Geographic Information Technologies for analysing the digital footprint of tourists

Toni Hernández, Rosa Olivella, Josep Sitjar, Lluís Vicens
University of Girona – SIGTE
Pl. Ferrater Mora, 1
17071 Girona (Spain)
info@sigte.udg.edu

Abstract

As part of a study on the use of the city by visitors, this paper discusses the technical solution adopted in order to analyse and exploit the geo-digital footprint generated by the said visitors using GPS devices.

1 Background

For the purpose of analysing the use of the city by tourists, the tourism research group (INSETUR) of the University of Girona planned a project in the city of Girona that involved the collection of movement data and its subsequent analysis using geospatial techniques applied by the SIGTE (the GIS Centre of the University of Girona), the results of which are presented in this document.

The methodology used by the research group to collect this data was a GPS device provided by the Tourist Office that visitors had to carry on their person throughout the one-day visit of the city. At the end of the visit, the tourists returned the device and answered a qualitative questionnaire.

The project has collected tourist movement data over the course of 9 months, generating a total of 1,339 tracks or 4,752,804 waypoints and 1,339 questionnaires. This paper describes the GIS technical solution that has been adopted in order to carry out the geospatial analysis of this data.

2 Introduction

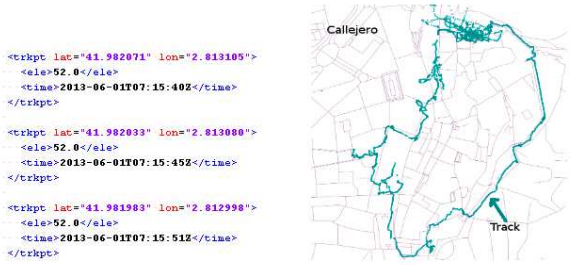
In order to fulfil the stated purpose, the research group carried out an analysis to identify the city streets most visited by tourists and the capacity of attraction of certain heritage elements.

The term "city" in this case refers to the historical centre, clearly delimited by natural elements (river) or anthropic elements (walls), which has been established as the area of study.

In order to carry out this study, over 1,339 GPS tracks have been captured thanks to the collaboration of tourists. These files (in .gpx format) contain, with a certain margin of error, all the information on the itinerary completed by the visitors to the city (route, distribution of time along the route, etc.) and form the basis of this project.

A track is an ordered list of points. The coordinates (latitude and longitude) of each point are known, along with the date and time when each coordinate was captured.

The image below shows a fragment of the track file in gpx format. The contents of the track file are displayed on the left and the graphic representation of the track on the city street map is displayed on the right.



3 Methodology

3.1 Tools

Having evaluated several work tools (both proprietary and open source), the research team has opted to import the files in .gpx format into a **PostgreSQL/PostGIS database** and to

use **PostGIS spatial tools** to analyse the information contained in the .gpx files on the basis of **SQL sentences**.

3.2 Technical challenges

The main challenge of the project consists of breaking down each track into a list of streets (or street arrays) along which each track passes. As such, it will be possible to know the sections of the streets through which each track passes and the length of time invested in each of these sections.

The area of study itself poses a challenge, since the typical dim, narrow streets of Girona old quarter are susceptible to poor satellite coverage [1]. It is therefore necessary to correct the tracks generated in order not to lose too much data for the subsequent analysis.

This challenge will consist of structuring the data in such a way that the planned analysis is both possible and agile.

All of the processes carried out are listed below.

