SIDE RATIO CONSTRAIN BASED PRECISE BOUNDARY TRACING ALGORITHM FOR DISCRETE POINT CLOUDS

HUANG Xianfeng, CHENG Xiaoguang, ZHANG Fan, GONG Jianya

State Key Laboratory of Informaiton Engineering in Surveying, Mapping and Remote Sensing (Wuhan University), 129 Luoyu Road, Wuhan 430079, Hubei, China –
(hxf_whu, chengxiaoguang985 zhangfan128, geogjy)@163.com

KEY WORDS: Laser Scanning (LiDAR), Edge detection, Aerial Photogrammetry, Feature Extraction, Buildings, Tracking

ABSTRACT:

Boundary tracing of discrete points is an important step to building model reconstruction using LiDAR data, its result directly effects the location regularization of building corners and the reconstructed building models. At present, the convex hull based boundary tracing algorithm is not suitable for buildings with many concave part and grid index based algorithm is too complicated and not stable enough. This paper proposes a side ratio constrain based boundary tracing algorithm for discrete points. This algorithm can effectively trace the boundary of concave polygons with holes. It doesn't depend on point densities heavily since it uses side ratio as qualification. This algorithm was finally proved in experiments.

INTRODUCTION

LiDAR (Airborne Light Detection and Ranging) is a kind of led point clouds). The building model reconstruction based on LiDAR data is a mean of increasing importance in the 3D model reconstruction of digital cities (Xianfeng Hwang, 2006). During the analyzing and processing of LiDAR data, after determining the building area by classifying and extracting complicated building roof surfaces, how to trace the boundary in discrete points that belongs to the same roof surface and get the vector boundary of model directly effects the regularization of building corners and expression of building model. This problem can be abstracted as how to extract outer and inner boundaries from discrete point clouds. It has broad applications in GIS, compute geometry, computer emulation, etc.

Previous methods can be mainly classified into two categories: convex hull algorithm and grid based searching algorithm. The research to convex hull of discrete points is much and the most classical algorithm is Graham (Graham, R, 1972), but convex hull algorithm can't resolve the problem of concave boundaries which is very common in reality. Not all buildings shape are convex boundary, there always exists concave boundary in the corners of buildings. Sampath used the improved convex hull algorithm raised by Jarvis (Jarvis, 1977) to trace the building boundary (Aparajithan Sampath, J.S., 2007). Although in this way concave boundary can be gotten, when searching for the next point, program needs to judge whether all the candidate boundary points are in the circle described by current point and radius threshold and calculate the radial slope (current point as start and point in the circle as end), it is inefficient because of the vast computation. (Chen Tao, Li Guangyao, 2004) raised a grid index based discrete points boundary tracing algorithm, despite it could get outer boundary and inner hole boundary with good precision, but its performance largely depends on the width of search box, it's also very hard to determine the search direction and not easy to gain a closed boundary. While there are several points in a search box, how to confirm the connect sequence is also difficult. Some other methods interpolate points to regular grid (Deng Fei, 2006), these methods are easier because they translate boundary tracing of discrete point clouds to edge detection of images, but accuracy will be reduced in the process of interpolation.

measurement system that has been greatly developed in recent years; it can quickly acquire large amounts of 3D points (cal

In essence, the difficulty of discrete point's boundary tracing lies in how to make use of the spatial relationship between points and set reasonable constrains or principles to judge which points are possible boundary points and then connect them. Creating TIN for discrete points and utilizing the rich information, for example, the spatial relationship between points and points, points and sides, sides and triangles, triangles and triangles is possible to solve this problem reasonably. This paper proposes a new side ratio constrain based discrete points boundary tracing algorithm. The method could trace the outer boundary and inner hole boundary of point clouds with great efficiency and speed.. Compared with previous methods, the proposed method's dependence on point density is relatively less, so the algorithm is of much practical value.

