Where Have I Been - Visualizing Personal Geolocation Data

Jorge Miguel Saldanha Filipe
jorge.s.filipe@tecnico.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

November 2015

Abstract

Big volumes of data are being produced with the large number of portable devices that exist nowadays and are capable of collecting GPS data. Despite the fact people track their mobility, most approaches on the analysis of spatiotemporal data are too complex and technical. It is very easy and common to collect spatio-temporal data but it is difficult to analyze it and extract things with personal relevance. To make the analysis of personal geolocation data easier, we devised a visual language for accessing and querying that data, including support for personal semantics of locations. This visual language was validated, before developing the system, to see if users could use and understand it. We then implemented a system that integrated our visual language with result viewing and map interaction. An evaluation showed users could use and understand the whole system, thus certifying our initial objectives.

Keywords: movement data, spatio-temporal data, scalable visualization, geovisualization, personal semantics

1. Introduction

Recently, the massive spread of GPS tracking devices, such as smartphones, smartwatches, tablets, dedicated devices, has been producing huge volumes of data that not only represent the mobility of people, but also animal behavior or natural phenomena. This massive collection of data makes it possible to say that, at some point, everyone is a spatio-temporal analyst [3], either by planning a journey, looking for a job or searching for restaurants. With the advance of technology people tend to plan more and more their life: find the days of the week that will be rainy, understand terrain prices in order to build a house, search for epidemic contagion.

Despite all those features and the fact that people track their mobility, most approaches on spatio-temporal data, including GIS systems, don't consider the personal side of the data. That is, data having its own private meaning for the user (either spatial: like "my home", "kids school" or temporal, like "my birthday"). There are however, some exceptions worth mentioning: spatio-temporal location in social networks and lifelogging.

So it is very easy and common to collect spatio-temporal data, but it is difficult to analyze this data and extract events of personal relevance. Most important, there is a lack of tools that are capable of visualizing and querying personal geo-temporal information.

Based on these tools, users could find information and standards that really are important to them personally. For example, one of the main benefits for users is the ability to have insights about their personal location data. Users can query about places they have been, find out the places where they spent more time or go more often.

Since a simple and specific tool, for which there is no need for users to have expert level knowledge, is missing, we propose to fill that gap. Our proposal is a tool able to query personal geolocation data in order to find specific routes, including the personal semantics of the locations. In order to achieve this, we must first identify the different data inputs we need to process and avoid the direct use of SQL language in the querying system, so the user can deal with it with ease and, instead, create a visual language. After that, we created a system based on that language and tested it with users.

We believe a query must be visually represented in order to be easily understandable. We studied how to create a visual language that has full support in querying geo-temporal data and also supports user assigned semantics. There is also an obvious relation between geo-temporal data and maps - so we came up with the idea to integrate maps with the query system, allowing the user to both directly select locations and see the query results in a map. Because the language must be understood and used by users, we have validated it with users

before proceeding into the actual implementation.

After the visual language definition, we also need to know how we could offer the user a good interface for the incoming results of a query. Since the results can extend for a large period of time, a user must be able to quickly understand an overview of all the data. We, therefore, studied how to present the results. To do both result presentation and query language definition we will extensively review related literature, so that we can perceive existing patterns and ideas that may be applicable to our solution, as well as existing problems faced by solutions dealing with geo-spatial data.

Lastly, we evaluated the system as a whole to understand to what extent is it easy to use. There was particular interest in evaluating how users deal with the query interface, because it is the most relevant part of the solution.

We are only focusing on showing the data of one person - this is not a tool related to crowd-sourcing. Despite being the data of only one person, this data can be extensive and spawns many years of life. There is also no concern on how the user gathers its data. One of the things a user should be able to do is to have insight of his/her personal mobility information and perceive patterns in the daily life. Thus, our main objective was:

Create a visual query language for personal mobility data that allows users to visually query their personal spatio-temporal information, with personal semantics.

Several secondary goals arise from the analysis of the main goal. We will discuss those goals below.

