A method to automatically identify road centerlines from georeferenced smartphone data

George Henrique Rangel Costa, Fabiano Baldo

Graduate Program in Applied Computing
Computer Science Department
Santa Catarina State University - UDESC/Joinville

{dcc6ghrc,baldo}@joinville.udesc.br

Abstract. *Digital road maps have gained fundamental role in population's daily life, so they need to be accurate and up-to-date. A viable solution is to generate maps by processing GPS data. However, one of the most challenging tasks regarding this approach is how to extract road centerlines from the cloud of georeferenced GPS points. The literature presents various methods that do it, but none have been found that is prepared for the continuous update of the map and refinement of its roads' accuracy. In this context, the objective of this work is to propose a method to identify road centerlines using an evolutive algorithm in order to generate and update road maps. This work uses as source of data GPS traces collected by smartphones. Although suitable to obtain a large amount of georeferenced data, these devices bring the additional problem of identifying the transport vehicle used along each trace. Preliminary results indicate that the method's performance is satisfactory, with an average variation of 2.95 meters in relation to satellite images.*

1. Introduction

Digital road maps are supporting tools that have gained fundamental role in population's daily life. For this reason it is essential that the maps reflect reality as well as possible, that is, they must be generated from accurate data. Periodic updates are also necessary to adapt them to modifications on the roads. These factors must be taken into consideration when describing a solution to generate maps.

Usually, maps are generated and updated by photogrammetric methods applied on pictures taken by airplanes and, more recently, on satellite images [Jang et. al. 2010]. The results are good, but need manual adjustments in areas that are hard to map, e.g. where treetops block visibility of the road. Another disadvantage is that these maps tend to not be updated regularly, so after some time they might contain inconsistencies in relation to the actual roads. Due to these limitations new solutions are being promoted, and the GPS is one of the options.

By storing the sequence of georeferenced points collected by a GPS receiver, a trace that represents the trajectory of the moving object is created [Braz and Bogorny 2012]. Combining traces from one or more moving objects, it is possible to build a map containing the paths they traveled. Assuming that the moving object is a motor vehicle, the resulting map can be considered a road map.

Collaborative projects such as OpenStreetMap [OSM 2013] follow this idea. They allow users to upload traces and to use them to create or update maps. However, all map

editing is done manually. An automatic solution would be more effective, since it could allow maps to be updated more quickly. Several studies demonstrate the feasibility of this approach, like Brüntrup et. al. (2005) and Cao and Krumm (2009).

One of the main challenges in this approach is how to analyze the collected traces in order to identify road centerlines. Since civilian GPS receivers are not completely accurate the traces form a cloud of points along the roads, and the only way to find their centerlines is by approximation. In this context, a suitable technique may be an evolutive algorithm that can approximate a good solution after many iterations. This technique is part of the area of Evolutive Computing and can be classified into the subarea of search and optimization.

As more traces are collected, the existing road centerlines can be improved and new ones can be found. Thus, a new – updated – map should be generated. The literature contains many map generating methods, but none have been found that is prepared for the continuous update of the map and refinement of its roads' accuracy. This depends on how new are the traces used to generate maps. Therefore, a method to find road centerlines must take into consideration the date when the traces have been recorded.

Another question that raises when thinking about generating maps is how to obtain the georeferenced data, and one possible solution is to use smartphones. The adoption of this type of device has been increasing fast over the years and this stimulates its usage in this work. They have several sensors, including a GPS receiver, all of which can be used to obtain a range of information about how, when and where the users are moving. As the device is generally carried close to its user during the whole day, the volume of data collected tends to be high. On the other hand, this brings the additional problem of identifying the transport vehicle used along each trace.

Based on all the challenges and assumptions presented, the objective of this work is to propose a method to identify road centerlines using an evolutive algorithm in order to generate and update road maps.

The methodological procedure utilized in this work begins with a literature review of related work. In parallel, a system to record and collect data for testing purposes is developed. Based on the knowledge acquired an initial solution is defined. Then, this solution evolves cyclically, with the gradual implementation of improvements and analysis of results. The evaluation of the proposed method is performed comparing the road centerlines with satellite images, where the test scenario is the city of Joinville-Brazil.

The paper is structured as follows. Section 2 presents related work and compares it with the proposed solution. Section 3 details the proposed method. Section 4 presents the evaluation of the results obtained. Finally, conclusion and possible future work are drawn.

