Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Projekt 2 Ausarbeitung

Truong Vinh Phan

Immersive Visualization of Multi-dimensional Data in a 3D Environment

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Contents

1 Introduction

1.1 3D visualization - next step for big data

As ever more aspects of our daily life become connected in the webbed environments of urban landscapes, the sheer amount of information that is generated and consumed collects into massive databases and is set to bypass the zettabyte threshold by the end of 2016 (Annual global IP traffic, Cisco VNI report [2]). With the big data explosion in full swing, data is being resourcefully aggregated across multiple industries for various purposes, from business intelligence to military and scientific applications. Growing proportionally with the immense volume of data is the inherent complexity and cumbersomeness, as well as the number of data dimensions [1], which magnifies the analysis task to a serious challenge, sometimes goes beyond today's capability.

Representing data in a 2D plane is still the predominant format of visualizations today. While most data can be well represented in tables and a variety of charts, the 2D environment limits the amount of data variables and properties that can be encoded, thus limit the ability of knowledge delivery and discovery. A TED Talk by D. McCandless from 2010 [5] showcased how aesthetics can add value to 2D data visualizations, but interactive 3D representation is the next step in this evolution, specifically in the big data landscape.

This paper aims to present the Stream Visualization, which is an approach to exploratory and interactive visualization in a 3D environment for chronological data. The basic concept is that chronological data can be represented in a stream-like flow form, and with the availability of a third dimension, more data variables can be encoded and meaningfully visualized.

2 The Stream Visualization

2.1 Introduction

The StreamViz is an approach to interactive and exploratory visualization which exploits new visualization paradigms, such as 3D graphics and animations, to allow for richer user experiences when visualizing time-based data that can be organized chronologically. Originally, the StreamViz is designed with online, commercial shop data in mind. In a large online shop system, there exists a large amount of meta data that can be all linked to sales data in many ways. As an example, ad campaigns or user click/conversion rate can be linked to sales performance data, or data from manufacturers can be linked to products in an order. The sheer amount of data can be overwhelming, and it is problematic to get the trends and relationships behind this data or derive actionable insights from it, even with traditional 2D visualizations and dashboards. The goal of the StreamViz is to try to solve this problem by providing the user with an interactive visualization that utilizes new paradigms, such as 3D graphics and animations to allow for a richer user experience in data exploration.

The ultimate goal of this approach would be to improve effectiveness in knowledge transfer and discovery process with regards to big, complex data. This approach should eventually prove to be more effective and deliver a richer experience than traditional 2D visualization approaches.

2.2 Initial design, requirements and challenges

The initial StreamViz was designed with the goal to visualize data from online commercial platforms. In the main use case, the user choose to visualize the sales data warehouse. This data warehouse has a star schema and aggregate sales data from many database tables, from order details to customer details and product details. The main requirement is that the user should be able to read out key facts, e.g. sales performance over a specified time period, from the visualized data. Because the data warehouse can be potentially huge, the data set is expected to be big that can scale up to millions of data rows. This presents a challenge to the usability of the visualization. An approach to this problem is to incorporate multiple views into the visualization that implements various levels of aggregation. See figure 6

2.2.1 Aggregated view

Aggregate all data points (e.g. purchase orders) in a specified timeline and visualize them. The x-axis shows the timeline with each tick can represent a month or a year in that time period. The y-axis's values show sales performance, represented by revenue. The Stream has a cylindrical form (a tube), and is segregated into different clickable areas according to the ticks on the timeline. Various filters can be applied to the visualization for better usability, including data type and data set filter. A filter for meta data (e.g. store location, manufacturer, etc.) can be useful to reduce clutter in the stream. Main use case in this view is that the user wants to view the details on the performance of a particular time window (e.g. a month), so the user clicks on the part of the stream corresponding to that time window to go into Detail view. See figure 7

2.2.2 Detail view

Detail view shows more details for data points in a selected time window. The corresponding part of the stream should be put into focus and the other parts grayed out. In this view, all individual data points are clickable, which will take the user into Context view. Statistical indicators like average, min, max can be overlaid and toggled. See figure 8

2.2.3 Context view

In the context mode, each individual data point (e.g. a purchase order progress) is shown together with all meta-data (context) associated with that data point. For example, relevant marketing data (e.g. Google Ad Words) can be linked to a purchase order based on date and content. Correspondence (e.g. emails) during purchase process can also be linked to the timeline, together with tracking data and/or customer feedback. Additional information pertaining to order items can be aggregated and shown based on user interaction (e.g. hovering). See figure 9

The user should also be able to add comments/notes throughout the visualization. Comments/notes can be stored using standard techniques, including in traditional SQL or No-SQL databases. See figure 10

3 Use Case: Visualizing the development of Refugee Camps over time with StreamViz

3.1 Introduction

Because real world sales databases are rather difficult to acquire, and the visualization can potentially become overly complex such that special knowledge in finance might be required for testers to evaluate and compare visualizations, which in such cases will eventually introduce bias into the end result, the StreamViz will be used to visualize the development of refugee camps in Germany (or possibly on a regional basis) for demonstration and study purposes. The data can be acquired through various sources. Dummy data can also be used to fill the data set in case real data is insufficient. The benefit of this approach is that the visualization is greatly simplified and no special prior knowledge is required, so that the study conducted later will remain as objective and unbiased as possible.

With the recent influx of immigrants and refugees into Germany, more and more camps are being built around cities to try to cope with the sudden surge of demand for accommodation. This visualization aims to show this process by visualizing the camp's capacity together with the actual occupation over time.

3.2 Design

To take into account for possibly big data sets, the visualization can be separated into different views and aggregation levels. The camps are put on a map according to their locations. The map will be varied based on different aggregation levels (e.g. country map with states, state map with cities or city map). The timeline is limited to a two-year period with each tick represents a month. The camp's planned capacity is represented by the outer cylinder stretching the timeline. The camp's actual occupation is represented by the inner cylinder. Colors are used to encode the ratio between planned capacity and actual occupation, as well as different camps.

