A Displacement Method for Maps Showing Dense Sets of Points of Interest

Sarah Tauscher and Karl Neumann

Abstract In the past, point data only play a minor role in map generalization, as points are either already the result of generalization or are used for objects which are only shown on large scale maps. Now, with the growing availability of web mapping services the role of point data has changed: Besides route planning, the most common function of web maps is the visualization of user queries for points of interest. The limited size of commonly used displays often results in a smaller scale as would be appropriate for the maps content. The state of the art to resolve occurring cluttered point sets, is on the one hand interactivity and on the other hand the selection of points. Thus, often the available space is not optimally used. Therefore, we propose a displacement method to improve the readability of dense sets of points of interest.

Keywords Displacement · Point symbol · Voronoi diagram

1 Introduction

Nowadays the use of web mapping services to present search results for points of interest (POIs) or to create maps showing the locations of personally relevant places, e.g. favorite places of one's last holiday, is a matter of course. Most often the locations are marked by uniform symbols, but the web mapping services also support the use of different customized symbols. Due to the limited size of commonly used displays, the map scale is often smaller as would be appropriate for the maps content. The state of the art to resolve occurring cluttered point sets, is on the one hand interactivity (pan and zoom) and on the other hand the selection of points. The selection has either to be performed manually or in the case of the presentation of search results it is rather based on the rankings of the results than on spatial criteria. Thus, often the available space is not optimally used.

S. Tauscher (✉) · K. Neumann
Institute for Information Systems, Technische Universität, Brunswick, Germany
e-mail: tauscher@ifis.cs.tu-bs.de

G. Gartner et al. (eds.), *Progress in Cartography*, Publications of the International Cartographic Association (ICA), DOI 10.1007/978-3-319-19602-2_1

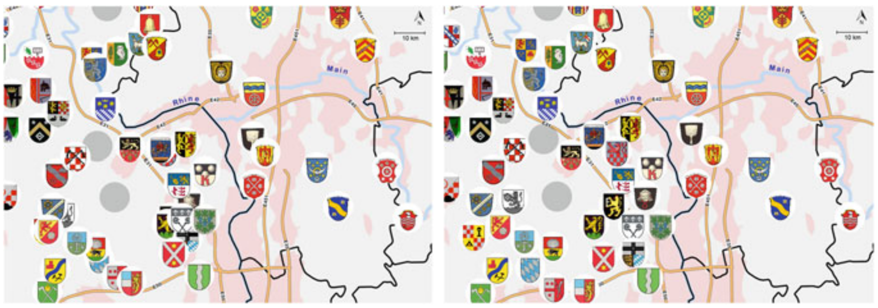

Fig. 1 Emblems of German cities before and after displacement (Parts of base map of Figs. 1, 6, and 10 adopted from Natural Earth)

In order to support the creation of maps, on which not only the location of important places is marked, but also other relevant information is visualized by the map symbols, a displacement is necessary which guarantees the sufficient visibility of symbols and a certain positional accuracy. Figure 1 exemplifies the potential benefit of displacement for a map with individual map symbols. It shows some German cities, which have been symbolized by their emblems. On the left side, the center of each map symbol has been placed on the coordinate of the corresponding city, whereas on the right side the symbols are displaced, but the coordinate of the city remains covered by the symbol.

Surprisingly, there is no established generalization method, which could be readily applied to this use case. The reason for this is that point data only play a minor role in map generalization, in the past. In traditional maps, points are either already the result of generalization (by symbolization) or they are used for objects which are only shown on large scale maps (e.g. real estate maps). Consequently, conflicts arise generally between point and line or areal data and not between different points. Therefore we propose a displacement method to improve the readability of dense sets of POIs, which can be easily applied to web mapping services without expecting further cartographic knowledge from the user.

The rest of the paper is organized as follows: In Sect. 2 we summarize existing displacement methods, before we describe the method we developed in Sect. 3. In Sect. 4 some evaluation results are presented and discussed. A possibility to improve the efficiency of our proposed method for large data sets is described in Sect. 5, before Sect. 6 concludes the paper.

