v. 3, n. 33, 2023

Journal of
Engineering
Research

AN ACTION APPLICATION FOR IMPROVING SOCIAL SECURITY USING GPS AND INTERNET EVENTS

Marcela T. Shinzato
Universidade de São Paulo (USP) – São Carlos, SP – Brasil

Geraldo P. Rocha Filho
Universidade Estadual Do Sudoeste Da Bahia – UESB – Vitória da Conquista, BA – Brazil

Luis H. V. Nakamura
Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, Brazil

Rodolfo Meneguette
Universidade de São Paulo (USP) – São Carlos, SP – Brasil

Abstract: In recent years there has been a huge increase in the number of cars and consequently the people who use them. This way, applications related to Intelligent Transport Systems play a key role in driver safety and have been gaining ground to assist with issues related to accidents, congestion and traffic problems. Given the above, this work proposes an application to capture data on events that are taking place in the city, disseminating these events to citizens who can take some action to try to minimize the amount of congestion that may occur due to an event on some road.

INTRODUCTION

In recent years there has been a large increase in the number of cars and this can impact mobility and traffic safety in a city. Since the disorderly growth of vehicles, without an adequate mobility plan, could lead to an increase in the number of congestion, accidents and other problems in the city's transport system [Meneguette and Boukerche 2020, de Souza et al. 2015].

The current trend is to provide vehicles and roads with resources to make transport infrastructure safer, more efficient and make passengers' experience on the road more pleasant. This trend can be achieved with an Intelligent Transport System (ITS), which uses information and communication technologies capable of optimizing and managing a city's transport system, collecting data to generate contextualization and infer about aspects of mobility management [Rocha Filho et al. 2020, Maschi et al. 2018, Meneguette et al. 2021]. Therefore, ITSs seek to improve resource management and increase people's convenience through information services and alerts [Dalarmelina et al. 2020]. Therefore, this improvement contributes to facilitating flow in the city, reducing time spent in congestion, reducing fuel consumption, CO_2 emissions and monetary losses [Hina et al. 2022, Meneguette et al. 2012, Lourenco et al. 2019].

Equipped and properly connected vehicles can collect, transmit and interpret information to assist in data acquisition and action by the driver and devices [Meneguette et al. 2021, Meneguette and Boukerche 2017]. Being able to sense and act on an environment, vehicles are a relevant tool for smart cities, not only in traffic management, but also for capturing information in real time that can be used in resource management [Zhou et al. 2022].

A smart city is made up of technologies in physical objects, transforming the environment more interactive and interconnected, allowing the development of solutions to urban problems [P and Mathew 2022, Duygan et al. 2022]. To achieve this, various urban data are collected and disseminated through communication infrastructures.

Although some commercial applications have the ability to indicate information about events that may be occurring on the road, this information is generated by the

users themselves and, therefore, there may be an inconsistency of information that will result in the indication of an event that is not occurring or occurred a certain time ago, which leads the driver to take an erroneous action.

Within this scope, this article proposes the development and use of an application called SAIR: System that Warns about Interdictions, which allows collecting vehicle mobility information. Consequently, with the wide use of the application it will be possible to monitor the mobility of an ITS across an entire city, providing better use of computational resources in the management of urban infrastructure, since the interdiction information placed by the device comes from a larger body such as, for example, the city council. Therefore, the work seeks to meet citizens' desires for a more dynamic, efficient and reliable mobility management infrastructure.

The article is organized as follows: in section 2 we present the solution architecture and describe how the proposed application works. Section 3 we evaluate the application on real devices and comment on the results obtained. Drying?? we summarize the proposed work; and finally, we show the references used in the article.

AN APPLICATION FOR IMPROVING SOCIAL SECURITY USING GPS AND INTERNET EVENTS

This section initially describes the general architecture of the proposed application and its main application functionalities and finally implementation details. At the end of the section, links to the source code, application installation file (APK), manuals, documentation, explanatory video and presentation requirements are available.

The architecture is presented in Figure 1, consisting of a mobile device using the Android platform, to collect (every three seconds) and store (every five minutes) location data - latitude and longitude - as a function of the clock time. user, in addition to the type of vehicle, until the user closes the application or if their position remains unchanged for two minutes. To do this, the user needs to enable GPS and have access to the Internet.

