# A Delay-Aware Coverage Metric for Bus-Based Sensor Networks

Pedro Cruz[a,b], Rodrigo S. Couto[a], Luís Henrique M. K. Costa[a], Anne Fladenmuller[b], Marcelo Dias de Amorim[b]

[a] *GTA/PEE/COPPE – Universidade Federal do Rio de Janeiro*
[b] *LIP6/CNRS – Sorbonne Université*

**Abstract**

Embedding sensors in urban buses is a promising strategy to deploy city-wide wireless sensor networks. By taking advantage of the mobility of buses, it is possible to achieve extended spatial coverage with fewer sensors as compared to a static setup. The trade-offs are that urban buses only cover part of the city, and that the frequency of the buses, and consequently of the data collection, is inhomogeneous across the city. Depending on the communication technology, buses may be unable to deliver the collected data on time. In this paper, we propose a coverage metric that takes into account the delivery delays of sensed data and the frequency at which a given region is sensed. The metric indicates, for a given time window, the proportion of streets that can be sensed under the requirements of an application. We apply the metric to more than 19 million GPS coordinates of 5,706 buses in Rio de Janeiro, mapping the coverage achieved for different application needs during a week. We build an abacus relating the coverage to different application requirements. We also calculate the coverage of a scenario only with static sensors. We show, among several observations, that bus-based sensing increases the coverage of applications by up to 13.2 times, in the worst case analyzed, when compared to a pure static scenario.

*Keywords:* Vehicular sensing, Internet of Things, Wireless Sensor Networks

*Corresponding author

*Email addresses:* `cruz@gta.ufrj.br` (Pedro Cruz), `rodrigo@gta.ufrj.br` (Rodrigo S. Couto), `luish@gta.ufrj.br` (Luís Henrique M. K. Costa), `anne.fladenmuller@lip6.fr` (Anne Fladenmuller), `marcelo.amorim@lip6.fr` (Marcelo Dias de Amorim)

## 1. Introduction

A clear advantage of mobile wireless sensor networks (MWSN) is that they can use mobility to expand the coverage area of a smart city [1]. When compared to static sensors, the deployment of mobile sensors involves two main challenges. Firstly, communication links between nodes and gateways are intermittent. Therefore, sensors must store sensed data until there is a connection to a gateway, delaying the transmission and delivery of sensed data. The second challenge concerns the measurement frequency. As zones of the target area are covered only when a mobile sensor passes nearby, intervals between readings are inherently heterogeneous.

The communication delay and the measurement frequency determine whether a specific application may run on top of the MWSN. In this paper, we consider the case where mobile sensors are embedded in urban buses and gateways are located at bus stops. The delivery delay depends on the itinerary of bus lines, the traffic conditions, and the location of bus stops.

Previous studies have considered the coverage of MWSNs. Liu *et al.* [1] and Ekici *et al.* [2] investigate the effects of mobility on coverage and on data delivery. Mosaic [3, 4, 5] and Opensense [6] projects show the feasibility of such networks for air quality applications, using machine learning techniques to predict missing data. The works from Ali *et al.* and Cruz *et al.* [7, 8] propose a coverage metric for bus-based WSNs, but do not take into account the delays generated by the mobile coverage and transmission. The work from Zhao *et al.* [9, 10] characterizes opportunistic coverage, proposing a metric called Inter-Cover Time, but does not take into account delays in data delivery.

We propose a coverage metric that takes into account the delivery delay and the measurement frequency of a given street section. Since applications have different requirements of data delivery delay and measurement frequency, the metric considers that buses can cover a street section if and only if they gather data with a *minimum measurement frequency* and deliver the gathered data

with a *maximum delivery delay*. We apply the metric to real mobility traces of buses from the city of Rio de Janeiro. We present the results in the form of an abacus, relating coverage to different tolerances in terms of delivery delay and measurement frequency. We map the coverage obtained by buses to the applications of waste management, air quality monitoring, and noise monitoring. We also compare the coverage obtained to the scenario where bus stops serve as gateways to static sensors. Taking the application of waste management as an example, our results show that buses can cover up to 25.5 times the region that it would cover with static sensors. This value corresponds to a coverage of 35.8% of the streets of Rio de Janeiro.

The main contributions of the paper can be summarized as:

- We propose a coverage metric that takes into account the delivery delay and measurement frequency required by applications.
- We measure the achievable coverage provided by a MWSN using bus traces collected in Rio de Janeiro.
- We compare the coverage obtained by a bus-based mobile WSN and a static WSN, for different applications.

This paper is organized as follows. Section 2 describes the scenario of bus-based urban sensing. In Section 3, we propose a coverage metric designed to deal with the delays and sensing frequencies of the studied scenario. In Section 4, we apply the metric to real mobility traces of buses of the city of Rio de Janeiro. In Section 5, we position this paper in relation to the scientific literature. Section 6 concludes the work and points out directions for future work.

## 2. Bus-based urban sensing: Goals and assumptions

There is a wide range of applications that can benefit from data gathered by buses. Section 5 lists existing approaches in bus-based urban sensing. These approaches serve applications of pothole detection, air quality monitoring, and

Table 1: Smart city applications and their data needs in terms of minimum measurement frequency and maximum tolerated delay.

| **Application** | **Measurement frequency** ($day^{-1}$) | **Tolerated delay (s)** | **Source** |
|---|---|---|---|
| Waste management | 24 | 1,800 | [12] |
| Air quality monitoring | 48 | 300 | [12] |
| Noise monitoring | 144 | 300 | [12] |
| Electricity meter | 96 | not defined | [13] |
| Gas meter | 96 | not defined | [13] |
| Temperature | 96 | not defined | [13] |
| Weather | 48 | not defined | [13] |

noise monitoring. Some of them also serve as data mules for sensors in buildings, electricity meters, smart lamps, smart trashcans, and others. For such applications to work properly, it is important that data meets certain requirements. In the case of a MWSN with delay-tolerant data delivery, the critical requirements are related to temporal adequacy of data [11]. In this context, temporal adequacy refers to the sensing frequency of each area, and the time it takes since a bus collects the data until data is available to applications and users. The survey conducted by Zanella *et al.* [12] identifies applications, the frequency of measurement needed, and the delay tolerated by those applications. Sinaeepourfard *et al.* [13] also provide the frequency of measurement needed for some smart city applications, but gives no information on delays. Table 1 lists applications and its data requirements, according to these studies.

