MobMetrics: A Modular Architecture for Computing and Visualizing Mobility Metrics

Márcio P. C. Filho[1*], Lucas Novais[2*], Leon L. Lausberg[4],
Peter Sevenich[4], Paulo H. L. Rettore[3,4], and Bruno P. Santos[1]
[*]Authors contributed equally to this research.
[1]Department of Computer Science, Federal University of Bahia, Brazil
[2]Department of Computer and Systems, Federal University of Ouro Preto, Brazil
[3]Department of Computer Science, Federal University of Minas Gerais, Brazil
[4]Department of Communication Systems, Fraunhofer FKIE, Bonn, Germany
Email: {marciofilho, coelhoj, bruno.ps}@ufba.br, lucas.novais@aluno.ufop.edu.br,
s6lelaus@uni-bonn.de, {peter.sevenich, paulo.rettore.lopes}@fkie.fraunhofer.de

***Abstract*—To support the analysis of mobility traces and the processing of mobility metrics, we propose MobMetrics, a web-based system designed to process mobility traces and extract comprehensive mobility metrics. The system is an open-source application that enables users to upload raw mobility data (geospatial or n-dimensional Cartesian data) and obtain a wide range of metrics, categorized into four groups: social, spatial, kinematic, and temporal. These metrics enable detailed pattern analyses of entities' movement behavior. Furthermore, the platform integrates a metric-based multidimensionality reduction technique and graphs/charts to visualize both metrics and traces. As a modular, easy-to-extend, and open-source project, it allows users to add their own metrics and visualizations. We analyzed mobility traces with different characteristics by comparing mobile entities and their respective labels, aiming to analyze, compare, and cluster them.**

***Index Terms*—Mobility Metrics, Mobility, Traces, IoT**

I. Introduction

Mobility is an important factor in our lives. When its understanding is embedded into computational devices and systems, more flexible, adaptive, and ubiquitous applications emerge. The analysis of mobility patterns is essential both for developing efficient computational protocols and routines, and for identifying entities with complex behaviors, which enables inferences across several applied domains such as smart cities and ubiquitous computing [1], [2], mobile networks and security [3], [4], or high-performance processing [5]. However, despite the wide applicability of such analyses, open tools that integrate both metric computation and visualization remain scarce. It is still common to rely on ad-hoc solutions, limited to specific contexts and with low reusability.

Thus, we propose *MobMetrics*, an intelligent tool that processes raw mobility data and generates interactive and exportable visualizations. The user can feed *MobMetrics* with georeferenced or n-dimensional Cartesian mobility data. The system then processes these data using a broad set of metrics grouped into four main categories: spatial, temporal, kinematic, and social. In addition, *machine learning* techniques and dimensionality reduction methods are applied to enhance analytical capacity, pattern detection, and decision support. As a result, the user can graphically interact with the system, exporting either visualizations or processed tabular data, which facilitates reuse.

MobMetrics is open-source, developed in Python-Web using the *Django framework* [6], and easily extensible, allowing users to add or modify functionalities. *MobMetrics* also supports seamless integration with Python libraries for machine learning and artificial intelligence. The main contributions of this work are:

- *MobMetrics* is an open-source system[1] for processing, analyzing, and visualizing mobility data:
  - Computes spatial, temporal, kinematic, and social mobility metrics;
  - Uses machine learning techniques to compare mobility traces;
  - Source code and data are available in a public repository, with documentation for installation and basic system usage.
- An interactive web interface that allows users to analyze and export processed data and generated visualizations.
- A practical, executable example included in the project repository, enabling users to easily test the main functionalities.

The remainder of this paper is organized as follows: Section 2 presents the related work; Section 3 describes the structure of *MobMetrics*; Section 4 details the computed metrics; Section 5 discusses the use cases; and finally, Section 6 presents the conclusions and future directions.

II. Related Work

Tools and frameworks aimed at mobility data analysis have become increasingly relevant, especially given the growing availability of geographic traces collected from mobile devices. In this context, several approaches have been proposed for extracting meaningful information from such data. These approaches are compared with **MobMetrics** in Table I, which highlights their main features and limitations.

[1]https://github.com/Marcio-Carvalho27/MobMetrics

TABLE I: Related Work

Tools	Metrics				Visualizations	Web Interface
	Social	Temporal	Spatial	Kinematic		
Bandicoot	✓	✓	✓	-	-	-
BonnMotion	✓	▲	✓	-	-	-
MobVis	▲	▲	▲	-	▲	✓
MOCHA	▲	▲	▲	▲	-	-
MovingPandas	-	▲	✓	▲	✓	-
scikit-mobility	-	▲	▲	-	▲	-
Trackintel	-	✓	✓	-	✓	-
MobMetrics	✓	✓	✓	✓	✓	✓

Legend: ✓ = Fully supported ▲ = Partially supported

Bandicoot [7] is an open-source Python library for extracting metrics from mobile phone data. The tool offers functionalities for individual and group visualization, spatial and social analysis, as well as filters and automated checks that identify inconsistencies such as missing locations or invalid dates. Although Bandicoot stands out for its ease of use and its ability to handle common issues in mobility data, its application is limited exclusively to telecommunication data. In contrast, *MobMetrics* adopts a more flexible approach, processing different formats of geographic traces such as GeoJSON and CSV files making it applicable to a broader range of studies and domains.

