An Implementation of Street Selection with Self-Organizing Maps

Gail Carmichael 100327911

Abstract

In today's digital world, maps are viewed on the computer at widely varying resolutions all the time. Therefore, a mechanism to select features from the model behind the map is required in order to display the most important items at a given resolution. Categorizing roads using self-organizing maps is a good way to accomplish this. In this paper, a Java implementation of this is described.

1 Introduction

Maps have become an essential part of many of our lives, helping us to get from one place to another for centuries. With the widespread availability of relatively recent digital technologies, such as Google Maps[1], the need to display maps at different resolutions has never been greater.

Generally speaking, when changing the scale of a map, the level of detail displayed changes as well. When zooming out, for example, it is often not desirable or even possible to display all features of a map in the pixel space available for rendering. Therefore, some mechanism for choosing a set of features to display is required.

Figure 1: An example of three different resolutions of a map from Google Maps Canada.

To illustrate this, Figure 1 shows an example of three different map resolutions. In the first map, only water features and political boundaries are visible. As it is zoomed in, major roads and highways begin to appear, and large cities are marked. In the last map, the resolution allows many more roads and cities to be displayed, and bodies of water are more detailed.

While the features shown on a map can vary greatly, as was seen in the previous example, this paper will concentrate on only those found in basic street networks.

[1] http://maps.google.com/

2 Background

As pointed out in [4], generalizing a map to be viewed at a lower resolution involves two stages. First, the most relevant features to be displayed are chosen based on the model of that information. In the example in Figure 1, it is decided that at certain zoom levels, roads are not selected at all from the model to be rendered. The second step involves analyzing the geometric properties of those features that are used to simplify the graphical rendering for the final map. Again the example demonstrates this well: the bodies of water have rougher shapes on the lower resolution maps, and become increasingly detailed as the maps are zoomed in.

Early research on map generalization (1960's and 1970's) was concentrated on the second step: line simplification. As Jones *et al.* state, "very much less research effort was devoted to solving the map generalization problem in a holistic manner that takes account of cartographic constraints on the relationships between multiple map objects" ([5]). More recent effort has been spent on the first, model-based stage of generalization. This is the area this paper will concentrate on.

Jiang and Harrie, again in [4], claim that a common technique for selecting streets from a road network for generalization is to simply use the classification of the roads (for example, choose to display highways over residential streets). There is no doubt that the type of road is important, but there are other properties as well that should not be overlooked.

The work of Mackaness and Beard in [7] is pointed out at notable as it makes use of the connectivity properties of a street network in the generalization process. Jiang and Claramunt also make use of connectivity in [3], analyzing the topological properties of the network using a graph with streets as nodes and edges between streets that intersect. Non-structural properties can be used to model the importance of various streets as well, as seen in [8] where Morisset and Ruas use an agent-based simulation to select streets based on how much they were used. More recently, Hu *et al.* use mesh density as a measure to control street selection.

Jiang and Harrie propose in [4] a selection method that categorizes streets using a self-organizing map. Here, this idea will be implemented using free, open source software packages that could be integrated to a system using these tools in a self-contained way.

3 Street Selection Using Self-Organizing Maps

Self-organizing maps are a type of artificial neural network, and are used to cluster and visualize data. Given a set of d-dimensional vectors, the learning process will arrange these into an output space usually of one or two dimensions, retaining any patterns found in the data. A brief explanation of the algorithm can be found in [4], or a more complete description can be found in [6].

The key for this discussion is the determination of the properties from a road network that will be used as input to the self-organizing map. Once these are known, they can be fed to a generic self-organizing map algorithm (along with some other parameters that define the map's learning process), and the results can then be used to select the most important roads.

As seen in section 1, there are many properties beyond road class that should be considered when selecting roads. The properties used here are degree, closeness, betweenness, length, lanes, speed, and class [4]. Each of these is discussed below.

The first three properties are chosen because they describe something about the topological nature of the road network. To obtain these values, a connectivity graph must first be constructed as follows: Every road in the network will be represented by one node, and an edge will exist between two nodes if the roads they represent intersect in any way. It is important to note the distinction between this connectivity graph and the original road network. No geometric properties are stored in it, and any algorithms such as shortest path must be purely graph-based.

The degree of a particular road in the connectivity graph describes how many other roads it intersects with. More important roads will tend to pass over a large number of (probably) less important roads.

The measure of closeness reveals something about the structural relationship between a given road and all other roads around it. While degree is a local property of the road, closeness is global. It is calculated as the number of other roads in the network divided by the sum of the shortest distances between this road and all the others: $\sum_{k=1}^{n} \frac{n-1}{d(v_i,v_k)}$

The last topological property is betweenness. This one is important because the number of connections to other roads cannot alone define how important a particular street is. A street with few connections, for example, may be necessary in keeping the network connected. Betweenness is defined as the sum of the proportions of shortest paths between all other roads that pass through the given road: $\sum_{j=1}^{n} \sum_{k=1}^{j-1} \frac{p_{ikj}}{p_{ij}}$.

The remaining properties are geometric and semantic in nature. Because roads that are longer and have more lanes tend to be more important, length and width (which here will equal number of lanes) are chosen. Furthermore, much can be learned from the class of road (which can be described with terms like avenue, highway, motorway, and so on), so it too is chosen.

These seven properties are normalized into a range of [0,1] to ensure that each property is given the same relevance in the self-organizing map learning process, since there can be a very high variance between the ranges. It is important to point out that for each property must have higher numbers indicating higher importance. Weights can then be optionally applied if certain properties are deemed more important for particular applications. When ready, a set of property values for each road is passed as input to the self-organizing map process.

