Dynamic GPS Maps

Cosmin VARLAN
"Alexandru Ioan Cuza" University of Iaşi
Bulevardul Carol I, nr.11, RO-700506 Iaşi
vcosmin@info.uaic.ro

***Abstract*—GPS are usually built by only one company that provides the software and the maps as a single solution. Because the maps are made especially for that kind of software and because the format is not made public, to update such a map additional files (from the producer) are needed. Even though, the producer is not always updating all the roads as fast as they appear or as fast as their properties are changing.**

This paper is proposing a new method of creating maps which provide the dynamism needed by every GPS: the maps are created by every driver, collected on the same website, parsed and modified and redistributed to the other users that are interested in vectorized maps. This way the maps are refreshed every day and that could be done even in real time.

***Index Terms*—Dynamic Maps, Global position System on Play Station Portable, Intelligent Transport Systems, Recording maps in vectorized format, Social network.**

I. INTRODUCTION

Intelligent Transportation Systems are defined as those systems utilizing synergistic technologies and systems engineering concepts to develop and improve transportation systems of all kinds.

In road transportation, the main issues of ITS are: improve the traffic safety, increase the mobility, minimize the impact on the environment, ensure the inter-operability and integration in the European transport networks, efficiently manage the entire transport process [1]. To increase the mobility, devices inside cars must communicate between them or between them and an electronic infrastructure implemented in such a way that cars would easily access it.

My approach on the problems is based on dynamically build map which will be displayed on GPS inside cars and used for some computations like shortest way, fastest way, less occupied road etc.

II. MAP DATA REPRESENTATION

The GPS is a tool which can provide enough data for a driver so he can't get lost in the traffic. Usually, the software part of the GPS has two components: the software interface and the map. Both of the components are developed and updated by the company that produces the software.

The interface's job is to show to the driver his accurate position on the map. This is done by using the data from 3 or more satellites and by deducing his exact location through triangulation method. While driving, the user must not be distracted by the GPS, and because of that, the interface must be easy-to-read or to offer information in such a way that the driver would understand them without paying any special attention. I could say that the GPS interface must be implemented in the same way as the accessibility methods are implemented in most operating systems: devices must have big screens and audio capabilities.

To find the best way from current location to the destination point, the GPS must first find his location on the map and second it must compute the correct path, computation that is usually based on graph theory. For that theory to be applied, it is necessary to have a correct representation of the map: the intersections must be considered the nodes of the graph and the roads must be arcs.

The GPS software can have the map represented in two ways: as a raster image – the image is in BMP or JPG format or as a collection of vectors.

The first type of representation is not a very useful because it can't be parsed and understood by the GPS's software. This kind of maps is good only if the user wants to know where it is but can't be used for computations as finding paths or locations (to get from one point to another the software must "understand" the map and to understand a raster format map means to parse it into vectors – and this job is not suitable for small devices with low frequency processors as the ones build in cars or in GPS).

The second type of representation is used by most GPS software and is better because the map is already represented as a directed graph (it is necessary to represent it as directed graphs because one way streets must be also displayed).

The problem that still remains is how to build maps in the vector format?

III. BUILDING VECTORIAL MAPS

There are several ways to create a vectorized map:

First method to is by making a vectorized representation of a real map: the real map (that can be on a piece of paper or as an accurate image taken from satellite) is scanned as a raster image. Although there are software that can transform the image in vector format there is nothing that so good so it can provide the correct map. That's why people are paid to make the vector map from the raster format. The algorithm that is used is: a scalar image is imported and shown on the screen; the positions of the corners of the map are introduced: the upper-left and the bottom-right (usually). The user then clicks on two intersections on the map and the raster to vector tool will know that between those two points there is a road (or a vector). The correct latitude and longitude of the intersection is easily calculated using the same method as the GPS: knowing the distance from the corners of the map (4 different points), the position where the map was clicked can be unique identified. If the two points representing the intersections are established, what remains is make a correct representation the road (because the road is not necessarily a straight line between the two

points). This is realized by clicking some more points on the path between intersections usually where the road has curves. This approach even though is not too accurate is successfully used in most of GPS because it provide sufficient information so the software can find the best way and tell the driver where to go when he arrives at the next intersection.

