Generating Topologically Correct Schematic Maps

Silvania Avelar, Matthias Müller
Institute of Theoretical Computer Science
Swiss Federal Institute of Technology
8092 Zurich, Switzerland
{avelar,muellerm}@inf.ethz.ch

Abstract

We create schematic maps from traditional vector-based, cartographic information. The lines of the original map are modified to meet geometric and aesthetic criteria in the resulting schematic map. However, in order to preserve topology after the transformations have been applied, usually topological information must be computed and consulted during processing. The known, existing methods for preserving topology during map transformation generally involve computing several constrained Delaunay Triangulations [Gol94, JBW95, Rua95, Har98]. This paper shows how to compute a cartographic schematic map, while preserving topology using simple geometric operations.

Keywords. Schematic maps, digital cartography, topology, computational geometry

1 Introduction

Schematic maps are designed to convey only essential features of network routes. They indicate important topological information on transportation or utility maps such as connectivity and stops, while preserving only a general sense of the original geometry. The length and shape of routes in schematic maps need not be faithful to reality, as long as essential topological information is preserved. This cartographic flexibility is possible because geometric accuracy is considered to be less important than linkages, adjacency, and relative position [Mon96, Bra98]. In addition, such maps do not have the mass of details usually presented in conventional topographic maps and thus present the essential information clearly. Figure 1 and Figure 2 illustrates the topological nature of schematic maps.

Generating schematic maps *on demand* can be seen as a geometric constraint problem that should be solved algorithmically, in interactive time. In general, to create schematic maps automatically, first geometric line detail should be removed via smoothing or filtering. Then the new, straightened line network should be adjusted, to improve the legibility of important information, following some common sense geometric and aesthetic criteria. We propose a technique to preserve the topology of the map during this transformation process. To the best of our knowledge, there does not seem to be a public, generally acknowledged solution for preserving topology while moving points, such as we do to generate topologically correct schematic maps automatically.

Some recent suggestions and theory concerning the automation of schematic maps can be found in the literature. Morrison [Mor96] proposes rules to govern the schematic mapping

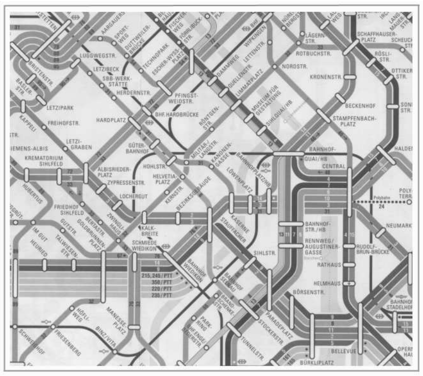

Figure 1: Example of a public transportation map of Zurich, Switzerland.

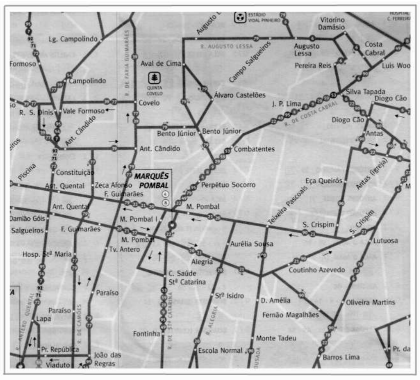

Figure 2: Example of a bus map of Porto, Portugal.

method based on characteristics of transport services. Gordon Kennedy in [GK99] presents cartographic design problems with mapping route data.

2 Characteristics of schematic maps

Routes on schematic maps are drawn as generally straight lines with cartographic microcrenulations removed, and vary in direction only via gentle, stylized angles. Changes at fixed angles of 0, 45 and 90 degrees or 0, 30, 60 and 90 degrees are common, though some schematic maps have merely simplified lines with arbitrary, but few angles. Network lines with overlapping routes are separated by a minimum legibility distance. This distance can either be zero or a constant chosen for the map. Sometimes adjacent schematized lines also have smooth, artistic, circular arcs around bends to preserve the graphic proximity distance for the greatest length possible (see Figure 1, top left corner where the lines curve in between the two white cylinders).

We also observe that the straight network lines are generally not too long. To provide a better visualization of the routes by adding a sense of the original geometry, a small number of breaks or changes in line direction are added. In some schematic maps, all lines are presented in one color and the route of a particular network can be traced by number only. In other maps, contrasting colors differentiate the various transportation lines.

It is easy to find differences in style, where differing geometric and aesthetic criteria were used to design the schematic map (e.g. compare Figure 1 and Figure 2), but all seem to share the need for graphic simplicity, while retaining network information content and presentation legibility.

