

Simultaneous selection and displacement of buildings and roads for map generalization via mixed-integer quadratic programming

Leon Rosenberger, Yilang Shen & Jan-Henrik Haunert

To cite this article: Leon Rosenberger, Yilang Shen & Jan-Henrik Haunert (2025) Simultaneous selection and displacement of buildings and roads for map generalization via mixed-integer quadratic programming, International Journal of Geographical Information Science, 39:7, 1567-1596, DOI: [10.1080/13658816.2025.2461602](https://doi.org/10.1080/13658816.2025.2461602)

To link to this article: <https://doi.org/10.1080/13658816.2025.2461602>



© 2025 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.



[View supplementary material](#)



Published online: 03 Mar 2025.



[Submit your article to this journal](#)



Article views: 903



[View related articles](#)



[View Crossmark data](#)

RESEARCH ARTICLE



Simultaneous selection and displacement of buildings and roads for map generalization via mixed-integer quadratic programming

Leon Rosenberger^a, Yilang Shen^b and Jan-Henrik Haunert^a 

^aInstitute of Geodesy and Geoinformation, University of Bonn, Bonn, Germany; ^bSchool of Geospatial Engineering and Science, Sun Yat-sen University, Zhuhai, Guangdong

ABSTRACT

Research on map generalization has led to many algorithms for multiple elementary processes, such as object selection, aggregation, simplification, and displacement. Algorithms for different processes are usually combined to workflows or orchestrated using multi-agent systems. Here, we present a novel approach integrating object selection and displacement at an algorithmic level. We model both processes together as an optimization problem in the form of a mixed-integer quadratic program and demonstrate that it can be optimally solved using a mathematical problem solver. Moreover, we present an efficient heuristic. In experiments with roads and buildings from OpenStreetMap, our methods showed a good capability to unselect a small set of buildings whose inclusion in the output map would have caused large displacements or proximity conflicts. For a quantitative evaluation, we solved a benchmark instance once with our new model integrating selection and displacement and once with a variant of our model where the selection of objects was prescribed based on a solution found with an existing approach via simulated annealing. Comparing the two models, our integrated model yielded a solution of 33% less total cost. We conclude the article with a discussion of possible follow-up work.

ARTICLE HISTORY

Received 5 July 2024
Accepted 28 January 2025

KEYWORDS

Map generalization;
displacement; selection;
optimization; mathematical
programming

1. Introduction

Map generalization aims to reduce the scale of a map while preserving the most characteristic information in it. To satisfy minimal dimensions and distances, the map has to undergo several processes. These processes include object selection, displacement, aggregation, and simplification, among others. In the past decades, optimization has repeatedly proven itself for automating different processes of map generalization. Researchers have emphasized, in particular, that optimization approaches can handle

CONTACT Jan-Henrik Haunert  haunert@igg.uni-bonn.de

 Supplemental data for this article can be accessed online at <https://doi.org/10.1080/13658816.2025.2461602>.

© 2025 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.

This is an Open Access article distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. The terms on which this article has been published allow the posting of the Accepted Manuscript in a repository by the author(s) or with their consent.

diverse constraints that cartographers have formalized as mathematical representations of cartographic knowledge (Sester 2005; Harrie and Weibel 2007). These constraints deal with the legibility of the output map or with the preservation of information given with the input map (Burghardt *et al.* 2007). Since a perfect satisfaction of all constraints is rarely possible, one usually considers some or all constraints as soft and tries to satisfy them as much as possible.

The optimization techniques that have been successfully applied to map generalization include mathematical programming, specifically least-squares adjustment (Harrie 1999; Harrie and Sarjakoski 2002) and (mixed) integer linear programming (Haunert and Wolff 2010a, 2010b). Also, it is common to apply meta-heuristics, including hill climbing and simulated annealing (Ware and Jones 1998; Ware *et al.* 2003) and genetic algorithms (Wang *et al.* 2017). To deal with multiple processes of map generalization in an integrated way, multi-agent systems are nowadays frequently used (Maudet *et al.* 2014). They are often driven by an optimization engine that iteratively modifies the current map based on mathematically defined goals. A discussion of different optimization strategies in the context of a multi-agent system for polygon generalization is provided by Galanda (2003).

Despite the success of optimization approaches for map generalization, there remains a larger scientific challenge that we address with this work. This challenge is the simultaneous treatment of multiple processes of map generalization in an optimization approach, where some processes deal with discrete decisions (e.g. whether to select an object or aggregate a set of objects) and other processes deal with continuous decisions (e.g. by how much to displace or enlarge an object). A prerequisite for tackling this challenge is a unified mathematical model integrating the different processes. Once such a model has been created, the problem will become accessible for a whole range of algorithmic techniques that are constantly being developed in the growing discipline of algorithm engineering (Sanders 2011). These techniques include both exact algorithms and efficient algorithms for computing approximate solutions.

To deal with both continuous and discrete decisions, the approach most commonly used in the context of map generalization is to discretize the continuous solution space. Then, a framework for discrete optimization is applied. For example, to deal with object displacement in a simulated annealing framework, Ware *et al.* (2003) defined for each map object an integer number of 'trial positions', from which in each iteration of the algorithm a candidate position is randomly sampled, evaluated, and possibly accepted for the next iteration. Less common but also possible is to model discrete decisions of map generalization in an optimization framework that uses only continuous variables. In particular, Harrie and Sarjakoski (2002) dealt with line simplification in their framework based on least-squares adjustment by introducing a constraint that tries to move a node towards the line segment connecting its predecessor and successor node. If the constraint is perfectly satisfied, the node can be removed without affecting the shape of the line. With this, a simplification of the line is achieved. Recently, Zhou *et al.* (2023) have presented a method based on deep learning that simplifies building footprints by simultaneously moving and removing nodes.

In contrast to previous work, we here investigate the potential of an optimization framework that can deal with both discrete and continuous variables and, in particular, that does not apply any unnatural discretization of continuous variables. Specifically, we apply mixed-integer quadratic programming to the simultaneous selection and displacement of objects, where ‘mixed-integer’ refers to the fact that we use both continuous variables (to model the displacement of polygon and line nodes) and integer variables (which in our approach are constrained to values 0 or 1, to model whether an object is unselected or selected, respectively).

By formalizing our problem as a mixed-integer quadratic program (MIQP), we can directly apply existing mathematical solvers. For our experiments, we used the solver Gurobi (Anand *et al.* 2017), which guarantees an optimal solution as output. While this guarantee comes at the cost of an exponential worst-case running time, our exact method is fast enough for generating optimal solutions that provide meaningful insights about the effectiveness of our mathematical model and that can serve as benchmarks for heuristic methods. Specifically, we present an efficient heuristic based on a relaxation of our MIQP.

To summarize, we make the following concrete contributions:

1. We present a mathematical model unifying the selection of objects and the movement of their nodes (displacement). The aim is to resolve a set of proximity conflicts that have been found in advance, meaning that the detection of conflicts is not part of our contribution. In our experiments, the conflicts are detected with an existing triangulation-based method (van Dijk *et al.* 2013). Our model has the form of an MIQP, which can be solved with exact methods implemented in mathematical solvers.
2. We show how our exact method can be easily transformed into an efficient inexact method. Specifically, we relax the MIQP in the sense that we admit fractional values for the discrete variables. The relaxed model has the form of a convex quadratic program (CQP), which can be solved efficiently. We compute an optimal solution to the CQP and apply a rounding heuristic to it.

As it is common in the literature on mathematical programming, we use the term *constraint* exclusively for hard constraints, i.e. constraints that have to be strictly satisfied. Subject to a set of constraint, our methods minimize a function measuring to which degree a set of mathematically defined goals are unfulfilled, e.g. the goal to keep nodes of objects at their original positions. Note that in the literature on map generalization, such goals are also sometimes referred to as constraints (Harrie 1999; Harrie and Sarjakoski 2002).

Our methods combine the displacement and selection of buildings and roads but do not aggregate buildings. This approach is appropriate if the geometric accuracy of the output map is not too important (hence displacement), the completeness of the set of buildings is not too important (hence selection), but the buildings should be represented individually (hence no aggregation). Individual buildings are commonly shown in large-scale topographic maps. At a scale of 1:25,000, displacement is necessary and some buildings need to be omitted to avoid graphic conflicts (Ware *et al.*

2003). In maps that are designed for navigation tasks, even large geometric distortions can be acceptable (Agrawala and Stolte 2001). Visual representations of individual buildings are highly relevant for navigation as they can serve as landmarks (Elias *et al.* 2005; Kapaj *et al.* 2023).

We evaluate both our exact and our heuristic methods on datasets of building footprints and roads from OpenStreetMap. For both methods, we assess the running times and the quality of the solutions. Moreover, we also compare solutions of our methods with solutions that we extracted from the publication of Ware *et al.* (2003), which also aimed at a combination of displacement and selection. However, the combination of displacement and selection is not our ultimate goal. We rather consider our work, as an important step towards a new holistic map generalization method, which should ultimately also include the aggregation of individual buildings to larger built-up areas as well as the enlargement of objects and simplification of lines. Moreover, although we evaluate our approach for datasets of buildings and roads, we see the potential of our method to also generalize other line or area features, such as rivers and lakes.

In the following, we discuss related work on selection and displacement (Sect. 2). Next, we review existing mathematical programming approaches for related tasks and discuss the mathematical relationships between mixed-integer quadratic programming and mathematical programming techniques more commonly used in cartography (Sect. 3). We then introduce our new methods (Sect. 4) and present the experiments we conducted with them (Sect. 5). Finally, we conclude the article and discuss possible future work to address limitations of our approach (Sect. 6).

2. Related work on cartographic displacement and object selection

Resolving proximity conflicts among buildings and roads represented in a map is a common challenge of map generalization. Often this is addressed using displacement and (un)selection of objects. Since we follow the same strategy, we review existing methods for displacement (Sect. 2.1) and selection (Sect. 2.2) in the following.

2.1. Cartographic displacement

Cartographic displacement refers to the resolution of proximity conflicts in maps through the movement of objects or their nodes. A common approach is to detect the conflicts first by analyzing the context of the map elements. Next, displacement vectors are calculated based on specific constraints or rules. Finally, the spatial conflicts are resolved by moving the map elements with conflicts. For example, Mackaness (1994) developed a radial displacement method for solving the spatial conflicts of point sets by cluster analysis. To preserve the spatial distribution patterns of point sets, a density function that can handle the displacement decay was applied. Ruas (1998) presented a displacement method, emphasizing the importance of evaluating the input map and the map after each step of an iterative displacement algorithm. Basaraner (2011) developed an iterative method for building displacement based on Voronoi-based generalization zones. A spatial analysis and multiple criteria were applied to determine the displacement distances and directions. Ai *et al.* (2015)

applied a vector field model inspired by physics for the displacement of building conflicts. In this method, the displacement distance, displacement decay, and secondary conflicts are handled by simulating a force field.