We assume that sensors carried by buses gather data about the city and store them until a connection with a gateway is possible. We assume that all stored data can be delivered in a single connection. Gateways, located at bus stops, receive data when buses are within communication range, and use the Internet to send this data to a cloud server. The server processes the data and makes it

available to applications. We approximate the time that data remains stored in the sensing node by the time a bus takes to travel between two consecutive bus stops. The time for data to travel over the Internet is negligible when compared to the time buses take to travel between bus stops. In this regard, we consider that the delivery delay is the time since data was acquired until it is delivered to a gateway. The paths followed by buses are often cyclic, therefore, the same bus may visit the same location several times a day. In addition, bus lines have overlapping sections and a certain number of trips per day, meaning that different buses may pass by the same location several times a day. Therefore, we define the measurement frequency as the number of times a certain section is measured, considering that data is delivered on time.

As Table 1 shows, different applications have different requirements of measurement frequency and maximum delivery delay. We can notice that the application of waste management tolerates a delay up to 6 times the delay tolerated by the applications of air quality monitoring and noise monitoring. This is due to the fact that the conditions of waste management infrastructure change more slowly than the conditions of the air quality and the noise in a city.

In a bus-based MWSN with delay-tolerant data delivery, different regions of the city are not sensed with the same frequency. Additionally, their information suffers different delivery delays. This means that not all applications can benefit from data gathered in the same region of a city. Therefore, it is important to use a coverage metric to reflect these requirements. The next section proposes a coverage metric that is aware of delays and frequency measurements.

## 3. Delay-aware coverage metric

A common way of representing the road map of a city consists of modelling it as a directed graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$. The set $\mathcal{V}$ contains the vertices of $\mathcal{G}$. Each vertex $x_1, \ldots, x_{|\mathcal{V}|} \in \mathcal{V}$ represents an intersection, a curve or other point of interest in the topology of the streets. The set $\mathcal{E}$ contains the edges of $\mathcal{G}$. An edge $(x_i, x_j) \in \mathcal{E}$ exists if and only if it is possible to follow a street from $x_i$ to

Table 2: Notations used in this work.

| Notation | Description | Type |
|---|---|---|
| $\mathcal{E}$ | Street segments of the city | Set |
| $\mathcal{V}$ | Vertices of the map of the city | Set |
| $\mathcal{G}$ | Graph representing the road map of a city | Graph |
| $\mathcal{B}$ | Urban buses that serve a city | Set |
| $\mathcal{P}$ | The set containing all the paths of the buses of a city | Set |
| $(x_i, x_j)$ | The street segment that starts in vertex $x_i$ and ends in vertex $x_j$ | Parameter |
| $l_{(x_i,x_j)}$ | The length of street segment $(x_i, x_j)$ | Parameter |
| $P_b$ | The path $((x_i, x_j), (x_j, x_k), (x_k, x_l) \ldots)$ of bus $b$, in terms of edges in $\mathcal{E}$ | Sequence |
| $P_b[m]$ | The $m^{th}$ edge visited by bus $b \in \mathcal{B}$ in its path $P_b$ | Parameter |
| $D_{\max}$ | The maximum tolerated delivery delay for the considered application | Parameter |
| $F_{\min}$ | The minimum measurement frequency required by the considered application | Parameter |
| $v^T_{(x_i,x_j)}$ | The number of visits received by street segment $(x_i, x_j)$ on a given time interval | Parameter |
| $F_{(x_i,x_j)}$ | The frequency at which the street segment $(x_i, x_j)$ is visited | Parameter |
| $\mathcal{E}_c$ | The subset of $\mathcal{E}$ that is covered for the considered application | Parameter |

$x_j$, not visiting any other vertex. A weight $l_{(x_i,x_j)}$ is associated to each edge $(x_i, x_j)$, representing the distance between $x_i$ and $x_j$. Since a street is an ordered collection of points of interest, a street is a sequence of vertices. A vertex can be part of more than one street, when this vertex is an intersection between two or more streets. The notations used in this paper can be found in Table 2.

In this paper, we use the edges of $\mathcal{G}$ to represent the street segments of the city. The street segments are used as an atomic unit of the measured region. The concept of a street segment is used in our previous work [8], while similar concepts are used by Ali and Dyo [7] and by the project Opensense [6].

### *3.1. Opportunistic sensing and data delivery*

Each bus in set $\mathcal{B}$ follows a fixed path through the street segments of the city. The path $P_b$ of a bus $b$ can be represented as a sequence of the edges in the graph $\mathcal{G}$, $((x_i, x_j), (x_j, x_k), (x_k, x_l) \ldots)$, where $P_b[n]$ is the $n^{th}$ edge reached by $b$. While passing through a street segment, a bus gathers data and stores it

until the bus reaches the next gateway. When a connection is achieved, the bus delivers the data. Figure 1 illustrates this situation[1]. The path of bus $b \in \mathcal{B}$ is a sequence of vertices that includes street segment $(x_j, x_k) \in \mathcal{E}$. Later, when passing by the vertex $x_l \in \mathcal{V}$, $b$ delivers data to gateway $g_2$. The time elapsed between the moment $b$ reaches $x_j$ and $x_l$ represents the delivery delay suffered by the data collected by $b$ in $(x_j, x_k)$.

