Building simplification using offset curves obtained from the straight skeleton

Martijn Meijers

GIS-technology, OTB Research,
Faculty of Architecture and the Built Environment
Delft University of Technology, The Netherlands

Abstract

We propose a conceptual simple algorithm based on offset curves obtained from the straight skeleton to simplify building outlines. We present initial results with some real world data and show that the approach can be used to simplify and amalgamate building outlines. We discuss how this approach can be extended to generate smooth transitions for continuous zoom.

1 Introduction

In this paper we present our on-going investigations in the application of the straight skeleton structure for continuous/smooth building generalization. We propose a process similar to erosion–dilation (expand–shrink), but we use offset curves obtained via the straight skeleton. Recently it was shown that the straight skeleton can be obtained efficiently from Planar Straight Line Graphs (PSLG) with a practical implementation by means of a kinetic triangulation (Palfrader, 2013). Once this skeleton structure has been obtained, it allows to generate offset curves, similar to what one can obtain by a normal GIS buffer operation (Minkowski sum with a disc). However, there are differences in the resulting geometry, e. g. right angled corners are better preserved, which is useful for building simplification.

We try to answer the question how well our algorithm (that uses offset curves derived from the straight skeleton) performs for building simplification. We aim at simplification of individual buildings, as well as aggregation of multiple buildings into built-up areas (building blocks). The initial results we obtained look promising: Aggregation and simplification can both be carried out. Also, the resulting data with different levels of detail should be suited for constructing continuous generalizations (for smooth zoom) and initial ideas are described for this.

The remainder of this short paper is structured as follows: Section 2 presents related work and surveys state of the art work with respect to building generalization and straight skeleton construction. Section 3 describes our proposal for building simplification. Section 4 gives an overview of the experiments we carried out. Section 5 concludes the paper with a discussion of the obtained results and our plans for future work.

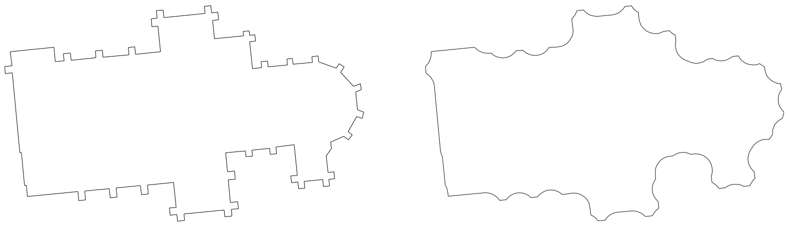


Figure 1: (left) Input. (right) Erosion–dilation with a standard GIS buffer operation (Minkowski sum with a disc)

2 Related work

2.1 Building generalization

Sester (2000) describes an algorithm for building simplification method based on least squares adjustment. In the algorithm small edges are removed, while the general structure of a building (with quite many 90° angles) is preserved by means of least squares adjustment. This algorithm is implemented in the CHANGE software, that is commercially available from the University of Hannover (Bobrich, 2001).

Damen et al. (2008) describe another algorithm, based on a Minkowski sum operation using the morphological operations opening and closing (erosion and dilation). They apply these operations with different elements that are added to the input buildings: discs, axis aligned squares and mostly aligned squares (by rotating the building to its main axis). Figure 1 illustrates that performing the Minkowski sum operation with a disc element as input, does not lead to satisfactory results for buildings (as this results in circular arcs in the generalized outlines).

However, they conclude that their approach works well when a square is used as element and the input building is rotated along its main axis. Also, Kada (2011) uses erosion and dilation for building ground plan simplification and notes as well that it is needed to rotate the input geometry along the main axis of the buildings to obtain good results, but that in practice this is not always possible due to certain geometric configurations.

Haunert and Wolff (2010) present an optimization approach to simplify sets of building footprints represented as polygons. The basic idea of their approach is to select a subsequence of the original edges of an input polygon; the intersections of the selected and possibly extended edges form then a new, simplified polygon. The method proposed does guarantee a solution within a geometric error tolerance ϵ as specified by the user, however, symmetries present in the input can be lost.

