Cooperative estimation of Vehicular Traffic using Mobile Applications

Alfredo Campuzano
Facultad Politécnica
Universidad Nacional de Asunción
San Lorenzo - Paraguay
acampuzano@outlook.com

Rubén López
Facultad Politécnica
Universidad Nacional de Asunción
San Lorenzo - Paraguay
rubenlop88@gmail.com

Joaquín Lima
Facultad Politécnica
Universidad Nacional de Asunción
San Lorenzo - Paraguay
joaquin.lima@pol.una.py

***Abstract*—The increasing amount of vehicles in developing countries, where its cities do not have a well-planned road infrastructure nor installed technology to monitor traffic conditions, tends to make difficult the daily traffic and to reduce the quality of life. Because of this, collect and analyze traffic information in an inexpensive and easy manner becomes necessary. This paper presents the implementation details of a system called Autotracks, which collects information from vehicular traffic through mobile devices in order to approximate the traffic conditions in real time. A Floating Car Data approach is used in combination with activity recognition to monitor the location of users in moving vehicles. The real path of the vehicles is approximated by using a Map Matching process. Then, all the trajectories are aggregated in order to approximate the traffic condition in the last period of time and finally the traffic status information in real time is obtained.**

Keywords*—*Map Matching, Floating Car Data, GIS, Activity Recognition, Traffic State Estimation

I. INTRODUCTION

Traffic congestion is one of the most serious problems in urban areas nowadays, the continuous increase of vehicles in traffic, the elevated use of particular cars and the lack of town planning are some factors that negatively affect life quality of its citizens. In addition, many developing countries have no traffic control systems due to the high investments cost that are required. Therefore, the possibility to collect and analyze information on traffic in an inexpensive and easy manner becomes evident. Traffic information systems based on *Floating Car Data* (FCD) have proven to be a viable alternative for obtaining this information efficiently. [1], [2].

Present FCD systems are based on the use of probe vehicles equipped with GPS sensors. For this purpose, taxi fleets [1], [2] and cars with GPS devices installed by insurance companies [3] have been used as probe vehicles. However, in developing countries there are no public programs or private efforts that promote the use of such systems.

Alternatively, the collecting of FCD information can be based in GPS devices of drivers' smart phones. This kind of applications has already been used for vehicle tracking [4], predicting arrival time of buses [5] and for travel time estimation [6]. In such cases, these studies have demonstrated the feasibility of using this technology to estimate the traffic conditions in real time and some of them proposes a penetration between 2 and 3% to provide an accurate estimation of traffic flow speed [7], [8].

To estimate a traffic condition, the first step is to determine the trajectory of vehicles through the city streets, this process is known as Map Matching (MM). There is a great variety of MM algorithms, from the simple ones, based only on geographic information [9], to the complex ones, based on statistical models and other advanced techniques [10], [11]. Due to the limitations in mobile platforms, such as battery consumption restrictions, limited or intermittent connectivity, accuracy errors, among others, the use of specialized MM algorithms result appropiate to process sparse and inaccurate samples [12].

This paper presents an implementation of a system to estimate traffic conditions in real time using smart mobile devices. The proposed solution is based on FCD information that is collected by a mobile application called Autotracks, which is installed in smart devices and periodically sends the collected FCD information to a central server. To address the impact of battery consumption, an activity recognition mechanism is applied in order to track the users only when they are in moving vehicles. Later, the collected trajectories are processed using a MM algorithm. The resulting information is stored in a GIS database and then is aggregated in order to approximate the traffic condition.

The rest of the paper is organized as follows: Section II describes the techniques used to collect traffic information. Section III describes the MM problem and the different algorithms that exists today. Section IV explains the fundamental parameters of traffic flow. Section V presents the proposed solution, its architecture, the selected techniques used in each step of the process. Section VI describes the evaluation of the obtained experimental results. Finally, Section VII presents the conclusions of this paper and the propose of future works.

II. TRAFFIC DATA COLLECTION

In present exists a variety of technologies for traffic data collection. These technologies can be splitted into two categories. The first one is called *in-situ technologies* and takes traffic data through detectors installed within or over the way. The second one, called *Floating Car Data* (FCD), obtain traffic data using probe vehicles equipped with measuring devices [13].

The in-situ traffic detectors technologies are based on data collection by dedicated devices for this purpose, which are physically arranged in places subject to measurement. They are divided into two categories: *a)* intrusive technologies, which are mounted in or below the surface of the routes and whose installation and maintenance causes a potential disruption of traffic (e.g. pneumatic tubes and magnetometers), and *b)* non-intrusive technologies, which are mounted above or on the surface of the routes and whose installation and maintenance does not generate a traffic disruption (e.g. photographic or video cameras).

