Generalization of the "Douglas and Peucker" Algorithm for Cartographic Applications

Xavier BARILLOT, Jean-François HANGOUËT, Hakima KADRI-DAHMANI

Laboratoire Cogit, Institut Géographique National, 2-4 Avenue Pasteur, 94165 Saint-Mandé Cedex, France
Xavier.Barillot@ign.fr, Jean-Francois.Hangouet@ign.fr, Hakima.Kadri-Dahmani@ign.fr
Fax. (+33) 1.43.98.81.71

Abstract
In this paper, the famous Douglas & Peucker algorithm for line generalization is mathematically revisited and shown to be applicable to sets of objects of any geometric kinds.

Keywords
Cartographic Selection, Generalization, Hausdorff distance, Spatial Analysis

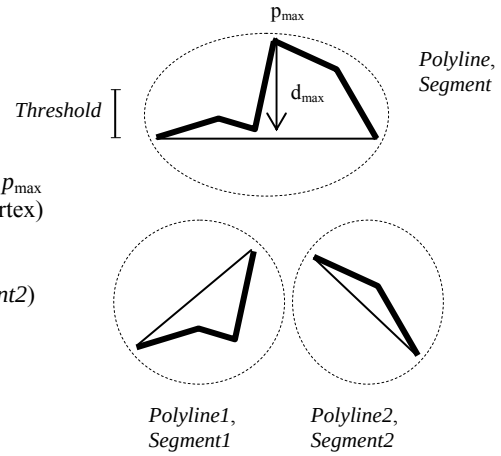
I. Introduction
In his "*Theory of the Cartographic Line*", Thomas K. Peucker explains that the algorithm he designed in collaboration with David H. Douglas was meant for line generalization (Peucker 1976 : 138), and contemporarily though independently developed in "*at least two other papers*", yet for other purposes. In vector cartography today, the Douglas & Peucker algorithm is certainly the most popular line simplification operator, widespread both in commercial GISs (because it is easy to program and it provides tolerable results) and in research publications (because it can be altered, refined or indeed improved in a variety of directions). Reviews can be found in (Cromley-Campbell 1991), (Hershberger-Snoeyink 1992), (Plazanet 1996), (Visvalingam-Herbert 1999).

The Douglas & Peucker algorithm takes for input a *Polyline* (a chain of vertices or, equivalently, of segments), a *Segment* (usually the first and last vertices of the *Polyline*), and a *Threshold* (a distance threshold). By recursively splitting the *Polyline*, it returns a succession of segments. The algorithm runs as follows:

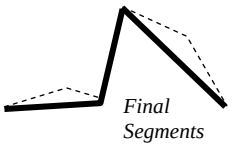
DP(*Polyline*, *Segment, Threshold*)
d_{max} ← greatest distance from *Polyline* to *Segment*
p_{max} ← corresponding farthest *Polyline* vertex

if d_{max} > *Threshold* then
    Polyline1 ← All *Polyline* segments from p_i (first vertex) to p_{max}
    Polyline2 ← All *Polyline* segments from p_{max} to p_f (last vertex)
    Segment 1 ← [p_i, p_{max}]
    Segment 2 ← [p_{max}, p_f]
    Return DP(*Polyline1*, *Segment1*) ⋃ DP(*Polyline2*, *Segment2*)
else
    Return {Segment}
end if

end program



Douglas and Peucker have generalized their own algorithm to terrain simplification, using a DTM instead of the *Polyline*, *triangles* instead of *segments*, and computing the maximal distance from DTM vertex to current triangular facet instead of computing the maximal distance from polyline vertex to segment; the initial approximation consisting of the corners of the DTM linked triangularily to the highest vertex (Douglas-Peucker 1975).



The aim of this paper is to show that the Douglas & Peucker algorithm, though originally designed for line or surface generalization, may be used for the generalization (either abruptly, or as a preliminary phase) of any kinds of objects, with a few mathematical arrangements only.

