A GPS simulation framework on smartphones for elderly assistance applications

German Paul-Adrian
Polytechnic University of Bucharest
Faculty of Computer Science
Bucharest, Romania
paul.german94@gmail.com

Fernando Luis-Ferreira
CTS, UNINOVA, Dep.º de Eng.ª Electrotécnica
Faculdade de Ciências e Tecnologia, FCT,
Universidade Nova de Lisboa
flf@uninova.pt

João Sarraipa
CTS, UNINOVA, Depº de Eng.ª Electrotécnica
Faculdade de Ciências e Tecnologia, FCT,
Universidade Nova de Lisboa
jfss@uninova.pt

Ricardo Jardim-Goncalves
CTS, UNINOVA, Dep.º de Eng.ª Electrotécnica
Faculdade de Ciências e Tecnologia, FCT,
Universidade Nova de Lisboa
rg@uninova.pt

Abstract— Ageing is a natural process for all living beings implying some cognitive and motor changes with time passing. The later stages of life have some associated disabilities and pathologies that can impose limits to the way of life that previously was a routine. Medical research and technological developments address those limitations with pharmaceutical drugs and treatments that overcome such limitations while providing support to extend, as much as possible, that previous normality. As people live longer lives, some conditions become more frequent as is the case of dementia and particularly of Alzheimer. While clinical staff try to find treatments and apply existing medical knowledge, it is important to help people in such conditions to continue doing their routines with minimal risk and distress for them and for relatives and caregivers. Displacement and posture are critical aspects for elder as sometimes they forget places and routes, getting lost with physical risks of accidents and exposing themselves to potential dangerous situations. This paper presents a framework to support development and testing of applications that use devices to keep track on displacement, using smartphone sensors as the pervasive built in GPS sensors, within the lab environment.

Keywords—Ambient Assisted Living, GPS simulation,

I. INTRODUCTION

Lately more and more applications on the market are location based and operate in order to provide users with the best solutions based on local context awareness [1]. Because every Application (APP) needs testing before release, it becomes necessary to analyze how it reacts to different locations, speeds and lack of signal. Those are critical processes that usually need testing on the road, by real users or someone playing their role. Developers need a way to create realistic scenarios during the testing phase to verify how the APP handles them. So far, this would be done by just assigning a test team to go on the field and walk like a real user would do. It gets even more complicated in certain circumstances of displacement as when it involves driving to perform the aimed tests. All this process costs time and money. It is possible, however, to use pre-recorded GPS data and just do a playback in a simulator but that limits your testing flexibility by the amount and quality of pre-recorded data you have. Developers and testers should have a way of generating such simulations, without further expenses, from their labs and offices, with great flexibility and various editable parameters.

In section II are presented the existing solutions and what approaches they take to the problem but also what features they lack or could have been done better. In section III the new framework is presented along with details of every step in the process of creating simulations. Section IV shows a working example of the framework, from the beginning until the final simulation is executed on the device. Section V presents the final conclusions and what can be done to further improve our current project.

II. SMARTPHONE SENSORS; ACCELEROMETERS, GPS DATA AND SIMULATIONS

Current simulation solutions on the market are either too focused on the deep technical details of the GPS sensor – NMEAsoft GPS simulator [2] - or too simple when it comes to building the simulation – Lockito [3].

NMEAsoft GPS simulator is “a product to produce a virtual GPS data”. It is great when it comes to creating a custom track on the map, using just a few buttons. It allows setting up a

custom fixed speed and then generate the simulation. But it lacks on the ability to make the speed variable and to add other types of parameters like accuracy or altitude levels. It also includes in depth technical details of the GPS sensor and how it receives data from the satellites, which is not suitable for a standard developer aiming to perform tests.

Lockito on the other hand provides a simple interface using Google maps API and allows the user to save the created simulations on the app and view them as a list. It is used to make the smartphone to follow a fake itinerary, with total control over the speed and GPS signal accuracy and offers more parameters: altitude and location accuracy. The problem is that, as in the above mentioned NMEAsoft simulator, these parameters are fixed throughout the simulation. The track is being built directly on the device and cannot be easily transferred to other devices. Another limitation is the fact that it only allows the input of points between which driving directions are fetched, from Google Maps API on the roads, and not from custom pedestrian routes. This kind of limitations makes it impossible to create specific paths and custom turns in walking areas (e.g. a garden with no roads).