In other hand, many traffic management systems use devices mounted on vehicles, known as *Automatic Vehicle Location* (AVL). These devices can provide two types of information: *a)* location information, identifying vehicles that cross certain points in the net of streets, or *b)* continuous information, identifying the vehicle routes through the streets. Probe vehicles can be equipped with AVL devices or use mobile devices as AVL devices when the user travels in a car.

Some FCD-based systems take into account the recollection and processing of traffic data obtained in real time through AVL devices. All vehicles equipped with these devices act as mobile sensors in the street network. Data such as vehicle locations, speed and direction of travel are recollected for its processing. Then, useful information, such as the traffic condition and alternative routes, can be calculated and distributed.

III. Map Matching

The process of identifying the real path followed by a vehicle on a street network from imprecise samples of its location is known as *Map Matching* (MM). This process is used in a wide range of services and geographic information systems, such as predicting user trajectories [14], in-vehicle navigation systems [11], monitoring traffic conditions [6], estimate bus arrival times [4], among others.

In order to carry out a MM process the following concepts should be considered:

- **Sample or Point**: A sample or point p is the set of all measurements collected by the vehicle in a given time. These measurements include the vehicles location, its direction of movement, speed, etc.
- **Trajectory**: A trajectory T is an ordered sequence of samples or points that belongs to an observed vehicle and that is collected during a journey of the same.
- **Street Network**: A street network $G(V, E)$ is a directed graph representing the shape and properties of a system of streets in a particular geographic area. Each vertice $v \in V$ describes an intersection between streets and each edge $e \in E$ represents the form and attributes of a street.
- **Reconstructed Path**: A reconstructed path R is an ordered sequence of interconnected streets through which a vehicle may have traveled.

Thus, the problem of MM can be defined as follows: *Given a trajectory T and a street network $G(V, E)$, find the path R which matches T with its more realistic reconstruction over $G(V, E)$.*

According to the information and the techniques used for its implementation, MM algorithms can be classified as: *1) Geometric*, that only use information about the position and distance between streets and points [9], they are simple to implement and are often used as an initial step in the implementation of another algorithms. *2) Topological*, which incorporate topological information about the street network [12], [15], [16], [17], such as streets connectivity information, turn restrictions, directions and speed limits. *3) Statistical*, which define probability regions around each point and then analyze the street segments within these regions [18]. And *4) Advanced* ones, which combine the geometric, statistical and topological techniques with other concepts such as Kalman Filters, Hidden Markov Models, Fuzzy Logic, among others [6], [10], [19], [20].

With respect to the time when the data processing is performed, there are two categories: *1) Incremental* or *On-line* algorithms, that perform the process of MM as new points are obtained [6], [19], [16], [17], [10] and are used in real time applications such as personal navigation assistants. And, *2) Global* or *Off-line* algorithms, that perform the MM process after all the points have been collected [12], [15], they are used in traffic analysis applications or in studies about users behavior.

Depending on the sampling frequency, two more categories can be identified: *1) high-sampling* algorithms, which typically work with sample intervals in the range of a few seconds and are generally executed *on-line* [16], [17], [10]. And, *2) low-sampling* algorithms, that work for sampling intervals of several minutes and are usually executed *off-line* [12], [15]. In general, all algorithms can work with samples of high or low frequency, but certain algorithms become less effective as the sampling frequency decreases.

IV. Traffic Measures

Vehicular Traffic (or Traffic for short) is the phenomenon caused by the flow of vehicles on a road, street or highway. When traffic flow is higher in a particular area, circulation congestion occurs and derives primarily in time loss and extra fuel consumption [21].

The traffic flow measurements can provide useful information about its nature. These measures can be classified mainly in: *a)* quantity measurements, including density and traffic volume; and *b)* quality measurements, such as speed. In Addition, a measure of traffic flow can be: *a)* macroscopic, characterizing the traffic as a whole; or *b)* microscopic, identifying the behavior corresponding to individual vehicles. The main measures for traffic flow are: *speed*, *volume*, and *density* [22]

The speed is defined as the traveled distance per time unit. The observed speed usually varies from one vehicle to another. To represent this variation, several types of speed can be defined [22]. The most important are:

1) *running speeds*, it refers to the observed average speed for a route whenever the vehicle is in motion, not considering the time intervals in which the vehicle is stationary.

2) *journey speed*, which is given by the distance between two points, divided by the total time used by the vehicle to complete the trip, including the time that the vehicle is momentary stopped.

3) *time mean speed*, defined as the average speed of movement of all vehicles passing through a point on the road over a period of time.

4) *space mean speed*, defined as the average travel speed of all vehicles passing through a section of the road for a specified period of time.

The volume is defined as the number of vehicles passing through a road during a time interval. The measurement is carried out by counting the number of vehicles passing through a particular point for a defined time.

The density is defined as the number of vehicles occupying a length of the road or lane in a given time and is usually expressed in vehicles per kilometer.

The most relevant measure for estimating real-time traffic is speed. This information is generally used to reduce congestion on the roads, helping drivers to make decisions based on the traffic conditions.

