DOI: 10.3745/KIPSTB.2009.16-B.1.47

3차원 측정점으로부터의 객체 자동인식

안 성 준†

요 약

측정점으로부터의 3차원 객체 자동인식은 컴퓨터비전, 지능형로봇 등의 분야에서 주요 연구주제이다. 본 논문에서 저자는 측정오차가 포함되어 있으며 정렬되지 않은 대용량 3차원 측정점으로부터 객체를 자동적으로 추출하며 그 형상계수를 추정하는 소프트웨어 기술에 대한 소개를 하고자 한다. 해당 소프트웨어는 기능적으로 상호 연결된 형상모델 제시, 측정점 분할, 형상모델 맞춤의 세 부분으로 이루어졌으며 최단거리 최소제곱법(ODF)이 핵심요소이다. ODF는 형상모델과 측정점 사이의 최단거리의 제곱합을 최소화하는 형상모델 계수를 추정한다. 무작위로 선정된 부분 측정점에 대한 임시 형상모델로서 이차 곡면이 ODF에 의하여 구하여지면 우리는 이로부터 3차원 객체를 자동적으로 추출하는 과정인 최종 형상모델 제시, 측정점 분할, 형상모델 맞춤에 필요한 초기값을 제공할 수 있다. 소개된 소프트웨어 기술을 실제 3차원 측정점에 적용함으로써 그의 성능을 확인하고자 한다.

키워드 : 객체인식, 3차원 측정점, 최소제곱법

Automatic Object Recognition in 3D Measuring Data

Sung Joon Ahn†

ABSTRACT

Automatic object recognition in 3D measuring data is of great interest in many application fields e.g. computer vision, reverse engineering and digital factory. In this paper we present a software tool for a fully automatic object detection and parameter estimation in unordered and noisy point clouds with a large number of data points. The software consists of three interactive modules each for model selection, point segmentation and model fitting, in which the orthogonal distance fitting (ODF) plays an important role. The ODF algorithms estimate model parameters by minimizing the square sum of the shortest distances between model feature and measurement points. The local quadric surface fitted through ODF to a randomly touched small initial patch of the point cloud provides the necessary initial information for the overall procedures of model selection, point segmentation and model fitting. The performance of the presented software tool will be demonstrated by applying to point clouds.

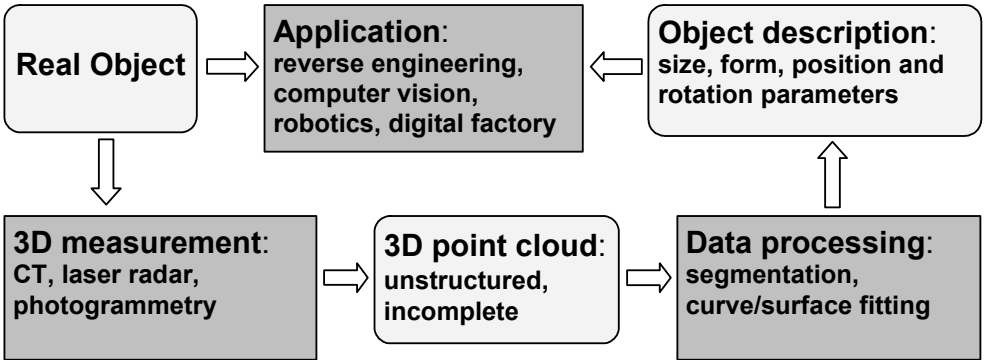
Keywords : Object Recognition, Point Cloud, Least Squares

1. Introduction

One of the main tasks of computer vision and robotics is to extract and recognize geometric objects from the environmental scene, where an object should be represented through geometric information such as size, position and orientation. Especially the parametric model recovery (PMR, Fig. 1) is aiming to represent the shape and size of an object in term of mathematical formulas. A fully automatic and generally applicable solution to parametric model recovery might be realized only through a highly sophisticated software technique analyzing all the available information on objects, such as point cloud, object database, object surface color and texture. In this paper we present a software tool for increasing the accuracy and automation degree of PMR by exploiting the 3D point cloud of objects.

