Visualization of Streaming Data: Observing Change and Context in Information Visualization Techniques

Miloš Krstajić, Daniel A. Keim
University of Konstanz
Konstanz, Germany
{*milos.krstajic,daniel.keim*}*@uni-konstanz.de*

***Abstract*—Visualizing data streams poses numerous challenges in the data, image and user space. In the era of big data, we need incremental visualization methods that will allow the analysts to explore data faster and help them make important decisions on time. In this paper, we have reviewed several well-known information visualization methods that are commonly used to visualize static datasets and analyzed their degrees of freedom. By observing which independent visual variables can change in each method, we described how these changes are related to the attribute and structure changes that can occur in the data stream. Most of the changes in the data stream lead to potential loss of temporal and relational context between the new data and the past data. We present potential directions for measuring the amount of change and loss of context by reviewing related work and identify open issues for future work in this domain.**

***Keywords*-streaming visualization, data streams, incremental visualization, dynamic data**

I. Introduction

Visualization of data streams in the era of big data has an important role. The streams are currently being created everywhere - from personal logs, where people track their travels over the world or training routines, to large network and sensor infrastructures, from financial transactions to social media text streams and this will be an ongoing trend in the years to come. Efforts in databases and data management communities on how to efficiently transfer and store all that data have been joined in the last years by the efforts in data mining community on how to deal with the automated analysis algorithms on such a large scale. What we are seeing now are the challenges that arise in the domain of data science and visual analytics - how to visualize, explore and make sense of all these vast amounts of data. Therefore, visualizations that can help the human to efficiently analyze the part of the current and the past data are of great importance.

Visualization of streaming data is strongly related to its temporal context and very often methods that map time to the horizontal axis are used to visualize the data stream. How do we define which part of past data is relevant for the current data and the current point in time? Although, the data being generated and delivered in the streams has a strong temporal component, in many cases it is not only the temporal component that the analysts are interested in. There are other important data dimensions that are equally important and time might be just an additional aspect that they care about. In those cases, we might want to rely on other visualization methods that can show other attributes better than temporal visualizations.

How should the visualization change when we add new data? Does the whole layout have to be recomputed when we add just one element like in force-directed graphs, or we can easily add it like in scatterplots? But what if the new attribute value is out of current minimum/maximum ranges? Can i identify what is new and where did my old data go?

In this paper, we have explored the degrees of freedom of several well-known visualization techniques and tried to identify how these changes relate to the attribute and structural changes that can occur in a data stream. The most of the changes can lead to loss of context to the past data, from simple cases where the visualization is just minimally changed to significant changes where the whole layout has to be recomputed, each visual element moves on the screen and the previous order in each dimension is not preserved.

II. What can change?

Established methods in information visualization are typically used to visualize historical datasets. We have reviewed several well-known methods that are commonly used to visualize static datasets and analyzed the degrees of freedom for each method, i.e. we have looked at which independent visual variables can change. Even though these methods work with different data types, use different visual features and sometimes have completely orthogonal approaches to visualize data, they share some commonalities when used to displayed dynamic data. For each method, we have also discussed which changes in the data structure and data attributes (dimensions) can occur and differentiated between the *changes within range* and *changes out of range*. Our observations show that they all suffer to certain extent from the loss of context, with a degree that depends on the type of change that occurs in the data stream and few of them have issues that also exist when working with static datasets, such as overplotting and visual clutter. The summary of the results is given in the Table.