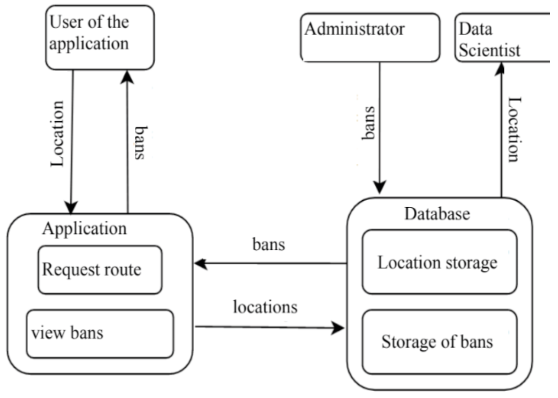


Figure 1. System architecture

Bans are entered by an administrator in the database to be stored and retrieved by the application with the aim of informing users. They can request directions from the application and their locations are collected and stored in the database to be used by data scientists.

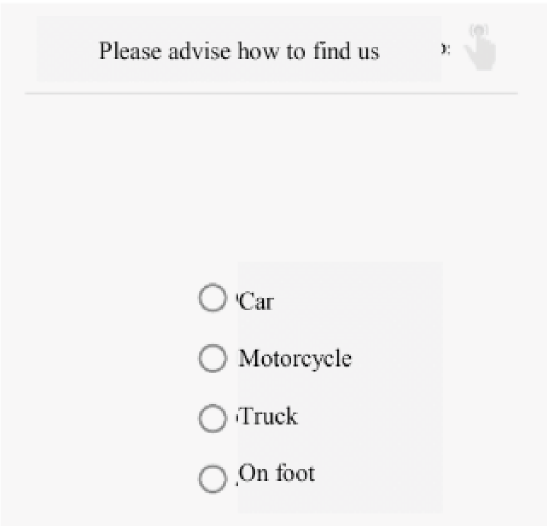
The application was called SAIR: System that warns about Interdictions. When executing it, the user must inform the means of transport used. The application creates an ID for the vehicle, stores the movement start date and the selected means of transport. During the route, the application allows communication with the GPS of the mobile device used, and thus collects the user's location (latitude and longitude), in addition to the date and time.

The user is directed to a new screen where the map with active closures can be viewed. A space for inserting origin and destination is also available to the driver where, after filling in the fields, the application will generate an alternative route. You can run other applications and perform other activities on the device while executing the proposed application.

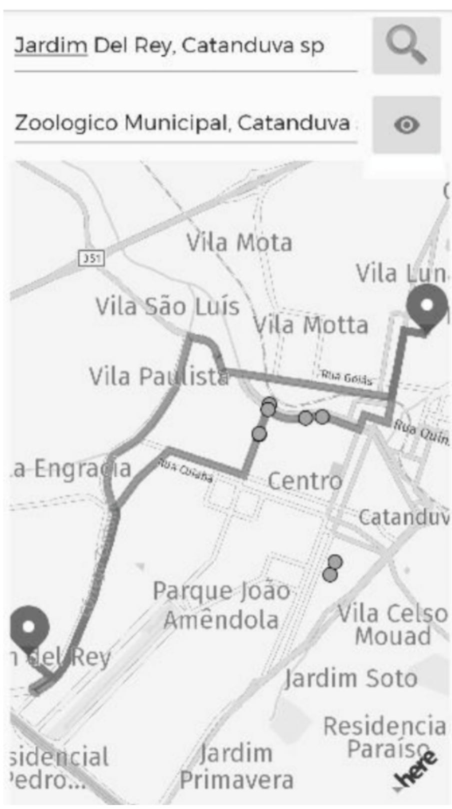
The application was implemented in the Java language, using the Android Studio[1] IDE, the Here[2] map service to show routes and closures to the user, the Firebase Realtime

1. https://developer.android.com/studio/
2. https://developer.here.com/

Database[3] database service and the JSON data storage structure to store information about closures and locations collected.



(a)Application home screen



(b)Map with closures and routes for the user, in addition to the search menu and button that takes you to the list of closures

Figure 2. System images

On the application's initial screen, shown in Figure 2a, the user must choose the form of transportation they will use: car, motorcycle, or on foot. Then the data collection service begins and the user is taken to another screen, shown in Figure 2b, containing: map with closures; a search menu with text boxes to enter origin and destination points, and a button with a magnifying glass symbol to search for the route between these points; and a button with an eye symbol to see a list of information about the bans.