Several optimization methods have been proposed to deal with cartographic displacement together with other tasks of map generalization. Ware and Jones (1998) proposed a method for resolving spatial conflicts based on simulated annealing and gradient descent. The authors considered different generalization operators in these methods, including exaggeration, reduction, and displacement, to choose the new position of a building from a set of candidate positions. Wilson *et al.* (2003) considered object displacement and scaling in a genetic algorithm approach, where candidate solutions are encoded with chromosomes. The solution is iteratively modified using different genetic operations, including mutation, crossover, and selection. To improve the method by Wilson *et al.* (2003), Sun *et al.* (2016) and Huang *et al.* (2017) extended the genetic algorithm by considering different arrangement patterns of buildings, their spatial relationships, and topological structures. In addition, some scholars also introduced ideas from other subject fields such as engineering and physics to build mathematical models for displacement. For example, Harrie (1999, 2003) and Sester (2005) applied the least-squares adjustment method for building displacement and simplification. Højholt (2000) introduced the deformation of an elastic body to model the displacement of buildings. The finite element method is applied to discretize the map space, and boundary constraints are used to change the size and shape of the map elements. This method can effectively preserve the distribution patterns and spatial relationships of buildings, without distorting the map elements too much.

Since least-squares adjustment turned out to be appropriate for cartographic displacement, we developed our new method based on a similar principle. Specifically, we use quadratic energy terms to quantify distortions and displacements.

2.2. Selection of objects

To realize selection, many methods designed for point clusters, roads, and buildings have been proposed by previous studies. Töpfer and Pillewizer (1966) developed the radical law for determining the number of map elements at different levels of detail, which has been widely applied for selection tasks in map generalization. Subsequently, methods based on Voronoi diagrams (Yan and Weibel 2008; Lu *et al.* 2019), Delaunay triangulations (Ai and Liu 2004), and quadtrees (Peters 2013) were introduced for the selection of points. Selection methods for linear features were mainly developed for roads and rivers (Jiang and Harrie 2004; Chen *et al.* 2009; Li *et al.* 2019; Mazur and Castner 1990; Thomson and Brooks 2000; Ai *et al.* 2006; Li *et al.* 2018). For example, Chen *et al.* (2009) developed a road selection method by calculating the mesh density of roads, especially considering semantic, topological, and geometrical properties. Li *et al.* (2019) proposed a railway selection method that can well preserve the structural features of railways and the connectivity of railway stations. For river selection, Mazur and Castner (1990) applied Horton's rule to order the streams during generalization. Ai *et al.* (2006) used the Delaunay triangulation to detect the watershed regions of rivers and perform the selection by calculating the sizes of the

watershed regions. The design of building selection methods including building typification has also attracted the attention of many scholars (Regnauld 2001; Burghardt and Cecconi 2007; Gong and Wu 2018; Wang and Burghardt 2019). For example, Regnauld (2001) proposed a building typification method combining different Gestalt rules, which can well preserve the distributions, shapes, sizes, and density of buildings. Based on graph theory and the Delaunay triangulation, Gong and Wu (2018) also developed a building typification method that can maintain linear distribution patterns of buildings. In recent years, Shen *et al.* (2022a, 2022b) developed raster-based methods for building selection and typification methods based on superpixel segmentation, which considers the distribution patterns as well as the geometric and semantic characteristics of buildings.

Our main conclusion for the development of our model is that the selection of buildings should not only be driven by proximity conflicts but also by patterns that should be preserved, such as multiple buildings aligned in a row (Gong and Wu 2018). The selection of roads should be based on connected sequences of segments, and the connectivity of the road network has to be preserved (Thomson and Brooks 2000).

3. Mathematical programming and its applications in map generalization

Mathematical programming means to state a computational problem as a mathematical program, which consists of a set of variables, an optimization objective, and a set of constraints. One commonly distinguishes different types of mathematical programs based on the types of the variables and the form of the objective function and constraints. Many types of mathematical programs can be solved with existing solvers.

In the context of cartographic displacement, least-squares adjustment is frequently applied (Harrie 1999; Sester 2000; Harrie and Sarjakoski 2002; Zhang *et al.* 2006; Harrie and Weibel 2007; van Dijk and Haunert 2014; Touya and Lokhat 2022), where the aim is to satisfy an overconstrained system of equations $A\mathbf{x} = \mathbf{b}$ with unknowns $\mathbf{x} \in \mathbb{R}^u$ and constants $A \in \mathbb{R}^{n \times u}$ and $\mathbf{b} \in \mathbb{R}^n$ as much as possible. To achieve this, the aim is to minimize $f(\mathbf{x}) = \|A\mathbf{x} - \mathbf{b}\|^2$.

Least-squares adjustment is a special case of convex quadratic programming, which in turn is a special case of convex programming (Boyd and Vandenberghe 2004). Generally, the term 'convex' in convex programming refers to the fact that (i) the set of feasible solutions defined with the constraints is convex (i.e., for every $\alpha \in [0, 1]$ the convex combination $\alpha \cdot \mathbf{x}' + (1 - \alpha) \cdot \mathbf{x}''$ of any two solutions $\mathbf{x}'$ and $\mathbf{x}''$ is again a solution) and (ii) the objective is to minimize a convex function f (i.e., f bounds a convex set from below). Due to the convexity of the objective function f and the convexity of the solution space, any local optimum of a convex program is automatically a global optimum. This observation has led to highly efficient convex programming solvers.

In a convex quadratic program (CQP), the variables are $\mathbf{x} \in \mathbb{R}^n$, the objective is to minimize a function $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A \mathbf{x} + \mathbf{b}^T \mathbf{x} + c$, and the constraints are $\mathbf{x} \geq 0$ and $D\mathbf{x} \geq \mathbf{e}$, where $A \in \mathbb{R}^{n \times n}$, $\mathbf{b} \in \mathbb{R}^n$, $c \in \mathbb{R}$, $D \in \mathbb{R}^{m \times n}$, and $\mathbf{e} \in \mathbb{R}^m$ are prescribed constants and A is positive semidefinite. Haunert and Sering (2011) used this form to enlarge focus regions in road network maps and to optimally distribute the resulting distortions. Another prominent special case of convex programming is linear programming, where

other than in convex quadratic programming the minimization objective has no quadratic term, i.e., the objective function of a linear program (LP) has the form $f(\mathbf{x}) = \mathbf{b}^T \mathbf{x} + c$. Hirono *et al.* (2013) used linear programming to displace three-dimensional buildings in order to disocclude routes and landmarks in maps for navigation tasks. More recently, Nickel *et al.* (2022) applied linear programming to displace squares representing countries in Demers cartograms.

Non-convex programs are much harder to solve than convex programs, but also their expressiveness is much larger. Especially mixed-integer linear programming has been applied for various tasks of map generalization and cartographic label placement that have a high computational complexity (in particular, for NP-hard problems). The difference to linear programming is that some of the variables are constrained to integer values, i.e., a mixed-integer linear program (MILP) has variables $\mathbf{x}^T = (\mathbf{x}_{\text{real}}^T, \mathbf{x}_{\text{int}}^T)$ with $\mathbf{x}_{\text{real}} \in \mathbb{R}^{n_{\text{real}}}$ and $\mathbf{x}_{\text{int}} \in \mathbb{Z}^{n_{\text{int}}}$. With integer variables constrained to 0 or 1 it is possible to model discrete states, such as the unselection or selection of objects. This principle has found wide application in cartography, e.g., to select point features (Schwartges *et al.* 2013), edges of graphs representing road networks (Chimani *et al.* 2014), local simplifications of lines called shortcuts (Haunert and Wolff 2010b), and text labels (Zoraster 1990; Haunert and Wolff 2017). Moreover, integer linear programs with 0–1 variables have been developed for clustering tasks, where the variables model possible assignments of objects to cluster centers. This principle has been applied in map generalization to group multiple polygon objects and aggregate the elements within each group to a single output polygon (Haunert and Wolff 2010a; Oehrlin and Haunert 2017; Peng *et al.* 2021; Gedicke *et al.* 2021). Guercke *et al.* (2011) applied the same concept to the generalization of 3D building models.

The mixed-integer quadratic programming formulation that we contribute in this article has the form of a convex quadratic program with both continuous and integer variables, i.e. $\mathbf{x}^T = (\mathbf{x}_{\text{real}}^T, \mathbf{x}_{\text{int}}^T)$ as stated above. In other words, the non-convexity of the problem is due to the discrete nature of its variables and not due to its objective function or constraints. Relaxing the restriction that $\mathbf{x}_{\text{int}}$ has to be integer yields an efficiently solvable CQP. We use this property of our MIQP to derive an efficient rounding heuristic (see Sect. 4.7).

To our knowledge, mixed-integer quadratic programming has not been applied to map generalization yet. In the more general context of geometry processing, Huang *et al.* (2023) presented an MIQP-based method for the symmetrization of polygons.

4. Methodology

In this section, we give a detailed presentation of our new approach. We present the design goals underlying the approach (Sect. 4.1) and provide an overview of the workflow (Sect. 4.2). Next, we discuss the preprocessing steps that are part of this workflow in detail (Sect. 4.3). To describe the main step of our approach, the optimization, we explain the fundamental mechanism of our model (Sect. 4.4), a basic MIQP that deals with the most important design goals (Sect. 4.5), and extensions to deal with the remaining design goals (Sect. 4.6). Finally, we present a heuristic (Sect. 4.7) and conclude the section with an overview of the parameters of our methods (Sect. 4.8).

4.1. Design goals

On an abstract level, our approach follows seven design goals.

Design goals G1–G4 deal with the selection of objects:

- G1: Priorities of map objects that are provided as input should be respected.
- G2: Dependencies between pairs of objects have to be respected (e.g., if a building is selected, the road leading to it must also be selected).
- G3: When unselecting buildings, patterns of multiple buildings should be preserved.
- G4: When unselecting roads, the road network has to remain connected.

Design goals G5–G7 deal with the positions of the nodes of the selected objects:

- G5: Positions close to the original node positions should be favored.
- G6: Proximity conflicts should be resolved.
- G7: Distortions should be low. We here understand distortions as deviations from preferred relative positions of nodes. They are measured (G7a) at edges of objects and (G7b) at pairs of nodes that represent characteristic proximity relations of objects.

Our approach is flexible with respect to how the priorities of objects are set for G1 and which dependencies between objects are considered for G2. The priorities of buildings could be determined with importance rating algorithms that consider application-specific criteria and the spatial context. For example, if the map is used for navigation, important buildings are those that can serve as landmarks, e.g., buildings that are clearly visible and distinguishable from other buildings in the vicinity (Elias 2003). However, in the experiments, we used a basic setting, simply prioritizing buildings according to their areas.