If we restrict our interest field to industrial environment, we find out that a large portion of industrial objects including manufacturing facilities and work pieces can be modeled as exact features, i.e. planes, spheres, cylinders or cones[8]. Thus, even when we limit the range of our interest model features to geometric primitives (exact features), there is still a large demand on a fully automatic identification of these features, especially from the

† 정 회 원 : 성균관대학교 정보통신공학부 조교수
논문접수 : 2008년 12월 2일
수 정 일 : 1차 2008년 12월 11일
심사완료 : 2008년 12월 11일



(Fig. 1) Parametric Model Recovery of real objects via 3D point cloud

fields of reverse engineering, robotics and digital factory.

Under the circumstances mentioned above, we have developed a software tool for a fully automatic extraction of exact features from point cloud, based on our previous work on a semi-automatic solution[2]. The functionality of the software tool is analogous to that of the human intelligence searching for objects of exact feature in a dark room environment. In this paper we describe in detail the algorithmic techniques implemented in our software tool. The performance of the software tool is tested on point clouds generated by X-ray CT technology.

2. Parametric Model Recovery

2.1 Model Features for Real Object

There are three analytic forms of describing curve/surface, i.e., explicit, implicit and parametric form[3,4,10]. In general, diverse applications handling dimensional models or objects use the implicit or the parametric form. In addition, many applications e.g. the bin-picking and the obstacle-avoidance task in robotics involve describing the real objects in terms of shape, size, position, and orientation. Thus, we group the model parameters $\mathbf{a}$ of a curve/surface into form $\mathbf{a}_g$, position $\mathbf{a}_p$, and rotation parameters $\mathbf{a}_r$ as follows:

$$\begin{cases} f(\mathbf{a}_g, \mathbf{x}) = 0 & : \text{implicit feature} \\ \mathbf{x}(\mathbf{a}_g, \mathbf{u}) & : \text{parametric feature} \end{cases} \quad (1)$$

$$\mathbf{X} = \mathbf{R}^{-1}_{\omega,\varphi,\kappa}\mathbf{x} + \mathbf{X}_o \quad \text{or} \quad \mathbf{x} = \mathbf{R}_{\omega,\varphi,\kappa}(\mathbf{X} - \mathbf{X}_o), \quad (2)$$

$$\mathbf{a}^T \equiv (\mathbf{a}_g^T, \mathbf{a}_p^T, \mathbf{a}_r^T) = (a_1, \ldots, a_l, X_o, Y_o, Z_o, \omega, \varphi, \kappa). \quad (3)$$

The form parameters, e.g. half lengths a, b, c of an ellipsoid, represent the shape and size of the canonical model feature (1) defined in model coordinate frame xyz. They are invariant to the rigid body motion (2) of the model feature in reference coordinate frame XYZ. Our software tool extracts exact features from given point cloud and estimates their model parameters in terms of form, position, and rotation parameters (3).

2.2 3D Point Cloud

The optical 3D measuring devices available on the market can generate millions of dense 3D points in a few seconds[12]. However the point cloud is usually not dense enough to cover the details of the object surface. And it is generally assumed that the point cloud is not ordered. Furthermore, because of the limited accessibility of the measuring devices to the object surface, the point cloud covers only partially the object surface. Our software tool can handle such unordered incomplete and complex point cloud with a large number of data points.

A measurement point is the probable observation of an unknown nearest point on the object surface to the measurement point[1]. The distance between the measurement point and the unknown object point is the true measurement error. In practice, because the true object surface is unknown, it is substituted by the associating model feature[8] and the true measurement error is substituted by the minimum distance (geometric distance, Euclidean distance) between the model feature and the measurement point. This error definition outlines the algorithmic functionalities which should be implemented in the software tool for a reliable and accurate PMR from point cloud. The minimum distance should be used not only as the decision measure between the inliers and the outliers of the model feature (segmentation), but also as the error measure to be minimized by the estimation of model parameters (model fitting)[1,8]. Although the calculation and minimization of the minimum distances are computationally expensive, they are of vital importance to a reliable and accurate PMR from point cloud.

2.3 Orthogonal Distance Fitting

We briefly describe the orthogonal distance fitting (ODF) that estimates the model parameters by minimizing the square sum of the error distances between the model feature and the given points. Interested readers are referred to [3] for a complete description of the ODF algorithms.