In [8], the authors proposed *BonnMotion*, an open-source tool developed in Java for generating and analyzing synthetic mobility scenarios. The tool implements several classical mobility models, such as *Random Waypoint*, *SWIM (Small World in Motion)*, *Random Walk*, and *Gaussian Markov*, among others. As a result, BonnMotion is widely used by researchers interested in simulating artificial mobility behaviors. Unlike *MobMetrics*, which operates on real or simulated tracking data and focuses on empirical analysis of urban mobility, BonnMotion centers on the simulation of synthetic traces, making it more suitable for studies based on hypothetical or controlled scenarios.

One of the most similar works in this field is MobVis [9], a tool designed for visual exploration of mobility data. *MobMetrics* was conceived as a continuation and expansion of this proposal. MobVis allows the analysis of traces through visualizations; however, it does not incorporate systematic metric computation mechanisms or support for machine learning and AI-based analyses. In contrast, *MobMetrics* adopts an extensible approach focused on the quantitative characterization of traces through spatial, temporal, kinematic, and social metrics, in addition to integrating *machine learning* techniques and modular analysis components, significantly enhancing the analytical potential of the tool.

MOCHA [10] is an open-source tool composed of modules for extracting, classifying, and comparing mobility trace features whether real or synthetic. Its main goal is to enable comparisons between different mobility models or between synthetic models and real data. To this end, the tool computes various spatial, temporal, and social metrics, stored in separate output files. One of its distinguishing features is the use of the *t-SNE* technique for comparison and dimensionality reduction, facilitating pattern identification across datasets. However, MOCHA presents visualization limitations: its representations are restricted to trace comparisons, without native support for more detailed spatial or temporal visualizations. *MobMetrics*, on the other hand, provides a broader set of interactive visualizations, enabling rich, exploratory analyses of both the traces and their *stay points* (see Section III-B3).

MovingPandas [11] is a Python library that supports operations on trajectory data. Its features include trajectory modeling, stop detection, and interactive visualization generation. Although it provides important resources for trajectory manipulation and visual exploration, *MovingPandas* is more oriented toward interactive spatial analysis than toward the systematic definition of mobility metrics. In contrast, *MobMetrics* proposes a more structured and extensible approach, integrating metric computation with customized visualizations.

scikit-mobility [12] is a Python library focused on human mobility analysis, emphasizing aggregated metrics, synthetic trajectory generation, and privacy risk assessment. Despite its analytical robustness, *scikit-mobility* is primarily oriented toward exploratory analyses conducted within *notebooks* and does not natively support integration with web interfaces or modularization of customized metrics. In contrast, *MobMetrics* provides a production-oriented solution emphasizing metric extensibility and visualizations tailored for comparative entity analysis.

Trackintel [13] is a Python library designed for analyzing spatiotemporal data of individual mobility, without a focus on social metrics. In contrast, *MobMetrics* also performs the identification of *stay points* and offers a broader set of metrics while enabling the analysis of interactions between entities, such as contact detection. Both tools provide visualization capabilities.

Therefore, this work stands out by offering a unified infrastructure for processing, computing, and visualizing trace metrics, combining flexibility and modularity—features still rarely integrated into existing solutions.

III. Mobility Metrics

In this section, we describe the metrics natively implemented in *MobMetrics*, presenting the required parameters and the corresponding calculation methods. From a functional

TABLE II: Relationship between Metrics and Calculation Groups

	Kinematic			Temporal			Spatial									Social		
Groups	**JAS**	**SVC**	**TAS**	**JT**	**TT**	**VTVC**	**AVC**	**JD**	**TD**	**RG**	**SP**	**SPI**	**TC**	**TADA**	**SC**	**Ctt**	**Etp**	**QEtp**
Local Metrics	x	x	x	x	x	x	x	x	x	x	x	-	-	x	x	x	-	x
Global Metrics	-	x	-	-	-	x	x	-	-	-	-	-	x	-	x	-	-	x

standpoint, metrics can be classified as **Local**, which analyze the individual behavior of an entity, and **Global**, aimed at the joint analysis of multiple entities. There are also **Mixed** metrics, whose use may vary between individual or collective contexts, depending on how they are applied. In addition, some metrics do not refer directly to entity traces but to precomputed metrics, such as the *Stay Point Importance Degree* (see Section III-B3). Table II presents an overview of the implemented metrics and their classifications.

Complementarily, the metrics are organized according to the type of information they analyze, grouped into four main categories: **Spatial**, **Temporal**, **Kinematic**, and **Social**. This structure aims to facilitate a comprehensive characterization of the mobility patterns observed in the traces.

A. Kinematic, Temporal, and Transversal Metrics

Certain metrics share similar concepts across different categories, which justifies presenting them together in this section. In *MobMetrics*, entity displacements are segmented in two main ways: *journeys* and *travels*. Based on these segmentations, metrics associated with time, distance traveled, and average speed are computed, enabling a more refined analysis of mobility dynamics.

