

Evaluating the Consistency of Cartographic Generalization*

Michiel Jansen

Marc van Kreveld

Abstract

Automated cartographic generalization has been studied much better than the automated evaluation of it. In this paper a tool is proposed that helps to evaluate certain perceptual characteristics in generalization. This tool is the clutter function, which quantifies the amount and location of clutter (or congestion) on a map. We analyze which aspects contribute to clutter on a map, and how this leads to a definition of the clutter function. We show how to compute the clutter function efficiently, and how to use it to evaluate cartographic generalization.

Keywords: Cartographic generalization, evaluation, consistency, clutter.

1 Introduction

Automated map generalization is a topic into which a lot of effort has been put. Although it is still considered a difficult problem and far from solved, some understanding of the issues has been acquired in the past years. Three books have appeared exclusively on different aspects of generalization [2, 8, 9].

In a rather simplistic view, map generalization proceeds in a few steps. Firstly, the need for generalization is detected. Secondly, the locations on the map or the organization of the data where generalization is necessary is determined. Thirdly, for each such location, one must decide how the problem is overcome by choosing a suitable generalization operator. This operator is implemented by an algorithm, a well-defined sequence of actions that performs a specific task. Running the algorithm is the fourth step.

One could add to these steps a fifth one: evaluation of the result. Only few papers discuss the issue of evaluation of map generalization, or the (semi-)automated version of this task [4, 6, 1]. The report of the ICA Workshop on Map Generalization [7] discusses several of the issues and identifies a need for quality assessment techniques for map generalization.

In this paper a simple tool is provided that can help with the evaluation of maps generalized for display purposes (cartographic generalization). A very similar tool was suggested before by Mackaness [6] for computer-assisted generalization. It can be used, for instance, after automated generalization to test whether the process was done consistently throughout the map. If not, a warning can be sent to the cartographer with the information where too much or too little generalization has been done. Or it can be redirected to the automated generalization package to undo generalization steps, or to take extra steps.

The idea of the evaluation is simple; intuitively, we take a tessellation into squares and overlay it with a map, as in the bottom right picture of Figure 1. Then we determine the

*This research was partially supported by the ESPRIT IV LTR Project No. 21957 (CGAL). Authors' address: Dept. of Computer Science, Utrecht University, P.O.Box 80.089, 3508 TB Utrecht, the Netherlands. E-mail: marc@cs.ruu.nl. Phone: +31-30-2534119. FAX: +31-30-2513791.