The ODF task can be interpreted as an energy minimization problem illustrated in (Fig. 2), with which the energy (cost) function is defined as:

$$\sigma_0^2 \equiv (\mathbf{X}-\mathbf{X}')^{\mathrm{T}}\mathbf{P}^{\mathrm{T}}\mathbf{P}(\mathbf{X}-\mathbf{X}') \quad \text{or} \quad \sigma_0^2 \equiv \mathbf{d}^{\mathrm{T}}\mathbf{P}^{\mathrm{T}}\mathbf{P}\mathbf{d} \quad , \quad (4)$$

where the vectors $\mathbf{X}^{\mathrm{T}} = (\mathbf{X}_1^{\mathrm{T}}, \cdots, \mathbf{X}_m^{\mathrm{T}})$ and $\mathbf{X}'^{\mathrm{T}} = (\mathbf{X}_1'^{\mathrm{T}}, \cdots, \mathbf{X}_m'^{\mathrm{T}})$ are the coordinate row vectors of the m given points and of the m corresponding points on the model feature, respectively. $\mathbf{d}^{\mathrm{T}} = (d_1, \cdots, d_m)$ is the distance row vector with $d_i = \|\mathbf{X}_i^{\mathrm{T}} - \mathbf{X}_i'^{\mathrm{T}}\|$. The diagonal elements of the weighting matrix $\mathbf{P}^{\mathrm{T}}\mathbf{P}$ correspond to the spring constants $\{k_i\}_{i=1}^m$ in (Fig. 2).

To minimize the cost functions (4) the ODF algorithm minimizes not only the square sum but also every single distance $\{d_i\}_{i=1}^m$ between the model feature and the given points. Because the minimum distances $\{d_i\}_{i=1}^m$ are nonlinear to the model parameters, the ODF task is inherently a nonlinear minimization problem that must be solved through iteration. The computing cost and the memory space usage of the ODF algorithms in [3] are proportional to the number of data points, thus the algorithms are suitable for processing a massive point cloud.

3. Automatic Feature Extraction

For a given set of points $\{\mathbf{X}_i\}_{i=1}^m$ the feature extraction procedure consists of two substantial sessions of segmentation and model fitting, respectively (Fig. 1). At this point we confront a chicken-and-egg dilemma. Namely, without the geometric information on the model feature we cannot decide between the inliers and the outliers of the model feature (segmentation), and reversely, without the inliers we cannot get the geometric information on the model feature (model fitting). To resolve this information deadlock, either of the two sides should provide the seed information triggering the other side.

3.1 Model Selection and Initial Model Parameters

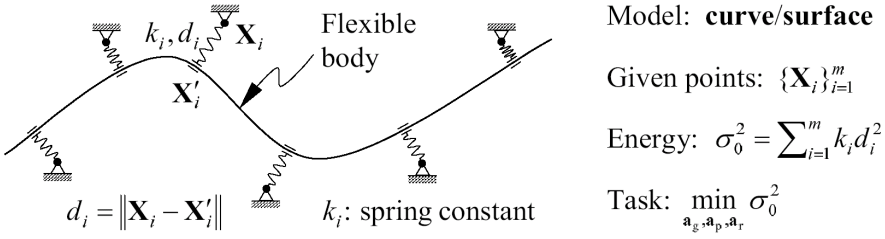
We obtain the geometric seed information on the model feature from a small patch of point cloud, which is comparable with touching an object and then guessing its geometry in a dark room environment:

1. Cut (touch) a small initial patch from the point cloud.
2. Fit a plane to the patch through the moment method (non-iterative linear ODF)[9].
3. Fit a quadric surface to the patch through ODF starting from the plane parameters.
4. Get the orthogonal footing point on the surface from the mass center of the patch.
5. Calculate the surface normal, principal curvatures at the footing point[5].
6. Choose the model type for the patch by analyzing the signed curvature radii (Fig. 3).
7. Derive the initial values for size, center, and orientation of the chosen model feature from the surface normal, curvature radii at the footing point.
8. Fit the initial model feature to the initial small patch through ODF starting from the model parameters derived in the last step.

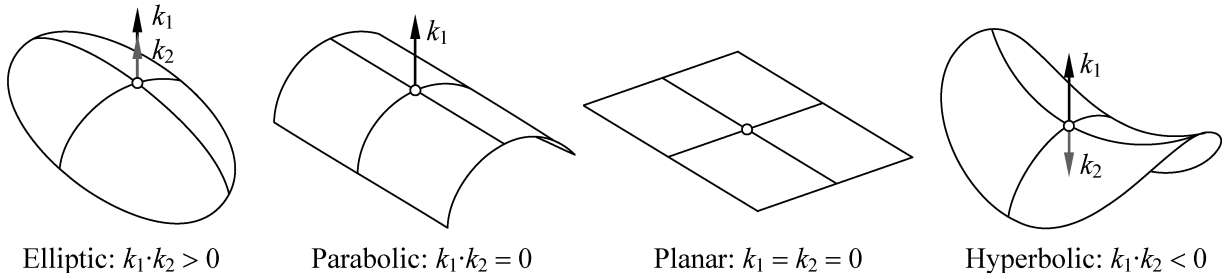
The quadric surface in the third step is a second order surface:

$$f(\mathbf{a},\mathbf{x}) = Ax^2 + By^2 + Cz^2 + 2Dxy + 2Eyz + 2Fzx + 2Gx + 2Hy + 2Iz + J = 0. \quad (5)$$

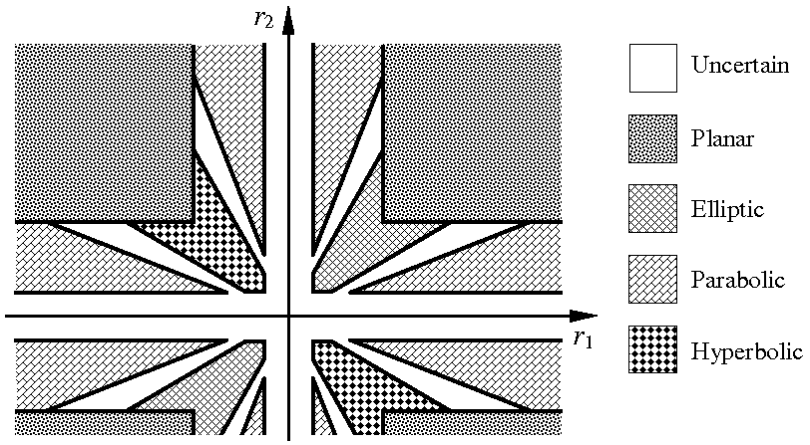
If $f(\mathbf{a},\mathbf{x}_i) \neq 0$ the given point $\mathbf{x}_i$ does not lie on the surface (5). By applying the ODF algorithm to the point set $\{\mathbf{x}_i\}_{i=1}^m$ of the initial small patch we obtain the parameter values of $\{A,B,C,D,E,F,G,H,I,J\}$. The footing point $\mathbf{x}_o$ on the surface (5) from the mass center $\bar{\mathbf{x}} = \frac{1}{m}\{\mathbf{x}_i\}_{i=1}^m$ of the patch can be determined by using a



(Fig. 2) Interpretation of the orthogonal distance fitting as an energy minimization problem



(a)



(b)

(Fig. 3) Classification of local surface types according to the two principal curvatures k_1 and k_2. (a) Elliptic for sphere/torus, parabolic for cylinder/cone, planar for plane, and hyperbolic for torus[5]; (b) Curvature radius map for local surface types ($r_1 = 1/k_1$, $r_2 = 1/k_2$)

constrained minimization method (method of Lagrangian multipliers[7]). The footing point $\mathbf{x}_o$ is the closest point on (5) from the point $\bar{\mathbf{x}}$:

$$\mathbf{x}_o = \arg\min_{\mathbf{x}} = (\bar{\mathbf{x}} - \mathbf{x})^T (\bar{\mathbf{x}} - \mathbf{x})$$

Subject to

$$f(\mathbf{a}, \mathbf{x}) = 0 .$$

We solve this problem by minimizing the Lagrangian function below:

$$L(\lambda, \mathbf{x}) = (\bar{\mathbf{x}} - \mathbf{x})^T (\bar{\mathbf{x}} - \mathbf{x}) + \lambda f . \quad (6)$$

A necessary condition for a minimum of (6) is

$$\begin{pmatrix} \nabla L \\ \frac{\partial L}{\partial \lambda} \end{pmatrix} = \begin{pmatrix} -2(\bar{\mathbf{x}} - \mathbf{x}) + \lambda \nabla f \\ f \end{pmatrix} = \mathbf{0}$$
$$\text{with } \nabla = (\partial/\partial x, \partial/\partial y, \partial/\partial z)^T . \quad (7)$$

