Geologic Surface Reconstruction Based on Fault Constraints

Suyun Li
Nanjing University
lisuyun@graphics.nju.edu.cn

Jie Tang
Nanjing University
jietang@graphics.nju.edu.cn

Gangshan Wu
Nanjing University
gswu@graphics.nju.edu.cn

Abstract

Some applications in 3D mesh surface modeling, e.g. geologic surface reconstruction, require linear discontinuous mesh pieces reconstructed from input point cloud. Unfortunately as far as we know, those previously developed approaches failed in reconstructing such mesh directly from point cloud. This paper presents a new method which using mesh fitting to create discontinuity during geologic surface reconstruction. After improved Hoppe's reconstruction method, each vertex located at discontinuous area is adjusted to a new position based on quadric error metrics. Experiment results show that our method could generate high quality surfaces which satisfy both geometric and geologic constraints.

Keyword: *surface reconstruction; mesh fitting; quadric error metrics (QEM)*

1. Introduction

Surface reconstruction from point cloud is a well studied problem in computer graphics. And in particular on an abstract problem defined by Hoppe et.al[1], the input is a point cloud X in the Euclidean space $\boldsymbol{R}^3$, without any additional structure or other information, and the desired output is an initial triangular surface M_0 which approximates the unknown surface represented by X . In practice, point cloud for surface reconstruction come from a variety of sources: laser range scanners, contact probe digitizers, seismic prospecting method or medical imagery.

In 3D geology modeling, surface reconstruction also is a prominent problem. There are two basic surfaces, horizon and fault. Horizon is a mesh surface composed of several linear discontinuous pieces. And the discontinuities are the outstanding feature and often occur in geology[2]. Better approximation should be obtained near the discontinuous area. Some important geologic elements, e.g. geology fracture, block and strata are generated from the discontinuities. Consequently, to some degree surface reconstruction in 3D geology modeling is different from other mesh surface applications. Unfortunately as far as we know, those previously developed approaches failed in reconstructing such horizon surface directly from point cloud. According to our approach presented in this paper, firstly initial mesh surface is reconstructed from X using our improved Hoppe's method, and then discontinuous horizon pieces are generated from fault constraints. In order to better approximate the unknown surface represented by X, fitting process is applied to the initial discontinuous pieces based on fault constraints using QEM.

The input to the surface reconstruction problem is a point cloud X in the Euclidean space $\boldsymbol{R}^3$ from the surface W of a 3D object, and the output surface M_0 should be a piecewise-linear approximation of W.

The mesh fitting problem considered in this paper can be roughly stated as follows: After surface reconstruction which results in M_0, find a mesh M of the same topological type as M_0 that better fits the unknown surface W near the discontinuous area and based on fault constraints.

In a nutshell the principal contributions of our reconstruction method are the following:

1. Improving Hoppe's algorithm on normal computing and triangle extraction
2. Optimizing horizon pieces based on fault constraints using QEM

The remainder of this paper is organized as follows: In Section 2 we review related work in surface reconstruction and fitting approaches. Improvements to Hoppe's algorithm are presented in Section3. We introduce QEM and our fitting approach in Section 4 and Section 5. And provide approximation error evaluation and results in Section 6 and Section 7 respectively. Finally we conclude in Section 8 by summarizing our approach and future research.

2. Related Work

Surface reconstruction from unorganized point cloud is a wide research topic, and it has motivated a

0-7695-3065-6/07 $25.00 © 2007 IEEE
DOI 10.1109/DMAMH.2007.58

large body of research in computer graphics and previous approaches can be broadly grouped into three categories[3]:

(1) Address the surface reconstruction problem through the use of computational geometry techniques[4] [5]. These methods proceed by computing either the Delaunay triangulation of the point cloud or the dual Voronoi diagram and using the cells of these structures to define the topological connectivity between the points.