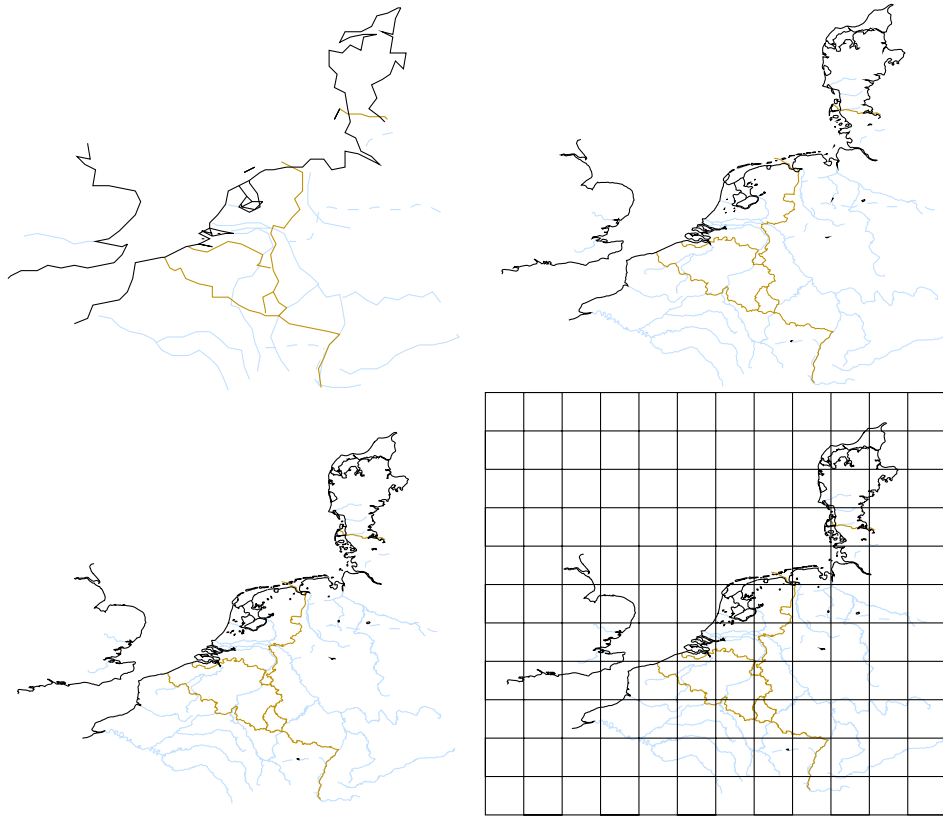


Figure 1: Maps of a part of Europe as in the CIA World Bank at three different levels of detail. The fourth picture shows an overlay with a grid.

amount of clutter inside each square. Clutter is some suitably chosen formula that has the number and complexity of the map features as an input and gives a value. This results in a matrix of values. If we apply this idea both to a map made from an original data set and one made from a generalized data set, we can compare the two matrices. Generalization has been done consistently, in a sense, if the values in the squares after generalization are, say, $1/4$ of the values in the corresponding squares before generalization.

The clutter function has been suggested before by Mackaness [6], who called it a feature density surface. He introduced it to estimate how much work is needed for generalization in different regions of a map. Our contribution is a formal study of what causes clutter, an operational definition for clutter, an efficient algorithm to compute it, and more ideas of how it can be applied for automated cartographic generalization.

2 Generalization and evaluation

Weibel [13] listed four types of constraints that generalization methods must respect, namely metric, topological, semantic, and Gestalt. He studied these four types of constraints in particular for line simplification, but they can be used for other generalization operators as well. Metric and topological constraints can usually be defined formally. A good definition of semantic constraints is more subtle, but the constraints of the type Gestalt are rather

ill-defined, because they deal with the perception or impression of a map. Therefore, the algorithms that implement a generalization operator can usually be designed so that they obey metric, topological, and semantic constraints, but not Gestalt constraints. Another reason why the algorithm cannot take this into account is that Gestalt constraints should be evaluated after all generalization operators have been performed, not in the intermediate stages. It will be the case, for instance, that after application of generalization operators on the left side of the map first, a strange impression of the map results (due to inconsistency), until the generalization operators have been applied to the right side of the map as well.

2.1 Metric, topological, and semantic constraints

Metric constraints are constraints like avoiding too small features (imperceptibility), and avoiding that two objects are too close (coalescence). Such constraints can easily be tested for. Imperceptibility or coalescence shows a need for generalization. The generalization operator that deals with the problem makes sure that it is solved. For example, when displacement is used to solve coalescence, one object is moved over such a distance that the objects are sufficiently far apart.

Topological constraints include avoiding self-intersections of a polyline, and keeping a road network connected. Such problems can occur when a generalization operator is not designed well, or as a temporary artifact that is resolved later. line simplification routine should be such that a polyline does not get self-intersections. Removal of a road from a network should not be done if it makes the network disconnected, or the connectedness should be restored later on.

Semantic constraints are necessary to guarantee that the map is meaningful and does not contain apparent contradictions, like a river that runs up a slope after line simplification of the river. As another example of a semantic constraint, consider road map generalization. It is important that no large detours be created, since these suggest the semantically wrong conclusion that there is no short connection.

Although the three types of constraint listed above are by not means easy to guarantee, the design of a generalization operator can often test for the constraints and take them into account.

