Evaluation of the dual half-edge data structure for implementation of a vario-scale model

Amin Gholami [1], Pawel Boguslawski[1], Martijn Meijers[2], Peter van Oosterom[2]

[1] Wroclaw University of Environmental and Life Sciences, Institute of Geodesy and Geoinformatics, Grunwaldzka 53, 50-357 Wroclaw, Poland – amin.gholami@upwr.edu.pl, pawel.boguslawski@upwr.edu.pl
[2] GIS Technology, Faculty of Architecture and the Built Environment, Delft University of Technology, Julianalaan 134, Delft, The Netherlands- b.m.meijers@tudelft.nl, P.J.M.vanOosterom@tudelft.nl

Keywords: vario-scale, LOD, multi-dimensional modelling, topology, DHE.

Abstract

This paper discusses the representation of the scale dimension using tGAP/SSC-DHE, emphasizing its role in creating vario-scale representations. It outlines the process of integrating the scale dimension as the third dimension in a 2D map before transitioning to 3D or 4D maps. DHE is introduced as an effective data structure for this approach, maintaining crucial connections across different levels of detail and offering enhanced topology. Its strength lies in the ability to modify specific parts of the model while retaining a clear structure, thanks to well-defined operators such as Euler operators. This method provides a versatile framework for representing complex spatial data, allowing for detailed analysis while preserving overall coherence and it updates the shape and structure of the model at the same time through local modifications. Through practical implementation, it demonstrates the potential for improved visualization and understanding of multi-dimensional spatial relationships. Also, the DHE data structure supports slicing by either a horizontal plane to produce a map at a specific scale (depending on the height of the horizontal plane) and tilted planes to produce perspective views with mixed scales.

1. Introduction

The incorporation of maps into many facets of daily life has been further accelerated in recent years by technical developments, especially with the increasing popularity of map-centric smartphone applications. This explosion in the use of interactive maps emphasizes the importance of continuous generalization techniques, which, in contrast to traditional approaches that rely on predefined scale increments, dynamically modify map representations in response to user interactions (Van Oosterom et al., 2014).

While earlier studies have made significant strides toward investigating continuous generalization methods (Cecconi and Galanda, 2002; Sester and Brenner, 2005; Van Kreveld, 2001), the quest for the best answer is still ongoing. The concept of the genuinely smooth vario-scale structure for geographic information was introduced by Van Oosterom et al. (2014), whose groundbreaking work represents a substantial advancement in this field. This creative method makes it easier to move between map scales smoothly, guaranteeing that representational changes closely match user expectations. Moreover, the vario-scale framework enables the development of mixed-scale visualizations, in which features that are both highly detailed and broadly applicable coexist peacefully within the same map context.

Research and development efforts are primarily focused on addressing the problems related to continuous generalization and mixed-scale visualization, especially in the context of 3D computer graphics. By employing the vario-scale approach strategically, researchers hope to reduce geometric data redundancy and eliminate temporal delays in data availability at different map scales, which will improve user experiences and maximize the use of geographic information resources (Van Oosterom et al., 2014).

In the variable-scale approach, the foundation lies in a data structure where each section of the map can be depicted by a topological face (Meijers, 2011). Essentially, any topological data structure, such as Winged-Edge, can serve this purpose. The authentic vario-scale structure, known as a Space-Scale Cube (SSC), is an extension of the topological Generalized Area Partitioning (tGAP), which utilizes a hierarchical tree structure. Vario-LoD is delineated as an additional dimension (the third dimension) for a series of maps spanning various scales; horizontal slice planes can be employed to create new maps in different scales. Similar to tGAP terminology, the third dimension is conceptualized through the notion of "importance" (representing divergence for scale and LoD content) (Meijers, 2011). This "importance" of objects relies heavily on the classification of features and the size of the object (Meijers, 2011). The updated importance value is computed post-generalization (merging) from two preceding objects, as depicted in Figure 1 shows four map fragments that were derived from their tGAP structure.

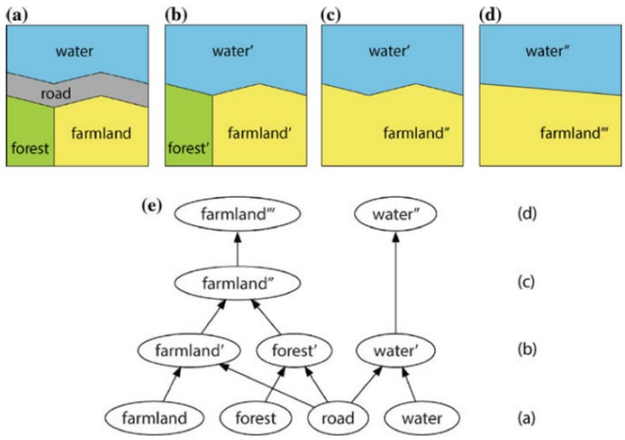

Figure 1. The four map fragments and corresponding tGAP structure. (a) Original map. (b) Result of collapse. (c) Result of merge. (d) Result of simplification. (e) Corresponding tGAP structure (Van Oosterom et al., 2014).

The tGAP model addresses the issue of data redundancy by constructing a hierarchy from the most detailed dataset to the coarsest one. Additionally, it maintains the topology of the horizontal plane while offering Levels of Detail (LoDs) along the scale axis (Meijers, 2011). The initial tGAP structure (Van Oosterom, 2005) is adept at establishing connections between objects across different levels of detail.

