GeoInformatica 9:2, 181–204, 2005
© 2005 Springer Science + Business Media, Inc. Manufactured in The Netherlands.

Construction of the Planar Partition Postal Code Map Based on Cadastral Registration

FRISO PENNINGA,* EDWARD VERBREE, WILKO QUAK AND PETER VAN OOSTEROM
Delft University of Technology, OTB—GIS Technology, Jaffalaan 9, 2628 BX Delft, The Netherlands
E-mail: {F.Penninga, E.Verbree, W.Quak, Oosterom}@otb.tudelft.nl

Received May 17, 2004; Revised December 23, 2004; Accepted January 28, 2005

Abstract

Accurate postal code maps have many applications within GIS as the postal code has the potential to link the address description of buildings to their location in a specified global reference system in a more natural way. This relationship is possible in both directions: geocoding and reverse-geocoding. These operators demand a mechanism for translating an exact geometric position (i.e., WGS84 coordinate) into a location indication (town, street, house number) and vice versa. As most built-up parcels are provided with a postal code, this indicator can be used as the linkage. This paper describes the procedure, based on the Dutch cadastral registration, to obtain a reliable 6-position (i.e., 2628BX, the highest level of detail possible) planar postal code map for the Netherlands. Problems with existing, Voronoi-diagram based, postal code maps, like intersected houses and arbitrary derived (and thus unrecognizable) boundaries are avoided. The reliability of the derived planar postal code map is discussed and results are illustrated by figures. For a planar coverage, non built-up parcels having no postal code should be assigned a plausible postal code. Furthermore special attention is given to infrastructural parcels. These parcels are divided at their (approximated) skeletons first and then these subdivided infrastructure parcels are piecewise attached to their neighbour parcels. This new approach results in very reliable postal code maps, which are visually attractive too as infrastructure lines can be recognized. The procedure is generic and can be applied to other administrative parcel information as well. The algorithm is implemented using the Computational Geometry Algorithms Library (CGAL), and the possibilities and limitations of this library are addressed as well. Also a number of non-implemented alternatives or improvements are given.

Keywords: skeletonization, reverse geocoding, aggregation, postal code map, algorithms, applications

1. Introduction

Postal code maps could be used within GIS as they can link address descriptions of houses and offices to their location in a geometric reference system. This link is bidirectional: reverse-geocoding is used to get a certain address specified by a coordinate and geocoding is used to allocate a coordinate to a given address. As in most other countries the postal code is introduced in the Netherlands for efficiency in postal services. By demanding a

* Corresponding author.

uniform description of identification of houses and offices a more automatic dispatching of letters could be developed. The Dutch postal code consists of four digits and two characters, e.g., 2628BX. This so-called 6-position postal code is assigned to an average of 16 neighbouring postal addresses, or part of a street. In comparison with other countries the Dutch postal code system is relatively dense.

By the combination of a postal code with a house number each house or office can be identified uniquely. For instance the location of the GIS Technology group at Delft, the Netherlands, is identified by 2628BX9. The postal code could accordingly be used as an address identifier. However, the actual position of the postal address, and thus the postal code, is not obvious. The 'centre' of a house identified by the postal address could be considered as the location of the postal code, but this derived point location of the postal code will be only a 'best guess.' That is because the coordinates are mostly arbitrarily allocated or calculated (i.e., the centroid) within the boundary of the building. And how to construct a 6-position planar postal code map out of several postal address locations? One approach is to take the average location of all address coordinates with the same postal code and use these points as input for the computation of a Voronoi diagram. However, this can result in serious consequences regarding the readability: these postal code areas will intersect with built-up parcels and do not always consider boundaries like roads and rivers that restrict postal code areas in a natural sense (as shown in Figure 1).

To solve these problems an approach [9] based on the cadastral registration is proposed and implemented. The 6-position postal code is now the efficient intermediary between an address and its location. Given a specific coordinate the reverse-geocoding algorithm determines by a point in polygon search firstly the postal code area and given the postal code it will subsequently determine the parcel within the postal code area. This parcel points through the cadastral registration to the proper house description by town,

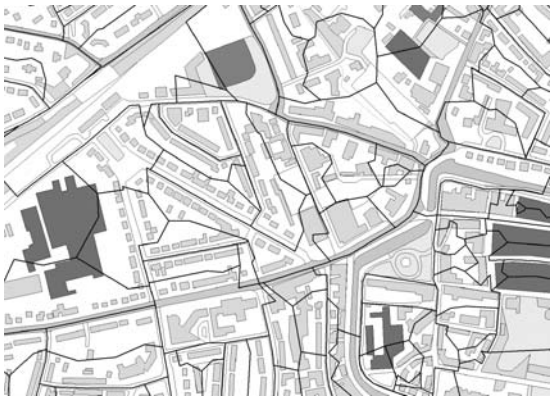

Figure 1. One of the existing postal code maps: (partly) Voronoi-diagram based polygons intersect with buildings which results in bad postal code areas (source: Bridgis).

address and number of the house. To the opposite the geocoding algorithm will now use the parcel as mean to obtain the proper 6-position postal code polygon. This method works without any problems for built-up parcels. The buildings are not only described by town (municipality), address and house number, but also by a postal code provided by the postal services. The parcels, which are not accommodated with a building, do not have a postal code. For a full coverage the most probable postal code should be attached to the remaining non-built parcels as well. To reflect the need to take the boundary of the infrastructural objects into account a procedure is developed where these are divided at their (approximated) skeleton (medial axes diagram) first. The remaining parts are then aggregated to their most likely neighbour parcel and thus postal coded. The results of this procedure are in line with the fact that in practice addresses on one side of the road have different postal codes than addresses on the opposite side of this road. The final result will be a planar map that partitions the full coverage into non-overlapping adjacent polygons, where each polygon identifies a certain postal code and where the centre-line of the infrastructural objects are on the boundary of these postal code polygons. This idea is illustrated by Figure 2.

This paper describes this procedure—based on the Dutch cadastral registration—to obtain a reliable planar partition postal code map for the Netherlands. The map is also more useful and attractive because of the expected coincidence of the created postal code polygons with the centrelines of the infrastructure. The complete procedure from the skeletonization to the assignment of the polygons has been programmed with the C++ language using the Computer Graphics Algorithms Library (version 2.4). CGAL [1], [4] is a collaborative effort of several universities and research centres in Europe and Israel. The goal is to make the most important of the solutions and methods developed in computational geometry available to users in industry and academia through a C++ library. Our algorithm extensively uses the ‘Planar Map’ and ‘Constrained Delaunay Triangulation’ classes of CGAL. Within the Planar Map, the ‘Trapezoidal Map’ is used