2.2 Gestalt constraints

A fourth type of constraint that should be respected when performing generalization are the Gestalt constraints. They relate maintaining the overall impression of a map, which implies maintaining global shape of features, global locations and distribution of features, and keeping consistency in generalization. It has been observed that Gestalt principles “are often hard to translate into operational terms and can be expected to be considerably more demanding than other constraints in terms of database preparation and structural analysis” [13].

2.3 Local vs. global

Some of the generalization operators have a local effect on the map; others operate globally. This distinction influences whether a constraint should be enforced during the design of the algorithm that implements the operator, or whether the constraint can better be evaluated later.

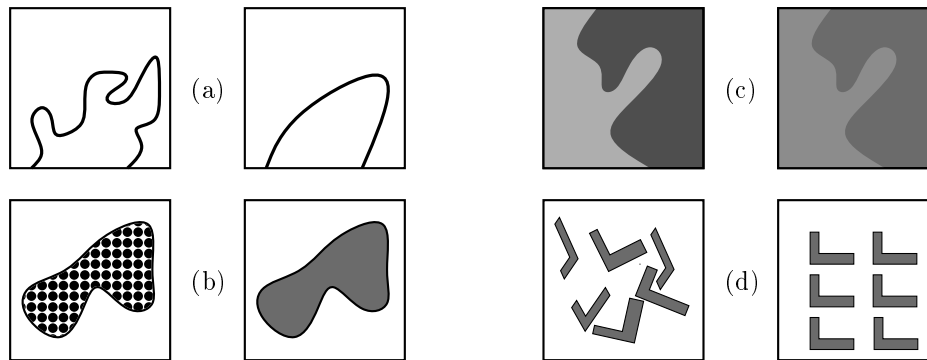


Figure 2: Some factors that influence the perceived clutter: (a) length of a line, (b) texture of a region, (c) contrast between regions, and (d) regularity of a group of features. The left pictures appear more cluttered than the right pictures.

For example, reclassification is a global operator. When deciduous and evergreen forest classes are joined for display on a generalized map to the class forest, then this applies to the whole map.

On the other hand, exaggeration and displacement are local operators. They change the map only locally, and therefore they may cause inconsistent map generalization. Which appears as a violation of Gestalt constraints, and sometimes of semantic constraints as well.

3 The clutter function

In this section we introduce a tool to evaluate Gestalt constraints in part, namely, the aspects that relate to feature density, distribution, and clutter. We first analyze what visual characteristics influence these aspects, then we define the clutter function, and then we discuss how the clutter function can be visualized and used in fully automated or computer-assisted generalization.

3.1 Definition

Clutter or congestion will be defined for any point on a map. The clutter at a point is determined by the number and complexity of map objects in the neighborhood of that point. We've chosen to take a square centered at the point to define neighborhood. Clutter is a concept that can occur at various scales. The whole map can appear cluttered, or only some section of it. Global clutter is considered by taking a large square centered at the point, and local clutter is considered by taking a small square.

In view of texts on the perception of maps (e.g. Part V from Robinson et al. [12]), the following aspects of the map influence the presence and amount of clutter in a square (see also Figure 2):

1. Point feature visual variables: size, color, and contrast.
2. Line feature visual variables: length, color, thickness, and contrast. A longer stretch of line within a square gives rise to more clutter in that square. Similarly, the color, thickness, and contrast with the background of the line matter.

3. Area feature visual variables: size, color, and texture. If a clutter square contains a region that is bright red, this is perceived as more cluttered than if that region were grey. The texture used for a region also influences clutter.
4. Boundary length and contrast. A boundary is a change in color on the map, without the presence of a line feature. Since the boundary does not have a thickness or color, only its length and the contrast of colors of the incident regions matter for the amount of clutter.
5. Regularity in groups of features: repetition, alignment, orientation, and symmetry. The presence of many small features in a clutter square obviously adds to the amount of clutter by the aspects just mentioned. However, if the many small features have the same size, shape, and orientation, and they are aligned, then much less clutter is perceived. So the aspects above may overestimate clutter if there is regularity of some sort.
6. Labels. Obviously labels add to the perceived clutter on the map.

