FastGeo - Visualizing large amounts of geospatial data in real time

João Rafael
Instituto Superior Técnico
Lisbon, Portugal
joaolucasrafael@tecnico.ulisboa.pt

ABSTRACT

The availability of devices that can record locations, and their connection to the Internet, creates a large amount of geospatial data that are continuously being streamed. However, the quantity of data produced makes visualizing it a challenge. Its processing has to be fast enough to handle a constant stream of new data. The visualization methods must also be displayed at interactive frame rates. To study a way to show this data in a way that is visually clean but efficient enough to handle large amounts of data being streamed in, we conceived FastGeo to visualize this data by displaying it in three different periods. We gradually simplify data, using different representations in each period, such as heatmaps and trajectory bundling, so that each period looks distinct and maintains good performance. We developed a prototype to implement these concepts and tested it with users, and we concluded that they understood the way data was represented. We evaluated this prototype's performance as well, which let us conclude that although the visual performance was effective, the scalability of trajectory bundling is limited and requires further improvement.

Author Keywords

Data Visualization, Visual Analytics, Trajectory Visualization, Geospatial Data.

INTRODUCTION

With the advancements in computing that have happened over the past few decades, access to location data has become increasingly available. Everyone can know where they are in a few seconds, using modern GPS mechanisms that have errors in the order of fewer than ten meters. The assessability of this kind of data has led researchers to study it in the field of Big Data, which is an increasingly studied area in Computer Science, and the amount of geospatial data is by no means small [5].

We have not only large amounts of data at our disposal, but this data is also produced in real-time. By visualizing this data, we can identify trends. Geospatial data can be divided into subgroups such as spatial events (combinations of locations and time intervals) or trajectories (sets of positions with associated timestamps). Trajectories can show where someone has been, and they may show patterns when in great numbers.

Trajectory data, however, is complex. Therefore, we must find some way to visualize it so that it is easy to understand. When there is a massive amount of it, visualizing trajectory data is no small feat, though, and there are several challenges regarding how to show and analyze it. The more constraints we put on this data (such as there being a large amount of it), the more problems arise.

This is the most important problem: we have to show large amounts of data, make it understandable, and avoid visual clutter, but ensure that we are not simplifying it too much as well. Moreover, if we want to visualize the data in real-time and show the temporal side of trends and patterns, more problems will arise, such as how to differentiate more recent data from older data or how to view recurring events and changes in routines over time.

Motivation

If we solve the problems that come with visualizing trajectories in large amounts and in real-time, we will end up with a visualization that can show spatiotemporal patterns, as we have discussed. If we can observe these patterns, we can study them and ask ourselves questions about why they happen.

Urban planners can study these patterns and try to identify roads that need more lanes due to being overly congested, and when they build these lanes, they can use this data to identify what streets and roads to use as detours to minimize the impact of construction on traffic. Advertisers can study the most congested streets during rush hour and focus on those streets when putting up billboards. If people can visualize this data successfully and easily, they can understand patterns and use this understanding for myriad goals.

Objective

Our work's primary objective is **to study how to clearly visualize large numbers of trajectory data as it is streamed in real-time, allowing the recognition of spatiotemporal patterns while the data is updated over time.**

To achieve this objective, we will have to show time and space clearly, enabling the user to analyze data without feeling overwhelmed by its quantity. Since we are handling large amounts of data, computational performance is a concern as well. Before data is displayed, it has to be processed, and this processing has to be fast enough to keep up with regular data updates. Data will be updated gradually, the amount of it growing over time. We will use a graceful degradation concept, which gradually simplifies data as it gets older and reduces the amount of data in a controlled manner [6]. This gradual simplification consists of dividing time into different periods and simplifying the data when it transitions from more recent periods into older ones.

CONTRIBUTIONS

To achieve our objective, we tested different ways to visualize data and tested them with users to evaluate their usability. Additionally, we prototyped a system that would handle the data processing required for each visualization and simulate real-time data streaming using a static dataset. Our contributions are the following:

- Several visualization techniques that use trajectory bundling and grid binning to represent data.
- Time period concept that gracefully degrades geospatial data, transitioning it between each time period and representing each period with a different visualization technique.
- A prototype system to process trajectory data and prepare it for visualization.
- Testing analysis regarding the understandability of these visualization techniques.
- Performance evaluation for the data processing systems.

RELATED WORK

In this section, we will discuss the state of the art in Visual Analytics (VA) related to trajectory and movement data. We will start by introducing some basic concepts related to this area and then analyze and discuss several approaches to visualizing the aforementioned types of data.

There are three fundamental types of spatiotemporal data [1]. Trajectories are the most complex data type of the three (and this work's main focus), describing positions of moving objects at sampled time moments.

There are many ways to visualize spatiotemporal data. While point-based visualizations are the most direct type of visualization for presenting and analyzing it, discrete sample points can be converted into continuous line forms that display paths from objects such as vehicles, humans, or marine life. Such line-based methods can also encode related variables with visual channels such as width or color hue, saturation or luminance, facilitating the depiction of spatial patterns [9].

Real-Time and Streaming

There is little research related to visualizing spatiotemporal data in real-time. The simplest way to represent this data is by showing points for each object's current location and lines representing where they have been [14].

Since GPS data often has associated errors, it has to undergo preprocessing, such as cleaning inaccurate, duplicate, or unrealistically high-speed points, or precomputation of attributes such as distance and speed. These attributes can be represented by color, for example. The points can be animated in real-time as well [7]. To ensure interactive framerates when handling larger amounts of data, GPU acceleration may be required for rendering this data.

