Visualization of Hierarchies in Space and Time

S. Hadlak, C. Tominski, H.-J. Schulz, and H. Schumann

Abstract— When visualizing data, spatial and temporal references of these data often have to be considered in addition to the actual data attributes. Nowadays, structural information is becoming more and more important. Hierarchies, for instance, are frequently applied to help dealing with large and complex data. Hence, a visual depiction of hierarchical structures in space and time is required. While there are several techniques addressing specific aspects of spatio-temporal visualization, approaches that cope with space, time, data, *and* structure are rare. With this paper we take a step to fill this gap. By combining various well-established concepts we achieve a reasonably complete visualization of all of the aforementioned aspects, where our focus is on hierarchical structures. We embed hierarchies directly into regions of a map display using a novel variant of a point-based layout. Layering and animation are applied to visualize temporal aspects. Depending on analysis goals, users can switch between representations that emphasize data attributes or hierarchical structures. Interaction techniques support users in navigating the data and their visualization.

Index Terms—hierarchy visualization, geo visualization, time visualization.

1 Introduction

Hierarchical data structures are useful in many domains. Hierarchies predominantly help us in structuring large information spaces (e.g., the ACM classification to catalogue computer science literature) and in solving problems that are too complex to cope with otherwise (e.g., agglomerative clustering to handle large data sets). To facilitate the use of hierarchies it is only natural to work with interactive visual representations. Classic tree visualizations like node-link diagrams (e.g., tree views in user interfaces) or treemaps (e.g., The Map of the Market[1]) have been developed for this purpose. The literature provides a wealth of other approaches to create expressive visual mappings of tree structures. Most of the known techniques consider hierarchies as isolated abstract data structures. Hierarchies that are embedded in a spatial context and that, moreover, may be time-varying have not yet been addressed sufficiently.

We describe an approach for visualizing attributed hierarchies that have an anchor in geographic space and change over time. Spatial reference is encoded by embedding a layout of the hierarchy into regions of a map display. Our goal was to find an embedding that uses the available screen real estate in maps efficiently. The main challenge in accomplishing this goal are the usually irregular shapes of geographic areas. A point-based layout algorithm is used to address this issue. The visual encoding of temporal aspects is based on creating separate map layers, one for each time step. A three-dimensional arrangement of the layers, similar to the classic space-time-cube approach, is applied to visualize a series of successive time steps. Additional visual indicators help users to recognize specific changes between time steps. Data attributes that might be attached to the hierarchy are visualized by color-coding and dedicated 3D glyphs. Interaction facilities allow users to navigate in space and time.

Next, we briefly explain basic notations and review previous work. Section 3 will detail on our approach, in particular on the visualization design and on interaction. Conclusion and possible next steps for future work are given in the last section.

2 Basic Notations & Related Work

2.1 Basic Notations

Graph structures in general describe entities and relations between these entities. Our approach focusses on hierarchies (i.e., connected acyclic graphs) and associated data given in a spatial and temporal frame of reference. We assume that geographic *space* is modelled as a set of disjoint two-dimensional *areas*

$$S = \{A \mid A \subset \mathbb{R}^2\}$$

where $A \neq B \rightarrow A \cap B = \varnothing$ holds for any two areas $A, B \in S$. The time axis is defined as an ordered set of time steps

$$T = (t_1, \ldots, t_n)$$

with $n \in \mathbb{N}$. We define

$$H_{A,t_i} = (V_{A,t_i}, E_{A,t_i})$$

as the hierarchy for area $A \in S$ at time step $t_i \in T$. Such a hierarchy consists of a set of *nodes*

$$V_{A,t_i} \subset V, \quad V = \bigcup_{A \in S,\, t_i \in T} V_{A,t_i}$$

of which one is a designated *root* node $r_{A,t_i} \in V_{A,t_i}$ and a set of edges

$$E_{A,t_i} \subseteq (V_{A,t_i} \times V_{A,t_i}) \subset E, \quad E = \bigcup_{A \in S,\, t_i \in T} E_{A,t_i}.$$

Data attributes of nodes and edges are defined by means of mappings

$$d_V : S \times T \times V \rightarrow \{D_1^V \times \ldots \times D_p^V\}$$
$$d_E : S \times T \times E \rightarrow \{D_1^E \times \ldots \times D_q^E\}$$

where D_*^V and D_*^E are the domains of *node attributes* and *edge attributes*, respectively.