3 Generating schematic maps

The purposeful deformation of a geographic map is a common operation in cartography. Two types of map deformations can be considered [DEG98]: one for bringing two maps together, such as combining a satellite image and a geographic map via rubber-sheeting or minor geometric adjustment; and the other, to highlight quantities other than geographic distance and area, such as in cartograms. Our schematic maps can be thought of as the first type of deformation, but we use techniques for transforming lines similar to [HK98, Dor96] for constructing continuous area cartograms.

The difficult aspect of generating schematic maps is to reposition the network routes of the original map to locations which fulfill the set of criteria described in the previous section. For this step, we propose a solution, described in the next section.

To begin the automatic generation of our schematic maps, we assume the database selection has already occurred and the network lines to schematize are used as input to our algorithm. We have chosen to create straight lines which can be horizontal, vertical, or at 45 degrees turns. For readability and aesthetic considerations, we have set a user-defined minimum distance between the lines and a maximum length for a straight line. The cartographic data structure we use contains nodes and polylines.

As a preprocessing step to remove line crenulations, we perform point and line filtering (simplification) to reduce the number of visualization points that constitute the detailed map. For this study, we used the Douglas–Peucker algorithm [DP73]. This algorithm is easy to implement and flexible, with a user definable threshold that controls the amount of simplification. However, there could be some topological and self-intersection errors introduced by using this algorithm. Variants of the Douglas–Peucker algorithm which take topology into account [Saa99] could be tried instead.

The next step is to find a new location for each point of each pre-simplified line according to the other criteria constraints of the schematic map.

For the constraint of stylized angles, we use a dynamic iterative algorithm, similar to those found in [Mue99, HK98]. Using this algorithm, we calculate the distance to move the points of a line segment for each one of the allowed locations. An allowed location is one that fits the line in one of the allowed schematic directions. The nearest distance $dist$ determines the new schematic direction for the line segment (see pq in Figure 3a). This calculation is also repeated for the other line segments in which the point p exists. The new location p' for a point p will be obtained by the arithmetic average of the required distances to move the point for all of its line segments (Figure 3b).

We then check if the newly found location fulfills the other constraints of the map, performing adjustments when necessary. The line segments may be stretched or shortened to reach the new locations. However, the topology of the map has to be preserved during this modification process.

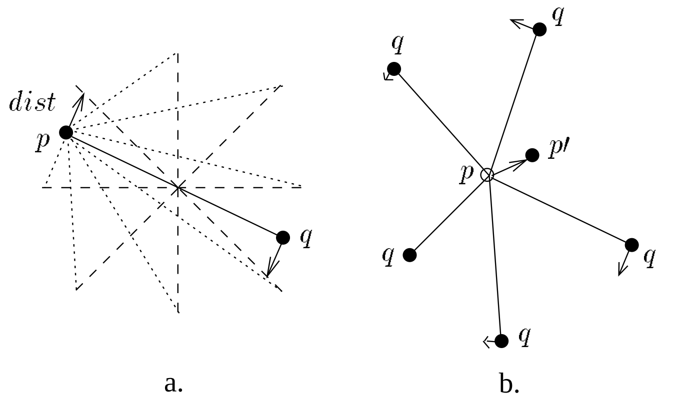

Figure 3: (a) Fitting a line segment pq to the nearest schematic direction and (b) finding the new location $p\prime$ for p.

4 Preserving the map topology

Before moving a point from its original position, p, to the new schematic location, $p\prime$, we perform a test to detect situations that can lead to a change in the map topology. For it we first create a triangle. The vertices of the triangle are p, $p\prime$, and the other endpoint of the line segment being analyzed, q [Ave97]. We have to find out if there is any line segment of the map crossed by the boundary edge $\overline{pp\prime}$ of the triangle $T(pp\prime q)$. We also have to check if the triangle T contains inside it any point.

If the topology would change, the move of p must be smaller than $p\prime$ to avoid this change. It is important whether a point moves at all. If a point moves, this will affect other points in the polyline and so the algorithm tries to find a better location for them. A better location is one where the polyline conforms to a solution that preserves the map constraints.

We distinguish the following cases to check the topology of a line segment pq:

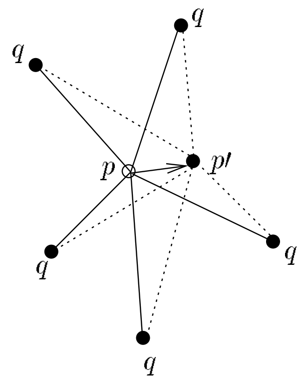

Figure 4: Point p can be moved to $p\prime$.