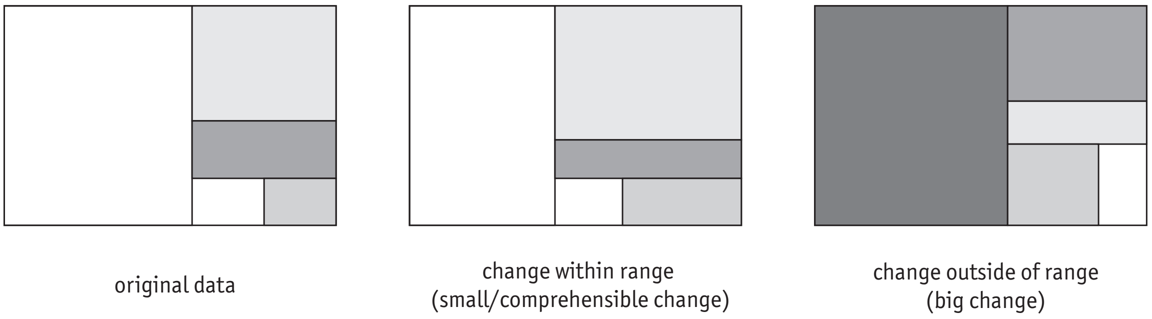

Figure 1. Changes in treemap. The changes in the data stream can lead to adjustments of the rectangle size, color and opacity and positioning. Although small changes might not lead to a significant loss of context (middle), bigger changes can lead to complete rearrangement of the layout, since the ranges and volume of the data stream is not predictable (right).

We have worked under following basic assumptions: first, the amount of new data is significantly smaller than the amount of the already processed data or it is just one new element. This way we avoid potential issues arising during the visualization initialization stages and minimize the complexity of the problem that would occur when the change between two time steps is bigger than the amount of the past data. Although some of the conclusions from our observations could also apply to big changes, we believe that it would be very difficult to describe and define such complex changes in the visualization. Therefore, we decided to concentrate on small incremental changes, which can, in some cases, already lead to significant loss of context. Our second assumption is that it is important to understand the relational and temporal context between the new and old objects. Otherwise, the visualization of streaming data would be visualization of each new increment separately, which is not different from visualizing a historical dataset.

A. Treemap

Treemap is a well-known space-filling technique used to visualize hierarchical data. Each node in the hierarchy is represented by a rectangle, whose area corresponds to (one of) the node's attribute(s). The fill color of the node can be used to show another data attribute. This technique can display both the hierarchy and node values without overlap and has been used extensively since its introduction in [1]. There exist several popular layout algorithms, such as squarified [2], pivot, slice-and-dice [1], spiral [3] or ordered treemap [4], which try to balance between aspect ratio, order and layout stability.

The obvious advantage of the technique to improve visibility of small items in a single layout is overshadowed by its instability when applied on streaming data and other design issues [5]. In many applications, such as monitoring stock market data, abrupt layout changes will lead to loss of relational and temporal context of the past.

1) Degrees of freedom: Treemap belongs to the class of *relative* visualization techniques in which the features of the visual objects used to depict data items *always* depend on the features of other data items. The basic visual object in a tree map is the rectangle, whose size is always proportional to the other rectangles and the fixed size of the visualization. Typically, the first dimension from the hierarchical dataset is mapped to the **area** of the rectangles and can change as the underlying dimension in the data stream change over time.

Next, the second dimension from the data is usually mapped to the color according to a predefined colormap. In a streaming data scenario, the data is potentially unbounded, which means that the colormap needs to be readjusted in every rectangle when only one value in the whole dataset is out of minimum/maximum range. An early attempt to map dynamic data changes in a treemap can be found in [6], where color is used to show high activity in the node. In this example, the rectangle areas do not change their fixed size and time is not mapped to any of the visualization primitives.

Treemap applied on streaming data would belong to the class of visualization techniques that provide *implicit* temporal context [7], which means that it could be used to represent new data in the context of the past data, although it would hardly be possible to reconstruct the history/evolution of the data. The color (or opacity) of the rectangle could be used to reflect recent changes by mapping the time of last change to it.

Finally, the aforementioned issues appear when the changes in the stream occur in the attribute space. Additional problems may arise when there are structural changes in the hierarchy, i.e. when the new nodes are added to the tree, the nodes are removed from the tree, or the nodes move in the hierarchy. This would lead to similar abrupt changes in the layout, which occur when the rectangles are resized.

