Constructing 2D Curves from Scanned Data Points Using *B*-spline Wavelets

Y. F. Wu[1], Y. F. Zhang[2], Y. S. Wong[3] and H. T. Loh[4]

[1]National University of Singapore, engp1547@nus.edu.sg
[2]National University of Singapore, mpezyf@nus.edu.sg
[3]National University of Singapore, mpewys@nus.edu.sg
[4]National University of Singapore, mpelht@nus.edu.sg

ABSTRACT

This paper describes a method for automatic reconstruction of concise polygonal curves from unorganized dense planar points. In reverse engineering, these planar points may be generated by slicing 3D data points and projecting them onto a plane. It is necessary to approximate these points by constructing 2D polygons, while keeping the shape error within a given tolerance. These 2D polygons can be used for fabrication using rapid prototyping processes. With the method outlined in this paper, the curve to fit these data points can be constructed without considering the structure, orientation and topology information of the points. The final 2D polygon obtained possesses the minimum number of points while keeping the shape error within a given tolerance. This is accomplished in several steps: firstly, the planar points are sorted by a tangent-vector based method, which uses a fixed neighbourhood size to estimate the tangent vector of a point. Secondly, the sorted points are decomposed into different levels by using *B*-spline wavelets. Finally, the polygonal curve is constructed from coarser to finer level under the control of shape error between the original planar points and constructed curve.

Keywords: curve construction, unorganized data points, wavelets, reverse engineering.

1. INTRODUCTION

In reverse engineering, one of the effective methods to model data cloud for fabrication using rapid prototyping techniques is to adaptively slice the data points, along a specific direction, into a number of layers and the points in each layer are treated as planar. These planar points then need to be represented by one or several 2D polygons while the shape error must be kept within a given tolerance. At the same time, the number of segments in each polygon should be kept to the minimum. This problem can be stated as follows: Given a planar data point set **D** that lie on an unknown curve **UC**, create a curve **C** to approximate **UC**, such that the constructed curve **C** should have the same topology as **UC**, and can be everywhere close to **UC,** i. e., the shape error, which is estimated by the largest distance between **D** and **C**, meets the requirement of shape tolerance ε.

To reconstruct a curve from an unordered 2D data set, different approximation approaches have been presented, which can be classified into global and local methods. Global methods assume that the **UC** is a continuous curve, and usually a least square method is used to achieve an approximated curve **C** to fit **UC**. Fang and Gossard [6-7] presented a method to reconstruct a smooth parametric curve from the unorganized data points by simulating the deformation of elastic beam under the application of spring forces. This smooth curve is achieved by nonlinear minimization of spring energy, which is solved by successive quadratic programming. However, this method requires previous knowledge of the data topology and good initial curve estimates. Hence it is not suitable for curve reconstruction from an unknown or arbitrary topology. Local fitting methods use piecewise curves to fit the "nearby points" piece by piece, such that a complex shape can be approximated [1, 9-10, 12, 14]. Apparently, the selection of the "nearby points", i.e., neighbourhood points, is an important and difficult issue. Two general methods can be used. One is to use a fixed neighbourhood size [10], which can give a fast computation. However, it causes severe problems in practice. The other method is to use adaptive neighbourhood size [9, 14], in which the correlation coefficient of neighbourhood points is used to decide the neighbourhood size. However, to select an appropriate coefficient value as a threshold to determine the maximum neighbourhood remains a difficult task.

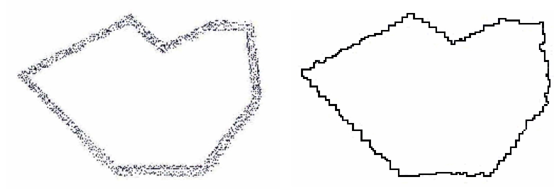
Recently, a new multi-scale technique for shape representation has been developed based on wavelets [2, 4, 8]. In computer graphics, wavelet methods are developed for the multi-resolution representation of parametric curves and surfaces. It is mainly used as a powerful tool for curve and surface hierarchical design. Wang, et al. [13] presented a multi-scale curvature-based shape representation using *B*-spline wavelets. They introduced a coarse-to-fine matching algorithm that automatically detects the dominant points to compress the curve. Esteve, et al. [5] presented a multi-resolution method for implicit curves and surfaces based on wavelets to simplify the topology.

