Towards a true vario-scale structure supporting smooth-zoom

Peter van Oosterom   Martijn Meijers

Abstract

This paper presents the first true vario-scale structure for geographic information: a delta in scale leads to a delta in the map (and smaller scale deltas lead to smaller map deltas until and including the infinitesimal small delta) for all scales. The structure is called *smooth tGAP* and its integrated 2D space and scale representation is stored as a single 3D data structure: space-scale cube (SSC). The polygonal area objects are mapped to polyhedral representations in the smooth tGAP structure. The polyhedral primitive is integrating all scale representations of a single 2D area object. Together all polyhedral primitives form a partition of the space-scale cube: no gaps and no overlaps (in space or scale). Obtaining a single scale map is computing an horizontal slice through the structure. The structure can be used to implement smooth zoom in an animation or morphing style. The structure can also be used for mixed-scale representation: more detail near to user/viewer, less detail further away by taking non-horizontal slices. For all derived representations, slices and smooth-zoom animations, the 2D maps are always perfect planar partitions (even mixed-scales objects fit together and form a planar partition). Perhaps mixed-scale is not very useful for 2D maps, but for 3D computer graphics it is one of the key techniques. Our approach does also work for 3D space and scale integrated in one 4D hypercube.

1 Introduction

Technological advancements have lead to maps being used virtually everywhere (e. g. mobile smartphones). Map use is more interactive than ever before: users can zoom in, out and navigate on the (interactive) maps.

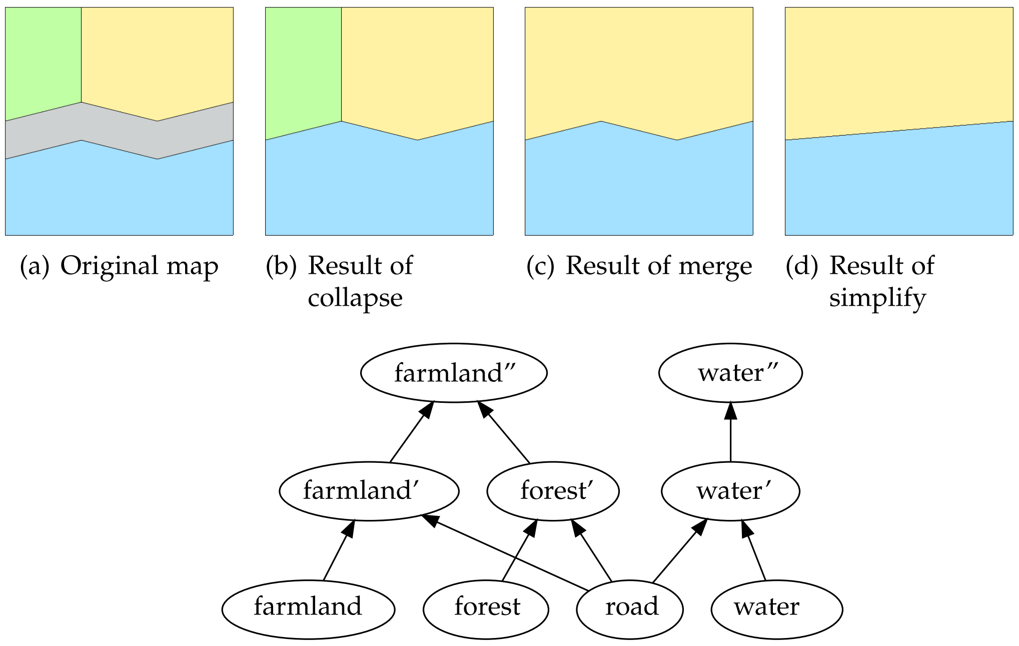
Therefore recent map generalization research shows a move towards continuous generalization. Although some useful efforts (van Kreveld, 2001; Sester and Brenner, 2005; Nöllenburg et al., 2008), there is no optimal solution yet.

This paper introduces the first true vario-scale structure for geographic information: a small step in the scale dimension leads to a small change in representation of geographic features that are represented on the map. From the structure continuous generalizations of real world features can be derived and can be used for presenting a smooth zoom action to the user. Furthermore, mixed-scale visualizations can be derived (more and less generalized features shown together in one 2D map) that are consistent with each other. Making such a transition area is mostly one of the difficulties for 3D computer graphic solutions (e. g. using stitch strips based on triangles, like in Noguera et al., 2010).

The remainder of this paper is structured as follows: Section 2 contains a discussion how the classic tGAP structure can be adapted to store more continuous generalization of line boundaries. Section 3 extends this reasoning to make also other generalization operations, such as merge and split/collapse, smooth, leading to a new structure: smooth tGAP. Section 4 explores drawbacks of the proposed approach, the paper is concluded, together with a summation of a long list of open research questions, in Section 5.

2 Vario-scale and tGAP structure

The tGAP structure has been presented as a vario-scale structure (van Oosterom, 2005). In summary, the tGAP structure traditionally starts with a planar partition at the most detailed level (largest scale). Next the least important object (based on geometry and classification) is selected, and then merged with the most compatible neighbour (again based on geometry and classification). This is repeated until only a single object is remaining, the merging of objects is recorded in tGAP-tree structure and the last object is the top of the tree. The (parallel) simplification of the boundaries is also executed during this process and can be recorded in a specific structure per boundary: the BLG-tree (binary line generalization). As assigning the least important object in certain cases to just a single neighbour may result in a suboptimal map representation, the weighted split (and assigning parts to multiple neighbours) was introduced. This changed the tGAP-tree into a tGAP Directed Acyclic Graph (DAG) and together with the BLG-tree, this is called the tGAP structure. The tGAP