- **Journey Metrics**: the Journey Time (JT), Journey Distance (JD), and Journey Average Speed (JAS) metrics are classified, respectively, as temporal, spatial, and kinematic metrics. They represent the total time, distance traveled, and average speed in each entity *journey*.
- **Travel Metrics**: the Travel Time (TT), Travel Distance (TD), and Travel Average Speed (TAS) metrics follow the same classification (temporal, spatial, and kinematic, respectively) and represent the same aspects, but segmented by *travel* rather than *journey*.

In addition, three metrics use the statistical concept of coefficient of variation to quantify the variability of different aspects of displacement:

- **Speed Variation Coefficient (SVC)** [14]: the ratio between the standard deviation and the mean of observed speeds along the trace; useful for identifying pace variations during displacement.
- **Angle Variation Coefficient (AVC)**: measures variability in the direction of the entity's displacement, based on angles between consecutive point pairs.
- **Visit Time Variation Coefficient (VTVC)**: assesses the consistency of dwell times at different *stay points*, indicating whether there are regular or irregular visitation patterns.

Despite the similarity in their general formula (σ/μ), these metrics serve distinct purposes and belong to different categories (kinematic, spatial, and temporal), reflecting different aspects of entity behavior.

B. Spatial

This group of metrics focuses on the geographic distribution of traces, analyzing spatial coverage and dispersion.

*1) **Travel Average Direction Angle (TADA)**:* The TADA metric represents the entity's average displacement angle along the trace, measured with respect to true north. For each pair of consecutive points, the direction angle is computed, and the mean of these values is then obtained considering the trigonometric circle.

$$TADA = \frac{\sum_{i=1}^{m} \theta_i}{m}$$

where θ_i is the direction angle between points i and $i+1$ along the trace, and m represents the total number of consecutive point pairs.

This metric is used, for example, in the calculation of the Angle Variation Coefficient.

*2) **Trajectory Correlation (TC)**:* This metric computes the degree of similarity among an entity's trajectories using cosine similarity. It can be obtained using the following expression:

$$TC = 1 - \sigma\left(1 - \cos(\mathbf{v}_i, \mathbf{v}_j)\right)$$

$$\forall i < j$$

In other words, trajectory correlation is given by 1 minus the standard deviation of *one minus* the cosine similarity across all pairs of trajectories performed by an entity.

*3) **Stay Points (SP)**:* This metric represents locations where the entity remained for a minimum period of time [15]. Two parameters are required for identification: **Distance Threshold** (D_{ths}) and **Time Threshold** (T_{ths}).

A *Stay Point* P is an ordered set of points $P = \{p_1, p_2, \ldots, p_n\}$ that satisfies:

$$\forall\, 1 \leq i \leq n,\ ||p_1 - p_i|| \leq D_{\text{ths}}$$

$$\Delta T_P \geq T_{\text{ths}}.$$

That is, all points in P are at most D_{ths} away from p_1, and the total time ΔT_P is equal to or greater than T_{ths}.

*4) **Stay Point Importance Degree (SPI)**:* This metric computes the Importance Degree of a *Stay Point*, taking into account metrics related to that point.

$$SPI = \alpha \cdot (1 - \text{Etp}) + \beta \cdot T_{visits} + \gamma \cdot Ctt_{sp}$$

where T_{visits} is the total time of visits to the *stay point*, Ctt_{sp} is the number of contacts in the region, and Etp represents the entropy associated with the *stay point*.

The values of α, β, and γ are parameters that define the relative weight of each metric in the SPI composition, respecting $\gamma > \alpha > \beta$. By default, these values are set to 0.4, 0.4, and 0.2, respectively.

*5) **Radius of Gyration (RG)**:* The Radius of Gyration (RG) [16] is the radius of the circle centered at the trace's centroid that contains most of the entity's displacements:

$$\text{RG} = \sqrt{\frac{1}{N}\sum_{i=1}^{k} \|t_i - t_m\|^2}$$

where $T = \{t_1, t_2, \ldots, t_k\}$, t_m is the center of mass of the travel, and N is the number of points.

*6) **Spatial Cover (SC)**:* Spatial cover is a metric derived from Quadrant Entropy; both are computed together, counting how many of the n distinct quadrants an entity has visited. The value of n depends on the user-specified *quadrant parts* parameter (Q_{parts}), as shown in Figure 2: if each axis is divided into Q_{parts} parts and there are 2 axes (2D), then $n = Q_{\text{parts}}^2$.

It allows computing the total spatial coverage considering all traces as well as the spatial coverage of each entity.

C. Social

This group of metrics aims to quantify interactions among entities, considering proximity, encounters, and co-presence patterns.

*1) **Contacts (Ctt)**:* Computing Contacts (Ctt) [9] requires the user-defined *radius threshold* parameter (R_{ths}) at submission time (Fig. 2). Given R_{ths}, there is a contact between two entities when, at the same *timestamp*:

$$\|e_n - e_m\| \leq R_{\text{ths}},$$

where e_m and e_n are points from distinct entities.

To reduce the quadratic verification cost across all pairs, *MobMetrics* filters traces by *timestamp*, ignoring instants with only one entity and avoiding unnecessary comparisons.