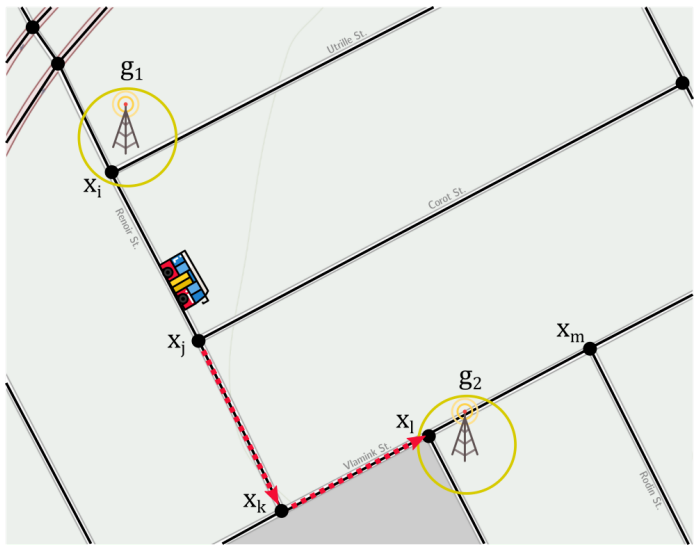



Figure 1: Path of bus $b$ through the segment $(x_j, x_k)$ and subsequent delivery in $x_l$.

Several variables might influence the delivery delay of data collected in a street segment $(x_i, x_j)$. Among them, the traffic conditions, the number of times a bus stops to serve passengers, and the distance between $(x_i, x_j)$ and the gateway where data from $(x_i, x_j)$ is delivered. As shown in Table 1, different applications can tolerate different data delivery delays. This means that, depending on the delivery delay, data collected on section $(x_i, x_j)$ may be useful or not for a given application. Since different applications may have different tolerances to delay, the same data may be useful to some applications and not

[1]Shuttle icon made by Skyclick and antenna icon made by Freepik from `https://www.flaticon.com/`. Licensed by Creative Commons BY 3.0

for others.

Buses might sense a street segment $(x_i, x_j)$ several times during a given interval of time $T$. This can happen either because $(x_i, x_j)$ is in the path of more than one bus but also because some buses can pass over the same street segment several times during $T$. In this case, it is possible to say that there is a certain measurement frequency of street segment $(x_i, x_j)$. As shown in Table 1, applications might need a certain measurement frequency to provide a reasonable service to its users.

### *3.2. A delay-aware coverage metric*

It is important that a coverage metric reflects application requirements in terms of delays and measurement frequency. To define the coverage for a given application, we use its maximum tolerated delay, $D_{\max}$, and its minimum tolerated measurement frequency, $F_{\min}$. We say that a bus $b \in \mathcal{B}$ has visited street segment $(x_i, x_j) \in \mathcal{E}$ if, after passing by $(x_i, x_j)$ and collecting data, $b$ is able to deliver the data before $D_{\max}$. We can define the visiting frequency $F_{(x_i,x_j)}$ of $(x_i, x_j)$ as the number of times $(x_i, x_j)$ has been visited in a given period $T$. A street segment $(x_i, x_j)$ is *covered* if and only if its visiting frequency is greater than or equal to the minimum visiting frequency $F_{\min}$ required by the target application. It is possible, then, to define $\mathcal{E}_c$ as the subset of $\mathcal{E}$ containing all the covered street sections, which is evaluated in terms of $D_{\max}$ and $F_{\min}$.

Algorithm 1 formalizes the construction of $\mathcal{E}_c$. The algorithm receives as inputs the set $\mathcal{B}$, the set $\mathcal{P}$, and the set $\mathcal{E}$. Set $\mathcal{B}$ contains all the buses that serve the considered city. The array **visits_counters** is an array indexed by section. In the while loop starting at Line 4, the visits of each section are accumulated in **visits_counters**. The function *get_delivery_delay* receives as input a path and an index $m$ in the path. The function returns the delivery delay for the $i^{th}$ street section of the path. The for loop starting in Line 10 adds to the subset $\mathcal{E}_c$ the sections that were visited with at least the minimum frequency of measurement $F_{\min}$. The algorithm returns the subset $\mathcal{E}_c$, containing the street sections that are covered for the considered application.

**Algorithm 1** Algorithm to construct the subset $\mathcal{E}_c$

**Require:** $\mathcal{B} = \{b_1, \ldots, b_n\}, \mathcal{P} = \{P_{b1}, \ldots, P_{bn}\}, \mathcal{E} = \{(x_i, x_j), \ldots\}$

**visits_counters** $\leftarrow$ **0**
**for** $P_{bi} \in \mathcal{P}$ **do** ▷ For the bus path of every bus
    m $\leftarrow$ 1
    **while** $m \leq |P_{bi}|$ **do** ▷ For each street section along the path of $b_i$
        $delivery\_delay \leftarrow get\_delivery\_delay(P_{bi}, m)$
        **if** $delivery\_delay \leq D_{\max}$ **then**
            **visits_counters**$[P_{bi}[m]] \leftarrow$ **visits_counters**$[P_{bi}[m]] + 1$ ▷ Count the number of visits for each section
        i $\leftarrow$ i+1
$\mathcal{E}_c \leftarrow \emptyset$
**for** $section \in \mathcal{E}$ **do** ▷ Verify the measurement frequency for each section
    $measurement\_frequency \leftarrow$ **visits_counters**$[section]/T$
    **if** $measurement\_frequency \geq F_{\min}$ **then**
        $\mathcal{E}_c \leftarrow \mathcal{E}_c \cup section$
**return** $\mathcal{E}_c$

Given the reasoning above and the construction of $\mathcal{E}_c$, Equation 1 defines the coverage $C$ of a city for a given application as the sum of the lengths of street segments that are visited within a minimum visiting frequency, normalized by the sum of the length of all the street segments of the city. In Equation 1, $\mathcal{E}_c$ is the set of covered street segments and $L$ is the sum of street segments lengths.

$$C = \sum_{(x_i,x_j)\in\mathcal{E}_c} \frac{l_{(x_i,x_j)}}{L}. \tag{1}$$

## 4. Experimental analysis

To show the feasibility of the proposal, we apply the metric to a hypothetical network, derived from real mobility traces of the buses of the city of Rio de Janeiro. Initially, we split the streets of Rio de Janeiro into street segments. After that, we collect traces consisting of GPS coordinates of buses, refreshed every minute. To calculate the coverage using the traces, we must filter the traces and adjust them to the topology of the street segments. To obtain the delivery delay of each measurement, we must access the times when each bus can make contact with a gateway and, therefore, can deliver data. Finally, we

can compute the coverage of the city, for different application requirements.

