The Four-Level Nested Model Revisited: Blocks and Guidelines

Miriah Meyer
University of Utah
miriah@cs.utah.edu

Michael Sedlmair
University of British Columbia
msedl@cs.ubc.ca

Tamara Munzner
University of British Columbia
tmm@cs.ubc.ca

ABSTRACT

We propose an extension to the four-level nested model for design and validation of visualization systems that defines the term "guidelines" in terms of blocks at each level. Blocks are the outcomes of the design process at a specific level, and guidelines discuss relationships between these blocks. Within-level guidelines provide comparisons for blocks within the same level, while between-level guidelines provide mappings between adjacent levels of design. These guidelines help a designer choose which abstractions, techniques, and algorithms are reasonable to combine when building a visualization system. This definition of guideline allows analysis of how the validation efforts in different kinds of papers typically lead to different kinds of guidelines. Analysis through the lens of blocks and guidelines also led us to identify four major needs: a definition of the meaning of block at the problem level; mid-level task taxonomies to fill in the blocks at the abstraction level; refinement of the model itself at the abstraction level; and a more complete set of guidelines that map up from the algorithm level to the technique level. These gaps in visualization knowledge present rich opportunities for future work.

Categories and Subject Descriptors

H.5.2 [**Information Systems Application**]: User Interfaces—*Evaluation/methodology*

Keywords

Nested model, validation, design studies, visualization

1. INTRODUCTION

In 2009, Munzner proposed the four-level nested design model as a framework for thinking about the design and validation of visualization systems at four cascading levels [23]. This model makes explicit the negative impact of poor design decisions early on in a project and stresses the importance of choosing appropriate validation techniques at each level. The nested model has provided guidance, motivation, framing, and ammunition for a broad range of visualization papers, including problem-driven design studies [7, 11, 25, 28, 30, 33], technique-driven work [19], evaluation [1, 32], models [8, 10, 18, 20, 29, 31, 36], and systems [4, 12].

We use the nested model extensively as a way to guide and reflect about our own work, and propose an extension of the model motivated by our desire to clarify the meaning of the term *guideline*. This term is loosely defined within the visualization literature to describe knowledge that guides how we make design decisions. One of our goals with this work is to clarify the meaning of this term for visualization research in order to assess the impact of both problem- and technique-driven work on guidelines.

The extension proposes *blocks* as a generic term for the outcomes of the design process at the three lower levels: abstractions, techniques, and algorithms. Concrete examples of blocks at each of these levels are that a network is a data abstraction block, a node-link diagram is a visual encoding block, and a specific force-directed layout approach such as GEM [13] is an algorithm block. We can then define *guidelines* as statements about the relationships between blocks, such as a node-link diagram is a good visual encoding of small graphs, or a specific force-directed layout algorithm is faster than another. We consider *guideline* and *characterization* to be synonyms.

Guidelines may pertain to blocks within a single level, and we call these *within-level* guidelines. Guidelines may also cross between levels; we call these *between-level* guidelines. Both types of guidelines often arise from reflection after evaluation and validation efforts. Within-level guidelines are the result of comparing one block to others at the same level, and often stem from validation efforts in papers that present a new technique or algorithm. Between-level guidelines provide guidance on how a block at one level is a match or mismatch with a block at an adjacent level. These guidelines typically emerge from the validation of design studies. Evaluation papers may result in either kind of guideline, between- or within-level.

The primary contribution of this paper is our proposed extension to the nested model and the resulting implications, presented in Section 2. Section 3 presents an analysis of open problems in our field illuminated by these extensions: the need to define the meaning of *block* at the problem level, to create mid-level task taxonomies at the abstraction level and possibly to refine the model itself at that level, and to establish a more complete set of guidelines that map up from the algorithm level to the technique level. Thus, a secondary contribution is the elucidation of these gaps in our collective knowledge and a call for action to close them.

BELIV 2012 Seattle, WA, USA

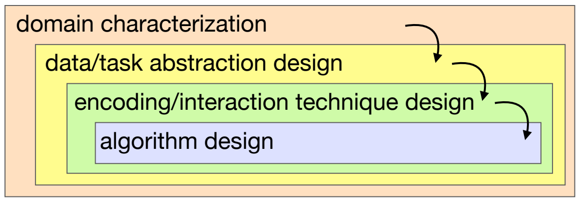

domain characterization
data/task abstraction design
encoding/interaction technique design
algorithm design

Figure 1: Original depiction of the four-level nested design model [23], with arrows indicating the cascading effects of decisions made at higher levels.

