



JamVis: exploration and visualization of traffic jams

Elio Rodriguez^{1,a}, Nivan Ferreira², and Jorge Poco^{1,3}

¹ Department of Computer Science, Universidad Catolica San Pablo, Arequipa, Peru

² Centro de Informática, Universidade Federal de Pernambuco, Pernambuco, Brazil

³ School of Applied Mathematics, Getulio Vargas Foundation, Rio de Janeiro, Brazil

Received 18 September 2021 / Accepted 15 December 2021

© The Author(s) 2022

Abstract Traffic jams are a significant problem in urban cities that cause pollution and waste fuel, money, and time. Therefore, there is an urgent need to build tools that enable authorities to monitor and understand traffic dynamics and their causes. However, exploring these large complex data presents a challenge to domain experts. This paper proposes JamVis, a web-based visual analytics framework that leverages Waze's multi-modal spatio-temporal data to this end. JamVis comprises two main components designed based on requirements elicited from domain experts. The first one supports the exploration of Waze's traffic jam information through multiple linked views. The second component allows identifying events through alerts reported by Waze users about different problems (e.g., potholes, floods, or heavy traffic). A new algorithm called TST-clustering is introduced to perform event detection, which is an adaptation of the DB-Scan algorithm that allows clustering alerts by space, time, and type. Furthermore, to provide an overview of this algorithm's spatio-temporal results, we introduce a novel visualization called ST-Heatmap. JamVis is validated through three usage scenarios analyzing different events in Rio de Janeiro.

1 Introduction

Traffic jams are a severe problem in urban cities that cause pollution and waste fuel, money, and time and, therefore, severely impact the population's quality of living. This problem is significant in major urban centers where traffic jams cause billions of dollars in losses every year. For this reason, it is essential for traffic planning and urban mobility experts to monitor and understand the leading causes of traffic congestions to plan policies and, therefore, identify solutions [1–4]. Due to the importance of this problem, many cities^{1,2} acquire and publish traffic data from different sources such as road sensors. While this approach is widely used, these data often cover a small fraction of the roads and are not uniform among cities. Also, this type of data lacks semantics (i.e., we do not know what type of events are associated with the recorded data). Therefore, understanding the cause of many traffic patterns is difficult or impossible.

Popular GPS navigation applications, such as Waze have revolutionized the way humans relate to traffic and mobility, engaging hundreds of millions of users worldwide. Like other popular vehicle-based traffic data such as taxi data, these applications transform their

users' vehicles into moving city sensors. However, what sets Waze's data apart from ordinary taxi data is that they provide real-time route information and allow social interaction via crowdsourced information and alert functionalities [5]. As a result, such applications produce an enormous amount of multi-modal data (e.g., spatio-temporal and textual) covering many cities in high resolutions, which has generated much interest from academic and governmental researchers [6, 7]. All these data can allow traffic managers to understand traffic jams' behavior and know the cause that produces them. For example, jams contain multiple geometries (e.g., points, poly-lines) that do not fit into a city's street network; also, alerts have a position with a comment that needs to be structured.

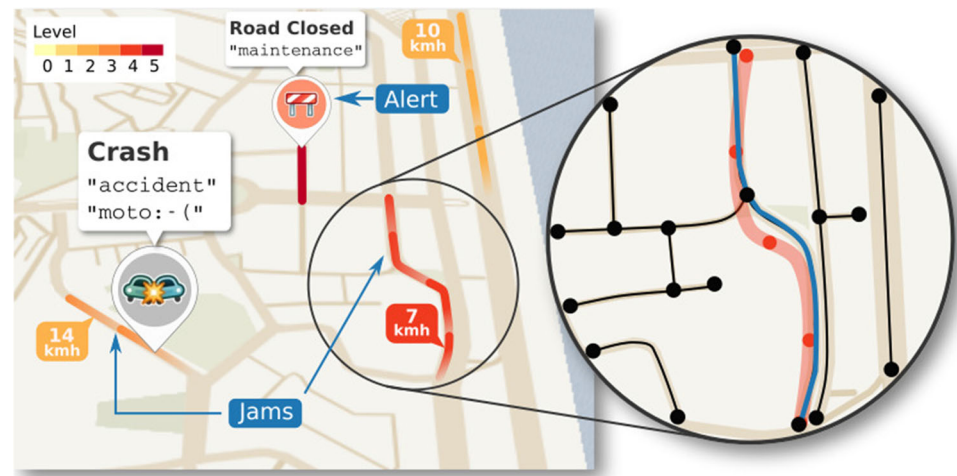
Working alongside domain experts, we developed JamVis, a visual analytics system targeted to explore and analyze congestions using Waze data. This system enables multi-modal traffic jam data exploration at different levels (segments, roads, and regions) interactively through multiple linked views (MLV) [8]. JamVis comprises two main modes; each comprises multiple linked views and targets Waze's specific type of data. The first one supports the exploration of Waze's traffic jam information and has the primary goal of understanding the spatiotemporal nature of jam patterns. The second one explores the Waze alerts, which consist of information volunteered by Waze's users. Alerts allow knowing the causes of traffic jams through both their reported type (e.g., potholes, floods, or heavy traffic) and free textual information provided by the users.

^a e-mail: elio.rodriguez@ucsp.edu.pe (corresponding author)

¹ NYC Open Data—<http://nycopendata.socrata.com>.

² City of Chicago Data Portal—<https://data.cityofchicago.org>.

Fig. 1 Jams and Alerts reported by Waze users. Here, there are two types of alerts (crash and road closed) displaying users' comments. Also, colored road segments indicate the traffic level. Figure magnification illustrates how we map the traffic information to the road network (black lines) to enable temporal traffic analysis



In addition, this work proposes TST-clustering (i.e., type-spatial-temporal clustering), an adaptation of the ST-DBScan algorithm (spatial-temporal DBScan) [9] that clusters alerts by space, time, and type to organize and effectively explore alert patterns. Results produced by this algorithm are represented in a novel visualization called ST-Heatmap that is introduced in this work. This technique leverages a linear sorting of the alerts' geographical coordinates using a space-filling curve to construct representations of the alerts like a Gantt chart. Moreover, this manuscript describes three usage scenarios to demonstrate the usefulness of the system. These usage scenarios show that our approach enables novel traffic congestion data exploration and their causes to uncover problems. Our domain expert collaborators validated these usage scenarios and also reported positive feedback on our system.

In summary, our work contributions are (i) an algorithm to discover hidden events related to traffic jams, based on comments reported by Waze users. (ii) JamVis, a visual analytical system that allows the exploration and analysis of multi-modal traffic jam data alongside a novel visualization called ST-Heatmap. (iii) Three usage scenarios showing an analysis of congestion in Rio de Janeiro.

2 Related work

Our work draws on prior research in the areas of mobility and traffic data analysis and visualization. We point the reader to the surveys by Chen et al. [10] and Mehdizadeh et al. [11] for a broader reading in these areas.

2.1 Visual analysis of mobility data

Proposing techniques and systems to support mobility data visual analysis has been a trending research topic recently. Such works' general goals can be classified as visual monitoring, pattern discovery, route planning, and situation-aware exploration; this last category is

the one where our work lies. In this class, many works focus on developing tools for the analysis of mobility and traffic data.

For instance, Ferreira et al. [12], Kruger et al. [13], and Wang et al. [14] propose visual querying capabilities that enable analysts to specify complex queries easily and filter the data interactively. Buchmüller et al. [15] propose a visualization technique called Motion-Rugs that consists of a temporal heatmap that encodes multiple objects' spatial distribution over time. Traffic flow is another important aspect of mobility; that is why Sun et al. [16] propose a technique to integrate spatio-temporal information in a selected road to visualize flow patterns. Also, Guo et al. [17] show a visual analytics system for finding traffic flow patterns at a road intersection. While the previous systems serve as inspiration for our work's development, they concentrate on general mobility and not analyze traffic jams and their causes, which is our goal.

Many works have used pattern extraction techniques to extract traffic data patterns and better support interactive visual analysis. For instance, the studies of Huang et al. [18] support pattern discovery via graph modeling approaches. Furthermore, Chu et al. [19], Liu et al. [20] use different methods to encode trajectories as documents and apply the latent Dirichlet allocation (LDA) method to extract hidden topics. Another common strategy is the use of clustering algorithms to extract mobility patterns. For example, some works use some variant of the DBScan algorithm, a widely used algorithm for clustering data with spatial attributes based on density. In this category, we have the work proposed by Wang et al. [21], which proposes a C-DBSCAN algorithm, that uses a spherical distance and calculates a cluster center by averaging all the data points in a cluster. In the same line, Bargar et al. [22] use a variation of DBScan called ST-DBScan [9]. In addition to spatial information, this technique uses temporal information to create the clusters; they apply this approach to group trips and detect patterns in bike-sharing data. In our work, we use the clustering strategy by proposing an adaptation of the ST-DBScan algorithm to detect hidden events in the alerts posted