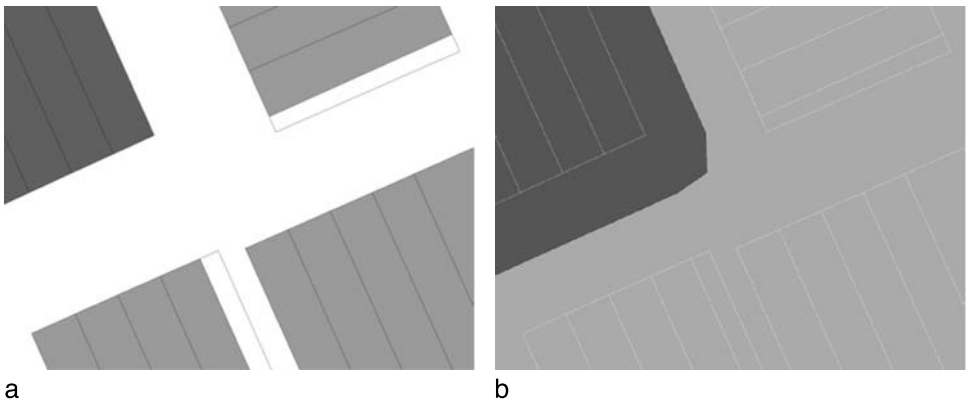

Figure 2. (a) Cadastral situation, two different postal codes. (b) Desired postal code regions are present (parcels without postal code are white).

to locate points within the Planar Map. Our aim in using CGAL was besides implementing and testing our algorithm also to explore the possibilities of CGAL and to answer the question whether CGAL compete with commercial GIS packages. A final conclusion could not be drawn, but it is clear that commercial GIS packages are more user friendly. Another drawback is that CGAL does not (yet) offer a conformal TIN. However, in general, if you have a spatial problem that cannot be handled within a GIS, either for performance or functionality reasons, CGAL will be a good option to look at.

This paper is organised as follows. It starts in Section 2 with the introduction of the algorithm. First results from the algorithm are analysed in Section 3. Based on these results several alternative approaches are suggested in Section 4. The paper ends with Section 5, which summarizes the main results and gives further suggestions for alternative or improved computations of the postal code areas (not implemented).

2. Creating the planar partition postal code map

The algorithm, presented in this section, is based on the assumption that in order to build postal code areas with clear-cut, in real life identifiable boundaries, one has to use an input data source that contains both postal code data and distinguishable boundaries. The cadastral registration meets both requirements, as an object address is available for built-up parcels and parcel (i.e., ownership) boundaries are often visible in the terrain. Although a lot of parcels are built-up (and therefore a postal code is known), also a large number of parcels have no postal code attached. As our approach to address matching requires a postal code map that covers the entire plane, it is necessary to attach a postal code to parcels where no postal code is given yet. As a last step all parcels with the same postal code can be merged to form postal code regions. If these operations are performed on infrastructure like roads an undesirable situation occurs. Many houses (and thus parcels) located on one side of the road have a different postal code in comparison to the houses on the opposite site. Adding a single postal code to the road will result in strange-looking (or perhaps even wrong) maps. As a result reverse-geocoding for point locations on the road will be quite difficult to perform. Intuitively the boundary between the two postal code areas is located at the road centreline. Therefore the left part of the road should be assigned to the postal code area at the left side and the right part of the road to the postal code area at the right side. To locate the road centrelines this algorithm uses skeletonization. Skeletonization was originally developed as a generalization technique to derive a representation of an object with a lower dimension [14], usually representing polygon shaped road features as lines. Its results are very close to the results of the collapse operation as used in a raster based approach [12]; only road ends are treated differently. The road centrelines obtained from skeletonization are now used to split road parcels in order to add these separate parts to different postal code areas. Within the Dutch cadastral registration a system of 80 different culture codes indicates whether a parcel might be built up, or is used as infrastructure (roads, etc.) or the parcel is in fact water (rivers, canals), etc. Based on this code it can be decided whether a parcel has to be

skeletonized or not. Below a summary of the main steps of the proposed algorithm to create a postal code map (next subsections 2.1 to 2.4 will refine these steps):

```
Main algorithm create postal code map:
M1. (L) Loading cadastral map data into the CGAL Planar Map
M2. (S) Skeletonization of infrastructural parcels
M3. (A) Assigning postal codes to the skeletonized and unas-
        signed parcels
M4. (U) Merging the parcels to postal code regions
```

2.1. Loading source data into the CGAL Planar Map

```
M1. (L) Loading source data into the CGAL Planar Map
    L1. Read edges from LKI and insert segments into CGAL topo-
        logical map
    L2. Read parcel data from AKR and relate to faces in CGAL
```

In the Dutch cadastral system administrative data such as ownership, culture code and build-up code of parcels are registered in the AKR[1]-system and the geometrical counterpart such as boundaries (in a winged edge data structure) are stored in the LKI[2]-system, see [8]. Both parcel and boundary data are needed from the LKI as input for the calculation process. From the LKI parcel data attribute data like object_id and a location point (a 2D internal point inside the parcel) are retrieved. Due to the chosen winged edge data structure the actual geometry of a parcel is stored in the LKI boundary tables. These boundaries are stored as polylines. As both cadastral parcels and postal code regions form planar partitions, CGAL's planar map data structure [1] is chosen during implementation. CGAL considers the planar map as an embedding of a topological map into the plane. A topological map is a graph that consists of vertices (nodes), edges, faces and an incidence relation on them. As a result a planar map is created by inserting edges, whereas one might expect a planar map to be constructed by inserting polygons. Therefore as a first step, step L1, all boundaries as retrieved from the cadastral source data are inserted into the planar map structure, resulting in faces similar to the cadastral parcels.

The AKR parcel data holds the postal code in case of built up parcels, as for each built up parcel an AKR object address is mandatory, including a postal code. If the parcel does not have an AKR object address no postal code is known. Or, if the parcel has multiple AKR object addresses, this may also result in multiple postal codes. The last situation (i.e., a large apartment complex) is not discussed further in this paper. In this implementation a separate region with multiple postal codes is used, while 'better' solutions, trying to subdivide this region into postal code areas with exactly one postal code, are described in the recommendations. Based on the AKR object addresses a list linking one or more postal codes to each built up parcel id is created. By querying each

parcel's culture code it is determined whether the parcel contains infrastructure and thus has to be skeletonized or not. These attributes need to be attached to the faces created in the CGAL Planar Map as step L2. However, as these faces are constructed by the insertion of edges, parcel attributes cannot be attached directly to the faces. After completing the construction of the Planar Map the internal location point attribute from the LKI parcel data is used to identify for each parcel its corresponding face and to add the proper parcel attributes to this face.