2 Related Work

If features on a map are too close to each other to be distinguishable, a way to resolve overlaps is to displace them. To do so there are two types of displacements methods: incremental improvements and holistic approaches.

In incremental methods often a set of predefined candidate positions is tested and either local search (Mackaness and Purves 2001) or simulated annealing (Ware and Jones 1998) are applied. Ruas (1998) proposed a deterministic greedy algorithm, which evaluates at each step the current worst conflict, which might be a side effect of a previous step and displace the involved objects. Another deterministic approach is the application of a gradient descent procedure in combination with on the fly generation of displacement actions (Lonergan and Jones 2001).

Holistic approaches process multiple map features. Mackaness (1994) applied a cluster analysis in order to detect all features which are involved in a conflict. These features are then spread radially outward from the conflict center. Thus, only the distances between conflicting objects are enlarged without distorting the topology.

Other studies formulate the problem as a system of equations that can be solved using an optimization method e.g. least squares methods (Harrie 1999; Sarjakoski and Kilpeläinen 1999; Sester 2001), finite element analysis (Højholt 2000), or by the use of snakes (Burghardt and Meier 1997).

Bereuter and Weibel (2013) utilized quadtrees in order to support real-time generalization of point data. Their algorithm reallocates points of a quadnode not satisfying cartographic proximity constraints to neighboring quadnodes.

3 Algorithm

Given are a set of geocoded POIs and their circular map symbols of a fixed size for a given scale. To guarantee a certain accuracy of the positioning, we require that the coordinates of each point are inside its corresponding map symbol. Thus, the distance between an original location and the center of its symbol is at most the radius of the symbol. The quality of the placement is measured by the sum of the areas of the intersections of all map symbols. The optimal value is zero, as it means that the symbols do not overlap at all. If different placements achieve the same value ($v > 0$), we consider the placement to be best, where the area of the largest intersection is the smallest. So the strategy is to assign a distinct area to each point, which is large enough to completely contain the symbol. In order to do so, we chose an iterative method, which applies Voronoi diagrams as auxiliary structures. The initial assignment of areas to the POIs is the Voronoi diagram of their coordinates and the center of the symbols are placed on the coordinates. Figure 2 visualizes three different cases, which have to be considered: First, the symbol already fits into the Voronoi cell. Second, the symbol overlaps with its nearest neighbor(s), but the Voronoi cell is large enough, so that the symbol could be placed completely within the Voronoi cell. Third, the Voronoi cell is too small to cover the symbol. In the latter two cases, the symbols have to be displaced and the Voronoi diagram is adapted. Then the new positions have to be evaluated again.

The method stops, if there are no more overlapping map symbols, or if no point can be moved without exceeding the given threshold (radius of the symbols), or if a given number of iterations is exceeded. So the algorithm consists basically of two

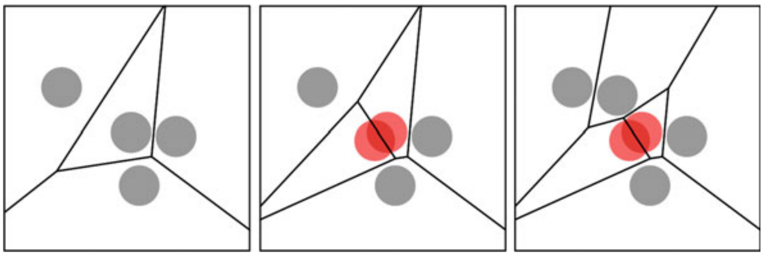

Fig. 2 Initial placement of POIs in Voronoi cells: possible constellations

nested loops (see Algorithm 1). In the inner loop new positions for all overlapping symbols are calculated and in the outer loop the Voronoi diagram is recalculated.

```
1: procedure displacePOIs(point set P, radius r)
2:     $P' \leftarrow \emptyset$
3:     $P'' \leftarrow P$
4:     $iter \leftarrow 0$
5:     while ($P' \neq P''$ and $iter < maxiter$) do
6:         $iter \leftarrow iter+1$
6:         $P' \leftarrow P''$
7:         $P'' \leftarrow \emptyset$
8:         vd ← calculate Voronoi diagram of $P'$
9:         for (each $p \in P'$) do
10:            if (symbol(p) inside vd.cell(p)) do
11:                $P'' \leftarrow P'' \cup \{p\}$
12:            else
13:                $P'' \leftarrow P'' \cup \{displace(p, r)\}$
14:            end if
15:        end for
16:    end while
```

Algorithm 1. Principle of point set displacement.

