MobVis: A Framework for Analysis and Visualization of Mobility Traces

Lucas N. Silva[†], Paulo H. L. Rettore[*], Vinicius F. S. Mota[‡], Bruno P. Santos[†]
[†]Dept. of Computer and Systems, Federal University of Ouro Preto, Joao Monlevade, Brazil
E-mails: lucas.novais@aluno.ufop.edu.br, bruno.ps@ufop.edu.br
[*]Dept. of Communication Systems, Fraunhofer FKIE, Bonn, Germany
Email: paulo.lopes.rettore@fkie.fraunhofer.de
[‡]Dept. of Computer Science, Federal University of Espirito Santo, Vitória, Brazil
Email: vinicius.mota@inf.ufes.br

***Abstract*—Due to the increasing location-aware devices, mobility traces datasets have become an essential source for smart cities planning. Given this scenario, we propose MobVis, a framework to characterize mobility traces through different metrics, allowing comparisons between different mobility traces in a simplified way. Furthermore, MobVis can extract and visualize spatial, temporal, and social aspects of mobility data through a Web interface. MobVis architecture has five main components: input data; data preparation; data processing and analysis to extract mobility metrics; visualization; and a web interface. To demonstrate the framework's process, we created a use case analyzing the characteristics of two distinct traces (Taxi and IoT-Objects). Then, through different metrics, we evaluated the data in two aspects: i) descriptive, through a set of graphics and quantitative data that enables characterizing each trace; and ii) comparative, presenting the main differences between the traces.**

***Keywords* - Mobility, IoT, ITS, Smart Cities.**

I. Introduction

Mobility is a fundamental aspect of the development of any society. We have seen exponential growth in the number of mobile devices able to sense the mobility of people and goods from different perspectives. As a consequence, a massive amount of data has been generated. Now, the Smart Mobility paradigm is emerging, which seeks to use those data to analyze and optimize the transportation of people and goods safely, while reducing travel time, emissions, and fuel consumption.

Mobility information has been shared in an increasingly efficient and ubiquitous manner. This can be seen with the increase of Internet of Things (IoT) devices and the evolution of Intelligent Transportation Systems (ITS) that collect large amounts of data regarding mobile entities [1]. Solutions to process, fuse information, visualize and analyze this data are mostly purpose-specific (applications to classify mobility or optimize routes, for example). Performing the study by fusing information to extract knowledge from these data is important not only to understand the mobility of entities itself, but to perform the design and implementation of smart solutions such as communication protocols in mobile networks, smart cities mobility planning, ITS, and Smart Infrastructures.

Recent literature presents different ways of extracting and fusing mobility data such as spatially, temporally, or socially characteristics of mobile entities and their interactions [2]. Fusing mobility information, different metrics can be explored to recognize individual and group mobility patterns and propose applications and protocols that best suit these mobile entities. However, general-purpose solutions that centralize, in a single framework, the data preparation, processing and fusing, analysis, and visualization of mobile entity data, especially open access, are scarce in the current literature.

Therefore, in this paper, we propose Mobility Visualization framework (MobVis), a framework for filtering, processing, analyzing, visualizing, and interacting with mobility data. MobVis aims to provide a simple, interactive, extensible, and open-source environment[1] to compute and visualize mobility metrics in a single environment. Different types of mobility data are supported by the MobVis (e.g. position tracks and contact tracks) and can be easily extended to other formats using the format conversion module. MobVis provides several mobility metrics by fusing the node's information, outputting visualizations based on different aspects such as spatial, temporal, and social. MobVis has a Command Line Interface (CLI) and an easy-to-use Web interface, where the user can interact with visualizations and configuration parameters.

The contributions of this work are listed below:

- The MobVis, a framework for preparing, processing, analyzing, and visualizing mobility traces.
  - It implements spatial, temporal and social metrics, allowing mobility trace characterization.
  - Enabling comparison between mobility traces and an overview of the mobility characteristics of each trace.
- A web service that allows hosting MobVis in a powerful computer to process large datasets.
- A comparative and descriptive evaluation of an use case that considers two mobility traces: real taxi data (*SFCab*) and synthetic IoT-Objects motion trace generated from *SWIM* mobility model.