There are various examples for data that follow this definition. One example regards spatio-temporal data that are too large to handle in a simple manner. Hierarchical aggregation often helps in dealing with huge data sets. Depending on analysis tasks and goals, it is not uncommon that an aggregation hierarchy is computed for each area and each time step, that is, the hierarchy may be different from area to area and from time step to time step. Another example are data that are given with respect to an existing hierarchy. Human health data, for instance, may be structured according to the ICD10 classification of diseases[2]. In this case, the hierarchical structure remains fixed, but the attributes of the nodes of the hierarchy (e.g., number of cases or number of deaths per disease) varies over space and time. For a last example, we may think of organizational structures, e.g., prime minister, ministers, chief officers, etc. of administrative geographical units. Such structures obviously differ among countries and change over time.

• *The authors are with the Institute for Computer Science, University of Rostock, Germany.*

[1] www.smartmoney.com/map-of-the-market/

[2] http://www.who.int/classifications/icd/en/

From the previous definition and examples one can see that several aspects need to be taken care of for visualization. First, multiple hierarchies must be embedded in the spatial frame of reference. In fact, each geographic area has its own sequence of hierarchies. How to visualize the sequences in order to represent changes over time is the second question to be investigated. Finally, the data attributes associated with nodes and edges should be communicated. The attributes may be external data attached to the graph structure or may be internal system-computed properties derived from the hierarchy (e.g, depth).

Spatial and temporal references also demand for appropriate interaction techniques: Browsing back and forth in time to view different parts of the time axis, as well as navigating in space to visit different geographic regions are crucial for data exploration.

2.2 Related Work

To our knowledge, the aforementioned needs have been satisfied only partially in previous work. Nonetheless, the literature provides a variety of interesting solutions for specific aspects.

Spatio-Temporal Data Visualization The visualization of geospatial data is mainly based on map displays [7]. Cartography and geovisualization provide guidelines that help in generating efficient map displays in concert with expressive techniques for presenting data on (interactive) maps. When visualizing time-dependent data there are basically two options to follow [1]. First, one can map time (in the data) to time (in the real world), commonly implemented as animations or slide shows, or as interactive navigation in time. Second, it is possible to map data time to presentation space. All techniques that align data items with a time axis belong to this class.

Approaches that have to visualize both spatial and temporal dependency commonly pursue a combination of map displays with additional temporal encoding [2]. Animated maps are used in many cases, since they provide a good overview and convey major trends in the data. However, detailed visual comparison is difficult using animated maps. In this case, techniques that show several time steps in a single visual representation are better suited; small multiple displays or the classic space-time-cube are prominent examples.

Most techniques for spatio-temporal visualization focus on data, not on hierarchical structures. Such structures are often seen as isolated objects for which separate visualization techniques can be applied.

Approaches to Represent Hierarchies Classic node-link displays require a spatial layout of the hierarchy [3]. Various algorithms are known to generate linear or radial layouts in two-dimensional, three-dimensional, and also hyperbolic space (e.g., [9], [10], [8]). In contrast to node-link displays, which explicitly visualize node relations (i.e., edges), implicit hierarchy visualizations generate special node arrangements that allow users to derive relationships mentally. The most prominent example are classic treemaps [12].

Just like spatio-temporal data visualization often neglects structural aspects, so does classic hierarchy visualization disregard spatio-temporal aspects. Nonetheless, first steps have been taken to combine hierarchies and space as well as hierarchies and time.

Representing Hierarchies in Space Multiple views are one possibility to represent geo-spatial hierarchical data. The idea is to provide interactive linking facilities to allow user to mentally associate map display and hierarchy visualization, which are shown in separate views. An expressive example is the combination of treemaps and choropleth maps in [6]. On the other hand, adapted maps like cartograms or RecMaps [5] are used. Such maps are better suited to integrate hierarchies into the map display. A similar approach is pursued in [18]. They modified treemaps in a way such that positions of nodes in the treemap layout reflect node locations in geographical space. Despite these examples, however, not much research has investigated a direct integration of hierarchy representations into regions of maps.