II. Notations

This section, where all our notations are introduced, may be skipped upon reading if appearing too tedious. Upon writing however we cannot skip it, as the mathematical fundamentals of our algorithm all belong here.

II.1 General

Set difference: $\mathsf{X}\backslash\mathsf{Y}$ denotes the set of elements in set X which are not in set Y.

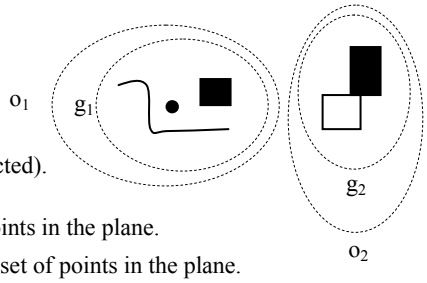
Set of subsets: $\mathsf{P}(\mathsf{X})$ denotes the set of all (finite) subsets of X.

II.2 Geometry and Objects

P is the plane, identified to R^2, measured by the euclidean distance (R is the set of real numbers).

p is a point in the plane: $p \in \mathsf{P}$.

$d(p,q)$ is the euclidean distance between points p and q.



g is a closed and bounded set of points in the plane.
o is the set of a unique g : $o = \{g\}$.
Practically: o is an object, g a geometry (allowed by definition to be disconnected).

G stands for the set of all possible g : G is the set of closed bounded sets of points in the plane.

O stands for the set of all possible o : O is the set of sets of a closed bounded set of points in the plane.

A *Figure* is a finite set of elements of O. A *Figure* belongs to $\mathsf{P}(\mathsf{O})$ (eg. *Figure* = {o1, o2}).

To provide a handle on the geometry of a given object, the γ operator is introduced as :

$$\gamma : \quad \mathsf{O} \to \mathsf{G}$$
$$o \mapsto \gamma(o) = g \in \mathsf{G} \mid \{g\} = o$$

To retrieve the geometry of a given *Figure*, the Γ operator is defined as:

$$\Gamma : \quad \mathsf{P}(\mathsf{O}) \to \mathsf{G}$$
$$\mathit{Figure} \mapsto \Gamma(\mathit{Figure}) = \bigcup_{o_i \in \mathit{Figure}} \gamma(o_i)$$

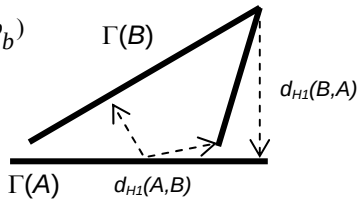
II.3 Hausdorff Measures

The following function d_{Hu1} is introduced to measure the greatest spatial distance from an object to a set of objects:

$$d_{Hu1} : \quad \mathsf{O}\times\mathsf{P}(\mathsf{O}) \to \mathsf{R}^+$$
$$(o,B) \mapsto d_{Hu1}(o,B) = \mathit{Max}_{p_o \in \gamma(o)} \ \mathit{Min}_{p_b \in \Gamma(B)} \ d(p_o, p_b)$$

The following function d_{H1} is introduced to measure the greatest spatial distance from a set of objects (A) to another set of objects (B):

$$d_{H1} : \quad \mathsf{P}(\mathsf{O})\times\mathsf{P}(\mathsf{O}) \to \mathsf{R}^+$$
$$(A,B) \mapsto d_{H1}(A,B) = \mathit{Max}_{p_a \in \Gamma(A)} \ \mathit{Min}_{p_b \in \Gamma(B)} \ d(p_a, p_b)$$



d_{H1} is not symmetric.
It is worth recalling that function d_H :

$$d_H : \quad \mathsf{P}(\mathsf{O})\times\mathsf{P}(\mathsf{O}) \to \mathsf{R}^+$$
$$(A,B) \mapsto d_H(A,B) = \mathit{Max}\ \left(d_{H1}(A,B) \ , \ d_{H1}(B,A) \right)$$