The paper is organized as follows: Section II reviews the literature, showing the tools that process and analyze mobility data. In Section III, the MobVis design as well as the implementation details are presented. An evaluation of MobVis through use cases is shown in Section IV. Finally, Section V presents our conclusions as well as future perspectives.

[1]MobVis is available on: https://github.com/lucNovais/MobVis

II. Related Work

In this section, we review the current literature of tools that analyze mobility traces through spatial, temporal and social metrics. In [3] the authors proposed the BonnMotion, an open-source tool developed in Java for generating and analyzing synthetic mobility scenarios. Currently, BonnMotion holds a wide set of synthetic mobility models implemented such as *Random Waypoint*, SWIM (*Small World in Motion*), *Random Walk*, *Gausian Markov* among others. Therefore, the tool is useful in researchers that intends to explore synthetic mobility.

BonnMotion implements metrics to analyze the generated traces such as spatial distribution of nodes and their respective contacts. However, there are many well-known metrics that the tool does not implement, e.g., the radius of gyration [4]. Thus, BonnMotion can be considered a limited tool for analyzing mobility data. Furthermore, BonnMotion's analysis module is designed to extract metrics from the mobility traces generated by the application itself, this imposes difficulties to perform analysis based on data from different sources. Another disadvantage is that BonnMotion does not implement a module to visualize the generated and extracted data, thus it is necessary to use external tools to visualize the trace and metrics.

Bandicoot [5] is an open-source library written in Python to extract features from mobile phone data. The library allows users to perform different data visualization such as individual/group, spatial, and social as well as data filtering and verification. Bandicoot can detect problems in mobile phone data, such as lost locations and incorrect dates. The tool is easy to use for extracting metrics, data visualizing and filtering. However, it is limited to mobile phone data, and so far there are no plans to extend the project to support different data types.

Another open-source tool that has modules for extracting, classifying, and comparing features of real or synthetic mobility traces is MOCHA [2]. The main goal of MOCHA is to compare different mobility models with each other or with real data. MOCHA extracts various spatial, temporal, and social metrics from mobility trace into different output files. In addition, there is a specific module to perform comparisons that use t-SNE [6] technique, which was chosen to highlight the relationships between different mobility data. However, the visualizations implemented in MOCHA are limited to one that does the comparison between different mobility data. MOCHA does not cover spatial and temporal visualizations.

Recently, Scikit-mobility [7], an open-source library written in Python was released. The library aims to represent mobility data in a specific data structure, perform pre-processings, compute mobility metrics, generate synthetic data from mobility models, and estimate privacy risks contained in a dataset. The library works from two main data structures, called *TrajDataFrame*, used to represent movement data per individual, and *FlowDataFrame*, used to indicate flows of several simultaneous individuals. From this, all tasks performed by the tool are conducted separately according to the data types.

Two data formats are supported by the tool: the individual mobility data (e.g. latitude, longitude, identifier, and date), and flow data (e.g. flow origin, flow destination, and the number of entities involved). Although there are many metrics to be computed such as social measures (e.g.: nodes' contacts and social topologies), the tool does not support them, since Scikit-mobility was not built to handle contacts between nodes. It is also worth mentioning that despite extracting the metrics, the library has no module for visualizing the metrics and does not support many data formats.

TABLE I: Comparison of features among the current tools.

