

# Medical Data Point Clouds Reconstruction Algorithm Based on Tensor Product B-Spline Approximation in Virtual Surgery

Guo-Dong Chen<sup>1</sup> · Fei-Fei Wang<sup>1</sup>

Received: 12 January 2016 / Accepted: 25 March 2016 / Published online: 7 January 2017  
© Taiwanese Society of Biomedical Engineering 2017

**Abstract** A complete 3D visualization method for virtual liver lesion model was proposed by taking patients' abdomen CT slices as data source. Firstly, Gradient Vector Flow-Snake algorithm that combines with region force was adopted to fast and accurately extract the contour points in patients' CT slices. Then 3D cloud data of these contour points was simplified uniformly, and a distance-field-based method of distribution field fitting for B-spline surface was presented to fast establish the lesion model roughly. Then, an interactive node fine tuning method Interactive Marching Nodes was proposed, so the model can be optimized according to requirements of users. Finally, realistic lesion model was generated through texture mapping. Experiments suggest that this approach is suitable for various kinds of lesions with the whole process taking only a few minutes, and the generated model has high precision, which is of some significance in the study of virtual surgeries.

**Keywords** CT image processing · Virtual surgery · GVF-snake model · Tensor product B-spline approximation

## 1 Introduction

In modern clinical medicine, the diagnosis of liver diseases remains in the stage of observing 2D CT images layer by layer. This method becomes increasingly important in

disease diagnosis by virtue of its convenient acquisition of source data and high resolution ratio. The manual image processing method, with low efficiency, requires a heavy workload, and the accuracy would be affected in the process. In 1987, Kass et al. proposed the active contour model Snake [1], taking a key step for medical image processing. However, the trapping force of traditional snake model is so small that the model cannot fulfill the convergence at the edge of depressed lesions. In 1998, Xu et al. [2] then proposed the GVF-Snake model, whose traditional form enhances the trapping force. But this model needs multiple iterations to realize convergence, which is time consuming, thus it cannot meet virtual surgeries' requirement for timeliness.

Currently, due to the limitations of 2D medical images, scientists often use 3D visualization by computer graphics and virtual reality technology and do interactive processing depending on the requirement, which helps the diseased tissue analysis, clinical treatment formulation in our liver virtual surgery, and provides doctors with some intuitive and reliable references. Traditional surface reconstruction technology is limited by the volume resolution, and it should maintain the topological consistency. In recent years, with the development of graphics hardware and software platform, volume visualization technology is moving on to the stage of points. However, because the coordinate information of surface point sets has noise and the resolution between layers and images is different, so the density of surface data is heterogeneous. In conclusion, to find a precise and efficient method suitable for medical images has currently become a research hotspot in the field of 3D reconstruction. There are many different surface reconstruction methods based on point cloud data, which can be divided into the explicit one and the implicit one in general. The explicit method gives precise surface location

---

✉ Fei-Fei Wang  
ifafei@126.com

Guo-Dong Chen  
fzucgd@126.com

<sup>1</sup> Department of Physics and Information Engineering, Fuzhou University, Fuzhou 350108, China

directly, which pays more attention to solving triangle patch by Delaunay Triangle and Voronoi-Diagram. Seo et al. [3] made outstanding contribution to that and proposed the  $\beta$ -shape algorithm. But the detail information is sometimes lost locally. Cazals et al. [4] presented the Power Crust algorithm to reconstruct model through signing a Delaunay tetrahedron. However, it is not suitable for the noise. The implicit method, formed by combinations of basis functions, is applicable to the complex model but time consuming. Vaněček [5] compressed the mass point cloud data by the combination of RBF and greedy algorithm. Wang et al. [6] made quadratic approximation on local surface by PHT-spline method and realized the adaptive reconstruction for the mass point cloud. However, the precision needs to be improved.

Our main contribution is proposing a new method for lesion volume visualization based on CT images. Figure 1 shows the whole process in brief. First of all, this paper puts forward an improved GVF-Snake algorithm based on region force to extract lesion contours in batches. Then, a tensor product B-spline surface reconstruction method [7] with IMN optimization is proposed for obtaining a high-quality lesion model. In general, our method has two advantages. The first one is that extraction process is well suitable for the complex lesion area and the large number of CT images. Secondly, the process is simple and interactive with high precision, which meets the requirements of virtual surgery.

