VisMillion and change

Tiago Pereira
tiago.b.pereira@tecnico.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

November 2019

Abstract

The amount of information produced by the immense activities of today's world has been increasing exponentially. With this increase, the exploration and the analyzes of representations that allow the visualization of this vast amount of information have been growing to follow its evolution. However, with the emergence of real-time Big Data, it has become imperative to simplify its analysis due to the huge amount of information so that the user can interpret the various patterns in the data as it evolves rapidly and can still react to possible variations of data flow trends, always maintaining the context of the information that is represented in the visualization. We propose the VisMillion and Change that focuses on the study of smooth animated transitions between different visualization techniques, allowing the analyzes in real-time of large amounts of information comprised in multiple modules positioned side by side, following the concept of graceful degradation. It provides different time intervals to each module in which the data flows through, from right to left, and gets aggregated as it evolves in time ensuring a bigger history and a detailed visualization for the most recent data. It is intended to provide a visualization of real-time Big Data and a set of smooth transitions that make it easier to analyze large amounts of information as it crosses over different visualization techniques associated to various statistical measures and their matching aggregation levels over time.
Keywords: Visualization Technique, Real-Time, Big Data Pattern, Streaming, Animated Transitions

1. Introduction

The automation of various activities performed nowadays is producing more and more information. This increase is largely due to the data that is generated every day and anywhere, from simple personal records to complex networks and financial transactions [10]. Also, sensors and other electronic systems have a huge impact on the volume of produced information because they are constantly producing more data at smaller intervals. As a result of this rapid and exponential increase, exploring this vast density of information has become a complex task and a lot of research has been done to find a way to extract only useful information.

Given the importance of facilitating the analysis and interpretation of Big Data, visualizations play an essential role, allowing a better understanding and recognition of interesting patterns, behaviours and correlations between the various data [1]. However, representing large amounts of data may constrain the user's ability to analyze their entire domain. Likewise, the resolution of conventional displays itself, which is already limited, may be insufficient to view all this information. As a solution to this problem, some systems have been applying data reduction methods, statistical measures to aggregate information and also scalability characteristics at both detail and visualization levels.

When the data is continuously generated with a connection to its temporal context, existing tools that once processed static information and allowed it to be visualized, now have to adapt their techniques to follow the evolution of this data received in streaming, so that they can generate real-time visualizations. However, most of the current systems have not yet adapted to this imposition of streaming data, making the analysis of these visualizations difficult or even suffering from too dense representations which can lead the system to "freeze" for a certain period or even making it crash [3].

The domains and types of each data set should be considered and reduced using multiple aggregation techniques, as well as the modifications and visualization changes should be minimized and smoothed to avoid visual leaps during the analysis and also avoid the loss of context of data.

To tackle these challenges, we propose VisMillion and Change, a set of transitions between visualization techniques targeted at time series, in order to represent large amounts of information in real-time, with its data comprised between multiple visualizations associated to different aggregation methods

and time intervals allowing the users to perceive the global context of the information, as well as a more detailed analyses of the most recent data and its patterns evolution. For this, the representation of the information was provided through elements that perform a graceful degradation metaphor, aggregating the information as it gets older in the visualization, passing it to another visualization type through transition techniques. This gives the user a more detailed view of the latest data while reducing the older one, saving memory but keeping its context and patterns always available. As such, the contributions of this work are:

1. A set of conceptualized visualization and transition techniques to allow the representation of Streaming Big Data.
2. A prototype to exhibit each technique.
3. An evaluation to deal with the efficiency of the developed techniques.
4. A user evaluation to check whether the techniques are effective or not.

The rest of the document is organized as follows. Section 2 presents and discusses the related work about the current techniques to visualize Streaming Big Data. Section 3 describes the conceptualized visualization and transition techniques. In Section 4 we detail the architecture of the developed prototype and its implementation process. Section 5 summarizes the performed evaluation of the system. Finally, in Section 6 we present the conclusions of this work, as well as some future work pointers.

2. Background

Regarding the visualization of real-time Big Data, a lot of new challenges have been imposed. Although many existing systems can represent streaming data, the amount of information that can be represented while maintaining the coherence state of the visualization is quite limited. It is, therefore, necessary to reconcile these types of visualizations with the amount of data obtained in real-time. Stream Squeeze [11], MeDICi [4] and Event Visualizer [6] systems can display data volumes in 10^5 data/hour ($\approx$ 28, 63 and 145 data per second, respectively), which is a limitation when viewing larger densities of information at smaller time intervals. According to the amount of information, the type of data also affects how it is represented.

Most systems are limited to the representation of numerical value sequences, i.e. quantitative data. Although Topic-aware [12] and Streaming LogData [15] systems can represent heterogeneous data, they end up imposing the existence of lengthy pre-processing, due to the need to standardize all received formats, in addition to solving potential problems that may exist, both in the structure and formatting of each one.