2. NESTED MODEL EXTENSION

The original description of the nested model [23] breaks down the design and evaluation of a visualization project into four nested *levels*, shown in Figure 1. The highest level is to characterize the domain and problem of interest; the next level is to design the data and task abstractions for that characterization; the third level is to design visual encodings and interaction techniques for those abstractions; and the lowest level is to design algorithms to implement those techniques programmatically. The focus of this original work is on the cascading implications of design decisions made at different levels, where the decisions made at one level become the assumptions at the next level. These implications are shown as arrows in Figure 1.

Although we find this model useful for structuring how we think about building and designing visualization systems, it falls short when we try to reason about both the wealth and dearth of knowledge that we have about design guidelines. In this paper we propose an extension to the original model that helps us do so.

2.1 Blocks and Guidelines

We extend the nested model with the ideas of blocks and guidelines, as illustrated in Figure 2. A **block** is the outcome of a design decision at a specific level: an algorithm, a visual encoding and/or interaction technique, a data abstraction, or a task abstraction. The term *block* allows us to refer to these different kinds of outcomes in a generic way that can be used for any level. Figure 2 shows these blocks as individual shapes within the levels.

Examples of blocks at the algorithm level are different algorithms for direct volume rendering: ray casting [21], splatting [39], and texture mapping [6]. At the technique level, examples of blocks for visually representing text are phrase nets [34], wordles [35], and word trees [38]. At the abstraction level, blocks include the tasks of finding outliers and trends [2]; the dataset types of tables, networks, and text; and the attribute types of categorical, ordered, and quantitative [24].

We chose the term *blocks* as an allusion to the experience of playing with building blocks. The builder is guided in choosing one particular block out of the bin of options by noticing that some blocks fit together nicely while others do not. By combining individual blocks together, the builder is able create more complex structures.

A **guideline** is a statement about the relationships between blocks. Guidelines help designers make choices about which blocks are appropriate versus which blocks are a mismatch with their requirements. **Within-level guidelines** are for choices within a particular level, such as selecting the fastest algorithm from several contenders within the algorithm level. **Between-level guidelines** are

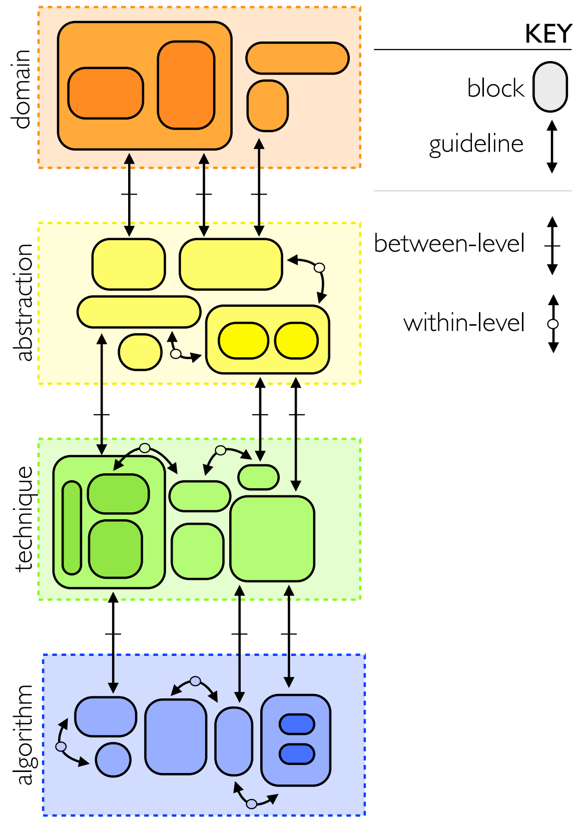


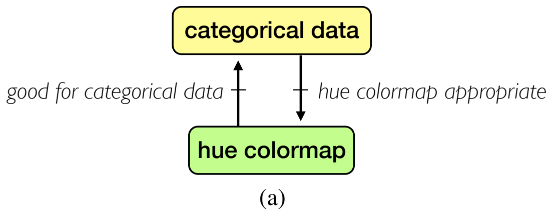
Figure 2: The extended nested model explicitly includes *blocks* that are the outcomes of the design process within a level, represented by individual shapes within each level, and *guidelines* for making choices between these blocks, represented by arrows. Between-level guidelines map blocks at adjacent levels, and within-level guidelines compare blocks within a level. The faded depiction at the domain problem level implies a knowledge gap, discussed in more detail in Section 3.

for choices about how to move between levels, such as selecting which visual encoding technique to use for a particular data and task abstraction.