In its current form, our method does not fulfill G3 for arbitrary patterns of buildings. However, we introduce weak dependencies between adjacent buildings. More precisely, we add a term to the objective function that penalizes the unselection of a building if an adjacent building is selected, where two buildings are considered adjacent if their polygons share at least one edge. This, in particular, has the effect that sequences of multiple row houses are preserved. The preservation of other types of building patterns, such as a set of buildings arranged on a regular grid, is a topic for future research. Generally, we consider it promising to detect patterns before setting up and solving the optimization problem. For example, if it was determined that a set of buildings in the input data set forms a grid pattern, the optimization problem should be set up with additional constraints or objectives, favoring solutions where the selected subset has a similar characteristic.

4.2. Overview of the workflow

Figure 1 provides an overview of our workflow. As input for our method we require a geometric graph $G = (V, E)$ that represents the roads and boundaries of buildings; see Figure 1(a). We apply preprocessing steps (from Figure 1(a) to (d)) to achieve an

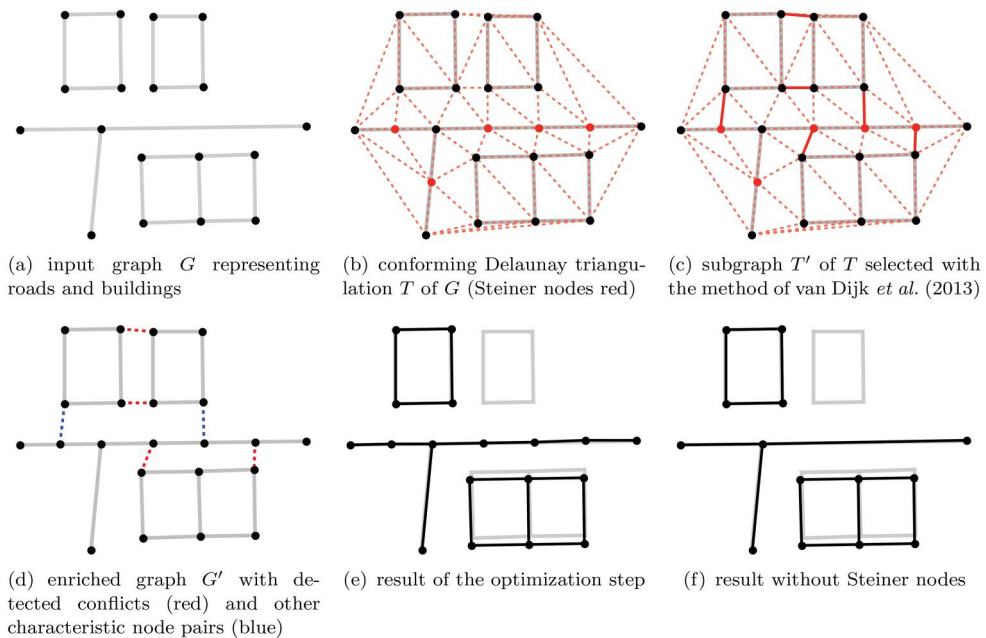


Figure 1. Overview of our workflow. The steps from (a) to (d) are referred to as preprocessing. The step from (d) to (e) is the optimization step, which is the focus of our work. The step from (e) to (f) is referred to as postprocessing.

enriched graph $G' = (V', E')$ that we use as input for the optimization step. These preprocessing steps are dealing with computing a conforming Delaunay triangulation of G (Saalfeld 1991), selecting a subgraph of interesting edges from the triangulation (see Sect. 4.3 for details), and removing redundant nodes in the selected subgraph (i.e., nodes of degree two whose two incident edges have the same orientation). The result of the subsequent optimization step is shown in Figure 1(e), where one building has been unselected and some displacement occurred. In a final postprocessing step, the nodes that were introduced by the conforming Delaunay triangulation algorithm (i.e. Steiner nodes) are removed from the graph; see Figure 1(f).

In some situations, it can be reasonable to apply a simplification algorithm prior to our workflow, in order to avoid that the enriched graph G' represents too many details. However, we did not simplify the data that we retrieved for our experiments (see Sect. 5) since its level of detail seemed appropriate.

A detailed presentation of the workflow for an example is given in Appendix.

4.3. Preprocessing

To compute the enriched geometric graph G' from the input graph G , we first compute a conforming Delaunay triangulation T of G ; see Figure 1(b). While many methods for displacement are based on a triangulation, not all methods use the full triangulation for the optimization step. For example, Sester (2005) uses all edges of a triangulation, but Harrie and Sarjakoski (2002) consider only the edges that belong to

objects or represent conflicts. They argue that the lengths of other triangle edges are not interesting from a cartographic point of view. We choose a compromise between these two approaches by selecting a preferably small subgraph T' of T that satisfies the following requirements:

- (R1) T' must include every edge of T that is covered by an edge of G (i.e. edges belonging to objects).
- (R2) For every edge $\{u, v\}$ of T it must hold that $d_{T'}(u, v)/d(u, v) \leq t$, where $d_{T'}$ denotes the geodesic distance in T' (i.e. the geometric length of a geometrically shortest path in T'), d denotes the Euclidean distance, and $t \geq 1$ is a parameter that can be chosen to control how densely T' is connected. Setting $t = 1$ yields $T' = T$. For increasing t , the density of T' decreases.

The motivation for Requirement R2 is that two nodes u, v with large $d_{T'}(u, v)/d(u, v)$ are closely related to each other (geometrically near) but are lacking a good connection in T' . Without improving this connection, there is the risk that the characteristic proximity relationship between the two nodes will be lost in the optimization step. By adding too many edges to T' , however, we would likely lose the flexibility that is needed to appropriately resolve the conflicts.

To satisfy Requirements R1 and R2 when selecting T' from T , we choose an iterative greedy heuristic proposed by van Dijk *et al.* (2013). It initially sets T' to include all edges belonging to objects, thus satisfying Requirement R1. Then, the method iterates over the edges in T in increasing order of their lengths. In every iteration, it checks the current edge $e = \{u, v\}$ and computes $d_{T'}(u, v)/d(u, v)$ with respect to the current graph T' . If this ratio exceeds t , the method adds e to T' .

With this method, we find bottlenecks of the faces of the current graph T' , i.e. every edge that we add cuts one of the faces where the face is relatively narrow. Therefore, we call the additional edges *bottleneck edges*. Bottleneck edges that are shorter than a prescribed threshold $\varepsilon > 0$ represent conflicts that should be resolved by displacement. Bottleneck edges of length at least ε represent characteristic relative positions between nodes that should be preserved during generalization. Consequently, we will measure the distortion of the map at the edges belonging to objects (design goal G7a) and at the bottleneck edges (design goal G7b). The resulting graph T' is shown in Figure 1(c).

To avoid superfluous nodes in the input for the optimization step, we remove every Steiner node v from T' for which no incident edge was selected as a bottleneck edge. When removing v , we replace its two incident edges $\{u, v\}$ and $\{v, w\}$ with a single edge $\{u, w\}$, resulting in the enriched graph G' , see Figure 1(d).

Finally, we impose an arbitrary direction on each edge of G' . The direction of an edge $(u, v) \in E'$ does not have any meaning, except that it allows us to unambiguously distinguish between the source node u and the target node v of e . Therefore, in the following, we refer to G' as a directed graph.

We would like to point out that the edges of the conforming Delaunay triangulation T of G only approximately represent smallest distances between objects. However, the approximation error is relatively small if the level of detail of the input data is

high. For a better representation of smallest distances, one could subdivide edges of G .

4.4. Fundamental mechanism

Once the enriched graph $G' = (V', E')$ has been computed, the optimization method continues with the distortion of G' and the selection of objects. For this, we assume that we are given a partition of E' into a set E_{obj} that represents the roads and buildings (gray edges in Figure 1(d)), and the set E_{bot} of bottleneck edges (red and blue edges in Figure 1(d)).

Each building and each road are given as a subset of E_{obj} . The edge set for a road may either follow the definition of a road object in the source data or correspond to a sequence of line segments that has been identified as a stroke in a pre-processing step (Thomson and Brooks 2000). The edge sets for two buildings overlap if the buildings share a wall. With B , we refer to the set of all buildings and with R to the set of all roads. The following definition summarizes the fundamental mechanism underlying our model.

Fundamental mechanism. Every edge $e \in E'$ depends on a set of objects, in the sense that a deviation of e from its *desired extent* is penalized in the objective function if and only if a certain condition over the selection states of the objects holds. An edge satisfying this condition is called *active*.

We first define the desired extent of the edges and then specify the conditions for active edges.

- For every bottleneck edge $e = (u, v)$ that is shorter than the minimally allowed length ε , the desired extent of e corresponds to the original extent of e scaled by $\varepsilon/d(u, v)$, where $d(u, v)$ is the distance between the two incident nodes u and v of e . With this we try to resolve proximity conflicts (design goal G6).
- For every other edge, the desired extent is equal to the original extent. This includes edges belonging to objects (design goal G7a) but also bottleneck edges of length at least ε (design goal G7b), which are depicted blue in Figure 1(d).

The condition under which an edge is active is defined as a logical expression that consists of at most two OR clauses that are combined with AND. Each OR clause for an edge e corresponds to a set of objects. Together, these sets of objects for e constitute a set $S(e)$. For example, for edge e in Figure 2(a), we define $S(e) = \{\{b_1\}, \{r_1, r_2\}\}$. Consequently, e is active if and only if (i) b_1 is selected and (ii) r_1 or r_2 is selected. More generally, we use the following rule:

- If e is a bottleneck edge, we add for each of its incident nodes u a set to $S(e)$. If u belongs to a single road or building, the corresponding set in $S(e)$ contains only this object. If u belongs to multiple objects, the corresponding set in $S(e)$ contains all of them.

For edges belonging to objects, the following two rules are used:

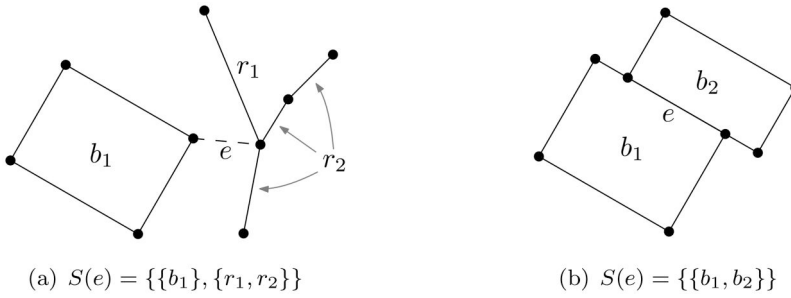


Figure 2. The set $S(e)$ determining the activity of an edge e for two cases. Generally, e is active if for each set of objects in $S(e)$ at least one object is selected.

- If e belongs to a single building or road a , then $S(e) = \{\{a\}\}$. This means that the distortion of e is considered if and only if a is selected.
- If e belongs to two buildings b_1 and b_2 , as it is the case for edge e_1 in [Figure 2\(b\)](#), we set $S(e) = \{\{b_1, b_2\}\}$. This means that the distortion of e is considered if and only if b_1 or b_2 is selected.