However the above mentioned methods for curve representation started with a source of data in digital ordered form. This paper presents a method to construct a curve from unorganized data points based on wavelets under the control of shape error.

This paper is organized as follows. In section 2, a sorting algorithm is outlined. In section 3, the curve decomposition based on wavelets is addressed. Section 4 describes the algorithm of polygonal curve reconstruction from coarser level to finer level under the control of shape tolerance. Section 5 gives two case studies. Finally, conclusions are given in section 6.

2. DATA SORTING

From a planar point cloud $\mathbf{D} = \{\mathbf{D}_i\}$, we start from a randomly selected point and use a fixed neighbourhood radius to find the first neighbourhood. The point that is closest to the centre of the points is used as the start point. We then construct a straight line segment that locally fits the points within the neighbourhood. Here, we use a least-square method to compute a regression line, which passes the start point and best fits the points within the neighbourhood. To make sure that the polygon fits the original point set well within the given shape error tolerance, the fixed neighbourhood radius is assigned a small value, e.g., the shape error tolerance. This small neighbourhood size helps to keep sufficient information on sharp corners. However, it also results in zigzag shapes of the polygon. An example of data sorting is shown in Fig. 1 in which the original data cloud for the polygon has 2,214 points (see Fig. 1a). Employing a fixed neighbourhood size of 0.08mm, we obtained 489 points after sorting as shown in Fig. 1b.



(a) (b)

Fig. 1. An example of data sorting

3. CURVE DECOMPOSITION

The resultant data points obtained from the sorting process are generally dense. Further processing is required such that the resultant polygon has the minimum number of segments (subject to a given shape error). Wavelets can be applied here to achieve this purpose. Wavelet transform and multi-resolution analysis can be used to represent signal scale by scale. In this section we focus on the aspects of 2D curve decomposition using wavelets. For a more detailed reading on wavelets, readers can refer to text books [3, 11].

Denoting the data points of the polygon as $\mathbf{S}_i$ (i = 0, 1, …, n), they can be reasonably considered as the control points of the unknown *B*-spline curve **f**, which nearly passes these data points. This curve **f** can be represented by a limited number of cubic endpoint-interpolating *B*-spline bases as

$$\mathbf{f}(u) = \mathbf{S}_0\phi_0(u) + \mathbf{S}_1\phi_1(u) + \ldots\mathbf{S}_n\phi_n(u)$$
$$= [\phi_0(u)\ldots\phi_n(u)] \bullet [\mathbf{S}_0\ldots\mathbf{S}_n]^{\mathbf{T}} = \phi(u) \bullet \mathbf{S} \quad (1)$$

where $\phi(u)$ is a row vector of the basis functions called scaling functions and coefficient $\mathbf{S}$ is a column vector of the control points. If these basis functions are considered as scaling functions at scale j, and **f** becomes $\mathbf{f}^j$, that is within the space $\mathbf{V}^j$, then Eqn. (1) becomes

$$\mathbf{f}^j(u) = \mathbf{S}_0^j\phi_0^j(u) + \mathbf{S}_1^j\phi_1^j(u) + \ldots\mathbf{S}_n^j\phi_n^j(u)$$
$$= [\phi_0^j(u)\ldots\phi_n^j(u)] \bullet [\mathbf{S}_0^j\ldots\mathbf{S}_n^j]^{\mathbf{T}} = \phi^j(u) \bullet \mathbf{S}^j \quad (2)$$

$\mathbf{f}^j$ represents the original shape that has the maximum number of data points. On the other hand, $\mathbf{f}^j$ can be approximated by using fewer number of data points (the dominant points). This is called curve decomposition and the resultant curve is $\mathbf{f}^{j\text{-}1}$ that represents the original shape at scale j-1 in space $\mathbf{V}^{j\text{-}1}$. Mathematically, $\mathbf{f}^{j\text{-}1}$ can also be expressed using Eqn. (2) with $\phi^{j-1}(u)$ and $\mathbf{S}^{j\text{-}1}$.

