AN ALGORITHM FOR GENERATING ROAD CENTER-LINES FROM ROAD RIGHTS-OF-WAY

Naven E. Olson
Digital Mapping Services
Computer Services Division
University of South Carolina
Columbia, SC 29208

ABSTRACT

A totally vector geometric algorithm was developed for generating road center-lines from road rights-of-way The algorithm finds pairs of parallel arc sections within a given distance. The user passes the name of the right-of-way layer, the name of the new center-line layer, a road width, and an expanded multiplier. Pre-processing is done on a new copy of the right-of-way layer linework The main algorithm then creates a table of segments perpendicular to right-of-way arcs. This table is used to generate a table of intersection points which can be sorted and used to generate the basic linework for center-lines Post-processing then joins nodes and center-lines to create road topology and windows out any "center-lines" outside of rights-of-way. The algorithm was developed as part of Digital Mapping Services' in-house mapping system and tested on City of Columbia right-of-way data. The source code is copyrighted by the University of South Carolina.

INTRODUCTION

One problem that is often encountered in computerized mapping is that of generating road center-lines from road right-of-way data. For instance it may be desired to generate center-lines from right-of-way and parcel data generated from a tax mapping program.

Traditional approaches have been of three types:

a) standard manual digitizing;

b) center-lining in a CAD or COGO environment;

c) vector to raster conversion, followed by skeletonization, followed by raster to vector conversion.

This paper sets forth a new, strictly vector, approach which does an automatic generation based on fundamental geometric and trigonometric analysis.

LITERATURE REVIEW

Various skeletonization and other algorithms related to this approach have been developed in the past. Blum(1967) first proposed "medial axes" or skeletons for regions. Raster algorithms for medial axis skeletons have been developed by Rosenfeld and Pfaltz(1967) and Montonari(1969). Brassel and Heller(1984) propose a vector algorithm for bisector skeletons. This could be used to develop "line segment proximity polygons" dividing a region into polygons closest to polygon line segments and having a rough equivalence to Thiessen polygons. It does not, however, produce a line network which would expand to the right-of-way area upon appropriate buffering.

Brassel (1985) has listed area-to-line conversion as a desirable map generalization operator. McMaster (l991) lists Beard (1987), DeLucia and Black (1987), Nickerson and Freeman (1987), and McMaster and Monmonier (1989) as others including this function in their "wish list".

Brassel and Weibel(1988) advocate "processing based on understanding". Mark (1989:76) states that "in order to generalize a cartographic line, one must take into account the geometric nature of the real-world phenomenon which that cartographic line represents." Weibel quotes these and other authors in advocating comprehension of underlying phenomena.

This paper presents an approach to area-to-line collapse suitable for road, railroad, and power-line rights-of-way. It could perhaps be used as a starting point for algorithms for stream collapse or even interpolation between contour intervals.

TRADITIONAL APPROACHES

The problem of generating road center-lines from road rights-of-way has traditionally been handled by standard manual digitizing, center-lining in a CAD or COGO environment, or a rasterization-skeletonization-revectorization approach. All of these approaches, as used in current state-of-the-art practice, have severe weaknesses

Standard manual digitizing is labor-intensive. Due to the generally small distance between road sides there may be significant variation from "parallelism" of the center-line to the rights-of-way.

Center-lining, as in the center-line option under the add arc command in arc edit using ARC/INFO, generally eliminates this non-parallelism problem and will produce a good center-line if used properly It is also less tedious than table digitizing However, it is still labor intensive and will produce errors if bad choices in picking right-of-way points are made and not corrected.

The rasterize-skeletonize-revectorize approach is the most automated of the three. This approach is, however, troubled by several artifact problems. Some well known raster-to-vector artifact problems are aliasing, dimpling, and spurs. (Gao and Minami (1993) and Zhan (1993)) Artifacts common in this rasterize-skeletonize-revectorize approach include incorrect handling of off-center intersections and the gaps illustrated in Figures la and lb.

Figure la.

Figure lb.

NEW ALGORITHM

The fundamental approach to the center-line problem involves making a copy of the right-of-way layer and pre-processing the copy, generating the basic line work, and post-processing to close intersections and create topology and window out any "centerlines" that are not inside any right-of-way.