The quality of the displacement depends wholly on the calculation of the displacement vectors. So, as to improve the initial placement, small Voronoi cells

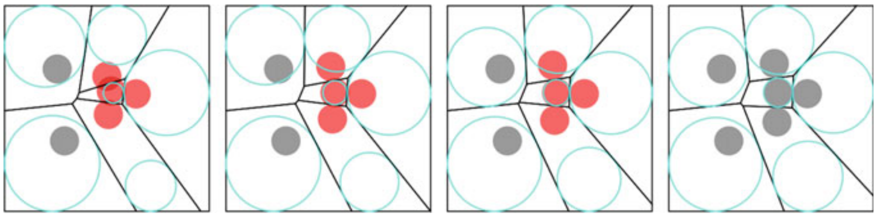

Fig. 3 Iteration one to three and final result applying heuristic (a)

(case 3) should be enlarged and the sites should be closer to the center of their Voronoi cells (case 2). In order to achieve this, we apply two alternative heuristics:

(a) Move point towards the center of a largest inner circle of its Voronoi cell (see Fig. 3). This is a greedy approach, which guarantees that the visibility of the least visible symbol is increased in every step.
(b) Move point towards the centroid of its Voronoi cell (see Fig. 4). This approach might decrease the visibility, but it is possible to overcome local optima.

If the Voronoi cells are elongated, as it is shown in Fig. 5, both heuristics will fail, because of the threshold. This problem can be avoided by restricting the Voronoi cell with a square whose edges measures two times the diameter of the symbol (see Fig. 5 right side).

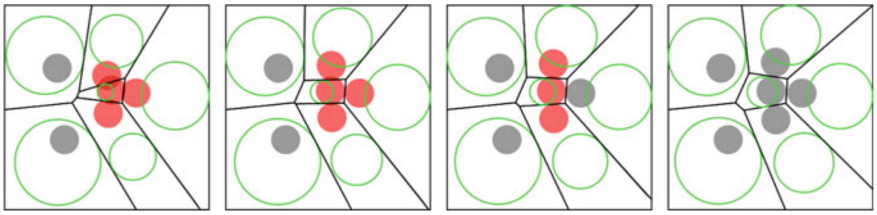

Fig. 4 Iteration one to three and final result applying heuristic (b)

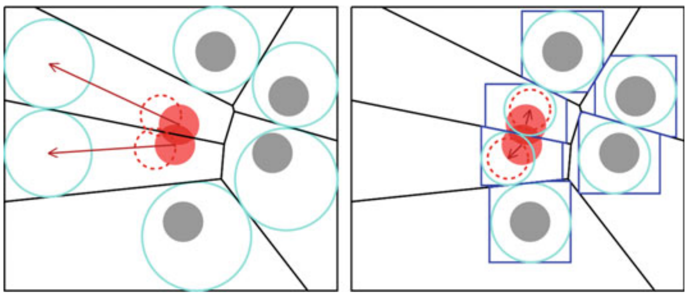

Fig. 5 Application of the threshold to the Voronoi cells

The same problem arises if the displacement result for a small scale is applied to a larger one. Although it is possible to use the displacement vector calculated for a larger scale and the radius of the actual scale to calculate a new valid position of a map symbol, the quality of the result will be decreased, as the direction of the vector is based on a larger restriction than appropriate for the actual scale. Furthermore, each symbol will be placed as far as possible from its POI. In the case that a displacement result of a large scale is applied to a smaller one, some overlaps cannot be resolved as the length of the displacement vectors is restricted according to the large scale radius.