The relationship between $\phi^{\mathbf{j}}(u)$ and $\phi^{\mathbf{j}-1}(u)$ can be expressed as:

$$\phi^{j-1}(u) = \phi^{j}(u)\mathbf{P}^{j} \quad (3)$$

Suppose space $\mathbf{W}^{j\text{-}1}$ is the complement space of $\mathbf{V}^{j\text{-}1}$ in space $\mathbf{V}^{j}$, and $\psi^{j-1}(u)$ is the basis matrix in $\mathbf{W}^{j\text{-}1}$, then we have

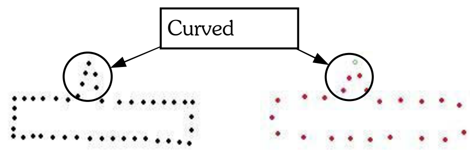
$$\psi^{j-1}(u) = \phi^{j}(u)\mathbf{Q}^{j} \quad (4)$$

According to Eqns. (3)-(4), we have the following,

$$\begin{bmatrix} \mathbf{A}^{j} \\ \mathbf{B}^{j} \end{bmatrix} = [\mathbf{P}^{j} \mid \mathbf{Q}^{j}]^{-1} \quad (5)$$

$\mathbf{P}^{j}$ and $\mathbf{Q}^{j}$ are called synthesis filters, while $\mathbf{A}^{j}$ and $\mathbf{B}^{j}$ are called analysis filters [11]. Thus, $\mathbf{f}^{j}$ can be decomposed to $\mathbf{f}^{j-1} = \mathbf{A}^{j}\mathbf{f}^{j}$ and $\mathbf{d}^{j-1} = \mathbf{B}^{j}\mathbf{f}^{j}$, in which $\mathbf{f}^{j\text{-}1}$ is the approximation of $\mathbf{f}^{j}$, and $\mathbf{d}^{j\text{-}1}$ is the detail that is lost because of this approximation. Similarly, further decomposition of the curve can be carried out to lower scales. The number of the data points in $\mathbf{f}^{j\text{-}1}$ is half of that in $\mathbf{f}^{j}$, and the positions of these data points at level *j*-1 are slightly different from the corresponding data points at level *j* because of the filtering nature of $\mathbf{A}^{j}$. The detail coefficients $\mathbf{d}^{j\text{-}1}$ stores the lost information so that $\mathbf{f}^{j}$ can be recovered from $\mathbf{f}^{j\text{-}1}$. Fig. 2 shows an example of curve decomposition in which the shape in Fig. 2a is approximated by the shape in Fig. 2b.

Therefore, we can use the end-point interpolating *B*-spline as the scaling basis and *B*-spline wavelets to decompose the initial polygon into lower levels at, *j*-1, *j*-2, and so on. However, the remaining challenge is to determine the number of levels of decomposition needed to achieve a shape that is represented by the minimum number of points while maintaining the shape error within the given tolerance. For a curve, its flat area may need more levels to decompose, while the curved area may need fewer. As shown in Fig. 2a, a curve at level *k* is within shape error tolerance and it is decomposed into level *k*-1, as shown in Fig. 2b. Obviously, the curved region at level *k*-1 will be out of shape error tolerance, which will stop the further decomposition to level *k*-2, even though the flat region can be compressed further. Hence, using the shape error tolerance as the only criterion may stop decomposition process too early. Therefore, the ideal situation is that the decomposition is carried out adaptively for flat and curved regions respectively. In practice, however, this is difficult to achieve. In our approach, the decomposition is carried out to the minimum level in which the shape error in the flattest regions is just within the given tolerance. At the same time, the information for each level of decomposition is recorded. Based on the given tolerance, the curved regions can be reconstructed until the level where the shape error meets the tolerance. In this way, other regions can be reconstructed and the decomposition can achieve an optimal result globally.