The issues related to the changes that can occur in the

	What can change?	**Loss of context**
treemap	- Rectangles can change size/color/opacity - Rectangles could move in the hierarchy - Rectangles added/removed from the screen	- significant when size changes a lot - significant when rectangles move in the hierarchy - change of color (when values are out of range
scatterplot / map	- Ranges of the axes - Points added/removed from the screen - Existing points can change their properties (size, color, opacity, position)	- only when outside min/max range - minimal due to point order preservation - minimal because of the assumption: amount of change << past data - other problems: overplotting, clutter
streamgraph	- Ranges of the axes - Streams reordered - Streams added/removed from the screen	- significant due to stream reordering
line chart(s)	- Ranges of the axes - Lines added/removed from the screen	- only when outside min/max y-axis range - other significant problems: overplotting, clutter, low resolution
horizon chart(s)	- Ranges of the axes - Color ranges for new max/min values - Number of charts - Charts reordered	- change of color (when values are out of range - chart reordering
pixel-oriented (recursive pattern)	- Pixels added/removed from the screen - Color change - Dimension reordering - Recursion levels changed	- color change when values outside range - if dimension reordering needed (rare)
word cloud	- Word size, position - Words added, removed	- spiral: only for a big change - force-directed: significant due to layout recomputation

Figure 2. Which components of different visualization techniques change when new data is added? The table lists common information visualization techniques, the visualization properties that can change in each technique and summarizes the loss of context that occurs when new data arrives in the data stream

streams of hierarchical data and major issues related to the loss of relational and temporal context are summarized in Figure 1 and the table shown in Figure 2.

B. Scatterplot / Map

Scatterplot is typically used to visualize multidimensional data in order to find correlations between dimensions and detect visual clusters. Two data attributes are mapped to Cartesian coordinates, creating a point, whose visual features (size, color, shape...) can be mapped to other data attributes.

1) Degrees of freedom: Scatterplot belongs to the class of *absolute* visualization techniques, where each visual object is placed in the visualization independently of other objects.

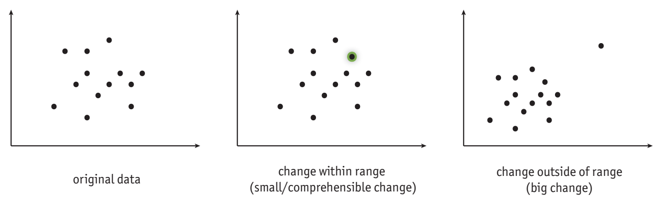

Figure 3. Changes in scatterplot. Adding a data point whose values are within the axes ranges (middle) causes a minimal change of the display, while adding a data point outside of the axes ranges requires the whole display to be adjusted (right).

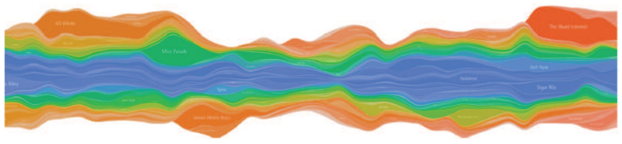

Figure 4. Streamgraph, also known as ThemeRiver or stacked graph. The optimal ordering of layers requires offline calculation, such that the streams with the highest "wiggles" are laid on the outside. Image taken from [8]

In a data streaming scenario, if a new data item arrives, whose main attributes are within current dimension ranges, it can be placed in the visualization without losing neither the temporal nor the relational context to past data. This is shown in Figure 3 (middle). Minimal loss of context might occur when the data attributes are out of range (Figure 3 (right), which can be also applied to changes in other data attributes (point size, color, opacity or shape). Although the loss of context might not necessarily be significant, this technique suffers from the problem of overplotting and clutter, which is a disadvantage that also exist when working with static dataset.

Scatterplot, similarly to treemap, belongs to the class of visualization techniques with implicit temporal context [7], and one possibility to encode time (age) of the visual elements on the screen would be to use color or opacity.