V. Architecture

The proposed solution is based on FCD information that is obtained by mean of mobile devices. The FCD information is processed on the server side using a MM algorithm and the result is stored in a GIS database in order to approximate the traffic condition.

Figure 1 shows the modular architecture of the developed system. The system has three main components: *1)* a *GIS database* that stores the street map and the recollected trajectories of the vehicles, *2)* a *Mobile Application* that captures FCD information about the path traveled by the vehicles, and *3)* a *Server-side Application* that processes the recollected data and calculates the traffic condition.

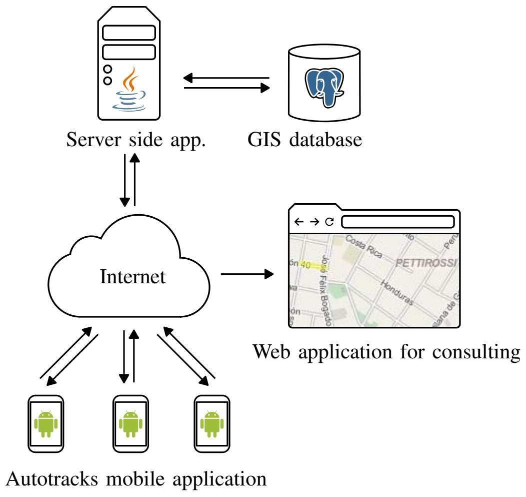


Figure 1: Modular Architecture of the system.

A. Data Storage

In order to generate information about the traffic in a particular geographic area, both a street map and a set of paths obtained by FCD mechanisms over a period of time are necessary.

The street map is obtained from Open Street Maps and stored in a GIS database as a directed graph, where the vertexes correspond to street intersections and the edges correspond to segments of streets.

The information of user trajectories are stored in the GIS database in the following tables:

1) **Locations**: stores the location data received from the mobile application. Date, time, latitude, longitude, speed, direction, accuracy, altitude and trajectory to which it belong.
2) **Trajectories**: a set of sample locations that serve as input for the process of MM. Each user trajectory have an input in this table.

B. Implementation of FCD

The FCD information is obtained from a mobile application called Autotracks[1] and that was developed as part of this work. This application captures, stores and periodically sends the user trajectories. For this purpose, Autotracks has three main functions:

1) Activity recognition: The activity recognition is used to determine the action that a person is doing based on a pattern of movements observed by the sensors of its mobile device, such as the accelerometer, the gyroscope and the GPS. In Autotracks an implementation provided as part of Google Play Services[2], called Activity Recognition[3], is used to determine when a user is in a moving vehicle.

For each activity recognition result, Autotracks checks if the user tracking is running or not and handles the start or end of the user tracking according to the detected activity. The activity detected is stored for its use in the next invocation of the procedure. The time when the detection result occurs is also stored. The Algorithm 1 shows the procedures that are performed in Autotracks for each activity recognition result that is obtained.

2) Tracking: The user location can be performed by a mobile device using triangulation of cellphone towers, from detected WiFi hotspots with known location or by using GPS sensors. Autotracks automatically work with any of these options using an implementation called Fused Location Provider[4].

When a location is obtained via GSM or WiFi the speed information is not available. To address this lack of information, the speed is estimated by using the previous location, if it is available, as follows:

$$v = \frac{d}{t} \qquad (1)$$

where d is the distance between the current location and the previous one, and t is the elapsed time between these locations.

To prevent erroneous user locations three filtering conditions were considered. The first discards user locations with low accuracy (> 200m). The second consists in determining the relative speed between two consecutive user locations, if

[1]https://play.google.com/store/apps/details?id=py.com.fpuna.autotracks
[2]http://developer.android.com/google/play-services/index.html
[3]http://developer.android.com/training/location/activity-recognition.html
[4]http://developer.android.com/training/location/receive-location-updates.html

Algorithm 1 Procedures for Activity Recognition.

```
 1: procedure DETECTEDACTIVITY($activity$)
 2:     $inVehicle$ = ISINVEHICLE($activity$)
 3:     if $inVehicle$ then
 4:         $lastState$ = VEHICLE
 5:         if TRACKING is not running then
 6:             Start TRACKING
 7:         end if
 8:     else
 9:         $lastState = null$
10:         if TRACKING is running then
11:             Stop TRACKING
12:         end if
13:     end if
14: end procedure

15: function ISINVEHICLE($activity$)
16:     if $activity$ == ON_FOOT then
17:         return $false$
18:     else
19:         Inicialize $t$ with current time
20:         if $activity$ == VEHICLE then
21:             $t\_lastDetection = t$
22:             return $true$
23:         else
24:             if $lastState$ == VEHICLE then
25:                 $\Delta t = t - t\_lastDetection$
26:                 if $\Delta t \leq$ TOLERANCE then
27:                     return $true$
28:                 else
29:                     return $false$
30:                 end if
31:             else
32:                 return $false$
33:             end if
34:         end if
35:     end if
36: end function
```

its result greater than a preset value (140km/h) then the newer location is discarded. The third filter discard new user locations that are closer to the last one (< 50m). The procedure used to register a location is described In the Algorithm 2.