… is a true mathematical distance between sets of objects considered for their geometry. Usually, this distance is called the *Hausdorff distance* between sets A and B. As it expresses *remoteness*, it is often used in feature-matching

in image processing (Galton 1997), and for controlling the results of a generalization operation (Hangouët 1995). It is used implicitly by the Douglas & Peucker algorithm: when the greatest distance from polyline to segment is computed, this is indeed the computation of $d_{H1}(\{\text{Polyline}\},\{\text{Segment}\})$, the "A to B" component of the Hausdorff distance…

III. The Douglas & Peucker Algorithm Generalized

The latter observation has suggested the generalization of the Douglas and Peucker algorithm we propose in this paper: the Hausdorff distance may be computed between sets of objects of any kind, not restrictedly between polylines. By merely changing the domain of the input parameters, and using a generic splitting method, while preserving the general structure of the algorithm, we will be equipped with a new tool for automatically approaching any given figure with a not-too-distant approximation. We will first list the structure of the algorithm, then provide a few but important comments as to what "splitting" and "approximating" mean, and also indications on how it may be programmed. Cartographic applications will be given in section IV.

III.1 Douglas & Peucker Generalized

The structure of our algorithm is exactly that of the Douglas & Peucker algorithm: the first part consists in computing the greatest distance from *Figure* to *Approximation*; the second part in checking whether the distance is smaller than a distance threshold or not and in acting consequently (respectively: accepting the *Approximation*, or segmenting the figure into sub-figures with new approximations and applying the algorithm on each sub-figure with new approximation). The major difference between this and the Douglas & Peucker algorithms lies in the nature of the parameters: no limitation is made now as to the geometry of the *Figure* (no longer a polyline only) nor as to that of the *Approximation* (no longer a segment only).

Algorithm DPG (for: Douglas and Peucker Generalized):

Input: $Figure \in \mathcal{P}(O)$, $Proxy \in \mathcal{P}(O)$, $Threshold \in \mathbb{R}^+$

Output: $Proxy' \in \mathcal{P}(O)$,

DPG(*Figure*, *Proxy*, *Threshold*)

$d_{\max} \leftarrow d_{H1}(Figure, Proxy)$ / Hausdorff component from *Figure* to *Proxy* /

$P_{\max} \leftarrow \{\, p \in \Gamma(Figure) \mid d_{Hul}(\{p\}, Proxy) = d_{\max} \,\}$ / Points of the *Figure* farthest from *Proxy* /

if $d_{\max} > Threshold$ / Distance test /

SubConfigs ← SPLITS(*Figure*, *Proxy*, $P_{\max}$) / Relevant subparts and possible approximations /

return $\cup_{(Fig_j, Prox_j) \in SubConfigs}$ DPG(Fig_j, $Prox_j$, *Threshold*) / Recursive application /

else

return *Proxy* / Sufficient Approximation /

end if

end program

III.2 Comments

Splitting

The algorithm makes use of a generic function SPLITS (*Fig*, *Approx*, *Pm*), whose characteristics must be as follows:

Input: $Fig \in \mathcal{P}(O)$, $Approx \in \mathcal{P}(O)$, $Pm \in \mathcal{P}(G)$

Output: a set of couples: (Fig_j, $Approx_j$) such that :

a/ $\bigcup_i Fig_i = Fig$ and $\forall i, \forall j, i \neq j, \quad Fig_i \cap Fig_j = \{\}$

(Through SPLITS, figure *Fig* is entirely segmented into subparts ie. into separate sets of objects).

b/ to each subpart is associated a particular approximation such that :

$\forall i, \quad d_{H1}(Fig_i, Approx_i) < d_{H1}(Fig, Approx)$

… which ensures the convergence of the algorithm.