C. *Streamgraph*

Streamgraph (shown in Figure 4, also known as *stacked graph* or *ThemeRiver*) is a well-known technique that is used for visualizing data streams [8] [9] [10] [11]. It shows several "streams", i.e. variables that change their values over time and are layered on top of each other and symmetrically along the timeline. This is a good example of a popular technique where the term "stream" is used incorrectly from data streaming perspective, because of its strong limitations in terms of *streamability*, which we will discuss below.

1) Degrees of freedom: As with other visualization techniques used for time series data, the ranges of the axes can change. Since the new data is added to the right of the visualization, the viewport on the timeline (x-axis) can either rescale to keep all the past data, or just shift and remove the same period length on the left that has been added to the right. If the new data is out of range for the maximum values on the y-axis, the streamgraph would have to rescale vertically. These shifts and size changes do not introduce major losses of context and we treat them as negligible in our observation.

However, an important property of a "good" streamgraph is the optimal ordering of individual streams, i.e. the ordering that provides the best legibility of the visualization. The legibility criteria is discussed in [8]. In order to achieve good legibility, the visual objects (layers) with the least amount of change are positioned in the middle, while the most "bursty" layers are on the upper and lower side of the

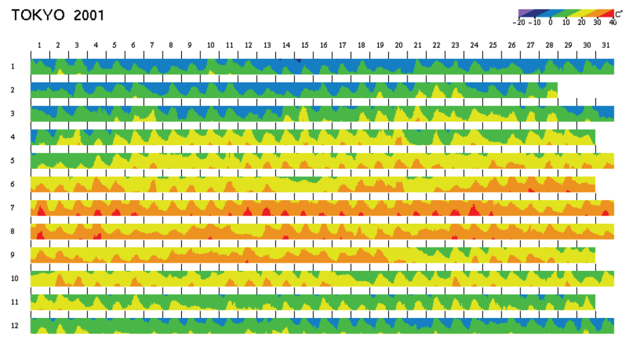

Figure 5. Horizon graph is very often used to visualize multiple time series. When the new data points are outside the predefined range, the colormap has to be readjusted, thus leading to the loss of context of the past data. Image taken from [12]

graph. This ensures minimum "wiggles" of the surrounding layers. Having a layout in which the positioning of each visual object is dependent on all the other objects, leads to the following issues when applied to the data stream: first, when new data comes in, layers have to be reordered, which requires recomputing of the whole visualization. This does not just increase the performance costs, but it can also cause confusion for the user, since the mental map of the visualization will not be preserved. Additionally, the unexpected bursts in each layer, which are characteristic for data streams, still might not result in a legible visualization even if reordering is applied. Second, if the layers are not reordered, the stability and optimal layout of the visualization is disrupted. In that case, the visualization depends on the initial layout that doesn't represent the current state of the underlying data.

Having so many undesired properties and serious limitations, the streamgraph can hardly be a good choice for streaming data. However, an exception could be made if the following criteria is fulfilled:

- the bursts in each layer are not too extreme and not too frequent,
- the number of layers is low,
- the past data is discarded or approximated using, for example, multi-resolution time windows,
- the time for user exploration between two updates is relatively long,
- there is enough time to inform the user about the new reordering (for example, using animation).

D. *Horizon graph(s)*

Horizon graph 5, also known as "two tone pseudo coloring" [12] is a visualization method very often recommended for displaying multiple time series due to its efficient use of space and color than other time-series techniques [13] [14]. Losing the context of the past data when working with data streams can occur in two cases. First, if the new values are

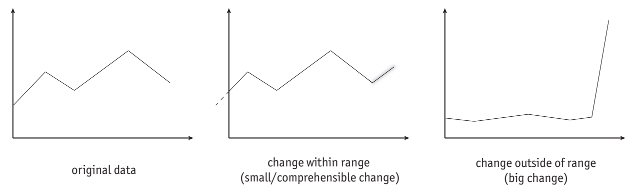

Figure 6. Changes in line chart. Adding new data to the current line requires readjustment of the display when the data is out of axes ranges, while adding new lines can cause overplotting and clutter.

outside of range, the whole color map has to be reapplied to the whole visualization again. Second, the order of the horizon charts can change. Although this is not the problem of the visualization itself, it is an issue since this method is usually used to visualize multiple time series and not just one.