Given the amount of information to be visualized, as well as the limitations imposed by the data types, there is a need to use simplification and reduction methods. It is crucial to ensure no loss while viewing the results from the application of the previous methods, compared to the original state. Sometimes, even after performing these techniques of simplifying, clearing or reducing existing data, its magnitude may still be too large to be represented on current devices due to limited screen space. As such, some systems like I^2 [13] instead of processing all the values and fit them into a single pixel column, uses the M4 aggregation technique [8] to generate four values per pixel column, MeDICi [4] reduces the dimensions of streams using the Symbolic Aggregate Approximation technique and generates a set of glyphs showing the trends of the data, underlining the importance of data aggregation and simplification methods.

Other systems choose to use techniques that provide visualizations to fit the available screen space, like News Streams [9] and Event Visualizer [6] that select the more interesting categories and events and remove the others to optimize the efficiency, Density Displays [7] uses a secondary view that hides a small part of the primary one showing the incoming data before inserting it into the main view. However, these systems can cause some loss of information that could be crucial to the mental pattern created by the user during analysis.

Transitions play one of the most important roles while keeping the user up-to-date on transformations that can occur in the visualization, as well as on the entire context of the information. Having smooth transitions help the user to not lose the context of the view throughout changes that may exist and make sure the existing patterns are always available. Systems like Event Visualizer [6], Streaming LogData [15] and StreamSqueeze [11] adopted smooth transitions to help tracking the information while data is added to the visualization and then evolves. Streamit [2] uses gradual and animated transitions to help maintaining the mental image of the visualization created by the user while its data gets aggregated by similarities. There must be a greater concern about how new data is obtained and represented so that the user maintains the "big picture" and at the same time sees new data and compares it with the existing history.

In short, it appears that the systems that exist today still have some difficulty representing dynamically large flows of information, either because their goal is to represent smaller amounts of data, or be-

cause the systems are not prepared for such dimensions or to create visualizations that in their composition have large chunks of information, obtained in real-time. Given the relevance of transitions, there is still some development to help the users analysing and exploring the information while it evolves.

In this work, we propose a set of smooth transitions that make it easier to analyse large amounts of information as it crosses over different visualization techniques associated to various statistical measures and their matching aggregation levels over time, allowing a dimensional data reduction and an easier recognition of the latest data but at the same time, maintaining all the patterns of the previous records, so that they can be compared during the analysis of the information in real-time.

3. VisMillion and Change

Targeting a set of transitions between pairs of visualization techniques, we propose VisMillion and Change. This set is composed of multiple transition alternatives to create a single and uninterrupted view that joins multiple visualization types associated to modules organized side by side and representing simultaneously all the data that has been received and a history that can hold a larger interval of time. Those visualization techniques represent data that has been aggregated with some kind of statistical measure. Based on VisMillion [5], it follows the concept of graceful degradation, aggregating the older data and showing the recent one in a more detailed visualization.

Based on visualization types, VisMillion and Change presents the data in different techniques: Scatterchart, Linechart, Heatmaps, Streamgraph and Barchart. We chose to use Scatterchart as a technique that presents the raw data, i.e. without aggregation, facilitating the analysis of the most recent data visualized in maximum detail before it gets transferred to next modules. Linechart was used to represent the average value of data per time interval. Streamgraph was used to represent multiple boxplots [14] over time, joining the values from each boxplot and creating coloured areas representing the maximum and minimum values per time interval, as well as its median and interquartile range. Finally, Heatmaps and Barchart were used to group multiple values in order to measure their density. The first one has two options, a normal Heatmap to represent a matrix relating two variables (time and value intervals) with different tones of colours to show the frequency and an accumulator Heatmap that only have one variable (just like the Barchart), relating value intervals with its frequency.

Between the Scatterchart and the other visualization techniques that can be applied to create a single representation of the data, we applied multiple techniques in order to make the transition more appealing to the user and easier to understand as the time evolves.

3.1. Scatterchart - Heatmap Transition

Heatmap squares are formed after the transition of the points corresponding to the most recent data and according to their density by time interval. That is, squares get toned depending on how many points there are in that range.

(a) Non-Animated (b) Fade-in Fade-out

Figure 1: Non-Animated and Fade Transitions

Starting from a transition without animations, as depicted in Figure 1(a), we created a set of alternatives focused on differences in time intervals and resulting velocities of the two halves of the visualization, like the Fade-in Fade-out technique that fades the dots while they decelerate into the Heatmap, gradually forming the squares (Figure 1(b)). In order to present the points that belong to each square, the Agglomeration in Squares technique forms squares of multiple aggregated points, clustering them in small groups, which themselves have a form of a square, then moving the points with the correct acceleration and direction, each group will fit into the Heatmap squares (Figure 2(a)).