Features		**Tools**				
		Bonn Motion	Bandicoot	MOCHA	Scikit mobility	**MobVis**
Metrics	Spatial	✓	❒	✓	✄	✓
	Temporal	✄	❒	✓	✄	✓
	Social	✓	❒	✓	-	✓
Visualiz.	Metrics	-	❒	-	-	✓
	Trace	-	❒	-	✓	✓
	Interaction	-	❒	-	✓	✓
Diff. Input Formats		-	-	-	-	✓
Web Interface		-	✓	-	-	✓
Synthetic Model		✓	-	-	✓	-

✓aspect covered
✄ aspect slightly covered
❒ aspect covered but for specific use case

The Table I summarizes the related work with MobVis in perspective. Several projects can benefit from the information provided by MobVis. For example, in [8], [9], the authors study a priori movement of IoT's objects to design mobility aware protocols. In [10], [11], the authors show that mobility pattern analyses using mobility traces are useful in the project of solutions for ITS. It is important to note that MobVis is a facilitator to the study and visualization of mobility traces and it does not do inferences about the data.

III. Design

This section describes the architecture of MobVis, its components, and the data flow diagram as shown in Figure 1. The MobVis is composed of two main parts the *back-end* and the *front-end*. The former, the focus of this paper, is responsible for manipulating, preprocessing, processing, exporting, and generating graphs and tabulated information. The second part, *front-end*, allows users to interact with the different visualizations and statistics about the mobility data.

In Figure 1, each component presents the fundamental activities performed by MobVis. The framework is implemented in Python and its architecture allows easy extensions, enhancements, and maintenance. The *front-end*, a web interface, is developed following the Model-Template-View (MTV) convention used by the framework Django[2]. The MobVis has the MIT license and it is available in a repository online [1].Each component of the architecture is detailed below.

A. Input

The MobVis has two data entries: i) the mobility trace and ii) the configuration parameters to start the necessary processing. For the mobility trace, it is necessary to indicate the order in which some information appears (e.g. entity

[2]https://www.djangoproject.com/

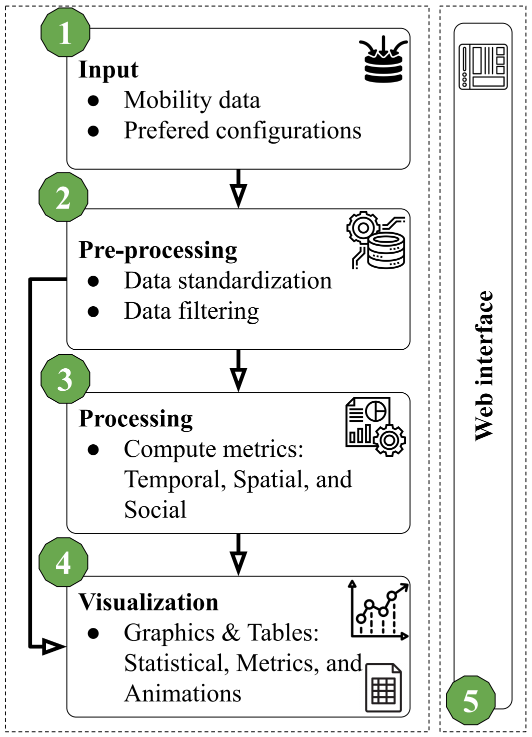

1
Input
• Mobility data
• Prefered configurations
2
Pre-processing
• Data standardization
• Data filtering
3
Processing
• Compute metrics: Temporal, Spatial, and Social
4
Visualization
• Graphics & Tables: Statistical, Metrics, and Animations
5
Web interface

Fig. 1: MobVis pipeline.

identifier, latitude, longitude, and timestamp). This order is important so that the module can read the information correctly and perform the framework's next steps.

The configuration parameters are important for example to perform or not actions on the data pre-processing step (e.g. temporal sorting) or to indicate which types of visualization will be created or exported. A non-exhaustive list of parameters is: sorting the trace, specifying the metrics to be computed, visualizations and tables to be displayed. It is important to note that, depending on the size of the trace and the parameters indicated the processing time can be affected.

Both mobility trace file and configuration parameters can be entered via a web interface (or command line in the local setup). After uploading that information, the data is directed to the pre-processing module, which is detailed below.