### *4.1. Data collection and processing*

To divide the streets of Rio de Janeiro into street segments, we use a map provided by OpenStreetMap [14]. The area of Rio de Janeiro is selected from the map, using a square delimited by the coordinates (-23.07,-43.7) and (-22.78,-43.16). The map is a graph that uses the same model described in Section 3. Therefore, each edge of the graph is a street segment. The sum of all street segment lengths in the map is 13,852 km.

The city administration of Rio de Janeiro provides bus mobility traces through an Application Programming Interface (API) [15]. The API offers a list containing the GPS coordinates of each bus, refreshed every minute. Each element of this list is a tuple containing the bus identification, its coordinates, and a timestamp of the moment when the coordinates are obtained. We have collected 29,155,221 GPS positions of 5,706 buses during the week between November, $5^{\text{th}}$, 00:00, and November, $11^{\text{th}}$, 23:59, in the year of 2,018. Since GPS coordinates are prone to errors, we eliminate inconsistent records. We consider that the records outside the square defined by the map and the records with timestamp out of the gathered period are inconsistent. Also, to reduce the cardinality of the dataset, we eliminate consecutive records of the same bus that differ from less than 10 meters. It is possible to discard these records because the difference between them is within the GPS error, i.e. 10 meters [16]. After these filters, there are 19,979,537 records. Table 3 shows the number of active buses for each day of the week. A bus is considered active if there is at least one entry for this bus in the day. It is possible to note that the largest difference in the number of buses is between Wednesday and Sunday. This difference is of 216 buses, which represents only 4% of the buses. Based on this result, the dataset is not divided into weekdays and weekends.

We also analyze the effect of the time of day when the buses are operating. To do so, we compute the cumulative distance traveled by buses at each moment of the day. Figure 2 shows the result of the total distance traveled as a function

Table 3: Number of active buses in the different weekdays.

| **Day of the week** | **Number of buses** |
|---|---|
| Monday | 5,540 |
| Tuesday | 5,546 |
| Wednesday | 5,573 |
| Thursday | 5,563 |
| Friday | 5,547 |
| Saturday | 5,435 |
| Sunday | 5,357 |

of the time of the day. It is possible to observe that between 0h and 4h, the distance traveled is close to zero. Therefore, we choose to eliminate this period of the day of our analysis, trusting that the buses are not capable of collecting data during this interval.

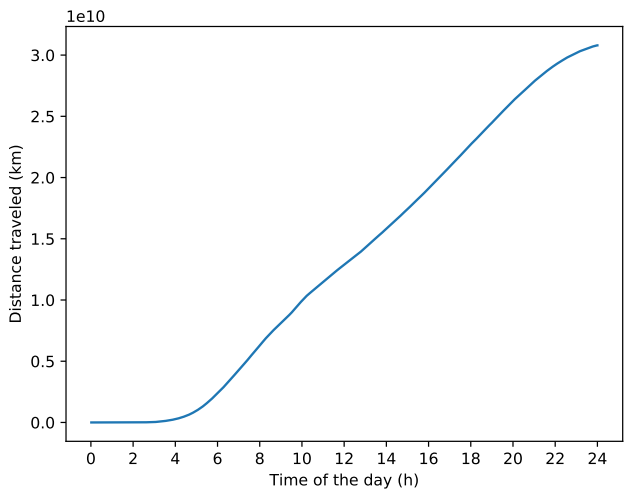
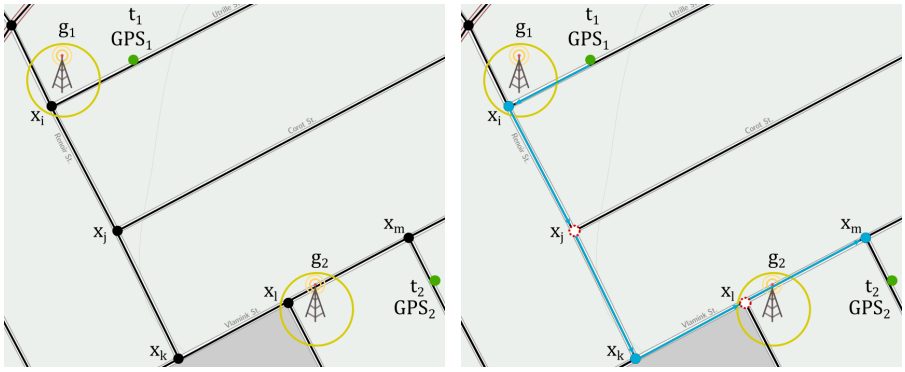



Figure 2: Cumulative distance traveled on each time of the day.

Bus traces have one GPS-coordinate sample per minute per bus. As illustrated in Figure 3(a), the GPS sampling rate does not allow the detection of all the street segments in the path of each bus. It is necessary to obtain the path of a bus in terms of all the edges in $\mathcal{G}$ that are part of the path of the bus, i.e.,



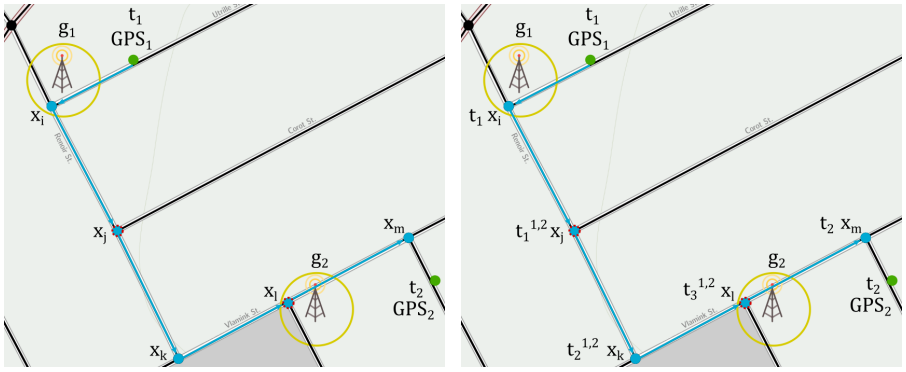

(a) Sequence of GPS coordinates from a single bus.