Pre-processing

Step 1. Angle correction. Correct angles on short arcs, as defined in the later section on Arc classification, which resemble Figure 2a so that they resemble Figure 2b. This will prevent good line work from being lost during Intersection points table correction.

Figure 2a.

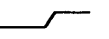

Figure 2b.

Step 2. Arc extension. Extend arcs ending at dangles two standard buffer widths past the dangle node.

Step 3 Arc classification. Classify arcs The original arcs are given attributes of "< *arc number*>,0" if they are standard arcs and attributes of "<*arc number*>,-2" if they are short arcs (less than a buffer width in length) having a sum of angles at their ends greater than about 180 degrees. See Figures 3a and 3b.

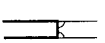

Figure 3a.

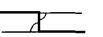

Figure 3b.

Extension arcs would be given attributes of "<*original arc number*>,- 1 " or "<*original arc number*>,1 " for arcs extended from the start (-1) or end (1) of original arc <*original arc number*>.

Main line work generation.

Step 4. Perpendicular segments table generation Generate a table of segments perpendicular to all standard arcs at all discrete points on the arcs and perpendicular to all extension arcs at their extended ends. Perpendicular directions are generated by a process of numeric differentiation The table would consist of the following properties:

a) arc or original arc number;

b) distance along arc from beginning of arc (negative for "<*original arc number*>,-1 " arcs);

c) x- and y-coordinates of center-point on arc;

d) x- and y coordinates of both endpoints, each endpoint being located a standard buffer length times an expansion multiplier away from the center-point along the perpendicular line.

Step 5. Spatial index generation. Generate a spatial index of this table of perpendicular segments.

Step 6. Intersection points table generation. Intersect these perpendicular segments with the right-of-way arcs to form a table (File 6.1) of intersection points calculated by averaging the coordinates of the center point of the perpendicular segment and the coordinates of the point of intersection of the perpendicular segment with the other arc. Each point has the following properties:

a) x- and y-coordinates;

b) arc number of arc the perpendicular segment was generated from;

c) arc number of arc intersected by the perpendicular segment;

d) distance from the beginning of the arc to the perpendicular segment.

An additional table (File 6 2) is also created for the intersection points containing the following properties:

a) the intersection number of the intersection;

b) the segment number of the perpendicular segment;

c) the x- and y-coordinates.

When the tables are generated, intersections that are not close to perpendicular are flagged as invalid. This prevents spurious line work from being created in later steps.

Step 7. Intersection points table correction. Eliminate intersections which are not the first intersection on the ray going out from the center point of the perpendicular segment

that they are located on These intersections can cause spurious linework as shown in Figure 4.

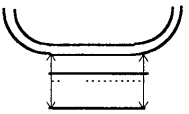

Figure 4.

Sort File 6.2 by perpendicular segment number. Mark each intersection that has any intersections on the same segment between it and the center point of the segment as invalid. Reorganize File 6.1 (the intersections point file) deleting invalid intersection points.

It should be noted that points flagged as invalid because they were generated by non-perpendicular intersections are written out in Step 6 and deleted in Step 7 because such points are sometimes the first intersections on a perpendicular segment. It should also be noted that this step makes Angle correction necessary.

Step 8. Intersection points table sort. Sort File 6.1 (the intersection points file) with field 6.1.b) being the primary key, field 6.1.c) the secondary key, and field 6.1.d) the tertiary key. Sort in ascending order on all fields.

Step 9. Auxiliary intersection points table generation. This is a key step and prevents the loss of a high percentage of the linework.

Generate intersection records with 6.1.b) and 6.1.c) transposed for arcs where the duplicate with lower primary key does not exist or has only one point. Also generate intersection records with 6.1.b) and 6.1.c) transposed for arcs where only one point exists in each duplicate arc.

The first step is necessary because Passing of intersections files uses only intersection points with the primary key less than the secondary key to prevent duplicate arc problems. Due to anomalies some arcs will only appear as the duplicate arc with the primary key greater than the secondary key.

