Computers & Industrial Engineering 63 (2012) 1189–1200

Contents lists available at SciVerse ScienceDirect

Computers & Industrial Engineering

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

A framework for 3D model reconstruction in reverse engineering

Jun Wang [a,*], Dongxiao Gu [b], Zeyun Yu [c], Changbai Tan [a], Laishui Zhou [a]

[a] College of Mechanical and Electrical Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China
[b] School of Management, Hefei University of Technology, Hefei 230009, China
[c] Department of Computer Science, University of Wisconsin, Milwaukee, WI 53211, USA

ARTICLE INFO

Article history:
Received 26 August 2011
Received in revised form 21 February 2012
Accepted 1 July 2012
Available online 7 September 2012

Keywords:
Model reconstruction
Mesh denoising
Mesh segmentation
Feature reconstruction
Reverse engineering

ABSTRACT

We present a framework for 3D model reconstruction, which has potential applications to a spectrum of engineering problems with impacts on rapid design and prototyping, shape analysis, and virtual reality. The framework, composed of four main components, provides a systematic solution to reconstruct geometric model from the surface mesh of an existing object. First, the input mesh is pre-processed to filter out noise. Second, the mesh is partitioned into segments to obtain individual geometric feature patches. Then, two integrated solutions, namely solid feature based strategy and surface feature based strategy, are exploited to reconstruct primitive features from the segmented feature patches. Finally, the modeling operations, such as solid boolean and surface trimming operations, are performed to "assemble" the primitive features into the final model. The concepts of "feature", "constraint" and "modeling history" are introduced into the entire reconstruction process so that the design intents are retrieved and exhibited in the final model with geometrical accuracy, topological consistency and flexible editability. A variety of industrial parts have been tested to illustrate the effectiveness and robustness of our framework.

1. Introduction

In many industries, such as the aerospace, automobile, and die and mold industries, plenty of efforts have been put on the design and manufacture of advanced products that deliver superior performance. The ability to rapidly design those products with improved characteristics is vital to success in the competitive environment of continuous change, and to respond quickly to the changing markets. However, conventionally, the product is designed on the shop floor instead of on the computer screen, and is represented by real physical prototypes, rather than virtual digital models. For example, style artists often work on physical mediums such as clay or wood rather than digital models to achieve a favorable shape design in the automobile industries (Huang and Menq, 2002). With the advance of CAD/CAM technology, creating geometric models of existing objects plays a substantial role in reverse engineering, especially when the prototype is created or modified on the shop floor and when the CAD model does not exist. Therefore, there are increasing demands to achieve 3D model reconstruction of existing objects in various industrial applications. Typically, the applications of model reconstruction consist of three categories: (1) the reproduction application; (2) the quality control application and (3) the redesign and modification application (Langbein, Marshall, and Martin, 2004). In the first two cases, the low level design intents are sufficient to their purposes, where only the exact shape information is needed. In the case of last one, the high level design intents, such as the geometric properties and relations, are required. Specifically, the high-level design intents refer to the information regarding features and constraints. Our framework aims to capture the design intents from existing objects and hence represent them in reconstructed models.

1.1. Previous works

Many model reconstruction methods have been proposed for converting scanning data to geometric models (Au and Yuen, 1999; Benko et al., 2001; Bernardini et al., 1999; Chivate and Jablokow, 1993, 1995; Franke, 1982; Hoppe et al., 1992; Park and Kim, 1996; Pottmann and Randrup, 1998; Pratt, 1987; Protopsaltis, 2009; Varady et al., 2007; Varady et al., 1997), which can be classified into two types: (1) surface-based methods and (2) feature-based methods. Comparatively, the surface-based methods have been studied more maturely. For instance, most of the state-of-the-art commercial reverse engineering systems such as CATIA, CopyCAD, Geomagic, Imageware, Rapidform and RE-SOFT adopt this surface-based strategy to reconstruct geometric models. The surface-based methods are typically suitable for products comprised of freeform surfaces, e.g., the surface parts of automobiles and aircrafts. However, they are not applicable to complicated industrial parts consisting of geometric features and constraints.

The feature-based strategy was proposed to carry out model reconstruction of industrial products. Werghi, Fisher, Ashbrook, and Robertson (1998), Werghi, Fisher, Robertson, and Ashbrook

* Corresponding author.
E-mail address: davis.wjun@gmail.com (J. Wang).

0360-8352/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cie.2012.07.009

(1999) and Fisher (2004) studied the constrained reconstruction of 3D geometric models of objects from range data and proposed a new technique of global shape optimization on the basis of feature positions and geometric constraints. They pioneered in giving such a large framework for the integration of geometric relationships in object reconstruction. The basic idea is to minimize a function containing a least-squares term and a penalty term associated with the constraints. This method uses a complex formulation of the constraint function, which heavily relies on the convexity of the constraint space and also needs accurate initial estimates of the solution. Benko, Kos, Varady, Andor, and Martin (2002), Kos, Martin, and Varady (2000) and Lukacs, Marshall, and Martin (1998) conducted the research of solid model reconstruction with boundary representation and constrained quadratic surface fitting in reverse engineering. Thompson, Owen St, and Germain (1999) worked on the REFAB project to reconstruct models of manufactured parts. The constraints and features are first introduced in REFAB so that high accuracy models are reconstructed efficiently. Because only simple features are used, this method is incapable of handling complicated parts. Ke et al. (2006) proposed feature-based reverse engineering strategies for modeling industrial components from point clouds to surfaces, in which the sectional feature based strategy and surface feature based strategy are exploited respectively. The main idea is to construct surface features and thereby perform surface modeling operations to generate the final model. However, it is non-trivial to impose the global constraints to surface features to generate a highly accurate and topologically consistent model. Ye et al. (2008) proposed a reverse engineering innovative design methodology (namely RID), which has introduced the definitions and construction of feature-based parametric solid models from scanned data. The RID methodology makes design and knowledge reuse possible for 3D digital design applications. The concept of surface feature reconstruction is incorporated into the methodology, while the solid features are not considered. Durupt, Remy, and Ducellier (2010) came up with a knowledge based reverse engineering (KBRE) methodology, which includes the functionality of managing and fitting manufacturing and functional features in mechanical products. The method categorizes and reconstructs features from manufacturing point of view, not from geometric perspective. Beccari, Farella, Liverani, Morigi, and Rucci (2010) presented a reverse engineering method for fast and interactive acquisition and reconstruction of a digital 3D model representing an existing physical object. This method uses a pen-based active stereo acquisition system to capture curve network of cubic splines and reconstruct a smooth surface using the Catmull–Clark subdivision technique. This method takes as an input the curve network of contours of 3D model, instead of the scanned point cloud. Consequently, minor features may be discarded and the reconstruction result is not accurate.

2. Overview of our framework

Our framework provides a systematic solution to reconstruct a 3D model from the surface mesh of an existing object. First, the input mesh is processed using the mesh smoothing technique, and mesh segmentation is performed on the mesh to extract individual geometric feature surfaces. Then, two types of feature reconstruction schemes are applied to the geometric feature surfaces to construct the primitive features. Finally, according to the topological and connectivity relationships of primitive features, a series of modeling operations are performed on those features to generate the final geometric model using the geometric modeling kernel: Open CASCADE (http://www.opencascade.org/, xxxx). Meanwhile, the history of model building operations is stored. Fig. 1 gives a systematic flowchart for model reconstruction from a surface mesh and an example of the fandisk model reconstruction. In summary, the main contributions of this paper are:

- An effective framework is proposed to produce the unique boundary representation of a complex 3D object. It is capable of automatically extracting geometric features and reconstructing CAD models from low quality mesh.
- An effective mesh denoising method is presented to filter out noises, capable of preserving smooth and sharp features.
- The combination of solid feature based and surface feature based strategies is proposed, allowing for reconstructions of all kinds of industrial products.
- Parametric features, constraints and modeling history are incorporated into the model reconstruction so that the original design intents are captured and embedded in reconstructed models. The strategy brings fast and flexible editing capabilities to models and hence facilitates model reuse and redesign for innovation applications.

3. Mesh processing

With the advance of scanning techniques, the 3D scanning device has become a powerful and popular tool to capture point clouds from an existing object. The surface mesh of the object is thereby generated from tessellation of the point clouds. During scanning, it is inevitable to introduce some noises to the acquired data, and consequently the surface mesh is noisy. Such noises usually result in errors to the subsequent model reconstruction. Therefore, removing the noises from the mesh, namely mesh denoising, is crucial prior to further processes. Over the last two decades, mesh denoising has been studied deeply and a number of methods have been proposed (Desbrun, Meyer, Schroder, and Barr, 1999; Fleishman, Drori, and Cohen-Or, 2003; Hildebrandt and Polthier, 2004; Jones, Durand, and Desbrun, 2003; Lange and Polthier, 2005; Ohtake, Belyaev, and Seidel, 2002; Sun, Rosin, Martin, and Langbein, 2007; Taubin, 1995, 2001; Zheng, Fu, Au, and Tai, 2010). Botsch et al. (2007) gave an insightful survey on general mesh smoothing and denoising. Using the Laplacian operator, Taubin (1995) proposed a mesh smoothing method by using an isotropic scheme to improve the smoothness of a surface mesh, while alleviating the shrinkage problem. Desbrun et al. (1999) extended Taubin's work to smooth irregular mesh by using geometric flows and re-scaling the mesh to preserve its volume. However, features are often blurred or filtered out in both methods. Ohtake et al. (2002) defined an error function over the mesh, the new position of each vertex is calculated through the minimization of the function. They also designed a diffusion-type smoothing method on the normal field. Jones et al. (2003) designed a non-iterative, feature-preserving smoothing algorithm by adopting local first-order predictors statistically defined on triangular surface meshes. Similarly, a bilateral filter is applied to the signed distances of neighborhood to the tangent plane on a vertex and the vertex is updated along its normal vector with the displacement obtained from the filter (Fleishman et al., 2003). The two bilateral methods work well in the presence of low noises but become ineffective with high level of noise. Hildebrandt and Polthier (2004) used mean curvature flows to remove noises while retaining features and volumes. Sun et al. (2007) designed a truncated function to filter the normal vector and update the vertex position, which is fast and effective to remove mesh noises. These approaches work well at low noise levels. When the noise level became reasonably high; however, it would oversharp or create features that do not exist in the original mesh.