Representing Hierarchies in Time Two concepts assist in visualizing time-varying hierarchies. One is the cumulative approach, which creates a superhierarchy that unites nodes and edges of all time steps. This super structure can be visualized to gain a static overall view of the hierarchy. An example are Contrast Treemaps [16], which support visual comparison. The other class of approaches is based on generating sequential views on time-varying hierarchies. The idea is to visualize the hierarchy at each time step, for instance by means of animations showing one time step after the other (e.g., TimeTrees [4]), or by drawing the hierarchy on different temporal layers (e.g., Code Flows [13]). Conceptually both examples corresponds to mapping time to time (in an animation) and time to space (for layering), respectively.

In summary, we see that partial solutions do exist to visualize data in space and time and to visualize hierarchies (isolated, in space, *or* in time). However, techniques to combine data and hierarchy visualization and representations of spatial and temporal contexts are still scarce. In the next section, we present an approach that offers such a combined strategy as a piece to fill this gap.

3 Visualizing Hierarchies in Space and Time

Our approach addresses four main issues: (1) Embedding of the hierarchy representation into the map display, (2) visualization of temporal aspects, (3) encoding of data associated with the hierarchy, and (4) appropriate interaction for navigation in space and time.

3.1 Embedding Hierarchies in Irregularly Shaped Areas

Due to the number of aspects that need to be visualized, we have to use screen real estate efficiently. Therefore, we went for a direct embedding of the hierarchy into the map display. The biggest problem concerning this embedding are the usually irregular shapes of geographic areas. Most hierarchy layouts, however, are restricted to specific shapes, like rectangles or circles, and cannot be adapted easily to handle irregular shapes.

The layout we use here is an adapted version of the point-based layout described in [11]. The point-based approach uses a space-efficient layout that allocates a unique pixel to each node. The layout is fixed and can be pre-computed independently. To visualize a concrete hierarchy, the only necessary step is to assign nodes to the pre-computed node positions, which is done in a well-defined way. Due to pre-computation and use of unique pixels, larger hierarchies can be visualized effectively. Moreover, we can quite easily adapt the final shape of the hierarchy representation by neglecting certain locations during the assignment phase. To achieve better approximation of the hierarchy layout to the underlying geographical area, we operate on subareas and assign to them hierarchy subtrees depending on the sizes of subarea and subtree. We suggest the following recursive procedure for finding a good placement of the subtrees, and hence, for the layout of the hierarchy:

1. compute center of area
2. subdivide area into subareas
3. map subtrees onto subareas
4. apply procedure recursively

To embed a hierarchy H_{A,t_i} into the area $A \in S$ one starts with the computation of the center $c \in A$. Secondly, A is subdivided into k subareas $A^1, \ldots, A^k$, where k is the number of subtrees H^j_{A,t_i} $(1 \leq j \leq k)$ below the root r_{A,t_i}. Then, subareas and subtrees are sorted by size and are assigned to each other accordingly. This procedure continues recursively for each subarea and its associated subtree. While steps 3. and 4. are straight-forward, steps 1. and 2. should be explained in more detail.

Compared to regular and convex shapes, it is not a simple task to determine a central position c of an irregular area A, which is necessary to place the root node of a subtree. Obviously, a center has to be inside the area and it should have a maximum distance to the area boundary. Several possibilities exist to choose a center point, but none is without problems. For example, the barycenter of a concave area may be outside of that area, while the center of the largest inscribed circle may turn out to be far from being central. We utilize the barycenter, the

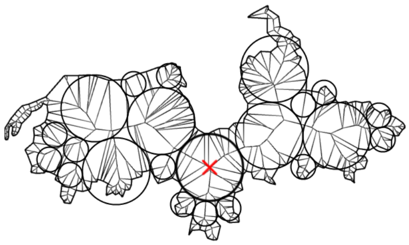

Fig. 1. Subdivision of an irregular area along the area's skeleton.

center of the largest inscribed circle, and the skeleton point with the highest importance [14] and evaluate the results of the different methods. For each center candidate, we compute the visibility polygon, namely that part of the area that can be seen from the candidate. Since one can interpret this as a measurement of centrality, we choose the candidate with the largest visibility polygon.

The chosen center is the starting point for area subdivision. The first subarea A^1 is defined as the largest inscribed circle around the area's center $c \in A$ (red mark in Figure 1). In a greedy manner, we then try to find further subareas A^i of the same size as A^1 along the area's skeleton. If $i < k$, that is, we have not yet found enough subareas to layout k subtrees, the greedy procedure continues with a search for smaller subareas[3]. If still $i < k$, the algorithm starts splitting the largest A^i into further subareas. This procedure is guaranteed to terminate and to generate an appropriate number of $m \geq k$ subareas. If $m \gg k$ we can even reunite some A^i to get closer to k, and thus, to use space more efficiently.