The arrows in Figure 2 represent guidelines that connect individual blocks, in contrast to the arrows in Figure 1 that represent dependencies and go between entire levels. For visual clarity, the extension depiction orders the levels vertically rather than explicitly nesting them.

We consider within-level guidelines as directly comparing one block to another. For example, a within-level guideline at the visual encoding level is to choose—for reasons of avoiding visual clutter—node-link diagrams when visualizing small networks and matrix diagrams when visualizing large ones [14]. An example of a within-level guideline at the algorithm level is to choose the newer Voronoi treemap algorithm of Nocaj and Brandes [27] over the original algorithm of Balzer and Deussen [5] because it is independent of display resolution and faster to compute.

Movement from one level to another is guided by between-level guidelines. These guidelines map blocks at one level to those at an adjacent level. Figure 3(a) shows a well-established guideline



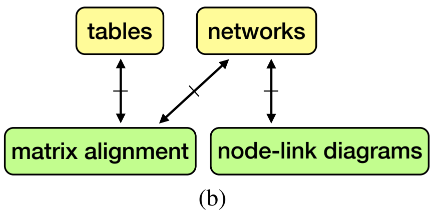


Figure 3: a) Example of a simple pairwise between-level guideline that can be inverted to go upwards or downwards. b) Guidelines can also be many-to-one or many-to-many; these more complex guidelines are not trivially invertible. The example shows a many-to-many between-level guideline.

between hue colormaps and categorical data [37]. In the literature, sometimes the term *characterization* is used to describe moving upward from a lower to a higher level, and the term *guideline* for moving downward from higher to lower—we consider these concepts to simply be two sides of the same coin. The simple upward characterization "hue-based colormaps are appropriate for categorical data" can be trivially restated as the downward guideline "if your data is categorical, then hue-based colormaps are appropriate". We propose guidelines as a more generic word to describe any stated relationship between blocks.

Although Figures 2 and 3(a) illustrate one-to-one pairwise guidelines for visual simplicity, these guidelines can be many-to-one or many-to-many. Most of these more complex guidelines are not trivially invertible, such as the between-level guideline example shown in Figure 3(b): matrix alignment is a block at the visual encoding technique level that is suitable for multiple blocks at the abstraction level, tables and networks. The technique-level block of node-link diagrams, however, does not match up with the abstraction-level table block.

A particular visualization system can be decomposed into a stack of blocks, with one or a small set of blocks chosen at each of the different levels. Guidelines help a designer make these choices, and also help an evaluator analyze whether a designer's choices are reasonable. Figure 4 illustrates this idea. Figure 4(a) shows two different simple visualization system designs, where different choices of blocks were made at each level. In keeping with the original nested model, the choice of a blocks on a higher level constrains the suitable choices of blocks at a lower level. Figure 4(b) shows a more representative real-world example, where the domain problem is decomposed into multiple subtasks and thus the full system includes multiple stacks of blocks.

In summary, guidelines encapsulate a relationship between two or more blocks, where within-level guidelines compare blocks within a single level and between-level guidelines map blocks at adjacent levels. Without this extension to the model, we had difficulty in reasoning about guidelines; these terms allow us to express these

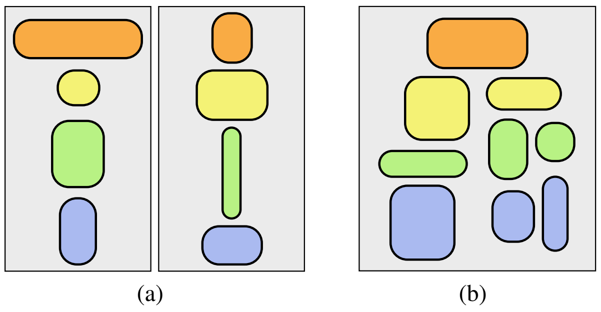


Figure 4: Constructing visualization systems with blocks. a) A designer stacks one or more blocks at each level, making choices based on guidelines; the two simple examples show that the choices at higher levels constrain the choices beneath. b) A real-world visualization system is likely to include multiple stacks of blocks.

ideas crisply.