SPLITS consists in segmenting both the *Figure* and the *Approximation* given particular points indicated in *Pm*. How the SPLITS function may work in deed on the basis of the *Figure*, the *Approximation* and some particular points, is a matter of external expertise. In the case of Douglas & Peucker, *Pm* contains the farthest vertex from *Line* (ie. *Figure*) to *Segment* (ie. *Approximation*), and SPLITS consists both in cutting the line in two at this point, and in recomputing a segment for either section. In the applications of section IV, another SPLITS method is used, which however has the same input/output characteristics.

Starting the algorithm. The choice of the initial *Approximation* with which to start the algorithm is always tricky, because the result will obviously vary accordingly. The classical Douglas & Peucker starts with the two endpoints of the polyline as initial segment; when the polyline is closed, the two points of its *diameter* (the pair of vertices that are farthest apart) are usually chosen. The most reliable general strategy, for DPG, consists in starting with an object or a few objects with a large extension across the map. This is consistent with the "philosophy" of *remoteness* which structures the interpretation of the Hausdorff distance.

Tuning the threshold. Choosing the threshold may appear to be less tricky, being more obviously scale- and purpose-dependent: Scale-dependent because selection aims at freeing space for legible symbolization at smaller scales; purpose-dependent because symbolization itself is purpose-dependent. However, setting right away the best threshold (to say nothing of automatically tuning the threshold), the one for which output images look most harmonious at desired scale and symbolization, is still a matter of luck, not of determinism.

Programming. Voronoi diagrams on points help quickening the exact search for farthest extra point (Kreveld *et al*. 1997); on lines they help quickening the exact computation of the Hausdorff distance between lines (Alt *et al*. 1991) as well as between areas (Hangouët 2000). In the absence of Voronoi tools, approximate computations can be made by discretizing the geometry of the objects: for each object, a few points are made to stand for it, and any distance computation is made on points.

IV. Cartographic Applications

In the various "cartographic" applications shown in this section both to illustrate the algorithm and to prepare the justification of its usefulness for automated map-making, a same, quite simple, SPLITS method is used. It is described in the first next paragraph. It is remarked that this splitting method may lead to a non-recursive version of the DPG algorithm, the formulation of which is given in the second paragraph. Cartographic results are then illustrated in the following paragraphs, on a dot-map, on a line-map, on a land-use map.

IV.1 Approximating

The SPLITS function used here is the most straightforward we could conceive in our eagerness to carry out the first tests for our intuition: SPLITSPRIME(*Fig*, *Approx*, *Pm*) = { ($Fig, Approx \cup \{o_i \in Fig \mid Pm \cap \gamma(o_i) \neq \{\}\}$) }.

Only one couple (*Fig*,*Approx*) is returned, *Fig* being the entire figure (no splitting in fact!), and *Approx* being augmented with objects from *Fig* whose geometries contains the points indicated in *Pm*. This SPLITSPRIME function has the characteristics of a SPLITS function (notably, it is clear that the DPG will converge).

IV.2 Non-Recursive Formulation

As the figure is never split by SPLITSPRIME, integrating this function into the DPG algorithm leads to a non-recursive algorithm. Here is the formulation of the "Non-Recursive Douglas & Peucker Generalized" algorithm used for the cartographic examples to follow:

Input: $Figure \in \mathcal{P}(O)$, $Proxy \in \mathcal{P}(O)$, $Threshold \in \mathbb{R}^+$

Output: $Proxy' \in \mathcal{P}(O)$,

NRDPG(*Figure*, *Proxy*, *Threshold*)

$d_{\max} \leftarrow d_{H1}(Figure, Proxy)$ / Hausdorff component from *Figure* to *Proxy* /

While $d_{\max} > Threshold$ / Distance test /

$P_{\max} \leftarrow \{ p \in \Gamma(Figure) \mid d_{Hu1}(\{p\}, Proxy) = d_{\max} \}$ / Points of the *Figure* farthest from *Proxy* /

$Proxy \leftarrow Proxy \cup \{o_i \in Figure \mid P_{\max} \cap \gamma(o_i) \neq \{\} \}$ / *Proxy* augmented with remotest *Figure* components /

