The Shortest Hypotenuse-Based Centerline Generation Algorithm

Pan Yuqing[1,2] , Chen Wenchao[3] , Sheng Yehua[1]

Lab of Virtual Geographical Environment, MOE, Nanjing Normal University,China[1]
School of Computer Science and Telecommunication Engineering,Jiangsu University, Zhenjiang 212013,China [2]
Department of Computer Science, Qingdao Harbour Vocational and Technical College,Qingdao 266404, China[3]

Summary
Centerline is a basic element of road networks. Generally, it is generated from the existing road contours. In consideration of contour features in urban area, this study proposes the shortest hypotenuse-based centerline generation algorithm. It constructs right triangles with the key points of contours and connects the midpoints of hypotenuses to form centerlines. The algorithm is quite efficient and produces fewer redundant points. Based on the centerline generation algorithm, this study also proposes an intersection point identification algorithm via polygon construction, and it can fix the problem of inaccurate intersection point generation. Experimental result prove the efficiency and accuracy of the algorithms proposed.
Key words:
Shortest Hypotenuse, Centerline, Intersection identification ,GIS

1. Introduction

Centerline is the most important component of alignment, also the core of highway, urban roads or interchanges and the basis of other parameter setting. Accurate centerline is the precondition for accurate and reliable spatial data of transportation network.

Currently, two kinds of digital maps are available in the market: one is the maps of scanned remote sensing images or common paper maps, and the other is digital maps surveyed by mapping institutions. Centerline is generally generated from these two kinds of digital maps via reprocessing of road contours. Conventional centerline is obtained manually, but it has poor efficiency and accuracy and may have some topology errors that are hardly to be aware of. As a result, many researchers have made intensive study on centerline generation algorithms.

Road information is generally automatically extracted from digital satellite maps via typical algorithms[1,2,3,4]. Automatic road information extraction is pilot in theory and can control information extraction based on knowledge. It is highly automatic and with little manual involvement. But it is mainly applicable to satellite images, so the application scope is limited.

There are several typical algorithms about Generating Centerline from Vector maps too,such as waterlining method (Christensen, 1996)[5], Straight Skeleton Method(Felkel and Obdrzalek ,1998)[6], Thinning Method(Thomas,1998)[7] and Triangulation Method (McAllister and Snoeyink, 2000)[8]. But if the shape of the road contours is complex, application of these method is very difficult and they did not taik about the intersection points

A three-dimensional reconstruction algorithm for multi-contour was proposed by Tang (1999)[9]. It generates centerlines with minimum spanning tree. However, it is not applicable to complicated maps, as (1) it cannot gain contours of both sides of a road, and other entities (such as rivers, lakes), whose contours are similar to road contour, are also present in a vector map; (2) a part of centerlines instead of all centerlines can be generated with minimum spanning tree.

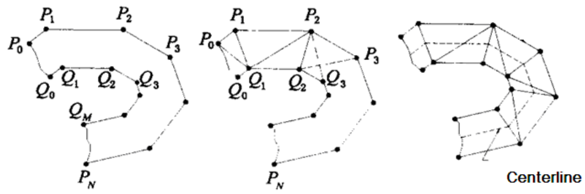
A centerline generation algorithm based on road contour was systematically introduced by Zhu, Wang, and Wan (2003)[10]. This algorithm describes centerline generation in five aspects, including initial contour setting, seeds-based initial contour growth algorithm, common centerline generation algorithm in case of no intersections, intersection identification method and tracking algorithm for vector line with high-frequency noise. This algorithm is much complete, easy to use and stable. It also has some weaknesses, including: (1) the algorithm cannot generate curved centerline in case that both side contours are curved; (2) centerlines generated by this algorithm have too many redundant points (i.e. there will be vertices that have no contribution to the spatial features of centerlines); (3) the algorithm cannot accurately recognize intersection points.

