FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Bus travel time estimation using mobile devices

Ana Filipa Barroso Pinto

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Teresa Galvão Dias

February 28, 2019

Bus travel time estimation using mobile devices

Ana Filipa Barroso Pinto

Mestrado Integrado em Engenharia Informática e Computação

February 28, 2019

Abstract

Automatic Vehicle Location is a combination of technology and intelligence allowing the determination and communication of the geographic position of a vehicle. In major cities, public transport operators install dedicated in-vehicle devices (AVL unit) to determine the GPS position and communicate to a back-office component. The GPS position and the planned schedule of each bus trip are used together to estimate the travel time of each bus, allowing to provide real time information to the users.

However, the costs involved in the acquisition and installations of such a system make it unaffordable for smaller public transport operators. The idea of this dissertation is to study how a mobile device (smartphone) equipped with GPS positioning could be an effective alternative to the traditional systems.

Smartphones are equipped with sensors such as GPS, accelerometers, gyroscopes and magnetometers. Furthermore, they cost only a fraction of the AVL units. The development of a solution to replace these devices using smartphones can cut costs dramatically, which, as a result, could allow smaller transportation companies to also provide their clients with real time estimations, allowing the commodity and usefulness of this information to be available to many more people. Furthermore, this information allows operators to manage and analyze their vehicles in real time.

This dissertation intends to analyze and review the technology available to automatic vehicle location, in particular using mobile devices; design an architectural model and a system that will be able to collect and integrate GPS traces using mobile devices in order to provide estimates of real time schedules taking into account the planned timetables and network information; design and implement an algorithm to produce accurate estimates of route schedules and, finally, test the achieved solution using real world data.

Resumo

Automatic Vehicle Location é uma combinação de tecnologia e inteligência que permite a determinação e comunicação da posição geográfica de um veículo. Nas grandes cidades, as operadoras de transportes públicos instalam dispositivos dedicados (unidades de AVL) dentro dos seus veículos de forma a determinar a posição de GPS e comunicá-la a um componente de *back-office*. A posição de GPS e o horário planeado de cada viagem são usados em conjunto para estimar o tempo de viagem do autocarro, permitindo fornecer aos utilizadores informação em tempo real.

No entanto, os custos envolvidos na aquisição e instalação tornam estes sistemas insustentáveis para empresas de transportes públicos mais pequenas. A ideia desta dissertação é estudar como um dispositivo móvel (smartphone) equipado com GPS, entre outros sensores, pode ser uma alternativa eficaz aos sistemas tradicionais.

Os *smartphones* estão equipados com sensores tais como GPS, acelerómetros, giroscópios e magnetómetros. Além disso, estes custam apenas uma fração das unidades de AVL. O desenvolvimento de uma solução capaz de substituir estes aparelhos por smartphones pode reduzir bastante os custos associados, o que, por sua vez, pode permitir que as companhias de transportes mais pequenas possam também fornecer aos seus clientes informação em tempo real, fazendo com que a comodidade e utilidade desta informação esteja disponível para muitas mais pessoas. Além disso, essa informação permitirá aos operadores gerir e monitorizar as suas viaturas em tempo real.

Com esta dissertação pretende-se analisar e rever a tecnologia disponível para a *Automatic Vehicle Location*, em particular, usando dispositivos móveis; conceptualizar um modelo de arquitetura e um sistema capaz de recolher e integrar dados de GPS, através de dispositivos móveis, de forma a fornecer estimativas de horários em tempo real, tendo em conta os horários planeados e informação da rede; conceptualizar e implementar um algoritmo capaz de produzir estimativas de horários de rotas precisas e, finalmente, testar a solução encontrada através de informação do mundo real.

Acknowledgements

I would like to thank everyone who helped me to complete this dissertation and achieve this milestone in my academic path. Without these people, none of this would ever be possible.

First of all, I would like to acknowledge my endless gratitude and admiration towards my parents, for their unconditional love and for everything they sacrificed to support me. I would also like to thank and acknowledge the support, advice and guidance of my supervisor, Teresa Galvão, who helped me to achieve all this work. Moreover, I would like to thank all my friends who have been with me throughout my academic path and my life, especially João Nogueira, who always offered to help, not only during the progress of this dissertation but also throughout our higher education path, and André Barros, who helped me to conceive the smartphone application that is part of this work. I would also like to thank my boyfriend António Silva, for his encouragement and his support, which goes beyond words. Finally, I would also like to thank *Optimização e Planeamento de Transportes S.A.* for the information that helped me to consolidate my work and for the availability demonstrated.

Filipa Barroso

"Let me say quite categorically that there is no such thing as a fuzzy concept... We do talk about fuzzy things but they are not scientific concepts. Some people in the past have discovered certain interesting things, formulated their findings in a non-fuzzy way, and therefore we have progressed in science."

Rudolf E. Kálmán

Contents

List of Figures

List of Tables

Abbreviations

AVL	Automatic Vehicle Location
GPS	Global Positioning System
WPS	Wi-Fi Positioning System
GSMPS	Global System for Mobile Communications Positioning System
LED	Light Emiting Diode
GSM	Global System for Mobile Communications
TTP	Time Travel Prediction
SVR	Support Vector Regression
kNN	k Nearest Neighbor
PPR	Project Pursuit Regression
ANN	Artificial Neural Network
ETA	Estimated Time of Arrival

Chapter 1

Introduction

1.1 Context

Automatic Vehicle Location (AVL) is a combination of technology and intelligence allowing the determination and communication of the geographic position of a vehicle. In major cities, public transport operators install dedicated in-vehicle devices (AVL units) to determine the GPS position and communicate to a back-office component. The GPS position and the planned schedule of each bus trip are used together to estimate the travel time of each bus, allowing to provide real time information to the passengers and used to monitor and control vehicle operations.

However, the costs involved in the acquisition and installation of such a system make it unaffordable for smaller public transport operators. The idea of this dissertation is to study how a mobile device (smartphone) equipped with GPS positioning could be an effective alternative to the traditional systems.

1.2 Motivation and Goals

Smartphones are everywhere. In 2016, around 2.10 billion people worldwide used a smartphone [eMa16]. These devices are equipped with sensors such as GPS, accelerometers, gyroscopes and magnetometers. Furthermore, they only cost a fraction of the AVL units price. The development of a solution to replace these devices using smartphones could cut costs dramatically, which, as a result, could allow smaller transportation companies to provide real time information to their clients, allowing the commodity and usefulness of this information to be available to many more people. Furthermore, this information also allows companies to manage and monitor their vehicles in real time.

This dissertation intends to analyse and review the current technology available in Automatic Vehicle Location, particularly using mobile devices; design an architectural model and a system that will be able to collect and integrate GPS traces using mobile devices in order to provide real time estimates of bus passings; design and implement an algorithm to produce accurate estimates of route schedules and, finally, test the achieved solution using real world data.

1.3 Dissertation Structure

In addition to this introduction, this dissertation has 5 more chapters. In chapter 2, the literature review is presented and we reference some related works. In chapter 3, we go over the architecture of the proposed solution. In chapter 4, we go into detail about how our Travel Time Prediction algorithm was designed and implemented. In chapter 5, we present the results of the GPS measurements collected by our smartphone application and the results of our TTP algorithm, and discuss those results. Finally, in chapter 6 we go over our conclusions and the future work.

Chapter 2

Literature Review

2.1 Introduction

This chapter provides some information about the concepts of AVL and GPS, as well as their relevance. Furthermore, some methods and algorithms for the prediction of travel times are analysed. Finally, we go over some Time Travel Prediction applications, by reviewing the different information types to take into account in travel time prediction and methods to obtain it and reviewing some works with a focus related to that of this dissertation.

2.2 Global Positioning System (GPS)

Global Positioning System is a satellite-based radio navigation system constituted by a network of satellites, that provides information about its position to a mobile receiving device. This information is available independently of atmospheric conditions, time of day or location, as long as the receiving device has a minimum of 4 satellites in its reach.

The concept of GPS is based in the time and the position of specialised satellites. Satellites are continuously broadcasting information about their position and time, with the latter being made possible by extremely accurate atomic watches, whose small deviations with the real time are corrected daily.

On the other hand, the receiving devices monitor several satellites and solve equations to determine their exact location and the deviation with the real time. The position is calculated through a method known as triangulation. Knowing the distances between the receiving device and 3 satellites, which represent a radius in which the receptor may be positioned, the receiving device position is obtained through the intersection of the 3 radius, as represented in Figure 2.1. A fourth satellite is used to avoid possible small mistakes caused by deviations in the clocks.

The continuous calculation of a mobile device through GPS has an elevated energetic cost. However, smartphones are equipped with sensors, such as accelerometers, capable of identifying user activity. There are also some alternatives to determine a device's location, such as Wi-Fi Positioning System (WPS) and Global System for Mobile Communications Positioning System

Figure 2.1: Triangulation process of a GPS system

(GSMPS). However, these systems are not as accurate as GPS. WPS and GSMPS precision measures around 20-30 meters and 70-200, respectively. These uncertainties make these methods less adequate for situations that require accurate results.

2.3 Automatic Vehicle Location

Automatic Vehicle Location is the determination of the geographical position of a vehicle and its communication to a monitoring system, in order to provide information about the vehicle trips.

This technology is commonly used for vehicle fleets and driver crews management in multiple contexts such as emergency crews and vehicles, service vehicles or public transportation fleets. For instance, regarding emergency vehicles and crews, this technology can be employed to determine the nearest crew to a reported emergency.

On the matter of public transportation, this technology is used to provide passengers real time information about buses estimated arrival times to stations. Furthermore, the information gathered can be useful in decision making support and real time operations control.

With the increase of traffic jams in large cities and, consequently, the demand for public transportation, many public transportation companies have been trying to improve their operations by investing in this promising technology for services planning, schedules management and performance analysis [FHMS03].

Nowadays, AVL systems installed in buses include the central capacity of locating a vehicle, as well as other features, some of which obtained through sensors and devices in the bus. Some examples of these features are [BNAoSM08]:

- management of the fare collecting boxes and of the LED signs that show the bus destination
- next stop voice announcement, automatically activated when the bus is close to the next stop
- automatic passenger counting
- display of monitoring messages about the vehicle state
- sending emergency messages, sometimes with the help of a microphone
- providing real time information about estimated buses arrival times through screens with dynamic messages

These features are made possible by all the sensors and components in an AVL device, such as a GPS receiver and antenna, "dead reckoning" devices to complement the GPS receiver for vehicle positioning, a vehicle logic unit computer, wireless local area network (WLAN) card and antenna, an automated passenger counting subsystem and a transit signal priority emitter [BNAoSM08].

One of the data types obtained by AVL systems is location. In most cases, location is determined through GPS. The most common channel to broadcast this information is Global System for Mobile Communications (GSM). This is due, mostly, to the low amount of data required by AVL systems and the low cost and the almost omnipresent nature of this public network [PM10].

Despite the benefits and the possibilities this technology brings, the costs involved in the acquirement and installation of the most common devices for this technology make these systems unsustainable to smaller public transportation companies, hence the relevance of alternatives to the devices used nowadays.

2.4 Methods and algorithms for Travel Time Prediction

Time Travel Prediction (TTP) is one of the most common problems in transportation [MMMMDG15].

In large cities, estimated times for the arrival of public transportation vehicles to their stations are frequently displayed in screens (Figure 2.2) and also provided through smartphone applications.

This information is, in most cases, obtained through AVL and it allows to increase passengers satisfaction and reliance in public transportation. TTP can be used in several other contexts, such as vehicle fleets management, individual navigation, logistics, monitoring and control.

Figure 2.2: Screen with buses arrival times

There are several approaches to solve TTP problems. Moreira-Matias *et al* [MMMMDG15] divide those methods into 4 categories:

- Machine Learning and Regression Methods
- State-Based Models and Time Series
- Conservation Equations and Dynamic Traffic Models
- Historical Databased Models

These approaches are summarised in the next sub-sections.

2.4.1 Machine Learning and Regression Methods

These methods are proposed to infer arrival times (dependent variable) through a mathematical function based on a set of independent variables.

In addition to their application in TTP, regression methods are also used to analyse the impact of each independent variable on the target variable. Complex models such as SVR, kNN, PPR and ANN are the most popular approaches in this kind of analytics [MMMMDG15].

Despite some advantages, like the long training process and the overfitting possibility, ANN is the most successful regression method for the context of TTP. [DHNM99], [JR05] and [Gur10] used this method with location based data, while [CLXC04] and [Pat06] used automatic passenger counting data.

Recently, promising models based on trajectories that use machine learning techniques have been proposed to solve TTP. [TJ08] presented a nearest neighbour trajectory technique that identifies the most historically similar trajectory to the current partial trajectory of a vehicle. A time travel prediction is then provided, inferring the future trajectory of a vehicle.

2.4.2 State-Based Models and Time Series

This type of approaches is based only on the most current data samples, disregarding the remaining data [MMMMDG15]. Time series models assume that the travel time of a vehicle is a combination linear/non-linear of their historical values. State-based models frequently assume that the future state of the dependent variables is only conditioned by the most recent states.

This group of methods is less dependent on the amount of data, in comparison to the other groups. Furthermore, they do not need a large training period, since they are, for the most part, online training algorithms. These algorithms are known by their capability to react to unexpected events, such as heavy rains, traffic jams, car accidents and sports events. Consequently, they are powerful short term predictors, thanks to their capability of learning and updating in real time. However, their performance decreases when faced with long term prediction problems.

Despite their recurrent use to traffic flux prediction, time series models are unusual in the context of buses time travel predictions. One of the possible explanations is their high sensibility to changes in the relation between historical data and real time, especially when a fixed distribution of the data is assumed [CC08].

State-based models are frequently used in TTP problems, due to their capability to handle with traffic jams. The most common state-based model is Kalman filter [CD03], [CLXC04], [WD99]. The main advantage of this method when compared to Markovian approaches is its ability to filter noise in the data, which is of extreme importance in online learning techniques for TTP problems.

2.4.3 Conservation Equations and Dynamic Traffic Models

This set of techniques applies relations between traffic variables, obtained through traffic flux theory, to predict travel times through flow data [Mor06]. These relations seek to formulate the travel time between two points as the sum of the travel times between several segments, such as streets, belonging to the route to be taken. These predictions focus on traffic density (vehicles/km) estimations, which, in turn, are based on the difference between the arrivals and departures of vehicles in each segment to analyse [MMMMDG15].

In order to correctly apply these methods, a good knowledge about them is essential, since the reliability of the predictors depends on the skills of the user to establish relations between travel time and the related selected input features [Mor06]. Moreira-Matias *et al* [MMMMDG15] state that only recently has someone worked with these models in the context of TTP through AVL [DZZ13]. These methods are more frequently employed to estimate urban traffic conditions than to TTP.

2.4.4 Historical Databased Models

This last set of approaches consist in simple averages and other types of Poisson processes whose average travel time or average speed depend on the type and/or time of day.

The simplicity of these methods is pointed as a downside in the matter of representing the complex relations between travel times and other urban public transportation variables. Consequently, these methods are a weak approach to TTP problems.

2.5 Travel Time Predictions applications

2.5.1 Different information types

Besides the TTP algorithms, it is also important to know the different methods and types of information that can be used in TTP problems. Although it is intended, with this dissertation, to obtain GPS information through a smartphone installed in each bus and, with that, to predict the buses arrival times to their stops, there are other approaches worth mentioning.

One of these methods is the passenger community participation, as described by Zhou *et al* [ZZL12]. Instead of GPS information, the researchers use information such as cell towers signals, movement states, audio recordings, among others. Furthermore, the users of the developed platform were also questioned about the arrival time of the buses they used. All this information was stored in a server and posteriorly processed, with the goal of monitoring the bus routes and predicting their arrival times. The results of the evaluation performed to the system, over the course of 7 weeks, demonstrated that this system can predict arrival times with accuracy.

Another type of information useful to TTP problems is the events that are going to happen in a city, such as concerts, sports events, parades and conferences. These special events have a potential disruptive impact in transportation networks, since they represent a public transportation demand high above usual. Through information about these events available online and with the use of data mining algorithms, Pereira *et al* developed a model capable of predicting the arrival times of public transportation vehicles to areas where there are happening special events [PRBA14]. The authors of this study conducted a test to their model in Singapore and achieved promising results.

2.5.2 Related Works

There are some previously developed works in this area that might be an added value and a good support base to the development of this dissertation.

One of these works is a Master's thesis by a *Faculdade de Engenharia da Universidade do Porto*'s alumni [Sea17], who intended to obtain TTP using a smartphone application for collecting GPS measurements. However, the results of this dissertation were not thoroughly explored.

Another work in the context of using smartphones to obtain GPS data is that of Herrera *et al* [HWH+10]. For their article. Herrera *et al* developed an application, tested in 100 smartphones, each allocated to a vehicle moving in loops over a 10 km highway segment. The results were promising and the authors concluded that with the help of smartphones, it was possible to

reach their goals. However, this study was conducted with the goal of monitoring traffic flow, not predicting travel times.

Precisely in the domain of public transportation TTP and obtaining GPS information using smartphones, Biagioni *et al* developed EasyTracker[BGME11]. In their article, the authors describe how their application is able to infer the line and schedule of a bus through several algorithms, without the requirement of any kind of input from the user. Although the authors were able to infer buses routes and schedules with positive results, the algorithm employed to predict the arrival of a bus to its next stop is fairly simple. This algorithm assumes that all buses are equipped with a smartphone running the application and then, through its location, calculates the arrival of bus nearest to the given station through simple averages.

2.6 Conclusions

Through this literature review, we can verify the usefulness and relevance of AVL systems in public transportation, bringing essential information to passengers, drivers and companies.

From the revision of the algorithms for TTP problems, it is possible to conclude that the Kalman filter is one of the most relevant algorithms and that implementing them produces accurate results. This is also the most commonly used algorithm by traditional AVL systems [Sea17].