## 2 Methods

### 2.1 Batch Extraction of Contour

#### 2.1.1 Setting the Initial Contour

The first step of GVF-Snake model extraction is to determine the initial contour. In clinical medical diagnosis, most

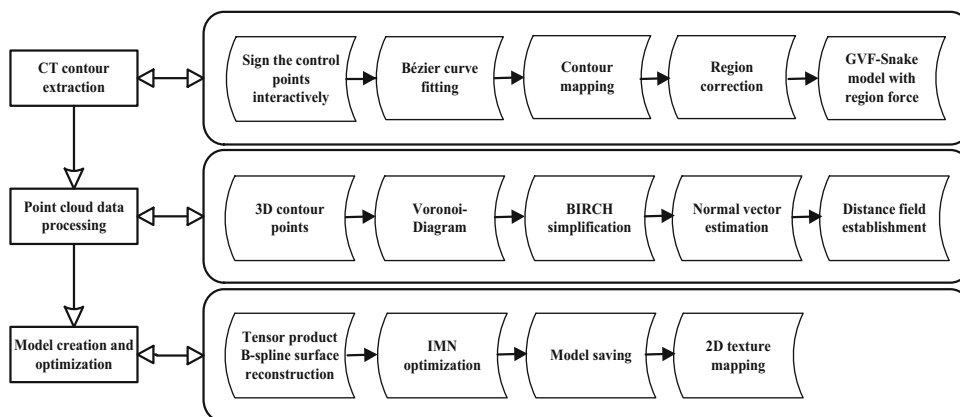
of the CT lesion is massive shadow for small area, and the gray level of edge is very similar within the neighbors. All these make it difficult to determine it automatically. In consequence, we choose a human interaction technology in this paper, like a part of experiment we researched before [8], and do some improvements. As shown in the top of Fig. 1, firstly we probably need a doctor to click on about ten lesion contour points on each of the CT images through the mouse, then conduct a Bézier curve fitting for these points. After this, the system will map each contour curve to the next CT image according to the position coordinates of points. So we will get lesion contours accurately only by correcting the contour according to the difference of regions, as described in the next paragraph. Moreover, the number of lesion CT images is probably twenty.

Here, we firstly need to contrast two contour areas before and after mapping, and define the same parts as the “public” area, the different one as the “particular”. And then calculate grayscale average  $\alpha$ ,  $\alpha_0$  of the public areas and particular areas, respectively, using the digital picture processing technique. We have known that the gray level of segmentation area  $\Omega_1$  and background area  $\Omega_2$  is  $\varpi_1$  and  $\varpi_2$  respectively. If a mild lesion area meet the conditions of  $|\alpha - \alpha_0| < \lambda$ , where we set  $\lambda = \text{Argmax}(\Omega_1\Omega_2 / (\Omega_1 + \Omega_2)^2)$  through the experiments, and its gray level is  $\varpi$ , so  $\varpi_1 < \varpi < \varpi_2$ . This moment we should adopt object filling method to convert  $\varpi_1$  to  $\varpi_2$ , which can help us prevent from missing segmentation. Likewise, we can adopt the method of filling the background area that converts  $\varpi_2$  to the average gray of the area for the condition of segmenting some background areas.

#### 2.1.2 GVF-Snake Model with Region Force

It’s very common that the shadow block of non-target lesions has a pseudo boundary effect on patients’

**Fig. 1** The whole framework of this paper



abdominal CT images, but as we all know, traditional GVF model cannot reduce this effect [9]. Xu [10] et al. proposed an automatic medical image segmentation technique with the concept of region force in 2012, which could improve accuracy and sensitivity of active contour model. In this paper, in order to make our model be more adaptive to the complex region like lesion, we simplify Xu’s method and add it to our algorithm to improve the traditional GVF-Snake model. Set R as a lesion area on one CT image  $I(x, y)$ , so we can define its gray information as:

$$S_R(x, y) = \begin{cases} I(x, y), & (x, y) \in R \\ 0 & (x, y) \notin R \end{cases} \quad (1)$$

If the contour and energy of R area is denoted by  $\Gamma(s)$  and  $E_R$  respectively, and C is the weighted value, so the energy function of model can be written as:

$$E = \int_0^1 E(\Gamma(s))ds + CE_R \quad (2)$$

Set  $H(S_R(x, y))$  as the conversion factor between image grayscale and active contour fitting, so according to the positional relations between R area and the initial contour, we can use Green’s identities derived by the divergence theorem to implement a conversion between the region field and the boundary field:

$$\begin{aligned} E_R &= \iint_R H(S_R(x, y))dxdy \\ &= \frac{1}{2} \oint P_R(x, y)dx + Q_R(x, y)dy \\ &= \frac{1}{2} \oint \left[ - \int_0^y H(S_R(x, z))dz \right] dx \\ &\quad + \left[ \int_0^x H(S_R(z, y))dz \right] dy \end{aligned} \quad (3)$$

Here we define the region force as follow, which may contribute to fitting to the deep concave area of lesion:

$$F_R = C(P_R(x, y) + Q_R(x, y)) \quad (4)$$

The energy becomes minimum when the contour is in a state of balance, which means that  $F_{int} + F_{GVF} + F_R = 0$ . Here  $F_{int}$  is internal force of the contour and  $F_{GVF} = F_{ext} = [u(x, y), v(x, y)]$  is external force generated by the gradient vector field.