(b) Route returned by OSRM, as a function of map nodes relevant for drivers.

Figure 3: Transformation of GPS coordinates into bus paths using OSRM.

the street segments of the path. Therefore, we use the Open Source Routing Machine (OSRM) to obtain the most likely route followed by a bus between two consecutive positions in a path [17]. OSRM returns a route as a sequence of vertices of the map. Since OSRM is tuned for road navigation, the paths returned are not the complete sequence of vertices that a bus drives by in the map. The software omits vertices that are important to describe the topology of a street but not needed to describe a route. Figure 3(b) illustrates this situation. This means that two consecutive vertices in the paths returned might not share an edge in $\mathcal{G}$. This also means that it might be impossible to know the street segments that are part of a path. Using the graph that represents the map, we apply a shortest path algorithm to every two consecutive vertices in the routes generated by OSRM, as illustrated by Figure 4(a). This way, we obtain the paths of the buses as a list of all the vertices where the buses pass by. With this list, we can derive the street segments visited by each bus.

After obtaining the paths in terms of map vertices, we must also associate to every edge $(xi, x_j)$ in a path $P_b$ the instant when $b$ passes by this edge. Since we used every two consecutive GPS positions to generate a route with OSRM, we associate the timestamp $t_1$ of the first GPS position to the first vertex returned




(a) Route as a function of all the map nodes that the bus passes by.

(b) Interpolation of GPS timestamps for the map nodes that the bus passes by.

Figure 4: Transformation of paths generated by OSRM into sequences of street segments and subsequent inference of the time of sensing and data delivery.

by OSRM, $x_i$. Similarly, we associate the timestamp $t_2$ of the second GPS position to the last vertex returned by OSRM, $x_m$. We use interpolation to associate an instant to the other vertices, employing the time between vertices as weights. In other words, we assume the bus traveled at a constant speed between those two points. Figure 4(b) illustrates this interpolation, where $t_i^{1,2}$ is the $i^{th}$ instant estimated by the interpolation.

To determine the time at which a bus can deliver data, it is necessary to detect the instants when a bus is in contact with a gateway. Hence, we define that a bus can deliver data when it reaches a vertex that is within communication range of a bus stop. We also define that a node is within communication range of a bus stop when it is in a distance of 10 m or less from this bus stop. Therefore, the bus can deliver data when it is within communication range of at least one bus stop. After the processing, it is possible to build the set $\mathcal{E}_c$ and, finally, evaluate the coverage for different applications.

### *4.2. Coverage analysis*

With the data obtained in Section 4.1, we build an abacus of the network coverage. Figure 5 shows the coverage of Rio de Janeiro as a function of $F_{\min}$, for

different maximum delays $D_{\max}$ of 12 s, 120 s, 300 s, 600 s, 1,800 s, and 72,000 s. To calculate $F_{(x_i,x_j)}$, we count the number of visits received by $(x_i, x_j)$ and divide it by the period $T$. The period used to calculate the measurement frequency is of 20 h, since it is the period considered in the traces.

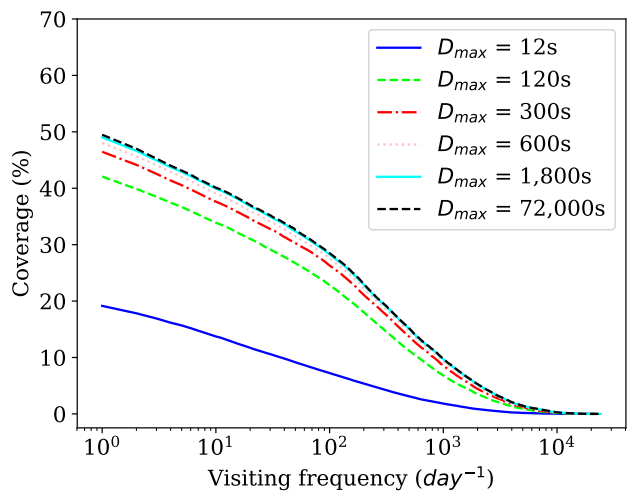



Figure 5: Abacus of the coverage of Rio de Janeiro in function of $F_{(x_i,x_j)}$, for different $D_{\max}$ over one week.

As shown in Table 1, a $D_{\min}$ of 300 s represents the applications of air quality and noise monitoring, while a $D_{\min}$ of 1,800 s represents an application of waste management. Figure 6 illustrates the coverage for the central region of Rio de Janeiro, for the applications of waste management, air quality monitoring, and noise monitoring. The streets in blue are the coverage of noise monitoring application. Since air quality monitoring application is less restrictive than noise monitoring, the coverage of this application is equal to the coverage of the noise monitoring application plus the street sections in green. The application of waste management is even less restrictive than the application of air quality monitoring. Its coverage is the coverage of air quality monitoring plus the street sections in red. The street sections in gray could not be covered. The area illustrated in Figure 6 has 22.26 km, representing about 1.86% of the total area of Rio de Janeiro.

Table 4: Coverage obtained by different Smart city applications.

| **Application** | **Coverage (%)** |
|---|---|
| Waste management | 36.2 |
| Air quality monitoring | 30.5 |
| Noise monitoring | 23.8 |

Other values of maximum delay are added to Figure 5 to represent applications that do not have a fixed delay value in the literature. It is possible to observe that the maximum coverage of the network is about 49% of the length of the streets of Rio de Janeiro.