One of the objectives is to **analyze and study the expressiveness of the queries** we want our system to support. We will also need to analyze the best way to **visually represent the resulting information** (this is the information a user wants to see about a specific track - duration, average speed, etc.). And, perhaps the most important issue, is to **analyze the best way to visually query the information**, that is, the query interface. It is important that this interface has enough expressiveness for a non-expert user, and that no SQL queries have to be written.

Since we're dealing with personal data, we also have the concern of privacy. There is the need to identify which information can be extracted from various types of data [4] and privacy-preserve that data.

2. Related Work

Related work was divided in three different areas: Visualization, Specifying queries and Conceptual frameworks.

2.1. Visualization

Regarding scalability, our work requires that a huge amount of data (potentially a person's whole life) can be queried and visualized with ease. There are not many solutions that allow this - most of them rely on importing data that spawns around a week long. More than that time and the interface would become messy with too much information for a user to understand. The most notable exception is AprilZero [1] - this work's focus is on a person's whole life, and therefore allows to display the information in a clear and understandable way.

The field of personal semantics, that is, the support for locations and times that only make sense for the user (my house, my work, mom's house, lunch time, etc.) is only supported by two works - Visual Mobility [7] and AprilZero [1]. As stated in the Objectives section, our work aims to allow the user to query his/her personal information. The said information has concepts that only make sense for that user (mom's birthday, my house location) thus it is important to support this feature.

For the visualizations methods we have chosen not to include any works that use automatic animation, because the widely spread opinion is that it fails to be more helpful than other visualizations [9] [10]. Many usual techniques are used, such as 2D maps and space time cubes. There is however, one different and interesting way of representing the information - in a timeline (Visits[2], LifeLines[8], GeoTime[6] and AprilZero[1]) - that seems to work well.

Another important feature is the data quality and cleaning issue. It is important, for a successful visualization experience, that the data shown is previously analyzed and divided in parts that make sense in the current context (eg. dividing a big GPS track into several tracks that correspond to a trip). It is also important to ensure that the information is real - when working with devices that gather GPS signals, it is often common to get many types of errors, including Visual Mobility[7]: *GPS errors* (causing misrepresentation of the actual route taken) and *Forgotten start/end* (when the persons forgets to turn on/off the data collection - leading to wrong detection of trip ends). Despite being an essential feature, very few works worry about dealing with them.

To summarize, the most important topics in visualization of geo-temporal data our solution will address are:

- Scalability (the support to potentially show a person's whole life data) of the solution.
- The support of personal semantics while querying the data (it is one of the main goals).

- Data quality - the tracks must be pre-processed for error-removal and simplification.

2.2. Specifying queries

There are two types of scope in the analyzed works: Spatial and Temporal. The first one refers to works that only allow us to query spatial features, either by specifying boundaries or selecting a zone on a map. The second type of works focus on querying temporal information, either by making use of a timeline, some widget or some other metaphor.

Some works may consider both temporal and spatial features, but only allow queries of one type. This is the case of Time Automaton[5]. It allows querying temporal data, and the resulting information is shown with the help of a 2D map.

Besides these two types of scope, we also focus on the support of recurrent queries and the possibility to support queries with personal semantics.

Only Time Automaton[5] allows a very useful feature in our work - the possibility to have queries of recurrent events. That is, a user might want to know things like "show all the places near a shopping mall where I went more than three times".

None of the studied works has personal semantics in their query system. Our work is closely related to personal information because we need to allow the user to include his/her places/times that have a personal semantic (eg. my house, lunch time).

2.3. Conceptual frameworks

The two main concepts of our solution are spatial and temporal queries. Personal information is not considered a query type, because it will be included in the spatial and temporal strands. It is also important to allow recurrency in our system - people want to ask things like "show all the places near a shopping mall where I went more than three times".

Regarding a spatial scope, an emergent pattern is the difference between endpoints and paths. Both concepts are very different: the endpoints are what is commonly referred to the end and start of the trip. The path is the way that connects the start to the end.

Our solution also relies on this approach: while formulating the spatial part of a query, it is important to distinguish the endpoints from the path because there are attributes that an endpoint can have and a path cannot.

3. Where Have I Been

Below we explain the components of our application.

3.1. **Data collection**

Our main goal is to explore a users' personal mobility data: for this purpose the user must collect GPS data on a daily basis. Data collection is independent of the device as long as it produces some file output, which will be used as input for our application.