(2) Address the surface reconstruction problem by directly fitting a surface to the point cloud[6] [7]. These approaches represent the base shape as a collection of points with springs between them and adapt the shape by adjusting either the spring stiffness or the point position as a function of the surface information.

(3) Address the surface reconstruction problem by defining an implicit function to the point cloud and then extracting the reconstructed surface as an isosurface of the function[1] [8].

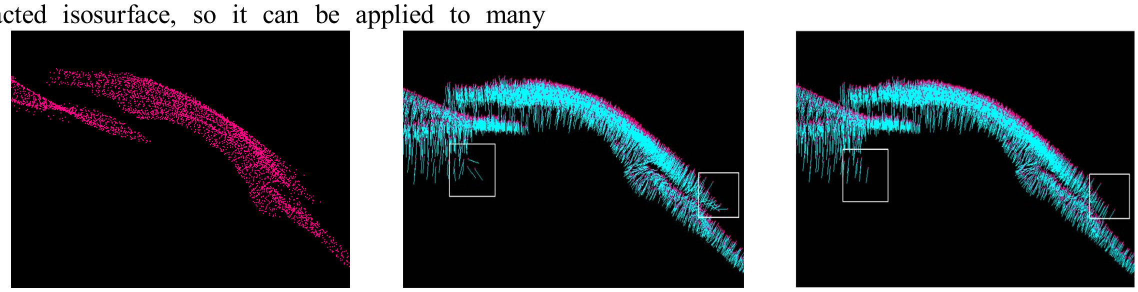
In brief, there is no clear recipe how to proceed for obtaining the best mesh surface reconstructed from point cloud X for all applications. It is relatively easy to check whether the surface M is within a tolerance or not, it is more difficult to decide whether it is reconstructed well enough.

In general, our surface reconstruction approach belongs to the third category, and has following advantage. First, the approach can process unorganized and unevenly distributed point cloud, and return a model with hole which is water-tight or not. And second, the use of implicit function does not place any restrictions on the topological complexity of the extracted isosurface, so it can be applied to many different 3D models.

3. Improvements to Hoppe's Algorithm

According to Hoppe's algorithm[1], the fist step toward defining the implicit function is to compute an oriented tangent plane for each sample point. And the tangent plane for point p is determined by its k-neighborhood. Here k-neighborhood means k points of X which have nearest Euclidean distance to p. But in 3D geology modeling, as Figure 1(a) shows, point cloud is often separated into several discontinuous parts by faults which are not included in Figure 1(a). If we find k-neighborhood of each sample point without constraint as Hoppe algorithm did, some points' normals will be incorrect, as shown in Figure 1(b)(inside the rectangles). The reason is that those points' k-neighborhood crossed a fault. Figure 2 shows the detail. p_2 is one of p_1's k-neighborhood. The solid line indicates a fault. Therefore, p_2 and p_1 are located across the fault. Using p_2 to compute the normal of p_1 will lead to an incorrect result. Here we have to find a way to improve it.

In our proposed method, since X is assumed to be a ρ–dense, δ-noise sample of W, we make a rule that $d(q, N)< \rho+\delta$, where q is the candidate neighbor point of p, N is neighbor point set of p which has been searched, and d means minimum distance between q and N. So each neighbor point of p will not cross fracture or hole, and the normal will be more precise. Figure 1(c) demonstrates the normal computing result using our improvement.