To simplify the discussion in the following, if $S(e)$ contains a set of two or more objects, we consider this set itself as an object. We call every such additional object a *collector*. With C , we refer to the set of all collectors. Furthermore, we introduce $O = R \cup B \cup C$ as the set of all objects (i.e. roads, buildings, and collectors). Since a collector corresponds to an OR clause, we consider it selected if at least one object in it is selected. For every edge e , we introduce a set of at most two associated objects $O(e) \subseteq O$. For every set in $S(e)$ that contains only one object, that object is contained in $O(e)$. For every other set in $S(e)$, $O(e)$ contains the corresponding collector.

4.5. A basic mixed-integer quadratic program

In addition to the enriched graph $G' = (V', E')$, the partition of E' into the set of object edges E_{obj} and the set of bottleneck edges E_{bot} , the set of objects O and its partition into the three sets B , R , and C , and the set $O(e)$ of associated objects for each edge e , we require the following parameters as input:

- $\varepsilon \in \mathbb{R}_{\geq 0}$: The minimal length required for the edges representing conflicts.
- $w_{\text{select}}, w_{\text{pos}}, w_{\text{edge}} \in [0, 1]$: The weights expressing the general priorities of design goal G1, design goal G5, and the goal to achieve the desired edge extents (modeling design goals G6 and G7).
- $w_{uv} \in \mathbb{R}_{\geq 0}$ for each $(u, v) \in E'$: An edge-specific weight expressing the priority to achieve the desired extent for edge (u, v) (also allowing the differentiation between design goals G6 and G7).
- $w_o \in \mathbb{R}_{\geq 0}$ for each $o \in B \cup R$: A weight representing the importance of object o .

Concrete parameter settings will be presented with our experiments, in Sect. 5.

With this, we set up the optimization problem in the form of an MIQP, which consists of the variables, the objective function, and the constraints specified as follows.

The *variables* are of three different types:

- For each node $v \in V'$, the continuous variables $dx'_v, dy'_v \in \mathbb{R}$ encode the movement of v .
- For each object $o \in O$, the binary variable $z_o \in \{0, 1\}$ models whether o is selected for the output map ($z_o = 1$) or not selected ($z_o = 0$).
- For each edge $e = (u, v) \in E'$, we introduce the continuous variables $\delta x'_{uv}, \delta y'_{uv} \in \mathbb{R}$. If e is active, we ensure with constraints that $\delta x'_{uv}$ and $\delta y'_{uv}$ measure the residuals for e , i.e., the difference between the desired extent of e and the extent of e after its distortion. If e is not active, we ensure that the constraints imposed on $\delta x'_{uv}$ and $\delta y'_{uv}$ are relaxed, such that $\delta x'_{uv} = 0$ and $\delta y'_{uv} = 0$ in every optimal solution.

The *objective function* consists of three parts.

$$\begin{aligned} \text{Minimize } f_{\text{basic}} = & w_{\text{pos}} \cdot \sum_{v \in V'} \left((dx'_v)^2 + (dy'_v)^2 \right) \\ & + w_{\text{edge}} \cdot \sum_{(u, v) \in E'} w_{uv} \cdot \left((\delta x'_{uv})^2 + (\delta y'_{uv})^2 \right) \\ & + w_{\text{select}} \cdot \sum_{o \in B \cup R} w_o \cdot (1 - z_o) \end{aligned} \quad (1)$$

The first, second, and third part of the objective functions are weighted by w_{pos} , w_{edge} , w_{select} , respectively. The reason for these parts is as follows:

- The first part measures the sum of the squared movements in x - and y -direction over all nodes, which means that each node contributes a cost that is proportional to the squared Euclidean distance between its original and new position. This has the effect of a quadratic energy attracting the nodes to their original positions.
- The second part of the objective penalizes, for every active edge $e \in E'$, the deviation of the extent of $e = (u, v)$ from its desired extent. We apply the edge-specific factor $w_{uv} \in \mathbb{R}_{\geq 0}$ to the cost for e to take into account that the same amount of distortion may be more or less tolerable for different edges.
- The third part adds a cost of w_o for every non-selected object o . With this important objects (i.e., objects with large weights) are likely selected.

The *constraints* of our basic model first of all ensure that the map stays within the prescribed box $[x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}]$. More precisely, we require for each $u \in V$

$$x_{\min} \leq x_u + dx'_u \leq x_{\max} \quad (2)$$

$$y_{\min} \leq y_u + dy'_u \leq y_{\max}, \quad (3)$$

where $x_{\min} = \min_{u \in V'} \{x_u\}$, $x_{\max} = \max_{u \in V'} \{x_u\}$, $y_{\min} = \min_{u \in V'} \{y_u\}$, and $y_{\max} = \max_{u \in V'} \{y_u\}$. Furthermore, we introduce a constraint for each collector $c \in C$ and each object $o \in c$, to ensure that c is selected if o is selected and, thus, that c has the effect of an OR clause.

$$z_c \geq z_o \quad (4)$$

Next, we ensure that the different types of variables are correctly coupled, such that the variables' values in an optimal solution conform to the variables' definitions stated above. We achieve this with four inequality constraints for each edge $e = (u, v) \in E'$, which we first introduce and then explain.

$$\delta x'_{uv} \geq ((x_u + dx'_u) - (x_v + dx'_v)) - s(x_u - x_v) - M \sum_{o \in O(e)} (1 - z_o) \quad (5)$$

$$\delta y'_{uv} \geq ((y_u + dy'_u) - (y_v + dy'_v)) - s(y_u - y_v) - M \sum_{o \in O(e)} (1 - z_o) \quad (6)$$

$$\delta x'_{uv} \geq -((x_u + dx'_u) - (x_v + dx'_v)) + s(x_u - x_v) - M \sum_{o \in O(e)} (1 - z_o) \quad (7)$$

$$\delta y'_{uv} \geq -((y_u + dy'_u) - (y_v + dy'_v)) + s(y_u - y_v) - M \sum_{o \in O(e)} (1 - z_o) \quad (8)$$

where

$$s = \begin{cases} \frac{\varepsilon}{d(u, v)} & \text{if } d(u, v) < \varepsilon \text{ and } (u, v) \in E_{\text{bot}} \\ 1 & \text{else} \end{cases} \quad (9)$$

and M is a large constant. More precisely, we set

$$M = \max\{2(x_{\max} - x_{\min}), 2(y_{\max} - y_{\min})\}. \quad (10)$$

To explain the effect of Constraints (5)–(8) for edge $e = (u, v)$, we first consider the case that the objects in $O(e)$ are selected, i.e. $z_o = 1$ for all $o \in O(e)$. In this case, the term with factor M in each inequality vanishes. Moreover, the right-hand sides of Constraints (5) and (7) differ only with respect to their signs, thus together they imply $\delta x'_{uv} \geq |((x_u + dx'_u) - (x_v + dx'_v)) - s(x_u - x_v)|$. In an optimal solution, $\delta x'_{uv}$ will be set to the smallest possible value, which implies $\delta x'_{uv} = |((x_u + dx'_u) - (x_v + dx'_v)) - s(x_u - x_v)|$. Here, $((x_u + dx'_u) - (x_v + dx'_v))$ is the extent of edge e in the x -dimension after the displacement of nodes. By choosing s according to Equation (9), we implement the resolution of proximity conflicts (design goal G6) in such a way that an edge representing a conflict should be scaled to achieve a length of ε , while every other edge should preserve its extent. Hence, $\delta x'_{uv}$ correctly models the difference between the resulting extent and the desired extent of edge e in the x -dimension. Constraints (6) and (8) ensure the same for $\delta y'_{uv}$ with respect to the y -dimension. Next, we consider the case that at least one object $o \in O(e)$ is not selected, i.e., $z_o = 0$. In this case, the constant M is subtracted at least once from the right-hand side of each of the four inequalities. With the definition of M in Equation (10) and assuming $\varepsilon < x_{\max} - x_{\min}$ and $\varepsilon < y_{\max} - y_{\min}$, this subtraction implies a relaxation of the four constraints, i.e. they are always fulfilled. Consequently, in an optimal solution, it holds that $\delta x'_{uv} = \delta y'_{uv} = 0$. This means that, if e is not active, there is no cost for its distortion.

4.6. Model extensions

4.6.1. Coupled selection of roads and buildings

We can model dependencies between any two objects (design goal G2) with additional constraints. In particular, in the context of buildings and roads, we allow the selection of a building only if the road leading to it is selected as well. Therefore, we compute for every building $b \in B$ the road $r \in R$ nearest to the centroid of b and denote it with $r(b)$.

We then add the following constraint to couple the selection of these two objects.

$$z_{r(b)} \geq z_b \quad (11)$$

Alternatively, the road $r(b)$ associated with building b could be defined using additional address information.

4.6.2. Coupled selection of multiple buildings

From our experiments with the basic model, we learned that the set of buildings selected for an optimal solution sometimes contains unfavorable gaps. For example, a single house in the middle of a sequence of row houses was removed, which led to a solution with relatively little distortion of the active edges. Although this solution conforms with the design goals based on which the basic model was set up, it does not conform with the overarching goal of map generalization to preserve characteristic patterns in the data. In particular, the sequence of row houses is not preserved as a sequence. Therefore, we model as an additional objective that adjacent buildings should be treated the same. More precisely, we consider the adjacency graph of all buildings, i.e. the graph with node set B and with an edge between every two buildings in B that share an edge in E' . We reduce this graph to its connected components of at least three buildings, resulting in a graph whose edge set we denote with A . For every two buildings b and b' connected by an edge in A , if one building is selected and the other one not selected, we charge a certain cost. To express this objective in mathematical terms, we introduce an additional variable $z_{bb'} \in \{0, 1\}$ for each $\{b, b'\} \in A$. Using the following two constraints, we ensure that $z_{bb'} = 1$ if one of the two buildings is selected and the other one not selected.

$$z_{bb'} \geq z_b - z_{b'} \quad (12)$$

$$z_{bb'} \geq z_{b'} - z_b \quad (13)$$

With this, we are able to add a fourth term to our objective function that penalizes solutions in which a building is selected and a building adjacent to it is not selected.

$$f_{\text{advanced}} = f_{\text{basic}} + w_{\text{depend}} \cdot \sum_{\{b, b'\} \in A} z_{bb'} \quad (14)$$

Here, $w_{\text{depend}} \in [0, 1]$ needs to be chosen to express the priority of the additional term.

Instead of only looking at adjacent buildings, we could introduce weak dependencies for arbitrary pairs of buildings, e.g. a pair of buildings that from given thematic information are known to belong to the same hospital or university.

4.6.3. Connectivity of the road network

An important requirement when selecting roads is to keep the road network connected. We enforce the global connectivity of the road network by adapting the flow model of Shirabe (2005). Generally, with this model one can enforce the connectivity of a graph H' when selecting it as a subgraph of a given graph H . In our application, H has the node set R and its edge set E_H contains an edge for every two roads that share a node in G . Since the model of Shirabe is well documented in the literature,

and its adaptation is rather straight-forward, we only present the additional variables and constraints coming with the model and give a very brief explanation.