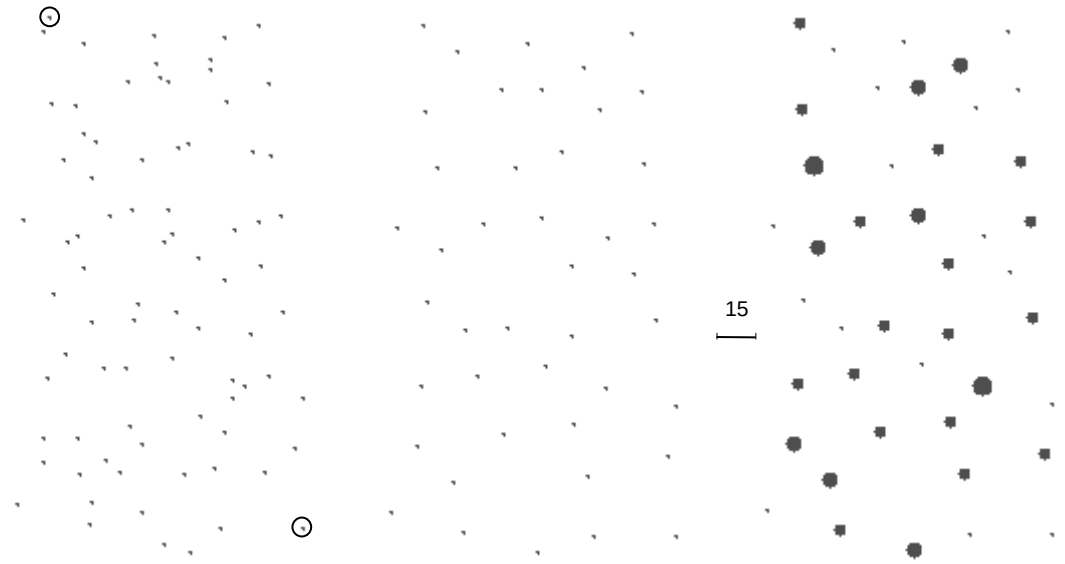
$d_{\max} \leftarrow d_{H1}(Figure, Proxy)$ / Hausdorff component from *Figure* to new *Proxy* /

end while

return *Proxy* / Sufficient Approximation /
end program

IV.3 On Points

Here, the *Figure* is made up of points – it may be a dot map, in cartography. A good initial approximation, consistent with the idea of maximal remoteness engineered by the Hausdorff distance, is the two points of the *Diameter* of the set (ie. the two points on the convex hull lying farthest apart). As illustrated below, the algorithm little by little selects the farthest points until any new point falls within the specified distance from some already selected point.



Initial *Figure* and its *Diameter*

NRDPG(*Figure*,*Diameter*, 15)

NRDPG(*Figure*,*Diameter*, 15) + Dot size proportional to the number of unshown neighbours around.

This may find direct applications in the generalization of dot maps at small scales where "*lack of space requires severe selection*" (Dahlberg 1967:133) and "*much of the locational specificity inherent in the dot* (…) *is lost*" (*id.*:132). Choosing a *Threshold* in relation to the minimal ground distance allowed by legibility at the small scale and applying the algorithm will lead to a both locational-consistent and readable selection of representative dots. As shown in the illustration above, graphic "tricks" may help make the final map even more readable.
A similar method, yet differing somewhat as to the test made (1/ start from the *Diameter* dots, 2/ put them in *Approximation*, 3/ find dot farthest from *Approximation*, 4/ repeat step 3/ until a certain number of dots is reached),

and in fact not recognized for its kinship with Douglas and Peucker, has been designed by (Kreveld *et al.* 1997) for the scale-dependent automatic display of settlement data, and provides agreeable results.