The algorithm as presented above does not handle foreground-background constraints, as the calculation of the displacement vectors is solely based on the Voronoi diagram of the POIs. However, some constraints may easily be included by adding further restrictions that correspond to the background objects. E.g. it is possible to ensure that the center of a POIs symbol remains within the boundaries of a certain area, if the intersection of its Voronoi cell and a convex progressive approximation of the area is used for the calculation of the displacement vector. If the symbol should be completely within the area, the distance between each border point of the approximation and the nearest border point of the area has to be at least the radius of the map symbol.

4 Evaluation

We evaluated heuristic (a) with five datasets (European towns, airports, towers, hills and peaks) derived from Geonames for two different scales, thus resulting in ten test sets. We implemented our method in Java using the Java Topology Suite, an open source Java software library, to model geometries and for the calculation of Voronoi diagrams. In Fig. 6 on the left side the initial placement of symbols for one test set (airports in the smaller scale) is shown and on the right side the result of the displacement.

Table 1 gives an overview of our evaluation experiments using at most 1000 iterations. It lists the size of the point sets, the number of conflicts as well as four values concerning the visible area of symbols before and after displacement.

The results vary depending on the distribution of POIs. Hills, peaks and towers are strongly clustered whereas towns and airports are distributed more evenly across the area. As expected not only the number of initial conflicts increases for clustered data, but also the percentage of unsolved conflicts. Nevertheless the readability of map symbols is greatly improved for all test sets. Korpi et al. (2014) stated that an occlusion level of 25 % for point symbols decreases neither the accuracy nor the efficiency during map reading tasks and that an occlusion level of 50 % only effects the efficiency. Consequently, for five test sets the remaining conflicts do not decline the readability of the symbols significantly, as more than 90 % of each symbol is visible and for two test sets only the efficiency is slightly influenced. For the

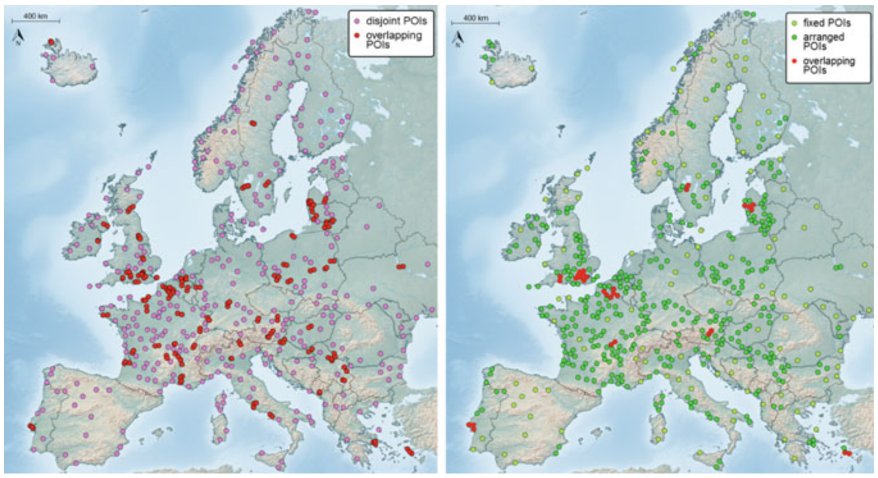

Fig. 6 Airports in Europe, initial placement and displacement result

remaining test sets the number of map symbols which are half-occluded is also reduced by more than 58 %.

Furthermore, we applied kernel density estimation to qualitatively investigate, how well the point distribution is maintained. Figure 7 shows a part of the kernel density map of the airport dataset before (a) and after (b) displacement, as well as a part of the peak dataset in (c) and (d) respectively.

After displacement the areas, where the local maxima of the densities are located, are blurred, their extents are increased and the maximum values are reduced. Nevertheless, the underlying spatial pattern is maintained.