Additional variables:

- For each edge $\{r, r'\} \in E_H$ that share a node, the variables $f_{rr'}, f_{r'r} \in \mathbb{R}_{\geq 0}$ model the amount of a commodity that flows from r to r' and from r' to r .
- For each road $r \in R$, the variable $s_r \in \{0, 1\}$ encodes whether r acts as a sink of the flow network ($s_r = 1$) or not ($s_r = 0$).

Additional constraints:

$$\sum_{r \in R} s_r = 1 \quad (15)$$

$$s_r \leq z_r \quad \text{for each } r \in R \quad (16)$$

$$\sum_{\{r, r'\} \in E_H} f_{rr'} - \sum_{\{r, r'\} \in E_H} f_{r'r} \geq z_r - |R| \cdot s_r \quad \text{for each } r \in R \quad (17)$$

$$\sum_{\{r, r'\} \in E_H} f_{rr'} \leq (|R| - 1) \cdot z_r \quad \text{for each } r \in R \quad (18)$$

The connectivity of the subgraph of H induced by the selected roads is ensured by enforcing that there is a single sink (Equation (15)), that this sink is in the selected subgraph (Equation (16)), that a non-selected node has neither incoming nor outgoing flow (Equations (17) and (18) for $z_r = 0$), and that all selected nodes except the sink contribute a positive amount of flow (Equation (17) for $z_r = 1$).

4.7. Heuristic method

Our heuristic approach consists of the following steps:

1. Solve the relaxation of the MIQP in which the integer variables are allowed to receive fractional values. This relaxation is a convex quadratic program (CQP), which can be solved efficiently. For each object o in the solution to this relaxation, the variable z_o has a value in $[0, 1]$.
2. For each object $o \in O$ fulfilling $z_o \geq \theta$, round z_o up to 1 and thus select o , where $\theta \in [0, 1]$ is a user-set threshold. For each other object o , set $z_o = 0$.
3. Solve the CQP again but keep the value of z_o for every object $o \in O$ fixed to the value resulting from the previous step. Return the solution found.

In Step 3, when fixing $z_o = 1$ for some object o , we apply several simplifications to the CQP to speed up its solution. In particular, in Equations (5) and (6), the terms with M disappear. The inequality relation can be replaced by equality since the sign of $\delta x'_{uv}$ and $\delta y'_{uv}$ has no effect on the optimization if the objects are fixed. Thus, Equations (7) and (8) can be omitted.

Also for the heuristic, we can formulate goals supporting the connectivity of rows of buildings in the same manner as before, but now the variables $z_{bb'}$ are continuous with values in the interval $[0, 1]$.

In order to guarantee the connectivity of the road network, we apply the following strategy between Steps 2 and 3: If the roads chosen in Step 2 are disconnected,

Table 1. Input parameters of our methods.

t	Threshold used in the preprocessing to control the density of the graph G' used in the optimization step. With larger t , G' gets less dense, which implies that distortions are measured only at most critical bottlenecks.
ε	Bottleneck edges shorter than this threshold are considered as conflicts.
w_{pos}	Weight expressing the priority to keep original node positions
w_{edge}	Weight expressing the priority to generally keep distortions at edges low.
w_{select}	Weight expressing the priority to generally keep objects selected.
w_{depend}	Weight expressing the priority to respect weak dependencies. (We used a weak dependency for each two adjacent buildings to preserve rows of buildings, but also other pairs of objects could be considered.)
w_o	Weight expressing the priority to select object o .
w_{uv}	Weight expressing the priority to keep the distortion at edge uv low.
θ	Threshold is used by the heuristic to decide which variables are rounded up.

iteratively add roads from the set of the unselected roads, ordered descendingly by their corresponding values z_o . Skip roads which are only adjacent to roads which have already been added and lie in one connected component.

4.8. Summary of parameters

We conclude this section with an overview of the parameters that we have introduced along the development of our model; see Table 1. Concrete parameter settings will be given in the next section, which deals with the experiments that we conducted.

5. Experiments

We implemented our method in Java using the Gurobi library for solving the mathematical programs. In this section, we show some results of our method applied to real-world data that we retrieved from OpenStreetMap.¹ In total, we use 20 datasets of road and building data split into two groups: 15 ‘urban’ datasets where the objects lie within the city of Bonn and mostly cover one or two building blocks and 5 ‘rural’ datasets, each consisting of a small village in the Eifel region south of Bonn. While in the urban datasets most buildings lie within tight meshes of the road network, they are rather grouped around crossings of the roads in the rural datasets. Unless stated differently, the parameters introduced in Sect. 4 are set as follows: $t = 5$, $\varepsilon = 7.5$ m, $w_{\text{pos}} = 0.0001$, $w_{\text{edge}} = 0.8$, $w_{\text{select}} = 0.1999$, $w_{\text{depend}} = 0.5$. This setting was found through the experiments that we discuss in Sect. 5.1. We weight the roads proportionally to their lengths and the buildings proportionally to their areas. By normalizing these weights, the smallest building finally obtains a weight $w_o = 1$ and the shortest road obtains a weight $w_o = 10$. We set the edge-specific weight w_{uv} to 1 unless (u, v) is a bottleneck edge of length $\ell > \varepsilon$: In this case, we set $w_{uv} = \varepsilon^2 / \ell^2$. This means for sufficiently long bottleneck edges design goal G7 is less strict, which allows these edges to be contracted to a larger degree. We found that in some situations this helps to better make use of free map space and thus preserve more objects.

Moreover, we apply the coupled selection of roads and buildings from Sect. 4.6.1. With this, the road network remains connected in our examples, even without the strategies described in Sections 4.6.3 and 4.7.



Figure 3. Example dataset for different parameter settings (grey: input graph G , black: output graph, blue: bottleneck edges of length at least ε , red: bottleneck edges shorter than ε).

In Sections 5.1 and 5.2, we discuss the results of our exact approach and compare it to the approach by Ware *et al.* (2003). Section 5.3 examines the quality of the heuristic approach. In Sect. 5.4, we discuss the running time of the exact and the heuristic approach.

5.1. Exploring different weightings

We now show some results for one of the urban datasets and how they are influenced by the choice of the weights w_{pos} , w_{edge} and w_{select} ; see Figure 3. For that, we start with an example where the displacement of nodes is penalized with a relatively large cost w_{pos} (see Figure 3(a)): The objects stay roughly at their initial locations, but some are removed to resolve the conflicts. The number of unselected objects is relatively large since displacement and distortion come along with large costs and selection remains as a relatively cheap operation to resolve conflicts. In contrast, when using a small value for w_{pos} and a large one for w_{edge} , some objects are displaced but roughly keep their original shapes, since the distortion of the graph edges is penalized; see Figure 3(b). Hence, w_{pos} should be small to allow for both selection and displacement being applied to resolve conflicts. Also, w_{edge} should not be too small. In the extreme case $w_{\text{edge}} = 0$, the bottleneck edges have no influence and no conflict is resolved. In comparison to Figure 3(b), we show in Figure 3(c) the result for the case that unselecting objects is not permitted. In this case, some buildings are visibly distorted, which happens mostly for those that are unselected in Figure 3(b). This shows that the combined approach tends to unselect objects only if they are involved in conflicts and if these conflicts cannot be resolved by only displacing the objects.

In order to visualize the displacements of nodes and the distortions of edges, we define the *absolute displacement* d_v of a node v and the *absolute distortion* δ_{uv} of an edge (u, v) as follows:



Figure 4. Absolute distortion δ_{uv} of each active edge (u, v) of G' .

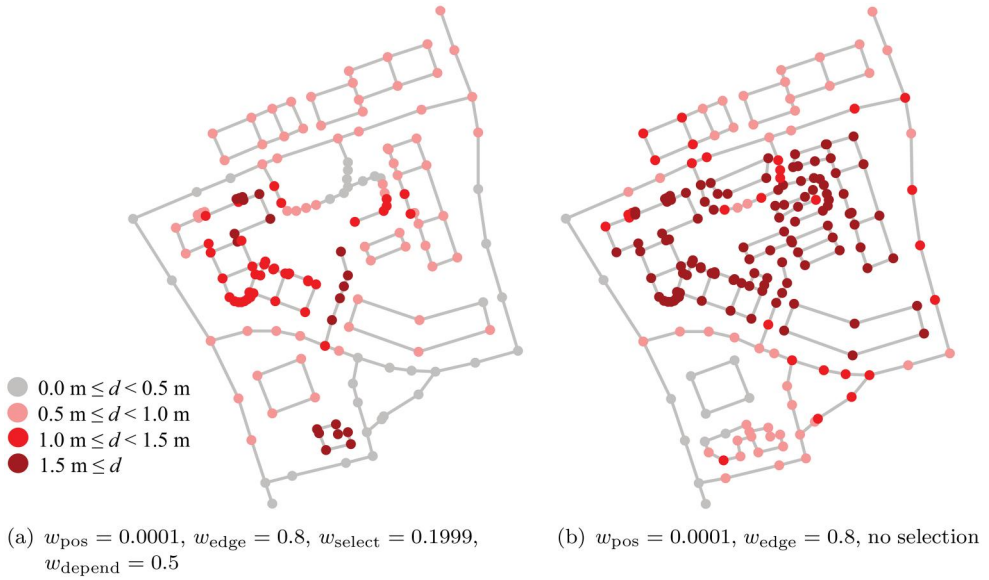


Figure 5. Absolute displacement d_v of each node v of G' that belongs to selected objects.

$$d_v = \sqrt{(dx'_v)^2 + (dy'_v)^2} \quad (19)$$

$$\delta_{uv} = \sqrt{(\delta x'_{uv})^2 + (\delta y'_{uv})^2} \quad (20)$$

Note that these terms occur squared and weighted in the objective function in Equation (1).

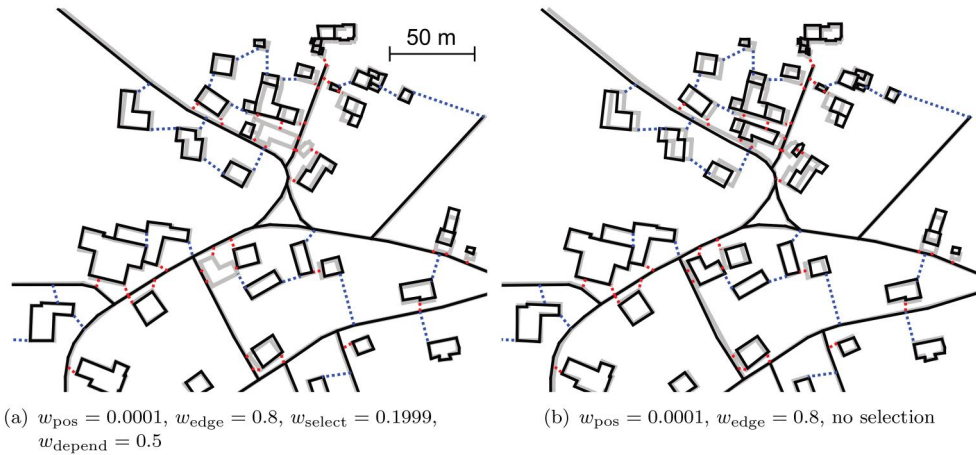


Figure 6. An example of a rural dataset.