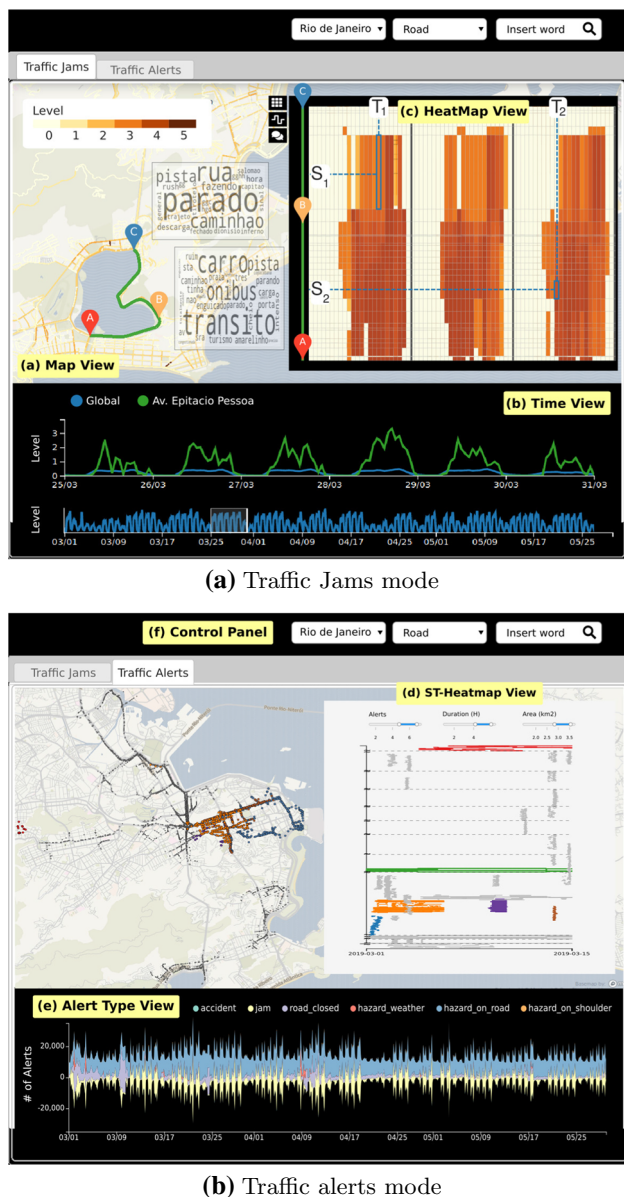


Fig. 2 An overview of JamVis. Our system has two modes. The traffic jams mode (a) allows us to explore and analyze the traffic jams registered by Waze. In contrast, traffic alerts mode (b) will enable us to explore and identify events detected through alerts reported by Waze users. JamVis is composed of a control panel and five linked views: map view, time view, HeatMap view, ST-Heatmap view, and alert type view

by Waze users. Our algorithm detects spatio-temporal evolving patterns and therefore allows us to identify causes in traffic jam data.

2.2 Traffic jams detection analysis

A common approach to the analysis of traffic jams is to derive traffic jam occurrence from mobility data. Andrienko et al. [23] detect traffic jams using a thresh-

old in vehicle speed and clustering it by space and time. Wang et al. [24] use a similar approach to detect traffic congestions, but they go deeper into the analysis by considering traffic jam propagation phenomena. Lee et al. [25] detect congestions and propagations and use this information to predict and, therefore, monitor congestions. They use a long short-term memory (LSTM) neural network to predict bottlenecks and propose a visualization system to explore vehicle flow and jams through the road network. These works have orthogonal goals to ours because we do not focus on traffic jam detection or propagation. As described later, the data used in our scenario already contain traffic jams detected by Waze. Our work aims to perform an analysis of the traffic jams at multi-level (i.e., road, segment, and region) and explore the causes of traffic congestions via social textual data provided by Waze alerts.

Knowing the cause of traffic jams is very important for domain experts because it provides an efficient way to monitor traffic, infrastructure, and events that affect traffic conditions and city life. In recent work, Pi et al. [1] propose a method to find the cause of traffic jams based on deep learning. More clearly, they use traffic flow data from taxis to classify patterns in four classes: accidents, traffic lights, large traffic jams, and free flow. The main drawback of this approach is that the reasons for traffic patterns are limited to only certain types of problems. To deal with this constraint, many works perform a semantic analysis by considering textual data, e.g., geo-located social media data (e.g., Twitter and Facebook), to infer the causes of traffic jams. For example, Kosala and Adi [2] present a system that shows tweets related to traffic according to the user's region selection. Sakaki et al. [4] present a system that shows essential information related to traffic events from social networks.

Our work uses a more flexible approach because we have multi-modal data. Waze provides us with a more significant number of types and subtypes of problems (e.g., potholes, weather), as well as user's comments, which supply us with more details about the causes of traffic jams. As described later, our system allows easy and interactive exploration of these comments via powerful visual summaries created using both the spatio-temporal and textual data.

3 Data and preprocessing

This section describes the Waze dataset used in this work and the pre-processing steps. In the rest of the paper, Rio de Janeiro (one of the largest and more crowded cities in Brazil) is used as a running example since this is one of the main towns our domain collaborators are interested in. However, it is important to mention that our system can be generalized to explore Waze's data from different cities.

Fig. 3 Example of the construction or design process of the ST-Heatmap View. Each subfigure corresponds to a step of this process sequentially. **a** Event detection; we have detected five different events (groups) in this case. **b** We index group' centroids to corners of the Hilbert curve. **c** Sort groups in heatmap according to the Hilbert position. **d** Scale the vertical axis of the heatmap to the size (number of points) of the Hilbert curve. **e** Eliminate vertical gaps. **f** Scale the height of each rectangle to be proportional to the convex hull area of each event

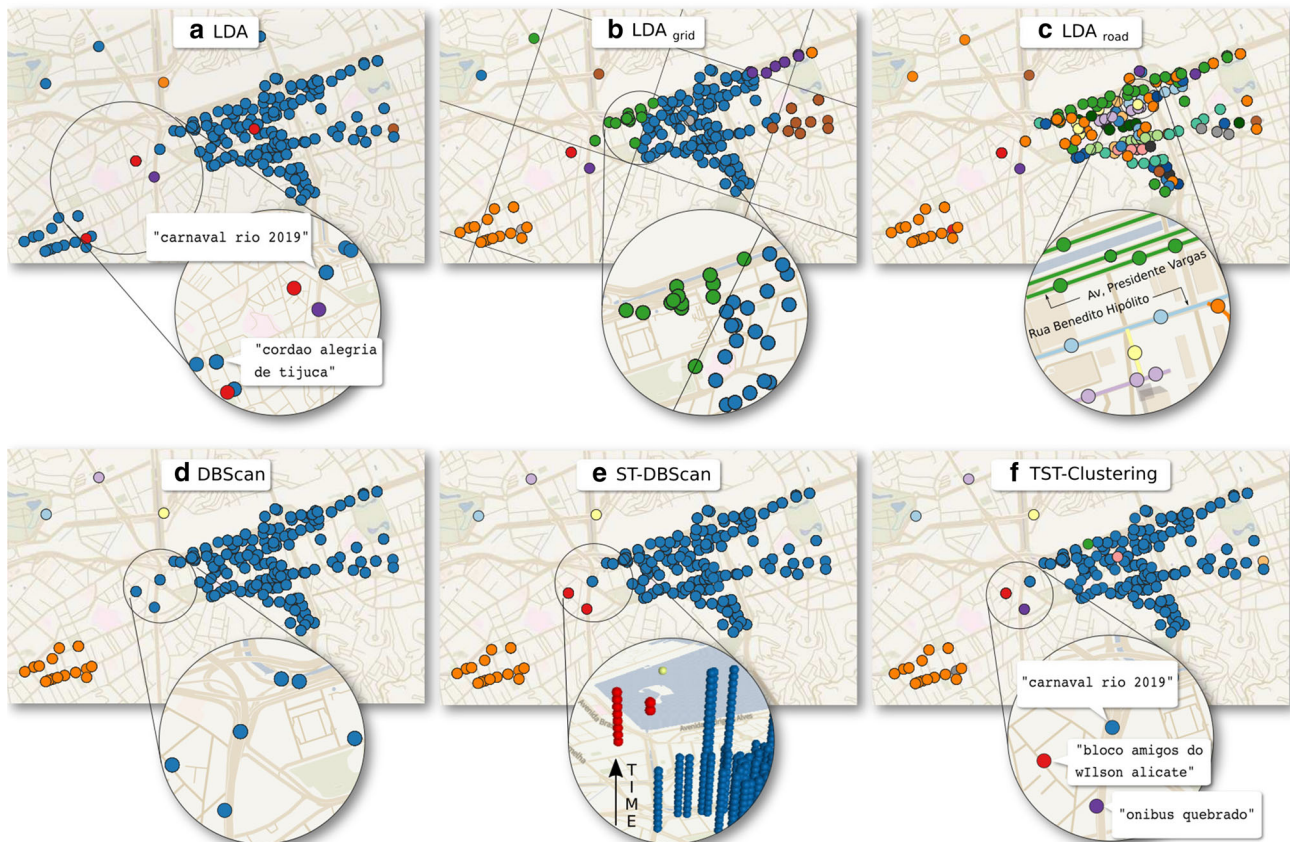
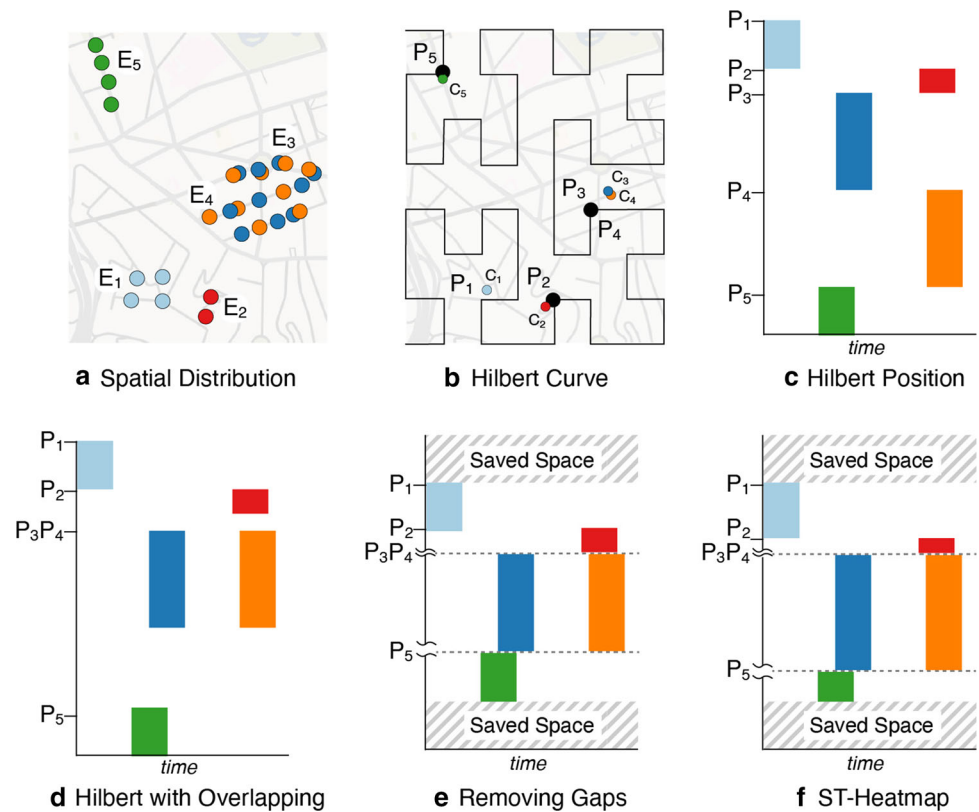