(a) Agglomeration in Squares (b) Data Columns

Figure 2: Agglomeration in Squares and Data Columns Transitions

Similar to the previous technique, the Data Columns transition (Figure 2(b)) allows to observe the position of the points relative to the vertical axis and thus a better relation to their values. The dots corresponding to the last column of the Heatmap are decelerated until all of them are between the

Figure 3: Grainy Transition

beginning and the end of that column. Finally, the Grainy transition intends to make the dots bigger and bigger as the time evolves until they have the same dimensions of a Heatmap square, inducing the creation of these squares gradually, like the Figure 3 illustrates.

3.2. Scatterchart - Linechart Transition

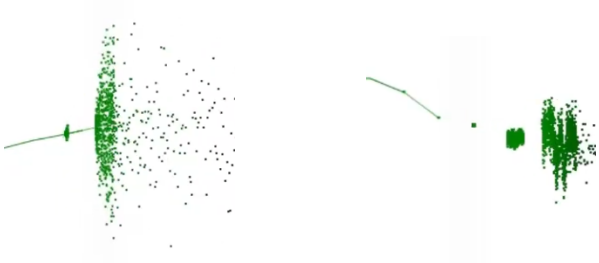
The Linechart is formed after the transition of the points corresponding to the most recent data and according to their average by time interval. That is, the higher the average in a given time interval, the higher (relatively to the vertical axis) will be the connection point of the line in that interval.

(a) Non-Animated (b) Fade-in Fade-out

Figure 4: Non-Animated and Fade Transitions

Starting from a transition without animations, as depicted in Figure 4(a), we created a set of alternatives focused on differences in time intervals and resulting velocities of the two halves of the visualization, like the Fade-in Fade-out technique that fades the dots while they decelerate into the Linechart gradually forming the line (Figure 4(b)). To allow the convergence of the points in multiple intervals and generate a new dot that will later be joined with the line, we created the transition Bottleneck (Figure 5(a)). This one intends to accumulate the points that belong to a time interval, to later start converging them to the point that represents the average value of that interval.

Similarly to the Bottleneck transition, Points Contraction transition (Figure 5(b)) contracts each group of points from the previous intervals, concentrating them and forming a shrinking set of points that will end up on a single one representing the



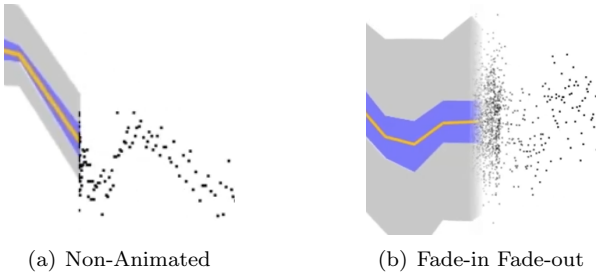
(a) Bottleneck (b) Points Contraction

Figure 5: Bottleneck and Points Contraction Transitions

average of values on that set and then will merge him with the existing line.

3.3. Scatterchart - Streamgraph Transition

The Streamgraph is formed after the transition of points and is intended to illustrate the maximum and minimum values, median, 1st and 3rd quartiles per time interval.



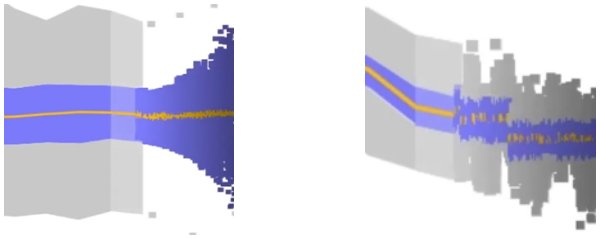
(a) Non-Animated (b) Fade-in Fade-out

Figure 6: Non-Animated and Fade Transitions

Starting from a transition without animations, as depicted in Figure 6(a), we created a set of alternatives focused on differences in time intervals and resulting velocities of the two halves of the visualization, like the Fade-in Fade-out technique that fades the dots while they decelerate into the Streamgraph gradually forming the areas (Figure 6(b)). As an alternative, we created the Narrowing transition to delimit which points will be relevant for the Streamgraph areas, converging them into the areas representing the first and third quartile of each interval. The dots are initially slightly increased in size and change colour to match the colour of the area they represent as shown in Figure 7(a). The points representing the maximum and minimum values of each interval are represented in the visualization without being converged into the previous areas.

Finally, the Stamping transition (Figure 7(b)) contrary to the last technique, keeps all points received in the transition and at the same time tries to adapt them to the Streamgraph areas, i.e. changing the colours to correspond with the ones from the Streamgraph. Also, to reduce the visual leap it was

decided to gradually increase the area of each point providing slight modifications in their position so that at the time of the transition into the Streamgraph, the points were aligned with the areas.