(1) If there is no point inside the triangle $T(pp\prime q)$ and no line segment crossing the edge $\overline{pp\prime}$ of T, the topology will be preserved, so the move of p to $p\prime$ is allowed (see Figure 4).

(2) If there is at least one line segment intersecting edge $\overline{pp'}$ of T, map topology will change, then a new location for p' has to be obtained. The new p' takes the nearest intersection point, say u, to p plus or minus a pre-defined distance d, which is an aesthetic constraint of the map (see Figure 5a).

(3) If there is at least one point v inside the triangle $T(pp'q)$, map topology will also change. To calculate the new location for p', we define a straight line l through v and q and calculate the intersection point u of l and the edge $\overline{pp'}$ of T. p' will be the nearest intersection point u to p plus or minus a pre-defined distance d (see Figure 5b).

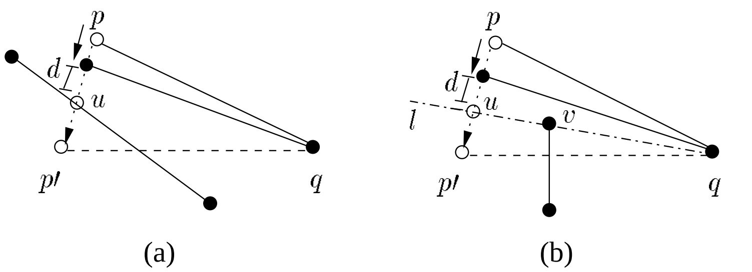

Figure 5: Topology checking.

The cases 2 and 3, as well more cases of each one, can occur together. In such case, the new location for p' will also be calculated taking the nearest intersection point to p: $p' = \min \operatorname{dist}(p, u) + d$.

We next present the algorithm as developed so far:

Procedure SchematicMap (map, maxSegment, tolerance);
Input: N points of the conventional map, the maximum line segment, the simplification tolerance
Output: n points for the schematic map
begin
    simplify lines with Douglas–Peucker algorithm using given tolerance;
    cut long line segments according to maxSegment;
    repeat
        for each point p of the map do
            calculate a new schematic location p' for it;
            if there is any segment collision with $\overline{pp'}$ then
                calculate new location for p';
            for each point q that makes a line segment with p do
                if there is any point inside triangle p,p',q then
                    calculate new location for p';
            end for;
            change location of p to p';
        end for;
    until reached desired appearance;
end;

5 Results and Conclusion

We have implemented on a Sun workstation in the Java programming language the main ideas of the schematic map generator with encouraging first results. In Figure 6, we show an example of the results obtained with our prototype using a database containing streets of Zurich. The topology of the resulting schematic network compared to the standard map is preserved. For this study, map labels were not included. We plan to extend the prototype to add cartographic features and labels to the visualization.