3.3 Importing the GPS data

In order to import the .gpx files, the ogr2ogr tool has been used [2]. From a command console the following command is keyed in to import the .gpx track file into the track table within the TURISMO database.

ogr2ogr -f "PostgreSQL" PG:"host=localhost user=postgres port=5433 dbname=TURISMO schemas=originals password=contraseña" track.gpx -overwrite -lco GEOMETRY_NAME=geom track_points -nln "track"

This command uses the Postgres username and password to connect to the TURISMO database. Once connected it generates a new track table with all the information contained in the .gpx track file.

gid integer	time character varying(254)	lat double precision	long double precision	fecha character varying(254)	hora character varying(254)	geom geometry(Point)
1	2013/03/13 9:41:00	485399.529832	4648107.63786	2013/03/13	9:41:00+00	01010000201F759
2	2013/03/13 9:41:02	485398.275501	4648109.36234	2013/03/13	9:41:02+00	01010000201F759
3	2013/03/13 9:41:03	485392.842089	4648147.86035	2013/03/13	9:41:03+00	01010000201F759
4	2013/03/13 9:41:04	485392.452281	4648144.58313	2013/03/13	9:41:04+00	01010000201F759
5	2013/03/13 9:41:05	485391.753827	4648140.47258	2013/03/13	9:41:05+00	01010000201F759
6	2013/03/13 9:41:06	485390.931959	4648138.78613	2013/03/13	9:41:06+00	01010000201F759
7	2013/03/13 9:41:08	485390.044762	4648137.80044	2013/03/13	9:41:08+00	01010000201F759

The image above shows a fragment of the contents of the track table after it has been imported. The *gid* field functions as the primary key and serves to identify unequivocally each point in the table. This attribute will subsequently be used in new SQL sentences. The geometry of the points is found in the *geo* column, while the *time* column contains both the date and the time at which each track point was captured.

The bulk import of the .gpx files, of which there are over 1,600, is carried out through a command file (.bat). This command file is generated from a script that reads the contents of a folder (where the tracks are located) and generates an import sentence for each .gpx file.

Once each track is imported it is necessary to convert it from geographic coordinates (GPS) to projected coordinates (UTM). This step is obligatory in order to carry out a subsequent Snap operation indicating a tolerance in metric units (corresponding to the projected coordinates).

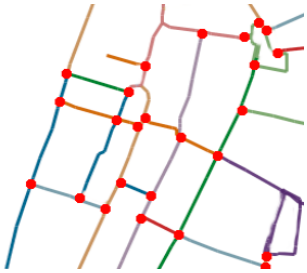
With the following SQL sentence a new table is created with all the initial columns, plus the geometry contents re-projected to the ETRS89 coordinate reference system (srid = 25831).

create table track_reproyectado as
select st_transform(geom,25831) as geom_reproyectada, track. from track;*

3.4 Street map

In order to carry out this operation it is necessary to import a street map of the area of study into the database. The main characteristic of this street map is that every street is split into various sections. Each section of each street is represented by a specific spatial entity. In other words, the number of sections into which each street is split is equal to the number of intersections that this street has with other streets. By splitting the streets in this way it is possible to detect the specific street section along which each track passes, while at the same time distinguishing the sections of the same street along which the track does not pass.

The image below displays the points of intersection between streets (marked in red). The various sections of the same street are displayed in a single colour.



The shp2pgsql tool has been used [3] (included in PostGIS) to import this shp format street map into the database.

From a command console we execute:

Shp2pgsql callejero_shape tabla_callejero > callejero.sql

This command will generate the callejero.sql file with the contents of the .shp file translated into SQL expressions. The next step consists of executing the callejero.sql file within the TURISMO database.

The file can either be executed from pgAdmin or, from a command console, by using the PostgreSQL command line interactive client, called psql, as shown below.

psql -d TURISMO -U postgres -W -f callejero.sql

This command will request the password of the corresponding user (-U).

3.5 Assigning the track points on the street map

Each point located on the track of the GPS device must be assigned to the nearest street. As such we can generate a list with the names of the streets along which the track passes. Only the track points located no more than 7 metres from a street have been assigned; the remaining track points have been discarded.