The works related to this dissertation not only provide a good foundation for the development of this work, but also show it is possible to obtain the GPS information needed in order to solve TTP problems, through mobile devices. The conclusion is, then, that there is room for improvement in AVL systems through mobile devices, a solution capable of cutting costs and bringing this technology and its benefits to a higher number of users.

Chapter 3

Proposed system architecture

3.1 Introduction

As previously seen, the current AVL devices in use are expensive and only available to the large public transportation companies with the budget for this system. Therefore, only passengers in large cities have access to real time information about the Estimated Time of Arrival of their public transportation vehicle.

In order to help solving this problem, we propose a solution that uses smartphones to collect the required data for TTP, replacing traditional AVL devices, and to inform users of the estimated times of arrival of the buses going to their stops.

3.2 User requirements

The first step adopted in the planning of the system was to identify the different types of users, the system platforms, as well as the platform each user interacts with. We identified three types of users for our system: passengers, drivers and administrators, each of which will interact with a different platform. After that, we proceeded to the elicitation of the requirements, resorting to two common elicitation techniques [PR15]: observation and brainstorming. The different types of users, the platforms they interact with and their requirements are enumerated in the next three sub-sections.

3.2.1 Passenger requirements

The passenger is a generic user with access to public information. He interacts with the passenger application only. The requirements for this type of user were mostly gathered by the observation of an existing application that allows users in Porto, Portugal to access estimated arrival times for their public transportation stops: MOVE-ME.AMP (http://www.move-me.mobi/).

The user stories concerning this user can be seen in Table 3.1

Identifier	Name	Priority	Description
US101	Search by name	very high	As an user, I want to search a stop by its name, in order to obtain information about it.
US102	Search by code	very high	As an user, I want to search a stop by its code identifier, in order to obtain information about it.
US103	Go to history	medium	As an user, I want to access my stops viewing history, in order to easily consult recent stops.
US104	Save as favourite	low	As an user, I want to save a stop as favourite, in order to easily consult a frequent stop.
US105	Go to favourites	low	As an user, I want to access my favourite stops, in order to easily consult frequent stops.
US106	View arrivals	very high	As an user, I want to view the estimated time of arrival of the next buses stopping at my station.
US108	Update stop page	high	As an user, I want to refresh a stop page, in order to view the most recent information.

Table 3.1: User requirements for passengers

3.2.2 Driver requirements

The driver/operator is the user responsible for setting the Driver application to gather data about a bus trip. It is intended for the interaction between this user and this platform to be as easy and short as possible, since the user should be focused on driving the bus, thus the need for an interface that provides little to no distraction. The requirements for this user's needs were collected over the course of some brainstorming sessions.

The user stories translated from those requirements can be seen in Table 3.2

3.2.3 Administrator requirements

The administrator is an authenticated user with access to all the information of the system. This user interacts with the Backoffice platform in order to monitor the system, the devices connected to it and the data gathered by all devices. The requirements for this user's needs were also collected over the course of some brainstorming sessions.

The user stories translated from those requirements can be seen in Table 3.3

Identifier	Name	Priority	Description
US201	Indicate line	very high	As an operator, I want to indicate the line I'm going to go through, in order to allow the app to gather data about my path.
US202	Indicate direction	very high	As an operator, I want to indicate the direction of a line I'm going. in order to allow the app to gather data about my path.
US203	Data gathering feedback	high	As an operator, I want to have access to information about the data gathering process, in order to know if the app is gathering data without any issues.
US204	Error notifications	low	As an operator, I want to get a notification about errors/warnings, in order to know if there is a problem even if the app is minimised.
US205	Terminate/Cancel trip	medium	As an operator, I want to terminate/cancel a trip whenever the data gathering should be stopped.

Table 3.2: User requirements for drivers

Identifier	Name	Priority	Description
US301	View devices	high	As an administrator, I want to see the number of devices gathering data, as well as the line and direction they're going, in order to monitor the data gathering.
US302	View errors	high	As an administrator, I want to access a list of the errors/warnings emitted by the devices connected to the system, in order to monitor the data gathering.
US303	View gathered data	very high	As an administrator, I want to view the data gathered by devices, in order to check possible errors or discrepancies.
US304	View estimates	very high	As an administrator, I want to view previously calculated estimates, by chronological order (most recent first).
US305	Compare estimates and passings	very high	As an administrator, I want to access a comparison between the calculated estimates and the actual time at which buses passed by a stop
US306	Ask for estimate	high	As an administrator, I want to request the calculation of an estimate for a stop, in order to obtain estimates even if there are no estimates requests by passengers.

Table 3.3: User requirements for administrators

3.3 Architecture

The proposed solution is composed by the three user interfaces mentioned in the last section, a server and a database, as illustrated in Figure 3.3.

The three user interfaces never communicate with each other. Instead, they communicate with the server, which is responsible for managing the gathered data and providing information to the user interfaces.

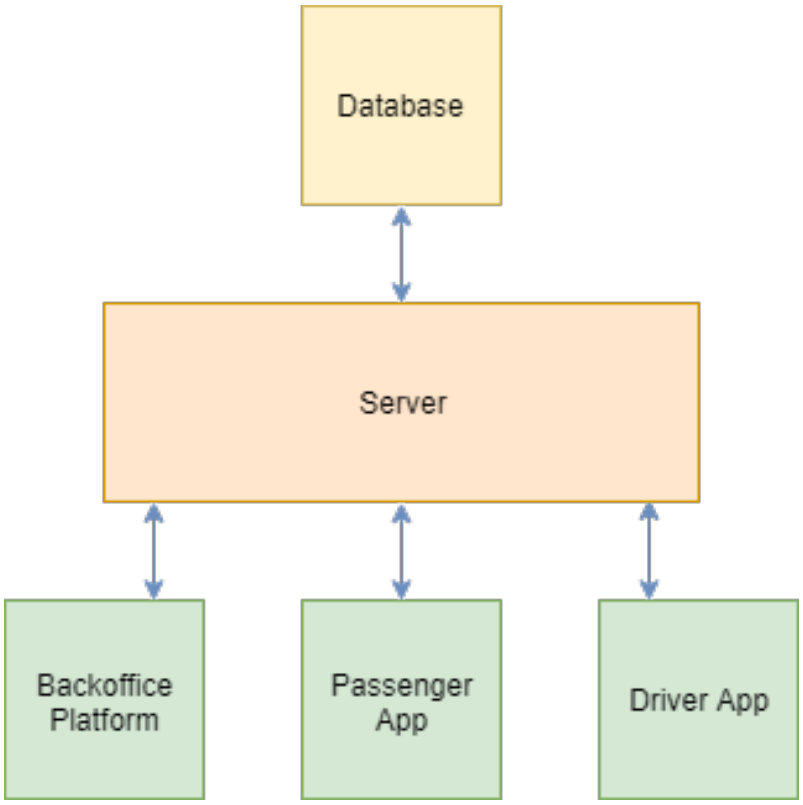

Figure 3.1: System's architecture

3.3.1 Database

The database stores all the gathered data in the system. The database communicates with the server in order to store new data and to retrieve stored information when requested. This data includes information about:

- Bus lines: the code, name, direction and total length of all lines, as well as the sequence of stops each line goes through
- Stops: the code, name, latitude and longitude of each stop
- Schedules: information about scheduled bus passing times, including a line's code and direction, a stop code, a scheduled passing time, the start and end date for that schedule, as well as the day of the week that the scheduled passing time refers to

- Bus location updates: each stored location update contains an unique identifier for each bus trip, an identifier for the device that collected that GPS update, a latitude, longitude and time-stamp, as well as a line code and line direction
- ETA predictions: for each prediction calculated by the system, the database stores the stop code it refers to, the unique identifier for the bus trip for which the algorithm is trying to apply a prediction, the line's code and direction and, of course, the estimated time of arrival

In Figure 3.2, we have an UML class diagram for this database and its components.

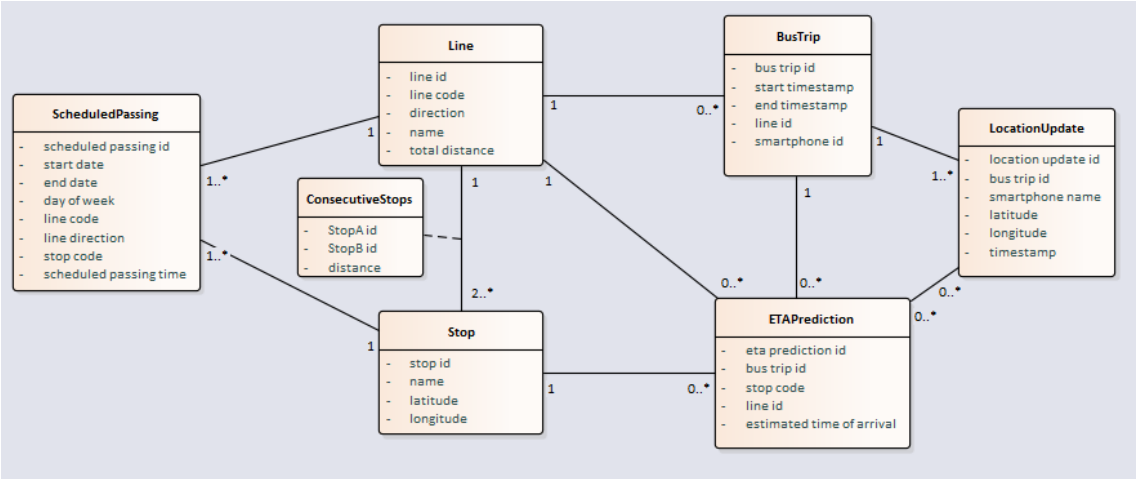

Figure 3.2: UML class diagram for the system's database

3.3.2 Server

The server is responsible for accessing the database in order to provide data to the user interfaces and to store the information being gathered by the driver application. It also communicates with the Backoffice platform by providing information about the system to the administrator, such as connected devices, gathered data, calculated estimates and comparisons with vehicles passing times. The server is also the system component where the arrival times estimates are calculated before they are provided to the passenger application. These predictions are calculated through an implementation of a Kalman filter,

3.3.3 Backoffice platform

The backoffice platform is a management tool for the system administrator. It aggregates a set of tools to monitor the mobile devices connected to the system (as seen in Figure 3.3), the gathered data and the calculated estimates. To obtain these estimates, the backoffice platform communicates with the server.

The backoffice platform also allows the user to ask the system to calculate estimates, in order to test the prediction algorithm and to have more data for its analysis.

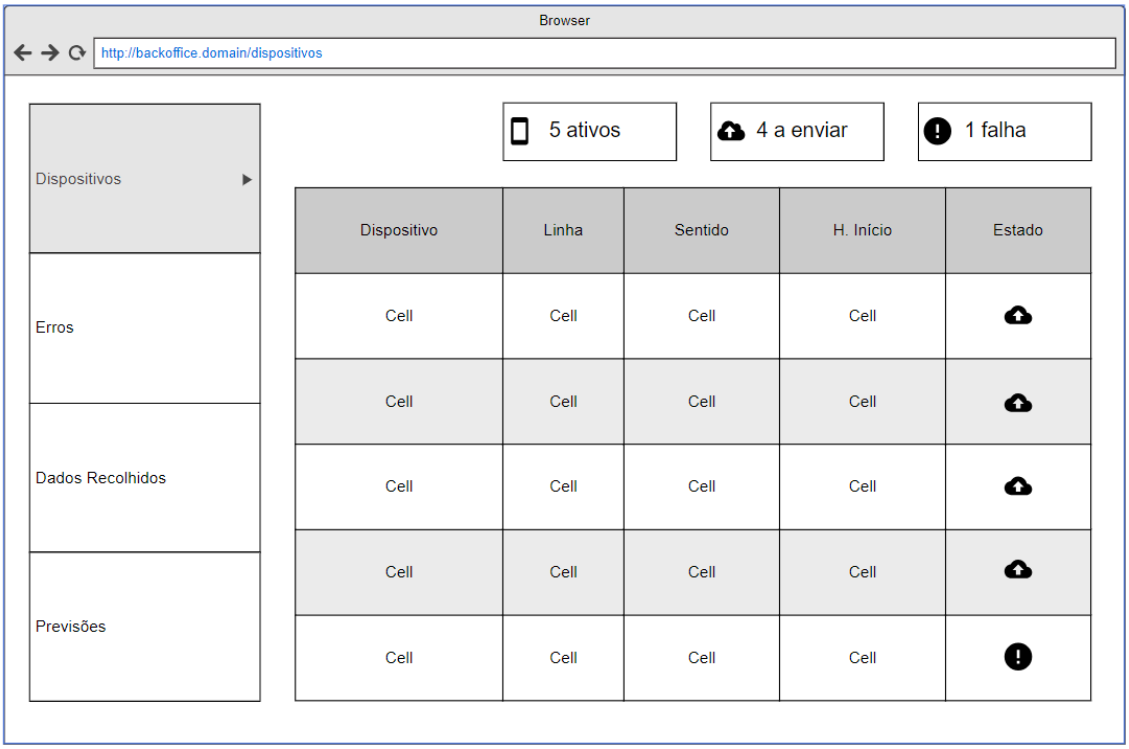

Figure 3.3: Mock-up of the backoffice platform's devices page

3.3.4 Passenger application

The passenger application allows users to view estimated arrival times of all buses who are going to pass through a stop for the next minutes, as seen in Figure 3.6. The user can either access the estimated arrivals to a stop by searching for a stop name or code and by accessing his history and his favourites, as we can see in Figure 3.4. The application then communicates with the server in order for it to provide this information.

3.3.5 Driver application

The driver application is the component of the system responsible for sending a vehicle's information to the server, in order for the latest to be able to calculate predictions. The data gathered by the application contains the line and the direction of a vehicle, its GPS positions through its course, as well as a time stamp for each GPS position.

In order to allow the device to collect the required data, the user needs to indicate the line and direction he is about to take and to click on a button to start the data collection for that trip. If the data collection process is occurring without any problem, the application should display a success loading icon, as seen in Figure 3.7. Otherwise, the application displays error messages regarding the problems affecting the collection process.

In Figure 3.5, we have the state diagram for this application.

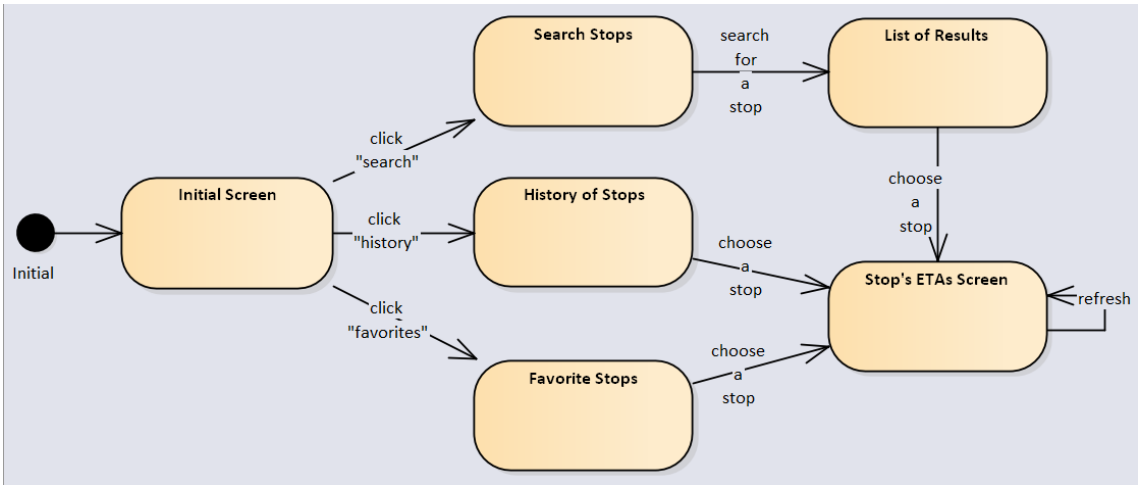

Figure 3.4: State diagram for the passenger application

3.4 Conclusions

With the proposed solution, TTP can be applied to public transportation networks with substantially reduced costs, compared to traditional AVL devices. In this chapter, we went over all the system requirements and its architecture, providing a solid foundation for the solution implementation.

The driver application allows the system to obtain the necessary information for the calculations of estimates with minimal input from the driver. This is an important feature as the driver should be able to do his job with as little distraction as possible. The greatest obstacle to this part of the solution is battery depletion. Due to the high GPS usage and the need to have a mobile data connection through the entire trip, most smartphone batteries would deplete in a few hours. However, this problem is easily solved by having the smartphone connected to a lighter charger or a power bank.

The passenger application allows users to know the calculated estimates with ease, thanks to the several options available in order to access a stop's page and the simplicity of the application, with no log-in required and no unnecessary menus and options.

Finally, the backoffice platform provides the administrators with an overview of the system and the tools to analyse the TTP algorithm's results.