## 2.2 Preprocessing of Point Cloud Data

The scanning of CT images for each layer is at the same slice thickness. As a result, we can get thickness information of each slice according to the actual performance parameters of medical equipment. Then we will acquire 3D information of point set from 2D. Because the sampling resolution of each CT image is larger than that of inter-layer, we need to homogenize point cloud data to meet the requirement of building distance vector field subsequently. Here we firstly adopt the points interpolation method provided in PCL (Point Cloud Library) [11], its core idea is: calculating MLS (Moving Least Squares) surface and the Voronoi diagram according to point cloud, so the new sampling point is the farthest vertex.

The spatial distribution of enormous data after interpolation is fairly dense, if we apply it directly to the following work will cause a lot of time and memory. So we should simplify the data on the premise of keeping the geometry features of model to improve the operation efficiency. In this paper, we divide and merge the spatial point cloud data according to the category based on BIRCH algorithm framework [12], so as to realize the adaptive simplification of point cloud. The basic idea of algorithm is regarding the point cloud space as a class, and making a splitting processing continuously through the calculation of covariance. Finally we choose the representative points which satisfying the MCD (Minimum Covariance Matrix) as results.

## 2.3 Establish the Distance Vector Field

### 2.3.1 Estimation and Orientation of the Normal Vector

As there is no connection information among spatial points, the normal vector of points can be only estimated by using k-Nearest Neighbors algorithm, namely, the normal vector is represented by the eigenvector corresponding to minimum Eigen value of its covariance matrix.

It is then required to determine the direction of normal vector and set up consistent interior and exterior orientations. An improved redirection algorithm based on local surface fitting is applied in our proposed method. As shown in Fig. 2, first, the point with the largest component of

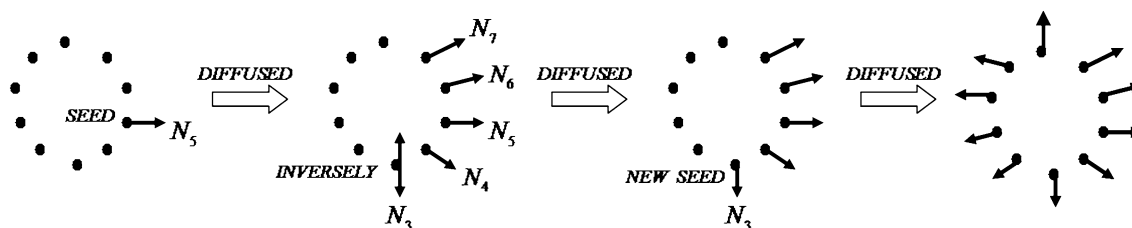


Fig. 2 An diagram of the normal vector establishment

$x$  coordinate is selected from sampling points as the initial point. Its normal vector is forced to point toward the positive direction of  $x$  axis. The direction of the normal vector is then diffused. Tangent constraint criterion is adopted to define its direction of diffusion. Based on normal vector similarity of adjacent points, the normal vector that is most similar to the current normal vector is searched to correct directions. If inner product of the two normal vectors is negative, the most similar normal vector shall be defined inversely. Otherwise, no additional definition is required.

### 2.3.2 Calculate the Distance Value for Grid

Given that space grid density is directly related to graphic accuracy and calculation, users shall first specify grid density interactively and divide spatial domains roughly into several small voxels according to the complexity of lesion shape before the space is subdivided. Distance value is then allocated to grid points of all subdivided voxels. The value is defined as the shortest distance  $d_{\min}$  between grid points and the model. In order to improve the efficiency of algorithm without compromising the basic structure of model, the approximate algorithm of distance field is used for calculation, i.e. as shown in Fig. 3, the nearest points  $P_i(x_i, y_i, z_i)$  to the grid point  $G(x_0, y_0, z_0)$  is chosen from point set and their Euclidean distance is used to approximately replace  $d_{\min}$ :

$$d_{\min} \approx \rho(G, P_i) = \sqrt{(x_i - x_0)^2 + (y_i - y_0)^2 + (z_i - z_0)^2} \tag{5}$$