2. Other proposals to identify road centerlines

The literature related to generation of maps contains many different proposals on how to identify road centerlines. Brüntrup et. al. (2005) do it from traces from any GPS receiver, but assume that they contain only motor vehicles trajectories. After the client device collects the traces, the server filters them to remove noise, based on limits of speed,

acceleration, and relation between distance and time for two sequential points. After that, it divides each trace into segments and, for each of them, uses a clustering technique based on Artificial Intelligence to identify which points should be added to the map. This method does not depend on an initial map, so it can create maps from scratch. Also, it can add roads to previously existing maps as new traces are collected, but no details have been given about how it updates already existing road centerlines in case there is any change.

Cao and Krumm's (2009) work uses GPS data recorded with in-vehicle GPS loggers. The traces are filtered by the server based on limits of distance, time and direction of movement between two sequential points. After that they find the road centerlines, which are divided by direction of movement. To do that, they developed a technique based on principles of physical attraction and repulsion and applied it to each of the points collected, relating them to all the other points nearby. As in the previous work, this method also does not depend on an initial map. The authors do not discuss about updating the map.

Jang et. al. (2010) collect traces using mobile devices equipped with GPS receivers. Like Brüntrup et. al. (2005), they also assume that all traces contain only motor vehicle trajectories. No steps for filtering the traces have been described. To generate the map, they first divide the area where traces have been recorded in squares of one meter. After that, the points on each square are clustered based on the distance to nearby squares. The clusters are then connected to make the road network, and the analysis of the shape and angle of the streets is used to correct any problems. This step is also not detailed by the authors. As in the previous works, they do not use an initial map. Finally, the authors comment that the method to update the map has not been automatized and, therefore, is inefficient.

Niu et. al. (2011) use smartphones (Blackberry and iPhone) as client devices to collect data, but also assume that all traces contain only motor vehicle trajectories. Their method to identify road centerlines begins by filtering traces based on limits of accuracy and direction of movement. Points with speed equal to zero or repeated points in the same coordinates have also been discarded. Finally, they use a combination of Robust Loess and subtractive clustering. Differently from the previous works, this one depends on an initial map to serve as reference. The authors do not discuss about how to update the map.

Although each work proposes a different method to find road centerlines, there are a few similarities between these solutions. Some are interesting and the present work follows the same ideas. One such example is the independence of initial maps. Only Niu et. al. (2011) use a map as reference, but that is because their objective is to improve existing maps.

Another interesting similarity is related to the filtering of traces to remove noise. Except for Jang et. al. (2010), all other works defined heuristics based on the attributes provided by the GPS, such as speed, acceleration, distance and so on. The present work uses traces collected with smartphones, therefore, the most relevant heuristics are those defined by Niu et. al. (2011) – i.e. filtering based on accuracy, direction of movement, speed and proximity of other points – since they too used this type of device.

On the other hand, the present work approaches some steps of the solution differently. For example, all these proposals assume that the traces contain only motor vehicle

trajectories. This is not true when collecting data with smartphones, so it is necessary to identify the transport vehicle used along each trace.

Another concern is that these works rarely discuss about updating the map. This is an important question because new roads can be created and existing ones might be altered in some way, even if temporarily. It is possible to think that generating a new map as more traces are collected would be enough. However, this is not true, because when these four related works are identifying road centerlines they do not take into consideration the date when the traces have been recorded. Because of that, the result would be a mixture of old trajectories and new ones. Therefore, the present work analyzes the traces collected over time to identify changes and correctly reflect them on the map.

3. Proposed solution

The traces with georeferenced data must be (i) collected and (ii) preprocessed to filter out noise. Then, (iii) the road centerlines are identified by applying an evolutive algorithm. The following subsections detail each of these steps, with emphasis given to the identification of road centerlines.

3.1. Data collection

To test the solution it is necessary to collect traces with georeferenced data. To this end, a service oriented system has been developed. This system is composed of a smartphone application that collects the data and a web application that stores it. The smartphone application runs in devices with Android operating system and is available at <http://bdes.dcc.joinville.udesc.br:100/coletor>. The user decides when to start and stop recording data and no personal information is logged. The collected data are the GPS coordinates, recorded once per second, and the accelerometer oscillation, twenty times per second. That means that each point of the trace is composed of one GPS coordinate and twenty accelerometer oscillations. When the smartphone connects to the Internet the traces are sent to the server, where they are kept stored until the next step begins.