Van Oosterom et al. (Van Oosterom et al., 2014) created a 3D model by extrusion of the 2D space with 1D scale which the tGAP model is represented in the two kinds of Space-Scale Cube (SSC); Classic SSC and smooth SSC, called classic tGAP/SSC(Figure 2. a) and smooth tGAP/SSC (Figure 2. b), Within the SSC, 3D volumes (prisms) represent vario-scale 2D area objects, while 2D vertical faces depict vario-scale 1D line objects, such as a collapsed road. Additionally, a 1D vertical line represents the vario-scale 0D point object (Van Oosterom et al., 2014).

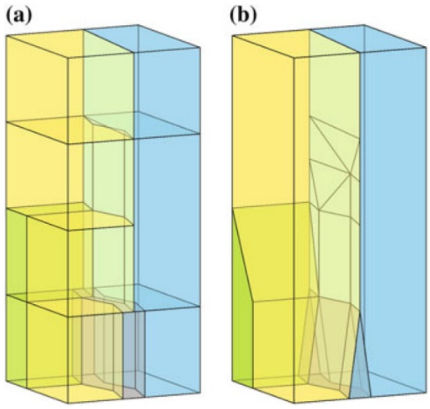

Figure 2. The space-scale cube (SSC) representation in 3D: (a) a wireframe classic tGAP/SSC and (b) a wireframe smooth tGAP/SSC (Van Oosterom et al., 2014).

In the realm of higher-dimensional modeling, numerous data structures exist capable of representing models in four or more dimensions, including polytopal meshes (Sobhanpanah, 1989) and decompositions of polytopes (Bulbul et al., 2009). Despite their capacity to preserve diverse topological relationships, none of these structures furnish a navigable network conducive to efficient model navigation (Karim et al., 2017).

Arroyo Ohori et al. (Arroyo Ohori et al., 2013) recognized two potential data structures suitable for modelling higher dimensions: Generalized Map (G-Maps) and Dual Half-Edge (DHE). The DHE represents a spatial 3D GIS data structure, akin to radial-edge, facet-edge, and half-edge data structures (Boguslawski, 2011).

The complexity of 4D models continues to pose challenges regarding computational and memory demands. These models place significant strain on storage resources due to their heightened complexity. The addition of extra dimensions leads to an exponential surge in the number of entities needed to represent a model. For instance, consider a square, a fundamental 2D representation of a building, which comprises four edges. In contrast, a 3D cube requires 12 edges, and a 4D hypercube necessitates 32, showcasing the escalating complexity with each dimension added. In (Boguslawski, 2024), an extended version of the dual half-edge structure for topological representation of 4D cell complexes is proposed. This feasibility study demonstrates the practical implementation of the Poincaré duality theorem. The proposed data structure remains straightforward, utilizing only two atomic elements in the construction process: nodes and edges. This approach paves the way for future research, which aims to leverage topological links in the fourth dimension to connect consecutive object representations of varying granularity.

The showcased representation possesses a dual characteristic, exhibiting full symmetry between the two structures. These structures encompass two spaces: the primal and the dual, both depicted as graphs. Every element (volume, face, edge, and vertex) within one space corresponds to a counterpart in the dual space, adhering to the 3D Poincaré duality principles (Boguslawski, 2011). A primary cell is denoted by its dual vertex, while a primary face is identified by its dual edge. Conversely, dual faces and cells, when required, are depicted by their primal edges and vertices. Adjacent cells of a complex are connected by a shared face, which is represented by a dual edge (Boguslawski, 2011).

Van Oosterom and Meijers (2014, section 5) introduced the concept of a 4D scale-space hypercube for representing 3D spatial models with varying scales. They discussed various operators within a unified 3D+scale framework, including creation and slicing operators. These operators enable the visualization of spatial data from different perspectives, allowing for detailed views closer to the observer and less detail further away. Additionally, they highlighted the potential of using non-horizontal scale slices to enhance perspective view visualizations, providing detailed views near the viewer and less detailed ones further away. This approach ensures a seamless transition between different levels of detail in computer graphics, addressing common issues such as gaps or overlaps. However, no implementation details were provided, specifically the data structure and algorithms implementing the operators were not provided. The 4D DHE data structure would be the first implementation of the 4D SSC.

The DHE also possesses the ability to manage thematic attributes, facilitating the semantic integration of information. For example, a room can be represented by a dual vertex, while attributes of the primal geometry (such as a room name or room volume) can be associated with the dual vertex of the primal cell. This integration of thematic attribute semantic information is ingrained within the DHE data structure, serving the needs of GIS spatial data management. Navigation within the DHE data structure is enabled by the interconnection of all model elements through pointers and from a navigational point of view, primal and dual spaces are identical – without an extra flag it is not possible to tell which space is navigated (Boguslawski, 2011). The fundamental set utilizing pointers directly comprises four operators which are introduced in Table 1.

Navigational pointer	To navigate
D	from a half-edge in one space to the associated half-edge in the dual space
S	from one half of the edge to the second one
N_V	around a shared vertex (anticlockwise)
N_F	around a shared face (anticlockwise)

Table 1. The meaning of navigational pointers

In order to facilitate access queries across Levels of Detail (LoDs), establishing links between objects in one LoD and their

https://doi.org/10.5194/isprs-archives-XLVIII-4-W11-2024-25-2024

corresponding objects in the adjacent LoDs is imperative (Paul and Bradley, 2015). The DHE dual edge, which is affiliated with the primal face (a loop of XY coordinates), can serve dual roles in hierarchical navigation and querying, alongside storing attribute and thematic semantic information (Karim et al., 2017). Karim et al. (2017) explores the intricacies of scale modeling and proposes novel solutions to address the associated challenges. They emphasise the importance of accommodating different scale models to meet the diverse needs of users and applications. Through an in-depth review, the authors examine the limitations of existing data structures in adequately representing scale dimensions, particularly in GIS modeling contexts. They introduce the DHE data structure as a promising alternative, emphasizing its adaptability to support variable LoD representations and dynamic updates based on Euler operations. Moreover, the paper highlights the versatility of the DHE data structure, not only in facilitating 3D spatial modeling but also in providing a robust platform for integrated space-scale data modeling. By leveraging the advantages of the DHE approach, researchers and practitioners can potentially enhance the efficiency and accuracy of scale modeling across various domains, including GIS and computer graphics. They just reviewed the potential of DHE data structure for integrated 2D-space and scale modeling.