3) Sending Data: The application sends user locations data via Internet to a central server. To reduce battery consumption caused by the use of the data network, Autotracks temporarily stores the captured used locations and send them to the server periodically.

To send the user locations, a fully completed travel is not necessary. All user locations captured over a defined period of time may be sent to the server. The rest of the locations of the same route are sent in the following periods. Thus, partial trajectories can be used to estimate the traffic in real time.

C. MM Implementation

A Map Matching (MM) process is performed each time that a trajectory is received in the central server in order to calculate the approximate path traveled by the vehicle. The applied algorithm is known as ST-Matching[12] with the improvements proposed by [23] and [24].

Algorithm 2 Procedures for consideration of user locations.

```
 1: procedure RECORD($location$)
 2:     $isValid$ = ISVALID($location$)
 3:     if $isValid$ then
 4:         $lastLocation = location$
 5:     end if
 6: end procedure

 7: function ISVALID($location$)
 8:     Inicialize $p$ with the $localizacion$ accuracy
 9:     $\Delta d$ = DISTANCE($location, ultimaLocalizacion$)
10:     $\Delta t$ = TIEMPO($localizacion, lastLocation$)
11:     $v = \Delta d / \Delta t$
12:     if $\Delta d >$ D_MIN $\wedge v <$ V_MAX $\wedge p <$ P_MAX then
13:         return $true$
14:     else
15:         return $false$
16:     end if
17: end function
```

ST-Matching is a topological algorithm specifically designed to work off-line and with samples of low frequency. This algorithm consists of three main parts [12]: *a)* the *Candidate Selection*, *b)* the *Spatial and Temporal Analysis* and *c)* the *Result Matching*.

Thus, the inputs for the MM process are the street net and a vehicle trajectory. The result of this processing is an approximation of the actual path traveled by the vehicle, that is stored in the GIS database as a input for a posterior estimation of the actual traffic condition.

D. Traffic estimation

At the end of each MM process, each point of the approximated trajectory is associated to an edge of the map of streets. Each point contains the speed at which its corresponding vehicle had traveled. In order to estimate the actual Traffic Condition, all associated speed values in each street segment are used to calculate its corresponding local average speed respect to an interval of interest Δt as:

$$V^j_{ave}(t - \Delta t, t) = \frac{1}{n^j_{t-\Delta t,t}} \sum_{k=1}^{n^j_{t-\Delta t,t}} \hat{v}_k(j) \tag{2}$$

where $\hat{v}_k(j)$ is the k-essime estimated speed at street j in the interval $[t - \Delta t, t]$, $n^j_{t-\Delta t,t}$ is the total number of samples available for the street j during the interval and t is the current time.

Finally, to represent the calculated information in a simple way, four possible levels of speed on each street segment were defined:

1) **Red:** for speeds between 0 and 14 km/h.
2) **Orange:** for speeds between 15 and 29 km/h..
3) **Yellow:** for speeds between 30 and 39 km/h..
4) **Green:** for speeds of 40 or more km/h.

Thus, segments of roads on the map are colored according to their level of speed as seen in Figure 2.

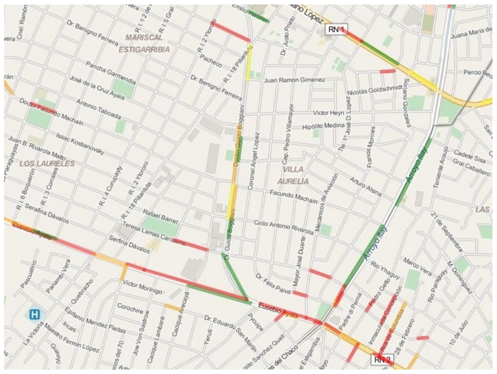


Figure 2: Streets state in Autotracks

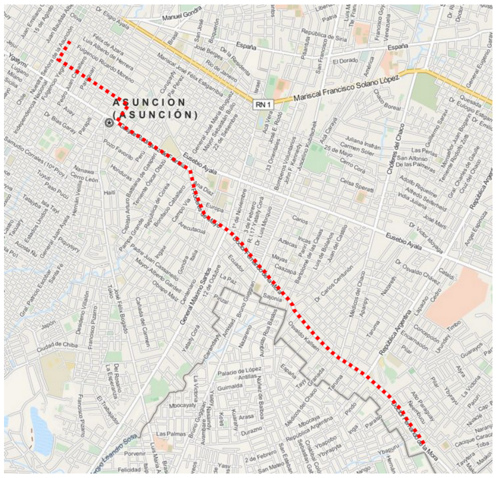


Figure 3: Test path for experiments

VI. Empirical tests and results achieved

This section describes the field tests and the experimental results obtained. First, a group of empirical tests were conducted in order to determine the most appropriate values for the parameters used in the collection of FCD. Then, a second group of empirical tests were performed in order to determine the appropriate values for the parameters involved in the detection of the reconstructed paths. Finally, by means of the distribution of Autotracks application, we proceeded to capture and process a set of FCD in order to derive a Traffic Condition.