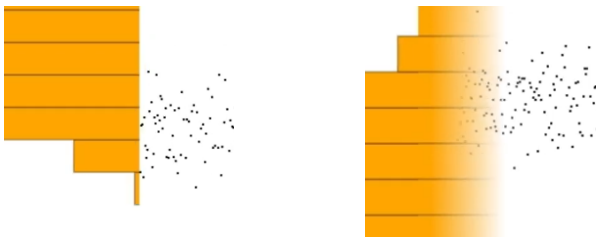


(a) Narrowing (b) Stamping

Figure 7: Narrowing and Stamping Transitions

3.4. Scatterchart - Barchart Transition

Each bar from the Barchart is formed and incremented after the transition of points corresponding to each interval of values. That is, considering that the Barchart was used as an accumulator, the bigger the number of points in that range, the larger the representative bar.

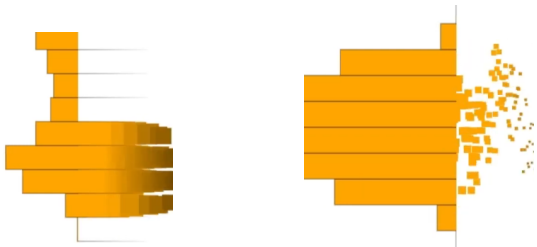


(a) Non-Animated (b) Fade-in Fade-out

Figure 8: Non-Animated and Fade Transitions

Starting from a transition without animations, as depicted in Figure 8(a), we created a set of alternatives focused on differences in time intervals and resulting velocities of the two halves of the visualization, like the Fade-in Fade-out technique that fades the dots while they decelerate into the Barchart gradually forming the bars (Figure 8(b)). To allow a better and more accurate tracking of the points while they move into the bars, inducing that they are being routed into the bars, we created the technique Pilot Lines (Figure 9(a)) which firstly increases the size of the points so that they end up with the same dimension as the height of the bars from the Barchart, reducing the visual leaps during the transition. This technique also uses lines that disappear gradually providing the idea of a pilot line or a route which points the direction of the bars to the dots.

With the purpose of moving the points into the bars but keeping their vertical position unchanged



(a) Pilot Lines (b) Points Consumption

Figure 9: Pilot Lines and Points Consumption Transitions

before the bars increase and at the same time avoiding the visual leap of the direct transition from the points to the bars, we decided to create a transition (Figure 9(b)) that would cause a slight increase in the points, thus avoiding the formation of large groups that could overlap the others and adapting their color to the color of the bars. Finally, the points are moved into each bar, suggesting an effect of their consumption.

3.5. Scatterchart - Heatmap Accumulator Transition

The Heatmap Accumulator has an infinite time range which leads to one column with multiple intervals on the vertical axis, resembling horizontal bars that have the same behavior as the Heatmap. These bars change their colors after the transition of the points according to their concentration.

(a) Non-Animated (b) Fade-in Fade-out

Figure 10: Non-Animated and Fade Transitions

Starting from a transition without animations, as depicted in Figure 10(a), we created a set of alternatives focused on differences in time intervals and resulting velocities of the two halves of the visualization, like the Fade-in Fade-out technique that fades the dots while they decelerate into the Heatmap gradually forming the squares (Figure 10(b)). To increase the size of the dots and reduce the visual leap, the Point Forwarding technique (Figure 11(a)) intend to route the points into each bar while increasing its size and while they grow toward the corresponding bars, they gradually change their colour to the coincident one. Then, the points are shifted

(a) Point Forwarding (b) Electrocardiogram

Figure 11: Point Forwarding and Electrocardiogram Transitions

into each bar causing a consumption effect on them.

The Electrocardiogram transition (Figure 11(b)) divides the data in the Heatmap Accumulator ranges, where for each one the respective points interconnect, forming as many lines (with the colour of the corresponding bar) as the number of bars in the Heatmap. For each interval, the user can see the variation and the peaks in the values before they coincide with the bars. Finally, the Dilation transition (Figure 12) intends to increase the points until they have the same height of the bars and changes their colours so that they match the colour of the interval to which they belong in the Heatmap. When there are a lot of points in a row belonging to the same range, the idea of a bar is projected right at the transition.

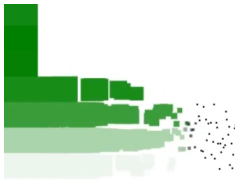


Figure 12: Dilation Transition

4. Prototype

To demonstrate and evaluate the conceptualized transition techniques, we developed a prototype to represent large amounts of data in real-time divided into multiple modules associated with the various visualization types described earlier in Section 3. The prototype architecture is based on a client-server approach. The server (streamer) creates and sends data packets to the interface, simulating an information source that distributes data in real-time to the receiver. The client presents the visualization interface to the user.

4.1. Server-side Implementation

The server is based on the VisMillion concept [5] and is implemented as a python script to send data packets to the interface via Web Sockets. This technology is a two-way connection that allows a simple but persistent exchange of information between the client and the server, operating between a single communication channel. After reading CSV files containing the information, data streams are generated and then sent to the interface with the corresponding configured delay between packages. Since our main focus is the data representation, the server is not responsible for any kind of data simplification or reduction. However, the server is ready to send packets of data even though its records have an old timestamp, making it possible to view information regardless of the original timestamps.