(a) Level *k* (b) Level *k*-1

Fig. 2. An example of curve decomposition

To achieve the optimal decomposition, we use criteria to check whether the final level of decomposition is achieved. The first one is the shape error. For the *i*-th segment $\mathbf{S}_i^m$ at *m* level, the shape error is calculated as:

$$\mathbf{E}_i(\mathbf{s}) = \max_{\mathbf{p}_k \in \mathbf{P}^i} \mathbf{dist}(\mathbf{p}_k, \mathbf{S}_i^m) \quad (6)$$

where, $\mathbf{P}^i$ is the subset of the originally sorted data points $\mathbf{f}^{j}$, which are near to segment $\mathbf{S}_i^m$ compared to any other segments at level *m*. The shape error measures the distance between the segment and the points of the original level which are nearer to this segment. If any of the errors of the segments are larger than the given tolerance, there is a shape-error violation. To further check whether this happens in a flat region or a curved region, it is necessary to check the curvature at the violation region. Here, we use a simplified method of curvature checking by using a relative shape error. For segment $\mathbf{S}_i^m$, its relative shape error is defined as:

$$\mathbf{E}_i(\mathbf{r}) = \mathbf{E}_i(\mathbf{s}) / \mathit{length}(\mathbf{S}_i^m) \quad (7)$$

Obviously, the relative shape error is larger in the curved region than that of the flat region. A threshold value is needed to check whether there is a relative-error violation for that segment. If both shape-error violation and relative-error violation happen to a segment at the same time, the decomposition stops and the final level of decomposition (or the coarsest level of curve) is one level higher than the current one.

4. CURVE RECONSTRUCTION FROM COARSER TO FINER LEVEL

With the algorithms introduced in section 3, a curve with dense points can be decomposed to coarser levels. Obviously, at the coarse level, some data points may be out of shape tolerance. Hence, it is necessary to find and keep the points that meet shape tolerance from coarser level to finer level, such that the reconstructed curve will be as concise as possible within the given tolerance. For implementation, we start with the coarsest level r by extracting the points that form continuous segments, which meet the tolerance. If the segments do not form a closed shape, we continue extracting points at level $r+1$, until the maximum level j.

4.1. Extract Storable Coefficients at the Coarsest Level

At this level, among the scaling coefficients set $\mathbf{C}$, given a coefficient $\mathbf{c}_i$ and its two adjacent coefficients $\mathbf{c}_{i-1}$ and $\mathbf{c}_{i+1}$, $\mathbf{c}_i$ is termed as a storable point associated with the original planar data points $\mathbf{P}$ if the following criterion is satisfied:

$$\min_{\mathbf{p}\in\mathbf{P}_{ci}} \{|\mathbf{p}-\mathbf{c}_{i-1}\mathbf{c}_i|, |\mathbf{p}-\mathbf{c}_i\mathbf{c}_{i+1}|\} \le \varepsilon \tag{8}$$

where ε is the given tolerance and the point set $\mathbf{P}_{ci}$ represents the neighbourhood points of $\mathbf{c}_i$, formed from $\mathbf{P}$. $\mathbf{P}_{ci}$ can be formed by finding all the data points in $\mathbf{P}$ that are closer to $\mathbf{c}_i$ than to any other coefficients $\mathbf{C}$ at this level, namely,

$$|\mathbf{p}-\mathbf{c}_i| = \min_{\mathbf{c}_k\in\mathbf{C}} |\mathbf{p}-\mathbf{c}_k| \tag{9}$$

If four adjacent coefficients $\mathbf{c}_{i-1}$, $\mathbf{c}_i$, $\mathbf{c}_{i+1}$ and $\mathbf{c}_{i+2}$ are storable, we can link $\mathbf{c}_i$ and $\mathbf{c}_{i+1}$ to form a line segment and $\mathbf{c}_i$ and $\mathbf{c}_{i+1}$ are extracted as the desired points. Thus, the scaling and detail coefficients at higher levels corresponding to these two coefficients can be flagged. We can then delete the data point from $\mathbf{P}$ if this point satisfies:

$$|\mathbf{p}-\mathbf{c_i}\mathbf{c_{i+1}}| \le \varepsilon \quad \text{and} \quad \mathbf{p}\in\mathbf{P_{ci}} \cup \mathbf{P_{c,i+1}} \tag{10}$$

where, $\mathbf{P}_{ci}$ and $\mathbf{P}_{c,i+1}$ are obtained with Eqn. (9). In cases where there are three or less consecutive storable points found in a region, these storable points are ignored.