Fig. 4 Comparative analysis of different clustering algorithms (LDA, LDA_{grid}, LDA_{roads}, DBScan, ST-DBScan, and our proposed TST-clustering) used on the alerts in Rio de Janeiro on March 2, 2019

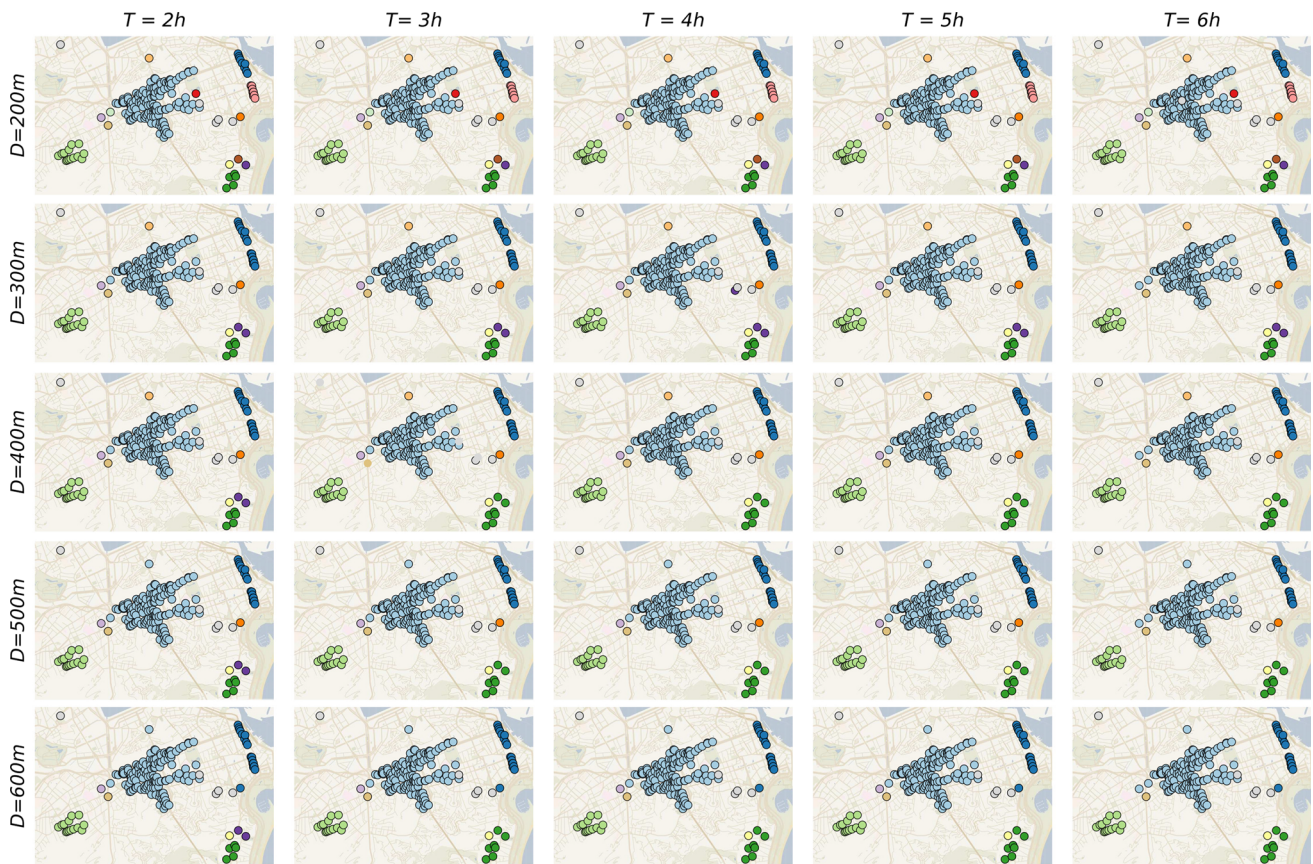


Fig. 5 Sensitivity analysis for the parameters of time and distance of TST-clustering. Each column corresponds to a different time value: 2, 3, 4, 5, and 6 h. Each row corresponds to a different distance value: 200, 300, 400, 500, and 600 m

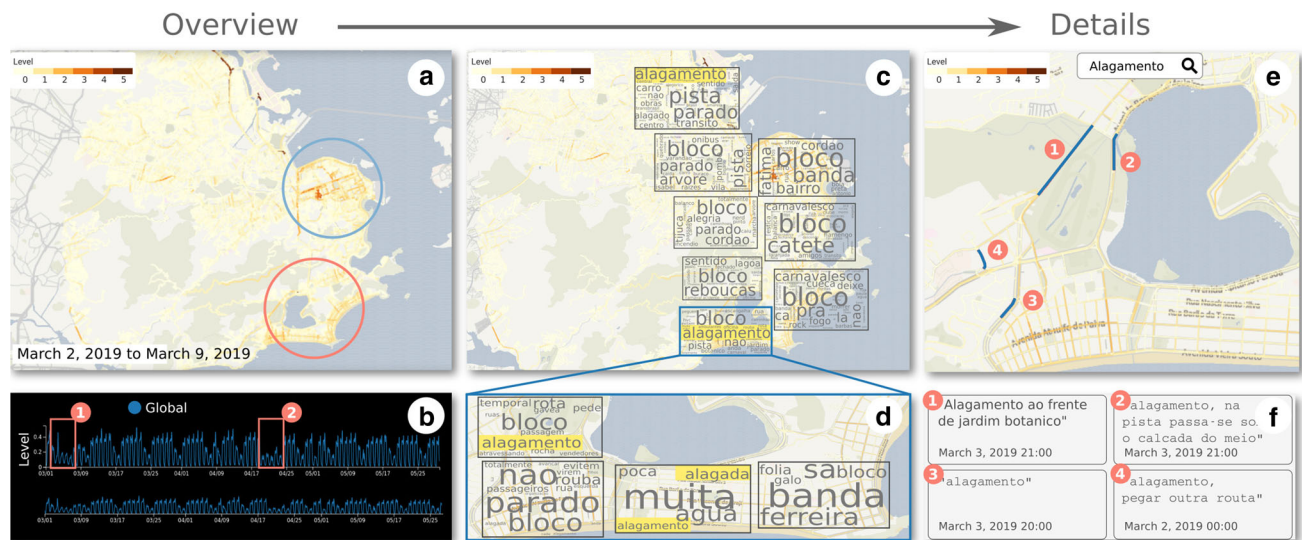


Fig. 6 Analysis of the traffic jams during the week of the Carnival of Rio de Janeiro in 2019. **a** Map View with the level of traffic jams during Carnival Week. **b** We can see two anomalies in Time View. **c** Tag clouds with the most frequent words in the users' comments in different regions.

d Detail of the selected area; note the word "alagamento" (flood). **e** Blue segments correspond to segments where users reported flooding. **f** We can see the date and the comment of one of the segments reporting flooding

3.1 Waze traffic data

Waze data are composed of two parts: traffic jams and alerts. We describe these data attributes in the following.

3.1.1 Traffic jams

Each traffic jam is represented by a poly-line geometry (i.e., sets of connected points) containing a timestamp, a delay time, a speed, and a traffic level (jams' severity ranging from 0 for free flow to 5 for road-closed.). In Fig. 1, we can see multiple jams reported by Waze. A color scale encodes the jam's level where the light red line represents a jam with level 4 (20% of free-flow speed), while the light orange line represents a jam with level 1 (80% of free-flow speed). It is worth mentioning that Waze defines a traffic congestion level as a jam level (0 = free flow, 1 = Low, 2 = Medium, 3 = High, 4 = Very High, 5 = closed), so we decided to keep it.