4.2. Client-side Implementation

The interface is implemented as a browser-based application, thus allowing the user to analyse the prototype developed using only a browser. It is defined by a configurable area in an HTML document, where through specific methods for each desired technique, one can draw elements on the screen. This using the HTML5 Canvas API which allows numerous changes to figures, colours, sizes, etc. Nevertheless, this API doesn't have any predefined transitions or animations.

There is a Manager entity responsible for managing all the modules of the prototype and it is also through him that the domains are defined. It receives the packets from the server and transfers the data to the first module and then to the subsequent ones, once the data no longer belongs the its temporal domain. The Manager unifies the various modules and is responsible for the invocation of their update and redraw methods.

Each Module can be initialized as a normal or a transitional module, as well as one can define the dimensions and time interval that it is intended to represent the information. Each module can be assigned with a visualization or a transition technique that can be changed in real-time. After the manager starts the updating and drawing threads, it will also update and draw on the associated visualization or transition technique, which will only chain alterations if they are sufficient to move the represented particles on the screen. Every module has a start and end time, corresponding to its duration, and if the data doesn't belong to that interval, then it will be filtered, leaving the older one for the Manager to transfer to the next module and this way there won't be repeated data stored in the memory of the browser.

The visualization techniques implemented on the prototype have some configurations that can be modified to define specific measures and colours according to users' preferences. Since the horizontal axis corresponds to a timescale, and time passes to

maintain its relationship with real-time, all the elements drawn in the visualization are also evolving, thus moving left to the corresponding original time. This movement follows the Uniform Rectilinear Motion formula, presented in Equation 1.

$$x = x_0 + \frac{width}{duration} * elapsedTime \quad (1)$$

The transition techniques implement on the prototype are responsible for aggregating the information that is obtained by the Scatterchart, to transfer it to the next module.

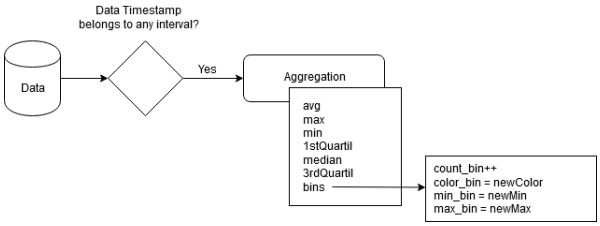


Figure 13: Aggregation by time intervals

The aggregation technique applied (Figure 13) depends on a pre-configured dimension in pixels that will divide its width into multiple intervals, each one containing an initial and final timestamp that will then be used to aggregate the information that belongs to the existing range. For each interval, the statistical measures of that data set are calculated using D3 methods, finding their values and associating them to the respective variables. Thus, this data will, together with the others that belong to the range, define the mean, maximum and minimum value, first and third quartile, as well as the median. If it is a transition to a Heatmap, then the bin to which the data belongs will also be detected to increment its data counter and update the corresponding colour through a linearly scaled colour interpolation of the data, using d3-scale API.

The movement of the graphical representation of the data in the transition is given through a formula that calculates the position of the points according to their relation with time. Just as in visualization techniques we used the Uniformly Varied Motion formula, as shown in Equation 2b. The duration of the transition had to be calculated using the Equation 2a to calculate the acceleration used in the determination of the position.

$$t = \frac{2 * position}{v_i + v_f} \quad (2a)$$

$$x = x_0 + v_i * elapsedTime + \frac{1}{2} * a * elapsedTime^2 \quad (2b)$$

Given the complexity of the proposed transitions, the position calculation using the Uniformly Varied Motion formula only allowed to reach the next module velocity at the end of the transition, which turned out to be a disadvantage for transitions where it was intended to achieve this velocity in an earlier phase. As a solution, we applied an interval before and after the time interval in which the deceleration occur. Thus, it would be possible to have an initial interval where the velocity was equal to the initial one, then a deceleration moment and again a final interval with the velocity equal to the final one, as depicted in Figure 14 and the resulting position as follows, $position = v_i * t_i + pos_a + v_f * t_f$.

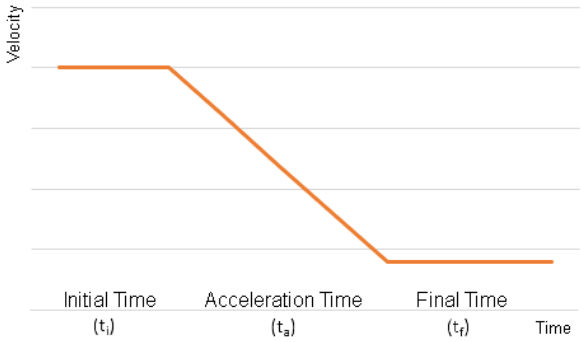


Figure 14: Velocity as the time evolves on the transition

5. Evaluation

To understand if VisMillion and Change fulfils its objective, we tested the usability and the efficiency of the developed techniques. We used the Google Chrome browser (*version 64.03865.90*) installed on a Windows 10 Pro system with an *Intel Core i7-4770K processor (3.50 GHz)*, 16 GB of RAM and a 1920x1080 resolution. The rendered view (1110x512 px) was divided into three modules with widths of 455, 200, and 455 px.