## 2.4 Tensor Product B-Spline Surface Reconstruction

### 2.4.1 Basic Theory

A tensor product B-spline surface  $f(u, v)$  is expressed by the form of B-spline basis functions [13] and defined by:

$$f(u, v) = \sum_l \sum_r C_{lr} L(u) R(v) \tag{6}$$

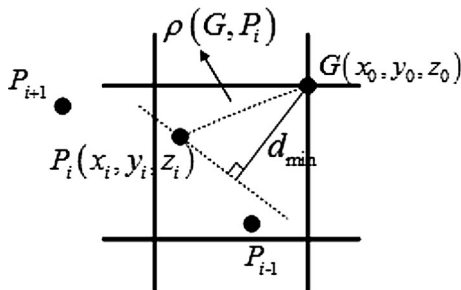


Fig. 3 Calculation of the distance value

where the  $C_{lr}$  are a series of fitting factors, and  $l = 1, 2, \dots, q, r = 1, 2, \dots, h$ , the  $L(u)$  and  $R(v)$  are B-spline basis functions, which are corresponding with B-spline functions of model's isometric node orders  $\{\varphi_l\}$  and  $\{\zeta_r\}$ , respectively. Given  $\Theta$  as the closed region of distance field, the fitting surface, denoted by zero-set of  $f(u, v)$ , is given by  $\{f(u, v) = 0 | (u, v) \in \Theta\}$ .

### 2.4.2 Step-by-Step Dis-Field-Based Algorithm for Surface Fitting

$$P_{i,j} = \{(x_{i,j}, y_{i,j}, z_{i,j}) | i \in [1, N_i], j \in [1, N_j]\}$$

Let be point sets from space division of distance field, and then we use the B-spline surface mentioned above to fit these points, which is expressed as:

$$P_{i,j} = \sum_{l=1}^q \sum_{r=1}^h C_{lr} L(u_i) R(v_j) \tag{7}$$

In order to obtain this fitting surface, we should find the relation between each point and the undetermined coefficients. In the space of distance field, let  $d(u_i, v_j)$  be a distance value distributed to grid point  $G(u_i, v_j)$ ,  $\omega$  be the smoothing factor of model and  $E_s$  be the surface energy, so the error function between tensor product B-spline function and point cloud is denoted as the following expression:

$$ERF = \sum_{i=1}^{N_i} \sum_{j=1}^{N_j} (P_{i,j} - d(u_i, v_j))^2 + \omega E_s \tag{8}$$

Our target is to balance the fitting surface by minimizing this error function. Firstly we rewrite the tensor product B-spline surface function as:

$$f(u, v) = \sum_{l=1}^q L(u) \left( \sum_{r=1}^h C_{lr} R(v) \right) = \sum_{l=1}^q C_l L(u) \tag{9}$$

$$\tilde{h}_i(v) = \sum_{r=1}^h \mu_r R(v)$$

Known from the above formula, we can ignore the factor  $\omega$  and calculate the tensor product B-spline surface by two steps. Firstly, a set of grid points which have the same  $u$  component are fitted by the B-spline function and  $\mu_i \in [1, h]$ . After that we obtain:

$$\Phi(\mu_i) = \sum_{j=1}^{N_j} [\tilde{h}_i(v_j) - d(u_i, v_j)]^2 = \sum_{j=1}^{N_j} \left[ \sum_{r=1}^h \mu_r R(v_j) - d(u_i, v_j) \right]^2 \tag{10}$$

$$\lambda_j(u) = \sum_{l=1}^q \tau_j L(u), \quad \tau_j \in [1, q]$$

Then fit the factor  $\mu_i$  by function and get:

$$\begin{aligned}\Psi(\tau_j) &= \sum_{i=1}^{N_i} [\lambda j(u_i) - \mu_{ij}]^2 \\ &= \sum_{i=1}^{N_i} \left[ \sum_{l=1}^q \tau_j L(u_i) - \mu_{ij} \right]^2\end{aligned}\quad (11)$$

$$\frac{\partial \Phi(\mu_i)}{\partial \mu_i} = 0 \quad \frac{\Psi(\tau_j)}{\partial \tau_j} = 0$$

As a result, we must decide parameters of each fitting point by making a solution of two linear equations and of order  $h$  and  $q$  respectively. That way we will reduce the order and the amount of calculation must be saved greatly.

## 2.5 Model Optimization and Rendering

In the actual process of modeling, the smoothing coefficient  $\omega$  of reconstructive surface is an important parameter [14] and it can reflect smoothness of the model. However, the smoothness and accuracy are often contradictory. If  $\omega$  is too large, some details of the model will be ignored. But if  $\omega$  is too small, the model will seem to be very rough and have obvious seams effect. Therefore, in virtual surgery, we need to optimize the model interactively according to actual conditions, and put in some realistic texture effect to strengthen the visual perception, which can meet the needs of medical diagnosis.