Ideally, the user would turn *on* GPS data collection when moving from one place to another and turn it *off* when arriving at some place. This behavior is ideal because of two different reasons: batteries don't last forever, and therefore, reducing tracking to the essential minimum would spare some battery. The second reason is that tracking location indoors does not work - because the GPS signal cannot reach the device and generated GPS errors, thus it only makes sense to record tracks while moving outdoors. This behavior generates a single GPX file every time a user is moving, which is a better result than having a whole file for a single day. However, it introduces some problems. For example, a user might forget to turn recording *off* when not moving - this will feed false information to our application (we will lose the one to one relation between trips and tracks). Furthermore, most today's devices are not capable of using the GPS sensor during all day without recharging the battery.

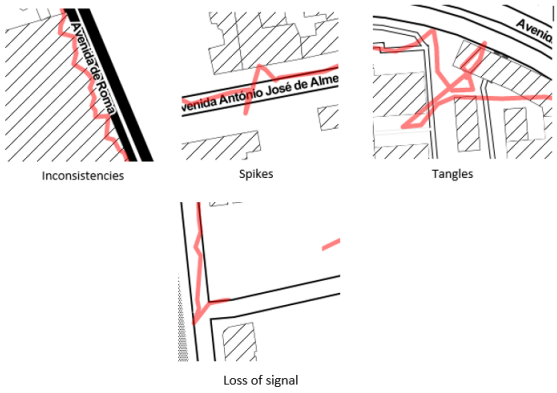

Figure 1: Common GPS problems

It is also important to assign personal meaning to the data collected by GPS sensors. This data must require some degree of user interaction, for only the user knows which places and tracks have a special personal meaning. To illustrate this idea we could assign *My home* to a specific coordinate, from then on, every time the user wanted to search for his home he would refer to it as such, and not use direct coordinates.

We propose a format, inspired by an existing one, used by some supporters of lifelogging, in order to address this question. The purpose of the format is to assign semantic meaning to a period of time. In the case of this application the purpose is to assign a name to a time a user was indoors.

3.2. **Expressiveness**

Our primary input will be loose GPX tracks that have little usefulness for querying personal spatio-

		Temporal	Recurrence	Spatial	
				Endpoint	Path
Accuracy	Fuzzy	A time interval	Some amount of time	A range	The same route, within some limits
	Accurate	A specific time	Specific amount of time	A specific location or distance	A particular route
Relativity	Relative	Time relative to another event	Comparison	Places in relation to other ones	A route within a certain distance from a baseline, or with similar features
	Absolute	Concrete times / intervals	Specific amount of time	An exact place	A particular route
Concreteness	Abstract	A "class of times" (time description)	-	A class of places or rough description	A class of possible routes
	Concrete	A concrete time	Specific amount of time	A concrete place	A concrete route

Table 1: Spatio-Temporal classification for personal geolocation data

temporal data. It is essential to systematize how to handle these tracks, so we can find relevant information to query.

Table 1 shows a summary, based on the analysis of all the previous works, of how to systematize personal data regarding space and time.

Three ways in which a user might query it's data are addressed: **Accuracy**, **Relativity** and **Concreteness**. Each of this concepts can be applied to four different areas: **Temporal**, **Recurrence**, **Spatial Endpoint**, **Spatial Path**. This was thought to give a fairly complete coverage of all things that may evoke reasonable questions in this context.

Accuracy refers to whether or not the user wants to refer to something precise or losely. In the case of time, a user could ask for an accurate time (09:00) or something more vague (*around* 12:00), in the case of space, if it is an endpoint a user might want to refer to a precise place (e.g. exactly at home) or a range around a place (no more than 10m from home); if it is a path, a user might want to refer to a precise route (accurate - an actual GPS route) or a route within some limits (fuzzy - e.g. a route that doesn't deviate 100m from another route). When applying recurrency, a query is accurate if the user wants to know something like "where I went exactly 3 times", or fuzzy if the user asks "where I went *about* 3 times".