Both APPs lack the capability of exporting/importing open formats like GPX (GPS Exchange Format)[4] or GEOJSON[5].

Another solution, developers only, is an Android Studio plugin, Mock Location Plugin [6], which can generate a straight route between 2 geographical coordinates. It allows specifying the number of points to be generated and the time between them. It feeds the data directly to the Android OS through an ADB connection. This solution is even more simple than the ones before, allowing only a straight line and fixed speed.

Thus, summing up, most important feature lacking on the existing solutions is the ability to specify various parameters in different points of the simulation. As an example, an app that traces a person's daily routine would need a variable ground speed for the simulation which should allow points where the GPS signal is getting bad, but also slight deviations from the route due to changes in tracking accuracy. The solutions on the market right now only allow performing ideal simulations that have fixed parameters and that can instantly be recognized as being generated. Such are not the best candidates when the objective is to test a sensitive location based application executed in various scenarios.

The main idea driving this project is to maintain the simplicity of generating GPS data without caring about the technical details of the GPS sensor and satellite signal but at the same time being able to generate a simulation with parameters variable in time. A simulation generated with this project will hardly be distinguished from real recorded data. As a requisite, the user who makes the simulation must have control over the exact path generated and over the parameters at any point in time along the path. Afterwards, the generated simulation will be used for playback on Android devices. The player APP will use the Android sensors simulation framework to input the data directly into the operating system so that the whole ecosystem will use the generated location instead of the real one.

Google Maps will be the map API used for the creation of the tracks and simulations. The reason for the choice of Google Maps API is the availability of well documented sources and the support for it can be found everywhere on the web. Other alternative was OpenStreetMap API but this requires separate APIs to display the map and to get the routes. Although google maps API has a usage limit (25.000 map loads per 24 hours), such limit is high enough to provide an uninterrupted experience during the usage of the platform because it is only used when generating new simulations, and not while playing back the simulations.

III. Framework for App Development

The proposed system aims to have all the main features from the previous analysis in a simple and consistent user interface that incorporate dynamic parameter changes throughout the simulation. The simulation will be generated in a browser instead of directly on the phone, in order to have a centralized place where it can be seen, edit and exported to any number of devices from anywhere, thus using any computer without the need to install specific software. In Figure 1 it is possible to observe the framework that allows the preparation (left) and the execution (right) of the proposed system as next described.

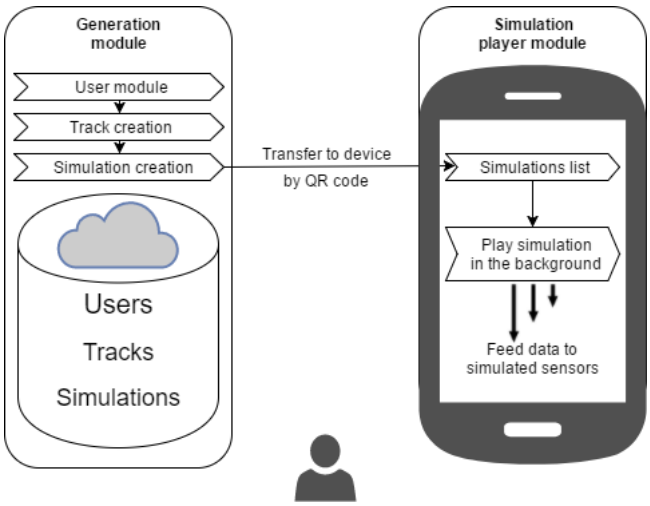

Figure 1: Framework diagram

The framework consists of two deliverable modules and one testing and development module. The deliverable modules are the Generation Module (web server), which will handle the creation and storage of the tracks & simulations and the Android Simulation Player, which will fetch the simulations from the web server and play them on the device using the sensors simulation framework from the operating system. The Testing and Development Module will be used to gather real life data from the sensors in order to compare them to the generated simulations and to improve the system to generate more real life like data.

The framework has 3 main entities

- Users
- Track – a user defined path on the map
- Simulations – the actual playback data starting from a path

Those can be observed in Figure 2, with example instantiations of what will be generated in a typical execution.