We have chosen to take into account only the first four aspects. They seem most important, and most straightforward to define. There is no fundamental reason why the other aspects can't be incorporated. But it is far from easy to detect regularity in a group of features, and to define how it should influence the clutter function. It is unclear whether the clutter caused by labels should be used to evaluate the quality of generalization, which is the primary purpose of designing the clutter function. If the primary purpose were evaluation of the clutter on a map itself, then certainly labels must be taken into consideration.

We define the amount of clutter at a point p for a chosen scale as follows. The point p and chosen scale define a square S centered at p with a given size. The clutter at p is:

$$\begin{aligned} \text{clutter}(p, \text{scale}) = & \sum_{\text{points } q \text{ in } S} \text{VisVar}(q) + \sum_{\text{lines } l \text{ in } S} \text{Length}(l) \cdot \text{VisVar}(l) + \\ & \sum_{\text{regions } r \text{ in } S} \text{Area}(r) \cdot \text{VisVar}(r) + \sum_{\text{boundaries } b \text{ in } S} \text{Length}(b) \cdot \text{Contrast}(c_1, c_2) . \end{aligned}$$

In the formula above, $\text{VisVar}(\dots)$ gives the cost of one unit of the feature inside the square and is determined by the visual variables. For lines, the unit is the unit of length, and for regions, the unit is the unit of area. The first summation is over all point features inside square S . The second summation involves the line features that intersect square S , but only for the part that is inside S . The third summation is over all regions that intersect square S , and $\text{Area}(r)$ gives the area of region r inside square S . The fourth summation is over the boundaries between regions, again restricted to the parts inside square S . The colors c_1 and c_2 are the two colors to the two sides of the boundary, and the function $\text{Contrast}(c_1, c_2)$ gives the clutter cost per unit length of a boundary between two regions with the colors c_1 and c_2 .

The formula for clutter as given above is somewhat restricted. It doesn't deal with the interaction of different map features, and how that affects the perception. But it can serve well as a first attempt to quantify clutter or congestion on a region.

Now that we have defined a clutter value for every point on the map, the clutter function simply is the function that maps points on the map, given by two coordinates, to the corresponding clutter values. The function is in a sense a digital elevation model where elevation is clutter. It usually is not continuous for a given data set.

3.2 Use and visualisation

In order to compute, use, and visualize the clutter function, we need to evaluate the amount of clutter at every point. Clearly this is not possible since a map consists of an infinite number of points. Therefore we evaluate the amount of clutter only at some subset of points, placed in a grid on the map. If we place these points the square size apart, we consider the amount of clutter in each square of a regular, square tessellation of the map, see Figure 1. Typically, 15×15 squares to cover a map detects clutter globally, and 50×50 squares gives a local determination of clutter. So the clutter function is approximated by a two-dimensional array, a matrix with entries that represent the amount of clutter. We call this matrix the clutter matrix.

The clutter matrix is a tool to help evaluate cartographic generalization. It needs to be interpreted before one can say that generalization is done correctly as far as certain Gestalt aspects are concerned. The values of the clutter matrix of a map by itself are not so useful; it is normal for a map to have more cluttered parts and less cluttered parts. How the clutter matrix can help is by computing it for both the base map and the generalized map. Then we can see if generalization has caused some parts of the map to become a lot less cluttered, relatively, than other parts. Which tells something about the consistency of generalization throughout the whole map.

In more detail, suppose that a base map and a generalized version of it are given. We choose a scale at which clutter will be evaluated, and then compute the clutter matrix for both maps. Clearly, the clutter matrix on the base map will have greater values than on the generalized map. To compare consistency in generalization, we *normalize* both clutter matrices separately. No clutter corresponds to the value zero and the maximum occurring clutter value is transformed to the value 1. All other clutter values are scaled to fit in this interval.

We then subtract the corresponding entries of the two normalized matrices. Ideally, all subtractions give values close to zero, which implies that generalization has been performed consistently. Large values in the positive or negative indicate parts of the map where generalization was applied considerably more than the average, or considerably less than the average.