B. *Pre-processing*

Converting the input data to a standard format is the main operation among the activities performed in the preprocessing module. it is necessary to standardize the data at this stage because, sometimes, the data may not be temporally ordered, or not have a proper column alignment. The basic standard uses by MobVis is {id, t, x, y}, more information can be found in the framework repository [1].

The sorting is a key step in the pre-processing step. This step is fundamental for the correct information processing and extraction algorithms. For the MobVis the data must be sorted by identifier and timestamp, thus algorithms that need to iterate over the data to their job following the better ordering logic. The sorting method used by MobVis is the Quicksort.

At the end of this step, the data will be standardized and ready for the processing step. It is worth mentioning that, for the creation of some visualizations involving, exclusively, the position of the nodes on the map or spatial plan, the result of this preprocessing step can be forwarded directly to the visualization module.

C. *Processing*

The processing step consists of extracting mobility metrics and characteristics from the data. Therefore, the module performs the extraction of three main characteristics for the mobility analysis: i) geo-locations, ii) the entities' "homes" (home locations), iii) and contact detection.

According to [12], a geo-location can be determined as a stay-location in which one or more entities of the mobility trace have spent a time greater than a certain predefined threshold τ. Whereas a stay-location is a set of location points $sl = p_m, p_{m+1}, p_n$ such that $\forall\ m < i \leq n, ||p_m - p_i|| \leq D_{max}$, where D_{max} is a pre-defined distance threshold. From the definition of geo-locations, we also define the home-locations that are the geo-locations in which the nodes spend most of time. This definition for home-locations is widely used in the literature [2].

A fundamental mobility aspect to be analyzed is the topology built based on the contacts between the mobile entities [10]. These contacts indicate that the entities would be able to communicate and exchange data in a network. Here, a contact is considered whenever two entities are at a distance less than a threshold R at the same instant of time in the mobility trace. The threshold R is commonly called transmission radius and the distance between the entities can be computed using either the Haversine or Euclidean distance, depending on the type of trace. Both R and distance are configuration parameters of MobVis.

After that, it is possible to perform the calculations related to the mobility of entities individually or in groups [2], [4]. Three types of metrics can be extracted by MobVis such as spatial, temporal, or social metrics. Spatial metrics reflect the behavior of entities concerning the space they occupy; temporal metrics concern the time series in which the movements happen and; social metrics concern the possible interactions between entities and collective behaviors. Table II presents a summary of the metrics currently supported by MobVis. We emphasize that MobVis is flexible enough to support easily new mobility metrics additions.

D. *Visualization and Data Extraction*

Once the metrics are computed, the visualization and export module starts running. At this point, the information is ready to be exported into tabular files (e.g.: csv) or visualized in static graphics (e.g.: png, jpeg, or pdf) and interactive graphics (via web interface) that allows manipulation for better viewing of the results.

TABLE II: Available MobVis metrics.

Metrics	Type	Short description
Radius of Gyration	Spatial	Dimension or depth of a trajectory
Travel Distance	Spatial	Distance traveled between consecutive stopping points
Visit Time	Temporal	Time spent in each geo-loc.
Travel Time	Temporal	Time spent on travel distance
Visit Sequance	Spatiotemporal	Order of geo-locs. visited and the time spent at each stop.
Contact Duration	Social	Time when two entities are within a transmission radius
Inter-contact Time	Social	Time between two consecutive contacts

To export the files and compile scatter plots, histograms, probabilistic plots, among others, the MobVis makes use of a set of Python libraries such as *Lib csv* and *Plotly*. Thus, with the processed trace it is possible to produce visualizations that depict the movement of the nodes in the area of interest or even display regions that have a higher concentration of movement. With the calculated metrics, it is possible to obtain statistical graphics that reflect the characteristics of the entities. It is worth mentioning that the geo-locations, home-locations, and the contacts between entities, although not specifically treated as metrics in this work, are also used to generate different visualizations within this module.