2.2 Within and Between Levels

One goal of visualization evaluation is to provide *actionability*, which is guidance for a particular audience to take action based on research results [15]. Guidelines are one such form of actionability for visualization designers, resulting from validation in both technique-driven work and design studies.

Within-level guidelines often arise from validation efforts when a new block is the main research contribution, at either the technique or algorithm design level. Between-level guidelines are often the result of validation in design studies, where the emphasis is on justifying choices from the set of existing blocks by showing that blocks match up properly with each other across levels. Guidelines that map between levels are thus a central concern in design studies, and most existing design study papers do indeed emphasize them.

Both kinds of guidelines may arise from evaluation papers. For example, Heer, Kong, and Agrawala provide within-level guidelines on how to choose between horizon graphs and bar charts based on the available screen space [17]. These guidelines are based on reflections from extensive empirical evaluation of different visual encoding techniques across a range of resolutions. In another evaluation paper, Heer and Bostock provide between-level guidelines on how to choose visual encoding techniques appropriate for different abstract data types [16], following in the footsteps of Cleveland and McGill [9]. These between-level guidelines are also based on empirical evaluation.

An interesting question is whether within-level guidelines are even a meaningful idea; one could argue that all within-level comparisons at lower levels of the model rely on assumptions made at higher levels, and thus the ostensibly within-level guidelines are in fact heavily influenced by between-level guidelines that map blocks down to these lower levels. For example, testing the run-time speed of a new algorithm might appear to be a within-level comparison with other algorithm blocks. That comparison, however, relies on choices made for blocks at the data and task abstraction level. What *are* these assumptions about data and task blocks? Are these the right set of blocks to be testing with? If these blocks are changed, would the test results be significantly different?

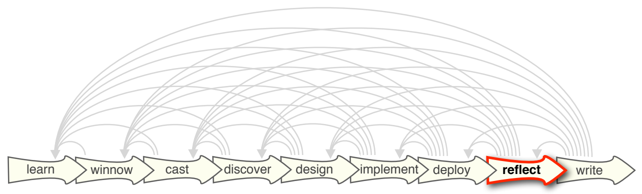


Figure 5: The nine-stage framework for conducting design studies [31]. The eighth stage is to reflect on lessons learned by confirming, refining, rejecting, or proposing guidelines. The extended nested model clarifies the scope of this stage.

We maintain, however, that the distinction between within-level and between-level is a useful way to reason about guidelines. Echoing points made in the original nested model paper, we argue that research progress in the field as a whole will be accelerated by explicitly documenting the assumptions made at higher levels when working at lower levels. That is, even technique-driven work with validation focused on within-level comparison would be strengthened by explicitly considering and documenting the assumptions about blocks at higher levels.

2.3 Guidelines and Reflection

The nested model is a *design model* that describes different levels of design that are inherent to, and should be considered in, the creation of a visualization tool. Recently, we proposed a *process model* for conducting design studies, called the nine-stage framework [31].

Figure 5 shows this framework, with the penultimate *reflect* stage highlighted. Our thinking about the meaning of *guidelines* was sparked by our claim that at this stage a visualization researcher should reflect on how lessons learned relate to the larger research area of visualization by confirming, refining, rejecting, or proposing new guidelines. The extension of the nested model proposed here clarifies that this reflection may involve both between- and within-level guidelines. Reflection can, and should, include guidance for the use of individual blocks at all levels of the nested model.

3. GAPS IN KNOWLEDGE

Extending the nested model with the concepts of blocks and guidelines clarifies why certain parts of the design process are particularly difficult by exposing gaps in our visualization knowledge. These gaps present rich opportunities for future work.