Our implementation shows the clutter matrix by showing the matrix of squares in shades of grey, see Figure 3. Dark grey shows more clutter than light grey. The subtraction of two normalized clutter matrices can also be shown as a matrix of squares, where light shades show consistent generalization and dark shades show less consistent generalization, see Figure 4. It would be best to show positive values in shades of red and negative values in shades of blue. In this paper plus and minus signs are used in the squares, because colors cannot be used here.

Detection of parts where generalization was done differently can lead to a warning to the human cartographer that inconsistency has been detected, or it may be redirected to the automated generalization package to generalize certain map parts even more, or to undo some generalization steps. So the clutter matrix can have use in computer-assisted evaluation and in fully automated evaluation of map generalization.

The clutter matrix can also be used to assess the conspicuity of a map feature. If an important map feature lies in a rather cluttered square it may be overlooked by the map reader. This may be a reason to apply more generalization in the square, or exaggerate the important map feature.

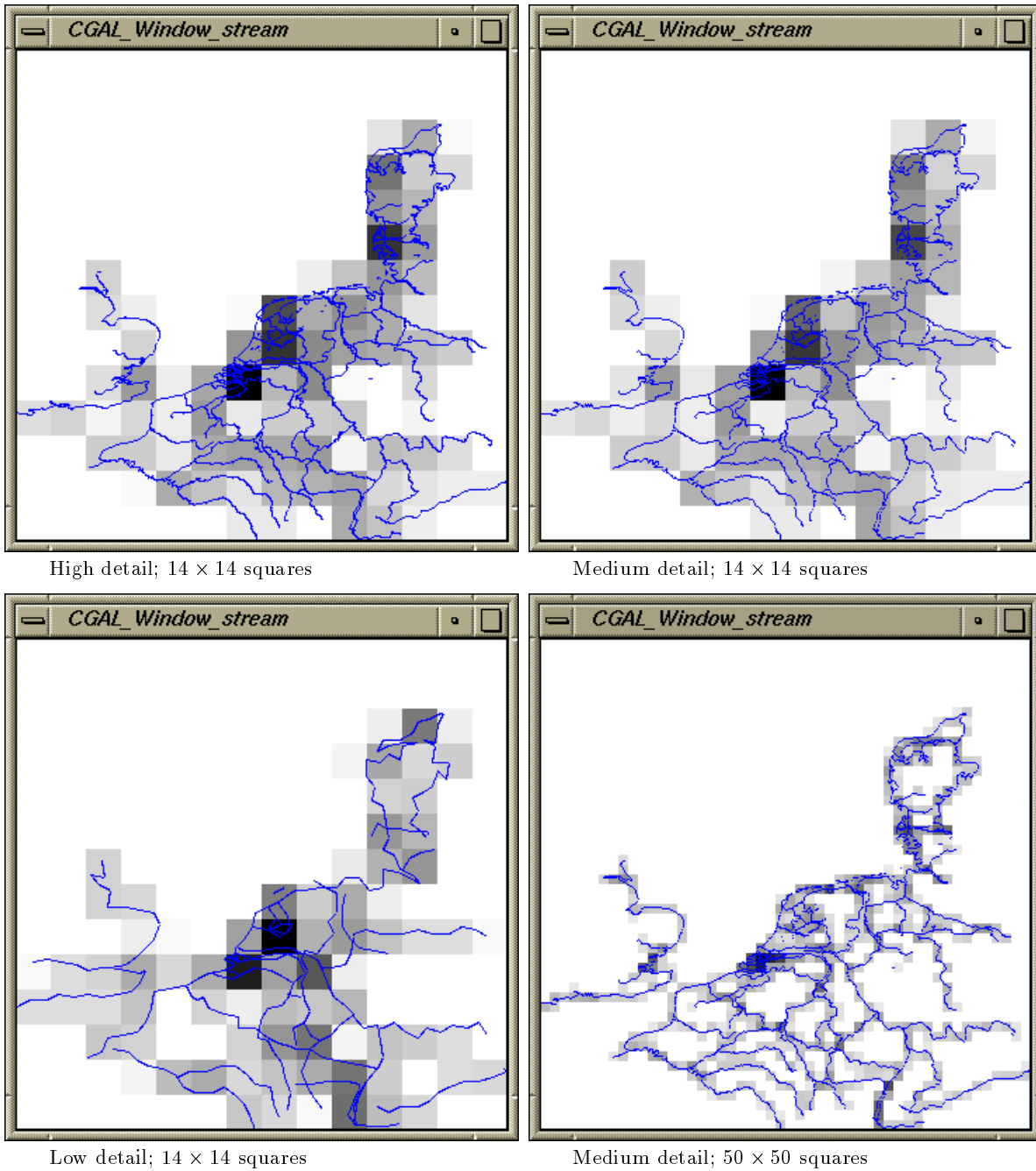


Figure 3: The clutter matrix shown by shades of grey on the background for high detail, medium detail, and low detail maps. The fourth map shows the clutter matrix at a different scale of clutter (square size).

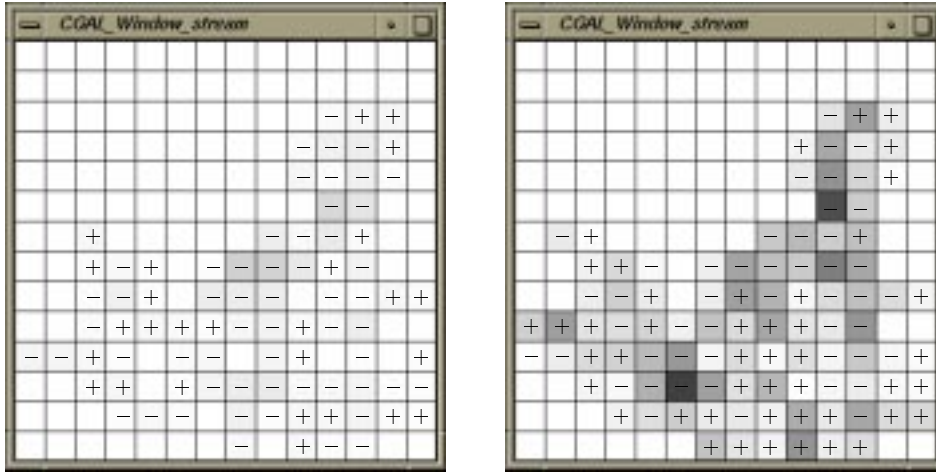