(a)original point cloud

(b)normals computed using Hoppe's method

(c)normals computed using our method

Figure 1. Comparison of two normal computing methods

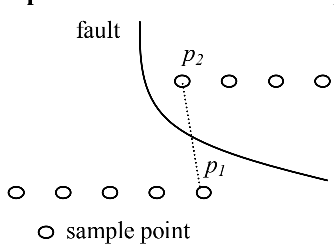


Figure 2. Imprecise *k*-neighborhood search

Marching Cubes (MC) algorithm[9] is the most popular and classical isosurface extraction algorithm. During the implementation we found that if some points of X distribute as Figure 3 shows, some triangles will be wrongly extracted. Figure 4(a) shows the wrong surface extraction result, and Figure 3 demonstrates the reason. Signed distances of cube vertices c_1 and c_2 are positive while c_3 and c_4 are negative. So there should be two interpolate points between c_1c_3 and c_2c_4, and then two obviously wrong triangles will be extracted from this cube. In Figure 3, the red line is a line wrongly extracted from the one of cube's six faces. In order to avoid this, we introduce a user-specified parameter-flip angle f. According to each cube vertex v, if the found tangent plane whose center is closest to v doesn't intersect with the cube, we label v semi-undefined. During the extraction process, if any one of eight cube vertices is labeled semi-undefined, a mean tangent plane normal $\boldsymbol{n_p}$ is computed in terms of those vertices which are not label semi-undefined. The extracted triangle's normal $\boldsymbol{n_t}$ is evaluated. If the angle between $\boldsymbol{n_p}$ and $\boldsymbol{n_t}$ is greater than f, in inequality $\boldsymbol{n_p}\cdot\boldsymbol{n_t} > f$, where $\boldsymbol{n_p}$ and $\boldsymbol{n_t}$ has been unitized, none triangle will be extracted from this cube. We make such improvement based on the assume, that in terms of local shape, the extracted triangle will not flip too much against unknown surface W, and the tangent plane is a local linear approximation of W. Figure 4(b) demonstrates the improved result.

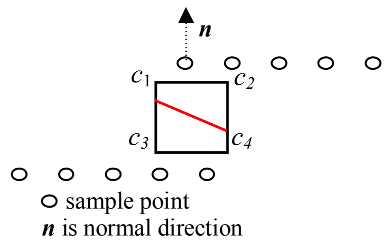
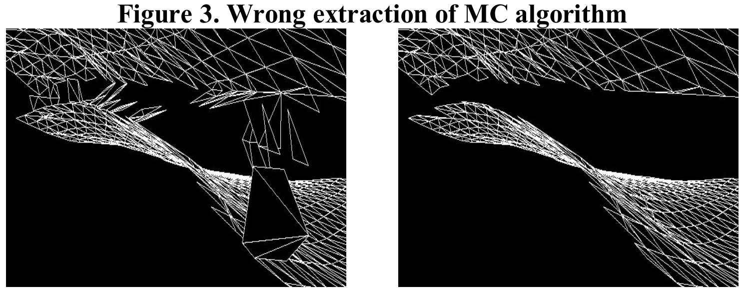


Figure 3. Wrong extraction of MC algorithm



(a) f=180º (b) f=30º

Figure 4. Comparison of two different triangle extraction results

4. QEM

4.1. QEM Introduction[10]

Garland and Heckbert first proposed QEM(Quadric Error Metrics) which was originally used in mesh simplification based on the iterative contraction of vertex pairs. Each vertex in mesh is the solution of the intersection of a set of planes—namely, the planes of the triangles that meet at the vertex.

The standard representation of a plane is the set of all points for which $ax + by + cz + d = 0$ where $\boldsymbol{n} = (a, b, c)^T$ is a unit normal and d is a scalar constant. So the squared distance of a vertex $\mathbf{v} = (x, y, z)^T$ from the plane is given by the equation

$$D^2(\boldsymbol{v}) = (\boldsymbol{n}^T\boldsymbol{v} + d)^2 = (ax + by + cz + d)^2$$

For a vertex p with an associated set of triangle planes, the error metric at that vertex can be defined to be the sum of squared distance of the vertex to all the planes in the corresponding set

$$QEM_p = \sum_i D_i^2(\boldsymbol{v}) = \sum_i (\boldsymbol{n}_i^T\boldsymbol{v} + d_i)^2$$

And $D_i^2(\boldsymbol{v})$ can be rewrited as follows.