Notice, MobVis has a web interface for accessing the static and interactive resources. This web interface was developed using the Django framework to facilitate the interaction with the *back-end CLI* of the MobVis, allowing a user to access through a web page, the functionalities of the framework. Although the scope of this paper is focused on the *back-end* of the MobVis, the web interface is a relevant part of interacting with and facilitating the use of the framework.

IV. USE CASE

Mobility analysis usually requires researchers to load, filter, and visualize the data by using a combination of tools and/or programming languages. Hereby, we present a use case of MobVis to study two distinct, representative, and well-known mobility dataset traces from the literature: i) a synthetic IoT-Objects motion trace, referred as "*IoT-Obj*", generated by the *Small Worlds in Motion* mobility model (*SWIM*), a model that mimics human mobility based on the idea that people visit nearby locations and points of interest (such as work and home) often than locations far away [13]; ii) a real fleet taxicab mobility trace from San Francisco city (referred as "*SFCab*") [14]. Both traces contain at least a timestamp, an identification of the node, and coordinates.

Table III summarizes MobVis parameters for each mobility trace. For the *SFCab* trace, we took a sample of the first 50 taxis ordered by ids to enable faster processing and better visualization in small graphics. For the IoT-Obj trace, we used a sample of first 100 IoT-objects ids taken from the work [13]. The authors changed the distance scale where the position of the nodes is in a square area of 0 to 1, where 1 represents 4 km of the original simulation area. The MobVis analyzes the traces in two aspects: i) an overview and visualization of basic mobility characteristics, and ii) a comparative analysis to observe the differences of the traces through the metrics calculated by the MobVis.

TABLE III: Use case parameters.

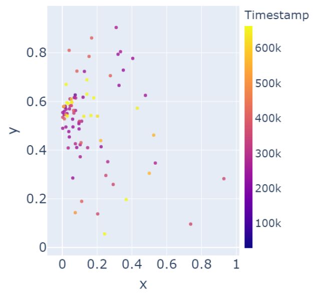
Parameters	IoT-Obj	SFCab
Number of nodes	100	50
Max Dist. (km)	0.014	30
Max Pause (minutes)	10	10
Contact radius (meters)	0.130	20
Type of Distance	Euclidean	Haversine

A. Overview of Mobility Characteristics

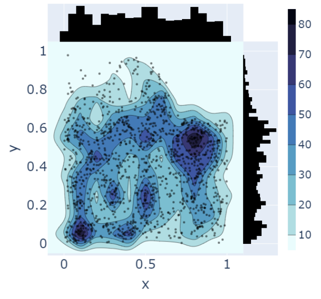
1) IoT-Obj Trace: Figure 2 presents an overview of IoT-Obj trace mobility characteristics. Figure 2a shows the behavior of one node in the simulation space over time, in which the color bar on the right side of the figure represents the simulation time in seconds. As can be seen, the node concentrates its movement in a region, representing a characteristic of the IoT-Obj model which has short mobility most of the time. Figure 2b presents the density of all nodes in each region. In this figure, the different contour levels and color intensity indicate the number of times that waypoints were detected in the regions of the map. Moreover, the densest regions may represent the points of interest defined by the trace. The visualization of the node's movement and density allows observing the area of interest of a node and the busiest region of the area.

As mentioned, MobVis extracts different metrics from the mobility trace such as the spatiotemporal characteristic shown in Figure 2c. This figure shows the histogram of the Travel Distance metric to all nodes of the IoT-Obj trace. It is possible to notice that the nodes tend to make short trips with few occurrences of long distances traveled. In addition, Figure 2d shows the top 10 geo-locations, where a mobile entity spends more time. One can see in such visualization how sparse are the points of interest from one or more nodes. This analysis may be interesting for the design of solutions that need to capture the spots where elements meet each other or find popular places.