5.1. Usability

We conducted a set of usability tests to compare each transition developed and understand which techniques were preferred by the participants.

1) *Method:* The tests consisted of a Google Forms' questionnaire. First, we introduced the questions to the participants. Then, we did a brief introduction to the utilized visualization techniques, as well as the different modules and the movement of the data flow. Participants were then encouraged to ask questions, followed by their explanation so that they could start the questionnaire. It started with a set of profiling questions and over the followed sections, they had to watch a video (1 minute each) for each transition. Then, they had to answer a set of questions about that transition. In the end, participants were asked to give some suggestions.

2) *Data sets and Settings:* We created multiple data sets using a data generator to avoid their rep-

etition over the various techniques. The duration and domain used to create each of these data sets were chosen to create 1-minute videos where the desired patterns were verified.

3) *Tasks:* During the completion of the questionnaire, participants were asked to answer a set of questions about each video. Videos made it possible to view trends and compare them with the evolution of data over time. Each set of questions consisted of 4 multiple choice questions on observed trends and 2 questions which were answered according to a Likert scale. The question sets were repeated over the various existing transitions for each pair of visualization techniques. Each question corresponded to different aspects:

1. **Compare Trends** - Analyse the data flow and compare the evolution of the trends over 1 minute.

2. **Finding Trends: Incoming data** - Analyse of trends as they were evolving over the right module.

3. **Finding Trends: Previously received data** - Analyse of trends visible in the left module.

4. **Identifying statistical measures** - Analyse of statistical measures resulting from aggregation in the left module.

5. **Rate: Understanding Data Flow** - Rate using a Likert Scale (1-5), the ease of analyzing data flow evolution with the transition technique in question.

6. **Rate: Overall Transition Rating** - Rate using a Likert Scale (1-5), to understand whether the participant liked the transition or not.

7. **Global Rank** - At the end of each group of transitions, the participants were asked to rank them in their according to their preference.

4) *Participants:* We had a total of 28 participants (11 female and 17 male), whose ages ranged from 17 to 26 and from 51 to 56 years (82% between 17-26 years, 18% between 51-56 years). 89% already knew the term Big Data. 10 participants took the test in person, while the remaining 18 took it remotely.

5) *Results:* After a statistical comparison between the answers given by the participants to the questionnaire, the answers given to the Heatmap transitions were the ones that allowed the largest number of significant differences, but none of them were enough to draw big conclusions about the users' favourite transitions. The trends visualized in the left module proved to be better interpreted as they evolve in transitions where there are animations for visualization types that maintain a relationship with time. Except for Linechart transitions, most animated techniques showed better results in trend comparisons. However, in the transitions developed for Streamgraph, Heatmap Accumulator and Barchart, participants marked through the ratings provided, their favourite transitions:

- **Stamping** - transition to **Streamgraph**
- **Point Forwarding** - transition to **Heatmap Accumulator**
- **Pilot Lines** - transition to **Barchart**

Table 1 lists each Streamgraph transition technique developed with the various aspects addressed in the questionnaire, showing the percentage of right answers up to the fourth aspect and from the fifth, the median and interquartile range values. Checking the relation between right and wrong answers to the first aspects considered in the questionnaire, we applied the Cochran test, which showed significant statistical differences in the identification of trends in older data ($\chi^2(2)$=18,439, p<.0005). Comparing the various pairs of transitions, when performing McNemar tests with Bonferroni corrections, we concluded that transition A (rate=32%) was worse than transition C (rate=75%, p=.048) and transition D (rate=79%, p=.012) to identify trends in the old data. For the final three aspects, the Friedman test was applied, which showed significant statistical differences in the overall classification according to the participant's order of preference ($\chi^2(2)$=26,527, p<.0005). Comparing the various transition pairs, when performing Wilcoxon tests with Bonferroni corrections, we concluded that transition D was the one with the best overall classification, beating transition A, B and C (transition A: Z=-3.281, p=.006; transition B: Z=-4167, p<.0005; transition C: Z=-3.676, p=.0014).

	TA	**TB**	**TC**	**TD**
P1	68%	71%	75%	61%
P2	36%	61%	61%	57%
P3 *	32%	43%	75%	79%
P4	32%	36%	32%	29%
P5	4 (2)	4 (1)	4 (2)	4 (1.75)
P6	4 (2)	4 (1)	4 (2)	4 (1)
P7 *	3 (1)	3 (1.75)	3 (2)	1 (0)

Table 1: Percentage of right answers up to P4. The median and interquartile range values for transitions to Streamgraph, from P5 onwards. * denotes significant statistical differences.