3.1.2 Alerts

Alerts are notifications related to traffic reported by Waze users. Each alert contains its geographical coordinates, a timestamp, a short textual comment posted by the users, and two categorical entries called type and subtype. The types are jam, accident, road closed, hazard on the road, and weather hazard. Each type has multiple subtypes; for instance, an accident has two subtypes: major and minor accidents. We provide a complete list of types and subtypes in Appendix A Table 1. The comment field is often a short text, and in most cases, it is empty. For example, in Fig. 1, we can see two alerts: the first one is a crash, which has comments "moto:-(" and "accident"; the second one is a road closed with a comment "maintenance".

Note that Waze data can be abstracted as a set of points (alerts) and a set of polylines (jams). Thus, our system could support analysis with datasets formatted in this manner. Concerning our example of Rio de Janeiro, Waze provided us access to a real-time stream containing traffic and alert data, which we collected from March 1 to May 30, 2019 (three months). The stream's temporal resolution is one minute, so Rio de Janeiro generates around 136 million jams and 40 million alerts in total for the three months. Jams cover around 60% of the roads in Rio de Janeiro. Furthermore, considering alerts, only 12% of the total contain non-empty textual comments (95% of the comments have between 5 and 40 characters).

3.2 Pre-processing

In JamVis, we use the city's road network as the base of our visualization. In general, each node of this network represents a city intersection characterized by its geographical coordinates. Besides, each network edge represents a road segment (from now on *segment*, for short) that connects two intersections. We obtain the road

network data from OpenStreetMap (OSM) using the OSMnx library [26]. For our analysis in Rio de Janeiro, the road network contains 15,447 nodes and 34,814 segments. Both the Waze data and the road network are the input of the following pre-processing steps:

1. We perform a map-matching process. Map-matching is the process of associating a sequence of GPS positions observed to its nearest road segment in a road network. For example, in Fig. 1 we can see that the red jam corresponds to the blue segment. To this end, we used a similar process to the ST-matching (Spatial-Temporal matching) algorithm described by Lou et al. [27].
2. Once the map-matching is complete, we group jams by segment and hour and, for each group, we compute and store the mean jam level and the number of jam records. For segments where there are no traffic jams, we assumed that there was no congestion, and we completed them with null values. Furthermore, in the case of the alerts, we group them by segment, hour, type, subtype, and comment. For each group, we store the number of alert records.
3. To accelerate queries for average values for the different attributes of traffic jams, we pre-compute and store accumulated sums over time. This is a common strategy used for speeding-up spatiotemporal aggregations [28]. In more detail, for each time step and segment, we store the sum of the values of each attribute from timestamp 0 to timestamp t . To calculate the mean value of an attribute (e.g., level) of a segment s in a continuous-time interval $[t_i, t_j)$ we use the following equation: $\overline{\text{level}}_{(s, [t_i, t_j))} = (\sum_{t=0}^{t_j} \text{level}(s, t) - \sum_{t=0}^{t_i} \text{level}(s, t)) / (t_j - t_i)$, where the sums in this formula are the pre-computed terms.

4 Methodology

Based on the needs of the domain experts, we defined a set of solutions: (i) a clustering algorithm to help us describe events in the traffic data and (ii) an interactive visual tool to help analysts explore the data. Throughout this section, we will look at each of these components in detail.

4.1 Identifying system requirements

During the last 2 years, we have been part of a larger research project on urban mobility. As part of this collaboration, we had a series of meetings with two traffic engineers interested in using Waze's data to monitor and explore traffic patterns in different cities. One of them is the general operations coordinator at the Municipality of Rio de Janeiro, and the other is a traffic engineer at the Traffic Engineering Company of Rio de Janeiro. Based on these meetings and the needs of

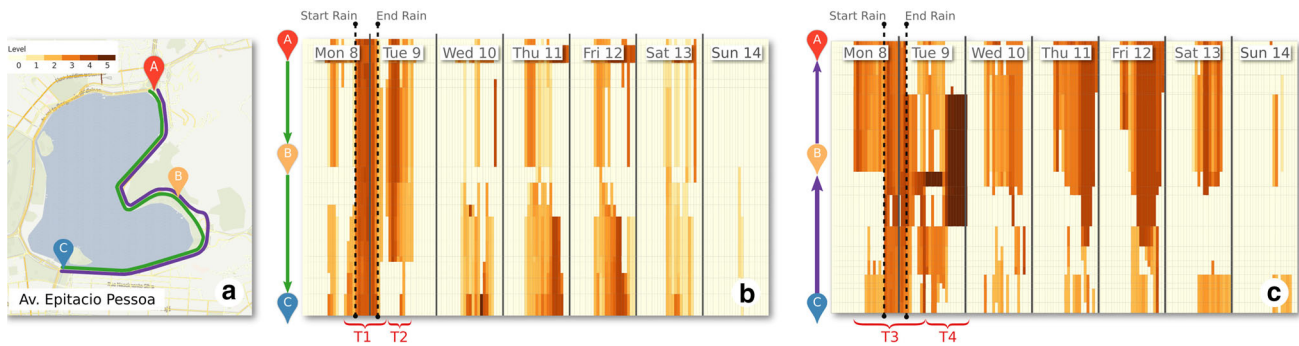


Fig. 7 Analysis of Av. Eptácio Pessoa from April 8 to April 15. **a** Selection of the two directions of Av. Eptácio Pessoa in Map View, from A to C and from C to A. **b** HeatMap View of Av. Eptácio Pessoa from A to C. **c** HeatMap View of Av. Eptácio Pessoa from C to A

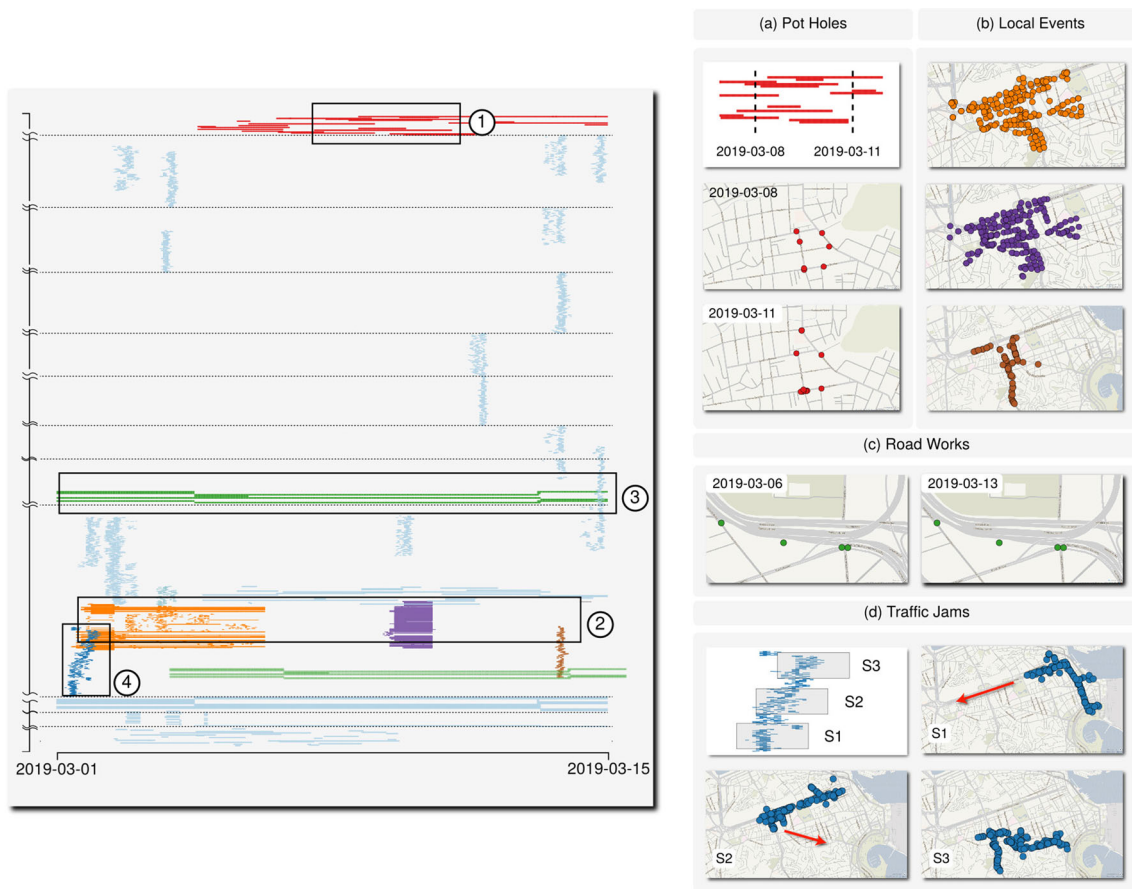


Fig. 8 Example of the automatic event pattern identification and exploration of patterns identified between March 1 and 15, 2019. **a** Pot-holes events: maintain similar spatial distribution over time. **b** Local events: three different events in the same place but at different times. **c** Road works events: maintain the same spatial distribution throughout the whole period. **d** Traffic jams events: spatial displacement over time. The ST-Heatmap (left) depicts the spatio-

temporal distribution of the different patterns identified. Notice that the highlighted regions represent far-away geographical locations (y -axis). The purple, orange, and blue events have spatial intersections but happen in different periods. Finally, the blue event (4) presents left to right displacement on the temporal axis, suggesting the geographical movement (d)

the domain experts, we defined a set of requirements that will be described below.