We propose an efficient, feature-preserving mesh denoising approach based on anisotropic neighborhood searching and surface fitting techniques. First, a new filter is designed to operate on the normal vector fields. For each vertex, we choose as a seed face

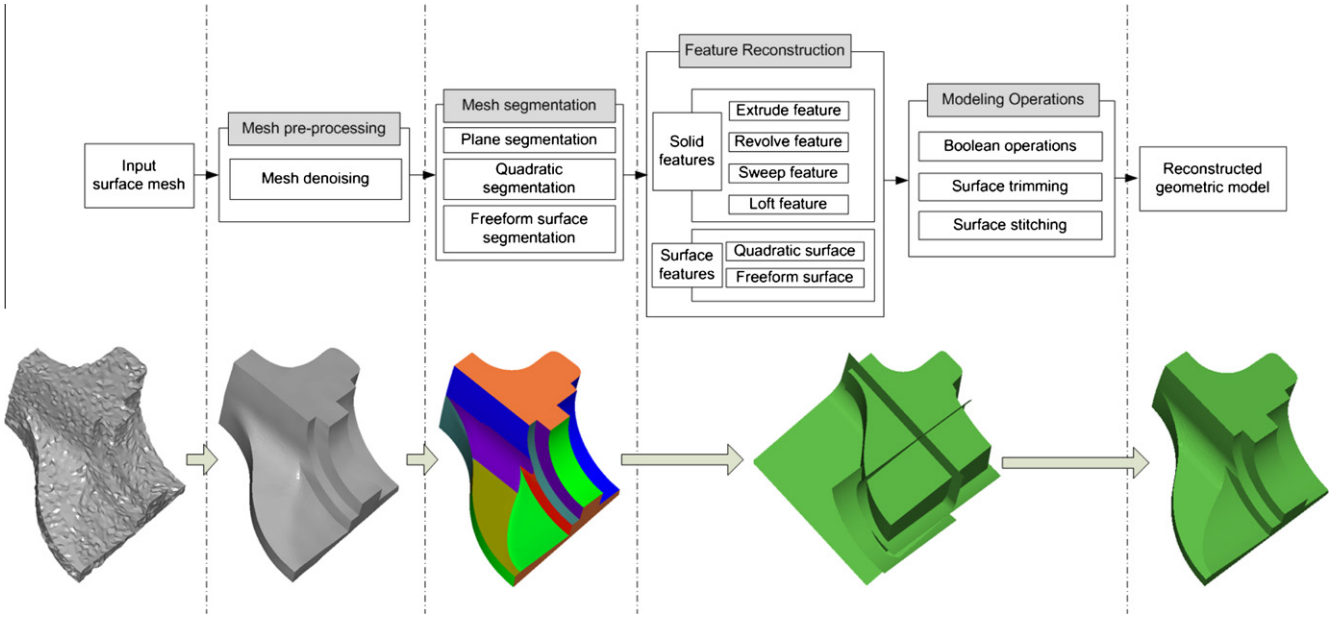

Fig. 1. A systematic flowchart for model reconstruction from a surface mesh.

one of its incident faces whose normal is closest to the vertex normal, and search the anisotropic neighborhood of the seed face as the neighborhood of the vertex. Over the neighborhood, the vertex is updated using the quadratic surface fitting and projection methods. The above procedure is repeated until convergence, i.e., the Hausdorff distance between the meshes from the ith and $(i+1)$th iterations is less than a pre-specified threshold. Here, we introduce the face normal filter firstly. Given a face f_i, let $NFF(i) = \{j | f(i) \cap f(j) \neq \emptyset\}$ be the index set of neighboring faces of f_i, then the updated normal of f_i is given as:

$$\mathbf{nf}(i) = normalize\left(\sum_{j \in NFF(i)} h_j \mathbf{n}_j\right) \tag{1}$$

where $\mathbf{n}_j$ is the normal of face f_j ($j \in NFF(i)$), and h_j is a weighted function defined by:

$$h_j = \begin{cases} f(\|\mathbf{n}_i - \mathbf{n}_j\| - T) & \text{if } \|\mathbf{n}_i - \mathbf{n}_j\| > T \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

where $0 < T < \sqrt{2}$ is a pre-defined threshold, and $f(x)$ is a monotonically increasing function for $x \geqslant 0$. According to our experiments, $f(x) = x^4$ always produces preferable results.

Next, we design a normal-weighted distance metric to search the anisotropic neighborhood of vertex. The normal-weighted distance is defined for each surrounding face around a seed face. Let f_i be a seed face and f_j, f_k are two adjacent faces within the neighborhood of f_i, see Fig. 2. $\mathbf{n}(i)$, $\mathbf{n}(j)$ and $\mathbf{n}(k)$ are the normals of f_i, f_j and f_k, respectively. c_j, c_k are the centroids of f_j, f_k, and mp_{jk} is the middle point of the common edge of f_j and f_k. The distance between two adjacent faces, f_l and f_{l+1}, is defined as $d_{l,l+1} = \|c_l - mp_{l,l+1}\| + \|c_{l+1} - mp_{l,l+1}\| = \|c_l - mp_{l,l+1}\| + \|c_{l+1} - mp_{l,l+1}\|$. Then, the normal-weighted distance from the face f_k to the seed face f_i is calculated as:

$$Dist(f_i, f_k) = Dist(f_i, f_j) + \omega_1 \|\mathbf{n}(i) - \mathbf{n}(k)\| \sum_{l=i}^{j} d_{l,l+1} + \omega_2 \|\mathbf{n}(j) - \mathbf{n}(k)\| d_{j,k} \tag{3}$$

where ω_1 and ω_2 are the non-negative values within the range of (0, 1). In our experiments, $\omega_1 = \frac{1}{3}$, $\omega_2 = \frac{1}{6}$ give good results. Note that there are typically more than one path going from f_i to f_k. In this case, the distance is calculated for each path and the shortest one can be chosen using Dijkstra's algorithm. After the normal-weighted distance is calculated for every triangular face in the neighborhood of the face f_i, the anisotropic neighborhood is chosen as those triangles whose distances to the seed face are less than a pre-specified distance threshold. For each vertex v, we search one of its incident triangular faces, denoted by f, whose normal is the most similar to that of v. The selected triangular face f is used as a seed face to search its anisotropic neighborhood using the distance metric above, which is considered as the neighborhood of v. Then we fit the searched neighborhood of each vertex with a quadratic surface, followed by projecting the vertex along its normal vector onto the fitted surface to obtain the projection point. The new position of the vertex is updated with the projection point. The entire mesh is processed by performing the fitting and projection operations on all vertices. The above procedure is repeated iteratively until the Hausdorff distance between two meshes from the ith and $(i+1)$th iterations is less than a pre-defined threshold. As a result, the final denoised mesh is obtained. In our algorithm, the parameters mainly consist of the number of iterations in face normal filtering: n_1, the sharpness threshold T, the number of anisotropic neighboring vertices in surface fitting: n_2, and the convergence threshold: d. To demonstrate the effectiveness of our method, we compare it with several recently related methods, see Fig. 3. From the figure, our method achieves a better result than others.

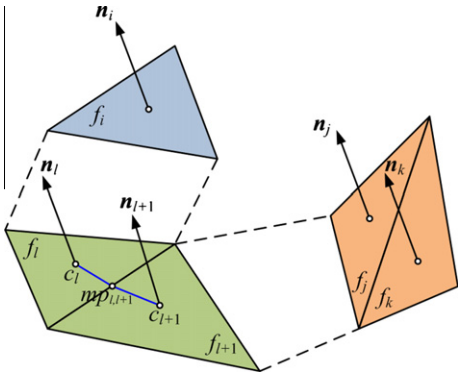

Fig. 2. Illustration of the normal-weighted distance.