This approach detects instantaneous contacts only. To identify continuous contacts, the *Contact Time Threshold* parameter ($CttT_{tsh}$) is used to group successive contacts into a single event if the interval between them is less than or equal to $CttT_{tsh}$.

*2) **Entropy (Etp)**:* Entropy [17], computed for *stay points*, measures the diversity of movements of a point P and is based on Shannon entropy:

$$\text{Etp} = -\log_2\big(p(P)\big), \quad p(P) = \frac{V(P)}{V_t},$$

where $V(P)$ is the number of visits to P and V_t is the total number of visits to all *stay points*.

*3) **Quadrant Entropy (QEtp)**:* Derived from Stay Point Entropy, QEtp divides the trace into n quadrants and computes the entropy for each one. The value of n is the same as presented in the Spatial Cover metric.

D. Derived Metrics

Some metrics can be derived from others already defined. For example, Local Metrics can be aggregated via the mean to compose Global Metrics. In addition, certain information such as dwell time at *Stay Points* or the number of identified journeys although not metrics in the strict sense, can provide relevant *insights* for mobility analysis; these are factors encountered while computing other metrics.

IV. MobMetrics Structure

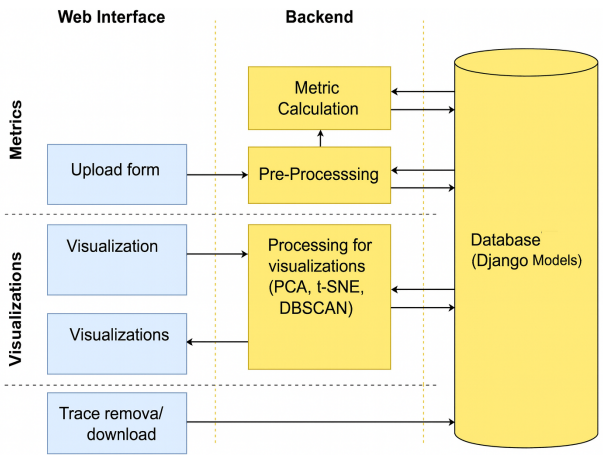


Fig. 1: MobMetrics Operating Flow

Caption Note: The architecture follows a modular pipeline consisting of five main stages: (1) **Data Upload**, where mobility traces are submitted and validated; (2) **Preprocessing**, responsible for formatting data and identifying key structures such as *Stay Points*, *Travels*, and *Journeys*; (3) **Metric Computation**, which applies spatial, temporal, kinematic, and social analysis methods; (4) **Persistent Storage**, managed through Django's *Models* to ensure data integrity and reusability; and (5) **Visualization and Analysis**, providing interactive, exportable representations of the results.

MobMetrics is organized as shown in Figure 1. Due to space limitations, we present the two main parts of *MobMetrics* in greater depth. Thus, Sections IV-A and IV-B address the interface level, indicating what the user needs to run *MobMetrics* and which visualizations are obtained as a result. Section V presents an exploratory study through a reproducible use case.

A. System Configuration

MobMetrics requires configuration of certain parameters to operate correctly. The parameters are presented below.

*1) **Upload Form**:* Figure 2 illustrates the submission form and the configuration parameters of *MobMetrics*.

- *Trace File*: a real or synthetic trace file input to be analyzed [18], [19], in *.csv* format, containing columns *id*, *x*, *y*, *z*, and *time*. *id* refers to a unique identifier associated with the record. Parameters *x*, *y*, and *z* indicate a geographic or Cartesian coordinate for the record. *time* refers to a timestamp, which may be in *DateTime* format. If the *id* and *z* columns are absent, identifier 1 is automatically assigned to the entire trace (assuming a single entity), and the value 0 is assigned to the *z* coordinate.
- *Name*: a user-defined name to save the trace in the database, allowing it to be differentiated from other traces.
- *Label*: the type of entity represented in the trace (e.g., car, person, or phone).
- *Geographical Coordinates*: must be checked if the coordinates are geographic (latitude and longitude). In this case, distance calculations consider Earth's curvature (the *Haversine* formula [20]); otherwise, Euclidean distance [21] is used.

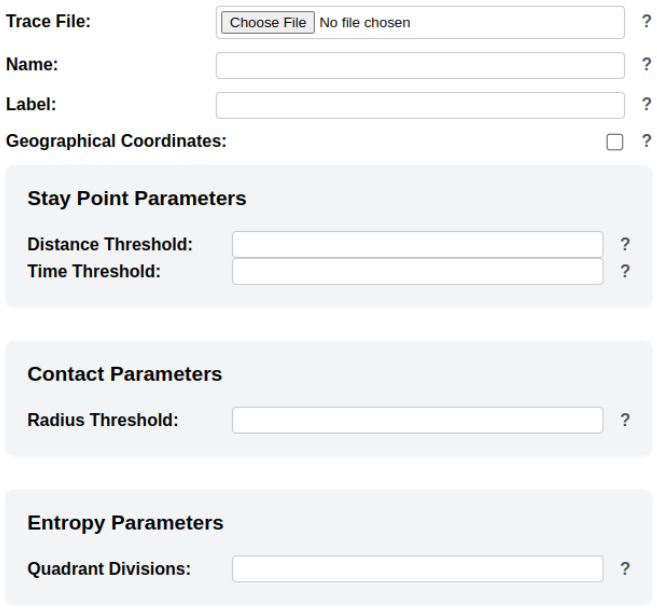


Fig. 2: MobMetrics graphical interface containing the submission form for the mobility trace to be analyzed

It is worth noting that this is a non-exhaustive list of parameters. Other metrics described in Section III also require specific parameterization for correct execution.