3.2.1 Front view (2D)

The front view of the visualization is shown as a 2D map with various circles. The location of the circles on the map corresponds to the location of the camps. Each camp is represented by two nested circles, with the inner one shows the actual occupation and the outer one shows the planned capacity. Details is shown using labels on hovering over the camps. See figure 11

3.2.2 Side view (3D)

The camp's development stream is shown as 3D nested tubes with different radius denote the difference between planned capacity and actual occupation. More details and meta data can be shown around the 3D scene and as labels on hovering, e.g. legends, raw data... The timeline is shown at the bottom of the map in monthly-steps and should rotate as the user change view perspectives. The parts of the stream can be clicked on to show more detailed data about a particular camp at a particular time period.

3.2.3 Data point connection

There are two varieties for data point connection. The point-specific connection project the tube's radius onto the next plane consistently, i.e the tube size stays the same between two planes. Whereas with the flow-specific connection, the tube size on the other end will gradually increase or decrease to match the radius of the next cylinder on the next plane. The latter approach has the advantage of being more aesthetic and transfer immediate knowledge of the deviation in the camp size between two time units, thus slightly improve usability. The disadvantage being that it is very difficult to implement other feature like showing detail camp information for a specific time unit when the user clicks on the relevant tube.

3.3 Implementation

The visualization is implemented mainly in JavaScript using the ThreeJS framework. ThreeJS is a JavaScript framework that allows to build complex 3D scenes and objects and support

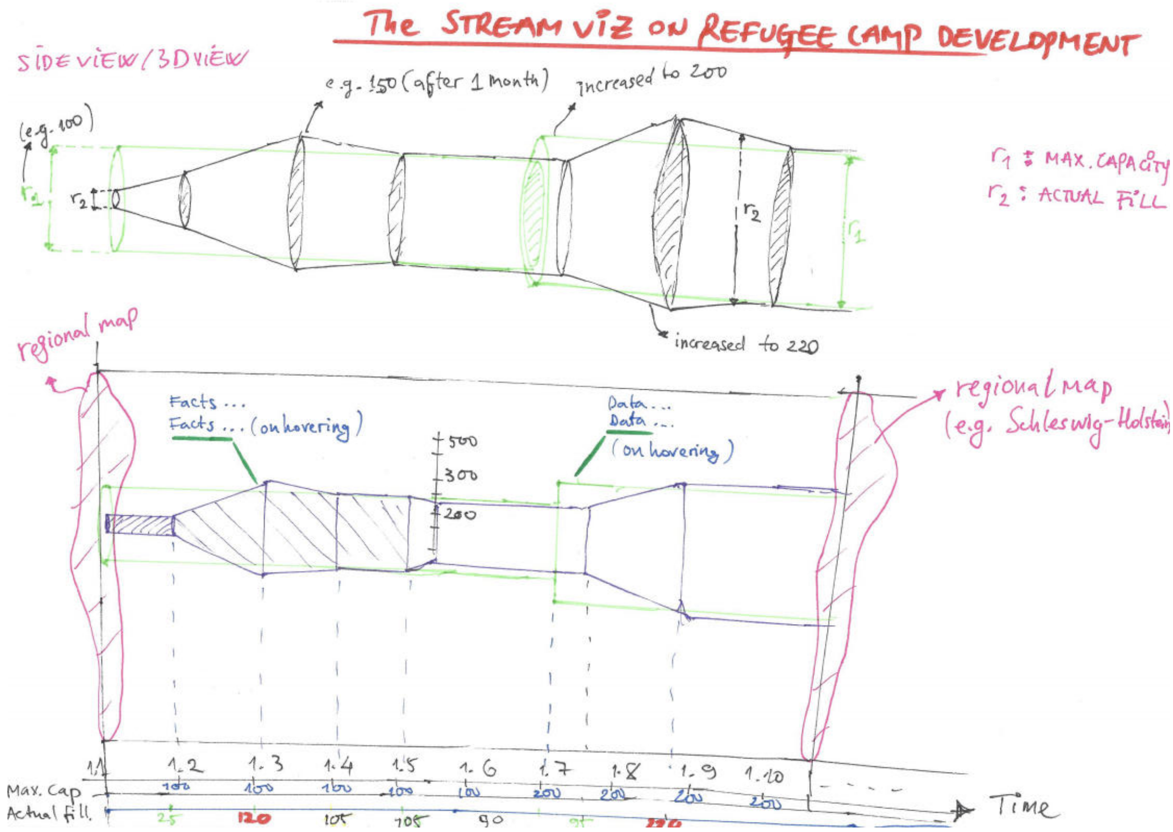

Figure 1: Refugee camp's development visualization - Side view

multiple controls. The rendering is done client-side in the browser using WebGL. Initially, the geographical coordinates (lat, long) of the administrative divisions (e.g. city, state, country, etc.) are queried against Google Map API for the map images. This approach is proved to be more difficult than using LeafletJS framework with OpenStreetMap API. Using this approach, the coordinate pair with a specific zoom level is sent to OpenStreetMap API via LeafletJS, then converted into a static image using Leaflet-Image library from MapBox. Because this process is time consuming, the result map image can be stored in a database or cached into the client's Local Storage for future retrieval, which will significantly reduce the overhead and thus the overall rendering time of the visualization.

The data is partially fictitious due to the difficulty in acquiring real data. Currently four data dimensions (variables) are encoded, using color, size, position and shape. The third dimension in a 3D environment allows for more variables to be encoded meaningfully, should more data is available. At the root of the data set is a JavaScript object named "data", which contains child objects for further administrative divisions. Each child object has a "coords" property to store the geographical and 3D coordinates. A camp object has a "timeline" property to store chronologically ordered data on the development of the camp over time.