When focusing on smaller scales, such as a single intersection, different attributes can be displayed, such as the direction each object enters and exits the intersection or the type of object (be it a car, bicycle, or pedestrian) [8].

We can find more works that deal with streamed data visualization should we look beyond geographical data. One paper [6] uses a concept called graceful degradation, which gradually simplifies data as it gets older, doing so in a controlled manner by splitting time into different periods.

Large Amounts of Data

Should we remove the constraint of real-time in visualizing large scale geospatial data, we will find many more works. In these works, we find a greater variety of methods being employed.

A simple way of showing large amounts of trajectory data is to represent it in a low opacity [17]. This may make a map less visually cluttered, but there are more ways of reducing clutter, namely using aggregation.

Aggregation can help deal with large amounts of data, distill noise, and enable an overall view of multiple movements' spatiotemporal distribution. Clustering algorithms function well in loading large amounts of data, which facilitates displaying macro information without data loss [1].

One such type of aggregation is spatial heatmaps, which create continuous surfaces by interpolating discrete points. These heatmaps can be used on point data [4] or line data [17], but the result in both cases is to simplify the data visually and show only the heatmap surface. Cluster heatmaps, which represent data in discrete squares, are often used as well, though for non-spatiotemporal data [16].

Similar to spatial heatmaps, density maps aggregate data into continuous surfaces, though this is done by creating density fields that aggregate smoothed trajectories, as opposed to interpolating points. Color mapping can be used to encode variables, such as the time of day [12].

Map matching can be used to encode streets with data, as opposed to representing individual trajectories. The result is that different streets and roads have a different color based on how congested they are [15]. This data can be represented as a graph as well, showing only a graph view apart from the map [10].

Edge bundling clusters similar lines and represents them as only one, greatly reducing visual clutter. It is more commonly used for origin-destination flows (sets of disconnected lines) and not full-fledged trajectories. However, it can be used for this type of data, producing results somewhat similar to the map matching-based techniques described above (Below, we will refer to the edge bundling of complete trajectories as trajectory bundling for clarity). Width can be used to represent how many lines are contained in each bundled line [13].

Of course, space is not the only important aspect to represent. Several of these works also represent time in their own ways. For example, space-time cubes, which are used in VAUD [4], transform a 2D spatial view into a spatiotemporal 3D view, with the z axis representing time.

In some works, time is represented by separate temporal views displaying line charts [16, 10]. In others, time may be encoded into radial charts for the 24 hours of the day [3], or color gradients [13].

Discussion

From what we have discussed above, we can see a distinction between visualizing geospatial data in real-time and visualizing large amounts of it, as no work does both of these things at the same time. Papers that propose real-time visualization focus on representations that do not scale visually (representing points and lines with no aggregation) and papers that present Big Data visualization in ways that reduce clutter and perform well are restricted to static datasets. Ideally, we want something that can do both of these things, showing data evolving in real-time, differentiating recent data from older, yet also maintaining an uncluttered map view.

Additionally, concepts that are used in specific areas could be adapted to our problem. For example, the concept of graceful degradation [6] can be used for geospatial data, and edge bundling [13] could be adapted to work in real-time.

FASTGEO

In this section, we present the different visualization techniques and the data processing algorithms that support them.

As we have already discussed, the goal of this work is to visualize large amounts of data in real-time. To do so, we divide time into different periods and represent each period using different visualization techniques. As data passes from one period to another, it will be gracefully degraded [6], aggregating trajectories and gradually reducing the amount of memory being used (Fig. 1).

Time Periods

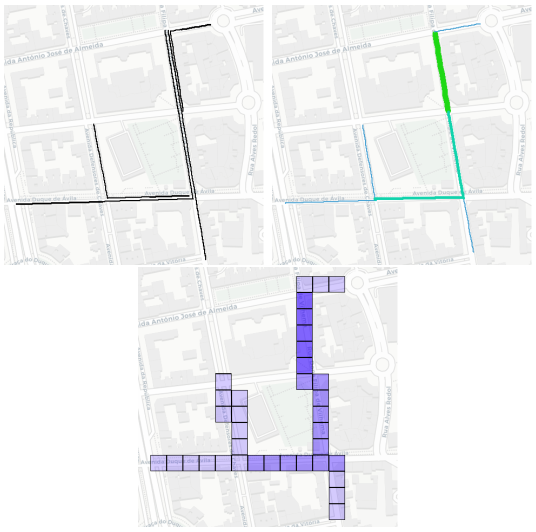
Figure 1. Data undergoing graceful degradation (left to right to bottom).

If we are to visualize large amounts of data as they are streamed in real-time, we will quickly realize that we cannot keep the entire detail of every single datum we receive. Therefore, we must find a way to gradually simplify the data to save on memory and keep a readable visualization. This is where the concept of time periods comes into play.

Time periods allow us to group the data received based on different factors, displaying each group differently. As data passes through each period, it loses detail, allowing it to occupy less memory without losing so much information that it becomes impossible to understand. The idea is to gracefully degrade data as it becomes older.

Each period represents data with specific granularity. Naturally, we want to do this as the data gets older since this is a streaming visualization, and we are more concerned with exactly what is happening currently than the exact happenings of two years ago. The most recent data, therefore, is what should have the most detail possible.

We displayed the time period representations on a low-contrast, mostly white and light-grey map, as we wanted to use color in our representations and did not want to use a more colorful map that might confuse users. Each time period uses different representations.

After consideration, we conceived three periods.

