Geometric Segmentation of 3D Scanned Surfaces for Multi-Sensor Coordinate Metrology

Zhiqiang Yu*, Mao Zhang and Jiaoyu Xiao
China Helicopter Research and Development Institute, Tianjin, 300000, China
Email: 1223588099@qq.com

Abstract. In modern industry, multi-sensor metrology methods are increasingly applied for fast and accurate 3D data acquisition. These methods typically start with fast initial digitization by an optical digitizer, the obtained 3D data is analyzed to extract information to provide guidance for precise re-digitization and multi-sensor data fusion. The raw output measurement data from optical digitizer is dense unsorted points with defects. Therefore a new method of analysis has to be developed to process the data and prepare it for metrological verification. This article presents a novel algorithm to manage measured data from optical systems. A robust edge-points recognition method is proposed to segment edge-points from a 3D point cloud. The remaining point cloud is then divided into different patches by applying the Euclidean distance clustering. A simple RANSAC-based method is used to identify the feature of each segmented data patch and derive the parameters. Subsequently, a special region growing algorithm is designed to refine segment the under-segmentation regions. The proposed method is experimentally validated on various industrial components. Comparisons with state-of-the-art methods indicate that the proposed method for feature surface extraction is feasible and capable of achieving favorable performance and facilitating automation of industrial components.

Keywords. 3D point cloud; quadric surface approximation; geometric segmentation; region growing.

1. Introduction
Modern industry tends to increase product quality while reducing production time. Functional, ergonomic and aesthetical parameters are under constant optimization, thus forcing the industrial components to be of more complex shape. All of these put forward higher requirements on the precision and efficiency of measurement techniques. Even though tactile and optical sensing technologies are widely used in data acquisition in dimensional metrology, it has been shown that each technique has its own characteristics and limitations, which lend them to particular applications. To meet the increasing requirement of flexibility and the level of automation of the whole digitization process, multi-sensor methods based on the integration of contact probes with high-speed non-contact digitizers are increasingly applied for 3D data acquisition [1].

Most objects in industry have surface shapes that may be represented by portion of primitive shapes including planes, cylinders, cones, spheres, and so on. Explaining 3D data using simple geometric primitives makes easier any subsequent analysis that would be performed, with consequences on both performances and the ability to perform high-level tasks. In the context of multi-sensor coordinate metrology, when a shape containing multi-patched surfaces is scanned by a optical sensor, it is imperative to divide the acquired point cloud into several smooth regions, each of which logically
belongs to a single primitive surface. On one hand, the extracted features can be used to guide tactile probe to perform high precision measurement [2]. On the other hand, fusion of segmented data patches belonging to different primitive surfaces with the corresponding contact measurement data is helpful to improve the accuracy of non-contact measurement [3]. Therefore, geometry processing, which aims at extracting information about topology, geometry and shape from the measured data is one of the critical issues to achieve multiple sensors integration in coordinate metrology.

There have been quite a few approaches for the geometric segmentation of 3D data have been reported, which can be mainly categorized into three types: edge-based segmentation, region-based segmentation, hybrid methods.

Edge-based methods try to detect the boundary lines between the geometric features shaping the model. This identification is performed based on the analysis of the differential geometric properties, such as curvature [4–8], normal vectors [9,10], local neighborhood distribution [11,12], and so forth. Points are usually classified into two opposite classes depending on certain thresholds: feature points versus non-feature points. However, due to the sensitivity of local surface properties to noise and the smooth effect in estimation results, edge-based methods often produce not accurate results. In 3D space, such methods often detect disconnected edges making the identification of closed segments difficult without a filling or interpretation procedure [13].

Region-based segmentation [14–18] aims to directly identify those specific zones characterized by homogeneous geometric properties. The methods based on this approach generally start from one or more seed points and merge around neighbouring points with similar characteristics, such as surface orientation, curvature, etc.. Region-based methods are relatively less sensitive to the noise in the data, and usually perform better when compared to edge based methods. However, here are some demerits of region-based methods, including that the possibility of over- or under-segmentation, sensitive to inaccurate estimations of the normals and curvatures of points near region boundaries, and the sensitivity to the choice of the initial seed regions.

In order to overcome the limitations involved in the above two kinds of approaches, some hybrid methods combining both edge- and region-based methods have been developed. In [19], K-means clustering method is used to divide points into redundant regions according to points’ principal curvature values. A region growing algorithm is then used to merge adjacent similar regions. In [20], the standard deviation of the point normal vectors within each cell is used in subdivisions of initial cells. The edge points are extracted by selecting the points contained in the small-sized cells. Finally, the segmented point-based data patches are obtained after these edge points have been removed. In [21], triangles in the vicinity of sharp edges are identified and discarded by plane fitting, the remaining triangles are segmented by a particular hierarchy of tests that split the point cloud into increasingly smaller sub-regions. In [22], points are classified as surface, edge, or corner points by applying PCA on the normal vector distribution matrix. Then surface point set are divided by using region growing method with a distance constraint. Finally, the RANSAC algorithm is used to analyze the implicit expression on patches, and the remaining edge and corner points that are not classified are substituted into the implicit expression to expand the patches. A similar approach is put forward by Sitnik et al. in [23], wherein the points are classified as surface, edge points base on local shape coefficient. A rough segmentation algorithm is proposed to split the cloud along the edges. A refine segmentation method is proposed to deal with complex regions that are composed of many primitive and free form shapes. The success of these hybrid methods depends on the success of the underlying methods.