(a) Original map (b) Result of collapse (c) Result of merge (d) Result of simplify

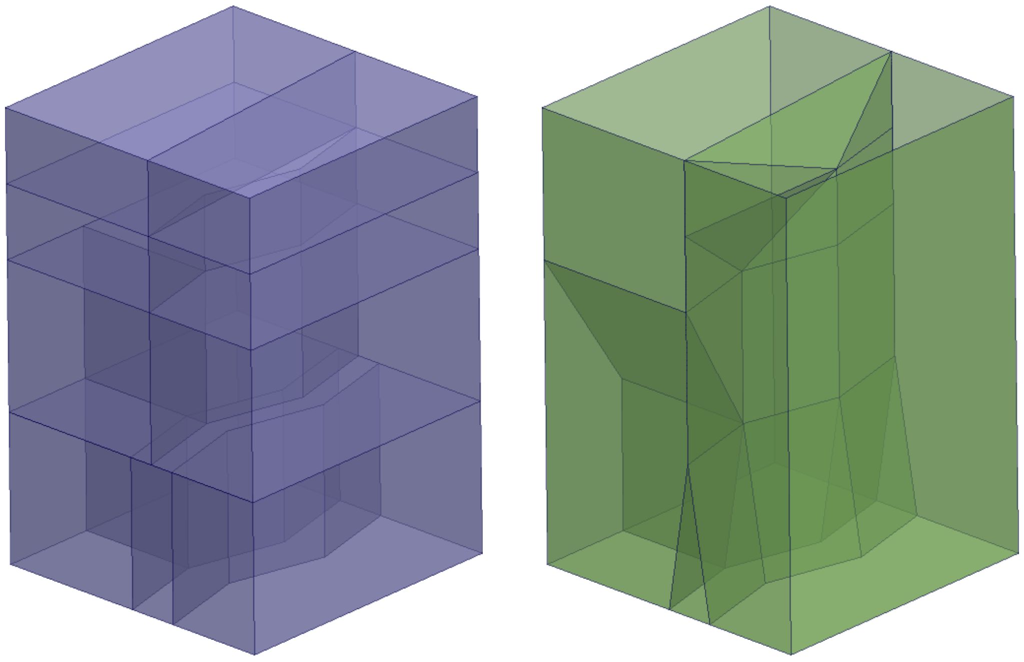
(e) Corresponding tGAP structure

Figure 1: The 4 map fragments and corresponding tGAP structure

structure can be seen as result of the generalization process and can be used to efficiently select a representation at any required level of detail (scale or importance). Figure 1 shows 4 maps fragments and the tGAP structure in which the following generalization operations have been applied:

1. Collapse road object from area to line (and split and assign free space to neighbours);
2. Remove forest area and merge free space into neighbour farmland;
3. Simplify boundary between farmland and water area.

The tGAP structure is a DAG and not a tree structure, as the split causes the road object to have several parents; see Figure 1(e). In our current implementation the simplify operation on the relevant boundaries is combined with the remove or collapse/split operators and is not a separate step. However, for the purpose of this paper it is more clear to illustrate these operators separately. For the tGAP structure, the scale has been depicted as third dimension — the integrated space-scale cube (SSC) representation (Vermeij et al., 2003; Meijers and van Oosterom, 2011). Figure 2(a) shows this 3D representation for the example scene of Figure 1.

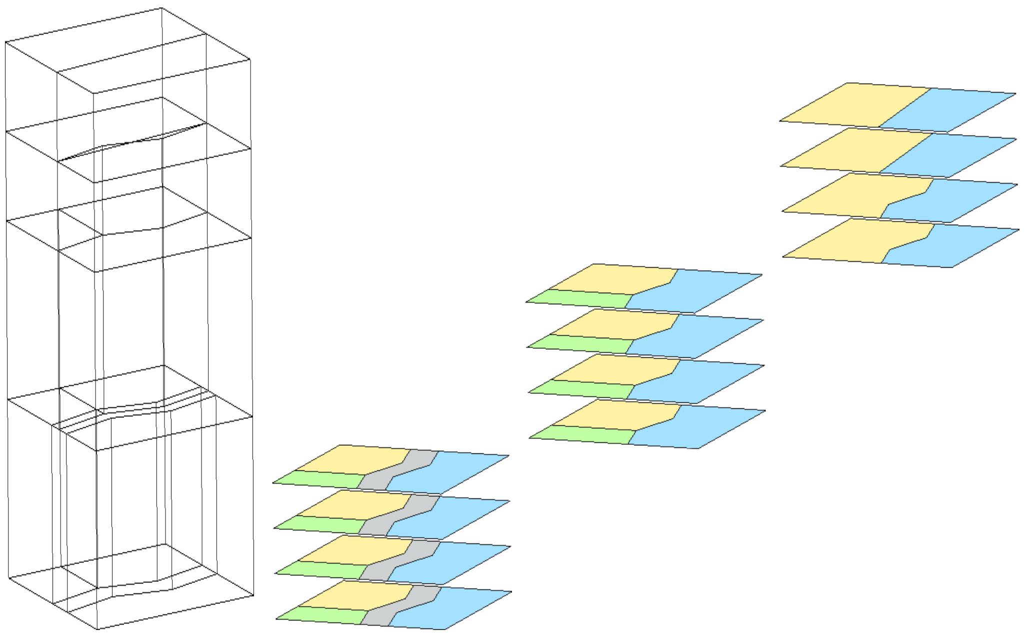


(a) SSC for the *classic* tGAP structure (b) SSC for the *smooth* tGAP structure

Figure 2: The space-scale cube (SSC) representation in 3D

Though many small steps (from most detailed to most coarse representation — in the classic tGAP, $n-1$ steps exist, if the base map contains n objects), this could still be considered as many discrete generalization actions approaching vario-scale, but not *true* vario-scale. Split and merge operations do cause a sudden local 'shock': a small scale change results in a not so small geometry change; e. g. leading to complete objects disappearing; see Figure 3. In the space-scale cube this is represented by a horizontal face; a sudden end or start of corresponding object. Furthermore, polygon boundaries define faces that are all vertical in the cube, i. e. the geometry does not change at all within the corresponding scale range (resulting in the collection of fitting prism shapes, a full partition of the space-scale cube).