2.2 Straight skeleton

Aichholzer et al. (1995) define the straight skeleton structure by a so-called wavefront propagation process. Consider a simple (not self-intersecting) polygon. Every edge of the polygon sends out 2 parallel wavefront edges. Between two adjacent wavefront edges s_1, s_2 a wavefront vertex moves along the angular bisector of s_1 and s_2.

Figure 2 illustrates that during this propagation process multiple events can happen (that represent changes in the wavefront): 1. a wavefront edge can shrink to zero length or 2. a split event can happen where an opposing wavefront edge is split into parts. The straight skeleton is now defined as the set of traces (arcs) that were left behind by the moving wavefront vertices (the blue lines in Figure 2).

Aichholzer and Aurenhammer (1996) have generalized the straight skeleton concept to be able to not only take polygons as input, but also Planar Straight Line Graphs (PSLG).

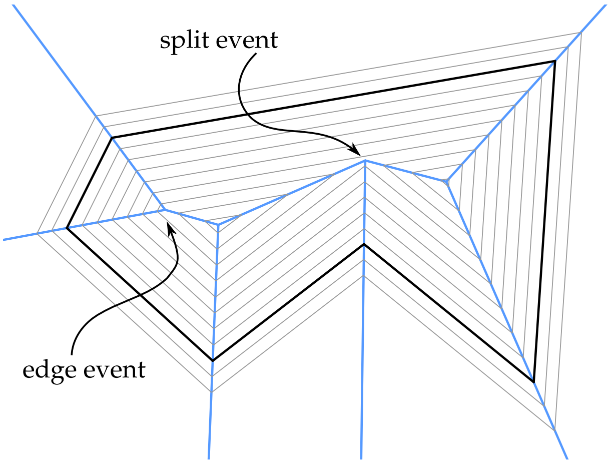


Figure 2: Polygon (black) with straight skeleton arcs (blue) and some offset curves (grey). During the wavefront propagation multiple events can happen: Edge events and split events. Figure after Palfrader (2013).

A PSLG defines a set of points, and some non-intersecting straight line segments. They give a description of an algorithm based on kinetic triangulations for obtaining the straight skeleton. For the triangles that are part of the kinetic triangulation their area changes over time, while the wavefront is progressing.

Palfrader and Held (2015) have taken this description and transformed it into an implementation that can cope with real-world data and can generate mitered offsets. For polygonal input, these offsets can be generated inwards, as well as outwards as Figure 2 shows. To be able to generate offset curves, a data structure is used by means of which it is known at every time *t* of the wavefront propagation process what are for each wavefront vertex the two neighbouring wavefront vertices. Based on this information it is straightforward and fast to generate an offset curve — once the straight skeleton is known, time is linear in the number of input segments.

Haunert and Sester (2007) use the straight skeleton to obtain a generalised, linear representation for polygonal input. The skeleton operation is applied to linear features (networks), such as road and water. Next to obtaining the centerlines from a cadastral dataset, they also propose a *partial* collapse method (mainly used for collapsing narrow water parts). The wavefront propagation of the straight skeleton algorithm is stopped, after it has travelled a user defined distance, leading to a smaller polygon, with certain areal parts collapsed/removed. They mention that this operation is 'very similar to the morphologic opening operator, which works on raster data' (Haunert and Sester, 2007, p. 178) and which was also investigated by Damen et al.

3 Algorithm for building simplification and aggregation based on offsets obtained via the straight skeleton

Inspired by the works of Haunert and Sester (2007) and Damen et al. (2008), and given the description of Palfrader (2013) and Palfrader and Held (2015) for computing the straight skeleton and a set of offset curves, we present here a straightforward algorithm for simplifying a (set of) building outline(s):

1. Compute straight skeleton for a (set of) input polygon(s).

2. Generate offset curves outwards, giving a user defined ϵ threshold distance (i. e. perform dilation, expand the input polygon).

3. Compute again the straight skeleton for this resulting set of already simplified curves.

4. Generate offset curves inwards, again using the same ϵ as distance (i. e. perform erosion, shrink back the expanded and somewhat simplified input).

Note that for this general description of the procedure a. the order of erosion and dilation can be swapped and b. that 2 more processing steps can be added:

5. Compute again the straight skeleton, but now for the offset shape.

6. Compute once more an offset curve.

When we employ this strategy, we halve the ϵ threshold in step 2 and 6.