Table 2 lists each Heatmap Accumulator transition technique developed with the various aspects addressed in the questionnaire, showing the percentage of right answers up to the fourth aspect

and from the fifth, the median and interquartile range values. Checking the relation between right and wrong answers to the first four aspects of the questionnaire, we applied the Cochran test, which showed significant statistical differences in trend comparison ($\chi^2(2)$=16,393, p=.003). Comparing the various pairs of transitions, when performing McNemar tests with Bonferroni corrections, it was concluded that transition C (rate=82%) was better than transition E (rate=39 %, p=.02) to compare trends across the data stream view. For the three final aspects, the Friedman test was applied, which showed significant statistical differences in the overall classification considering the participant's order of preference ($\chi^2(2)$=36,652, p<.0005). Comparing the various transition pairs, by performing Wilcoxon tests with Bonferroni corrections, we concluded that transition C was better than the other transitions in terms of the overall ratings given by participants (transition A: Z=-4.107, p<.0005; transition B: Z=-4,364, p<.0005; transition D: Z=-4,378, p<.0005; transition E: Z=-4,365, p<.0005).

	TA	TB	TC	TD	TE
P1 *	54%	79%	82%	68%	39%
P2	64%	54%	71%	89%	54%
P3	68%	54%	61%	75%	57%
P4	50%	54%	57%	50%	43%
P5	4 (2)	4 (1.75)	4 (2)	4 (2)	4 (1.75)
P6	3.5 (2)	4 (2)	4 (1)	4 (1.75)	4 (2)
P7 *	4 (3)	3 (1)	1 (1)	3 (2)	3 (2.75)

Table 2: Percentage of right answers up to P4. The median and interquartile range values for transitions to Heatmap Accumulator, from P5 onwards. * denotes significant statistical differences.

5) *Difficulties and Suggestions:* In the open questions, participants were asked to report their biggest difficulties during the realization of the tasks. Some participants reported difficulties in the interpretation of statistical measures represented in some of the visualization types. Others found it difficult to follow the data from modules where there was no relation with time, such as Barchart and Heatmap Accumulator.

As for suggestions, participants suggested the addition of a colour representation mechanism for data representation, to signal (or alert) when there is a change in trends of the data flow. To allow a better analysis of some transition techniques, it was also proposed to increase the associated width, making it easier to interpret some visual transformations and clarify the identification of the statistical measures represented in the next module. In addition, it was all participants' opinion that the use of animated transitions allows a better understanding of the evolution of the data flow.

5.2. Performance

We conducted a set of efficiency tests applied to each of the developed transition techniques to understand their limitations regarding the drops in frames per second (FPS).

1) *Method:* We chose the various visualization techniques that allow the assignment of time intervals and the transitions developed, to verify the resulting FPS using generated streams of data and tested the prototype for 20 seconds after all modules were filled with information. Finally, we tested the streams with different flows: 100, 200, 1000 data points per second and also different time intervals on the left module: 20, 50, 100, 200, 300, 400 seconds and 1 second for the right module.

2) *Results:* The best results were obtained when the flow was no higher than 200 data points per second, which corresponds to 720,000 data points per hour. The duration of the Heatmap, Linechart and Streamgraph, was normally better when it did not exceed 3 minutes. However, increasing flow speed and the module's duration may influence the system's ability to maintain its fluidity while moving graphical representations of the data. The complexity of the developed animations aggravates the number of operations that are required to animate the visual transformations. Although the techniques can be used, the transitions that had least influence on the system performance, were the **Fade-in Fade-out techniques** for the Heatmap, Linechart and Streamgraph transitions and their FPS values were very similar. The FPS variation for the Heatmap is depicted in Figure 15. The fact that these techniques do not apply computation to vertical movements, increases in size or variations in colours, avoids complex operations while the other techniques have to apply them.

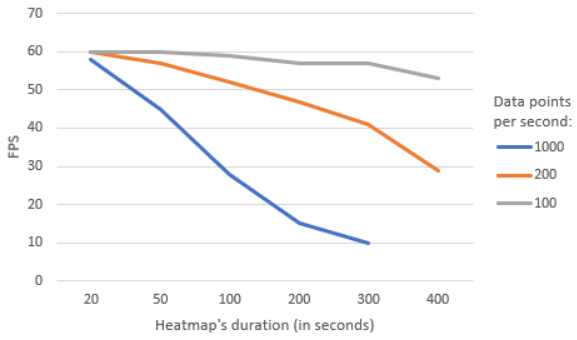


Figure 15: Frames per second achieved by VisMillion and Change for Fade-in Fade-out Transition between Scatterchart and Heatmap

Regarding the users' preferred transitions in the usability tests, the Stamping Transition between Scatterchart and Streamgraph (Figure 16) didn't

obtain the best performance for flows over 200 data points per second, maintaining a history of over 3 minutes. For the Heatmap Accumulator and Barchart, there is no limitation imposed by the performance of the prototype. It reflects that future work is needed to make some user's preferred techniques more efficient, such as the transition to Streamgraph.

