation matrix R is described by using these parameters as follow.

$$R = \begin{pmatrix} s^2 + u^2 - v^2 - w^2 & 2(uv + sw) & 2(uw - sv) \\ 2(uv - sw) & s^2 - u^2 + v^2 - w^2 & 2(vw + su) \\ 2(uw + sv) & 2(vw - su) & s^2 - u^2 - v^2 + w^2 \end{pmatrix} \tag{6.16}$$

While quaternion has 4 components, it is adequate to consider only 3 components since there are 3 independent variables. Based on Eq. 6.9 we are to deal with the parameter u, v and w. The parameter s is an induced variable of u, v and w, that means $s = s(u, v, w)$. Then, let us consider the first and second-order derivatives of s with respect to other parameters.

$$\begin{aligned} \frac{\partial s}{\partial u} = \frac{\partial}{\partial u}\sqrt{(1 - u^2 - v^2 - w^2)} &= \frac{1}{2}(-2u)\sqrt{(1 - u^2 - v^2 - w^2)} \\ &= -\frac{u}{\sqrt{(1 - u^2 - v^2 - w^2)}} = -\frac{u}{s} \end{aligned} \tag{6.17}$$

$$\frac{\partial s^2}{\partial u} = \frac{\partial}{\partial u}(1 - u^2 - v^2 - w^2) = -2u. \tag{6.18}$$

similarly,

$$\frac{\partial s}{\partial v} = -\frac{v}{s}, \ \frac{\partial s^2}{\partial v} = -2v, \ \frac{\partial s}{\partial w} = -\frac{w}{s}, \ \frac{\partial s^2}{\partial w} = -2w. \tag{6.19}$$

Therefore, the derivatives of the ratation matrix R with respect to u, v and w are written as follows.

$$\frac{\partial R}{\partial u} = \begin{pmatrix} 0 & 2(v - \frac{uw}{s}) & 2(w + \frac{uv}{s}) \\ 2(v + \frac{uw}{s}) & -4u & 2(s - \frac{u^2}{s}) \\ 2(w - \frac{uv}{s}) & 2(\frac{u^2}{s} - s) & -4u \end{pmatrix} \tag{6.20}$$

$$\frac{\partial R}{\partial v} = \begin{pmatrix} -4v & 2(u - \frac{vw}{s}) & 2(\frac{v^2}{s} - s) \\ 2(u + \frac{vw}{s}) & 0 & 2(w - \frac{uv}{s}) \\ 2(s - \frac{v^2}{s}) & 2(w + \frac{uv}{s}) & -4v \end{pmatrix} \tag{6.21}$$

$$\frac{\partial R}{\partial u} = \begin{pmatrix} -4w & 2(s - \frac{w^2}{s}) & 2(u + \frac{vw}{s}) \\ 2(\frac{w^2}{s} - s) & -4w & 2(v - \frac{uw}{s}) \\ 2(u - \frac{vw}{s}) & 2(v + \frac{uw}{s}) & 0 \end{pmatrix} \tag{6.22}$$

C.2 Non-linear Optimization

The current famous minimization tools are gradient descent method, Newton method, non-linear conjugate gradient (CG) method by giving out the integral energy gradients. CG method is often advocated with better convergent direction for the optimization, let us introduce the implementation of using CG method for our energy minimization problem.

In the previous section, we explained a descent gradient method. This method guarantees that the gradient at the current step is orthogonal to the gradient at the previous step, but it may be not orthogonal to other former gradients. So it is possible that the same gradient is computed during the minimization process. Namely, the minimization search cannot be done in wide area because it is done in the same direction over and over. This leads to possible inefficiency.

The conjugate gradient method avoids this kind of inefficiency. In this method, the conjugate gradient is calculated by modifying the descent gradient. This method guarantees that one any gradient is orthogonal to all other gradients.