E. Line chart(s)

When applied to streaming data, this low-resolution technique has the similar properties that were described for streaming graphs - rescaling the visualization when the new data is out of range, as shown in Figure 6. Since it uses the explicit notion of time, the loss of temporal context would be minimal. However the well known disadvantages from static datasets, such as overplotting and clutter, exist here as well when the new variables show up and new lines have to be added to the screen.

F. Pixel-oriented visualizations

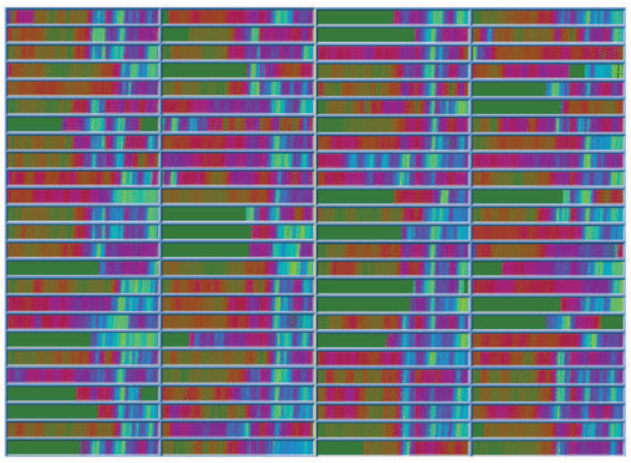

Figure 7. Pixel-oriented visualization: Recursive pattern technique, image taken from [15]

Pixel-based visualizations (Figure 7) are a set of well known techniques [16] for visualization of multidimensional data whose main idea is to use one pixel for one attribute value by assigning the value to the color of the pixel. The pixels are arranged according to different criteria, with recursive pattern being one of the most popular arrangements [15]. The pixels are arranged according to user-defined parameters for each recursion level.

Since the data value is mapped to color, all the pixels would have to be repainted if the new data value is out of maximum range for the chosen colormap. The ordering of dimensions might be another problem that could lead to the loss of context.

G. Word cloud

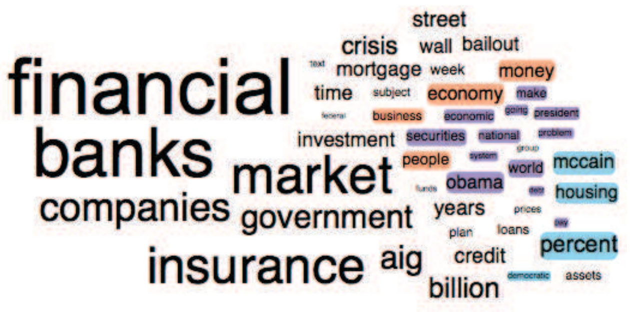

Figure 8. Word cloud visualization. The words are usually positioned without overlap using the force-directed method or spiral arrangement. Image taken from [17]

Word cloud, shown in Figure 8, has been extensively used to visualize text data [18] [19] [20] [17]. The layout algorithm tries to pack the words as tightly as possible, and two approaches are commonly used: force-directed and spiral. Applying the force-directed layout to streaming text data can lead to a significance loss of context, since the new words that appear in the stream would require the whole layout to be recomputed. On the other side, spiral layout, which places the word with the highest weight in the center and then arranges the rest on a spiral path, would lead to loss of context only if the change between the updates is big.

III. Change/Context Metrics and Criteria

In the previous section we have described which properties of different well-known visualization methods can change when working with data streams. As it can be seen in the Table 2, the majority of problems that occur are related to the loss of context, while few methods suffer from problems that already exist with large static datasets, such as overplotting or clutter in scatterplots and line charts. Our first step in identifying and describing these problems should be extended to quantifying the loss of context and defining other optimization, stability and aesthetic criteria for streaming data visualization.