The second step is necessary because other anomalies will cause some arcs to appear as two one-point arcs. The two points will generally be close to the two ends of the arc. The distance of the point from the lower-numbered arc must be converted to the distance along the higher-numbered arc using the following procedure

a)Find the point that distance along the lower-numbered arc.

b)Use a perpendicular segment to find the corresponding point on the higher-numbered arc.

c)Calculate the distance along the higher-numbered arc of this point.

Step 10. Passing of intersections files. Pass the intersections files, starting a new arc each time either arc in the arc pair changes. Use only intersections where the primary key is less than the secondary key to eliminate duplicate arcs.

It should be noted that several indices are created in this program. It is necessary to know what the primary and secondary right-of-way arcs are for each center-line generated. There must be tables of corresponding primary right-of-way arcs, secondary right-of-way arcs, and associated center-lines. One of these tables is sorted by primary arc number, and the other is sorted by secondary arc number.

Post-processing.

This stage proved harder to develop than any other stage of the process. The first algorithm, Ring generation, proved especially difficult. Several tables were used and a great deal of complex indexing was involved.

It should be noted at this time that pairs of right-of-way arcs which surround a centerline do not necessarily have the same directionality. Failure to account for this fact can cause major problems in Ring generation, Tee solution, and Auxiliary two-arc-ring generation.

Step 11. Ring generation. Form "rings" around open intersections as in the hypothetical example in Figure 5.

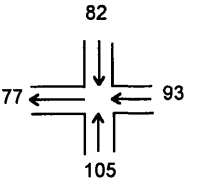

Figure 5.

Here the "ring" has four ends in this order: 77,-82,-93,-105. The procedure for generating a ring was as follows·

a) Pick a one-arc node.

b) Note which two right-of-way arcs were used to generate the arc.

c) Pick one of these right-of-way arcs for the target arc, the finding of which will close the ring.

d) Note which right of-way arc meets the other right-of-way arc at the appropriate end

e) Find which center-line and opposite right-of-way arc pair with this right-of-way arc.

f) Continue this process until the "ring" is closed by reaching the target arc or it is clear the "ring" cannot be closed.

False intersections are a significant problem in Ring generation. Two helpful checks for false intersections are the previous endpoint test (i) and the on-extensions test (ii).

(i). Take the endpoint of the proposed center-line. Find the point on the known right-of-way corresponding to this end point. Use this point to find the point on the proposed right-of-way arc corresponding to the endpoint. Construct a line segment starting at the corresponding point on the known right-of-way and perpendicular to the segment connecting it to the corresponding point on the proposed right-of-way. Intersect this newly constructed segment with the end segment of the right-of-way section which connected to the known right-of-way going backward on the ring. If the intersection is too far from the actual node of the known right-of-way arc, the pairing is invalid.

(ii). Take the intersection of the two end segments of the center-lines. Check to be sure it is on both rays extending outward from the relevant end points. If it is not, the pairing is invalid. This test is especially helpful in "downtown" type neighborhoods where most blocks are square in preventing loss of line work resulting from invalid arcs inside blocks, which will later be deleted, being incorrectly chosen for pairing.

Step 12. Tee solution. Find missed intersections at "T" junctions where there is no pseudo-node at the "top" of the "T" For an example, see Figure 6.

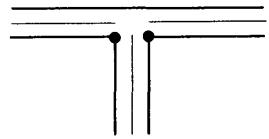

Figure 6.

The procedure for "solving" a "tee" is as follows:

a) Pick a one-arc node.

b) Note which two right-of-way arcs were used to generate the arc.

c) Note which right-of-way arcs meet these right-of-way arcs at the appropriate ends.

d) If either of these arcs is non-existent or these arcs are identical, stop.

e) Find a right-of-way arc which pairs with both of these arcs and the corresponding center-line for each pair.

f) If the center-lines are identical, stop.

g) Check for "false tees" using the on-extensions test used for false intersections during Ring Generation. Both center-line pairs should be checked

h) If the "tee" passes both tests in g), write the "tee" to the "ring" table.

Step 13. Ring closure. Edit the arcs to close the "rings" generated in Step 11 and Step 12. The procedures for closing two-arc and three-arc "rings" are well understood. No attempt has yet been made to handle five-or-more-arc intersections, although the file structure allows up to seven arcs in a "ring".