In our implementation, we use the Fletcher-Reeves method and the Polak-Ribiere method [Polak71] [Jacobs77] [Stoer80]. These two methods are closely related and have well-known algebraic properties. Define A as a positive constant symmetry matrix of $n \times n$, which can be considered to be the Hessian matrix of our minimization function. Two vectors is considered: descent gradient $\mathbf{g}_i$ and conjugate gradient $\mathbf{h}_i$, which satisfy the following:

$$\mathbf{g}_{i+1} = \mathbf{g}_i - \lambda_i A\mathbf{h}_i, \;\; \mathbf{h}_{i+1} = \mathbf{g}_{i+1} + \gamma_i \mathbf{h}_i. \tag{6.23}$$

λ_i and γ_i are determined such that

$$\mathbf{g}_{i+1} \cdot \mathbf{g}_i = 0, \;\; \mathbf{h}_{i+1} \cdot A \cdot \mathbf{h}_i = 0. \tag{6.24}$$

From equations (6.23) and (6.24), λ_i and γ_i can be calculated as:

$$\begin{array}{rl} if & \mathbf{g}_i \cdot A\mathbf{h}_i \neq 0 \;\; \mathbf{h}_i \cdot A\mathbf{h}_i \neq 0, \;\; \lambda_i = \dfrac{\mathbf{g}_i \cdot \mathbf{g}_i}{\mathbf{g}_i \cdot A\mathbf{h}_i}, \;\; \gamma_i = -\dfrac{\mathbf{g}_{i+1} \cdot A\mathbf{h}_i}{\mathbf{h}_i \cdot A\mathbf{h}_i}. \\ otherwise & \lambda_i = 0, \; \gamma_i = 0. \end{array} \tag{6.25}$$

As a result, we can obtain the following equation.

$$\mathbf{g}_i \cdot \mathbf{g}_j = 0, \ \ \mathbf{h}_i \cdot A\mathbf{h}_j = 0 \ \ (i \neq j). \tag{6.26}$$

Eq. (6.23) is a type of Gram-Schmidt bi-orthogonalization; its first equation makes each $\mathbf{g}_i$ orthogonal to the previous $\mathbf{g}_{i-1}$, and the second makes each $\mathbf{h}_i$ be the conjugate gradient to the previous $\mathbf{h}_{i-1}$.

From Eq. (6.23) and (6.26), γ_i and λ_i can be found by

$$\gamma_i = \frac{\mathbf{g}_{i+1} \cdot \mathbf{g}_{i+1}}{\mathbf{g}_i \cdot \mathbf{g}_i} = \frac{(\mathbf{g}_{i+1} - \mathbf{g}_i) \cdot \mathbf{g}_{i+1}}{\mathbf{g}_i \cdot \mathbf{g}_i}, \tag{6.27}$$

$$\lambda_i = \frac{\mathbf{g}_i \cdot \mathbf{h}_i}{\mathbf{h}_i \cdot A\mathbf{h}_i} \tag{6.28}$$

Let P_i be a registration parameter that minimizes our objective function. Then we can acquire conjugate gradients inductively as follows:

$$\mathbf{g}_0 = - \nabla f(P_0), \ \ \mathbf{h}_0 = \mathbf{g}_0, \ \ P_1 = P_0 + \alpha \mathbf{h}_0. \tag{6.29}$$

$$\mathbf{g}_i = - \nabla f(P_i), \ \ \mathbf{h}_i = \mathbf{g}_i + \frac{\mathbf{g}_i \cdot \mathbf{g}_i}{\mathbf{g}_{i-1} \cdot \mathbf{g}_{i-1}} \cdot \mathbf{h}_i, \ \ P_{i+1} = P_i + \alpha \mathbf{h}_i \ \ (i = 1, 2, \ldots) \tag{6.30}$$

α is determined by 1-dimentional search in the direction $\mathbf{h}_i$ as shown in next. Thus the conjugate gradient can be obtained without the use of the Hessian matrix A.

References

[ABCO*03] Alexa M., Behr J., Cohen-Or D., Fleishman S., Levin D., Silva. C. T.: Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphic 9*, 1 (2003), 3–15.

[BaKS*05] Bustos B., a. Keim D., Saupe D., Schreck T., v. Vrani'c D.: Feature-Based Similarity Search in 3D Object Databases. *ACM Computing Surveys 37*, 4 (2005), 345–387.

[BKAA04] Boughorbel F., Koschan A., Abidi B., Abidi M.: Gaussian Energy Functions for Registration without Correspondences. *In 17th International Conference on Pattern Recognition (ICPR)* (2004).

[BLC00] Blane M., Lei Z. B., Cooper D. B.: The 3L Algorithm for Fitting Implicit Polynomial Curves and Surfaces to Data. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 22*, 3 (2000), 298–313.

[Blo97] Bloomenthal J.: *Introduction to Implicit Surfaces*. The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling, 1997.

[BM92] Besl P., McKay N.: A Method for Registration of 3-D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 14*, 2 (1992), 239–256.