One such metric could be similar to the layout distance change function proposed by Bederson and Shneiderman [4]. This metric tries to capture how much two treemap layouts differ on average by measuring the Euclidean distances between each pair of corresponding rectangles. This metric could easily be extended by taking into account variance distance change, as proposed in [3] to detect changes when few items move by large distances while the most items do not move at all.

Hao et al. [21] propose a similar metric, *constancy of display*, which ”counts the number of pixel changes per time unit” and *usage of display space*, which suggests that the empty space in the visualization should be avoided. However, they suggest that these might not be the best metrics and that other metrics should be developed. One proposal is to use the *pixel coherence*, which captures the neighbouring pixels that form a pattern and are percieved as an object. It has been suggested that measuring the data shifting to evaluate the usefulness of a visualization is strongly data dependent.

However, these metrics might not be appropriate for scatterplots and other techniques in which rescaling of axes occurs (see Figure 3). In such cases, the movement/change of the objects (points, lines, etc), on the screen might be significant, although it wouldn't necessary lead to the loss of context. This example shows how an abstract structural information can be easily formed when the data allows it and relates to the well known *mental map preservation* criterion.

In any case, these criteria have to be taken into account together with aesthetic criteria, which are often conflicting (as seen in the streamgraph example). Beck et al. define different aesthetic criteria for graphs in [22], which, beside mental map preservation, include reduction of cognitive load and minimization of temporal aliases that occur when a node is positioned after the update in the same place where another node was before the update. A survey on visualization of large graphs [23] reviews general, dynamic and scalability criteria for dynamic graphs and discusses two categories of visual display of time changes on graph elements: animation and static displays.

IV. Conclusion

In this paper, we have described the changes that can lead to loss of context in common visualization techniques when working with streaming data. We have addressed the relationship context between the new and old elements and temporal context that distinguishes the new and old elements. We have analyzed the usage scenarios under the assumption that the data is being updated one element at the time or *delta* is much smaller than the rest of the visualization. For each technique, we have differentiated between the updates that occur within the existing visualization structure and the updates with new elements in the visualization. We have identified possible research directions for measuring the loss of context.

Research in streaming data visualization is a new and challenging field, which tackles complex problems in data, visualization and user spaces. Future work should address the challenges that occur in transition stages between the updates, the propagation of streaming data through the information visualization pipeline and process times needed to get from the raw data to the visualization, the interplay between user interaction and automatic updates, but also the different types of updates (automatic and user triggered). Finally, an important research direction is how to deal with the visualization when the volume of the data is too big and data approximation occurs.

References

[1] B. Johnson and B. Shneiderman, “Tree-maps: A space-filling approach to the visualization of hierarchical information structures,” in *Visualization, 1991. Visualization'91, Proceedings., IEEE Conference on.* IEEE, 1991, pp. 284–291.

[2] M. Bruls, K. Huizing, and J. J. Van Wijk, “Squarified treemaps,” in *Data Visualization 2000.* Springer, 2000, pp. 33–42.

[3] Y. Tu and H.-W. Shen, “Visualizing changes of hierarchical data using treemaps,” *Visualization and Computer Graphics, IEEE Transactions on*, vol. 13, no. 6, pp. 1286–1293, 2007.

[4] B. B. Bederson, B. Shneiderman, and M. Wattenberg, “Ordered and quantum treemaps: Making effective use of 2d space to display hierarchies,” *AcM Transactions on Graphics (TOG)*, vol. 21, no. 4, pp. 833–854, 2002.

[5] D. Turo and B. Johnson, “Improving the visualization of hierarchies with treemaps: design issues and experimentation,” in *Visualization, 1992. Visualization'92, Proceedings., IEEE Conference on.* IEEE, 1992, pp. 124–131.

[6] G. Chin Jr, M. Singhal, G. Nakamura, V. Gurumoorthi, and N. Freeman-Cadoret, “Visual analysis of dynamic data streams,” *Information Visualization*, vol. 8, no. 3, pp. 212–229, 2009.