Four-arc intersections were classed as one of two basic classes. Figure 7 shows several variants of the most common class. Figure 8 shows two variants of the rarer class.

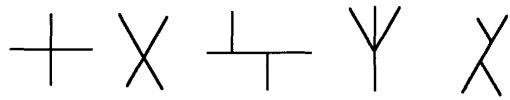

Figure 7.

Figure 8.

The rarer case is distinguished by the fact that the sum of the two largest opposite angles minus the sum of the two smallest opposite angles is greater than 1.5 radians and the difference between the two angles in the larger opposite angle pair is greater than 1.5 radians.

For the most common case the procedure was:

a) Intersect the two opposite angle pairs where the sum of the distances from the endpoints to the intersections is smallest.

b) If the difference between these two intersections is smaller than a closure tolerance, move both intersections to their average.

c) Move all "ring" nodes to the appropriate intersections.

d) If the intersections are not within a closure tolerance, add an arc between them.

For the rarer case the following procedure was used:

a) Intersect the arcs in the smaller angle of the large angle pair.

b) Intersect the arcs in the most perpendicular angle of the smaller angle pair to get the intersection for the largest angle.

c) If the intersections are within a closure tolerance, move both intersections to their average.

d) Move all "ring" nodes to the appropriate intersections.

e) If the intersections are not within a closure tolerance, add an arc connecting them.

Step 14. Auxiliary two-arc-ring generation. Generate two-arc rings to cover the cases in Figure 9a and Figure 9b. The case shown in Figure 9a is more common and is basically due to minor mismatch between pseudo-nodes on matching right-of-way arc pairs. The problem in figure 9b arises in situations where a pseudo-node occurs on one side of a road in the right-of-way line work but not on the other. Both of these cases occur at what should be pseudo-nodes in the center-lines and cause minor errors.

Figure 9a.

Figure 9b.

Step 15.Auxiliary two-arc-ring closure. Edit the arcs to close the "rings" generated in Step 14.

Step 16. Overlay and windowing. Intersect these center-line arcs with a right-of-way polygon layer and window out "center-lines" which are not inside any right-of-way

CONCLUSIONS

The programs tested well. The great majority of line work was captured. Artifacts were essentially non-existent. Figures 14 through 17 show some rights-of-way and center-line output.

The chief problem not solved at this point is the anomaly illustrated in Figure 10. This results in duplicate one-point arcs which have ends corresponding to the same endpoint.

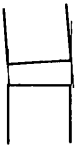

Figure 10.

Figure 11.

Compare Figures 10 and 11 with Figures 12 and 13. In these figures translation error, rather than rotation error occurs. Intersections with perpendicular segments occur within the paired arcs and no linework is lost. The translation errors in Figures 12 and 13 and the rotation errors in Figures 10 and 11 are greatly exaggerated to illustrate the point. These errors are not normally visible to the naked eye. In real linework, translation error is greater than rotation error in the great majority of cases. In the rare cases where rotation error is greater than translation error, perpendicular segment intersections are outside

paired rights-of-way and linework is lost. Such cases occur most commonly in "downtown" type neighborhoods.

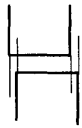

Figure 12.

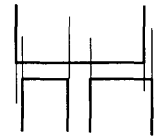

Figure 13.

Work on this anomaly and on the related anomaly shown in Figure 11 is planned This will involve modification of the Auxiliary intersection points table generation step. It is expected that it will shortly be completed.

Other problems observed were:

a) Intersections failed to close in a minority of cases.

b) Problems occurred where paired right-of-way sections deviated significantly from parallel.

c) Problems sometimes occurred at map edges.

d) No provision has yet been made for five-or-more-arc intersections, cul-de-sacs, or filleted intersections.

Future plans include dealing with these remaining problems, including an error report as a final step, and examining what related problems can be treated using similar approaches.

