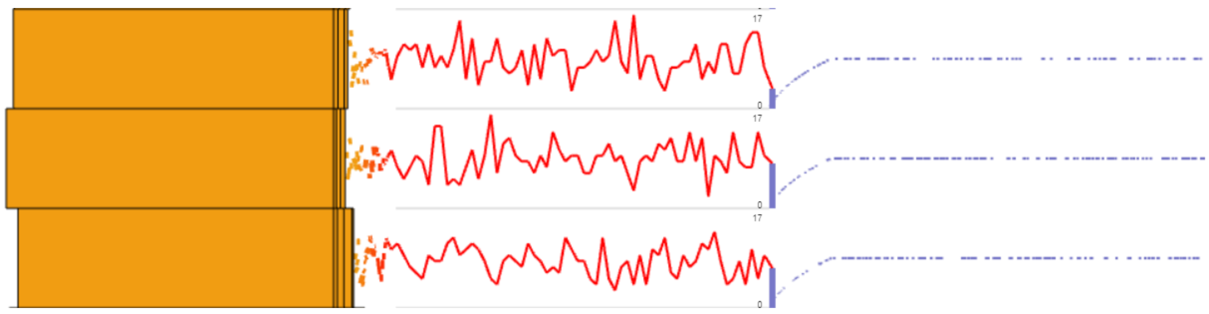
TÉCNICO LISBOA



OrderWarp

Visualizing ordinal data in Big Data Streaming

Henrique Lourenço do Espírito Santo Rosa Ferreira

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisor: Prof. Daniel Jorge Viegas Gonçalves

Examination Committee

October 2021

This work was created using $\LaTeX$ typesetting language
in the Overleaf environment (www.overleaf.com).

Acknowledgments

For their support and guidance through the development of my dissertation, I would like to thank my professors and colleagues in VisBig, with a special regard to Prof. Daniel Gonçalves and Prof. João Moreira. Their availability and feedback throughout the work made it possible, teaching me a lot in the process.

I want to thank my family who accompanied me from the beginning of my studies until today, and I am sure they will continue to do so in the future. A special thanks to my mother, father and brother, whose love and support shaped who I am today.

To all my friends, those who already knew me or become friends in these complicated 5 years, I would like to express my deepest gratitude for your help and all the memories created in the process.

This work was partially supported by FCT through project PTDC/CCI-CIF/28939/2017.

Abstract

The creation of platforms that support Big Data and Streaming domains is nowadays an important topic, since the amount of created and stored data at any moment in time increased substantially with the development of new information technologies. However, finding the best visualization techniques which work in these contexts is a complex challenge. The excessive amount of data forces the visualization's system to find the best data aggregation techniques to explicitly display information, and that at the same time does not impact the overall performance of the system. Furthermore, these techniques cannot be products of pre-processing algorithms, because the data in a Streaming visualization is received during its runtime. To deal with this issue, there is a concept called Graceful degradation, a technique that depicts information with different levels of aggregation and detail for different time periods.

In this work, we present OrderWarp, a system that displays ordinal Big Data applying the above-mentioned Graceful degradation, while using WebGL technologies for performance enhancement. The system's visualization uses different task-oriented idioms for ordinal data depiction, accompanied by animated transitions to represent changes in aggregations between periods. Using a binning strategy, the aggregation method ensures the visualization is able to always represent the whole dataset, and to run indefinitely. The study confirms the performance boost in Big Data Streaming visualizations, resulting of the system's architecture. It also finds the most effective idioms to represent ordinal data in these domains, as well as the best transitions between said idioms, and suggestions for designing these visualizations.

Keywords

Big Data, Streaming, Ordinal Data, Graceful Degradation, Information visualizations, Idioms and Transitions

Resumo

Hoje em dia são cada vez mais necessárias plataformas que suportem *Big Data* e *Data Streaming*, isto porque a quantidade de dados criados e armazenados a qualquer momento aumentou substancialmente com o desenvolvimento de nova tecnologias de informação. Encontrar técnicas de visualização que funcionem nestes contextos apresenta-se como um desafio complexo. A elevada quantidade de registos obriga as visualizações a utilizar técnicas de agregação de dados para explicitamente representar informação, e que, em simultâneo, não impactem a performance do sistema. Estas técnicas não podem ser produtos de algoritmos de pré-processamento, visto que os dados são apenas recebidos durante a execução da mesma. Para resolver este problema, apareceu o conceito de Degradação graciosa, uma técnica que retrata a informação com diferentes níveis de agregação para diferentes períodos de tempo.

Neste trabalho, apresentamos o OrderWarp, um sistema que exibe dados ordinais em Big Data e em tempo real, aplicando a Degradação graciosa anteriormente referenciada, e utilizando tecnologias WebGL para melhoramento da sua performance. A visualização do sistema usa idiomas centrados em tarefas diferentes para representação de dados ordinais, acompanhados por transições animadas que representam as mudanças nas agregações entre períodos. Utilizando uma estratégia de *binning*, o método de agregação assegura que a visualização é capaz de representar todo o *dataset*, e correr indefinidamente. O estudo confirma ainda o melhoramento da performance em visualizações de Big Data Streaming, resultantes da arquitetura do sistema. Encontra ainda, os idiomas que mais eficazmente representam dados ordinais nestes domínios, assim como as melhores transições entre idiomas, e oferece sugestões para o design destas visualizações.

Palavras Chave

Big Data, Informação em tempo-real, Dados ordinais, Degradação graciosa, Visualização de Informação, Idiomas e Transições;

Contents

List of Figures

List of Tables

Acronyms

LoD	Levels of detail
IoT	Internet of things
KDE	Kernel density estimation
t-SNE	t-Distributed Stochastic Neighbor Embedding
PCA	Principal component analysis
D3.js	Data Driven Documents
SVG	Scalable Vector Graphics
DOM	Document object model
XML	Extensible markup language
HTTP	Hypertext Transfer Protocol
fps	frames-per-second
RAM	Random-Acess Memory
WSS	WebSocket Secure
SSL	Secure Sockets Layer
TLS	Transport Layer Security
IQR	interquartile range
XML	Extensible Markup Language

1 Introduction

Contents

Information analysis is extremely important for data recognition and decision-making in multiple areas of scientific study, playing an important role in their development. The idea behind this field of study, is to create visual representations that simplify the process of discovering insightful information in a dataset, through visualizations. The advancement of information technologies originated an increasing amount of accessible and valuable data, created and stored by more and more entities. Nowadays, everyone can easily generate some sort of analyzable and storable data, such as a location or a user rating, through the most varied technologies, such as personal computers, smartphones or Internet of things (IoT) devices.

When the dimensions of these datasets computationally defy its representation, it is safe to say that the dataset belongs to the **Big Data** domain, even if there is no defined limit to when a dataset is considered within the domain. Big Data is usually characterized by the 5V's [8]: High data Volume; high data creation Velocity; big data Variety; data Veracity, or in other words, information's integrity; and Valuable and informative data.

Understanding information present in a dataset within the Big Data domain, can be an excessively time-consuming task or even impossible. The dataset's dimension, therefore, affects the effectiveness of the visualization, since it should not exclude information. If so, the visualization does not respect the expressiveness principle, which dictates that a visualization should show all data and nothing but the data [9]. This means that Big Data visualizations are obliged to [U+FB01]nd adequate aggregation techniques. If the aggregation is too small, the visualization is forced to consume additional time drawing more data representative elements, affecting the system's computational performance, or the overdrawing of elements will make the visualization illegible, in both cases the user's analysis is compromised. Oppositely, if the aggregation is too large, the visualization might lose the ability to present useful information. The aggregation level is therefore a tradeoff between the intended detail and the visualization's performance.

The ease of producing data also means that at every moment new data is being generated and plenty of systems experience an increase in traffic. This makes the analysis of the information in real-time one of the utmost important challenges of information-seeking entities, in order to quickly and accurately make decisions. This domain is called **Streaming**, and it is characterized by the data's moment of creation being during the observation of the visualization, whose main concern is the representation of the data as soon as it is received and without it constantly abruptly changing the visualization with new data, but rather effectively smoothly represent the arrival and/or exit of data.

If we join both domains, the result is an analysis with the advantages of the two, which will address an even bigger computational challenge. Since, in Streaming visualizations, the data can not be processed before the beginning of the visualization, because they have not been created yet, it is necessary to group them in the visualization's runtime. However, the moment where the grouping occurs is also important to study. If grouped too soon, the system could, once again, lose important information, if too belatedly the visualization will hold too many visual elements. Moreover, the moment where it was created, should be explicit in the visualization. Newer data is usually the focus of Streaming visualizations, yet older data can present important information too, so both data periods should be represented. For instance, a user rating of a system's usability needs to focus on the most recent evaluations which will entail the current performance of the system, yet comprehending the differences between the previous iterations of the system and the current one, might improve it even further. This implies that more data needs to be represented, hence the system requires, once again, an aggregation technique that should group data according to its age.

With all this in mind, VisBig is a set of works that explore the concept of the **Graceful degradation** technique, presented initially in Pires et al. [6], whose objective is to represent Big Data Streaming data in different Levels of detail (LoD), with different visualization techniques, in separate timespans, from the beginning of the visualization until current real-time data. All data types have different applications, and consequently different adequate visualization techniques. To further explore the value of the Graceful degradation technique, we should analyze its contributions and limitations to other data types. The previous iteration of the work, only took into account quantitative data. In this work's case, the studied data type is the **ordinal data**, which until this point, to the author's best knowledge, was yet to be studied on the abovementioned technique. Ordinal data is defined as data divided in categories, where these present a defined discrete order. Examples of these types of data are seen in clothing sizes or in the *Likert* scale, with the last example being the most common in social sciences' studies [9]. Understanding the best visual representations for ordinal data in the Graceful degradation technique is the focus of this work.

1.1 Objectives

Our objective is to **study the concept of Graceful degradation for ordinal data using different visualization techniques in the context of Big Data and Streaming domains**. We focus on the design, implementation and analysis of techniques which allow the study of ordinal data, while also discretizing the data based on the data's moment of creation, presenting them in different levels of detail, with more emphasis in the most recent data. Additionally, the visualization should depict all the data since its beginning, by aggregating them in a final visual representation.

1.2 Contributions

This work contributes to the Information Visualization's field in a couple of aspects. First, the creation of a visualization capable of displaying large quantities of data in real-time, using up to date WebGL technology for a performance enhanced system. The work provides an evaluation of the system's measures, to ensure the overall performance, as well as its durability. The system's visualization is generated, with a flexible and dynamic definition of the viewed idioms and transitions between them.

Furthermore, the study complements the Graceful degradation technique with an analysis of the idioms that better express the information of ordinal datasets. And lastly, the work proposes transitions, which aid the user's comprehension of the information being aggregated between idioms. All the proposed visualization techniques are studied through user testing of unspecialized users, to filter out the better representations.

1.3 Organization of the Document

The document is organized as follows: Chapter 2 will present the state-of-the-art works in the fields addressed by this work. In chapter 3, the design and implementation of OrderWarp are addressed, following its evaluation on chapter 4. Finally, Chapter 5 provides an overview of the work, accompanied by its limitations and future work.

2 Related Work

Contents

In this section, we will evaluate the state-of-the-art techniques and technologies in the previously referred subject areas of our work, Big Data, Streaming and ordinal data.

This section will be broken into the first two themes because while ordinal data works on their own have been a popular topic in past years, they are only useful for our work if they are contained in either Big Data or Streaming. We will talk about the contributions each effort makes to OrderWarp at the end of this section. Despite the fact that each work is assigned to a single subject, they may contribute to others, they are, however, assigned to the subject they contribute the most.

2.1 Big Data

As referenced before, Big Data corresponds to the high cardinality of information that challenges its the computer analysis. Before initializing a work in this domain, it is crucial to fully understand the problems set by these types of visualizations. It is with this goal, that Gorodov et al. [7] analyses and defines five problems affecting these types of visualizations.

1. *Visual Noise* - When too many elements are drawn in the visualization, the user is unable to comprehend the information, since what the user sees is an indiscernible agglomeration of points. This concept is also commonly referred to as *clutter*;

2. *Large Image Perception* - Refers to the problem caused by an excessive amount of information on the visualization, even if legible, the user isn't humanly capable of processing it;

3. *Information Loss* - A common solution for the above-mentioned problems is to filter or to group data to reduce the amount of visible information, however if the used techniques aren't properly implemented this can lead to a misinterpretation of the data by only showing subsets of information instead of the full dataset, as required in any work of information visualization.

4. *High Performance Requirements* – With a bigger set of data, the data processing performance becomes an increasingly crucial aspect of the structure of the system, otherwise the user information analysis could be jeopardized by a slow or skip-filled representation;

5. *High Rate of Image Change* – Also applicable to streaming visualizations, this type of problem happens when there is so much data being updated, resulting in constant changes of information impossible for a user to follow along.

Lastly, the authors evaluate the capabilities of some idioms to visualize large data volumes, variety of data and the data dynamics. This evaluation is explained in the tables below.

When you begin the implementation of a visualization, it is common to start by structuring the ways in which the information is represented, before actually being visualized, and when we factor in the

	Large data volume	Data variety	Data dynamics
Treemap	+	-	-
Circle packing	+	-	-
Sunburst	+	-	+
Circular network diagram	+	+	-
Parallel coordinates	+	+	+
Streamgraph	+	-	+

Method name	Big Data class
Treemap	Can be applied only to hierarchical data
Circle packing	Can be applied only to hierarchical data
Sunburst	Volume + Velocity
Circular network diagram	Volume + Velocity
Parallel coordinates	Volume + Velocity + Variety
Streamgraph	Volume + Velocity

Table 2.1: Gorodov et al. [7]'s idiom analysis tables of Big Data support. The first table evaluates idioms according to its ability to support large data dimensions, data variety and data dynamics. The second table classifies the idioms according to the 5V's.

Big Data domain, it becomes mandatory since accessing data in datasets with very extensive numbers of records is time-consuming. The chosen representation has implications on the performance of the system e studying the most effective representations leads to the most effective responses to the *High Performance Requirements* [7] problem. Opting not to study the best structuring alternatives will impact the system's performance and subsequently likely compromise the user's ability to comprehend the information. However, there is no such thing as an ideal alternative to the representation of all datasets, and the best alternative will depend on the context of the intended visualization.

One of the possible alternatives is the ImMens. The ImMens [10] is a system that allows large quantities of data while still maintaining 50 frames-per-second (fps) with datasets with millions of records. To achieve this goal, the system stores data in pre-computed multidimensional projections, while also maintaining processing and parallel rendering techniques. The study focuses on idioms that intrinsically hold binning techniques, arguing that these idioms allow the analysis of global patterns as well as local characteristics such as discovering outliers.

The multidimensional projections represent the various dimensions held by the data, in a way that its access is done much quicker, however, the access speed is traded not only with the pre-processing time, but also with a significantly larger memory space needed to hold the projections. The ImMens reduces the required memory space by transforming the N dimensions cube in various tri or quadridimensional cubes, yet the authors point out that datasets with numerous dimensions make this solution ineffective.

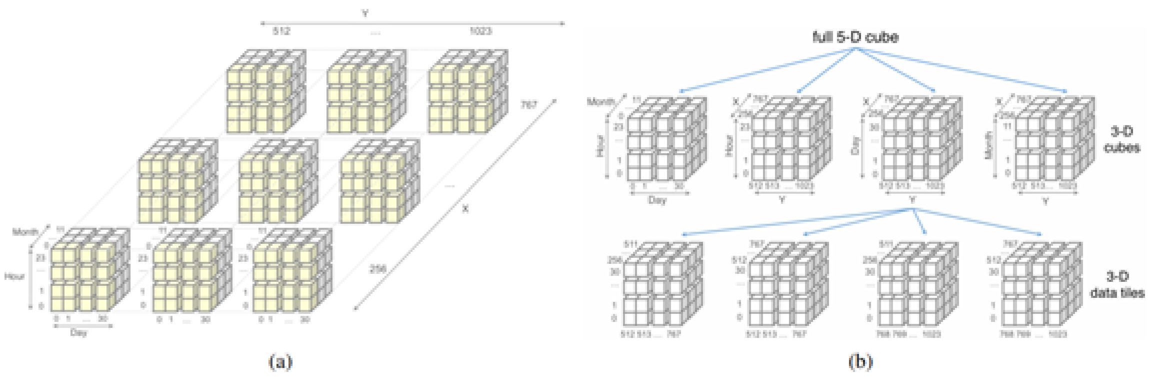


Figure 2.1: ImMens dimension reduction technique for multidimensional projections of the dataset.

If the dimensionality of the datasets is a visualization requirement, then another alternative needs to be studied. Sansen et al. [1] present a solution for large a quantity of data in the form of a parallel coordinates [11]. This idiom is known to be an effective solution for multidimensional comparison between records, while also not requiring plenty of user prior knowledge. In addition to supporting high dimensionality, it also supports practical interactivity techniques that simplify the user's data analysis. These techniques make the parallel coordinates an interesting study subject in information visualization. The authors test the idiom's versatility on two distinct case studies, one of them is represented in figure 2.2.

Parallel coordinates is a vertically scalable idiom, this means you can add extra information to the system has one more dimension and still get good results, however, it isn't in its natural for, a horizontally scalable idiom, which means that a large increase of data records affects its performance, as well as its legibility, since the data representative lines would overlap in the thousands and would clutter the visualization before reaching Big Data levels. To solve these issues, the authors use two strategies for a fluid observation of the data, the first being the Pre-Computation, that, as seen in other works, reduces the time needed for in real-time interactivity by computing possible data combinations instead of doing it on the moment. The second strategy, named on-demand computation, is, as the name implies, a technique to reduce the time it takes to compute the queries done in real time that could not be covered in the previous strategy, because of the high number of possible combinations.

For the actual visualization of the data, it is too time-consuming to draw each individual line, therefore the lines are bundled up according to the pre-processing grouping, and the thickness of the lines represents the amount of data records in the line's grouping. Furthermore, the intersections generated by different valued lines, might compromise the user comprehension of the data, therefore the lines are drawn using Bézier curves to simplify the intersections.

Ordinal data is well represented by this idiom, and this work shows one way they could be represented in large quantities.

The solution given by Sansen et al. [1] requires, however, consuming a lot of time in the pre-processing stage, as one case study took over 32 hours to complete the dataset aggregation.

A third alternative to the representation of the dataset is the Sye-MinChan et al. [12]'s approach. Being one of the first works on Big Data visualizations, the authors present Atlas, a system which in one case study could visualize an internet network traffic with about 1,28 billion records.

Atlas divides itself in four parts, the first one being the database organized in kdb+ columns, and it takes into consideration that the majority of queries searched by users only require a few couple of columns, and only these need to be store in cache. Kdb+ is a database specifically designed for time-series data, which is the subject data type of Atlas.

The second part of the system is a query distributing server that runs on the client side. This system

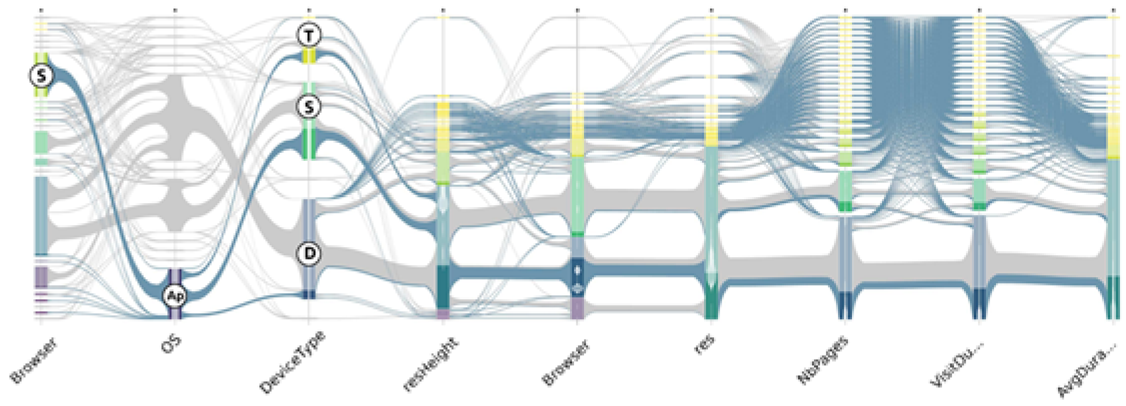


Figure 2.2: Case study applied to the parallel coordinates idiom on Big Data. The system allows users to easily observe tendencies like the use of the Safari Browser (S) is exclusive to Apple's (AP) Operative system (OS).

brings data from the available databases through a balancing query algorithm. The data is then transferred to the visual interface. This interface is the third part of the system, and is responsible for the user interactive operations, since they take such a crucial role in a detailed analysis of the dataset. To avoid over cluttering in the visualization, the different levels of zoom possible to the user are accompanied by different LoD's.

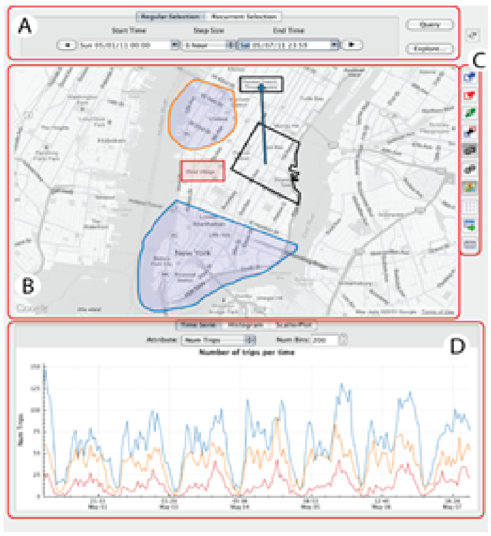
Since the Atlas objective is a fluid and response latency free visualization, the last part of the system is a predictive caching module, meaning that when an interaction with the visualization occurs this module discovers which are the most likely following interactions and immediately copies the data to cache and analyses which data should be substituted next. The objective of the module is to contain the least amounts of data possible, ensuring the system's efficiency. This allows for a simplification of the representation of the dataset to be done during the visualization rather than previously.

The most adequate techniques for visual representations of the data depend on the properties of each dataset and the domains that it covers. In Big Data visualizations there is a need to develop new representations that answer the visual noise, large image perception and information loss problems previously described by Evgeniy Yur'evich Gorodov et al. [7], while considering the study subjects faced by the visualization.

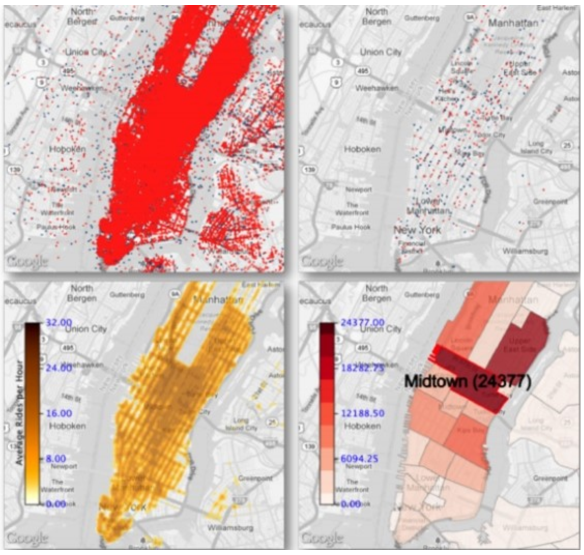
In Ferreira et al. [2], the authors analyzed different visualizations techniques for origin-destination and geospatial data in large quantities. The data comes from a dataset of a taxi network in New York, that, for its abundance in daily taxi trips, make for an interesting study subject.

For a deep analysis of the information present in the dataset, the authors begin by exploring the composition of the queries, since these are pivotal elements of user interaction, and its excessive number of possibilities makes the synthesis of the data exclusively in the pre-processing stage impractical. The authors opt to separate the queries in two, the atomic queries - a set of temporal, attribute or spatial restrictions; and the complex queries - a set of atomic queries through disjunction.

The system is composed of a point cloud map, as well as auxiliary line charts, that can both be filtered with the help of the visualization's widgets (A and C in figure 2.3). While A allows to choose the limits of the analyzed data, C allows drawing shapes in more detail, where the visualization filters the data inside the resultant area.