Ongoing

The ongoing period is the most detailed, containing trajectories as they are received. It corresponds to trajectories that have not ended: if an object stops, its trajectory until that point will no longer be part of this period and be moved to the recent period. If it starts to move again, then its trajectory, starting with the point where it last stopped, will be part of the ongoing period.

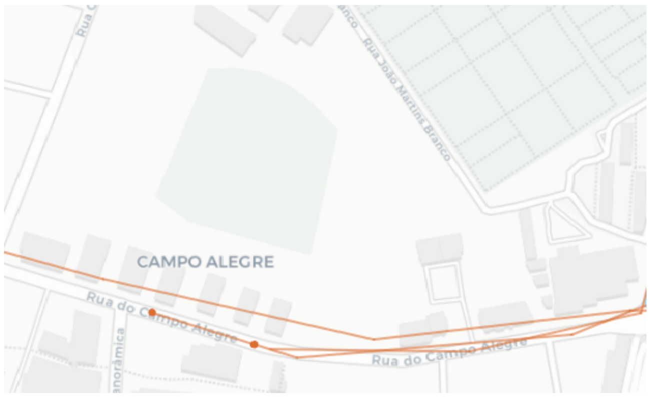

Figure 2. Representation for the ongoing period.

This period's data is not simplified, so it is shown as-is. Like in a paper focused on real-time visualization we discussed [7], we show each object's current location with a small circle, and its complete trajectory is shown using lines (Fig. 2). To make it clear to the user that data in the ongoing period corresponds to something that is currently moving, we animate lines: rather than appearing in their entirety immediately, the animation mimics a movement from one point of the line to the other.

Recent

The recent period covers an adjustable period of time (e.g., stopped trajectories to one hour ago, or stopped trajectories to one day ago). Instead of keeping a trajectory's complete data, we will use trajectory bundling to reduce memory usage and visual clutter.

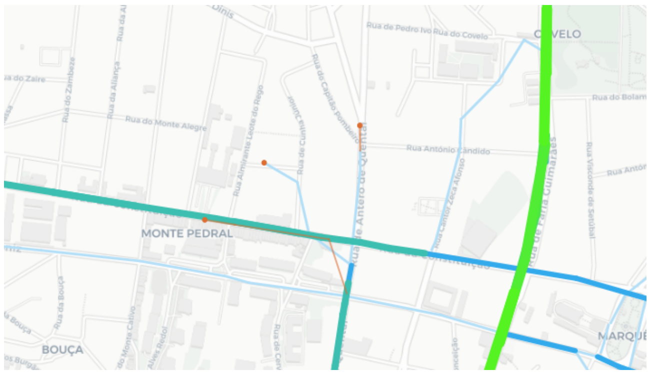

Figure 3. Representation for the recent period.

In this period, lines are simplified with trajectory bundling, and therefore each bundled line has information regarding how many lines are bundled in it. We encode this attribute with both width and color, with a color scale that ranges from blue to green (Fig. 3). The brighter green helps draw the user's eye immediately to streets with higher traffic, while width is used to make segments with low total thinner so as not to clutter the screen too much.

In both recent and ongoing periods, the size of both lines and circles adapt to how zoomed in the map is. This is done so that lines are never too thin or too wide for whatever zoom level is on the map at any given moment, avoiding occlusion problems (which happen if the map is zoomed out and lines are too wide, covering each other) or lines being too small when zoomed in (since if a user zoomed in they would probably like to look at a line in more detail).

History

The history period is the final accumulator for the data, and every line that is older than the recent period's limit is represented here. Rather than storing lines like in other periods, in the history period, we fit each line into the squares of a grid encompassing the entire dataset's boundaries.

For this history period, we developed two representations. One such representation uses a cluster heatmap, which consists of a square matrix grid where each grid tile is shaded on a color scale to represent an attribute's value. This is a clear fit for grid binning, as it simply consists of shading each grid square in a different color (Fig. 4).

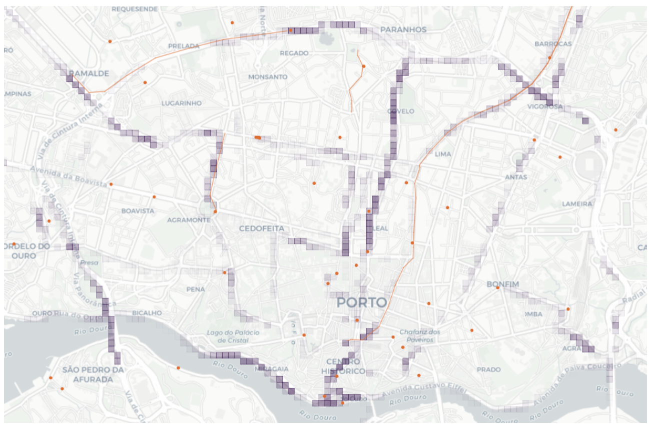

Figure 4. Cluster heatmap representation for the history period.

Whereas the cluster heatmap presents data in discrete squares, the spatial heatmap creates continuous surfaces from interpolating discrete points. This is also a good fit for grid binning, as each grid square can be represented as a point and then interpolated into one such continuous surface (Fig. 5).

In both representations, the different resolution grids are used in conjunction with map zoom (Fig. 6): as the user zooms out, the visualization will show a lower-resolution version, while zooming in shows a higher-resolution version. This helps the grid detail accompany the map's overall detail.