Relativity refers to comparisons between things. In the case of time, a relative time is one relative to another event (one hour after...) and an absolute event is the same as a concrete or accurate one (9:00). In the case of space, if considering an endpoint, we can be relating a place to another (e.g. a place near the shopping mall), or an exact place (e.g. at home); if considering a path, something like (e.g. a route that doesn't deviate more than 100m, overall, from route X - or endpoint X) is relative and a particular route (e.g. a similar route to an actual GPS route) is absolute. In relativity, recurrency is easy to understand: a user might want to compare the number of times he went to a certain place - "went more times to X than to Y".

Concreteness defines the target. In the case of time, it can be an abstract date, such as "mom's birthday", or a concrete time like "9:00". In the case of space, if considering an endpoint, we might give a rough description of a place (e.g. generic shopping mall) or its absolute position (e.g. mom's house). If considering the path, an abstract path is a class of possible routes (e.g. highway) and a concrete refers to a concrete route (e.g. highway A5). In the case of recurrency, we believe there is no such thing as an "abstract" recurrency. A concrete recurrency is the same as an absolute or accurate one.

This table will serve as the basis for our query mechanism, because it specifies the expressiveness we want the system to have. Thus, it specifies all the possible query types our system will have to support.

3.3. Visual Language

In order for users to make queries we developed a visual language allowing them to search for their personal spatial data.

As explained in section 3.2, our language will have to support three different concepts - time, space (in two ways: place and path) and recurrence. Each of these concepts will be divided into different degrees - accuracy, relativity and concreteness.

Our language metaphor is a timeline. Users can sketch their query in this timeline with two different types of objects: *paused time* and *moving time*. As a guiding principle, everything related to time will be drawn on the horizontal axis and everything related to space will be drawn on the vertical axis. Another important consideration is that larger areas imply a larger values.

Routes cannot exist on their own, they must have a start point and a destination point, thus, every time two *paused time* are created, a route (*moving time*) connecting the first to the second will automatically be added.

Paused time (represented in figure 2 as a blue rectangle) reflects a period of time when the user was not moving - usually means the user was at some place: home, restaurant, school, etc.

Moving time (represented in figure 2 as a gray line between the rectangles) is a period of time when the user was moving. There is *always* a line between any two such rectangles.

To support each different concept (time, space,

Figure 2: Where Have I Been query area

recurrence) each object has fields that can be filled in. Every parameter is optional - in case the user doesn't fill it in it will not be considered.

3.3.1 Temporal concept language specification

Below we explain how the language was formulated to support the time expressiveness represented in table 1.

- **Accuracy support** A time is either accurate or fuzzy. To represent an accurate time a user must fill in a time box, located at the left and right sides of an *paused time* (figure 3). By just selecting an hour, the user will be referring to *precisely* that hour. However, a user may want to know something like "Where was I after 17h?" - in that case the mathematical symbols (>, <, <=, >=) should be used.

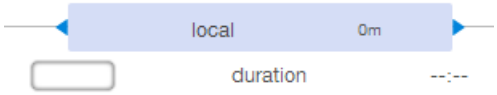

Figure 3: Editing a start time

  To represent a fuzzy time, a user must first specify an accurate time and then specify the desired range. Times are shared between different objects, for example, in figure 2 the first *paused time* shared its ending time with the route object (this time as the start time). Instead of specifying a start/end time, a user might just want to specify the route/stay. In this case no conflicting information can be provided: that is, the duration cannot be longer than the specified (if any) start/end times.

- **Relativity support** Relativity only makes sense when there is more than one object. To achieve a relative specification a user has to specify a duration in the route connecting two locations. For example, if the user wants to know where he was one hour after being at home, he should draw two rectangles and specify the duration of one hour in the route connecting both rectangles.

- **Concreteness support** An abstract time (a class of times) is specified in the same box as start/end time - but as soon as the user types, a dropdown of existing abstract times is shown for the user to select.

3.3.2 Recurrence concept language specification

Below we explain how the language was formulated to support the recurrency expressiveness represented in table 1.

- **Accuracy support** Recurrence is represented by an arch above a stay. It is created by selecting the rectangle and then the icon representing the arch. Recurrence is applied daily - that is, all results will shown how many times the user went to the specified place during a specific day. As with time, recurrence can also be fuzzy. Selecting fuzziness follows the same principle as in time, using mathematical symbols.