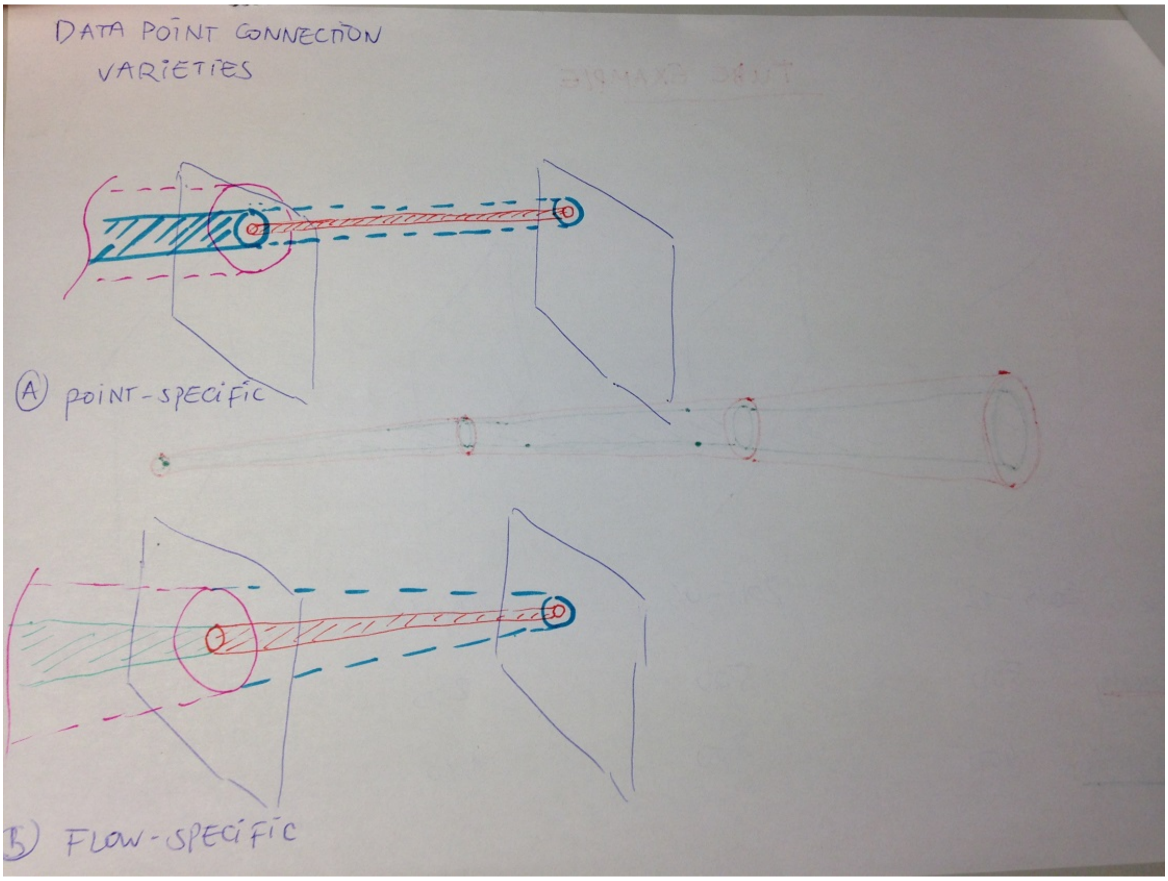

Figure 2: Data point connection varieties

For development purpose, the data is initially stored in a Google Spreadsheet and loaded into the visualization via Google Data API. Current approach stores the data as a JavaScript object inside a JavaScript file. This approach has several advantages over that using Google Spreadsheet, one of which is the speed improvement and also less effort is required later when moving the data set into a database (e.g. MongoDB).

The animation is implemented using the TweenMax/TimelineMax API from GreenSock, which provides functions to animate various properties of a JavaScript object.

Lists of camps and administrative divisions are provided for better navigation and improve usability.

3.4 Visualization challenges

There are various difficulties when working in a 3D environment. Unlike common 2D visualizations, the positioning and distribution of objects in a 3D-space pose a significant challenge.