Overview

The three representations are all rendered, the history period being drawn below the recent period, which is then drawn below the ongoing period (Fig. 7). This

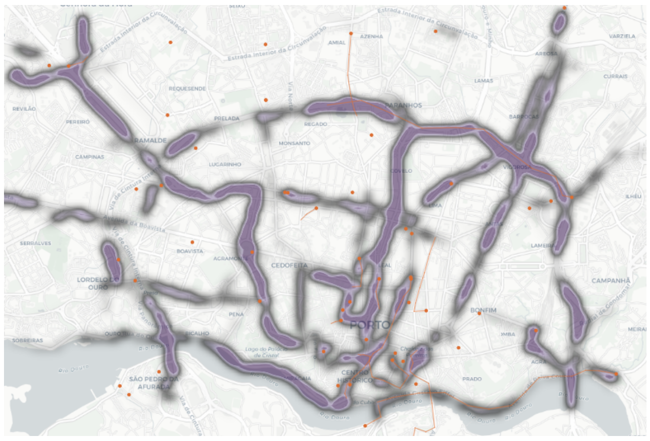

Figure 5. Spatial heatmap representation for the history period.

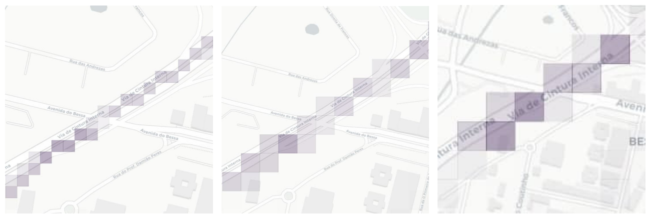

Figure 6. Different grid levels in the history period. As the user zooms out (left to right), the grid will decrease in resolution.

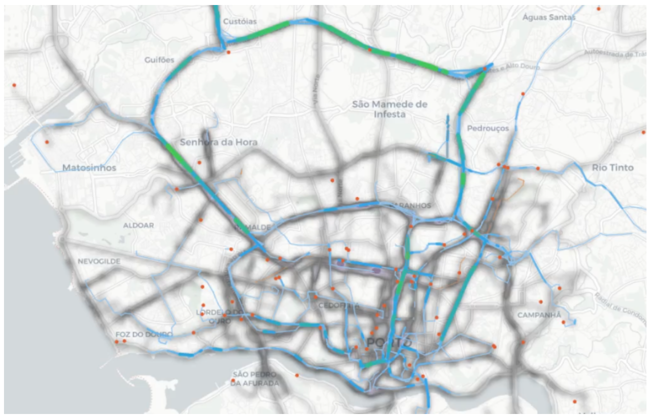

Figure 7. Visualization with all periods.

order is the same as period recency, as we want the more recent data to stand out.

Trajectory Bundling

As we have discussed, we use trajectory bundling in the recent period to simplify data by bundling similar lines into one, this single line storing data related to how many lines it contains.

The trajectory bundling algorithm described in Understand My Steps [13] is well-suited for a static visualization and dataset, but we are more constrained. Since new data is added every few seconds, trajectory bundling must be performed at the same rate. Moreover, this data may constantly be accumulating, meaning that even if the algorithm is extremely optimized, it will eventually become too slow as more data is streamed in, making it so that bundling cannot be performed at a steady rate.

Therefore, there were two important problems to address when creating our trajectory bundling algorithm: we want to focus on efficiency, to make sure we can process data as fast as possible, and data filtration, to have it deal with the least data possible and avoid the necessary time for its execution from growing too much as the amount of data increases.

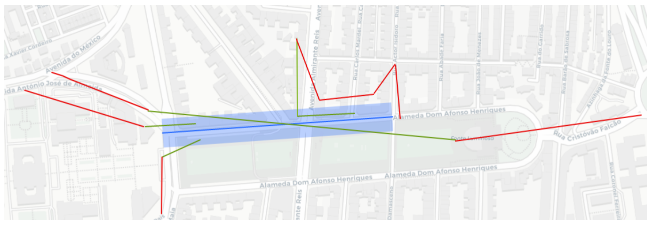

Figure 8. Data filtration for trajectory bundling. Green segments intersect with the blue area and are compared with the blue line, while red segments are not.

To reduce the number of comparisons, we filter the data using two criteria. We only compare segments that are close to each other (like in Fig. 8), as segments that are too distant consist of overly different trajectories (e.g., different streets). Additionally, we do not compare segments whose angles are too different, as they will likely consist of different trajectories as well (for example, segments on intersections or roundabouts may be close to each other, but their bearings are likely different).

After comparisons, segments may be merged. The results will vary depending on the segments: there may be a full merge, different types of partial merges (Fig. 9), or no merge at all.

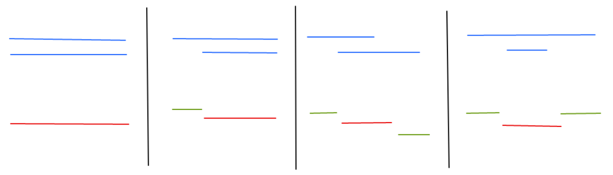

Figure 9. Different results of trajectory bundling merges. Blue segments are the originals, red segments are the fully merged part of both segments, and green segments are unmerged parts of the original segments.

Grid Binning

In the history period, lines are processed through grid binning. The result is that the notion of individual lines is lost from the data; only a set of grids are stored. As opposed to one, several grids are stored because we will show a different grid based on zoom level. If the user is more zoomed in, then a higher-resolution grid will be shown; otherwise, a lower-resolution grid will be shown.

The grid binning process consists of dividing each line into a set of points through interpolation, then fitting each point into a grid square (Fig. 10). As more data is added, the grid may expand, but only if the amount of data outside the grid is significant.