2.2. The skeletonization of infrastructural parcels

```
M2. (S) Skeletonization of infrastructural parcels
    S1. Select all infrastructure parcels (faces)
    S2. For all selected faces do
        S2.1. Compute constrained Delaunay Triangulation
        S2.2. Count constraint edges for all triangles inside
              infra parcels
        S2.3. Calculate skeleton segments based on number of
              constrained edges
        S2.4. Calculate additional skeleton segments for mul-
              tiple split
        S2.5. Merge skeleton segments and split infrastructure
              parcel
```

As all data is now available in the planar map data structure, the next step is to calculate skeletons of all infrastructural parcels in order to be able to assign different postal codes to both halves of the roads. Using skeletons to find road centrelines is suggested amongst others by Oosterom [7]. Selecting infrastructure parcels in step S1 is easy as each face in the Planar Map has an attribute indicating whether it has to be skeletonized or not. Each infrastructure parcel is skeletonized separately and the skeletonization will be performed using a constrained triangulation. Therefore a constrained Delaunay triangulation is created for each infrastructure parcel in step S2.1, where each constrained edge is part of the parcel boundary. The Delaunay criteria were used to create triangles as equilateral as possible, as this improves skeletonization results. Skeletonization is performed on each parcel using the method of DeLucia and Black [3]. This method was preferred to the skeletonization method of Chithambaran [2], as Chithambaram was not useful for our purpose. Although Chithambaram describes his method's basic principle as a collapse of the polygon's boundaries, it is implemented as a collapse-like operation on the points spanning the polygon. The skeleton is composed of parts of the voronoi diagram calculated from these points. As a result the shape of the skeleton depends on the configuration of the polygon's points and not on the polygon's actual boundary shape. The method of DeLucia and Black does not have this shortcoming as its resulting skeleton is composed of parts lying parallel to its outer boundary segments. Figure 3

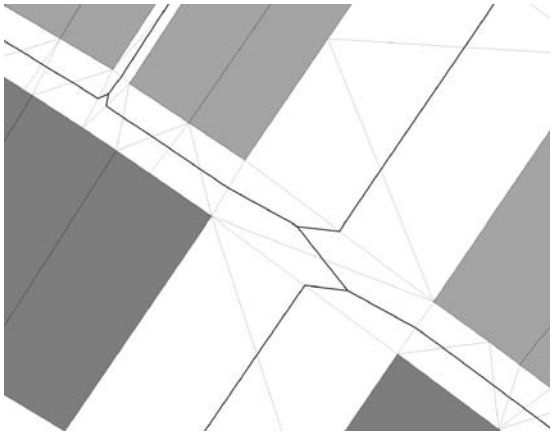

Figure 3. Skeletonized T-junction (upper left) and crossing (centre).

shows an example of the results of this method. It should be noted that both methods give a straight-line based approximation of the true skeleton. The true skeleton is defined by the fact that the distance to both sides are equal for every point on a skeleton segment. Due to this definition the true skeleton is composed of both straight-line segments and parabolic curved segments.

The DeLucia and Black method is parameterized by the number of constrained edges in a triangle, which is determined in step S2.2. Three separate cases can be distinguished and these cases are illustrated in Figure 4. It visualises triangles with two, one and no constrained edges (thick black edges) and their skeleton segments. A triangle with two constrained edges is called a 2-triangle and is split by the skeleton into two parts, where the skeleton segment is defined as the line between the midpoint of the non-constrained edge and the node shared by two constrained edges. A triangle with one constrained edge, a 1-triangle, is split into two parts by a skeleton segment connecting the midpoints of the two non-constrained edges. Internal triangles with no constrained edges, the 0-triangles, are split in three parts by skeleton segments connecting the midpoints of each non-constrained edge with the triangle centre. The creation of a skeleton segment for each triangle is performed in step S2.3. In practice the 1-triangle is the most common type. A 0-triangle is related to a junction, as can be seen in Figure 3 (none of the triangle edges is part of a parcel boundary). 2-triangles occur at road ends.

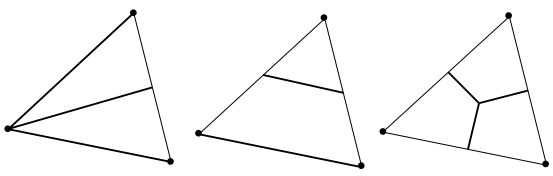

Figure 4. 2-, 1- and 0-triangle with their skeleton segments.

With the skeleton resulting from the previous step it is now possible to split an infrastructure parcel at its centreline. In Dutch practice opposite sides of a road have different postal codes and with the centreline split it is possible to assign half of the road to one postal code and the other half to the other postal code. However the possibility exists that at one side of a road multiple postal codes exist. In this situation one wants to split the infrastructure parcel in more than two parts. To enable this, additional skeleton segments are required and they are calculated in step S2.4. With this additional skeleton branches an infrastructure parcel is split, see Figure 5. As one can see in the lower left of this figure, sometimes more than one postal code appears at one side of the road (light en dark grey parcels). As one wants to add the part of the road directly in front of the light grey parcel and the rest of the road to the postal code region of the dark coloured parcels, it is necessary to split the road half into more separate pieces using skeleton branches.

The algorithm of this step (S2.4) is given below in a little more detail.

```
S2.4. Calculate additional segments for multiple split
      S2.4.1 Determine if and where additional segments are
             needed
             - node is topological node
             - node not already connected to skeleton by
               segment
             (- optional: node lies on postal code boundary
               (better))
      S2.4.2 Compute addition skeleton segment from 'problem'
             node
             - select shortest incident triangle edge in face
             - use this triangle edge from node to 'mid-
               point' on edge

      S2.4.2 alternative to compute additional skeleton seg-
             ment (better)
             - find closest point (w.r.t. node) on current
               skeleton
             - connect closest point and node to create the
               additional segment
```

Step S2.4.1 is to decide whether an additional skeleton branch is needed between the skeleton and a point on the polygon boundary. Therefore the type of point has to be determined. This point can be a topological node, i.e., it defines also another boundary (at least three parcels touch in this point) or it is a point that solely contributes to the shape of the polygon, but has no topological meaning. A skeleton branch is needed in the first case only and whether this is the case is determined by counting the number of edges spanned by the point. If this number equals two the point is a shape point, if this number is larger the point is considered to be a topological node. Note that in our implementation a skeleton branch is added to every topological node, irrespective of the question whether

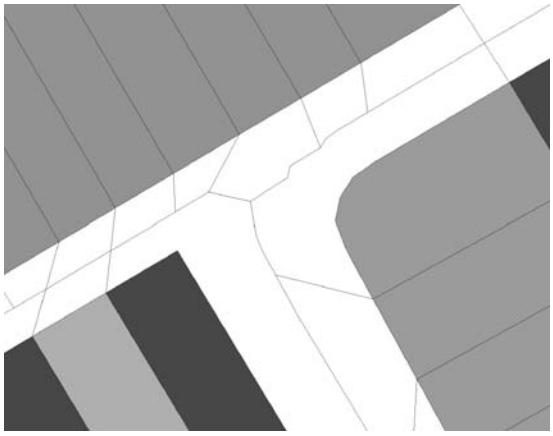

Figure 5. Additional skeleton branches enable splitting infrastructure into more separate pieces.

this point actually is part of a postal code region boundary. For means of simplicity testing whether the edge separates parcels with different postal codes is omitted, whereas it is possible to reduce the number of branches and thus the number of newly created faces by testing the necessity of the skeleton branch. The additional skeleton branches calculated in step S2.4.2 connect the node on the polygon boundary with the midpoint on one of the internal triangle edges spanned by the point. These branches are inserted together with the original skeleton into the planar map, thus creating several new faces by splitting the original one. An alternative approach for step S2.4.2 is not reusing the triangle edges (which results in possible 'fuzzy' splits), but to find the closest point on the skeleton with respect to the 'problem' node and add this as a skeleton segment. As all skeleton segments from applying DeLucia and Black and the additional skeleton segments are calculated these complete skeletons can be used to split (step S2.5) infrastructure parcels into multiple parts. All attributes of the old infrastructure face are transferred to the new faces to enable one to trace back the origin of these parts.