(a) Figure A



(b) Figure B

Figure 2.3: In I, (a) and (c) represent different data manipulation features, in (b) the user can observe data as well as draw filtering polygons, and in (d) there is a line chart with the lines showing the polygon's evolution through time. In II it is possible to observe the clutter problem, on the top-left square, on a dot map without any visual noise reduction technique. In contrast, the top-right shows an LoD alternative, the bottom-left shows the same information in a heatmap and on the bottom-right the user can compare between geospatial zones.

As seen in figure 2.3 (II) the problem related with the point cloud is that it could turn into an illegible visualization. Consequently, the authors implemented various LoD's, through a hierarchical data tree, and this difference can be seen in the first and second squares. Alternatively, the user can also visualize the data in different forms. In the third square, it is possible to understand the exact areas with the most taxi trips using the heat map, and in the fourth square we can compare between zones using the choropleth map. To allow comparison between different time periods, it is possible to divide the visualization in various time intervals and in slices, and visually verify the traffic evolution.

A common type of data domain to study is attributes with temporal properties. LiveRac [13] is a real-time visualization of large scale informatic systems using commonly known idioms like bar charts, line charts or heat maps. In these systems, the diagnosis of problems in a computer system must be done quickly, for that reason LiveRac treats data as soon as they are presented in the visualization, with the help of an exterior backup server that processes the data accumulation and its storing. This solution allows a Big Data visualization to be viewed in real-time on a common company computer.

The authors define principles in which the visualization should be ruled by, such as attribute comparison, multiple views or interactivity. From these principles derives a highly dense interface in a matrix format, that applies the semantic zoom technique previously referred as LoD's. This interface is, therefore, initially composed by a heat map that shows the totality of the dataset with plenty of interactive levels, where if one wants to further analyze an attribute or a record, it would simply need to expand the heat map row or column and the affected cells would transform themselves into sparkline charts or line charts depending on the wanted detail. LiveRac commits in presenting all the information of the dataset even if it means a cell taking up only one pixel, however, when the existing screen space is insufficient, the visualization solution is to group data blocks and representing them as one, until another change happens in the visualization space.

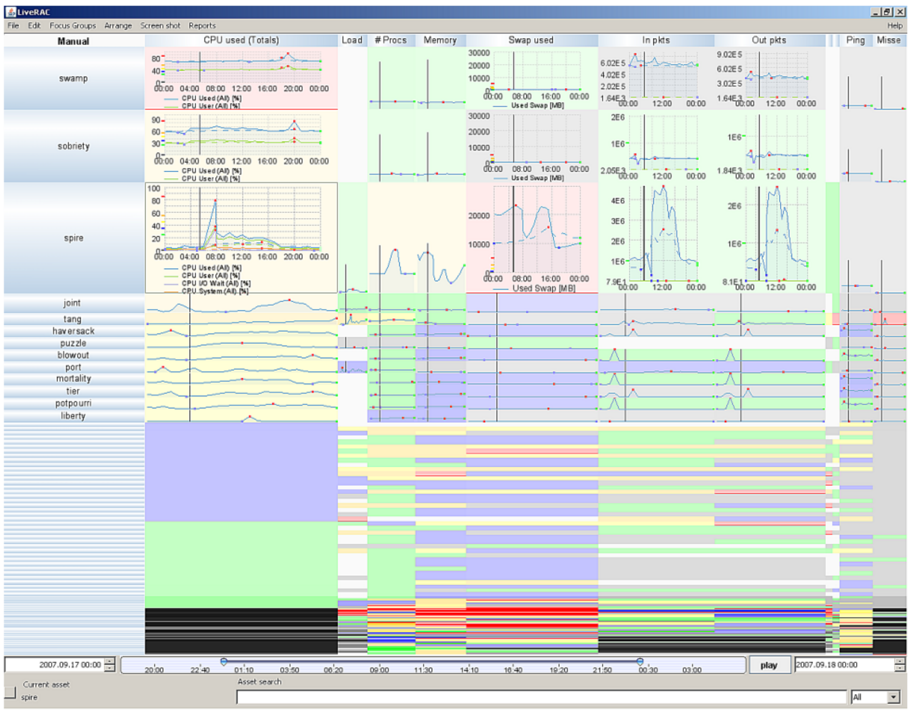
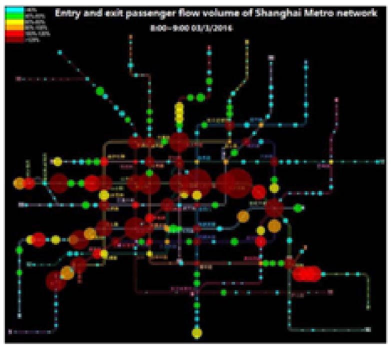


Figure 2.4: McLachlan et al. [13]'s LiveRac allows to observe the total dataset with different details according to the user's interaction with the system

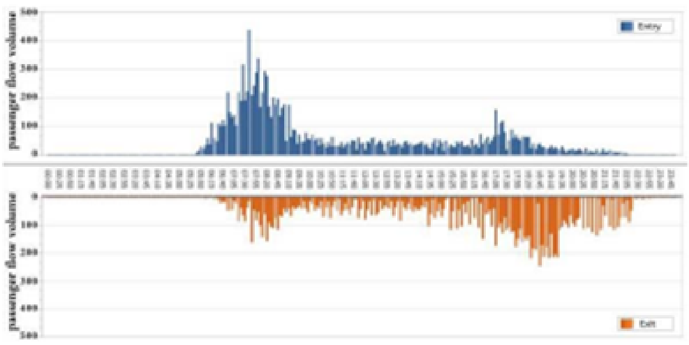
A great deal of found studies that approach the Big Data theme work with quantitative or geospatial data, and only a few study other data types. Topic-aware [14] visualization focuses on the large dimension of data in a companies email network. This system objective is to create a connections graph with thousands of nodes and connections between them, without resulting in an excessively cluttered graph.

Topic-aware's implementation in order to find a stable and visually comprehensible network is divided in three phases: the initialization of the visualization, the network scheme adjustment and lastly the post-processing. In the first phase, after creating the first network, the graph is separated with the Kernel density estimation (KDE) algorithm into different communities and for graph size reduction the t-Distributed Stochastic Neighbor Embedding (t-SNE) algorithm is applied. Following this, in phase two, the graph's scheme is incrementally adapted to better show salient structures. Finally, to improve legibility, the edge bundling technique is applied, same as in Sansen et al. [1]. The graph's goal is to easily identify *who* spoke with *whom*, *when* and about *what*, and with that in mind the authors allow a more thorough analysis techniques such as zoom and color scheming. Although the age of the email is studied, it is not possible to analyze the information in real-time, as that would require a restructuring of the network.

The studied works until this point all examined visualization techniques with specific idioms to their problem. Alternatively, Zhiyuan et al. [3] approaches various problems with different idioms. Using real-time data from the Shangai metro network, with the help of the automatic fare collection, the authors created a visualization that allows the study of large quantities of data generated in short intervals of time by the cities inhabitants. Not only is the dataset's dimension large, the number of attributes is quite extensive as well, and instead of implementing one dashboard where its dimensionality is reduced, the authors present various idioms with Big Data techniques that answer different problems. Through idioms such as bar charts, heat maps, or circle packing, the visualizations allow a deep analysis of the dataset.


Entry and exit passenger flow volume of Shanghai Metro network
8:00~9:00 03/3/2016

(a) Dot map of Shangai's metro station afluence



(b) Histogram with Entry and exit numbers for one metro station

Figure 2.5: a and b are just two idioms Zhiyuan et al. [3] presents for different data visualization of Shangai's metro network

2.2 Streaming

Just like in the previous section, before defining the future work, an effort must be made to understand the problems faced by streaming techniques. Since these techniques are characterized by real-time data analysis, meaning the studied information is created during the observation of the visualization, this means that we can no longer apply, in the same way, some techniques presented previously, such as pre-processing methods simply because there is no data prior to the beginning of the visualization. On that note, when we factor in the Big Data domain into a visualization, it becomes crucial to find new data aggregation strategies that do not require excessive computational time in order to maintain the visualization's performance and thus preserving an effective users' analysis.

The I2 [15] is a system that allows cluster applications to run in real-time side by side with visualization. I2 implements an algorithm that reduces the data's dimensions to the necessary elements at the specific moment where they are needed. The authors further prove that the used algorithm is both correct and minimal in terms of transferred data.

I2's name is due to its interactivity capacities in two dimensions, through code changes in the visualizations runtime or through an interactive analysis. The users can, therefore, alter the platform while the visualization is running and the cluster applications adapt to the new and modified visualization. The used algorithm is composed by a four-step pipeline adaptation of the previously known M4 [16] algorithm, so that it can be applied in streaming context.

This work is a great solution for streaming platforms for its ability of significant reduction of the dataset to just the necessary data, as well as a good example of an adaptable visualization when facing both the users and other developers' interaction, without the need to restart the system.

Fujiwara et al. [17] is another alternative of an in real-time Big Data aggregation technique, by focusing once again in methods of reducing the dataset's dimension to ensure the visualization's fluidity. The work reduces the necessary computation by applying aggregation algorithms every time the system receives data. The authors implemented an incremental version of the Principal component analysis (PCA) technique that calculates one iteration of the algorithm through the previous PCAiterations.

Furthermore, the work tries to solve two problems caused by dimension reduction techniques, being them, the preservation of the user's mental map and non-uniform data dimensions. To solve the first issue, the authors apply the *Procrustes* transformation, to make the transition of each data point's position easier to follow. And for the second issue, the system implements a position estimation method, that estimates where the positions of data points with incomplete numbers of features would be in the PCA result of another data points with the full set of features.

To test the work, a dashboard composed of a scatter plot, a parallel coordinates and a scatter plot matrix was created. The scatter plot is where it is possible to analyze the PCA derived data, and the other idioms serve as a tool for a more detailed analysis. When new data arrives or when there is a view alteration such as a zoom action, the areas where the data changes are highlighted inside the idioms, and to further express this, the authors use animations that follow the change, however if the flow of data is too high, the animations could harm the user's mental map instead of assisting it. Moreover, the work applies anomaly detection techniques that allow the user to explore the outliers separately.

Although the work presents important solutions for streaming visualizations, it does not try to analyze the time of the data, only focusing on the newer data. And, even though, it applies effective data change observation techniques, if the streaming data flow is excessive, the visualization becomes cluttered, as the work does not implement any visual noise reduction techniques.

Ragan et al. [4] analyzes possible techniques one may use on visualizations independent of its data flow which can also convey the data's timestamp, and does so by presenting a scatter plot as a study subject. The work's objective is a test to different focus-plus-context techniques, displaying their advantages and disadvantages in a temporal analysis context.

The authors explore four different techniques, being them a) a normal version with one simple values y-axis and one time x-axis limited to 40 seconds of data; b) a stretched version with the x-axis this time representing an 85-second interval, with the obvious space consuming disadvantage compared to a); c) a more compact version where the time axis grows exponentially solving the previous space problem, but at the same time this version is more likely to lead to visual clutter, as well as to more confusion to users not accustomed to exponential axes; d) a last focus-plus-context version that works as a combination of the normal scatter plot for more recent data, and a heat map for older data. This version allows a visualization of the entirety of data, and avoiding the clutter problem referenced in version c);

This versions where testes along four distinct tasks: the estimated average, average comparison, outlier comparison, and tendency recognition. In addition, two different variants of time intervals were tested. The results show that the only significant differences between the versions were in the tendency recognition, where the focus-plus-context version had negative results. The authors alert that even though the benefits of this technique are obvious, it also encounters problems, specially in long duration datasets. They also suggest that the presented limitations in these versions could be mitigated through complementary idioms.

Alternatively, Krstajić e Keim [18]'s goal is to study the functionality and the different faced problems to the most common idioms in the context of streaming. Streaming visualizations tend to represent the arrival of new data through changes in the idiom, and one of the objectives of the work is to understand which aspects of idioms can be altered, as well as which are the differences between types of idioms. The work also studies the perceptibility of the changes and how these could influence, or not, the user's comprehension of the data. Even though the work does not explore too much specific idioms, these represent the various existing of types of idioms.

Using one of Krstajić e Keim [18]'s examples, again with the known scatter plot, the authors conclude that possible changes in the idiom are the axes intervals, the addition or removal of dots and a change in the dots' properties. They conclude that the idiom is quite effective in context perseverance, unless an unpredicted alteration of the axes minimum or maximum occurs. Since there is a constant alteration of

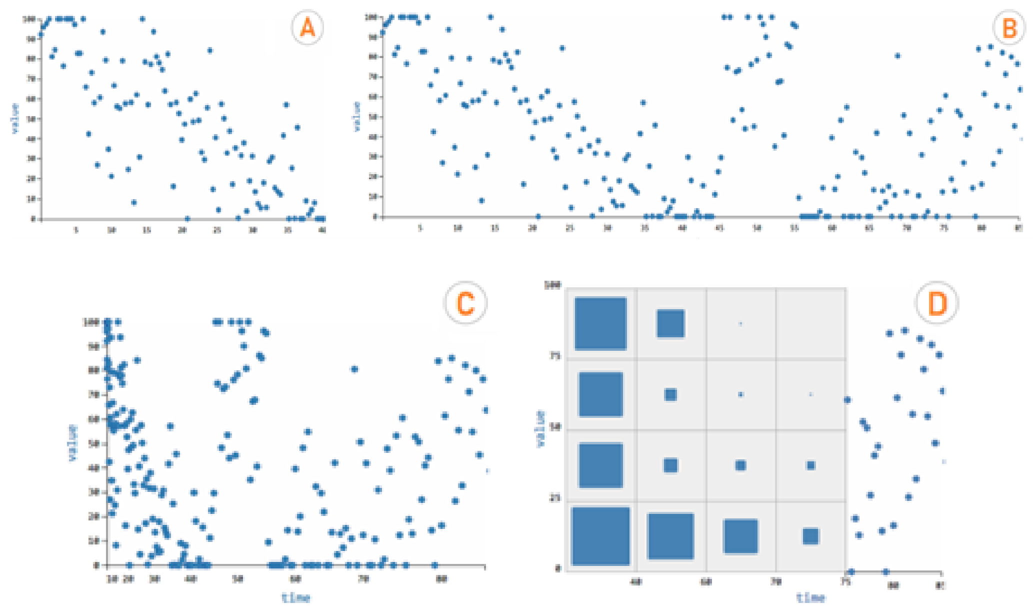


Figure 2.6: In A is the typical scatter plot where the x-axis represents time, in B the same axis is stretched to enlarge the visualized time period, in C the x-axis is now a logarithmic representation, in D is the focus-plus-context solution of a union of both the heat map and a scatter plot.

the timespan of the x-axis, its alteration does not affect the context in any way. Furthermore, the works alert for the previously explored problems of overplotting or visual clutter.

There are plenty of representation techniques for geospatial data with Big Data characteristics, however in the majority of works these techniques require pre-processed data, thus it is not common to find both Big Data and Streaming characteristics in visualizations. With this goal in mind, Lampe and Hauser [5] is a work that focuses in real-time analysis of geospatial data, avoiding the clutter issues in point-based strategies. The system groups the data, for facilitated read of the data, using the KDE concept.

The work offers countless equations on how to generate points according to KDE as well improvements on its typical implementation that allow the addition of points during the visualization. The authors, also address the possible interactivity mechanisms of the system to aid the user's analysis following the Shneiderman's task list [19]: *Overview first* - the visualization ensures it shows data in two dimensions spanning all the records and appropriate bandwidth; *Zoom and filter* - the system has implemented an individual zoom and panning for each axis, and allows the users to filter through time intervals; Lastly *details on demand* - as KDE is not an item based visualization, the user can not directly select a single data point, hence the authors solve this issue through drawable bounding boxes, which manage to filter data through given weights in the system's algorithm.

As referred previously, the type of data in a dataset also influences the applicable visualization techniques, and understanding the differences between these types results in more effective techniques for

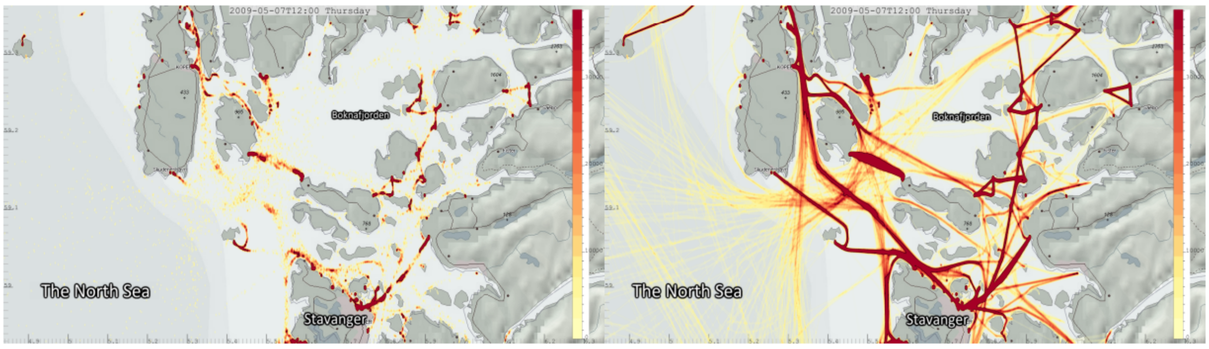


Figure 2.7: Kernel density estimation representation of geospatial data by Lampe and Hauser [5]

each type. For categorical data, most idioms are in part incapable of representing large quantities of data. As a response to this issue, Mansmann et al. [20] creates StreamSqueeze, a platform centered on streaming information regarding events/news, where its date dictates the position and space occupied in the visualization, and consequently its information detail. The system is represented as a matrix, where its columns represent new data in the most left column of the screen, and while time passes and the data becomes older in the context of the visualization and thus ”less” relevant, the events advance towards the right. To allow storing more data, each column holds double the amount of rows as its left column if half the space, therefore the user can analyze a good amount of data with more detail in newer data. To preserve the user’s mental map, each data event or news, when changes columns, tries to follow an approximately rectilinear path, which is also accompanied by a smooth and noticeable animation.

Since the system study subject is streaming events, the authors assign each event with one category encoded by the color of each matrix’s cell, and allow interactivity such as sorting according to its category. However, the authors explain that an analysis of the older data is difficult unless the user sorts the columns, which can also compromise the user’s mental map considering the leaps required to sort it.

This system can effectively display high amounts of data, around 100 thousand per day through various sources, yet, there is a limit on how long the data is visualized, since upon reaching the last column, the system does not implement any techniques to further store the data, and the events fall out of view. Because StreamSqueeze’s studied type of data is categorical, and therefore harder to represent in large scale, when the system faces higher flows of data its representation capabilities are affected, considering that the data matrix has a maximum number of cells, and at a rapid pace the events would enter and exit the visualization in a period of time that just would not allow its analysis.

To achieve a correct representation of new domains, it is often necessary to create new visualization techniques that work on top of proved techniques. Hashimoto and Matsushita [21] creates a system that joins the best of two idioms, the stream graph [22] and the heat map, both well versed in displaying time-series data, and in allowing the analysis of large amounts of data.

The streaming graph is a powerful idiom when it comes to tendency identification, something that heat map is not capable of representing, and heat map is effective in pattern recognition and data correlation, that stream graph struggles to represent. To explore the advantages of both idioms and to solve their disadvantages, the authors create a joint version of them, which they named the Heat Map Scope. The resulting idiom enables the stream graph's division both horizontally and vertically. Horizontally, the heatmap substitutes one of the stream graph's layers, hence it becomes possible to analyze the temporal correlations in the subbed layer. Vertically, the stream graph is split into a heat map in a defined interval, and here the user is able to compare different layers of the stream graph.

To preserve the user's mental map, all transitions between layers or time intervals from the stream graph to the heat map or vice-versa are done through perceptible interpolations of the visualization elements. This application adds value to streaming platforms, since it allows a temporal continuity without any big data related problems, because both idioms tolerate large data dimensions.

2.3 VisBig

The content presented in this dissertation is all part of the VisBig set of works, where fellow students have presented contributions that will serve as the base for OrderWarp. Orderwarp's goal is to extend the study further while maintaining the fundamentals of VisBig.

The first proposed work in VisBig was the VisMillion platform [6]. The aim of this work was, similarly to OrderWarp, to summarize Big Data streaming datasets, meaning that the proposed visualization should display all the information generated since its beginning, while maintaining the system's performance. To accomplish this goal, the data is grouped throughout its time inside the visualization, in order to reduce the needed amount of required data. This means that the visualization is presented in various LoD's, with the newer data shown in more detail and less grouped, and the older data with less detail and being grouped together in a final idiom.

The visualization is composed by three idioms, representing different time intervals, organized from right to left, where the idiom displayed on the right shows newer data, and the idiom on the left stores all the data that as entered the visualization and as reached the idiom (Figura 2.8).

For the newer data, the authors opt for a scatter plot, seeing as this idiom is the one that best represents the flow type that is being transmitted. Scatter plot represents all the received data points as single dots, which implies that no grouping is done, and therefore this idiom will have the biggest impact on the performance of the visualization. For mid-range timespans, the authors chose a line chart, representing the mean of the data by time, that allows the understanding of the data evolution, and pattern recognition, that is not possible in smaller timespans. Lastly, the idiom on the left is a bar chart, that is both a good solution for data comparison and its structure is ideal to store the full dataset as it enters the idiom. As it is observable, the data grouping levels rise and the detail levels lower with each idiom.

Upon defining the idioms, the authors realized that when the data flow is too high, both the line chart and scatter plot idioms became incomprehensible, and, as a solution, two modules were created, the Low flow module and the High flow module. The Low flow module maintained the previous idiom sequence, where the High flow module replaced the line chart with an adapted version of the same line chart with a box plot idiom. The new resulting idiom, draws the minimum, the maximum, the mean and the quartiles as different lines, with the both quartiles displayed as colored layers, covering the mean in the middle. The High flow module also replaced the scatter plot with a heat map, where its cells have a smaller timespan and a color representing the number of data points inside the timespan, and each cell moves along the visualization at the same rate as the scatter plot's dots. The heat map could also serve as an additional helper in the Low flow module by overlapping the existing scatter plot.

Above the three central idioms, there is one more scatter plot, with the goal of displaying the outliers of the visualization. Since the amount of outliers should never be enough to compromise the visualization's performance, the timespan represented by the scatter plot can be much larger than the idiom below without affecting the visualization.

As the authors mention, the focus of VisMillion is to study the applied visualization techniques and not necessarily to improve the system's performance. For this reason when the data flow is too high the system performance is compromised.

VisMillion's idioms were implemented so that each idiom would connect to the others with a small transition, since these transitions weren't the test subject of the work, the effectiveness of the transitions, as well as possible alternatives were not studied. Knowing this, Pereira et al. [23] is a work developed within the VisBig organization, where the work's goal is to comprehend which transitions between idioms are the most effective, and the authors do so by analyzing different techniques of transitions starting from the scatter plot idiom to a number of other idioms.

Vismillion and change, as the work is named, tests the transitions that best maintain the user's mental map, by examining the users' answers in trend comparison and recognition tasks, and in personal rating exercises. After analyzing and identifying the best transitions for each idiom, the work further solidifies VisMillion's capabilities in providing information through different techniques and different detail levels of the data for distinct time periods.

The limitation of this work, is its focus on just the transitions beginning in the scatter plot, since there are many more combinations of idioms that still require transition analysis.

A posterior iteration of these works was presented in Castanheira et al. [24]. The focus of FastViz, as it is named, was to explore, once again, transitions, but this time instead of the horizontal transitions presented in Pereira et al. [23] and defined by the authors as ”the transitions that deal with the data passage from a visual idiom of one module to a visual idiom of another module", the work studies vertical transitions. This means that FastViz studies the passage of data inside the same module, or in other words, the visual transition of the representations of one idiom to the other when there is an idiom change.

The authors choose to identify the best working vertical transitions between the combinations of the possible middle idioms in VisMillion, which are the heatmap, the line chart and the stream graph. The study finds the options that most help the user to keep up with the changes in the data between idioms by analyzing the user's answers in different tasks, such as the identification of the mean or the observation of the user's global effort.