In this paper the tGAP/SSC reconstruction processes and challenges, by DHE data structure are described, the drawback of the tGAP/SSC implementation is that it was not tested for higher dimensions than 3D, but our objective is to extend the concept to 4D (e.g. 3D model with 1D scale). Therefore, this paper considers reconstructing the tGAP/SSC 3D model by using DHE data structure implementation for a 2D map (3D SSC when adding scale as dimension) to better understand what operators are required, before going to 3D space + 1D scale (4D SSC). A construction process uses Euler operators, which update the geometry of the model while preserving its topology, e.g. MEV, Make Edge Vertex, creates a new vertex and edge, where the edge is added to the existing structure; MEF, Make Edge Face, splits an existing face into two faces; MEVVFS, Make Edge Vertex Vertex Face Shell, to create a new edge in empty space; MergeByFace, merge two cells in the shared face to be one cell; SplitByFace, split one cell in one internal face into two cells, etc. (Boguslawski and Gold, 2016).

2. Methodology

The main purpose of the article is to study the feasibility of the DHE data structure for implementing a vario-scale map/model (see Figure 3).

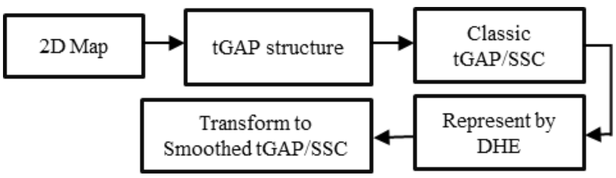

Figure 3. Research methodology.

In this research we reconstruct the modified classic tGAP/SSC model shown in Figure 3, by using a 3D DHE structure; after reconstructing tGAP/SSC by DHE, it needs to transform the classic model to a smooth model to determine a set of Euler-Operators necessary for reconstruction and transformations which this transformation is described in 2.3 and 2.4 section. The construction operators are arranged in layers (see Figure 4). Two more layers than the (Boguslawski and Gold, 2016) layers are added upon layer 4,5 to support Euler operators, 4th layer is divided into two sections, left section is related to which Euler-operators are used to reconstruct modified classic tGAP/SSC by DHE data structure and right section is related to how we can convert vertical faces to smoothed faces and 5th layer is last level that explains how we can transform from classic to smoothed tGAP/SSC.

Each layer depends on the operators from the layer below it. The lowest-level operators rely on pointers and fundamental operations, while those at higher levels become more intricate and specialized (Boguslawski and Gold, 2016).

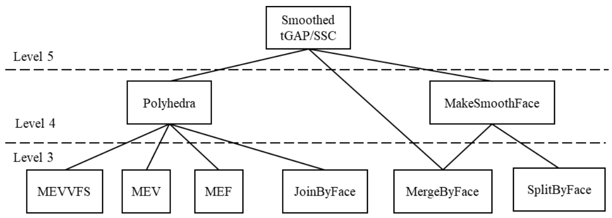

Figure 4. Construction operators are organized in layers: Level 5—Merge all smoothed and classic polyhedral cells Level 4—Create classic polyhedral and convert it to be smoothed; Level 3—the Euler and extended Euler operators.

There are two methods to create the tGAP/SSC structure with the DHE data structure: 1. Convert an existing tGAP/SSC directly into the DHE data structure, 2. Make DHE data structure for the large-scale map, apply map generalization operations and step by step transform the initial structure into the tGAP/SSC. In this paper, the first method is applied (see Figure 3) and the second method will be implemented in the future to automate the whole process. Inside the Euler-operators some navigational pointers are used (Table 1).

Some Euler-operators for reconstructing modified classic tGAP/SSC by DHE are created which are explained in the following sections.

2.1 *MakeFaceShell* operator

This operator creates an edges list in one face (pList) by getting the primal vertices list of a polygon and two dual vertices, it uses *MEVVFS, MEV,* and *MEF* operators, *MEVVFS* operator for constructing a new edge in a space to create a new cell (shell) is used, *MEV* operator creates a vertex and an edge which is linked to the existing model in such a way that one end of the edges is free (not connected to any other edge), *MEF* operator splits a face loop into two parts by adding an edge between two input edges (Table 2).

```
Function MakeFaceShell( pList , I , E ) {
e:=  MEVVFS ( pList(0) , pList(1) , I , E);
eList.Add(e);
for P:=all vertices from pList starting from the third element
to one before the last element
    e:= MEV(P, e.S);
    eList.Add(e);

e := MEF(e.S, eList(0))
eList.Add(e);
return := eList ;
}
```

Table 2. The MakeFaceShell operator

2.2 *Polyhedra* operator

This operator creates a polyhedron cell by getting two primal vertices lists of two faces (ToppList and DownpList) and two dual vertices (External (E) and Internal (I) dual vertices), inside this operator first of all by MakeFaceShell operator, the top and down horizontal faces are created and therefore those primal half-edges lists of two faces are acquired and then by *MEF* operator vertical faces will be created (Table 3).

```
Function Polyhedra( ToppList , DownpList , I , E) {
TopeList := MakeFaceShell ( ToppList, I , E );
DowneList := MakeFaceShell ( DownpList, I , E );
for f1:=all edges from TopeList and f2:=all corresponding edges from DowneList
       MEF(f1 , f2.NV)
}
```

Table 3. The *Polyhedra* operator

After creating the polyhedron cells by using the *Polyhedra* operators, the *JoinByFace* operator should be used to make the topological connection between shared faces (common faces) of two cells, this connection is possible only if the faces have the same number of edges, and also it makes the topological connection between two dual vertices cells, it means it can make hierarchical tree connection in tGAP (one condition is passed). After using these operators, modified classic tGAP/SSC implemented by DHE is represented, and then to transform the classic model to the smoothed model, additional operators are used.