3.3.3 Spatial concept language specification

Below we explain how the language was formulated to support the spatial expressiveness represented in table 1.

Endpoints Spatial concept is divided in Endpoints and Paths. Endpoints are actual locations either identified by the user, or represented by geographical coordinates.

- **Accuracy support** To provide a fuzzy specific location a user can write a location name in the middle of the rectangle and select existing locations from a dropdown menu. If a user wants to specify a coordinate instead of a name, he must click on the box and then on the location he wants on the map. As explained before, space is represented vertically. So, in order to specify a fuzzy location (that is, a location with some range) a user can vertically drag the upper and lower borders of the rectangle (or just input a number - in meters, in the box on the upper right corner), see figure 4.

- **Relativity support** To specify a place in relation to another, a user can specify a location (either by name or coordinate) and then specify a range in which the wanted location is.

- **Concreteness support** As with times, locations can also be grouped into classes. The process to choose a class of places is similar. While writing in the location box, categories will also appear on the dropdown menu.

Figure 4: Editing a spatial range

Paths A path is the actual route made by the user to get from one point to another.

- **Accuracy support** To specify a specific route of comparison a user can upload a GPX file containing that route. To specify a fuzzy route, also needs to upload a file, but then has to vertically drag the line, following the principle that larger areas mean larger values, to specify a desired fuzziness.

- **Relativity support** Relativity in this case means using a location (coordinate) as a passing point of the route. To specify it, a user must pick click on the route box and pick a coordinate from the map. Doing so will automatically add a range (defined in the settings).

- **Concreteness support** As with time and endpoints, paths can also be grouped into classes. In order to select a class a user needs to start typing in the route box - as usual a suggestion will appear on a dropdown, allowing the user to choose the desired route.

3.4. System

Here we present our approach - first by explaining the scenario in which it will be used. After that, by explaining all the components of our implementation.

Our typical user will use a smartphone every day, doesn't matter which brand, as long as it has a GPS receiver and a good enough battery. Each time a user moves from point A to B, either walking, driving, etc, the device will record the path - a very simple application for this, in Android is *My Tracks*, this application is simple because it allows to record and stop the path with a single click, without having to write anything or go through complicated menus.

The main idea is, each time a user moves from one place to another, a different track should be recorded. This means that if, for example, a user goes from home to a coffee-shop, spends five minutes drinking coffee, and then goes to school, it would result in 2 different tracks: home → coffee-shop *and* coffee-shop → school.

As stated before, we support personal semantics related to locations a user is familiar with. For this to happen the user will have to register at which place he was at a certain time - with this mechanism (that is intelligent enough to understand patterns and suggest routes that happened in similar conditions) we will be able to categorize the locations in a personal semantic way.

We adopted a client-server approach in order to improve scalability. The main functions of our server are to clean GPX files, store data in a spatial database and process the queries.

As explained before, our main objective is not caring about the source of the data, but it is also important to provide quality data in order for the queries and visualizations to be optimal. Thus, every GPX file representing a track will be processed by a library (GPX Lib) that, among other things, cleans the data, removes error points, smooths the trajectory, etc.

All our server code is in the Python language. This choice was made because it has great community support and many libraries are publicly available. We provide this API at GitHub, because this library will not be specific for this work, and we think it could be used by others to remove errors from GPX tracks.

After being processed, track data is stored in a PostgreSQL + PostGIS database. It is not a common relational database because we are handling a particular data type: geographical data. While relational databases are made to deal with types like integer, dates, string, currency, we need a convenient database to handle points, edges, polygons (areas). Several extensions to known database systems provide spatial support. We've chosen PostgreSQL extension - PostGIS because it is considered a mature service and it also supports point, linestring and polygons. Furthermore it follows the SQL specification from the Open Geospatial Consortium.