Second method is getting the data with especially equipped cars that can record the route of their pass. Those cars have some computer systems attached to several GPS sensors and while driving, the software is getting and recording information from those devices (the GPS provide data like longitude, latitude, altitude, moment of the day, the direction of the car, etc.). For the car that builds the map only the longitude and latitude are relevant (actually only those data are recorded).

GPS devices are not totally accurate – usually the error is between 3 and 5 meters. Using more than one GPS and driving very slow, the car can get more accurate positions and recording those positions is able to recreate the road in a vectorized format.

In the end, users are only adding some more data like names of the streets, positions of the intersections or POIs[1].

There are some inconveniences with the two approaches explained: the first one is not very accurate and even though the raster image can be in very big resolution, the data that is recorded is the basic as needed – the employee that draws the vectors can't be as accurate as the real path (for example which is taken by the second method). More than this, this way of building a map is time consuming and requires a big number of employs that must be paid (which will increase the cost of making a map). The second approach is better somehow but there are some problems with this approach too: a car that is especially equipped for getting data from the field is very expensive, the car must travel slow (to get an accurate path) and will also be time consuming, the car will not be able to get on all existing small streets (or on streets temporally closed), if after mapping new streets are build, the car must return and map those one too.

What is observed is that both methods are not so good responsive at changes. Both are expensive (that makes a vectorized map really expensive too – 30.000e for Europe). The real problem lies in the fact that the map is made in one pass. A new vector-map building method is necessary:

The third method (our way) is to convince the driver in every car to create the map for us: the GPS is getting the correct position so it can know what is the current location and while driving, it continues to get data like longitude, latitude, altitude etc. If those data can be used to get the location, then the GPS device can also save them, creating this way a path-of-points that can be treated like start and end points of vectors. This way the map is not generated by especially equipped cars but from every-user cars: if a driver can get to a certain point with his car then it is a fact that the path the car tripped on is a road (of course, the roads can be differentiated from car to car: for example if the driver has an off-road car, it can select that kind of car from his GPS device and only maps taken from other off-road cars will be displayed).

Is the third method accurate enough? Well, if only one car passes a road, chances are that the GPS provides data that contains the 5m-error we were talking about but if more cars are using the same route and record the route while passing by, creating the media of all their data will provide the accurate road points and directions.

The software I create is platform-dependent and uses the Play Station Portable (PSP) as a GPS. Although the ideas can be implemented in other devices as well, I chosed this platform because it has the capabilities to deploy a good user interface and it really needs a software for the recently build GPS sensor (also built by SONY).

Figure 1. GPSP running.

The GPS sensor can take the current position from time to time. In the case of GPSP this data are token once every second. Data are recoded internally as long XML files containing only the points and their characteristics with no additional information:

```
<?xml version="1.0"?>
<data>
    <point>
        <lat>47.168976</lat>
        <long>27.588182</long>
        <alt>112.800003</alt>
        <spd>48.189041</spd>
        <course>328.089996</course>
        <sat>5</sat>
        <qual>1</qual>
    </point>
    <point>
        <lat>47.169079</lat>
        <long>27.588083</long>
        <alt>113.300003</alt>
        <spd>48.189041</spd>
        <course>328.089996</course>
        <sat>5</sat>
        <qual>1</qual>
    </point>
    ..................
</data>
```

Every point recorded contain information about the speed of driving in that certain point, the angle of the car, the number of satellites and the quality of the signal. Although the data is recorded, it only contains linear points and has no other information attached: if this is re-loaded into GPS as a map, it will not know where an intersection is or the names of the roads.

[1] Point of Interest

The recorded data will be uploaded on our web-site where will be processed by special algorithms that can (for example) suggest the positions of intersections or create the media between the current uploaded path and the ones already uploaded by other users.

IV. USER INTERACTIONS AND FINAL MAP REPRESENTATION

Why would users upload maps?