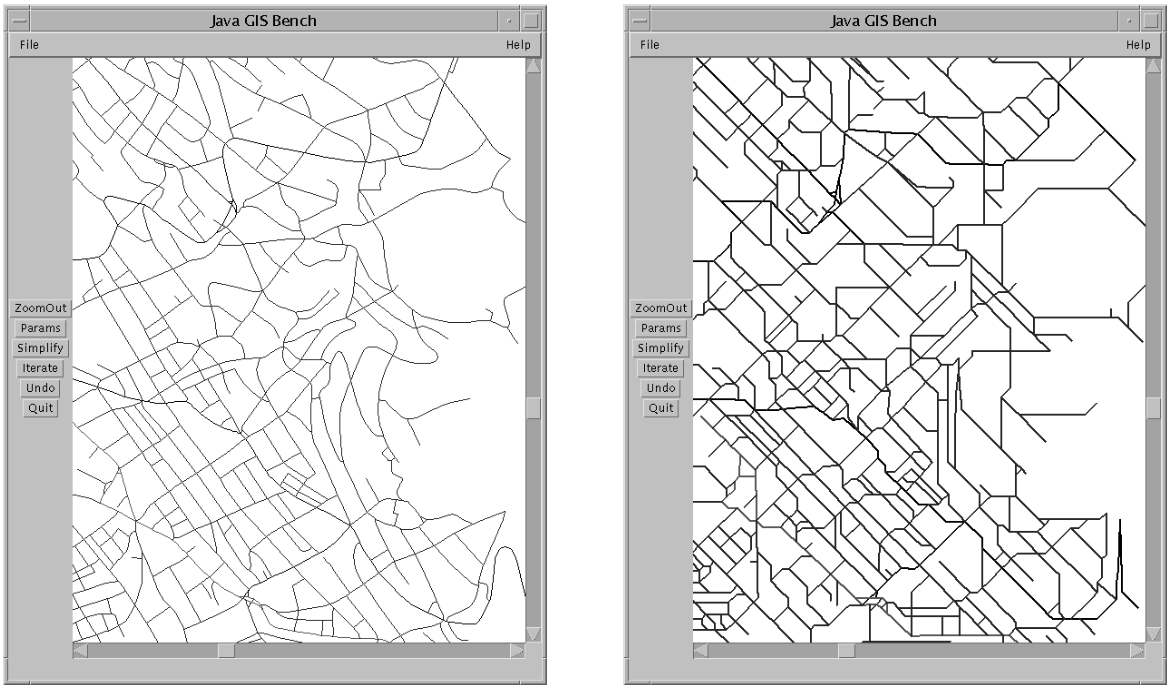

Figure 6: Example of streets in a standard map and a derived schematic map.

The Porto bus map in Figure 2 has less aesthetic treatment and relatively less geometric simplification than the public transportation map of Zurich in Figure 1. For this reason, we hoped to achieve, in early stages, a map of comparable quality to Figure 2. Preliminary inspection of our results indicates our schematic maps have a similar design appearance and feel to Figure 2.

We have presented a new method for preserving topology while generating schematic maps. We hope to continue this work by finding and implementing remaining cases of the described algorithm, and by studying and defining a better formulation of aesthetic constraints to generate automatic schematic maps.

Acknowledgments

The authors of this paper are grateful to Bernd Gärtner for discussions, and Ambros Marzetta for helpful comments. Special thanks go to Frank Brazile for proof-reading this paper and references.

References

[Ave97] Silvania Avelar. The problem of contour in the generation of digital topographic maps. In *Auto-Carto 13*, pages 397–403, Seattle, 1997. ACSM/ASPRS.

[Bra98] Francisco Brazile. A generalization machine design that incorporates quality assessment. In *8th Int. Symp. on Spatial Data Handling*, pages 349–360, Vancouver, 1998.

[DEG98] Tamal K. Dey, Herbert Edelsbrunner, and Sumanta Guha. Computational topology. In B. Chazelle, J.E. Goodman, and R. Pollack, editors, *Advances in Discrete and Computational Geometry*, Providence, 1998. AMS.

[Dor96] Daniel Dorling. Area cartograms: their use and creation. *Concepts and Techniques in Modern Geography (CATMOG)*, 59, 1996.

[DP73] David H. Douglas and Tom K. Peucker. Algorithms for the reduction of the number of points required to represent a line or its caricature. *The Canadian Cartographer*, 10(2):112–122, 1973.

[GK99] R. Gordon-Kennedy. Problems of cartographic design in geographic information systems for transportation. *Cartographic Perspectives*, 32:44–60, September 1999.

[Gol94] Christopher Gold. Three approaches to automated topology and how computational geometry helps. In T.C. Waugh and R. G. Healey, editors, *Advances in GIS Research I*, pages 145–158, London, 1994. Taylor and Francis.

[Har98] Lars Harrie. Generalisation methods for propagating updates between cartographic data sets. Master's thesis, Department of Surveying, Lund Institute of Technology, Lund University Sweden, 1998. Licentiate Thesis.

[HK98] Donald H. House and Christopher J. Kocmoud. Continuous cartogram construction. In *9th IEEE Visualization Conference*, pages 197–204, North Carolina, October 1998.

[JBW95] Christopher B. Jones, Geraint L. Bundy, and J. Mark Ware. Map generalization with a triangulated data structure. *Cartography and Geographic Information Systems*, 22(4):317–331, 1995.

[Mon96] Mark Monmonier. *How to lie with Maps.* The University of Chicago Press, 1996.

[Mor96] Alastair Morrison. Public transport maps in western european cities. *The Cartographic Journal*, 33(2):93–110, September 1996.

[Mue99] Matthias Mueller. *The Structure of Dense Polymer Systems, Geometry, Algorithm and Software.* PhD thesis, Department of Computer Science, ETH–Zurich, 1999.

[Rua95] Anne Ruas. Multiple paradigms for automating map generalization: Geometry, topology, hierarchical partitioning and local triangulation. In *Auto-Carto 12*, pages 69–78, Charlotte, 1995. ACSM/ASPRS.

[Saa99] Alan Saalfeld. Topologically consistent line simplification with the douglas-peucker algorithm. *Cartography and Geographic Information Science*, 26(1), 1999.