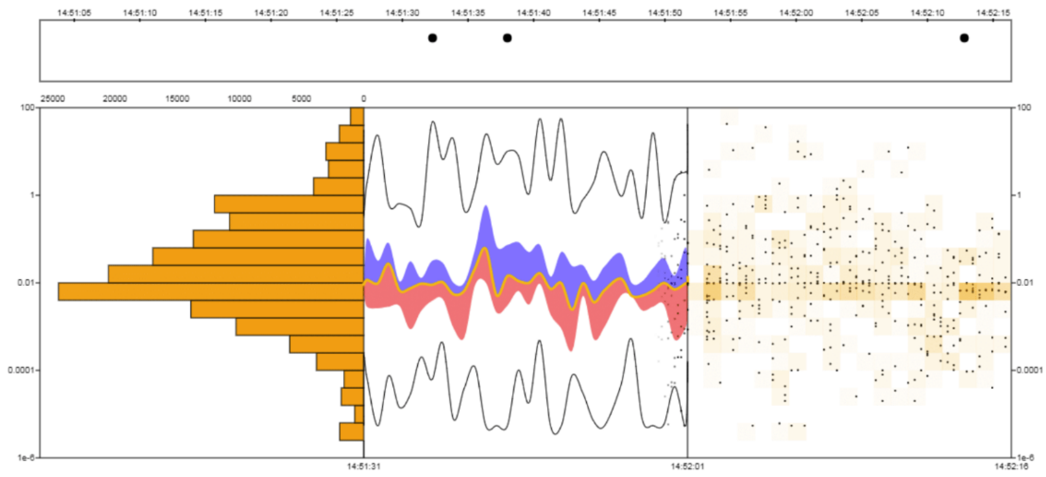


Figure 2.8: VisMillion visualization. Recent data shown in the left scatter plot idiom, passing through line chart for tendency and evolution study, and finally the data is accumulated in the right bar chart idiom. On top of the visualization, it is possible to see the outlier detection scatter plot.

2.4 Discussion

After presenting the state-of-the-art, it is important to discuss the main aspects that make these works relevant in Big Data and streaming, how they may contribute to our approach, and what is still lacking in these fields. Before the discussion, Table 2.2 summarizes all the works in the different study subjects of OrderWarp.

As previously described, Big Data's characteristics cause visual complications in unprepared visualizations described in Evgeniy Yur'evich Gorodov et al. [7] as visual noise and large image perception. The chosen techniques to solve these issues depends on the data type being analyzed. For quantitative data McLachlan et al. [13] presents the full dataset with different LoD's depending on the chosen space

Work	Dataset Reduction	Big Data	Streaming	Time-Series	Idioms	Data type
Zhicheng Liu and Heer [10]	X	X			Binning idioms	Multivariable
Sansen et al. [1]	X	X			Parallel coordinates	Quantitative, ordinal, categorical
Sye-Min Chan et al. [12]	X				Line charts, scatterplots	Multivariable
Ferreira et al. [2]		X			Dotted map, choropleth, line charts, heatmaps	Geospatial
McLachlan et al. [13]		X	X		Heatmap matrix, line charts, sparklines	Time-Series
Tim Repke [14]	X	X			Network diagram	Node-link relations
Zhiyuan et al. [3]		X			Varied	Multivariable
Traub et al. [15]	X	X	X		Not specified	Multivariable
Lampe and Hauser [5]		X	X		Heatmap	Geospatial
Mansmann et al. [20]			X	X	Matrix	Categorical
Hashimoto and Matsushita [5]		X	X	X	Heatmap, stream graph	Time-Series
Pires et al. [6]		X	X	X	Bar chart, line chart, scatter plot	Quantitative

Table 2.2: Summary of the state-of-the-art contributions to the studied fields

inside a matrix, Pires et al. [6] uses a bar chart that supports the aggregation of the complete dataset, with the data first passing through a scatter plot and a line chart, or a heat map and a box plot line chart if the data flow is too high. In geospatial data Ferreira et al. [2] allows the visualization of data in various forms, either being heat maps with different LoD's or by a choropleth map, while Lampe and Hauser [5] applies the Kernel density algorithm, as does Tim Repke [14], but in this case the objective is to group different communities in a data network representation. For multivariable data, Sansen et al. [1] uses a parallel coordinates with Bézier curves implemented in its lines to lower the amount of intersections and improve the legibility of the visualization, and, Zhiyuan et al. [3] uses different visualizations for different data types. These works represent large amounts of data successfully, however, no work was found exploring Big Data techniques for ordinal data visualization. The potential of the Graceful degradation technique makes its analysis a relevant subject, that lacks in most of the presented works.

Since the limit of where a dataset is considered Big Data is ill-defined, affirming that the visualization is capable of representing this domain is not enough, we need to design a system with the intent of displaying the most amount of data points possible, and understand the limits of the system. VisMillion [6] and VisMillion and change [23] decided to implement the visualization using the Data Driven Documents (D3.js) [1] JavaScript library for manipulation of documents based on data, and found that their visualizations could support up to 1000 dots per second.

In Streaming visualizations, it is important to analyze works that explore Time-series, where the timestamp of the data is fundamental for the visualization. The majority of visualizations explain age through, just one finite axis, that, in some idioms such as stream graph used in [21, 25], pass this idea

[1] https://d3js.org/

with clarity. However, this idiom is not suitable for every data type nor does it show additional details. Mansmann et al. [20] and Pires et al. [6] forward their data through the visualization, from one side to the other, displaying with more detail the newer data. In Mansmann et al. [20], because it addresses categorical data, the authors were unable to find a technique that would group the data in a final full aggregation, and so each data point, upon reaching the last column of the matrix, is forced to leave the visualization, which means that if in a high flow context, the data points might be forced to leave before being possible its analysis. In Pires et al. [6] the data passes through the different idioms, that work as temporal LoD's since they might only make sense in certain time orders, and they end up in the final cumulative bar chart.

Regarding Big Data, there is another field that requires attention, which is the system scalability when interacting with higher amounts of data. A common response to this issue is to apply pre-processing techniques to the dataset [1, 10, 12] that reduce the access time to the data by grouping it. Zhicheng Liu and Heer [10] accomplishes this using multidimensional projections, Sansen et al. [1] by applying known and extensive cluster algorithms, trading pre-processing time for user interactivity, Tim Repke [14] uses machine learning algorithms accompanied by the t-SNE algorithm. However, pre-processing is not enough to ensure an interactive interaction. As described before [1], the interaction possibilities are usually so high that it is not practical to apply these algorithms for the entire range of possibilities, therefore there is a need to study the interaction resulting queries. Sansen et al. [1] presents the on-demand processing module and Sye-Min Chan et al. [12] implements a technique for predictive caching, that foresees the next data necessary for the most likely succeeding interaction. However, these techniques require prior knowledge of the dataset. When real-time scalability is required, any pre-processing techniques are not useful, since the data is created as the visualization runs, so it is essential to analyze information aggregation techniques applied while the data arrives. This is the reason a good part of the works, capable of visualizing Big Data, do not address the streaming domain. When approaching real-time visualizations, the focus of the work should be fast and repeatable data treatment techniques. Traub et al. [15] implements a variation of the M4 algorithm to extract necessary data in the specific moments, and Fujiwara et al. [17] uses the incremental PCA to reduce the new data's dimensions. These solutions work with high flows of data, therefore they are effective for both Big Data and streaming domains.

Once more, the depicted idioms of a visualization depend on the data types of the dataset, but they also depend on the experience of the target users with the idioms, this is one of the reasons plenty of the studied works implement line charts, bar charts, heat maps and scatter plots [2, 5, 6, 12], because these idioms are known by the majority of the population. However, for more experienced users, Sansen et al. [1] uses the versatility of the parallel coordinates for a concise analysis of various types of data, McLachlan et al. [13] and Mansmann et al. [20] use matrices that make the most of the visualization's space by compacting the data and manipulating its size with different LoD's, and lastly, Hashimoto and

Matsushita [21] combines two idioms, a heatmap and a stream graph, retrieve the advantages of both to effectively demonstrate time-series as large quantities of data. The goal of OrderWarp is for the visualization to be accessible to every user, so the idioms have to be chosen accordingly regarding the overall most familiar and least complex idioms.

Lastly, the majority of analyzed works have referenced that, to maintain the user's mental map, any alterations to the data in the visualization should be accompanied by a transition from the first representation to the second. Of all the works, Pereira et al. [23] and Castanheira et al. [24] are the only ones that focus on studying and identifying the best transitions, and they do so, on top of VisMillion's system. Pereira et al. [23]'s centers its work in the horizontal transitions, by studying the transitions applied between modules which start in the scatter plot, while Castanheira et al. [24] searches for the most adequate vertical transitions applied within each module. VisMillion and change shows the necessity of studying transitions between idioms on graceful degradation visualizations, a subject that is also a requirement in OrderWarp, and FastViz expands the study to allow idiom changes in the visualization, without compromising the users' perception of the data.

The Graceful degradation technique as proved to be an effective technique regarding Big Data Streaming domains, yet it still lacks its study when specifically visualizing ordinal data, which is also the case in the field of Information visualization. No work, to the authors' best knowledge, as yet analyzed the impacts of this type of data, and these domains on the chosen idioms and transitions.

3 Proposed solution

Contents

Information visualizations can give powerful insights about data, useful in various situations, however, before idealizing the visualization, it is first necessary to analyze our goals according to four design levels presented in Munzner's Visualization Analysis and Design [26]. The domain situation level covers the understanding of what is being visualized, but since one of OrderWarp's goals is to create a dynamic solution, that allows the study of multiple datasets, analyzing this level becomes unfeasible. The next design level is the data/task abstraction, where the focus is to analyze what is going to be observed, what data is going to be visualized and what tasks should the visualization be responsible for answering.

The third design level manages the definition of the visual encoding and the systems' architecture required to accomplish OrderWarp's goals, and the last covers the implementation of the OrderWarp. Each following section presents the various phases of work the visualization went through, addressing these four design levels.

3.1 Approach

To achieve a perdurable and dynamic solution, it is necessary to previously define the tasks OrderWarp should respond according to the described visualization's goals in section 1.1. Therefore, the system's goal tasks are the following:

- Display large amounts of data (Big Data) received in real time (Streaming).
- Focus the analysis in ordinal data.
- Represent data in a single comprised visualization, grouping the data along a timeline, using the concept of **Graceful Degradation**.
- Produce a visualization that could run infinitely by adequately storing and deleting data in order to maintain a constant amount of allocated memory, thus keeping the visual stability throughout time.
- Allow manipulation of the visualization using different idioms that provide different insights to the data, by changing the number of idioms and its position along with the transitions of the data from one idiom to the other.
- Representing, if required by the user, all the data received in the visualization's runtime with a high dimension grouping technique.
- Accurately represent different timespans, to have multiple time periods to analyze.
- It should represent the transition between two different timespans, allowing the user to understand the leap between the time differences.

3.2 Architecture

OrderWarp's resulting architecture is represented in figure 3.1. The visualization is composed by a limitless number of different **modules**, all commanded by the **Modules manager**. The Modules manager is responsible for linking the modules by sending and receiving data to the respective ones, updating the current module's timings and for storing other important variables relevant for the visualization, such as the time it takes to complete an animation when there is a visual change in the representation, the start time of the visualization or the number of ordinal values in which the visualization will be divided. It is also responsible for the update of the common time axis.

Each module is created given a width and a timespan configurable by the user in order to accommodate the needs of the visualization, this way the user can study the data for different intervals of time (timespans) in more or less observable space (width) with different analysis methods, for different datasets. With the known width and timespan, each module calculates the position of an element inside the module through the equation $p = vModule + b$, where the velocity ($vModule$) is the division of the width by the timespan. The modules are responsible for an **idiom** that is the visual depiction of data. They are also in charge of storing, sending and removing data to and from its visualization for as long as it is inside the module's timespan.

A module can also be a **horizontal transition** module, which is the entity responsible for continually transforming the data in the form of the previous module to the next module's requirements. This is accomplished by the creation of bins, the structures responsible for grouping data. Bins are created one after the other, and will hold all the received data during the bin period. Since the transition is the link between two modules, the position of the data also has to decelerate from the first module's velocity to the second module's velocity. Opposite to the other modules, the module's timespan (Δt) is not given as an input by the user, but rather calculated with the remaining available visualization width (w) divided by the number of existing transitions, by the following equations:

$$a = \frac{vModule2^2 - vModule1^2}{2w} \tag{3.1}$$

$$\Delta t = \frac{vModule2 - vModule1}{a} \tag{3.2}$$

Once again, the **visualizations** are representations of the data using different techniques as well as different marks and channels where the users can obtain information about the data. Because these representations are purely visual, this means that we can change the module's idiom by removing every element inside the representation and starting a new idiom, making the system more dynamic.

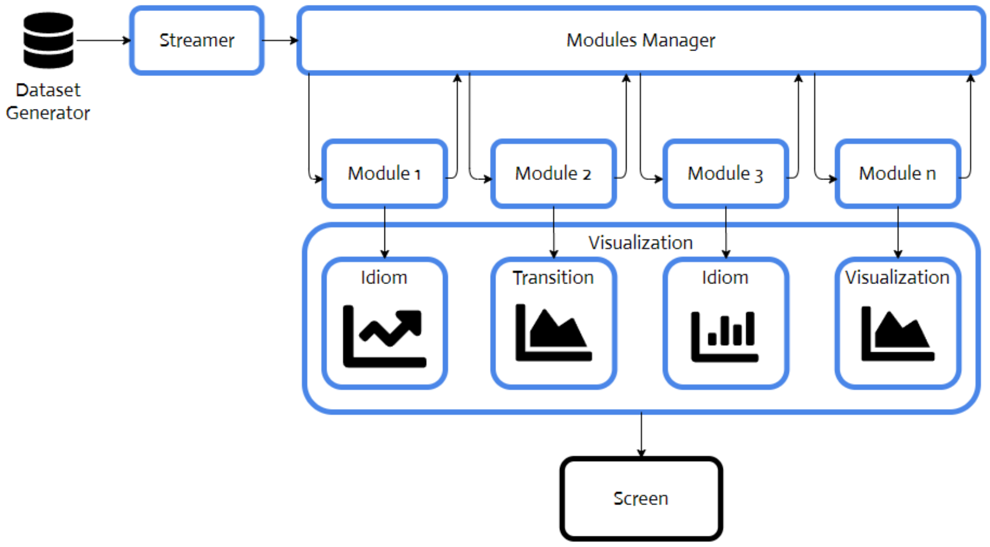


Figure 3.1: OrderWarp's architecture

3.3 Migration

In OrderWarp, the plan was to extend even further the amount of supported data from the previous works, and for this reason the initial focus of the system, was to consider up-to-date technology alternatives, and migrate the previous works onto the most effective ones.

The D3.js [1]library, used in VisMillion and in VisMillion and Change, makes it exceptionally easy to create and control visualization elements by binding data to HTML elements, as well as defining animation and interaction behaviors. This library usually uses Scalable Vector Graphics (SVG) technology, a Extensible Markup Language (XML) based vector image format, which means it creates Document object model (DOM) elements with the given instructions. Another possible alternative is canvas, more specifically HTML5 canvas, which instead of drawing according to vectors, it draws elements pixel by pixel inside the canvas' dimensions.

SVG's alternatives are a better and simpler solution for the representation of larger objects, since it draws the element as a whole and has already built-in manipulation and interactivity functions. However, when the number of objects is too great, this solution becomes ineffective, since the manipulation of the amount of DOM elements generated would be much slower. On the other hand, canvas draws by pixel, this means the scale of the dimension of elements does not alter the drawing phase of the system, and for larger amounts of data this solution is better than the previous one. Canvas is only a graphics container, therefore, the actual visual depictions need scripting, which leads to much more spent time in

[1]https://d3js.org/

implementation and increased quantity of lines of code. Knowing this, Pires et al. [6], migrated VisMillion from a SVG based technology to a canvas technology while maintaining the D3.js for data manipulation.

Our proposed solution implements the THREE.js library [2], that works as a framework used to create, manipulate and display 3D or 2D elements. This technology is also built on top of WebGL [3]. WebGL is a JavaScript API that accelerates the rendering of graphic elements in a web browser by shifting the drawing of the elements using the GPU's computational power. Apart from the JavaScript control code, a WebGL program includes shader code in OpenGL Shading Language (GLSL) [4]. WebGL, alongside the canvas technology, is shown by Kee et al. [27] as the best solution for displaying complex visualizations, as depicted in figure 3.2, which shows WebGL and canvas (omitted) as the best alternative with an SVG purely for better user interaction.

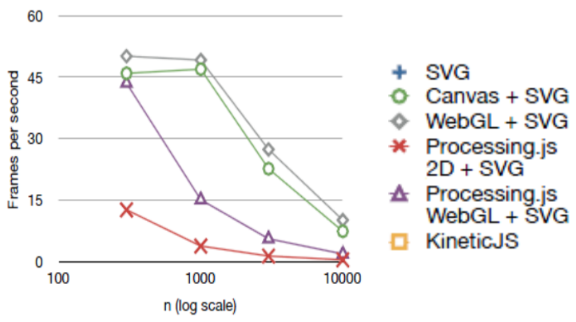


Figure 3.2: Alternative rendering techniques shown in Kee et al. [27], comparing the resulting fps per elements in the visualization. WebGL shows the best results

The first phase of implementation, after restructuring the architecture of the visualization depicted in figure 3.1, was to implement the visualization techniques studied in the VisBig organization [6, 23, 24] in the best way possible, within a WebGL environment.

THREE.js requires the creation of a scene which will be the subject displayed in the canvas through the view of a virtual camera. A scene, on its own, does not represent anything unless within it are objects called meshes, which can be of various shapes and forms in both 3D or 2D, whose behavior is dictated through scripting. To animate the scene, the application updates every object inside the scene and then renders it continuously for as long as the system runs.

3.4 Dataset Generator

To study the visualization's ability to fulfill the proposed requirements, the dataset has to be robust and extensive, so that the visualization is capable of accurately studying various tasks within the data. A

[2]https://threejs.org/
[3]https://www.khronos.org/webgl/
[4]https://www.opengl.org//

dataset like this, adding the fact that it needs to have ordinal data, is hard to find, and it is for this reason that we developed a dataset generator or data streamer which creates data according to commands that can be manipulated previously or in real time, giving us more flexibility with our testing.

The Dataset generator sends data through a python script, that will act as the server, with the help of the Flask library, a micro web framework, and its extension **Socket.IO** library [5]. Socket.IO is a real-time bidirectional and event-based communication module, based on WebSocket technologies, present in multiple coding languages. WebSocket is an open-source communication protocol between a server and browser provided that the browser supports the protocol, which is the case in 97% of the browsers as of 2020.

The WebSocket protocol was chosen because of its advantages over the known Hypertext Transfer Protocol (HTTP). As explained in [28], WebSockets improve network communications by implementing the following properties:

- Bidirectional: WebSockets allows a full-duplex communication, removing the client's need to send a request on every information exchange. It also allows continuous data streams to be sent by both the client and server, which is the case for OrderWarp.

- Persistency: The client-server connection is persistence, in the sense that it maintains the communication channel, instead of terminating it after providing the data. This is possible through constant pings from the server to the client, and only terminates the connection if the client says so or if it can not reach the client.

- Low-latency: Since there is no need for a new connection on every communication, the information sent on each message, after the handshake on connection, is reduced to only the necessary data, and therefore reducing the latency.

- Extensible: WebSocket is an open-source protocol, that can potentially implement a number of subprotocols, this way the technology can stay up-to-date since it is highly-adaptable.

- Secure: WebSockets can implement a secury protocol WebSocket Secure (WSS) that uses standard Secure Sockets Layer (SSL) and Transport Layer Security (TLS) encryption for secure connections and safe message exchanging.

The server was created with the intent of allowing both ordinal data and quantitative data to be created, and the data values are assigned by one of multiple functions of randomness. These functions are responsible for a different random distribution, such as a binomial distribution or a fully random probability, and they can be controlled with parameters, specific for each distribution. The chosen distribution

[5]https://socket.io/docs/v2/

could itself be changed or chosen according to a given probability. This implementation allows flexibility in point generation and manipulation in the visualization's runtime.

After creating a data point, the server sleeps until it needs to create a new one. The time slept by the server is given by dividing 1 second by the data point flow per second ($t = \frac{1}{flow}$), which is a configurable feature. However, the sleep function consumes time itself, so for bigger flows this means the server becomes unable to create points at a sufficient rate, therefore instead of generating one point at a time the server needs to generate a proportional amount of points (n) to the time spent between the start of the sleep function ($tStart$) and its end ($tEnd$) as shown in the following equation:

$$n = \frac{tEnd - tStart}{flow} \tag{3.3}$$

The data sent by the server is then received by the **Streamer** module, which is the platform of communication between the visualization and the server. This means the Streamer module is responsible for both client-server and server-client communications. While server-client communications only involves the transmission of data, the client can send instructions to the client as well as the visualization's details. The instructions cover operations such as flow manipulation or the change of the randomness function used for data points creation. The Streamer module also periodically returns details of the visualization, them being, the number of fps, the amount of received data, the amount of used memory and finally the detail's gathering timestamp, for the server to collect and store in a performance file, that could be used to understand the evolution of the visualization or for debugging.

3.5 Elements

The visual representations of data, displayed in the visualization, are individual elements created by both idioms and transitions which manipulate these representations to convey information as intended by them. All elements are instances of the THREE.js library, which allows the creation and manipulation of geometries and their properties effectively and efficiently.

For our visualization's needs, the required elements were the following:

- **Dots:** Simple, small rectangular planes that represent individual data points.
- **Lines:** Rectangular plane, where its height and angle of rotation is given by two x and y coordinates. The line thickness is also modifiable since it corresponds to the rectangle's width.
- **Rectangles:** Rectangular plane once again, and a single position just like a dot. However, in this case, the rectangle's size or color is modifiable. Another difference is the possibility of drawing borders surrounding the rectangle, these are accomplished by the creation of lines for each border, this way we can easily manipulate different borders as needed. Therefore, the rectangles

are comprised of five geometries instead of one, impacting the performance more than the other elements. However, this element is used for the representation of larger amounts of data and will presumably be in a smaller number.

- **Polygon:** This element is needed to create non-rectangle polygons, and it is possible through a buffer geometry. Instead of indicating a width, a height and a position, we have to provide a list of vertices positions. This element allows more flexibility since it does not restrict the shape of the visual representation, however its manipulation is much more complex and requires the specification of the position of each vertex at every update.

When using the dots element in the idiom implementation, on a preliminary test phase, it was possible to verify that the performance of the system in terms of fps's, would be heavily reduced. To improve the performance of the idioms that use this element, we substituted the dot element implementation with *instanced meshes*. These new elements allow the reduction of the number of manipulated elements to represent every data point, by using a single instance created to represent every data point that is yet to be processed between two updates. The previous representations of dots are inserted inside this instance once and from that point on, they are treated as one, meaning that all the dots are renderer as a single object, and the needed position calculations for each dot are also reduced to just one. This is accomplished using the THREE.js's *Instanced Mesh* class [6]. This class requires the provision of a fixed number of points prior to knowing the number of exact dots which will be inserted in the instance. For this reason, every surplus point is hidden outside the visualization.

The use of this class means that a substantial amount of points can be computed as a single one scaling the amount of visible points considerably. However, the *Instanced Mesh* comes with a few limitations, some of its properties like the opacity and color of the dots can not be changed unless you manipulate each dot, and we lose the capability of controlling the individual dot's position, that is so valuable for transitions. As an alternative we can change the scale of the instanced dot to try and mimic the merging of the dots, but this changes the size of the dots as well.

Ultimately, whether it be in drawing or scripting, its the amount of elements inside the visualization that dictates the performance of the visualization. The created scene holds all the elements' meshes inside a normal array, and calls the update function for each one, therefore, more meshes equals more update time. On top of that, each time an element is removed, to delete it from the array takes $O(n)$, and in OrderWarp this is a common practice of every module each time the element exits its representation, making it an unfeasible solution. To solve this issue, and since the position of the mesh in the scene's element array is not relevant, each element class presented above stores a static array that holds removed elements from the module without removing them from the scene's element array, and when a new element is required anywhere, instead of creating a new mesh each time, it reutilizes the ones al-

[6]https://threejs.org/docs/#api/en/objects/InstancedMesh

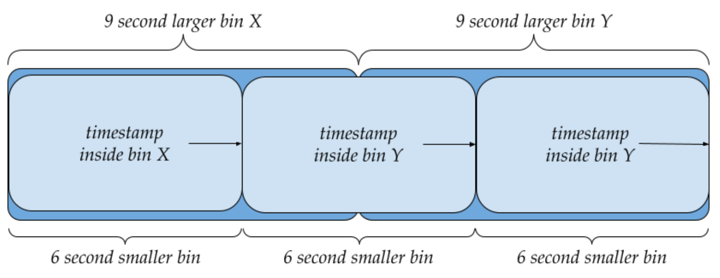
ready stored in the array, lowering the time of creation of meshes and removing the need to delete each element.