[BPKH05] Blackall J., Penney G., King A., Hawkes D.: Alignment of sparse freehand 3-D ultrasound with preoperative images of the liver using models of respiratory motion and deformation. *Trans. on Medical Imaging 24*, 11 (2005), 1405–1416.

[CF01] Campbell R. J., Flynn P. J.: A survey of free-form object representation and recognition techniques. *Computer Vision and Image Understanding (CVIU) 81*, 2 (2001), 166–210.

[CLMC92] Catte F., Lions P.-L., Morel J.-M., Coll T.: Image selective smoothing and edge detection by nonlinear diffusion. *SIAM J. Numer. Anal. 29*, 1 (1992), 182–193.

[CLSB92] Champleboux G., Lavallee S., Szeliski R., Brunie L.: From Accurate Range Imaging Sensor Calibration to Accurate Model-Based 3D Object Localization. *Proc. IEEE Conference Computer Vision and Pattern Recognition (CVPR)* (1992), 83–89.

[CM94] Chen Y., Medioni G.: Surface Description of Complex Objects from Multiple Range Images. *Proc. IEEE Conference Computer Vision and Pattern Recognition (CVPR)* (1994), 153–158.

[CSSK02] Chetverikov D., Svirko D., Stepanov D., Krsek P.: The trimmed iterative closest point algorithm. *In International Conference on Pattern Recognition (ICPR)* (2002).

[dAJ04] de Araujo B., Jorge J.: Curvature dependent polygonization of implicit surfaces. *Computer Graphics and Image Processing, 2004. Proceedings* (2004), 266– 273.

[Eva98] Evans L.: *Partial Differential Equations*. Providence: American Mathematical Society, 1998.

[FMZ*91] Forsyth D., Mundy J., Zisserman A., Coelho C., Heller A., Rothwell C.: Invariant Descriptors for 3D Object Recognition and Pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 13*, 10 (1991), 971–992.

[GO00] Golub G., O'Leary D. P.: Tikhonov Regularization and Total Least Squares. *SIAM J. Matrix Anal. Appl. 21* (2000), 185–194.

[Gos93] Goshtasby A.: Design and Recovery of 2-D and 3-D Shapes Using Rational Gaussian Curves and Surfaces. *International Journal of Computer Vision (IJCV) 10*, 3 (1993), 233–256.

[HBM04] Helzer A., Barzohar M., Malah D.: Stable Fitting of 2D Curves and 3D Surfaces by Implicit Polynomials. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 26*, 10 (2004), 1283–1294.

[HBZM00] Helzer A., Bar-Zohar M., Malah D.: Using Implicit Polynomials for Image Compression. *Proc. 21st IEEE Convention of the Electrical and Electronic Eng* (2000), 384–388.

[HJ85] Horn R. A., Johnson C. R.: *Matrix Analysis: Section 2.8*. Cambridge University Press, 1985.

[IF03] Ilic S., Fua P.: From Explicit to Implicit Surfaces for Visualization, Animation and Modeling. *In ISPRS Workshop on Visualization and Animation of Reality-based 3D Models* (2003).

[IHN*04] Ikeuchi K., Hasegawa K., Nakazawa A., Takamatsu J., Oishi T., Masuda T.: Bayon Digital Archival Project. *Proceedings of the Tenth International Conference on Virtual Systems and Multimedia* (2004), 334–343.

[IKK*07] Iwashita Y., Kurazume R., Konishi K., Nakamoto M., Aburaya N., Sato Y., Hashizume M., Hasegawa T.: Fast Model-Image Registration using 2D Distance Map for Surgical Navigation System. *Advanced Robotics 21*, 7 (2007), 751–770.

[JBS06] Jones M. W., Baerentzen J. A., Sramek M.: 3D Distance Fields: A Survey of Techniques and Applications. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 12*, 4 (2006), 581–599.

[JXL*07] Jiang X., Xu W., Li Y., Sweeney L., Gross R., Yurovsky D.: 2D Image Database Indexing: A Coefficient-Based Approach. *IEEE Int. Conf. Multimedia and Expo 21* (2007), 2210–2213.

[KC94] Keren D., Cooper D.: Describing Complicated Objects by Implicit Polynomials. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 16*, 1 (1994), 38–53.

[Ker94] Keren D.: Using symbolic computation to find algebraic invariants. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 16*, 11 (1994), 1143–1149.

[Kna94] Knanatani K.: Renormalization for Computer Vision. *The Institute of Elec., Info. and Comm. eng. (IEICE) Transaction 35*, 2 (1994), 201–209.