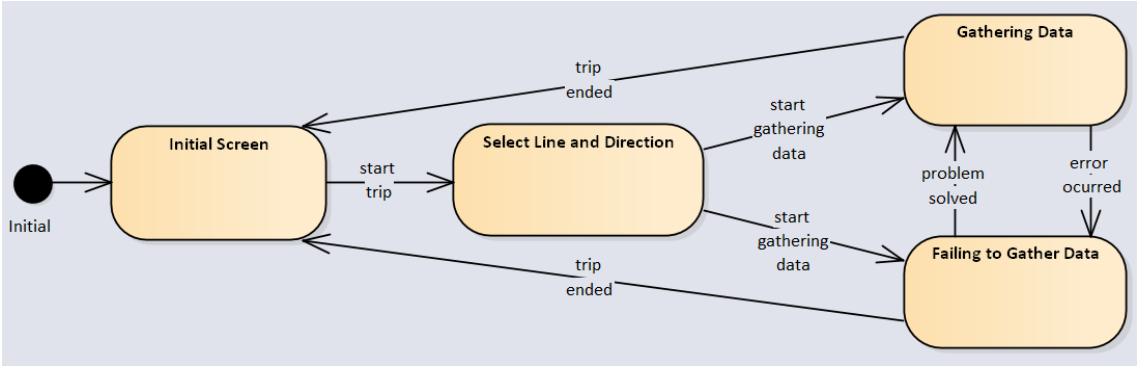

Figure 3.5: State diagram for the driver application

Figure 3.6: Mock-up of a passenger application's page for a stop

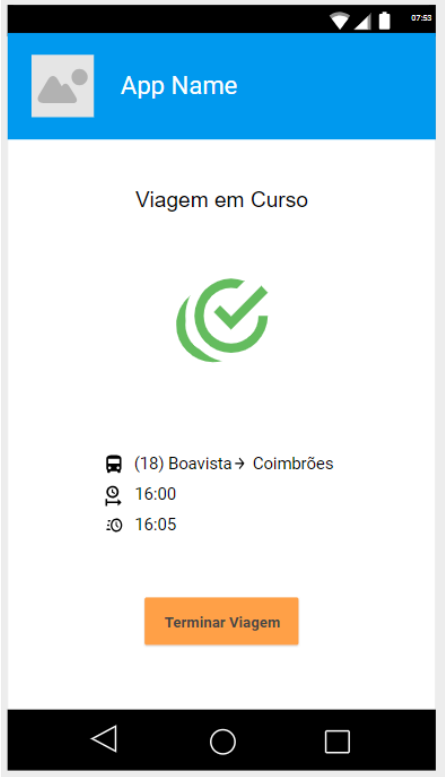

Figure 3.7: Mock-up of the driver application's page for a successfully on-going data gathering

Chapter 4

A new Travel Time Prediction algorithm

4.1 Introduction

As mentioned in 2.4, the Kalman filter is a state-based model commonly used in TTP problems, with an important advantage in its ability to filter noise in the data. That noise in GPS measurements can be a problem, if not handled correctly, by generating inaccurate results. As such, the Kalman filter was chosen as the algorithm to be implemented to predict the future state (distance travelled, velocity and acceleration) of a bus during its trajectory.

In this chapter, we go over how the Kalman filter works, how it was applied to the this problem and how its output is used to produce an ETA for a given bus.

4.2 The Kalman filter algorithm

The Kalman filter is an algorithm that uses measurements, containing noise and other inaccuracies, observed over time, to predict the next state of a system, that is, to estimate the future value for given variables, using a joint probability distribution over the variables for each timespan [Kal60].

The Kalman filter works recursively through 2 stages. In the **prediction** stage, the algorithm calculates an estimate of the current state variables, taking into account its uncertainties. Once the actual value of the estimated variables are observed, with some amount of error in the measurement, the algorithm enters the **update** stage. In this step, the estimates calculated in the prediction step are updated using a weighted average, with estimates with higher certainty having more weight.

4.2.1 Prediction step

In the prediction stage, the algorithm predicts a state estimate a the error covariance as follows:

$$\text{Predicted state estimate: } \hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_k \hat{\mathbf{x}}_{k-1|k-1}$$

$$\text{Predicted error covariance: } \mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^{\mathsf{T}} + \mathbf{Q}_k$$

where

- k is the current time-step
- $\mathbf{F}_k$ is the state transition function
- $\hat{\mathbf{x}}_{k-1|k-1}$ is the last updated state estimate
- $\hat{\mathbf{P}}_{k-1|k-1}$ is the last updated error covariance
- $\mathbf{Q}_k$ is the covariance of the process noise

4.2.2 Update step

When there is a new measurement for the predicted variables, the algorithm updates the state estimate, combining the measured values with the predicted state estimate. The calculations done in the algorithm during this phase are as follows:

$$\begin{gathered}
\text{Measurement pre-fit residual: } \tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k|k-1} \\
\text{Pre-fit residual covariance: } \mathbf{S}_k = \mathbf{R}_k + \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^{\mathsf{T}} \\
\text{Kalman gain: } \mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^{\mathsf{T}} \mathbf{S}_k^{-1} \\
\text{Updated state estimate: } \hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \tilde{\mathbf{y}}_k \\
\text{Updated estimate covariance: } \mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1} (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)^{\mathsf{T}} + \mathbf{K}_k \mathbf{R}_k \mathbf{K}_k^{\mathsf{T}} \\
\text{Measurement post-fit residual: } \tilde{\mathbf{y}}_{k|k} = \mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k|k}
\end{gathered}$$

where

- $\mathbf{z}_k$ is the observation of the true state $\mathbf{x}_k$
- $\mathbf{H}_k$ is the observation model
- $\mathbf{R}_k$ is the covariance of the observation noise
- $\mathbf{I}$ is the identity matrix

4.3 Kalman filter algorithm for Travel Time Prediction

For our TTP problem, the state estimate $\mathbf{x}_{k|k-1}$ represents a matrix with estimates for the distance travelled by a bus on a given line since the start of that line, as well as the velocity and the acceleration of that bus, so that $\mathbf{x}_{k|k-1} = \begin{bmatrix} \mathbf{d}_{k|k-1} & \mathbf{v}_{k|k-1} & \mathbf{a}_{k|k-1} \end{bmatrix}$.

The state transition function $\mathbf{F}_k$ is a function with the equations of motion for a particle moving in a straight line with constant acceleration, so that:

$$\mathbf{F}_k = \begin{bmatrix} 1 & \Delta t & \frac{1}{2}\Delta t^2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix}$$

. The smartphone application gathering GPS information collects and saves the latitude and longitude of a bus during each trip. The distance between 2 consecutive measurements is determined by applying the *Vincenty's formulae*, a formulae used to calculate the distance between two points, based on the assumption that the Earth is an oblate spheroid instead of a sphere, hence being more accurate than other methods [Vin75]. This calculation gives us the direct distance between two GPS measurements, which is why $\mathbf{F}_k$ uses the equations for a particle moving in a straight line. Even though a bus trajectory is not a set of straight lines, the small value of Δt, that is, the time between two consecutive GPS measurements, means that the difference between the straight line distance and the actual distance travelled is minimal.

The reason why we chose to assume a constant acceleration between 2 measurements is that Kalman filters can only be applied to linear systems, and to represent our problem as a linear system we need to represent the acceleration as linear. To work around this more accurately, our algorithm could be converted to an Unscented Kalman Filter, as will be discussed in 6.2.

The observation model $\mathbf{H}_k$ represents which variables of $\mathbf{x}_k$ are actually measured by our system and which variables are calculated from our measurements. Since our system only measures the distance travelled of a bus, $\mathbf{H}_k = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$

$\mathbf{R}_k$ represents the accuracy of the measured values, in this case, GPS positions. Nowadays, smartphones have a GPS accuracy of about 5 meters [vDE15], so $\mathbf{R}_k = \begin{bmatrix} 5 \end{bmatrix}$.

With the values predicted with our Kalman filter, our system can produce an ETA for a given bus and a given stop. To do so, we use the following equation:

$$ETA = \frac{TotalDistance - TraveledDistance}{AvgSpeed}$$

where

- TotalDistance is the length, in meters, of a bus line between its start and the stop for which we are calculating an ETA. This distance is know because the bus travels through a fixed path and those fixed distances should be stored in the database of our system.

- TraveledDistance corresponds to $\mathbf{d}_{k|k-1}$, from our Kalman Filter.

- AvgSpeed is the average speed, calculated by dividing the last measured distance travelled by the duration, in seconds, of the bus trip since it started. The calculation of this variable can be improved by taking into account more factors, as it will be mentioned in 6.2.

4.4 Internal validation of the new Travel Time Prediction algorithm

To test our algorithm before real data could be gathered, we simulated some measurements. To do so, we created an array, $measurements$, with 300 evenly spaced numbers from 0 to 3000, representing 300 total distance measurements over the entire trip, so that $measurements = [0, 10, 20, ..., 3000]$. Then, we multiplied that array by an array of random numbers between 0.2 and 1.9, $RandomValues$, generated by our program when running, to apply some randomness to our measurements and so that the different distance between 2 consecutive measurements could represent different speeds. For instance, multiplying $measurements$ by $RandomValues = [0.3, 1.5, 0.2, ..., 1.2]$ gives us an array with random measurements, so now $measurements = [0, 15, 4, ...3600]$. After that, we sort the values in the array in ascending order, as this array is supposed to represent cumulative sum of distances between consecutive GPS measurements.

For this scenario we defined $\Delta t = 5seconds$, so we also created an array with 300 evenly spaced numbers between between 0 and 1500, representing the total time since the beginning of the trip at each measurement, so that $TimeMeasurements = [0, 5, 10, ..., 1500]$. This means we have a simulated trip with a total duration of 1500 seconds, or 25 minutes. It is important to reference that Δt, the time between each measurement, should be as small as possible, so that the difference between the straight line distance between two GPS measurements and the actual distance travelled by a bus is minimal. However, a smaller value for Δt means that the smartphone taking those measurements will spend more battery, since it will be taking more measurements.

Our first travelled distance measurement is 0, which represents a velocity of 0. Also, the first few measurements provide a sample too small to calculate an average speed accurate enough to produce an ETA. With that in mind, it was decided that the ETA for the first 10 measurements would correspond to the scheduled bus arrival in its timetable. Since we do not have a schedule, as this is a simulation, $ETA = 1500$ for the first 10 iterations of our algorithm, which corresponds to the total duration of this simulated trip.

With this, we can finally test our simulation. We tested our predictions for what would be the last stop of the trip, as it would be the hardest stop to predict, since it is the most distant stop of the bus line. In Figure 4.1 we can see the results of that simulation. The first chart of Figure 4.1 shows us the Kalman filter predictions for the distance travelled, in meters, and our $measurements$ over the time of this trip. On the second chart of Figure 4.1, we have our ETA predictions for what would be the final stop and, since we know that this simulated trip has a duration of 1500 seconds, we know the actual time of arrival at any given time-step. So, at each new prediction, every 5 seconds, the time of arrival decreases 5 seconds, as we can see in the orange line in this graph. In Figure 4.2, we have the results of a similar simulation, but with $RandomValues$ containing values from 0.5 to 1.5, instead of 0.2 to 1.9. We can easily compare our ETA prediction with the actual time to arrival by looking at the difference between the two lines at any given elapsed time. The closer the orange and the blue line in the second graph of Figures 4.1 and 4.2 are, the more accurate is our prediction. The same applies to the Kalman filter predictions and the distance travelled measurements.

4.5 Conclusions

In this chapter, we went over how the Kalman filter works, how we applied it to our problem and incorporated it in our ETA solution.

As we can see in Figures 4.1 and 4.2, our Kalman Filter produces estimates for the distance travelled very close to its real values. However, the ETA predictions are not so close from the actual time of arrival. Shortening the interval of *RandomValues* reduces this gap, as we can see from the difference between Figures 4.1 and 4.2. We can also see, as expected, that the ETA prediction gets closer to reality as the bus gets closer to, in this case, its last stop.

These results are not worrying, as the randomness applied might have generated measurements more disperse and harder to predict than those of a real life scenario. So, with our TTP algorithm implemented, we are ready to test it with real data, collected by a smartphone.

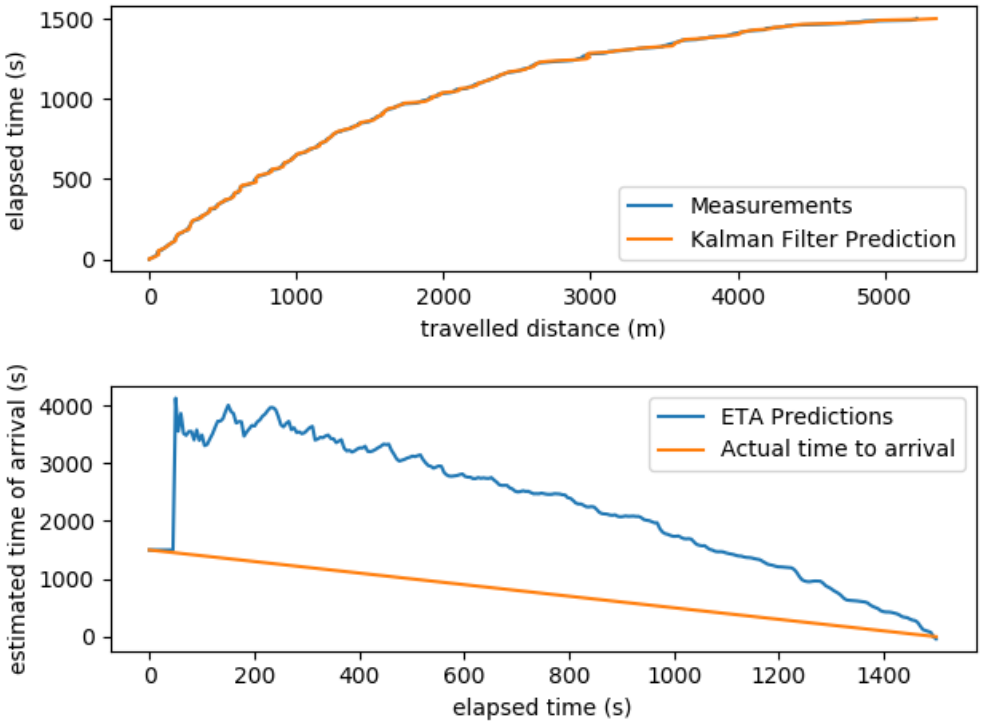

Figure 4.1: Results from simulation with *RandomValues* between 0.2 and 1.9

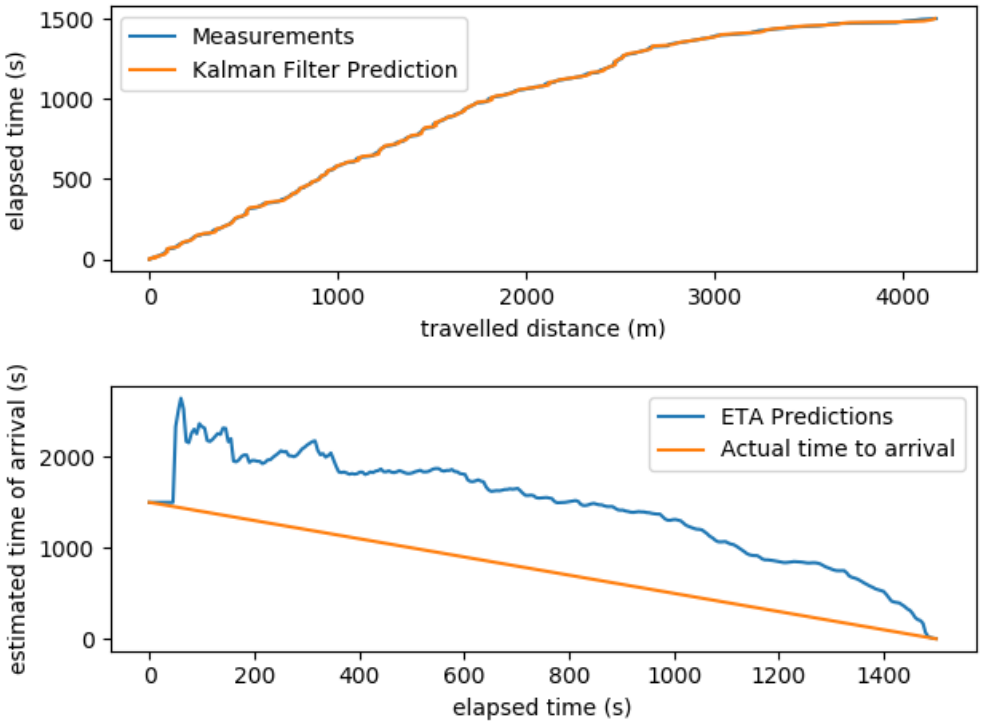

Figure 4.2: Results from simulation with *RandomValues* between 0.5 and 1.5

Chapter 5

Case study

5.1 Introduction

In this chapter, we present the results of the GPS data collection done by our smartphone and the results of the predictions made by our TTP algorithm, as well as how they were analysed.

To analyse the GPS data collected by our smartphone application and to test our TTP algorithm, we selected *Espírito Santo*'s[1] bus line 18. This line goes from *Afurada*, in Vila Nova de Gaia, to *Casa da Música*, in Porto. The reason for this being the selected line for the analysis is that it passes through *Ponte da Arrábida*, one of Portugal's busiest bridges, and the difference in traffic between rush hour and regular hours, as well as the difference in traffic between the two directions of the bridge, gives us a variety of scenarios for testing our solution. We then made several data collections and experiments, between January 9 and January 22, the results of which will be discussed in Section 5.4

5.2 GPS data analysis

In order to analyse the GPS data collected by our smartphone application, a program in Python was developed to plot the gathered GPS data on a map, as well as the bus stops for the selected line. The files needed for this analysis are the following:

- `lines.csv` — Information about *Espírito Santo*'s lines, containing line's codes, names, directions and the codes for the line's stops, as we can see in Table 5.1. This information was provided by *OPT* [2].

- `stops.csv` — Information about *Espírito Santo*'s bus stops, containing codes, matching with the stop codes in `lines.csv`, names and GPS locations, as we can see in Table 5.2. This information was also provided by *OPT*.