2.3. Assigning postal codes to the skeletonized and unassigned parcels

```
M3. (A) Assigning postal codes to the skeletonized and unas-
        signed parcels
    A.1 Until all faces have postal code do
        A.1.1 For every face currently without postal code
              - Find neighbours with postal code
              - If neighbours with postal code exist:
                - Compute sum of length of common boundaries per
                  postal code
                - Assign postal code with largest sum
```

The infrastructural parcels are now split into separate parts. These parts need to be assigned with the most likely postal code of one of the neighbours. The same holds for non-infrastructure parcels without an AKR object address. It is obvious that the assigned postal code should be based on the postal codes of the surrounding parcels, as a postal code represents a clustered collection of addresses. Several implementations based on this neighbourship relation are possible, e.g., selecting the postal code that most neighbouring parcels have. Our implementation uses another criterion as it selects on the largest (sum of) shared boundary length. Therefore for each neighbouring postal code the sum of lengths of the common boundaries is calculated. Note that different neighbour parcels can contribute to the same postal code. As not all (parts of) parcels without postal code have neighbours with a postal code, the procedure will repeat until all parcels have adopted a postal code, shown in Figure 6. This iterative procedure can be considered as a parcel-based region growing operation.

A disadvantage of this procedure is the possibility to create two or more disjoint areas with the same postal code. Although this is an unwanted situation as it complicates for instance geocoding (which area to choose given a certain postal code), this result might actually fit in perfectly with the real world situation.

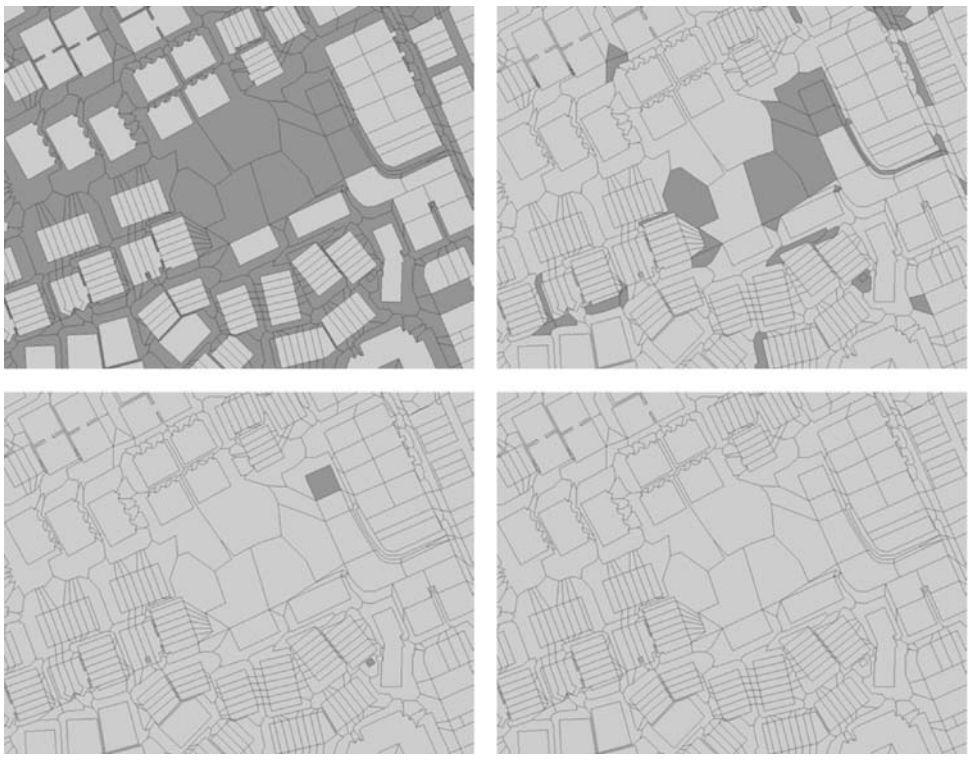

Figure 6. Iterative assignment of postal codes. In each phase more unassigned parcels (dark grey) get a postal code.

2.4. *Uniting the parcels to postal code regions*

```
M4. (U) Uniting the parcels to postal code regions
    U.1 Remove all edges with same postal code on left and right
        side
```

As all (parts of) parcels now have a postal code, the postal code regions can be formed by merging all parcels with the same postal code. Since the planar map data structure in CGAL is edge oriented, uniting parcels is implemented by removing all edges with the same postal code at both sides. This results in postal code regions bounded by recognisable features such as roads and hedges or henches placed on ownership boundaries. As buildings are rarely crossing these ownership boundaries, the new postal code region boundaries will, in contrast with current postal code maps, intersect buildings only in exceptional cases. The complete process from cadastral situation to postal code regions is graphically summarized in Figure 7.

The created postal code regions contain both parcels that originally had a postal code and parcels without postal code. In rural areas the number of parcel without postal code is relatively high, thus leading to postal code regions that are less reliable. In order to get an idea of the spatial distribution of this reliability a map is drawn in Figure 8. For this map a quality indicator is calculated as follows:

Quality indicator $Q = \frac{1}{n}\sum_{m=1}^{n}\frac{1}{\sqrt{i_m + 1}}$ with number of parcels n and number of iteration steps i

3. First results

The planar partition postal map algorithm is tested with a data set of the cadastral municipality of Zevenhuizen, in the southwest of the Netherlands. To show the possibilities of the proposed method a part of the generated postal map of Zevenhuizen, combined with the buildings layer from the topographical map, is shown in Figure 9.

Figure 9 clearly indicates the connection between the houses and their postal code. If one compares the shape of the postal code areas between Figure 1 and 9 it can be seen that the proposed new method results in more meaningful postal code regions as most boundaries coincide with roads and houses almost never intersect with parcel boundaries. In some exceptional cases houses are intersected by the postal code boundaries, as can be seen in Figure 10. The house in the southwest and the one in the north are build upon a parcel that has been incorrectly classified within the AKR-registration as being not built-up. The one building is a public school and the other is the town hall and both buildings are located on a very large parcel owned by the local authorities. Besides these buildings this parcel also contains about ten different roads and the village square and therefore this parcel is marked as infrastructural parcel. As a result the parcel is split and the skeleton unfortunately intersects these buildings. However this is only the case with a few of the about 1500 buildings in this test data set.