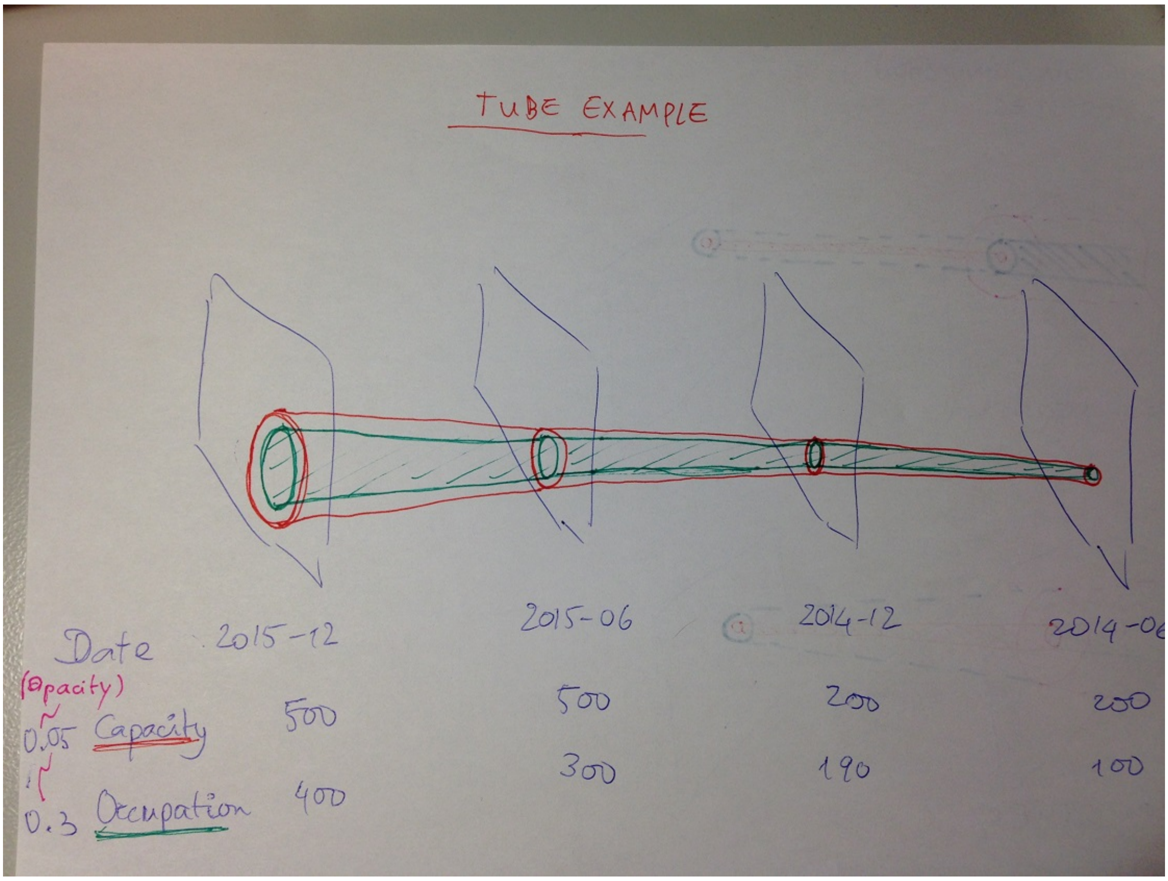

Figure 3: Tube example, Flow-specific

As in the case of refugee camp visualization, if the number of camps is large and there is no big difference in their geographical distribution then the visualization must be broken down into several views, otherwise camp overlapping issue will cause a negative impact on usability. For instance, a country view should only show aggregation (i.e. camps) for its next smaller administrative division (e.g. a state), a state should show aggregation for its cities, and so on. This approach also facilitates interactivity.

According to design, the 3D cylinders are nested into each other to emphasize the proportion between the planned capacity of a camp and the actual occupation. In order to maintain an acceptable level of usability, transparency must be employed to allow better view through these cylinders. However, overuse of transparency is known to have caused different problems, one of which is non-deterministic rendering, when being applied to multiple nested objects.

Another challenge is the placement of the camps geographically onto a 2D map. With the geographical coordinates (latitude, longitude) of the camps known, the translation into pixel-

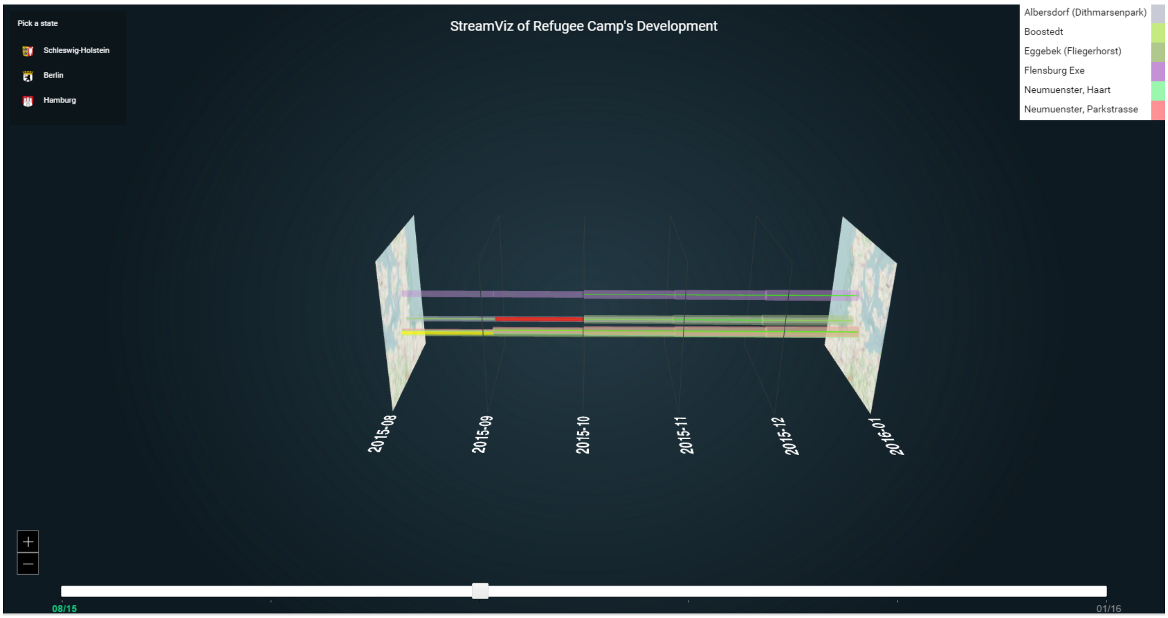

Figure 4: Stream visualization for refugee camp's development

coordinates for a 2D display can be done easily. However, since objects in a ThreeJS scene has a different scale of measurement, the translation from geographical coordinates to 3D unit requires a different approach. This approach requires the geographical bounds of the map to be known then a transformation factor is calculated as the ratio between the geographical coordinates and the 3D coordinates.

The actual capacity and occupation of a camp must also be mapped to the radius of the cylinder because the real number could go up to as high as tens of thousands. The scaling parameters must be correctly determined so that the proportions between the mapping and mapped objects are retained and reflected accurately in the 3D scene.

There are also a variety usability considerations, one of which is whether to allow the user to freely explore the 3D space with the mouse using the OrbitControl, or limit that ability and instead provide to the user a fixed set of on-screen controls, e.g. tilt, pan, zoom, left/right rotation, etc. The latter option would allow for a more streamlined experience, whereas the former option allows for a more natural and immersive experience. If the user is allowed to change the camera view freely to explore the 3D scene, all objects in the visualization should also adapt to the camera change to retain all important information in the user's view, in a manner so that key information is not distorted. This proves to be far more complex in a 3D-scene.

To make navigation easier, a slider is provided to help the user move along the visualization's timeline. Performance is also a potential problem when the timeline or the number of objects in a 3D scene grows too large. The visualization has been tested with a thousand-month