After the subdivision has been computed, we assign subtrees to subareas and continue with the procedure recursively. In summary, our approach is capable of embedding a hierarchy layout directly into a 2-dimensional geographic area of irregular shape, while being computationally and space efficient.

3.2 Representing Temporal Aspects

For the hierarchies as we consider them here, several changes may occur over time. First, the sets of nodes and edges may change from one time step to another, that is $V_{A,t_i} \neq V_{A,t_{i+1}}$ and $E_{A,t_i} \neq E_{A,t_{i+1}}$ for $A \in S$ and $t_i, t_{i+1} \in T$. Nodes can be added or deleted. The same holds true for edges, which also implies that a node can move in the hierarchy – exactly then when it is connected via different edges. As node and edge sets change, so do global properties of the hierarchy (e.g., maximum depth) and node attributes (e.g., number of children). We also indicated that external data attributes d_V and d_E, which are associated with nodes and edges, respectively, are usually subject to change. Often the analysis of the evolution of data attributes over time is the primary goal of visualization.

We combine several concepts to communicate the temporal aspects of the data. For an initial overview, straight-forward animation of the map display has proven useful. When it comes to discerning details in the course of time, animated maps are not the best choice [17].

Visualizations that show a sequence of times steps are better suited in this case. We follow the idea of using the third dimension of the presentation space to represent the time axis (analog to the space-time-cube approach). To that end, we consider a subset of successive time steps $(t_i, \ldots, t_{i+s})$ with $1 \leq i < i+s \leq n$ and render for each of them a separate layer L_i representing the map and the embedded hierarchy layouts. These semitransparent layers are aligned with the time axis that emanates perpendicularly from the base map layer L_i. To facilitate identification of the aforementioned changes in between two layers, visual cues are added. Differently colored links between subsequent layers are used to indicate nodes that have moved or whose attribute values have changed significantly. Significance is determined by a user-selectable threshold. According to a thermometer scale, we visualize positive attribute changes as red links, negative changes are shown in blue. Links representing node movements are colored with a shade of grey. Addition or deletion of nodes and edges is indicated by spikes that start at a layer L_i and point either to the next layer L_{i+1} or the previous one L_{i-1}. In particular, spikes that represent deletion leave a layer in the direction of the time axis and are shown in blue. Those that mark addition enter a layer and are shown in red. Figure 2 illustrates links and spikes. The layering approach in combination with the described visual cues allows users to compare successive time steps more closely.

However, to avoid problems caused by overplotting and visual cluttering still a restriction must be imposed: only a smaller subsequence of time steps (e.g., $s < 10$) can be represented in a single visualization. To overcome this restriction and to allow for comparison of time steps that may be located anywhere on the time axis, multiple views are provided. Each view may be configured independently to show separate portions of the temporal domain, either a single timestep using a base map or a subsequence of time steps using the layering method. Thanks to a docking framework[4], views can be arranged arbitrarily to suit the users' needs. It is possible to align views in a row to resemble a temporal sequence, or to set up larger focus views and smaller views for those parts of the time axis the user deems to be contextual.

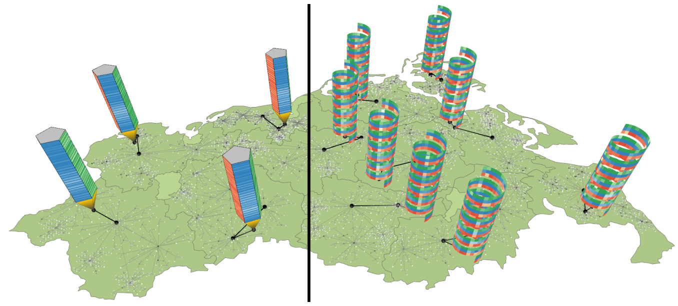

Fig. 3. Pencil and helix glyphs visualize data attributes associated with hierarchy nodes.