The map appears centered on the user's location, if it is possible to obtain it. The closures are retrieved from the database, which contains the closures authorized by the larger entity (city hall) and displayed on the map and in a list of closures that can be accessed by clicking on the button with an eye design, with the map and the list updated in real time. The list is shown in Figure 3, whether or not it contains active bans.

The user can search for locations that are within the visible area of the map, with suggestions appearing as they type, which is shown in Figure 4a, and being able to move and change the zoom of the map. When touching the magnifying glass button, one or two (if it is possible to obtain an alternative route) routes appear from the origin to the destination, taking into consideration, the user's way of moving to obtain the paths faster. If it is not possible to find the origin or destination, warnings are shown, otherwise, the approximate time and distance of the routes are shown, shown in Figure 4b.

To do this, a flag called: Origin flag was used, the use of which is shown in Figure 5. The flag is initially 0, indicating that the origin has not yet been searched. Then the search for the origin is carried out. If it is unsuccessful, its value changes to 1, otherwise,

The search for the destination is carried out. It is checked whether the source was

3. 3https://firebase.google.com/

unsuccessful, which means that, if the destination was also not found, both must be revised, otherwise, if only the destination was not found, it is indicated that it must be revised. If the search for the destination is successful and the origin was not, it is indicated that it must be revised, otherwise, if both were found, the route between them is requested. Regardless of the search results for origin and destination, the flag value returns to 0, allowing the user to re-enter origin and destination as many times as desired, generating an infinite loop.

The user can tap the origin and destination markers to confirm their information, and, if they wish, request routes to different origins and destinations, as many times as they want.

The bodies responsible for the interdictions must send an email containing information regarding description, start and end locations, and date and start and end times of the closures, so that they can be placed in a JSON with the time in GMT format by the administrator. Figure 6 shows what the JSON must look like for a ban that starts at 7am and ends at 7pm.

Data collection is explained in Figure 7a. Two counters were used: one to pop when the user shows no activity for 2 minutes (inactive Counter) and another that pops every 5 minutes of activity (active Counter), these values were chosen so that the application did not consume a lot of battery since it doesn't find events next to the application. Initially, the user is considered inactive, and, upon detecting a change in location of at least 50 meters, a list of locations called list Of Location is started. The location is added to the list and the counters are activated, making the user active, which is represented by the action of waiting. Once active, after 5 minutes the active Counter overflows. To prevent the operation from being interrupted by another counter overflowing, the counters are deactivated. Then, the locations are sent to the database, creating a new node. We want the user to become inactive to have continuity in the loop. To do this, you need to restart the list.

Furthermore, while active, every 3 seconds your location is collected, if there is a change in location of at least 50 meters, being added to the list, and restarting the inactive Counter, so that it only pops if you restart it. is not called for 2 minutes. If this occurs, the user becomes inactive. This way, there is a loop of states that is only broken when closing the application. When this closure occurs, the on Destroy () method is called, creating a new node with the collected locations and leading to the end of the service. The location update values were chosen in order to minimize the amount of information stored without losing the accuracy of the application's location. Figure 7b shows what the JSON of collected locations looks like.

The on Location Changed (Location location) method must be called when a change is detected in the user's current location, and it can also define a minimum time interval and/or distance between one call and another. However, it is not necessarily called respecting the minimum time interval and distance defined between one call and another. This implies calls without changing location and without requests for a minimum time interval. To get around this problem, as in the program flow you only want to know if the location has changed to reactivate the inactive Counter or change from inactive to active state, simply compare whether the returned location is the same as the previous one.