IV.4 On Lines

Here, the *Figure* is made up of lines – it could be a street map. A good initial approximation, because extension is consistent with the idea of maximal remoteness engineered by the Hausdorff distance, is the line with the widest span, or a set of a few lines. In the illustration below, the *Figure* is composed of the 16 major and minor lines of the Parisian métro (this is a rough, self-made sketch only. Official métro plans *are* nice-looking). Let us remark that if the original data format is topological (with nodes and arcs between them), we need to build complex objects composed of arcs to get each metro line, or each street of a town: in other words, topological structure is not necessary.

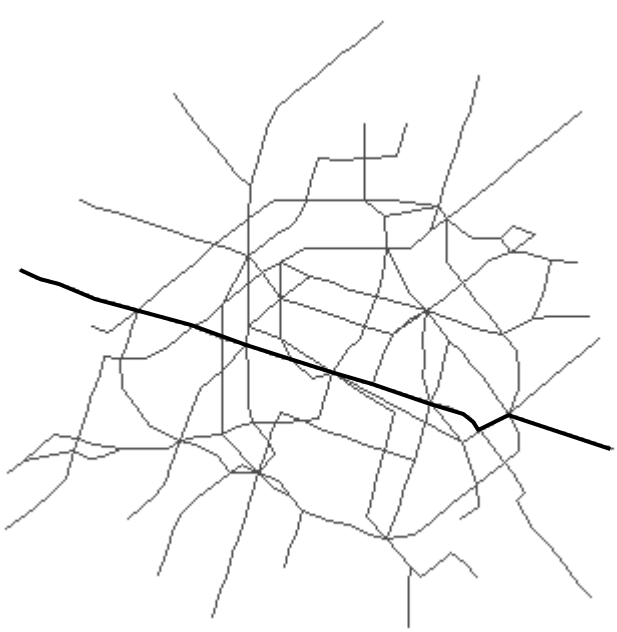
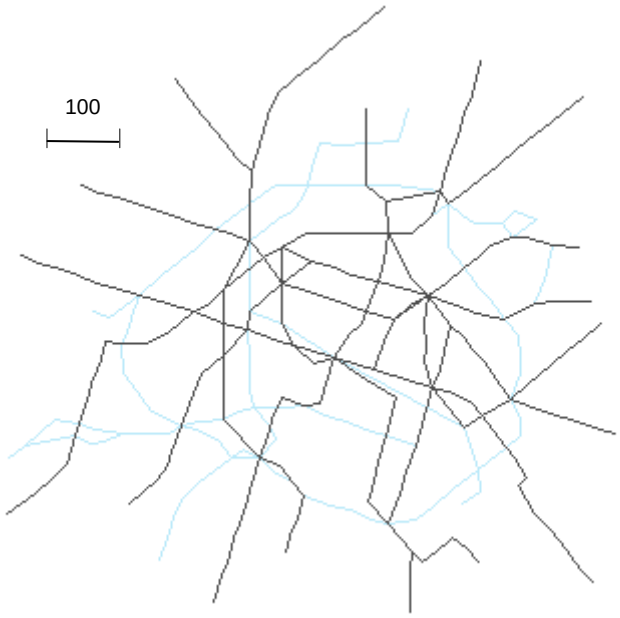


Figure = set of métro lines
Initial approximation = "Ligne 1"



NRDPG(Figure, "Ligne 1", 100)
Mean discretization: 215 points per métro line

It is a bit hard to grasp what happened between these plans, in spite of their similarity: lines are removed in order to minimize the degradation of the servicing of Paris by metro. This process could be compared with spillikins, but with opposite constraints: whereas at spillikins we begin by picking up isolated sticks, the first lines to be "removed" are the ones which are the closest to another one.
This example (metro plan), if a good one to show how NRDPG works, is not very interesting for map generalization: a street plan of a city like Montréal could show better results. Let us remark that, according to the purpose in using NRDPG, it is possible to define differently the distance between two lines, taking into account for instance the area of the convex hull of both lines; it could also be interesting to weight each line according its importance (length, semantic, …).