3.2. Preprocessing

Many points of the traces contain noise. These points must be discarded, in order to find more accurate road centerlines. There are several types of noise and each of them can be filtered using different methods. Following, the types of noise identified are presented.

A single trace might contain data from different means of transport, i.e. recorded while the user is walking, riding, running or driving. However, in order to generate road maps it is necessary to use only motor vehicle trajectories. That is, points that have not been recorded while using motor vehicles must be discarded. This can be achieved using many different methods, such as generating a Decision Tree combining GPS and accelerometer data, as proposed by Reddy et. al. (2010).

Many points have bad accuracy or incorrect information that can decrease the accuracy of the whole map, therefore they should not be used to identify road centerlines. One strategy to filter them out is to simply define a limit value for each of those characteristics and eliminate any points that go above these limits. Points too close to each other or with speed equal to zero can be considered unnecessary and should be discarded as well.

Lastly, there are algorithms that can compress traces by removing less significant points, such as the Douglas-Peucker and Opening Window algorithms [Hershberger and Snoeyink 1992; Meratnia and de By 2004]. These algorithms can substantially reduce the number of points of a trace without changing its shape.

On average, more than 75% of the points of each trace are discarded after applying these filters. Although this percentage is high it is important to consider that, usually, there will be many traces passing through the same road, and all their remaining points must be combined to find the road centerline.

3.3. Identifying road centerlines with an evolutive algorithm

After the preprocessing step, the roads are covered with points from all traces that pass through them. Since GPS has variable accuracy, the points are scattered all around the road, in the shape of a cloud. For this reason, this work assumes that none of the existing points is correctly positioned and therefore new points must be created to represent the road centerlines.

For each portion of the cloud that is analyzed, one road centerline is defined. These points cannot be defined deterministically, so it is necessary to use an approximation method. A suitable approach is to use evolutive algorithms. They are based on the concept of evolution by selection of the most adequate individuals after each generation. That is, given a set of candidates, the best ones are selected and used to create new sets.

In this work, each candidate (individual) contains only a geographic coordinate. They are selected according to the results of a fitness function, that weights them considering three values extracted from the points in the same portion of the cloud: (i) recording date, (ii) accuracy and (iii) distance between the candidate and the points nearby. The recording date is essential to correctly update the map, since it helps identify changes on the roads. Combining it with the other values, the candidate selected to be in the road centerline will be the one closest to the highest concentration of recent and accurate points.

The fitness function is composed of equations that define the influence of each of those characteristics (distance, accuracy and date) in finding the road centerlines. Each characteristic can have a different weight, so it is possible to control which has more influence. After calculating the fitness of all candidates, the selection procedure identifies which will be kept for the next generation, which will be discarded, and which will be used to create new candidates through random modifications to its latitude and longitude. That way, it is possible to avoid local maximums [Haupt and Haupt 2004].

This cycle of executing the fitness function and creating new generations is repeated many times. At the end, the best candidate from the last generation is chosen to represent the road centerline for that segment of the road. Algorithm 1 gives an overview of the entire method. It is worth mentioning that all of the method's parameters have been chosen based on tests with a set of 1500 points.

With this contextualization in mind it is possible to detail the method's execution. Each of the q traces collected is composed of n points. The traces and their respective points can be represented as $P_{j,k}(j = 1, 2, .., q; k = 1, 2, .., n)$. After sorting the traces from most recent to oldest and from most accurate to least, the process starts from $P_{1,1}$

(first point of the most recent and accurate trace) and is repeated to every point of every trace.

```
1  Initialize road_centerlines as an empty list;
2  Query database to get all traces ordered by date and accuracy;
3  foreach trace do
4      foreach point P of the trace do
5          if P has not yet been marked as used then
6              Create the S set by querying the database to identify points near P
               and with similar direction of movement;
7              Add P into the S set;
8              Define the domain based on the S set;
9              Create first generation based on the S set;
10             repeat
11                 foreach candidate in that generation do
12                     Calculate the candidate's fitness;
13                 Order candidates according to their fitness;
14                 Create new generation based on the best candidates and the
                   domain;
15             until 60th generation has been created;
16             Add the best candidate into road_centerlines;
17             foreach point P' in the S set do
18                 Mark P' as used;
```

Algorithm 1: Pseudocode of the road centerlines identification method

First, it is necessary to define what is the area considered close to $P_{j,k}$. Experimentally, it has been defined that this area is a circle with 15-meter radius, centered on $P_{j,k}$. The same idea is also used to represent the points in the cloud: their circle's radius is their accuracy and they are centered on their own coordinates. To simplify the explanation, from now on the circle of a point in the cloud will be called its "circle of accuracy".