2.3 *MakeSmoothFace* operator

As shown in Figure 5 where one of the objects in scale (b) is removed (from 2D object transit to 0D object) in the next scale (c), in classic polyhedral cell this transition does not make sense (in top polyhedral cell in scale d, this object does not exist), but in the smoothed polyhedral cell this transition dimension between scales b and c is smoothly (Figure 3 and 5).

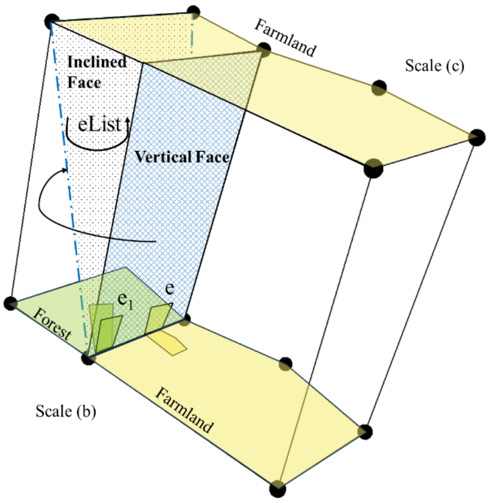

Figure 5. Transform the vertical face to the Inclined face in a classic polyhedra cell

In this paper, in order to transform the classic polyhedral cell to a smoothed polyhedral cell, first, we need to transfer as many vertical faces as possible to inclined faces., In Figure 5, we just need to transfer one vertical face to an inclined face, for this step just first need to create an inclined face by connecting two extra two DHEs (two dashed blue lines) by the *MEF* operator and then split the forest polyhedral cell to two polyhedral cells by *SplitByFace* (eList), between the inclined and horizontal faces is one of those split cells, which we should create a new temporary dual vertex for its DHEs, while this new dual vertex has the farmland attributes, so next step is to merge this temporary polyhedral cell with farmland polyhedral cell by the *MergeByFace* operator (Table 4).

```
Function MakeSmoothFace (eList, E) {
e1:=eList(0);
e:=e1.Adjacent;

SplitByFace (eList)

AllCellEdgesList:= Get all new Polyhedra cell edges for e1
NewDualVertex:= create new dual vertex by new coordinate and e.D.V attributes
for e:= all edges in AllCellEdgesList
     e.D.V= NewDualVertex;

MergeByFace(e1,e, E)
}
```

Table 4. The *MakeSmoothFace* operator

To convert some classic polyhedral cells into smoothed polyhedral cells, multiple vertical faces must be transformed into inclined faces. In such cases, the *MakeSmoothFace* operator can be employed for each face transition individually.

2.4 Smoothed tGAP/SSC

After converting the classic polyhedral cells into smoothed polyhedral cells by the *MakeSmoothFace* operator, it is essential to merge identical objects using the *MergeByFace* operator along their shared faces. This merging process specifically targets horizontal shared faces. As depicted in Figure 3b, the final polyhedral cells are both smoothed and merged. Each object contributes to a single smoothed polyhedral cell, maintaining a topology structure defined by their vertices, edges, and faces.

3. Implementation

This paper employs the Python programming language for coding and relies on the Panda3D library for visual representation. Figure 5 shows how a new edge is constructed in a new shell by the *MEVVFS*(P_0, P_1, I, E) operator. In this paper, black lines are primal HE, red lines are DHE, black circles are primal vertices and orange circles are dual vertices.

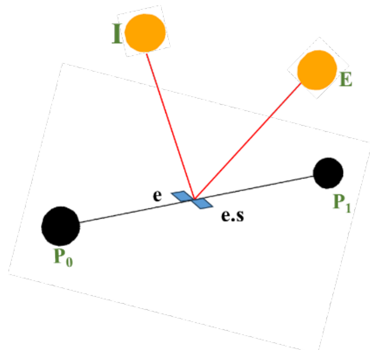

Figure 5. The *MEVVFS* operator

Figure 6 shows the construction of a face by the *MEFFVS*(pList, I, E) operator. In this paper, to simplify the visualization of one face rather than visualize DHE for all edges in one face (Figure 6a) we just visualize it once, and also the DHEs for external dual vertex will not visualize (Figure 6b).

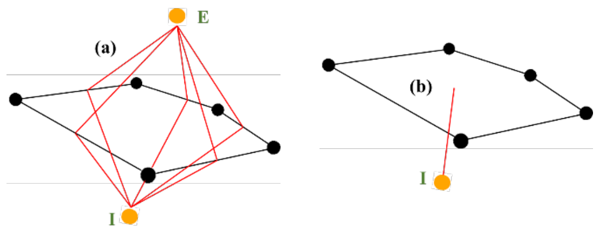

Figure 6. Construct a face with the MakeFaceShell operator (a) Comprehensive DHE face visualization (b) Simplified DHE face visualization

As mentioned in the methodology section, the paper data set is shown in Figure 1, in which the first(bottom) layer contains four polygons: forest, farmland, road, and water body. the second(top) layer contains three polygons: forest, farmland, and water body. Furthermore, by the connection between the top and bottom(down) polygons(faces), the SSC will be created.