3.6 Bins

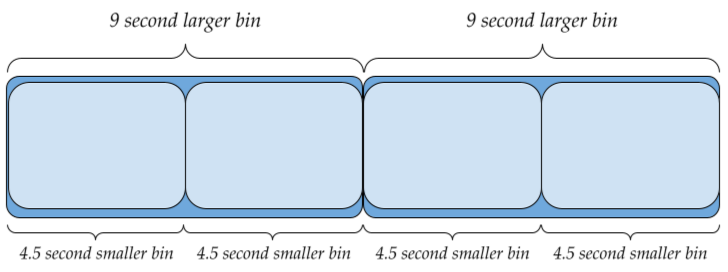
The concept of Graceful degradation is only possible with the help of bins. They are responsible for reducing the amount of data in the visualization by representations of the same, lowering the computational load and allowing more information to be displayed. Bins are simple agglomerations of all the data received during a period. After the bin duration has passed and no more points are added, the bin will hold various properties that will act as representations of the data inside the bin to simplify its computation, as well as a timestamp which is equal to the ending date of the time period. Examples of these properties are the average value of the bin, or the number of points inside the bin, which are ultimately the most needed properties to summarize the data in most idioms.

The bins are created in the transition modules when the previous bin has surpassed the transition's beginning date. They can hold either single data points or other bins, which is true whenever there is more than one transition, if their timestamps are comprised within the new bin. In this case, the bin duration of the next transition has to be a multiple of the previous one, if not, the bin durations might misalign and the data inside the bin might not reflect the actual distribution of the data but rather a different number of received bins. See the examples in figure 3.3 where in a) two same size larger bins receive a different number of smaller bins (bin X receives 1 bin, while bin Y receives 2), simply because their durations do not match, something that does not happen in b) since both receive the same amount.

The bins are created considering the start time of the visualization, and they can only be created in a timestamp equal to the start time plus a multiple of the bin duration. If this was not the case, this could mean that a portion of the smaller bin might not be fully inside the larger bin interval, yet still considered part of it, because its timestamp is within the larger bin's interval. Ultimately, the data point inside this portion would be represented inside the larger bin, as its timestamp is inside the smaller bin, which is then inside the larger one. This event compromises the visualization's credibility, since some representations of the data points would be shown in a timestamp that would not be truly the data point's timestamp. Figure 3.4 represents the above problem, with the darker tone showing data points created between 10 seconds and 9 seconds inside a smaller bin, being integrated in the larger bin on the right since the smaller bin's timestamp is 5.5 seconds.



(a) Non-multiple bin sizes



(b) Multiple bin sizes

Figure 3.3: Depiction of the difference between non-multiple and multiple bin duration times.

3.7 Idioms

Idioms are the visual representation of data, who's job is to allow the user to accurately understand the information present in the data, and take insightful information about said data. The idiom's suitability relies on the tasks relevant to analyze, such as the evolution of the data or the comparison between ordinal values, while also taking into account their volume or in which timeframe they are being studied. As explained above, our goal was to create a visualization containing ordinal data, big data and data streaming techniques over different time periods. After careful deliberation over the adequate marks and channels for the required characteristics, in this section we present the studied and implemented idioms.

3.7.1 Cleveland plot

The first proposed idiom was the Cleveland plot as a variation of the quantitative solution of the scatter plot. In both idioms the visual representation is achieved by drawing an individual dot (a small rectangle) for every data point received, with its y position encoding the data point value, and the x position encoding the timestamp of the data, since the common x-axis represents the time between the start of the visualization and the present time, thus being constantly updated, the dots x position will be updated with

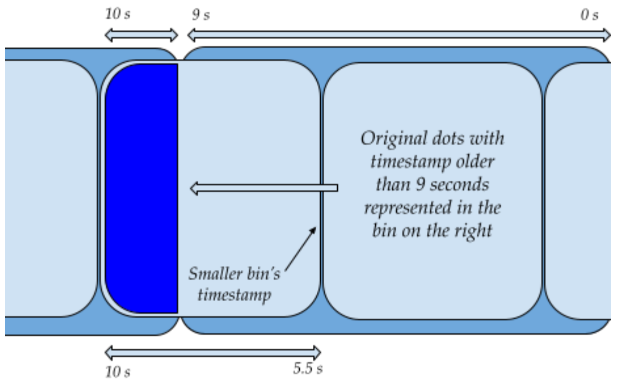

10 s
9 s
0 s
Original dots with timestamp older than 9 seconds represented in the bin on the right
Smaller bin's timestamp
10 s
5.5 s

Figure 3.4: Depiction of data being misrepresented because bins do not start at time multiples

the axis, moving towards the idiom's data exit boundary position. The difference between both idioms is simply the domain presented, in the scatter plot the y-axis expresses the data points' values inside a continuous interval between two numbers, meaning the dots will be dispersed freely in two dimensions, whereas in the Cleveland plot the dots are restrained into invisible lines representing each existing ordinal value. Although they are considered different idioms, in terms of implementation they do not differ, as the ordinal values are converted to a position in the same quantitative interval.

Because the idiom represents its data points individually, this means the number of elements being drawn every update is considerably high, and its impact on the visualization's performance will therefore also be higher than other solutions. To increase the idiom's performance, the previously used dot mesh was substituted for the instanced dot mesh 3.5 allowing a much greater amount of dots inside the idiom. The increase in number of dots also raises the concern of dot clutter, when the number is so high that the dots just overlap and the data's distribution is no longer clear, consequently affecting the user's data perception.

The Cleveland plot is a great solution to understand the arrival of data points and its distribution over time, as the user sees one dot for each data point representing its exact value and timestamp. This idiom currently has no solution for the representation of bins instead of individual data points, therefore it is only suitable for the beginning module. It is also more effective for smaller timespans, both computationally and visually, since longer time periods will result in more elements to draw, and with it a bigger visual agglomeration of points.

3.7.2 Heat map

The common heat map resembles a table of colors, where the rows and columns encode different attributes of the data and their intersection returns a number encoded with a color inside a color spectrum.

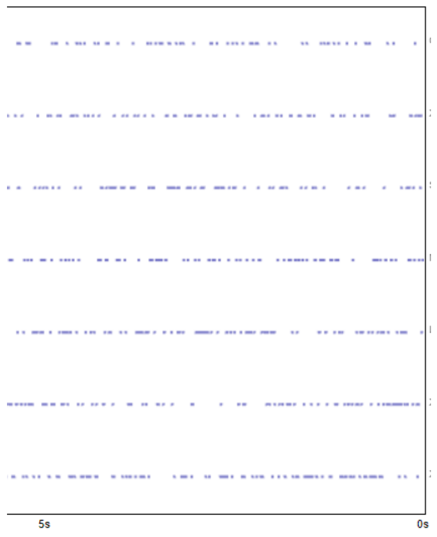


Figure 3.5: Cleveland plot idiom: The dots are moving from right to left representing its timestamp along the timespan of the idiom, and each ”row” of dots depicted represents one ordinal value.

In OrderWarp’s implementation of the heat map, the position on the vertical axis will encode the ordinal value and the position on the horizontal axis will encode a fixed time period, where the number of points inside the period will yield the saturation of a rectangle to which we called a cell. The cell’s time period is a fixed date, so this means that the cell’s position will shift to the left just like the horizontal axis until they reach the left boundary of the idiom, where they are sent to the next module representation or simply eliminated out of the visualization.

The number of points inside a cell can not only vary with the data’s distribution, but also with the overall data flow of the visualization. This means we can not know the optimal maximum value for the value scale, so the scale needs to be adaptable to the idiom’s data, increasing the maximum value if its maximum was surpassed and also lowering it when the value no longer accurately reflects the data. In order not to compromise the user’s comprehension of the data, we need to avoid any visual leaps in the representation, that is why instead of instantly changing the maximum/minimum value and the colors changing with it, the idiom’s scale values are gradually increased/decreased to the new value and with it a smooth color transition.

This idiom is suitable for various time periods, but presumably most suitable as a replacement of the Cleveland plot 3.7.1, since instead of drawing every single data point, only draws one element representative of a small aggregation of the data. However, it also somewhat limits the temporal discretization of the data that makes the Cleveland plot a great solution.

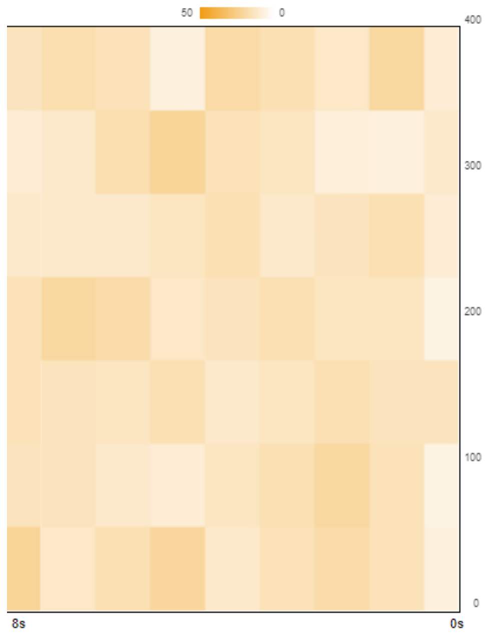


Figure 3.6: Heatmap idiom: Each cell's color represents the number of data points between a smaller timespan, and each cell moves leftwards since it represents an interval between two timestamps.

3.7.3 Ordinal line chart

As explained before, ordinal data shares a lot of characteristics with quantitative data, so it is understandable that an idiom is suitable for both data types, however, in ordinal data, the concept of a mean does not have any intrinsic value, and for this reason, the solution of a mean line chart used in Pires et al. [6], where the position of a point in the line encodes the mean value of a series of data points, needs rethinking to suit ordinal data.

The goal of a mean line chart is to understand the distribution of points in one instance and the evolution of the data throughout an interval. It is easy to maintain both tasks if we change the mean to the number of total points, which still allows the understanding of the distribution if we split the one line into the number of separate ordinal values, as if there are different line charts stacked on top of each other, this way the distribution will be perceived as the comparison of positions in the multiple row's lines, and the study of the data evolution is not compromised since the marks and channels of both idioms stay the same.

So in this idiom implementation, the received data bins will inform the number of points for each ordinal value that will be encoded in a position on the y-axis split for each ordinal value, where the maximum scale position of one ordinal value is immediately before the zero position for the next ordinal value, only divided by a ground line for each different "line chart". Each position of the ordinal value is connected to the previous bin's position, forming a line out of line segments, and it is this line's slope that encodes the evolution of the data. The x-axis encodes the bin's timestamp, so the lines will be constantly moving towards the idiom's boundary, where they will be expelled. For the comparison of positions between ordinal values, the same scale is required for every single value, otherwise the user's comprehension of data would be affected since two values in the same position would encode different values.

As implemented in the previous idiom (section 3.7.2), the number of points can vary immensely between different time periods, and to be able to understand the data in all periods, the scale needs to be adjusted to the number of points inside the idiom. If the amount of points surpasses the maximum scale value, the scale's upper limit has to increase to the new value plus a fraction of $\frac{1}{10}$ this maximum. This fraction serves to avoid the lines reaching the upper limit line, which could possible overlap with the above line. To avoid constant scale changes, when the maximum value leaves the idiom, the scale's upper limit only changes if the new maximum in the idiom is less than 70% of the previous value. The scale's new maximum decreases to the new maximum plus the previously described interval.

The idiom's capability of transmitting the evolution of the data, makes the idiom suitable for periods with somewhat of a time magnitude. If the idiom's timespan is too little, the evolution might be insignificant, and on the other hand, the idiom does not allow for a full grouping of data since it is constantly moving with its timestamp and renewing the depicted data.

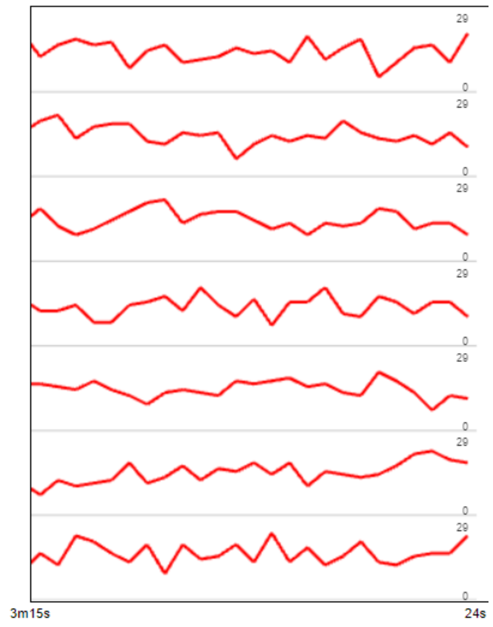


Figure 3.7: Ordinal line chart idiom: There is one line for each ordinal bin, updated after every bin duration of the previous transition.

3.7.4 Stacked bars

Stacked bars' goal is to effectively show proportions of the data values between every existing value group (typically different categories) encoded with a different color and their values are comprised in stacked lengths of a single bar, and then compared to other stacked bars. The bigger the value of the group, the bigger the proportion in the number of total values of that specific bar. However, this does not mean it has more data points than the same group in another bar, since the data quantity is not encoded, but rather the percentage of data inside each value group.

In OrderWarp each bar corresponds to an ordinal value, so they are encoded by the y-axis, and the group values are encoded in the x-axis, which means the groups correspond to time intervals defined as **"Eras"**. The length of each stacked bar is the full width of the idiom, and the Eras are encoded with a color from a list of colors that are reused along the visualization's age. Each Era has the same timespan, and every data point that enters the idiom will belong to the latest one. When the new Era's timestamp surpassed the idiom's beginning date, a new one is created. Since the Eras also have a timestamp, this means they will also move along the x-axis. Furthermore, the first Era will not be fully represented most of the time because their time intervals have not entered the visualization's timespan yet, therefore, not all points of the Era are yet represented in the bar. The opposite thing happens for the last Era, where the points have already surpassed the timespan, in this case a proportion of the points has already moved to the next idiom.

The movement of data in and out of the module is represented by the change in the Era's visual length, which should not compromise the idea of a data path presented by the Graceful degradation concept. This means the idiom is suited for middle idioms, since the Eras need a considerable interval to represent relevant visual cues. The idiom is not capable of accumulating data unless the Era's timespan would change during the visualization's runtime and incorporate other Eras with the passing of time. However, this idea was disregarded seeing that the longer the Era the bigger timespan it will presumably have, and in the future of the visualization there would be an overwhelming amount of points in a single bar with no visual value for the user.

3.7.5 Bar chart

A bar chart is an idiom represented by bars with its length or height, depending on the bars being horizontal or vertical respectively, representing a quantity on one axis and the different bars representing the other axis, this makes the idiom the perfect candidate for comparison between values.

The bar chart is another example of an idiom appropriate for both quantitative and ordinal data. Where in VisMillion this was possible by dividing the values in intervals, with ordinal data the division is already defined by the ordinal values. In our implementation, to keep a visual continuity the bar

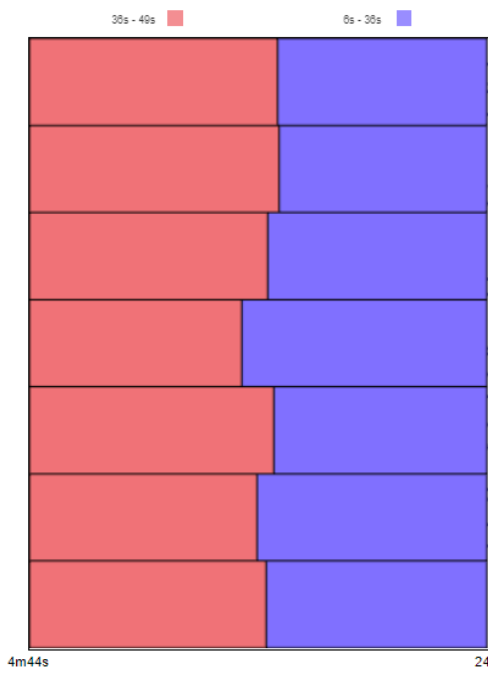


Figure 3.8: Stacked bars idiom: The various bars in the vertical axis represent each ordinal value, with each bar in the horizontal axis representing a bigger time period called an **Era** also encoded by a color, and the width of these bars represents the percentage out of the total values for that ordinal value in the idioms timespan.

chart is horizontal, and the y-axis encodes the different ordinal values and, therefore, there are two different x-axes. The first one is a scale of data points that varies from 0 to a value above the maximum value, otherwise the maximum value bar would always have the length of the idiom, and it could be misinterpreted by the user. The second axis is the common axis that represents the timespan covered by the idiom.

To represent the arrival of new data, the bars gradually (along the transition time interval) increase in size to their new values on the idiom, however the number of points will increase presumably more than the initial maximum scale value of 1000 points, so to keep up with the maximum increase, when the value hits 90% of the previous maximum value, this value changes to a new maximum plus 30% avoiding a constant change of the maximum value every time new data arrives. When the scale changes, the bars' length once again changes, this time by gradually decreasing in size, since the maximum was increased.

This idiom is only suitable for a last idiom period, since there is no visual connection between the data to the left boundary of the idiom, to keep the idea of a data path through time. For a last period idiom, it works perfectly, as the constant update of bar size and scales allows the data to be grouped indefinitely since the start of the visualization.

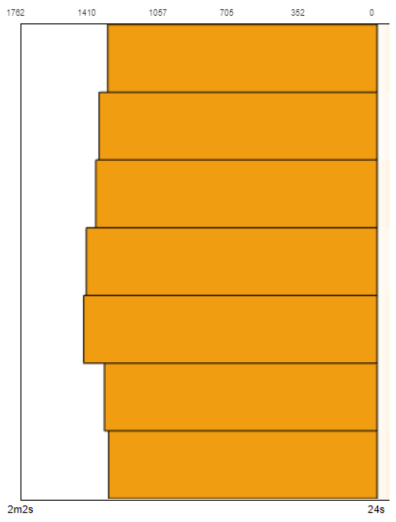


Figure 3.9: Bar chart idiom: Each bar is placed horizontally, connected to the visual beginning of the idiom. Its width enlarges every time data points arrive to the idiom representing the number of data points since the beginning of the visualization.

3.8 Transitions

The transition's role in a visualization is to provide a visual continuity between changes in the data, whether it is an update in the scale or the change of its presentation. OrderWarp applies various types of transitions, but the term transition will be used to refer to **horizontal transitions**.

Horizontal transitions are, once again, the continuity between two idioms, where it is the transition's objective to seamlessly transform the elements that represent the data in the former idiom **(A)** to the next idiom's **(B)** element properties [24], while it visually communicates the differences between channels in both idioms. The goal is for the transition to perceptibly show that the information present in both idioms is the same, with a change in the aggregation level.

The velocity of each idiom varies with its timespan and width in the visualization, and for this reason, it is also the transition's responsibility to gradually transform the visual element's velocity from idiom A's velocity (vA) to idiom B's (vB). Therefore, this means that the transition has to decelerate the elements, using the calculated acceleration a in equation 3.1, to then determine the element's position p with the equations:

$$\Delta t = |timestamp - idiomStartTime| \tag{3.4}$$

$$p = offsetA + vA + a\frac{\Delta t}{2} \tag{3.5}$$

Two different transitions were developed to explore the properties of an effective transition, for each

Idiom	Cleveland plot	Heatmap	Ordinal Line chart	Stacked Bars	Bar chart
Cleveland plot			Growing Bars	Stacking Dots	Stacking Dots
			Grouping Dots	Pilot Lines	Pilot Lines
Heatmap			Line Morphing	Stacking Bars	Stacking Bars
			Squares	Squares	Squares
Ordinal Line chart		Cell Morphing		Stacking Bars	Stacking Bars
		Line squeeze		Dissolving Lines	Dissolving Lines
Stacked Bars		Cell Morphing	Line Morphing		Stacking Bars
		Squares	Squares		Squares
Bar chart					

Table 3.1: Transitions table

studied idiom combination, apart from the already studied Cleveland plot to heatmap in Pereira et al. [23]. Also taken out of the equation, was every combination where the bar chart is the A idiom, since it is always the accumulator and final idiom. Some transitions are adaptations of previous ones for different combinations of idioms, and others take inspiration from the previous VisBig works [23, 24]. The goal with these transitions is to connect the combination of idioms, such that no information seems lost in the process. The number of transitions between combination was chosen to provide alternative transitions and to avoid an extensive combinatorial explosion of transitions, which would limit the implementation time as well as its analysis.

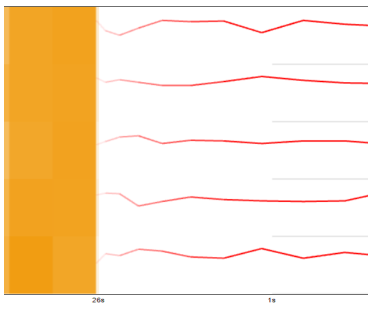
The transitions applied towards the bar chart idiom are the same for the stacked bars idiom, as both use the same marks and the same height channel. Therefore, in each following described combinations, both transitions with these idioms as the B idiom, will be merged. In addition, each combination of idioms also has a default transition, which is a simple fade out of the elements from idiom A.

Table 2.2 shows a summary of each implemented transition for each combination of idioms, excluding the same idioms combinations and Cleveland plot to heatmap combinations previously studied in Pereira et al. [23].

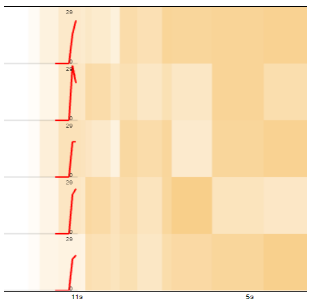
Below, the studied and chosen transitions are explained in more detail and sectioned by the start idiom of each transition. All transitions are accompanied by figures to ease their comprehension, as well as a link to a small video where it is possible to visualize exactly how the transitions were implemented.

3.8.1 Default Transition - Fade out

If no transition is selected, a fade out transition is created. This transition receives the elements from the previous idiom and moves them while gradually decreasing its opacity until it reaches the next idiom's start boundary. This works for both the heatmap and ordinal line chart, since these idioms represent data in a linear flow, where the representing elements exit the idiom by continuing their path at the same speed. This is also the case for Cleveland plot. However, this idiom utilizes *instanced meshes* and as previously explained it is not possible to efficiently change the opacity of these elements, thus they simply move towards the idiom's left boundary. In the other idioms, when data exits the idiom, it is represented by one element. This is not the case for the stacked bars idiom. The representing rectangles do not leave the visualization unless the Era also leaves the idiom's timespan. Therefore, when a bin exits the idiom this does not mean that its representing elements did the same, since there is not necessarily one. For this reason, the default transition after a stacked bars will result in no element leaving the idiom, and consequently there is no visual transition. Finally, since there is no bar chart beginning transitions, the default transition is not possible as succeeding transition for this idiom as well. These limitations in the default transition, adding the fact that they were already studied in Pereira et al. [23], caused this transition to be removed from user testing.



(a) Ordinal line chart's fade out



(b) Heatmap's fade out

Figure 3.10: Fade out transition, starting with ordinal line chart and heatmap

3.8.2 Cleveland plot

In the next transitions, it is important to mention that to manipulate the dots according to its ordinal value, the logic behind the transitions divides the *instanced meshes*, in smaller instanced meshes that represent the points for just the ordinal value instead of the whole domain. This multiplies the number of existing dots by the number of ordinal values, and consequently worsening the performance of the visualization, although still much more efficient than using individual dots as a representation.

3.8.2.A Cleveland plot to ordinal line chart

Growing Bars: The dots converge on top of a bar in the left boundary of the transition, causing the bar to grow to the value it will have in the ordinal line chart. Additionally, attached to the top of the bar is a line that joins the next idiom's latest line, creating a seamless connection where there is always a line being produced, and its value grows with time before being transferred to the next idiom. The dots entering the bar give the idea that they are being grouped by pilling up in a bar whose height is the number of dots inside it.