To find the point that represents the road centerline in the area close to $P_{j,k}$, all n' points with a circle of accuracy that intersects this 15-meter radius circle are identified, no matter from which trace they are. The points with a direction of movement different than $P_{j,k}$ are eliminated, and the remaining ones (n'') compose the set of selected points $S_i(i = 1, 2, .., n'')$ (line 06 of Algorithm 1).

The domain specifies the range of valid coordinates that can be assigned to the candidates created in each generation, and it is defined (line 08 of Algorithm 1) as a rectangle bounded by the coordinates of the four most distant points in S. After defining it, the first generation is created and each candidate is represented as $C_x(x = 1, 2, .., z)$ (line 09 of Algorithm 1). In the first generation z is equal to n''; in every other generation z is fixed to 20. This exception occurs because the first generation is formed by a copy of the full S set, that is, for each S_i there is a candidate C_x located exactly on the same coordinates. This is done to speed up the creation of the first generation. On the other hand, if S has less than 20 points, those will be used as seed to create the points remaining to complete the C set.

After that, the fitness of each candidate is calculated (lines 11-12 of Algorithm

1). For each C_x, its relationship to each S_i is analyzed individually. The final fitness of C_x is the sum of the contributions of each point from S_i, as represented by equation (1). That equation is composed of equations (2), (3) and (5), that calculate the influence of recording time (IT), accuracy (IA) and distance (ID), respectively. Their results are in the range between 0 and 1, inclusive ($[0..1]$). The influence of those characteristics is weighted through the variables named MT, MA and MD, where $MT+MA+MD = 1$. In tests, $MT = 0.4$, $MA = 0.3$ and $MD = 0.3$.

$$FITNESS(C_x) = \sum_{i=1}^{n''} IT(S_i) \cdot MT + IA(S_i) \cdot MA + ID(C_x, S_i) \cdot MD \tag{1}$$

Equation (2) has been created to define the influence of time. The function $T(S_i, S_r)$ calculates the elapsed time (in days) between the recording of S_i and the recording of the most recent point of the S set (S_r). The variable t_{max} represents the maximum elapsed time allowed. In tests it has been used $t_{max} = 90$ days, and points older than t_{max} are removed from S to keep the map updated. This value has been chosen because this work aims to quickly identify modifications made on roads and reflect them on the map. Thus, recently collected traces must have a greater fitness value than older ones.

$$IT(S_i) = \frac{t_{max} - |T(S_i, S_r)|}{t_{max}} \tag{2}$$

Equation (3) defines the influence of accuracy, where $A(S_i)$ is the accuracy of the selected point S_i. A quadratic equation has been chosen because it allows points below a certain threshold (considered good accuracy) to be overestimated while points above it are underestimated. This equation has the following conditions: (i) if $A(S_i) = 0$, then $IA(S_i) = 1$; (ii) if $A(S_i) = a_{lim}$, then $IA(S_i) = a_{limV}$; (iii) if $A(S_i) = a_{max}$, then $IA(S_i) = 0$. In tests, $a_{max} = 20$ meters and $a_{lim} = 10$ meters, while $a_{limV} = 0.75$. With these values, points with really good accuracy ($A(S_i) \leq 5$) contribute a lot to the candidate's fitness value, points with good accuracy ($5 < A(S_i) \leq 10$) contribute not too much, and points with average accuracy ($10 < A(S_i) \leq 20$) contribute little. Applying these values to equation (3), equation (4) is reached.

$$\begin{aligned} IA(S_i) &= \alpha A(S_i)^2 + \left(\frac{-1 - \alpha \cdot {a_{max}}^2}{a_{max}} \right) A(S_i) + 1 \\ where \quad \alpha &= \frac{-a_{lim} - a_{max}\left(a_{limV} - 1\right)}{a_{max} a_{lim}\left(a_{max} - a_{lim}\right)} \end{aligned} \tag{3}$$

$$IA(S_i) = 1 - 0.0025 A(S_i)^2 \tag{4}$$

Lastly, equation (5) defines the influence of distance, where $D(C_x, S_i)$ is the distance between a candidate C_x and a point S_i. A quadratic equation has also been chosen because it increases the fitness of candidate points located near the selected points (within a threshold) while decreasing the fitness of candidate points far from them. This equation has the following conditions: (i) if $D(C_x, S_i) = 0$, then $ID(C_x, S_i) = 1$; (ii) if $D(C_x, S_i) = d_{lim}$, then $ID(C_x, S_i) = d_{limV}$; (iii) if $D(C_x, S_i) = d_{max}$, then