### 2.5.1 Basic Idea

$\mathbf{n}_p = (n_x, n_y, n_z)$   $\mathbf{P}^* = \mathbf{P} + d\mathbf{n}_p$ . Assuming that  $\{\mathbf{P}_i | i = 1, 2, \dots, N\}$  is point cloud on the tensor product B-spline surface. Let  $\{\mathbf{P}_i^* | i = 1, 2, \dots, N\}$  be the point cloud after IMN optimization. So, where  $d$  is the points' relative distance before and after optimization, is a marching direction of nodes.

$$\begin{aligned}\mathbf{W} &= \left( \sum_{i=1}^N \omega_i x_i, \sum_{i=1}^N \omega_i y_i, \sum_{i=1}^N \omega_i z_i \right) d(\mathbf{n}_p) \\ &= \mathbf{q} \cdot \mathbf{n}_p / \|\mathbf{n}_p\|^2 \mathbf{q} = (q_1, q_2, q_3) = \mathbf{W} / \sum_{i=1}^N \omega_i - \mathbf{P} \mathbf{n}_p\end{aligned}$$

If we assume that  $d$  is a function about, and. So we can get, where is the weight vector.

### 2.5.2 Realization of IMN Algorithm

$d(\mathbf{n}_p)$ . In order to simplify calculation, we need to convert the optimizing progress to a solving procedure for the minimum of. For this, we can firstly solve the follow equation by Lagrange multiplier method:

$$\begin{aligned}\frac{\partial d(\mathbf{n}_p)}{\partial \mathbf{n}_p} &= 0 \\ \|\mathbf{n}_p\|^2 - 1 &= 0\end{aligned}\quad (12)$$

By setting  $\lambda$  as a Lagrange multiplier, we obtain the following equation under the constraint condition that :

$$L = d(\mathbf{n}_p) + \lambda \left[ \|\mathbf{n}_p\|^2 - 1 \right] = \mathbf{q} \cdot \mathbf{n}_p + \lambda \left[ \|\mathbf{n}_p\|^2 - 1 \right] \quad (13)$$

By solving partial derivative of  $L$  for  $n_x, n_y, n_z$  and  $\lambda$  components, we obtain

$$\mathbf{n}_p = \left( -\frac{q_1}{2\lambda}, -\frac{q_2}{2\lambda}, -\frac{q_3}{2\lambda} \right) \quad (14)$$

According to the actual conditions, we choose that:

$$\omega_i = \frac{1}{\|\mathbf{P}_i - \mathbf{P}\|^{4+1}} \in (0, 1] \quad (15)$$

$$\mathbf{Q}_p = \left( \sum_{i=1}^n Z_i \mathbf{Q}_i \right) / \left( \sum_{i=1}^n Z_i \right)$$

If, here,  $Z_i$  is the area of the  $i$ th patch,  $\mathbf{Q}_i$  is its normal vector. Thus the optimal marching direction for nodes is:

$$\mathbf{n}_{p(\text{opt})} = \frac{\mathbf{Q}_p}{\|\mathbf{Q}_p\|} \quad (16)$$

### 2.5.3 Texture Mapping

Texture is mostly in the form of two-dimension in computer graphics, which reflects the detail information about the structure of surface. In this part, we mainly use the idea of "lapped textures" [15], which is proposed by Praun et al. in 2000, so we only provide a brief introduction as follows. By observing some instances, we find that the texture of liver lesion model is an unstructured homogeneous texture. Therefore, we assume that the shape of texture can be pre-computed, and then we distribute these created texture blocks to Alpha mask for eliminating the seams effect [16].

Next, we need to assign tangential field on model surface to determine the size and direction of textures. Because of homogeneousness of the lesion texture, we only need to specify global scale field of space grid interactively. Users can freely control the scope of space to achieve the purpose of efficient mapping. In the progress of texture mapping, we firstly need to determine the transformation relation between each vertex of model and the texture space, which means a section mapping relation from 3D to 2D. Then the texture will spread over the whole model through adding patches. We should use the matching between the surface vector field and texture coordinates to make the model covered with texture in the end.

### 3 Results

In order to prove the feasibility presented above, we have applied this algorithm to some examples of liver complaint. The algorithm is implemented in VC++6.0 and OpenGL4.1.0. The CPU is Core i7-4700 HQ 2.4 GHz processor with 4G RAM. The graphics card is NVIDIA GeForce GT 745 M.

#### 3.1 Contour Extraction Module

4Firstly, we need to number slices according to the scanning order of original CT images and store them in the same folder. Users need to read the first slice when program is running. And then they should click on the contour of lesion interactively, about a few key points, to be the control points of Bézier curve. Results show that choosing 8–10 points in the first slice can achieve to a better fitting effect. After mapping the first fitting curve to the second slice, do a region filling and correction. If the difference of fitting area exceeds threshold, system will prompt users to choose some control points again. In this way we can determine each preliminary initial contour. At last, using the improved GVF-Snake algorithm and the accurate fitting results are shown in Fig. 4. The left one is a liver tumor, and the right one is a liver cystic lesion.