Video: `https://youtu.be/uGGXSPwl6g8`

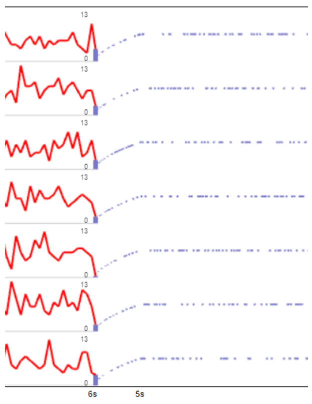


Figure 3.11: Cleveland plot to ordinal line chart's Growing Bars

Grouped Dots: Similar to Pereira et al. [23]'s scatter plot to line chart. The dots converge at the beginning of the respective line segment, giving the idea that the dots group together and return the group's position on the ordinal line chart. The last line in the transition, is again connected to the latest line in the ordinal line chart, and the lines' opacity grows the closer they get to the next idiom.

Video: `https://youtu.be/2sFZH_bYrm8`

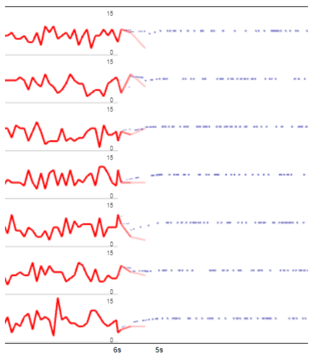
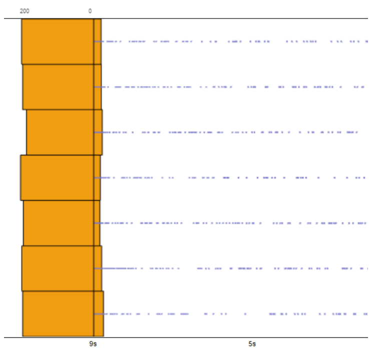


Figure 3.12: Cleveland plot to ordinal line chart's Grouped Dots

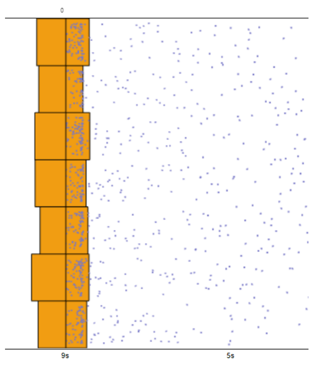
3.8.2.B Cleveland plot to stacked bars/bar chart

Stacking Dots: Once more, a similar approach to Pereira et al. [23]'s scatter plot to heatmap transition, where the dots move throughout the idiom until they reach the idiom's boundary or other points, forming a crowd of points. These points also enlarge a horizontal bar that will be "pushed" forward into the next idiom. This means that, upon reaching the bin's time to be transferred to the next idiom, the rectangle, which represents that bin, will translate to the idiom according to the Modules Manager animation time.

Each transition was implemented, with the possibility of also representing quantitative data, however, in this case the transition effectiveness, even before user testing, was already comprehensibly unclear for ordinal values distinction, since the formed crowd of points, in this case, would be a small and clustered line, where there were no point distinction nor the idea that the amount of points dictates the size of the bar. Since OrderWarp will only user test ordinal data, this transition was set aside. Figure 3.13 shows this transition's alternatives for both types of data.



(a) Ordinal data



(b) Quantitative data

Figure 3.13: Cleveland plot to stacked bars/bar chart's Stacking dots transition

Pilot Lines: Pilot lines is a variation of the once again Pereira et al. [23]'s tested pilot lines. Because instanced meshes are a collection of points rather than just one, it is not possible, without affecting the execution time, to change the size of a single point, and thus it is impossible to replicate the transition. To solve this problem, our variation of this transition, gradually scales down the instanced meshes' height, until the first third of the transition. By scaling the height to zero, the dots convey the idea that they are being merged into a single point. However, this is only the case for quantitative data, as the ordinal values follow a line and the scaling will almost be unperceivable.

After the points disappear (scale at zero), a single rectangle is created with the size the dots would occupy in the next idiom. Once these rectangles hit the idiom boundary, just like stacking dots, they enlarge a horizontal bar that will "pushed" to the next idiom.

Video: `https://youtu.be/Al1veEQx2gg`

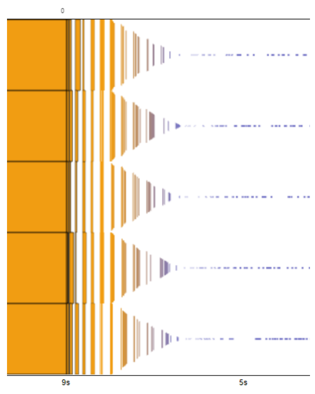


Figure 3.14: Cleveland plot to ordinal line chart's Pilot Lines

3.8.3 Heatmap

3.8.3.A Heatmap to ordinal line chart

Line Morphing: The expelled cells from the heatmap idiom will morph into the shape the cell will have in the ordinal line chart, by gradually reducing their size to the ordinal line chart's line thickness, whilst also changing to the line's color and translating the edges of the rectangle to their points in the next idiom by rotating the element. If the cell's timespan is smaller than the line's timespan, then the cell won't represent the totality of the line, but the proportion that its timespan represents in the line. This gradual morphing creates a transformation of the previous elements to new ones, which eases the visual transition. The representation of these elements in proportions clarifies the idea of the data being grouped.

Video: `https://youtu.be/poE7jPwr_4Y`

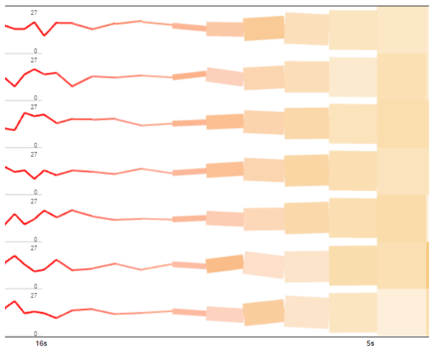
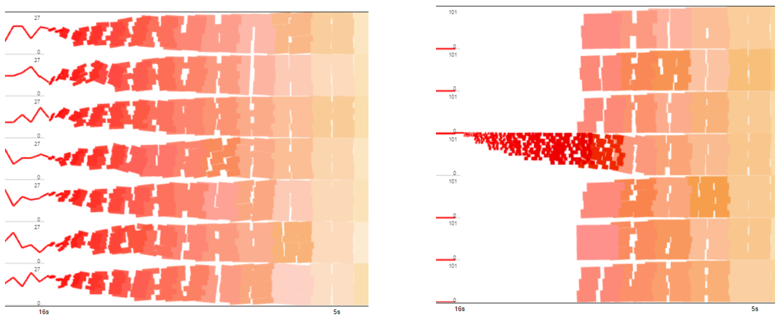


Figure 3.15: Heatmap to ordinal line chart's Line Morphing

Squares: In order to understand the quantity of points inside the expelled cell which will translate to a higher point position in the ordinal line chart, in this transition the cell divides itself in smaller squares proportional to the total received points in that interval. This way, the former color channel of the cell is now perceived as the quantity of squares. These squares initially spread out to understand their quantity, and then they gradually travel to the position they will occupy inside the line, as well as gradually transforming their color and rotation to the line's.

Video: https://youtu.be/yteTrr0Mm7c



(a) Similiar Distribution **(b)** Single Value

Figure 3.16: Heatmap to ordinal line chart's Squares transition

3.8.3.B Heatmap to stacked bars/bar chart

Stacking Bars: The cells gradually reduce size to the length they will occupy inside the next idiom, while joining the other cells that represent data inside the same bin. Close to the transition's boundary and when the data inside the bin is represented by what seems a single bar, the bar's outline gradually appears and when it hits the idiom's boundary it is "pushed" to the next idiom, as described in the Cleveland plot to stacked bars/bar chart's Stacking dots 3.8.2.B. The grouping of data is perceived when the bars stack on top of each other creating a new and larger one, and in doing so, the transitions eases the transformation between the elements of both idioms.

Video: https://youtu.be/soUzq0LncLI

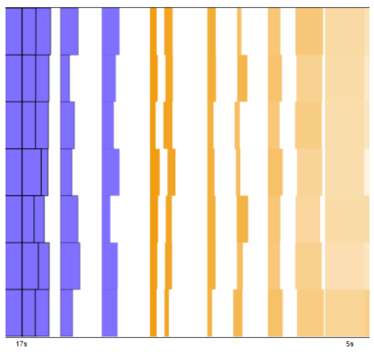


Figure 3.17: Heatmap to stacked bars/bar chart's Stacking Bars

Squares: Similar to heatmap to ordinal line chart's *Squares* 3.8.3.A, the cells are divided in smaller squares that represent its proportion of points, but in this transition the points will expand the size of a bar, created every time the respective bin is generated, before it is, once again, "pushed" to the next idiom.

Video: `https://youtu.be/zXNOP3QWmcQ`

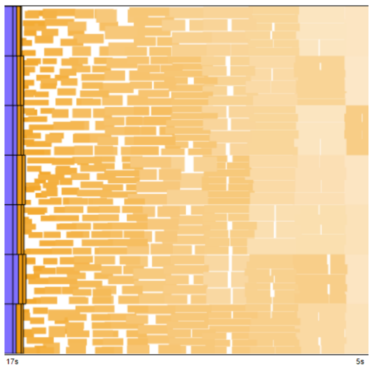


Figure 3.18: Heatmap to stacked bars/bar chart's Squares

3.8.4 Ordinal Line chart

3.8.4.A Ordinal Line chart to heatmap

Cell Morphing: The lines that leave the line chart progressively morph into the cells they will represent in the heatmap, by changing its thickness and color of the line while vertical straightening the line. Each line will, once again, only visually represent the proportion of cell length that the line's timespan corresponds to inside the cell's timespan, so multiple lines could make a single cell. The idea is the same as in the previous Morphing transition, where there is an ease on the transformation between idioms and the proportions imply the grouping of data.

Video: `https://youtu.be/ewBNThdRwqk`

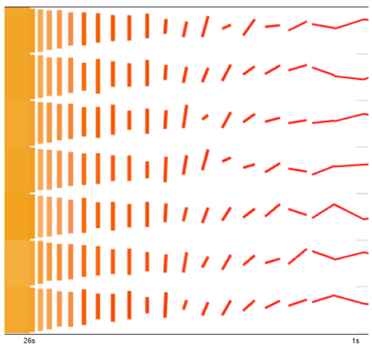


Figure 3.19: Ordinal line chart to heatmap's Cell Morphing

Line squeeze: In this transition the lines will move towards the boundary while horizontally straightening, and after they reach their position in the heatmap, they increase in height. If the cell's timespan is greater than the line's timespan, multiple lines will create a single cell, just like Cell Morphing. The idea of the transition is to form one line with the all the segments that will represent the cell, and upon reaching the end of the transition, it seems as if the line got squeezed against the boundary to form a cell.

Video: https://youtu.be/deEI8w6J6LA

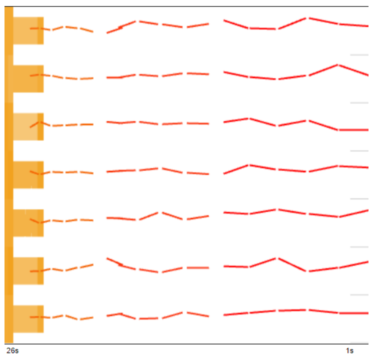


Figure 3.20: Ordinal line chart to heatmap's Line squeeze

3.8.4.B Ordinal Line chart to stacked bars/bar chart

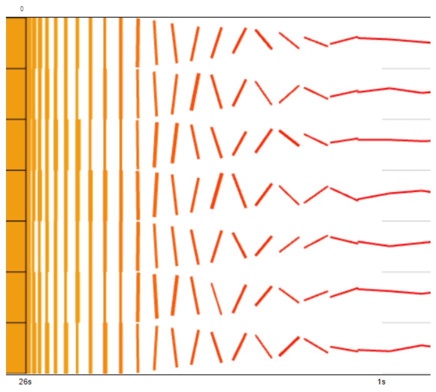
Stacking Bars: The lines from the ordinal line chart rotate until they straighten vertically, at the same time the lines' width increases until it reaches the respective size in the next idiom. Just like heatmap's to stacked bars/bar chart stacking bars 3.8.3.B the recently modified lines will stack, forming larger bars that, upon reaching the idiom's boundary, are "pushed" to the next idiom.

An important difference between this transition and both existing Stacking bars, Heatmap to stacked bars/bar chart 3.8.3.B and Stacked bars to bar chart 3.8.5, is the elements used in them. Since this transition uses lines instead of rectangles, it is unable to form borders around the produced stack bar, which could lead to distinction complications between these bars if they happen to overlap each other.

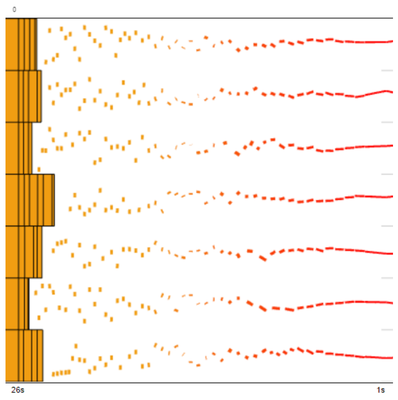
Video: https://youtu.be/FB2hwtBnCdQ

Dissolving Lines: When the lines reach this transition, they are continuously split apart, making it seem like the lines are dissolving into smaller pieces. These "dissolved" pieces move towards the left boundary where, upon reaching the bar that represents its bin, will increase the size of said bar. The bar is, then again, "pushed" to the next idiom. The idea behind this transition is very similar to the Squares transitions where the element gets split after entering the idiom, while here the element splits while entering it.

Video: https://youtu.be/esOtMnamziE



(a) Stacking Bars



(b) Dissolving Lines

Figure 3.21: Ordinal line chart to bar chart's transitions

3.8.5 Stacked bars

Because, as detailed in the introduction of section 3.8, stacked bars do not expel any elements out of its idiom, for every transition beginning with this idiom, an element is created to represent the expelled bin. This idiom works with rectangles as their visual elements and for that reason its transitions have the same functionality as the heatmap's transitions, with the difference being the creation of these elements. So, for both line chart and bar chart the transitions are the same as in 3.8.3.A and 3.8.3.B, it is only necessary to study the transitions for the heatmap idiom.

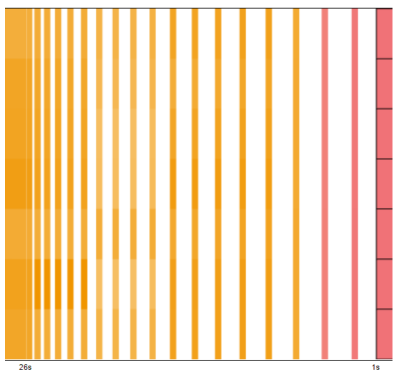
3.8.5.A Stacked bars to heatmap

Cell Morphing: This transition behaves the same way as the line chart's Cell Morphing 3.8.4.A, only this time there is no need to change the rotation of the element since it already resembles the cells. The rectangles created to represent the data leaving the stacked bars will represent a proportion of the cell, depending on the time that rectangle represents in the new cell's timespan. To achieve the cells form, the rectangles must change their colors and width.

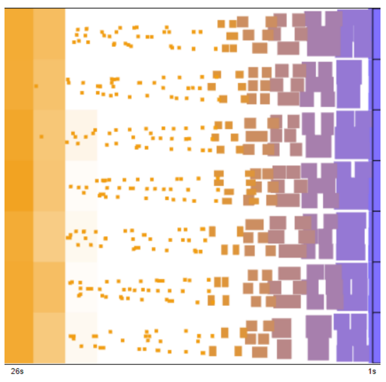
Video: https://youtu.be/t1XxkN3PYS0

Squares: In this transition, we apply a similar approach as the Squares transitions, the difference being that instead of increasing the size of a bar, the cells are created in their future positions on the heatmap with zero **saturation** so that they begin invisible, and then, the squares increase the cell's saturation upon reaching their position, whilst the cells move at the same pace they would in the heatmap.

Video: https://youtu.be/UPKnnhwohYc



(a) Cell Morphing



(b) Squares

Figure 3.22: Stacked bars to heatmap's transitions

4 Evaluation

Contents

It is important to analyze the implemented work on two fronts, the system performance, to understand if the system can handle big data in real-time, and the user testing, to validate the chosen visual idioms and transitions to present ordinal data. In the next sections, the evaluation methods are addressed.

4.1 System performance tests

Performance is a requirement in most non-static visualizations, without it the users' data comprehension will presumably be affected. A poor system performance will most likely be represented by visualization frames freezing and/or frame leaps, and in worst-case scenario in a system crash.

The performance's speed in dealing with both processing the data and rendering the visual elements is one of the most important aspects, and we measure it by the amount of fps. If the number of fps is too low, it means the system is taking an excessive amount of time displaying the required frame, and the lower the number, the more leaps and frame freezes will occur. The value in which the human eye is able to distinguish between frames is considered to be 24 fps yet for OrderWarp the lower limit set for a fluid visualization was 30 fps.

The second measure of performance, required to evaluate the system, is the amount of used *memory*, or in other words the space necessary to store all the needed data structures to ensure the functioning of the system. The retrieval of this measure is specific to the used browser, and in OrderWarp's case it was the by Google Chrome's memory variables, which are only obtained by starting the browser with a specific flag. Different browsers use different amounts of memory, hence the studied measure will not be the exact dimensions of memory, but rather how they develop with time.

OrderWarp is always prepared to received new data and is also committed to run indefinitely, which means that to ensure the system's duration, both described measures have to evolve into a constant limit. If not, after a period of time, the visualization would not be able to maintain its functionality, and would eventually crash.

All the performance tests were executed using Google Chrome version 94.0.4606.81 in a Windows 10 environment with an Intel(R) Core(TM) i5-8400 CPU @2.80GHz, 16Gb of Random-Acess Memory (RAM) memory in a 1920x1080 resolution, with NVIDIA GeForce GTX 1660 GPU.

4.1.1 Results

The results presented below were products of the two tests of the visualization using three idioms. In the first test the right most **(A)** idiom was the Cleveland plot, followed by an ordinal line chart and finally a bar chart, with Growing Bars and Stacking bars transitions respectively. In the second, the visualization **(B)** was composed of a heatmap, a stacked bars and a final bar chart, separated by two Stacking Bars transitions. The idioms positions were chosen based on the theorized most suitable positions. The idiom combinations, as well as their transitions, were chosen arbitrarily to test all the idioms since their functionality is independent of the other idioms. The bar chart idiom is repeated on both tests to ensure that the visualization is being tested with all the data in the dataset. For both visualizations, the idioms corresponding to each other's positions had the same width of 30% (420x400) of the full length of the visualization and the same timespan, and so did the transitions with the remaining 5% for each (70x400).

The transitions themselves also impact the performance of the visualization and therefore were also subjects of testing. The evaluation of the best performing transitions, only makes sense if the same measures are applied between the same combination's transitions. On this test, the same timespans used for each idiom in the combination are repeated on the next transition test. In doing so the same timespan of the transition is ensured. Furthermore, the same widths of the idioms are kept, each representing 40% of the visualization's span, and resulting in a remaining 20% width for the transition (280x400).

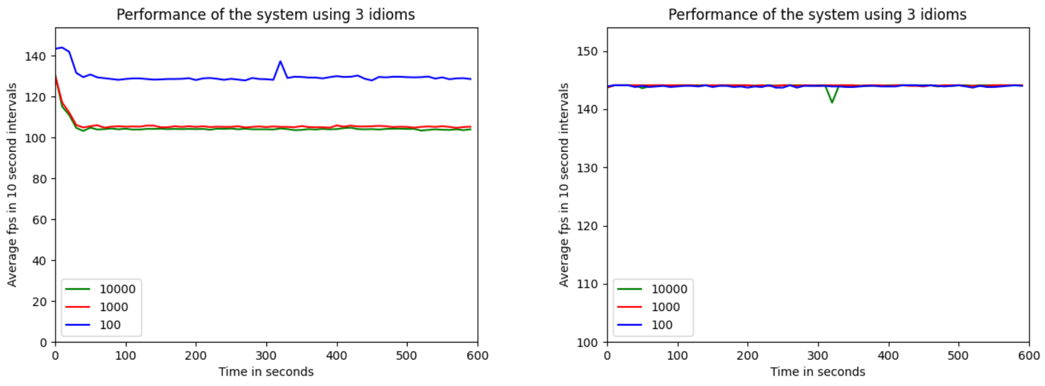
All the information retrieved in the tests was the result of a 10-minute period, where data was saved for analysis every 10 seconds. This process was repeated for 100, 1000, and 10000 data points per second to observe the system's response to different data flows. The most important analysis on these performance tests is whether the system is able to maintain its longevity, and although 10 minutes is a small timeframe in a visualization that as the goal of running infinitely, it is enough to study the evolution of both the fps and memory usage of the system.

4.1.1.A Frames-per-second

The value for the fps is computed by counting the number of updates of the system, whose last step is to render the current frame, during a 10-second period and then upon its end, divides the value by a factor of 10 and stores it. The resulting value will be an average of the fps on the interval, which ensures the data does not show punctual frame spikes. Furthermore, retrieving the values only once every 10 seconds ensures that the performance is not compromised by communication procedures.

For the first two tests, the results are depicted in figure 4.1(a) and figure 4.5(b), respectively. Upon analyzing the figures, it is easy to infer that the system's goal of the lower limit for the fps was accomplished distinctively, with the lowest fps value reached being 103.2 and an average value of 104.8 $\pm$ 3.8 for the highest data flow in visualization A, which is the lowest performing visualization out of the two. Analyzing its evolution, it is possible to conclude that the system's performance stabilizes quickly after the initial drop for all three data flows. The second test's visualization performs better than the first's, as expected, since it does not have the Cleveland plot idiom, which is the creator of the biggest amount of representation elements. The number of visual elements in visualization B is always the same for every data flow, and that is why, as seen in figure 4.5(b) the values are almost identical.

Regarding the transitions' tests, just like before, all the combinations stabilize in a certain value of fps, and unsurprisingly the transitions which hold the lowest values of this measure are the ones which



(a) A - Cleveland plot, ordinal line chart and bar chart

(b) B - Heatmap, Stacked bars, bar chart

Figure 4.1: fps's evolution through time results for the system's performance tests

contain more visual elements, such as the Squares transitions, over the ones with less, such as the Morphing transitions. However, even the transitions with more elements were designed to create the elements according to proportions of records of the ordinal values, instead of depending on the amount of data. This is visible when comparing the different data flows 4.2 which hold approximately the same values of fps regardless of the magnitude of the flow. This is not the case for transitions starting in Cleveland plot which generate elements the same way as the idiom, thus creating elements according to the data quantities in the transition.