$$D^2(\boldsymbol{v}) = \boldsymbol{v}^T(\boldsymbol{nn}^T)\boldsymbol{v} + 2(d\boldsymbol{n})^T\boldsymbol{v} + d^2$$

where $\boldsymbol{nn}^T$ is the outer product matrix

$$\boldsymbol{nn}^T = \begin{bmatrix} a^2 & ab & ac \\ ab & b^2 & bc \\ ac & bc & c^2 \end{bmatrix}$$

A triple is defined as $\boldsymbol{Q} = (\boldsymbol{A}, \boldsymbol{b}, c)$ where $\boldsymbol{A}=\boldsymbol{nn}^{\mathrm{T}}$ is a 3×3matrix, $\boldsymbol{b}= d\boldsymbol{n}$ is a 3-vector, and $c=d^2$ is a scalar.

The triple $\boldsymbol{Q}$ is called a quadric error metrics(QEM) or quadric matrix. Given a vertex $\mathbf{v} = (x, y, z)^T$

$$\boldsymbol{Q}(\boldsymbol{v}) = D^2(\boldsymbol{v}) = \boldsymbol{v}^T\boldsymbol{A}\boldsymbol{v} + 2\boldsymbol{b}\boldsymbol{v}^T + c$$

$\boldsymbol{Q}(\boldsymbol{v})$ is also called quadric error, and it represents the squared distance of vertex v to a particular plane.

The QEM can provide a useful characterization of local surface shape. And it requires only modest storage space and computation time. This is because:

$$\boldsymbol{Q}_i(\boldsymbol{v})+\boldsymbol{Q}_j(\boldsymbol{v})=(\boldsymbol{Q}_i+\boldsymbol{Q}_j)(\boldsymbol{v})$$

$$\boldsymbol{Q}_i+\boldsymbol{Q}_j=(\boldsymbol{A}_i+\boldsymbol{A}_j,\boldsymbol{b}_i+\boldsymbol{b}_j,c_i+c_j)$$

Some QEMs is added, but storage space doesn't increase.

4.2. Geometric Interpretation of QEM

In this section the geometric interpretation of QEM will be discussed. Given a set of k planes defined by equations of the form $\boldsymbol{n}_i^T\boldsymbol{v}+d_i=0$ (i=1, …, k). Let $\boldsymbol{N}$ be the $k\times 3$ matrix of normals.

$$\boldsymbol{N}=\begin{bmatrix}\boldsymbol{n}_1{}^T\\ \boldsymbol{n}_2{}^T\\ \dots\\ \boldsymbol{n}_k{}^T\end{bmatrix}=\begin{bmatrix}a_1\ b_1\ c_1\\ a_2\ b_2\ c_2\\ \dots\\ a_k\ b_k\ c_k\end{bmatrix}$$

And let $\boldsymbol{d}$ be the corresponding k-vector of offsets

$$\boldsymbol{d}=\begin{bmatrix}d_1\\ d_2\\ \dots\\ d_k\end{bmatrix}$$

Suppose that a point $\mathbf{v}=(x,y,z)^T$ is to be located which lies at the intersection of these planes. Such a point would satisfy equation $\boldsymbol{N}\boldsymbol{v}+\boldsymbol{d}=0$

In general if k>3 this system of equation will be over-constrained. It is impossible that all the planes will intersect at a single point. In order to find the point which best fits this set of planes, the least squares method is applied. So the essence of QEM is to find the optimal approximation point by applying the least squares method.