R1—global analysis: Domain experts want to know what areas in the city have more congestions in a specific time interval. In this way, this tool should inform the user where they should focus their analysis.

R2—multi-level: Domain experts want to visualize and analyze traffic jams at different levels: region, road (e.g., Av. Main Street), or segment. In the case of a road, they want to visualize the information of each segment.

R3—jams analysis: They want to know traffic jams' behavior (i.e., propagation, causes). The experts emphasized the need to know why congestion occurs.

R4—events exploration: Experts want to explore the different traffic-related problems (i.e., accidents, rains, works) in the city. For example, the location of the event and how long it lasted.

R5—interactive exploration of big data: Experts often use different analytical systems that do not support large amounts of data. Therefore, they require a tool to refine their queries by showing a global vision of the city and interactivity.

4.2 TST clustering algorithm for events discovery

One of the main challenges in visual analytics of spatio-temporal data is to support pattern discovery. It is especially true in our case since we are dealing with complex multi-modal data. Previous works have shown that putting together interactive visualization capabilities with automated methods is an excellent strategy to support efficient pattern discovery [29,30]. In this work, we use this strategy to guide users towards patterns that explain traffic jams and point to important events in the city. More clearly, we devise a method that identifies semantically meaningful spatio-temporal traffic events. We propose a modification of the ST-DBScan algorithm [9] that considers Waze's alert type as part of the clustering process, as described in the rest of this section.

We start our discussion by presenting the original density-based spatial clustering of applications with noise (DBScan) [31] algorithm (we provide pseudo-code snippets for this algorithm and its variants in Appendix B). It allows automatic detection of arbitrary-shaped clusters in any dataset with spatial attributes using a distance metric chosen by the user (usually Euclidean distance). DBScan has two input parameters: a searching radius (ε_s) and the minimum number of points required to form a cluster (MinPts). The algorithm starts with a first point p in a dataset D and retrieves all neighbors of point p using the *RetrieveNeighbors* procedure. This corresponds to all points within a distance of at most ε_s from p . If p has not a sufficient number of neighborhoods (i.e., larger than MinPts), then p is classified as a noise point and is not further considered. On the other hand, if the neighborhood of p is large enough, then the clustering process starts, using the *ExpandCluster* procedure. This procedure consists

of iteratively collecting the neighbors within ε_s distance from each neighborhood point. We repeat this process until all of the points have been processed.

One of the disadvantages of the DBScan algorithm is that it only takes into account spatial attributes. However, when studying spatio-temporal phenomena, one is also interested in understanding the patterns' temporal aspect. To solve this, Birant and Kut [9] propose a modification of the DBScan algorithm called spatial-temporal DBScan (ST-DBScan). This algorithm takes three parameters: ε_s , MinPts (which are analogous to the ones in the original DBScan), and ε_t , which represents the threshold for the temporal distance. The ST-DBScan procedure is essentially the same as the original DBScan, with the only change being a new version of the *RetrieveNeighbors* function. In more detail, this modified version first retrieves the set X of all points within a temporal interval of amplitude ε_t centered at the timestamp of p and, among the points in X , returns all points within (spatial) distance ε_s from p .

One important fact to notice is that different events that affect traffic can happen nearby in space and time. The ST-DBScan algorithm is oblivious to this fact and might cluster together these events. To solve this issue, we modify the *RetrieveNeighbors* function once again to include the semantics given by the type field in Waze alerts. This final version of the function retrieves all neighbors of a point p within ε_s and ε_t distance and with the same alert type of point p . We call this algorithm TST-clustering, because it group object based on type, space and time attributes.

4.3 JamVis

We now discuss the design and implementation of the JamVis system. The main interface (shown in Fig. 2) comprises a control panel and five linked views: map view, time view, HeatMap view, ST-heatmap view, and alert type view. The control panel allows to select the city to be analyzed, toggle selection mode, and use the text query capability. In the following, we describe the visual components.

4.3.1 Map view

This view consists of an interactive map that depicts the geographical context (allowing zooming and panning) and traffic/alert information for the entire city (T1). In Fig. 2(left), we can see the average traffic level color-coded on the road network (from light yellow—free flow to dark red—blocked). Traffic jams can be analyzed using three types of selection (chosen in the control panel): segment, road, and region selection. To select a *segment*, we need to click on the desired road segment. To select a *road*, the user can search the road name in the Control Panel or click a road segment; the system automatically detects all the segments associated with that road (Fig. 2a shows the Av. Epitacio Pessoa selected). To select a *region*, we can draw a rectangle on top of the map. This selection allows us to inspect

the content of the comments registered by Waze users by rendering a tag cloud that shows the most frequent words present in these comments (see the tag clouds on the map in Fig. 2a). This was inspired by the Tag Cloud Lenses technique proposed by Ferreira et al. [32]. In Fig. 2(right), we can see the location of traffic alerts, where the colors represent the different spatio-temporal events (groups) identified with the algorithm described in Sect. 4.2. Here we can also use the region selection to analyze the user comments.

4.3.2 Time view

This view allows us to visualize the traffic level, aggregated by hour, throughout the entire period (T1). As we can see in Fig. 2a, this view follows the concept of focus-plus-context-visualizations.³ The bottom plot shows the average traffic for the entire city (blue line) to help users keep a temporal overview. The selection period on this plot creates on the top a detailed (zoomed) representation of the corresponding time interval on the top. Furthermore, the selected interval is used to filter the overall period considered, and the other views changes accordingly (T2). The plot at the top also shows the average traffic level of a segment, road, or region selected on Map View (using multiple lines with different colors). In the case of roads or regions (multiple segments), the plot shows the aggregated information. For instance, In Fig. 2a, we visualize the average traffic level for Rio de Janeiro (blue line) at the bottom line chart. The top line chart shows the traffic level of Av. Epitacio Pessoa road (green line) and the traffic level for Rio de Janeiro (blue line) for the visible spatial area and the whole period.

4.3.3 HeatMap view

The HeatMap view aims to visualize and compare the congestion level in a road (set of segments) in a given period. For that, we use a matrix format where each row corresponds to a segment (with height proportional to the segment length). Each column represents one hour of the selected period (the column width is set to one pixel to display possibly long periods compactly). The value of each cell's traffic level is encoded using the same color scale used in Map View. Figure 2 shows an example of this view analyzing the Av. Epitacio Pessoa in three days (a gray line limits each day). To make it easier to see the correspondence between the segments in this view and the Map View, we place indicators at the beginning, middle, and end segments (see red, orange, and blue pins with letters A, B, and C, respectively) In addition, when the user selects a cell in this view (e.g., segment S_1 at time T_1 in Fig. 2), a tooltip appears with detailed information about the chosen segment.

4.3.4 Alert type view

This view is displayed when using the traffic alert mode of JamVis. It consists of a Theme River plot that shows the temporal distribution of each type of alert as we can see in Fig. 2e. Similar to the time view, this view also allows us to filter a continuous period, and the information in the other views changes according to the selected period (T2).

4.3.5 Spatio-temporal HeatMap view

This view aims to provide an overview of the result (groups) obtained using our proposed algorithm TST-clustering. It consists of a heatmap that shows the temporal distribution of alert clusters and tries to depict spatial information using a linear ordering obtained via a space-filling curve. We inspire our design on the MotionRugs technique proposed by Buchmüller et al. [15] in the sense that we use a space-filling curve to sort the geographical space linearly. Buchmüller et al. use the Hilbert Curve as a space-filling curve since this technique allows to represent 2D spaces in 1D trying to maintain neighborhoods. That is, if two points are close in 2D, likely, they are also close in the 1D ordering.

However, different from MotionRugs, our application requires representing the position of a time-varying number of alerts. To solve this, we used a heatmap-like metaphor described in the following.

In Fig. 3a, we can see the spatial distribution of five different events (groups) (E_1, E_2, E_3, E_4, E_5) obtained using TST-clustering. We then use a Hilbert curve to sort the locations corresponding to the centroid of each event. In more detail, Fig. 3b shows that the group' centroids (C_1, C_2, C_3, C_4, C_5) are indexed to points (P_1, P_2, P_3, P_4, P_5) of the Hilbert Curve (P_3 and P_4 are the same points). The obtained order is used to vertically sort the rectangles representing each group on the heatmap (the height of each rectangle represents the number of alerts in the group), as shown in Fig. 3c. To represent in the heatmap the fact that two or more events occur in the same location (e.g., E_3 , and E_4), we first scale the vertical axis of the heatmap to the size (number of points) of the Hilbert curve. Using this approach, we can uniquely identify points in the Hilbert curve in the y -axis (in Fig. 3d). As a result, events that occur in nearby spaces (E_1 and E_2) are in a close (vertical) position (P_3 and P_4); and events in the same space (E_3 and E_4) are in the same (vertical) position (P_3 and P_4) in the heatmap. One harmful side product of this scaling is that it generates blank spaces that correspond to extents in the Hilbert curve, where no close event occurs. As shown in Fig. 3e, we eliminate these gaps by vertically translating the events as long as they don't overlap. We can quickly implement it by eliminating the indices corresponding to intervals of empty points in the Hilbert curve. To indicate a spatial gap before removing the blank spaces, we place a dotted horizontal line and an icon consisting of two wavy lines in the vertical axis. We encode the separation between

³ <https://infovis-wiki.net/wiki/Focus-plus-Context>.

the wavy lines with the size of the blank space removed. Finally, to have a better perception of the area covered by each event, we scale the height of each rectangle to be proportional to the area of the convex hull of each event. Additionally, we apply the Hilbert curve ordering within the alerts of each event. We do this because, in events that occupy large areas, it is good to keep the spatial information of the internal alerts. Figure 3f shows an illustration of the final design. Note that E_5 and E_1 have the same number of alerts but different areas.