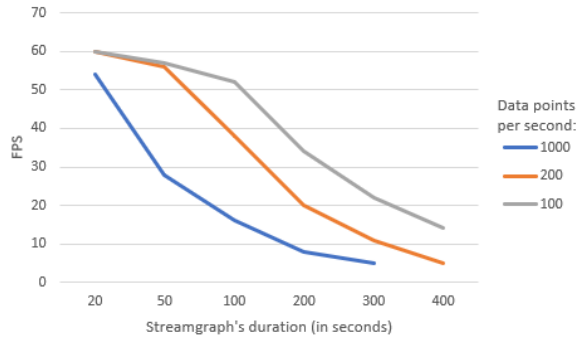


Figure 16: Frames per second achieved by VisMillion and Change for Stamping Transition between Scatterchart and Streamgraph

6. Conclusion and Future Work

Faced with the challenge of data visualization imposed by the amount of existing information and the difficulty of clearly maintaining data patterns and trends over time and as new data is obtained in real time, we proposed a set of multiple transition techniques with smooth animations that allow the transformation and movement of the data to be followed along various visualization techniques associated to different aggregation levels, velocities and time intervals, in order to maintain a coherent state of the visualization.

To validate the conceptualized techniques, we implemented a browser-based prototype following a graceful degradation metaphor. This prototype allowed the development of the transitions and the creation of efficiency and usability tests. We tested the prototype with 100, 200 and 1000 data points per second and obtained the best records using a history of 3 minutes, receiving 200 data points per second (720000 data points per hour), even though it's not a big flow, it was sufficient to check the efficiency of the developed transitions. Usability tests were conducted with 28 participants and showed that to achieve some users' preferred techniques there is still space for efficiency improvements as a future work and also showed that the number of complex operations and variations in sizes and colors is directly related to the breaks of the frames per second and therefore to the fluidity of the system.

As future work, we intend migrate the system into another technology like WebGL since it can use the GPU to speed up processing. It is also proposed to create a set of animated and smooth transitions, to follow the changes of a specific visualization type inside a module, as well as the consecutive change of the created transitions between modules described in this dissertation.

References

[1] S. M. Ali, N. Gupta, G. K. Nayak, and R. K. Lenka. Big data visualization: Tools and challenges. In *2016 2nd International Conference on Contemporary Computing and Informatics (IC3I)*, pages 656–660. IEEE, 2016.

[2] J. Alsakran, Y. Chen, D. Luo, Y. Zhao, J. Yang, W. Dou, and S. Liu. Real-time visualization of streaming text with a force-based dynamic system. *IEEE Computer Graphics and Applications*, 32(1):34–45, 2011.

[3] L. Battle, M. Stonebraker, and R. Chang. Dynamic reduction of query result sets for interactive visualizaton. In *2013 IEEE International Conference on Big Data*, pages 1–8. IEEE, 2013.

[4] D. M. Best, S. Bohn, D. Love, A. Wynne, and W. A. Pike. Real-time visualization of network behaviors for situational awareness. In *Proceedings of the seventh international symposium on visualization for cyber security*, pages 79–90. ACM, 2010.

[5] G. Fialho. Vismillion and change (thesis). 2018.

[6] F. Fischer, F. Mansmann, and D. A. Keim. Real-time visual analytics for event data streams. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pages 801–806. ACM, 2012.

[7] M. Hao, D. A. Keim, U. Dayal, D. Oelke, and C. Tremblay. Density displays for data stream monitoring. In *Computer Graphics Forum*, volume 27, pages 895–902. Wiley Online Library, 2008.

[8] U. Jugel, Z. Jerzak, G. Hackenbroich, and V. Markl. M4: a visualization-oriented time series data aggregation. *Proceedings of the VLDB Endowment*, 7(10):797–808, 2014.

[9] M. Krstajić, E. Bertini, F. Mansmann, and D. A. Keim. Visual analysis of news streams with article threads. In *Proceedings of the First International Workshop on Novel Data Stream Pattern Mining Techniques*, pages 39–46. ACM, 2010.

[10] 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*, pages 41–47. IEEE, 2013.

[11] 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*, volume 8294, page 829404. International Society for Optics and Photonics, 2012.

[12] T. Repke and R. Krestel. Topic-aware network visualisation to explore large email corpora. In *EDBT/ICDT Workshops*, pages 104–107, 2018.

[13] J. Traub, N. Steenbergen, P. Grulich, T. Rabl, and V. Markl. I2: Interactive real-time visualization for streaming data. In *EDBT*, pages 526–529, 2017.

[14] D. F. Williamson, R. A. Parker, and J. S. Kendrick. The box plot: a simple visual method to interpret data. *Annals of internal medicine*, 110(11):916–921, 1989.

[15] J. Xia, F. Wu, F. Guo, C. Xie, Z. Liu, and W. Chen. An online visualization system for streaming log data of computing clusters. *Tsinghua Science and Technology*, 18(2):196–205, 2013.