4 Experiments and results

Results we present in this paper have been obtained by our implementation of this algorithm that takes a PSLG as input (where we followed the guidelines of Palfrader (2013)), and which we named GRASSFIRE. We can obtain both the straight skeleton arcs as well as a set of offset curves of a given input PSLG. Implementation is an ongoing effort to improve robustness of the algorithm (not all cases in practice are handled correctly yet in our current implementation).

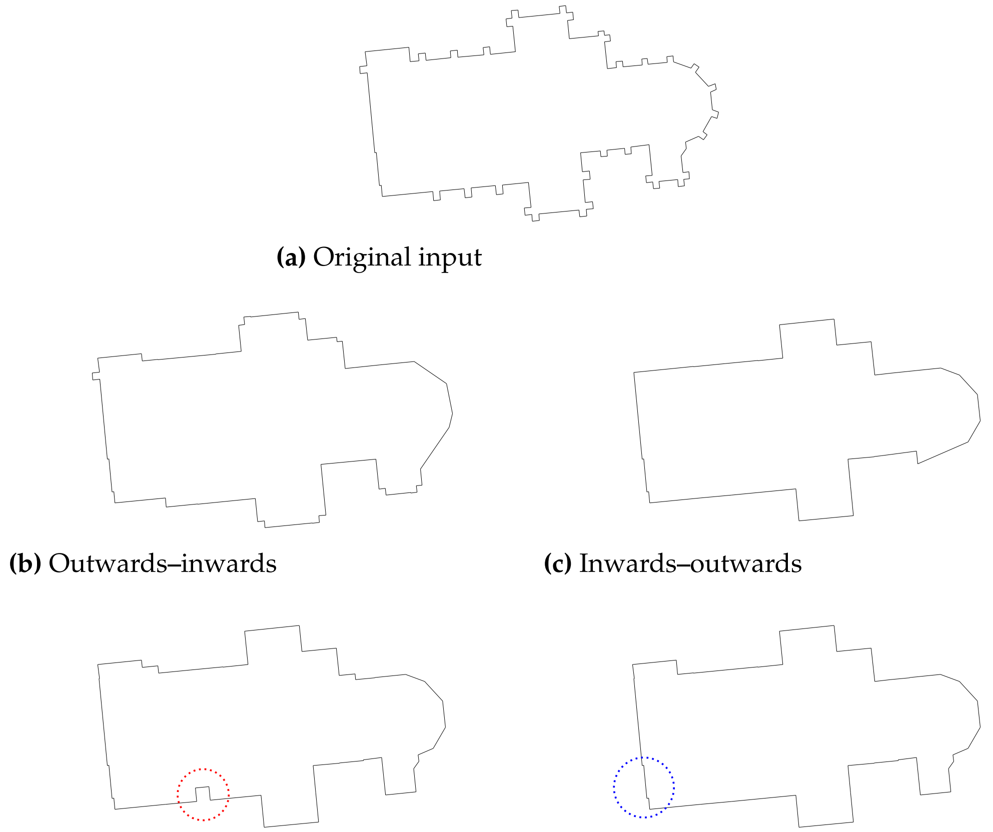
As input we have taken buildings from the BAG (the key register of buildings and addresses in The Netherlands, in Dutch: Basisregistraties adressen en gebouwen). Figures 3a and 4a show this input.

Figure 3 shows the outcome of the different variants of the algorithm, which we described in Section 3, applied on an individual building. Figure 3b illustrates that by first going outwards and then inwards, small details (corners) can be preserved. This is caused by the the wavefront vertices part of these corners that move in parallel, or even diverge, along their bisector lines. Hence we tried to shrink first and then move back outwards again. This operation did indeed lose all small details, but also a quite significant building part is lost (cf. Figure 3c). For this particular case, this omission could have been resolved by choosing a smaller ϵ threshold.