5. Mesh Fitting Based on QEM

5.1. Basic Conception

Our fitting algorithm is based on QEM, which is introduced in Section 4.1. For the sake of convenience, each sample point in point cloud is associated with a symmetric 4×4 matrix $\boldsymbol{Q}$ instead of the triple QEM. If a point's coordinate is $\mathbf{v}=(x,y,z)^T$ and its tangent plane's unit normal is $\mathbf{n}=(a,b,c)^T$ which can be evaluated by Principle Component Analysis method[1], then

$$\boldsymbol{Q}=\begin{bmatrix}a^2 & ab & ac & ad\\ ab & b^2 & bc & bd\\ ac & bc & c^2 & cd\\ ad & bd & cd & d^2\end{bmatrix}$$

where d can be evaluated from the equation $ax+by+cz+d=0$. To find an optimized position according to k points' $\boldsymbol{Q}$ matrices, we simply sum up these matrices ($\boldsymbol{Q}_{sum}=\boldsymbol{Q}_1+\boldsymbol{Q}_2+\ldots+\boldsymbol{Q}_k$), then get the optimized position from $\boldsymbol{Q}_{sum}$ by using least square method. If $\boldsymbol{Q}_{sum}$ is not invertible, the optimized position is one of the k points with the least error.

In order to make horizon pieces better approximate the unknown surface W featured by $\boldsymbol{X}$ near discontinuous area, some vertices in horizon are located and adjusted. During the adjusting procedure, firstly the $\boldsymbol{Q}$ matrix of each sample point is computed, and then the neighbor points of each vertex in discontinuous area are located with fault constraints. The optimized adjust position is evaluated according to the neighbor points' $\boldsymbol{Q}$ matrices. Finally each vertex is assigned the optimized coordinate.

Figure 5 illustrates one vertex adjust procedure. A new position is evaluated based on the five points' $\boldsymbol{Q}$ matrices, then the vertex is updated to the new position.

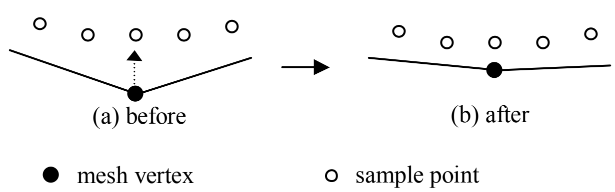


Figure 5. Vertex adjust

5.2. Summary of Algorithm

The algorithm can be quickly summarized as follows:

1. Compute the $\boldsymbol{Q}$ matrices for all the sample points in X
2. Get an unprocessed vertex v_i in discontinuous area
3. Get neighbor sample points of v_i with fault

constraints

4. Compute optimized position according to neighbor points' $\boldsymbol{Q}$ matrices, and assign the optimized position to v_i
5. If all vertices in discontinuous area have been processed, algorithm exists successfully. Else return to Step 2.

It should be noted that since the point cloud maybe not evenly distributed, it may be better if vertex only be adjusted along its normal direction.

6. Approximation Error Evaluation

As stated earlier, the primary aim of mesh fitting is to produce a surface approximation which should be as same as possible to the unknown surface represented by point cloud. In order to evaluate the quality of approximation produced by the introduced algorithm, we need an error measurement. We have chosen a metric which measures the average squared distance between the approximation surface and the original point cloud. This is very similar to the E_{dist} energy term used by Hoppe[11]. We define the approximation error $E=E(M,X)$ as:

$$E = \frac{1}{|X_S|} \sum_{p \in X_S} d^2(p, M)$$

where M and X are approximation surface and point cloud respectively, and X_s is sets of point which are the neighbor points of discontinuous area in M. The distance $d(p,M) = \min_{f \in M} \|p - f\|$ is the minimum Euclidean distance from p to the closest face of M.

According to Figure 7 (b) and (c), $E(M,X)$ is 35.6473 and 22.0067 before and after fitting respectively. $E(M,X)$ is reduced by about 38%.