Manuscript received August 5, 2009
Manuscript revised August 20, 2009

To fix the defects and weaknesses of the algorithm proposed by Zhu, Wang and Wan (2003), this study proposes a new algorithm to construct a network including centerlines and intersection points based on road contour network. This new algorithm is applicable to both vector digital maps and survey maps..

2. Contour identification

Centerline generation algorithm proposed by Zhu, Wang and Wan (2003) defines two non-closed contours P and Q as inner and outer contour, and P_0, P_1, …, P_N and Q_0, Q_1, …, Q_M are two vertex queues on two counters respectively (figure2(a)). If the vertices have completely same trend, centerlines generation via a group of adjacent contours can be described as: (1) connect vertices of two contours to generate a group of triangles (figure 2(b)); (2) connect the mid-points of two sides of each triangle to generate a centerline (figure 2 (c)).



(a)Contours and vertices (b) Triangle construction (c) Centerline

Fig. 1 shorst diagonal algorithm

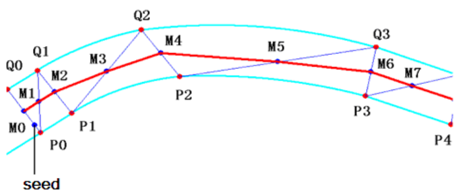


Fig. 2 Triangular interpolation in middle contour

This method has several weaknesses, including (1) it cannot generate curved centerlines. There will be big errors in processing curved contours, the more curved the contour is, the bigger the error will be. Take figure 3 for example, centerline between P_2Q_2 and P_2Q_3 has significant error; (2) the centerline generated has too many redundant points (points that have no contribution to the shape and direction of the centerline). For example, M_1, M_3, M_5 and M7 in figure 3 are redundant points. Too many redundant points will increase topology complexity of centerlines, and will take more time to generate centerlines.

To improve the above algorithm, this study proposes the smallest hypotenuse algorithm which can effectively fix problems such as too many redundant points and curved centerline generation. Assume that vertex queues of two side contours are P_0, P_1, … , P_N and Q_0, Q_1, … , Q_M, and are aligned in the same direction, then the algorithm is as the follows:

(1) assume point P_i of contour P is the current point to be tracked, then obtain the closest point P_i' to P_i on contour Q. If P_i' is the vertex of contour Q, it indicates that hypotenuse P_iQ_j overlaps with the leg P_iP_i' of the right triangle, execute step (3); if P_i' is not the vertex of contour Q, then execute step (2);

(2) compare the length of $Q_{j-1}P_i$' and P_i' Q_j. If $Q_{j-1}P_i$'>P_i'Q_j, construct right triangle P_iP_i'Q_j and execute step (3); if $Q_{j-1}P_i$'<P_i' Q_j and Q_{j-1} is not tracked, construct right triangle $Q_{j-1}P_iP_i$'and execute step (3), and if Q_{j-1} has been tracked, then execute step (4);

(3) identify the mid-point M_k of the hypotenuse. If two side contours tracked by two adjacent hypotenuses are curved, then connect M_{k-1} and M_k with a curve of the same radian;

(4) compare P_iP_{i+1} with P_i'Q_j. If P_iP_{i+1}< P_i'Q_j, the current tracking point is P_{i+1}; otherwise current tracking point is Q_j; and execute step (1).

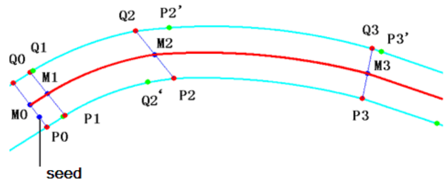


Fig. 4 The process and outcome of generating centerline via the shortest hypotenuse method

Figure 4 represents the progress of executing the above algorithm with a program. Here S denotes the seed point defined by the user, P_0 and Q_0 denotes starting points of tracking gained via searching method, and execution starts from the step (3) of the algorithm with P_0Q_0 as the first hypotenuse.