Figure 7 shows a farmland polyhedral cell which is created by *Polyhedra*(ToppList, DownpList, I, E) operator, regarding Table 2, the DownFace and TopFace (horizontal faces) have been created by the MakeFaceShell operator and they respectively are the farmland 2D polygon shown in scale (a) and scale (b) of Figure 2, And the vertical faces have been created by the *MEF* operator.

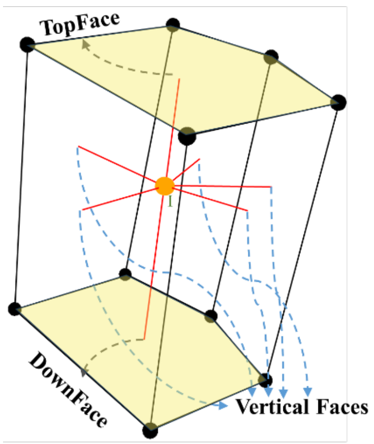

Figure 7. Construct the first farmland polyhedral cell with the *Polyhedra* operator

After constructing the first forest polyhedron by *Polyhedra* operator shown in Figure 8, for the topological connection between forest and farmland polyhedral cells, the *JoinByFace*(e_1,e_2) operator (Boguslawski, 2011; Boguslawski and Gold, 2016, 2010) has been used.

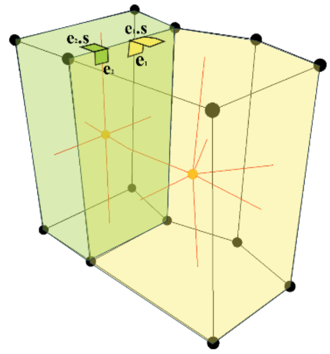

Figure 8. The *MEVVFS* operator

The *Polyhedra* operator is just applicable when the top and down objects are 2D, but e.g. in Figure 1 , the road object in scale (a) is 2D (down object) but in scale (b) is 1D (top object), it means the object is disappeared completely. As shown in Figure 2, there are two kinds of polyhedral cells: classic and smoothed, which in classic this polyhedral cell created by top and down 2D polygons (Figure 3a between scale (a) and (b)) and in smoothed SSC this polyhedral cell created by down 2D polygon and top 1D line. As whereas in these classic polyhedral cells (for this example is first layer), the down object of the next layer (for this example is the second layer) should be a 1D line. Therefore, in this paper some of the smoothed polyhedral cells are created in the classic tGAP/SSC step to avoid split faces in the next classic polyhedron cells on the top layer. In future research, we aim to reconstruct these steps automatically.

For this case, there are several ways to construct the road polyhedron cell by Euler-operators, one of them as shown in Figure 9, first, create two vertical faces (Figure 9a) and then connect them with *MEV* and *MEF* operators (Figure 9b).

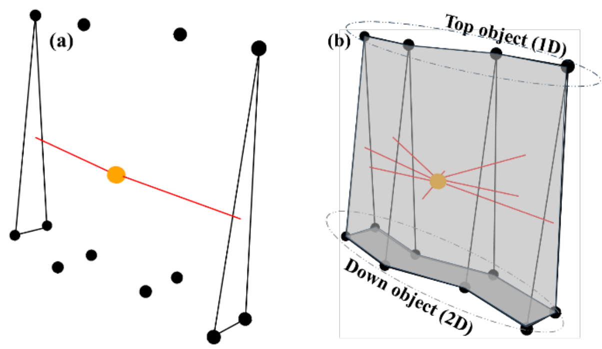

Figure 9. Construct a smoothed polyhedral cell with the Euler-operators (a) create two vertical faces with the MakeFaceShell operator (b) complete the cell with *MEV* and *MEF* operators.

In certain instances of transitioning between vertical faces, unplanned faces emerge as a challenge. Addressing this concern we have two different cases; case (1): when the upper and lower vertices possess an unequal count. In this case, Van Oosterom et al. (2014) introduced a method involving the addition of interpolated vertices followed by triangulation (refer to Figure 10a). However, this paper proposes an approach to rectify such geometric anomalies. Conversely, if the vertex counts differ, our proposed method involves connecting each lower vertex to its nearest upper counterpart by the *MEF* operator. In cases where unplanned faces persist in the final model, they are systematically resolved through a process of triangulation (depicted in Figure 10b).

In case (2), when the upper and lower vertices possess an equal count, we have two conditions; (A) if their corresponding half-edges are paralleled (Figure 10c); a simple edge connection between corresponding vertices suffices by the *MEF* operator. After making every face between corresponding parallel half-edges they will be planned (as exemplified in the road vertical faces depicted in Figure 9b), (B) if their corresponding half-edges are unparalleled (Figure 10d); after making a face between corresponding unparallel half-edges, split the face into two triangulates by the *MEF* operator, after triangulation for case(2).B the final faces are planned but after slicing between them it makes more number of half-edges(6 half-edges) than number of half-edges in lower scale level(4 half-edges), so it means rather than going to simplify more, it goes to be more complicated, however in this paper we are not going to resolve this problem.