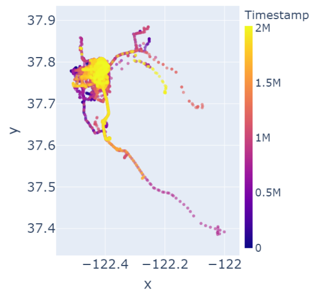
2) San Francisco Cab Trace: To understand the *SFCab* trace, Figure 3 presents its overview characteristics, where Figure 3a shows the mobility performed by a specific cab. In this case, we can see that the cab moves most of the time in the central area of San Francisco, making few trips beyond. In addition, the density of movements is shown in Figure 3b.



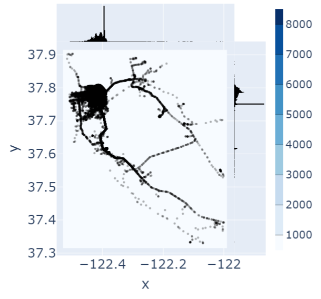
(a) Node movement



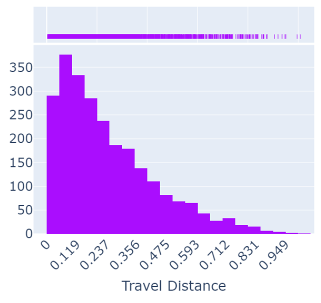
(b) Density



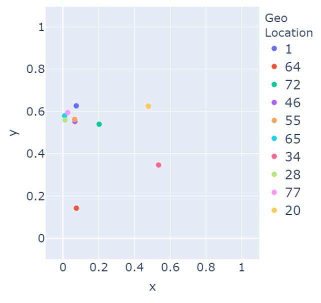
(a) Node movement



(b) Density

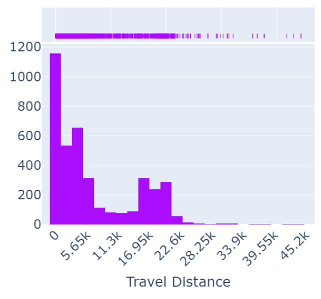


(c) Travel distance

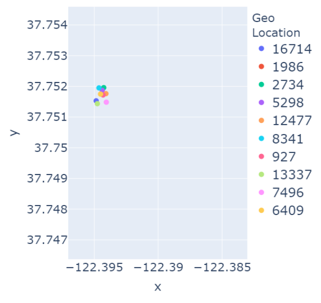


(d) Geo-locations

Fig. 2: Overview of IoT-Objects mobility trace.



(c) Travel distance



(d) Geo-locations

Fig. 3: Overview of *SFCab* trace.

Due to the road network characteristics of San Francisco, we can see a higher density in a small region. Turning now to the characteristic of the movement of cabs, Figure 3c depicts the histogram of the travel distances for all cabs. Again, it is possible to notice a greater preference for short trips. However, the histogram also shows a high frequency of travels between 16 and 22km, which may refer to travels for neighboring cities.

Figure 3d presents the top stay spots where one taxi spends more time. A common and expected pattern observed in the *SFCab* dataset is that the stay points are close to each other. This behavior is expected because usually taxis stand in the same location until pick someone up and, after a typically short travel distance, drop someone off.

B. Comparative Analysis

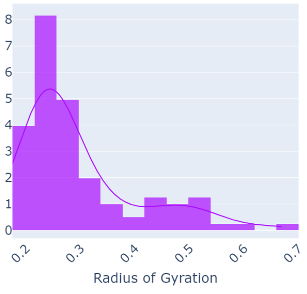
The comparative analysis comes with the capability of MobVis to characterize the mobility traces through different metrics, as shown in Table II, and expose the results enabling an easy comparative analysis between mobility traces. Therefore, we compared IoT-Obj and *SFCab* traces using the following metrics: the Radius of gyration, Visit Time, Travel Time, and Inter-Contact Time as shown in Figure 4. We use 4 groups of sub-figures to turn easy the visualization, each sub-figure shows two visualizations, where the leftmost concerns to IoT-Obj dataset and the rightmost *SFCab* trace.