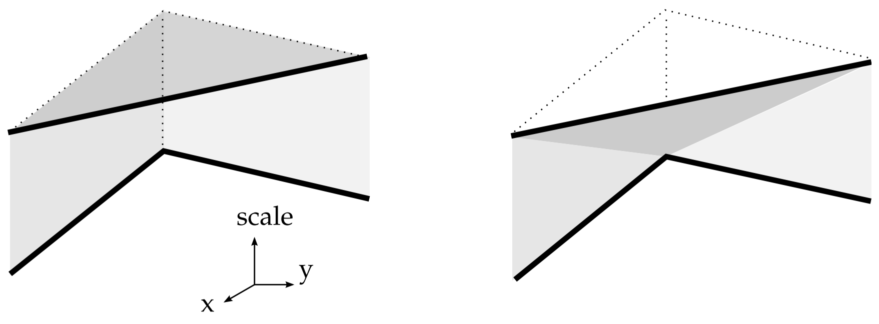
In order to obtain more gradual changes when zooming, i.e. in a morphing style (c. f. Sester and Brenner, 2005; Nöllenburg et al., 2008), we first realised that the line simplification operation could also output non-vertical faces for the space-scale cube and that this has a more true vario-scale character; e. g. when replacing two neighbouring line segments by



(a) Wireframe of (classic) space-scale cube (b) Slices for Step 1 (c) Slices for Step 2 (d) Slices for Step 3

Figure 3: The map slices of the classic tGAP structure: (b) step 1 (collapse), (c) step 2 (merge) and (d) step 3 (simplify). Note that nothing changes until a true tGAP event has happened.

a single new line segment (omitting the shared node), this can be represented by three triangular faces in the space-scale cube; see Figure 4. Note that both the sudden-change line simplification and the gradual-change line simplification have both 3 faces in the ssc: sudden-change has 2 rectangles and 1 triangle and gradual-change has 3 triangles. When slicing a map (to 'slice' means taking a cross-section of the cube) at a certain scale, a delta in scale leads to a derived delta in the map. That is, a small change in the geometry of the depicted map objects and no sudden change any more, as was the case with the horizontal faces parallel with the bottom of the cube, which were the results of the merge or split operations. Note that the more general line simplification (removing more than one node of a polyline) can be considered to consist of several smaller sub-steps: one step for the removal of each of the nodes.



(a) Sudden-change line simplification: 2 rectangles and 1 triangle

(b) Gradual-change line simplification: 3 triangles

Figure 4: Line simplification in the SSC: (a) sudden removal of node, (b) gradual change. The dashed lines in (b) only illustrate the difference with the sudden-change variant.

3 Supporting smooth zoom

The split and merge operations can, similar to the gradual line simplification operation as sketched above, be redefined as gradual actions supporting smooth zoom. For example in case of the merge of two objects: one object gradually grows and the other shrinks — in a space-scale cube this corresponds to non-vertical faces (and there is no more need for a horizontal face, i. e. a suddenly disappearing feature); see Figure 2(b). All horizontal faces in the cube are now gone, except the bottom and top faces of the cube. Note that adjacent faces in the same plane belonging to the same object are merged into one larger face, e. g. the big front-right face in Figure 2(b) corresponds to four faces in Figure 2(a). The same is true for the involved edges, several smaller edges on straight lines are merged, and the shared nodes are removed. This can be done because they carry no extra information. Perhaps the most important and elegant consequence is that the merging of the different polyhedral volumes belonging to the same real world object is that also the number of volumes is reduced: there is a one-to-one correspondence between a single object and its smooth tGAP polyhedral representation, valid for all relevant map scales. The benefit of a smaller number of primitives, the nodes, edges, faces and volumes, is that there are also less topology references needed to represent the whole structure. In previous investigations it was reported that the storage requirements for topology structure may be as high, or even higher, than the storage requirements for plain geometry (see previous tests, described in Louwsma et al., 2003; Baars et al., 2004; Penninga, 2004). This is even more true for topology based vario-scale data structures (c. f. Meijers et al., 2009). Lighter structures are more suitable for (progressive) data transfer and high(er) performance.