*2) **Preprocessing**:* This stage formats the data and computes certain metrics:

- **Stay Points**: This metric is among the first execution steps in *MobMetrics*, as its result is used transversally by other metrics throughout processing. *MobMetrics* is careful to avoid repeated *Stay Points*; thus, when a new *Stay Point* is found, the system searches the database for another *Stay Point* that is sufficiently close and refers to the same mobility trace. If such a point exists, i.e., the distance between their centroids is less than or equal to D_{ths} both are considered to represent the same location, and the "new" *Stay Point* is recorded as a *Visit* to the preexisting point.
- **Travels** and **Journeys**: This conceptual distinction separates two levels of displacement segmentation, computed during preprocessing. *Travels* represent the complete path traveled by an entity, from the first to the last recorded point. *Journeys* correspond to the segments between two consecutive *Stay Points*, enabling a more detailed analysis of displacements between dwell locations.
  Thus, if the start of an entity's trace is not associated with a *Stay Point* and/or the end is not either, *MobMetrics* considers as a *Journey* the segment between the starting point and the first *Stay Point* and/or between the last *Stay Point* and the final point of the trace.
- **Trajectory Center**: Computed during preprocessing, this corresponds to the centroid obtained from all points in an entity's trace. This center is used as a reference in the calculation of the *Radius of Gyration*, serving as a basis for measuring the spatial dispersion of displacement.
- **Mobility Profile**: Computed during preprocessing, the mobility profile corresponds to a normalized average, between 0 and 1, of the metrics extracted for a trace. This value compactly summarizes the entity's overall behavior throughout the displacement.

*3) **Metric Calculation**:* In this phase, all metrics described in detail in Section III are computed based on the preprocessed data and the parameters provided by the user. The results of these metrics are then stored in persistent data structures via the *Django* framework's *Models*, ensuring integrity, reuse, and availability for subsequent visualizations or additional analyses within the system.

B. Visualizations

To enable trace analysis, *MobMetrics* provides visualizations. The visual elements are organized into different categories according to the nature of the presented data.

- **Raw Trace Visualizations**: These visualizations display the entities' traces individually, as well as in comparison with other entities present in the same file, when applicable.
  The main goal is to allow visual inspection of displacements and *Stay Points* over time.
- **Metric Visualizations**: These visualizations focus on the quantitative analysis of traces based on the computed metrics. They can be used to compare behaviors among entities, identify statistical patterns, and assess deviations relative to references.
- **Social Visualizations**: This group includes visualizations aimed at analyzing interactions among entities, enabling investigation of contact, proximity, and collective behavior over time and space.

- **Comparative Visualizations**: Comparative visualizations employ dimensionality reduction and clustering techniques to represent relationships among entities based on multiple metrics. They are especially useful for exploring data structure and detecting non-trivial patterns.

V. Use Case

In general, analyzing datasets requires initial processing followed by evaluation through indicators and charts. Accordingly, we present a use case of *MobMetrics*, applying the tool to traces from three different sources: (i) a sample of *Microsoft GeoLife* [22] classified by movement type (car, taxi, bus, and walking); (ii) a collection of public GPS data gathered from *OpenStreetMap*, also classified by type of movement; (iii) a sample of the bus dataset for the city of Rio de Janeiro obtained from *Kaggle*. Therefore, the tool is applied to obtain insights on vehicular data (both private cars and taxis), public transport data (buses), and pedestrian data in urban centers.

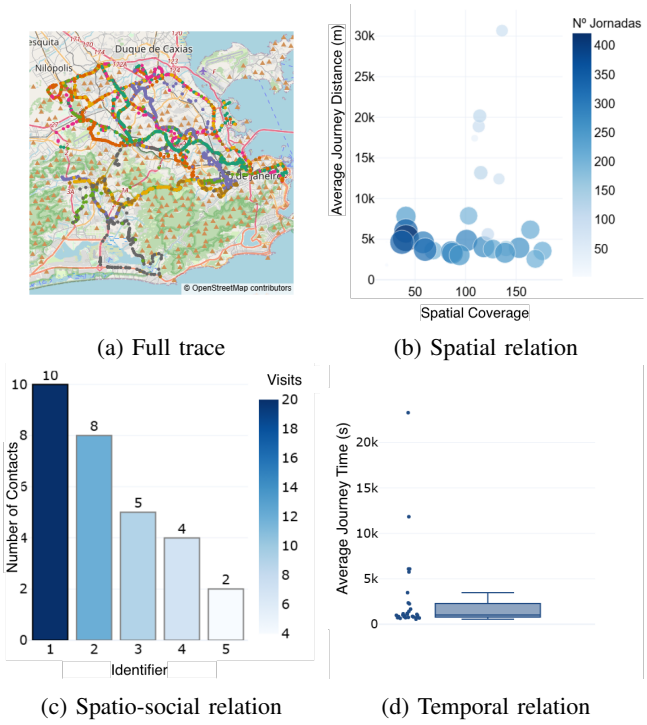
All traces correspond to different locations; however, differences in the original dimensionality of each dataset required sampling such that the amount of information would be comparable, i.e., a similar number of points across datasets. To this end, each dataset spans a two-week interval and reflects the movement of 20 distinct entities, as well as a similar temporal sampling grain.