Here, we use CvPoint, which is basic data types of OpenCV, to represent point’s coordinate as an 2D integer. Its constructed function is: inline CvPoint cvPoint(int x, int y). “Vector” is a dynamic array in std(standard library) of C ++, which can store and operate a variety of data types, and we can define a “vector” template class to store these extracted contour points as std::vector < CvPoint > CounterPoints. Table 1 shows some extreme values of these points’ 3D coordinates.

#### 3.2 Surface Reconstruction Module

PCL (Point Cloud Library) is a specialized library for processing point cloud data, and the web site of the PCL library is <http://pointclouds.org/about/indicated>. As shown in Fig. 5, we have created a PointCloud class to convert point cloud format to.PCD as: pcl::PointCloud < pcl::PointXYZ > cloud. And then we take the algorithm proposed in the second and third section for simulation through OpenGL technology. In terms of texture mapping, thanks to our medical image library for providing some precious material, we synthesis 2D texture based on some realistic texture samples of the liver lesion surface. Finally we obtain the high-quality lesion model as shown in Fig. 6. Drag the left mouse, and we will observe the spatial structure in every direction.

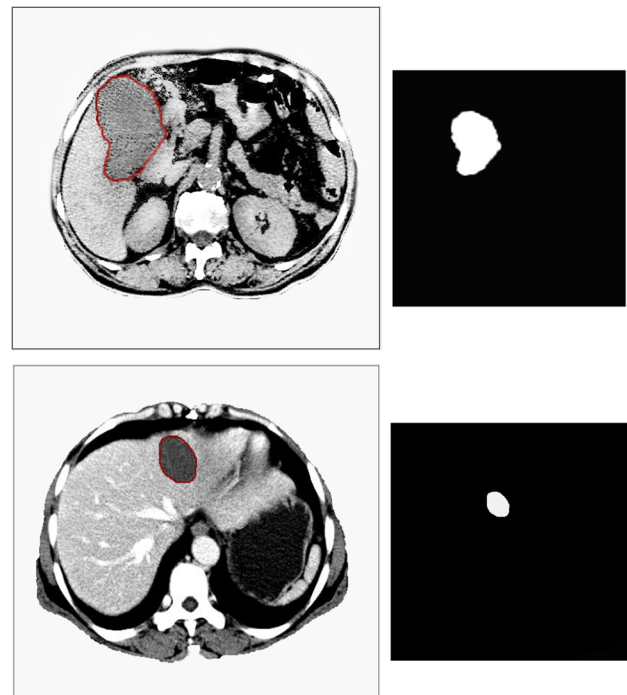


Fig. 4 Lesion contour in one liver tumor CT image

Table 1 The extreme value information of point cloud coordinates

	X	y	z
Liver tumor Fig. 5a			
Min	-27.234591	-46.291348	-61.853020
Max	50.708458	38.685719	46.187515
Liver cystic lesion Fig. 5b			
Min	61.113152	-32.761583	-23.675243
Max	96.251297	4.837264	18.436517

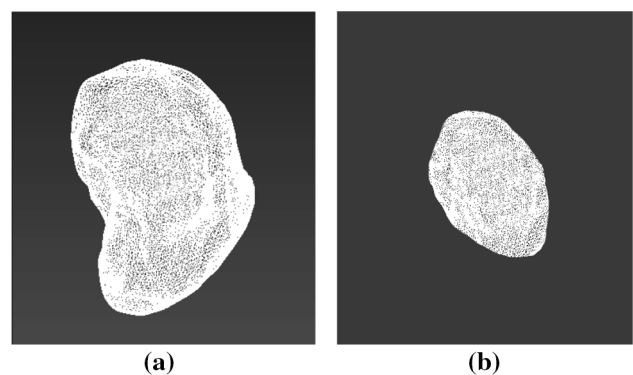
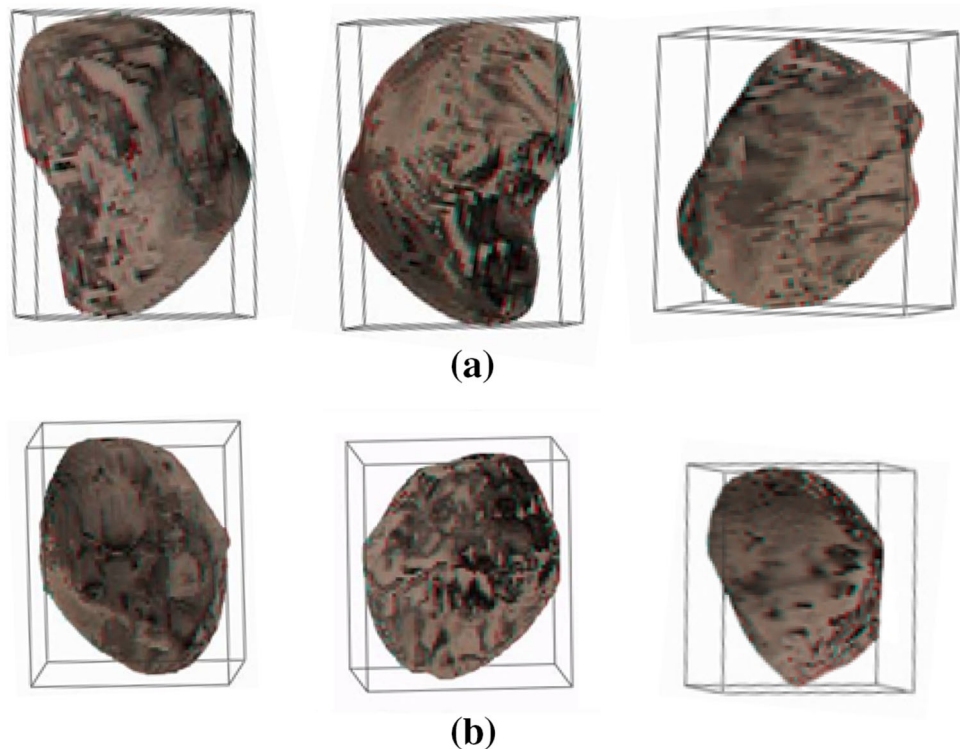