Comparison between figure 3 and figure 4 shows that the shortest hypotenuse method employs fewer vertices and produces a more accurate centerline for the same curved road. When figure 3 and figure 4 have the same contours, figure 4 requires only half vertices of figure 3, and it also comes with shorter processing time and improved efficiency.

3. Intersection identification

It is important and difficult to identify intersections in centerline generation. Figuring out intersections in tracking is quite critical, and it is pertinent to the accuracy of other centerlines of interactions and the automation of centerline generation algorithm.

Generally, two side contours of a road are almost parallel. It is generally considered there will be an intersection in the way ahead if two side contours have great angle difference. According to the shortest hypotenuse algorithm, assume that the leg crossing with two current contours is P_iP_i', the line segment to be tracked is P_iP_{i+1} and $P_i'Q_j$, and the threshold of the included angle between two contours is θ_0, then the existence of an intersection can be confirmed if θ, the included angle between P_iP_{i+1} and $P_i'Q_j$, is bigger than threshold θ_0.

An intersection point is the ending point of the centerline currently being tracked and also the starting point of the next centerline. Only intersection point identification can ensure the tracking algorithm transit well in the intersection area and start tracking the next road section.

Based on analysis of previous research results, this study proposes a polygon positioning algorithm for intersection points, i.e. identifying an intersection point through constructing a polygon. First of all, we should establish a relationship table of contours and intersection points, and its storage is shown as table 1.

Table 1 Relationship table of contours and intersection points

Contour 1	Contour 2	Point A	Point B	Intersection points
Contour 3	Contour 4	Point C	Point D	Intersection points
…	…	…	…	…

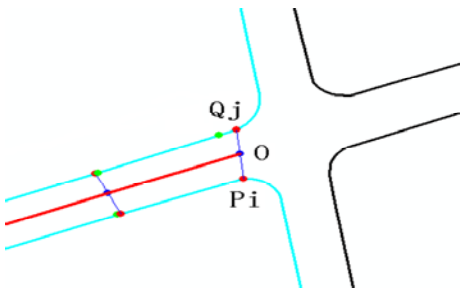
Point A: The inflection point on contour 1 where exists and entries are formed with contour 2；Point B: The inflection point on contour 2 where exists and entries are formed with contour 1

The algorithm is as the follows:

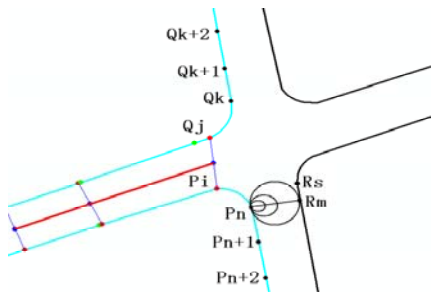
(1) if current tracking algorithm tells the existence of an intersection when reaching P_iQ_j, then the tracking stops and centerline ends at point O (as shown in figure 5(a)). Search for any matched record in the relationship table of intersection points and contours. If a record is found, execute step (2); if no record is found, then obtain the interception point which should be considered as the end of current centerline, finish centerline tracking, delete the record in the table, and execute step (10);

(2) assume that the angel difference threshold of contour at one side is β_0 and the longest distance tracked by intersection inflection point is L_0. Select inflection point to be tracked in contour P or Q, take contour P as an example, add Q_j into the inflection point queue and calculate the tracking distance. Consider θ_1 as the included angle between P_nP_{n+1} and $P_{n+1}P_{n+2}$, if $\theta_1>\beta_0$ and $L<L_0$, then n=n+1 and continue the tracking; if $\theta_1>\beta_0$ and $L\geqslant L_0$, then stop the tracking, take P_n+L as intersection inflection point when tracking reaches L0 , and execute step (3); if $\theta_1\leqslant\beta_0$, then stop the tracking, take Pn as inflection point and execute step (3);