A state variable that represents the state of the location collection filters whether on Location Changed calls must be considered or ignored, which is shown in Figure 8 and Table 1. At the moment the user logs in, becomes inactive (start of service or overflow of a counter) its value is 0, indicating no registered location. When receiving the on Location

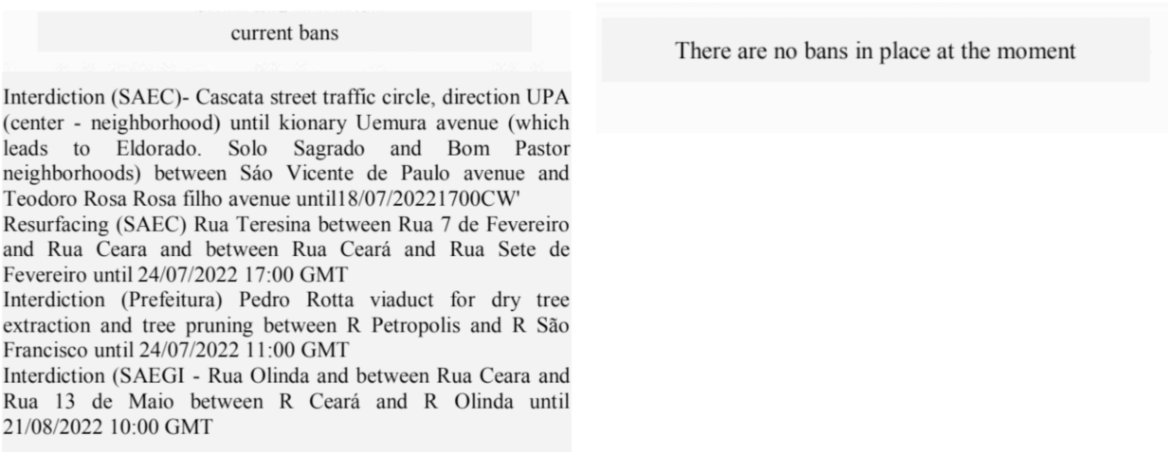
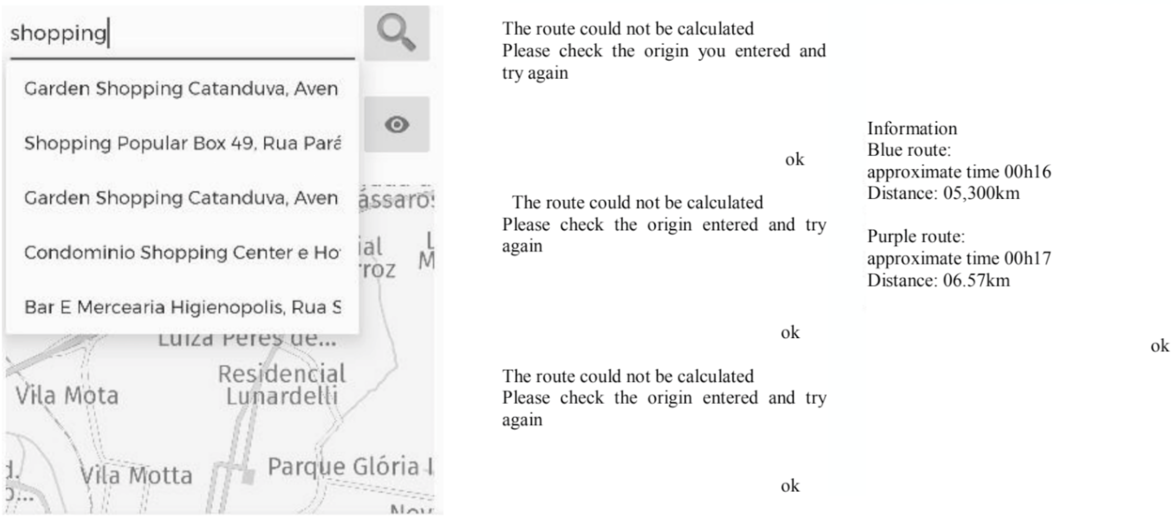