The above method are mostly focus only on segmentation, geometric information contained in point clouds is ignored. Primitive type recognition and model derivation are of great significance for the subsequent analysis of point clouds. Some methods based on random sample consensus (RANSAC) have been proposed to extract primitive shapes from unstructured point cloud [24–27]. RANSAC-based methods is quite fast and robust to outliers. However, the method may result in over or under-segmentation or even wrongly identified primitive types since the RANSAC-based approach only looks for local cues [28]. Li Y et al. [29] proposed a improved RANSAC method by take a global approach to constrain and optimize the local RANSAC-based primitives.
Although various segmentation methods have been proposed, segmentation of raw point clouds is still a challenging problem due to lack of the topological structures or the mathematical model of the input data, geometry shape complexity and noise (outliers). In this study, we propose a novel segmentation framework to extract feature surfaces from from raw 3D point cloud of industrial components. This method is designed to improve the automation and reliability of feature surface extraction, extract high-level surface geometric feature information from the original point cloud data, and provide effective routes for multi-sensor coordinate metrology and data fusion.

2. Methods
The proposed method uses statistics-based method to classify the points into two categories: surface points and edge points. For the set of surface points, Euclidean distance clustering is applied to obtain initial divided patches. These patches can be classified as geometric, complex and free-form surfaces according to their shape. For complex surfaces, we propose a special region growth strategy based on standard deviation of points’ normals in a neighbourhood to realize refine segmentation. In this study, a simple random sample consensus (RANSAC) algorithm is used to reliably recognize a variety of geometric primitives such as plane, cylinder, cone and sphere surface. For each recognized geometric feature, least squares best fitting method is used to derive surface parameters. Finally, a patch optimization strategy is proposed to optimize the segmentation results according to the derived parametric equations. The flowchart of the proposed method is shown in figure 1.

![Flowchart of the geometric segmentation method](image)

**Figure 1.** Flowchart of the geometric segmentation method.
2.1. Feature Points Detection

2.1.1. Normal Estimation. Principal component analysis (PCA) of the covariance matrix of a local neighborhood is widely used to estimate local surface properties, such as normal and curvature [30]. Assuming that the point cloud \( P = \{p_i\}_{i=1}^N \) is sampled from piecewise smooth surfaces, \( p_i \) is a point on this surface, and let \( N_i \) be the neighborhood of \( p_i \). For each point \( p_i \) a neighborhood \( N_i \) of size \( K_0 \) is computed. The corresponding covariance matrix \( C \) is defined as:

\[
C = \begin{bmatrix}
p_1 - \bar{p} \\
\vdots \\
p_{K_0} - \bar{p}
\end{bmatrix}^T \begin{bmatrix}
p_1 - \bar{p} \\
\vdots \\
p_{K_0} - \bar{p}
\end{bmatrix}, p_i \in N_i
\]

where, \( \bar{p} \) is the centroid of the set of neighbors in \( N_i \).

The eigenvalues \( \{\lambda_0, \lambda_1, \lambda_2\} \) of the covariance matrix can be estimated by analyzing the following eigenvector problem:

\[
C \cdot \hat{v}_j = \lambda_j \cdot \hat{v}_j, j \in \{0,1,2\}
\]

where \( \{\hat{v}_0, \hat{v}_1, \hat{v}_2\} \) are corresponding eigenvectors.

Assume which \( \lambda_0 < \lambda_1 < \lambda_2 \), then the surface variation \( \sigma_i \) of the underlying surface at point \( p_i \) and an initial normal \( n_i \) can be estimated. The surface variation \( \sigma_i \) is defined as:

\[
\sigma_i = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2}
\]

The normal \( n_i \) is defined as the eigenvector \( \hat{v}_0 \) corresponding to the smallest eigenvalue \( \lambda_0 \).

2.1.2. Feature Points Detection. In [31-33], \( \sigma_i \) is considered as a curvature and used to distinguish feature points, which have a more complicated neighborhood, from non-feature points. The coefficient of a point near sharp features is considered to be larger than that of a point in smooth regions. Therefore, each point \( p_i \) with \( \sigma_i \) greater than a threshold \( \sigma_\tau \) is viewed as a feature point.

However, it has been shown that PCA is sensitive to noise and outliers due to using classic covariance matrix. As a result, attributes calculated through PCA are highly sensitive to noise and outliers [34]. Figure 2 shows the results of feature point identification under different noise levels using above curvature-based method. For each point cloud and noise level, the curvature threshold \( \sigma_\tau \) is set to the optimal value. It can be seen that when the noise level is up to 70%, curvature-based method fails to accurately identify feature points. Large tracts of points in smooth regions are mistakenly identified to be feature points, which will subsequently affect the accuracy and efficiency of our normal estimation algorithm.