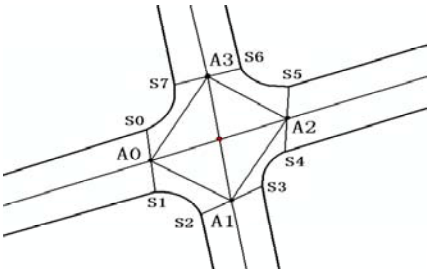
(3) if tracking along contour P, then compare Pn with the head point of inflection point queue. If different, add Pn into the queue, otherwise, execute step (9). If tracking along contour Q, then compare Q_k with the head point of inflection point queue. If different, then add Q_k into the queue, otherwise, execute step (9) as shown in figure 5 (b); (a) Recognizing the existence of an intersection(b) Searching for the inflection point of the other side with circular search method(c) Constructing a polygon in the intersection



(a) Recognizing the existence of an intersection



(b) Searching for the inflection point of the other side



(c) Constructing a polygon in the intersection

Fig. 5 Intersection point identification with polygon construction method.

(4) assume tracking along contour P (tracking along contour Q in the same way) and the inflection point of contour P is Pn, then the tracking direction is Pn->Pn+1, draw a search circle inside the contour with the line segment perpendicular to P_nP_{n+1} as the diameter, Pn as point of contact and P_nP_{n+1} as tangent line, and the circle crosses with R_m in contour R (as shown in figure 5(b));

(5) if $P_{n+1}R_{m+1}$ crosses with P_nR_m, it indicates that current contour R to be tracked is in opposite direction against current tracking, and the tracking should continue; otherwise, turn over the current contour R to be tracked;

(6) obtain the included angle θ between R_mR_{m-1} 和 R_mR_{m+1}. If $\theta > \beta 0$, execute step (7); otherwise execute step (8);

(7) based on the gradient angle between adjacent line segments, obtain inflection point Rs via opposite tracking from Rm in contour R in a way similar to step (2) , add contour P and R and intersection inflection point Pn and Rs into the relationship table of intersection points and contours. If R_s is different from the head of inflection point queue, add R_s into the queue, otherwise execute step (9); obtain inflection point R_t via forward tracking from Rm in step (2), and execute step (3);

(8) let m=m+1, continue forward tracking along the current contour till θ>β0, add contour P and R and intersection inflection point Pn and Rm into the relationship table of intersection points and contours, if Rs is different from the head point of inflection point queue, then add Rs into the queue, otherwise execute step (9); let m=m+1, continue forward tracking along the current contour till $\theta<\beta_0$ to obtain intersection inflection point Rm, execute step (3);

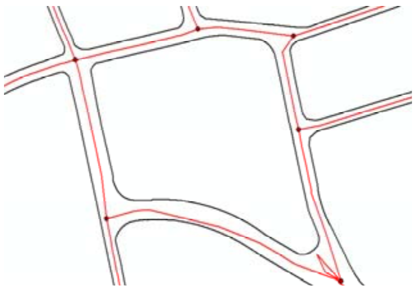
(9) construct line segments S_0S_1, S_2S_3, ••• , S_iS_{i+1}, $S_{i+2}S_{i+3}$, •••by connecting the vertices of the intersection inflection point queue, obtain a midpoint queue of the above line segments A_0, A_1, ••• , A_n, connect the midpoints to construct a intersection polygon as shown in figure 5(c). The center of the polygon is the intersection point, take the intersection point as the temporary end of centerline of step (1) and finish the centerline tracking; enter the intersection point into relevant records in the relationship table of intersection points and contours;

(10) select one record of current intersection point in the relationship table of intersection points and contours, and start tracking the next centerline.