Communication between the client (browser) and the server are made using Websockets. Websockets were chosen instead of Ajax requests because they allow the server to communicate with the browser without the browser having to request it. The browser can also send data to the server. This is

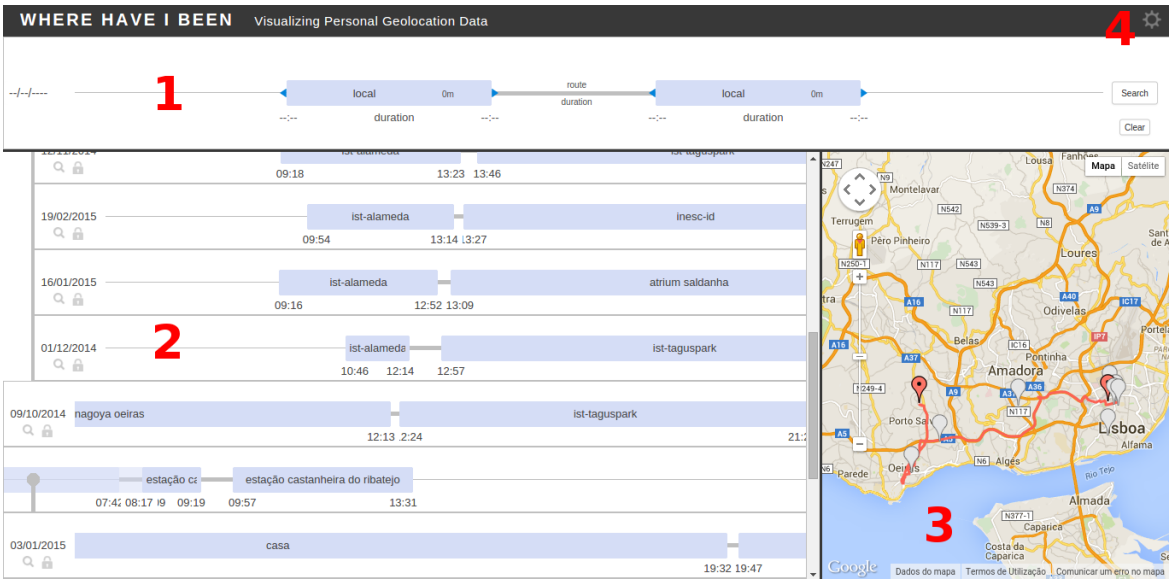

Figure 5: Where Have I Been's interface

quite useful for sending notifications and updates because the server can send them when it gets them, instead of waiting for the browser to ask for them.

The front-end side will exchange data from both queries and results, from the server with JSON messages, in particular, messages that contain spatial data will be encoded in GeoJSON. We choose JSON because it is a common standard to transmit data between a server and a web application, furthermore, having a specific standard for dealing with spatial structures is also useful and can benefit this solution in the future, integrating with other projects. Another advantage of choosing JSON is a larger number of tools already supports JSON messages.

After data is stored, our server is ready to wait for queries from the client. When a user enters a query in the interface and then presses the search button, a JSON object containing all the fields is sent to the server. Upon receiving this object, the server has to translate that information into a SQL query. The translation mechanism is quite simple: each part of the query system has a direct translation to SQL, we just need to compose all the different parts in a single query. Furthermore, different query types have different SQL templates, thus generating a full SQL statement implies combining all of the above. If a user draws a query based on temporal fuzzyness, we know the main structure of our statement will need a WHERE clause. And because we are dealing with fuzzyness, we will also need a BETWEEN clause. After generating the template, the properties that are inside the JSON object must be converted to the correct types and injected into the SQL statement.

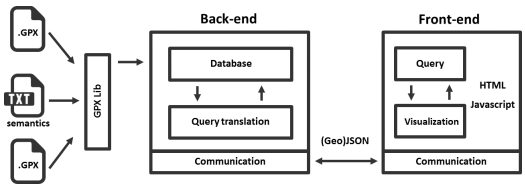

Figure 6: Solution schema

4. Evaluation

The main objective of the evaluation was to understand if users, in practice, understood and could use our visual language and if they could, in general, use the system in a useful way that would be personally relevant.

To evaluate the usability, a set of users was given the system to try and comment on its use. This evaluation consisted in a session, composed of three parts:

1. Initial interview to trace testers' profile (e.g. age, gender, studies, etc.);

2. Tasks, users were presented a set of actions to perform on the application that covered all the possible solutions to a certain problem;

3. Debriefing questionnaire, aimed at usability qualitative assessment of the system.

The mobility data used to evaluate our work consisted of two months of daily tracking.