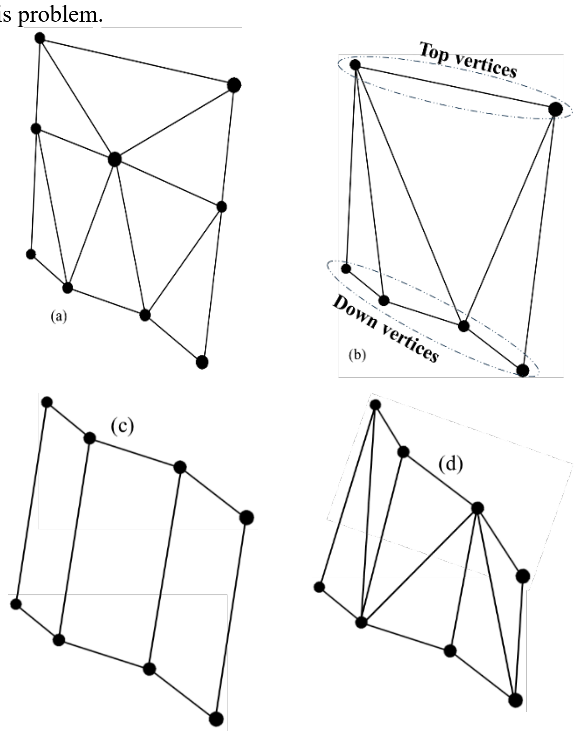

Figure 10. Unplanned vertical faces, Case(1); (a) Triangulated by extra interpolated vertices (b) Triangulated by their vertices. Case(2); (c)Paralleled half-edges (d) Unparalleled half-edges

After constructing all polyhedral cells by the mentioned Euler-operators, the Classic tGAP/SSC is reconstructed (Figure 11) by DHE data structure, the next step is to transfer the classic polyhedral cells to smoothed polyhedral cells and then merge all polyhedral cells from the same objects.

The next step is converting classic tGAP/SSC to smoothed tGAP/SSC, e.g. in scale (b) (Figure 1) we have a classic forest polyhedral cell which should convert to smoothed forest polyhedral cell (Figure 12a) by the *MakeSmoothFace*(eList, E) operator, the first step is split the classic forest polyhedral cell to two smoothed forest polyhedral cells and second step is merge one of smoothed forest polyhedral cell to its neighboured farmland polyhedral cell.

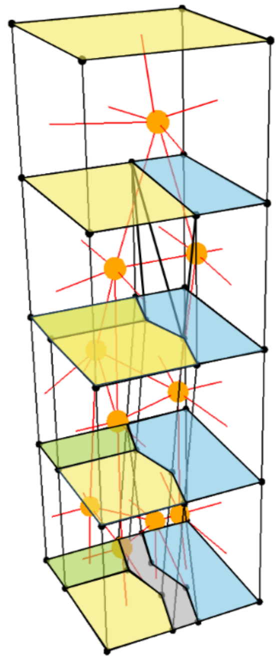

Figure 11. Reconstruct the classic tGAP/SSC by DHE data structure

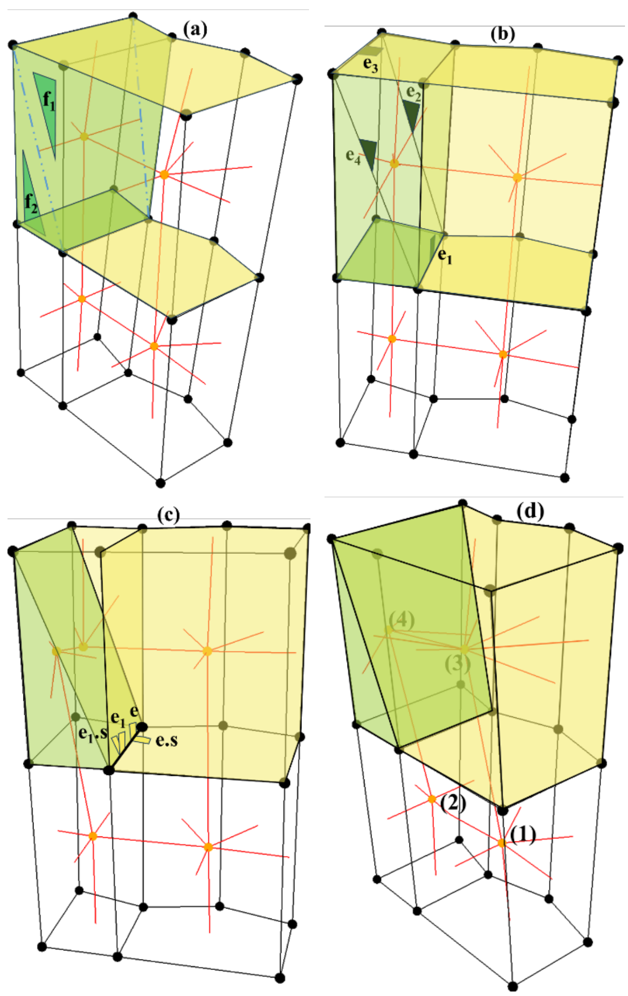

Figure 12. Convert processing of classic polyhedral cells to smoothed polyhedral cells by the *MakeSmoothFace* operator (a)Classic Forest and farmland polyhedral cells (b) Make inclined e_2 and e_4 edges by the *MEF* operator (c) after the *SplitByFace* operator (d) after the *MergeByFace* operator

https://doi.org/10.5194/isprs-archives-XLVIII-4-W11-2024-25-2024

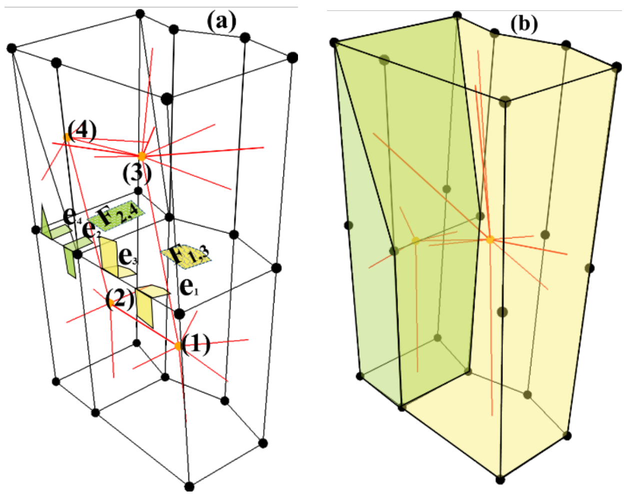

Figure 13. Merge processing of polyhedral cells (a) 4 polyhedral cells (b) Merged polyhedral cells by the *MergeByFace* operator