We solve (7) through iteration as below:

$$\begin{pmatrix} 2\mathbf{I} + \lambda \mathbf{H} & \nabla f \\ \nabla^T f & 0 \end{pmatrix}_k \begin{pmatrix} \Delta \mathbf{x} \\ \Delta \lambda \end{pmatrix} = \begin{pmatrix} 2(\bar{\mathbf{x}} - \mathbf{x}) - \lambda \nabla f \\ -f \end{pmatrix}_k \text{ with } \mathbf{H} = \frac{\partial}{\partial \mathbf{x}} \nabla f$$
$$\begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix}_{k+1} = \begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix}_k + \alpha \begin{pmatrix} \Delta \mathbf{x} \\ \Delta \lambda \end{pmatrix} . \quad (8)$$

Now given $\mathbf{x}_o$, we can calculate the surface normal $\mathbf{n}$ and the principal curvatures k_1 and k_2 of the surface at the footing point $\mathbf{x}_o$. If we set a temporary coordinate system xyz on $\mathbf{x}_o$, of which z-axis is parallel to the normal of the point patch plane of $\{\mathbf{x}_i\}_{i=1}^m$, then the implicit surface (5) holding $f_z = \partial f / \partial z \neq 0$ could be regarded as a parametric surface patch around $\mathbf{x}_o$ as below[6]:

$$\mathbf{x}(x, y) = \begin{pmatrix} x \\ y \\ z(x, y) \end{pmatrix}, \quad \mathbf{x}_x = \frac{\partial \mathbf{x}}{\partial x} = \begin{pmatrix} 1 \\ 0 \\ z_x \end{pmatrix}, \quad \mathbf{x}_y = \frac{\partial \mathbf{x}}{\partial y} = \begin{pmatrix} 0 \\ 1 \\ z_y \end{pmatrix},$$
$$\mathbf{x}_{xx} = \frac{\partial^2 \mathbf{x}}{\partial x \partial x} = \begin{pmatrix} 0 \\ 0 \\ z_{xx} \end{pmatrix}, \quad \mathbf{x}_{xy} = \frac{\partial^2 \mathbf{x}}{\partial x \partial y} = \begin{pmatrix} 0 \\ 0 \\ z_{xy} \end{pmatrix}, \quad \mathbf{x}_{yy} = \frac{\partial^2 \mathbf{x}}{\partial y \partial y} = \begin{pmatrix} 0 \\ 0 \\ z_{yy} \end{pmatrix}. \quad (9)$$

Where, we get z_x, z_y, z_{xx}, z_{xy}, and z_{yy} from (5) as below:

$$\frac{\partial f}{\partial x}+\frac{\partial f}{\partial z}\frac{\partial z}{\partial x}=f_x+f_z z_x=0,\quad z_x=-f_x/f_z,$$
$$\frac{\partial f}{\partial y}+\frac{\partial f}{\partial z}\frac{\partial z}{\partial y}=f_y+f_z z_y=0,\quad z_y=-f_y/f_z,$$
$$z_{xx}=\frac{2f_{zx}f_z f_x-f_{xx}f_z^2-f_{zz}f_x^2}{f_z^3},$$
$$z_{xy}=\frac{f_{yz}f_z f_x+f_{zx}f_y f_z-f_{xy}f_z^2-f_{zz}f_x f_y}{f_z^3},$$
$$z_{yy}=\frac{2f_{yz}f_y f_z-f_{yy}f_z^2-f_{zz}f_y^2}{f_z^3}.$$

Given (9), we derive the coefficients E, F, G of the first fundamental form of the surface[5,6]

$$ds^2=Edx^2+2Fdxdy+Gdy^2,$$
$$E=\mathbf{x}_x\mathbf{x}_x=1+z_x^2,\quad F=\mathbf{x}_x\mathbf{x}_y=z_x z_y,\quad G=\mathbf{x}_y\mathbf{x}_y=1+z_y^2.$$

And, the coefficients L, M, N of the second fundamental form of the surface are

$$k\cos\varphi ds^2=Ldx^2+2Mdxdy+Ndy^2,$$
$$L=\mathbf{n}\mathbf{x}_{xx}=\frac{z_{xx}}{D},\quad M=\mathbf{n}\mathbf{x}_{xy}=\frac{z_{xy}}{D},\quad N=\mathbf{n}\mathbf{x}_{yy}=\frac{z_{yy}}{D},$$
$$\mathbf{n}=\frac{\mathbf{x}_x\times\mathbf{x}_y}{D}=\frac{1}{D}\begin{pmatrix}-z_x\\-z_y\\1\end{pmatrix}=\frac{1}{\sqrt{f_x^2+f_y^2+f_z^2}}\begin{pmatrix}f_x\\f_y\\f_z\end{pmatrix}=\frac{\nabla f}{\|\nabla f\|},$$

where D is the dscriminent of the surface.