For the evaluation we asked for a set of tasks to be completed. We measured the time users took to finish the task, record the screen, and asked them to voice their difficulties aloud. Those tasks were related to the issues below:

- Evaluating the **temporal part** of a query - users should evaluate and describe (talking aloud) the difficulties they encounter while elaborating the temporal part of a query. The focus was on the different ways a temporal query can be done, such as *time intervals* or *specific times*.

- Evaluating the **spatial part** of a query - users should follow the same principle as before: talk aloud about the difficulties in making a spatial query, with particular focus on *a range around a place* and *a specific place*.

- Evaluating how the **results** are shown - tasks asked users to find a precise result in the result list. Users should identify it and comment aloud on the doubts they have.

- Evaluating the way a route is shown in the **map**.

- Evaluating the **settings** interface.

Data collected during the task execution includes: task duration time, number of errors, number of clicks and other relevant annotations. In the end we asked users to fill a questionnaire that qualitatively evaluates the implemented functionality. Users commented and classified the utility of the features mentioned above. We also asked users to fill a questionnaire regarding the global feeling of the solution. This questionnaire was based on System Usability Scale (SUS).

Before starting the first stage, we explained the motivation behind this tool and asked users to briefly explore the tool's user interface and various functionalities in order to get comfortable with the tool and reduce errors and long task execution time periods on the first tasks.

The task set consists of 12 different tasks focusing different areas of the interface and different expressiveness query levels. An example of a task related to temporal expressiveness is "*Search when I was at ist-alameda for more than 2 hours*". Another example, this time related to exploring the results panel is "*Repeat the first query. Point all the results in which I went from castanheira do ribatejo to ist-alameda*".

Tests were executed by 21 subjects, 47.6% were male and 52.4% were female. 71.4% were aged between 18 and 25 years old, 23.8% between 26 and 35, and only one person was more than 35 years old. Regarding studies, most of the users had a background in Science and Engineering (90.5%), while the rest had a background in Social Sciences.

task	avg clicks	stdev click	avg time (s)	stdev time (s)	avg error	stdev error
1	12,7	3,1	36,4	12,9	0,5	0,9
2	10,9	4,1	32,4	10,1	0,3	0,6
3	20,6	5,8	47,5	12,1	1,0	1,4
4	16,0	3,3	31,0	8,1	0,3	0,6
5	9,0	2,0	26,7	8,0	0,4	0,7
6	7,2	4,4	13,2	15,7	0,4	0,8
7	15,9	5,0	41,5	13,4	1,0	0,9
8	7,5	1,9	18,4	3,8	0,2	0,4
9	7,0	2,0	18,7	3,3	0,2	0,4
10	10,8	1,6	28,7	4,2	0,0	0,2
11	12,8	1,2	24,0	3,3	0,1	0,3

Table 2: Results by task

Our overall SUS Score is 80.75, which means that we have a good system and we are on the right track.

On the second part of the questionnaire, 57% of the users gave a global appreciation (on a scale from 1 - terrible to 5 - very good) of 4, while 33% gave the maximum classification. As for the questions about smartphone usage, only one user did not have one. From those who had one, 66.7% said they used the GPS of their phone. The most common usage of the GPS was map navigation, with 85.7% of the users using it. When asked if they would track their daily geolocation data, 52.4% of the users answered no. The main reason that motivated this answer was the short battery life duration.

Generally, the results of our experiments were positive, to the extent that most people considered that our application is helpful and the exercises were adequate. Also, having a score of 80.75 on SUS is a very good result, although we know that there is a lot to improve.

Overall the task results were good and corroborate that our efforts to build a simple and flexible interface were well applied. Usability tests show that there are still some finetunings to be made (mainly related to the way results are shown and map integration), but overall feedback from users is very positive. The usability of the system was proven with the tests, since all users completed all tasks. Situations to be tuned include allowing the user to order results and clearly stating that an entry has no map data.

Overall users' feedback was also very positive. They stated they gained a perspective on mobility patterns and would frequently use the tool if the collection method was simple and did not drained the phone's battery, thus proving the system to be of utility.