During the first step, first of all, inclined half-edges(blue dash lines in Figure 12a) should be created by the *MEF* operator which this operator splits related faces to f_1 and f_2 faces, after making these half-edges the eList={e_1,e_2,e_3,e_4}will be chosen from nearest forest polyhedral cell neighbour with the farmland polyhedral cell (Figure 12b) and then split the classic forest polyhedral cell two smoothed forest and farmland polyhedral cells by *SplitByFace(eList)* operator (Figure 12c) and then merge two farmland polyhedral cells by *MergeByFace(*e_1*,e, E)* operator (Figure 12d), the final model is smoothed polyhedral cells.

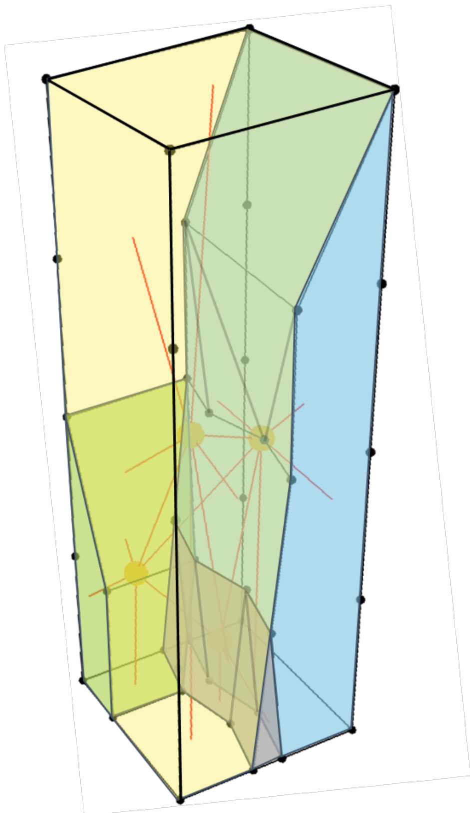

Figure 14. Reconstruct the smoothed tGAP/SSC by the DHE data structure

During the second step of the process, the merge of all polyhedral cells from each object takes place. For example, in Figure 13a, two forest polyhedral cells from the 2nd and 4th objects in Figure 12d, and two farmland polyhedral cells from the 1st and 3rd objects in Figure 12d, require merging along their shared horizontal faces labeled as $F_{2,4}$ and $F_{1,3}$ respectively. This merging operation is carried out using the *MergeByFace*(e_2,e_4, E) and *MergeByFace*(e_1, e_3, E) operators, as demonstrated in Figure 13b.

The tGAP/SSC method demonstrates its adaptability by effectively generating detailed map layers across a range of scales, as indicated in this paper by the interval [a, d]. This is achieved through the implementation of the slicing technique. By employing this method, finely detailed maps are produced, and a smooth integration is achieved across different scales, thereby improving the accuracy and completeness of geographic representations.

This paper outlines the slicing of horizontal and tilted planes, as illustrated in Figure 15. The subsequent research papers will delve into the step-by-step processing of these slicing planes and introduce new slicing operators. The final Smoothed tGAP/SSC is shown in Figure 14.

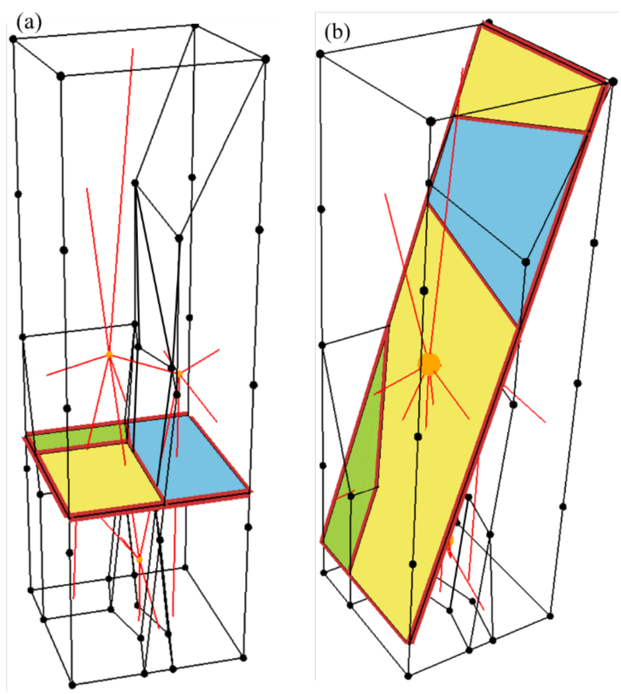

Figure 15. Slice the smoothed tGAP/SSC by (a) a Horizontal plane and (b) a tilted plane

Slicing with the tilted plane may result in a model with the topology that did not exist in any of initial 2D models, i.e. individual horizontal layers. In Figure 15b, the farmland area is divided into two parts by the water body. This issue is not taken into consideration in this paper, as the focus is put on construction operators for 2D+1D model and geometrical slicing with planar surfaces. However, this issue was addressed by Peng et al. (2023).