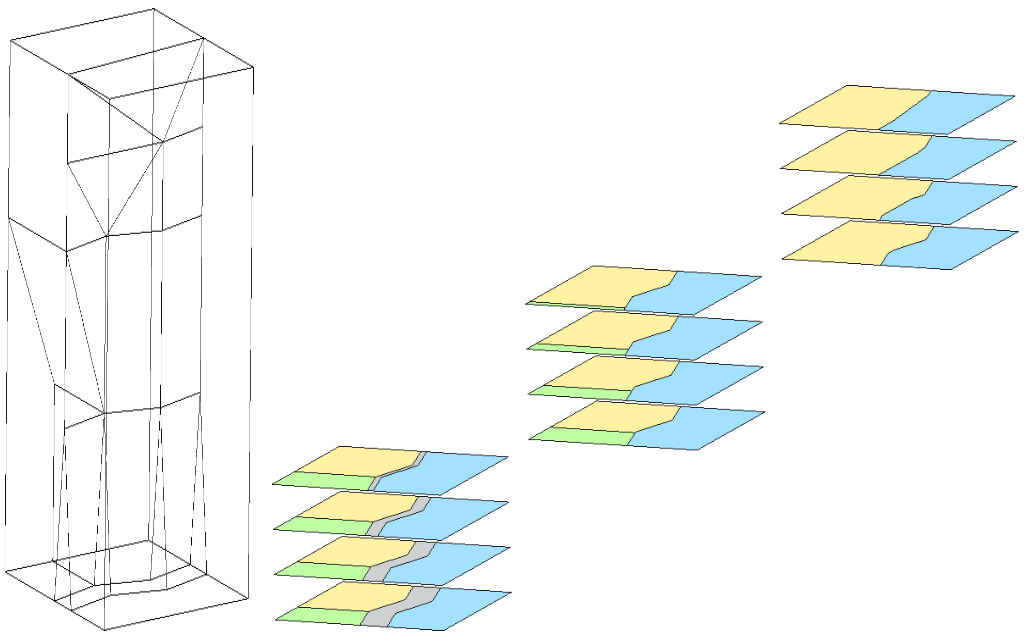
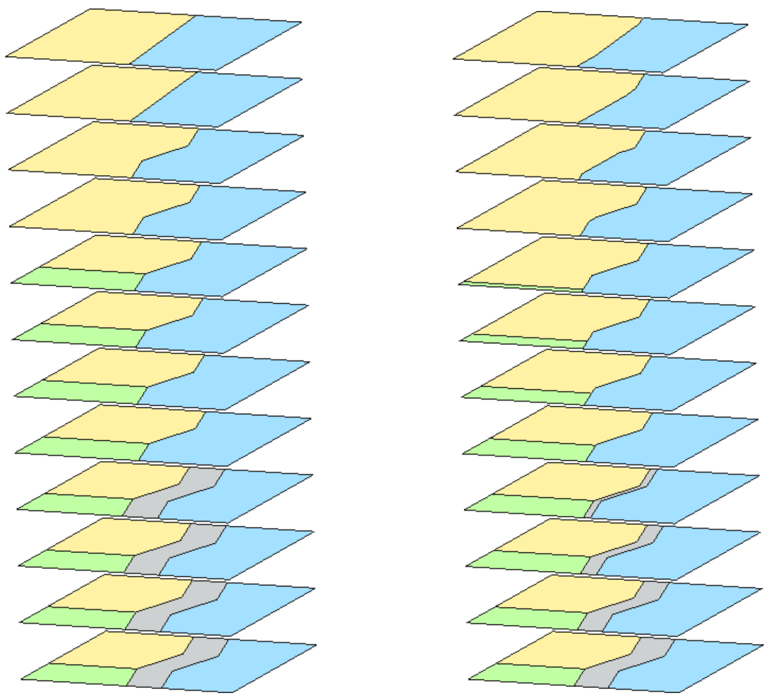

(a) Wireframe of (smooth) space-scale cube (b) Slices for Step 1 (c) Slices for Step 2 (d) Slices for Step 3

Figure 5: The map slices of the smooth tGAP structure: (b) step 1 (collapse), (c) step 2 (merge) and (d) step 3 (simplify). Note the continuous changes, also in between the 'true' tGAP events.

Figure 5 illustrates the resulting true vario-scale structure: small deltas in scale will give small deltas for map areas. Figure 6 shows that if all slices of the classic tGAP and the smooth tGAP space-scale cubes are compared, the differences and the benefits of the later become clear.

So far, only horizontal slices parallel to the bottom and top of the cube were discussed and used for creating 2D maps. It is not strictly necessary to do parallel slices, nothing prevents taking *non-horizontal* slices. Figure 7 illustrates a mixed-scale map derived as a non-horizontal slice from the SSC. What does such a non-horizontal slice mean? More detail at side where slice is close to bottom of the cube, less detail at the side where slice is closer to top. Compare to 3D visualizations, where close to the eye of the observer lots of detail is needed, while further away not so much


(a) Slices for the classic tGAP structure

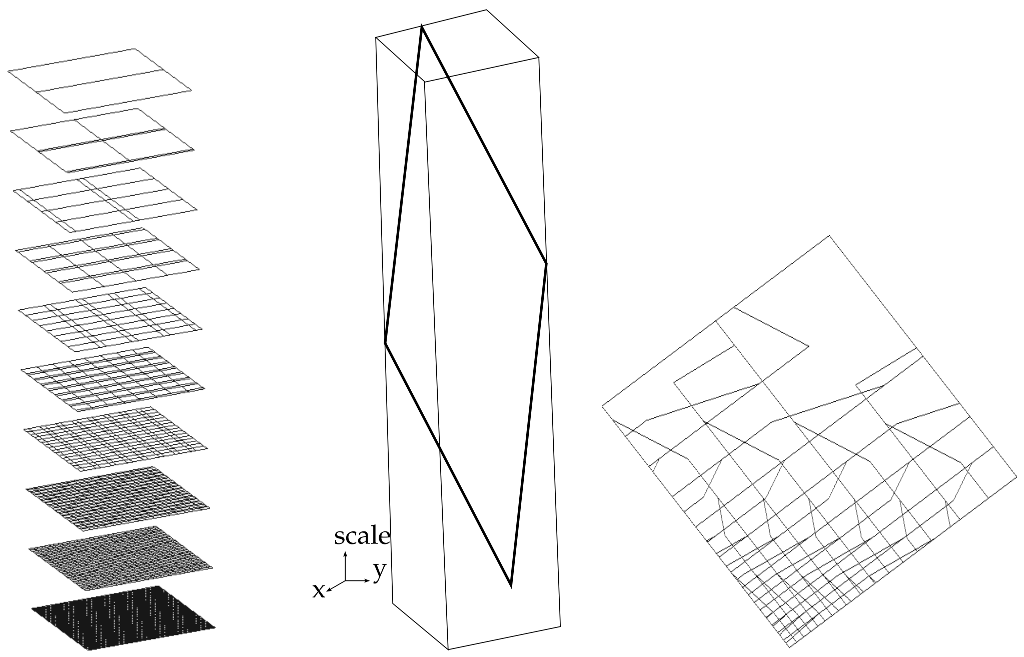
(b) Slices for the smooth tGAP structure