Finally, we can calculate the principal curvatures k_1 and k_2 of the surface by solving the second order equation below:

$$k_{1,2}=\left\{k\,|\begin{vmatrix}kE-L & kF-M\\ kF-M & kG-N\end{vmatrix}=0\right\},$$
$$\begin{aligned}&(kE-L)(kG-N)-(kF-M)(kF-M)\\&=(EG-F^2)k^2-(EN-2FM+GL)k+(LN-M^2)\\&=0\end{aligned}.$$

By investigating the Gaussian curvature

$$k_1k_2=\frac{LN-M^2}{EG-F^2},$$

and the mean curvature

$$k_1+k_2=\frac{EN-2FM+GL}{EG-F^2}$$

we can discriminate the type of the surface as shown in (Fig. 3). For example, if k_1 and k_2 are both nonzero and have the same sign, the surface is concave or convex. It could be a part of a sphere or ellipsoid, etc. If k_1 and k_2 have different signs, the surface is saddle shaped and could be a part of a torus.

The classification of the local surface types (Fig. 3a) according to the local curvatures (including the mean and Gaussian curvatures) of a curved surface fitted to a small point patch is known in literature[11]. Instead of the curvatures k_1 and k_2, our software tool employs the curvature radii $r_1=1/k_1$ and $r_2=1/k_2$ which correspond to the feature radius (Fig. 3b).

3.2 Overall Process and Experimental Result

Once the model type and parameters are initialized, the interaction loop between the segmentation and the model fitting can be triggered. As noted in Sect. 2, the minimum distance of a given point to the model feature should be used as the decision measure whether a point is inlier point of the model feature. However, with regard to a specific model feature, the large part of a point cloud is occupied by plain outliers, causing a high computing cost of unnecessarily calculating the minimum distances. Through utilizing the parameter grouping (1)-(3) and the properties of the implicit model description, we can efficiently eliminate the plain outliers from the point cloud. The overall process of the automatic feature extraction can be described below (see Fig. 4):

1. Initialize the model feature (model type, size, position, and orientation).

2. Put a domain box enclosing the interest volume of the model feature in xyz frame.

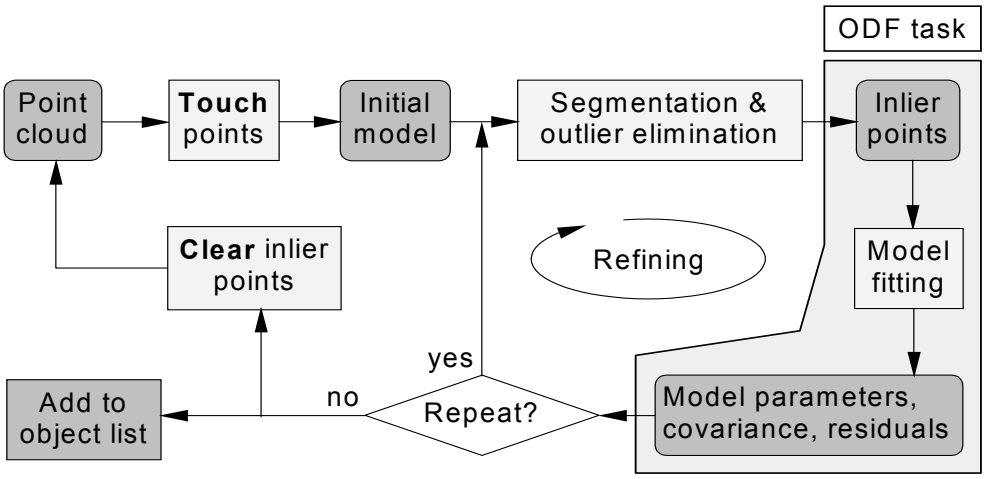
$x_{\text{left}}\le x\le x_{\text{right}},\quad y_{\text{front}}\le y\le y_{\text{back}},\quad z_{\text{bottom}}\le z\le z_{\text{top}}$.

For example, the domain box encloses an ellipsoid with extra margin of 10% of the half-lengths of the ellipsoid.