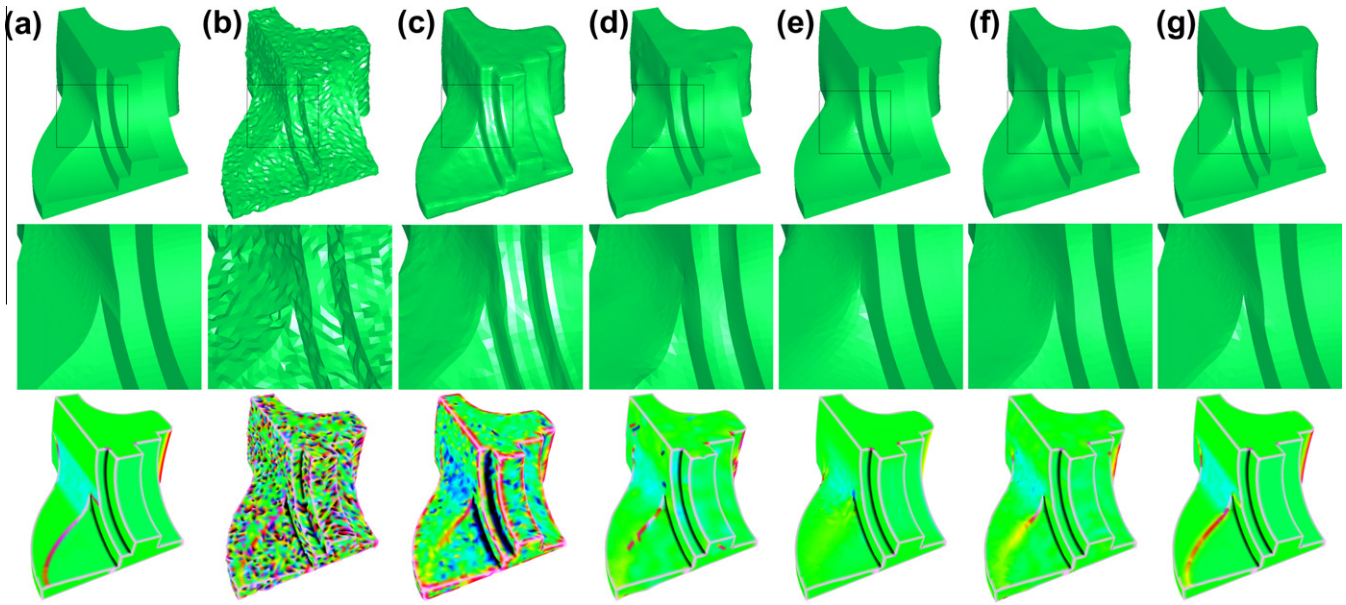

Fig. 3. Mesh denoising on the fandisk model. (a) Original mesh, (b) noisy model (Gaussian noise: 0.25× the mean edge length), and the results from (c) Fleishman et al. (2003) ($n = 10$), (d) Hildebrandt and Polthier (2004) ($n = 70$), (e) Sun et al. (2007) ($n_1 = 10$, $n_2 = 20$, $T = 0.4$), (e) Zheng et al. (2010) ($n_1 = 5$, $\sigma_s = 0.3$, $n_2 = 10$) and (f) ours ($n_1 = 5$, $T = 0.6$, $n_2 = 10$, d: 0.05 $*$ the mean edge length of the mesh). The second row shows sharp feature details of denoised meshes, and the third row gives the mean curvature distributions of denoised meshes. From the comparisons, we can see that sharp features are better preserved with our method.

4. Mesh segmentation

In the conventional design, a model is created by assembling a series of basic geometric features. To reconstruct the geometric model with high accuracy and capture high level information, namely the design intents, the surface mesh needs to be segmented into patches, each of which corresponds to a single feature surface.

Mesh segmentation has been studied by many research scientists. Different criteria and methods of mesh segmentation have been comprehensively summarized by Attene et al. (2006), Agathos, Pratikakis, Perantonis, Sapidis, and Azariadis (2007), Shamir (2008), Chen, Golovinskiy, and Funkhouser (2009). Vieira and Shimada (2005) presented an automatic algorithm to partition scanning data into regions closely approximated by single surfaces. It is fairly suitable for freeform surface mesh models, such as the sheet metal models, but not appropriate to process regular mechanical parts. Attene, Falcidieno, and Spagnuolo (2006) proposed a hierarchical face clustering method to decompose a triangular mesh on basis of plane, sphere and cylinder fitting technology. This method attempts to partition a triangular mesh into components which can be faithfully fitted with one of those three primitives. However, it has apparent limitation to deal with the mesh model including freeform surfaces. To address those issues, we have proposed an effective mesh segmentation method for complicated models, which are not only applicable to regular surface models, but to arbitrary freeform surface models as well. The segmentation algorithm is given below.

4.1. Plane segmentation

The region growing technique is used to carry out plane segmentation firstly. By choosing an arbitrary, un-segmented triangular face as the first seed, the region is initialized with this seed face. The region grows by adding one of three neighboring faces whose normal is closest to the seed face normal, and the process repeats iteratively until no more faces could be added into the current region. This region is referred to as a segment. By repeating the above process until all faces of mesh are segmented, the initial segmentation is thereby achieved. For each segment, the standard deviation of all face normals is used to determine whether the segment is plane or not. If not, the further segmentation is needed.

4.2. Quadratic surface segmentation

The curvatures of vertices are calculated by fitting the local neighborhoods with quadratic surfaces. According to Besl and Jain (1998), the triangular faces of mesh can be completely labeled with eight types (i.e. peak, pit, ridge, valley, saddle ridge, saddle valley, minimal surface and flat) in terms of curvatures, and hence divided into different patches. The minimal curvature of vertices on the cylinder surface is zero, while the maximal curvature is a non-zero constant. For a sphere surface, the Gaussian and mean curvatures are constant. The Gauss mappings of the normals of faces on the cone surface form a small circle of the Gauss sphere. Based on these observations, the statistics and Gauss mapping technology is exploited to carry out further segmentation using the region growing method, followed by recognizing the cylinder, sphere and cone. Due to noises and computational errors, the real data may not strictly comply with those observations. For example, the mappings of all face normals on a cone surface are not likely located exactly on a small circle. To address this issue, we fit the mappings with a circle and calculate the standard deviation of the fitting error. If the standard deviation is less than a small value and the radius of the fitted circle is less than the radius of Gauss sphere, i.e., 1, the mappings are considered to be around a small circle and hence the region is regarded as a cone surface segment. As a result, the quadric surface can be recognized and decomposed after performing the region growing processes on all non-planar segments from the initial segmentation. The remaining segments, which are neither planar nor quadric surface features, need to be further refined in the next subsection.

4.3. Freeform surface segmentation

In industrial applications, freeform surfaces of models are usually represented with the B-spline surface. Accordingly, we

exploit the bicubic B-spline surface fitting method to carry out segmentation refinement. We first choose a few connected triangular faces within freeform surface segments as a seed region (named an original seed region), and fit its vertices with a bicubic B-spline surface. Taking the fitted B-spline surface as an input, the region grows by adding the vertices which are not included in but adjacent to the current region. Each added vertex should satisfy three compatible conditions with the region: (1) the projection distance of the vertex onto the fitted face is less than a small value; (2) the normal vector of the projection point on the fitted surface should be close to that of the vertex; and (3) the principle curvatures of the projection point should also be close to those of the vertex. The region growing process terminates when no more vertices can be added into the region, followed by fitting a new B-spline surface on the region. We clear all faces in the current region and update it with the original seed region. The region growing process above is performed again by taking the newly fitted B-spline surface as an input. This procedure is repeated until the maximal region is achieved and hence a final segment is obtained. Essentially, this region growing method attempts to maximize the number of topologically connected vertices that can be faithfully fitted by a single underlying B-spline surface. According to our experiments, the original seed region affects the final segmentation result to some degree. In our implementation, we always choose as an original seed region the connected triangular faces which possess the lowest variance of the principle curvatures. Technically, we calculate the mean curvature variance of a fixed-size neighborhood of each vertex, and choose the neighborhood with the lowest variance as a seed region. By applying this region growing process to all freeform surface segments, the final segmentation refinement is achieved.

5. Solid feature reconstruction

In 3D models, the primitive features are composed of solid features and surface features. In this section, we introduce the corresponding reconstruction methods for solid primitive features, including extrude, revolve, sweep, and loft features. Specifically, we convert the problem of feature reconstruction into extracting feature parameters.

5.1. Extrude feature

In the conventional modeling, an extrude feature is generated by extruding a 2D contour along a constant direction with a certain distance, where the direction is always perpendicular to the underlying plane of the 2D contour. Then, we observe that the normal of each point on the side surface of the feature is perpendicular to the extrude direction, based on which the extrude direction can be extracted. Let $P = \{p_1, \ldots, p_m\}$ be the points on the side surface of an extrude feature, $\mathbf{N} = \{\mathbf{n}_1, \ldots, \mathbf{n}_m\}$ the corresponding normals of P, then the direction $\mathbf{v}_a$ can be computed as follows:

$$\begin{cases} \min \left[\mathbf{v}_a^T \left(\sum_{i=1}^{m} (\mathbf{n}_i \cdot \mathbf{n}_i^T) \right) \mathbf{v}_a \right] \\ \text{s.t. } \|\mathbf{v}_a\|_2 = 1 \end{cases} \tag{4}$$

Then, we calculate the centroid p_a of P, and construct a sectional plane determined by $\mathbf{v}_a$ and p_a. Having this sectional plane, we can reconstruct the 2D contour of the extrude feature using the constraint-based fitting method. Next, we project all the points of the side surface onto the line determined by $\mathbf{v}_a$ and p_a, and choose the two extreme points of those projection points as the start, end points of the extrude feature. As a result, the extrude distance is determined.

5.2. Revolve feature

In the conventional modeling, a revolve feature is constructed by revolving a 2D contour around an axis with a certain angle. Then, the lines along the normals of all points on the side surface of the revolve feature are coplanar with the axis. Based on this observation, we can extract the revolve axis. Let $P = \{p_1, \ldots, p_i, \ldots, p_m\}$ be the points on the side surface of a revolve feature, $\mathbf{N} = \{\mathbf{n}_1, \ldots, \mathbf{n}_i, \ldots, \mathbf{n}_m\}$ the corresponding normals of P. Let p_a be a point on the axis, $\mathbf{v}_a$ the vector of the axis, φ_i the angle between $\mathbf{n}_i$ and $\mathbf{v}_a$, and $dist_i$ is the distance between the axis and the line determined by p_i and $\mathbf{n}_i$, which could be expressed as: (See Fig. 4)

$$dist_i = \frac{|(p_a - p_i) \cdot (\mathbf{v}_a \times \mathbf{n}_i)|}{|\sin \varphi_i|} \tag{5}$$

Theoretically, the distance is zero between the line of each normal to the axis. Therefore, by solving the following optimization model:

$$\begin{cases} \min \sum_{i=1}^{m} dist_i^2 = \min \sum_{i=1}^{m} [(p_a \times \mathbf{v}_a) \cdot \mathbf{n}_i + (p_i \times \mathbf{n}_i) \cdot \mathbf{v}_a]^2 \\ \text{s.t. } \|\mathbf{v}_a\|_2 = 1 \end{cases} \tag{6}$$

the axis of the revolve feature is obtained.