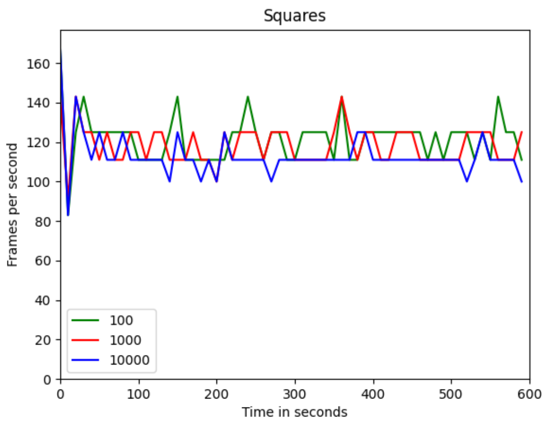


Figure 4.2: Values are approximately the same in each data flow, since the transition creates proportional amounts of elements to the amount of records in the visual representation, resulting in the same number of elements for different data flows

There was one exception for the Pilot lines transitions, which saw a decline below the 30 fps limit for the 1000 and 10000 data flows, see in figure 4.3. **Which means that these transition is not suitable for higher data flows, and should either be rethought in terms of its implementation or its width should be smaller for these flows.**

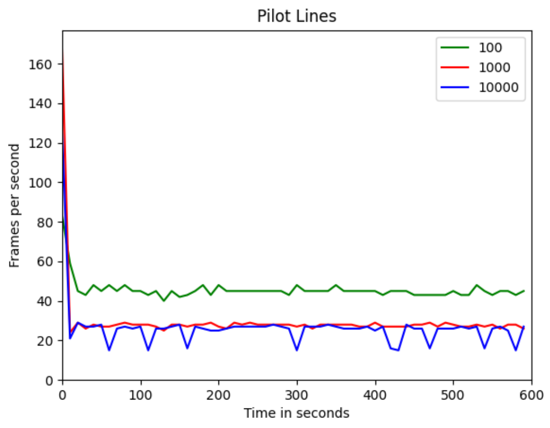


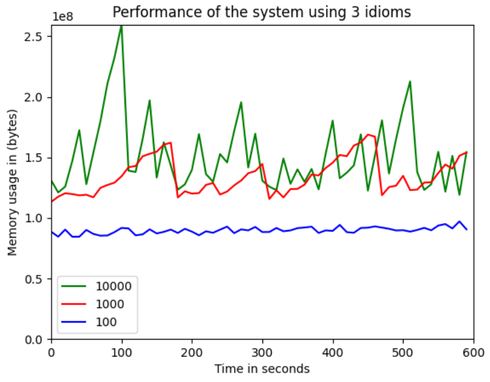
Figure 4.3: fps evolution for Cleveland plot to Bar chart Pilot lines' transition, for different data flows

4.1.1.B Memory

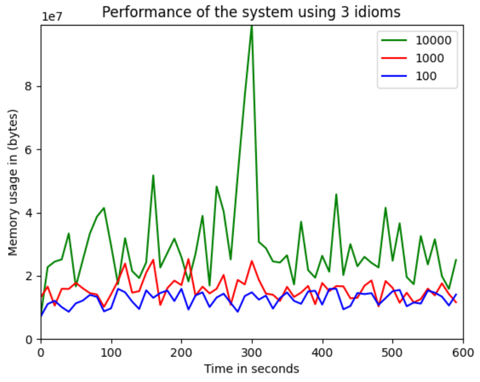
Oppositely to the fps measure, on the used memory it is not common to find spikes on the amount of stored data, especially since this measure does not depend on the computer components. For this reason the resulting showed memory is retrieved on the moment, instead of an average.

For the first two tests, and upon studying figures 4.4(a) and 4.4(b), it was quite clear, other than the fact that higher data flows held the highest amount of memory stored as expected, that this increase in the stored amounts was also accompanied by the biggest leaps in the magnitude of these amounts. Other than that, both alternatives kept constant amounts of memory in the long run, and so it is safe to assume that the memory usage does not compromise the system's performance in terms of time, therefore allowing the visualization's infinite run. Furthermore, the figures allow the analysis of the cyclic process of freeing data, more prominent in higher data flows, which is probably related to the exiting of the bins from the transition onto the next idiom, that, as described in 3.6, summarize the data inside the bin to variables when it reaches the end of the transition, and in this process releases all the data that is no longer necessary after computing the variables.

No surprising information was discovered when studying the memory usage in the transitions. The transitions who hold the largest quantities of data stored, were the ones which required more data structures to represent their data, which, again, implies that more visual representative elements in the transition will lead to higher memory usage. Two examples can be seen in figure 4.5.



(a) A - Cleveland plot, ordinal line chart and bar chart



(b) B - Heatmap, Stacked bars, bar chart

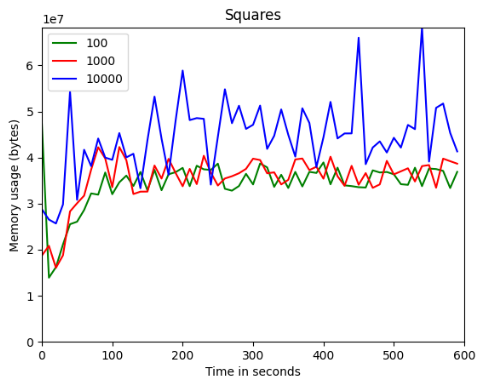
Figure 4.4: Memory usage evolution through time results for the system's performance tests

4.1.2 Discussion

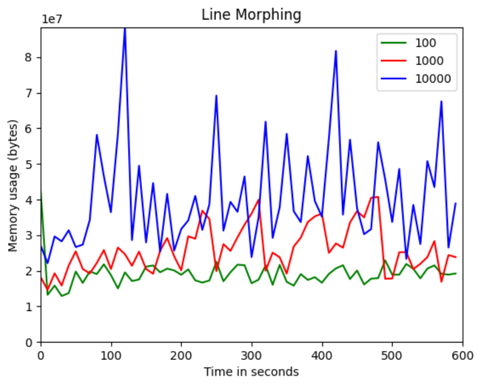
The purpose of this evaluation chapter was to analyze if the goals set in section 1.1 were met. OrderWarp's initial phase migrated VisMillion's system to WebGL technologies, and so, the first required evaluation was the study of the new architecture's performance. In this aspect, the system holds good results, with a balanced visualization of three idioms and two transitions, maintaining more than 100 fps in a 10-minute timeframe window. Memory-wise, the system's stored information expectedly increased with bigger streaming flows, but still was able to secure a constant evolution. Both measures weighted together validate the system's capacity to run indefinitely and support large orders of data flow magnitudes.

The study of the transitions' performance showed that these representations impact the system's overall performance, even though for most cases this impact is not enough to lower the value of fps below 30. This does not mean the transitions can be implemented without thinking of performance, since it was clear that for transitions with more visual elements, the average value of fps was considerably lower than transition with fewer elements, and the opposite happened for memory usage.

Since the tested data flows reached 10000, the dataset generator surpassed the Pires et al. [6]'s limitation of 1000 data points per second.



(a) Stacked bars to ordinal line chart's Squares



(b) Stacked bars to ordinal line chart's Line Morphing

Figure 4.5: The Squares transition's line chart presents a similar evolution as the Line Morphing's, the difference is in the amount of used memory

4.2 User testing

To evaluate our idioms and transitions, our user study was conducted via questionnaires created through Google forms[1], since the tool allows easy and dynamic creation of forms, as well as its analysis, and it covered all the necessary types of questions required.

On total, there were 13 questionnaires, an initial user profile questionnaire, followed by one for each presented combination of idioms addressed in section 3.8. Each user could complete any amount of different testing questionnaires out of all 12, throughout a two-week period, without the need to answer that amount in one session, since that could take a considerable amount of time. For these reasons, the questionnaires were designed to be completed online, with the added benefit that the tests could reach more people and there is not a need for the presence of a guide. The tests were assigned to each person via a given ID, and with a 12 by 12 *Latin square* distribution, to ensure the order in which the questionnaires were given did not bias the results.

The questionnaires' questions were based on videos pre-recorded of the visualization in the respective idioms and transitions. This alternative was chosen over allowing the user to interact with the interface to preserve users time, by only focusing on the visualization instead of unnecessary procedures of choosing the correct visualizations, their timespan and their width. Since the interaction of the user is not the test subject of this work, videos are the most effective way to show what is being tested.

[1] https://www.google.com/forms/about/

4.2.1 Questionnaires

The 12 questionnaires all followed the same structure, and an example of one can be seen in A. The questionnaires were written in Portuguese, as this was the target user's native language. The users are first greeted with a description of what they will be subject to and the purpose of the study, as well as an explanation of the OrderWarp system. They are then presented with an edited video [2] that allows the visualization and understanding of the system, through detailed writing and covering the different parts of OrderWarp step by step. All the videos referenced in this section were recorded in a visualization with a starting data flow of 100 data points per second.

Following this, and since each questionnaire studies one idiom combination, the next questionnaire sections focus the analysis in each idiom in the combination individually. These sections start off with another video showing only the functioning of the target idiom, accompanied by a series of objective questions that allow the quantification of the idiom's effectiveness. The next set of questions are identical for all the idioms, and ask - *"How confident are you on the previous answers?"* and - *"How many times did you have to rewatch the video?"*. These questions give insights on the complexity of the representations, since visualization's goal is not only to accurately display the information, but also, ensuring the user they understood the information present in it. Finally, the user is encouraged to identify any problems with the idiom and/or offer suggestions that could improve the visualization.

The next two questionnaire sections serve to evaluate the two proposed transitions for the studied idiom combination on the questionnaire. It followed the same logic as the idiom section by beginning with a video, which in this case is the same presented after each transition's description in section 3.8. Next in line are the objective questions specific for each transition, and the same user confidence and video review questions, with the open question for problems or suggestions. In the second transition, the last question asks the user's preferred transition between both given solutions.

Since one user can do more than one questionnaire and since these have multiple coincident idioms, because they represent its combinations, it is not in the analysis best interest for the user to repeat answers for what has already been tested, otherwise the results might be compromised by the answer repetition. This is the reason why users, before entering the idioms section, are asked if they already answered questions about that idiom on another of OrderWarp's questionnaires. If they have, they skip the questionnaire section for the following one, and in doing so there will not be any repeated questions and the users' time spent answering the questionnaire is shortened.

[2] https://youtu.be/RcwU2Bc3KT8

4.2.2 Participants

Twenty-four people between the ages of 18 to 30 (91.7%) and 50 to 60 (8.3%) responded to the above described questionnaires. Out of the 24, two thirds (16 users) identified as male, 29.2% as female and the remaining person identified as non-binary. Overall, the users' experience with information visualizations was considerably dispersed, with a third (33.3%) using it only occasionally and another third (37.5%) using it every week, 12.5% uses it every day, and equal proportions (8.3%) for never using them and using them once a month. As expected, almost every user knew the concept of the bar chart, line chart, and scatter plot with 95.8%, 83.3% and 75% respectively, and a smaller, yet considerable proportion of users being familiar with a heatmap (58.3%) and stacked bars (54.2%).

4.2.3 Results

4.2.3.A Idioms

The efficiency of the idioms was determined by the correct answers given by the users in correct or incorrect questions, which were based on the tasks that each idiom attempted to address. All following idioms present their results in a table with objective questions represented in percentage values, and the Likert scales questions showing both the median and the interquartile range (IQR).

Cleveland plot

For the **Cleveland plot** idiom, the displayed video [3] showed an almost binomial distribution on one of the values, and then suddenly an agglomeration of points emerges on a distinct ordinal value.

The first question asked - *"For which value are there more points during the most part of the video?"* (Q1) and it targeted the user's distinction of the distribution of the ordinal values. The next question focused on the recognition of agglomerations and asked - *"Was there a sudden agglomeration at any moment in time"* (Q2), followed by identification of which values were affected by the agglomeration with - *"If you answered yes, for which ordinal value(s)"* (Q3). The results are shown in table 4.1 alongside the average confidence (QC) and average rewatched times (QR) mentioned above.

Question	Q1	Q2	Q3	QC	QR
Results	100%	100%	83.3%	4 (1)	2 (2)

Table 4.1: Cleveland plot effectiveness results

Q1 proved the visualization to be quite effective since all the users accurately identified the most common value in the idiom. Q2 was, once again, identified by every user. However, a lower portion, yet still substantial (83%), was able to point out, in Q3, which values were affected by the agglomeration. One user's observation considered the idiom the most suitable to represent the arrival of new data. QC

[3] https://youtu.be/uKTqZ0lQWDg

got a median value of 4 with the first and third quartile of 4 and 5, respectively, and QR got a median value of 2 with the first and third quartile of 1 and 3, which indicates that the answers were given with confidence and without the use of many video repetitions.

Heatmap

Heatmap's video [4] followed the same logic as the Cleveland plot's, with a different dataset and agglomeration instant. Since the analyzed tasks are identical, the three first questions were the same, adapted to the new dataset. The idiom's questionnaire presented two extra question, the first asked - *"Is there any ordinal value without any point or with very few dots in any moment of the video?"* (Q4), and it researched the comprehension of the lack of records in the idiom. The second asked - *"How perceptible were the changes of the values in the scale"* (Q5), and its purpose was to understand the animation's effectiveness on the scale's value update with the help of a 1 to 5 Likert scale.

Question	Q1	Q2	Q3	Q4	Q5 (median)	QC (median)	QR (median)
Results	100%	95.7%	87.0%	82.6%	4 (1)	4 (1)	2 (1)

Table 4.2: Heatmap effectiveness results

The results can be found in table 4.2. The three first questions' results showed similar findings to Cleveland plot's, with a slight decrease in agglomeration identification balanced by a small increase in the recognition of the affected values. The idiom's scale was perceptible for the users, with 82.6% responding correctly when no values were shown, and when modifications to the scale's limits occurred, since the users found these changes very perceptible, with a narrow distribution around a median value of 4. This idiom had the same distribution as Cleveland plot with the same quartiles and median values, and a smaller distribution in rewatch times (first quartile 1, median 2, third quartile 2).

Ordinal line chart

The test of the ordinal line chart was slightly different. The video's [5] dataset was designed so that one of the values presented a larger amount of points tested with the same question as Cleveland plot's Q1, which also allows us to understand if the comparison of ordinal values is possible with the idiom. Two of the other values tested the analysis of the idiom's evolution when asked - *"Which value(s) continually increased or decreased with time?"* (Q2), with one irregularly increasing linearly and the other decreasing. Lastly, one of the ordinal values' line showed a sudden spike, and it was the user's job to identify the maximum amount of points hit by that value in the question - *"What was the approximate maximum value hit by the XL value during the video?"* (Q3). The results can be found in table 4.3.

The results were positive for the first couple of questions, with 95.7% and 78.3% respectively. However, the third identified a problem on either the logic or the implementation of the scale, with only 56.5%

[4] https://youtu.be/2KNuAX6ZmBI
[5] https://youtu.be/ozKu4JvQ1-A

Question	Q1	Q2	Q3	QC (median)	QR (median)
Results	95.7%	78.3%	56.5%	4 (1.5)	2 (2.5)

Table 4.3: Ordinal line chart effectiveness results

finding the correct answer. The confidence in the answers was positive, with a slightly wider distribution than the previous idiom, the difference being half a point in the first quartile of 3.5, but the number of needed views for the video increase considerably with a first quartile of 3.5, median of 2, and lower quartile of 1.

Stacked bars

As described before in section 3.7.4, the idiom's goal is to understand the proportions of points arrived in different time intervals called Eras, which were the test subject of the questions. Stacked bars video [6] showed a normal distribution of the data through various Eras, and then, similarly to Cleveland plot's, a sudden agglomeration on the S value.

The first question's (Q1) objective was to study the comparison between Eras when asked - *"What was the Era that received, more points, in more ordinal values?"*. The second question's (Q2) asked - *"The previous question was not correct for one ordinal value, which one?"*, and its goal was to identify an outlier Era in an ordinal value, that received an agglomeration of records much greater than the other values.

The third question - *"What is the approximate percentage that the green Era occupies in the L value at the end of the video?"* (Q3), served to evaluate the identification of the bar's percentage values. In the last question, when asked the true or false question - *"The first and last Era are completely represented?"* (Q4), the users had to understand that these two Eras' timespans were not fully inside the idiom's timespan.

Question	Q1	Q2	Q3	Q4	QC (median)	QR (median)
Results	72.7%	59.1%	86.4%	20.0%	4 (1.75)	2 (2)

Table 4.4: Stacked bars efficiency results

This idiom presented somewhat of negative results. 72.7% found the most common Era for most ordinal values, but 40.9% said that it was not clear in which Era this did not occur. Furthermore, the identification of the individual values was clear for most users, with 86.4% finding the correct answer, however, only 20.0% understood that the Eras were not fully represented in the idiom. The confidence question got lower results, at a median value of 4, yet 3 and 4.5 for first and third quartile. The users had to rewatch the video more times to answer QR as well, with 1, 2 and 3 for first, median and third quartile values.

[6]https://youtu.be/ZVYVDIICzAY

Bar chart

Bar chart's video [7] demonstrates a simple binomial distribution of points around the L value. The idiom was then tested with just two questions, the first, once more, tested the efficiency of the idiom on comparison by asking - *"Which ordinal value has more points at the end of the video"*, and the second (Q2) asked to identify the number of records in a given ordinal value, with the question - *"How many points are approximately represented in the XL value bar at the end of the video?"*. The results can be found in table 4.5.

Question	Q1	Q2	QC (median)	QR (median)
Results	100%	88.9%	5 (0)	1 (1)

Table 4.5: Bar chart efficiency results

Both questions showed good results in the two measures with 100% and 88.9% correct answers respectively. These positive results are also supported by a high confidence and a very narrow answer distribution at a median value of 5, just like both quartiles. The rewatch distribution was low, with a median and first quartile of 1 and third quartile of 2.

An additional analysis on the comparison of answering confidence values for each idiom and the amount of times the users had to rewatch the videos, represented in the box plots on figure 4.6, shows the levels of comfort the users have on visualizing information in each idiom.

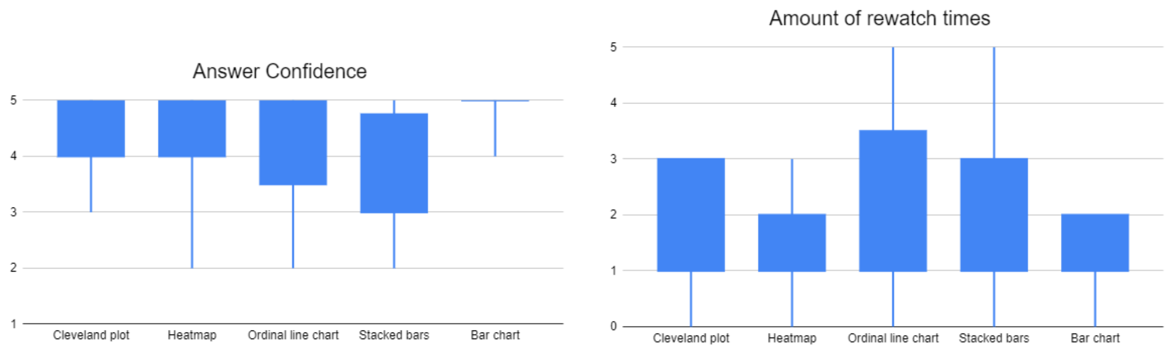


Figure 4.6: Comparison of the answer confidence and amount of rewatch times between idioms

4.2.3.B Transitions

The goal in testing the transitions is, not only, testing the efficiency of each transition, but also evaluating the best transition between two idioms. In OrderWarp's questionnaires, all transitions were tested using the same questions, with every user testing both transitions according to three different measures:

[7] https://youtu.be/FrvASuI2RE4

- Aggregation: If the transition is able to represent the binning of the data to the next idiom. This is tested by asking the user to evaluate the aggregation clarity in said transition, using a 1 to 5 Likert scale.
- Fluidity: If the transition is fluid, i.e. if it seems a continuous transformation between the visual elements from the first idiom to the next. This measure is, once again, evaluated by asking the user to attribute a value from 1 to 5 in terms of fluidity.
- Logic: If the logic of the transition is clear to the user. This is evaluated by asking the user to choose one of three options of plausible logic descriptions and analyzing the amount of correct responses.

To study the first and second measures, each combination of transitions underwent a *Wilcoxon signed-rank test* except for one transition that found a non-normal difference median distribution, and therefore the statistical test used was the *sign test*. For the logic measure, since the results are dichotomous instead of ordinal, the applied test was the *McNemar's Test*, however no statistically significant differences were found in any of the combinations, and for this reason will not be shown in the following results.

Additionally, the last question asks the user's preference of the two studied transitions, useful as an overall measure of the visual appeal of each transition. The abovementioned statistical tests and this last question are only relevant for transition comparison, and as explained in section 3.8.2.B, both combinations of transition starting from Cleveland plot and ending in either stacked bars or bar chart, only have one testable transition since the Stacking dots transition was taken out of the user testing phase, and therefore can only be evaluated individually.

Cleveland plot

The first tested combination was between Cleveland plot and ordinal line chart's, with both **Growing Bars** and **Grouped Dots**' results shown in figure 4.7. Individually, Growing bars presented a very positive aggregation and fluidity with the same median result of 5 (1), with 95% understanding the logic behind the transition. On the other hand, Grouped dots got a median of 4 (1) in aggregation and just 3 (2) in fluidity, and just 72.7% of the users understood the logic of the transition. In the end, an unsurprising 90.9% of users preferred Growing bars over Grouped Dots.

Comparing both transitions, for the aggregation variable, this combination was the exception cited above. Using the sign test for the aggregation, out of the 22 users, 13 found the second transition worst in representing the aggregation of data when compared with the first, and the other 9 considered it the same as the former. For the fluidity, the Wilcoxon signed-rank test could be used, and here, 18 users found the second transition worse than the first in terms of visual fluidity, 3 tied the values and one found it better than the first transition. There was a statistically significant median decrease in the attributed

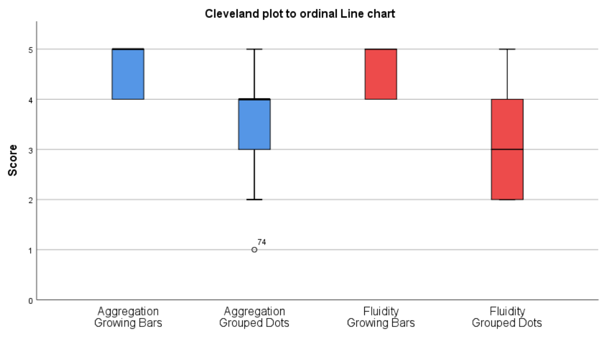


Figure 4.7: Cleveland plots to ordinal line chart Growing Bars vs Grouped Dots

value of representation of the aggregation from, as well as for fluidity, with $p < .0005$ for both measures and $z = -3.678$ for the second. These results imply that for both measures the first transition is a better representation, and since the understanding of the logic does not play a part due to its statistically insignificant results, the study concludes that **Growing Bars is the most suitable transition out of the two, for the Cleveland plot to ordinal line chart combination.**

Moving on to Cleveland plot to Stacked Bars, the only studied transition was **Pilot lines**, and this transition received a positive user evaluation, with 5 (1) and 4 (1) for aggregation and fluidity respectively, and 72.0% accurately understood the logic of the transition. Both distribution can be seen in figure 4.8.

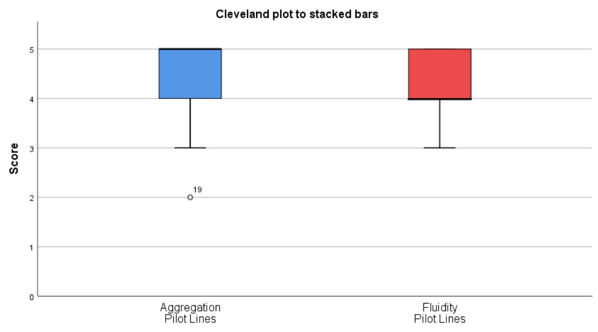


Figure 4.8: Cleveland plot to stacked bars' Pilot lines

The same transition, but for the combination ending in bar chart, received almost the same median aggregation value of 5 (1) and a higher fluidity value of, also, 5 (1). The answer's distribution can be seen in figure 4.9. The logic comprehension also increased to an 86.4%, comparing to the previous combination's Pilot lines.

Heatmap

For Heatmap starting transitions, the first analyzed transitions were **Line Morphing** and **Squares** which end in the ordinal line chart idioms. In figure 4.10 It is observable that Line Morphing's aggre-

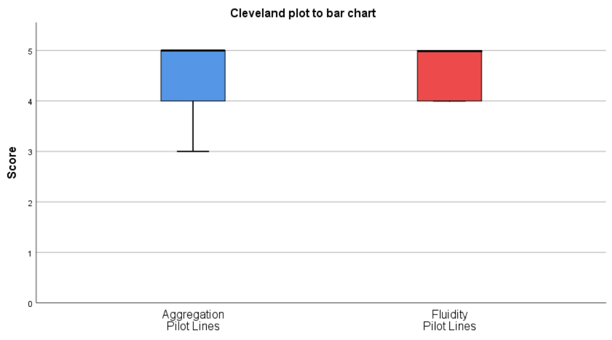


Figure 4.9: Cleveland plot to bar chart's Pilot lines

gation distribution is the same for both transitions, with a median value of 4 (1). In fluidity average, the distribution is more spread out, with the same median 4 (2) for the first transition. Squares's fluidity had a lower distribution than the previous transition, with the median value of 3 (1). Line Morphing's Logic levels can be considered negative with not even half the users comprehending the transition 48%, while Squares' is at 72.0%. Although this last value is higher for Squares, Line Morphing was the most preferred transition by a considerable difference (84.0%).