[1] http://www.carreiras.espiritosanto.com.pt/
[2] Optimização e Planeamento de Transportes, S.A. (http://www.opt.pt/)

line_code	name	direction	sequence	stop_code
18	Boavista - Afurada (Via Arrábida Shopping)	Ida	24	304
18	Boavista - Afurada (Via Arrábida Shopping)	Ida	25	186
18	Boavista - Afurada (Via Arrábida Shopping)	Ida	26	187
18	Afurada - Boavista (Via Arrábida Shopping)	Volta	1	187
18	Afurada - Boavista (Via Arrábida Shopping)	Volta	2	188
18	Afurada - Boavista (Via Arrábida Shopping)	Volta	3	185
18	Afurada - Boavista (Via Arrábida Shopping)	Volta	4	191
18	Afurada - Boavista (Via Arrábida Shopping)	Volta	5	192
18	Afurada - Boavista (Via Arrábida Shopping)	Volta	6	552
18	Afurada - Boavista (Via Arrábida Shopping)	Volta	7	193

Table 5.1: Excerpt of `lines.csv`

- `stops_afurada_casamusica.csv` — A file with the sequence of stops for the *Espírito Santo*'s line 18, for the direction from *Afurada* to *Casa da Música*, as we can see in Table 5.3. This file was created by combining the information from `lines.csv` and `stops.csv`.
- `stops_casamusica_afurada.csv` — A similar file to `stops_afurada_casamusica.csv`, for the other direction of line 18
- `gathered_data.csv` — GPS data for a single bus trip, gathered and stored by our application. For each GPS measurement, our application adds a row to this file containing a time-stamp for that measurement, in milliseconds, a date and time in text format, converted from that time-stamp, for better visualisation and the latitude and longitude of that measurement. Furthermore, each row also contains the number of stops passed. This value is updated by clicking on a button on our application prototype when passing a bus stop, and is there so we can separate the gathered data into subsets, if we want to apply our TTP algorithm to a stop in the middle of the bus line. We can see an example of the information stored by our app, in Table 5.4.
- `ticket_validation_measurements.csv` — GPS measurements collected from the bus' ticket validation device, for the same trip in `gathered_data.csv`. Since *Espírito Santo*'s buses do not have AVL devices, this is the information currently available in *Espírito Santo*'s transportation network for GPS positioning. Each row corresponds to a measurement and contains an identifier for the bus, an identifier for the last stop visited, a latitude and longitude, the direction of the bus (0 or 1), a timestamp for the last visit to a stop and a timestamp for that measurement, as we can see in the example of Table 5.5. This information is used so we can compare and validate our results with GPS data already in use for public transportation solutions. This information was also provided by *OPT*.

For this analysis, we used **PyPy**[3], an alternative implementation of the Python language with a Just-in-Time compiler and a better memory usage, making it faster. Furthermore, the libraries

[3]https://pypy.org/

stop_code	external code	name	latitude	longitude
158	158	Tv Barrosa Vi33	41.12495	-8.612287
159	159	Conservatório Vi33	41.12558	-8.614953
16	16	Sta Barbara I	41.1246	-8.628562
160	160	AFURADA	41.14179	-8.648458
161	161	R. da Praia I	41.14125	-8.64968
162	162	Tirone I	41.13757	-8.651016
163	163	MARQUES GOMES I	41.13445	-8.652231
164	164	Fontão I	41.13461	-8.648949
165	165	Pedra Alta I	41.13517	-8.64704
166	166	S. Paio I	41.13538	-8.645117

Table 5.2: Excerpt of `stops.csv`

sequence	code	name	latitude	longitude
1	160	AFURADA	41.14179	-8.648458
2	161	R. da Praia I	41.14125	-8.64968
3	162	Tirone I	41.13757	-8.65102
4	163	MARQUES GOMES I	41.13445	-8.65223
5	164	Fontão I	41.13461	-8.64895
6	165	Pedra Alta I	41.13517	-8.64704
7	166	S. Paio I	41.13538	-8.64512
8	167	Tripeira I	41.13437	-8.64515
9	168	R. de Bustes I	41.13085	-8.64468
10	133	Espinheiro I	41.12902	-8.64315
11	134	QUATRO CAMINHOS (Farmacia)	41.12819	-8.6394
12	65	Tenente Valadim I	41.12961	-8.63712
13	66	Fonte Lodosa I	41.13148	-8.63689
14	177	Dr Ribeiro de Magalhães I	41.13239	-8.63581
15	178	Jordão I	41.13546	-8.63584
16	179	AVE DOS ESCULTORES I	41.13853	-8.63729
17	270	Arrabida Shoping I	41.1391	-8.63546
18	180	Arrabida Shoping I	41.13921	-8.63557
19	181	Jardins da Arrábida I	41.13899	-8.63886
20	182	Afurada de Cima I	41.14059	-8.6419
21	183	Bairro Cavaco I	41.14209	-8.64139
22	184	Convívio I	41.15405	-8.63174
23	185	Bom Sucesso Iv	41.15511	-8.62861
24	304	Bom Sucesso Interface	41.15582	-8.628417
25	186	Ave França I	41.1593	-8.62876
26	187	BOAVISTA/CASA MÚSICA	41.16132	-8.63007

Table 5.3: Contents of `stops_afurada_casamusica.csv`

date	timestamp	latitude	longitude	stops passed
Wed Jan 09 17:01:32 GMT+00:00 2019	1547053292177	41.1264241	-8.6056105	0
Wed Jan 09 17:01:42 GMT+00:00 2019	1547053302057	41.1263859	-8.6056378	0
Wed Jan 09 17:01:52 GMT+00:00 2019	1547053312247	41.1263766	-8.6057144	0
Wed Jan 09 17:02:02 GMT+00:00 2019	1547053322250	41.1263665	-8.6057895	0
Wed Jan 09 17:02:12 GMT+00:00 2019	1547053332471	41.1263612	-8.6057841	0
Wed Jan 09 17:02:26 GMT+00:00 2019	1547053346641	41.1260607	-8.6067046	0
Wed Jan 09 17:02:36 GMT+00:00 2019	1547053356701	41.126054	-8.6068675	1
Wed Jan 09 17:02:44 GMT+00:00 2019	1547053364000	41.1260691	-8.6066261	1
Wed Jan 09 17:02:54 GMT+00:00 2019	1547053374000	41.1260869	-8.6062888	1
Wed Jan 09 17:03:01 GMT+00:00 2019	1547053381000	41.1261439	-8.6063869	1

Table 5.4: Excerpt from the data collected by the driver smartphone application during a bus trip

bus id	last stop id	latitude	longitude	direction	last stop time	measurement time
215	177	41.131932	-8.63618	0	22-01-2019 17:43	22-01-2019 17:43
215	178	41.134732	-8.635805	0	22-01-2019 17:43	22-01-2019 17:44
215	179	41.13755	-8.636665	0	22-01-2019 17:44	22-01-2019 17:44
215	180	41.138864	-8.636346	0	22-01-2019 17:44	22-01-2019 17:45
215	270	41.139656	-8.634231	0	22-01-2019 17:45	22-01-2019 17:45
215	180	41.139972	-8.634325	0	22-01-2019 17:45	22-01-2019 17:45
215	180	41.1389	-8.636383	0	22-01-2019 17:46	22-01-2019 17:47
215	181	41.138864	-8.638286	0	22-01-2019 17:46	22-01-2019 17:47
215	182	41.139652	-8.641366	0	22-01-2019 17:47	22-01-2019 17:48
215	182	41.14012	-8.641788	0	22-01-2019 17:47	22-01-2019 17:48
215	183	41.14143	-8.641831	0	22-01-2019 17:47	22-01-2019 17:48
215	183	41.1434	-8.64173	0	22-01-2019 17:49	22-01-2019 17:50

Table 5.5: Excerpt from the GPS measurements collected by ticket validation devices

pandas[4] and **folium**[5] were used. Pandas is an open-source library that provides high-performance, easy-to-use data structures and data analysis tools. This library was used to transform the necessary files into dataframes, a tabular data structure with labelled axes, and to perform the necessary tasks in that data. Folium is a library that allows us to visualise data that's been manipulated in Python on an interactive leaflet map.

The GPS data collected by our app was imported and saved as a *pandas* dataframe named `gps_df`. After, that, a function named `plotMeasurements()` is called to go through all the rows of `gps_df` and plot all GPS measurements on a map as black circles, as well as to plot a red line between all measurements, in order to better visualise the path of the bus.

We also import `stops_afurada_casamusica.csv` and `stops_casamusica_afurada.csv` as dataframes, and plot one of them on our map, accordingly to the direction of the trip, by calling a function named `plotStops(stops_path)`. The GPS coordinates of all stops are plotted as blue markers on our map, in order to give us reference points to better visualise if the path plotted from our GPS measurements passes through these reference points.

The created map is saved in an `.html` file, giving us an interactive visualisation of our information in a map we can zoom in on and drag. In Figure 5.1 we have a screenshot of the interactive map for the results of a bus trip from *Afurada* to *Casa da Música*, on January 9.

In order to compare our results with GPS measurements from ticket validation devices, we call a function named `plotTicketData(ticket_data_path)`. This function imports `ticket_validation_measurements.csv` and saves it as a dataframe named `ticket_measurements_df`. Then, the GPS measurements in `ticket_measurements_df` are plotted in our interactive map, as purple circles, as well as a green line between all consecutive measurements. To obtain the `plotTicketData(ticket_data_path)` for a given trip, we went through all the ticket validation devices' measurements for *Espírito Santo*'s line 18, which were provided by *OPT*, and filtered the data by the date and time of the first and last measurement in `gathered_data.csv`, as well as the bus identifier, which can be seen in the front of each bus and was annotated before entering a bus to conduct experiments.

The result of all the plotted data (application's GPS measurements, bus stops and ticket validation device's GPS measurements) can be seen in Figures 5.2 and 5.3, and in Appendix A.

[4] https://pandas.pydata.org/

[5] https://python-visualization.github.io/folium/

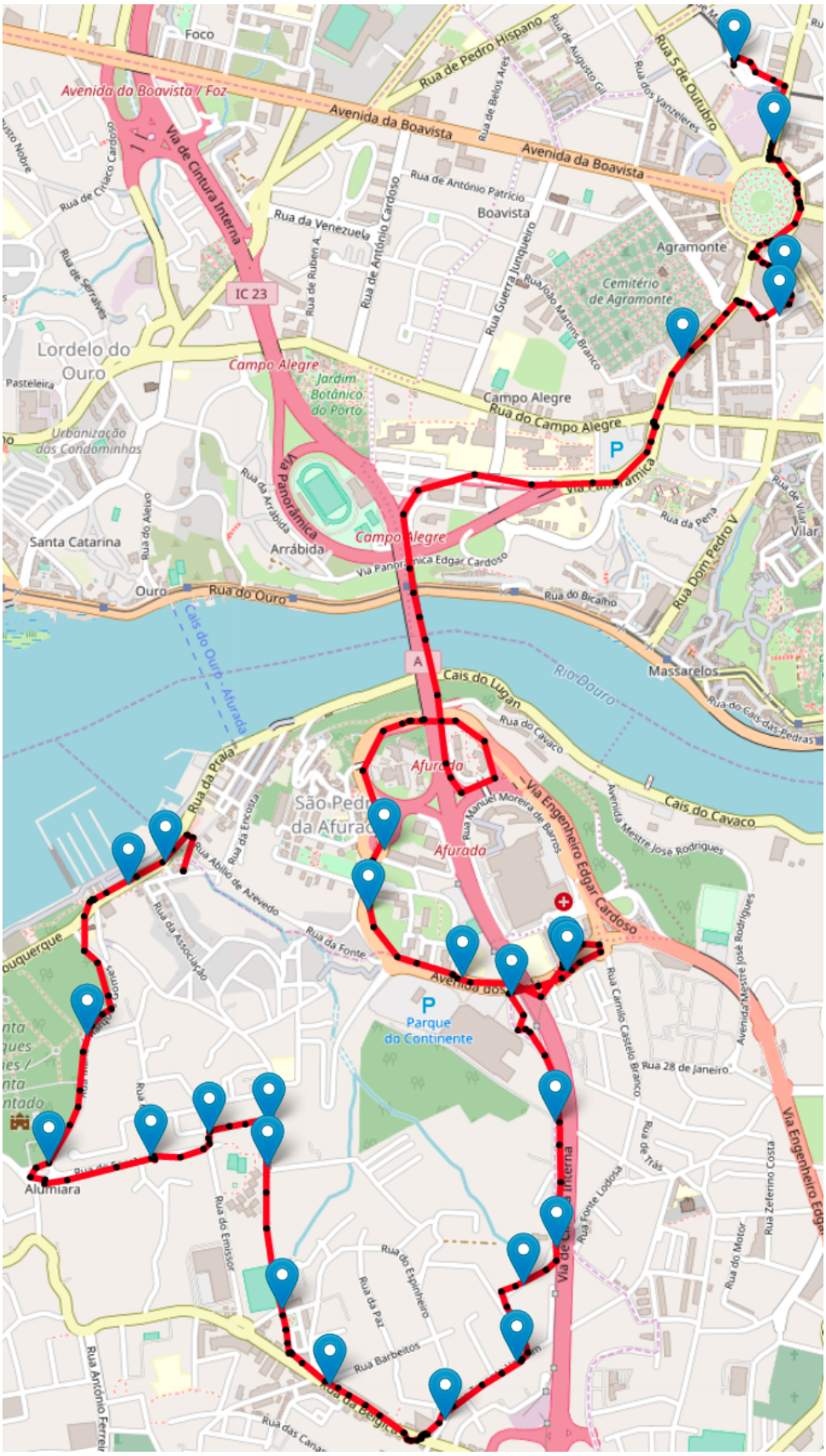

Figure 5.1: GPS data collected during a bus trip from *Afurada* to *Casa da Música*, on January 09, between 18:11 and 18:41

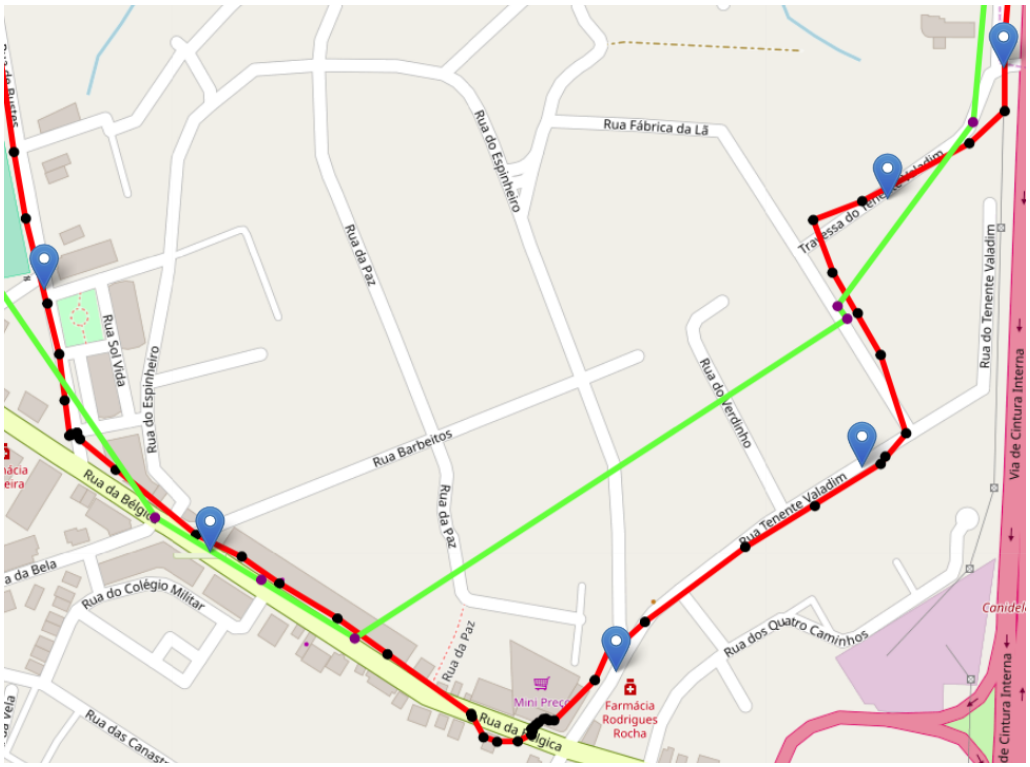

Figure 5.2: Zoomed in results of the same trip seen in 5.1, now with the ticket validation device's GPS measurements

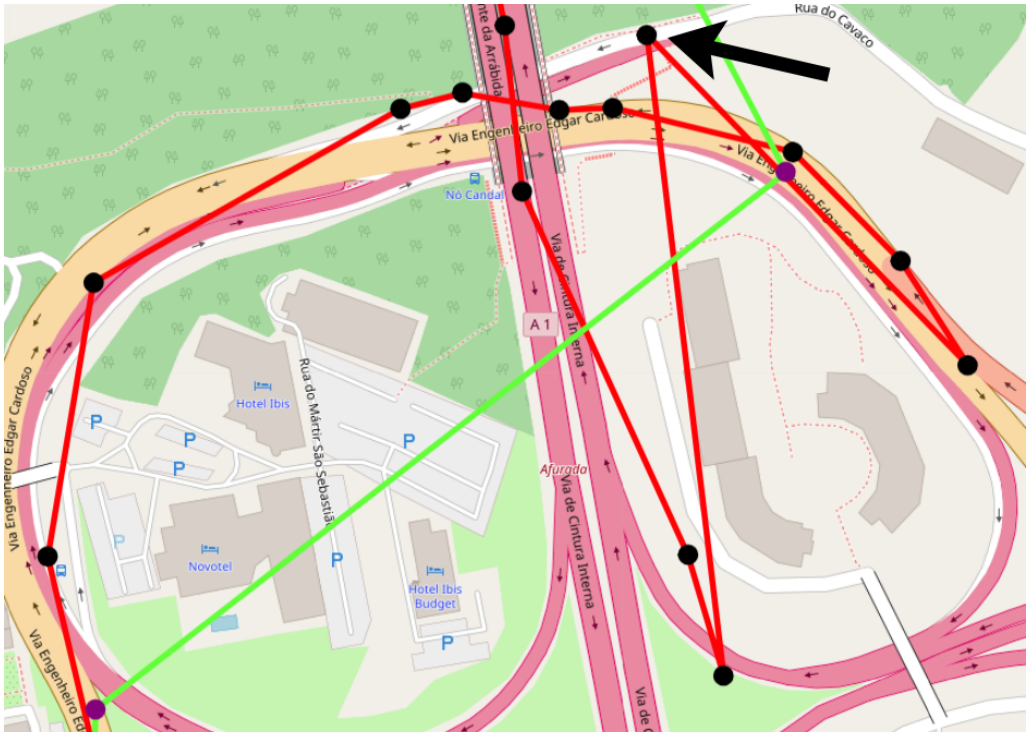

Figure 5.3: Example of GPS data collection results with one highly inaccurate measurement

5.3 Application of the TTP algorithm to real data

As previously mentioned, we conducted a series of experiments and data collections for *Espírito Santo*'s line 18, between January 9 and January 22. The results of the data collection for an entire bus trip are saved in a `.csv` file, as we can see in the example presented in Table 5.4.