Figure 6: The maps – slices of (a) the classic and (b) the smooth tGAP structure – compared

detail. Such a slice leads to a *mixed-scale* map, as the map contains more generalized features far away (intended for display on small scale) and less generalized features close to observer (large scale).

4 Critical reflection: possible drawbacks

This section explores possible drawbacks of the presented smooth tGAP. Four potential issues are presented in the subsections below: 1. will slivers occur when slicing for a 2D map (§ 4.1), 2. can use of a sequence of non-horizontal delta-slices lead to less gradual changes than expected (§ 4.2), 3. can multiple generalization operations be performed in parallel (§ 4.3) and 4. can square split and merge operations (horizontal faces) always be transformed into their smooth counterparts with non-horizontal faces (§ 4.4)?



(a) A set of smooth slices derived from the SSC

(b) How the non-horizontal slice of (c) is taken.

(c) Corresponding *mixed-scale* map (non-horizontal slice): top of map shows more generalized features than bottom

Figure 7: Checkerboard data as input: each rectangular feature is smoothly merged to a neighbour. Note that all merge operations have been executed in parallel, see § 4.3. Subfigures show: (a) a stack of horizontal slices, (b) taking a non-horizontal slice leads to a 'mixed-scale' map and (c) one mixed scale slice (non-horizontal plane).

4.1 Slivers

The first possible drawback of the smooth tGAP structure might be that if at certain scale a slice is taken, then one could get a sliver: just before a object to be merged is disappearing. If this tGAP structure is used for static 2D maps (and not smooth zoom), then such a sliver should be removed; either by finishing the operation or going back to the start state of the operation. This corresponds to moving the slice slightly up or down in the cube. So, this is no real problem.

4.2 Bad luck can happen...

Imagine that we have a smooth tGAP structure (so non-horizontal faces), then horizontal slices and their movement, up or down the scale-dimension, will give delta map changes. Now for the same structure imagine that we want to have a mixed-scale map using a non-horizontal slice plane and we also want smooth mixed-scale zoom (whatever this may be, doubtful if useful). If the slice plane has the same angle as one of the object faces, then a delta slice plane movement could result in a sudden big map change: a complete object disappearing at once and other objects reappearing. As this is an exotic use case and unlikely that exact same angles will ever occur (probability near 0% if there is no systematic preference for angles of object faces and/or slice plane), this drawback is not really considered a problem.

4.3 Parallel execution, instead of 1 by 1

Despite the fact that the proposed solution results in a true vario-scale structure, it has still an (old) tGAP drawback and that is the 1 by 1 sequencing of all generalization operations. This has the positive effect that it can be proven that all operations are validly represented in the SSC: both a start and end scale of an operation, but also in between. But all gradual changes are local when looking at the big picture: first one operation is (gradually) finished and then the next local operation is started, and so on. This might give a suboptimal smooth-zoom effect — more experiments with end users are needed to verify whether this is indeed suboptimal.

A solution for the 1 by 1 sequencing is not implementing the steps in the structure in a sequential manner, but to group them (to group size N_g) and then let all members in the group transform in parallel. Danger is now that neighbouring actions (each creating a valid part of the structure when executed alone), may together result in an invalid representation, that is, intersecting planes. This can be efficiently detected: prepare resulting faces of this gradual group step and put a 3D R-tree on the new faces. For every new face check for conflicts, that is, intersection with faces already present in the group. Through the use of the R-tree this takes $O(\log n)$ time with n the number of faces in the R-tree group). If case of any conflict, then undo the action that belongs to the last (smaller scale) action of the sequence. In total this takes this takes $O(n \log n)$ time, the time needed for the creation of the R-tree for the faces in the group. If no intersection is found, then the total result is correct, in next group of

N_g actions, the undone actions gets a new change and then have highest priority, so will not be undone this time.

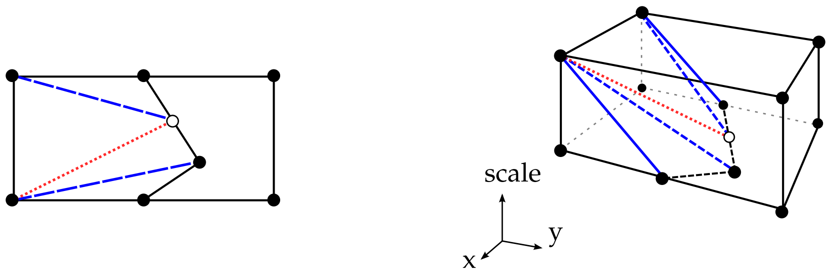
An alternative approach, not based on spatial searching (3D R-tree) is to exploit the topology structure when creating the parallel groups. The normal tGAP creation approach is followed: select the least important object, process this object, then select the next least important object, process this object and so on until enough objects in the group are found. However, when an object is selected, then it will be temporarily marked and also the neighbours will be marked. If a to be removed object or one of its neighbours is already in the set of marked objects, then skip this object and continue with next least important object (and in next grouping the skipped object will be first in line). By finding non-neighbouring objects, the local smooth zoom actions (faces) will not interfere.