IV.5 On Areas

Here, the *Figure* is made up of areas – it could be a land-use map. A good initial approximation, for the now familiar reason that extension is consistent with the idea of maximal remoteness engineered by the Hausdorff distance, is the area with the widest-spanning form, or a set of a few areas. In the illustration below, the *Figure* is composed of 295 buildings (from IGN's BDTopo® data on a southern French village), and the initial approximation consists of the

pair of remotest buildings. The discretization used was only one point per building (the centroid of corners). Tests are being elaborated to compare this roughest one-point discretization with discretization with more points and with "Voronoi-exact" results.

Figure = set of buildings (BDTopo® data)
Initial approximation = Diameter buildings

NRDPG(Figure, Diameter, 30)
Mean discretization: 6.7 points per building

Two main reproaches may be made to NRDPG used as a generalization algorithm. First, structural information is not preserved: regular patterns like alignments between buildings (Cf. illustration above, on the left) no longer appear; in fact, from the moment the chosen threshold is equivalent to the mean diameter of the "holes", a homogeneous distribution is obtained and alignments are often lost. Nevertheless, it has to be emphasized that the sensation of space occupation is well kept: the limits and the shape of the town are preserved successfully. The second reproach we can notice in the example above is that nothing prevents the removing of important buildings, either because of their size or their semantic attribute: a solution to that problem could be to include important buildings in the initial approximation or, as it has been said for NRDPG on lines, to assign a big weight to main buildings in distance computation.
In areas in which buildings are more or less equivalent to each other and in which there are no very strong patterns like alignments (typically, residential areas in French Riviera), NRDPG can be used without any modification.

V. Discussion

V.1 What Generality is Claimed for DPG

We have generalized the Douglas and Peucker algorithm to objects of any kind of geometry, yet keeping within some limits as to the kind of test performed and as to the distance used.
Test. The test is performed only on *Distance*. It could also be performed on the quantity of objects selected, compared to a desired selection ratio. The meaning of the selection would be altered; it would no longer be a selection on ground occupation and extension, but would integrate the idea of numeral quantity.
Distance. The component *Figure* to *Approximation* of the Hausdorff distance is used. The full Hausdorff distance could be used, but computation time would be doubled for very limited advantages – as shown in the statistical studies of (Abbas-Hottier 1993), the other component, *Approximation* to *Figure* (*Generalization* to *Original*, in their terminology) is usually shorter than the reverse, and the component *Figure* to *Approximation* statistically is the

Hausdorff distance between the two. Instead of Hausdorff, any other mathematical distance can conceivably be used, yet maybe without conveying the sense of *remoteness* that fits so well with the distance test performed.

V.2 Fields Open for Research

On mixed data. We will have to compare the result of the algorithm on a *Figure* of buildings and streets, and the results of the algorithm performed separately on the buildings on the one hand and on the streets on the other hand.

Splitting. For the moment, results have been produced using the quite simple SPLITSPRIME function - we will have to design and compare different methods for splitting objects and for constructing relevant approximations, methods which are bound to be specific to the geometry, but certainly also to the geographical nature, of the objects involved.

Tuning the threshold. How the "best" threshold may be systematically related to scale and purpose must be investigated, both mathematically and perceptively.

DPG as analysis tool. For quality cartography, the full potential of DPG is not claimed to lie in its raw generalization powers, but as the service it may bring in evaluating not only the presence but the extension too of cartographic matter within the plane, to guide more specific local transformation operators. It is more an upstream analysis tool than a self-sufficing transformation tool, and applications are not limited indeed to automated map-drawing. For example, for geographic data matching, the separate application of the DPG on the different datasets may provide information on the "matchability" of the features of these sets. Also when the two datasets are of different resolutions, the separate application of DPG can help us to define what kind of conceptual generalization operation may be said to be involved (typification, aggregation, regular selection,…).