Figure 4: Left, the subtraction of the normalized clutter matrices of the high detail and medium detail maps of Figure 3. Right, the subtraction of the normalized clutter matrices of the high detail and medium detail maps of Figure 3. Shades of grey are made twice as dark for illustration purposes. Plus and minus signs are used to indicate more and less than average generalization, respectively.

4 Computation of the clutter matrix

The input to the clutter matrix computation is a map, three tables (for points, lines and regions) to convert visual variables to the corresponding $VisVar(\cdot)$ -value, a table to convert pairs of colors to the contrast value, and a scale (represented by a square size) at which clutter is evaluated. How the tables and conversion are determined is not relevant for the methodology; we simply assume that the tables are given.

4.1 Raster

We'll first assume that the map is raster based. Since rasters don't explicitly have point or line features, the definition of the clutter matrix must be simplified. A point or line is regarded as a region so we only need to consider clutter by color of areas and clutter by contrast between areas. Now the clutter matrix can be computed rather easily. Each square in the clutter matrix covers a square group of pixels, say, $k \times k$ grid pixels in a square. See Figure 5 for an example. First we determine the number of pixels of each color in the $k \times k$ subgrid, and use a table to compute clutter by color. Then we determine the boundaries between pixels of different color. We consider all pixel boundaries inside the square, and also the boundaries at the bottom and right side of the square. For each such boundary we consult another table with the pair of colors of the pixels to compute clutter by contrast. We add these values to get the total clutter inside one square, and then we continue with the next square of the clutter matrix.