To implement our TTP algorithm and visualise our predictions, we developed a program in Python. In this program, calling `run_alg(file_name)`, replacing `file_name` by the name of the `.csv` file for which we want predictions to be made, gives us an image with the prediction results as output, as we can see in Figures 5.5 and 5.6.

When calling `run_alg(file_name)`, this function converts our file to a dataframe named `gathered_data`. Then, `gathered_data.getDistances()` adds the columns `last lat` and `last long` to our dataframe and uses the *Vincenty's formulae* to calculate all the distances, in meters, between consecutive measurements, as we can see in Figure 5.4. Then, this function adds the column `dist since last update` to the dataframe, as wee can see in the example presented in Table 5.7. Since our GPS measurements may have some inaccuracy, if the *Vincenty's formulae* determines there is a distance inferior to 5 meters between two consecutive measurements, that distance is discarded. The reason for that is that a smartphone's GPS has an accuracy of about 5 meters, as we have mentioned in Section 4.3, and, therefore, two consecutive GPS measurements with a distance inferior to 5 meters indicate that either the bus position either didn't change or, should those measurements be accurate, the bus position didn't change enough to considerably affect our predictions.

After that, `getTotalDistance()` calculates the cumulative sum of the previous distances and adds the column `total dist` to the `gathered_data` dataframe, so that, at any given row, representing a GPS measurement, we have the distance travelled until that measurement.

Similarly, `gathered_data.getTimeDifferences()` gives us the time differences, in seconds, between consecutive measurements, as we can see in Figure 5.4. Then, this function adds the column `sec since last update` to our dataframe. `gathered_data.getTotalTime()` then gives the cumulative sum of time passed, since the beginning of the trip, at any given measurement and adds the column `total time` to our dataframe, as wee can see in Table 5.7. Even though in our application, developed for Android[6] devices, we defined the time interval between GPS measurements to be equal to 5 seconds, Google's Fused Location Provider API[7], the API we used for GPS updates, takes more time to give a GPS update, in order to save battery, if it determines that the smartphone is moving too slowly for a GPS update to be relevant. This means that sometimes our application takes about the double of the desired time to give an update. For this reason, we have different time intervals between measurements that we need to pass as Δt to our algorithm, hence the need for knowing the elapsed time between consecutive updates.

With this, we finally have a dataframe with all the necessary information to apply our TTP algorithm. Since our TTP algorithm was applied to the gathered data *a posteriori*, we know the

[6] https://www.android.com/

[7] https://developers.google.com/location-context/fused-location-provider/

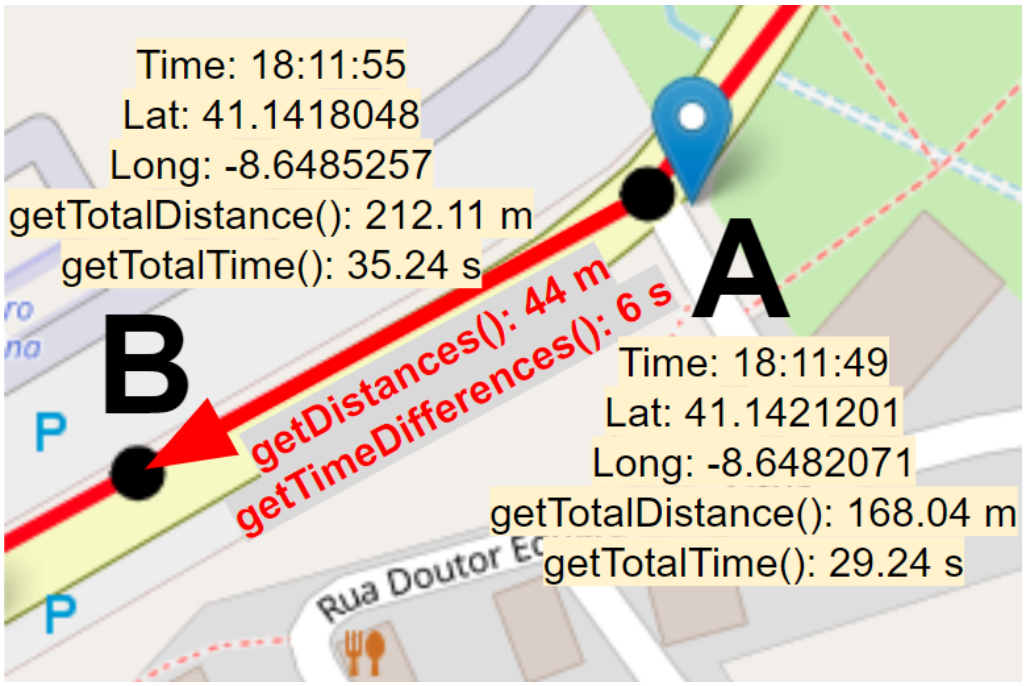

Figure 5.4: Example of the data from 2 GPS measurements and the values calculated by the functions applied to those measurements

duration of our trip and, therefore, the actual time to arrival at any given measurement, which allows us to evaluate and visualise our results in a similar way as we have in our simulated trip in Section 4.4.

In the first graph of Figures 5.5 and 5.6 we have the Kalman filter prediction for the distance travelled, in orange, and the real measurements for the distance travelled, from the `total dist` column of our `gathered_data` dataframe, in blue. In the second graph, we have our ETA predictions, in this case, for the last stop of the line, from the beginning until the end of the trip, in blue, as well as the actual time of arrival, in orange. In Table 5.6, we have an excerpt of the measurements and predictions plotted in Figure 5.5.

To better analyse our results, we also calculated the relative error between our Kalman filter predictions and distance travelled measurements, as well as the relative error between ETA predictions and the real times to arrival. The relative error is calculated for all predictions by $\frac{|measurement - prediction|}{measurement}$ and an example of these results can be seen in Figure 5.7.

More results of the application of our TTP algorithm to real data can be seen in Appendix B.

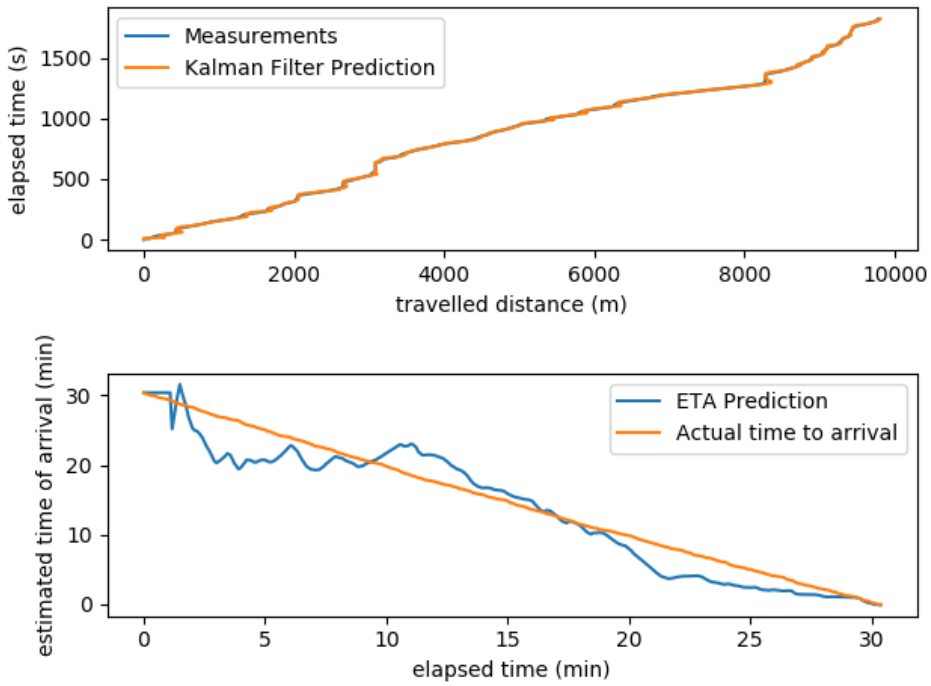

Figure 5.5: Predictions for the bus trip from *Afurada* to *Casa da Música* seen in Figure 5.1

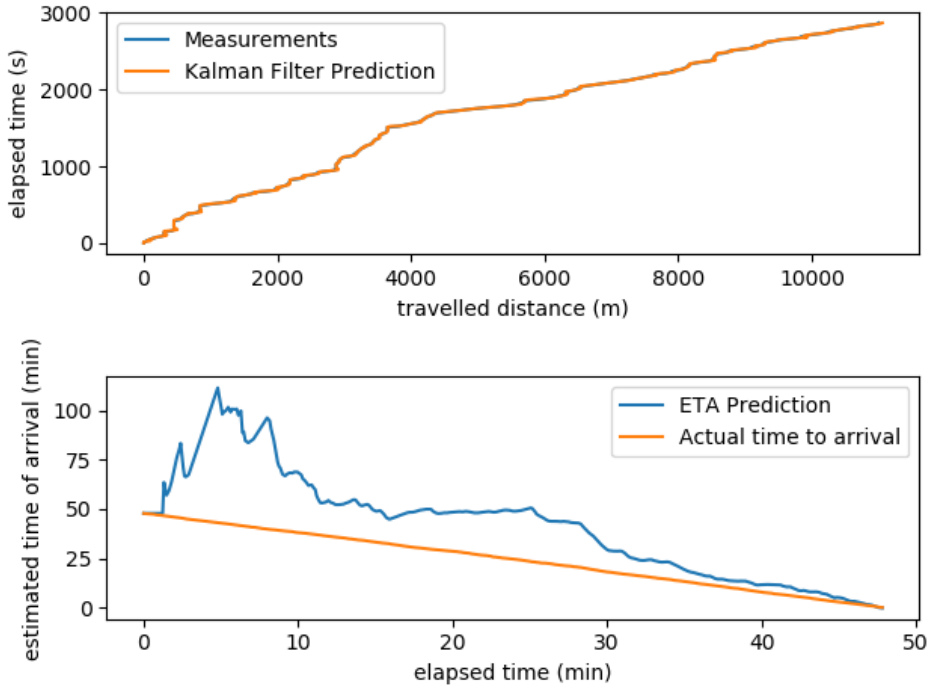

Figure 5.6: Predictions for a trip from *Casa da Música* to *Afurada*

distance travelled (m)	distance travelled prediction	actual time to arrival (s)	eta prediction
9393,079	9405,638	130,239	67,49
9404,227	9425,146	125,239	64,301
9404,227	9418,533	120,239	65,684
9410,471	9415,349	110,239	66,41
9418,483	9420,702	105,239	65,77
9418,483	9422,413	94,239	65,65
9424,993	9424,625	89,239	65,581
9436,621	9427,259	80,239	65,207
9444,773	9442,493	75,239	62,716
9444,773	9454,596	70,239	60,831
9461,223	9449,67	65,239	61,813
9488,82	9462,342	60,239	59,455
9568,614	9504,872	50,239	51,417
9600,608	9585,974	41,239	36,377
9635,209	9629,855	35,239	28,232
9688,493	9705,524	29,239	14,236
9718,973	9723,606	19,239	10,887
9756,696	9781,295	10,766	0,22
9767,233	9785,885	5,017	0
9782,482	9811,802	0	0