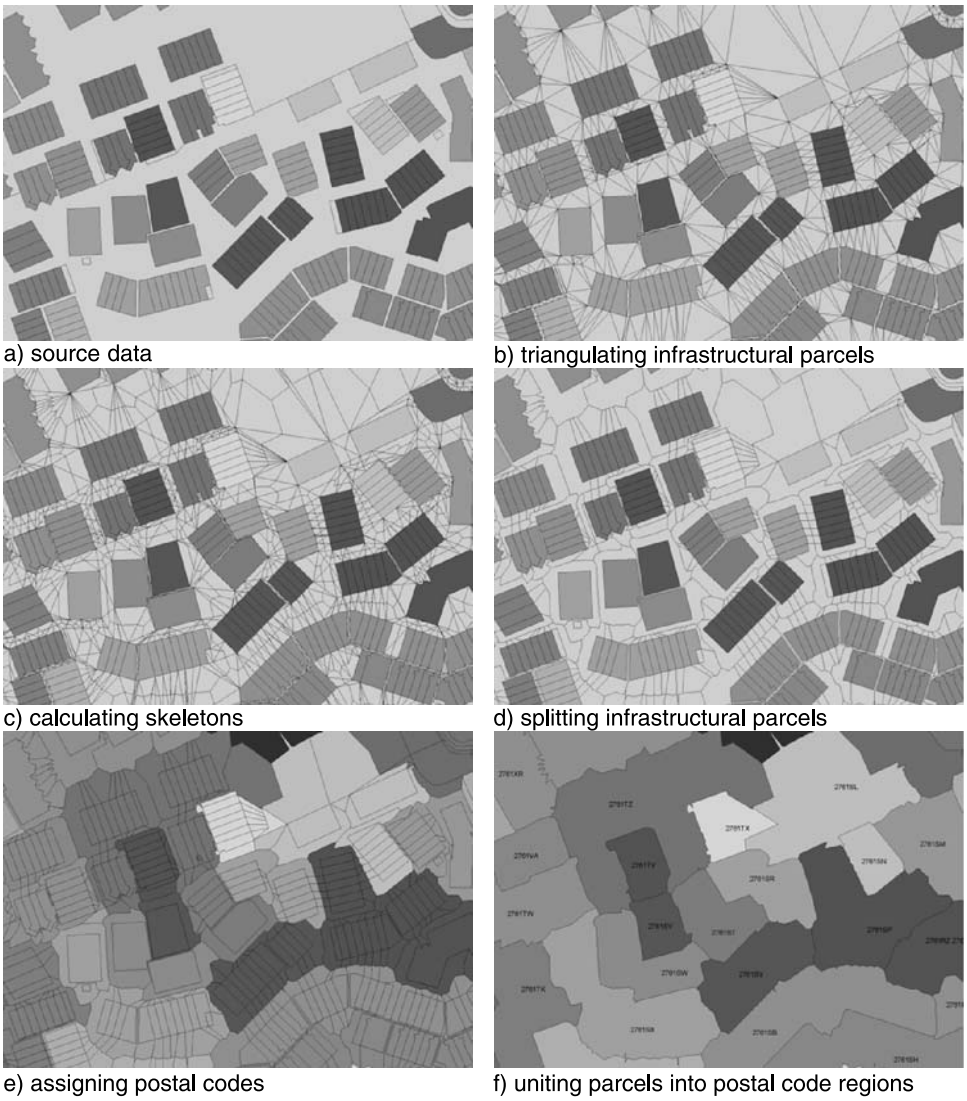

Figure 7. The algorithm in steps: a) source data; b) triangulating infrastructural parcels; c) calculating skeletons; d) splitting infrastructural parcels; e) assigning postal codes; f) uniting parcels into postal code

To review the overall results two different aspects have to be judged. In the first place the results have to be judged semantically. Some of the postal code regions consist of two or more disjoint areas due to irregular street patterns in urban areas and extremely large parcels in rural areas by which another postal code is located between the two disjoint areas. Also postal code regions with more than one assigned postal code appear,

Figure 8. Relative reliability of assigned postal code regions (the lighter the better).

due to for instance apartment buildings. These results are probably unwanted as they cause a less straightforward postal code map, but might actually be a proper description of the real world situation. As a result the question if and how these problems should be tackled can't be answered generically, the answers will differ for different purposes. Some ideas for improvements will be mentioned in the conclusions (Section 6).

In the second place the geometry of the derived postal code regions has to be judged. Although in general the result looks promising, this test also indicates some limitations in the use of the skeletonization method by DeLucia and Black. Firstly especially junctions are difficult objects to treat properly. A four way junction is indicated by two adjacent or near 0-triangles, both with no constrained edges, see Figure 3. This implicates at each crossing the creation of an edge of the skeleton between the centres of the two 0-triangles, thus creating a 'zigzag' shaped skeleton part. The second limitation are the

Figure 9. Result of the new algorithm, combined with the buildings layer from the topographic map.

Figure 10. Two houses are intersected by the postal code region boundaries.

'false 2-triangles' as shown in Figure 11. A little bulge in the parcel boundary causes the construction of a small additional triangle in the triangulation. As the skeleton represents the centrelines of a road network, this small triangle causes the creation of a non-existing side-road.

The third problem with the DeLucia and Black skeletons are the 'shifted 0-triangles,' as shown in Figure 12. As the lower boundary of the road did not contain any points except the endpoint, the triangles in the triangulations were stretched towards these

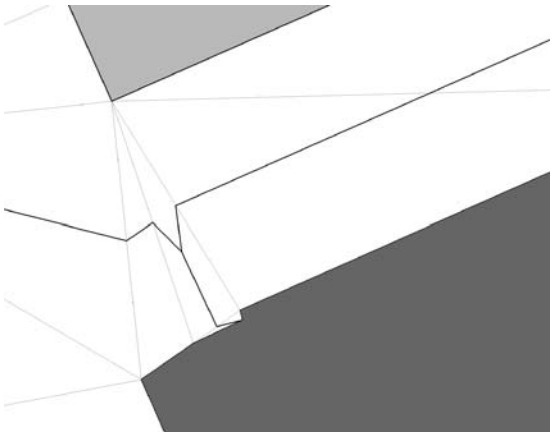

Figure 11. A false 2-triangle leads to the creation of an unwanted branch of the skeleton.

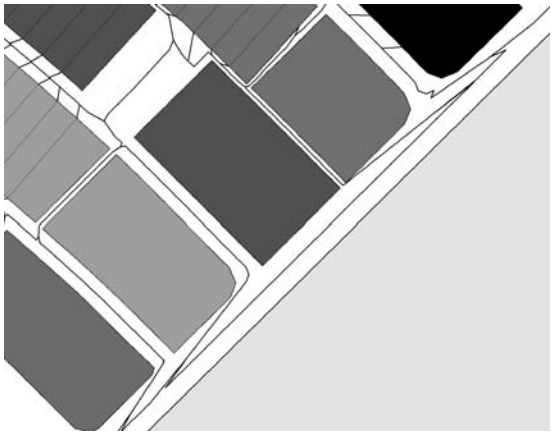

Figure 12. Shifted 0-triangle as the lower boundary contains few points.

endpoints thus dragging the skeleton towards these points. This results in unwanted bulges in the postal code regions, as one likes postal code regions to be as rectangular as possible.

In an attempt to eliminate or reduce these geometric problems several alternatives were developed and tested. These alternative approaches are described in the next section.

4. Alternative approaches

In Section 4.1 we investigate the usefulness of applying a conformal TIN (instead of a constraint TIN) and also try to see if using the true skeleton will give better results (than the approximated skeleton). As both alternative options do not solve all the problems mentioned in Section 2, some other alternatives are investigated (improved treatment of junctions).