7. Experiment Results

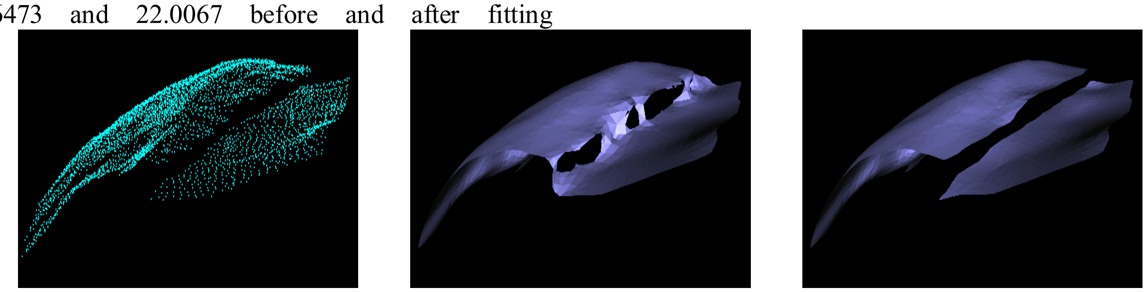
This section provides some examples to demonstrate the proposed approach. A prototype system has been developed for evaluating and testing the approach described in this article. The system is implemented on a DELL workstation with 2.0G MHz CPU and 1G RAM under a RedHat Enterprise operating system. Figure 6-8 illustrate some examples.

All surfaces and blocks in Figure 6-8 are rendered in flat shading.

Figure 6 demonstrates our improvements to Hoppe's algorithm. The point cloud is shown in Figure 6(a), the reconstructed horizon with Hoppe's algorithm is shown in Figure 6(b), and the reconstructed horizon with our improvements is shown in Figure 6(c).

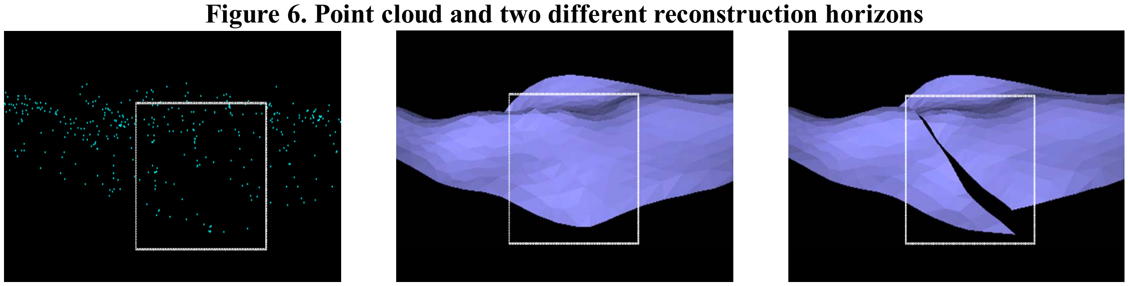
Figure 7 shows another reconstruction example of a typical horizon. Figure 7(a) is input point cloud, and (b) and (c) are corresponding reconstruction horizons. Compared with (b), there are two discontinuous horizon pieces in (c) because of the introduced fitting algorithm. It is obvious that (c) approximates point cloud better than (b).

Figure 8 also shows other examples for our reconstruction approach application. These geologic blocks are made up of geologic surfaces which are not fitted in (a) and (c) and which are fitted in (b) and (d). In light of (b) and (d) there are fractures between two blocks and fracture is a very important feature in geology. Especially as to (d), it is a cross-section and the two top horizon pieces in block are discontinuous.



(a) (b) (c)

Figure 6. Point cloud and two different reconstruction horizons



(a) (b) (c)

Figure 7. Point cloud(a), reconstruction horizons without fitting (b) and with our fitting method(c)