[7] C. Rohrdantz, D. Oelke, M. Krstajic, and F. Fischer, “Real-time visualization of streaming text data: Tasks and challenges,” in *Workshop on Interactive Visual Text Analytics for Decision-Making at the IEEE VisWeek*, vol. 201, 2011.

[8] L. Byron and M. Wattenberg, “Stacked graphs–geometry & aesthetics,” *Visualization and Computer Graphics, IEEE Transactions on*, vol. 14, no. 6, pp. 1245–1252, 2008.

[9] S. Havre, E. Hetzler, P. Whitney, and L. Nowell, “Themeriver: Visualizing thematic changes in large document collections,” *Visualization and Computer Graphics, IEEE Transactions on*, vol. 8, no. 1, pp. 9–20, 2002.

[10] M. Dork, D. Gruen, C. Williamson, and S. Carpendale, “A visual backchannel for large-scale events,” *Visualization and Computer Graphics, IEEE Transactions on*, vol. 16, no. 6, pp. 1129–1138, 2010.

[11] F. Wei, S. Liu, Y. Song, S. Pan, M. Zhou, W. Qian, L. Shi, L. Tan, and Q. Zhang, “Tiara: a visual exploratory text analytic system,” in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 2010, pp. 153–162.

[12] T. Saito, H. N. Miyamura, M. Yamamoto, H. Saito, Y. Hoshiya, and T. Kaseda, "Two-tone pseudo coloring: Compact visualization for one-dimensional data," in *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*. IEEE, 2005, pp. 173–180.

[13] 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 *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2009, pp. 1303–1312.

[14] W. Javed, B. McDonnel, and N. Elmqvist, "Graphical perception of multiple time series," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 16, no. 6, pp. 927–934, 2010.

[15] D. A. Keim, M. Ankerst, and H.-P. Kriegel, "Recursive pattern: A technique for visualizing very large amounts of data," in *Proceedings of the 6th conference on Visualization'95*. IEEE Computer Society, 1995, p. 279.

[16] D. A. Keim, "Designing pixel-oriented visualization techniques: Theory and applications," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 6, no. 1, pp. 59–78, 2000.

[17] W. Cui, Y. Wu, S. Liu, F. Wei, M. X. Zhou, and H. Qu, "Context preserving dynamic word cloud visualization," in *Pacific Visualization Symposium (PacificVis), 2010 IEEE*. IEEE, 2010, pp. 121–128.

[18] O. Kaser and D. Lemire, "Tag-cloud drawing: Algorithms for cloud visualization," *arXiv preprint cs/0703109*, 2007.

[19] C. Seifert, B. Kump, W. Kienreich, G. Granitzer, and M. Granitzer, "On the beauty and usability of tag clouds," in *Information Visualisation, 2008. IV'08. 12th International Conference*. IEEE, 2008, pp. 17–25.

[20] C. Collins, F. B. Viegas, and M. Wattenberg, "Parallel tag clouds to explore and analyze faceted text corpora," in *Visual Analytics Science and Technology, 2009. VAST 2009. IEEE Symposium on*. IEEE, 2009, pp. 91–98.

[21] M. Hao, D. Keim, U. Dayal, D. Oelke, and C. Tremblay, "Density displays for data stream monitoring," *Computer Graphics Forum*, vol. 27, no. 3, pp. 895–902, 2008.

[22] F. Beck, M. Burch, and S. Diehl, "Towards an aesthetic dimensions framework for dynamic graph visualisations," in *Information Visualisation, 2009 13th International Conference*. IEEE, 2009, pp. 592–597.

[23] T. Von Landesberger, A. Kuijper, T. Schreck, J. Kohlhammer, J. J. van Wijk, J.-D. Fekete, and D. W. Fellner, "Visual analysis of large graphs: State-of-the-art and future research challenges," in *Computer graphics forum*, vol. 30, no. 6. Wiley Online Library, 2011, pp. 1719–1749.