4.1. Conformal triangulation instead of constraint triangulation

As the described algorithm is constraint triangulation (parcel boundaries) based, some improvements were expected when changing to a conformal triangulation based approach [10], because the Delaunay empty circumcircle criterion has to be fulfilled for all points. Therefore long edges are spitted by adding Steiner points (and in this way the very flat an long triangles as depicted in Figure 11 are avoided). As this conformal TIN consists of more points and thus more triangles (than the constraint TIN), the resulting skeleton was expected to be smoother. As described earlier three types of problems appear in the shape of the skeleton: shifted 0-triangles, false 2-triangles and spikes on junctions. Figure 13 clearly illustrates that the impact of shifted 0-triangles is

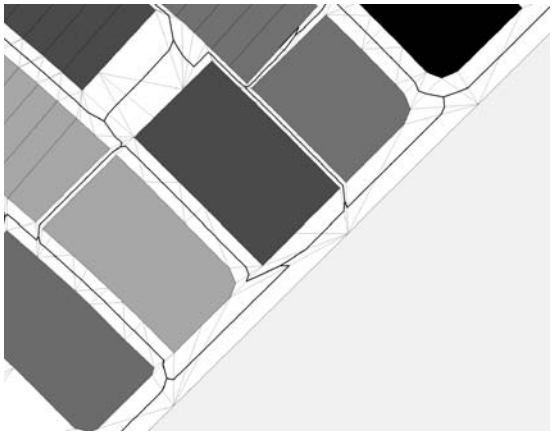

Figure 13. Steiner points in a conforming TIN reduce the influence of shifted 0-triangles effectively.

effectively reduced by the insertion of Steiner points. As Steiner points are added until the empty circumcircle criterion is fulfilled, these points are still distributed irregularly on the polygon boundaries. Uitermark et al. [13] suggest (without regard to a constrained or conformal TIN based approach) dividing parcel boundaries into regular parts by adding extra points. In order to attempt to obtain as equilateral triangles as possible he suggests adding these points every 15 metres, which is (for his test data) about the average road width. One can imagine that this will be an even more effective solution to the shifted 0-triangle problem, but this depends on the type and scale of data.

Unfortunately the other two problem types still appear in the new skeleton, which in retrospect was to be expected. As mentioned earlier false 2-triangles are caused by the

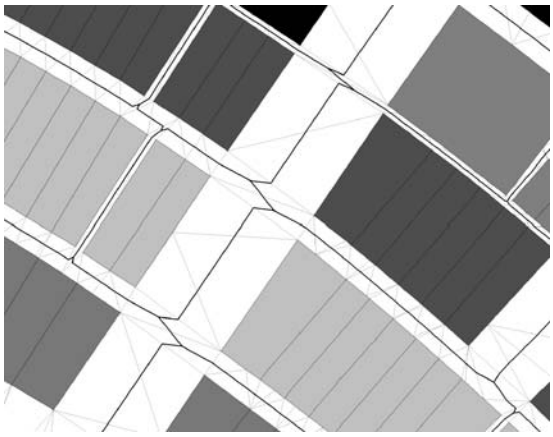

Figure 14. Spikes on crossings arise from stretched 0-triangles.

existence of small, detailed bulges in the parcel boundaries. This can be considered as a surplus of points on the polygon, which obviously will not be avoided by the addition of even more (Steiner) points. Generalization of the polygon boundaries might be a solution (as suggested in [13]), but this comes with the risk of intersection problems in case of close input data; e.g., very narrow alleys. The problem with spikes on junctions also still exists in the triangle-based approach. The size of these spikes on T-junctions and crossings depends on the size of the 0-triangle. The more stretched this triangle is, the larger the spike will be. As can be seen in Figure 14 the shape of most 0-triangles, part of a configuration related to a normal 4-way junction (or road intersection) is hardly influenced by the addition of Steiner points, as its stretched nature is mainly determined by the width of the road. Steiner points can't reduce this width as these points can only be added on parcel boundaries and not on the middle of a road.

Adding Steiner points leads to a result similar to the line Voronoi diagram inside the polygon that has to be subdivided. In a line Voronoi diagram the skeleton segments on crossings will be parabolic curves, which is also the case for the 'true skeleton.' One of the disadvantages of this 'true skeleton' is that the resulting boundaries do not only consist of straight-line segments, but also of parabolic curves. Also, the spikes (near T-junctions) do not disappear, and the resulting postal code regions will also contain parabola shaped spikes on T-junctions; see Figure 15.

4.2. Improving DeLucia and Black skeletonization rules

As both using conformal TINs or line voronoi diagrams do not completely solve the problem with the original DeLucia and Black skeletons, the remaining option is to improve the skeletonization rules of DeLucia and Black, particularly the rules for handling 0-triangles. Figure 14 illustrated the relationship between the actual shape of the 0-triangle and the size of the spike. Both on T-junctions and crossings these spikes appear, thus deteriorating the skeleton's smoothness. Jones [6] suggests an alternative approach of calculating skeleton segments in 0-triangles. The basic idea of this method is

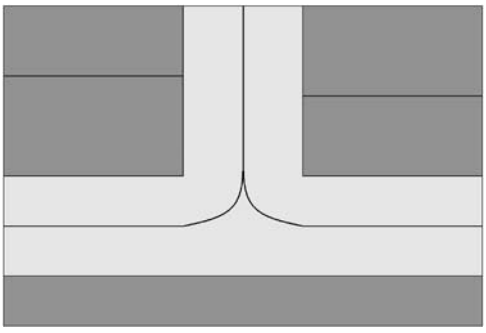

Figure 15. T-junctions defined by the true skeleton still contain (unwanted) spikes.

to calculate skeleton segments in the smoothest manner. This is realized by comparing the directions of the skeleton segments in all three adjacent triangles. The two segments that are the most aligned with each other are directly connected with a new skeleton segment. The third segment is connected with the midpoint of this new skeleton segment. In contrast with DeLucia and Black rules the intersection of the two newly calculated skeleton segments usually will not lie in the middle of the 0-triangle. Although this new method will prevent spikes on T-junctions, it still has a problem with crossings as in this case two 0-triangles are adjacent and thus no three adjacent skeleton segments are available. Therefore an alternative rule for crossings had to be defined, analogously to Jones' ideas. In this case (two adjacent 0-triangles) four skeleton segments are available in the adjacent triangles. These four segments can be divided in two pairs of best aligned segments and these segments will be connected to each other. As this operation is relatively laborious, an almost similar result can be obtained by connecting all four adjacent segments to the midpoint of the common boundary of the two 0-triangles. The results of the improved DeLucia and Black rules are illustrated in Figure 16, where the skeleton is much smoother on the T-junctions and crossings. The approach for crossings is not perfect, as it often happens that the 0-triangles on a crossing are not adjacent but separated by one or more very small triangles, caused by a rounded road corner. One might filter on the distance between the two 0-triangles in order to decide whether the form a crossing together and then ignore the small triangle in between, but this will complicate the algorithm significantly, while again it most probably will not be foolproof. Gao and Minami [5] describe similar solutions, also for the more general case on N-way junctions ($N > 4$) based on trend lines of the associated skeleton segments.