Similarly, when we identify more than four consecutive storable points, we will extract only the inner scaling coefficients (two end points will not be considered). The storable points extracted (2 or more) are called multi-storable coefficients (MSC). Within a MSC, we will reduce the data points in the original set using Eqn. (10). To store the extracted coefficients, we use a global dynamic list GDL to save the coefficients and its corresponding spatial index. For the example shown in Fig. 1, Fig. 3 shows the MSC extraction results at the coarsest level. It can be seen that there are 3 MSC segments extracted at the coarsest level.

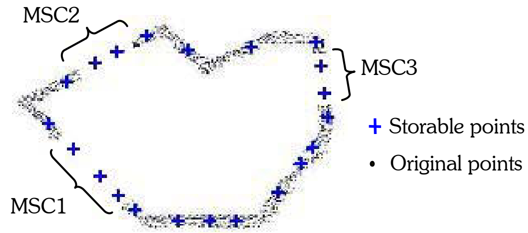


Fig. 3. Extracting multiple scaling coefficients (MSCs) at the coarsest level

4.2 Extract Scaling Coefficients at the Remaining Finer Levels

The method for extracting the storable coefficients at the remaining finer levels is slightly different from that at the coarsest level. At higher levels, we firstly need to reconstruct all the scaling coefficients from the scaling and detail coefficients at coarser level. This process can be implemented with the synthesis filters $\mathbf{P}$ and $\mathbf{Q}$ obtained from Eqns. (3)-(4) and the reconstruction from level j-1 to j can be done using:

$$\mathbf{c}^j = \mathbf{P}^j\mathbf{c}^{j-1} + \mathbf{Q}^{j-1}\mathbf{d}^{j-1} \tag{11}$$

For the smooth section of planar data points that are already approximated by the extracted coefficients at coarser level, the corresponding portion at the finer level need not be recalculated, i.e., the MSCs that are extracted at coarser levels remains unchanged at finer levels. For example, a MSC at level j is $\{\mathbf{c}_i^j, \mathbf{c}_{i+1}^j, \ldots\ldots \mathbf{c}_{i+k-1}^j\}$, after reconstruction to level $j+1$, we get $\{\mathbf{c}^{j+1}\}$ at level $j+1$ to replace the whole scaling coefficients $\{\mathbf{c}^j\}$ and the detail coefficients $\{\mathbf{d}^j\}$ at level j. This MSC has a corresponding list of coefficients $\{\mathbf{c}_{2i}^{j+1}, \mathbf{c}_{2i+1}^{j+1}, \mathbf{c}_{2(i+1)}^{j+1}, \mathbf{c}_{2(i+1)+1}^{j+1} \ldots\ldots \mathbf{c}_{2(i+k-1)}^{j+1}, \mathbf{c}_{2(i+k-1)+1}^{j+1}\}$, which is called a updated MSC. In the final set of MSCs, the updated MSCs will be replaced by their original MSCs. The updated MSCs are mainly used for finding the search space for new MSCs at the current level. Fig. 4 shows the updated MSCs from the MSCs in Fig. 3.

In between the updated MSCs, the coefficients need to go through the MSC extraction process introduced in section 4.1. This will produce a new set of MSCs at this level. Fig. 4 also shows some new MSCs extracted at the current level. They are stored into the GDL with their corresponding indices. Fig. 5 shows the data structure of the coefficients in Fig. 3 and Fig. 4. Fig. 5a shows 3 MSCs at level j, and Fig. 5b shows the reconstructed scaling coefficients at level $j+1$, and there are 3 updated MSCs. Fig. 5c shows 2 new MSCs extracted from the regions in between the update MSCs.

Furthermore, the data points in the original list **P** that satisfy Eq. (10) are deleted. The extraction process continues the next higher, until no data points are left in **P**. All the MSCs retained at all levels then form a complete shape approximating the original data set.