Table III presents the values used for the configuration parameters mentioned in Section IV-A. Traces of the same nature were grouped under identical parameters, except for the Rio de Janeiro bus dataset. Thus, in addition to balanced sampling, the measurement extraction settings were also aligned to highlight the inherent differences in entity movements. Based on this, *MobMetrics* was used to analyze the traces from two different perspectives: (i) an overview of the different bus fleets present in the Rio de Janeiro dataset; and (ii) a comparative and classificatory view of different traces according to the intrinsic characteristics of each mode of transportation.

A. Overview of RJBus

To provide a general overview of the Rio de Janeiro bus dataset, Figure 3 presents a sequence of plots generated from carefully chosen axes. In Figure 3a, we show a view of the Rio de Janeiro map with all bus line entities distributed across their routes, where each line is represented by a distinct color. This is the first view generated after metric computations, and placing it at the beginning of the results *dashboard* allows users to gain an overall sense of how entities are distributed on the map. It can also be used as a geographical reference for numerical results obtained from metrics and purely analytical charts.

Figure 3b presents a descriptive view of the traces, combining a spatial visualization with metrics extracted for each entity (bus). The subfigure showing the three-dimensional scatterplot (Spatial Cover × Average Journey Distance × Number of Journeys) reveals relevant patterns in the operational behavior of monitored buses. There is a significant concentration of entities with short average journeys (between 2,000 and 8,000



(a) Full trace (b) Spatial relation

(c) Spatio-social relation (d) Temporal relation

Fig. 3: Overview of the *RJBus* trace.

meters), suggesting the predominance of shorter lines, possibly local routes or feeder services to terminals. Regarding Spatial Cover, although the distribution is relatively uniform along the axis, there is a slight concentration of buses with smaller territorial reach, which may reflect geographic or operational constraints. As for the Number of Journeys, vehicles with higher journey counts tend to operate on routes with lower spatial coverage, indicating more intensive use of short lines in central or high-demand areas. Conversely, entities with greater spatial coverage generally performed fewer journeys, indicating long-distance or intermunicipal routes that run less frequently throughout the day. We also note the presence of outliers with average distances above 15,000 meters, and one extreme case around 30,000 meters, which nevertheless performed very few journeys reinforcing the hypothesis of atypical routes, operational tests, or special lines with low recurrence.

In Figure 3c, we focus on a small group of bus lines selected manually via certain configuration parameters in *MobMetrics* (specific to the *analytics* section), according to the criterion of the highest number of visited *stay points*. This visualization is constructed such that the y-axis shows the total number of contacts made by the bus, distributed along the x-dimension. In addition, the color axis indicates how many times each line visited a point of interest on the map. The insight here aligns with expected behavior: the more often an entity passes by a *stay point*, the higher the chances of contacts, especially if we consider that *stay points* may represent certain integration stations (line interchanges) in the city. Thus, the bar chart indicates a directly proportional relationship between the

TABLE III: Parameter values used in the case study.

Parameters	GL and OSM (Walking)	GL and OSM (Cars)	GL and OSM (Buses)	RJBus (Buses)
D_{ths}	20 m	25 m	30 m	35 m
T_{ths}	160 s	180 s	180 s	180 s
CR_{ths}	18 m	24 m	30 m	34 m
Q_{div}	20 m	20 m	20 m	20 m

number of contacts and the number of visits a line makes to a detected point of interest.

Finally, Figure 3d presents a boxplot of the Average Journey Time (in seconds) for each entity in the RJBus trace sample. The observed distribution shows strong positive skewness, with most buses operating at average journeys up to 3,000 seconds (50 minutes), suggesting a predominance of short- to medium-duration routes in the analyzed set. The median, around 650 seconds (approximately 11 minutes), indicates that at least half of the entities undertook notably short trips, consistent with movements in dense urban areas or circular routes. The lower portion of the boxplot is quite short, reflecting low variability among the shortest average times, while the upper portion extends to values near 4,000 seconds, suggesting greater dispersion among entities with longer journeys. Identified outliers, especially those exceeding 4,000, 6,000, and even 12,000 seconds, indicate the presence of atypical lines possibly long-distance, intermunicipal routes, or lines with low operating frequency. Although minority cases, they highlight operational diversity and reinforce the importance of considering the full distribution when interpreting aggregated metrics such as the average journey time.