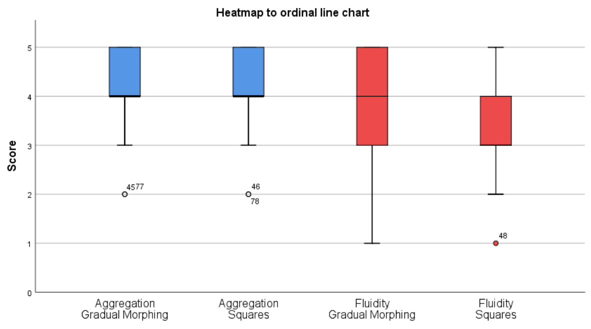


Figure 4.10: Heatmap to ordinal line chart's Line Morphing v Squares

Of the 23 different users, 10 considered the first transition more fluid than the following, just 1 considered the opposite and 12 users found no differences in this measure. For aggregation, no statistically significant differences were found between transitions, $z = 0.378$, $p = 0.705$, however for fluidity there is such significance, with a median decrease and $z = -2.658$, $p = 0.008$. Since the logic and aggregation measures do not find any valuable difference, the transition can only be distinguished by its fluidity, where **Line Morphing resulted in a better representation**.

The second combination addresses the **Stacking Bars** and **Squares** transitions to Stacked bars. The first transition's measures returned a median value of 4 for both measures, with a very spread distribution, with the interquartile range being 2.5 and 2, respectively (Figure 4.11). The second returned a median of 4 (1) and 3 (1.5) for aggregation and fluidity, respectively, returning a more narrow distribution. Logic-wise, 95.7% understood Stacking bars, and 87% understood Squares, with the former being a heavy favorite with 87.0% preference over the latter.

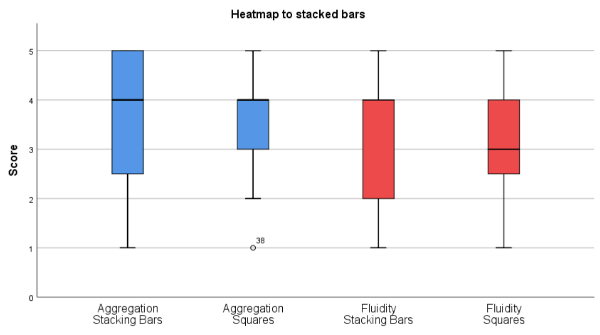


Figure 4.11: Heatmap to stacked bars' Stacking Bars v Squares

Both aggregation and fluidity, do not present a statistically significant difference in the mean, since they hold the statistical values of $z = 0.108$, $p = 0.914$ and, $z = -0.250$, $p = 0.803$ respectively. No statistical significant difference in all the measures means **that the study can not choose a better representation in any of the fronts, and for this reason the preference values decide that Stacking Bars is a more suitable representation**

The last combination presents the same transitions but this time with the ending transition being the bar chart. On these transitions, the median aggregation value rose to 5 (1) and the fluidity remained at 4 (1) for Stacking bars. Squares registered a median aggregation and fluidity value of 4 (2) and 4 (1) respectively, as seen in figure 4.12. The logic measure also increased to 95.7% and 91.3% for each transition, and the preference balanced out to 56.5% of the users preferring Stacking bars.

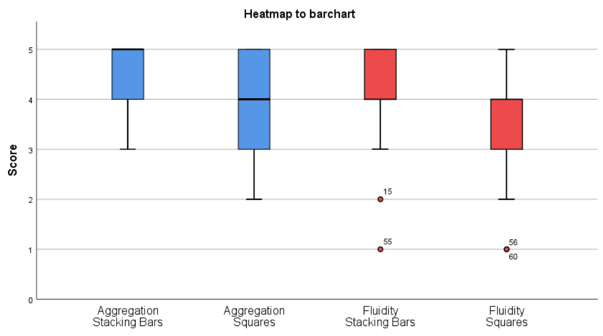


Figure 4.12: Heatmap to bar chart' Stacking Bars v Squares

Of the 23 users, for the aggregation measure, 9 identified the first transition a better aggregation representation, 3 disagreed, and the rest found it to have the same performance. Whilst the fluidity measure, had 11 finding the first transition more fluid than the latter, 1 disagreed, and the rest did not find them different. This means that for both measures, and in opposition to the previous combination, there was a statistically significant decrease in their values, with $z = -2.077$, $p = 0.038$ for aggregation and $z = -2.801$, $p = 0.005$ for fluidity. **These results suggest that the Stacking Bars was the most suitable transition of the combination, also supported by their positive individual results.**

Ordinal line chart

This first two studied transitions with the ordinal line chart as the starting idiom were **Cell Morphing** and **Line Squeeze** to the heatmap idiom. For the first transition, the aggregation and fluidity median values stood at 4 (2) in both cases. The second transition return lower values of 3 (2) and 3 (1) respectively (Figure 4.13). 81.8% of the users comprehended Cell Morphing versus 68.2% in Line Squeeze's case. Once again, unsurprisingly considering the results, the Cell Morphing has a much higher percentage of preference (90.9%) than Line Squeeze (9.1%).

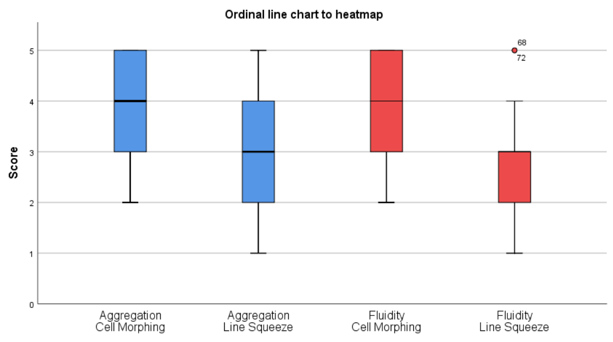


Figure 4.13: Ordinal line chart to heatmap's Cell Morphing v Line Squeeze

Of the 22 users, 14 considered the first transition better to represent an aggregation transition, 2 considered the opposite and 6 tied the representations. For fluidity, the values increase to 16 considering the first transition more fluid, only 1 disagreeing and 5 tied. Both measures found a statistically significant decrease in their values, with $z = -3.051, p = 0.002$ for aggregation and $z = -3.459, p = 0.001$ for fluidity. **Therefore, the Cell Morphing transition can be considered a better visual representation for the ordinal line chart to heatmap combination.**

The next combination, is ordinal line chart to the stacked bars idiom and the transitions are **Dissolving Lines** and **Stacking Bars**. For both transitions and in both measures, the medians were the same value of 4. What distinguished them was their distributions, which can be seen in figure 4.14. Stacking bars aggregation's IQR was a small 0.5, with the first quartile having the same value as the median and a slightly higher 4.5 value of the third quartile, whilst Dissolving Lines had both a lower first quartile and higher third quartile of 3 and 5 (IQR 2). The fluidities were very similar, with half a point difference in the first quartile and the exact same third quartile as the median. Both transitions' are easy to understand, according to the 82.6% and 95.7% results. And lastly, the preference in transitions tends to the second transition by just one user (52.2%).

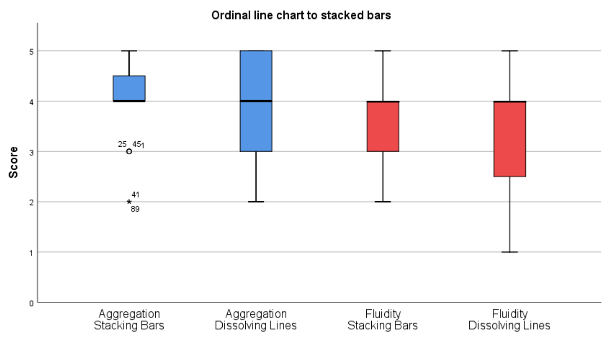


Figure 4.14: Ordinal line chart to stacked bars' Dissolving Lines v Stacking Bars

The balanced preference between transition, is also shown in the fact that for both aggregation and fluidity values no statistically significant difference is found, having, respectively, $z = -0.683$, $p = 0.495$ and $z = -1.020$, $p = 0.308$. Once again, the absence of statistically significant differences means, no transition can be considered a better representation, and their suitability has to be seen by the individual values and weighted by the user's preference, **which ultimately decides the Stacking Bars as more suitable**.

The last combination involving an ordinal line chart, is the similar ending bar chart. This similarity is also present in the results, with the transitions median values in both measures being the same, the difference being a slight increase in the distribution of the Dissolving Lines' measures (Figure 4.15). Both transitions are also easily understandable according to the 95.7% and 87.0% respective values. The preference in this transition shifts slightly more towards Stacking bars than the previous transitions.

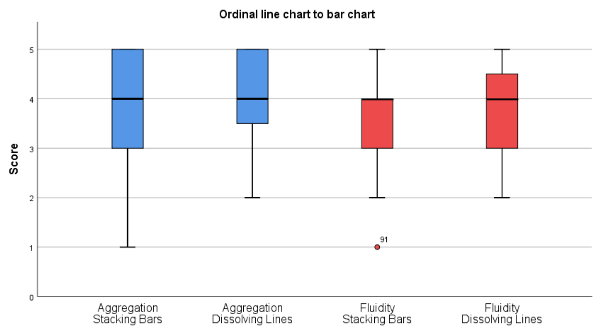


Figure 4.15: Ordinal line chart to bar chart's Dissolving Lines v Stacking Bars

Just like the previous combination, the similarity between both transitions once again shown in the none existing statistically significant difference with $z = 0.000$, $p = 1.000$ for the aggregation measure and $z = -0.638$, $p = 0.523$ for the fluidity. As before, the statistical difference of one transition over the other is non-existing, so the preference is taken into account, this means **Stacking Bars is therefore**

considered more suitable.

Stacked bars

In stacked bars to heatmap, the studied transitions were **Cell Morphing** and **Squares**, and were possibly influenced by stacked bars' poor performance described in section 4.2.3.A, since the results obtained in the different measures were lower when comparing to other transitions. Cell Morphing's values were a median of 3 (1) for both measures, and 72.7% percentage of users understood the transition's logic. Whilst Squares measures secured a median of 4 (1.5) and 4 (1) to the lower quartile. The better results in the Squares transition justifies its preference by the users. The distributions of the two first measures can be analyzed in figure 4.16.

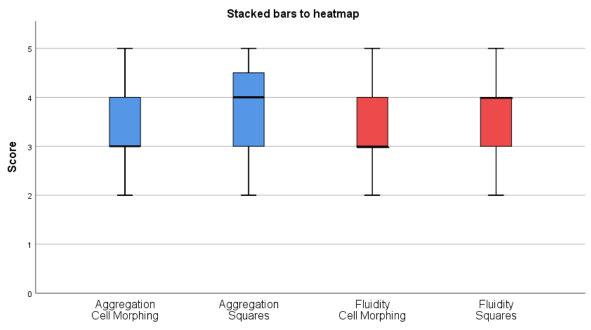


Figure 4.16: Stacked bars to heatmap's Cell Morphing v Squares

For this combination, no statistically significant mean difference occurred for both measures, with p values of $z = 1.340$, $p = 0.180$ for the aggregation measure and $z = 0.200$, $p = 0.842$ for the fluidity measure. **The preference of the Squares transition over Cell Morphing, decides that this transition is picked as most suitable.**

Line Morphing and **Squares** are the next studied transitions, and the preference of the first transition, with 63.6%, over the second is supported in the measures. Line Morphing's median values were 4 (1) and 3 (1) for aggregation and fluidity, respectively. For Squares, the same first two measures received identical values, with the only difference being a slight aggregation distribution increase, as seen in figure 4.17. The understandability of the transition stood at 90.9% for Line Morphing against the 86.4% of Squares. The preference of the users was in Line Morphing's transition with 63.6%.

Similarly to the previous combination, there were no statistically significant mean differences for both measures, $z = 0.237$, $p = 0.813$ and $z = -0.258$, $p = 0.796$ for aggregation and fluidity respectively. As before, the lower individual results question the representations' suitability, and since there is no significant difference for all three measures, only the preference can weight in on the choosing of the transition, **which decides that Line Morphing is more suitable**.

The last studied combination was between stacked bars and bar chart, again using the **Stacking**

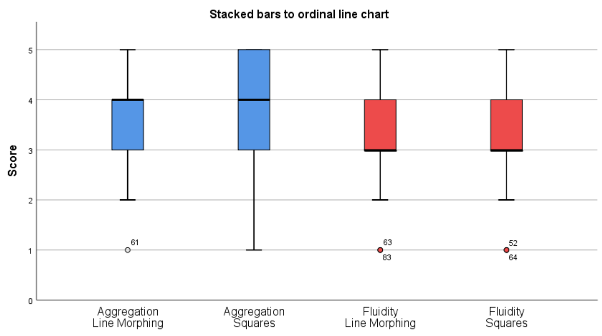


Figure 4.17: Stacked bars to ordinal line chart's Line Morphing v Squares

bars and **Squares** transition. Stacking bars' median aggregation value was 4 (1), the median fluidity was a low 3 (1) out of 5 and the comprehension of the transition's logic was 86.4%. Squares's aggregation median was the same value, however its distribution was concentrated in higher values as seen in figure 4.18. The fluidity distribution was slightly higher for Squares transition, with 3.5 (1). The improvement on the aggregation values from the first transition to the second one seems to justify the preference of the Squares transition over the Stacking bars, with 63.6%.

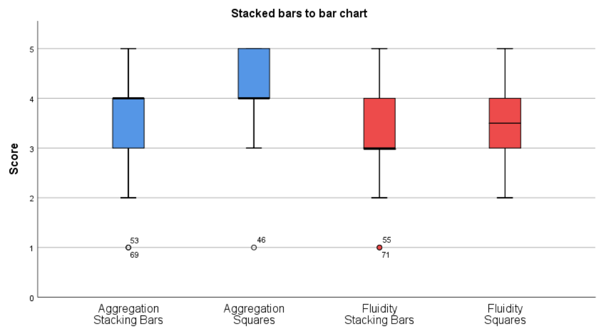


Figure 4.18: Stacked bars to bar chart's Stacking bars v Squares

Of the 22 users, 10 considered the second transition better to represent an aggregation transition, 2 disagreed and 6 tied the representations. Only the aggregation measure found a statistically significant increase in its value when compared to the previous transition. The aggregation's values were $z = 1.990$, $p = 0.047$, while the fluidity's $z = 0.254$, $p = 0.799$ surpassed the significance level. **This means, only the aggregation measure can distinguish the best overall representation between the two transitions in the combination, which, in this case, is the Squares transition**.

Table 4.6 summarizes the chosen transitions for each of the combination of idioms, based on statistical tests when possible and by individual results [8] when not, the latter case transition are identified on the table. A threshold of 3 for both aggregation average and fluidity average was set as the limit of an

[8] Chosen by user preference and individual results

acceptable transition's efficiency [9], has this value represents the neutral limit of an efficiently average transition. All the transitions with lower results are identified on the table.

Idiom	Cleveland plot	Heatmap	Ordinal Line chart	Stacked Bars	Bar chart
Cleveland plot	▨	▨	Growing Bars	Pilot Lines	Pilot Lines
Heatmap	▨	▨	Line Morphing	Stacking Bars [8]	Stacking Bars
Ordinal Line chart	▨	Cell Morphing	▨	Stacking Bars [8]	Stacking Bars [8]
Stacked Bars	▨	Squares [8]	Line Morphing [8] [9]	▨	Squares
Bar chart	▨	▨	▨	▨	▨

Table 4.6: Resulting transitions table

4.2.4 Discussion

The evaluation of the user testing phase addresses the effectiveness of the visual representations in displaying the information correctly. The idioms had positive results for the most part. The heatmap and bar chart showed positive results in all asked questions, with the questions being answered correctly by the most part of the users, proving to be effective representations for ordinal data on the Graceful degradation technique, as well as for quantitative data [6].

Cleveland plot had positive results as well, which proves that the idiom is a good representation of the new data's distribution for ordinal data. This idiom suffers even more visual clutter than the quantitative data's alternative scatter plot, since it only shows data following in one dimension, while the scatter plot disperses the dots according to two axes. This idiom is, therefore, not suitable when the data flow is too high, not only for its drawing challenges which are partially mitigated by the *instanced meshes*, but mainly for its visual clutter, where the dots seem to be just one connected line. For this reason, we suggest changing idiom to heatmap for higher data flow, just like VisMillion' quantitative alternative. Furthermore, the good results in both the Cleveland plot and its transitions, infer that, even with the *instanced meshes* limitations explain in section 3.5, these elements can still provide effective representations for this idiom.

Moreover, the ordinal line chart's approach saw good results in distinguishing the evolution, and comparing between ordinal values, yet the identification of the exact values proved to be challenging for the users. Since the evolution distinction and value comparison outweighs the negative exact value identification, this idiom is also considered effective in ordinal data representation, but it should adjust its scale implementation strategies before being used in a real context.

Oppositely to the previous results, the stacked bars idiom had lower results in all questions, except for proportion value distinction. The idea that the idiom shows proportions instead of actual numbers, was found to be confusing to the users according to their comments. The idea that the Eras are an

[9] Results below given transition efficiency threshold

interval of data which moves with the passage of time, added to the fact the representation bars' width also changed when new data arrived, compromised the data perception of the users. For these reasons, the stacked bars idiom, at least as it stands, can not be considered an adequate idiom and should not be used in a real Graceful degradation visualization.

Finally, in the confidence and video rewatch measures, it is clear that the least known or alternative idioms are more cognitive demanding than the others. On top of that, these idioms are also the ones with worse testing results, with the least known idiom, stacked bars, receiving the worst reviews. The idioms, which presented themselves as modified versions of other known idioms, even though they had less positive results than the identical idioms, were still found to be suitable for ordinal data representation. For these reasons, when designing alternative idioms with other unstudied task analysis, the developers should try and consider modifying known idioms, when they are not a possible solution.

In the transition user evaluation, the best alternatives out of the implemented transitions were found and summarized in table 4.6, however this does not mean there are no better alternatives. The individual evaluations of each transition showed out of the picked transitions which ones were adequate according to a value threshold, with the transitions below the limit being identified. For the most part, the resulting transitions implemented less visual elements in their representation and applied the same logic when that was a possibility, which introduced a familiarity to the transitions. Therefore, when designing transitions, we suggest finding alternatives with fewer elements and familiar to the user, if that is possible, by, for instance, using the suggested transitions in our work.

5 Conclusion

Contents

We conducted a study on the impact of Big data information generated in a streaming context, extending previous works to a study of said domains on ordinal data. To achieve this we first started by studying the state-of-the-art works in the referred fields. It was clear that the dimensions of the datasets compromised the performance of the analyzing systems, both in the computational complexity required to create the visualization, and in the visual clutter generated by over drawing information elements. The found solutions, usually applied aggregation techniques prior to the study of the visualization, which is not possible when the information is yet to be created, as in Streaming visualizations. For these visualizations, many of the works focus was on creating systems where the information was presented in various levels of detail, with the most relevant information being subject to small or no aggregations, and therefore, displayed with more detail, and the least relevant information subject to the opposite.

The goal of this work was to continue the analysis and implementation of the works in VisBig. These studies introduced the Graceful degradation technique, whose system's logic was characterized by a series of sequential and connected visual representations which guides the information through a path where the data passes through various levels of detail while being gradually grouped and displayed in different idioms. The data ends in a final grouping representation, which confirms that the visualization shows all the data since its start. However, this technique was only explored for a single data type, and it is this work's objective to expand the analysis for ordinal data.

For this reason OrderWarp is proposed, a web system that can visualize ordinal data in datasets with large dimensions in real-time with different levels of detail. The system creates an adaptable visualization through the flexible creation and manipulation of idioms and transitions for different time periods and view spans by user given specifications.

The first phase of the project was to migrate the work onto canvas + WebGL technologies and restructure the architecture behind the system. Then, we idealized and implemented representation idioms, specific to ordinal data, which could effectively display the information. The resulting idioms were the same heatmap and bar chart as already studied in VisBig, the alternative representations of Cleveland plot and ordinal line chart to the scatter plot and line chart, respectively, and a new part-to-whole alternative, stacked bars. Since these idioms are connected in different detail levels and different timespans, the transitions between them had to be analyzed as well, to maintain the user's mental map of the data path. For this reason, we present two alternative transitions, whose purpose is to find the best representations for each idiom combination.

The evaluation of OrderWarp focused on ensuring that the proposed architecture allows the system's indefinite run, and identifying the best representations of idioms and their transitions. The performance analysis retrieved the system's average fps and memory usage for 100, 1000 and 10000 data flows, and recorded its evolution through time. On the other hand, the user testing was split into two segments, the study of the idioms individually, and the evaluation of the two transitions per idiom combination.

The performance tests proved the system maintains the representation effectiveness when subject to different data flows, with the studied measures sustaining their performance through time, ensuring the visualization's longevity. In the user testing analysis, it was found that all the idioms proved to be suitable in the representation of ordinal data, except for the stacked bars idiom, which compromised the perception of the data path of the visualization. The work also concludes the best implemented transitions to accompany said idioms, yet it does not declare them as the possible better representations, showing which ones should be rethought and identifying the aspects which were found good in all transitions.

Looking back on the defined problems by Gorodov et al. [7] described in chapter 2, both *visual noise* and *large image perception* are answered by the proposed idioms with some sort of aggregation implied in its technique. The only idiom where this is not the case is the Cleveland plot, which can suffer with higher data flows, and for this reason should be replaced by the heatmap in these situations. The flexible definition of the idiom's timespan and width, allows the definition of the required representation detail, which ensures no *information loss* by the visualization. The concept of Graceful degradation, representing data through a path where it gets iteratively aggregated, means there is no *high rate of image change*, but rather a smooth data flow. It also means that the system can maintain its *high performance requirements*, which was also one of the main focuses of our work.

5.1 System Limitations and Future Work

Missing from the visualization, and as mentioned in plenty of the state-of-the-art works, is the capability to respond to user's interactivity by the visualization, which would help the user's understanding of the information. This could be done with the help of an SVG over the canvas, as explained in Kee et al. [27]. The focus on the implementation process of the visualization's capabilities, left the initial interface, that allows the users to pick and manipulate the idioms, incomplete. The full completion of this interface should be a priority if the work is to be used in a real context.

The analyzed idioms showed great results for the majority of the cases. However, ordinal line chart's scale still needs to be rethought as it saw poor results. If it is decided that the stacked bars should be kept for its part-to-whole observation value, the current implementation proved to need plenty of restructuring work. Additionally, as discovered in chapter 4, the stacked bars to ordinal line chart's Line Morphing transition did not fulfill the required minimums and should be replaced with better alternatives, if, once again, the beginning idiom is kept. More studies should also be performed to understand the impact of the visualization's width and/or timespan in the user's data comprehension.

Bibliography

[1] J. Sansen, G. Richer, T. Jourde, F. Lalanne, D. Auber, and R. Bourqui, “Visual exploration of large multidimensional data using parallel coordinates on big data infrastructure,” *Informatics*, vol. 4, no. 3, p. 21, Jul 2017. [Online]. Available: http://dx.doi.org/10.3390/informatics4030021

[2] N. Ferreira, J. Poco, H. T. Vo, J. Freire, and C. T. Silva, “Visual exploration of big spatio-temporal urban data: A study of new york city taxi trips,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2149–2158, 2013.

[3] H. Zhiyuan, Z. Liang, X. Ruihua, and Z. Feng, “Application of big data visualization in passenger flow analysis of shanghai metro network,” in *2017 2nd IEEE International Conference on Intelligent Transportation Engineering (ICITE)*, 2017, pp. 184–188.

[4] E. D. Ragan, A. S. Stamps, and J. R. Goodall, “Empirical study of focus-plus-context and aggregation techniques for the visualization of streaming data,” in *Proceedings of the International Conference on Advanced Visual Interfaces*, ser. AVI ’20. New York, NY, USA: Association for Computing Machinery, 2020. [Online]. Available: https://doi.org/10.1145/3399715.3399837

[5] O. Daae Lampe and H. Hauser, “Interactive visualization of streaming data with kernel density estimation,” in *2011 IEEE Pacific Visualization Symposium*, 2011, pp. 171–178.