3.1 Problem Characterization Blocks

We have not provided any examples of blocks at the outermost level of domain problem characterization, and Figure 2 depicts this level as less clear than the others. This omission is because we do not yet have an answer to the question: what exactly *is* a block at this level? Is it simply a problem domain, like bioinformatics or finance? Is it a set of specific, real-world questions that people in a problem domain have about a real dataset? For example, the MizBee design study [22] has a table of specific questions asked by researchers in comparative genomics, questions such as "What are the differences between individual nucleotides of feature pairs?" and "What is the density of coverage and where are the gaps across a chromosome?". Are these problem blocks outcomes of design decisions, or outcomes of some other sort? Are they fundamentally different than the blocks at lower levels?

These open questions persist when we further consider between-level guidelines that map from problem blocks to abstraction blocks—is there such a thing as a problem or domain guideline? The assumption underlying the abstraction level in the nested model is that all possible problems in different domains will map to a smaller set of task and data abstractions. Might we be able to establish the complete set of guidelines from the problems in a given domain to these abstractions, or will there always be more to find? These questions are central to our understanding of how to conduct design studies because they necessitate a careful selection of abstraction blocks that map to the target domain problem.

3.2 Abstraction Level

Considering the four levels in terms of their constituent blocks uncovers the dearth of identified blocks at the abstraction level. We have many blocks for visual encodings and interaction techniques, and perhaps even more blocks at the algorithm level, but we have far fewer established blocks for data and task abstractions. Without blocks we cannot have guidelines; blocks at both levels are a precondition for guidelines that map between them. We believe that this lack of both blocks and between-level guidelines at least partially accounts for the difficulty of going from a problem characterization to an abstraction when conducting a design study.

In particular, our knowledge of tasks is deeply incomplete. The well-cited work of Amar, Eagan, and Stasko [2] describes a taxonomy of low-level tasks such as *retrieve value*, *filter*, and *sort*, and was intended to provide a foundation for the development of richer task taxonomies at higher levels. Amar and Stasko [3] also presented a taxonomy of very high-level tasks such as *expose uncertainty*, *formulate cause/effect*, and *confirm hypothesis*. We lack task blocks, however, at the middle level. There is a pressing need to propose and validate mid-level task taxonomies that bridge the gap between finding the minimum value of an attribute and confirming or refuting a hypothesis.

The abstraction level itself could also benefit from more study. Considering the level in terms of its blocks exposes some shortcomings of the current model. Are task blocks and data blocks sufficiently different that the abstraction level should be further broken down into two separate levels? Might a further breakdown into original data versus derived data be useful? Currently the model treats data generically, not differentiating between data provided as input from the problem domain, and data derived as part of the design process. In our experience these two types of data appear at separate points of the design process, with input data giving rise to an initial task abstraction, which informs the design of derived data, which then often causes the task abstraction to change, and so on. In retrospect, however, the distinction between input data and derived data is often unclear, and there is not a crisp line between them. We identify the influence of data abstraction changes on task instability as an important aspect of design studies in our nine-stage process framework [31]. Would a finer-grained model at the abstraction level be useful in capturing and explaining this instability through further refinement of the nested model?

3.3 Algorithm Mappings

Establishing guidelines from algorithms up to techniques is sometimes straightforward, but other times remarkably subtle. In many cases the literature does not concisely describe the result of a specific algorithm in terms of a visual encoding technique. For example, different algorithms in the general family of force-directed network layout can give rise to quite different visual encodings in terms of the spatial proximity of classes of nodes to each other. Very few authors explicitly discuss these characteristics; Noack is a notable exception [26]. Fully characterizing the mappings up from specific algorithms to the visual encodings that they produce remains an important open problem. Until this goal is attained, problem-driven visualization designers will have a difficult time in making wise desicions about which algorithms to use.

4. CONCLUSIONS

We present an extension of the four-level nested model that identifies blocks within each level that are specific outcomes of the design process, allowing the definition of a guideline as a expressing a relationship between blocks that can hold either within a single level or between two adjacent levels. Differentiating between within- and between-level guidelines allows for a more explicit discussion of open problems in the literature, including questions about the role of validation.

5. ACKNOWLEDGMENTS

This work was funded in part by an NSERC Strategic Grant and a Microsoft Research Faculty Fellowship.