However, as going outwards and then inwards leads to a significantly different outcome from first shrinking (inwards) and then expanding the result (outwards), we investigated two other options: Move $\frac{1}{2}\epsilon$ outwards, move ϵ distance inwards and then $\frac{1}{2}\epsilon$ distance outwards, as well as vice versa (start with moving inwards, then outwards and then inwards again). Figure 3d and 3e illustrate the different results. At first sight both lead to reasonable results — the large details are preserved (the corners) and smaller details are eliminated (the abutments in the outline of the church). However, moving outwards–inwards–outwards does generate additional detail that was not present in the original input (in Figure 4d the building edge on the lower left side contains a dent inwards that was not present in the input, circled with red dots).

In all cases short edges still remain in the simplified output (the tiny 'stair edges' on the left side of the building, circled with blue dots in Figure 4e). These edges can possibly be removed by a short edge removal algorithm (e. g. skipping the vertex at the end of a short edge, while going around the polygon).

Figure 4 shows the same process, but now on a collection of building outlines as input. To process these efficiently, we first dissolved the internal edges of the adjacent buildings



(a) Original input

(b) Outwards–inwards

(c) Inwards–outwards

(d) Outwards–inwards–outwards

(e) Inwards–outwards–inwards

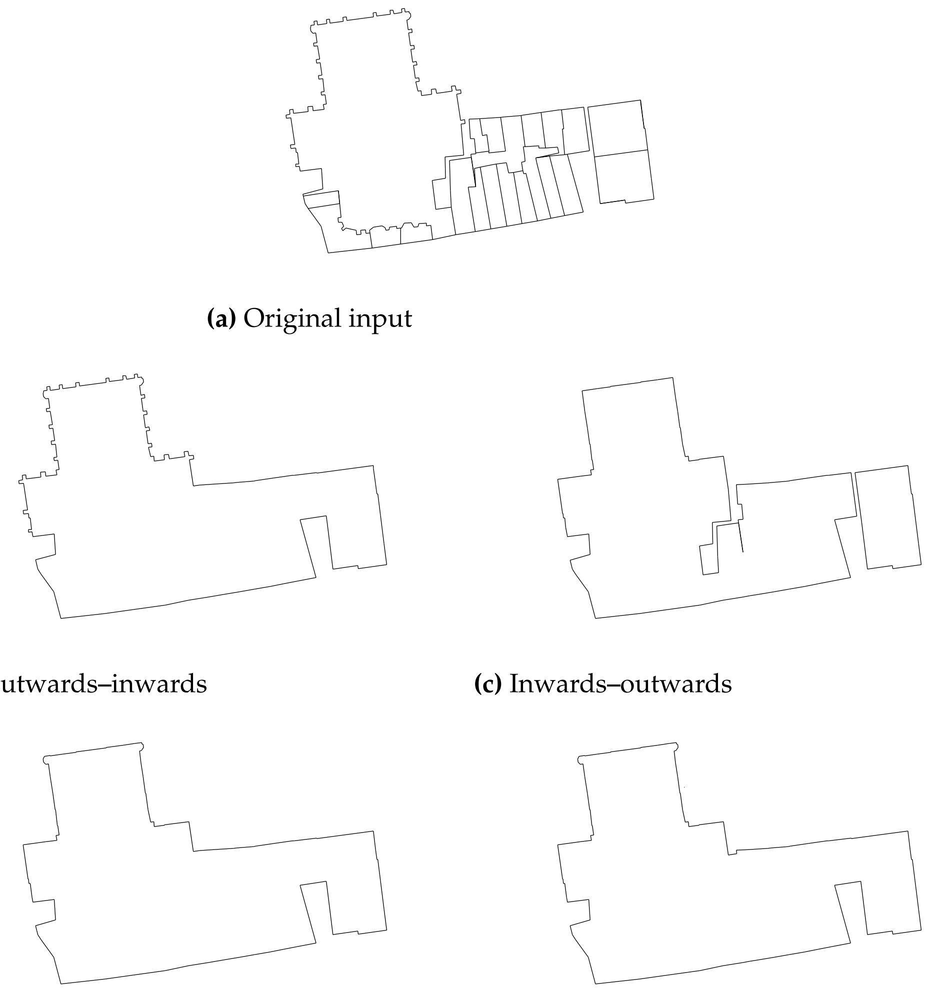
Figure 3: Church with many abutments in its outline

and used the resulting geometries as input for our building simplification algorithm. Figure 4b illustrates that by first going outwards and then inwards, also here small details are preserved. It also shows that this order of processing does amalgamate multiple building parts to one new polygon. In Figure 4c it is clear that the reverse order (inwards–outwards) does not lead to amalgamation (separate blocks in the input are kept separate in the output). Figures 4d and 4e show that both strategies where we move outwards, inwards and outwards again and vice versa lead to near identical results and that for both cases amalgamation takes place.