3.3 Encoding Associated Data Values

Now that hierarchies are embedded in space and time, the question that remains to be answered is how to visualize the data attributes. Since the presentation is already packed with information, only little space is available for data visualization. Therefore, it makes sense to follow a two-fold strategy: The user can decide if the visualization should focus on structural aspects or data aspects. If structure is selected, only one data attribute is encoded to the nodes and edges of the hierarchy layout. We use color, size, and shape variations sparingly to keep structural aspects in the focus. If data aspects should be emphasized, the hierarchy layout is dimmed and additional visual representations of data attributes are faded in. Pencil and helix glyphs as described in [15] fit well into our space-time-cube visualization. They are applied to show the development of multiple data attributes over time. Figure 3 shows pencil and helix glyphs side by side. Pencil glyphs are suited to visualize linear development, whereas helix glyphs are useful for exploring cyclic patterns. The glyphs are positioned with respect to nodes that users may select in the base map, and each glyph represents the data attributes of its associated node via color-coding. Optionally, the areas of the base map may be color-coded to visualize a single aggregated data value (e.g., to indicate high variance for an area over time).

3.4 Interaction for Data Exploration

The previous paragraphs describe visualizations that can be quite complex. As we visualize complex data, the visualization is endangered by visual cluttering. Therefore, it is crucial to provide effective methods to interact with the data and their representation.

To keep the visual load at an digestible level, our approach operates on user-chosen subsets of areas, time steps, nodes, and data attributes. Where possible, we allow for direct selection from the visualization

[3] Searching for subareas of size $\frac{1}{5}A^1, \ldots, \frac{1}{5^4}A^1$ has proved practical.

[4] www.infonode.net

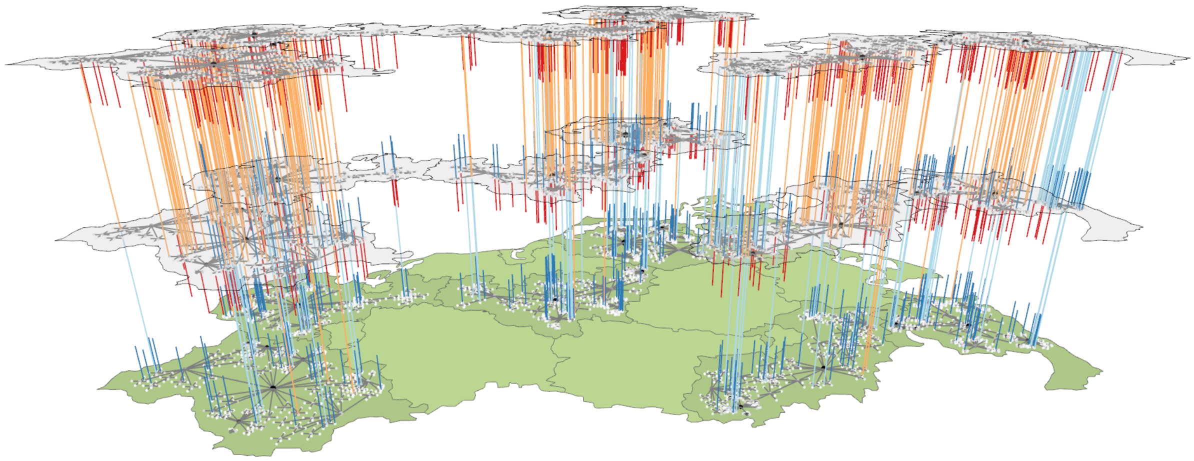

Fig. 2. Visualization of hierarchies for selected areas and three time steps. – Colored links between layers indicate significant increase (red) or decrease (blue) of node attributes. Addition and deletion of nodes is shown via red and blue spikes, respectively.

(e.g., selection of areas or nodes) and provide dedicated user interface elements (e.g., calendar views to select time spans) otherwise. Visualization parameters can be set analogously.

A particular concern is the navigation of the map display because we are facing the challenge of 3D navigation. Our approach combines visualization techniques that are best viewed as a 2D projection (e.g., hierarchy embedded in map areas, color coding) with techniques that intrinsically require 3D interaction (e.g., layering of time steps, 3D glyphs). Therefore, a single interaction metaphor alone can not satisfy interaction needs. As a consequence, we combine object-oriented navigation (i.e., the user manipulates objects in the 3D scene) with user-oriented navigation (i.e., the user moves through the 3D scene). To alleviate 3D occlusion problems, we allow for easy orbiting of points of interest (e.g., a selected glyph or node in a hierarchy layout). A special mode for moving along the z-axis facilitates navigation in time. We also provide a "panic button" that resets the visualization to a well-defined default view.