Regarding our objectives (initially specified in section 3) we can say that all of them were achieved. Below we make a careful analysis of each objective and the things that still need improvement. The main objective (*Create a visual query language for personal mobility data that allows users to visually query their personal spatio-temporal infor-*

mation, with personal semantics) was successfully achieved, only having minor situations to correct and improve. User tests showed that our language was easily manipulated and understood, and could easily generate complex queries in little time.

Another objective was to *visually represent the resulting information*. This was also accomplished, but it requires more improvement, in order for results to be easily explored. Users stated they would like results to be more dynamic, by having the possibility to order them and search through them.

5. Conclusions

The growing offer and massive spread of GPS tracking devices can generate huge volumes of personal mobility data. As we've seen in this report, there is no efficient way that provides a pleasant experience for the user to explore its mobility data.

Thus we found the need to develop a solution that allows users to query their personal mobility data while maintaining personal semantics of the locations. First we needed to find the essential components of our solution: the visualization and the query system. For the query system we had to understand what was the expressiveness we were looking for - we analyzed what were the main components (temporal, spatial and recurrency) and made a table that contemplates the supported queries.

In order to understand the best compromise to use in the formulation of visual queries we analyzed several works that also used this mechanism, so we could understand the advantages and disadvantages of different approaches. After considering the different options (comic strips, timelines and graphs) we chose the timeline, because it allows for an obvious natural time dependence between events.

We also analyzed several works to understand the best visualizations methods (space-time cubes, 2D maps, timelines, etc) and realized the best option was to integrate several of these components. Thus we have chosen *2D maps* to show the route, and a *timeline* to present all the results of a given query.

So, based on our review, we proceeded to devise the expression of queries that a personal system of this type should have. There we have one of our main contributions, as detailed in section 3.3. After specifying the visual language we had to determine if users could really use it, so we implemented a solution for users to test. The validation was successful, as stated in section 3.

After concluding that users understood the visual language, we implemented a whole system including that language, which is another contribution of our work. This system allows specifying queries and exploring their results.

This system worked, as our evaluation shows in section 4, thus our initial goals of *creating a visual query language for personal mobility data that allows users to visually query their personal spatio-temporal information, with personal semantics* and *visually represent the resulting information* were achieved.

Acknowledgements

The author would like to thank his supervisor, Daniel Jorge Viegas Gonçalves for his exemplary guidance, monitoring and constant encouragement throughout the course of this thesis.

References

[1] AprilZero. `https://aprilzero.com/`. Accessed: 2015-10-11.

[2] Visits. `http://v.isits.in/`. Accessed: 2015-10-11.

[3] G. Andrienko, N. Andrienko, U. Demsar, D. Dransch, J. Dykes, S. I. Fabrikant, M. Jern, M.-J. Kraak, H. Schumann, and C. Tominski. Space, time and visual analytics. *Int. J. Geogr. Inf. Sci.*, 24(10):1577–1600, Oct. 2010.

[4] G. Andrienko, N. Andrienko, D. Keim, A. M. MacEachren, and S. Wrobel. Editorial: Challenging problems of geospatial visual analytics. *J. Vis. Lang. Comput.*, 22(4):251–256, Aug. 2011.

[5] L. Certo, T. Galvão, and J. Borges. Time automaton: A visual mechanism for temporal querying. *J. Vis. Lang. Comput.*, 24(1):24–36, Feb. 2013.

[6] T. Kapler and W. Wright. Geotime information visualization. In *Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on*, pages 25–32, Oct 2004.

[7] F. F. Leandro. Visual mobility - visual exploration of personal mobility data, 2012.

[8] C. Plaisant, B. Milash, A. Rose, S. Widoff, and B. Shneiderman. Lifelines: Visualizing personal histories. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '96, pages 221–227, New York, NY, USA, 1996. ACM.

[9] G. Robertson, R. Fernandez, D. Fisher, B. Lee, and J. Stasko. Effectiveness of animation in trend visualization. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1325–1332, Nov. 2008.

[10] B. Tversky, J. B. Morrison, and M. Betrancourt. Animation: Can it facilitate? *Int. J. Hum.-Comput. Stud.*, 57(4):247–262, Oct. 2002.