There are several reasons why raster based clutter computation is not such a good idea. Firstly, most generalization operators are described as vector based operations. Secondly, the length of diagonal lines or boundaries would be overestimated. A diagonal line of length $Length$ would contribute in total to the clutter matrix by as much as $\sqrt{2} \cdot Length$ because

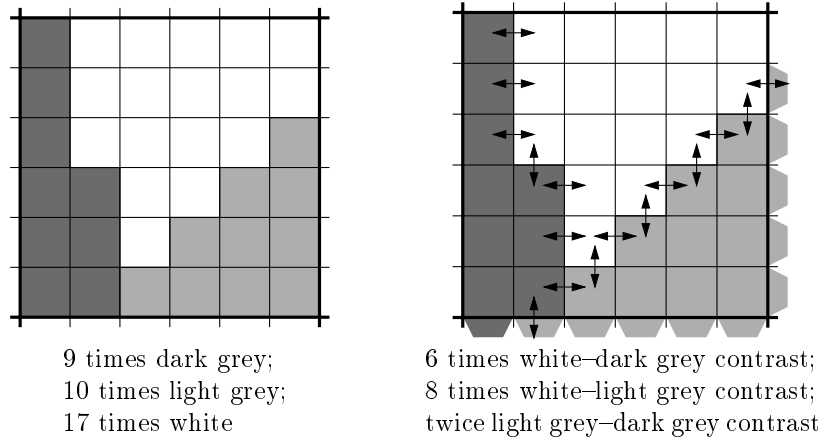


Figure 5: One square of the clutter matrix to compute clutter by pixel colors (left) and by boundary contrast (right).

of rasterization. The total clutter would be influenced by the orientation of the raster, an undesirable situation. To undo this effect one would have to recognize diagonal lines, which basically comes down to raster-to-vector conversion. Aspects like line display type, and texture of regions are impossible to incorporate directly as well.

4.2 Vector

To compute the clutter matrix for a vector based map, one first has to determine for every square of the clutter matrix which lines and regions of the map intersect the square. This basically comes down to map overlay of the vector map and the tessellation into squares. In our case, the overlay is easier to compute than usual, since one of the subdivisions is regular.

Every point feature can be located in the square tessellation simply by taking the coordinates, and transforming these to get the index of the square in the clutter matrix in which it falls. The visual variables of the point feature are used to determine the contribution to the clutter value in that square.

Every line feature can be located in the square tessellation by locating one endpoint first, and then tracing the rest of the line feature through the adjacent squares. For each piece of line feature, its length can be determined and its display parameters can be used to find the visual variables value per unit length in a table. It is easy to add up all contributions of the line features and add them to the values already present in the clutter matrix.

Every region feature can also be located in the square tessellation. This is done by locating the boundaries of a region first, and then computing the area of intersection of each region and each square. Some care has to be taken in this operation, because the intersection of a square with polygon with holes generally is a collection of polygons with holes, see Figure 6. Again the visual variables value per unit area of the region is found in a table, and the contribution can be determined.

The contribution of boundaries to the clutter values in square tessellation can be done in the same way as was done for line features. Only this time we use the contrast table on the colors of the incident regions. If the subdivision is stored in a topological data structure, the incident regions can easily be retrieved from the boundary.

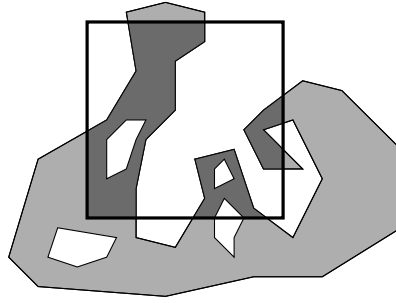


Figure 6: The intersection of a square with a polygon with holes. The summed areas of the dark grey regions is the contribution of the region to the clutter in the square.