4 Conclusion

This work described the visualization of hierarchical structures and associated data in time and space. We combined several techniques and concepts to arrive at a solution that integrates spatial, temporal, structural, and data aspects. Key points in our combined strategy are the point-based layout to embed hierarchies into areas of a map, the layering of time steps (and corresponding visual cues), the integration of dedicated data representations, and interactivity.

We understand our work as step towards visualizing hierarchies in time and space, not as a final solution. Many research questions need to be answered in the future. How can we deal with large numbers of unevenly distributed time steps? Is it possible to find better ways to tackle overplotting of layers? Does our approach scale for globe-like visualization and 3D terrains? How can data attributes and structural aspects be combined more efficiently?

The focus+context concept seems to be a good starting point in the search for valid solutions to these problems. Depending on data characteristics, analytical needs, and user interests, relevant parts should be automatically highlighted in the visualization, others should be dimmed. The natural structure of time (years, quarters, months, weeks, days, etc.) as well as different scales of space (continents, countries, counties, etc.) will surely prove useful in this endeavor. Moreover, knowledge about the effectiveness of visual encodings is required, which in turn means that evaluations need to be conducted. We believe that the described visualization and interaction facilities help users in exploring attributed hierarchies in time and space. However, we also think that much more research is required to understand visualization and interaction needs and to design solutions that adapt themselves in a user-centered and task-oriented manner.

Acknowledgements

We acknowledge support by "VisMaster" and "dIEM oSiRiS".

References

[1] W. Aigner, S. Miksch, W. Müller, H. Schumann, and C. Tominski. Visualizing time-oriented data - A systematic view. *Computers & Graphics*, 31(3):401–409, 2007.

[2] N. Andrienko and G. Andrienko. *Exploratory Analysis of Spatial and Temporal Data*. Springer, Berlin, Germany, 2006.

[3] G. D. Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, 1999.

[4] S. K. Card, B. Suh, B. A. Pendleton, J. Heer, and J. W. Bodnar. Time-tree: Exploring time changing hierarchies. In *IEEE Symposium on Visual Analytics Science And Technology*, pages 3–10, 2006.

[5] R. Heilmann, D. A. Keim, C. Panse, and M. Sips. Recmap: Rectangular map approximations. In *IEEE Symposium on Information Visualization*, pages 33–40, 2004.

[6] M. Jern, J. Rogstadius, and T. Aström. Treemaps and choropleth maps applied to regional hierarchical statistical data. In *International Conference Information Visualisation*, pages 403–410, 2009.

[7] A. M. MacEachren. *How maps work: Representation, Visualization and Design*. The Guilford Press, New York, 1995.

[8] T. Munzner. H3: Laying out large directed graphs in 3d hyperbolic space. In *IEEE Symposium on Information Visualization*, pages 2–10, 1997.

[9] E. M. Reingold and J. S. Tilford. Tidier Drawings of Trees. *IEEE Transactions on Software Engineering*, 7(2), 1981.

[10] G. G. Robertson, J. D. Mackinlay, and S. K. Card. Cone trees: animated 3d visualizations of hierarchical information. In *Proceedings of Human factors in computing systems (CHI)*, pages 189–194, 1991.

[11] H.-J. Schulz, S. Hadlak, and H. Schumann. Point-based visualization for large hierarchies. *IEEE Transactions on Visualization and Computer Graphics*, 2010. (under review).

[12] B. Shneiderman. Tree visualization with tree-maps: 2-d space-filling approach. *ACM Transactions on Graphics*, 11(1):92–99, 1992.

[13] A. Telea and D. Auber. Code flows: Visualizing structural evolution of source code. In *Computer Graphics Forum*, pages 831–838, 2008.

[14] A. Telea and J. J. van Wijk. An augmented fast marching method for computing skeletons and centerlines. In *Proceedings of the Symposium on Data Visualisation*, pages 251–260, 2002.

[15] C. Tominski, P. Schulze-Wollgast, and H. Schumann. 3D Information Visualization for Time Dependent Data on Maps. In *International Conference on Information Visualisation*, pages 175–181, 2005.

[16] Y. Tu and H.-W. Shen. Visualizing changes of hierarchical data using treemaps. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1286–1293, 2007.

[17] B. Tversky, J. B. Morrison, and M. Betrancourt. Animation: Can It Facilitate? *International Journal of Human-Computer Studies*, 57(4), 2002.

[18] J. Wood and J. Dykes. Spatially ordered treemaps. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1348–1355, 2008.