To improve the accuracy and robustness of feature point recognition, a statistics-based feature point recognition strategy is proposed in this paper. For each point \( p_i \), the initial normal is computed using PCA mentioned above, and then the standard deviation of the point normal values within the point’s \( K_2 \) neighborhood is computed. The standard deviation of point normal (SDN) is calculated by:

\[
\sigma_n = \sqrt{\frac{\sum_{j=0}^{K_2} (n_j - \bar{n})^2}{K_2}} = \sqrt{\frac{\sum_{j=0}^{K_2} (n_{j_i} - \bar{n})^2 + \sum_{j=0}^{K_2} (n_{j_e} - \bar{n})^2 + \sum_{j=0}^{K_2} (n_{j_z} - \bar{n})^2}{K_2}}
\]

where \( \bar{n}_{j_i} \) is the unit normal vector of each point in \( p_i \)’s neighborhood. \( \bar{n} \) is the average normal vector calculated from the normalized point normals within \( p_i \)’s neighborhood by:
\[
\bar{n}_i = \sum_{j=0}^{K_i} n_j / K_2
\] (5)

Similar to \(\sigma_i\), the standard deviation of points’ normals in a neighborhood indicates the level of changes in the shape of the underlying surface at point \(p_i\). Each point \(p_i\) with normal standard deviation larger than the user-defined threshold \(\sigma_{nT}\) is viewed as a feature point. In this way, the feature points are identified by statistical analysis of all points’ normals in the neighborhood, which is more robust than judging only by the curvature of a single point.

Figure 3 shows the result of our feature point identification algorithm. The standard deviation tolerance \(\sigma_{nT}\) for model Fandisk, Revolved part and Anchor are set to 0.21, 0.17 and 0.19 respectively. It can be seen that the proposed algorithm shows good robustness to noise, it can accurately identify feature points under different noise levels.

**Figure 2.** Feature points detection using curvature-based method. Points in red color are identified feature points, and non-feature points are shown in green points. From left to right, they are Fandisk, Revolved part and Anchor.

**Figure 3.** Feature points detection using statistics-based method. Points in red color are identified feature points, and non-feature points are shown in green points. From left to right, they are Fandisk, Revolved part and Anchor.
2.2. Rough Patch Segmentation

To improve the segmentation speed of our method, initial rough segmentation is used to split the cloud along the edges and limit the amount of points being processed at a time. In this paper, the Euclidean distance clustering method is applied to fast separate the surface point set \( S \). After separating the edge points from the point cloud, the remaining surface point set \( S \) is naturally divided into several separated point sets in space domain figure 4(a). Then Euclidean distance clustering [35] is used to segment the point set into several smooth data patches. The result of rough segmentation is shown in figure 4(b).

The key to surface point segmentation is the selection of distance threshold \( d \) used in Euclidean distance clustering. An inappropriate distance threshold may lead to over segmentation or under segmentation. The distance threshold usually changes with the density variation in the point cloud model or between models. To improve the algorithm’s performance and practicability, an adaptive distance threshold is essential. To this end, a sample distance threshold determination procedure is introduced. Firstly, one-tenth points are randomly selected from surface point set \( S \). And then, the local point spacing \( \delta \) is estimated as the average distance of these points from their nearest neighbors [36]. Finally, the distance threshold is adaptively set as \( d = 2.5 \delta \).

The relationship between \( d \) and \( \delta \) is determined by experimental test on different point cloud models. Experimental test on different point cloud models shows that the distance threshold \( d \) has a large value range (figure 5). This is because the identified edge points form a banded area with a certain width that separates different surface patches as shown in figure 3. Experiments show that the distance threshold is selected as 2-3 times the local point spacing is appropriate. Therefore, the distance threshold is determined to be \( d = 2.5 \delta \) in this paper.

---

**Figure 4.** Rough segmentation. **a** Initial surface point set, **b** result of Euclidean distance clustering.
Surface Feature Recognition and Parameterization

Geometric surfaces are typical structures in industrial components, 85% of mechanical component’s surface can be represented by plane, spherical, cylindrical, and conical surface [22]. Determining if the discrete points are on the surface is easy if the parameter expression is known. Therefore, it is necessary to recognise and derive the parameter expression of these geometric surfaces. In this paper, the RANSAC algorithm is adopted for this task.

The primitive shapes considered in this paper are planes, spheres, cylinders and cones which have between three and seven parameters. With the point coordinates and normal vector, two point samples are enough to estimate each of the considered primitive shapes. However, we using one additional sample for preliminary verification and eliminating the unnecessary evaluation of many low scored shapes, thus reduce calculation.

3D Plane Model

A plane can be specified by a point \((x_0, y_0, z_0)\) on the plane and the direction cosines \((a, b, c)\) of the normal to the plane. Any point \((x_i, y_i, z_i)\) on the plane satisfies:

\[
a(x_i - x_0) + b(y_i - y_0) + c(z_i - z_0) = 0
\]  

(6)

For a plane, three points \(\{P_1, P_2, P_3\}\) and the corresponding normal vectors \(\{n_1, n_2, n_3\}\) are used to estimate the parameter. The deviations of the points’ normals \(n_1, n_2, n_3\) are calculated, and the candidate plane is accepted only if all deviations are less than the predefined angle \(\alpha\). The average coordinates and normal vectors of the three points can represent the equation of the candidate plane.

3D Sphere Model

All points on the spherical surface have the same distance from the center of the sphere, and the normal vector of these points is toward the center of the sphere. On the basis of the geometry of the sphere, a sphere can be defined by two points with corresponding normal vectors:

\[
\begin{align*}
    P_1 - C_s &= R_s \cdot n_1 \\
    P_2 - C_s &= R_s \cdot n_2
\end{align*}
\]  

(7)
Center coordinates $C_s$ and radius $R_s$ of the sphere can be obtained by substituting points $P_1(x_1, y_1, z_1)$, $P_2(x_2, y_2, z_2)$ and their normal vectors $n_1(x_{n1}, y_{n1}, z_{n1})$, $n_2(x_{n2}, y_{n2}, z_{n2})$ into Equation (10).