3. Stamp all the points lying outside the domain box as plain outlier.

4. Except for the inlier candidates lying between the two (inner and outer) iso-surfaces $f(\mathbf{a},\mathbf{x})-const_{\text{inner}}=0$ and $f(\mathbf{a},\mathbf{x})-const_{\text{outer}}=0$ of the model feature $f(\mathbf{a},\mathbf{x})=0$, stamp all points as plain outlier. For ellipsoid, we could put $const_{\text{inner}}=f(\mathbf{a},\mathbf{x})\big|_{\mathbf{x}=(0.9a,0,0)}$ and $const_{\text{outer}}=f(\mathbf{a},\mathbf{x})\big|_{\mathbf{x}=(1.1a,0,0)}$.

5. For each inlier candidate $\mathbf{x}_i$ remained in the last step,



(Fig. 4) Automatic feature extraction in point cloud

we determine the footing point $\mathbf{x}_i'$ by using (8) and calculate the shortest distance $d_i = \|\mathbf{x}_i - \mathbf{x}_i'\|$ between $\mathbf{x}_i$ and the model feature. Evaluate the rms distance $d_{\text{rms}} = \sqrt{\frac{1}{m}\sum_{i=1}^{m} d_i^2}$ of the inlier candidates to the model feature (m: number of inlier candidates).

6. Stamp only the inlier candidates as inlier, of which distances d_i to the model feature are not larger than 2-3 times the rms distance d_{rms}.

7. Update the model parameters through ODF to the inliers.

8. If necessary, repeat from the second step ('Refining' in Fig. 4).

9. Save the model parameters, and, clear the inliers from the point cloud.

10. Repeat from the first step until no more dense point patch can be found (touched).

As an experimental example for the automatic feature extraction from point cloud, we applied our software tool to a point cloud generated by X-ray CT (Fig. 5a). All the relevant model features could be extracted fully automatically and correctly (Fig. 5b).

4. Summary

We have developed a software tool for a fully automatic extraction of geometric primitives from unordered incomplete and error-contaminated 3D point clouds. The necessary information for model selection, segmentation, and model fitting could be obtained from a local quadric surface fitted to a small initial patch of the point cloud. The geometric error measure is of vital importance to both the segmentation and the model fitting, although the required computing cost is relatively high. In order to save the computing cost of the segmentation, we exploited the parameter grouping and the properties of the implicit model description. Our software can handle a variety of model features, e.g. line, 2D/3D circle, 2D/3D ellipse, plane, sphere, ellipsoid, circular/elliptic cylinder, circular/elliptic cone, etc. We demonstrated the outstanding performance of the software tool on a set of real measurement points generated by X-ray CT.

Reference

[1] R. J. Adcock, "Note on the method of least squares," The Analyst, Vol.4, pp.183-184, 1877.

[2] S. J. Ahn, I. Effenberger, S. Roth-Koch and E. Westkämper, "Geometric Segmentation and Object Recognition in Unordered and Incomplete Point Cloud," Lect. Notes Comput. Sc., Vol.2781, pp.450-457, 2003.

[3] S. J. Ahn, 'Least-Squares Orthogonal Distance Fitting of Curves and Surfaces in Space,' Lect. Notes Comput. Sc., Vol.3151, pp.125, 2004.

[4] R. M. Bolle and B. C. Vemuri, "On Three-Dimensional Surface Reconstruction Methods," IEEE Trans. Pattern Analy. and Mach. Intell., Vol.13, pp.1-13, 1991.

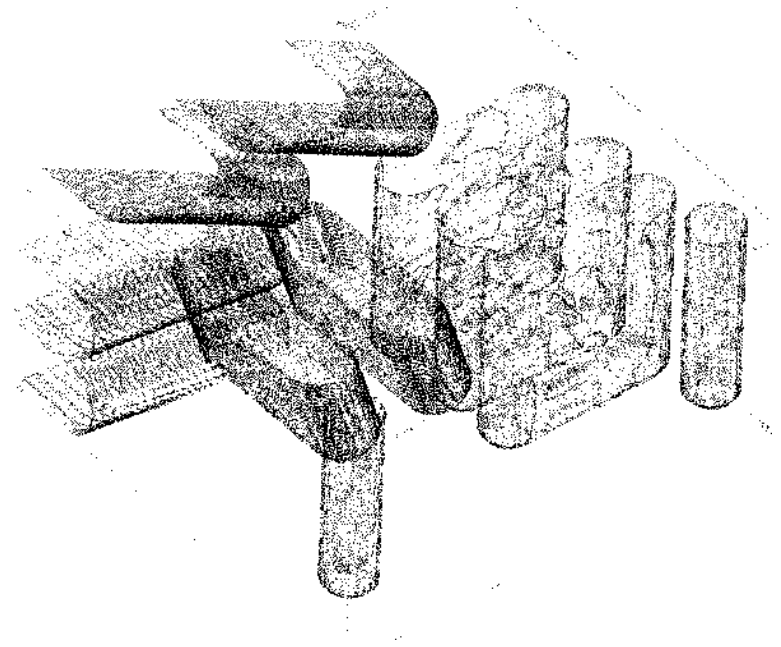
[5] M. P. do Carmo, 'Differential Geometry of Curves and Surfaces,' Prentice-Hall, 1976.