2. BOUNDARY TRACING ALGORITHM

The core of this algorithm is making full use of the topological relationship between points, sides, triangles and geometry features (length, angle, etc.) to analyze and judge. The basic steps include: (1) create TIN for discrete points using the method of interpolating point one by one; (2) get the primitive boundary that envelopes all the points; (3) search inwards for every line segment in the primitive boundary whose length exceeds the threshold and get the coarse outer boundary; (4) find the triangle with at least one side exceeds the threshold, push back the short side into the array of current hole boundary and expand longer side to another adjacent triangle iteratively, connect these short sides and get the coarse hole boundary; (5) gradually expand the coarse outer boundary and hole boundary with the side ratio constrain and get more precise boundary.

2.1 Primitive Boundary Tracing

TIN is generated from point clouds using classical method of interpolating point one by one (Tang Guoan, Liu Xuejun, Lv

Guonian, 2005). This method has the advantage that it could use rectangle envelope to get the primitive boundary. While creating the TIN, a fairly large square (also one kind of rectangle) is used as primitive envelope, as shown in Figure 1. The biggest x and y coordinate of the four vertexes is both 2147483647 and the smallest x and y coordinate is both -2147483647. These vertexes also connect with other points and construct Delaunay triangles. These triangles are called outer triangles and the others are inner triangles, as shown in Figure 1, edge P_1P_2 belonged to an outer triangle and an inner triangle, so it is a line segment that forms primitive outer boundary.

After creating TIN, analyze all the outer triangles, if a vertex of triangle T is one of the four envelope vertexes (A, B, C or D in figure.1) and another two are not, then record the line segment made up by the two vertexes not on envelope. After this step, connect all the recorded line segments and get a ring S. S is the primitive boundary of these discrete points, as the ring formed by wide solid line segments shown in figure 1. But this ring is far away from ideal situation, especially when the points show the shape with concave boundary, which needs further optimization.

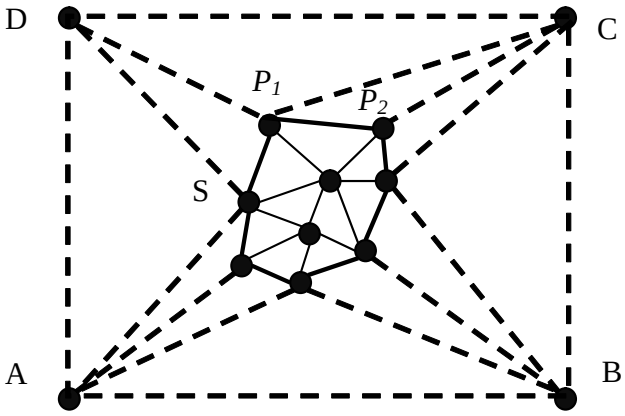


Figure 1. Create TIN using the method of interpolating points one by one

2.2 Discrete Points Primitive outer Boundary Shrinkage Based on Side Length Threshold

The ring got in the previous step can't reflect this kind of concave boundary in Figure 2, some points that shows concave part of the boundary are probably lost. Moreover, some of the line segments in S are too long to represent the precise boundary. Therefore, ring S needs further shrinkage operation, so it can be a concave polygon and reflect the real building boundary. The shrinkage is actually a process of using one or two short sides to substitute the longer side. Shorter sides can describe boundary more precisely than the longer, so shrinkage is also a process of approaching to real boundary step by step.

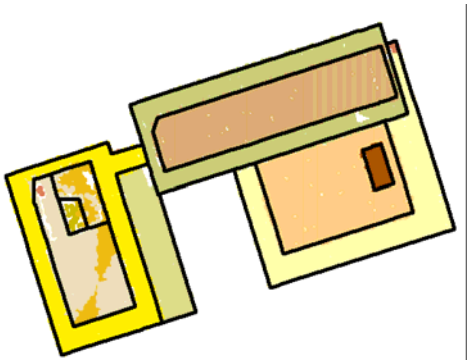


Figure 2. Concave boundary of a building

In the paper a threshold β is given for the maximum length of the line segments that forms the outer boundary of the discrete points, for example, 5m. If P_1P_2 (also adjacent to t_1, as shown in Figure 3) is one line segment that forms S and longer than β, then analyze P_2P_3 and P_3P_1 (which make up t_1). If P_3P_1 is shorter than β, push P_3P_1 into coarse outer boundary array, else expand to the other triangle t_2 that P_3P_1 is adjacent to and calculate whether P_1P_4 and P_4P_3 is longer than β. If not, put them into boundary array, else expand again, until current line segment is not longer than β and put into boundary array. Treat P_2P_3 in the same way. Do the same processing to all the other line segments that form S. After an iterative operation to all the line until no lines length longer than the given threshold line, then stop. That can produce a non-convex boundary and the boundary can be connected as a ring. This result reflects the character of real boundary better. β determines the level of detail, but the threshold couldn't be very small, once it is near or much smaller than the average point distance, the coarse boundary will break into several rings.