4. Conclusion and Future Work

This paper presents new construction operators and implementation of the vario-scale model - tGAP/SSC - using the DHE data structure, which results in a volumetric model. This allows for straightforward application of standard geometric operators, such as slicing with planar surfaces. his paper explains how to represent the scale dimension as the additional spatial dimension and provides a detailed example of a simple vario-scale model construction. It proposes DHE as a good candidate for the implementation , as the data structure is able to represent connections between different levels of detail and is able to provide a simple representation of spatial objects and to store spatial relations between them, which is not available in case of other data structures (Karim et al., 2017). The strength of this data structure lies in its ability to make changes to specific parts of the model while maintaining a clear structure, thanks to well-defined operators like Euler operators. Essentially, it is a straightforward and effective way to model scales since it updates both the geometry and topology of the model at the same time through local modifications.

The solution proposed in this paper supports standard geometrical functions, such as slicing using horizontal and inclined planes. In future research, we are going to introduce dedicated operators for slicing based on Euler operators.

In our research, we aim at the challenging task, which is an automated method of vario-scale model construction, where individual objects at consecutive layers represented in different scales are mapped to a space-scale cube. Another challenge is to introduce more dimensions into the model - to represent three spatial dimensions and scale in a consistent 4D model .

Acknowledgments

This research work was funded in whole by the National Science Centre (NCN), Poland as part of the research program OPUS, project no.: 2021/41/B/ST10/03178.

References

Arroyo Ohori, K., Boguslawski, P., Ledoux, H., 2013. Representing the Dual of Objects in a Four-Dimensional GIS, in: Abdul Rahman, A., Boguslawski, P., Gold, C., Said, M.N. (Eds.), Developments in Multidimensional Spatial Data Models, Lecture Notes in Geoinformation and Cartography. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 17–31. https://doi.org/10.1007/978-3-642-36379-5_2

Boguslawski, P., 2011. Modelling and Analysing 3D Building Interiors With the Dual Half-Edge Data Structure. Wrocław University of Environmental and Life Sciences.

Boguslawski, P., Gold, C., 2016. The dual half-edge—A topological primal/dual data structure and construction operators for modelling and manipulating cell complexes. ISPRS Int. J. Geo-Inf. 5, 19.

Boguslawski, P., Gold, C., 2010. Euler Operators and Navigation of Multi-shell Building Models, in: Neutens, T., Maeyer, P. (Eds.), Developments in 3D Geo-Information Sciences, Lecture Notes in Geoinformation and Cartography. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 1–16. https://doi.org/10.1007/978-3-642-04791-6_1

Bulbul, R., Karimipour, F., Frank, A.U., 2009. A simplex based dimension independent approach for convex decomposition of nonconvex polytopes, in: 10th Lnternational Conference on GeoComputation (GeoComputation 2009). UNSW, Sydney, Australia.

Cecconi, A., Galanda, M., 2002. Adaptive Zooming in Web Cartography. Comput. Graph. Forum 21, 787–799. https://doi.org/10.1111/1467-8659.00636

Karim, H., Abdul Rahman, A., Boguslawski, P., Meijers, M., Van Oosterom, P., 2017. The Potential of the 3D Dual Half-Edge (DHE) Data Structure for Integrated 2D-Space and Scale Modelling: A Review, in: Abdul-Rahman, A. (Ed.), Advances in 3D Geoinformation, Lecture Notes in Geoinformation and Cartography. Springer International Publishing, Cham, pp. 477–493. https://doi.org/10.1007/978-3-319-25691-7_27

Meijers, B.M., 2011. Variable-scale Geo-information. NCG Delft.

Paul, N., Bradley, P.E., 2015. Integrating Space, Time, Version, and Scale using Alexandrov Topologies. Int. J. 3- Inf. Model. IJ3DIM 4, 64–85.

Peng, D., Meijers, M., Van Oosterom, P., 2023. Generalizing Simultaneously to Support Smooth Zooming: Case Study of Merging Area Objects. J. Geovisualization Spat. Anal. 7, 12. https://doi.org/10.1007/s41651-022-00109-x

Sester, M., Brenner, C., 2005. Continuous Generalization for Visualization on Small Mobile Devices, in: Fisher, P.F. (Ed.), Developments in Spatial Data Handling. Springer, Berlin, Heidelberg, pp. 355–368. https://doi.org/10.1007/3-540-26772-7_27

Sobhanpanah, C., 1989. Extension of a boundary representation technique for the description of n dimensional polytopes. Comput. Graph. 13, 17–23.

Van Kreveld, M., 2001. Smooth generalization for continuous zooming, in: Proc. 20th Intl. Geographic Conference. pp. 2180–2185.

Van Oosterom, P., 2005. Variable-scale Topological Data Structures Suitable for Progressive Data Transfer: The GAP-face Tree and GAP-edge Forest. Cartogr. Geogr. Inf. Sci. 32, 331–346. https://doi.org/10.1559/152304005775194782

Van Oosterom, P., Meijers, M., Stoter, J., Šuba, R., 2014. Data Structures for Continuous Generalisation: tGAP and SSC, in: Burghardt, D., Duchêne, C., Mackaness, W. (Eds.), Abstracting Geographic Information in a Data Rich World, Lecture Notes in Geoinformation and Cartography. Springer International Publishing, Cham, pp. 83–117. https://doi.org/10.1007/978-3-319-00203-3_4

Van Oosterom, P, Meijers, M., 2014. Vario-scale data structures supporting smooth zoom and progressive transfer of 2D and 3D data, In: International Journal of Geographical Information Science, 28(3), pp. 455-478, http://dx.doi.org/10.1080/13658816.2013.809724.