The implementation of the clutter matrix computation made use of two libraries: the LEDA library [5, 10] for efficient data structures and algorithms, and the CGAL library [3, 11] for efficient geometric computation.

5 Final remarks

We have only done preliminary tests of the clutter matrix and its visualization. The CIA World Bank has been used to obtain an idea of the output. The data is provided at three different levels of detail, and for each, the clutter matrix was computed and visualized in shades of grey as the background of the map. Each shade of grey in a square is a conversion of the clutter value computed in the square. Squares with no map features will be white and the most cluttered square is black, which comes down to normalization of the clutter matrix. The results are shown in Figure 3. To visualize the difference in extent of generalization in each of the squares the entries in the normalized clutter matrices were subtracted. The subtraction of the high detail and medium detail, and high detail and low detail, are shown in Figure 4.

The clutter function and clutter matrix seem to be a useful tool for the evaluation of cartographic generalization. However, to really make the ideas useful in practice further research on the topic is necessary. This includes developing the best formulation of the definition of clutter, the precise contributions of various visual variables to the amount of clutter, incorporation of regularity in the definition, and studying how the clutter matrix can best be integrated in the whole process of cartographic generalization. Extensive experimentation is needed for each of these issues.

Note. The technical report version of this paper will soon be available at <http://www.cs.ruu.nl/docs/research/publication/TechList2.html>, e.g. to obtain an original print of the images with shades of grey.

Acknowledgement. The authors thank Frank Brazile and Robert Weibel for helpful comments and references.

References

- [1] F. Brazile. A generalization machine design that incorporates quality assessment. In *these proceedings*, 1998.
- [2] B.P. Buttenfield and R.B. McMaster, editors. *Map Generalization: Making Rules for Knowledge Representation*. Longman, London, 1991.
- [3] CGAL, Computational Geometry Algorithms Library. <http://www.cs.ruu.nl/CGAL/>.
- [4] R. Ehrliholzer. Quality assessment in generalization: integrating quantitative and qualitative methods. In *Proc. of the 17th Int. Cartographic Conference*, pages 2241–2250, 1995.
- [5] LEDA, Library of Efficient Data structures and Algorithms. <http://www.mpi-sb/LEDA/>.
- [6] W.A. Mackaness. A constraint based approach to human computer interaction in automated cartography. In *Proc. of the 17th Int. Cartographic Conference*, pages 1423–1432, 1995.
- [7] W.A. Mackaness, R. Weibel, and B.P. Buttenfield. Report of the 1997 ICA Workshop on Map Generalization, 1997. Gävle, Sweden, 19–21 June. Publication of the ICA Working Group on Map Generalization. <http://www.geo.unizh.ch/ICA/>.
- [8] R.B. McMaster and K. Stuart Shea. *Generalization in Digital Cartography*. AAG, Washington, D.C., 1992.
- [9] J.-C. Müller, J.-P. Lagrange, and R. Weibel, editors. *GIS and Generalization – Methodology and Practice*, volume I of *GISDATA*. Taylor & Francis, London, 1995.
- [10] S. Naeher and K. Mehlhorn. LEDA: A library of efficient data types and algorithms. In *Proc. Internat. Colloq. Automata Lang. Program.*, pages 1–5, 1990.
- [11] Mark H. Overmars. Designing the Computational Geometry Algorithms Library CGAL. In M. C. Lin and D. Manocha, editors, *Applied Computational Geometry (Proc. WACG '96)*, volume 1148 of *Lecture Notes Comput. Sci.*, pages 53–58. Springer-Verlag, 1996.
- [12] A.H. Robinson, J. Morrison, P.C. Muehrcke, A.J. Kimerling, and S.C. Guptill. *Elements of Cartography*. John Wiley & Sons, New York, 6th edition, 1995.
- [13] R. Weibel. A typology of constraints to line simplification. In M.J. Kraak and M. Molenaar, editors, *Proc. 7th Int. Symp. on Spatial Data Handling*, pages 9A.1–9A.14, 1996.