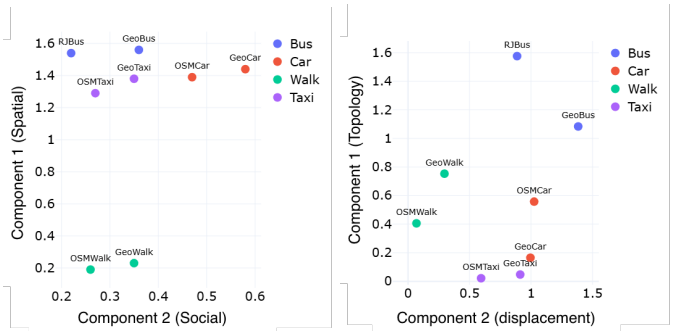
B. Comparative and Classificatory View

We now present a comparative and classificatory evaluation of the traces mentioned at the beginning of this section. The goal is to cluster traces based on modes of transportation, leveraging the fundamental properties of each movement (highlighted via the metrics). Accordingly, in Figures 4a and 4b, we show two scatterplots generated for a collection of different sets of traces, named to indicate both the source dataset and the type of movement. Additionally, we implemented a feature in the tool to color points according to the mode of transport specified in the dataset *labels*. Based on this, the methodology used to compose these plots was as follows: from the *MobMetrics frontend*, two metric sets were selected to construct each of the two components present in the plot (we varied these components from Figure 4a to Figure 4b).

Figure 4a highlights distinct mobility patterns by combining spatial and social metrics for different movement types. On the y-axis, by aggregating metrics related to distance traveled and the average radius of gyration, there is a clear separation between vehicular traces (cars, taxis, and buses) and pedestrian traces. This separation indicates that vehicles generally cover significantly longer paths and exhibit less angular movement, reflecting continuous displacements along structured roadways. In contrast, walking traces tend to cover shorter distances with greater directional variability, resulting in lower values on this component. The x-axis, composed of social metrics such as the number and total duration of contacts, reveals subtler nuances. There is a slight separation between private car traces and other vehicles, suggesting that cars tend to share more time in proximity with entities of the same type, possibly due to congestion or overlapping routes. Buses and taxis cluster more strongly on this axis, indicating more homogeneous contact patterns. Interestingly, walking traces lie close to buses and taxis in terms of social interaction, revealing that, despite spatial differences, pedestrians share mobility environments with similar levels of contact or temporal proximity to other entities.

In Figure 4b, combining topological (y-axis) and movement (x-axis) components offers a new perspective on differentiating traces. On the y-axis, by gathering metrics related to the topological structure of the routes, such as the number of *stay points* and quadrant entropy, bus traces stand out with the highest values, occupying the top area of the plot. This suggests greater spatial diversity and presence at well-defined stopping points, typical characteristics of public transit lines. Car and taxi traces appear grouped lower, with moderate topological values, reflecting continuous and less structured routes. Walking traces lie between these two categories, slightly above private vehicles, indicating that although pedestrian movements explore different areas of the urban space, they tend to generate less complex topological patterns than bus routes. On the x-axis, by aggregating movement-related metrics such as average speed, average journey time, and spatial cover, the separation among transport modes is more pronounced. Walking traces occupy the leftmost side with the lowest values, reflecting slower, shorter, and more localized movements. Next are buses with intermediate values, and finally private vehicles (cars and taxis) at the far right, indicating higher speed, spatial reach, and average journey time. This arrangement suggests that the metrics selected for this visualization were effective in capturing both the structure and dynamics of different transport modes.

Figure 5 demonstrates *MobMetrics*' capability to perform clustering automatically, without user intervention in component selection. For this, we fixed a subset of metrics preselected for their discriminative potential across transportation types, and then applied the *K-Means* algorithm to these data. The two main automatically extracted components were used to generate the scatterplot, in which points were colored based on the clusters identified by the algorithm itself. The resulting plot shows the effectiveness of the approach: traces were mostly classified into three distinct groups corresponding to buses, walking, and private vehicles (cars and taxis).



(a) Spatial vs. social relation (b) Topology vs. movement

Fig. 4: Comparative view.

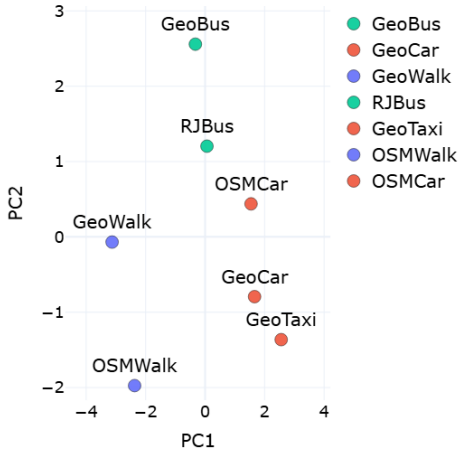


Fig. 5: Classification by K-Means

VI. Conclusion and Future Work

In this work, we presented *MobMetrics*, an open-source tool for processing and analyzing mobility traces. The proposal addresses the lack of unified and extensible solutions for extracting spatial and temporal metrics from location-based data. *MobMetrics* stands out for its modular architecture, which supports both the direct use of pre-implemented metrics and the inclusion of new ones through an extensible and well-documented model.