A. Deduction of parameters for FCD

For the collection of FCD information, an activity recognition mechanism is used in Autotracks to track the user locations only when they are in a moving vehicle. This relies in a parameter called *recognition interval* that defines the time interval in which the verification of user's activity is performed, and its value determines how fast is detected the state condition to start or end the tracking. Also is important to take into account that while shorter this interval is, higher would be the battery consumption, because the mobile device must turn on its sensors more often.

In order to determine the appropriate recognition interval value, the values of 15, 30, 45 and 60 seconds were considered. For each value, an empirical tests of 30 iterations were performed in the travel path illustrated by the Figure 3 in two travel conditions, in a bus and in a private car. The achieved results of this test are shown in Table I.

Table I: Average startup time of tracking.

Interval (s)	Iterarions	In bus (min.)	In car (min.)	Both (min.)
15	30	1.55	1.18	1.37
30	30	1.64	1.30	1.47
45	30	2.59	2.05	2.32
60	30	4.37	2.75	3.66

Based on the test results, the recognition interval of 30 seconds was selected, for being the longest interval that has a good response time and would demand a lower battery consumption.

During a trajectory, the vehicle may be temporarily stopped due to various causes, such as in a red light or in a bus stop. In these cases the tracking should not be erroneously stopped in order to avoid cuts in the capture of trajectories. Thus, a *tolerance time* must be considered. When a stop is detected, the time of the last of motion detection is considered and the tracking is stopped only if the tolerance time is exceeded. Is important to note that a higher tolerance time means that will be necessary a longer time to stop the tracking and may result in a negative impact on battery consumption.

Moreover, it is also important to consider the time percentage that device monitors the user in relation to the total duration of its trip, this is called *Tracking Rate*. The tolerance time and the recognition interval together influence the real tracking time, so they must be jointly selected.

With respect to the recognition interval previously selected, the tolerance time values of 3, 5, 7 and 10 minutes were considered. Then, a new empirical test of 30 iterations was performed for each value. The achieved results about the observed tracking rate for these values of tolerance time considered are shown in Table II.

Table II: Real tracking rate results

Interval (s)	Tolerance (min.)	In bus (%)	In car (%)
30	3	80.34	89.16
30	5	85.86	91.56
30	7	94.44	96.11
30	10	95.83	97.22

Based on the previous results, the tolerance time of 10 minutes was selected.

B. Verification of effectiveness of MM

The MM process takes as input a set of captured user locations. The time interval at which these points are obtained is called *sampling interval*. A large value of this interval results in a fewer number of samples and thus in a lower consumption of battery, but in turn, a poor approximation of the path could be obtained. However, with small intervals, a larger sample is obtained, which increases the quality of the approximation but results in a higher battery consumption.

Therefore, to measure the effectiveness of MM process, the amount of points that are correctly assigned to streets must be taken into account. This effectiveness was observed with values of 30, 60 and 120 seconds, and for each interval, 30 trips were done in the path indicated in Figure 3. For each sampling interval value, the total collected points was considered and compared to the number of points assigned correctly. The results can be appreciated in Table III.

Table III: MM process effectiveness

Interval (s)	Total points	Correct points	Hit Rate (%)
30	1227	1141	93
60	594	541	91
120	313	275	88

It can be noted that there is a slight difference in favor of the 30 seconds interval compared to the 60 seconds interval, and a slightly larger difference compared to 120 seconds interval. Based on these results, the 60 seconds interval was selected.

C. Traffic Analysis

To collect data for the traffic analysis, the Autotracks mobile application was distributed through Google Play[5] and users results were observed over a 5 week period, in which the app had an average of 212 active users per day, collecting a total of 123.419 samples. Figure 4 shows the number of active users per day during the test period.

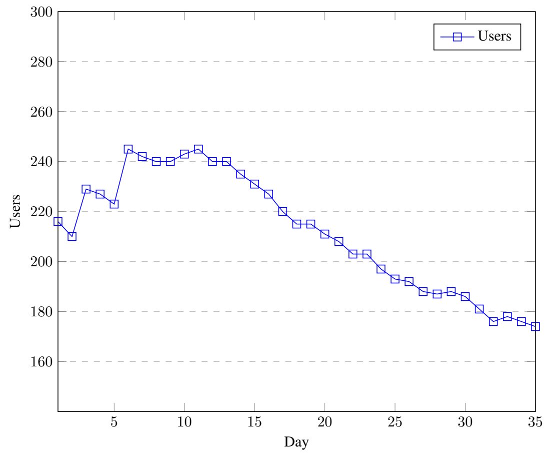


Figure 4: Active users per day

In order to characterize the activity of users over a week, the distribution of the number of points collected by day and by time of day was analyzed. Table IV shows distribution of collected samples per day, showing that Mondays are the days when most locations are received, while on Sundays fewer locations are received. In general, a greater number of locations is received during weekdays.