In order to assign each track point to a street the following SQL command has been executed, inspired by Paul Ramsey's post [4]:

```
create table track_callejero as
      select distinct on (punto_id) c.gid as calle_id, tp.gid as
punto_id
         from track_reproyectado tp
          inner join callejero c on st_DWithin(tp.geom, c.geom,
7.0)
      order by tp.gid, st_Distance(c.geom, tp.geom);
```

This command calculates, for each track point, which streets are less than seven metres away. If a track point has more than one street located less than seven metres away, it places the streets in order so that the nearest street appears in first place. Finally, the distinct (*punto_id*) clause indicates that we only wish to obtain one street for each point. The fact that the streets have previously been placed in order means that each track point is assigned exclusively to its nearest street.

The result of the sentence above is a new *track_callejero* table that lists each track point together with the nearest street. The matching of track points and streets is carried out on the basis of the alphanumeric attributes (gid) that identify each point and each street. These 'gid' attributes are renamed *punto_id* and *calle_id*, respectively, in order to facilitate their interpretation.

calle_id integer	punto_id integer
339	1
350	5
350	6
350	7
350	8
350	9
350	10

The streets from the above list for which fewer than five track points have captured have been omitted from the final result. As such we have omitted certain sections that are not very representative of the itinerary followed by the track.

3.6 Time variable

Having obtained the streets along which the track passes we can also find out the order in which the streets have been visited and the amount of time invested in each street.

In order to extract this information it is necessary to create a PL/PSQL function called 'itinerary'. The result returned by this function is an array data structure. Each element of the array contains three variables: *calle_id, punto_id* and *contador*, which is the variable used to count the number of times the same track passes along the same street.

```
CREATE OR REPLACE FUNCTION itinerario(_tbl character varying, _col1 character varying, _col2 character varying)
RETURNS SETOF integer[] AS $$
DECLARE
      ultima_calle integer;
      contador integer;
      fila RECORD;
      r boolean;
BEGIN
FOR fila IN EXECUTE 'SELECT ' || quote_ident(_col1) || ' as calle_id, ' || quote_ident(_col2) || ' as punto_id FROM ' || quote_ident(_tbl) || ' ORDER BY ' || quote_ident(_col2)
      LOOP
      IF ultima_calle IS NULL THEN
         ultima_calle:= fila.calle_id;
         contador:= 0;
      ELSE
         IF ultima_calle != fila.calle_id THEN
               ultima_calle:= fila.calle_id;
               contador:=contador+1;
         END IF;
      END IF;
       RETURN next ARRAY[ fila.calle_id, fila.punto_id, contador];
END LOOP;
END
  $$ LANGUAGE plpgsql VOLATILE;
```

As can be observed, the itinerary function receives three parameters: the name of the table and the names of the columns that contain the identifiers of the track points and the streets. These three parameters correspond to the previously created *track_callejero* table.

With the data structure returned by the itinerary function we can obtain a list of the first and last track points that pass along each street. For this purpose we execute the following command:

```
create table itinerario_seguido as
      SELECT calle_id, MIN(punto_id) as primer_pt_id, MAX(punto_id) as ultimo_pt_id FROM (
            SELECT ar[1] calle_id, ar[2] punto_id, ar[3] contador from (select itinerario('track_callejero','calle_id','punto_id') ar) as fo
          ) as foo
      GROUP BY calle_id, contador
      ORDER BY contador;
```

We obtain a new list, as can be observed in the image below.

calle_id integer	primer_pt_id integer	ultimo_pt_id integer
339	1	1
350	5	39
1636	40	42
352	43	65
2561	66	70
2560	71	76
1369	83	91

With this new table we can then calculate the time invested in each street, using the time that has elapsed between the first and the last point of each street.

Select calle_id, (t2.time – t1.time) as tiempo_invertido_en_calle
from track t1, track t2, itinerario_seguido i
where i.primer_pt_id= t1.gid and i.ultimo_pt_id = t2.gid;

calle_id integer	tiempo_invertido_en_calle interval
350	00:00:52
1636	00:00:02
352	00:00:39
2561	00:00:05
2560	00:00:07
1369	00:00:13
2557	00:00:03
2468	00:00:03
198	00:00:01

3.7 Assigning track points to heritage elements

Certain heritage elements of the city are defined as key attractions by the city's tourism promoters. As such, we have set out to analyse the time that visitors (object of the study) devote to each of them.