The sphere is accepted as a shape candidate only if all three points are within a distance of $e$ of the sphere and their normals do not deviate by more than $\alpha$ degree.

- **3D Cylinder Model**

A cylinder can be specified by a point $(x_0, y_0, z_0)$ on its axis, a vector $(a, b, c)$ pointing along the axis and its radius $R_c$. Given two points $P_1(x_1, y_1, z_1)$, $P_2(x_2, y_2, z_2)$ and their normal vectors $n_1(x_{n1}, y_{n1}, z_{n1})$, $n_2(x_{n2}, y_{n2}, z_{n2})$, the direction of the axis is estimated as: $N_c = n_1 \times n_2$. Then two parametric lines $P_1 + t n_1$ and $P_2 + t n_2$ are projected along the axis onto the $N_c \cdot x = 0$ plane and the center of the cylinder is estimated as the intersection of the two projected lines. The distance between $C_c$ and $P_1$ in that plane is set as the radius. Afterwards, the cylinder is verified by applying the thresholds $e$ and $\alpha$ to distance and normal deviation of the three points.

- **3D Cone Model**

A cone can be specified by the apex $(x_0, y_0, z_0)$, a vector $(a, b, c)$ pointing along the axis and the apex semi-angle $\theta$. All these parameter can be determined by two points with corresponding normal. The position of the apex $C_c$ is derived by the intersection of the three planes defined by the three point and normal pairs. Then the direction of the axis $N_c$ is set as the normal of the plane defined by the three points $\{ C_c + \frac{P_1-C_c}{\|P_1-C_c\|}, C_c + \frac{P_2-C_c}{\|P_2-C_c\|}, C_c + \frac{P_3-C_c}{\|P_3-C_c\|} \}$. The opening angle $\theta$ is given as $\theta = \frac{\sum \arccos((P_i-C_c, N_c))}{3}$. Again, the cone is verified in the same way as sphere and cylinder before becoming a candidate shape.

Given a segmented data patch $S_i$, three sample points are randomly selected to calculate the model parameters for the four models and the inliers of the four models using the RANSAC algorithm. The model type and inliers are recorded when the number of inliers is larger than the threshold, and the recording times are marked as the frequency of the model to which the patch belongs. After multiple cycles, the ratio of model frequency to the total number of cycles is recorded as the probability of the model to which the patch belongs. The model with the maximum likelihood is considered to be the optimal expression of the surface. The specific steps of the algorithm are as follows:

1. Three points are randomly selected from $S_i$, when the number of iteration is less than $n_1$, the angle deviation between the normal vectors of the three points are calculated, and Step 2 is executed. When the number of iteration reaches $n_1$, Step 4 is executed.

2. The patch may be a plane if the deviations of the three normal vectors are less than the predefined angle $\alpha$. The parameters of the plane are calculated, and the points satisfying distance and normal deviation constraints are considered as inliers of the parametric model. The frequency of the plane model is incremented by 1 when the number of inliers is greater than the threshold $n_2$, and the optimal parameters are updated and Step 1 is repeated. Otherwise, Step 3 is executed.

3. The parameters of the cylindrical, sphere and cone model are calculated. The inliers of the parametric model are counted when the preliminary verification is satisfied. The frequency of the cylindrical, sphere or cone model is incremented by 1 when the number of inner points is greater than the threshold, and the optimal parameters are updated. The frequency of the free-form surface is incremented by 1 when none of the four shapes’ inner points is greater than the threshold. Continue to Step 1.

4. The maximum probability model is calculated, all the model inliers under the corresponding optimal model parameters are optimized using the least squares method, and the obtained model parameters are the final outputs.

2.4. Refine Segmentation

Through rough segmentation and feature recognition, the point cloud is classified into three types of data patches according to their underlying geometric feature: geometric, free-form, complex data patches. A complex data patch is region that is composed of two or more geometric features and
free-form surfaces. The smooth transition between these features leads to the inability of rough segmentation to extract these features. Therefore, a refine process is needed to further segment the complex data patch.

In this paper, a special designed region growing algorithm is applied to realize refine segmentation of complex data patches. The proposed region growth algorithm is unique in that it uses SDN as the measure of similarity for region growth. The used feature is more robust than curvature or normal as stated in section 2.1 and is thus helpful to improve the reliability of segmentation. In addition, the criterion to measure whether a point can be selected as seed point is no longer a simple curvature threshold, but the difference between the SND of the considered point and the initial seed point. In this way, the points in the same region after segmentation have the same normal behaviour, which ensures that the different features of smooth transition are not under-segmented. The flowchart for dividing complex data patch by using the region growing algorithm is shown in figure 6.

![Flowchart](figure6.png)

**Figure 6.** Flowchart of refine segmentation for complex patches.
Given an under-segmented point set S, the method for refine segmentation is implemented as follows:

1. A classification flag \( f_c \) is set for each point in surface point set S. The flag is initialized to \(-1\), and the number of sorts, \( n \), is initialized to 0. Sort the index of points according to their NSD.