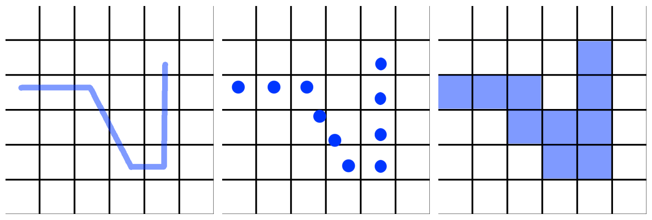
Figure 10. Lines being binned into the grid. On the left, the starting lines. In the middle, the resulting points from interpolation. On the right, the resulting grid.

Prototype

In the prototype we have created to study these concepts, each time period has its own Python process, ensuring that we take full advantage of multi-core CPUs. A node.js server manages these three processes and handles communication with the visualization.

For the visualization side of the prototype, we used HTML, CSS, and JavaScript. This includes the Mapbox JavaScript library, which displays maps using WebGL, fully taking advantage of GPU acceleration to display large amounts of geospatial data on-screen (like in a paper we discussed above [7]).

EVALUATION

In this chapter, we present how we tested the prototype regarding both performance and usability. We start by detailing the performance tests we made and draw comparisons between different versions of the prototype and discuss its limitations. Afterward, we describe how the prototype's usability was tested and present the test results.

Performance

We analyze two main facets of performance: server-side, that is, how fast it can process data, and the framerate on the client-side, that is, how fluid the visualization looks to the user.

Looking at the frame rate graphs (Figure 11), we can see that the frame rate maintains close to 60 frames per second (FPS) most of the time, periodically having moments where framerate diminishes. These dips are somewhat larger with the cluster heatmap representation for the history period, and they correspond to a short, periodic stutter when the visualization is receiving new data and processing the animations. When the visualization receives data, the CPU becomes busy with updating the visualization and cannot produce enough frames to keep the frame rate stable. When interacting with the visualization, it runs at a stable near-60 FPS, but it seems to have very short hitches when panning or zooming, which

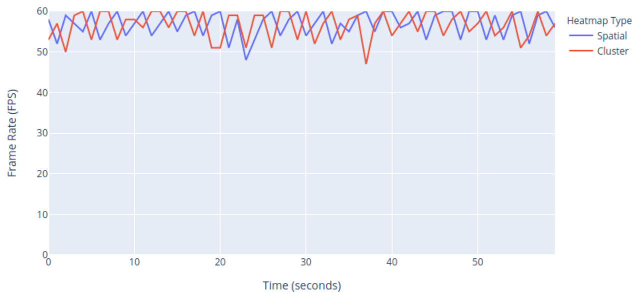

Figure 11. Framerate graph for both history period representations.

correspond to data updates. These stutters are slightly longer when using the cluster heatmap representation due to the larger volume of data received.

We can conclude that the visualization maintains a mostly stable framerate, with the occasional stutter. While these stutters detract slightly from the interaction's fluidity, the frame rate is otherwise extremely smooth.

The trajectory bundling we use for the recent period is based on an algorithm described in a previously discussed work [13] but optimized for periodic data updates. We decided to evaluate our improvements by running the prototype with each algorithm. The results (Figure 12) make it easy to conclude that our improvements greatly optimized the Understand My Steps algorithm, as the time the algorithm takes to process the data is much lesser and grows much less over time.

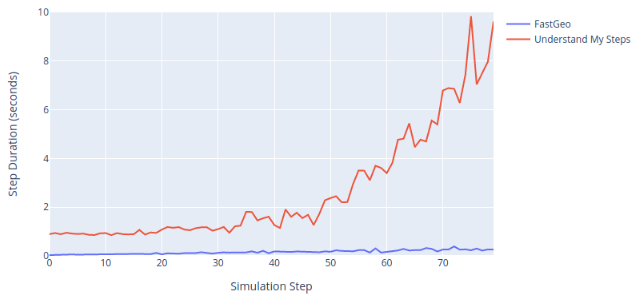

Figure 12. Comparison of bundling time between unoptimized (red) and optimized (blue) algorithms.

Our prototype simulates the passage of time by dividing it into discrete steps. If a step takes one second, then the next one will simulate 1 second of time passing. If a step takes 5 seconds, then the next one will simulate 5 seconds. We would ideally like to keep each simulation step below 5 seconds.

After simulating 24 hours of a dataset from a taxi company in Porto, Portugal [11], we conclude that the step duration rises over the 5 second goal as time passes. The vast majority of this time is spent on the recent period, performing trajectory bundling. If the prototype only consisted of an ongoing period and a grid binning-based history period, each step would seldom go above 0.5 seconds.

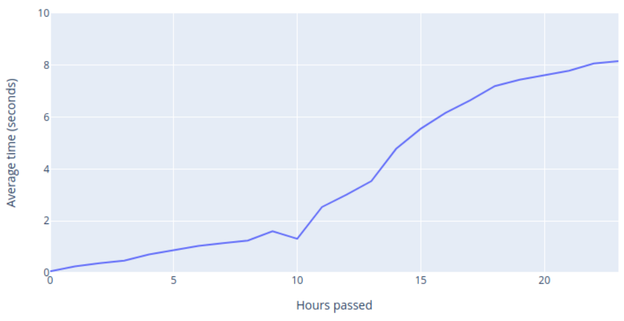

Figure 13. Average simulation step time at each hour.

The slow performance in the recent period is due not to the algorithm we are using but to database usage. We use a PostGIS database to perform data filtration and greatly reduce the number of comparisons made in bundling, but this comes at the cost of heavy overhead. This means that the most limiting factor is how long the recent period spans, as a longer interval implies a larger database table. Comparing different intervals with the hour with the most traffic, we can see that a smaller interval for the recent period results in much-improved performance.