5 Processing of Independent Subsets

One shortcoming of our method is that all POIs have to be considered in each iteration step. To improve the efficiency of our method, it would be desirable to consider only those sites which are actually occluded or might be occluded due to the displacement of other site symbols. Moreover, if the method fails, it is likely that in some areas the point set is too dense to be visualized in the given scale. So it would be beneficial to identify these areas, in order to either perform another generalization operation in advance or to give appropriate information to the user.

As the distance between a displaced symbol and its original coordinates is strictly limited, it is possible to identify fixed POIs, i.e. POIs that do not have to be displaced. Furthermore, the point set can be divided into disjoint subsets which might be processed independently. For both steps a Voronoi diagram can be used as auxiliary structure.

Table 1 Results of evaluation experiments

		Town		Hill		Peak		Tower		Airport	
Scale		1:10 m	1:5 m	1:10 m	1:5 m	1:10 m	1:5 m	1:10 m	1:5 m	1:10 m	1:5 m
# all sites		513	2050	513	2050	513	2050	513	2050	513	1435
#conflicts	Before	250	1094	425	1756	452	1823	334	1476	175	416
	After	87	277	365	1491	399	1575	224	1101	45	75
%visible area of least visible symbol	Before	4.78	1.25	0	0	0	0	0	0	0	0
	After	99.99	99.99	90.04	61.39	47.31	18.91	58.27	5.37	99.99	99.99
%visible area of symbols	Before	92.38	92.29	68.93	63.86	51.75	53.01	67.85	55.95	94.99	95.93
	After	99.99	99.99	99.96	99.47	91.51	83.10	98.17	90.21	99.99	99.99
#symbols w. visible area < ½	Before	31	92	155	714	244	976	142	808	20	33
	After	0	0	0	0	5	410	0	113	0	0
#symbols w. visible area < ¾	Before	53	244	210	1001	297	1201	191	949	28	52
	After	0	0	0	15	91	546	10	338	0	0

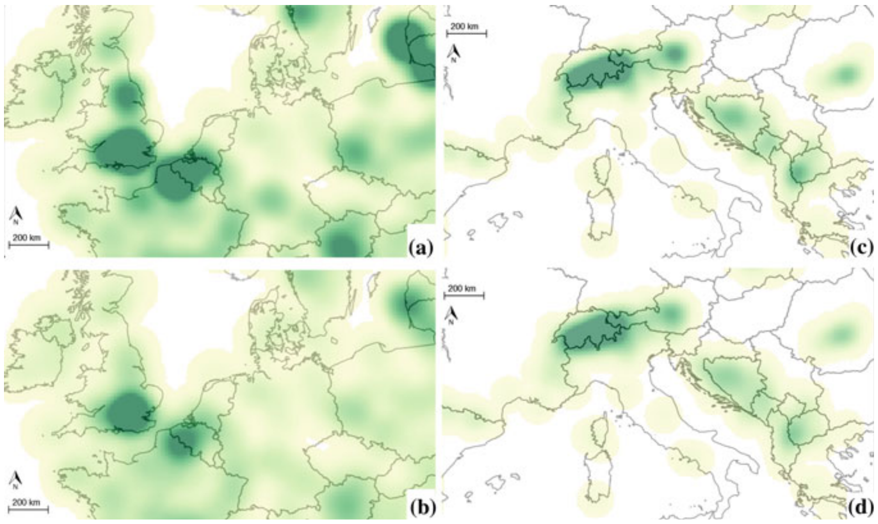

Fig. 7 Kernel density maps: airport/peak before (**a**, **c**) and after (**b**, **d**) displacement

To determine the fixed points, the distance between each POI and its nearest neighbor is calculated. If the distance is larger than two times the diameter of the map symbol, the nearest neighbor (and thus any other point) cannot be displaced in a way that it overlaps with the symbol.

To divide the point set into disjoint subsets, we calculate for each edge the distance to its two sites. If the distance is smaller than the diameter of the map symbol, the symbols might overlap after being displaced. Thus, they have to be processed in one subset and the edge cannot be used to split the Voronoi diagram and therefore is deleted. As an example for the deletions, in Fig. 8 on the left side the remaining edges of the Voronoi diagram of the airports in the smaller scale are shown. Afterwards, the diagram is recursively divided along an arbitrary path leading from one border point to another. In the first step, the border points are the intersection points of the Voronoi diagram and the clip polygon. In the following steps, the points of the path are also regarded as border points. The resulting subsets for the small scale airport test set are shown in Fig. 8 on the right side: The Voronoi cells of the original sites of each subset are filled with a different color and the Voronoi cells of fixed points are white.