The Radius of Gyration metric [4] can be visualized in Figure 4a, where it is possible to compare that IoT objects (they move like humans because *SWIM*) present shorter radius of gyration than Taxis. While nodes in IoT-Obj trace have a low Radius of Gyration, moving close to the point of interest (e.g., homes, market, etc), mobile nodes in *SFCab* dataset tend to move further around the city. Indeed, in the *SFCab*, most of the travels are greater than 4km and lower than 20km, covering higher density regions. Notice that, the density of radius of gyration is measured in μ, meaning 10^{-6}.

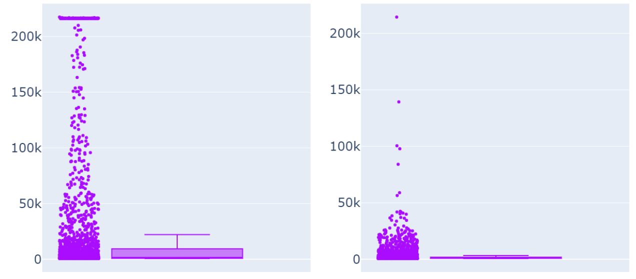
The Visit Time metric is shown in Figure 4b, it is possible to note that the dispersion is different between both traces. For IoT-Obj, it is possible to see a greater variation that can be explained by the attempt to mimic human beings who quickly visit several places and stands for a long period in points of interest (home, work, etc). For Taxis, as they usually are moving around, they do not present high variation. Moving to the Travel Time metric shown in Figure 4c. It is noticed that the mobile elements spend more time from a geo-location to another in the IoT-Obj dataset rather than *SFCab*. It can be explained by the fact the IoT-Obj simulates human-like walking from one place to other by foot, thus, it tends to spend more time between geo-locations.

To illustrate the visualization of a social characteristic, Figure 4d presents a histogram representing the social metric Inter-contact Time. This metric concerns the time interval when a pair of nodes re-encounter each other. In IoT-Obj trace, most pairs of nodes meet in small time intervals. This characteristic may reflect social aspects. For *SFCab*, the majority of cabs re-encounter each other within larger intervals compare to IoT-Obj nodes. Moreover, some cabs may take longer to re-encounter each other on average, spending up to a few days. Most of the cabs circulate in the downtown area of San Francisco, and we also must consider the cabs working hours. The inter-contact time can be useful for novel communications applications in ITS, for instance.

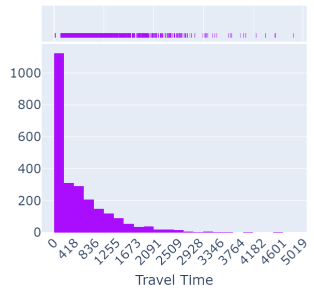
In summary, several metrics were extracted and arranged in statistical graphs to characterize and compare data from different sources and mobility patterns. All graphs presented



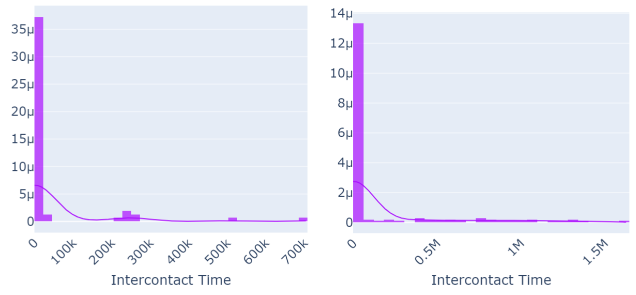
(a) Radius of Gyration



(b) Visit Time



(c) Travel Time



(d) Inter-contact Time

Fig. 4: Comparative analysis between the IoT-Objects (leftmost) and *SFCab* (rightmost) traces.

in the use case can be generated for any type of metric available in the MobVis, allowing a variety of ways to compare mobility data. Another important aspect is that mobility data can have inconsistencies such as location gaps. This issue is due to the fact that the data collected from *GPS* is susceptible to failures and may stop providing data for a long time. Such inconsistencies can be corrected by filtering and error correction before being processed by MobVis.