Fig. 5 The surface point cloud distribution map of liver tumor and liver cystic lesion respectively

**Fig. 6** Liver lesion models diagram from different angles. **a** Front view, back view and top view of the liver tumor. **b** Front view, back view and top view of the liver cystic lesion



**Table 2** Model information before and after IMN optimization compared with MC algorithm

	# Of nodes	# Of triangles	Average error (%)	Total time (s)
<i>Liver tumor</i>				
MC algorithm	62,948	31,474	5.32	27.62
Our proposed algorithm				
Before	23,433	11,716	1.58	4.12
After	51,664	25,832	0.20	4.16
<i>Liver cystic lesion</i>				
MC algorithm	20,583	10,292	3.29	9.97
Our proposed algorithm				
Before	6289	3144	1.23	1.78
After	14,754	7376	0.15	1.80

## 4 Discussion

$\{S_{\min_i}\}$  Firstly, we need to verify the accuracy of algorithm by calculating average error between reconstructive model and the original point cloud. Traverse all the points in point cloud space. When one point is not on model surface, we call it “offset point”. Map this point to the surface and we will obtain one “mapping point”. Then, search its nearest point on the surface, and we need to use orthogonally approximation technology to gradually make the “nearest point” close to the “mapping point”. Finally, we get a set

of minimum distances called, which is between the model and point cloud. So the average error can be written as:

$$\sigma = \frac{1}{M} \sum_{i=1}^M S_{\min_i} \times 100\% \quad (17)$$

where M is the sum of points. We have compared the basic situation before and after the optimization, as shown in Table 2, and made a comparison with standard MC (marching cubes). We see from the table that the MC algorithm takes a tremendous amount of time and is very susceptible to the complexity of lesion models. Our

**Table 3** Efficiency analysis

	# Of points	Modeling time (s)	Render time (s)
Liver tumor			
Ref. [17] algorithm	58,387	46.18	20.58
Ref. [18] algorithm	53,283	33.49	12.64
Ref. [19] algorithm	49,801	17.91	–
Our algorithm	51,664	4.16	8.32
Liver cystic lesion			
Ref. [17] algorithm	14,018	30.12	15.35
Ref. [18] algorithm	12,146	10.28	7.83
Ref. [19] algorithm	9875	8.77	–
Our algorithm	11,328	1.71	5.01

algorithm takes only a few seconds to get the high-accuracy model, where the optimization process has a satisfactory result within less than 0.05 s. Model's average error basically has a percentage-point decrease, which improves model's precision.

In terms of efficiency, we have compared our algorithm with previous work of CT visualization based on points [17–19] on the basis of model's average error less than 0.20%, as shown in Table 3, where refs [17]. shows a rapid algorithm of point clouds reconstruction, refs [18]. presents a tomographic surface reconstruction algorithm from point clouds, and refs [19]. proposes a method to efficiently and accurately reconstruct continuous surfaces from point clouds. It's not hard to see that our proposed method will greatly reduce the calculating time in surface reconstruction of medical images and will provide a solid foundation for real-time modeling of lesions.

## 5 Conclusion

In this paper, a complete visualization method for lesion organization was proposed aimed at improving the sense of reality and efficiency of virtual surgery. Our main process include CT image preprocessing, point cloud processing, surface model reconstruction and texturing. Experiment shows that our method can perform efficiently and interactively in our reconstruction work. Moreover, it can maintain good accuracy for some complex structures. Our algorithm also has some limitations. For example, there are still a few visual deviations between built model and the actual one, which is due to the low resolution of texture samples and we ignore the light factor. In addition, we can see in Table 3 that it still takes too much time to render, because in our method we should repeatedly render for completely covering. So how to improve the efficiency of model rendering will be the problem we need to study in the next stage.