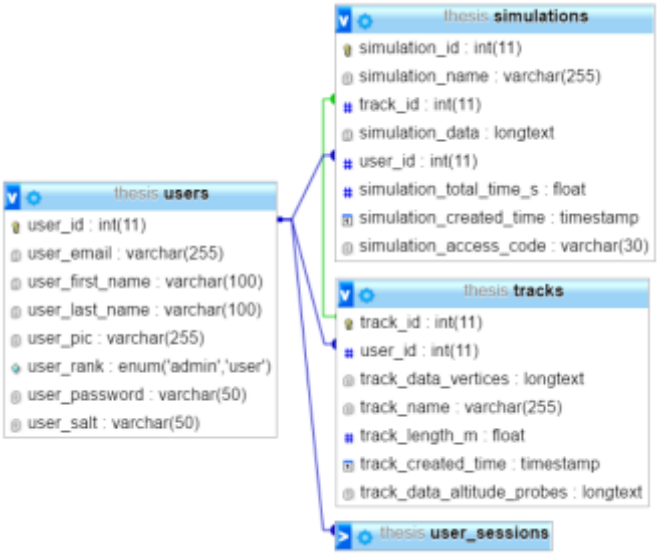

Figure 2: Data schema

On the webserver, the final user will have an account from where he can manage all the routes and simulations. For the GPS simulation, a map is displayed to the user with Google Maps Web API [7] with the tools to create a route.
The Generation Module, described in the next paragraphs, is the core of the whole system to create and simulate the track to be uploaded to the user device.

The Generation Module consists of the following submodules:

- **User module** – login and account creation
- **Track creation** – using Google maps directions API, the user can input a place of start and a destination and the best route is automatically created. If the route is not the desired one, the user can drag & drop the route to the desired one. After that there is a step of fine-tuning the route, where the user can edit every vertex of the generated route for precise movements. For this fine-tuning step, we need to split the path into smaller paths to allow smooth editing and not hang the browser. Splitting introduces new challenges to always keep the paths united into a single large path (whenever an edge vertex of the new split is being moved, the vertex corresponding to the neighbor path needs to move as well to not visually break the whole path). Finally, the track is saved in the user's account and can be exported to an open format – GPX [4] (GPS Exchange Format) or GEOJSON [5]. When the track is saved, altitude data is being requested from the API and saved along the track. This data consists of altitude values for a requested X equidistant points along the route. This will be later used in the simulation by interpolating the data to get the altitude value for any point in the track. For the best result X should be a value as large as possible but also should depend on the length of the track.
- **Simulation creation** – based on an already defined track, the user can create a simulation. The simulation consists of specifying the various parameters the project supports at any point during the route. This is done by selecting the desired points on the route. Total time of the simulation is calculated and displayed on the screen.
- **Simulation export** – using a generated QR code that contains a unique string ID related to the data, any simulation can be exported to the APP on the mobile phone.

The simulation player module, on Figure 1 (right), is execution in the user device and performs the play of the simulation as prepared before and allows the test to be performed as with values generated by the GPS sensors.
The simulation player module has the following functionalities:

- Display the available simulations
- Add simulation by QR code or unique string code
- Play any selected simulation on the device.

The APP will stay in the background all the time to give input to the simulated sensors. It will make use of the Android sensors emulation framework and 'trick' the whole OS into recognizing the simulation as real input data.

In order to execute the simulation, the proposed APP needs to feed the simulated sensors with regular input – location, altitude and others, and that will be refreshed at short intervals of time. To make this as believable as possible the intervals need to have a small random added value. For this, when the final simulation is generated on the server, every input for the simulated sensor will be generated. The result will be a few inputs per second of simulation. When playback occurs, the Android player APP has to wait for the amount of time until the next input and feed it to the simulated sensors. For the simulation to run flawlessly, the APP runs a separate service which handles the input to the simulated sensors. The service must run in "foreground mode" to prevent it from being killed by the system. Even if the APP is terminated by the OS because of low memory, the service still runs and continues the emulation. It can always be stopped by going back in the APP.

The technologies used for the development of the framework are PHP 7, MySQL – with JSON support, JavaScript ES6 &

ES2017 – for the server/browser side and Java for the development of the native Android application.
The server side has a JSONRPC endpoint, which is used by the APP to retrieve data but also by the simulation generation framework for the JavaScript client side to communicate with the server and fetch or save data.