It is considered that a heritage element has been visited when the tourist has remained at the site for a predefined length of time (15 minutes in the case of elements for which a short visit time is considered sufficient, and 30 minutes in the case of those for which more time is considered necessary).

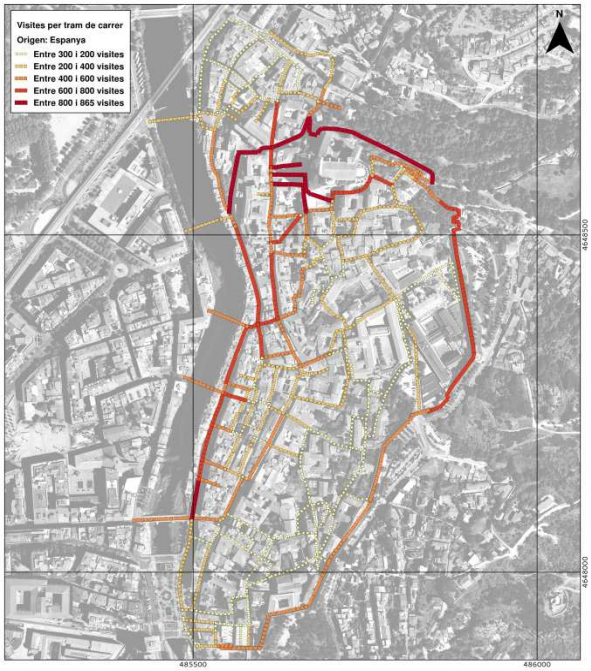
An area of influence has been generated around these heritage elements, and the first and last track points detected in this area of influence have been identified. Given that each track point contains information on the hour/minute/second in which it has been captured, it is straightforward to calculate the length of time devoted by the tourist to visiting each element.

Linear heritage elements, such as the city walls and certain streets, are considered visited when the track points closely follow them.

4 Results

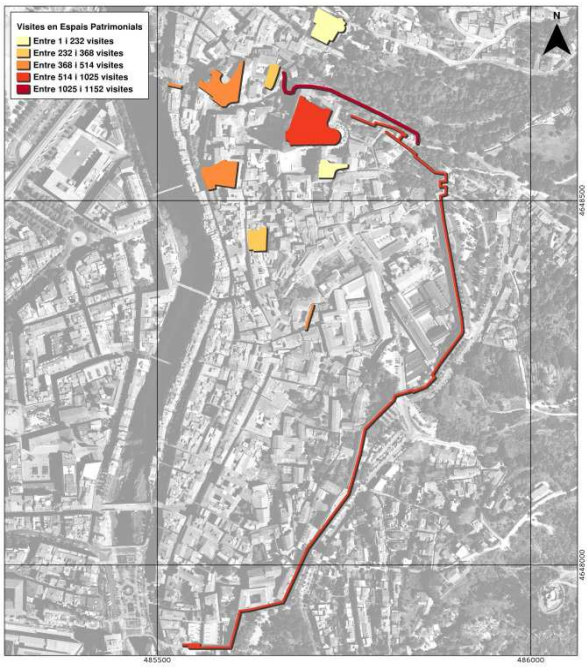
The possibility of comparing each visitor's track with a qualitative questionnaire has made it possible to obtain specific results in relation to some of the questions that were posed. As such it has been possible to represent the assignment of tracks on the street map (that is, represent the degree of intensity of use of the street map), not only for all the visitors who formed part of the study but also according to the following factors:

- Where they have stayed (accommodation)
- Nationality
- Number of times they have visited the city previously
- Reason for the visit