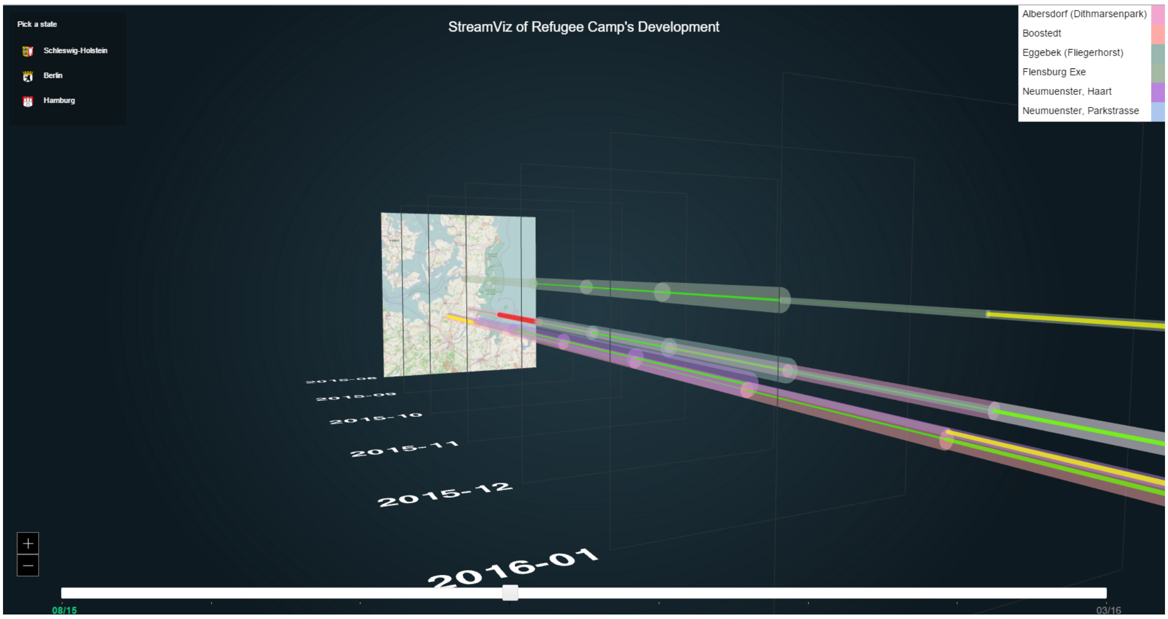

Figure 5: Stream visualization for refugee camp's development

timeline and under ThreeJS's WebGL rendering, the user would experience some minor lags while moving along the timeline, using the latest version of Chrome/Firefox browser on a mid-range machine.

Animation helps provide better user experience, although it is significantly more complex to implement in a 3D environment and may impact usability negatively if overused.

To keep the visualization dynamic, the map of the administrative divisions are queried from OpenStreetMap using their geographical coordinates and a fixed zoom level, then transformed into a static image. These map images are not quite user-friendly and also contain a lot of unnecessary details, which could affect usability negatively. An approach to this problem is to alter the generated map images using software tools like MapBox, which will generate more overhead. Other possible overhead come from the dynamic query of the camp's geographical coordinates based on their addresses.

4 Future work

Data such as social network data (e.g. Facebook posts, Twitter tweets, etc.) can also be visualized with the Stream Visualization as they can be ordered chronologically. Furthermore, real time visualization can be implemented with libraries like SocketIO, so that when the data set changes, the changes will be reflected in the visualization in real time.

Traditional visual representations like bar charts, pie charts, etc. can also be incorporated into the Stream Visualization in 3D form to allow for even more comprehensive visualizations, with the ability to encode more data and variables. Further, more complex user interactions can also be implemented.

5 Conclusion

Visualizing data in a 3D environment requires more effort and calculations than traditional 2D visualizations. Important features of the Stream Visualization's approach are the use of tri-dimensional techniques as visualization of a big data set may involve large amount of evolving information, and multiple views in order to provide different visions of complementary or overlapping information. Traditional visual representations such as bar charts can be adapted to a 3D environment to augment the exploration and knowledge discovery process. The work presented here is focused on visualization. Future work is needed to add more debugging functionalities, user interactions, animations and possibly real-time capability to the Stream Visualization. A first step in this direction could be to prepare a database/data warehouse system to house the evolving data set. Possible drawbacks of 3D approach include the complexity and rendering overheads, which is the subject for further usability studies.