Table 5.6: Excerpt of the predictions plotted in Figure 5.5

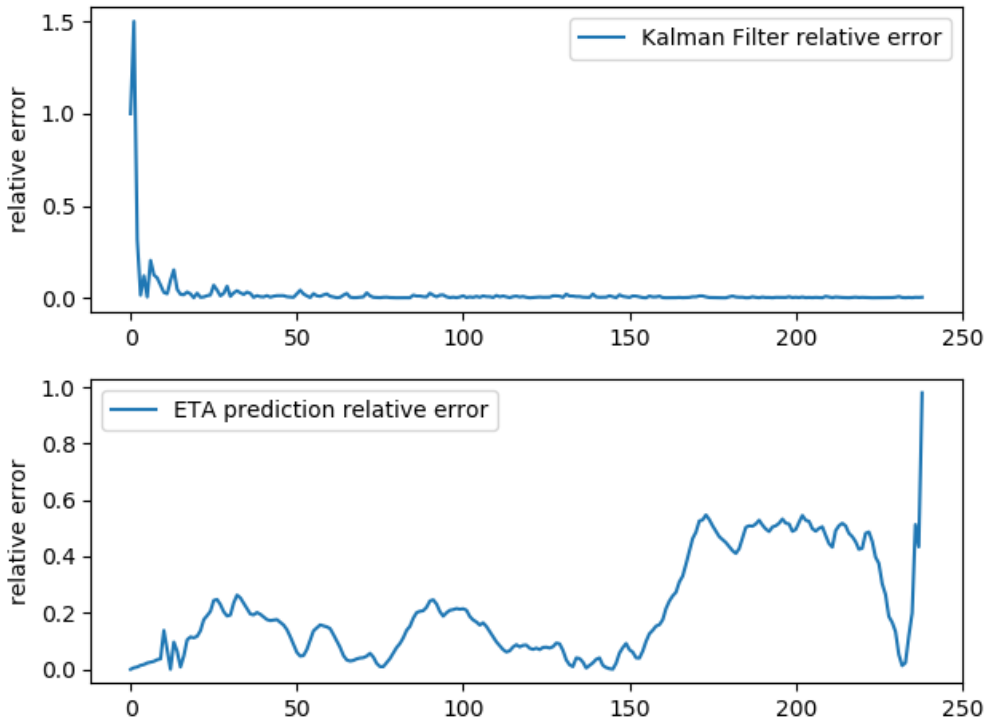

Figure 5.7: Relative error of the predictions in Figure 5.5

date	timestamp	latitude	longitude	n stops	last lat	last long	dist since last update	total dist	sec since last update	total time
Wed Jan 09 18:11:19 GMT+00:00 2019	1547057479761	41,1415917	-8,6478553	0			0	0	0	0
Wed Jan 09 18:11:24 GMT+00:00 2019	1547057484778	41,1415917	-8,6478553	0	41,1415917	-8,6478553	0	0	5,017	5,017
Wed Jan 09 18:11:30 GMT+00:00 2019	1547057490527	41,1424082	-8,6475471	0	41,1415917	-8,6478553	94,2971766933996	94,2971766933996	5,749	10,766
Wed Jan 09 18:11:39 GMT+00:00 2019	1547057499000	41,1424784	-8,6477342	0	41,1424082	-8,6475471	17,5360751557091	111,833251849109	8,473	19,239
Wed Jan 09 18:11:49 GMT+00:00 2019	1547057509000	41,1421201	-8,6482071	0	41,1424784	-8,6477342	56,2103833083725	168,043635157481	10	29,239
Wed Jan 09 18:11:55 GMT+00:00 2019	1547057515000	41,1418048	-8,6485257	0	41,1421201	-8,6482071	44,063367648703	212,107002806184	6	35,239
Wed Jan 09 18:12:01 GMT+00:00 2019	1547057521000	41,1414579	-8,6493673	0	41,1418048	-8,6485257	80,4769679465242	292,583970752708	6	41,239
Wed Jan 09 18:12:10 GMT+00:00 2019	1547057530000	41,1410632	-8,650344	0	41,1414579	-8,6493673	92,9800168436769	385,563987596385	9	50,239
Wed Jan 09 18:12:20 GMT+00:00 2019	1547057540000	41,1409048	-8,6507956	0	41,1410632	-8,650344	41,7966023297423	427,360589926128	10	60,239
Wed Jan 09 18:12:25 GMT+00:00 2019	1547057545000	41,1409027	-8,6507348	0	41,1409048	-8,6507956	5,10981635138503	432,470406277513	5	65,239
Wed Jan 09 18:12:30 GMT+00:00 2019	1547057550000	41,1408928	-8,6507664	0	41,1409027	-8,6507348	0	432,470406277513	5	70,239
Wed Jan 09 18:12:35 GMT+00:00 2019	1547057555000	41,140899	-8,6507919	1	41,1408928	-8,6507664	0	432,470406277513	5	75,239
Wed Jan 09 18:12:40 GMT+00:00 2019	1547057560000	41,1408848	-8,6508082	1	41,140899	-8,6507919	0	432,470406277513	5	80,239
Wed Jan 09 18:12:49 GMT+00:00 2019	1547057569000	41,1408373	-8,6508824	1	41,1408848	-8,6508082	8,16298231679142	440,633388594304	9	89,239
Wed Jan 09 18:12:54 GMT+00:00 2019	1547057574000	41,1404899	-8,6510671	1	41,1408373	-8,6508824	41,5807216871632	482,214110281467	5	94,239
Wed Jan 09 18:13:05 GMT+00:00 2019	1547057585000	41,1396454	-8,6510618	1	41,1404899	-8,6510671	93,7883680055306	576,002478286998	11	105,239
Wed Jan 09 18:13:10 GMT+00:00 2019	1547057590000	41,1393504	-8,6506348	1	41,1396454	-8,6510618	48,5648253283153	624,567303615313	5	110,239
Wed Jan 09 18:13:20 GMT+00:00 2019	1547057600000	41,1385538	-8,6502748	1	41,1393504	-8,6506348	93,4883931638425	718,055696779156	10	120,239
Wed Jan 09 18:13:25 GMT+00:00 2019	1547057605000	41,1382405	-8,6501856	1	41,1385538	-8,6502748	35,5908858394617	753,646582618617	5	125,239
Wed Jan 09 18:13:30 GMT+00:00 2019	1547057610000	41,1380964	-8,6504683	1	41,1382405	-8,6501856	28,626343180779	782,272925799396	5	130,239
Wed Jan 09 18:13:36 GMT+00:00 2019	1547057616000	41,1378209	-8,6508189	1	41,1380964	-8,6504683	42,45716228098	824,730088080376	6	136,239
Wed Jan 09 18:13:45 GMT+00:00 2019	1547057625000	41,1371851	-8,6510988	1	41,1378209	-8,6508189	74,4178010313082	899,147889111685	9	145,239
Wed Jan 09 18:13:55 GMT+00:00 2019	1547057635000	41,136246	-8,651192	1	41,1371851	-8,6510988	104,586356784504	1003,73424589619	10	155,239
Wed Jan 09 18:14:00 GMT+00:00 2019	1547057640000	41,1358395	-8,6512606	1	41,136246	-8,651192	45,51043037954	1049,24467627573	5	160,239
Wed Jan 09 18:14:05 GMT+00:00 2019	1547057645000	41,1354011	-8,6515326	1	41,1358395	-8,6512606	53,7773816766639	1103,02205795239	5	165,239
Wed Jan 09 18:14:15 GMT+00:00 2019	1547057655000	41,1344683	-8,6522027	1	41,1354011	-8,6515326	117,886547870727	1220,90860582312	10	175,239
Wed Jan 09 18:14:20 GMT+00:00 2019	1547057660000	41,1343595	-8,6526193	1	41,1344683	-8,6522027	37,0075275492105	1257,91613337233	5	180,239
Wed Jan 09 18:14:30 GMT+00:00 2019	1547057670000	41,1341301	-8,6527987	1	41,1343595	-8,6526193	29,5963609136547	1287,51249428598	10	190,239
Wed Jan 09 18:14:42 GMT+00:00 2019	1547057682000	41,1340688	-8,6523278	2	41,1341301	-8,6527987	40,1206473280803	1327,63314161406	12	202,239
Wed Jan 09 18:14:46 GMT+00:00 2019	1547057686179	41,1339667	-8,6523525	2	41,1340688	-8,6523278	11,5269714211048	1339,16011303517	4,179	206,418
Wed Jan 09 18:14:54 GMT+00:00 2019	1547057694000	41,1340553	-8,6516119	2	41,1339667	-8,6523525	62,9578052095436	1402,11791824471	7,821	214,239
Wed Jan 09 18:15:04 GMT+00:00 2019	1547057704000	41,134376	-8,6500188	2	41,1340553	-8,6516119	138,423842863902	1540,54176110862	10	224,239
Wed Jan 09 18:15:14 GMT+00:00 2019	1547057714000	41,1345944	-8,6488875	2	41,134376	-8,6500188	98,0361637953865	1638,577924904	10	234,239
Wed Jan 09 18:15:19 GMT+00:00 2019	1547057719000	41,1345648	-8,6487758	3	41,1345944	-8,6488875	9,93814045274979	1648,51606535675	5	239,239

Table 5.7: Excerpt from a dataframe with the GPS measurements collected by the driver application and the information calculated *a posteriori*

5.4 Discussion of results

As we can see in Figures 5.1, 5.2 and 5.3, the GPS data collected by our application allow us to draw a clear path for the bus in question. While those measurements may not always be as accurate as those from ticket validation devices, our application collects GPS measurements more frequently, which provides us with a path which is more articulated and closer to the real path of the bus and, therefore, gives us a more accurate measure for the distance travelled since the beginning of the trip until a given GPS measurement, which is essential for our ETA prediction.

There are, rarely, some highly inaccurate GPS measurements, as we can see in Figure 5.3. However, these inaccurate measurements could be filtered, for instance, by ignoring GPS updates which are unfeasible for the time passed since the last GPS update. Namely, if a new GPS measurement would imply that the bus is travelling at a speed highly above the legal speed, or even above a feasible speed, that GPS measurement could be discarded.

In addition to the path plotted by our GPS measurements, it is also possible to infer the speed of the bus by analysing the distance between the black dots representing GPS measurements, in our interactive map, since the time difference between those measurements does not vary much. Generating this interactive map in real time would mean that such information could be used by public transportation companies for fleet management. For instance, if a bus has been moving very slowly through a street where it isn't normal to be much traffic, a person responsible for fleet management could deduct there is an accident or roadworks in that street, and notify the drivers of the next buses to take a small detour.

The results from the Kalman filter algorithm are very clear: our Kalman filter is able to predict, with great accuracy, the distance travelled by the bus in question for the next GPS measurement. These results are very optimistic, and lead us to the conclusion that this part of our TTP algorithm is very solid.

In regards to the ETA predictions, these were more accurate than those of the simulated trips seen in Figures 4.1 and 4.2, specially towards the end of the trips, as we can see in Figures 5.5 and 5.6. This means, perhaps, that the simulated measurements were more scattered and unpredictable than real data.

In the second graph of Figure 5.7 we can observe that the relative error for this ETA prediction increases towards the end of the trip. This might seem odd, as we can see in Figure 5.5 that the ETA prediction is very close to the real time to arrival. This is easily explained by the fact that, when there is only a few seconds for the arrival of a bus, a prediction which is wrong by only a few seconds has a larger impact on the relative error, which is not troublesome in real life.

In Figure 5.6 we can see that in the first 10 minutes our ETA prediction was highly inaccurate. Now, to understand this, it is important to mention that this bus trip took place at rush hour. This trip started at *Casa da Música*, in Porto, and there was high, slow moving traffic throughout the entire bus line up until the bus entered *Ponte da Arrábida*. After that, there was less traffic and the bus moved at a faster speed. This means that, during that initial portion of the trip, the average speed of the bus was much slower than the average speed of the entire trip. As a result, our TTP

algorithm predicted that the bus was going to take much longer to complete the trip than it actually took, given the average speed at that time. This leads to the conclusion that using the average speed of the bus since the beginning of the trip, up until the time of our prediction, to produce an ETA prediction is insufficient and a naive method. In order to work around this, our algorithm could use, for instance, a weighted average between the current average speed and an history of average speeds for similar trips, that is, trips from the same line that took place at the same day of the week and the same time of day of the bus trip in question. To implement that, it would simply require that our system stored the average speed of the bus trips for which there was a smartphone collecting data with our application, and that our system collected data during a few weeks.

5.5 Conclusions

In this chapter, we presented the results of the tests to our GPS data collection method and the predictions made by our TTP algorithm. We made these tests with real data, using a bus line from a public transportation network and we compared our results with GPS data from ticket validation devices.

In regards to our GPS measurements, we conclude that, in addition to using them in our TTP algorithm, they can also have interesting applications for public transportation fleet management. Although there are, occasionally, some more inaccurate GPS measurements, these could be filtered as suggested in Section 5.4, so our system can be more resistant to errors.

Our results also demonstrate that our Kalman filter provides results with great accuracy. On the other hand, there is more room for improvement in our ETA predictions. With such adjustments, both to the GPS measurements and the ETA predictions, we believe a robust solution can be achieved.

Chapter 6

Conclusions and future work

6.1 Main achievements

Over the course of this dissertation, several aspects of the implementation of a system capable of producing bus travel time estimations using smartphones were studied and analysed, with the objective of developing a solution able to replace AVL devices, allowing smaller transportation companies to provide real time information to their passengers, thus bringing this technology to more people.

We reviewed several methods and algorithms for travel time prediction, from which we concluded that the Kalman filter is one of the best and most robust algorithms, an algorithm highly capable of filtering noise in the data. We then implemented this algorithm, adapted it to our TTP problem and integrated it into our ETA prediction algorithm.

During this dissertation, we also designed an architecture of all the components needed for a solution to our problem, with a database, a server, a backoffice platform, a passenger smartphone application and a driver smartphone application. We also listed all the system requirements for the users of this system: the passengers, the drivers and the administrators. With this, we achieved a solid foundation for the solution's implementation.

With the implementation of a prototype of the driver smartphone application, we were able to gather real data from a bus line of a public transportation network over the course of several trips and compare the GPS measurements gathered by our application with data from ticket validation devices. The data gathered by our application, while not always as accurate as the data from ticket validation devices, still allows us to get information about the position and speed of a bus, as well as the path and duration of the trip of that bus. Furthermore, the paths traced by our GPS measurements were more articulated and closer to the real path of the studied buses than the paths traced by GPS measurements gathered by ticket validation devices.

With the data gathered by the developed smartphone application, we were able to test our TTP algorithm. With these tests, we were able to observe that the implemented Kalman filter is able to produce estimates about the distance travelled by a bus over the course of a line, with remarkable

accuracy. With the values from those predictions, we were then able to calculate estimated time of arrivals for the studied buses.

With some additional fine tuning, namely to our GPS measurements, where more inaccurate measurements can be filtered, and to our ETA predictions, which can be improved to include more information in order to produce better estimates, we believe that we can achieve a robust solution to our problem. With those adjustments, and with the implementation of the full system presented in Chapter 3, we believe that it is possible to professionally use smartphones for travel time predictions, at a fraction of the price of the AVL devices currently used by public transportation companies in large cities. The results seen in this dissertation were obtained with a Xiaomi Mi A1, a smartphone which currently can be bought at around 160 Euros [Kua]. Therefore, smartphones, which cost a small amount when compared to the thousands of Euros an AVL device costs, are able to give us solutions for travel time estimations accessible to smaller public transportation companies, having the potential to bring this technology to a higher number of people.

6.2 Future work

In order to improve the work done over the course of this dissertation and to achieve an enhanced solution to the problem presented, some changes could be applied.

The implemented Kalman Filter, for once, can be improved. This algorithm was implemented with equations of motions for a particle moving with constant acceleration, the reason for that being that the Kalman Filter is an algorithm that can only be applied to linear systems. However, we know that the acceleration of a bus is not a constant value throughout an entire trip, which means the equations of motion for a bus moving over a given bus line do not represent a linear system. Therefore, in order to more accurately formulate this system, the implemented Kalman filter should be converted to an Unscented Kalman Filter [WVDM00], where the state transition and observation models do not need to be linear functions. Even though the predictions for the distance travelled by a bus made by our Kalman Filter presented us with positive results, implementing an Unscented Kalman filter could bring even better results, possibly allowing us to make predictions about a more distant future, with even more accuracy.

In regards to the GPS measurements collected by our smartphone application, although they gave us good results, there is also room for improvement. As seen in the results of those measurements, there are some more inaccurate GPS measurements, occasionally. Highly inaccurate measurements can be filtered, for instance, by discarding unfeasible GPS updates that would imply that a bus is travelling at a speed highly above the legal limit, or even a feasible speed.

In our ETA predictions, we use the average speed of the bus from the start of the trip to the moment when a prediction is being made, in order to produce an estimated time of arrival. As seen in our results, this is not always the best measurement to use in our predictions, since a bus can, for example, have a low speed during the first half of a trip, because of traffic, and an higher speed for the rest of the trip, as was the case in some of our experiments. This means that, during the first half of such a trip, our algorithm will estimate that the bus will arrive at the target stop

much later than it actually will, given the low speed at that time. In order to overcome this, our ETA predictions can be adapted to use the average speeds of trips from the same day of the week and time of day, from a given bus line. To do that, we only need to allow our system to gather and save GPS data for a few weeks, before being able to produce improved estimates.

In order to achieve a complete solution, fit for professional use, it would also be necessary to implement other components from the system architecture presented in Chapter 3. Namely, the passenger smartphone application, which would allow public transportation passengers to access estimated times of arrivals in real time, the backoffice platform, which would allow administrators to manage the system, the database, so the necessary information can be stored and accessed at all times, and the server, so all the components of the system can retrieve and store information.

References

[BGME11] James Biagioni, Tomas Gerlich, Timothy Merrifield, and Jakob Eriksson. EasyTracker : Automatic Transit Tracking , Mapping , and Arrival Time Prediction Using Smartphones. *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems SenSys 11*, pages 68–81, 2011.

[BNAoSM08] Transportation Research Board, Engineering National Academies of Sciences, and Medicine. *AVL Systems for Bus Transit: Update*. The National Academies Press, Washington, DC, 2008.

[CC08] Jonathan D Cryer and Kung-Sik Chan. *Time Series Analysis: With Applications in R*. Number January. 2008.

[CD03] F.W. Cathey and D.J. Dailey. A prescription for transit arrival/departure prediction using automatic vehicle location data. *Transportation Research Part C: Emerging Technologies*, 11(3):241 – 264, 2003. Traffic Detection and Estimation.

[CLXC04] Mei Chen, Xiaobo Liu, Jingxin Xia, and Steven I. Chien. A dynamic bus-arrival time prediction model based on APC data. *Computer-Aided Civil and Infrastructure Engineering*, 19(5):364–376, 2004.

[DHNM99] Maged Dessouky, Randolph Hall, Ali Nowroozi, and Karen Mourikas. Bus dispatching at timed transfer transit stations using bus tracking technology. *Transportation Research Part C: Emerging Technologies*, 7(4):187 – 208, 1999.

[DZZ13] Jian Dong, Lu Zou, and Yan Zhang. Mixed Model For Prediction OfBus Arrival Times. *2013 IEEE Congress on Evolutionary Computation*, pages 2918–2923, 2013.

[eMa16] eMarketer. Smartphone users and penetration worldwide, 2014-2020 (billions, % of mobile phone users and % change). Available at https://goo.gl/1j2826, accessed in May 19, 2018, April 2016.

[FHMS03] Peter G Furth, Brendon Hemily, T.H.J. Muller, and James G Strathman. Uses of Archived AVL-APC Data to Improve Transit Performance and Management : Review and Potential. *Transportation Research Board*, 23(June 2003), 2003.

[Gur10] Zegeye Kebede Gurmu. A Dynamic Prediction of Travel Time for Transit Vehicles in Brazil Using GPS Data . Chair of Advisor Committee. *Transit*, 2010.

REFERENCES

[HWH+10] Juan C. Herrera, Daniel B. Work, Ryan Herring, Xuegang (Jeff) Ban, Quinn Jacobson, and Alexandre M. Bayen. Evaluation of traffic data obtained via GPS-enabled mobile phones: The Mobile Century field experiment. *Transportation Research Part C: Emerging Technologies*, 18(4):568–583, 2010.

[JR05] Ranhee Jeong and Laurence Rilett. Prediction Model of Bus Arrival Time for Real-Time Applications. *Transportation Research Record: Journal of the Transportation Research Board*, 1927(1927):195–204, 2005.

[Kal60] Rudolf Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering (ASME)*, 82D:35–45, 01 1960.

[Kua] KuantoKusta. Smartphone xiaomi mi a1 dual sim 4gb/32gb mdg2 gold (desbloqueado). Available at https://www.kuantokusta.pt/comunicacoes/Telemoveis-Smartphones/Smartphones-Desbloqueados/Xiaomi-Mi-A1-Dual-SIM-4GB-32GB-MDG2-Gold-Desbloqueado-p-2-305878, accessed in January 29, 2019.

[MMMMDG15] Luis Moreira-Matias, Joao Mendes-Moreira, Jorge Freire De Sousa, and Joao Gama. Improving Mass Transit Operations by Using AVL-Based Systems: A Survey. *IEEE Transactions on Intelligent Transportation Systems*, 16(4):1636–1653, 2015.

[Mor06] Usue Mori Carrascal. A Review of Travel Time Estimation and Forecasting for Advanced Traveler Information Systems. 2006.

[Pat06] Athanassios Patnaik, Jayakrishna; Chien, Steven; Bladikas. Using Data Mining Techniques on APC Data to Develop Effective Bus Scheduling Plans. *Cybernetics*, 4(1):86–90, 2006.

[PM10] Dejan Predic, Bratislav;Rancic and Aleksandar Milosavljevic. Impacts of applying automated vehicle location systems to public bus transport management. *Journal of Research and Practice in Information Technology*, 42(2):79–98, 2010.

[PR15] Klaus Pohl and Chris Rupp. *Requirements Engineering Fundamentals: A Study Guide for the Certified Professional for Requirements Engineering Exam - Foundation Level - IREB compliant*, chapter 3. Rocky Nook, 2 edition, 2015.

[PRBA14] Francisco C Pereira, Filipe Rodrigues, and Moshe Ben-Akiva. Using data from the web to predict public transport arrivals under special events scenarios. *Journal of Intelligent Transportation Systems*, 19(3):273–288, 2014.

[Sea17] Cristiano Seabra. *SAE Light Controller – Automatic vehicle location using mobile devices*. PhD thesis, Faculdade de Engenharia da Universidade do Porto, 2017.

[TJ08] Dalia Tiesyte and Christian S. Jensen. Similarity-based prediction of travel times for vehicles traveling on known routes. *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems - GIS '08*, (c):1, 2008.

REFERENCES

[vDE15] Frank van Diggelen and Per Enge. The world's first gps mooc and worldwide laboratory using smartphones. *Proceedings of the 28th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2015)*, pages 361–369, September 2015.

[Vin75] T. Vincenty. Geodetic inverse solution between antipodal points, August 1975.

[WD99] Z. Wall and D. J. Dailey. An algorithm for predicting the arrival time of mass transit vehicles using automatic vehicle location data. In *in 78th Annual Meeting of the Transportation Research Board, National Research Council, Washington D.C*, 1999.