[6] G. Pires, D. Mendes, and D. Gonçalves, “Vismillion: A novel interactive visualization technique for real-time big data,” in *2017 IEEE International Congress on Big Data (BigData Congress)*, 2019.

[7] M. S. Evgeniy Yur’evich Gorodov, Vasiliy Vasil’evich Gubarev, “Analytical review of data visualization methods in application to big data.” Journal of Electrical and Computer Engineering, 2013.

[8] X. Jin, B. W. Wah, X. Cheng, and Y. Wang, “Significance and challenges of big data research,” *Big Data Research*, vol. 2, no. 2, pp. 59–64, 2015, visions on Big Data. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214579615000076

[9] V. E. Johnson and J. H. Albert, *Ordinal data modeling*, ser. Statistics for Social Science and Behavorial Sciences. New York: Springer, 1999. [Online]. Available: https://cds.cern.ch/record/1608780

[10] B. J. Zhicheng Liu and J. Heer, "immens: Real-time visual querying of big data," vol. 32, no. 3. Eurographics Conference on Visualization (EuroVis) 2013, 2013.

[11] A. Inselberg", ""the plane with parallel coordinates"," *The Visual Computer*, 1985.

[12] Sye-Min Chan, Ling Xiao, J. Gerth, and P. Hanrahan, "Maintaining interactivity while exploring massive time series," in *2008 IEEE Symposium on Visual Analytics Science and Technology*, 2008, pp. 59–66.

[13] P. McLachlan, T. Munzner, E. Koutsofios, and S. North, "Liverac: Interactive visual exploration of system management time-series data," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '08. New York, NY, USA: Association for Computing Machinery, 2008, p. 1483–1492. [Online]. Available: https://doi.org/10.1145/1357054.1357286

[14] R. K. Tim Repke, "Topic-aware network visualization to explore large email corpora." EDBT/ICDT Workshops 2018, 2018.

[15] J. Traub, N. Steenbergen, P. M. Grulich, T. Rabl, and V. Markl, "I2: Interactive real-time visualization for streaming data." EDBT, 2017.

[16] U. Jugel, Z. Jerzak, G. Hackenbroich, and V. Markl, "M4: A visualization-oriented time series data aggregation," *Proc. VLDB Endow.*, vol. 7, no. 10, p. 797–808, Jun. 2014. [Online]. Available: https://doi.org/10.14778/2732951.2732953

[17] T. Fujiwara, J. Chou, S. Shilpika, P. Xu, L. Ren, and K. Ma, "An incremental dimensionality reduction method for visualizing streaming multidimensional data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 1, pp. 418–428, 2020.

[18] M. Krstajić and D. A. Keim, "Visualization of streaming data: Observing change and context in information visualization techniques," in *2013 IEEE International Conference on Big Data*, 2013, pp. 41–47.

[19] B. Shneiderman, "The eyes have it: a task by data type taxonomy for information visualizations," in *Proceedings 1996 IEEE Symposium on Visual Languages*, 1996, pp. 336–343.

[20] F. Mansmann, M. Krstajic, F. Fischer, and E. Bertini, "StreamSqueeze: a dynamic stream visualization for monitoring of event data," in *Visualization and Data Analysis 2012*, P. C. Wong, D. L. Kao, M. C. Hao, C. Chen, R. Kosara, M. A. Livingston, J. Park, and I. Roberts, Eds., vol.

8294, International Society for Optics and Photonics. SPIE, 2012, pp. 13 – 24. [Online]. Available: https://doi.org/10.1117/12.912372

[21] Y. Hashimoto and R. Matsushita, “Heat map scope technique for stacked time-series data visualization,” in *2012 16th International Conference on Information Visualisation*, 2012, pp. 270–273.

[22] S. Havre, B. Hetzler, and L. Nowell, “Themeriver: visualizing theme changes over time,” in *IEEE Symposium on Information Visualization 2000. INFOVIS 2000. Proceedings*, 2000, pp. 115–123.

[23] T. Pereira, D. Mendes, and D. Gonçalves, “Vismillion and change,” 2019.

[24] F. Castanheira, D. Mendes, and D. Gonçalves, “Fastviz - visualizando big data em evolução dinâmica,” 2020.

[25] S. Cheng, K. Mueller, and W. Xu, “A framework to visualize temporal behavioral relationships in streaming multivariate data,” in *2016 New York Scientific Data Summit (NYSDS)*, 2016, pp. 1–10.

[26] T. Munzner, *Visualization Analysis and Design.* CRC Press, 2014.

[27] R. C. Daniel E. Kee, Liz Salowitz, “”comparing interactive web-based visualization rendering techniques”,” 2012.

[28] https://www.amx.com/en/site_elements/benefits-and-applications-of-websockets#:~:text=WebSockets%20are%20ideal%20for%20providing,via%20a%20web%20browser%20interface., accessed: 08-10-2021.

A

Questionnaire example

This chapter includes the tested profile questionnaire, followed by one of twelve system evaluation questionnaires, demonstrating the procedure that the users took for the initial test of OrderWarp. The chosen evaluation questionnaire represents the combination between Cleveland plot and the ordinal line chart idioms, as well as their respective transitions. Both questionnaires were written in Portuguese.

Informação do utilizador

As suas respostas vão ser privadas!

*Obrigatório

1. Qual é o seu id do teste? *

____

2. Qual é o seu grupo de idades *

Marcar apenas uma oval.

( ) <18
( ) 18-24
( ) 25-30
( ) 31-40
( ) 40+

3. Qual é o teu género *

Marcar apenas uma oval.

( ) Feminino
( ) Masculino
( ) Prefiro não dizer
( ) Outra: ______________________________

4. Quão frequentemente utiliza técnicas de visualização de informação? *

Ex: Gráficos de barras, gráficos de linhas, gráficos de distribuição (scatter plot), etc. Isto pode ser no trabalho, nas notícias, por interesse pessoal ou outras razões.

Marcar apenas uma oval.

- ◯ Nunca
- ◯ Ocasionalmente
- ◯ Todos os meses
- ◯ Todas as semanas
- ◯ Todos os dias

5. Com que idiomas de visualização está familiarizado? *

Sinta-se a vontade para pesquisar se não conhecer algum destes termos

Marcar tudo o que for aplicável.

- ☐ Gráfico de barras (Bar chart)
- ☐ Gráfico de linhas (Line chart)
- ☐ Gráfico de distribuição (Scatter plot)
- ☐ Mapa de calor (Heat map)
- ☐ Pie charts
- ☐ Choropleth map
- ☐ Box plots
- ☐ Barras empilhadas (Stacked bars)
- ☐ Streamgraph
- ☐ Mapa em árvore (Tree map)

Outra: ☐ ____________________

OrderWarp

Olá!

O meu nome é Henrique Ferreira, e de momento estou no processo de testagem com utilizadores do projeto para a minha dissertação, e para tal preciso da sua ajuda. O questionário deve demorar cerca de 20 minutos, mas à medida que faz mais questionários mais rápido ficará porque poderá passar perguntas à frente.

Se já fez um dos nossos questionários, já deve ter lido esta explicação, por isso pode passar para a próxima secção se acha que compreendeu bem o trabalho.

O OrderWarp, como é chamado, é a continuação do trabalho Vismillion, um sistema de visualização de dados para datasets de grande dimensão onde os dados chegam em tempo real, ou por outras palavras dados em Streaming. O objetivo do OrderWarp é expandir as capacidades do Vismillion e permitir representar outros tipos de dados, no meu caso dados ordinais. Os dados ordinais são caracterizados como dados divididos em categorias, mas estas categorias tem uma ordem definida, um exemplo destes dados seriam tamanhos de roupas (XS, S, M, L, XL)

Para compreender a informação apresentada dentro do dataset, são utilizados diferentes técnicas de visualização chamadas de idiomas, e visto que estamos a estudar dados em tempo real, o seu tempo de chegada é importante. A ideia por detrás do projeto é que os dados sigam um caminho desde que chegam a visualização passando por um número de idiomas, de forma a que seja estudado de diferentes formas, e, ao mesmo tempo, enquanto os dados tornam-se mais velhos são cada vez mais agregados, para permitir ao utilizador analisar todos os dados que chegaram à visualização desde que começou, com mais detalhe nos dados mais recentes.

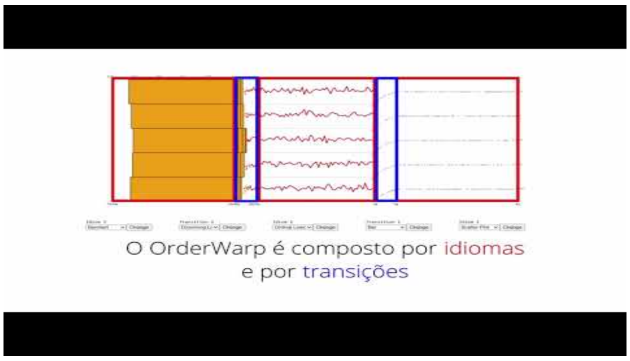
No vídeo apresentado abaixo, vai puder ver uma explicação do sistema OrderWarp.

Quando acabar de ver o vídeo, por favor siga para a próxima secção, onde começamos a testagem! É importante informar que todas as respostas vão ser privadas, e se continuar com alguma questão ou observação após responder ao questionário, pode-me sempre contactar em ferreira.henrique27@gmail.com, ou através das redes sociais, ou qualquer outra forma.

Muito obrigado,
Henrique Ferreira

*Obrigatório

Explicação do OrderWarp

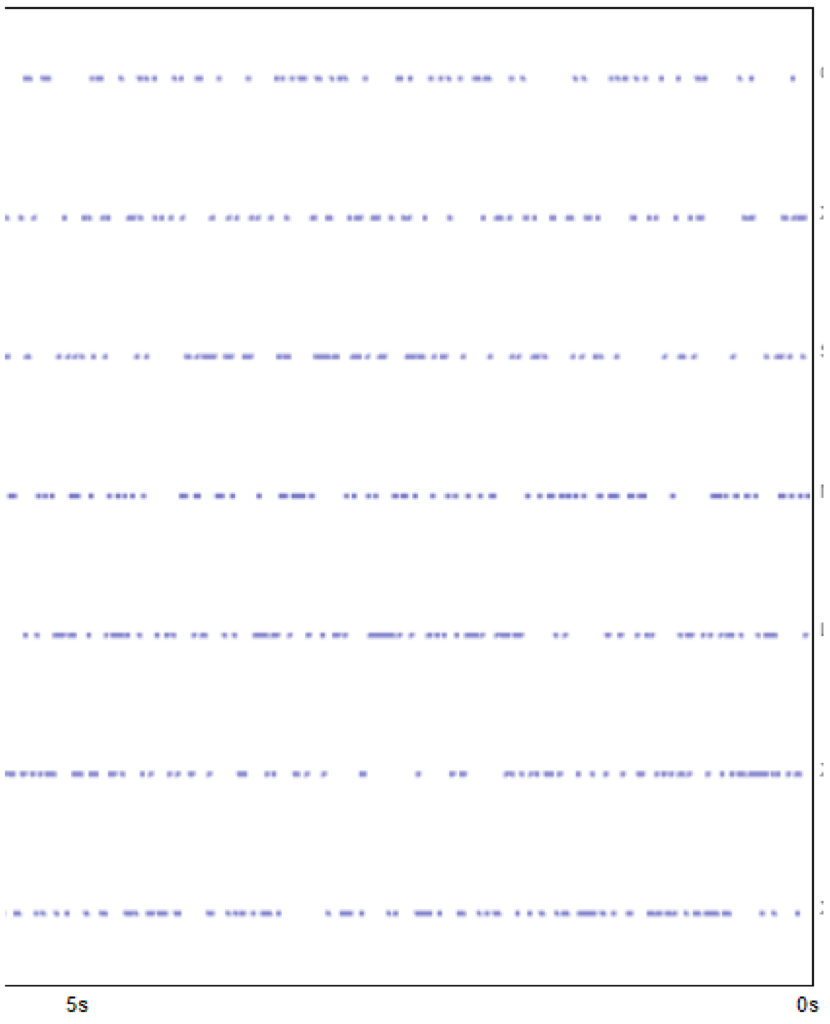

O OrderWarp é composto por idiomas e por transições

http://youtube.com/watch?v=RcwU2Bc3KT8

1. Qual é o seu id do teste? *

Cleveland plot

Já viu este idioma?



2. Já respondeu a questões sobre o Cleveland plot noutro dos nossos questionários? *

Marcar apenas uma oval.

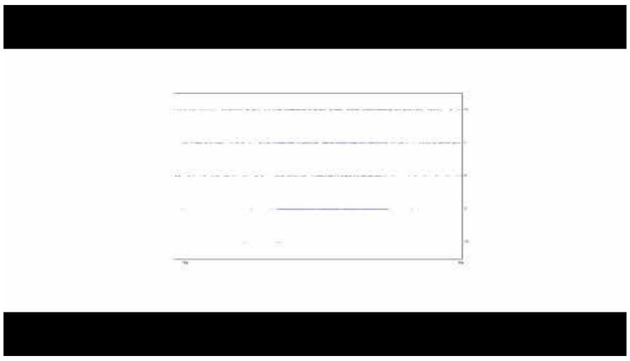
- ⬭ Não
- ⬭ Sim *Avançar para a pergunta 9*

Cleveland plot

O Cleveland plot é semelhante a um gráfico de dispersão (scatter plot), onde os dados são representados por pontos e posicionados de acordo com os seus valores em dois eixos. A diferença é que um destes eixos não é continuo, mas dividido em vários valores ordinais, como poderá ver em baixo no vídeo.

Por favor, assista a este vídeo com cuidado, e responda posteriormente às questões sobre o mesmo. Esteja à vontade para ver o vídeo tantas vezes como necessário, e por favor registe quantas vezes o reviu.

Cleveland plot



http://youtube.com/watch?v=uKTqZ0lQWDg

3. Para que valor existe mais pontos durante a maior parte do vídeo? *

Marcar apenas uma oval.

- ⬭ Não foi claro
- ⬭ XS
- ⬭ S
- ⬭ M
- ⬭ L
- ⬭ XL

4. Existiu uma aglomeração súbita em algum momento? *

Marcar apenas uma oval.

- Não foi claro
- Não
- Sim

5. Se respondeu que sim, para qual(is) valor(es) ordinais? *

Marcar tudo o que for aplicável.

- [ ] XS
- [ ] S
- [ ] M
- [ ] L
- [ ] XL

6. Quão confiante se sente nas respostas anteriores? *

Marcar apenas uma oval.

	1	2	3	4	5	
Nada confiante	○	○	○	○	○	Completamente confiante

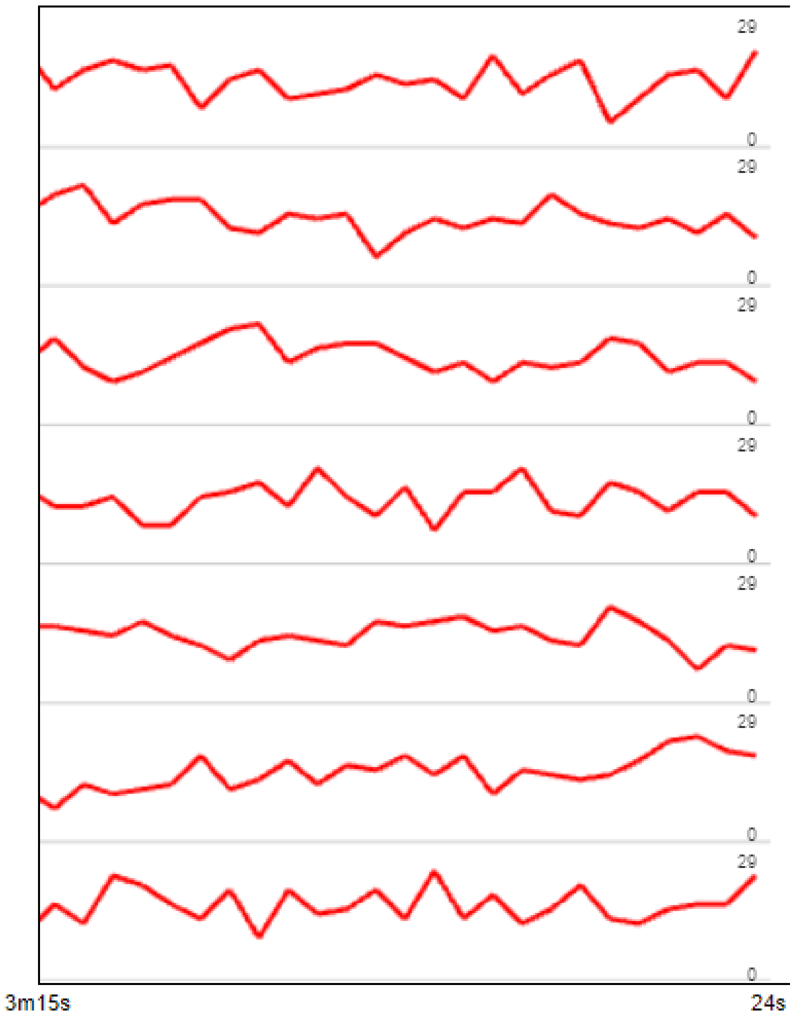
7. Quantas vezes teve que rever o vídeo? *

8. Se tiver quaisquer observações acerca do Cleveland plot por favor escreva-las aqui

Exemplos: Dificuldades que não foram abordadas, recomendações, bugs.

Gráfico de linhas (Line chart)

Já viu este idioma?



9. Já respondeu a questões sobre o gráfico de linhas noutro dos nossos questionários? *

Marcar apenas uma oval.

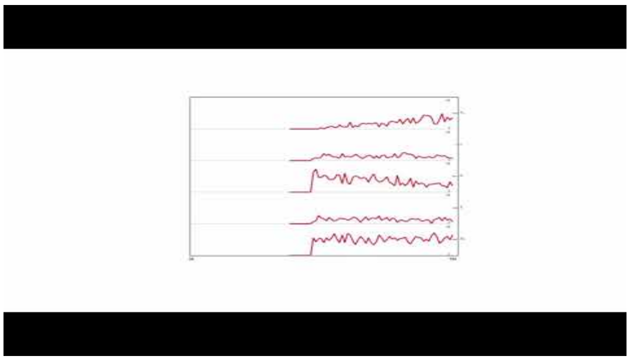
- ⬭ Não
- ⬭ Sim *Avançar para a pergunta 16*

Gráfico de linhas (Line chart)

O gráfico de linhas é um idioma muito comum, onde os dados ligam-se através de uma linha, e as suas posições no eixo do x e y representam valores dos dados em dois domínios. No OrderWarp o line chart é ligeiramente diferente. Existe uma linha diferente para cada valor ordinal, e cada ponto representa o número de pontos recebidos num pequeno intervalo, como vai puder ver no vídeo.

Por favor, assista a este vídeo com cuidado, e responda posteriormente às questões sobre o mesmo. Esteja à vontade para ver o vídeo tantas vezes como necessário, e por favor registe quantas vezes o reviu.

Line chart



http://youtube.com/watch?v=tcCLb-Rsvg0

10. Qual o valor que teve mais pontos durante grande parte do vídeo? *

Marcar apenas uma oval.

- ◯ Não foi claro
- ◯ XS
- ◯ S
- ◯ M
- ◯ L
- ◯ XL

11. Que valor(es) estiveram continuamente a crescer ou decrescer ao longo do tempo? *

Os valores não têm mesmo de estar a subir em todos os tempos, apenas tem de crescer ou decrescer com tempo!

Marcar tudo o que for aplicável.

- ☐ XS
- ☐ S
- ☐ M
- ☐ L
- ☐ XL

12. Qual foi aproximadamente o valor máximo atingido pelo valor XL em todo o vídeo? *

Marcar apenas uma oval.

- ◯ Não foi claro
- ◯ À volta de 15
- ◯ À volta de 30
- ◯ À volta de 50
- ◯ À volta de 90

13. Quão confiante se sente nas respostas anteriores? *

Marcar apenas uma oval.

	1	2	3	4	5	
Nada confiante	⬭	⬭	⬭	⬭	⬭	Completamente confiante

14. Quantas vezes teve que rever o vídeo? *

15. Se tiver quaisquer observações acerca do Line chart por favor escreva-las aqui

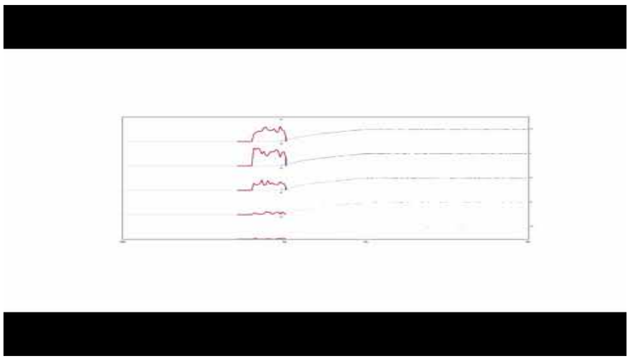
Exemplos: Dificuldades que não foram abordadas, recomendações, bugs.

Transição 1

No OrderWarp quando falamos de transições, falamos da união de dois idiomas, e a missão da transição é transformar os dados de um idioma para o outro suave e eficazmente.

Agora vamos testar duas alternativas de transições.

Cleveland plot para gráfico de linhas - Barras



http://youtube.com/watch?v=uGGXSPwl6g8

16. É claro que os pontos estão a ser agregados? *

Marcar apenas uma oval.

	1	2	3	4	5	
Nada claro	◯	◯	◯	◯	◯	Completamente claro

17. Quão fluída considera ser a transição? *

Marcar apenas uma oval.

	1	2	3	4	5	
Nada fluída	◯	◯	◯	◯	◯	Muito fluída

18. Qual destas observações acerca da lógica da transição é verdadeira? *

Marcar apenas uma oval.

◯ Os pontos são agrupados na parte de baixo da barra e fazem-la crescer, o tamanho da barra vai corresponder ao número de dados que a linha vai representar no gráfico de linhas

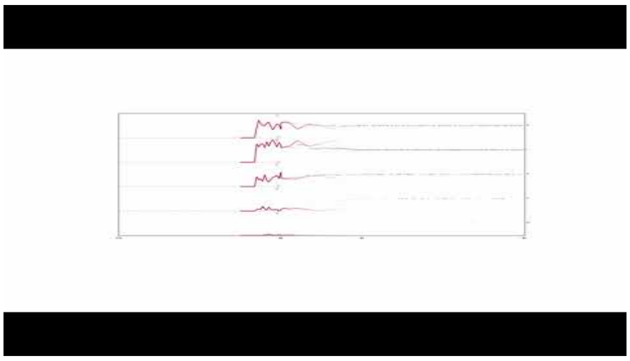
◯ As barras crescem continuamente, não influenciadas pelos pontos, ao mesmo ritmo até chegar o último ponto da barra, o tamanho da barra vai corresponder ao número de dados que a linha vai representar no gráfico de linhas

◯ Os pontos desaparecem assim que chegam ao lado direito da transição, e o tamanho das barras para aleatóriamente.

19. Se tiver quaisquer observações acerca da transição por favor escreva-las aqui

Transição 2

Cleveland plot para Line chart - Pontos agrupados



http://youtube.com/watch?v=2sFZH_bYrm8

20. É claro que os pontos estão a ser agregados? *

Marcar apenas uma oval.

	1	2	3	4	5	
Nada claro	⬭	⬭	⬭	⬭	⬭	Completamente claro

21. Quão fluída considera ser a transição? *

Marcar apenas uma oval.

	1	2	3	4	5	
Nada fluída	⬭	⬭	⬭	⬭	⬭	Muito fluída

22. Qual destas observações acerca da lógica da transição é verdadeira? *

Marcar apenas uma oval.

- ◯ Os pontos recriam a linha que vão representar ficando uns atrás dos outros.
- ◯ Os pontos vão para um ponto aleatório nas linhas criadas.
- ◯ Os pontos caminham para o ponto na linha que vão representar e agrupam-se nesse mesmo ponto

23. Que transição pensa que funciona melhor? *

Marcar apenas uma oval.

- ◯ A anterior - Barras
- ◯ Esta - Pontos agrupados

24. Se tiver quaisquer observações acerca da transição por favor escreva-las aqui