A small alteration on the DeLucia and Black rules can be the solution to the problem of false 2-triangles (en of network). Uitermark et al. [13] suggest adding a filtering on the triangle's surface area. As the fake 2-triangle does not span the entire road width its surface area will be smaller than the real ones. Based on a certain minimum required surface area criterion false 2-triangles can be ignored. The result is also that the a

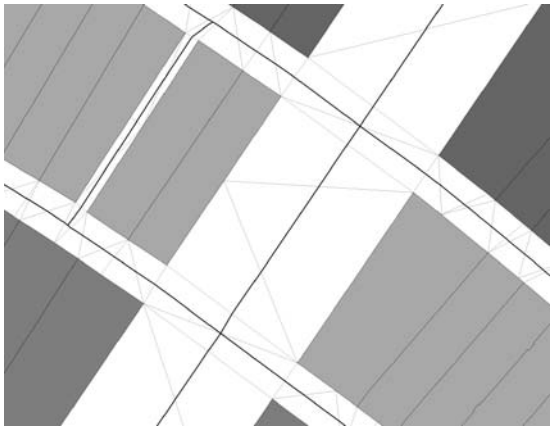

Figure 16. Smoother skeleton due to improved rules for 0-triangles.

neighbour triangle will change; most likely a fake 0-triangle (junction) will change into a normal 1-triangle (connection). The result is that the spike created by the (fake) 0-triangle, caused by the false 2-triangle, is now avoided. Another possibility is to smooth the parcel boundaries before triangulation, but this generalization might lead to intersection problems. Alternatively it may also be worthwhile to smoothen (generalize) the output boundaries and remove the spikes related to junctions. Again this holds the risk of unwanted (intersection) problems.

5. Conclusions and recommendations

We now conclude this paper by first summarizing our experiences with the creation of (several alternative) postal code maps. Afterwards we will end with suggestions for further research.

5.1. Conclusions

In this paper we presented a new method to derive a planar partition postal code map based on the geometric representation of cadastral parcels together with their associated administrative data, such as object addresses, culture and building codes. Infrastructural parcels (without cadastral address) are subdivided along their (approximated) skeleton. Sub parcels are created from this skeleton completed with some additional edges from the skeleton to the original neighbour nodes in the parcel map. The method has been implemented and tested on an actual data set. The results are very promising: better-shaped planar postal code maps than the existing Voronoi-diagram based products (better readable due to infrastructural line work, which is recognized by users). Also, visual checks indicate that far fewer buildings are intersected by postal code boundaries and the data owner has acknowledged in their first reaction this improvement. A quality indicator has been developed to indicate the reliability of the assigned postal code to the areas.

However, two semantic problems became apparent in the evaluation: some postal code regions result in disconnected areas and to some regions multiple postal codes were attached (due to apartment buildings). This could in fact be correct, but if possible it should be avoided as much as possible. Geometrically the method of DeLucia and Black offers good results, although three types of problems occur: shifted 0-triangles, false 2-triangles and unwanted spikes on T-junctions and crossings. Additional research showed that adding extra Steiner points to polygon boundaries effectively reduce the influence of shifted 0-triangles. Whether one chooses to use a conformal TIN (and therefore to add Steiner points) or adds points with regular intervals (suggestion Uitermark et al. [13]) is of less importance. The false 2-triangles can be either ignored based on a minimal surface area or prevented by generalizing the parcel boundaries. The first solution requires fine-tuning to find the correct minimal surface area value and the second solution incorporates the risk of unwanted intersection problems (in case of ‘narrow’ input). As the actual influence on the postal code region boundaries is limited,

it is an option to tolerate the occurrence of false 2-triangles. The problem of the spikes can be tackled by implementing Jones' idea for t-junctions and the proposed Jones-based approach for crossings. Another option is a post processing line generalization, but this will diminish the close relationship between postal code boundaries and road centrelines.

5.2. *Future work*

Besides a number of tested improvements, we also have a number of not-yet tested improvements (or alternative solutions): two phase merge, avoid disjoint postal code areas, avoids multiple postal codes per area, refined mesh based full triangulation, avoid intersection of postal code areas with buildings.

1. *Two phase merge:* This method is a relative simple adaptation of our current implementation: instead of only merging areas at the end of the algorithm, the algorithm now starts with a 'merge' step. The advantage of this method is that the postal codes do not cross the roads, which is also in the real postal code world not the case (in the Netherlands). Another advantage is that less additional skeleton edges are needed, which will save computing time. In pseudo code the algorithm looks like:

```
Step 1
Assign postal codes (to non-infrastructural parcels without
address) and merge parcels to larger areas until an infra-
structural parcel is reached.
Step 2
Compute skeleton of infrastructural parcels and additional
skeleton edges to the merged areas.
Step 3
Continue assigning postal codes to areas without address and
merge them together.
```

2. *Avoid disjoint postal code maps:* Another interesting field of research will be the question how to solve the disjoint postal code areas. One might think of an alternative assignment of postal codes. In the current implementation the most likely postal code is selected. One could now choose to select an ordered list of likely postal codes for each (sub) parcel and then try to calculate non-disjoint postal code areas with the 'best' postal code for as much parcels as possible using some kind of weight function. An easier but less desirable approach would be to skeletonize all non built-up parcels (so also the non infrastructure parcels as the agricultural or natural environment parcels), thus enabling the assignment of more postal codes to a single parcel and so connecting the disjoint postal code areas. This approach is less desirable because of the creation of meaningless (i.e., no physical meaning) boundaries, whereas the basic objective of this algorithm was to create recognisable, meaningful boundaries.

3. *Avoid multiple postal codes per area:* Solving the problem of multiple postal codes assigned to one parcel is easier. One could split the parcel in as many parts as needed, thus creating again meaningless boundaries. A more appropriate option would be to choose a cartographic solution, thus making clear that the mapped situation is not the same as the actual solution. One might think of creating some kind of a pie chart in the parcel, thus representing each postal code separately but at the same time making it obvious that these boundaries have no geometrical meaning.
4. *Refined mesh based full triangulation:* Instead of just triangulating the infrastructure parcels, triangulate everything. Further, refine the constraint triangulation; e.g., by a conformal triangulation or perhaps by also adding points to the internal area of faces that are too large [10]. Triangles belonging to infrastructure parcels are further subdivided by their skeleton (as described in this paper). Now all triangles that belong to a parcel with postal code, get that postal code. Now iteratively assign all remaining (parts of) triangles one (or more) postal codes. Once all (parts) of triangles have been assigned one (or more) postal codes try to create coherent postal code regions. In case there is only one postal code assigned no choices have to be made and the regions are defined (and still may not always be coherent, that is a single postal code may have to disjoint polygons). In case alternative postal codes are available choices can be made (in order to avoid multiple postal codes per area).
5. *Avoid intersection of postal code areas with buildings:* One of the remaining problems with all alternative methods presented so far, is that in some exceptional cases the postal code regions still intersect with the buildings on the map. Since buildings are known (or supposed) to be in one post region in most cases, this is unwanted behaviour. Therefore we are looking for ways to create boundaries that never overlap with buildings and stay far away from buildings. The buildings on a postal map are used as input for a constrained (or conformal) TIN algorithm. Then the associated skeleton segments (in the area on the outside of the buildings) are used to split the area into regions that are guaranteed not to overlap with buildings. The regions get the postal code of the building (assuming that we cab somehow obtain this from the cadastral map). Finally, regions with the same postal code are glued together. Though the region boundaries will not intersect buildings, it is unlikely that the map will be very clear or readable. The boundaries between the regions do not have a direct relationship with the infrastructure parcels (roads, waterways, etc.) anymore and may have very funny shapes. This is basically the same problem as with the original point-based Voronoi postal code maps.