[KTFC01] Kang K., Tarel J.-P., Fishman R., Cooper D.: A Linear Dual-Space Approach to 3D Surface Reconstruction from Occluding Contours using Algebraic Surfaces. *Proc. IEEE Conference International Conference on Compter Vision (ICCV) 1* (2001), 198–204.

[KWTM03] Kindlmann G., Whitaker R., Tasdizen T., Moller T.: Curvature-based transfer functions for direct volume rendering: methods and applications. *Visualization, 2003. VIS 2003. IEEE* (2003), 513– 520.

[LC97] Lorensen W., Cline H.: Marching cube: A high resolution 3-D surface construction algorithm. *Computer Graphics Proceedings 21*, 4 (1997), 163–169.

[LMPT04] Leroy A., Mozer P., Payan Y., Troccaz J.: Rigid registration of free-hand 3D ultrasound and CT-scan kidney images. *Proc. MICCAI2004 3216* (2004), 837–844.

[Mar05] Marola G.: A Technique for Finding the Symmetry Axes of Implicit Polynomial Curves under Perspective Projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 27*, 3 (2005).

[Mas06] Masuda T.: Registration and Deformation of 3D Shape data through Parameterized Formulation. *Doctral Thesis in the graduate school of computer science, the University of Tokyo* (2006).

[MF07] Munim H. E., Farag A.: Shape Representation and Registration using Vector Distance Functions. *Proc. IEEE Conference Computer Vision and Pattern Recognition (CVPR)* (2007), Y1–Y8.

[MYL92] Menq C., Yau H., Lai G.: Automated Precision Measurement of Surface Profile in CAD-Directed Inspection. *In IEEE Transactions on Robotics and Automation 8* (1992), 268–278.

[NNS99] Nguyen V.-D., Nzomigni V., Stewart C. V.: Fast and Robust Registration of 3D Surfaces Using Low Curvature Patches. *In Second International Conference on 3-D Imaging and Modeling (3DIM '99)* (1999), 201.

[OBA*03] Ohtake Y., Belyaev A., Alexa M., Turk G., Seidel H.-P.: Multi-level Partition of Unity Implicits. *ACM Transactions on Graphics (SIGGRAPH 2003) 22*, 3 (2003), 463–470.

[OBA05] Ohtake Y., Belyaev A., Alexa M.: Sparse Low-degree Implicit Surfaces with Applications to High Quality Rendering, Feature Extraction, and Smoothing. *Proc. of Eurographics Symposium on Geometry Processing 2005* (2005), 149–158.

[OEB03] Oden C., Ercil A., Buke B.: Combining implicit polynomials and geometric features for hand recognition. *Pattern Recognition Letters 24*, 13 (2003), 2145–2152.

[OF03] Osher S., Fedkiw R.: Level Set Methods and Dynamic Implicit Surfaces. *Applied Mathematical Sciences. Springer 153* (2003).

[ONKI05] Oishi T., Nakazawa A., Kurazume R., 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)* (2005), 476–483.

[PBC*06] Penney G., Barratt D., Chan C., Slomczykowski M., Carter T., Edwards P., Hawkes D.: Cadaver validation of intensity-based ultrasound to CT registration. *Medical Image Analysis 10*, 3 (2006), 385–395.

[Pet02] Petitjean S.: A Survey of Methods for Recovering Quadrics in Triangle Meshes. *ACM Computing Surveys 34*, 2 (2002), 211–262.

[Pha] http://www.atslabs.com.

[Pie91] Piegl L.: On NURBS: a survey. *Computer Graphics and Applications 11*, 1 (1991), 55–71.

[PK92] Ponce J., Kriegman D.: *Symbolic and Numerical Computation for Artificial Intelligence, chapter 5: Elimination Theory and Computer Vision*. Computational Mathematics and Applications. Academic Press, 1992.

[PME*99] Pagoulatos N., Member S., Edwards W. S., Haynor D. R., Kim Y.: Interactive 3-D Registration of Ultrasound and Magnetic Resonance Images Based on a Magnetic Position Sensor. *IEEE Trans. Information technology in biomedicine 3*, 4 (1999), 278–288.

[Red00] Redding N. J.: Implicit Polynomials, Orthogonal Distance Regression, and the Closest Point on a Curve. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 22*, 2 (2000), 191–199.

[RL01] Rusinkiewicz S., Levoy M.: Efficient Variants of the ICP Algorithm. *International Conference on 3D Digital Imaging and Modeling (3DIM 2001)* (2001).