In Figure 4, we colored each active edge according to its absolute distortion. It becomes clear that also the bottleneck edges are distorted more if only the displacement operator is applied. Similarly, Figure 5 shows the absolute displacements of the nodes belonging to the output graph. Again, using the combined selection and displacement approach leads to smaller node movements than when only displacement is allowed. Furthermore, objects can be displaced while not being distorted much. This can be seen especially for the bottom-most building in Figures 4(a) and 5(a).

For the shown dataset, the total objective value decreases by ca. 86% when selection of objects is permitted. Averaged over the urban datasets, a decrease of ca. 73% is obtained, compared to only 52% for the rural datasets. This reflects the structure of our datasets: In the urban datasets, the buildings are to a higher degree surrounded by other objects, which makes it more difficult to displace them without generating new conflicts. For comparison, the result for the rural dataset yielding the smallest decrease (ca. 21%) is shown in Figure 6.

5.2. Comparison to a simulated annealing approach

We now compare our approach to the simulated annealing approach by Ware *et al.* (2003). For that, we have to slightly change our model so that the results are comparable. First, in their approach all roads remain unchanged. Hence, we add the constraint $dx'_v = dy'_v = 0$ for each node v belonging to a road before solving the model. Secondly, in their model buildings can only be displaced as a whole object, i.e. their edges cannot be distorted. We achieve this by constraining $\delta x'_{uv} = \delta y'_{uv} = 0$ for each edge (u, v) belonging to a building. Thirdly, the authors distinguish between conflicts involving two buildings and conflicts involving a building and a road. The weight they assign to the latter group is 10 times the weight they assign to the first group. We consider this by setting the coefficient w_{uv} in Equations (1) to (10) for each bottleneck edge between a building and a road, and to 1 for any other edge. Since their input data are not available anymore, we manually generated a dataset that mimics the situation in Figure 6(d) of their publication. For our comparison, we run our approach on

Table 2. Objective values when using our approach for the selection of buildings and when enforcing the selection from Ware *et al.* (2003). The columns displacement, distortion, and selection refer to the first, second, and third term of Equation (1), respectively. The column total shows their sum. The weights are $w_{\text{pos}} = 0.0001$, $w_{\text{edge}} = 0.8$ and $w_{\text{select}} = 0.1999$.

	Displacement	Distortion	Selection	Total
Our selection	0.13	564.86	241.24	806.24
Selection from Ware <i>et al.</i> (2003)	0.18	585.87	616.96	1203.02



Figure 7. Our method applied to a dataset that we manually generated to mimic the situation shown in a figure by Ware *et al.* (2003): (a) Free selection of buildings within our framework. (b) Fixing the selection to the one made by Ware *et al.* (2003). Coloring as in Figure 3.

this dataset twice: Once without further restrictions and once with the restriction that we have to select exactly the same buildings as they did. We then compare the overall objective values achieved with both variants. Moreover, we look at the values of the three terms in Equation (1). We do this for the weighting which appeared most reasonable in Sect. 5.1 ($w_{\text{pos}} = 0.0001$, $w_{\text{edge}} = 0.8$, $w_{\text{select}} = 0.1999$), without the model extension from Sect. 4.6.2 since the dataset consists of isolated buildings only. The resulting objective values are listed in Table 2. In total, we achieve a reduction by approximately 33% of f_{basic} when we use our approach to select buildings instead of adopting the selection from Ware *et al.* (2003). The largest part of the reduction comes from the selection of buildings. In their approach, a larger number of buildings are unselected (12 vs. 11), and these buildings tend to be larger than the ones which are unselected in our approach (see Figure 7). This is interesting since also in their approach the cost of unselecting a building is proportional to its area. Hence, we can assume that with our global optimization we are more able to find the buildings which are most suited for being unselected. Also the displacement cost is larger for the restricted selection. This can be seen in Figure 7, since some buildings in the center obtain rather large displacements in the restricted selection. The largest displacement is 3.7 m for our selection, compared to 5.1 m for the restricted one. A reason for this may be that, in our approach, the cost for displacing a node grows quadratically with the amount of displacement, whereas Ware *et al.* (2003) apply a linear cost

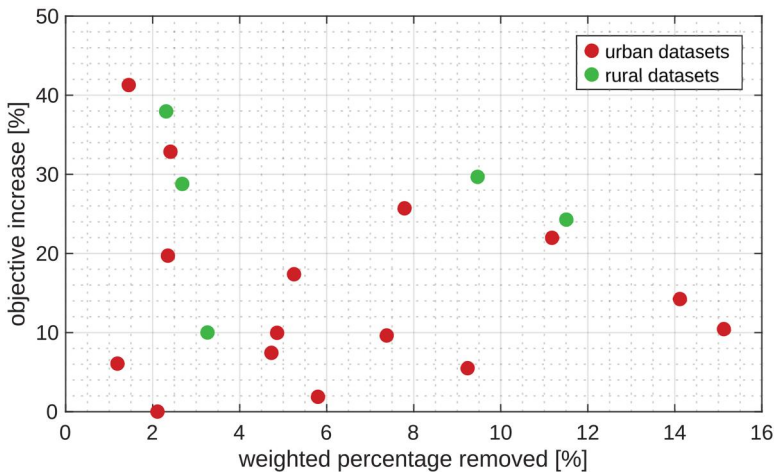


Figure 8. Increase in the objective function that comes along when using the heuristic instead of the exact method, for each of our 20 datasets. The x-axis shows the amount of unselection occurring in the exact method.

function. Hence, in their approach, displacing a few nodes by a large distance is less strictly penalized.

5.3. Performance of the heuristic

A heuristic optimization algorithm should provide results close to an exact algorithm measured in terms of the objective function. Yet, so far it is not clear to which degree this holds for our heuristic. Therefore, we now compare the objective values provided by our exact algorithm and the heuristic for our instances. For each instance, both the exact and the heuristic algorithm are run and the objective function from Equation (14) is evaluated on the results. The threshold θ in the heuristic is chosen experimentally for each instance by exploring values between 0.9965 and 0.9995 with a step size of 0.0005 and selecting the one yielding the smallest objective value. This always resulted in θ lying within the interval [0.998, 0.999]. Figure 8 shows the increase in the objective function coming along with the heuristic. Averaged over the 20 datasets, the objective value provided by the heuristic is by ca. 18% larger than the one of the exact solution. The datasets are sorted by the weighted percentage of removed objects (WP), which we define as the ratio of the summed weights of all unselected objects in the exact solution over the summed weights of all objects. Higher values of WP indicate more unfavorable configurations because in this case the optimal solution unselects objects of high weight. However, there seems to be no correlation between WP and the increase in the objective. This indicates that the approximation quality does not depend much on the spatial complexity of the instance. Also, the performance of the heuristic does not seem to differ much between the urban and the rural datasets.

Figure 9 shows the results for the dataset where the highest increase in the objective (ca. 41 %) was obtained. Using the heuristic, a block of garages in the upper part is completely removed, whereas the exact method removes only the first

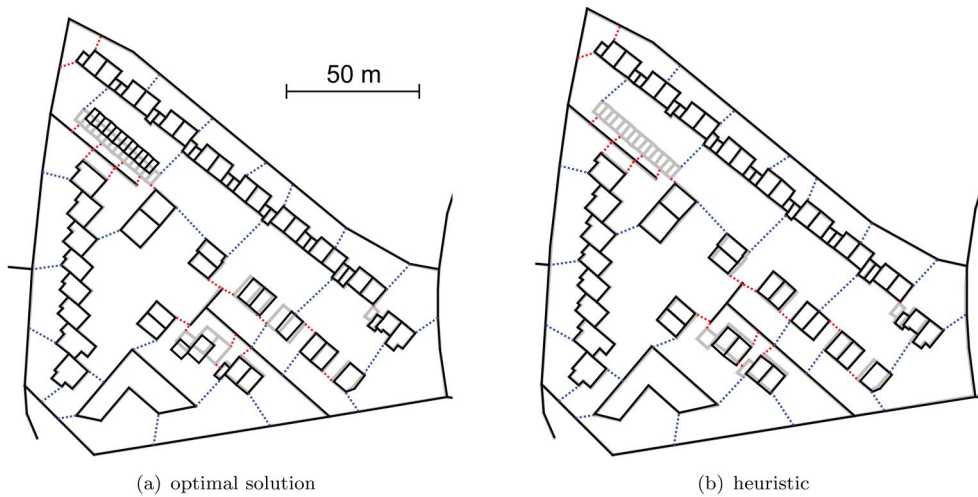


Figure 9. Dataset where the largest gap between the heuristic and the exact method in terms of the objective function occurred (coloring as in Figure 3).

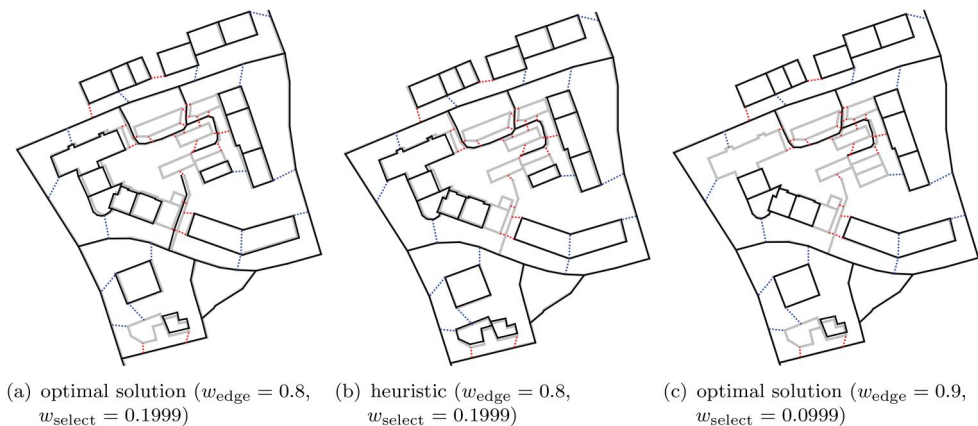


Figure 10. An example where a road is unselected ($w_{\text{pos}} = 0.0001$, $w_{\text{depend}} = 0.5$; coloring as in Figure 3).

and last garage. Apart from this, the two methods yield a similar result. Both the rectangular shape of the buildings and the connectivity of building blocks can be preserved. The results for another dataset are depicted in Figure 10. In this case, the heuristic unselects a road. The same also happens for the exact method when the weight for the selection of objects is decreased ($w_{\text{edge}} = 0.9$, $w_{\text{select}} = 0.0999$). However, this is only a minor road contributing to many conflicts, so in this case the removal is plausible.