Figure 3. Screen that shows a list of active bans



(a)Suggestions that appear while the user types

(b)Possible messages about locations not found or routes returned

Figure 4. Application screens

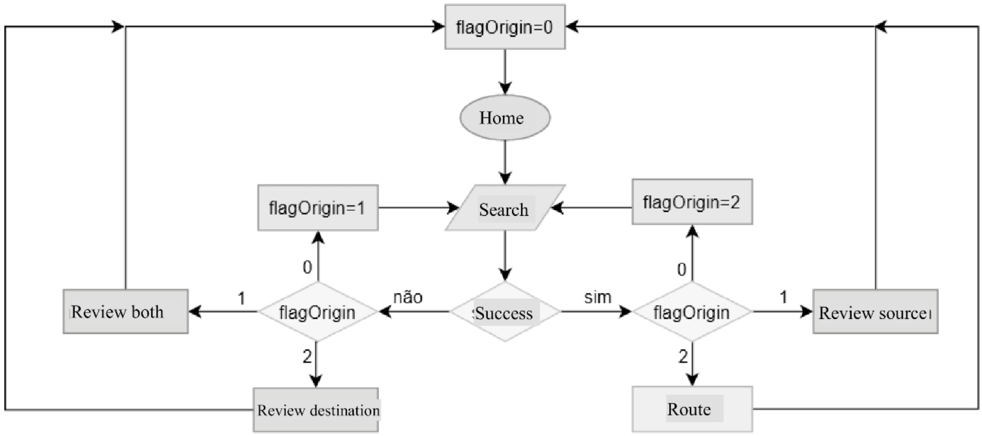
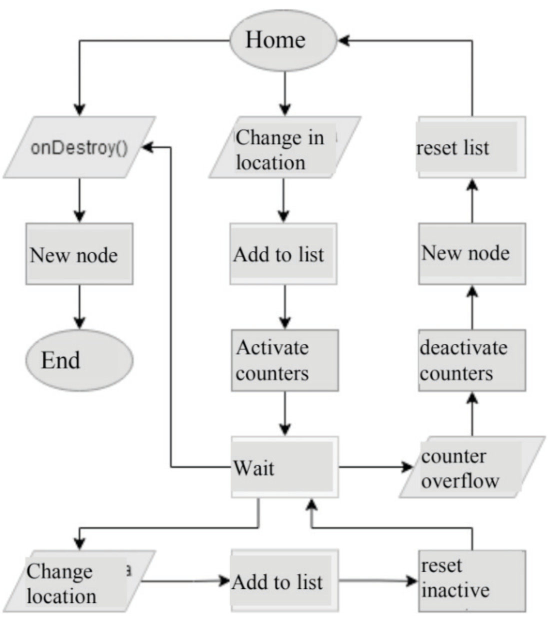


Figure 5. Diagram showing how the flag is used to request the route or return the error

Journal of Engineering Research ISSN 2764-1317 DOI 10.22533/at.ed.31733323250910

```
{
  "beginDate" : "17/02/2022 07:00",
  "description" : "Interdicao",
  "destination" : {
    "lat" : −21.133602410012063,
    "lng" : −48.987751907094065,
    "street" : "Av. Benedito Zancaner, 1705 − Jardim do Lago,
    Catanduva − SP, 15801−440, Brasil"
  },
  "endDate" : "20/04/2022 19:00",
  "organization" : "Administrador",
  "origin" : {
    "lat" : −21.134185330516875,
    "lng" : −48.98877785204226,
    "street" : "Av. Benedito Zancaner, 1481 − Jardim do Lago,
    Catanduva − SP, 15800−000, Brasil"
  },
  "status" : true
}
```

Figure 6. Example of ban JSON



(a)Program flow formulated to implement user activity states and desired functionalities

```
{
    "averageOfSpeed" : 1.7399999856948856,
    "cityName" : "Sao Carlos",
    "finishDate" : "11/03/2022 13:56:25",
    "initDate" : "11/03/2022 13:51:46",
    "listOfLocation" :
    [
        {
            "dateNow" : "11/03/2022 13:51:46",
            "latitude" : −22.00792041,
            "longitude" : −47.89580776
        },
        {
            "dateNow" : "11/03/2022 13:53:01",
            "latitude" : −22.0084575,
            "longitude" : −47.89609404
        },
        {
            "dateNow" : "11/03/2022 13:54:25",
            "latitude" : −22.00879623,
            "longitude" : −47.89663043
        }
    ],
    "meansOfTransport" : "OnFoot"
}
```

(b)Location JSON example

Figure 7. Information about data collection

Changed (Location location) call, the value is checked. If it is 0, the list is initialized with the location obtained, indicating that the user must still be considered inactive, but there is already a location to be compared with the next one, and then its value becomes 1. If it is 1, the current location is compared with the previous one, to determine whether the user is still inactive - keeping value 1 and returning to the beginning of the flow - or whether it must be changed to 2 and therefore, being able to consider the change of location and the active user.