2. The algorithm is terminated when all the points in point set S are extracted. Otherwise, \( n \) is incremented, and a point \( p_i \) is extracted from point set S according to the sorted point index to determine whether its classification flag \( f_c \) is \(-1\) one by one. \( f_c \) is regarded as a seed point, marked as \( n \), and stored in the temporary point set \( Q \) when its value is \(-1\). If \( f_c \) is not \(-1\), another point is obtained from S.

3. If set \( Q \) is empty, then Step 2 is repeated. Otherwise, element \( q_i \) is removed from \( Q \), and the \( R \) nearest neighbors set of \( q_i \) is searched.

4. Flag \( f_c \) of point \( r_j \) in the R nearest neighbors set of \( q_i \) is marked as \( n \) when its value is \(-1\) and the condition \( |p_i,\text{NSD} - r_j,\text{NSD}| \leq g_{th} \) is satisfied. \( r_j \) is regarded as a seed point and stored in set \( Q \) when the condition \( |p_i,\text{NSD} - r_j,\text{NSD}| \leq s_{th} \) is satisfied.

The points in set S with an equal flag \( f_c \) are on the same patch. The points are stored based on their category, and each element of \( S = \{S_1, S_2, \ldots, S_n\} \) is a category. The original point display is shown in figure 7(a) and (c), and the classification display is presented in figure 7(b) and (d), where point sets with different colors represent different patches.

**Figure 7.** The result of refine segmentation, under-segmentation regions are correctly divided into several independent regions via region growing algorithm.
2.5. Patch Optimization

2.5.1. Point Merging. Some small data patches containing only a small number of points are extracted after refine segmentation due to the presence of noise/outliers. In addition, some points located at the junction of different primitives are extracted and recognized as free-form surfaces. These regions often belong to the same regular surface as adjacent larger data patches, so we proposed a merging strategy to add these points to the adjacent data patches.

For each point that belongs to free-form surfaces or areas of low point count, its $R$ nearest neighbors is computed and which data patches do the points included in the neighbourhood belong to are counted. The parametric expression of each included patch is easy to obtain from the solved model parameters of the patch. The deviations from current considered point to each calculated solid of the neighbouring data patch are calculated after substituting the point coordinate into the parametric equations. The considered point is added to the data patch that satisfies the following two conditions:

- The deviation from the point to the calculated solid of the data patch is the minimum among all the included data patches.
- The calculated minimum deviation can’t be much greater than the average noise of the cloud, usually within 3 times the value of noise.

Point merging is a multi-pass operation and performed until there are no more possible points that can be merged. The segmentation result after point merging is shown in figure 8.

![Figure 8](image.png)

**Figure 8.** The result of point merging process, areas of low point count are merged into adjacent data patches.

2.5.2. Patch Expansion. Considering that edge point set contain several surface points during point cloud segmentation, these points are not assigned to any regions. The patch should be expanded to obtain a complete patch. Therefore, we present a patch expanding strategy to assign each edge point to the patch it belongs to.

For each edge point $p_i \in E$, we first add it to the surface point set $S$. Then its $R$ nearest neighbors is computed and which data patches do the surface points included in the neighbourhood belong to are counted. The deviations from current edge point to each neighbouring primitive are calculated. Finally, the point is deemed to belong to the surface patch corresponding to minimum deviation. Sometimes, noise/outliers are recognized as edge points. To filter outliers or high noise points, the deviation of the considered point from the calculated solid can’t be much greater than the average noise of the cloud. On this basis, the edge points can be classified into the patch they belong to.

The results of the original patch before and after expansion are displayed in figure 9. The regular surface exhibits significant expansion in space, as shown in figure 9(b).
Figure 9. The result of patch expansion process, edge points are added to the corresponding point data patches.

2.5.3 Patch Merging. Over-segmentation occurs in patch classification. The two patches selected by the black in figure 10(a) are divided into several surfaces, and the patches that belong to the same surface should be combined. On the basis of the surface parametric expression calculated by the least squares fit algorithm, we can determine if the patches belong to the same spatial surface. When patches belong to the same spatial surface and the distance between the nearest points of the patches is less than the distance constraint, the patches are merged.

In this manner, expansion and optimization of all regional surfaces are completed, and the point cloud model is divided into a number of feature surfaces according to shape characteristics (figure 10(b)).

Figure 10. The result of patch merging process, patches that belong to the same surface are combined.

3. Parameter Tuning
The parameters used in this paper are listed in table 1. In edge-points recognition, the main parameters are $K_0$, $K_1$ and $\sigma_{nr}$, the value of these parameters are fixed. Experiments show that the values of these parameters are suitable for all the models used in this article. Parameter $d_e$ used for Euclidean distance segmentation is adaptively determined using the method described in Section 2.2. It is worthy to note
that, incorrect set the parameter $d\tau$ only increase the run time of our method in refine segmentation stage, is unlikely to degenerate our result. The parameters used in refine segmentation are determined empirically. These are the only two parameters that need to be set manually. The value of the two parameters is affected by the neighbourhood size and noise level. Experiments show that the range of $s_{th}$ is 4–10, the value of $g_{th}$ is larger than that of $s_{th}$, usually 15–20. R used for patch optimization is set to a value that is 10 times larger than average point spacing $\delta$. A sample distance threshold determination procedure is introduced to adaptively determine the value of $\varepsilon$. Firstly, one percent points are randomly selected from surface point set $S$. And then, a least squares plane is computed for each selected point by applying PCA on the point’s $K_0$ neighbourhood and the median value of the deviations of all neighbourhood points from the fitting plane are computed as $d_{mi}$. The noise of the point cloud is estimated as: noise = $\frac{1}{m} \sum_{i=1}^{m} d_{mi}$, where $m$ is the number of selected points. Finally, the value of $\varepsilon$ is set as: $\varepsilon = 3 \times$ noise. The angle threshold is fixed as $\alpha = 5^\circ$. The larger the value of $n_1$ is, the more accurate the result is, and the lower the computational efficiency is. The cycle time is set as $n_1 = 500$ in this paper. The value of $n_2$ is set as the 90% the size of a point cloud. It can be seen that the key parameters of our algorithm are $g_{th}$ and $s_{th}$ used for region growing. Other parameters are either determined adaptively or set to a fixed value.