$ID(C_x, S_i) = 0$. In tests, $d_{max} = 50$ meters and $d_{lim} = 10$ meters, while $d_{limV} = 0.9$. With these values, points with a distance smaller than 10 meters contribute a lot to the candidate's fitness, and points farther than this gradually decrease their contribution. This equation is complemented by the previous one (influence of accuracy) because it is not enough for a candidate to be near many points, they must have good accuracy too. Applying these values to equation (5), equation (6) is reached.

$$\begin{gathered} ID(C_x, S_i) = \beta D(C_x, S_i)^2 + \left(\frac{-1 - \beta \cdot {d_{max}}^2}{d_{max}} \right) D(C_x, S_i) + 1 \\ where \quad \beta = \frac{-d_{lim} - d_{max}\,(d_{limV} - 1)}{d_{max} d_{lim}\,(d_{max} - d_{lim})} \end{gathered} \tag{5}$$

$$ID(C_x, S_i) = 1 - 0.0075 D(C_x, S_i) - 0.00025 D(C_x, S_i)^2 \tag{6}$$

After calculating the fitness of each candidate, the two best ones (highest values) are kept for the next generation and 18 new candidates are generated through small random modifications of the eight best results. A new candidate will be 2.5 meters away from its seed's location, at most (lines 13-14 of Algorithm 1). The same process (equation (1)) is applied to this new generation and so forth until 60 generations have been calculated. At this moment, the candidate with better fitness is considered the road centerline in the area close to $P_{j,k}$ (line 16 of Algorithm 1).

The next area where the road centerline would be calculated is the area close to $P_{j,k+1}$. As the GPS records one point per second, even after the preprocessing the chance of $P_{j,k+1}$ being near $P_{j,k}$ is very high. If this indeed happen, it means that $P_{j,k+1}$ has already been used to calculate the road centerline in the area close to $P_{j,k}$, because its circle of accuracy intersected the 15-meter radius circle centered on $P_{j,k}$ (as explained during the creation of the S set). Thus, it is not necessary to calculate the road centerline near $P_{j,k+1}$. The same thought is valid to every point included in the S set of $P_{j,k}$, so they are "marked" to not be used again (lines 17-18 of Algorithm 1). Therefore, before calculating the road centerline near any given point $P_{j,k}(j \in q; k \in n)$ first it is necessary to verify if it has been marked yet (line 05 of Algorithm 1). If so, then $P_{j,k}$ is ignored and the same verification is done to the following points. After processing every point of all traces collected, all road centerlines are found.

4. Results Assessment

As the aim of this work is to represent the road centerline based on a cloud of points collected by smartphone's GPS, its evaluation consists in comparing the resulting maps to satellite images. The test scenarios are regions of the city of Joinville-Brazil that contain complex road structures. These places can indicate the efficiency of the proposed method. If it works well for these scenarios, by induction, the same is valid for regions with simpler road structures.

The first test scenario presented in Figure 1a contains a region where two highways cross. Based on the collected data showed in Figure 1b, the difficulties to find road centerlines are related to three factors: (i) two-ways on each highway; (ii) one highway

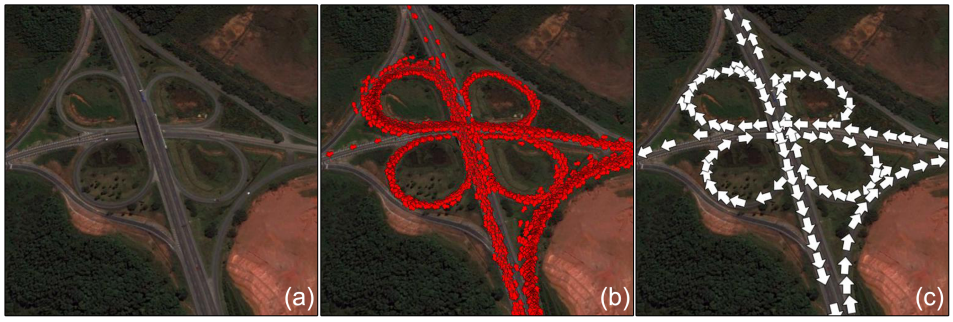

Figure 1. First test: (a) satellite image, (b) points after preprocessing, (c) road centerlines

with two lanes in each direction; (iii) cloverleaf interchange causing multiple join and disjoin intersections.