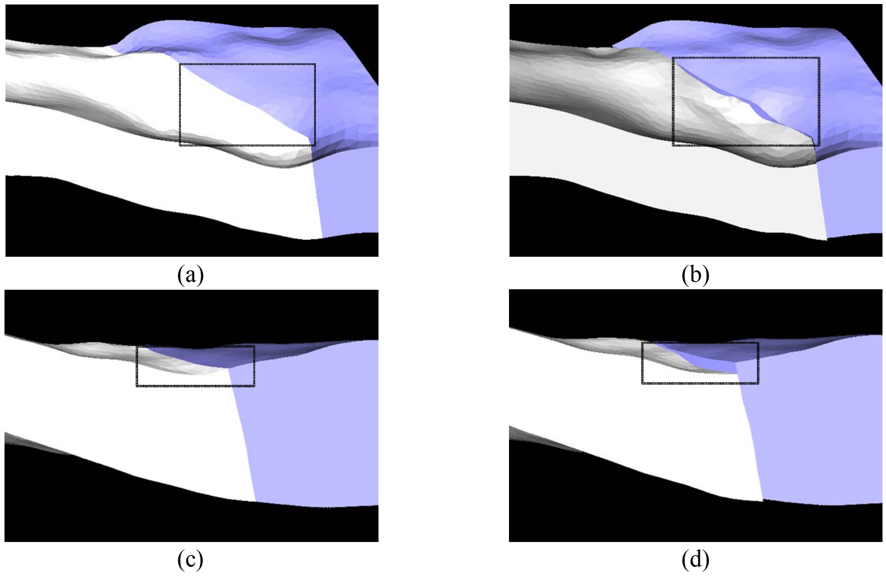


Figure 8. Blocks made up of mesh surfaces and its' cross-sections

8. Conclusions

In this paper we have presented a novel method for surface reconstruction in 3D geology modeling. Firstly we make some improvements to Hoppe's algorithm on normal computing and triangle extraction because of fault constraints. Fitting algorithm using QEM is introduced in order to make better approximation near discontinuous area. Experiment results show that our method is capable of generating geologic surface which satisfy the requirements of 3D geology modeling.

Future work may concern about how to avoid possibly producing sliver triangles during fitting process, which can result in poor approximations[12] and lead to instability during later process. And additional constraints like orientation data will be included into the developed surface reconstruction approach.

Reference

[1] H. HOPPE, T. DEROSE, T. DUCHAMP. et al, "Surface reconstruction from unorganized points", *Computer Graphics*, ACM Press, New York, 1992, pp. 71-78.

[2] T. FRANK, A.L. TERTOIS, J.L. MALLET, "3D-reconstruction of complex geological interfaces from irregularly distributed and noisy point data", *Computers & Geosciences*, 2007, pp. 932-943.

[3] M. KAZHDAN, "Reconstruction of solid models from oriented point sets", Eurographics Symposium on Geometry Processing, Vienna, 2005, pp. 73-82.

[4] N. AMENTA, M. BERN, M. KAMVYSSELIS, "A new Voronoi-based surface reconstruction algorithm", SIGGRAPH, ACM Press, New York, 1998, pp.415-421.

[5] T. DEY, S. GOSWAMI, "Tight cocone: A water tight-surface reconstructor", 8th ACM Symposium on Solid Modeling Application, 2003, pp. 127-134.

[6] D. TERZOPOULOS, M. VASILESCU, "Sampling and reconstruction with adaptive meshes", IEEE Conference on Computer Vision and Pattern Recognition, 1991, pp. 70-75.

[7] Y. CHEN, G.MEDIONI, "Description of complex objects from multiple range images using an inflating balloon model", *Computer Vision and Image Understanding*, 1995, pp.325-334.

[8] G. TURK, J. O'BRIEN, "Modeling with implicit surface that interpolates", ACM Transaction on Graphics, 2004, pp.855-873.

[9] W. LORENSEN, H. CLINE, "Marching cubes: A high resolution 3D surface construction algorithm", SIGGRAPH, ACM Press, Anaheim, 1987, pp. 163-169.

[10] M. GARLAND, "Quadric-Based Polygonal Surface Simplification", *Ph.D. dissertation*, Computer Science Department, CMU, 1999, 210.

[11] H. HOPPE, T. DEROSE, T. DUCHAMP. et al, "Mesh optimization", SIGGRAPH, ACM Press, New York, 1993, pp. 19-26.

[12] I. BABUSKA, A. AZIZ, "On the angle condition in the finite element method", *SIAM Journal of Numerical Analysis*, 1976, pp. 214-227.