5 Discussion

We have proposed an algorithm to simplify building outlines, by using offsets obtained via the straight skeleton. Our results indicate that the proposed algorithm, which is conceptually simple, performs reasonably well and can be used to simplify buildings and amalgamate buildings into building blocks (built-up areas).

The experiments showed that not all small details in the input are removed when we first move outwards and then inwards. Too much detail (i. e. a whole building part) can be lost, when moving inwards then outwards (admittedly depending on ϵ tolerance, which was set). Both the sequences outwards-inwards-outwards, as well as inwards-outwards-inwards seem to give reasonable results. However, both processing orders can introduce small details not present in the original input geometries.



(a) Original input

(b) Outwards–inwards

(c) Inwards–outwards

(d) Outwards–inwards–outwards

(e) Inwards–outwards–inwards

Figure 4: Individual building outlines to built-up area

A nice result is that no rotation of the input is needed for the straight skeleton-based erosion-dilation algorithm to obtain good results (contrary to the case with the Minkowski sum with a square as input).

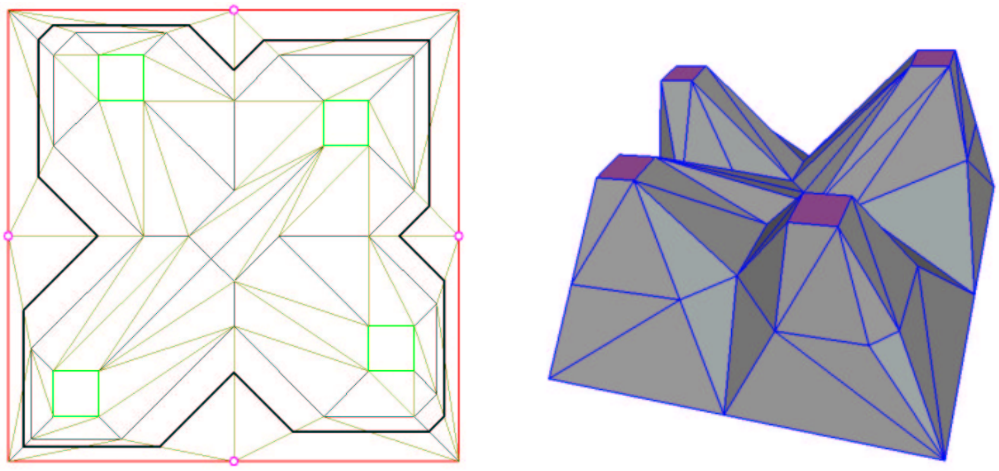
In general, we found it difficult to determine a good ϵ value in advance and to predict what would be the expected outcome with the set ϵ distance. We want to find out guidelines on what would be a good threshold value.

Further work is also needed to better benchmark the proposed approach: This means that we want to improve our initial straight skeleton implementation, as it is not yet fully robust. For this we need to test with significantly more and different real world input data. Once this work is finished, we also want to obtain statistics on the input and output (e. g. count number of vertices, determine area that is occupied, angle distribution, . . .).

Possibly, the algorithm can also generate a typified version (caricature) of an input polygon: First, grow building polygon outwards by means of the straight skeleton and then graphically scale the resulting polygon back to its original size (or leave it somewhat enlarged). A typified building is the result, that can be used to represent the more important buildings (points of interest).

Furthermore, we want to investigate to what other types of features our proposed algorithm can be applied. Haunert and Sester (2007) mention that when offsetting based on the straight skeleton is used the resulting polygons can have sharp angles and that this leads to less visually pleasing results. Haunert and Sester propose to use circular arcs obtained from a buffering process to replace these parts in the resulting shape. Palfrader and Held (2015) use the linear axis around vertices that have sharp angles and we want to investigate whether this can be used to our advantage as well.