Other future work, not directly related to the main area computation algorithm, will be:

1. Investigate the quality of the administrative data (culture and building code) on the resulting planar postal code map.
2. Derive beside the most detailed 6ppc (6 position) also the larger postal code area units: 5ppc and 4ppc (or even 3ppc, 2ppc or 1 ppc) maps. We already performed one little experiment: as one can see in Figure 17 it looks slightly overdone to use the exact cadastral boundaries for such inaccurate maps as 5ppc postal code regions.

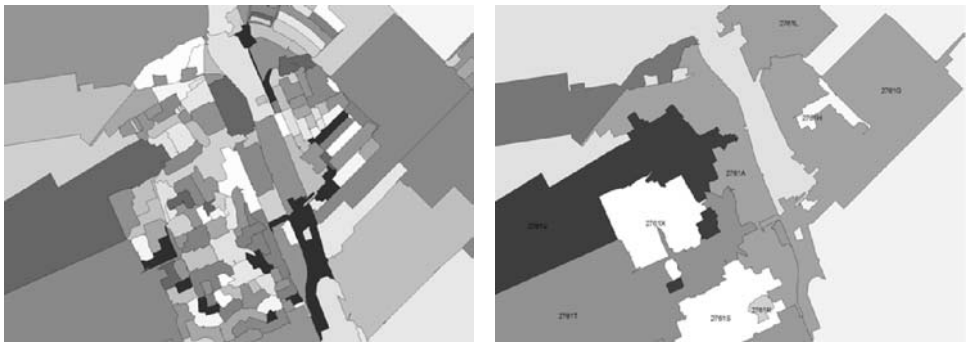

Figure 17. The 6ppc map and the derived 5ppc map of the same area: the accuracy of the boundaries seems to be out of proportion with the purpose.

Some generalization need to be applied here (and also suspicious enclaves and exclaves have to be investigated further).

3. Produce more quantitative proof of the improved quality (also for larger data sets and different types of regions).

Acknowledgments

The authors would like to express their acknowledgements to de Dutch Kadaster for use of their AKR and LKI databases for this research and all other research conducted by the Section GIS-technology within the Geo Database Management Centre of the Delft University of Technology.

Notes

1. AKR stands for Administratieve Kadatrale Registratiesysteem (in Dutch): 'Administrative Cadastral Registration System.'
2. LKI stands for Landmeetkundig Kartografische Informatiesysteem (in Dutch): 'Information System for Surveying and Mapping.'

References

1. CGAL Basic library manual. CGAL Consortium, 2002.
2. R. Chithambaram, et al. "Skeletonizing polygons for map generalization," *Technical papers, ACSM-ASPRS Convention, Cartography and GIS/LIS*, Volume 2. Bethesda: ACSM, 44–54, 1991.
3. A. DeLucia and T. Black. "A comprehensive approach to automatic feature generalization," in *Proceedings of 13th International Cartographic Conference*, International Cartographic Association, pp. 169–191, 1987.

4. A. Fabri, et al. On the design of CGAL, the Computational Geometry Algorithms Library (research report). Saarbrücken: Max-Planck-Institut für Informatik, 1998.
5. P. Gao and M.M. Minami. "Raster-to-vector conversion: A trend line intersection approach to junction enhancements," in *Auto-Carto* Vol. 13:297–303, 1997.
6. C.B. Jones, et al. "Map generalization with a triangulated data structure," *Cartography and Geographic Information Systems*, Vol. 22–4:317–331, 1995.
7. P.J.M. van Oosterom. "Hartlijnen van wegennetwerken." *Informatieblad Kadaster*, Vol. 7:10–11, 1999 (in Dutch).
8. P.J.M. van Oosterom, B. Maessen and C.W. Quak. "Generic query tool for spatio-temporal data," *International Journal for Geographical Information Science*, Vol. 16–87:713–748, 2002.
9. F. Penninga, E. Verbree, C.W. Quak and P.J.M. van Oosterom. "Construction of the Planar Partition Postal Code Map Based on Cadastral Registration," *Proceedings of the Eleventh ACM International Symposium on Advances in Geographic Information Systems, November 2003*, New Orleans, USA.
10. J.R. Shewchuk. "Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator," In First Workshop on Applied Computational Geometry, Philadelphia, Pennsylvania, USA, MAY 1996, pp. 124–133.
11. T.G. Schut, en B. Maessen, Haalbaarheid. 6PPC-vlakkenkaart (eindrapportage). Apeldoorn: Kadaster, 2000 (in Dutch).
12. F. Thomas. "Generating street center-lines from inaccurate vector city maps," *Cartography and Geographic Information Systems*, Vol. 25–4:211–230, 1998.
13. H. Uitermark, A. Vogels, and P. van Oosterom. "Semantic and geometric aspects of integrating road networks," in A. Vckovski et al. (Eds.), *Interoperating Geographic Information System: Second International Conference, INTEROP '99*, Berlin: Springer-Verlag, 117–188, 1999.
14. A. N. Wilschut, R. van Zwol, N. Brasa, J. Flokstra. "Road collapse in magnum," in R. Laurini, (Ed.), *6th ACM Workshop on Advances in Geographic Information Systems ACM-GIS'98*, Washington DC, USA, 6–7 November 1998.

Friso Penninga obtained a M.Sc. in Geodesy in 2003 from Delft University of Technology. Afterwards he became a Ph.D. student at the same university in the GISt department of the OTB Research Institute for Housing, Urban and Mobility Studies. His research focus will be on 3D topographic modelling.

Edward Verbree got his M.Sc. degree in Geodesy in 1992 at Delft University of Technology. After some year in industry he is since 1997 assistant professor at the department of GIS-technology of Delft University of Technology. His main research activities are 3D-surface reconstruction with TIN/TEN modelling and location-based services (LBS). Besides he lectures in GIS-design and LBS.

Wilko Quak got his M.Sc. degree in Computer Science from Utrecht University in 1992. After that he worked as a Ph.D. student at the University of Amsterdam on performance of spatial databases. Currently he is employed as an assistant researcher at GIS Technology department of the OTB Research Institute for Housing, Urban and Mobility Studies, Delft University.

Peter van Oosterom obtained a M.Sc. in Technical Computer Science in 1985 from Delft University of Technology. In 1990 he received a Ph.D. from Leiden University for his thesis "Reactive Data Structures for GIS.'' From 1985 until 1995 he worked at the TNO-FEL laboratory in The Hague as a computer scientist. From 1995 until 2000 he was senior information manager at the Netherlands' Kadaster, were he was involved in the renewal of the Cadastral (Geographic) database. Since 2000, he is professor at Delft University of Technology (OTB) and head of the section 'GIS Technology.'