As suggested in [4], the parameters specifically used for the self-organizing map algorithm can be found in Figure 2.

Parameter	Value
Size	100
Dimensionality	2
Shape	Sheet
Map lattice	Hexagonal
Neighbourhood	Gaussian
Learning rate	$\alpha(t) = \alpha_0/(1 + 100t/T)$
Initial learning rate	0.5 for coarse period
	0.05 for fine period
Initial neighbourhood radius	5
Final neighbourhood radius	1.25 for coarse period
	1 for fine period

Figure 2: Parameters for Self-Organizing Map algorithm.

The self-organizing map algorithm will output a set of cells based on the dimensionality provided in the parameters (in the case of those shown in Figure 2, the algorithm will return 100 cells). Each cell will contain a number of input vectors, clustered together based on their similarities.

To select the most important streets using these cells, the average values of the input properties found in each cell are used. The streets that belong to cells with the highest averages will be chosen.

4 Implementation With GeoTools and JSOM

Jiang and Harrie use ArcView GIS and Matlab in their implementation, and while these tools are industry standards, they are commercial and not freely available. This Java implementation makes use of several free open source packages and code bases instead, allowing for a complete and self-contained system. This section will outline a few details about the implementation and provide an example of its use.

The main component of the implementation is the GeoTools open source Java Toolkit[2]. This library is fully featured for the general manipulation of geospatial data and adheres to the OpenGIS® Specifications[3]. It is capable of opening several common data formats including ESRI shapefiles, geography markup language, and PostGIS, and has built in rendering capabilities.

JavaSOM[4] is the implementation component of Jav's work in [1]. It is a basic self-organizing map library that supports input and output of data via XML files.

Looking to the `shapeReader` package, the main starting point of the implementation is the `ShapeFile` class. It can load an ESRI shapefile and provide access to the features found within. In the example test run, a shapefile from Statistics Canada was obtained[5]. The file contains a road network for the province of Ontario.

The `AreaOfInterest` class facilitates a new view of a set of features filtered either by geometric bounds or a list of feature identifiers. The example creates an instance of this class in order to work with a smaller set of data (the original file has 473,252 features and the new interest contains just 239). The area of interest can be rendered from this class as well. Figure 3 shows the road network contained in the example's area of interest.

Figure 3: A subset of road networks in Ontario.

The `shapeReader.graph` package contains classes that are used to build a connectivity graph

[2]http://geotools.codehaus.org/
[3]http://www.opengeospatial.org/standards
[4]http://javasom.sourceforge.net/
[5]http://geodepot.statcan.ca/Diss2006/DataProducts/RNF2006_e.jsp

from a set of features and extract information about the graph (namely, degree, closeness, and betweenness). The graph is implemented as a part of the GeoTools graph framework, and as such, the algorithms for computing this information are limited by the capabilities of GeoTools.

Finally, the `streetClassifier` package contains the main driver for the street selection. The main class of note here is the `StreetSelector` class. It can perform the whole process beginning with an area of interest and a percentage of important roads to select. A new area of interest with only the selected streets is returned. Figure 4 shows a reduced map with the top 80% of roads selected.

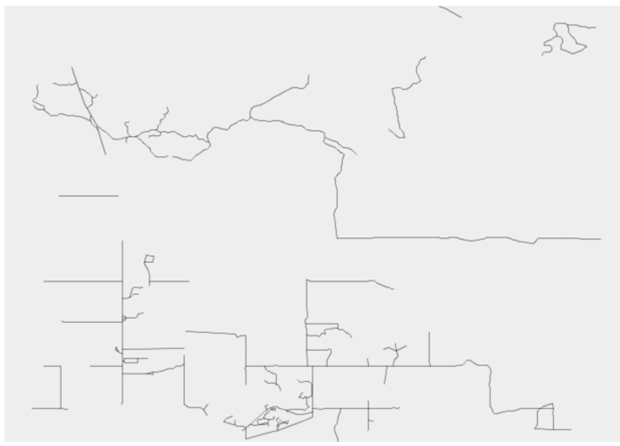

Figure 4: A reduced map having selected the top 80% of streets as chosen from the self-organizing map.

5 Conclusion

In this paper the idea of selecting streets from a network using self-organizing maps [4] was reviewed. A self-contained implementation in Java was created using free open-source software, making it usable by anyone implementing their own GIS. Future work includes improvement of the algorithms used to calculate connectivity properties as well as tweaking the weights for each property. The categorization of street types also needs improvement as the initial round involved estimation and guess work.

References

[1]

[2] Y. Hu, J. Chen, Z. Li, and R. Zhao, Selection of streets based on mesh density for digital map generalization. In *Fourth International Conference on Image and Graphics*, pp. 903–908, 2007.

[3] B. Jiang and C. Claramunt, A structural approach to model generalisation of an urban street network. In *Proceedings of the Fourth AGILE Conference on Geographic Information Science*, Mallorca, Spain.

[4] B. Jiang and L. Harrie, Selection of streets from a network using self-organizing maps. *Transactions in GIS*, **8(3)**:335–350, 2004.

[5] C. B. Jones, G. L. Bundy, and J. M. Ware, Map generalisation with a triangulated data structure. *Cartography and geographic information systems*, **22(4)**:317–331, 1995.

[6] T. Kohonen, *Self-Organizing Maps*. Springer, Berlin, third edn.

[7] W. A. Mackaness and M. K. Beard, Use of graph theory to support map generalisation. *Cartography and geographic information systems*, **(22)**:210–221, 1993.

[8] B. Morisset and A. Ruas, Simulation modeling for road selection in map generalisation. In *Proceedings of 18th ICC*, Stockholm.