Depending on the parameters used in TST-clustering, the number of groups can be large. Initially, we show all the groups, but at the same time, we have sliders that allow us to filter the groups based on the number of alerts, duration, and area of the groups (see the top of Fig. 2d). It is also important to mention that to facilitate the explanation, in Fig. 3, we represent each group as a rectangle (based on the number of alerts and the maximum and minimum time of the alerts). However, in usage scenario 3 and the video, we draw a line for each alert in the group (based on the duration).

5 Evaluation

This section describes two experiments we performed to evaluate the results of TST-clustering. The first one compares our proposal with variations of DBScan and LDA algorithms. The second one shows a sensitivity analysis of the distance and time parameters required by our algorithm.

5.1 Comparison with LDA, DBScan and ST-DBScan

To demonstrate the effectiveness of TST-clustering, we compare it against other methods previously used to find patterns in mobility/traffic data: DBScan, ST-DBScan, and three variants of LDA described in the following. LDA is a method initially proposed to unveil hidden topics in text corpora. Since Waze alerts contain users' comments, it is natural to use LDA to uncover patterns in these comments. The first and more basic variant consists of processing the comments, removing symbols and numbers, dividing them into tokens, and performing the stemming process. After that, this variant uses LDA as a clustering algorithm by grouping alerts with their most prominent topic (topic with the highest weight). The other variants are based on two recent approaches that used LDA to cluster taxi trajectories by representing each trajectory as a document. First, Tang et al. [33] divide the spatial region with a rectangular grid naming each grid cell with a unique id. Hence, they represent each trajectory as a document by collecting the cells' ids where the trajectory goes through. In our experiment, we divide the city of Rio de Janeiro with a regular grid where each cell is a square of side 800 m. The second method, proposed

by Liu et al. [34], represents the trajectory as a document by including the roads' names that the trajectory goes through. For further reference, we name these approaches LDA_{grid} and LDA_{road} , respectively.

In this comparison, we use alerts registered in Rio de Janeiro on March 2, 2019. In the case of the DBScan, ST-DBScan, and our proposed method, we use the parameters $MinPts = 1$ and $\varepsilon_s = 400$ m, and $\varepsilon_t = 4$ h. In the case of the LDA method, we use different numbers of topics (5, 10, 15, 20), but we select 20 because it shows better results. In Fig. 4, we can see the results by clustering the alerts with these methods; we encode each group using the color. In the highlighted area of (a), we see several blue points, which implies that they belong to the same topic (found by LDA). However, looking at the comments, we notice that the blue points on the right side refer to Rio de Janeiro's main event, "carnaval rio 2019", and the dots on the left side refer to a local parade away from the central area of the festivities "cordao alegria tijuca". It occurs because the LDA only considers the comments and found a similarity between them, but it does not consider the spatial or temporal attributes. In (b), we see the results for LDA_{grid} . Notice that the big group is divided into two smaller groups (green and blue) because they belong to two regions. However, these two clusters refer to the same event as we discuss later. We can explain this result, noting that most of the comments are very short (95% of them have between 5 and 40 characters). Hence, the words corresponding to the grid cell ID have higher importance, in general, than the original words from the comment. Similar behavior is seen in the results of LDA_{road} (c). We observe that this method divides alerts into many clusters because, in several cases, the roads' name is longer than the comment. Therefore, the street names have a high weight in the model, and the clusters are created based on this information. For example, the highlighted area in (c) shows that each street makes a different group. We also replaced the street names with an IDs and obtained similar results.

In (d), we can see the results of the DBScan algorithm. In the highlighted area, all points have the same color, which indicates that they belong to the same group. It happens because the algorithm only used spatial information to create groups. However, these alerts should be on different groups since they are in other times (as you can see in (e)) and have different types (as you can see in (f)). We present the results for ST-DBScan in (e). The highlighted area shows a 3D plot where each sphere represents an alert, and the height is proportional to the time. Here we see that the red and blue points are separated in time. Also, we can see that some red dots should not belong to the red group. If we look at their comments or types, we notice that they are related to different events, such as bus crashes or carnival festivities. Finally, in (f), we can see the results using the TST-clustering algorithm. Notice that our algorithm creates three clusters. The red cluster refers to the event "bloco amigos de wilson alicate" (type: road closed), the blue cluster refers to the event "carnaval rio 2019" (type: road closed), and the purple

cluster refers to an accident "onibus quebrado" (type: accident). As shown in this example, using this simple adaptation of the ST-DBScan algorithm, we obtained semantically enriched clusters that were not possible with other methods. Our domain experts validated our results by saying that each group was well defined and explained.

5.2 Parameter sensitivity

For this experiment, it is used the same alerts as the previous comparison. TST-Clustering is an extension of DBScan; so, it inherits the parameters from DBScan, i.e., MinPts, ε_s , and ε_t . In this experiment, we are only going to vary the values of the ε_s and ε_t parameters because they are the ones that could have the most significant impact. In addition, MinPts was set to 1, since in our case, there are isolated alerts that would be classified as outliers if we increase this parameter. As we saw in the comparison of the previous section, the parameters used were $\varepsilon_s = 400$ and $\varepsilon_t = 4$, so we decided to apply our algorithm by varying the parameters with two higher and two lower values. We used 2, 3, 4, 5, and 6 h, and for the distance, we used 200, 300, 400, 500, and 600 ms.

In Fig. 5, we can see the results for these combinations. In general, we can see that the groups are maintained in all cases. However, we can appreciate some differences. For example, for all cases where the distance is 200 ms ($D = 200$, first row), the dark blue group is small, but when varying the distance to 300 ms ($D = 300$), it grows, and the pink group disappears, this is because both groups are of the same type and occur at the same time. We can conclude that the parameters time and distance are not so sensitive for the results of our algorithm. Suppose the user selects an initial value of both. He will observe a general pattern of the distribution of events in the city, which will be maintained by varying these parameters.

6 Usage scenarios

We validated our system's usability through three usage scenarios (shown in the accompanying video), where we analyzed the traffic jams in the city of Rio de Janeiro. In the first one, we explored the causes of traffic jams. The second usage scenario examines the behavior of traffic jams on two different roads. Finally, the third use case describes the use of Spatio-Temporal HeatMap View to explore the automatically identified clusters of Waze alerts.

6.1 Exploring events

The Carnival period is one of the main festivities in Rio de Janeiro and attracts around 2 million people worldwide. The celebration happens every year and lasts one week. The famous street carnival parties, called *bloco*s, are parades organized by local groups that can last

several hours and produce traffic jams in different city regions. We used our system to explore this scenario for the Carnival of 2019.

Looking at Fig. 6b, we can notice two intervals where the value of the traffic level was irregular (1 and 2). The first period was from March 2 to March 9 and corresponded to the Carnival period. The second was from April 18 to April 25 and corresponded to the Holy Week period. These two periods were holidays in Rio, so fewer cars in the city compared to working days. Using the Time View to selected the first period (Carnival period), we can see in Fig. 6a the corresponding geographical distribution of traffic jams. We notice that the highest level of traffic jams (red segments) was in the city center (blue circle) and the Southern (red circle) regions of the city. Therefore, although the average level was lower than usual, there are regions where the traffic was still heavy. To explore the cause of these traffic jams, we use MapView's tag cloud component. As shown in Fig. 6c, most of them contain the word *bloco* as expected. However, two tag clouds prominently contain the word "alagamento" (flood in English), which indicates the presence of registered comments related to flooding in these two regions. To explore this in more detail, we zoom in on the South region and, again, used the tag cloud component to select four different subregions (see Fig. 6d). We can see that only two of them contain the word "alagamento". To get an even more detailed understanding, we decided to search all the segments where it was reported a comment with the word "alagamento" using the Control Panel's search capability. Figure 6e depicts the resulting (blue) segments. Finally, Fig. 6e shows that the original comments registered in these segments are related to floods. In addition, we can see that three of them occurred on March 3 between 8 pm and 9 pm. The fourth comment was recorded on March 2 at midnight. This comment corresponds to the floods that occurred in the city due to heavy rains.⁴

6.2 Traffic jam and road analysis

On April 8 of 2019, the city of Rio de Janeiro experienced heavy rainfall. It began at approximately 6 pm and continued until 3 am on April 9, causing floods, tree falls, landslides, and heavy traffic in different city areas. Due to the magnitude of the damages, the municipality of Rio de Janeiro declared April 9 a holiday. Due to this, many roads showed unusual traffic behavior. One of the most affected road was the avenue Epitacio Pessoa. We used our system to understand the impact of this heavy rain on both roads by focusing on the one week between April 8 and April 15.

First, we focus on Epitacio Pessoa avenue and select both directions of the road (from A to C and from C to A) in our Map View (Fig. 7a). The HeatMap View in

⁴ Globo G1—<https://g1.globo.com/rj/rio-de-janeiro/noticia/2019/03/03/chuva-forte-deixa-rio-em-estagio-de-atencao-no-domingo-de-carnaval.ghtml>.