As can be seen in Figure 1c, in order to have a more realistic result, it is important to distinguish the direction of movement of each roadway, otherwise the road centerline will tend to the side of the roadway with more traffic. In the method developed, the roadways are distinguished before executing the evolutive algorithm. The method does not try to identify the number of lanes in a roadway and only returns one centerline. Therefore, the centerline will be closer to the lane with more traffic. It is assumed that traffic is almost equally distributed across all lanes, so the final result should not be affected. Besides that, the roadway differentiation also identifies when a road has a join or a disjoin point, as well as when two roads overlap. Figure 1c shows that the proposed method distinguishes the centerline of the two highways and the cloverleaf interchange, without mixing them.

The second test showed in Figure 2 has been performed on a large roundabound that has access points to two universities and to various places of the city. From all test scenarios, this one is considered by the authors as the most complex because of: (i) roads with multiple lanes; (ii) proximity of the roads, with similar or different directions of movement.

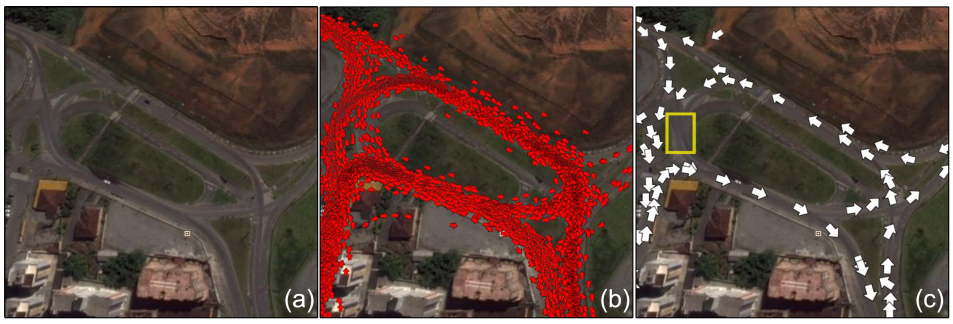

Figure 2. Second test: (a) satellite image; (b) points after preprocessing; (c) road centerlines with incorrectly mapped area highlighted

Once again, the differentiation by direction of movement is essential to achieve that result. Unfortunately, in some cases the parameters have not been enough to differ-

entiate nearby roads. As example, in Figure 2c there is a road – highlighted by the yellow rectangle – without centerline, but in Figure 2b it is possible to see that points have been collected in that area. This happened because the direction of movement of the road to the left is similar and they are close to each other, so the method treated them as two lanes from the same roadway. In other tests, the same problem is observed in highways with parallel roads nearby. It is believed that this problem can be fixed by tweaking the parameters of the preprocessing step and evolutive algorithm.

As can be roughly perceived in the two test scenarios presented in Figure 1 and Figure 2, the difference between the road centerlines and the satellite images in both tests is small. However, it must be taken into account that that the baseline images provided by Google Earth might be a little shifted from their correct position. This means that it is not possible to confirm whether the road centerlines or the satellite images is more correct, but at least it is possible to observe that there are no considerable differences.

In order to collect some statistical evidences the average perpendicular distance between the road centerlines and the road at the satellite images has been calculated using 100 randomly selected points. The result is an average distance of 2.95 meters.

The method has been implemented using the Python programming language. The DBMS chosen is PostgreSQL with PostGIS extension to support geographical data. The database scheme contains two tables. One stores the filtered traces and that is accessed by the method to retrieve information about points being processed. The other stores the circle of accuracy of each point and that is used to find the points that fill the S set (as explained in Section 3.3). The method's output is a Keyhole Markup Language (KML) file that is opened with the Google Earth program, used to evaluate the results and calculate the average distance.

To process all data collected between January 27 and June 15 2013 (6475 Kilometers collected in 301 hours; 4237 files with traces and a total of 966698 points), the Python implementation took four hours and 42 minutes. The computer used on the tests had an Intel Core i5 2400 3.1GHz, 10GB RAM and Windows 7 operating system. This execution time can be shortened, because the method has been implemented without any optimizations and without following the parallel programming paradigm.