Then, we construct sectional planes to extract the revolve contour. A base plane π is created, passing the line determined by p_a and $\mathbf{v}_a$. By rotating the base plane along the revolve axis with a series of angle intervals $\alpha, \ldots, k\alpha$, a set of sectional planes $pln_1, \ldots, pln_k$ are generated, followed by intersecting those sectional planes with the side surface, resulting in a series of sectional points. k is the number of sectional planes and is typically set to 90 in our implementation. The constraint-based fitting method is exploited to fit those points to obtain the revolve contour. Next, we project all points on the side surface of the revolve feature onto π and calculate the angle range of the projection points around p_a on π as the revolve angle.

5.3. Sweep feature

In the conventional modeling, a sweep feature is created by sweeping a 2D profile along a spatial path. Generally, the sweeping operation has two options: (1) the profile section remains perpendicular to the path at all times; (2) the profile section remains parallel to the beginning section at all times. The feature from the latter case could be considered as a loft feature, discussed in the next section. We consider the former case here. Theoretically, the associated sweep profile of each point on the sweep surface is orthogonal to the tangent vector in the Darboux frame of the point on the surface, based on which the sweep path can be extracted. Let $P = \{p_1, \ldots, p_i, \ldots, p_m\}$ be the points on the sweep surface, $\mathbf{N} = \{\mathbf{n}_1, \ldots, \mathbf{n}_i, \ldots, \mathbf{n}_m\}$ the corresponding normals of P. For each

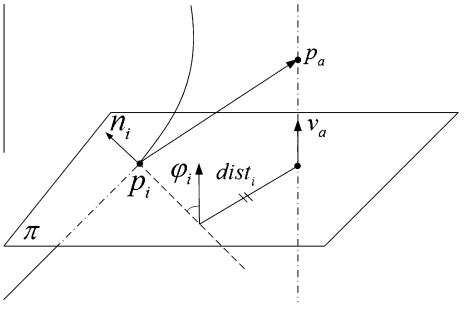

Fig. 4. Extraction of the revolution axis of the revolve feature.

point p_i, the tangent vector $\mathbf{T}_i$ is computed using local quadratic surface fitting method. Then the section plane π_i is created at p_i with the normal vector of $\mathbf{T}_i$. After that, the intersection points $IP_i = \{ip_{i,1}, \ldots, ip_{i,j}, \ldots, ip_{i,l}\}$ is obtained by intersecting π_i with P (l is the number of intersection points), whose normal vectors are $\mathbf{N}_i = \{\mathbf{n}_{i,1}, \ldots, \mathbf{n}_{i,j}, \ldots, \mathbf{n}_{i,l}\}$. Then, we calculate the associated point ap_i of p_i on the path, i.e., the intersection point between the sweep profile at p_i and the sweep path. We observe that the sum of the squared distances to the associated point on the path should be as small as possible from the extension lines of the normal vectors of the intersection points on a section plane. Solving this optimization problem yields an associated point of p_i on the path. For all points in P, we can get all the points on the path. Then, we fit those points with a cubic B-spline curve, which is the sweep path.

Next, we sample the sweep path into a set of points adaptively, which means there are more sample points in the high curvature areas on the path, while less in the low curvature areas. At each sample point, we create a section plane orthogonal to its tangent vector and intersect it with the sweep surface to get the intersection points. Having the associated intersection points of all sample points, we fit them to obtain the sweep profile. Fig. 5 presents the reconstruction result of a sweep feature.

5.4. Loft feature

In the conventional modeling, a loft feature is formed by blending multiple profile sections and transitioning them into smooth shapes between the profile sections. Generally, this operation can be transformed into skinning a series of profile Sections (2D contours). The shape, cut from the loft feature by two section planes with a sufficiently small distance, could be considered *generalized* rotationally symmetric. With this observation, we can extract the loft profiles. Specifically, we need to obtain the optimal section planes associated with all points on the loft surface. According to the characteristics of *generalized* rotational symmetry, the optimal section plane of each point features that its normal vector minimizes the variance of angles with the normal vectors at a set of relevant points intersected by the section plane with the surface of the feature. Accordingly, the optimal section plane of each point can be found iteratively. Tagliasacchi, Zhang, and Cohen-Or (2009) proposed an effective method to extract curve skeleton from incomplete point cloud, in which the similar cutting planes of points are computed. Here, we use this method to extract the optimal section plane. Let $P = \{p_1, \ldots, p_i, \ldots, p_m\}$ be the points on the loft surface, $\mathbf{N} = \{\mathbf{n}_1, \ldots, \mathbf{n}_i, \ldots, \mathbf{n}_m\}$ the corresponding normals of P. For each point p_i, we try to find an optimal section plane $\pi_i^\star$ such that the normal $\mathbf{n}_{\pi_i^\star}$ of $\pi_i^\star$ is most rotationally symmetric about the normals of the intersection points between $\pi_i^\star$ and the loft surface. It can be solved via an iterative method with the following model (Tagliasacchi et al., 2009):

$$\mathbf{n}^{(t+1)} = argmin_{\mathbf{n}\in\Re^3} var\left\{\langle \mathbf{n}, \mathbf{n}_{p_j}\rangle : p_j \in IP_i^{(t)}\right\}, \quad t \geqslant 0 \tag{7}$$

where $IP_i^{(t)}$ is the point set intersected by the section plane at the tth iteration with the loft surface, and $\mathbf{n}_{p_j}$ is the unit normal vector of p_j in $IP_i^{(t)}$. $\mathbf{n}^{t+1}$ is the normal of the plane from $t+1$-iteration. Solving this problem iteratively gives the optimal section plane of each point. Then, we intersect the section plane $\pi_i^\star$ of each point p_i with the surface, and obtain the intersection points $IP_i^\star$ of p_i.

After the intersection points of all points in P are found, the loft profile sections associated with all those points could be reconstructed with the constraint-based fitting method. As a result, there are a huge number of profile sections. However, it is not realistic to use all of those profile sections to generate the loft feature. Since there are many approximately congruent profile sections, we detect those similar profile sections and remove them using the dilation operation in mathematical morphology. Fig. 6 gives the reconstruction result of a faucet model.

6. Surface feature reconstruction

There are a number of industrial parts which are created by trimming solid geometry with surfaces. Therefore, prior to remodeling those parts, we need to reconstruct the surface features first. The surface features mainly consist of regular surfaces (i.e. plane and quadratic surfaces) and freeform surface (i.e. B-spline surface). Over the last ten years, the surface feature reconstruction has been studied profoundly. For the regular surfaces, Werghi et al. (1998), Werghi et al. (1999), Fisher (2004) studied the constrained reconstruction of quadratic curves and surfaces from range data and

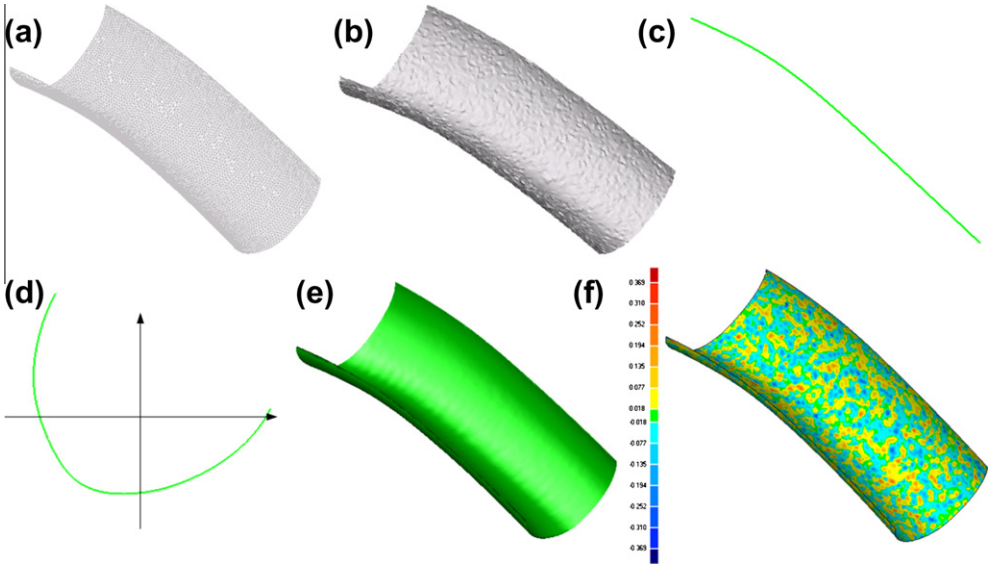

Fig. 5. Reconstruction of a sweep feature. (a) The point set on the surface of the feature; (b) the rendering effect of the input mesh of the feature; (c) the fitted sweep path; (d) the reconstructed sweep profile; (e) the reconstructed sweep feature; (f) the reconstruction error graph. The diameter of the bounding sphere of the feature is 82.32 and the average edge length of the input mesh is 0.713, while the maximum error is 0.369. From the graph, the reconstruction accuracy is sufficiently high.