In general, the complexity to find one path from one border point to another is linear to the number of edges. In our case the complexity to find all subsets is linear to the number of edges as well. The reason for this is that each edge only has to be considered once, as the following inspection of the only three possible cases, which are illustrated in Fig. 9a–c respectively, indicates:

(a) the search ends at a border point
(b) the path contains a circle (the algorithm stops as soon as it closes the circle)
(c) the path does not end at a border point, because of deleted edges

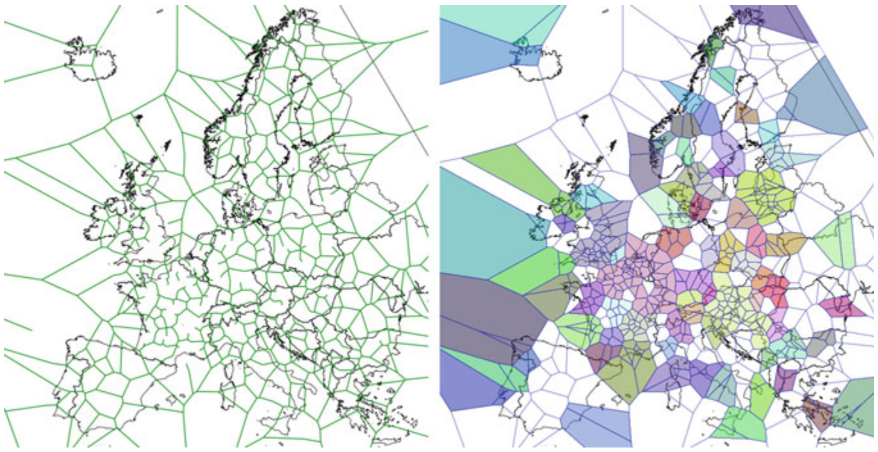
Fig. 8 Airports in Europe, Voronoi edges usable for divisions and resulting subsets

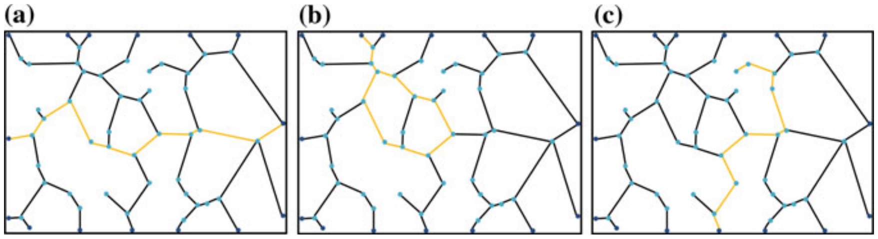

Fig. 9 Three different kinds of paths that might occur during division

In case (a) all visited edges are used to split the Voronoi diagram, i.e. only their endpoints are kept as border points for both divisions and the edges themselves are deleted. Same is true for case (b). The only difference is that the diagram is divided into the part inside and the part outside the circle instead of the part left and the part right of the path. In case (c) the subpath from the last branch to the end will be deleted, and the search continues from the last branch and then being classified again as (a), (b) or (c).

In Table 2 for each of the ten test sets (see Sect. 4) the number of fixed points as well as the minimal, maximal and average number of sites per subset is given.

As expected, the town and airport test sets can be divided into more and smaller subsets, whereas the hill and the peak test sets consists of considerably less subsets and the largest one contains roughly half of the POIs. This difference is exemplified in Fig. 10 showing the subsets for the airport and the peak test set in the larger scale.