**Acknowledgements** We would like to thank our team in the institution of digital image processing in Fuzhou University for help. Our research

was supported by the Natural Science Foundation of China (61471124), the Provincial Science Foundation of Fujian (2013J05090) and the Provincial Science and Technology Major Project of Fujian (2011H0027).

## References

- Bénard, P., Lu, J., Cole, F., Finkelstei, A., & Thollot, J. (2012). Active strokes: Coherent line stylization for animated 3D models. In *Proceedings of the symposium on non-photorealistic animation and rendering* (pp. 37–46).
- Zhang, X., Cai, K., Zhang, F., & Li, R. (2013). Contour segmentation based on GVF snake model and contourlet transform. In: *Proceedings of the 9th international conference on intelligent computing theories and technology* (pp. 503–508).
- Seo, J., Cho, Y., Kim, D., Kim, & D.-S. (2007). An efficient algorithm for three-dimensional  $\beta$ -complex and  $\beta$ -shape via a quasi-triangulation. In *Proceedings of the 2007 ACM symposium on solid and physical modeling*. ACM (pp. 323–328).
- Cazals, F., Kanhere, H., & Lorient, S. (2011). Computing the volume of a union of balls: A certified algorithm. *ACM Transactions on Mathematical Software*, 38(1), 734–747.
- Vaněček, P. (2010). Delaunay space division for RBF image reconstruction. In *Proceedings of the 26th Spring conference on computer graphics*. ACM (pp. 109–116).
- Wang, J., Yang, Z., Jin, L., Deng, J., & Chen, F. (2010). Adaptive surface reconstruction based on implicit PHT-splines. In *Proceedings of the 14th ACM symposium on solid and physical modeling*. ACM (pp. 101–110).
- Saunders, B. V., Antonishek, B., Wang, Q., & Miller, B. R. (2015). Dynamic 3D visualizations of complex function surfaces using X3DOM and WebGL. In *Proceedings of the 20th international conference on 3D web technology*. ACM (pp. 219–225).
- Jiexiong, W., Guodong, C., & Yi, C. (2015). Improved B-snake segmentation method for liver CT images. *Computer Engineering and Applications*, 51(9), 152–157.
- Ng, P., & Pun, C.-M. (2012). Skin segmentation based on human face illumination feature. *IEEE Computer Society*, 3, 373–377.
- Xu, T., Mandal, M., Long, R., Cheng, I., & Basu, A. (2012). An edge-region force guided active shape approach for automatic lung field detection in chest radiographs. *Computerized Medical Imaging and Graphics*, 36(6), 452–463.
- Alexa, M., & Adamson, A. (2009). Interpolatory point set surfaces-convexity and hermite data. *ACM Transactions on Graphics*, 28(2), 307–308.



12. Ackermann, M. R., Märtens, M., Raupach, C., Swierkot, K., Lammersen, C., & Sohler, C. (2012). StreamKM++: A clustering algorithm for data streams. *Journal of Experimental Algorithmics*, *17*, 254–257.
13. Gálvez, A., & Iglesias, A. (2012). Particle Swarm optimization for non-uniform rational B-spline surface reconstruction from clouds of 3D data points. *Information Sciences*, *192*, 174–192.
14. Patane, G., Li, X. S., & Gu, D. X. (2013). Surface- and volume-based techniques for shape modeling and analysis. *SIGGRAPH Asia Courses*. ACM (pp. 65–67).
15. Praun, E., Finkelstein, A., & Hoppe, H. (2000). Lapped textures. In *Proceedings of the 27th annual conference on computer graphics and interactive techniques*. ACM (pp. 465–470).
16. Chen, K., Johan, H., & Mueller-Wittig, W. (2013). Simple and efficient example-based texture synthesis using tiling and deformation. In: *Proceedings of the ACM SIGGRAPH symposium on interactive 3D graphics and games*. ACM (pp. 145–152).
17. Yoo, D. J., & Kwon, H. H. (2009). Shape reconstruction, shape manipulation, and direct generation of input data from point clouds for rapid prototyping. *International Journal of Precision Engineering*, *10*(1), 103–113.
18. Nagai, Y., Ohtake, Y., & Suzuki, H. (2015). Tomographic surface reconstruction from point cloud. *Computers and Graphics*, *46*, 55–63.
19. Liu, W., Cheung, Y., Sabouri, P., Arai, T. J., Sawant, A., & Ruan, D. (2015). A continuous surface reconstruction method on point cloud captured from a 3D surface photogrammetry system. *Medical Physics*, *42*, 238–240.