5. Conclusion and future work

Digital maps are becoming increasingly more important. However, producing and keeping them updated is a complex problem. Road maps can be created using georeferenced data provided by GPS receptors, and smartphones can facilitate its collection. Due to their popularization, these devices make possible to collect a large amount of traces, which can be used to generate a complete map of a city.

Related work propose methods to generate maps, but few of them use data from smartphones. This can be partially explained by the complexity of using data from a device that is not dedicated to such task. This makes necessary to use filters to identify and discard traces that are not motor vehicle trajectories. Besides that, no work has been found that take into consideration the influence of the traces' date of recording to the process of finding road centerlines. This is essential to automatically update the map, otherwise the result would be a mixture of the old path with the new one.

This work tackles one of the main challenges of automatic generating road maps, which is identifying road centerlines. The proposed solution assumes that an approximation method can be used to create points that better represent the centerline of a road. This approach has been implemented applying an evolutive algorithm that analyzes recording date, accuracy and distance between the points candidate to center of the road and the points collected with smartphones.

The method's parameters have been refined through many test cycles, and the results obtained showed little difference to the satellite images, with an average difference of 2.95 meters. However, it is still possible to optimize the parameters to achieve better results. It would be interesting to compare this method and the proposals from related work using the same dataset, but that was not possible because their implementations could not be found publicly.

Regarding future work, it is suggested to study ways to improve collected traces' reliability. For example, by applying a method to infer that the accuracy of a certain point is better than what the GPS informed. If this would be possible, fewer points would be eliminated during the preprocessing step and the evolutive algorithm could find more accurate road centerline results. In this context, the Kalman Filter [Grewal and Andrews 2001] seems to be a worthy approach to be investigated. Another future work would be to define an update policy. Traces are not collected everywhere with the same frequency. Therefore, different parameters should be used for each region. For example, in downtown only points with good accuracy could be used, since more traces are collected there. On the other hand, in rural regions points that are not so good could be used, since traces are not collected so frequently there. One last future work would be to analyze the data collected to extract more georeferenced information, such as the location of semaphores or potholes, so as to make the map more complete.

6. Acknowledgements

This project is supported by a research scholarship granted by the Brazilian Coordination for Enhancement of Higher Education Personnel (CAPES).

References

Braz, F. J. and Bogorny, V. (2012) Introdução a trajetórias de objetos móveis. Editora Univille.

Brüntrup, R., Edelkamp, S., Jabbar, S. and Scholz, B. (2005) "Incremental map generation with GPS traces". In: Proceedings of the 8th International IEEE Conference on Intelligent Transportation Systems, p. 413-418. IEEE.

Cao, L. and Krumm, J. (2009) "From GPS traces to a routable road map". In: Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS '09, p. 3-12. New York, USA: ACM Press.

Grewal, M. S. and Andrews, A. P. (2001) Kalman Filtering: Theory and Practice Using MATLAB. John Wiley & Sons, Inc. 2. ed. 410p.

Haupt, R. L. and Haupt, S. E. (2004) Practical genetic algorithms. Wiley-Interscience. 2. ed. 253p.

Hershberger, J. and Snoeyink, J. (1992) ”Speeding Up the Douglas-Peucker Line-Simplification Algorithm”. In: Proceedings of the 5th International Symposium on Spatial Data Handling, p. 134-143.

Jang, S., Kim, T. and Lee, E.(2010) ”Map Generation System with Lightweight GPS Trace Data”. In: International Conference on Advanced Communication Technology - ICACT, p. 1489-1493.

Meratnia, N. and de By, R. A. (2004) ”Spatiotemporal Compression Techniques”. In: Lecture Notes in Computer Science, 2992, p. 765-782. Springer-Verlag.

Niu, Z., Li, S. and Pousaeid, N. (2011) ”Road extraction using smart phones GPS”. In: Proceedings of the 2nd International Conference on Computing for Geospatial Research & Applications - COM.Geo ’11. New York, USA: ACM Press.

OSM. OpenStreetMap (2013) ”The Free Wiki World Map”. Disponível em: <http://www.openstreetmap.org>. Acesso em: 05 ago. 2013.

Reddy, S. et.al. (2010) ”Using mobile phones to determine transportation modes”. In: ACM Transactions on Sensor Networks, 6(2), p. 1-27.