Barequet et al. (2004) and Yakersberg (2004) use the straight skeleton to get smooth transitions between 2 slices of medical imagery (cf. Figure 5). By applying their algorithm to our initial input and the simplified output, we want to investigate whether it is possible to obtain smooth generalization for buildings. The idea is as follows: To make a continuous generalization possible between data for two discrete map scales, we overlay our input data, with the simplified version. The smooth transition zones can then be found by determining which polygonal parts are in either the initial or the derived polygon(s), but not in both. For every individual polygonal part that is found this way, we can obtain a straight skeleton that will divide these polygonal parts even further. This way a transition zone is defined, and the offsets based on the straight skeleton can subsequently give a gradual shrinking process (for the morphing) of the parts. The simplification, overlay and connection process (to connect different levels of detail) can be repeated multiple times to arrive at a series of smoothly transitions. An open question here is how many times to repeat to give reasonable results?



(a) Red (low level of detail) and green lines (high level of detail) are input. Black line is straight skeleton between two slices of input data, that models half way transition. The area is also triangulated, and Steiner points are inserted to obtain a smoother transition.

(b) 3D view of interpolated surface.

Figure 5: The transition zone is found by overlaying the two inputs (red / green). For the zone that is only in one (in this case the area between the red and green input curves) a smooth transiton is found. Figure taken from Yakersberg (2004).

Acknowledgements

This research was financially supported by: (i) the Dutch Technology Foundation STW, which is part of the Netherlands Organisation for Scientific Research (NWO) and partly funded by the Ministry of Economic Affairs (project code 11185) and (ii) the European Location Framework (ELF) project, EC ICT PSP Grant Agreement No. 325140.

References

Aichholzer, O., Alberts, D., Aurenhammer, F., and Gärtner, B. (1995). Straight skeletons of simple polygons. In *Proceedings of* 4th *International Symposium of LIESMARS*, pages 114–124, Wuhan, P. R. China.

Aichholzer, O. and Aurenhammer, F. (1996). Straight skeletons for general polygonal figures in the plane. In Cai, J.-Y. and Wong, C. K., editors, *Computing and Combinatorics*, volume 1090 of *Lecture Notes in Computer Science*, pages 117–126. Springer Science + Business Media.

Barequet, G., Goodrich, M. T., Levi-Steiner, A., and Steiner, D. (2004). Contour interpolation by straight skeletons. *Graphical Models*, 66(4):245–260.

Bobrich, J. (2001). Cartographic Map Generalization in Urban Districts. In *Proceedings of GIS Research UK,* 9th *Annual Conference*, pages 513–515.

Damen, J., van Kreveld, M., and Spaan, B. (2008). High quality building generalization by extending the morphological operators. In *Proceedings of the ICA Workshop on Generalization*, Montpellier, France.

Haunert, J.-H. and Sester, M. (2007). Area collapse and road centerlines based on straight skeletons. *GeoInformatica*, 12(2):169–191.

Haunert, J.-H. and Wolff, A. (2010). Optimal and topologically safe simplification of building footprints. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS*. Association for Computing Machinery (ACM).

Kada, M. (2011). Aggregation of 3D buildings using a hybrid data approach. *Cartography and Geographic Information Science*, 38(2):153–160.

Palfrader, P. (2013). Computing Straight Skeletons by Means of Kinetic Triangulations. Master's thesis, University of Salzburg, Austria.

Palfrader, P. and Held, M. (2015). Computing Mitered Offset Curves Based on Straight Skeletons. *Computer-Aided Design and Applications*, 12(4):414–424.

Sester, M. (2000). Generalization based on least squares adjustment. In Fritsch, D. and Molenaar, M., editors, *ISPRS Archives –* 19th *ISPRS Congress Technical Commission IV: Mapping and Geographic Information Systems*, volume XXXIII, Part B4 (/1-3), pages 931–938, Amsterdam, The Netherlands.

Yakersberg, E. (2004). Morphing between Geometric Shapes using Straight-Skeleton-Based Interpolation. Master's thesis, Technion – Israel Institute Of Technology, Haifa, Israel.