6. REFERENCES

[1] Y. ah Kang and J. Stasko. Characterizing the intelligence analysis process: Informing visual analytics design through a longitudinal field study. In *IEEE Conf. Visual Analytics Science and Technology (VAST)*, pages 21–30, 2011.

[2] R. Amar, J. Eagan, and J. Stasko. Low-level components of analytic activity in information visualization. In *Proc. IEEE Symposium on Information Visualization (InfoVis)*, pages 111–117, 2005.

[3] R. Amar and J. Stasko. A knowledge task-based framework for the design and evaluation of information visualizations. In *Proc. IEEE Symposium on Information Visualization (InfoVis)*, pages 143–149, 2004.

[4] D. Auber, D. Archambault, R. B. A. Lambert, M. Mathiaut, P. Mary, M. Delest, J. Dubois, and G. Melançon. The tulip 3 framework: A scalable software library for information visualization applications based on relational data. Technical report, INRIA Research Report 7860, 2012.

[5] M. Balzer and O. Deussen. Voronoi treemaps. In *IEEE Symp. Information Visualization (InfoVis)*, pages 49–56, 2005.

[6] B. Cabral, N. Cam, and J. Foran. Accelerated volume rendering and tomographic reconstruction using texture mapping hardware. In *Proc. Symp. Volume Visualization (VolVis)*, pages 91–98, 1994.

[7] C.-Y. Chiu and A. D. Russell. Design of a construction management data visualization environment: A top-down approach. *Automation in Construction*, 20(4):399–417, 2011.

[8] J. Chuang, D. Ramage, C. D. Manning, and J. Heer. Interpretation and Trust: Designing Model-Driven Visualizations for Text Analysis. In *Proc. ACM Conf. Human Factors in Computing Systems (CHI)*, pages 443–452, 2012.

[9] W. S. Cleveland and R. McGill. Graphical perception: Theory, experimentation, and application to the development of graphical methods. *Journal of the American Statistical Association (JASA)*, 79(387):531–554, 1984.

[10] S. Conversy, S. Chatty, and C. Hurter. Visual scanning as a reference framework for interactive representation design. *Information Visualization*, 10(3):196–211, 2011.

[11] I. Dillingham, B. Mills, and J. Dykes. Exploring road incident data with heat maps. In *Conf. Geographic Information Science Research UK (GISRUK)*, 2011.

[12] A. G. Forbes, T. Höllerer, and G. Legrady. behaviorism: A framework for dynamic data visualization. *IEEE Trans. Visualization and Computer Graphics (Proc. InfoVis 2010)*, 16(6):1164–1171, 2010.

[13] A. Frick, A. Ludwig, and H. Mehldau. A fast adaptive layout algorithm for undirected graphs. In *Proc. Graph Drawing (GD'94)*, volume 894 of *LNCS*, pages 388–403. Springer, 1995.

[14] M. Ghoniem, J.-D. Fekete, and P. Castagliola. On the readability of graphs using node-link and matrix-based representations: a controlled experiment an d statistical analysis. *Information Visualization*, 4(2):114–135, 2005.

[15] M. Gleicher. Position paper: Why ask why? considering motivation in visualization evaluation. In *Proc. VisWeek Wkshp. BEyond time and errors: novel evaLuation methods for Information Visualization (BELIV)*. ACM, 2012.

[16] J. Heer and M. Bostock. Crowdsourcing graphical perception: using mechanical turk to assess visualization design. In *Proc. ACM Human Factors in Computing Systems (CHI)*, pages 203–212, 2010.

[17] J. Heer, N. Kong, and M. Agrawala. Sizing the horizon: The effects of chart size and layering on the graphical perception of time series visualizations. In *Proc. ACM Human Factors in Computing Systems (CHI)*, pages 1303–1312, 2009.

[18] J. Hullman and N. Diakopoulos. Visualization rhetoric: Framing effects in narrative visualization. *IEEE Trans. Visualization and Computer Graphics (Proc. InfoVis 2011)*, 17(12):2231–2240, 2011.

[19] P. Kok, M. Baiker, E. A. Hendriks, F. H. Post, J. Dijkstra, C. W. Löwik, B. P. Lelieveldt, and C. P. Botha. Articulated planar reformation for change visualization in small animal imaging. *IEEE Trans. Visualization and Computer Graphics (Proc. Vis 2010)*, 16(6):1396–1404, 2010.