The track data and simulation data is stored in the DB as JSON. While a NoSQL DB might seem better, MySQL introduced native JSON support in the last versions, making it perfectly usable for this purpose.

Next in Figure 3 is presented the format of the stored points, which will represent the track in the same format as those generated by the GPS sensors and are stored in a local database transferred for the user's device.

A stored point in the track has the following format:

```
{
   "elevation":36.4005241394043,
   "location":{
      "lat":40.79057,
      "lng":-73.9692
   },
   "resolution":19.08790397644043
}
```

Figure 3 - Sample of a stored point

Even though geolocation data is being compressed before the call, when requesting altitude data for a track with many vertices, it is necessary to simplify the path because of the API limits (the endpoint for getting altitude data only accepts GET requests which are limited to around 8000 characters). To solve that limitation it is used the Ramer–Douglas–Peucker [8] algorithm which, given a Polyline (a curve composed of line segments), finds a similar Polyline which consists of a subset of the points that defined the original one.
When generating the final simulation points, elevation data from the samples collected at the track creation step is being interpolated to the generated points so that each point will have an approximation of the real elevation. Linear interpolation is being used for this.

IV. PILOT

With the built framework, a user account can be created and registered users are able to login as depicted in Figure 4.

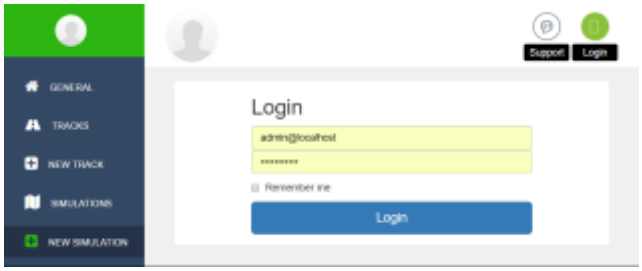

Figure 4: Login screen

After login, a new track can be created. When creating a new track, it is possible to input a specific address or just click on the map to set the desired start/end location. Also, the user can select if he wants to walk or transit and driving directions. The generated path is then manually editable, vertex-by-vertex, using drag and drop, Figure 5.

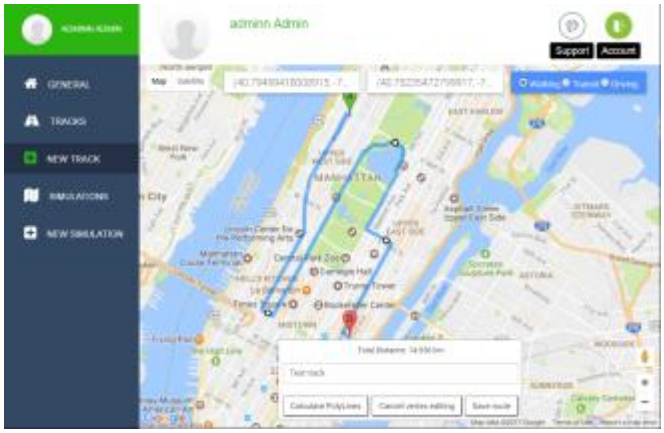

Figure 5: Route creation

It is possible also to view or edit existing tracks. When selecting a previously created track, there is an option to edit it or to create a new simulation starting from that path, Figure 6.

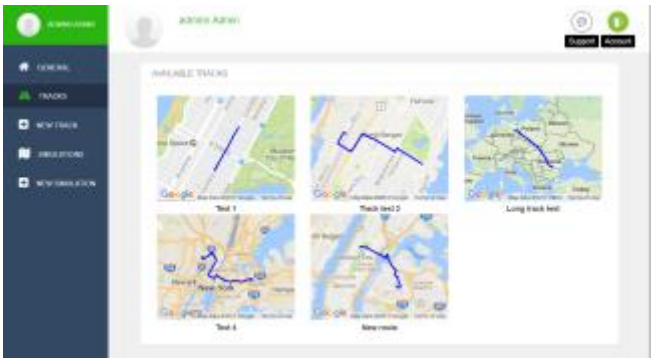

Figure 6: Tracks view

Based on an already created track, create a simulation can be created. Here any number of points can be added to the simulation by clicking on the path or near the path where we want the new point to be added and specify what speed we want to have, the tracking accuracy and altitude offsets. A process that can be performed with low effort, Figure 7.