Appendix A: Figures

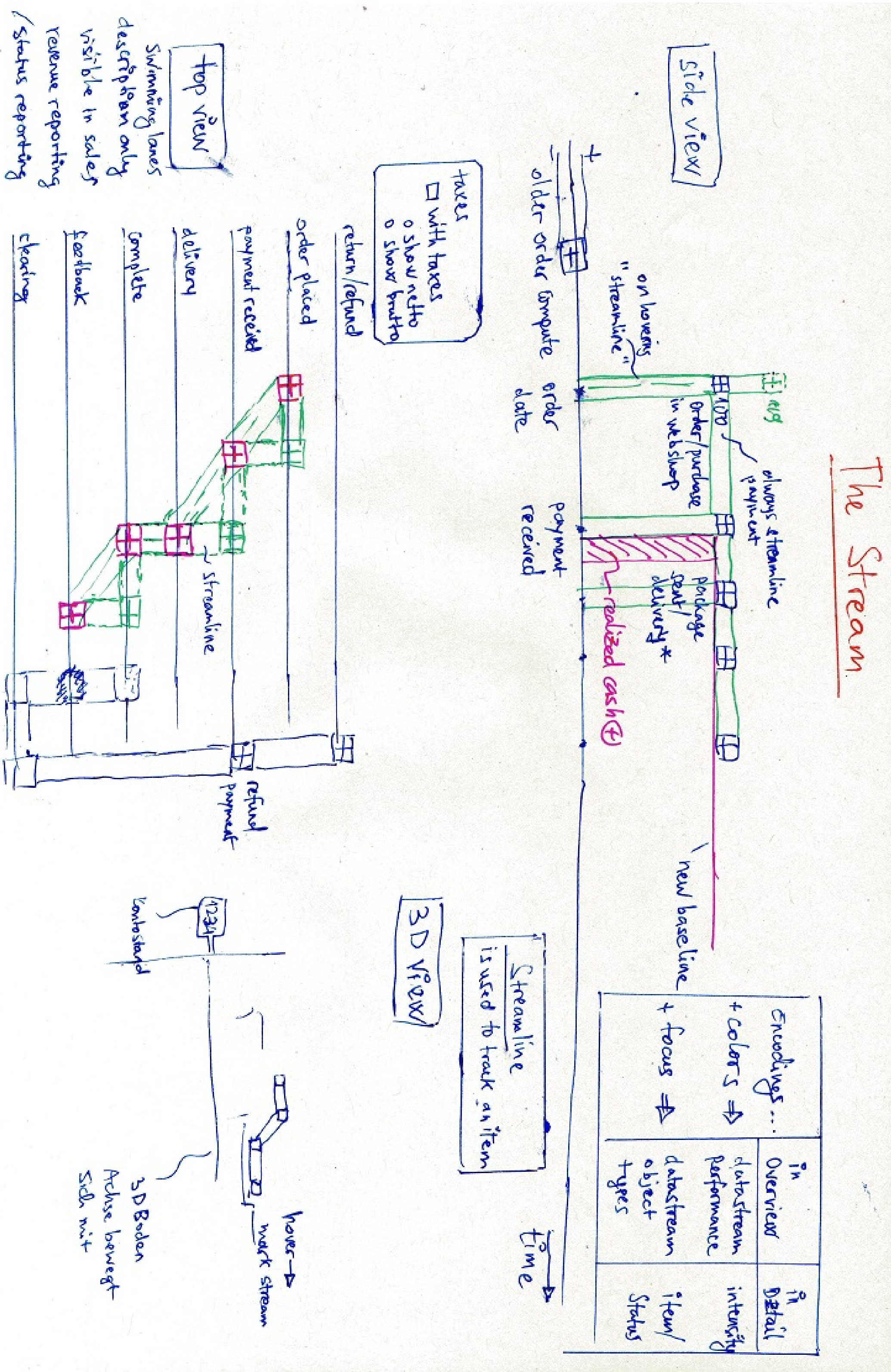

Figure 6: The StreamViz - initial mockup

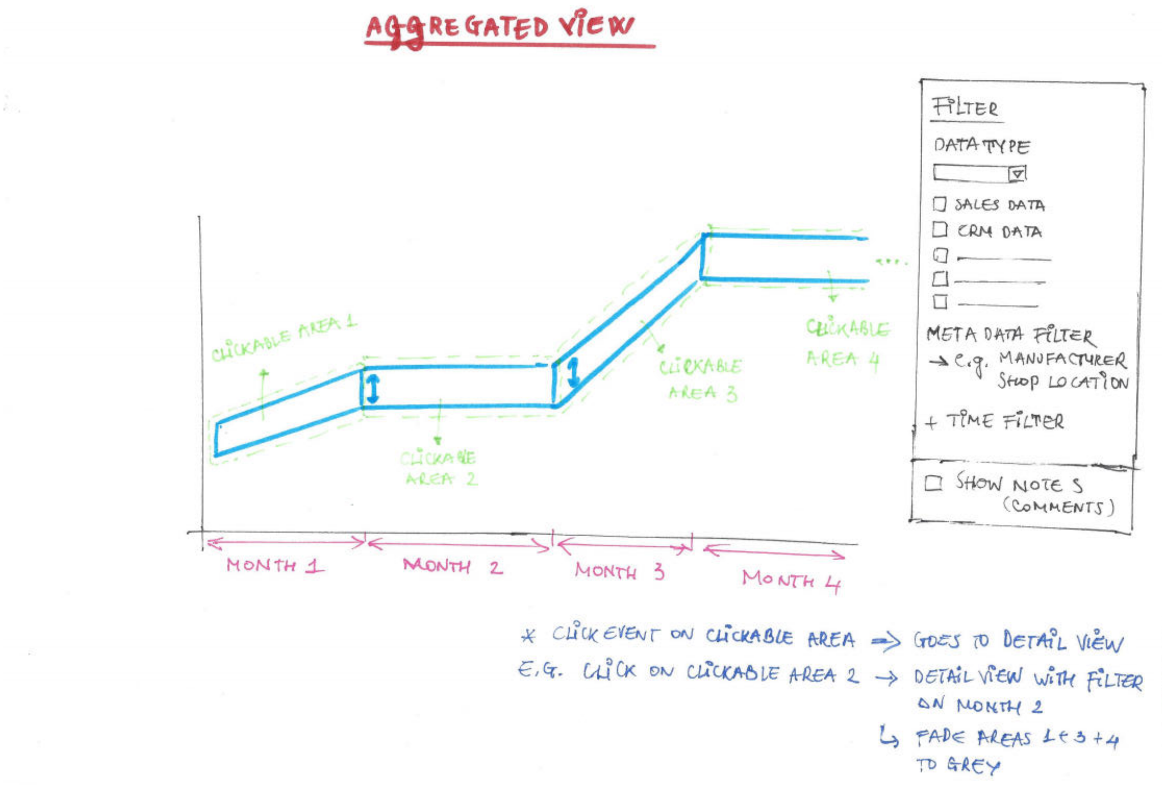

Figure 7: The StreamViz - Aggregated view

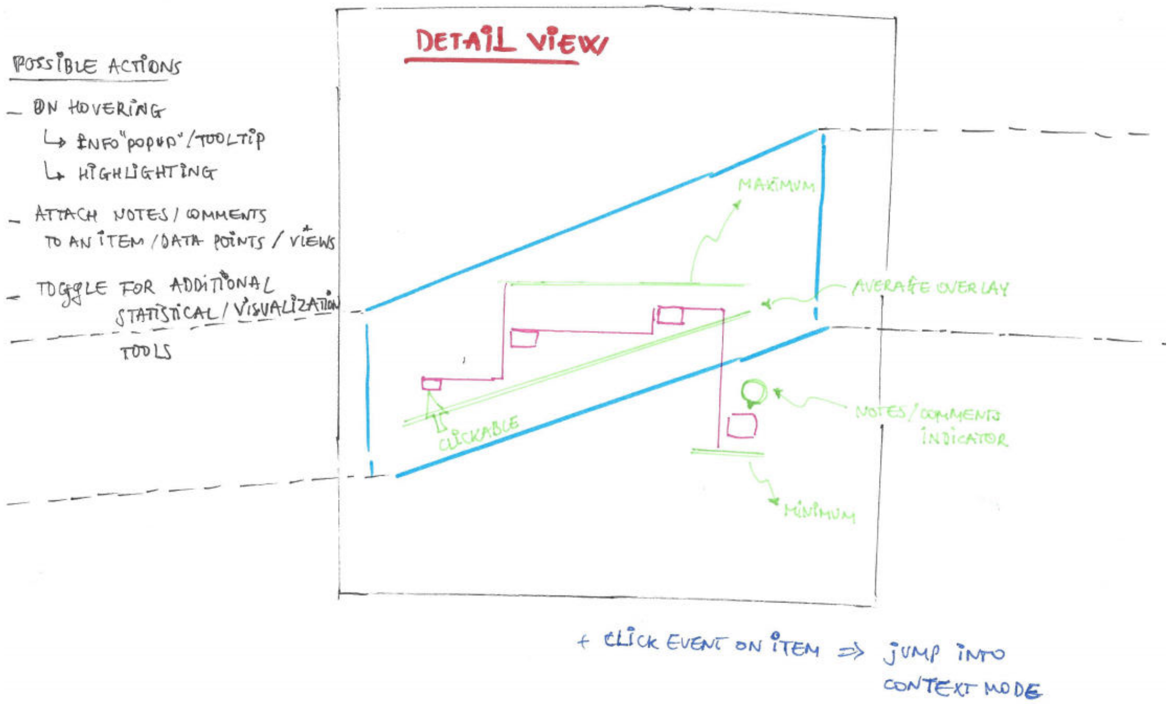

Figure 8: The StreamViz - Detail view

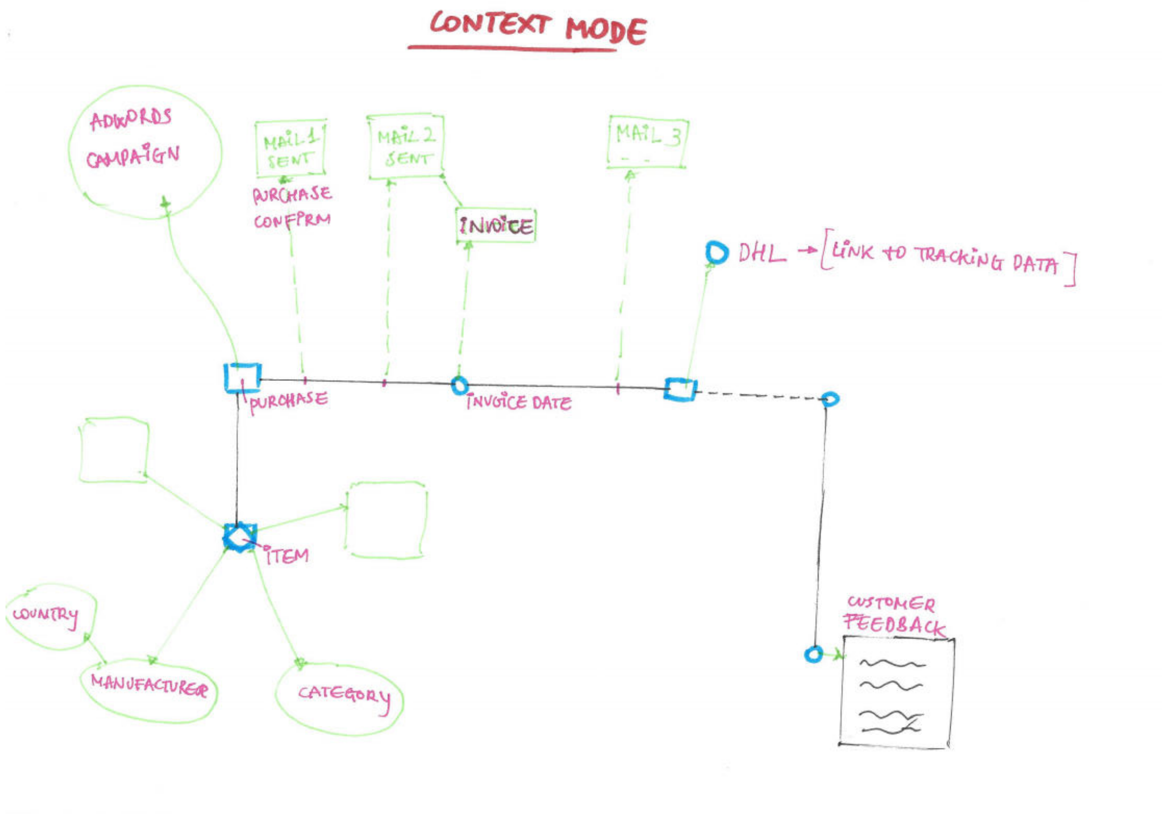

Figure 9: The StreamViz - Context view

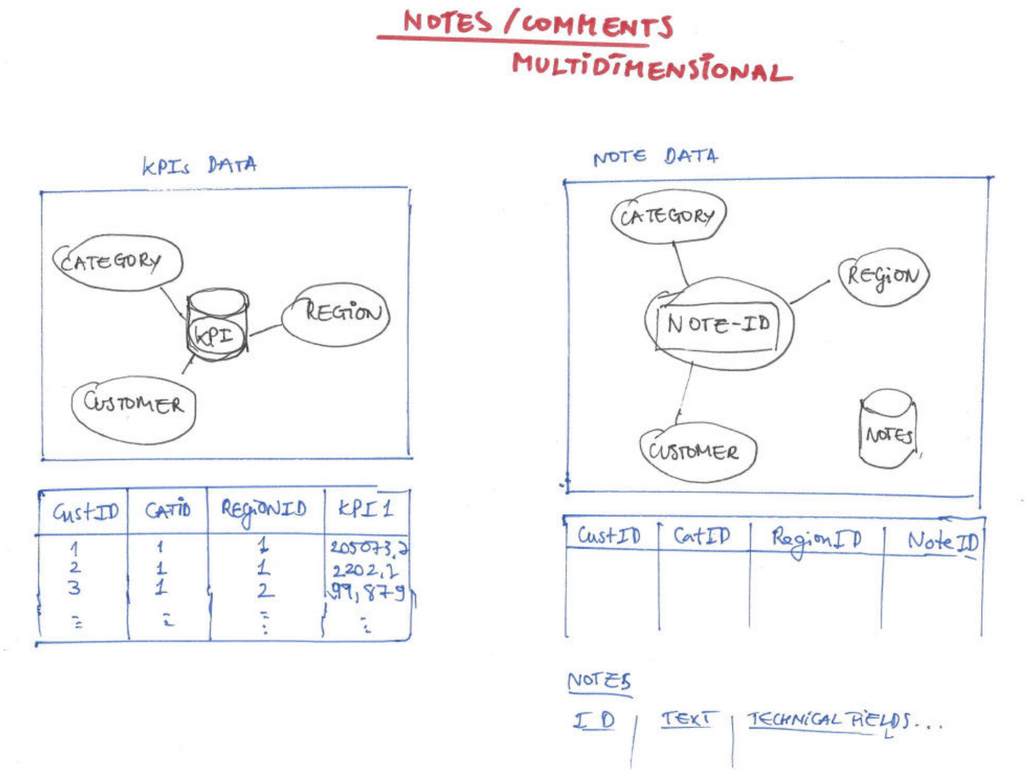

Figure 10: The StreamViz - Structure for comments/notes

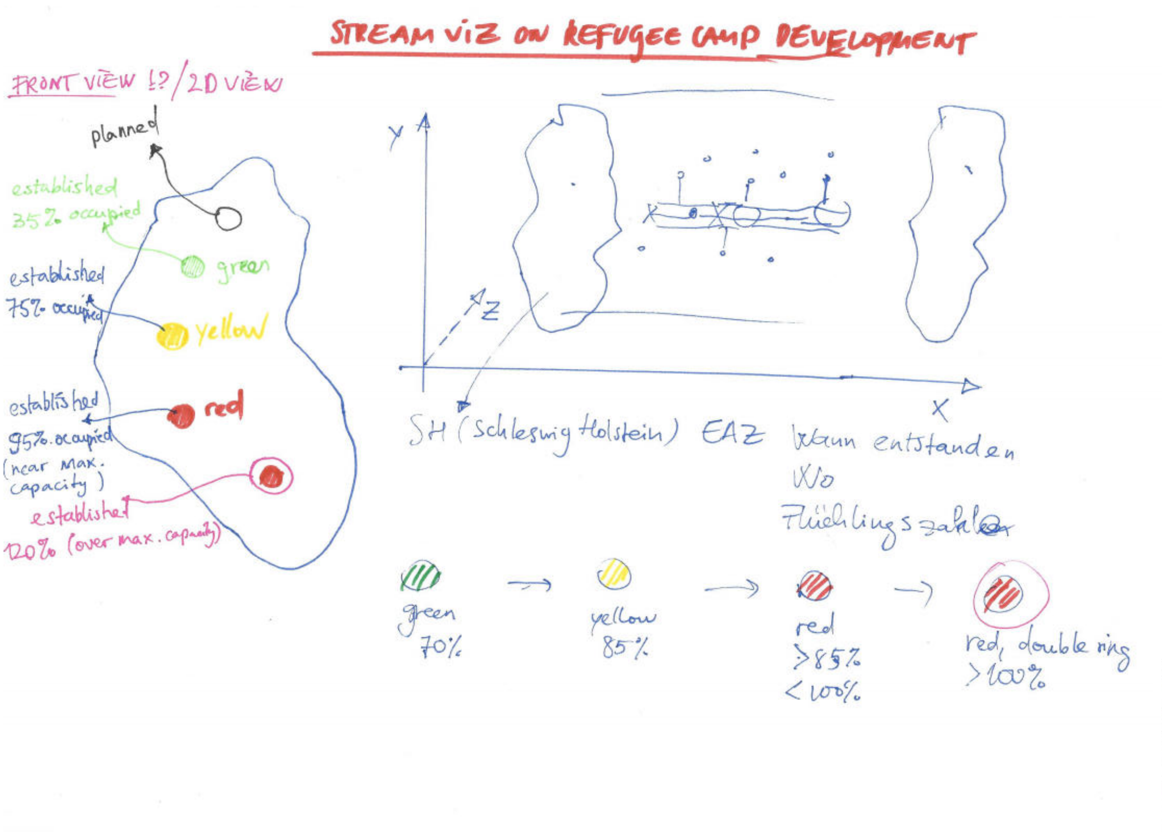

Figure 11: Refugee camp's development visualization - Front view

References

[1] CHAVEZ, David: Is 3D Visualization the Next Step for Big Data? In: *IEEE Computer Society of Silicon Valley* (2014)