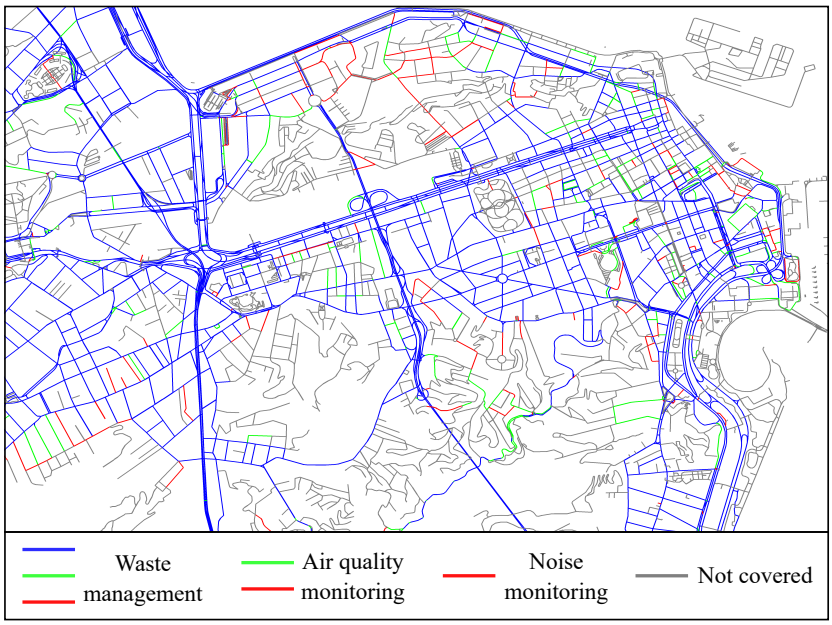



Figure 6: Coverage of the central region of Rio de Janeiro for different smart city applications.

Using the delay and frequency thresholds in Table 1, it is possible to define the coverage for different applications for smart cities. Table 4 shows the coverage for some of these applications.

### *4.3. Comparison with the static case*

The goal of using mobile sensors is to achieve better spatial coverage for certain applications. To quantify the coverage gain, we want to analyze the coverage obtained by leveraging buses mobility with the coverage obtained by a possible static scenario. In the case of bus-based mobility, sensors cover each street segment from its beginning to its end. This is not the case for static sensors. Therefore, it is important to use a method capable of considering the coverage of fractions of street segments. In this static scenario, bus stops are gateways and sensors are placed within their communication range. We consider as covered a fraction of street segment that is inside the communication range of at least one gateway. The total coverage is the sum of the lengths of all covered fractions.

To evaluate the static coverage, we treat each gateway as a circle of radius equal to the communication range. The union of such circles represents the area in the city where communication with at least one gateway is possible. We represent street segments as lines segments. The union of these line segments is the total road map of the city. In this metric, the total sensing coverage is the intersection between the area where communication with gateways is possible and the road map of the city. The idea behind it is that it is possible to place static sensors anywhere in the communication range of gateways. Figure 7 illustrates the coverage by static sensors placed in the communication range of gateways. The evaluations show a total static coverage of 1,7% of the total roads in the map. The static coverage obtained is equivalent to about 242 km of streets.

Figure 8 shows the gain of coverage obtained by the bus-based mobility of sensors over the course of a week. It is possible to notice that the coverage gain for an application of waste management ($D_{\max}$=1,800 s , $F_{\min}$=24 per day) is more than 20.5 times, while for the applications of air quality ($D_{\max}$=300 s, $F_{\min}$=48 per day) and noise monitoring ($D_{\max}$=300 s, $F_{\min}$=144 per day) are 17.2 and 13.2, respectively. These results show that the coverage of smart city applications can largely benefit from the mobility provided by buses.

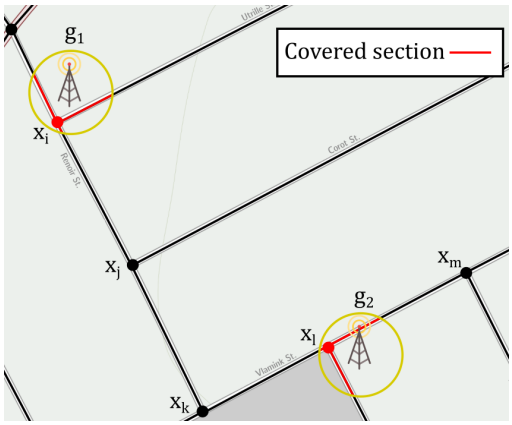



Figure 7: Example of covered streets by static sensors placed within communication range the gateways.

## 5. Related work

The literature presents some works in bus-based MWSNs and more generally, vehicle-based WSNs. It is possible to note that coverage is an important measure of the effectiveness of a WSN. There are still open challenges to the characterization of coverage, depending on the application.

### *5.1. Vehicle-based urban sensing*

Vehicle-based urban sensing is a relevant option to collect data for smart cities applications. The following works present different strategies to better adequate the data collection and delivery to the mobile scenario. The metric proposed in this paper could be used by these works to estimate the covered area by each one of them or as a comparison with each proposal and other alternatives.

The pioneer project BusNet [18] monitors road surface condition using sensors embarked in buses. Buses are used as data mules, taking raw data from a secondary station to another one until data reaches the main station. Once in the main station, the information about the road condition is retrieved and served to final users.

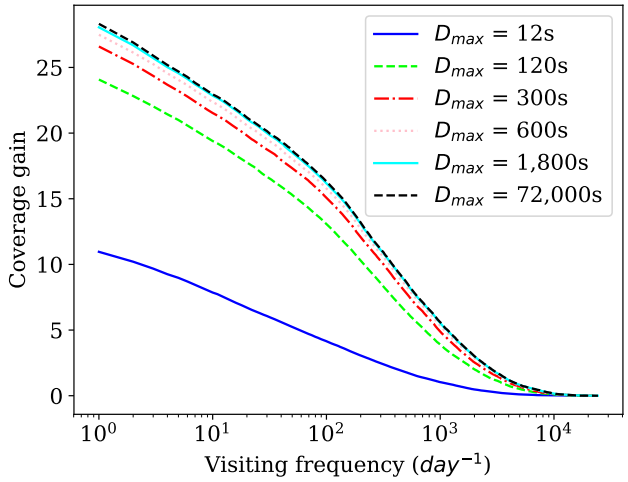



Figure 8: Coverage gain by buses of Rio de Janeiro in comparison to a static network, in function of $F_{(x_i,x_j)}$, for different $D_{\max}$ over one week.