[RPMA01] Roche A., Pennec X., Malandain G., Ayache N.: Rigid registration of 3-D Ultrasound with MR images: A new approach combining intensity and gradient information. *IEEE Trans Medical Imaging 20*, 10 (2001), 1038–1049.

[SCK96] Subrahmonia J., Cooper D., Keren D.: Practical reliable bayesian recognition of 2D and 3D objects using implicit polynomials and algebraic invariants. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 18*, 5 (1996), 505–519.

[Sha04] Shamir A.: A Formulation of Boundary Mesh Segmentation. *Second International Symposium on 3D Data Processing, Visualization and Transmission* (2004), 82–89.

[SK03] Srinark T., Kambhamettu C.: A Novel Method for 3D Surface Mesh Segmentation. *Proceedings of the 6th IASTED International Conference on Computers, Graphics, and Imaging* (2003), 212–217.

[SMFF07] Salvi J., Matabosch C., Fofi D., Forest J.: A review of recent range image registration methods with accuracy evaluation. *Image and Vision Computing 25*, 5 (2007), 578–596.

[Ste02] Stewart C.: Covariance-based registration. *Technical Report RPI-CS-TR 02-8, Department of Computer Science Rensselaer Polytechnic Institute* (2002).

[SU05] Sahin T., Unel M.: Fitting Globally Stabilized Algebraic Surfaces to Range Data. *Proc. IEEE Conference International Conference on Compter Vision (ICCV) 2* (2005), 1083–1088.

[Tau91] TAUBIN G.: Estimation of Planar Curves, Surfaces and Nonplanar Space Curves Defined by Implicit Equations with Applications to Edge and Range Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 13*, 11 (1991), 1115–1138.

[TC92] TAUBIN G., COOPER D.: *Symbolic and Numerical Computation for Artificial Intelligence, chapter 6*. Computational Mathematics and Applications. Academic Press, 1992.

[TC00a] TAREL J.-P., COOPER D. B.: The Complex Representation of Algebraic Curves and Its Simple Exploitation for Pose Estimation and Invariant Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 22*, 7 (2000), 663–674.

[TC00b] TASDIZEN T., COOPER D.: Boundary Estimation from Intensity/Color Images with Algebraic Curve Models. *Int. Conf. Pattern Recognition* (2000), 1225–1228.

[TCC98] TAREL J.-P., CIVI H., 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* (1998), 13–21.

[TO99a] TURK G., O 'BRIENI J. F.: Shape transformation using variational implicit functions. *SIGGRAPH* (1999), 335–342.

[TO99b] TURK G., O 'BRIENI J. F.: Variational Implicit Surfaces. *Technical Report GIT-GVU-99-15, Graphics, Visualization, and Useability Center* (1999).

[TRSB94] TROBAUGH J. W., RICHARD W. D., SMITH K. R., BUCHOLZ R. D.: Frameless stereotactic ultrasonography: Method and applications. *Computerized Medical Imaging and Graphics 18* (1994), 235–246.

[TTC00] TASDIZEN T., TAREL J.-P., COOPER D. B.: Improving the Stability of Algebraic Curves for Applications. *IEEE Transactions on Image Processing (TIP). 9*, 3 (2000), 405–416.

[TWC03] TAREL J.-P., WOLOVICH W., COOPER D. B.: Covariant-conics decomposition of quartics for 2D shape recognition and alignment. *Journal of Mathematical Imaging and Vision 19* (2003), 255–273.

[UA98] UNEL M., A.WOLOVICH W.: Pose estimation and object identification using complex algebraic representations. *Pattern Anal. Appl. 1* (1998), 178–188.

[UA99] UNEL M., A.WOLOVICH W.: A New Representation for Quartic Curves and Complete Sets of Geometric Invariants. *International Journal of Pattern Recognition and Artificail Intelligence 13*, 8 (1999).

[UA00] Unel M., A.Wolovich W.: On the Construction of Complete Sets of Geometric Invariants for Algebraic Curves. *Advances In Applied Mathematics 24* (2000), 65–87.

[Uns07] Unsalan C.: A Model Based Approach for Pose Estimation and Rotation Invariant Object Matching. *Pattern Recogn. Lett. 28*, 1 (2007), 49–57.

[WRN05] Wein W., Roper B., Navab N.: Automatic registration and fusion of ultrasound with CT for radiotherapy. *Proc. MICCAI 2005 2* (2005), 303–311.

[WU98] Wolovich W. A., Unel M.: The Determination of Implicit Polynomial Canonical Curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 20*, 10 (1998), 1080–1090.