[2] CISCO: The Zettabyte Era: Trends and Analysis. In: *White paper* (2015), May

[3] COUPAYE, T. ; RONCANCIO, C. L. ; BRULEY, C. ; LARRAMONA, J.: 3D Visualization of Rule Processing in Active Databases. In: *NPIV '97 Proceedings of the 1997 workshop on New paradigms in information visualization and manipulation* (1997)

[4] LI, W. ; EICKHOFF, C. ; VRIES, A. P. de: Interactive Summarization of Social Media. In: *IIiX '14 Proceedings of the 5th Information Interaction in Context Symposium* (2014)

[5] MCCANDLESS, David: TED: The Beauty of Data Visualization. . – URL http://www.ted.com/talks/david_mccandless_the_beauty_of_data_visualization?language=en

[6] PRAPAITRAKUL, N. ; PHITHAKKITNUKOON, S.: EQviz: a Visualization Tool for Monitoring World Earthquakes. In: *UbiComp/ISWC'15 Adjunct Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers* (2015)

[7] STASKO, John: Value-driven Evaluation of Visualizations. In: *BELIV '14 Proceedings of the Fifth Workshop on Beyond Time and Errors: Novel Evaluation Methods for Visualization* (2014)

[8] VORBROECKER, M. ; SAALFELD, P.: Einfluss ausgewählter Visualisierungen auf die Effizienz der Ablesbarkeit statistischer Informationen. . – URL http://wwwiti.cs.uni-magdeburg.de/iti_db/lehre/emcs/2012/projekte/MathiasPatrick.pdf

[9] WETTEL, Richard ; LANZA, Michele: CodeCity: 3D Visualization of large-scale Software. In: *Proceeding ICSE Companion '08 Companion of the 30th international conference on Software engineering* (2008)