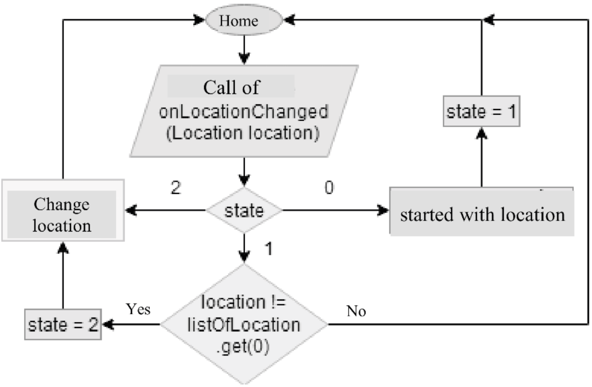


Figure 8. Program flow for the state variable

Value	Activity	Location registration
0	Inactive	Absent
1	Inactive	Present
2	Active	Present

Table 1. Meaning of state

ASSESSMENT AND RESULTS

This section presents how SAIR was evaluated. It is worth mentioning that SAIR was evaluated on real devices.

To carry out initial tests of SAIR, the App was installed on devices A5 from 2015 (Android 6.0.1), Note 10.1 from 2014 (Android 5.1.1), and G 1st generation from 2014 (Android 5.1). In the last two devices, the RAM memory is only reported at the time of the query, so it was noted every 5 minutes and its average was calculated and the Note does not inform about battery usage.

Three tests of 30 minutes each were carried out at ``Universidade de São Paulo``, in São Carlos-SP. In each of them, the mobile data of a device was connected throughout the route, and, for the others, the university's own wireless network was used, which suffered disconnections during the route, to connect to different points. This way it was observed whether it is really necessary to have an Internet connection throughout the journey without compromising the construction of the dataset.

Just to differentiate them in the database, the A5 was assigned as a means of transportation "on foot", for the Note car and for the G motorcycle. The data relating to the last two were removed, as the means of transportation was not consistent with reality and so that the data would not be duplicated. Table 2 displays the results of the experiments.

	A5	Note	G
Average RAM	43MB	43MB	46MB
Mobile data	81KB	60KB	78KB
Battery	4mAh	Not informed	3,mAh
CPU time	1m51s	Not informed	1m3s

Table 2. Average use of RAM, mobile data and device battery when running the application for 30 minutes

To check whether the construction of the dataset was compromised by the lack of Internet connection during the entire route, we searched Google Maps for the coordinates obtained, added labels to them, and connected the dots in the Paint software. The results are shown in Figure 9, noting that there were no losses, and that, therefore, the user does not need to be connected to the Internet during the entire journey.

Thus, we can observe that the system had greater consumption on the A5 and Note devices due to the use of the same memory mapping technology, unlike the G. The A5's mobile data was greater due to greater precision of your GPS, but very close to G. All mobile data consumption was much lower than that of Waze (115KB/minute), Google Maps (46KB/minute) and Apple Maps (92KB /minute) since the average result for the proposed application was 73KB in 30 minutes, that is, 2.4KB/minute. Regarding the battery, on all devices there was less than 1% battery drain, while, in 30 minutes, for Google Maps this value is 1.52%. Furthermore, the obtained value of 4mAh corresponds to 40mW, much lower than the 824mW of Waze and 745mW of Google Maps.

CONCLUSION

This work presented an infrastructure for monitoring vehicles and generating real-time data on urban mobility. This infrastructure is composed of a mobile application responsible for capturing, anonymously, the real geographic coordinates of users and storing this information so that it can be used in network simulators. Furthermore, it informs users about possible events on the roads.