[X, 07] X, Huang and N.A. Hill and J. Ren and G. Guiraudon and T.M. Peters: Intracardiac 2d us to 3d ct image registration. *Proceedings of the SPIE International Symposium on Medical Imaging 2007*, 6509-85 (2007).

[YGZS05] Yamauchi H., Gumhold S., Zayer R., Seidel H.-P.: Mesh Segmentation Driven by Gaussian Curvature. *Pacific Graphics 2005, Visual Computer 21*, 8–10 (2005), 649–658.

[YT99] Yngve G., Turk G.: Creating smooth implicit surfaces from polygonal meshes. *Technical Report GIT-GVU-99-42, Graphics, Visualization, and Useability Center, Georgia Institute of Technology* (1999).

[Zha92] Zhang Z.: Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision (IJCV) 13*, 2 (1992), 119–152.

[ZOF01] Zhao H. K., Osher S., Fedkiw R.: Fast Surface Reconstruction Using the Level Set Method. *In Proceedings of IEEE Workshop on Variational and Level Set Methods in Computer Vision (VLSM 2001)* (2001).

List of Publications

Journal Papers

Major contribution to this thesis

1. B. Zheng, J. Takamatsu and K. Ikeuchi, “3d model segmentation and representation with implicit polynomials”, IEICE Trans. on Information and Systems, Vol. E91-D, No. 4, pp. 1149-1158, 2008.

2. B. Zheng, J. Takamatsu and K. Ikeuchi, “An Adaptive and Stable Method for Fitting Implicit Polynomial Curves and Surfaces”, IEEE Trans. on Pattern Analysis and Machine Intelligence (submitted).

Minor contribution to this thesis

3. 曽我部知広，鄭波，橋本康，張紹良,「非対称 Toeplitz 行列のための置換行列による前処理」, 日本応用数理学会論文誌，Vol. 15，No. 2，2005，pp．159-168.

Conferences Papers

Major contribution to this thesis

1. B. Zheng, J. Takamatsu and K. Ikeuchi, “Adaptively Selecting Degrees for Shape-representing Implicit Polynomials”, 第 10 回　画像の認識・理解シンポジウム (MIRU’07) (poster presentation).

2. B. Zheng, J. Takamatsu and K. Ikeuchi, “Adaptively Determining Degrees of Implicit Polynomial Curves and Surfaces”, Proc. 8th Asian Conf. on Computer Vision (ACCV2007) , Nov. 2007 (oral presentation).

3. B. Zheng, R. Ishikawa, T. Oishi, J. Takamatsu and K. Ikeuchi, “A Fast 2D-3D / 3D-3D Alignment Using 3D Implicit Polynomials”, 第 11 回　画像の認識・理解シンポジウム (MIRU’08) (to appear).

Minor contribution to this thesis

4. B. Zheng, S.-L. Zhang, “ A Computing System for Solving Large-scale Linear Systems ”, Supercomputing 2004 (SC04) (poster presentation).

5. 曽我部知広， 鄭波， 橋本康， 張紹良,「非対称 Toeplitz 行列に対する法の適用」, 第 32 回 数値解析シンポジウム (NAS2003)

6. 曽我部知広， 鄭波， 橋本康， 張紹良,「非対称線形方程式のための Bi-CR 法」，日本応用数理学会年会（2003 年）

7. 鄭波，張紹良， 「連立一次方程式のための積型反復法の多様性」，日本応用数理学会年会（2004 年）.

Workshops

1. B. Zheng, R. Ishikawa, T. Oishi, J. Takamatsu and K. Ikeuchi, “6-DOF Pose Estimation from Single Ultrasound Image Using 3D IP Models”, 5th IEEE International Workshop on Object Tracking and Classification in and Beyond the Visible Spectrum (OTCBVS’08), in conjunction with CVPR’08, Anchorage, AK, USA, June 27, 2008.

Award

1. **Best Paper Award** sponsored by *IEEE Computer Society*, for the paper, “6-DOF Pose Estimation from Single Ultrasound Image Using 3D IP Models”, in 5th IEEE International Workshop on Object Tracking and Classification in and Beyond the Visible Spectrum (OTCBVS’08), in conjunction with CVPR’08.

Patent

1. 池内克史，高松淳，大石岳史，鄭波，石川亮（キャノン）,「陰多項式の勾配を利用した 2D-3D/3D-3D 高速位置合わせ」（特許申請中）