V. Conclusion

Mobility data analysis became fundamental to the emerging smart mobility concept. This paper presents MobVis, a framework for mobility data analysis and visualization, which aims to allow researchers to do fast and easy processing and visualization of mobility datasets. MobVis provides a library with a set of functions that allows to compute and visualize the temporal, spatial, and social metrics on a given trace. These metrics may allow researchers to better understand the node's mobility characteristics as well as allow easy comparison between different mobility traces. We demonstrated MobVis by analyzing and comparing a synthetic IoT-Objects trace and a taxicab fleet from San Francisco city (SFCab). We selected a set of graphics to characterize these traces demonstrating MobVis' capabilities.

As future work, we plan to introduce new visualizations, such as map overlay, and increase the number of useful metrics for mobility analysis. Furthermore, we plan to improve the online version of MobVis.

Acknowledgment

We would like to thank the research agencies Bundeswehr (BAAINBw/WTD81), Fapemig, FAPES, and CNPq/CAPES.

References

[1] Rettore, P. H. L., G. Maia, L. A. Villas, and A. A. F. Loureiro, "Vehicular Data Space: The Data Point of View," *IEEE Communications Surveys Tutorials*, vol. 21, no. 3, pp. 2392–2418, 2019.

[2] F. R. de Souza, A. C. Domingues, P. O. Vaz de Melo, and A. A. Loureiro, "MOCHA: A tool for mobility characterization," in *Proceedings of the 21st ACM International Conference on MSWIM*, 2018.

[3] N. Aschenbruck, R. Ernst, E. Gerhards-Padilla, and M. Schwamborn, "BonnMotion: A Mobility Scenario Generation and Analysis Tool," in *Proceedings of the 3rd ICST*, Brussels, BEL, 2010.

[4] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi, "Understanding individual human mobility patterns," *Nature*, 2008.

[5] Y.-A. De Montjoye, L. Rocher, and A. S. Pentland, "Bandicoot: A python toolbox for mobile phone metadata," *The Journal of Machine Learning Research*, 2016.

[6] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE." *Journal of machine learning research*, 2008.

[7] L. Pappalardo, F. Simini, G. Barlacchi, and R. Pellungrini, "Scikit-mobility: a Python library for the analysis, generation and risk assessment of mobility data," 2019.

[8] B. P. Santos, O. Goussevskaia, L. F. Vieira, M. A. Vieira, and A. A. Loureiro, "Mobile matrix: routing under mobility in IoT, IoMT, and social IoT," *Ad Hoc Networks*, vol. 78, pp. 84–98, 2018.

[9] B. P. Santos, P. H. Rettore, L. F. Vieira, and A. A. Loureiro, "Dribble: A learn-based timer scheme selector for mobility management in IoT," in *WCNC*. IEEE, 2019.

[10] F. D. Cunha, F. A. Silva, C. Celes, G. Maia, L. B. Ruiz, R. M. Andrade, R. A. Mini, A. Boukerche, and A. A. Loureiro, "Communication analysis of real vehicular calibrated traces," in *ICC*. IEEE, 2016.

[11] C. Celes, A. Boukerche, and A. A. Loureiro, "Mobility Trace Analysis for Intelligent Vehicular Networks: Methods, Models, and Applications," *ACM CSUR*, 2021.

[12] M. Zignani, S. Gaito, and G. Rossi, "Extracting human mobility and social behavior from location-aware traces," *WCMC*, 2013.

[13] C. Marche, L. Atzori, V. Pilloni, and M. Nitti, "How to exploit the Social Internet of Things: Query Generation Model and Device Profiles' Dataset," *Computer Networks*, p. 107248, 2020.

[14] M. Piorkowski, N. Sarafijanovic-Djukic, and M. Grossglauser, "CRAWDAD dataset epfl/mobility (v. 2009-02-24)," Feb. 2009.