| Parameter | Function module | Describe | Determination |
|-----------|----------------|----------|--------------|
| $K_0$     | Edge-points recognition | Neighbourhood size for initial normal vector estimation | Fixed (50) |
| $K_1$     | Neighbourhood size for normal vector standard deviation calculation | Fixed (20) |
| $\sigma_{nt}$ | Threshold for edge point recognition | Fixed (0.19) |
| $d\tau$  | Distance threshold for Euclidean distance segmentation | Adaptive |
| $g_{th}$ | Threshold of point merging for region growth | Empirical |
| $s_{th}$ | Threshold of seed point selecting for region growth | Empirical |
| $R$       | Patch optimization | Neighbourhood search range for patch optimization | Adaptive |
| $\varepsilon$ | Distance threshold for determining whether a point belongs to a model | Adaptive |
| $\alpha$ | Angle threshold for determining whether a point belongs to a model | Fixed (5°) |
| $n_1$     | Cycle time of RANSAC algorithms | Fixed (500) |
| $n_2$     | Frequency threshold to determine whether a candidate feature is accepted | Adaptive |

4. Results
To verify the effectiveness of our algorithm, we run our algorithm on a series of mechanical models (figure 11). It can be seen that our algorithm can deal with various complex models with different primitive shapes. The point clouds are correctly segmented into data patches, which illustrates the effectiveness and accuracy of our algorithm.
Figure 11. The test results of our algorithm on different models.

Figure 12 shows a side-by-side comparison of our segmentation results with that of the RANSAC-based approach [24] and region growing method [36]. As we can see region growing method fails to divide regions with smooth transition and is prone to under-segmentation. The RANSAC-based results are much better, but still produce some incorrect segmentation and under-segmentation when dealing with complex models. Our result, in contrast, segments a better level of details, point data patches belonging to different surfaces are accurately extracted.

Figure 12. Comparison between our segmentation, RANSAC-based segmentation and region growing segmentation method.
To quantitatively evaluate the performance of the proposed method, we refer to [22] and evaluate the results of our algorithm at the point level by using four scores: Corr (correctness rate of segmentation), Comp (completeness rate of segmentation), Qual (quality of segmentation), and T (execution time of the method). The first three scores pertain to accuracy, and the last pertains to time. The accuracy scores are defined as follows:

\[
\text{Corr} = \frac{N_{tp}}{N_{tp} + N_{mp}}
\]

(8)

\[
\text{Comp} = \frac{N_{tp}}{N_{tp} + N_{op}}
\]

(9)

\[
\text{Qual} = \frac{N_{tp}}{N_{tp} + N_{mp} + N_{op}}
\]

(10)

where \(N_{tp}\) is the number of points correctly classified, \(N_{mp}\) is the number of misclassified points, and \(N_{op}\) is the number of omitted points. The sum of the three types of points is the total number of points in the model. Larger values of Comp, Corr and Qual and a small value of T indicate good performance of the algorithm.

The joint, anchor and fandisk models are evaluated, the details of the three models are listed in table 2. To compute the four scores for evaluating the surface segmentation results, we count the number of the three types of points to which the detected points belong in each model and the execution time. The proposed method is implemented using MATLAB. Region growing method is programed based on the Point Cloud Library (PCL), while RANSAC-based method is obtained in C++ version. All the experiments have been performed on the same computer with 1 CPU Inter(R) Core(TM) i3-3240M 3.40GHZ and 4 GB RAM without parallel computing. The performance statistics are shown in tables 3.

As shown in table 3, the completeness of each method is similar, all exceed 99%. Region growing method achieves the lowest correctness and quality scores on all the three cloud models due to the under-segmentation. RANSAC-based method performs much better than region growing method, both of the scores tested on the three models exceeded 90%. Our method achieve highest scores in most cases, the three scores on all the three models exceeded 97%. There is no doubt that our method yields the best overall quality with very low variance.