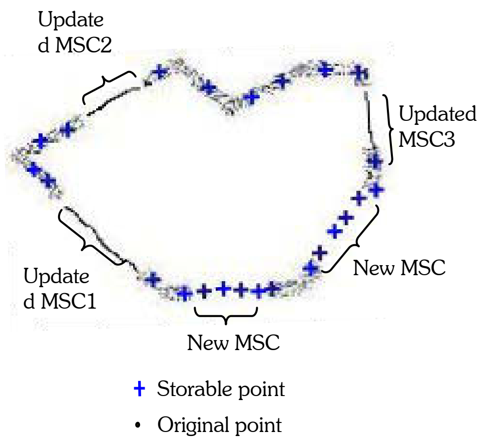


Fig. 4. MSC extracting at finer levels

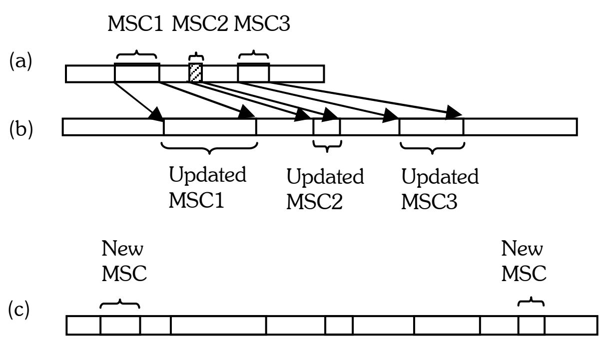


Fig. 5. Data structure of scaling coefficients

5. EXAMPLES

The algorithm for 2D curve reconstruction using *B*-spline wavelet has been implemented using C++ and OpenGL. The input to run the algorithm is a set of 2D points without any structure, plus the shape error tolerance and the relative error tolerance. Two examples are given here to show the efficacy of the algorithm.

In the first example, the 2D data set is shown in Fig. 1. The shape tolerance was set at 0.08 mm and the relative shape tolerance as 0.05. There are total 5 levels of decomposition. The algorithm ran on a PC with a 1.8GHZ CPU, and the memory is 256MB. It took less than 1 minute to complete the decomposition and reconstruction. The final constructed polygon shown in Fig. 6 has 64 points. It can be seen the corner points are very well retained.

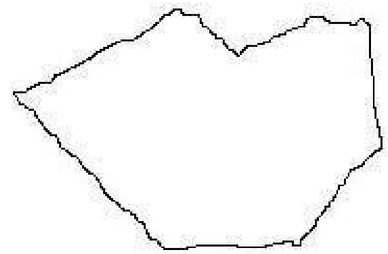


Fig. 6. The final 2D shape of example -1

In the second example, the 2D points are obtained by taking a slice of a set of scanned data points from an object with sculptured surface (see Fig. 7a). The points in the sliced layer are then projected onto a plane as shown in Fig. 7b. There are total 1,320 points. The shape error tolerance was set as 0.05 mm. Employing a fixed neighbourhood size of 0.05mm, we obtained 387 sorted points as shown in Fig. 7c. The relative error tolerance was then set at 0.05, and we obtained a final polygon curve with only 67 points as shown in Fig. 7d. This 2D curve reconstruction algorithm is part of an overall algorithm for construction layer-based model (with shape error control and thus adaptive layer thickness), from cloud data, for fabrication using rapid prototyping methods. Fig. 7e shows the final layer-based model for the cloud data in Fig. 7a, which has 49 layers.

6. CONCLUSIONS

A practical method for curve construction from unstructured planar data points with accuracy control has been described. The method commences with a quick sorting algorithm based on neighbourhood marching method. The small fixed neighbourhood size results in a dense and ordered data set. This data set is decomposed to different levels under the control of shape tolerance and relative error tolerance using *B*-spline wavelet. Finally, a concise curve is constructed by extracting the data points from coarser level to finer level under the control of a shape tolerance. Case studies show that our algorithm is efficient.

The remaining challenging issue is the determination of the relative error tolerance to control the maximum level of decomposition. Currently, it is a user-specified parameter. It is recommended that a small value should be used in order to ensure the maximum level of decomposition is reached. This may, however, result in that some coarse levels that are useless in the reconstruction process, i.e., no MSCs can be found in these levels. Therefore, a trade-off is inevitable between the optimality and the efficiency.