Fig. 7b depicts the traffic level in the direction from A to C. We can see that, usually, there is heavy traffic during morning and afternoon rush hours. However, as shown in the highlighted periods, T1 and T2 (part of April 8 and April 9), there were unusual traffic patterns due to the rainfall. During T1, which starts at 3 pm on April 8 and ends at 3 am on April 9, we notice that traffic jams began in the last Segment (C in the South) before the rainfall. After that, the jam propagated through the other segments until 6 pm when the rain started, and by then, the avenue was completely congested. The unusual congestion remained until 3 am on April 9, when the second period (T2) starts (and ends at 3 pm on the same day). During T2, we can notice that the last four segments were empty, but the other remained congested. Notice that starting from 3 pm (on April 9) till the end of the day, the road had no traffic jams. Finally, we highlight that traffic was still unusual on April 10 and went back to normal on April 11.

We now focus on the opposite direction (from C to A). As shown in Fig. 7c, we notice that the traffic behavior is different from the first case. We again focus on two-time intervals. In T3, (from 9 am on April 8 to 9 am on the following day), the traffic jams were more intense in the last segments (close to the A point) until 6 pm, when rains started, and all the segments became congested. On the other hand, looking at the period T4 (from 9 am on April 9 to 1 am on April 10), we see that only the middle segments (close to the B point) were continuously closed due to repair work which lasted until 1 am on April 10, according to comments of Waze users. The main reason for the different traffic behaviors in the two directions of the Epitacio Pessoa Ave. is that they follow different lanes which are located in a non-flat terrain. The segments close to point B on the purple lane are located at a lower elevation than the other lane; therefore, it is more affected by the floods.

6.3 Automatic event identification

In a large and complex city such as Rio of Janeiro, a large diversity of events take place daily. Looking at Fig. 2e, we can see during the first 2 weeks of March 2019, we had a significant increase in the number of alerts typed as road closed. To obtain an overview of this situation, we apply the proposed TST-clustering algorithm to the alerts that happened during this period using as parameters (see Sect. 4.2): $\varepsilon_s = 200m$, $\text{MinPts} = 5$, and $\varepsilon_t = 1h$ (we chose this values through experimentation). We use the ST-HeatMap view to visualize the resulting clusters. First, to eliminate noisy clusters, we selected the 100 largest (in terms of the number of alerts) clusters. Using the interactive filtering capabilities of the ST-HeatMap view, we set the sliders to filter the clusters with at least five alerts, that lasted more than four hours and cover a spatial region of at least 2km^2 . The resulting ST-HeatMap is shown at the center of Fig. 8. We finally selected four different patterns on the ST-Heatmap for further analysis (numbered 1, 2, 3, and 4). Clusters not involved in

the discussion are shown in light blue. We select these patterns to highlight a diverse set of features present on the heatmap. As the first example, we mention the red cluster (1) on the ST-Heatmap: a long-duration event covering a small spatial region (small height). This pattern is related to the reports of pot-holes in the roads. Figure 8a shows a zoomed-in view of the cluster on the heatmap (top), as well as the spatial distribution of this cluster corresponding to two different times, where we can see that the alerts have other spatial distribution but stay within the same small area. This is because Waze users do not always report alerts from the same location. Furthermore, these events have a long duration since they are infrastructure problems not immediately solved by the local authorities. The second example highlighted on the ST-heatmap (2) shows occurrences of events in the same space but at different times. These clusters are related to samba parades in the center of the city during Carnival. Figure 8b shows the spatial region covered by the three highlighted clusters, which is one of the most popular places for gatherings during the Carnival period. Notice how using the ST-Heatmap can instantly identify the presence of this spatial intersection. Still, it is also possible to distinguish between longer (light orange) and shorter events (purple and dark orange). The third pattern on our heatmap (3) is similar to the one described in Fig. 8a since it also has a small height but long duration. However, the number of alerts is constant throughout time, and we can see that they are registered in the same positions. This pattern is related to construction work carried out on certain roads which are reported to Waze by the local authorities on specific locations (Fig. 8c) during a prolonged period. Finally, the last pattern (4) is related to traffic jams. On the ST-Heatmap, we can see that this cluster has a considerable height and move over time (Fig. 8d top-left). By selecting different portions of this cluster (S1, S2, and S3 on Fig. 8d), we observe on the map that the alerts move spatially, which indicates propagation of the traffic jam over time.

7 Discussion and limitations

As we saw earlier, the usage scenarios show the usefulness of the system for analyzing traffic congestion. Among the functionalities, we have the detection of events and the identification of the cause of these events. In addition, it allows us to explore and visualize the behavior of congestion at the street level. Nevertheless, there is still space for improvement.

7.1 Real time

One important scenario usage for our system is monitoring the traffic state and detecting traffic events in real-time, as suggested by the domain experts. Our system currently relies on a pre-processing stage to prepare the data. The need for pre-processing prevents our

JamVis from being readily available for real-time analysis.

Initially, this was not an objective of our work, as we focused on a historical analysis of the stored data. However, given suggestions from our domain experts, we are currently working on strategies to remove this limitation. For example, some work has recently been done to speed up and efficiently execute map-matching using Apache Spark [35]. Nevertheless, it is not a simple job as it requires more computational power and high-performance computing specialists to perform a proper implementation.

7.2 Useless comments

Another critical point is that our system needs a pre-processing to remove useless comments (e.g., insults, emoticons, and out-of-context comments). In fact, in our experience, the vast majority of alerts contain text related to traffic jams and a few useless comments. Although this problem did not affect our analysis, we plan to use supervised learning algorithms that will allow us to distinguish this type of comment. This work would be equivalent to a spam classifier; however, it requires having a labeled dataset that we currently do not have.

7.3 Geometric computations

Another critical problem is that TST-Clustering involves expensive geometric computations and needs to be adapted to work in real time. An immediate future work would be to adapt our algorithm to process data streams and capture evolving spatio-temporal patterns of Waze alerts. On the other hand, it would also be possible to accelerate this algorithm using parallelism, but this is orthogonal work to our current proposal.

8 Conclusion

This paper proposes JamVis, a visual analytics framework that leverages Waze's multi-modal traffic data. We have two main contributions in this work: First, the TST-clustering algorithm is an adaptation of the ST-DBScan algorithm to identify events clustering alerts by space, time, and type. Second, the ST-HeatMap View is a novel visualization strategy to display spatial information on a temporal heatmap that allows us to depict both the duration of the events and an approximation of their spatial distribution. Our evaluations and usage scenarios demonstrate the usefulness of our approach, as well as being positively validated by domain experts.

Funding The authors acknowledge the financial support by Getulio Vargas Foundation and Concytec Project World Bank “Improvement and Expansion of Services of the National System of Science, Technology and Technological Innovation” 8682-PE, through its executing unit ProCien-cia for the project “Data Science in Education: Analysis of large-scale data using computational methods to detect and

prevent problems of violence and desertion in educational settings” [Grant 028-2019-FONDECYT-BM-INC.INV].

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

Appendix A: Types and subtypes of alerts

Waze classifies alerts reported by users into five main categories (i.e., jams, accidents, road closed, hazard on road and hazard on weather). Each category has a different number of sub-categories as we can see in Table 1. In addition, we can note that each of these categories describes a general problem related to traffic. We use these categories in our TST-clustering algorithm to better classify the alerts according to space and time.

Table 1 Waze alerts type and subtype

Type	Subtype
Jam	Moderate traffic
	Heavy traffic
	Stand still traffic
	Light traffic
Accident	Major accident
	Minor accident
Road closed	Road closed hazard
	Road closed construction
	Road closed event
Hazard on road	On shoulder
	On road object
	On road pot hole
	On road road kill
	On shoulder car stopped
	On shoulder animals
	On shoulder missing sign
Weather hazard	Weather fog
	Weather hail
	Weather heavy rain
	Weather heavy snow
	Weather flood
	Weather monsoon
	Weather tornado
	Weather heat wave
	Weather hurricane
	Weather freezing rain

Appendix B: TST-clustering algorithm

We can see in Algorithm 1 the original density-based spatial clustering of applications with noise (DBScan) algorithm. Algorithm 2 shows the retrieveNeighbors procedure of the ST-DBScan algorithm, which is an adaptation of the DBScan algorithm; ST-DBScan incorporates the attributes of time and space. Finally, in Algorithm 3 we can see our proposed algorithm. Our algorithm modifies retrieveNeighbors procedure of the ST-DBScan and incorporates the type attribute.