In terms of computational efficiency, RANSAC-based method is very effective, feature extraction in the three models can be completed within 10s. Region growing method is also computational efficiency. Our method is the most time-consuming. Table 4 gives the performance of our algorithm with detailed timings on each step. The performance depends on the complexity of the model and its size. The edge-points recognition and edge-point processing steps are very fast, because no complex algorithms are included and the edge-point processing step only has to deal with a relatively small input. The rough segmentation and refined segmentation steps are major time-consuming stages in our framework. It is worth noticing that our method applied in this paper is in MATLAB version, which has a negative effect in our algorithm. On the other hand, this means our method has more potential in efficiency.

| Model | Points | Surfaces | Average Point Spacing (dm) | Bounding Box Size |
|-------|--------|----------|---------------------------|-------------------|
| Joint | 60243  | 12       | 0.009423                  | 0.838 1.117 1.066 |
| Anchor | 99752  | 23       | 0.005223                  | 1.379 0.871 1.174 |
| Fandisk | 60105  | 21       | 0.0228363                 | 4.879 5.278 2.726 |
Table 3. Comparison of performance of different methods on different models.

| Model   | Method                | Segmented regions | N_{tp} | N_{mp} | N_{op} | Corr (%) | Comp (%) | Qual (%) | Time(s) |
|---------|-----------------------|-------------------|--------|--------|--------|----------|----------|----------|---------|
| Joint   | Region growing        | 10                | 54087  | 6156   | 0      | 89.78    | 100      | 89.78    | 31.013  |
|         | RANSAC                | 12                | 59016  | 1195   | 32     | 98.02    | 99.95    | 97.96    | **2.616** |
|         | Our method            | 12                | 59877  | 342    | 17     | **99.42**| 99.97    | **99.39**| 167.597 |
| Anchor  | Region growing        | 12                | 73943  | 25551  | 258    | 74.32    | 99.65    | 74.13    | 48.472  |
|         | RANSAC                | 23                | 98224  | 1036   | 492    | 98.96    | 99.5     | 98.47    | **5.09** |
|         | Our method            | 23                | 98780  | 816    | 156    | **99.18**| **99.84**| **99.03**| 458.763 |
| Fandisk | Region growing        | 13                | 34319  | 25614  | 172    | 57.26    | 99.5     | 57.1     | 20.213  |
|         | RANSAC                | 21                | 55507  | 3242   | 336    | 92.87    | 99.4     | 92.35    | **7.509** |
|         | Our method            | 21                | 58504  | 1287   | 59     | **97.43**| **99.9** | **97.34**| 157.245 |

Table 4. Timings statistics (in seconds) of each step in our method on processed models.

| Model   | Points | Edge-points recognition | Rough segmentation | Refined segmentation | Edge-point processing | Total |
|---------|--------|-------------------------|--------------------|----------------------|-----------------------|-------|
| Joint   | 60243  | 8.812282                | 85.302346          | 49.851530            | 3.238337              | 147.2045 |
| Anchor  | 99752  | 20.534734               | 138.047876         | 287.312412           | 4.6249                | 450.5199 |
| Fandisk | 60105  | 8.377107                | 85.108551          | 50.465183            | 10.324102             | 154.2749 |

Figure 13 shows the comparison between the RANSAC-based method, region growing method and ours on the joint model under different levels of noise. The noise used in this paper is Gaussian noise, with different standard deviation as % of the mean distance between points. When the noise level is less than 40% (the third column), each approach gives consistently segmentation result. However, when noise continues to increases, region growing method tends to omit points in the large curvature region. To correct this error, we try to increase the curvature threshold of the algorithm, but this result in more under-segmentation regions. Our method and RANSAC-based method keep consistently performance even when the model is heavily corrupted by noise (the last column) (σ = 100%), the primitives are extracted in good shapes.

Figure 13. Comparison of the robustness between region growing segmentation, RANSAC-based segmentation and our segmentation on the joint model. Noise (on both points’ positions and normals) ranges from 0% to 100% and increases at equal intervals from left to right.
5. Conclusions
The algorithm presented in this article can robustly segment a wide range of measured data from optical measurement systems and prepare it for subsequent application, such as metrological analysis, data fusion of different sensors, model reconstruction. The method analyzes the neighborhood normal vector standard deviation to segment edge-points from a 3D point cloud. Then, an Euclidean distance clustering based rough segmentation method is proposed to realize initial segmentation. A robust feature recognition and parameterization method is used to recognize the features of point data patches as geometric surface, free-form surface, complex regions. For under-segmented complex regions, a novel region growing strategy is designed to further segment the point clouds into geometric features and free-form surfaces. Experimental results show that our approach compares favorably with existing classical methods such as the RANSAC-based and region growing approaches in terms of robustness and accuracy.

In our future studies, we plan to extend our method to handle a broader class of primitives. In addition, although we have provided the guidance for parameter selection, setting appropriate parameters is still difficult for non-professionals because of the lack of adaptive setting of the parameter threshold. Subsequent studies may consider the adaptive adjustment of model parameters.