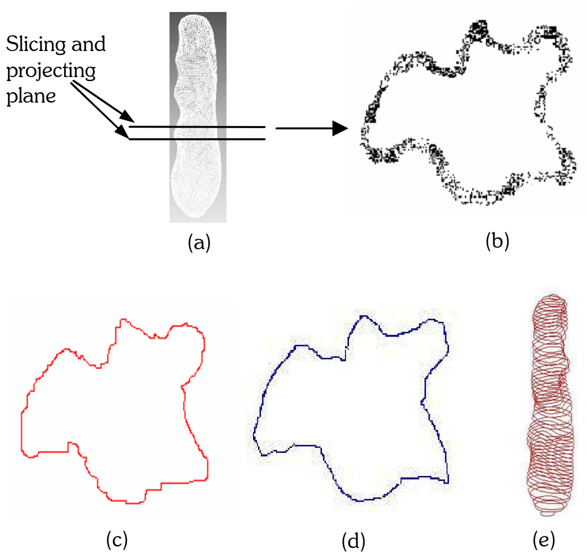


Fig. 7. The raw data and final shape of example-2

7. REFERENCES

[1] Cheng, S. W., and Funke, S., Glolin, M., Kumar, P., Poon, S. H. and Ramos, E. A., Curve reconstruction from noisy samples, *Proceedings of the 19th Symp. Computational Geometry ACM*, Jun 2003, pp 302-311.

[2] Chuang, G. and Kuo, C., Wavelet descriptor of planar curves: Theory and applications, *IEEE Transactions on Image Processing*, Vol. 1, No. 5, 1996, pp 56-70.

[3] Chui, C., *An introduction to wavelets*, Academic Press, San Diego, 1996.

[4] Chung, K., The generalized uniqueness wavelet descriptor for planar closed curves, *IEEE Transactions on Image Processing*, Vol. 9, No. 5, 2000, pp 834-845.

[5] Esteve, J., Brunet, P. and Vinacua, A., Multiresolution for Algebraic Curves and Surfaces Using Wavelets, *Computer Graphics Forum*, Vol. 20, No. 1, 2001, pp 47-59.

[6] Fang, L. and Gossard, D. C., Fitting 3D curves to unorganized data points using deformable curves, *Visual Computing* (*Proceedings of CG International '92*), June 1992, Springer, Berlin, pp 535-543.

[7] Fang, L. and Gossard, D.C., Multidemensional curve fitting to unorganized data points by nonlinear minimization, *Computer-Aided Design*, Vol. 27, No. 1, 1995, pp 48-58.

[8] Finkelstein, A. and Salsin, D., Multiresolution curves. Computer Graphics Annual Conference Series, July 1994, pp 261-268.

[9] Lee, In-Kwon, Curve reconstruction from unorganized points, *Computer-Aided Geometric Design*, Vol. 17, No. 2, 2000, pp 161-177.

[10] Liu, G. H., Wong, Y. S., Zhang, Y. F., and Loh, H. T., Error-based segmentation of cloud data for direct rapid prototyping, *Computer-Aided Design*, Vol. 35, 2002, pp 633-645.

[11] Stollnitz, E., DeRose, T. and Salesin, D., *Wavelets for computer graphics*, Morgan Kaufmann Publishers Inc., San Franscisco, 1996.

[12] Taubin, G. and Ronfard, R., Implicit simplicial models for adaptive curve reconstruction, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 18, No. 3, 1996, pp 321-325.

[13] Wang, Y.P., Lee, S.L. and Kazuo Toraichi, Multiscale Curvature-Based Shape Representation Using B-Spline Wavelets, *IEEE Transactions on Image Processing*, 1999, Vol. 8, No. 11, 1999, pp 1586-1592.

[14] Wu, Y.F., Wong, Y.S, Loh, H.T. and Zhang, Y.F., Modelling Cloud Data Using an Adaptive Slicing Approach, *Computer-Aided Design*, Vol. 36, 2004, pp 231-240.