Below a list of related research questions:

- With one of the approaches, can deadlocks occur in undoing intersections of the N_g actions?
- What is a good group size N_g? Too small approaches the normal tGAP, while too big N_g might result in a number of discrete stages (end of many non-vertical walls) at the same time. Also a too big N_g increases the change on conflicts between neighbour actions. Having a larger group size N_g might also give a strange artificial effect during the smooth zoom: if many disappearing features are visible in the image, then some kind of artificial climax moment is introduced, when they all disappear together. Tests should be conducted whether this is noticeable (because only very few actions will be visible at same time in one window; e.g. 1 or 2; and others are outside display window).
- Should the size of N_g vary due to: 1. stage in process (more to the top, smaller N_g; e. g. always be a percentage of the total number of objects at a given scale stage) or 2. relative to accumulated area change by actions in group?

4.4 Smooth zoom with real data

The example data set, as used in Sections 2 and 3, only shows very simple shapes. It is easy to imagine that when there are two neighbouring equal rectangles, how one rectangle gradually has to take the space of the other rectangle and that the resulting non-horizontal faces in the smooth tGAP structure will be flat. The question that arises: Is this always possible for any pair of strangely shaped neighbours or configurations with island polygons included? Answer: Yes. Proof: it is possible for strictly convex parts[1] to be 'removed', using the following algorithm (see Figure 8):



(a) Top-view (b) Space-scale view

Figure 8: The simple neighbour merge: one rectangular feature is smoothly merged into rectangular neighbour feature. Note that the plane that forms the boundary between the two features is composed of 2 triangular and 1 quadrilateral faces (these faces can be dissolved by post-processing into 1 face, as they are planar).

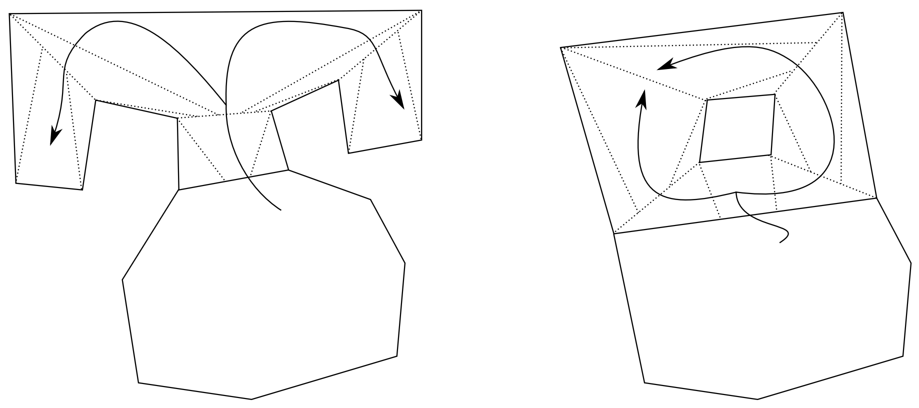
- Count the number of interior nodes on the boundary to be removed and on the boundary to be moved to (minimum number is 1, otherwise neighbour to be removed would have no area).
- If unequal, add the missing number of (fake) nodes fairly distributed to the boundary with the too low number. The number of intermediate nodes is called I and is equal in both boundaries.
- Now both boundaries have an equal number of nodes and add edges between pair of corresponding intermediate nodes (so at least one edge is added).
- This results in two faces with three nodes and $I - 1$ faces with four nodes (and also four) edges. If a face is not flat add an additional diagonal edge and the resulting two triangles are per definition flat.

Note that this 'simple' algorithm may add some unneeded (temporary) nodes. Imagine two equal shaped neighbour rectangles, then a single diagonal face is sufficient. However, our algorithm would add two

[1]A simple polygon is strictly convex if every internal angle is strictly less than 180 degrees (so not equal to 180 degrees).

intermediate nodes (on the shared boundary) and create two triangles and one 4-node face. In a planarity check it may be detected that these faces are co-planar and can be merged (and same for split edges and added node may be removed). So, the final result is equal after this post-processing.

Because of the convex shape, there will never be intersecting edges or faces. If the to be merged shape is concave, then decompose it in convex parts and treat the convex parts one by one. The order in which this should be done is to start with a direct neighbour part of the growing area (and repeat until all parts are processed); see Figure 9(a). Note that this algorithm also works when the to be merged neighbour has an island: creating the strictly convex parts and processing these with the algorithm above will give correct results in the SSC; see Figure 9(b).



(a) The processing of a m-shape neighbour, with growing area attached to middle leg of 'M'

(b) The example of neighbour with island: decompose in strictly convex parts.

Figure 9: The processing of complex shapes into vario-scale representations. Note that quadrilaterals will not be planar and will have to be decomposed into triangular faces by adding an extra diagonal.

5 Conclusion and Future work

In this section first the main results of our research are presented and then the paper is concluded with a long list of future work, aiming to resolve the open questions.

5.1 Main results

This paper has introduced the first true vario-scale structure for geographic information: a delta in scale leads to a delta in the map (and smaller scale deltas lead to smaller map deltas until and including the infinitesimal small delta) for all scales. The smoothness is accomplished by removing all horizontal faces of the classic tGAP structure. Recipes were given how to obtain data for the smooth tGAP structure: 1. performing generalization operations in such a way that the output given gradually changes the boundaries between the features being generalized and 2. grouping generalization operations for parallel execution while still guarding topological consistency. The smooth tGAP structure delivers true vario-scale data and can be used for smooth zoom. It is one integrated scale-space partition, and when using non-horizontal slices the resulting 2D maps will be a valid, mixed-scale planar partition: this is useful for use in 3D computer graphics.