References
[1] Lu K and Wang W 2015 A multi-sensor approach for rapid and precise digitization of free-form surface in reverse engineering The International Journal of Advanced Manufacturing Technology 79(9-12) 1983-1994.
[2] Sladek J, Blaszczyk P M, Kupiec M and Sitnik R 2011 The hybrid contact-optical coordinate measuring system Measurement 44(3) 503–510 doi:10.1016/j.measurement.2010.11.013
[3] Li F, Longstaff A P, Fletcher S and Myers A 2014 Rapid and accurate reverse engineering of geometry based on a multi-sensor system Int. J. Adv. Manuf. Technol. 74(1-4) 369-382. doi: 10.1007/s00170-014-5997-y
[4] Merigot Q, Ovsjanikov M and Guibas L 2010 Voronoi-based curvature and feature estimation from point clouds IEEE Trans. Comput. 17 743-756.
[5] Altantsetseg E, Muraki Y, Matsuyama K and Konno K 2013 Feature line extraction from unorganized noisy point clouds using truncated Fourier series Vis. Comput. 29 617-626.
[6] Demarsin K, Vanderstraeten D and Roose D 2013 Meshless extraction of closed feature lines using histogram thresholding Comput.-Aided Des. Appl. 5 589-600.
[7] An Y 2011 Geometric Properties Estimation and Feature Identification from 3D Point Cloud (Dalian: Dalian University of Technology).
[8] Zou D 2012 The Feature Extraction and Segmentation Analysis of Point Clouds (Nanjing: Nanjing Normal University).
[9] Zhang Y, Geng G, Wei X, Su H and Zhou M 2016 Point clouds simplification with geometric feature reservation J. Comput.-Aided Des. Comput. Graph. 28 1420–1427.
[10] Angelo L and Stefano P 2015 Geometric segmentation of 3D scanned surfaces Comput. Aided Des. 62 44–56.
[11] Liu Z, Jiang K and Lin J 2016 Interactive extraction of boundary of specified target feature on scattered point cloud Comput. Eng. Appl. 52 186–190.
[12] Qin J, Wan Y, He P, Chen M, Lu, W and Wang S 2015 An automatic building boundary extraction method of TLS Data Remote Sens. Inf. 30 3–7.
[13] Grili E, Menna F and Remondino F 2017 A review of point clouds segmentation and classification algorithms ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XLII–W3:339-344.
[14] Mizoguchi T, Date H, Kanai S and Kishinami T 2006 Segmentation of scanned mesh into analytic surfaces based on robust curvature estimation and region growing In Proceedings of the 4th International Conference on Geometric Modeling and Processing p 644–654.
[15] Vo A, Truong-Hong L, Laefer D and Bertolotto M 2015 Octree-based region growing for point cloud segmentation ISPRS J. Photogramm. Remote Sens. 104 88–100.
[16] Wang X, Zou L, Shen X, Ren Y and Qin Y 2017 A region-growing approach for automatic outcrop fracture extraction from a three-dimensional point cloud Comput. Geosci. 99 100–106.
[17] Jagannathan A and Miller E L 2007 Three-dimensional surface mesh segmentation using curvedness-based region growing approach IEEE Trans. Pattern Anal. Mach. Intell. 29(12) 2195–204.
[18] Gelfand N and Guibas L J 2004 Shape segmentation using local slippage analysis In: Proc. Eurographics Symposium on Geometry Processing p 219–228.
[19] Lavoué G, Dupont F and Baskurt A 2005 A new CAD mesh segmentation method based on curvature tensor analysis Computer-Aided Design 37(10) 975–87.
[20] Woo H, Kang E, Wang S Y and Lee K H 2002 A new segmentation method for point cloud data International Journal of Machine Tools & Manufacture 42(2) 167–78.
[21] Benko P and Varady T 2004 Segmentation methods for smooth point regions of conventional engineering objects Comput-Aided Des. 36(6) 511–23.
[22] Wang Y, Wang J, Chen X, et al. 2018 Feature surface extraction and reconstruction from industrial components using multistep segmentation and optimization Remote Sensing 10(7).
[23] Sitnik R and Blaszczyzk P M 2012 Segmentation of unsorted cloud of points data from full field optical measurement for metrological validation. Computers in Industry 63(1) 30-44.
[24] Schnabel R, Wahl R and Klein R 2007 Efficient RANSAC for point-cloud shape detection Comput. Graph. Forum 26 214–226.
[25] Arikan M, Wimmer M and Maierhofer S 2013 O-snap: optimization-based snapping for modeling architecture ACM Trans. Graph 32 1–15.
[26] Zeineldin R and El-Fishawy N 2017 A survey of RANSAC enhancements for plane detection in 3D point clouds Mechnofa J. Electron. Eng. 26 519–537.
[27] Wu S 2011 Feature Extraction and Shape Detection in Reverse Engineering (Hangzhou: Zhejiang University).
[28] Isack H and Boykov Y 2012 Energy-based geometric multi-model fitting Int. J. Comput. Vis. 97 123–147.
[29] Li Y, Wu X, Chrysaouthi Y, Sharf A, Cohen-Or D and Mitra N J 2011 GlobFit: consistently fitting primitives by discovering global relations In: ACM TOG, SIGGRAPH 52 1–12.
[30] Pauly M, Gross M and Kobbelt L P 2002 Efficient simplification of point-sampled surfaces In: Proceedings of the IEEE Conference on Visualization p 163-70.
[31] Zhang J, Cao J, Liu X, Wang J, Liu J and Shi X 2013 Point cloud normal estimation via low-rank subspace clustering Comput Graphics 37(6) 697-706.
[32] Liu X, Zhang J, Cao J, Li B and Liu L 2015 Quality point cloud normal estimation by guided least squares representation Comput. Graphics 51(1) 106-16.
[33] Cao J, Chen H, Zhang J, et al. 2017 Normal estimation via shifted neighborhood for point cloud J. Comput. Appl. Math. 329 57-67.
[34] Hubert M, Rousseeuw P J and Branden K V 2005 ROBPCA: a new approach to robust principal component analysis Technometrics 47 (1) 64-79.
[35] Assem I and Dupont G 2011 Friezes and a construction of the Euclidean cluster variables Journal of Pure & Applied Algebra 215(10) 2322-40.
[36] Point Cloud Library (PCL) 2019 Region Growing Segmentation Available online: http://pointclouds.org/documentation/tutorials/region_growing_segmentation.php#region-growing-segmentation