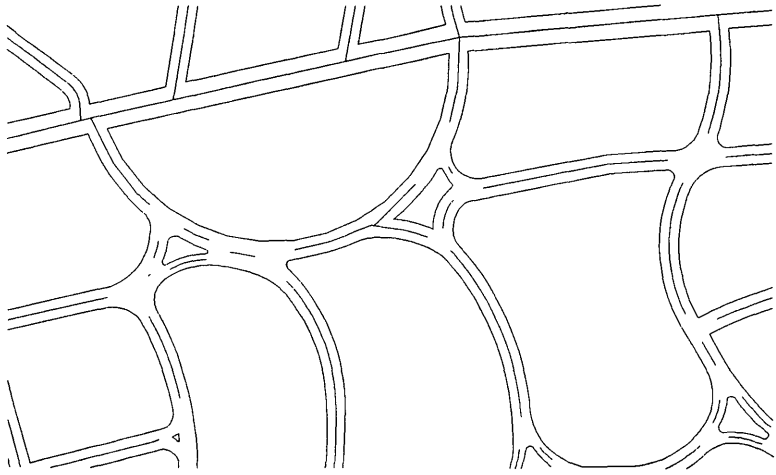

Figure 14.

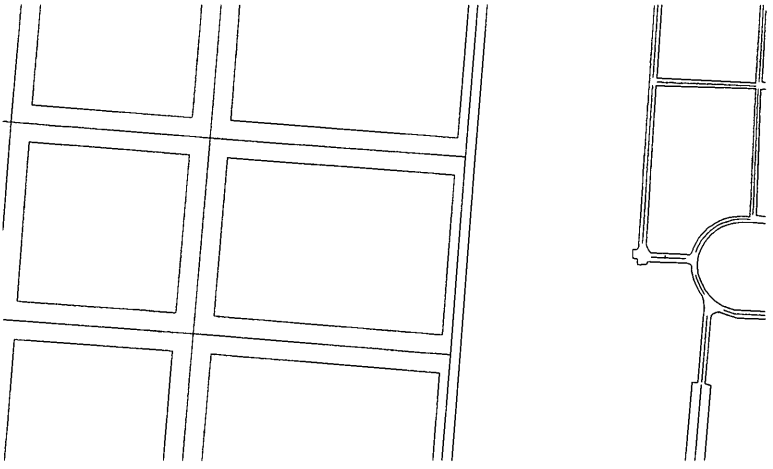

Figure 15.

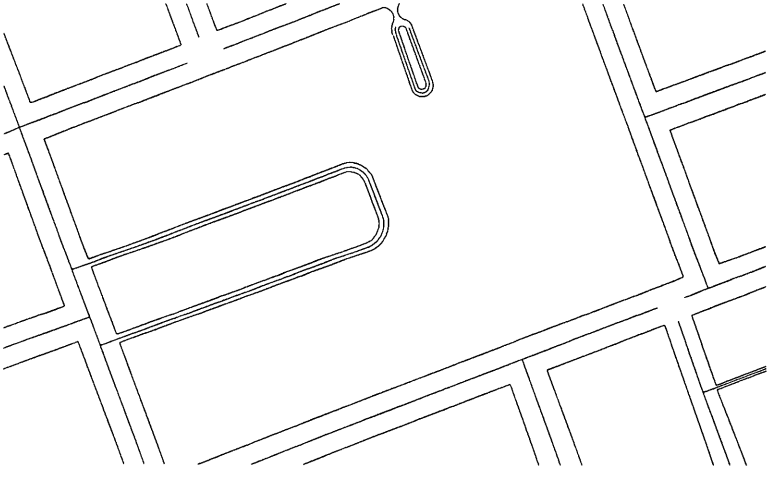

Figure 16.