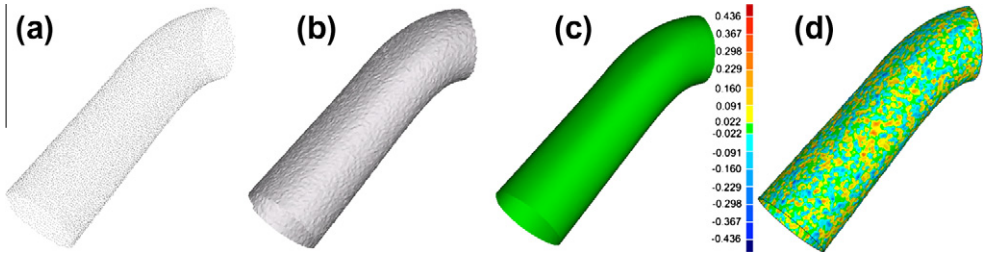

Fig. 6. Reconstruction of the loft feature in a faucet model. (a) The point set of the feature; (b) the rendering effect of the triangular mesh of the feature; (c) the reconstructed loft feature; (d) the reconstruction error graph. The diameter of the bounding sphere of the feature is 144.34 and the average edge length of the input mesh is 0.788, while the maximum error is 0.436. Comparatively, the error is small.

proposed a new technique of global shape optimization on the basis of feature positions and geometric constraints. The basic idea is to minimize a function containing a least squares term and a penalty term associated with constraints, which has a good performance when having accurate initial estimates of the solution. For the freeform surface, Milroy, Bradley, and Vickers (1995) and Krishnamurthy and Levoy (1996) fitted B-spline surfaces of freeform topology, while the user is required to manually specify the patch boundaries by drawing boundary elements on an approximating surface. In our system, we exploit an effective technique to estimate the initial parameters of quadratic surfaces and use the constrained optimization method (Werghi et al., 1999) to create quadratic surface features. In addition, the error-based adaptive fitting algorithm is used to reconstruct B-spline surface feature.

6.1. Quadratic surface fitting using squared distance minimization

Let S be the fitting surface of a point set $P = \{p_1, p_2, \ldots, p_m\} \subset R^3$, s is the closest point of a point $p \in P$ on S and d is the shortest distance (i.e. $d = \|s - p\|$). The principal curvatures of s on S are k_1, k_2, and the corresponding curvature radii are $\rho_1 = \frac{1}{|k_1|}$ and $\rho_2 = \frac{1}{|k_2|}$. According to surface theory, the Frenet frame of s on S consists of the two vectors $\mathbf{n}_1$, $\mathbf{n}_2$ of the principal curvature directions, and the normal vector $\mathbf{n}_3$. The squared distance from p to S can be expressed as Wang, Pottmann, and Liu (2006):

$$F(p) = \frac{d}{d-\rho_1}[\mathbf{n}_1 \cdot (p-s)]^2 + \frac{d}{d-\rho_2}[\mathbf{n}_2 \cdot (p-s)]^2 + [\mathbf{n}_3 \cdot (p-s)]^2 \tag{8}$$

Therefore, the squared distance function of each point $p_i \in P$ to S is:

$$\begin{aligned} F(p_i) &= \frac{d_i}{d_i-\rho_{i,1}}[\mathbf{n}_{i,1} \cdot (p_i - s_i)]^2 + \frac{d_i}{d_i-\rho_{i,2}}[\mathbf{n}_{i,2} \cdot (p_i - s_i)]^2 \\ &\quad + [\mathbf{n}_{i,3} \cdot (p_i - s_i)]^2 \\ &= \sum_{j=1}^{3}(\alpha_{i,j} \cdot [\mathbf{n}_{i,j} \cdot (p_i - s_i)]^2) \end{aligned} \tag{9}$$

where $\alpha_{i,1} = \frac{d_i}{d_i-\rho_{i,1}}$, $\alpha_{i,2} = \frac{d_i}{d_i-\rho_{i,2}}$ and $\alpha_{i,3} = 1$. $\mathbf{n}_{i,1}$, $\mathbf{n}_{i,2}$ and $\mathbf{n}_{i,3}$ are the principal curvature directions and the normal vector of p_i on S, respectively. s_i is the closest point of p_i on S and $d_i = \|s_i - p_i\|$. $\rho_{i,1}$ and $\rho_{i,2}$ are the curvature radii of p_i on S.

Let $\mathcal{P}$ be the parameter vector (namely, the unknown coefficient vector) of surface S, S^+ denotes the fitting surface with the updated parameter vector $\mathcal{P}^+ = \mathcal{P} + \mathcal{D}$, where $\mathcal{D}$ are incremental updates to S. The closest point of p_i on S^+ is s_i^+, which, strictly, is different from the shortest distance s_i of p_i to S. Since the difference is quite small, we can approximate s_i^+ with s_i. Suppose that $\mathbf{x}_i$ is the state vector of s_i, Eq. (9) can be transformed to Wang and Yu (2011):

$$E_i(\mathcal{P}^+) = \sum_{j=1}^{3}(\alpha_{i,j} \cdot [\mathbf{n}_{i,j} \cdot (p_i - S^+(\mathbf{x}_i))]^2) \tag{10}$$

By minimizing the sum of the squared distances of all points, i.e.:

$$\begin{aligned} \min F(\mathcal{P}^+) &= \min \sum_{i=1}^{m} E_i(\mathcal{P}^+) \\ &= \min \sum_{i=1}^{m}\sum_{j=1}^{3}(\alpha_{i,j} \cdot [\mathbf{n}_{i,j} \cdot (p_i - S^+(\mathbf{x}_i))]^2) \end{aligned} \tag{11}$$

the updated parameter vector $\mathcal{P}^+$ of S^+ is obtained, i.e. the surface is fitted. Using this method, the quadratic surfaces, such as cylinder, cone and sphere, can be reconstructed.

6.2. Quadratic surface feature

If quadratic surfaces exist independently in a part, we can simply adopt the fitting technique above to construct them individually. However, it is more common that there are more than one surface in a part and there are some geometric constraints among those surfaces. In this situation, we exploit the constraint-based surface fitting method to reconstruct them. At first, we still fit them individually, and recognize the potential constraint relationship of fitted surfaces based on the geometric properties of constraints.

Once the constraints are determined, we can apply those constraints to surface fitting so that a global optimization system is generated. Let $\mathbf{S} = \{S_1, \ldots, S_m\}$ be the set of individual surfaces (m is the number of surfaces). Suppose $\mathbf{X}$ is the parameter vector of all surfaces in $\mathbf{S}$, and $C_k(\mathbf{X})$ is the kth constraint function, then the constraint-based fitting problem could be expressed as:

$$\begin{cases} minF(\mathbf{X}) = min\sum_{i=1}^{m} f_i(\mathbf{X}) = min\sum_{i=1}^{m}\left(\omega_i \sum_{j=1}^{N_i} d^2(p_{i,j}, S_i)\right) \\ \text{s.t. } C_k(\mathbf{X}) = 0 \quad (k = 1, \ldots, t) \end{cases} \tag{12}$$

where $p_{i,j}$ is the jth point in the associated point set on the ith surface S_i, $d(p_{i,j},Si)$ is the distance from $p_{i,j}$ to S_i, and Ni is the number of associated points of S_i. ω_i is the fitting weight of a surface S_i (typically set to $\frac{1}{m}$). t is the number of constraints. Here, $d(p_{i,j},S_i)$ is represented by the algebraic distance, which is defined as: for a surface given by the equation $x^TAx = 0$, the algebraic distance from a point x' to the surface is simply x'^TAx'. Thus we have:

$$\begin{cases} minF(\mathbf{X}) = min(\mathbf{X}^T H\mathbf{X} + h\mathbf{X} + K) \\ \text{s.t. } C_k(\mathbf{X}) = 0 \quad (k = 1, \ldots, t) \end{cases} \tag{13}$$

where H, h and K are the coefficient matrix of surfaces in terms of its associated points. Using the penalty function method, this constrained optimization problem can be converted into the following non-linear optimization model by Leverberg-Marquarat algorithm (Werghi et al., 1999):

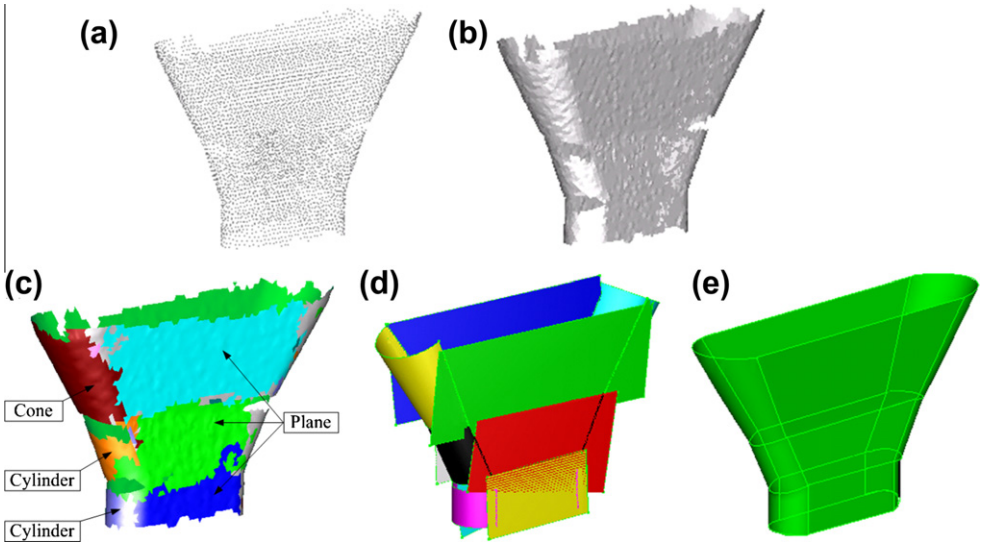

Fig. 7. Constrained fitting of multiple quadratic surfaces. (a) The point cloud of multiple quadratic surfaces; (b) the triangular mesh; (c) the segmentation result; (d) the individual fitting surfaces; (e) the constrained fitting result.