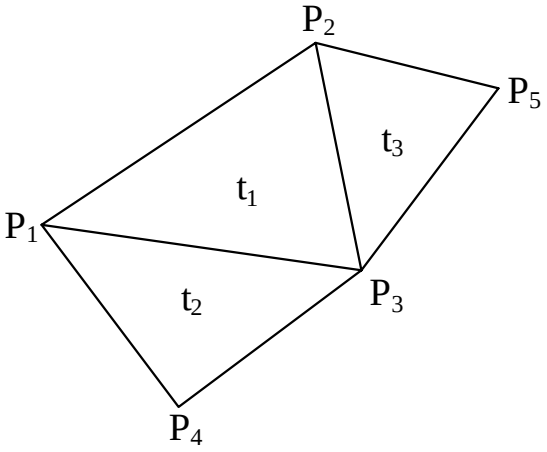


Figure 3. Method to optimize primitive boundary

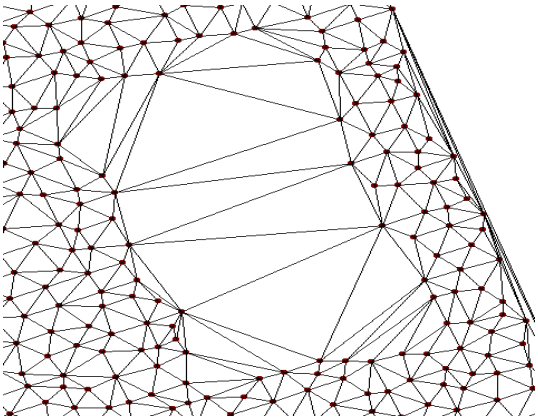


Figure 4. Character of hole

2.3 Inner Hole Coarse Boundary Detection of Discrete Points

Because the discrete points don't distributed evenly in surface, so it makes the existence of inner hole possible. The hole can be defined as in a certain range the point density is so small that a rectangle with certain size could be place in it without containing any points (as shown in Figure 4).

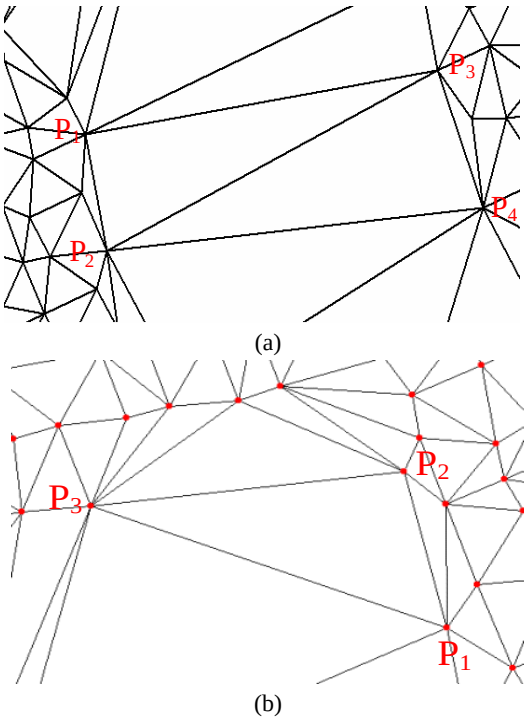


Figure 5. Two kinds of typical DT that made up of hole

The principle for TIN to judge hole: if there really exist a hole, then the points on hole boundary must link with each other (as shown in Figure 5). One common condition is that two points P_1 and P_2 on one side of the hole form a triangle with one point P_3 on the other side of the hole (Figure 5(a)). The length of P_1P_3 and P_2P_3 reflect the dimensionality of the hole. It's quite reasonable to assume that P_1P_3 is long enough and could reflect the dimensionality of the hole. Therefore, first, all triangles in TIN on coarse outer boundary need to be judged.