Despite all the optimizations that have been made to the trajectory bundling algorithm, its performance is somewhat short of meeting our goals. Only if the recent period consisted of a very short interval is the algorithm fast enough to handle incoming data without too much slowdown. As for the other two periods, their performance is more than fast enough to meet our goals.

Usability

When evaluating the prototype's usability, our goal was to observe if users understood the difference between each data representation, if they were able to identify trends, and if they were able to understand how trends changed over time. We also wanted to compare the two representations we created for the history period.

We performed tests with 21 people, resulting in 42 tests for both prototypes. Each participant was asked to perform 10 tasks (two of them twice, once for each history period representation) using the aforementioned Porto dataset. For each task, we measured the elapsed time, the success or failure in completing it, and the user's difficulty rating from 1 (very difficult) to 5 (very easy). After the tasks were performed, participants were asked to fill out a questionnaire with time period-related questions, as well as the System Usability Scale (SUS) and the Raw NASA Task Load Index (TLX) tests.

Tasks

When considering what tasks to require of users, we wanted the full set of tasks to have a good variety, comprising the evaluation of how users understand each individual period and the combination of them on a map that can change over time. We wanted to evaluate how users understood space and time as it changes on this prototype.

We started by creating a basic task for each period (ongoing, recent, history) to make sure users understood each period's basic representation (for the history period, this task would have to be made twice, once for each of the heatmap types). Afterward, we created more complex tasks requiring the user to compare different periods and compare the trends at different times or different parts of the map. Finally, we created two less constrained tasks to enable a more free-form analysis.

The final tasks, therefore, are:

1. Follow a specific vehicle and point out when its trajectory has transitioned to the recent period.
2. Indicate the most-used street recently.
3. Indicate the area that has seen the most traffic overall (this is, not just recently).
4. Explain whether, recently, there has been more traffic outside or inside the city.
5. (After showing the prototype at two different times in the simulation,) explain whether trends in traffic have changed between these two times or not.
6. Indicate an area that was used in the past, but not recently.
7. Go to Ponte da Arrábida (if the user did not know where this was, we explained it was the westernmost bridge in Porto) and indicate how many vehicles passed through it recently.
8. Indicate one vehicle that does not follow the trends displayed by the visualization.
9. A scenario will be shown. This scenario will have a gradual but significant change. When it happens, indicate it.
10. Explore the visualization and make any observations.

Task 3 and 6, which focus specifically on the History period, were performed twice (once for each representation) to compare both representations' clarity and understandability (naturally, for each representation, a different timestamp was selected to have different answers in each).

To avoid the order of questions possibly impacting statistics (for example, task 1 being negatively affected by users' lack of familiarity with the prototype), each user performed the tasks in a different order. This order was decided by Latin squares, which, given n tasks, present n rows and n columns of different orders to perform them.

Since we also did not want certain participants to be hindered by particularly difficult tasks being the first, we divided tasks into three sets with incremental difficulty, the order of tasks within each set following a Latin

Square design: the tasks start with an order of tasks 1 through 3, then another order of tasks 4 through 8, then task 9 and finally task 10.

Task 10 is a freeform task where the user is asked to make any observations about anything they desire. In this task, we did not impose a time minimum or maximum or a correct answer. The users were simply asked to voice their thoughts about whatever parts of the map or trends they noticed until they felt they had no further observations to present.

Results

Our statistical analysis consisted of finding any significant differences between the two different representations and between each task. We evaluated, for each task, the elapsed time, success, or failure in the task completion, and each user's reported difficulty from 1 (very difficult) to 5 (very easy). Additionally, since we asked each user questions after they completed the tasks, we used the results for these questions. For each graph, Tasks 3/C and 6/C correspond to Tasks 3 and 6 with the cluster heatmap representation of the history period, while 3/S and 6/S correspond to the same tasks with the spatial heatmap representation.

To analyze time and difficulty, we performed the Friedman test. In case this test indicated significant differences, we performed post-hoc tests, more specifically Wilcoxon tests on pairwise comparisons, with Bonferroni corrections. For test success, which is a dichotomous variable (success/failure), we used McNemar's test.

Comparing the two different representations, the only significant difference between both representations was found in the difficulty of Task 3. The spatial heatmap representation was the most successful overall: comparing the statistically significant difficulties in Task 3, more specifically, their means with a confidence interval of 95%, the spatial heatmap variant of the task has values between 3.96 and 4.42, while the cluster heatmap variant has values between 3.39 and 4.04.

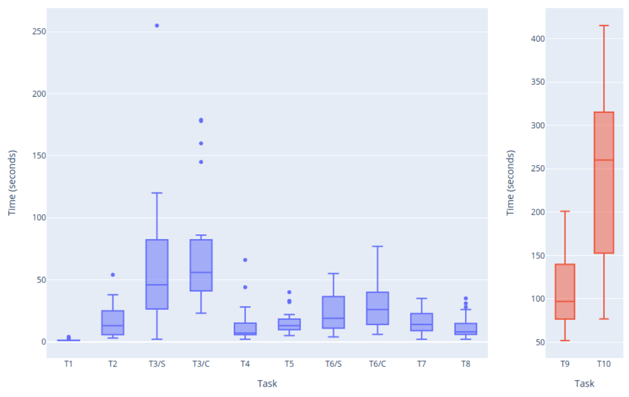

Figure 14. Time distribution for each task.