[WVDM00] Eric Wan and Ronell Van Der Merwe. The unscented kalman filter for nonlinear estimation. pages 153 – 158, 02 2000.

[ZZL12] Pengfei Zhou, Yuanqing Zheng, and Mo Li. How Long toWait?: Predicting Bus Arrival Time with Mobile Phone based Participatory Sensing. *Proceedings of the 10th international conference on Mobile systems, applications, and services - MobiSys '12*, 13(6):379–392, 2012.

REFERENCES

Appendix A

GPS data collection results

A.1 Results for a trip from *Casa da Música* to *Afurada*, between January 09 18:43:03 and January 09 19:30:53

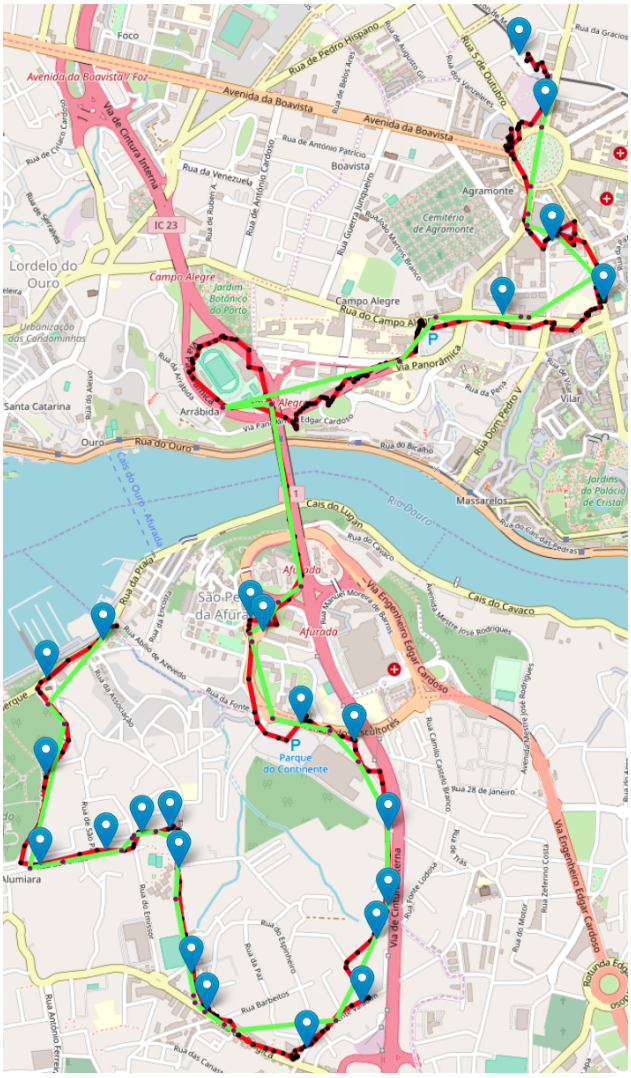

Figure A.1: GPS data collected during a bus trip from *Casa da Música* to *Afurada*

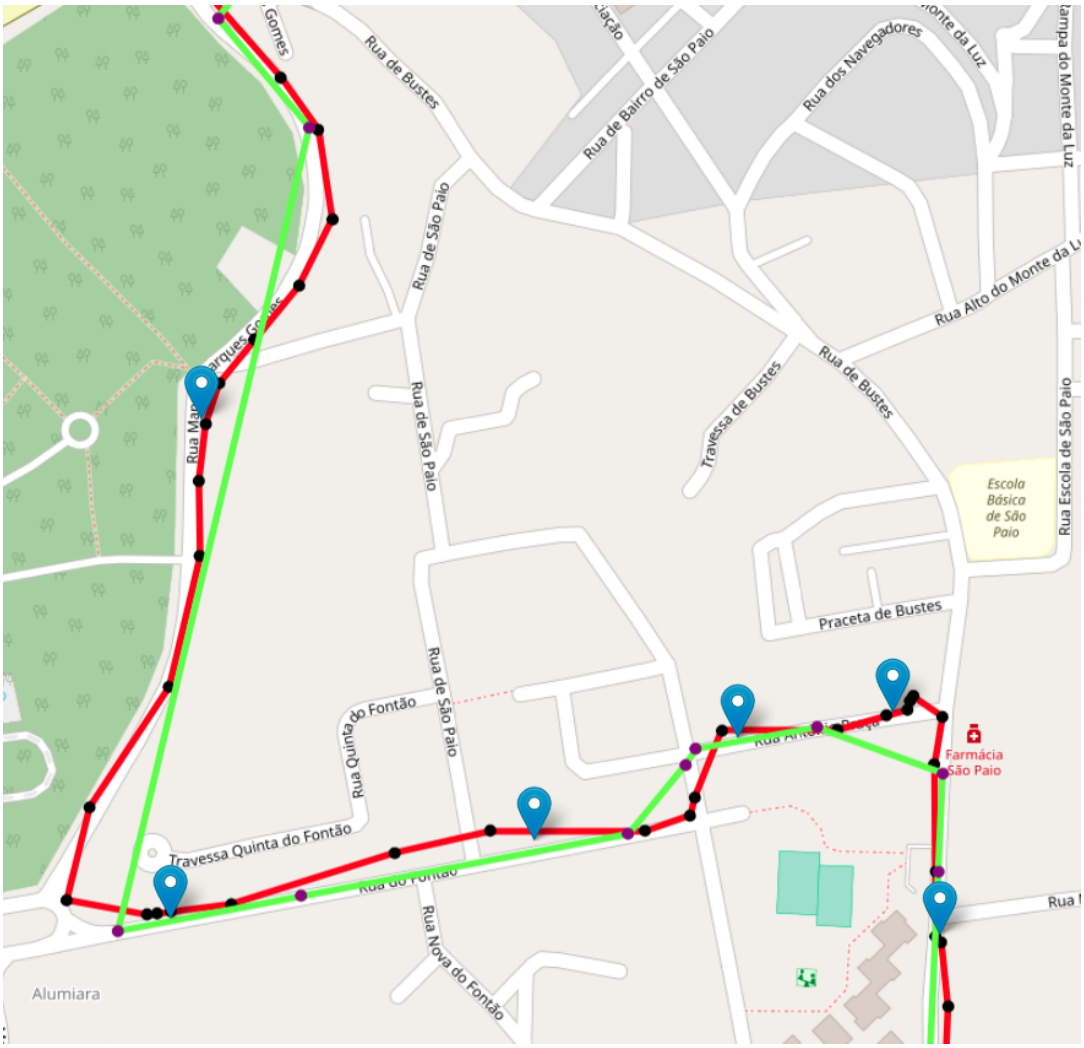

Figure A.2: Zoomed in portion of the results seen in Figure A.1

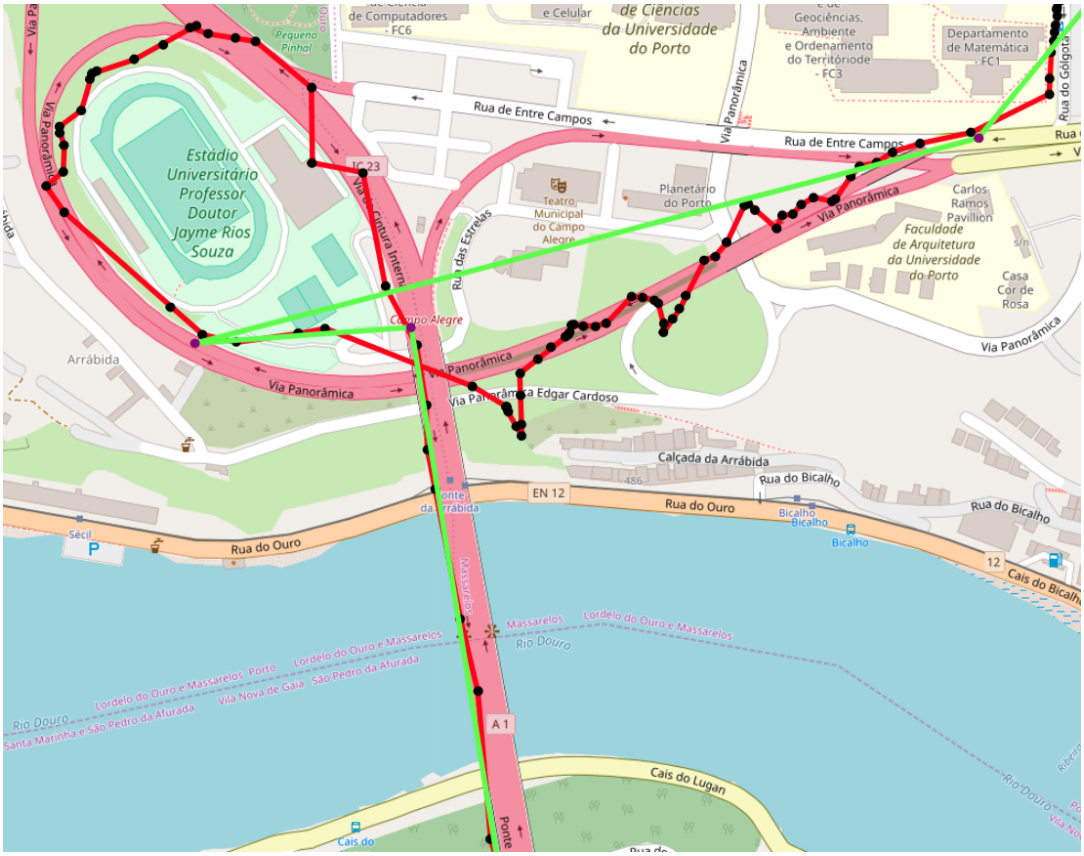

Figure A.3: Zoomed in portion of the results seen in Figure A.1

A.2 Results for a trip from *Casa da Música* to *Afurada*, between January 22 16:59:35 and January 22 17:31:49

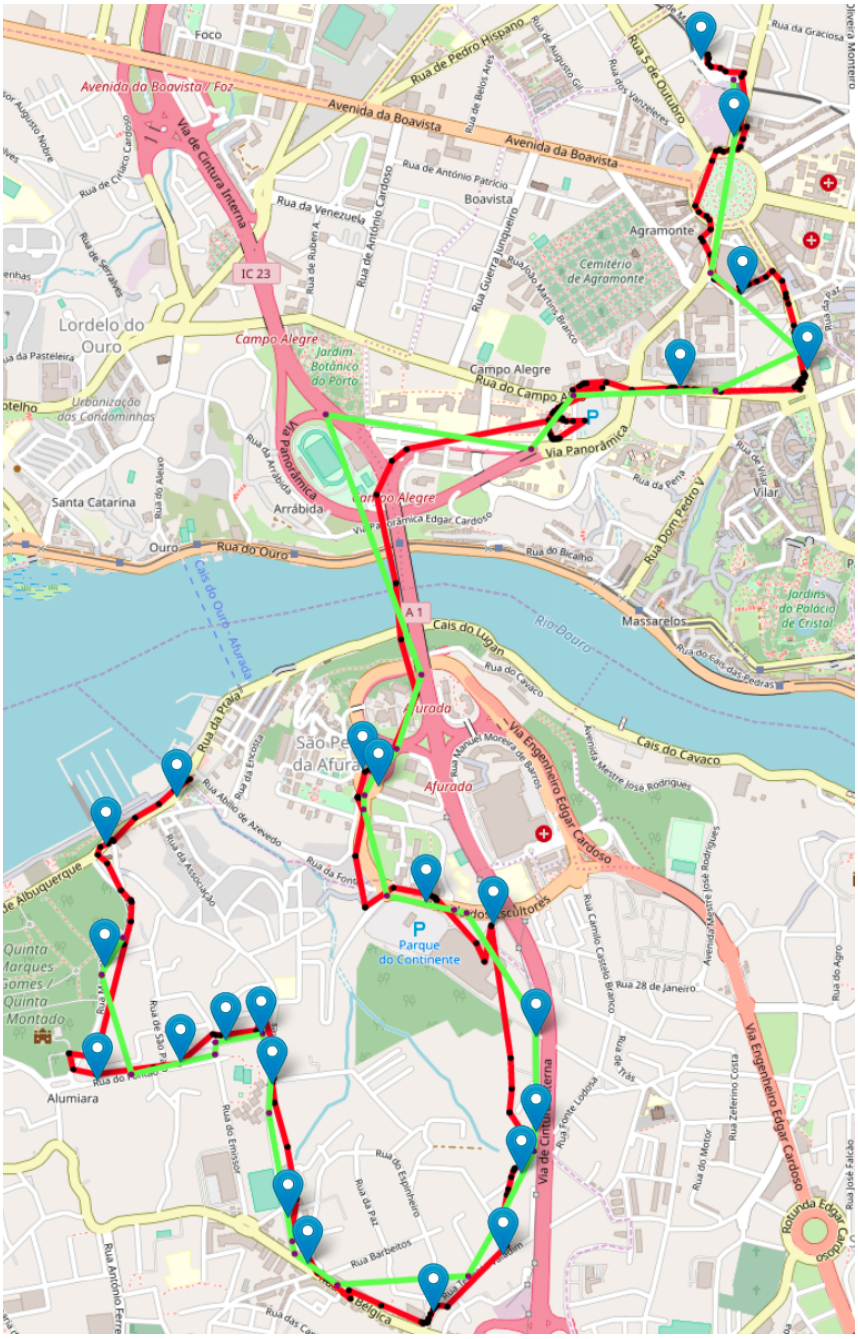

Figure A.4: GPS data collected during a bus trip from *Casa da Música* to *Afurada*