Condition 1: triangle t has three vertexes P_1, P_2, P_3, at least one of P_1P_2, P_2P_3 and P_3P_1 are longer than a fixed threshold γ, for example, 2 m. If one triangle t satisfies condition 1, then record and search t. The concrete steps are as follows:

(1)If P_2P_3 and P_3P_1 are longer than γ, set p_1p_2 = P_1P_2, go to step (2), set $p_2p_3 = P_2P_3$, go to step (3), set $p_2p_3 = P_3P_1$, go to step (3). Finally go to step (4). Other conditions can be disposed in this way: for sides not longer than γ, go to step (2); for sides longer than γ, go to step (2).

(2)Judge p_1p_2 is on found boundary or not. If not, push p_1p_2 into hole boundary array, else give up the search along this triangle. This is because one side can make up just one hole boundary at the most.

(3)For p_2p_3, the length of another two sides except p_2p_3 in another triangle that p_2p_3 is adjacent to need to be calculated to see whether they are longer than γ or not. If p_2p_4 is not longer than γ, then set $p_1p_2 = p_2p_4$, go to step (2), else set $p_2p_3 = p_2p_4$, go to step (3). If p_3p_4 is not longer than γ, then set $p_1p_2 = p_3p_4$, go to step (2), else set $p_2p_3 = p_3p_4$, go to step (3). The process iterates until the newly found line segment is not longer than γ and go to step (2).

(4)If the number of line segments got in previous steps are more than 3, these line segments can be connected as a ring and the ring area is larger than a previous set threshold ω, then these line segments is considered to be valid. That valid ring is the boundary.

This process can find the primitive boundary of holes, but this result is coarse and not satisfactory. The concave part of the boundary couldn't be traced precisely, so the boundary needs to be optimized just as coarse outer boundary.

2.4 Coarse Boundary Optimizing Algorithm Based on Side Ratio

The main purpose of this step is to optimize the coarse outer and inner hole boundary, to get better concave boundary and avoid the errors caused by only using side length threshold. The errors are always in this form: although one line segment is not longer than previously set threshold, the shape of the triangle adjacent to this line segment is not stable, conditions such as one side is very short while the other two are relatively long and the side on the coarse boundary is long while the other two are relatively short often appears.

TIN has strict requirement for the shape of TIN and the most important one is trying to make sure that three sides are of same length. Once points are distributed evenly in surface, the triangles in TIN almost satisfy this requirement. If unstable triangle appears, it's reasonable to think that the distribution of

points is not even and there is a hole. For example, in the corners of buildings, the points collected by LiDAR distribute along the wall. Obviously, the point density is not average, so the triangles here is not stable and the errors previously referred to occurs. This error couldn't be completely eliminated using only side length threshold. Figure 6 shows a concave corner of a building, the black line is the coarse boundary got in last step. It's quite evident that despite P_1P_2 is shorter than threshold, the shape of triangle t_2,t_3,t_4,t_5,t_6,t_7 is unstable, the distribution of points $P_1,P_4,P_7,P_6,P_9,P_8,P_5,P_3,P_2$ is not even, t_2,t_3,t_4,t_5,t_6,t_7 should be out of outer boundary. Using side length threshold can only get the black line which doesn't reflect the boundary accurately.

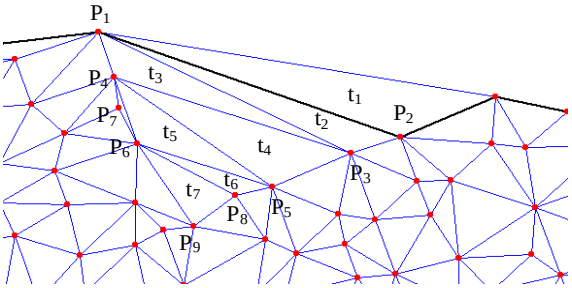


Figure 6. The error caused by only using line segment threshold to trace boundary

In order to get the boundary that is expressed by $P_1,P_4,P_7,P_6,P_9,P_8,P_5,P_3,P_2$, this paper employs an optimized and side ratio constrain based boundary tracing algorithm. When the length of the longest side is α or more times that of a shorter side, the two shorter sides will substitute the longest side. This method is brief and effective. The concrete steps are as follows:

(1)Dispose the entire line segment P_iP_{i+1} that makes up coarse boundary and go to step (2). Connect all the got optimized line segments and get a ring. This ring is optimized boundary.