5.4. Running time

We conducted the computations on a Windows notebook with 16 GB RAM and an AMD Ryzen 5 5625 U CPU, clocked at 2.3 GHz. For the exact method, the running times range

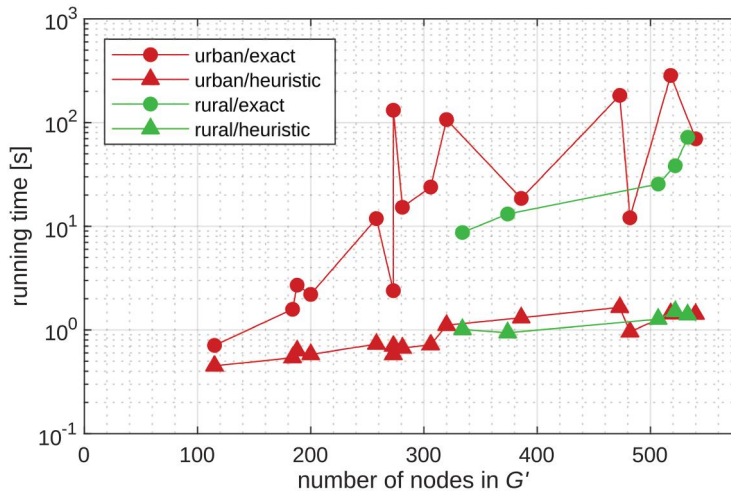


Figure 11. Running times of our methods as functions of the number of nodes in the enriched graph G' .

from 0.7 s to 284.5 s, whereas they range from 0.5 s to 1.7 s for the heuristic (see Figure 11). The ratio of the running times of both methods varies between 1.6 and 199.0 and, in general, increases with the size of the datasets. Beside the 20 small datasets, we tested our method on a larger dataset with 898 buildings and 88 road objects, covering an area of ca. 43 ha and comprising 5024 nodes in G' . There, the heuristic yielded a solution within 65 sec, whereas the exact approach was not completed within 9 h.

To sum up, the heuristic allows for a much faster computation while only slightly increasing the objective value.

6. Conclusion and future work

We have presented a novel optimization approach for solving two processes of map generalization, object selection and displacement, in an integrated way. From the experiments that we have conducted with building and road data, we conclude that the method is well capable to resolve a set of previously detected proximity conflicts. Specifically, our exact method reduced the overall cost by 33%, compared to a solution where the selected objects have been adopted from a solution of an existing method and only the displacement has been optimized using our model. The solutions found with our heuristic were on average 18% worse than the optimal solutions. However, this decrease in quality may be acceptable in view of the relatively short running time of the heuristic, which solved a problem instance of 898 buildings and 88 roads within 65 seconds. We understand our approach as a first step towards integrating discrete and continuous decisions of map generalization in a single mathematical programming framework and see the following opportunities for future work.

As discussed in Sect. 4.1, an important topic for future research is the preservation of building patterns other than sequences of row houses in the selection process, e.g. sets of buildings arranged in a regular grid. Moreover, it is important to integrate further processes of map generalization into our model. It would be relatively easy to

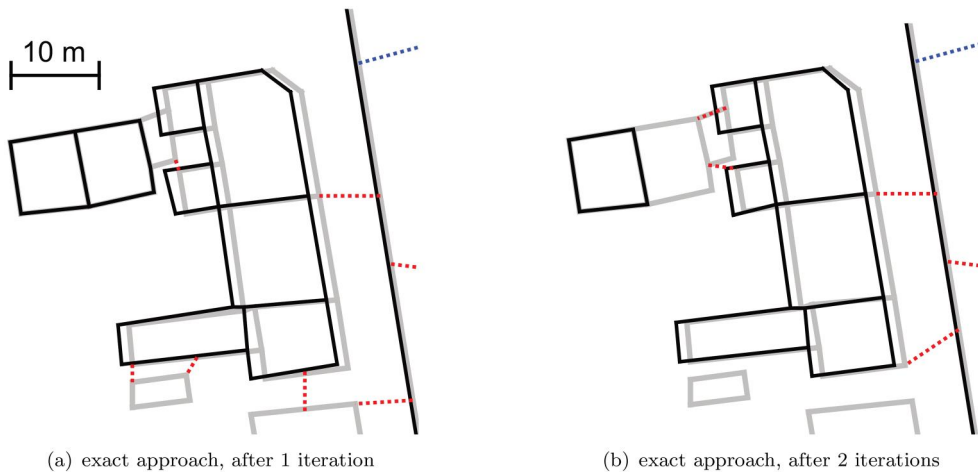


Figure 12. An example where new proximity conflicts arise from the displacement of objects. The bottleneck edges shown in (b) are those computed in the second iteration.

integrate processes that have already been tackled with least-squares optimization, such as smoothing and exaggeration (Harrie and Sarjakoski 2002), since our model for displacement has a similar form. A more interesting challenge would be to also include polygon aggregation, meaning that multiple buildings are replaced by a single output polygon. With this capability the method would become relevant for the production of maps of scales smaller than 1:25,000. We consider it promising to develop an aggregation approach where the output polygons are defined by selecting a set of triangles from a triangulation of the exterior of the input polygons (Jones *et al.* 1995; Rottmann *et al.* 2021). Here, the solution could be modeled with additional binary variables indicating which triangles are selected. Since even without these extensions our model contains many parameters that so far need to be set by an expert, automating the calibration of the model is an important task for future work. We consider it promising to model the calibration task as an optimization problem, aiming to find a set of parameters whose application yields an output map most similar to a reference solution, e.g., a map that was manually generated by an expert.

Future work is also needed to address the limited scalability of our method. For processing very large datasets, such as a digital landscape model of an entire country, it is promising to combine our method with methods for data partitioning (Chaudhry and Mackaness 2008; Thiemann *et al.* 2013; Berli *et al.* 2018).

Finally, future work is needed to deal with situations where large displacements occur. For example, in Figure 12(a), a small building lying between other ones is unselected. Since the amount of displacement of the selected buildings is almost half the width of the unselected building, a new proximity conflict arises that has not been detected in the preprocessing step. A straight-forward approach would be to reiterate on the solution found, i.e., to perform the detection of proximity conflicts and the solution of the optimization problem again; see Figure 12(b). However, an interesting question is whether one could predict the proximity conflicts that will likely arise. When considering all predicted conflicts in the model, it may be sufficient to solve the model only once.

Specifically for the prediction of conflicts, we see a large potential in machine-learning approaches, which are receiving a growing attention in cartography (Harrie *et al.* 2024).

Note

1. <https://www.openstreetmap.org/>

Author contributions

Leon Rosenberger, Yilang Shen, and Jan-Henrik Haunert generated the idea and conceptualized the methodology. Leon Rosenberger and Jan-Henrik Haunert worked out the mathematical details. Leon Rosenberger did all the implementation and experimentation and wrote the first version of the corresponding sections. Yilang Shen contributed to the review and discussion of related work. Jan-Henrik Haunert finalized the article.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported by the China Scholarship Council (CSC) under Grant 202006275019 and by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Projektnummer 459420781.

Notes on contributors

Leon Rosenberger studied Geodesy and Geoinformation at the University of Bonn from 2017 to 2023. His Bachelor's and Master's theses covered topics of geoinformation science, and he worked as a student assistant in the Geoinformation Group. Since 2024, he has worked at the national railway company of Germany (Deutsche Bahn) where he is involved in surveying and planning tasks in connection with the maintenance of the railway network.

Yilang Shen received B.S. and Ph.D. degrees in geographical information systems from Wuhan University, Wuhan, China, in 2013 and 2019, respectively. He did postdoctoral research at the University of Bonn in Germany and is now an associate professor at the school of geospatial engineering and science, Sun Yat-sen University. His research interests include cartography and remote sensing image processing.

Jan-Henrik Haunert received a doctoral degree (Dr.-Ing.) in geodesy and geoinformatics from the University of Hannover, Germany, in 2008. He then held a postdoctoral position at the University of Würzburg and later a professorship for geoinformatics at the University of Osnabrück. In 2016, he became a professor for geoinformation at the University of Bonn, where he and the Geoinformation Group chaired by him are focusing on algorithms for cartography and GIS.

ORCID

Jan-Henrik Haunert  <http://orcid.org/0000-0001-8005-943X>

Data and codes availability statement

The sourcecode and input data used for the experiments is publicly available under <https://doi.org/10.6084/m9.figshare.26243195.v1>. The input data for our experiments have been authored by OpenStreetMap under the Open Data Commons Open Database-Lizenz (ODbL). The only exception is the example in Figure 7, which we generated manually to mimic a situation displayed in a figure by Ware *et al.* (2003).

References

- Agrawala, M., and Stolte, C., 2001. Rendering effective route maps: improving usability through generalization. In: *Proc. 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '01)*. 241–249.
- Ai, T., and Liu, Y., 2004. Analysis and simplification of point cluster based on Delaunay triangulation model. In: Z. Li, Q. Zhou and W. Kainz, eds. *Advances in spatial analysis and decision making*. ISPRS. Lisse, NL: A A Balkema Publishers, 9–19.
- Ai, T., *et al.*, 2015. A vector field model to handle the displacement of multiple conflicts in building generalization. *International Journal of Geographical Information Science*, 29 (8), 1310–1331.
- Ai, T., Liu, Y., and Chen, J., 2006. The hierarchical watershed partitioning and data simplification of river network. In: *Progress in Spatial Data Handling: Proc. 12th International Symposium on Spatial Data Handling*. 617–632.
- Anand, R., Aggarwal, D., and Kumar, V., 2017. A comparative analysis of optimization solvers. *Journal of Statistics and Management Systems*, 20 (4), 623–635.
- Basaraner, M., 2011. A zone-based iterative building displacement method through the collective use of Voronoi tessellation, spatial analysis and multicriteria decision making. *Boletim de Ciências Geodésicas*, 17 (2), 161–187.
- Berli, J., *et al.*, 2018. Experiments to distribute map generalization processes. *Proceedings of the ICA*, 1, 1–6.
- Boyd, S., and Vandenberghe, L., 2004. *Convex optimization*. Cambridge, UK: Cambridge University Press.
- Burghardt, D., and Cecconi, A., 2007. Mesh simplification for building typification. *International Journal of Geographical Information Science*, 21 (3), 283–298.
- Burghardt, D., Schmid, S., and Stoter, J., 2007. Investigations on cartographic constraint formalisation. In: *Proc. 11th ICA Workshop on Generalisation and Multiple Representation*. <https://kartographie.geo.tu-dresden.de/downloads/ica-gen/workshop2007/Burghardt-ICAWorkshop.pdf>
- Chaudhry, O.Z., and Mackaness, W.A., 2008. Partitioning techniques to make manageable the generalisation of national spatial datasets. In: *Proc. 11th ICA Workshop on Generalisation and Multiple Representation*. https://kartographie.geo.tu-dresden.de/downloads/ica-gen/workshop2008/21_Chaudhry_Mackaness.pdf
- Chen, J., *et al.*, 2009. Selective omission of road features based on mesh density for automatic map generalization. *International Journal of Geographical Information Science*, 23 (8), 1013–1032.
- Chimani, M., van Dijk, T.C., and Haunert, J., 2014. How to eat a graph: Computing selection sequences for the continuous generalization of road networks. In: *Proc. 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. 243–252.
- Elias, B., 2003. Extracting landmarks with data mining methods. In: W. Kuhn, M.F. Worboys and S. Timpf, eds. *Proc. International Conference on Spatial Information Theory (COSIT 2003). Foundations of Geographic Information Science*. Lecture Notes in Computer Science, vol. 2825. Berlin, Heidelberg: Springer, 375–389.
- Elias, B., Hampe, M., and Sester, M., 2005. Adaptive visualisation of landmarks using an MRDB. In: L. Meng, T. Reichenbacher and A. Zipf, eds. *Map-based mobile services: Theories, methods and implementations*. Berlin, Heidelberg: Springer, 73–86.