[6] G. Farin, J. Hoschek and M.S. Kim, 'Handbook of Computer Aided Geometric Design,' Elsevier Science, 2002.

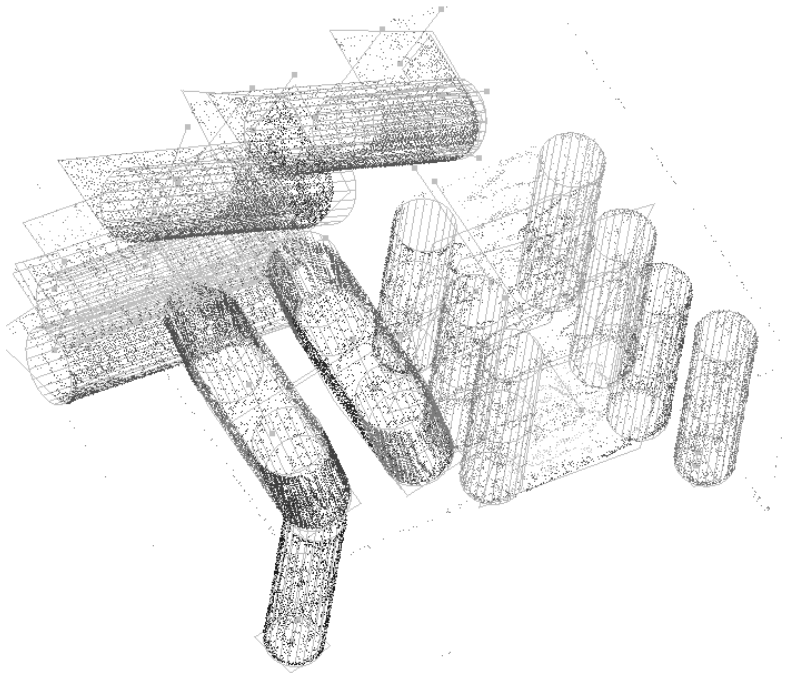
[7] R. Fletcher, 'Practical methods of optimization,' John Wiley & Sons, 1987.

[8] 'ISO 103606:2001: Geometrical Product Specification (GPS) - Acceptance test and reverification test for coordinate measuring machines (CMM) - Part 6: Estimation of errors in computing Gaussian associated features,' ISO, 2001.

[9] K. Pearson, "On Lines and Planes of Closest Fit to Systems



(a)



(b)

(Fig. 5) Feature extraction from a point cloud. (a) Unordered and incomplete point cloud obtained by X-ray CT technology; (b) Fully automatically extracted exact features

of Points in Space," The Philosophical Magazine: Ser.6, Vol.2, pp.559-572, 1901.

[10] D. H. von Seggern, 'CRC standard curves and surfaces,' 2nd Ed., CRC Press, 1993.

[11] B. C. Vemuri, A. Mitiche and J. K. Aggarwal, "Curvature-based representation of objects from range data," Image and Vision Computing, Vol.4, pp.107-114, 1986.

[12] http://laser.jadaproductions.net/, "POB's 2008 Laser Scanner Hardware and Software Surveys," Point of Beginning, 2008.

안 성 준

e-mail : finger@skku.edu

1985년 서울대학교 기계설계학과(공학사)

1987년 한국과학기술원 생산공학과(공학석사)

2004년 독일 Stuttgart University, Mechanical Eng. (Dr.-Ing.)

1985년~1990년 LG전자

1990년~2004년 Fraunhofer IPA, Stuttgart, Germany

2005년~현 재 성균관대학교 정보통신공학부 조교수

관심분야 : 3차원 측정, 공간정보처리