$$\begin{aligned} minE(\mathbf{X},\lambda) &= min\left(F(\mathbf{X}) + \lambda\sum_{k=1}^{t}[C_k(\mathbf{X})]^2\right) \\ &= min\left(\mathbf{X}^T H\mathbf{X} + h\mathbf{X} + K + \lambda\sum_{k=1}^{t}[C_k(\mathbf{X})]^2\right) \end{aligned} \tag{14}$$

where λ is the penalty term and set to be positive. Then this non-linear optimization problem can be further transformed to a linear equation system (Werghi et al., 1999):

$$\frac{\partial^2 E}{\partial^2 \mathbf{X}} \cdot \Delta\mathbf{X} = -\frac{\partial E}{\partial \mathbf{X}} \tag{15}$$

Here, the initial values of **X** are from the individual fitting results as discussed in the Section 6.1. After solving the equation system, the surfaces are obtained with constraints satisfied. Fig. 7 presents the constrained fitting result of multiple quadratic surfaces in a mechanical part. From the figure, the fitting result is satisfactory.

6.3. Freeform surface feature

For freeform surfaces, we use the error-based adaptive fitting method to reconstruct them. Given the initial number of control points, the least-squares method is adopted to fit the input points with a B-spline surface and the fitting error is calculated. In our implementation, the initial number is typically set t 16. If the error is less than a pre-defined threshold, the final surface is obtained; otherwise, using the current control points as initial values, we take advantage of the iterative method to optimize the control points or adjust the number of control points. The procedure repeats until convergence, i.e. the current fitting error is less than the threshold. Let ξ be the fitting error threshold, σ the error convergency threshold, N the maximal iterations, and i is the number of iterations, initialized as 0, then we have the following detailed steps to fit the input points P with a B-spline surface:

1. Calculate the knot vectors: U, V based on the number of control points: $(n_u + 1) \times (n_v + 1)$ and the surface orders: k, l.
2. Fit the input points with a B-spline surface S using the least squares method and hence obtain the control points $D = \{d_{i,j} | 0 \leqslant i \leqslant n_u, 0 \leqslant j \leqslant nv\}$.
3. If $i < N$, calculate the distance $dist_p$ from each $p \in P$ to S, the maximal distance $mDist_i$, the parametric pair $\left(u_p^\star, v_p^\star\right)$ of the projection point of p on S; otherwise, go to step (5).
4. If $mDist_i \leqslant \xi$, then exit the procedure; otherwise, if $mDist_{i-1} - mDist_i > \sigma$, optimize the control points using the projection parametric pair $\left(u_p^\star, v_p^\star\right)$ of each point p as the new parameter pairs of p. Suppose that $\Delta d_{i,j}$ is the update of $d_{i,j}$, the surface can be optimized by solving the following mathematical model:

$$min\sum_{p\in P}\left\|\sum_{i=0}^{n_u}\sum_{j=0}^{n_v}\left(d_{i,j} + \Delta d_{i,j}N_{i,k}\left(u_p^\star\right)N_{j,l}\left(v_p^\star\right) - p\right)\right\| \tag{16}$$

where $N_{i,k}(u)$ and $N_{j,l}(v)$ are the B-spline basis functions, determined by the Cox-de Boor recursion formula (Piegl and Tiller, 1997) in terms of U and V. Solving the optimization model, $\Delta d_{i,j}$ is calculated

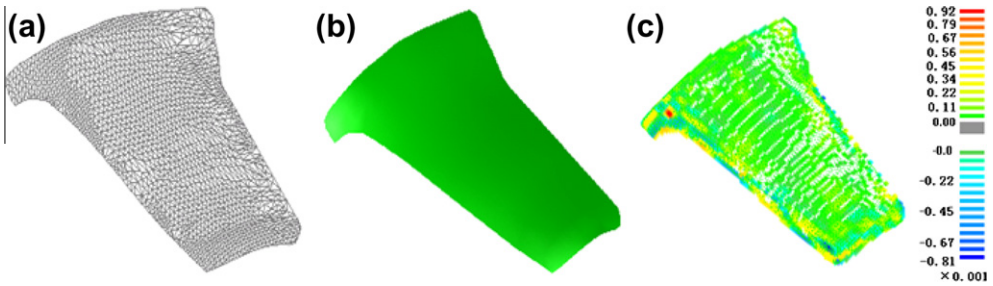

Fig. 8. B-spline surface fitting of a freeform surface. (a) The triangular mesh of a freeform surface; (b) the fitting result; (c) the fitting error distribution graph, where the average edge length is 0.0195, while the max error is 0.00092. From the graph, the fitting error is relatively small.

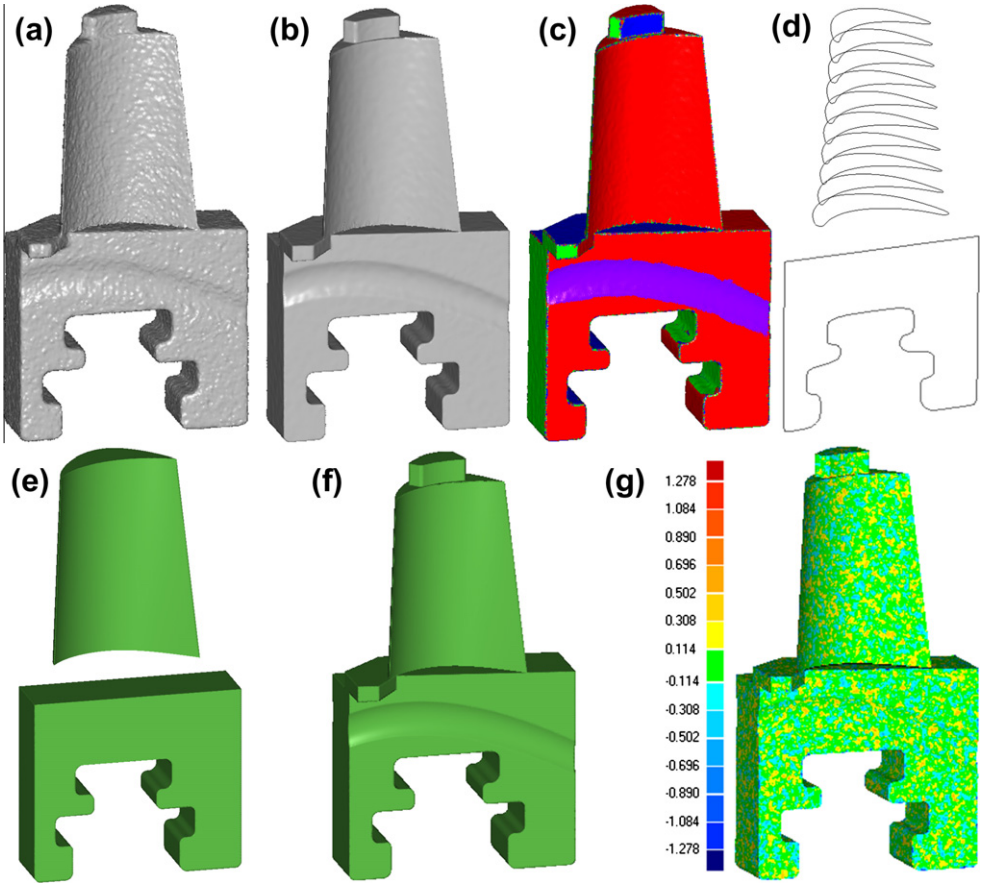

Fig. 9. Model reconstruction of a blade. (a) The noisy triangular mesh of the blade; (b) the denoised mesh; (c) the segmentation result; (d) a series of profiles of a loft feature and a profile of an extrude feature; (e) the corresponding loft feature and extrude feature to (d); (f) the final reconstructed blade model; (g) the reconstruction error distribution graph, where the bounding box of the model is roughly $310 \times 95 \times 635$ and the average edge length is 2.915, while the maximal error is 1.278, suggesting that the reconstruction is relatively accurate.