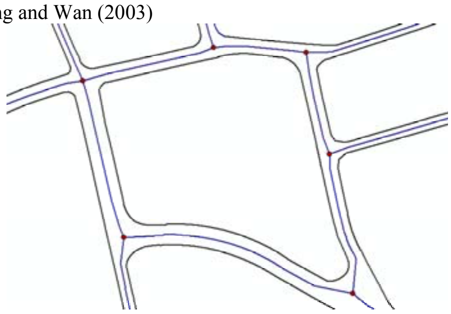
4. Experment and Result

Assume that the road network has n intersections and m sections. The above algorithm is generating centerlines and intersection points via contour tracking. It is obvious that the temporal complexity of the algorithm is m+n. In addition, the tracking also stores relevant intersection information in the table of relationship which can store no more than (k-2)n intersections (k is the mean of network exits and entries), so the spatial complexity of the algorithm is (k-2)n. Therefore, the centerline generation algorithm is linear in terms of both temporal and spatial complexity.

With a fraction of road contours of a city's survey map as the study subject, figure 6 is centerlines generated with the algorithm proposed by Zhu, Wang and Wan (2003) and the algorithm newly proposed in this study.



(a) Centerlines generated with the algorithm proposed by Zhu, Wang and Wan (2003)



(b) Centerlines generated with the algorithm proposed by this study

Fig6. Centerlines generated by two algorithms (partial)

Table 2.2 Results of two algorithms

Item	Zhu 's algorithm of (2003)	the algorithm of this study
Actual Number of sections	324	324
Actual Number of intersections	228	228
Number of sections obtained	328	326
Number of intersections obtained	232	230
Number of intersection errors	4	2
Number of section errors	97	28
Number of intersection errors	39	11
Time consumed (s)	577	469

5 Conclusion

The objective of this study is to propose an automatic centerline generation algorithm based on contours. With reference to previous studies, this study proposes the shortest hypotenuse-based centerline generation algorithm. It constructs right triangles with the key points of contours and connects the midpoints of hypotenuses to form centerlines. Based on the centerline generation algorithm, this study also proposes an intersection point identification algorithm via polygon construction. Experimental findings prove the algorithm is quite efficient and accurate. Centerline generation finish first step of spatial information processing for urban transportation network.

References

[1] Barzohar M, Cooper D B. Automatic finding of main roads in aerial images by using geometric-stochastic models and estimation [J].IEEE Transactions on Pattern Analysis and Machine Intelligence, 1996, 18(7): 707-721

[2] Laptev I, Mayer H, Linderberg T,et al.Automatic extraction of roads from aerial images based on scale space and snakes[J].Machine Vision and Applications, 2000, 12(1): 23-31

[3] Hu X, Tao C V.Automatic highway extraction from high resolution imagery by an energy minimizing based perceptual grouping method [J].Geomatica, 2004, 58(1): 41-50

[4] Kim T, Park S-R, Kim M-G,et al.Tracking road centerlines from high resolution remote sensing images by least squares correlation matching [J].Photogrammetric Engineering and Remote Sensing, 2004, 70(12): 1417-1422

[5] Christensen, A.H. Street Centerlines by A Fully Automated Medial – Axis Transformations, in Proceedings of GIS / LIS'96, Denver, November 19-21, 1996.p.107-116.

[6] Felkel, P., Obdrzalek, S. Straight Skeleton Implementation, Proceedings of Spring Conference on Computer Graphics, Budmerice, Slovakia, 1998,p210-218.

[7] McAllister, M.ve Snoeyink, J. Medial Axis Generalization of River Networks, Cartography and Geographic Information Science, 2000, v.27,n.2, p.129-138.

[8] Thomas, F. Generating Street Center-Lines from Inaccurate Vector City Maps, Cartography and Geographic Information Systems, 1998.v.25,n.4, p.221-230.

[9] Tang Zesheng, Chen Li, Deng Junhui. Visualization of 3D data sets [M]. Beijing: Tsinghua University Press, 1999.

[10] ZHU Zhuang-sheng,WANG Qing,WAN De-jun. A Study on Automatic Generation Road Center-lines Algorithm Based on Road Contour Journal of Image and Graphics[J]: A, 2007, 7(8): 792-797