[5]https://play.google.com/store

Table IV: Collected locations per day of week

Day	Total Points	Average
Monday	20262	4052
Tuesday	18385	3677
Wednesday	19361	3872
Thursday	17833	3567
Friday	18808	3762
Saturday	16834	3367
Sunday	11936	2387

Table V shows the distribution of locations collected per hour during the day. A greater number locations is received in the morning, between 6 and 9 hours, and in the evening, between 17 and 19 hours. This difference becomes more noticeable during weekdays. While during weekends the hours with the most locations are in the night.

Table V: Collected locations per hour of day

Hour	Total	Weekdays	Weekends
00	1947	1293	654
01	1061	424	637
02	1018	414	604
03	834	368	466
04	821	424	397
05	917	621	296
06	5236	4824	412
07	7931	7117	814
08	9518	8586	932
09	7326	5900	1426
10	4710	3353	1357
11	4904	3307	1597
12	6727	4489	2238
13	5636	3925	1711
14	5190	3546	1644
15	6128	4498	1630
16	5868	4418	1450
17	6744	5287	1457
18	10892	9204	1688
19	9639	8018	1621
20	7506	5815	1691
21	6435	4525	1910
22	4163	2933	1230
23	2214	1360	854

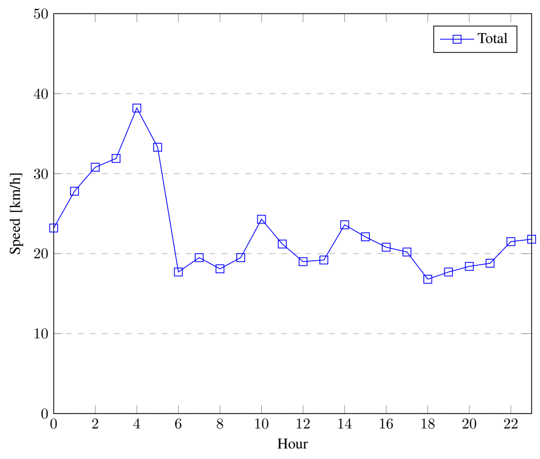


Figure 5: Average speed per hour of day

Table VI: Average speed per hour of day

Hour	Total(km/h)	Weekdays(km/h)	Weekends(km/h)
00	23.2	22.3	24.9
01	27.8	24.9	29.7
02	30.8	33.1	29.2
03	31.9	36.8	28.1
04	38.2	47.3	31.2
05	33.3	29.9	40.4
06	17.7	16.8	28.5
07	19.5	18.9	24.8
08	18.1	16.9	28.8
09	19.5	17.6	27.4
10	24.3	19.1	24.5
11	21.2	20.3	22.9
12	19	17.8	21.4
13	19.2	18.5	20.7
14	23.6	20.9	29.4
15	22.1	20.1	27.7
16	20.8	19.4	24.9
17	20.2	18.6	26,2
18	16.8	15.9	21.7
19	17.7	16.8	22.1
20	18.4	18.5	18.1
21	18.8	18.8	18.8
22	21.5	22.7	18.8
23	21.8	22.7	20.4

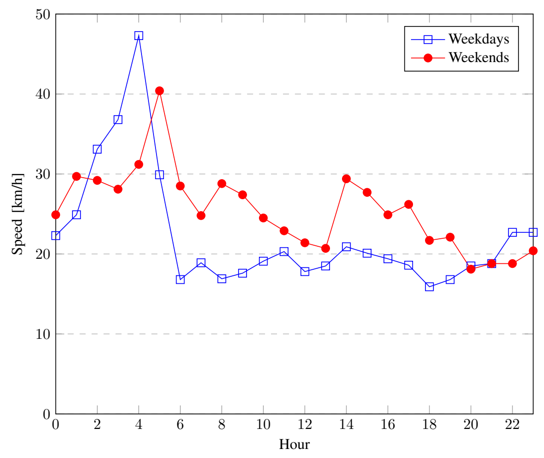


Figure 6: Comparison of average speed per hour

Finally, the average speed of the vehicle was chosen as the measure used to characterize the traffic flow,the highest values are considered best and lowest values are considered worse. Table VI and Figure 5 present the average speed by time of day, showing that in the mornings, the lowest average speed occurs between 6 and 9 am, while in the afternoons, this happens between 18 to 21 hours. A decrease in the average speed is observed during midday.