Table 2 Characteristics of point subsets

		Town		Hill		Peak		Tower		Airport	
Scale		1:10 m	1:5 m	1:10 m	1:5 m	1:10 m	1:5 m	1:10 m	1:5 m	1:10 m	1:5 m
# all sites		513	2050	513	2050	513	2050	513	2050	513	1435
# fixed points		68	275	33	125	26	67	73	286	126	529
Divisions	#	56	203	12	58	20	66	46	157	79	217
#sites per division	min	2	2	2	2	2	2	2	2	2	2
	max	45	298	389	1492	248	1035	114	432	53	46
	avg	7.95	8.74	40.00	33.19	24.35	30.05	9.57	11.24	4.90	4.18

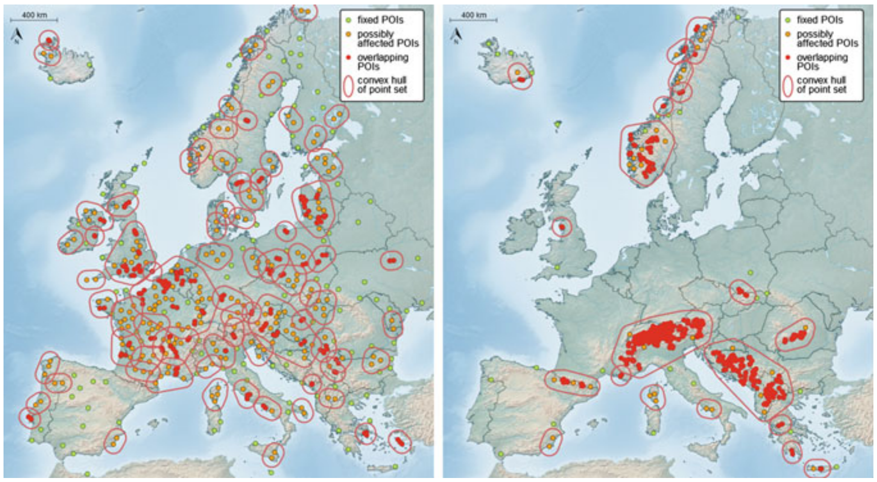

Fig. 10 Division of point sets, airports in Europe and peaks in Europe

The complexity of the algorithm is O(i*n log n), where i denotes the number of iterations and n the number of POIs. Although the complexity is not reduced, the runtime is improved due to several aspects. First, the calculation of a set of smaller Voronoi diagrams is faster than the calculation of a large one, as:

$$n \log n > \sum_{j=0}^{m} k_j \log k_j, \quad with \sum_{j=0}^{m} k_j = n$$

Second, the number of iterations to find the best solution according to our algorithm is reduced significantly for most point sets. The test set with the highest benefit was the airport data set, for which only for 9.16 % of POIs more than 10 iterations are necessary, and the least improvement was achieved for the peak dataset, for which the algorithm still iterates more than 10 times over 54.64 % of POIs.

Additionally, the division into subsets allows an easy parallelization leading to a further reduction of its runtime. Due to the iterative nature of the method on-the-fly behavior without the division into subsets is only possible for small and fixed values of i, which leads to worse displacement results. Preliminary tests show that on-the-fly behavior even for large numbers of iterations is probable, if the optimizations described above are applied. However, further investigations concerning the influence of characteristics of point sets on the runtime as well as on the quality of the displacement are needed.

6 Conclusion

In this paper we presented a method for the displacement of POIs in the context of user generated maps. In contrast to existing methods, the available space for each map symbol is modelled by cells of a Voronoi diagram. The POIs are iteratively moved within their Voronoi cell, with the objective to improve not only their visibility but also to achieve a more evenly distribution of the available space among adjoining POIs. Additionally, a threshold is introduced and directly applied to the Voronoi cells in order to guarantee a certain positional accuracy. We choose the radius of the map symbol as threshold, so that the map symbol still covers the original position of the POI. We proposed two different heuristics to determine the direction of movement of a POI and evaluated one on ten real world datasets. Moreover, we suggest a linear method to divide a point set into disjoint subsets that can be processed independently of each other. These subsets improve not only the efficiency of the algorithm, but they also allow identifying regions, which are too dense to be presented in the given scale.