Comparing all tasks, the most noticeable conclusion was that the time for Task 1 was significantly different from all the other times. This is due to a specific detail of measuring time in Task 1: the data for elapsed time in this task consists of how long it took the user to notice the track changing period. As for the other tasks, Task 10 is by far the longest on average. Other tasks that have several significant differences are both variations of Task 3 and Task 9. Apart from Task 10, these are the longest.

Looking at significant differences in difficulty, the pairwise comparisons indicate that they happen between two groups: Tasks 1, 2, 7, and 8, and Tasks 3 (both variants) and 9. These two groups are different extremes in difficulty: the first corresponds to the generally easier tasks (none of their values are below 4.63), and the second corresponds to generally harder tasks (none of their values are above 4.27, and the minimum lower bound is 3.39 in the cluster variant of Task 3).

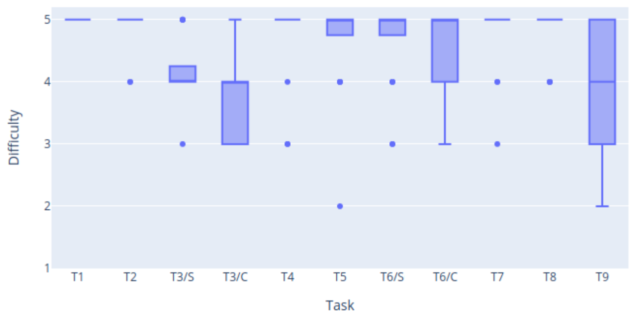

Figure 15. Difficulty distribution for each task. For this attribute, 1 was considered "very difficult", and 5 "very easy".

Considering success, the cluster heatmap variant of Task 3 had the lowest success rate, with the spatial heatmap variant of Task 3, Task 4 and 9 having a slightly higher rate, and all other tasks having 100%.

Overall, the most challenging tasks were Task 3 (both variants) and 9, and the most time-consuming consisted of the same group as well, with the sole addition of Task 10. Considering the different representations in the History period, the easiest to understand was the spatial heatmap variant.

After the tasks were performed, participants rated how well they understood each time period in a Likert Scale. We used the Friedman test on the results and concluded that there were significant differences ($p = 0$). After performing pairwise Wilcoxon tests with Bonferroni corrections, the cluster heatmap representation had a significant difference to every other representation/period, as it was the lowest-rated one for this question.

We also asked users to take the System Usability Scale, a set of ten questions that measure the prototype's usability. Having a score above 87.5 would correspond to the top fourth of responses [2]. The average SUS score for the prototype was 90.12.

Aside from SUS, users were asked to take the NASA Task Load Index as well. This test consists of six sub-scales that measure different facets of a task, overall letting users rate the perceived workload. The six subscales are used to calculate a final workload score: our result was 22.58.

Observations and Feedback

Apart from statistical data, the user testing process provided us with many observations regarding how users interacted with the prototype, as well as their opinions and feedback.

While the statistic analysis we performed shows a clear preference for the spatial heatmap representation, it is not unanimous. Many people preferred the cluster heatmap. The preference came down to whether a user preferred the more immediately understandable differences of the spatial heatmap (due to its wider range in color) or the higher quantity of information the cluster heatmap provides (since it lets users hover over each square and get the exact number of objects that passed through it).

Task 10, as we have discussed, was mostly open-ended, allowing users to explore the prototype freely and pointing out anything they wished. Their answers showed more than a simple understanding of the visualization. Rather than only understanding the base spatiotemporal context, they think not only of parts of time and space with no interconnection but of this data as the real traffic patterns of a city and connect these separate observations to form conclusions about how traffic flows throughout the city over time.

Users suggested that the map showed some points of interest (such as the aforementioned airport) and a feature that let them filter out a period and analyze it individually. While some users were pleased with the color choices, others suggested using more colors for the color scales, specifically for the History period (different users would suggest this for different representations).

Discussion

Given the results we obtained during the user testing process, we can conclude that users were generally able to understand how the prototype worked, the differences between each period, and how trends could evolve over time. Our goal was to make a visually clear and understandable way to show trajectory data, and users were not confused with the representations we created.

Participants had no difficulty understanding the ongoing and recent periods, though the History period was somewhat harder to understand, with tasks 3 and 6 focusing on it and having somewhat inferior results compared to the others.

We conclude that the **spatial heatmap** was both generally preferred by the users and easier to use, with better results regarding difficulty. Some users did prefer the **cluster heatmap** representation, however. Considering the SUS and NASA-TLX scores, we can conclude that the prototype's usability was overall excellent.

The visualization's performance was acceptable, only having small stutters when updating data and keeping a steady 60 FPS when not doing so. While trajectory bundling was greatly optimized for use in a real-time streamed data setting, limitations left its performance short of our goals, with scalability being a problem as well.

Given the performance, the recent period has to be kept at a short interval (30 minutes). However, this is still a lot of data, and it allows users to understand what is recent and what is not. However, the performance was well within our goals regarding processing for the History and Ongoing periods.

We previously saw that the field of real-time visualization of trajectory Big Data was limited. With this work and its results, we conclude that users can understand this type of data visualization and that processing it fast enough is mostly feasible (with some limitations).

CONCLUSION

After studying current works that try to solve the problem of real-time visualization of large amounts of geospatial data, we decided that to handle data as it is streamed in, we would have to divide this data into different periods of time and represent them in visually distinct ways. We then studied ways to represent each period. We used trajectory bundling for one of these periods, which bundles similar lines and represents them as one. For another period, we used grid binning, which creates a grid where each square has data pertaining to how many objects passed through it.