Algorithm 1 DBScan algorithm

```

1: procedure RETRIEVENEIGHBORS( $p, \varepsilon_s$ )
2:   return All points within  $p$ 's  $\varepsilon_s$ -neighborhood.
3: end procedure
4:
5: procedure DBSCAN( $\varepsilon_s, \text{MinPts}$ )
6:    $C = 0$ 
7:   for each unvisited  $p$  in database  $D$  do
8:     mark  $p$  as visited
9:     neighbors = RETRIEVENEIGHBORS( $p, \varepsilon_s$ )
10:    if  $\text{sizeof}(\text{neighbors}) < \text{MinPts}$  then
11:      mark  $p$  as noise
12:    else
13:       $C += 1$ 
14:      EXPANDCLUSTER( $p, \text{neighbors}, C, \varepsilon_s, \text{MinPts}$ )
15:    end if
16:  end for
17: end procedure
18: procedure EXPANDCLUSTER( $\varepsilon_1, \text{MinPts}$ )
19:   add  $p$  to cluster  $c$ 
20:   for each  $p'$  in neighbors do
21:     if  $p'$  is not visited then
22:       mark  $p'$  as visited
23:       neighbors' = RETRIEVENEIGHBORS( $p', \varepsilon$ )
24:       if  $\text{sizeof}(\text{neighbors}') \geq \text{MinPts}$  then
25:         join neighbors and neighbors'
26:       end if
27:       if  $p'$  is not yet member of any cluster then
28:         add  $p'$  to cluster  $c$ 
29:       end if
30:     end if
31:   end for
32: end procedure

```

Algorithm 2 retrieveNeighbors procedure of ST-DBScan

```

1: procedure RETRIEVENEIGHBORS( $p, \varepsilon_s, \varepsilon_t$ )
2:    $X =$  points in  $D$  within  $p$ 's  $\varepsilon_t$ -neighborhood.
3:    $X =$  points in  $X$  within  $p$ 's  $\varepsilon_s$ -neighborhood.
4:   return  $X$ 
5: end procedure

```

Algorithm 3 Proposed retrieveNeighbors procedure

```

1: procedure RETRIEVENEIGHBORS( $p, \varepsilon_s, \varepsilon_t, \varepsilon_c$ )
2:    $X =$  points in  $D$  within  $p$ 's  $\varepsilon_t$ -neighborhood.
3:    $X =$  points in  $X$  within  $p$ 's  $\varepsilon_s$ -neighborhood.
4:    $X =$  points in  $X$  which have same type of point  $p$ .
5:   return  $X$ 
6: end procedure

```

References

1. M. Pi, H. Yeon, H. Son, Y. Jang, Visual cause analytics for traffic congestion. *IEEE Trans. Vis. Comput. Graph.* **27**(3), 2186–2201 (2021)
2. R. Kosala, E. Adi, Harvesting real time traffic information from twitter. *Proc. Eng.* **50**, 1–11 (2012)
3. S.S. Ribeiro Jr, C.A. Davis Jr, D.R.R. Oliveira, W. Meira Jr, T.S. Gonçalves, G.L. Pappa, Traffic observatory: a system to detect and locate traffic events and conditions using twitter, in *Proceedings of the 5th ACM SIGSPATIAL International Workshop on Location-Based Social Networks*, pp. 5–11. ACM (2012)
4. T. Sakaki, Y. Matsuo, T. Yanagihara, N.P. Chandrasiri, K. Nawa, Real-time event extraction for driving information from social sensors, in *2012 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, pp. 221–226. IEEE (2012)
5. T.H. Silva, P.O.S. Vaz De Melo, A.C. Viana, J.M. Almeida, J. Salles, A.A.F. Loureiro, Traffic condition is more than colored lines on a map: characterization of Waze alerts, in *International Conference on Social Informatics*, pp. 309–318. Springer (2013)
6. M. Pack, N. Ivanov, Are you gonna go my Waze? *Inst. Transport. Eng. ITE J.* **87**(2), 28 (2017)
7. A. Pérez-Espinosa, A.L. Reyes-Cabello, J.L. Quiroz-Fabián, E. Bravo-Grajales, Trafico cdmx system: using big data to improve the mobility in Mexico city, in *Proceedings of the 2018 International Conference on Computing and Big Data*, pp. 13–17 (2018)
8. G. Wills, Linked data views, in *Handbook of data visualization*, pp. 217–241. Springer (2008)
9. D. Birant, A. Kut, St-dbscan: an algorithm for clustering spatial-temporal data. *Data Knowl. Eng.* **60**(1), 208–221 (2007)
10. W. Chen, F. Guo, F.-Y. Wang, A survey of traffic data visualization. *IEEE Trans. Intell. Transp. Syst.* **16**(6), 2970–2984 (2015)
11. A. Mehdizadeh, M. Cai, Q. Hu, A. Yazdi, M. Ali, N. Mohabbati-Kalejahi, A. Vinel, S.E. Rigdon, K.C. Davis, F.M. Megahed, A review of data analytic applications in road traffic safety. Part 1: descriptive and predictive modeling. *Sensors* **20**(4), 1107 (2020)
12. N. Ferreira, J. Poco, H.T. Vo, J. Freire, C.T. Silva, Visual exploration of big spatio-temporal urban data: a study of New York City taxi trips. *IEEE Trans. Vis. Comput. Graph.* **19**(12), 2149–2158 (2013)
13. R. Krüger, D. Thom, M. Wörner, H. Bosch, T. Ertl, Trajectorylenses—a set-based filtering and exploration technique for long-term trajectory data. *Comput. Graph. Forum* **32**, 451–460 (2013)
14. F. Wang, W. Chen, F. Wu, Y. Zhao, H. Hong, T. Gu, L. Wang, R. Liang, H. Bao, A visual reasoning approach

- for data-driven transport assessment on urban roads, in *2014 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pp. 103–112 (2014)
15. J. Buchmüller, D. Jäckle, E. Cakmak, U. Brandes and D. A. Keim, MotionRugs: visualizing collective trends in space and time, in *IEEE Transactions on Visualization and Computer Graphics*, vol 25, no. 1, pp. 76–86 (2019). <https://doi.org/10.1109/TVCG.2018.2865049>
 16. G. Sun, R. Liang, Q. Huamin, W. Yingcai, Embedding spatio-temporal information into maps by route-zooming. *IEEE Trans. Vis. Comput. Graph.* **23**(5), 1506–1519 (2017)
 17. H. Guo, Z. Wang, B. Yu, H. Zhao, 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*, pp. 163–170. IEEE (2011)
 18. X. Huang, Y. Zhao, C. Ma, J. Yang, X. Ye, C. Zhang, Trajgraph: a graph-based visual analytics approach to studying urban network centralities using taxi trajectory data. *IEEE Trans. Vis. Comput. Graph.* **22**(1), 160–169 (2016)
 19. D. Chu, D.A. Sheets, Y. Zhao, Y. Wu, J. Yang, M. Zheng, G. Chen, Visualizing hidden themes of taxi movement with semantic transformation, in *2014 IEEE Pacific Visualization Symposium*, pp. 137–144. IEEE (2014)
 20. K. Liu, S. Gao, L. Feng, Identifying spatial interaction patterns of vehicle movements on urban road networks by topic modelling. *Comput. Environ. Urban Syst.* **74**, 50–61 (2019)
 21. W. Wang, L. Tao, C. Gao, B. Wang, H. Yang, Z. Zhang, A c-dbscan algorithm for determining bus-stop locations based on taxi gps data, in *International Conference on Advanced Data Mining and Applications*, pp. 293–304. Springer (2014)
 22. A. Bargar, A. Gupta, S. Gupta, D. Ma, Interactive visual analytics for multi-city bikeshare data analysis, in *The 3rd International Workshop on Urban Computing (UrbComp 2014)*, vol 45 (New York, USA, 2014)
 23. G. Andrienko, N. Andrienko, C. Hurter, S. Rinzivillo, S. Wrobel, From movement tracks through events to places: extracting and characterizing significant places from mobility data, in *2011 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pp. 161–170 (2011)
 24. Z. Wang, L. Min, X. Yuan, J. Zhang, H. Van De Wetering, Visual traffic jam analysis based on trajectory data. *IEEE Trans. Vis. Comput. Graph.* **19**(12), 2159–2168 (2013)
 25. C. Lee, Y. Kim, S. Jin, D. Kim, R. Maciejewski, D. Ebert, S. Ko, A visual analytics system for exploring, monitoring, and forecasting road traffic congestion. *IEEE Trans. Vis. Comput. Graph.* **26**(11), 3133–3146 (2020)
 26. G. Boeing, OSMnx: new methods for acquiring, constructing, analyzing, and visualizing complex street networks. *Comput. Environ. Urban Syst.* **65**, 126–139 (2017)
 27. Y. Lou, C. Zhang, Y. Zheng, X. Xie, W. Wang, Y. Huang. Map-matching for low-sampling-rate gps trajectories, in *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*, pp. 352–361 (2009)
 28. L. Lins, J.T. Klosowski, C. Scheidegger, Nanocubes for real-time exploration of spatiotemporal datasets. *IEEE Trans. Vis. Comput. Graph.* **19**(12), 2456–2465 (2013)
 29. H. Doraiswamy, N. Ferreira, T. Damoulas, J. Freire, C.T. Silva, Using topological analysis to support event-guided exploration in urban data. *IEEE Trans. Vis. Comput. Graph.* **20**(12), 2634–2643 (2014)
 30. P. Valdivia, F. Dias, F. Petronetto, C.T. Silva, L.G. Nonato, Wavelet-based visualization of time-varying data on graphs, in *2015 IEEE VAST*, pp. 1–8. IEEE (2015)
 31. M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, KDD'96*, pp. 226–231. AAAI Press (1996)
 32. N. Ferreira, L. Lins, D. Fink, S. Kelling, C. Wood, J. Freire, C. Silva, Birdvis: visualizing and understanding bird populations. *IEEE Trans. Vis. Comput. Graph.* **17**(12), 2374–2383 (2011)
 33. Y. Tang, F. Sheng, H. Zhang, C. Shi, X. Qin, J. Fan, Visual analysis of traffic data based on topic modeling (Chinavis 2017). *J. Vis.* **21**(4), 661–680 (2018)
 34. H. Liu, S. Jin, Y. Yan, Y. Tao, H. Lin, Visual analytics of taxi trajectory data via topical sub-trajectories. *Visual Inform.* **3**(3), 140–149 (2019)
 35. A. Zeidan, E. Lagerspetz, K. Zhao, P. Nurmi, S. Tarkoma, and H. T. Vo, GeoMatch: efficient large-scale map matching on apache spark. *ACM/IMS Trans. Data Sci.* **1**(3) (2020). <https://doi.org/10.1145/3402904>