5.2 Open Research questions

Although the smooth tGAP structure is a breaktrough vario-scale data structure supporting smooth zoom, there is still a myriad of open research questions:

- Engineering: how to encode the space-scale (hyper) cube in an efficient manner? Also create metrics and collect statistics: how many nodes, edges, faces, and volumes in the space-scale cube (and which primitives and references explicitly stored, c. f. van Oosterom et al., 2002; Meijers et al., 2009).

- Formalize the structure and proof that all claims can indeed be backed by sound mathematical proofs. To be based on Meijers and van Oosterom (2011) and Thompson and van Oosterom (2011).

- Investigate mixed-scale slices that are non-planar; e. g. support for fish-eye type of visualizations (see Figure 10). Should be investigated with respect to the planar partition characteristic of the resulting maps. Probably OK, but it might be true that a single area object, in original data set, might result in multiple parts in the slice (but no overlaps or gaps will occur in the slice). What are useful slicing surface shapes? Folding back surfaces seam to be non-sense as this will give two representations of the same object on same location in one map/visualization.

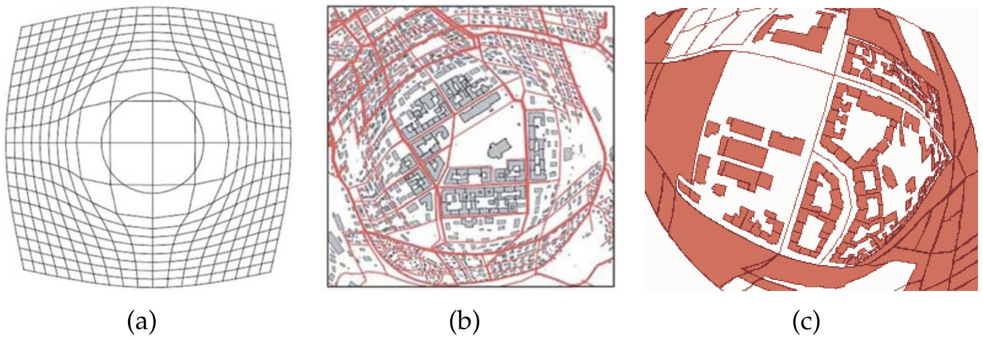


Figure 10: A 'mixed-scale' map. Harrie et al. term this type of map a 'vario-scale' map, while we term this a 'mixed-scale' map. Furthermore it is clear that there is a need for extra generalization closer to the borders of the map, which is not applied in (b), but is applied in (c). With our solution, this generalization would be automatically applied by taking the corresponding slice (bell-shaped, curved suface) from the ssc. Illustrations (a) and (b) taken from Harrie et al. (2002) and (c) from Hampe et al. (2004).

- Implementation of the smooth tGAP structure takes two main steps: 1. build classic tGAP and 2. transform from classic to smooth tGAP (space-scale cube). The smooth tGAP has the same building challenge as the classic tGAP with respect to applying the right sequence of generalization operators (remove or merge, collapse or split, simplify) to obtain cartographic quality. This has to be well tuned, otherwise the maps will be of (too) low cartographic quality despite the fact that they are perfect in topological sense and 100% consistent between scales. One option for this might be the constrained tGAP (Haunert et al., 2009). It is also clear that this requires 'understanding' (semantics) of the different types of object classes involved (and the map needs of the end-users).

- Testing with larger real world data sets and appropriate graphical user interfaces supporting smooth zoom visualization, mixed-scale visualization and observe end-user behaviour (this is a typical Human Computer Interaction study). Probably different devices/platforms (desktop, mobile) have to be tested and users have to be given a range of relevant tasks. Large datasets result in large cubes, a slice near the bottom will contain a lot of data (takes time) and is not what a user wants. So slicing should be combined with other (spatial) selection criteria; e. g. the bounding box (bbox). The bbox is most likely smaller at the bottom and larger near the top for 'sane' applications. For non-horizontal slices the lower edges of the bbox should be shorter than the higher edges of the bbox. This can be compared to the use of frustums in 3D computer graphics for perspective views.

- Further investigating the effects of the Collapse operator in the smooth tGAP structure. After the collapse of an area object to a line (or point) object, the same object lives on. In the SSC this object is then represented by a polyhedral volume to which a vertical surface is connected at the top (in case of collapse to point then in the SSC the polyhedral object is extended with a vertical line). All attributes are attached to the same object, which is represented in the SSC by connected multiple parts of respectively dimension 3 and 2 in case of collapse to line and 1 in case of collapse to point.

- Test overlay processing with two (or more) independent space-scale cubes – this 3D overlay resembles data integration: it is possible to geometrically overlay the two space-scale cubes and carry over the attribute information to the newly segmented space-scale partition. Note that before the actual overlay, the scale-dimension has to be first well aligned: only intersect the corresponding representations. However, for data integration this will not be enough, e. g. one of the difficulties will be to harmonise semantically the attribute values. Using space-scale cubes might give more clues for a data integration process than integrating just two separate 2D map sheets (e. g. which do not have the same reference scale) and can be helpful for performing both horizontal as well as vertical conflation at the same time. An example application could be creating a smooth tGAP based on a soil map 1:50,000 and a land cover map 1:100,000. Intersect the two SSC and use the result to answer the request to find the areas that are forest on sandy soils at scale 1:250,000.