The displacement algorithm can easily be adopted to handle circular map symbols of different size by using a power diagram (Aurenhammer 1987) instead of a regular Voronoi diagram. Moreover, it is possible to use our method to displace elliptical symbols more accurately than approximating them with circles as exemplified in Fig. 11. First, the transformation to convert the ellipse into a circle is determined and applied to the underlying map space. Second, the POIs are displaced by the algorithm described in Sect. 3. Finally, the inverse transformation is applied to the displacement result to restore the original map space.

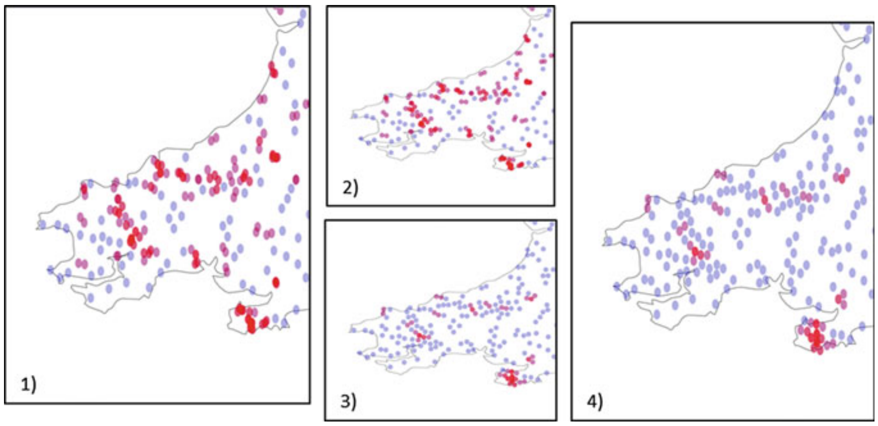

Fig. 11 Displacement of elliptical symbols. **1** Input. **2** Transformed map space. **3** Displaced POIs. **4** Restored map space

References

Aurenhammer F (1987) Power diagrams: properties, algorithms and applications. SIAM J Comput 16(1):78–96

Bereuter P, Weibel R (2013) Real-time generalization of point data in mobile and web mapping using quadtrees. Cartographic Geogr Inf Sci 40(4):271–281

Burghardt D, Meier S (1997) Cartographic displacement using the snakes concept. In: Proceedings of the Semantic modeling for the acquisition of topographic information from images and maps, Birkhauser Verlag, pp 59–71

Harrie L (1999) The constraint method for solving spatial conflicts in cartographic generalization. Cartography Geogr Inf Sci 26:55–69

Højholt P (2000) Solving space conflicts in map generalization: using a finite element method. Cartography Geogr Inf Sci 27(1):65–73

Korpi J, Haybatollahi M, Ahonen-Rainio P (2014) Identification of partially occluded map symbols. Cartographic Perspectives, (76):19–32. doi:10.14714/CP76.59

Lonergan ME, Jones CB (2001) An iterative displacement method for conflict resolution in map generalization. Algorithmica 30(2):287–301

Mackaness WA (1994) An algorithm for conflict identification and feature displacement in automated map generalization. Cartography Geogr Inf Syst 21(4):219–232

Mackaness WA, Purves R (2001) Automated displacement for large numbers of discrete map objects. Algorithmica 30:302–311

Ruas A (1998) OO-constraint modelling to automate urban generalization process. In: Proceedings of the eighth international symposium on spatial data handling, Vancouver, Canada, 12–15 July 1998, pp 225–235

Sarjakoski T, Kilpeläinen T (1999) Holistic cartographic generalization by least squares adjustment for large data sets. In: Proceedings of the 19th ICA/ACI conference, Ottawa, Canada, pp 1091–1098

Sester M (2001) Optimization approaches for generalization. In: Proceedings of geographical information systems research, University of Glamorgan, Wales, 18–20 April 2001 pp 32–35

Ware JM, Jones CB (1998) Conflict reduction in map generalization using iterative improvement. GeoInformatica 2(4):383–407