We conceived three different time periods: Ongoing (trajectories happening currently), Recent (trajectories of objects that stopped recently), and History (trajectories of objects that stopped a longer time ago). For the Ongoing period, we showed the data as-is. As for the Recent period, we presented the lines produced by trajectory bundling. Finally, we created two representations for the History period: a cluster heatmap that presents data in discrete squares and a spatial heatmap that creates continuous surfaces.

After developing a prototype to implement these techniques, we evaluated its usability by inviting users to perform a set of tasks and compare each History period representation. Each user answered a questionnaire, including the raw NASA-TLX and SUS tests, which let us conclude that our prototype's usability was overall very good. We also evaluated performance. The visualization's performance (frame rate) was very stable, with slight stutters when loading data. While we optimized our trajectory bundling algorithm significantly compared to Understand My Steps [13], its scalability of the Recent period was below our objectives.

While there were limitations regarding processing performance and scalability, these were only for one of the periods we developed. All other goals were met, meaning we succeeded at creating a visualization that shows large amounts of data as it evolves in real-time, and this visualization was easily understandable.

Future Work

Given the results of the performance and usability tests, we believe some changes could be performed.

The prototype structure could be changed so that trajectory bundled data was stored in memory and not the database, resulting in faster access times.

Additionally, data processing in the visualization could be further optimized to avoid stuttering when new data is received. The way animations for the Ongoing period line updates are prepared could be further optimized as well.

Regarding usability, further improvements could be made to the representations in the history period. For the spatial heatmap, adding a hover tooltip to indicate how many objects a certain part of the heatmap represents. For the cluster heatmap, the color scale could be changed to make it easier to differentiate squares.

Finally, the graceful degradation concept we discussed could be further explored. More granular forms of this degradation, allowing for more and more configurable periods, could be made. Our modular time period concept could be built upon to incorporate periods with different types of data simplification or different representations.

REFERENCES

1. G. Andrienko, N. Andrienko, P. Bak, D. Keim, and S. Wrobel. *Visual Analytics of Movement*. Springer, Berlin, Dec 2013.
2. Aaron Bangor, Philip T. Kortum, and James T. Miller. An empirical evaluation of the system usability scale. *International Journal of Human–Computer Interaction*, 24(6):574–594, 2008.
3. S. Chen, X. Yuan, Z. Wang, C. Guo, J. Liang, Z. Wang, X. Zhang, and J. Zhang. Interactive visual discovering of movement patterns from sparsely sampled geo-tagged social media data. *IEEE Transactions on Visualization and Computer Graphics*, Jan 2016.
4. W. Chen, Z. Huang, F. Wu, M. Zhu, H. Guan, and R. Maciejewski. VAUD: A visual analysis approach for exploring spatio-temporal urban data. *IEEE Transactions on Visualization and Computer Graphics*, Sep. 2018.
5. Z. Deng, Y. Hu, M. Zhu, X. Huang, and B. Du. A scalable and fast optics for clustering trajectory big data. *Cluster Computing*, Jun 2015.
6. G. Fialho and D. Gonçalves. Visbig: Visualizar bigdata em tempo real. Master's thesis, Instituto Superior Técnico, Oct 2018.
7. G. A. M. Gomes, E. Santos, and C. A. Vidal. Interactive visualization of traffic dynamics based on trajectory data. In *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images*, Oct 2017.
8. H. Guo, Z. Wang, B. Yu, H. Zhao, and X. Yuan. TripVista: Triple perspective visual trajectory analytics and its application on microscopic traffic data at a road intersection. In *2011 IEEE Pacific Visualization Symposium*, March 2011.
9. J. He, H. Chen, Y. Chen, X. Tang, and Y. Zou. A review of variable-based spatiotemporal trajectory data visualization. *IEEE Access*, 2019.
10. X. Huang, Y. Zhao, C. Ma, J. Yang, X. Ye, and C. Zhang. TrajGraph: A graph-based visual analytics approach to studying urban network centralities using taxi trajectory data. *IEEE Transactions on Visualization and Computer Graphics*, Jan 2016.
11. L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas. Predicting taxi–passenger demand using streaming data. *IEEE Transactions on Intelligent Transportation Systems*, 14(3):1393–1402, 2013.
12. R. Scheepens, N. Willems, H. van de Wetering, and J. van Wijk. Interactive density maps for moving objects. *IEEE Computer Graphics and Applications*, Jan 2012.
13. D. Sil and D. Gonçalves. Understand my steps: Using edge-bundling to visualize GPS tracks. Master's thesis, Instituto Superior Técnico, Jun 2018.
14. M. Steptoe, R. Krüger, R. Garcia, X. Liang, and R. Maciejewski. A visual analytics framework for exploring theme park dynamics. *Transactions on Interactive Intelligent Systems*, February 2018.
15. Z. Wang, M. Lu, X. Yuan, J. Zhang, and H. v. d. Wetering. Visual traffic jam analysis based on trajectory data. *IEEE Transactions on Visualization and Computer Graphics*, Dec 2013.
16. C. Yang, Y. Zhang, B. Tang, and M. Zhu. Vaite: A visualization-assisted interactive big urban trajectory data exploration system. In *2019 IEEE 35th International Conference on Data Engineering*, April 2019.
17. Ye Zhao, Shamal Al-Dohuki, Thomas Eynon, Farah Kamw, David Sheets, Chao Ma, Yueqi Hu, Xinyue Ye, and Jing Yang. TrajAnalytics: A web-based visual analytics software of urban trajectory data. In *2016 IEEE Visualization Conference*, Oct 2016.