Projects Mosaic [3, 4, 5] and Opensense [6] use urban buses to monitor the air quality of cities. Since pollution sensors are not reliable in the presence of mobility, Mosaic designs an algorithm to improve the accuracy of pollution sensors in a mobile scenario. Opensense proposes log-linear models to infer the air quality of uncovered regions. In Mosaic and Opensense, buses are connected all the time with the cloud, being able to send data with negligible delay.

Alsina *et al.* [19] design a bus-based WSN for noise monitoring. The paper evaluates the costs and equipment requirements to implement this network. They also propose strategies to build a noise map of the city canceling the noise from the bus carrying the sensors. The authors conclude that the application of noise monitoring can exploit the mobility of urban buses to improve its coverage, an assumption we share with them.

The work of Apte *et al.* and the work of Von Fischer *et al.* both explore the predictable mobility of Google Street View cars. Apte *et al.* embark sensors in the cars to measure the air quality in the city of Oakland [20], proposing methods to implement similar services in other cities. Similarly to this work, Apte *et al.* uses street sections of the city to define the regions of the city

that can be covered. Von Fischer*et al.* uses the improved coverage provided by Google Street View cars to detect gas leaks and rapidly communicate them to maintenance teams, avoiding accidents [21]. Von Fischer*et al.* infer the magnitude of the gas leaks, the measurements obtained by sensors and the speed of the wind are used by a prediction algorithm. In both cases,

SmartSantander uses vehicles as part of a multi-purpose WSN to gather data about the city of Santander [22]. Other urban objects are used to sense and actuate in the city, providing services to its citizens. The work presents a framework to integrate IoT in the smart city environment and deploys a city-wide network, showing the feasibility of this system and some applications that can benefit from it, such as environmental monitoring and smart parking. In the communication paradigm used, data is delivered immediately to the applications. This means that, for the coverage metric proposed in this paper, $D_{\max}$ is always respected.

SensingBus [23] uses a three-level architecture to collect, transmit and serve data on a smart city. Sensing nodes, embarked in buses, gather data from the city; fog nodes coupled to bus stops receive and pre-process raw data from buses, sending it over the Internet to the next level; the cloud level processes and serves data to users. Hence, SensingBus explores delay-tolerant communication. In another work, the authors propose a model for the delays and a selection algorithm to choose bus stops to place fog nodes minimizing the delivery delay respecting budget constraints [24]. The present work is in the same context of SensingBus, and has as main objective to discover the coverage provided by Sensingbus.

The works listed can apply the metric proposed in the present paper to define a notion of coverage for these works. The metric reveals which regions of the city can benefit from the applications offered by each of these systems.

### *5.2. Mobile wireless sensor networks coverage*

The coverage of MWSNs gives an important indication of the quality of data generated by these networks [11]. Mosaic and Opensense use different strategies

to study the coverage of their proposals. Mosaic divides the city using a grid and sets a score for every grid cell, depending on the number of routes that include the grid cell [5]. Additionally, Mosaic proposes an algorithm to select the best buses to receive sensing nodes and cover points of interest spread in the city. Opensense [6] divides the city into street segments and uses log-linear models to predict pollution data on segments that could not be directly measured. The results obtained by Opensense show that, for urban environments, the street segmentation is more appropriate than grid partitioning. The metric proposed in the present paper uses street segmentation and enables the estimation of the coverage of Opensense and Mosaic by taking account the number of measurements in each street segment. This makes possible to identify places where, even though there are a few measurements, they are not enough to serve as input to the applications.

Ali and Dyo propose a coverage metric for a bus-based WSN focused on road surface inspection [7]. Their proposed metric divides the city into street segments and considers a street segment as covered if at least one bus line passes through it. Ali and Dyo also propose a method to choose a limited number of bus lines while maximizing coverage. The authors test the method using a dataset containing the routes of London buses. In our previous work, we propose a similar approach, but we consider the lengths of the streets when calculating the coverage [8]. The basic idea of this metric is that longer streets produce more data. We also propose a method to maximize coverage when the number of buses that can be part of the network is limited. We apply the method to GPS traces of the city of Rio de Janeiro and obtain the maximum possible coverage for different budgets, in terms of the number of participating buses. Nevertheless, our previous work does not take into account the delays caused by intermittent connection or application requirements in terms of measurement frequency. The present work proposes a more general metric, that considers both delays and measurement frequency when calculating the coverage.

Zhao *et al.* propose a coverage metric for vehicle-based WSNs that takes into account the time between measurements [9, 10]. In their work, time is

discretized into slots and the area of the city is divided into a grid. A cell in the grid is covered for a given time slot if and only if a participating vehicle is inside the grid cell during the time slot. They also propose a metric called Inter-Cover Time, which is the time elapsed between two consecutive samples of the same grid cell. They also propose a metric called the opportunistic coverage ratio, which is the expected number of covered grid cells on a given time interval. In our work, we use the measurement frequency to determine whether there is enough information in time, taking advantage of the predictability of buses routes. Additionally, we add the delivery delay to our coverage metric, making sure that applications get data in a timely manner.

## 6. Conclusion

Mobile Wireless Sensor Networks (MWSNs) are an option to decrease the sensing cost of a large area. On the one hand, the mobility increases the region covered by each sensor and lowers the networking costs, through delay-tolerant data delivery. On the other hand, regions are not covered the whole time and sensed data is not delivered instantly. In the scenario where buses are used to enable sensors with mobility, there are different visiting frequency and delivery delay for different streets. Since smart city applications have different data needs, each application can benefit from data gathered from different streets. Consequently, the coverage of the network is not the same for different applications.

This work proposes a coverage metric that takes into account the minimum visiting frequency and the maximum delivery delay tolerated by an application. The metric is used with real traces from buses of the city of Rio de Janeiro, in a scenario where bus stops are gateways. We present our results in the form of an abacus. Therefore, it is possible to obtain the coverage of the network, given a certain application that requires a minimum visiting frequency and maximum delivery delay. We also obtain the coverage of applications that have known minimum visiting frequencies and maximum delivery delays in the literature.

We found that a waste management application can cover 35.8% of the city of Rio de Janeiro using the buses of the city. We also show that, for this application, buses increase the coverage of the network up to 20.5 times, when compared to the scenario where sensors are static.