- If instead of a 2D base map we start with a 3D base map (model) and then create in a similar manner a 4D space-scale hypercube, then this might be used for good perspective view visualizations by taking non-horizontal scale slices: near a lot of detail (low in scale) far not so much detail (high in scale). The intersection of this 4D hypercube with the hyperplane gives a perfect 3D topology: all representations do fit without gaps or overlaps. This solves a big problem as often the case in the transition from one Level of Detail (LoD) to the next LoD in computer graphics. Interesting 'implementation' issues will arise: How can the slicing in the 4D hypercube be done? Is this efficient enough for interactive performance (100 times per second)? The slice is a 3D model and still has to be rendered on a 2D display (or 3D stereo device). Would it be possible to combine the above two steps in a single operation on the 4D hypercube (selection and transformation for display). What steps can be done in hardware and what needs to be done in software?

- Make the structure dynamic: currently the tGAP structure (including the new smooth tGAP) is a static structure. When an update takes place, the structure has to be recomputed. Due to global optimization criteria, the impact of a local change is not guaranteed to have a local effect; e. g. limited to path in structure from changed object to root of structure, perhaps including sibling. The grouping approach of § 4.3 might be helpful for making more local updates possible.

  Making the structure dynamic also might result in a 5D hypercube (van Oosterom and Stoter, 2010). Again slicing issues arise when we want to create visualizations: slice from 5D to 4D with hyperplane (e. g. select a specific moment in time or alternatively select a specific scale).

Acknowledgements

The authors of this paper want to express their gratitude to the anonymous ICA reviewer for the constructive comments. The authors would further like to thank Dirk de Jong, European Patent Attorney at Vereenigde, for the inspiring questions and discussions during the process of writing the patent claim for the method and system description of true vario-scale maps (patent pending nr. OCNL 2006630).

References

Baars, M., Stoter, J., van Oosterom, P., and Verbree, E. (2004). Rule-Based or Explicit Storage of Topology Structure: a Comparison Case Study. In Toppen, F. and Prastacos, P., editors, *Proceedings of the 7th Conference on Geographic Information Science (CD-ROM)*, pages 765–769. Heraclion: Crete University Press. (Cited on page 7).

Hampe, M., Sester, M., and Harrie, L. (2004). Multiple representation databases to support visualization on mobile devices. In *Proceedings of the XXth ISPRS Congress*, volume XXXV of *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, pages 135–140, Istanbul, Turkey. (Cited on page 15).

Harrie, L., Sarjakoski, L. T., and Lehto, L. (2002). A variable-scale map for small-display cartography. *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, 34(4):237–242. (Cited on page 15).

Haunert, J.-H., Dilo, A., and van Oosterom, P. (2009). Constrained set-up of the tGAP structure for progressive vector data transfer. *Computers & Geosciences*, 35(11):2191–2203. Progressive Transmission of Spatial Datasets in the Web Environment. (Cited on page 15).

Louwsma, J., Tijssen, T., and van Oosterom, P. (2003). Topology under the microscope. GeoConnexion. (Cited on page 7).

Meijers, M. and van Oosterom, P. (2011). The space-scale cube: An integrated model for 2D polygonal areas and scale. Manuscript accepted as short paper at UDMS 2011. (Cited on pages 4 and 14).

Meijers, M., van Oosterom, P., and Quak, W. (2009). A storage and transfer efficient data structure for variable scale vector data. In *Advances in GIScience*, Lecture Notes in Geoinformation and Cartography, pages 345–367. Springer Berlin Heidelberg. (Cited on pages 7 and 14).

Noguera, J. M., Segura, R. J., Ogáyar, C. J., and Joan-Arinyo, R. (2010). Navigating large terrains using commodity mobile devices. *Computers & Geosciences*, In Press, Corrected Proof. (Cited on page 2).

Nöllenburg, M., Merrick, D., Wolff, A., and Benkert, M. (2008). Morphing polylines: A step towards continuous generalization. *Computers, Environment and Urban Systems*, 32(4):248–260. Geographical Information Science Research - United Kingdom. (Cited on pages 2 and 4).

Penninga, F. (2004). Oracle 10g Topology; Testing Oracle 10g Topology using cadastral data. Technical report, Delft University of Technology, Delft. (Cited on page 7).

Sester, M. and Brenner, C. (2005). Continuous generalization for visualization on small mobile devices. In Fisher, P., editor, *Developments in*

Spatial Data Handling, pages 355–368. Springer-Verlag. (Cited on pages 2 and 4).

Thompson, R. M. and van Oosterom, P. (2011). Modelling and validation of 3D cadastral objects. Manuscript accepted as full paper at UDMS 2011. (Cited on page 14).

van Kreveld, M. (2001). Smooth generalization for continuous zooming. In *Proceedings 20th International Cartographic Conference (ICC'01)*, pages 2180–2185, Beijing, China. (Cited on page 2).

van Oosterom, P. (2005). Variable-scale topological data structures suitable for progressive data transfer: The gap-face tree and gap-edge forest. *Cartography and Geographic Information Science*, 32:331–346. (Cited on page 2).

van Oosterom, P. and Stoter, J. (2010). 5D data modelling: full integration of 2D/3D space, time and scale dimensions. In *Proceedings of the 6th international conference on Geographic information science*, GIScience'10, pages 310–324, Berlin, Heidelberg. Springer-Verlag. (Cited on page 17).

van Oosterom, P., Stoter, J., Quak, W., and Zlatanova, S. (2002). The balance between geometry and topology. In Richardson, D. and van Oosterom, P., editors, *Advances in Spatial Data Handling, 10th International Symposium on Spatial Data Handling*, pages 121–135, Berlin. Springer-Verlag. (Cited on page 14).

Vermeij, M., van Oosterom, P., Quak, W., and Tijssen, T. (2003). Storing and using scale-less topological data efficiently in a client-server dbms environment. In *GeoComputation 2003*, University of Southampton, Southampton, UK. (Cited on page 4).