- Galanda, M., 2003. *Automated polygon generalization in a multi agent system*. Thesis (PhD). University of Zurich.
- Gedicke, S., Oehrlein, J., and Haunert, J., 2021. Aggregating land-use polygons considering line features as separating map elements. *Cartography and Geographic Information Science*, 48 (2), 124–139.
- Gong, X., and Wu, F., 2018. A typification method for linear pattern in urban building generalisation. *Geocarto International*, 33 (2), 189–207.
- Guercke, R., et al., 2011. Aggregation of LoD 1 building models as an optimization problem. *ISPRS Journal of Photogrammetry and Remote Sensing*, 66 (2), 209–222.
- Harrie, L., 1999. The constraint method for solving spatial conflicts in cartographic generalization. *Cartography and Geographic Information Science*, 26 (1), 55–69.
- Harrie, L., 2003. Weight-setting and quality assessment in simultaneous graphic generalization. *The Cartographic Journal*, 40 (3), 221–233.
- Harrie, L., and Sarjakoski, T., 2002. Simultaneous graphic generalization of vector data sets. *Geoinformatica*, 6 (3), 233–261. [Mismatch]
- Harrie, L., and Weibel, R., 2007. Modelling the overall process of generalisation. In: W.A. Mackaness, A. Ruas and L.T. Sarjakoski, eds. *Generalisation of geographic information*. Amsterdam, Netherlands: Elsevier, Ch. 4, 67–87.
- Harrie, L., et al., 2024. Machine learning in cartography. *Cartography and Geographic Information Science*, 51 (1), 1–19.
- Haunert, J., and Sering, L., 2011. Drawing road networks with focus regions. *IEEE Transactions on Visualization and Computer Graphics*, 17 (12), 2555–2562.
- Haunert, J., and Wolff, A., 2010. b. Optimal and topologically safe simplification of building footprints. In: *Proc. 18th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. 192–201.
- Haunert, J., and Wolff, A., 2010a. Area aggregation in map generalisation by mixed-integer programming. *International Journal of Geographical Information Science*, 24 (12), 1871–1897.
- Haunert, J., and Wolff, A., 2017. Beyond maximum independent set: An extended integer programming formulation for point labeling. *ISPRS International Journal of Geo-Information*, 6 (11), 342.
- Hirono, D., et al., 2013. Constrained optimization for disoccluding geographic landmarks in 3d urban maps. In: *Proc. IEEE Pacific Visualization Symposium*. 17–24.
- Højholt, P., 2000. Solving space conflicts in map generalization: Using a finite element method. *Cartography and Geographic Information Science*, 27 (1), 65–74.
- Huang, H., et al., 2017. Reducing building conflicts in map generalization with an improved PSO algorithm. *ISPRS International Journal of Geo-Information*, 6 (5), 127.
- Huang, J., Stoter, J.E., and Nan, L., 2023. Symmetrization of 2D polygonal shapes using mixed-integer programming. *Computer-Aided Design*, 163, 103572.
- Jiang, B., and Harrie, L., 2004. Selection of streets from a network using self-organizing maps. *Transactions in GIS*, 8 (3), 335–350.
- Jones, C.B., Bundy, G.L., and Ware, M.J., 1995. Map generalization with a triangulated data structure. *Cartography and Geographic Information Systems*, 22 (4), 317–331.
- Kapaj, A., et al., 2023. How does the design of landmarks on a mobile map influence wayfinding experts' spatial learning during a real-world navigation task? *Cartography and Geographic Information Science*, 50 (2), 197–213.
- Li, C., et al., 2019. An automated method for the selection of complex railway lines that accounts for multiple feature constraints. *Transactions in GIS*, 23 (6), 1296–1316.
- Li, C., Wu, W., and Yin, Y., 2018. Hierarchical elimination selection method of dendritic river network generalization. *PloS One*, 13 (12), e0208101.
- Lu, X., et al., 2019. An algorithm based on the weighted network Voronoi diagram for point cluster simplification. *ISPRS International Journal of Geo-Information*, 8 (3), 105.
- Mackaness, W.A., 1994. An algorithm for conflict identification and feature displacement in automated map generalization. *Cartography and Geographic Information Systems*, 21 (4), 219–232.

- Maudet, A., et al., 2014. Multi-agent multi-level cartographic generalisation in CartAGen. In: *Proc. 12th International Conference on Advances in Practical Applications of Heterogeneous Multi-Agent Systems*. 355–358.
- Mazur, R.E., and Castner, H., 1990. Horton's ordering scheme and the generalisation of river networks. *The Cartographic Journal*, 27 (2), 104–112.
- Nickel, S., et al., 2022. Multicriteria optimization for dynamic Demers cartograms. *IEEE Transactions on Visualization and Computer Graphics*, 28 (6), 2376–2387.
- Oehrlin, J., and Haunert, J., 2017. A cutting-plane method for contiguity-constrained spatial aggregation. *Journal of Spatial Information Science*, 15 (15), 89–120.
- Peng, D., Wolff, A., and Haunert, J., 2021. Finding optimal sequences for area aggregation: A* vs. integer linear programming. *ACM Transactions on Spatial Algorithms and Systems*, 7 (1), 1–4:40.
- Peters, S., 2013. Quadtree- and octree-based approach for point data selection in 2D or 3D. *Annals of GIS*, 19 (1), 37–44.
- Regnauld, N., 2001. Contextual building typification in automated map generalization. *Algorithmica*, 30 (2), 312–333.
- Rottmann, P., et al., 2021. Bicriteria aggregation of polygons via graph cuts. In: *Proc. 11th International Conference on Geographic Information Science (GIScience 2021) - Part II*. Leibniz International Proceedings in Informatics (LIPIcs), vol. 208. Wadern, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 6:1–6:16.
- Ruas, A., 1998. A method for building displacement in automated map generalisation. *International Journal of Geographical Information Science*, 12 (8), 789–803.
- Saalfeld, A., 1991. Delaunay edge refinements. In: *Proc. 3rd Canadian Conference on Computational Geometry*. 33–36.
- Sanders, P., 2011. *Algorithm engineering*. Boston, MA: Springer US, 33–38.
- Schwartges, N., et al., 2013. Optimizing active ranges for point selection in dynamic maps. In: *Proc. 16th ICA Generalisation Workshop*.
- Sester, M., 2000. Generalization based on least squares adjustment. In: *Proc. XIXth ISPRS Congress, XXXIII (Part B4)*, 931–938.
- Sester, M., 2005. Optimization approaches for generalization and data abstraction. *International Journal of Geographical Information Science*, 19 (8-9), 871–897.
- Shen, Y., Ai, T., and Zhao, R., 2022a. Raster-based method for building selection in the multi-scale representation of two-dimensional maps. *Geocarto International*, 37 (22), 6494–6518.
- Shen, Y., et al., 2022b. A raster-based typification method for multiscale visualization of building features considering distribution patterns. *International Journal of Digital Earth*, 15 (1), 249–275.
- Shirabe, T., 2005. A model of contiguity for spatial unit allocation. *Geographical Analysis*, 37 (1), 2–16.
- Sun, Y., et al., 2016. An immune genetic algorithm to buildings displacement in cartographic generalization. *Transactions in GIS*, 20 (4), 585–612.
- Thiemann, F., et al., 2013. Investigations into partitioning of generalization processes in a distributed processing framework. In: *Proc. 26th International Cartographic Conference*.
- Thomson, R.C., and Brooks, R., 2000. Efficient generalisation and abstraction of network data using perceptual grouping. In: *Proc. 5th International Conference on GeoComputation*. 23–25.
- Töpfer, F., and Pillewizer, W., 1966. The principles of selection. *The Cartographic Journal*, 3 (1), 10–16.
- Touya, G., and Lokhat, I., 2022. Rebankment: displacing embankment lines from roads and rivers with a least squares adjustment. *International Journal of Cartography*, 8 (1), 37–53.
- van Dijk, T., et al., 2013. Accentuating focus maps via partial schematization. In: *Proc. 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. 428–431.
- van Dijk, T.C., and Haunert, J., 2014. Interactive focus maps using least-squares optimization. *International Journal of Geographical Information Science*, 28 (10), 2052–2075.
- Wang, L., et al., 2017. Contextual building selection based on a genetic algorithm in map generalization. *ISPRS International Journal of Geo-Information*, 6 (9), 271.

- Wang, X., and Burghardt, D., 2019. A mesh-based typification method for building groups with grid patterns. *ISPRS International Journal of Geo-Information*, 8 (4), 168.
- Ware, J.M., and Jones, C.B., 1998. Conflict reduction in map generalization using iterative improvement. *GeoInformatica*, 2 (4), 383–407.
- Ware, J.M., Jones, C.B., and Thomas, N., 2003. Automated map generalization with multiple operators: a simulated annealing approach. *International Journal of Geographical Information Science*, 17 (8), 743–769.
- Wilson, I.D., Ware, J.M., and Ware, J.A., 2003. A genetic algorithm approach to cartographic map generalisation. *Computers in Industry*, 52 (3), 291–304.
- Yan, H., and Weibel, R., 2008. An algorithm for point cluster generalization based on the Voronoi diagram. *Computers & Geosciences*, 34 (8), 939–954.
- Zhang, J., Yuan, C., and Tong, X., 2006. A least-squares-adjustment-based method for automated map generalization of settlement areas in GIS. In: *Proc. Geoinformatics 2006: Geospatial Information Science*.
- Zhou, Z., Fu, C., and Weibel, R., 2023. Move and remove: multi-task learning for building simplification in vector maps with a graph convolutional neural network. *ISPRS Journal of Photogrammetry and Remote Sensing*, 202, 205–218.
- Zoraster, S., 1990. The solution of large 0-1 integer programming problems encountered in automated cartography. *Operations Research*, 38 (5), 752–759.