(2)Do the following processing to P_iP_{i+1}: for the triangle adjacent to P_iP_{i+1} and not searched (if P_iP_{i+1} is on coarse boundary, it is the triangle in outer boundary and adjacent to P_iP_{i+1}; if P_iP_{i+1} is on hole boundary, it is the triangle out of hole boundary and adjacent to P_iP_{i+1}) $P_iP_{i+1}P_j$, find the shorter one P_mP_j and the longer one P_nP_j in $P_{i+1}P_j$ and P_iP_j (if m is i+1, then n is i; if m is i, then n is i+1). Calculate the side ratio λ of P_iP_{i+1} and P_mP_j. If λ is bigger than α (generally bigger than 1.5 and smaller than 2.5), then this triangle is considered to be invalid, go to step (3), else go to step (4).

(3)Push P_mP_j into optimized line segment array. Set $P_iP_{i+1}=P_nP_j$, go to step (2).

(4)Push P_iP_{i+1} into optimized line segment array, go to step (1).

3. EXPERIMENTS AND ANALYSIS

The first experiment chose the point clouds shown in the left picture of Figure 7 to validate that the proposed algorithm's performance doesn't rely on the threshold very much. The point density is about 1 point per m^2 and there are obvious concave boundary and hole in the data. After getting the primitive boundary, $\beta=10m$ and $\gamma=3m$ is given to do further processing and the right part of Figure 7 is the result. In the final process of boundary optimizing, $\lambda=1.8$ and the result is shown in Figure 8(a) and seems satisfactory. When $\beta=7m$ and $\beta=15m$, the result can be seen in Figure 8(b) and Figure 8(c). Although β are quite different, points outer boundary and hole boundary are all effectively extracted, the results are of little difference. What is clear is that the effect of outer boundary tracing does not rely so much on β. If β is not too big or too small, good result still can be got. However, γ should be set a proper value since it's directly connected with the criterion of how large hole can be called hole. In the process of optimizing coarse boundary, $\lambda=1.8$, this value is empirical and can be considered as a constant value. When $\lambda=1.8$ and γ is different (hole boundary can be got), the results are nearly the same (as shown in Figure 8(a) and Figure 8(d)).

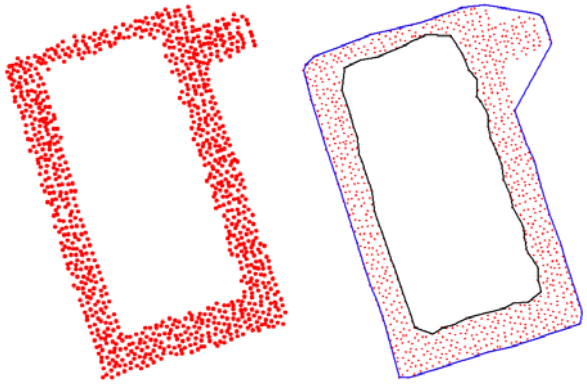
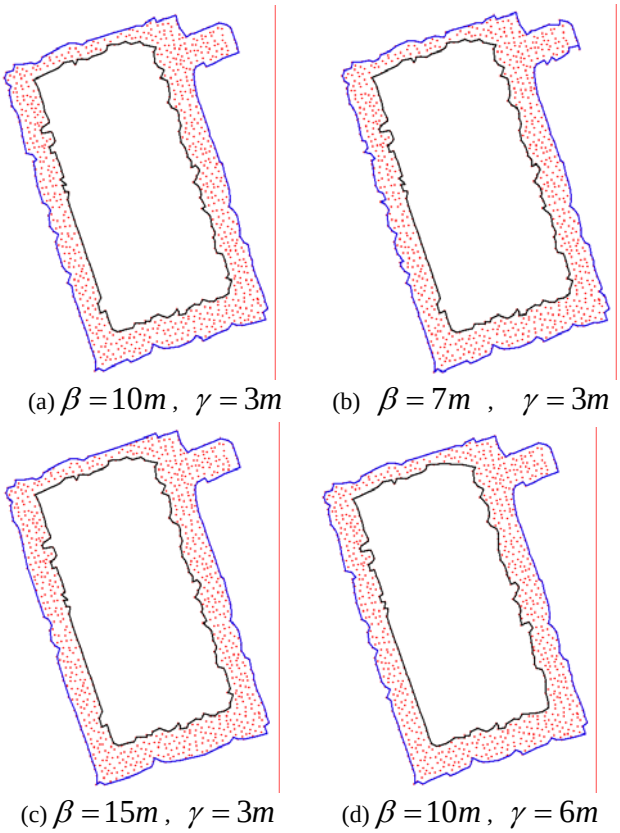