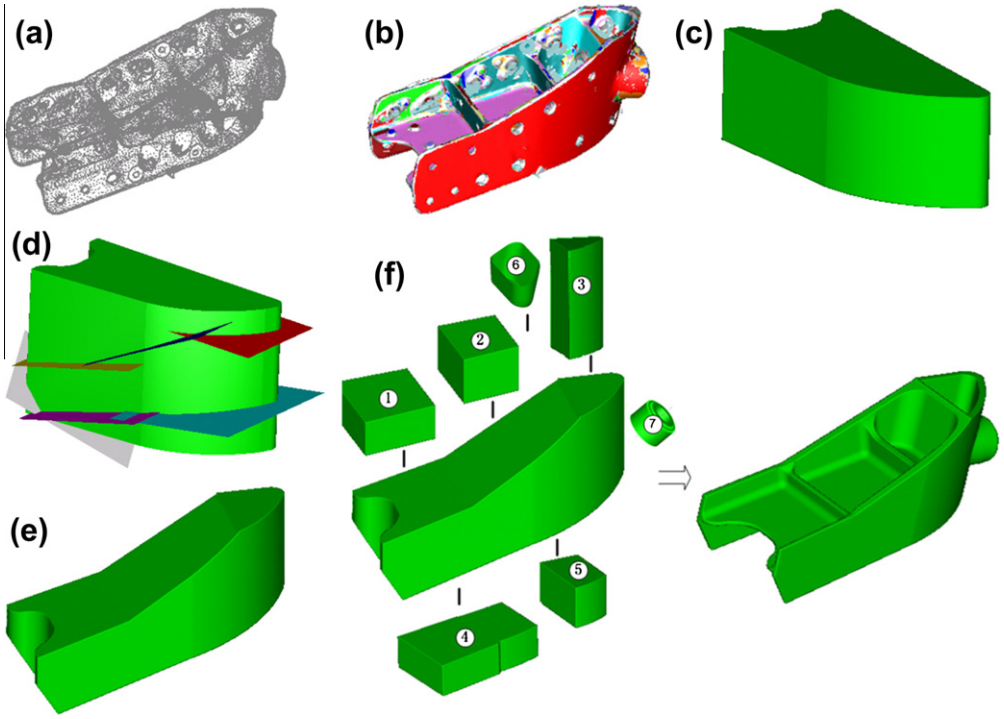

Fig. 10. Model reconstruction of a aircraft part. (a) The point cloud of the part; (b) the segmentation result; (c) the outer block, an extrude feature reconstructed from the outer surface; (d) multiple fitted planes from the corresponding points; (e) the intermediate result by trimming the outer block with the planes; (f) the final reconstructed model, generated by performing boolean and surface trimming operations among the primitive features and intermediate results.

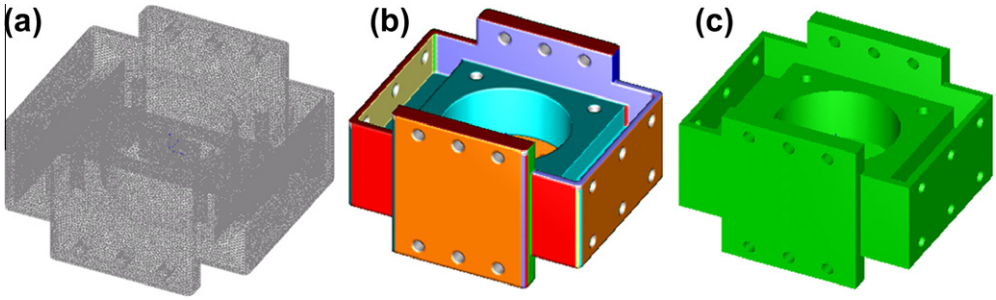

Fig. 11. Model reconstruction of a mechanical part. (a) The point cloud of the part; (b) the segmentation result; (c) the reconstructed model.

and every new control point is updated as $d'_{i,j} = d_{i,j} + \Delta d_{i,j}$. Accordingly, the new surface is obtained. Meanwhile, the number of iterations increases by 1, i.e. $i = i + 1$. Then, go to step (3);

5. Increase the number of control points and set $i = 0$, then go to step (2).

By running the steps above, the freeform surface is reconstructed within a given fitting error. Fig. 8 gives the fitting result of a freeform surface.

7. Construction of modeling operation history

History based modeling has many advantages in CAD technology. For example, the "shell" is an obvious modeling feature that benefits from a parent/child relationship. When a change is made to the parent, the child feature (shell) will conform to the modified parent during the model regeneration. Similarly, the modeling operation history is also significantly beneficial to model redesign, and model modification for the innovation applications in reverse engineering. Therefore, it is vital to construct the history of model building during reconstruction. In our framework, the bottom-up strategy is exploited to reconstruct geometric model, in which 2D profiles are constructed firstly according to parameters of primitive features, then the corresponding primitive features are created by modeling operations, and the final model is properly "assembled" from the created primitive features. Accordingly, the history tree is created by storing the whole modeling elements and operations. In the special tree, the nodes represent the basic elements, intermediate shapes, or final model. Specifically, the leaf nodes could be 2D profiles or the surface features, and the root node is the final model. The edges between nodes represent the operations between elements. Once the model is reconstructed completely, the history tree is also recorded thoroughly. Once one node is modified, it will notify its parent to update. This notification process is recursive and terminates when the root node gets notified and updated.

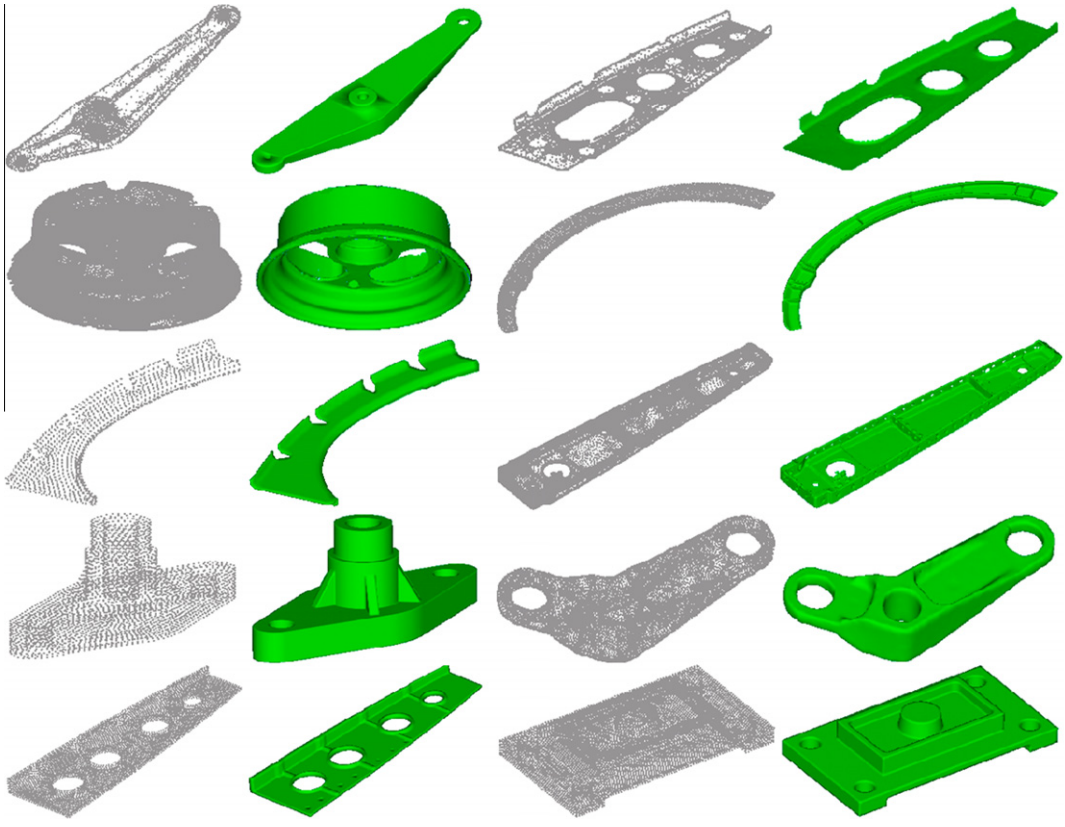

Fig. 12. A few more reconstruction results of industrial parts.

8. Results and discussions

All algorithms described have been implemented with Visual C++ and OpenGL, and run on a PC with 1.8 GHz CPU and 2 GB RAM. A user-friendly GUI has been created encapsulating the implemented algorithms and will be made publicly available to the research community. We have tested our method on a number of industrial models with triangular meshes, and presented a couple of results in this section.

Fig. 9 presents the reconstruction result of a blade part, which consists of 1 loft feature, 2 sweep features, 3 extrude features. There are 175,640 vertices and 351,276 faces in the mesh. To calculate the reconstruction error, the triangulation of the model is implemented. Then we measure as the error the distance from the reconstructed model to the input model. From the error graph, we notice that the reconstruction is relatively accurate. Since there are some human interactions during model reconstruction, we only count the segmentation and feature reconstruction running time here. The total reconstruction time from the input mesh to the final model takes 239.584 s, where the segmentation takes 41.832 s, and the reconstruction of all features takes 197.752 s.

Fig. 10 shows the reconstruction result of the mechanical part of aircraft. There are 1 revolve feature, 4 quadratics surface features, 6 extrude features, 10 plane features in this part. From the figure, the result is favorable. Fig. 11 gives the reconstruction result of a mechanical part. A number of industrial parts have been tested with our method and a few are listed in Fig. 12.

9. Conclusions

A framework is designed to create 3D models from the boundary surface meshes of industrial parts. An efficient mesh denoising method is proposed to filter out model noise and is capable of preserving features of the original model. A divide-and-conquer strategy is exploited to construct all primitive features of parts. Specifically, mesh segmentation is first performed to partition the whole part into primitive feature surfaces. The feature parameter extraction method is adopted to identify and create each type of primitive features. According to the geometric and topological relationship among features, the modeling operations are performed to obtain the final model. During the whole reconstruction process, the constraints and features are considered such that the design intents are fully captured and represented in the final model. Meanwhile, the history of reconstruction is stored to facilitate potential applications for model redesign and modification. With the comprehensive framework, almost all kinds of industrial products could be handled. Although blended features frequently seen in some parts are not discussed specifically here, they can be treated as sweep features (constant radius blend), or loft features (variable radius blend) within our framework.