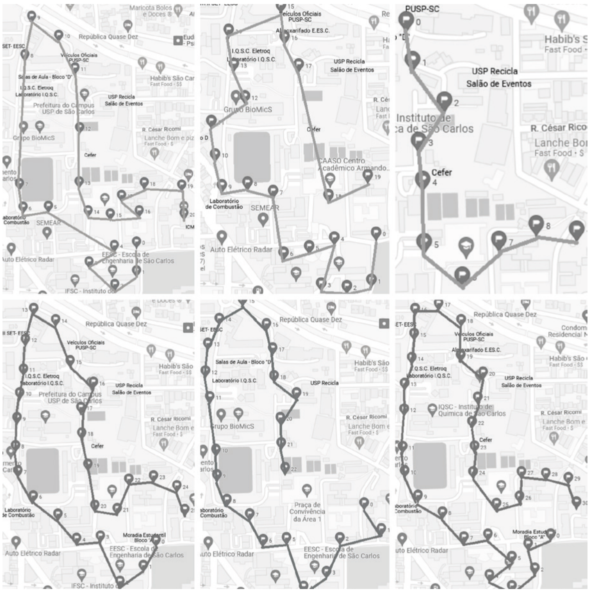


Figure 9. User path collected using mobile data throughout the journey, in the top line, and suffering from WIFI disconnections, in the bottom line. The columns refer to A5, Note and G, in this order.

As future work, a website can be implemented that allows users to see the closures on the map if they cannot consult their cell phones at the moment, in addition to allowing responsible entities to enter bans using login and password, which must be approved by the system administrator. On Android you can: check whether there is really a need for all dependencies; allow you to view the map in landscape mode; and change the code to minimize warnings.

REFEREENCES

Dalarmelina, N. d. V., Teixeira, M. A., and Meneguette, R. I. (2020). A real-time automa- tic plate recognition system based on optical character recognition and wireless sensor networks for its. *Sensors*, 20(1).

de Souza, A. M., Yokoyama, R. S., da Fonseca, N. L. S., Meneguette, R. I., and Villas, L. A. (2015). Garuda: A new geographical accident aware solution to reduce urban congestion. In *2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, pages 596–602.

Duygan, M., Fischer, M., PA˜ ¤rli, R., and Ingold, K. (2022). Where do smart cities grow? the spatial and socio-economic conFiguretions of smart city development. *Sustainable Cities and Society*, 77:103578.

Hina, M. D., Soukane, A., and Ramdane-Cherif, A. (2022). *Computational Intelligence in Intelligent Transportation Systems: An Overview*, pages 27–43. Springer International Publishing, Cham.

Lourenço, M., Souza, F. S., Silva, C. M., Meneguette, R. I., and Guidoni, D. L. (2019). A hybrid v2i and v2v approach for urban traffic management in vehicular networks. In *2019 IEEE Latin-American Conference on Communications (LATINCOM)*, pages 1–6.

Maschi, L. F. C., Pinto, A. S. R., Meneguette, R. I., and Baldassin, A. (2018). Data sum- marization in the node by parameters (dsnp): Local data fusion in an iot environment. *Sensors*, 18(3).

Meneguette, R., De Grande, R., Ueyama, J., Filho, G. P. R., and Madeira, E. (2021). Vehi- cular edge computing: Architecture, resource management, security, and challenges. 55(1).

Meneguette, R. I. and Boukerche, A. (2017). A cooperative and adaptive resource schedu- ling for vehicular cloud. In *2017 IEEE Symposium on Computers and Communications (ISCC)*, pages 398–403.

Meneguette, R. I. and Boukerche, A. (2020). Vehicular clouds leveraging mobile urban computing through resource discovery. *IEEE Transactions on Intelligent Transporta- tion Systems*, 21(6):2640–2647.

Meneguette, R. I., Madeira, E. R. M., and Bittencourt, L. F. (2012). Multi-network packet scheduling based on vehicular ad hoc network applications. In *2012 8th international conference on network and service management (cnsm) and 2012 workshop on systems virtualiztion management (svm)*, pages 214–218.

P, S. and Mathew, A. (2022). Cross layer design with weighted sum approach for exten- ding device sustainability in smart cities. *Sustainable Cities and Society*, 77:103478.

Rocha Filho, G. P., Meneguette, R. I., Torres Neto, J. R., Valejo, A., Weigang, L., Ueyama, J., Pessin, G., and Villas, L. A. (2020). Enhancing intelligence in traffic management systems to aid in vehicle traffic congestion problems in smart cities. *Ad Hoc Networks*, 107:102265.

Zhou, Z., Dong, X., Li, Z., Yu, K., Ding, C., and Yang, Y. (2022). Spatio-temporal feature encoding for traffic accident detection in vanet environment. *IEEE Transactions on Intelligent Transportation Systems*.