That issue was one of the main concerns when we started to implement the dynamic map creation tool. The motifs are that we can provide a map that is better than any other existing maps, because this is implemented on a (somehow) cheap device with good properties that can be used in many ways and because nobody implemented software for GPS on Play Station Portable (except the software that comes with the original GPS sensor which is made for China only).

The software is provided for free for every user but from time to time it will requires for the user to upload the recorded information or (if it does not have any recorded information) to add data to some paths uploaded by other users. If the user complies with the software request, this will allow further use of it; if not, the software will display a message asking the user to send data to our site. After sending the data, the website will provide a new code that will be built based on PSP's MAC address and the data the user sent and this code once introduced on GPSP will extends application's life for a certain number of days (depending of what user has done for the map) (See also [2]).

The parsing other user's data implies correct positioning of intersections, introducing the names for streets or adding Points of Interest. The information added by a user is not immediately spread, it will check with the data introduced by other users.

The data that a user will upload can't be home-made because the name of the file will also encrypt some unique characteristics of device (as MAC address) and some information about the file (certain bits of control etc.).

Every time a user logs in, he will be able to download the map as it was build until that moment and to upload it on his own device.

The final map is build in two files, both in format XML (we used the XML file format so other users will have access to our data). The first XML file contains information only about intersections. Every Intersection will have a unique ID and in the XML file that contains intersections, will be represented as that:

```
<?xml version="1.0" ?>
<intersection id="1">
  <coordinates>
     <longitude>27.606253</longitude>
     <latitude>47.137280</latitude>
  </coordinates>
  <neighbors>
    <inter id="3" distance="756"
    str_id="1" str_name="Bd. Indep" />
    <inter id="2" distance="6533"
    str_id="2" str_name="Bd. Carol" />
    <inter id="7" distance="5092"
    str_id="3" str_name="E53" />
    ............
  </neighbors>
  ............
</intersection>
```

Every intersection will be a node in a graph. His attributes are the coordinates (longitude and latitude). From one intersection – node, the software must know what options the driver has: what are the intersections he can reach? The other nodes the user can reach from the current one are represented as neighbors and identified by their unique ID. To get to the destination neighbor, it will be used a street with a certain ID (that will be found in the streets file) and also data as the name of the street and it's distance are added as attributes. Also the distances and the name can be also taken from the streets file the data are redundant in the intersection file. The reason is simple: this way, the GPS will not be forced to upload the entire streets file onto memory when the best way will be calculated – distances can be taken from this file too (or if the user wants to display the route, names of the streets will not be taken from streets file).

The street file is simpler and does not helps any calculations for the software, it is more a representation as point-by-point for every street. Although we recorded more information, we will provide less in the street representation:

```
<?xml version="1.0" ?>
<map name = "Iasi">
   <street id = "1" length = "6533"
                      name = "Bd.Carol">
      <point>
         <lat>47.137280</lat>
         <long>27.606253</long>
         <spd>1</spd>
      </point>
      <point>
         <lat>47.137398</lat>
         <long>27.606125</long>
         <spd>33</spd>
      </point>
      ............
    </street>
    ............
</map>
```

The last file which is not important for the way the map is represented is the Point of Interest List. This is also a file in XML format that has information about location and the nature of the POI (e.g. Hospital, Theatre etc.). Example of this file:

```
<?xml version="1.0" ?>
<poi_list>
  <poi id="1">
     <coordinates>
        <longitude>27.826153</longitude>
        <latitude>46.917055</latitude>
     </coordinates>
     <type>
        Hospital
     </type>
  </poi>
  ............
</poi_list>
```