Figure 6 shows a comparison between the average speed in weekdays and weekends to contrast the difference. During weekdays lower average speed is observed from 5 am until 19 pm, while on weekends, two periods with low average speed values are observed, one between 11 and 13 hours, and another between 20 and 23 hours.

D. Real-time traffic estimation

To approximate the traffic in real time, the average speed of all the points that have been associated with a particular street segment during the last hour is obtained. Thus, traffic information will be available on those streets that have recently been busy. In contrast, for those streets that do not have current information, an approximation of their status won't be presented. To illustrate this, Figure 7 shows the approximate traffic status in real time on a weekday between 9 and 17 hours. It can be noted that the status of street segments will vary throughout the day, according to data collected.

VII. Conclusions

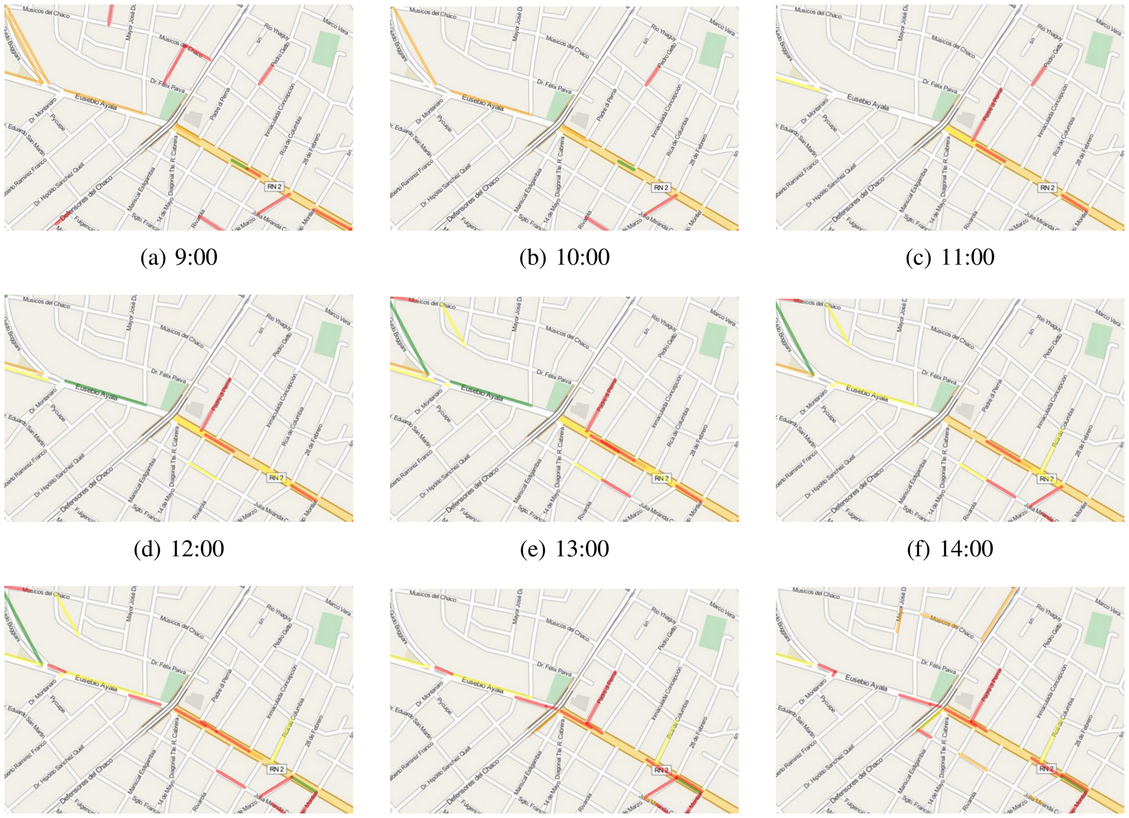
This paper describes the effective implementation of an intelligent traffic information system that may be applied as alternative to approaches that require of installed technology. Moreover, FCD information is obtained by a mobile application that uses an activity recognition approach in order to track the users only when they are within a moving vehicle.

The experimental results and data obtained during the tests show that the system is able to generate a valid approximation of the traffic condition in which it is possible to determine the peak and average speed and are also consistent with the everyday user perception.

It it also important to note that with enough users it is possible to provide a continuous real-time information about the traffic status. Therefore, the feasibility to estimate traffic conditions through smart mobile devices, such as smart-phones and tablets, is demonstrated.

As future work are proposed the following:

- Experiment with FCD information obtained by devices installed in continuous movement vehicle fleets such as buses and taxis.
- Implement other types of applications that make use of the generated traffic information, such as the calculation of less congested roads, intelligent traffic light control, among others.
- Analyze the collected data through data mining techniques to identify critical points that may be subject of improvements.