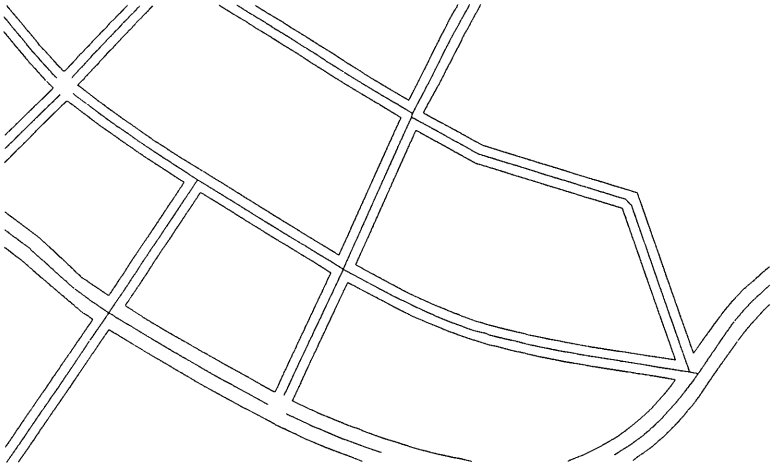

Figure 17.

ACKNOWLEDGMENTS

Acknowledgments are extended to Dr. David J. Cowen of the University of South Carolina Humanities and Social Sciences Computing Lab and his staff who were very helpful in preparing and reviewing this paper. Acknowledgments are extended to my boss, Stan Lawrimore, who encouraged me to write this paper and helped with final obstacles. I would also like to thank Paul Smith, who was very helpful in obtaining clearance, and David Smith I must also acknowledge the City of Columbia, South Carolina which made available the right-of-way data which were used for testing purposes. And I would extend very grateful acknowledgments to Carole Shealy, without whose help this paper could not have been recovered after I lost my disk.

REFERENCES

Beard, M. Kate 1987, "How to Survive on a Single Detailed Database", *Proceedings Auto-Carto 8,* Baltimore, pp. 211 -220.

Blum, H. 1967, "A Transformation for Extracting New Descriptors of Shape", *Symposium on Models for the Perception of Speech and Visual Form,* Weiant Whaten-Dunn (ed.), MIT Press, Cambridge, Mass., pp. 362-380.

Brassel, Kurt E. 1985, "Strategies and Data Models for Computer-Aided Generalization", *International Yearbook of Cartography,* 25, pp. 11-29

Brassel, Kurt E., and Heller, Martin 1984, "The Construction of Bisector Skeletons for Polygonal Networks", *Proceedings, First International Symposium on Spatial Data Handling,* Zurich, pp. 117- 126.

Brassel, Kurt E., and Weibel, Robert 1988, "A Review of and Conceptual Framework of Automated Map Generalization", *International Journal of Geographical Information 5ystems,* 2/3, pp. 229-244.

DeLucia, A., and Black, T. 1987, "A Comprehensive Approach to Automatic Feature Generalization", *Proceedings, 13th International Cartographic Association Conference,* Morelia, Mexico, 4, pp. 168-191.

Gao, Peng and Minami, Michael M 1993, "Raster-to-Vector Conversion: A Trend Line Intersection Approach To Junction Enhancement", *Proceedings Auto-Carto 11,* Minneapolis, pp. 297-303.

McMaster, Robert B. 1991, "Conceptual Frameworks for Geographical Knowledge", In Buttenfield, B. P., and McMaster, R. B.(eds.), *Map Generalization: Making Rules for Knowledge Representation,* Longman, London, pp. 21-39.

McMaster, R. B., and, Monmonier, M. S. 1989, "A Conceptual Framework for Quantitative and Qualitative Raster-Mode Generalization", *Proceedings GIS/LIS '89,* Orlando, Florida, 2, pp. 390-403.

Mark, D. M. 1989, "Conceptual Basis for Geographic Line Generalization", *Proceedings Auto-Carto 9,* Baltimore, pp. 68-77.

Montonari, U., "Continuous Skeletons from Digitized Images", *Journal of the Association for Computing Machinery,* 15/4, pp. 534-549.

Nickerson, B G., and Freeman, H. 1986, "Development of a Rule-Based System for Automatic Map Generalization", *Proceedings, Second International Symposium on Spatial Data Handling,* Seattle, pp. 537-556.

Rosenfeld, A., and Pfaltz, J.L. 1966, "Sequential Operations in Digital Picture Processing", *Journal of the Association for Computing Machinery,* 13/4, pp 471-494.

Zhan, Cixiang, 1993, "A Hybrid Line Thinning Approach", *Proceedings Auto-Carto 11,* Minneapolis, pp. 396-405