[20] H. Lam, E. Bertini, P. Isenberg, C. Plaisant, and S. Carpendale. Empirical studies in information visualization: Seven scenarios. *IEEE Trans. Visualization and Computer Graphics (TVCG)*, 18(9):1520–1536, 2012.

[21] M. Levoy. Display of surfaces from volume data. *IEEE Comput. Graph. Appl.*, 8(3):29–37, May 1988.

[22] M. Meyer, T. Munzner, and H. Pfister. MizBee: A multiscale synteny browser. *IEEE Trans. Visualization and Computer Graphics (Proc. InfoVis 2009)*, 15(6):897–904, 2009.

[23] T. Munzner. A Nested Model for Visualization Design and Validation. *IEEE Trans. Visualization and Computer Graphics (Proc. InfoVis 2009)*, 15(6):921–928, 2009.

[24] T. Munzner. Visualization. In *Fundamentals of Graphics*, pages 675–707. AK Peters, 3rd edition, 2009.

[25] T. Munzner, A. Barsky, and M. Williams. Reflections on QuestVis: A visualization system for an environmental sustainability model. In *Scientific Visualization: Interactions, Features, Metaphors*, volume 2, pages 240–259. Dagstuhl Follow-Ups, 2011.

[26] A. Noack. An energy model for visual graph clustering. In *Proc. Graph Drawing (GD'03)*, volume 2912 of *LNCS*, pages 425–436. Springer-Verlag, 2003.

[27] A. Nocaj and U. Brandes. Computing voronoi treemaps: Faster, simpler, and resolution-independent. *Computer Graphics Forum*, 31(3pt1):855–864, 2012.

[28] H. Piringer, W. Berger, and J. Krasser. HyperMoVal: interactive visual validation of regression models for

real-time simulation. *Computer Graphics Forum (Proc. EuroVis 2010)*, 29(3):983–992, 2010.

[29] M. Sedlmair, P. Isenberg, D. Baur, and A. Butz. Information Visualization Evaluation in Large Companies: Challenges, Experiences and Recommendations. *Information Visualization*, 10(3):248–266, 2011.

[30] M. Sedlmair, P. Isenberg, D. Baur, M. Mauerer, C. Pigorsch, and A. Butz. Cardiogram: Visual Analytics for Automotive Engineers. In *Proc. ACM Conf. Human Factors in Computing Systems (CHI)*, pages 1727–1736, 2011.

[31] M. Sedlmair, M. Meyer, and T. Munzner. Design Study Methodology: Reflections from the Trenches and the Stacks. *IEEE Trans. Visualization and Computer Graphics (Proc. InfoVis 2012)*, 18(12):2431–2440, 2012.

[32] D. Sprague and M. Tory. Exploring how and why people use visualizations in casual contexts: Modeling user goals and regulated motivations. *Information Visualization*, 11(2):106–123, 2012.

[33] A. S. Spritzer and C. M. D. S. Freitas. Design and evaluation of magnetviz – a graph visualization tool. *IEEE Trans. Visualization and Computer Graphics (TVCG)*, 18(5):822–835, 2012.

[34] F. van Ham, M. Wattenberg, and F. B. Viegas. Mapping text with phrase nets. *IEEE Transactions on Visualization and Computer Graphics*, 15:1169–1176, 2009.

[35] F. B. Viegas, M. Wattenberg, and J. Feinberg. Participatory visualization with wordle. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1137–1144, Nov. 2009.

[36] X. Wang, W. Dou, T. Butkiewicz, E. A. Bier, and W. Ribarsky. A Two-stage Framework for Designing Visual Analytics System in Organizational Environments. In *IEEE Conf. Visual Analytics Science and Technology (VAST)*, pages 249–258, 2011.

[37] C. Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann/Academic Press, 2nd edition, 2004.

[38] M. Wattenberg and F. B. Viegas. The word tree, an interactive visual concordance. *IEEE Trans. Visualization and Computer Graphics (Proc. InfoVis 2008)*, 14(6):1221–1228, 2008.

[39] L. A. Westover. Splatting: A parallel, feed-forward volume rendering algorithm. Technical report, University of North Carolina at Chapel Hill, 1991.