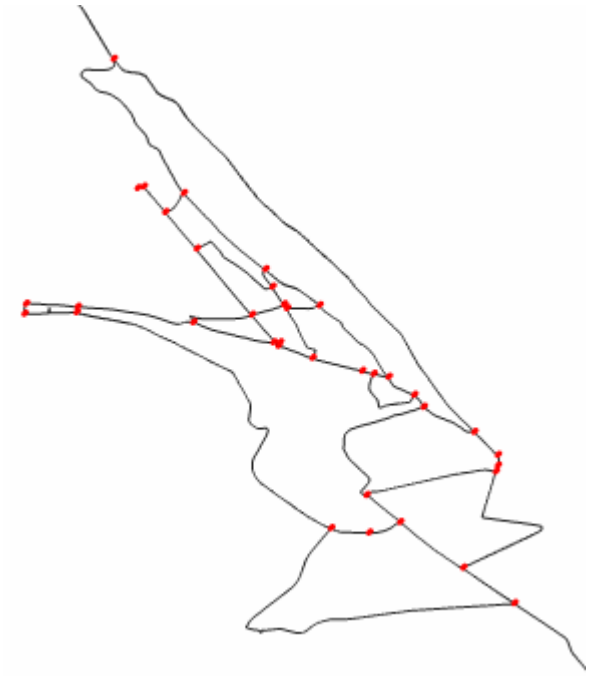

Figure. 2 Web Application – Map of Iasi, Romania.

V. DYNAMIC MAPS

One of the problems that appeared in both other two ways of creating vectorized maps was the high cost of the map. The method I proposed in this paper is totally money-less (except for the money the user invests in buying the PSP device / GPS sensor but the money they invest are less than the cost of a dedicated GPS hardware).

The other important problem was the map was made in one pass and nobody tries to correct the data. With this (third) approach, the data is always rebuilt and new streets are mapped as soon as a single car that records route passes by.

The user always uploads data; this is good to create new roads and update the ones that already are in our database so they can be more accurate – so even the information exists in our database, it will still interest us. If, for example, a certain street is no longer used by users (because of repairs for example) our system will update its knowledge about that street and the GPS will not advice drivers to go that way and will try to find a alternate way. This is actually the method for deleting streets: if no users use that street for more than one week, the street will not be exported anymore in the final map (the street is kept in DB but not shown to the GPS). If the street is in function again, it will fast be added because the system already knows about it.

Streets are exported with additional information as the speed in every of its points. On the GPS this information is very useful: the device could calculate the times needed to go from one point to another better. The roads are displayed on GPSP in two colors: red and green using gradients. The portions where the street is represented in red mean that the driver can't go very fast; the green indicates faster alternatives.

Based on the two last ideas, some other can be implemented: the intersections file will not contain only data as the name and the length of the street but also data about traffic analysis. Even if is not obvious, those data can be obtained from the ones that the user gave us: the file is saved at a certain point in time. Knowing the final time-stamp and the number of points, the web application can find the available speed and time passing of a street for each moment of the day, this way implementing a road-load balance on all available streets (of course, the driver can choose through more alternatives).

The GPS will be able to indicate the path from source to destination and not only the shortest one (from the point of view of distance) but also the cheapest one, the fastest one and will not include the roads that are closed.

VI. FUTURE WORK

In future we have 3 main goals:

First one is to record the temporal information: for example what roads must be used at rush hour (noon) when almost all roads are crowded. Other useful temporal information can be used for the GPS to learn in which days a road is closed (in weekends or on some special days).

The second is building a 3D interface (because we are also recording the altitude data). The 3D data will not serve only in representation of the map but also will be introduced to calculate the amount of gas that a car consume (a car consumes more fuel when it has to climb and almost nothing when it has to descend) – of course, altitudes don't change a lot so this part will not be really dynamically.

The third goal is making the GPSP communicate in real time with other devices. The GPSP will be connected during travel to a cellular phone that will send to a server the entire route that is going to be driven on, this way registering to any information that are usefully: if a big accident has happened on his way he can be redirected or if radars are discovered by other cars, that can be an type of announcement too (although illegal in some countries).

REFERENCES

[1] Intelligent Transport Systems Romania, Available: http://www.its-romania.ro/en/despre/index.html

[2] Luis Von Ahn, "Human Computation", Google Talks, July 06 2006: http://video.google.com/videoplay?docid=-8246463980976635143