As future works, we plan to investigate the effect of using other communication technologies and more realistic propagation models. We believe that changes in the communication links can change the trade-off between mobile and static scenarios. We also plan to evaluate the effects of recruiting other vehicles and pedestrians for coverage.

**Acknowledgments**

The authors would like to thank CAPES, CNPq and FAPERJ for their financial support to this work.

## References

[1] B. Liu, P. Brass, O. Dousse, P. Nain, D. Towsley, Mobility improves coverage of sensor networks, in: International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), ACM, 2005, pp. 300–308.

[2] E. Ekici, Y. Gu, D. Bozdag, Mobility-based communication in wireless sensor networks, IEEE Communications Magazine 44 (7) (2006) 56 – 62.

[3] W. Dong, G. Guan, Y. Chen, K. Guo, Y. Gao, Mosaic: Towards city scale sensing with mobile sensor networks, in: 21st International Conference on Parallel and Distributed Systems (ICPADS), IEEE, 2015, pp. 29–36.

[4] G. Guan, Y. Chen, K. Guo, Y. Gao, W. Dong, Low-cost urban air quality monitoring with mosaic, in: Conference on Computer Communications Workshops (INFOCOM WKSHPS), IEEE, 2016, pp. 642–643.

[5] Y. Gao, W. Dong, K. Guo, X. Liu, Y. Chen, X. Liu, J. Bu, C. Chen, Mosaic: A low-cost mobile sensing system for urban air quality monitoring,

in: 35th International Conference on Computer Communications (INFOCOM), IEEE, 2016, pp. 1–9.

[6] A. Marjovi, A. Arfire, A. Martinoli, High resolution air pollution maps in urban environments using mobile sensor networks, in: International Conference on Distributed Computing in Sensor Systems (DCOSS), IEEE, 2015, pp. 11 – 20.

[7] J. Ali, V. Dyo, Coverage and mobile sensor placement for vehicles on predetermined routes: a greedy heuristic approach, in: International Joint Conference on e-Business and Telecommunications (ICETE 2017), SCITEPRESS, 2017, pp. 1–6.

[8] P. Cruz, R. S. Couto, L. H. M. K. Costa, A. Fladenmuller, M. Dias de Amorim, On the coverage of bus-based mobile sensing, Sensors 18 (6) (2018) 1–12.

[9] D. Zhao, H. Ma, L. Liu, J. Zhao, On opportunistic coverage for urban sensing, in: 10th International Conference on Mobile Ad-Hoc and Sensor Systems (MASS), IEEE, 2013, pp. 231–239.

[10] D. Zhao, H. Ma, L. Liu, X.-Y. Li, Opportunistic coverage for urban vehicular sensing, Computer Communications 60 (2015) 71–85.

[11] J. Merino, I. Caballero, B. Rivas, M. Serrano, M. Piattini, A data quality in use model for big data, Future Generation Computer Systems 63 (2016) 123–130.

[12] A. Zanella, N. Bui, A. Castellani, L. Vangelista, M. Zorzi, Internet of things for smart cities, IEEE Internet of Things journal 1 (1) (2014) 22–32.

[13] A. Sinaeepourfard, J. Garcia, X. Masip-Bruin, E. Marín-Tordera, J. Cirera, G. Grau, F. Casaus, Estimating smart city sensors data generation, in: Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net), IFIP, 2016, pp. 1–8.

[14] OpenStreetMap contributors, South America dump retrieved from https://download.geofabrik.de/south-america.html , https://www.openstreetmap.org (2017).

[15] IPLANRIO, Descrição do dataset conjunto GPS ônibus, Available online: http://dadosabertos.rio.rj.gov.br/apitransporte/apresentacao/pdf/documentacao_gps.pdf (accessed on 12 June 2018) (in Portuguese) (Dec 2016).

[16] J.-M. Zogg, Gps - essentials of satellite navigation, Tech. rep., UBlox (2009).

[17] D. Luxen, C. Vetter, Real-time routing with openstreetmap data, in: 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS '11, 2011, pp. 513–516.

[18] K. D. Zoysa, C. Keppitiyagama, G. P. Seneviratne, W. W. A. T. Shihan, A public transport system based sensor network for road surface condition monitoring, in: Workshop on Networked Systems for Developing Regions (NSDR), ACM, 2007, pp. 9–15.

[19] R. M. Alsina-Pagès, U. Hernandez-Jayo, F. Alías, I. Angulo, Design of a mobile low-cost sensor network using urban buses for real-time ubiquitous noise monitoring, Sensors 17 (1) (2016) 57–78.

[20] J. S. Apte, K. P. Messier, S. Gani, M. Brauer, T. W. Kirchstetter, M. M. Lunden, J. D. Marshall, C. J. Portier, R. C. Vermeulen, S. P. Hamburg, High-resolution air pollution mapping with google street view cars: Exploiting big data, Environmental Science & Technology 51 (12) (2017) 6999–7008.

[21] J. C. von Fischer, D. Cooley, S. Chamberlain, A. Gaylord, C. J. Griebenow, S. P. Hamburg, J. Salo, R. Schumacher, D. Theobald, J. Ham, Rapid, vehicle-based identification of location and magnitude of urban natural gas pipeline leaks, Environmental Science & Technology 51 (7) (2017) 4091–4099.

[22] L. Sanchez, L. Muñoz, J. A. Galache, P. Sotres, J. R. Santana, V. Gutierrez, R. Ramdhany, A. Gluhak, S. Krco, E. Theodoridis, et al., Smartsantander: Iot experimentation over a smart city testbed, Computer Networks 61 (2014) 217–238.

[23] P. Cruz, F. F. Silva, R. G. Pacheco, R. S. Couto, P. B. Velloso, M. E. M. Campista, L. H. M. K. Costa, Sensingbus: Using bus lines and fog computing for smart sensing the city, IEEE Cloud Computing (2018) 1–11.

[24] P. Cruz, R. S. Couto, L. H. M. K. Costa, An algorithm for sink positioning in bus-assisted smart city sensing, Future Generation Computer Systems 93 (2017) 761–769.