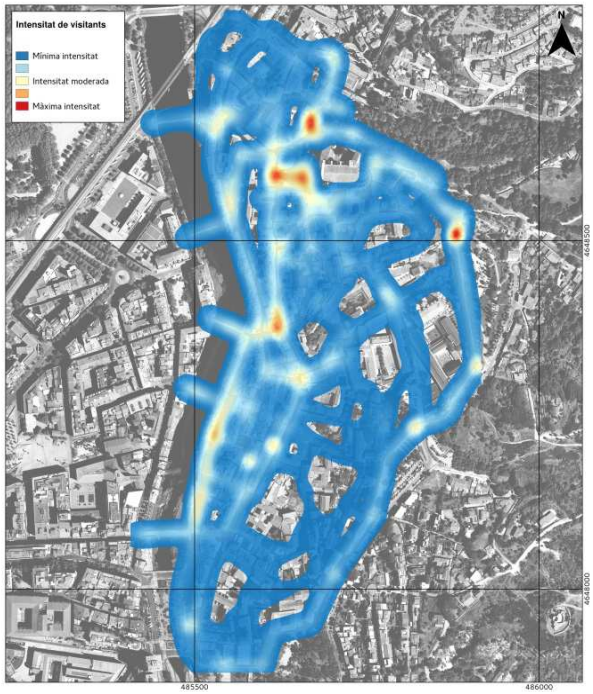
It has also been possible to represent other aspects thanks to the time data contained by the tracks, such as specific times of day or seasons of the year.

As in the case of streets (or street arrays), the visits to key heritage elements have also been considered in their totality, as well as in relation to the different variables gathered from the completed questionnaire. Thanks to the design and implementation of a relational database that contains all the information on the street arrays, questionnaire answers, heritage sites and tourist tracks, it is possible to produce specific theme-based cartography by combining the information provided by all of these elements.



Furthermore, a representation is also displayed of the track points prior to being assigned to the street map, in order to capture the total magnitude and for the purpose of comparing it with the 'manipulated' results after assigning the points to the street map.

The result of this general analysis of waypoints has been displayed through a density map [5]. This type of technique for the interpolation of point data compares the space with the number and position of each of the waypoints for the total area of study. A density map, which to a large extent offers qualitative results, in this case provides information on which areas or spaces of the city are most densely occupied by tourists, while at the same time offering information on where in the city it would be easiest to find the highest concentration of tourists at a given time of day or in a given season of the year.



5 Conclusions

The project is an example of how databases can be used to carry out spatial analyses. In this respect it should be taken into account that the entire project, except for the final graphic representation, has been carried out within the PostgreSQL/PostGIS spatial database using SQL (Structured Query Language) commands.

Through these SQL commands we have carried out the following operations:

- The street map has been broken down into a segmented street map in order to analyse which street sections are the most visited.
- The GPS tracks have been converted from geographic coordinates to projected coordinates. In this way we have been able to use Euclidean distances to assign each track point to the nearest street.
- An analysis has been carried out of the most visited areas of the city, taking into account specific times of day and seasons of the year.
- The results obtained have been compared with the results of the questionnaires completed by the tourists. As such it is possible to know what tourists' preferences are according to a certain parameter of the questionnaire (nationality, budget, etc.).

Without a doubt, this project is a successful case of a spatial analysis carried out not using a conventional desktop GIS (the most habitual solution) but rather a robust open source database (Postgresql) supplemented with a highly developed and stable spatial capability (POSTGIS).

This opens up a range of possibilities for working on and exploring other functions of analysis or visualisation that continue to add value to field-gathered data.

References

[1] M. Modsching, R.Kramer and K.ten Hagen. Field trial on GPS Accuracy in a medium size city: The influence of built-up in Proceedings of the 3rd Workshop on Positioning, Navigation and Communication. 2006

[2] http://www.gdal.org/ogr2ogr.html

[3] http://postgis.net/docs/manual-1.3/ch04.html, ch. 4.3.2 & http://manpages.ubuntu.com/manpages/natty/man1/shp2pgsql.1.html

[4] http://blog.cleverelephant.ca/2008/04/snapping-points-in-postgis.html

[5] V. Olaya, Sistemas de Información Geográfica (2013)