(a) 9:00 (b) 10:00 (c) 11:00

(d) 12:00 (e) 13:00 (f) 14:00

(g) 15:00 (h) 16:00 (i) 17:00

Figure 7: Traffic along a street segment during the day

References

[1] K.-U. y. W. P. Schäfer, Ralf-Peter y Thiessenhusen, "A traffic information system by means of real-time floating-car data," in *ITS world congress*, vol. 2, 2002.

[2] M. Reinthaler, B. Nowotny, F. Weichenmeier, and R. Hildebrandt, "Evaluation of speed estimation by floating car data within the research project dmotion," in *14th World Congress on Intelligent Transport Systems, ITS Word Congress*, 2007.

[3] L. Giovannini, "A novel map-matching procedure for low-sampling gps data with applications to traffic flow analysis," 2011.

[4] A. Thiagarajan, J. Biagioni, T. Gerlich, and J. Eriksson, "Cooperative transit tracking using smart-phones," in *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, pp. 85–98, ACM, 2010.

[5] P. Zhou, Y. Zheng, and M. Li, "How long to wait?: predicting bus arrival time with mobile phone based participatory sensing," in *Proceedings of the 10th international conference on Mobile systems, applications, and services*, pp. 379–392, ACM, 2012.

[6] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, and J. Eriksson, "Vtrack: accurate, energy-aware road traffic delay estimation using mobile phones," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, pp. 85–98, ACM, 2009.

[7] S. Tao, V. Manolopoulos, S. Rodriguez, A. Rusu, *et al.*, "Real-time urban traffic state estimation with a-gps mobile phones as probes," *Journal of Transportation Technologies*, vol. 2, no. 01, p. 22, 2012.

[8] J. C. Herrera, D. B. Work, R. Herring, X. J. Ban, Q. Jacobson, and A. M. Bayen, "Evaluation of traffic data obtained via gps-enabled mobile phones: The mobile century field experiment," *Transportation Research Part C: Emerging Technologies*, vol. 18, no. 4, pp. 568–583, 2010.

[9] C. E. White, D. Bernstein, and A. L. Kornhauser, "Some map matching algorithms for personal navigation assistants," *Transportation Research Part C: Emerging Technologies*, vol. 8, no. 1, pp. 91–108, 2000.

[10] M. A. Quddus, R. B. Noland, and W. Y. Ochieng, "A high accuracy fuzzy logic based map matching algorithm for road transport," *Journal of Intelligent Transportation Systems*, vol. 10, no. 3, pp. 103–115, 2006.

[11] S. Kim and J.-H. Kim, "Adaptive fuzzy-network-based c-measure map-matching algorithm for car navigation system," *Industrial Electronics, IEEE Transactions on*, vol. 48, no. 2, pp. 432–441, 2001.

[12] Y. Lou, C. Zhang, Y. Zheng, X. Xie, W. Wang, and Y. Huang, "Map-matching for low-sampling-rate gps trajectories," in *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 352–361, ACM, 2009.

[13] L. E. Y. Mimbela, L. A. Klein, and V. D. Clearinghouse, "Summary of vehicle detection and surveillance technologies used in intelligent transportation systems," 2003.

[14] J. Eisner, S. Funke, A. Herbst, A. Spillner, and S. Storandt, "Algorithms for matching and predicting trajectories.," in *ALENEX*, pp. 84–95, 2011.

[15] J. Yuan, Y. Zheng, C. Zhang, X. Xie, and G.-Z. Sun, "An interactive-voting based map matching algorithm," in *Proceedings of the 2010 Eleventh International Conference on Mobile Data Management*, pp. 43–52, IEEE Computer Society, 2010.

[16] J. S. Greenfeld, "Matching gps observations to locations on a digital map," in *Transportation Research Board 81st Annual Meeting*, 2002.

[17] M. A. Quddus, W. Y. Ochieng, L. Zhao, and R. B. Noland, "A general map matching algorithm for transport telematics applications," *GPS solutions*, vol. 7, no. 3, pp. 157–167, 2003.

[18] W. Y. Ochieng, M. Quddus, and R. B. Noland, "Map-matching in complex urban road networks," *Revista Brasileira de Cartografia*, vol. 2, no. 55, 2009.

[19] A. Thiagarajan, L. Ravindranath, H. Balakrishnan, S. Madden, L. Girod,

et al., "Accurate, low-energy trajectory mapping for mobile devices.," MIT, 2011.

[20] S. Fang and R. Zimmermann, "Enacq: energy-efficient gps trajectory data acquisition based on improved map matching," in *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 221–230, ACM, 2011.

[21] T. Litman, "Smart congestion relief: Comprehensive analysis of traffic congestion costs and congestion reduction benefits," 2011.

[22] A. D. May, *Fundamentals of Traffic Flow*. Prentice-Hall, 1990.

[23] B. Budigm, "An algorithm for map matching on incomplete road databases," 2012.

[24] E. Sakic, "Map-matching algorithms for android applications," 2012.