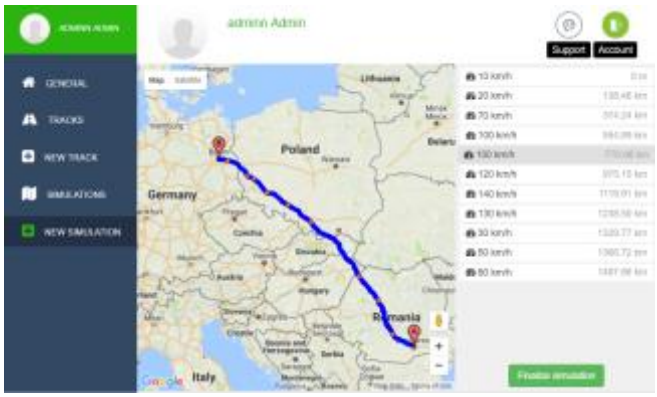

Figure 7: Simulation creation

When finalizing the simulation, data can be transferred to the device using the generated QR code and from the device APP we run the simulation, making the operating system use the prepared input in the same manner, as it would be by a normal displacement using the smartphone built-in sensors. That makes a complete framework where simulation can be prepared, loaded and executed in a smartphone environment.

V. Conclusions and Future Work

In conclusion, the framework as presented is a seamless choice for a developer who wants to thoroughly test an unlimited number of developed location based APP in various scenarios. The developer has the freedom of creating exactly the desired route and to control the variables of the simulation at any point. Furthermore, those tracks can be stored with the created parameters for each track and, based on it, create a set of multiple reusable and different track simulations. Another benefit is that the final simulations can be exported to Android devices used for testing by just scanning a QR code or by inputting the simulation ID. The presented work opens the opportunity for testing and executing software based on GPS simulated data, allowing the creation of many alternative simulated routes without the need of hiring people to perform such tests with the benefits of readiness and cost savings. The presented framework brings additional value to the market and to the application's technological development laboratories.

Future work can also incorporate generation of accelerometer data – to be able to simulate when a user is walking, driving, cycling or even sudden events like when a person falls on the ground or when a crash happens. The proposed solution can be explored in a CARELINK Project, where tracks can be established and simulated to ensure safety for older people suffering from Alzheimer or other kind of Dementia.

Acknowledgment

The research leading to these results has received funding from the European Union ERASMUS+ Program under grant agreement EAC/A04/2014 ACACIA. The work has also been promoted under the project CARELINK, AAL-CALL-2016-049 funded by AAL JP, and co-funded by the European Commission and National Funding Authorities of Ireland, Belgium, Portugal and Switzerland.

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 *3rd workshop on positioning, navigation and communication*, 2006, pp. 209–218.

[2] "software based gps/nmea simulator." [Online]. Available: http://www.nmeasoft.com/. [Accessed: 22-May-2017].

[3] "Lockito – Fake GPS itinerary - Android Apps on Google Play." [Online]. Available: https://play.google.com/store/apps/details?id=fr.dvilleneuve.lockito. [Accessed: 22-May-2017].

[4] "GPS Exchange Format." [Online]. Available: https://en.wikipedia.org/wiki/GPS_Exchange_Format. [Accessed: 22-May-2017].

[5] "GeoJSON." [Online]. Available: http://geojson.org/. [Accessed: 22-May-2017].

[6] "Android Studio. Simulate multiple GPS points with Mock Location Plugin – Jesús Amieiro." [Online]. Available: http://www.jesusamieiro.com/android-studio-simulate-multiple-gps-points-with-mock-location-plugin/. [Accessed: 02-Jun-2017].

[7] "Google Maps JavaScript API | Google Developers." [Online]. Available: https://developers.google.com/maps/documentation/javascript/. [Accessed: 22-May-2017].

[8] "Ramer–Douglas–Peucker algorithm," Aug-. [Online]. Available: https://en.wikipedia.org/wiki/Ramer–Douglas–Peucker_algorithm. [Accessed: 22-May-2017].