Figure 7. The original point data and primitive boundary detecting result



(a) $\beta = 10m$, $\gamma = 3m$ (b) $\beta = 7m$, $\gamma = 3m$

(c) $\beta = 15m$, $\gamma = 3m$ (d) $\beta = 10m$, $\gamma = 6m$

Figure 8. Optimized boundary

To test the effect of the proposed algorithm applied in data with different point density, the data shown in Figure 10. The point density in the top part of Figure 9 is about 0.8 m between two adjacent points, and that of the bottom of Figure 9 is about 0.1-0.2 m. It clearly shows that this algorithm can still achieve satisfactory result.

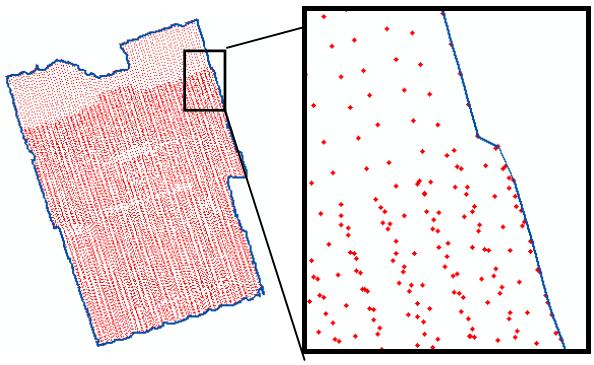


Figure 9. Boundary tracing for data with uneven point density

4. CONCLUSIONS

Boundary tracing of discrete point clouds with complex shape is an fundamental and important processing, however, the level of inosculate between tracing result and real data is often influenced by the value of threshold. The boundary tracing algorithm introduced in this paper is based on side ratio constrain. Tin is used to extract the outer boundary briefly and quickly. Side ratio which is a relative variable is employed as a parameter which can reduce the influence of threshold for result significantly. At the same time, boundary tracing of point clouds with holes is also considered. The following research will focus on adding more constrains, so boundary will not over shrink.

ACKNOWLEDGEMENTS

Thanks for the support from Natural Science Fund of P. R. China (40701154), Chinese National Project (863) (SQ2006AA12Z108506、2006AA12A115) and Open Fund of State Key Laboratory of Remote Sensing Science.

REFERENCES

Aparajithan Sampath, J.S., 2007. Building boundary tracing and regularization from airborne Lidar point clouds. *Photogrammetric Engineering & Remote Sensing*, **73**(7): pp. 805-812.

Chen Tao, Li Guangyao, 2004. Boundary tracing algorithm for discrete point set in plane. *Computer emulation*, 21(3), pp. 21-23.

Deng Fei, 2006. Research on LiDAR and digital images registration and objects extraction

Graham, R, 1972. An efficient algorithm for determining the convex hull of a finite point set. Information Processing Letters, 1: pp. 132-133.

Jarvis. 1977. Computing the shape hull of points in the plane. In: *Proceedings of IEEE Computer Society Conference Pattern Recognition and Image Processing*.

Tang Guoan, Liu Xuejun, Lv Guonian, 2005. Science Press, Beijing, pp. 117-119.

Xianfeng Huang, 2006. Research on 3D building model extraction from airborne LiDAR data. Wuhan University. Doctor Degree Thesis.