With the development of 3D scanning technology, our reverse engineering framework has broad industrial applications in automobile, aeronautics, medical areas. In the automobile industry, our method can be used to produce favorable surface models from clays or woods for automobile components. In the medical field, 3D imaging data of biological organs (e.g. heart, brain) can be scanned with medical CT and MRI devices. Quite often these data can be segmented manually or automatically on 2D slices. Our reconstruction algorithm may be adopted to generate 3D geometric models from such contours, which are particularly helpful to physicians or biologists for making medical decisions. Additionally, in the aeronautics industry, there are cases where the original CAD models may not exist, or no longer correspond to the physical parts that were manufactured because of subsequent undocumented modifications that were made after the initial design stage. Our technique can be utilized to reconstruct CAD models from those physically existing parts. More importantly, the models obtained from our framework are feature-based, providing a higher level description of part geometry and allowing easy redesign and reuse for innovation.

Missing data occur as a result of acquisition via laser scanning, due to self-occlusion or unideal conditions of surface materials. How to improve the effectiveness of the reconstruction methods on incomplete data is worthy of studying in the future work.

References

Agathos, A., Pratikakis, I., Perantonis, S., Sapidis, N., & Azariadis, P. (2007). 3D mesh segmentation methodologies for CAD applications. *Computer-Aided Design & Applications, 4*(6), 827–841.

Attene, M., Katz, S., Mortara, M., Patane, G., Spagnuolo, M., & Tal, A. (2006). Mesh segmentation – A comparative study. In *Proceedings of the IEEE international conference on shape modeling and applications* (pp. 14–25).

Attene, M., Falcidieno, B., & Spagnuolo, M. (2006). Hierarchical mesh segmentation based on fitting primitives. *Visual Computer, 22*(3), 181–193.

Au, C. K., & Yuen, M. M. F. (1999). Feature-based reverse engineering of mannequin for garment design. *Computer-Aided Design, 31*, 751–759.

Beccari, C., Farella, E., Liverani, A., Morigi, S., & Rucci, M. (2010). A fast interactive reverse-engineering system. *Computer-Aided Design, 42*, 860–873.

Benko, P., Kos, G., Varady, T., Andor, L., & Martin, R. R. (2002). Constrained fitting in reverse engineering. *Computer-Aided Geometric Design, 19*, 173–205.

Benko, P., Martin, R. R., & Varady, T. (2001). Algorithms for reverse engineering boundary representation models. *Computer-Aided Design, 33*, 839–851.

Bernardini, F., Bajaj, C. L., Chen, J., & Schikore, D. R. (1999). Automatic reconstruction of 3D CAD models from digital scans. *International Journal of Computational Geometry and Applications, 9*, 327–334.

Besl, P. J., & Jain, R. (1998). Segmentation through variable-order surface fitting. *IEEE PAMI, 10*(2), 167–192.

Botsch, M., Pauly, M., Kobbelt, L., Alliez, P., Levy, B., Bischoff, S., et al. (2007). Geometric modeling based on polygonal meshes. In *SIGGRAPH 2007 course notes*.

Chen, X., Golovinskiy, A., & Funkhouser, T. (2009). A benchmark for 3D mesh segmentation. *ACM Transactions on Graphics, 28*(3). http://dx.doi.org/10.1145/1531326.1531379.

Chivate, P. N., & Jablokow, A. G. (1993). Solid-model generation from measured point data. *Computer-Aided Design, 25*(9), 587–600.

Chivate, P. N., & Jablokow, A. G. (1995). Review of surface representations and fitting for reverse engineering. *Computer Integrated Manufacturing Systems, 8*(3), 193–204.

Desbrun, M., Meyer, M., Schroder, P., & Barr, A. H. (1999). Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of the 26th annual conference on computer graphics and interactive techniques* (pp. 317–324).

Durupt, A., Remy, S., & Ducellier, G. (2010). KBRE: A knowledge based reverse engineering for mechanical components. *Computer-Aided Design & Applications, 7*(2), 279–289.

Fisher, R. B. (2004). Applying knowledge to reverse engineering problems. *Computer-Aided Design, 36*, 501–510.

Fleishman, S., Drori, I., & Cohen-Or, D. (2003). Bilateral mesh denoising. In *Proceedings of ACM SIGGRAPH* (pp. 950–953).

Franke, R. (1982). Scattered data interpolation: tests of some methods. *Mathematics of Computation, 38*, 181–200.

Hildebrandt, K., & Polthier, K. (2004). Anisotropic filtering of non-linear surface features. *Computer Graphics Forum, 23*(3), 391–400.

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., & Stuetzle, W. (1992). Surface reconstruction from unorganized points. *Computers & Graphics*, 71–78.

http://www.opencascade.org/.

Huang, J., & Menq, C. H. (2002). Automatic CAD model reconstruction from multiple point clouds for reverse engineering. *Journal of Computing and Information Science in Engineering*, 160–170.

Jones, T., Durand, F., & Desbrun, M. (2003). Non-iterative, feature preserving mesh smoothing. In *Proceedings of ACM SIGGRAPH* (pp. 943–949).

Ke, Y., Fan, S., Zhu, W., Li, A., Liu, F., & Shi, X. (2006). Feature-based reverse modeling strategies. *Computer-Aided Design, 38*(5), 485–506.

Kos, G., Martin, R. R., & Varady, T. (2000). Methods to recover constant radius rolling ball blends in reverse engineering. *Computer-Aided Geometric Design, 17*, 127–160.

Krishnamurthy, V., & Levoy, M. (1996). Fitting smooth surfaces to dense polygon meshes. In: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques* (pp. 313–324).

Langbein, F. C., Marshall, A. D., & Martin, R. R. (2004). Choosing consistent constraints for beautification of reverse engineered geometric models. *Computer-Aided Design, 36*, 261–278.

Lange, C., & Polthier, K. (2005). Anisotropic smoothing of point sets. *Computer-Aided Geometric Design, 22*(7), 680–692.

Lukacs, G., Marshall, A. D., & Martin, R. R. (1998). Faithful least-squares fitting of spheres, cylinders, cones and tori for reliable segmentation. In *Proc. ECCV* (pp. 671–686).

Milroy, M. J., Bradley, C., Vickers, G. W., et al. (1995). G1 continuity of B-spline surface patches in reverse engineering. *Computer-Aided Design, 27*(6), 471–478.
Ohtake, Y., Belyaev, A. G., & Seidel, H. P. (2002). Mesh smoothing by adaptive and anisotropic Gaussian filter. In *Vision, modeling, and visualization* (pp. 203–210).
Park, H., & Kim, K. (1996). Smooth surface approximation to serial cross-sections. *Computer-Aided Design, 29*, 995–1005.
Piegl, L., & Tiller, W. (1997). *The NURBS book* (2nd ed.). New York, NY: Springer-Verlag New York, Inc..
Pottmann, H., & Randrup, W. T. (1998). Rotational and helical surface approximation for reverse engineering. *Computing, 60*, 307–322.
Pratt, V. (1987). Least-squares fitting of algebraic surfaces. *Computers & Graphics, 21*, 145–152.
Protopsaltis, A. (2009). Reconstructing 3D CAD models based on geometrically constrained cross sections. Ph.D. thesis, University of Ioannina.
Shamir, A. (2008). A survey on mesh segmentation techniques. *Computer Graphics Forum, 27*(6), 1539–1556.
Sun, X., Rosin, P., Martin, R., & Langbein, F. (2007). Fast and effective feature-preserving mesh denoising. *IEEE Transactions on Visualization and Computer Graphics, 13*(5), 925–938.
Tagliasacchi, A., Zhang, H., & Cohen-Or, D. (2009). Curve-skeleton extraction from incomplete point cloud. *ACM Transactions on Graphics*, 28.
Taubin, G. (1995). A signal processing approach to fair surface design. In *Proceedings of the 22nd annual conference on computer graphics and interactive techniques* (pp. 351–358).
Taubin, G. (2001). Linear anisotropic mesh filtering. Tech. Rep. IBM Reseasrch Report RC2213.
Thompson, W. B., Owen St, J. C., & Germain, H. J. (1999). Feature-based reverse engineering of mechanical parts. *IEEE Journal of Robotics and Automation, 15*, 57–66.
Varady, T., Facello, M., & Terek, Z. (2007). Automatic extraction of surface structures in digital shape reconstruction. *Computer-Aided Design, 39*, 379–388.
Varady, T., Martin, R. R., & Cox, J. (1997). Reverse engineering of geometric models – An introduction. *Computer-Aided Design, 29*, 25–568.
Vieira, M., & Shimada, K. (2005). Surface mesh segmentation and smooth surface extraction through region growing. *Computer-Aided Geometric Design, 22*(8), 771–792.
Wang, W., Pottmann, H., & Liu, Y. (2006). Fitting B-spline curves to point clouds by squared distance minimization. *ACM Transactions on Graphics, 25*(2), 214–238.
Wang, J., & Yu, Z. (2011). Quadratic curve and surface fitting via squared distance minimization. *Computers & Graphics, 35*(6), 1035–1050.
Werghi, N., Fisher, R. B., Ashbrook, A., & Robertson, C. (1998). Towards object modelling by incorporating geometric constraints. In *Proc IEEE international workshop on model-based 3D image analysis* (pp. 45–53).
Werghi, N., Fisher, R. B., Robertson, C., & Ashbrook, A. (1999). Object reconstruction by incorporating geometric constraints in reverse engineering. *Computer-Aided Design, 31*, 363–399.
Ye, X., Liu, H., Chen, L., Chen, Z., Pan, X., & Zhang, S. (2008). Reverse innovative design – An integrated product design methodology. *Computer-Aided Design, 40*(7), 812–827.
Zheng, Y., Fu, H., Au, O., & Tai, C. (2010). Bilateral normal filtering for mesh denoising. *IEEE Transactions on Visualization and Computer Graphics*. http://doi.ieeecomputersociety.org/10.1109/TVCG.2010.264.