VI. Conclusion

In his "*Theory of the Cartographic Line*", Thomas K. Peucker expressed his hopes that "*others* (would) *pick up the idea and expand it as part of a growing body of cartographic theory*" (Peucker 1976: 142). Starting, not from the line theory he proposed, but from the innermost characteristics of the algorithm he had designed a few years earlier, this paper has illustrated the fact that the reasons for the success of the Douglas & Peucker algorithm may not rest specifically on the nature of the *line* but on the nature of *whatever things in configuration*, and the algorithm elicits the configuration quite similarly as we human would see it: first the widest-spanning features, then within this span the second-widest-spanning features and so on… That Voronoi diagrams have been found on the way, even if apparently discreetly at the heart of Hausdorff (section III.2) is certainly no coincidence – but a full Voronoi-based spatial theory is still to be constructed…

References

Abbas, I., Hottier, P., 1993, *Contrôle du tracé planimétrique d'une carte – Contrôle ponctuel et contrôle linéaire*, Bilan Recherche'92, Bulletin d'Information de l'I.G.N., pp.30-36

Alt, H., Behrends, B., Blömer, J., 1991, *Approximate Matching of Polygonal Shapes*, Proceedings of the 7th ACM Symposium on Computational Geometry, pp.186-193

Cromley, R., Campbell, G., 1991, *Noninferior Bandwidth Line Simplification: Algorithm and Structural Analysis*, Geographical Analysis, Vol.23, No.1 (January 1991), pp.25-38

Dahlberg, R., 1967, *Towards the Improvement of the Dot Map*, Annuaire International de Cartographie, vol.VII, pp.158-167

Douglas, D., Peucker, T., 1973, *Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or its Caricature*, The Canadian Cartographer, Vol.10, No.2, December 1973, pp.112-122

Douglas, D., Peucker, T., 1975, *Detection of Surface-Specific Points by Local Parallel Processing of Discrete Terrain Elevation Data*, Computer Graphics, Visions and Image Processing, Vol.4, No.2, December 1973, pp.375-387

Galton, A., 1997, *Continuous Change in Spatial Regions*, in *Spatial Information Theory*, Proceedings of Cosit'97, Laurel Highlands, Pennsylvania, USA, October 1997. Hirtle & Frank (Eds.) Springer, Lecture Notes in Computer Science #1329, pp.1-13

Hangouët, JF., 1995, *Computation of the Hausdorff Distance Between Two PlaneVector Polylines*, Proceedings of AutoCarto 12, Charlotte N.C., pp. 1-10

Hangouët, JF., 2000, *Storing Voronoi Diagrams in Geographical Databases*, Proceedings of *GeoComputation 2000* (CDROM ISBN :0-9533477-2-9) University of Greenwich, Chatham Maritime, U.K., August 23-25

Hershberger, J., Snoeyink, J., 1992, *Speeding Up the Douglas-Peucker Line-Simplification Algorithm*, Proceedings of the 5th International Symposium on Spatial Data Handling, Charleston, South Carolina, vol.1, pp.134-143

van Kreveld, M., van Oostrum, R., Snoeyink, J., 1997, *Efficient settlement selection for interactive display*. Proceedings Auto-Carto 13: ACSM/ASPRS Annual Convention Technical Papers, 1997

Peucker, T., 1976, *A Theory of the Cartographic Line*, Annuaire International de Cartographie, vol.XVI, pp.134-143

Plazanet, C., 1996, *Enrichissement des bases de données géographiques: analyse de la géométrie des objets linéaires pour la généralisation cartographique (application aux routes)*, Thèse de doctorat, Université de Marne-la-Vallée

Rimbert, S., 1990, *Carto-graphies*, éditions Hermès

Visvalingam, M., Herbert, S., 1999, *A Computer Science Perspective on the Bendsimplification Algorithm*, Cartography and Geographic Information Science, Vol.26, No. 4, pp.253-270