The system already encompasses a wide range of fundamental metrics for mobility studies, including speed, distance, direction, stop points, and inter-entity contacts. In addition, it provides an interactive visualization interface that facilitates data exploration and interpretation of results.

As future work, we highlight three main development directions: support for multiple input formats, the addition of new metrics, and the introduction of new visualization capabilities.

Acknowledgment

We would like to thank the National Council for Scientific and Technological Development (CNPq), the São Paulo Research Foundation (FAPESP), the Brazilian Internet Steering Committee (CGI.br), the Espírito Santo Research and Innovation Support Foundation (FAPES), the Bahia Research Support Foundation (FAPESB), and the German Federal Armed Forces (BAAINBw and WTD 81) for the grants 444077/2024-3, 2018/23097-3, 2020/05182-3, TIC 002/2015.

References

[1] P. Zißner, P. H. Rettore, B. P. Santos, R. R. F. Lopes, and P. Sevenich, "Road traffic density estimation based on heterogeneous data fusion," in *2022 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 2022, pp. 1–6.

[2] I. Ullah, F. Ali, H. Khan, F. Khan, and X. Bai, "Ubiquitous computation in internet of vehicles for human-centric transport systems," *Computers in Human Behavior*, vol. 161, p. 108394, 2024.

[3] C. Prajisha and A. Vasudevan, "MSecTrust: A Mobility-Aware Secure Trust-Based Routing Protocol for RPL Based Internet of Things," *Journal of Network and Systems Management*, vol. 33, no. 2, p. 40, 2025.

[4] P. H. Rettore, J. Mast, T. Aurisch, A. C. Viana, P. Sevenich, and B. P. Santos, "Military IoT from Management to Perception: Challenges and Opportunities Across Layers," *IEEE Internet of Things Magazine*, vol. 8, no. 2, pp. 25–31, 2025.

[5] P. K. D. Pramanik, S. Pal, and P. Choudhury, "Mobile crowd computing: potential, architecture, requirements, challenges, and applications," *The Journal of Supercomputing*, vol. 80, no. 2, pp. 2223–2318, 2024.

[6] D. S. Foundation, "Django web framework," https://www.djangoproject.com/, 2024, acessado em mai. 2025.

[7] Y.-A. De Montjoye, L. Rocher, and A. S. Pentland, "bandicoot: A python toolbox for mobile phone metadata," *Journal of Machine Learning Research*, vol. 17, no. 175, pp. 1–5, 2016.

[8] N. Aschenbruck, R. Ernst, E. Gerhards-Padilla, and M. Schwamborn, "Bonnmotion: a mobility scenario generation and analysis tool," in *Proceedings of the 3rd international ICST conference on simulation tools and techniques*, 2010, pp. 1–10.

[9] L. N. Silva, P. H. Rettore, V. F. Mota, and B. P. Santos, "Mobvis: A framework for analysis and visualization of mobility traces," in *2022 IEEE Symposium on Computers and Communications (ISCC)*, 2022, pp. 1–6.

[10] 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 Modeling, Analysis and Simulation of Wireless and Mobile Systems*, 2018, pp. 281–288.

[11] A. Graser, "Movingpandas: efficient structures for movement data in python," *GIForum*, vol. 1, pp. 54–68, 2019.

[12] L. Pappalardo, F. Simini, G. Barlacchi, and R. Pellungrini, "Scikit-mobility: A python library for the analysis, generation, and risk assessment of mobility data," *Journal of Statistical Software*, vol. 103, pp. 1–38, 2022.

[13] H. Martin, Y. Hong, N. Wiedemann, D. Bucher, and M. Raubal, "Trackintel: An open-source python library for human mobility analysis," *Computers, Environment and Urban Systems*, vol. 101, p. 101938, 2023.

[14] E. R. Cavalcanti and M. A. Spohn, "Aplicando métricas de mobilidade na classificaçao de rastros de movimento em manets."

[15] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from gps trajectories," in *Proceedings of the 18th international conference on World wide web*, 2009, pp. 791–800.

[16] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi, "Understanding individual human mobility patterns," *nature*, vol. 453, no. 7196, pp. 779–782, 2008.

[17] D. Hristova, M. J. Williams, M. Musolesi, P. Panzarasa, and C. Mascolo, "Measuring urban social diversity using interconnected geo-social networks," in *Proceedings of the 25th international conference on world wide web*, 2016, pp. 21–30.

[18] "Towards a Real Vehicular Data Cloud."

[19] T. E. Alves, P. H. Rettore, and B. P. Santos, "A mobility model of the internet of things," in *Proceedings of the 29th Brazilian Symposium on Multimedia and the Web*, 2023, pp. 221–229.

[20] M. T. Scripts, "Haversine formula," https://www.movable-type.co.uk/scripts/latlong.html, 2002, acessado em 15 de maio de 2025.

[21] H. Anton, I. C. Bivens, and S. Davis, *Cálculo com Geometria Analítica*, 9th ed. São Paulo: LTC, 2013.

[22] Y. Zheng, X. Xie, and W.-Y. Ma, "Geolife: A collaborative social networking service among user, location and trajectory," *IEEE Data Eng. Bull.*, vol. 33, no. 2, pp. 32–39, 2010.