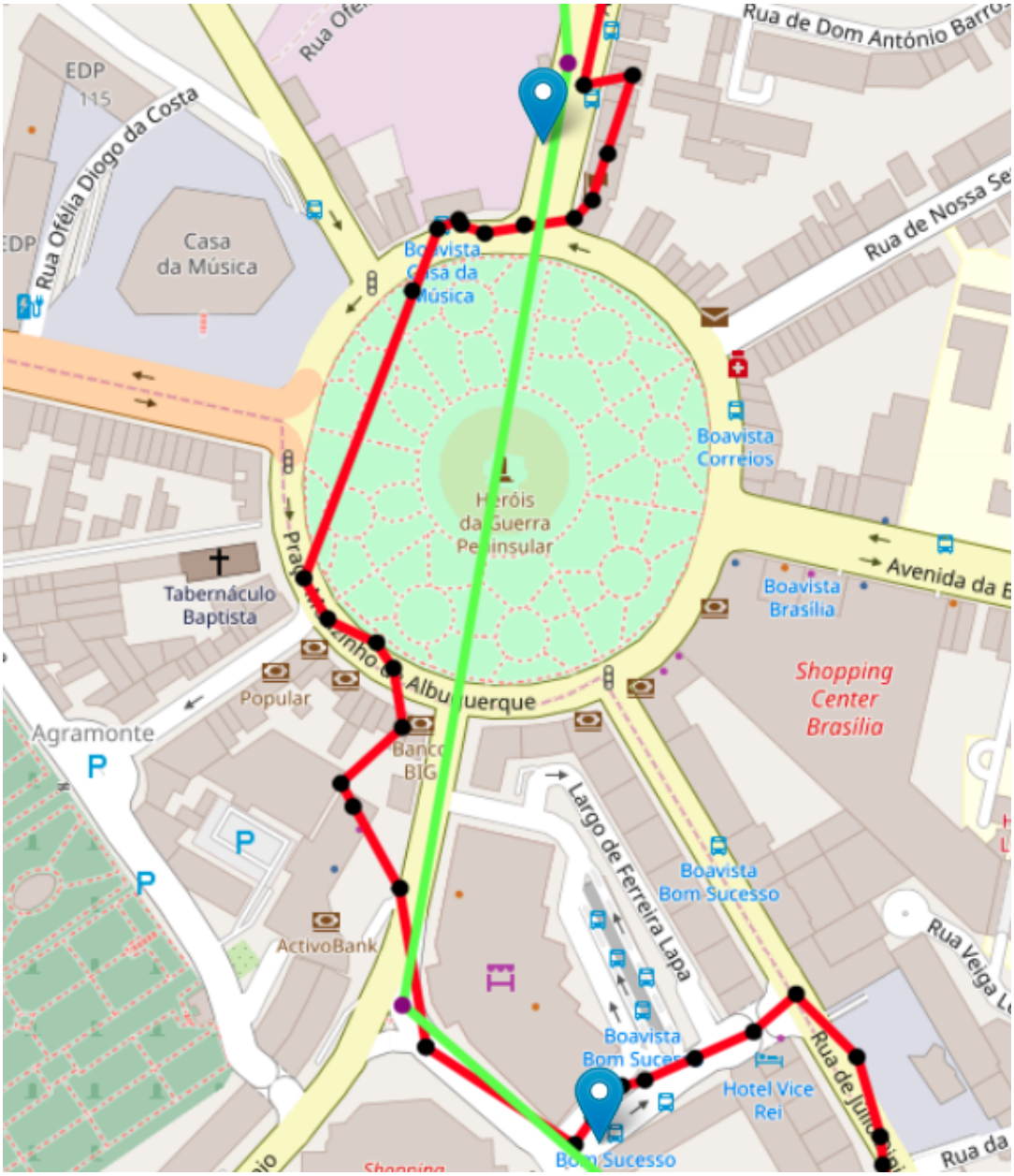

Figure A.5: Zoomed in portion of the results seen in Figure A.4

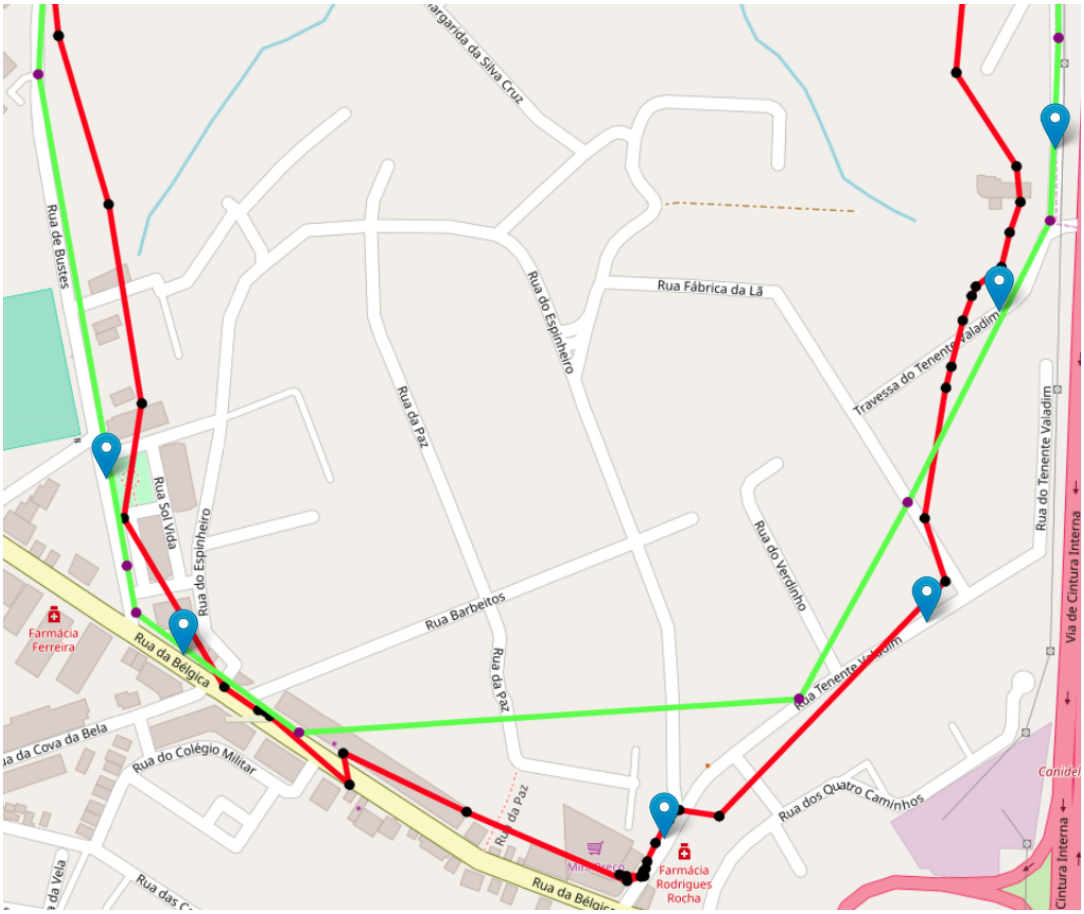

Figure A.6: Zoomed in portion of the results seen in Figure A.4

A.3 Results for a trip from *Afurada* to *Casa da Música*, between January 22 16:31:26 and January 22 16:58:50

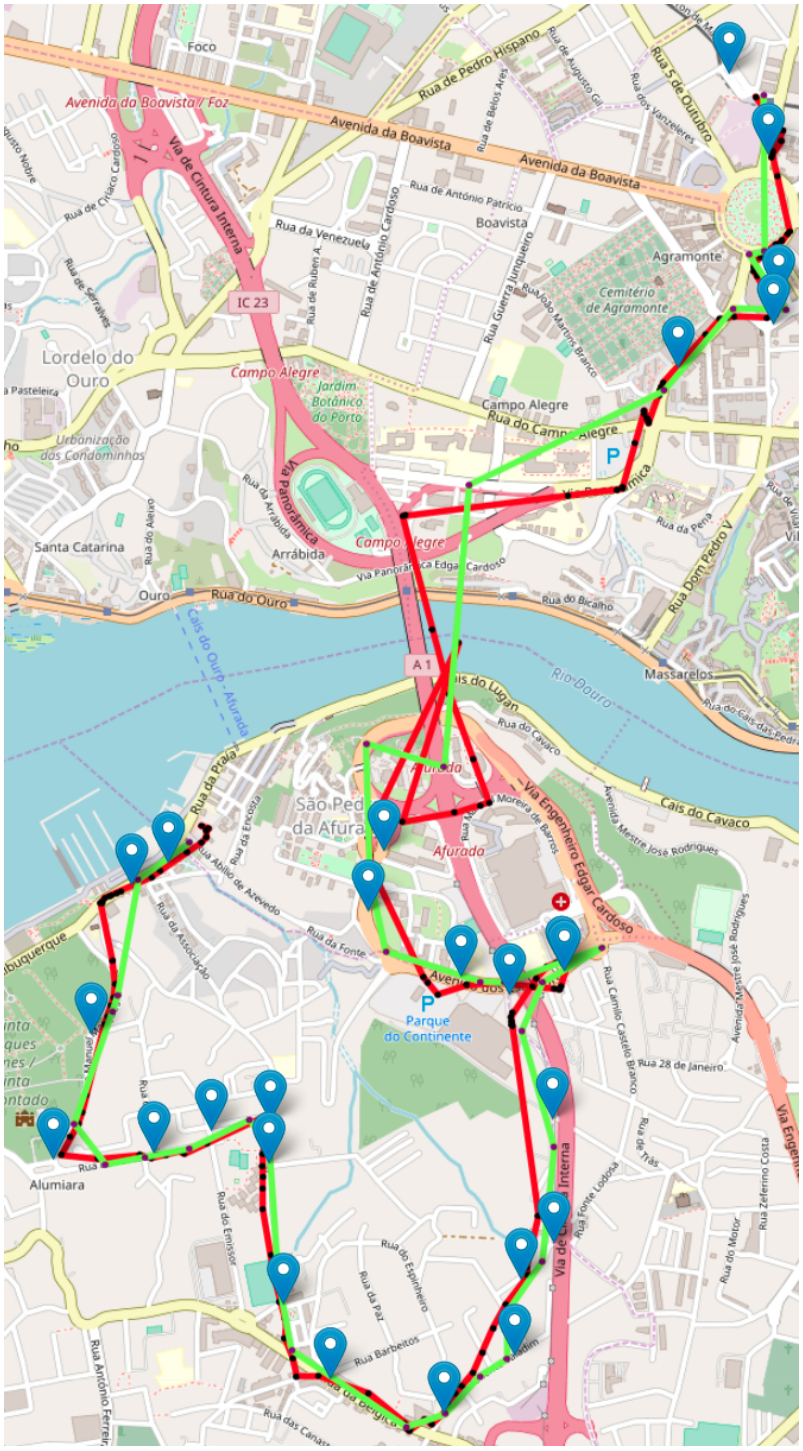

Figure A.7: GPS data collected during a bus trip from *Afurada* to *Casa da Música*

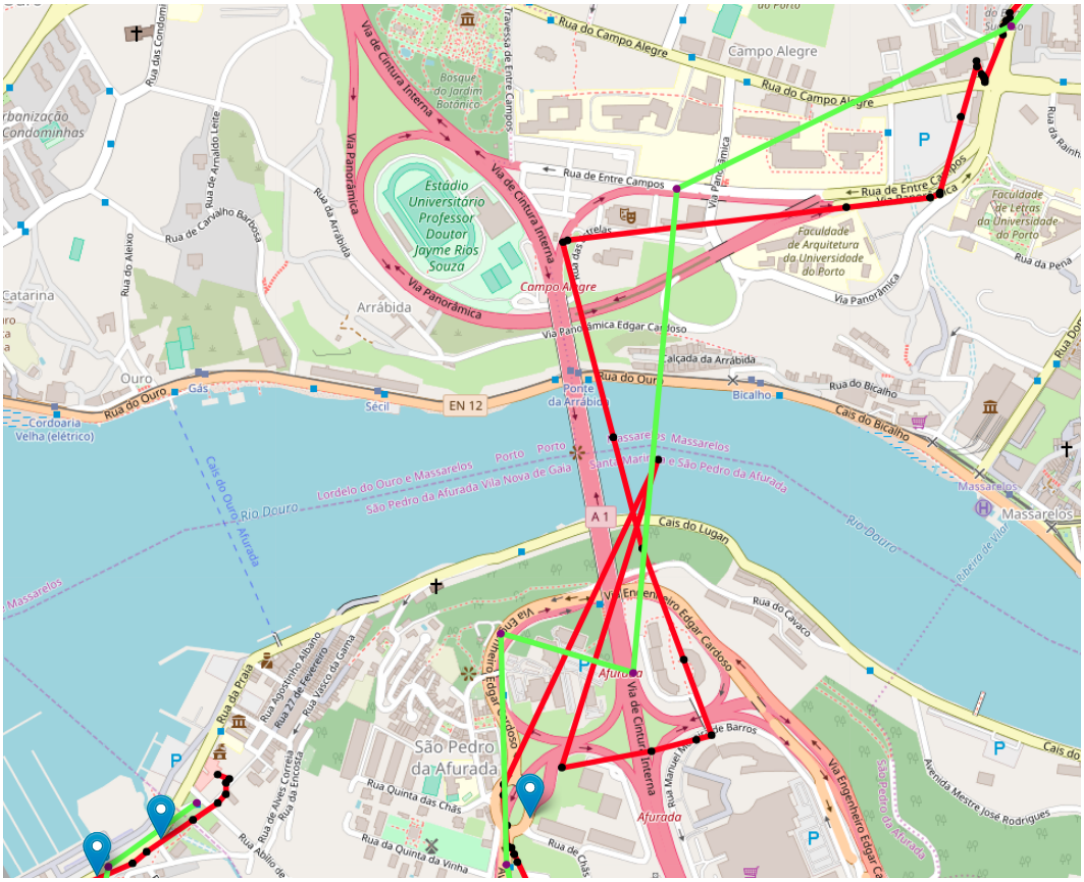

Figure A.8: Zoomed in portion of the results seen in Figure A.7

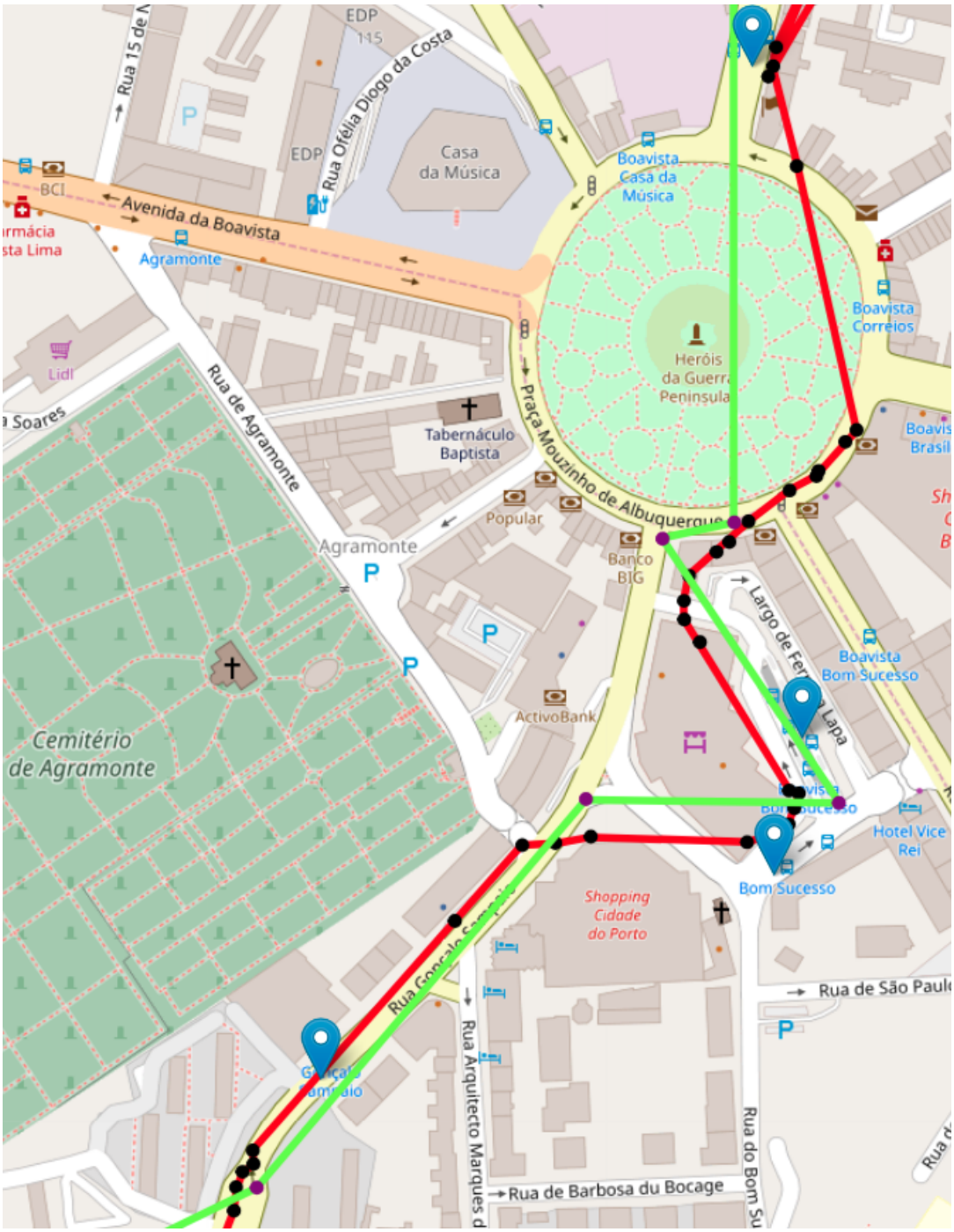

Figure A.9: Zoomed in portion of the results seen in Figure A.7

A.4 Results for a trip from *Afurada* to *Casa da Música*, between January 22 17:32:23 and January 22 18:01:19

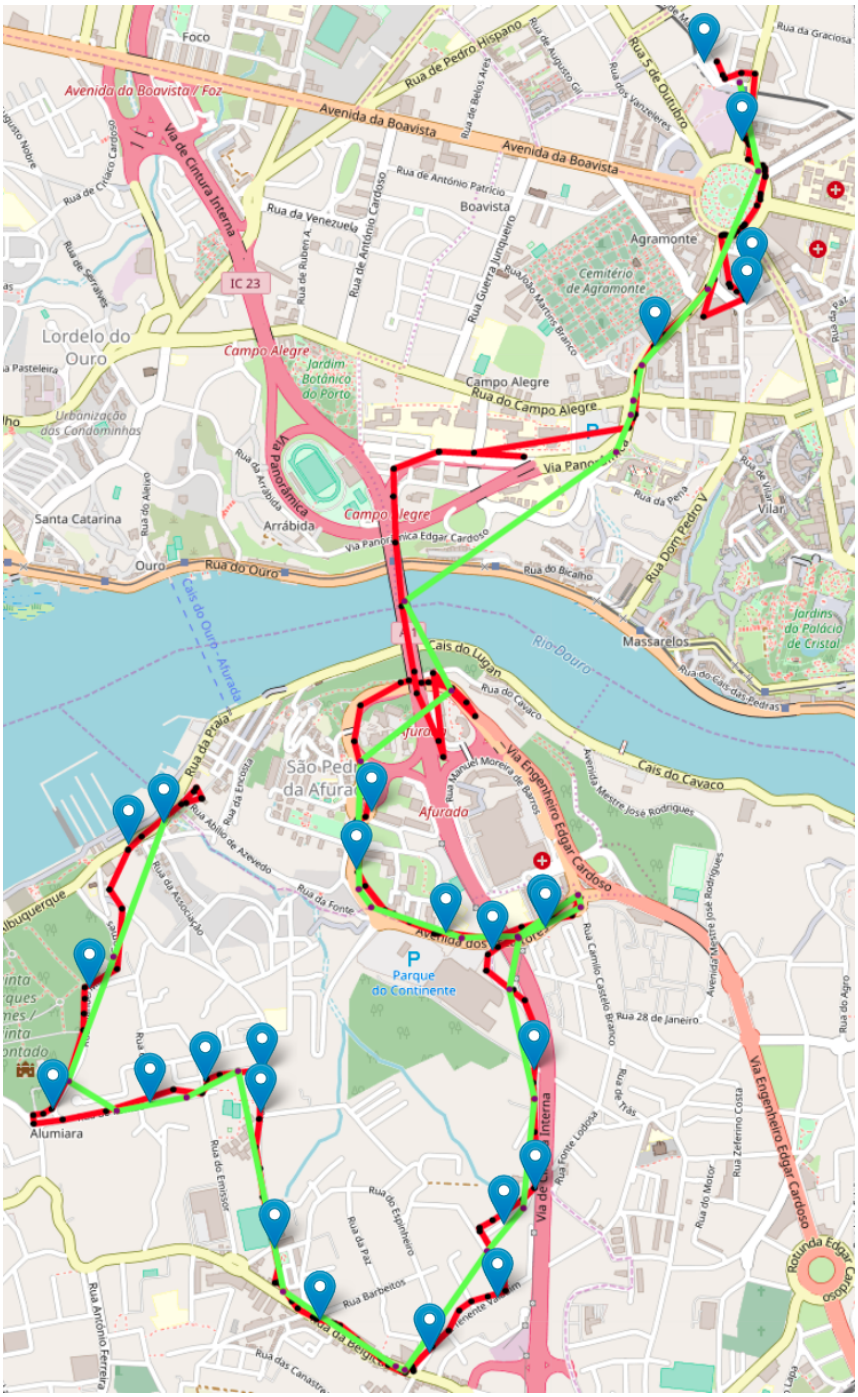

Figure A.10: GPS data collected during a bus trip from *Afurada* to *Casa da Música*

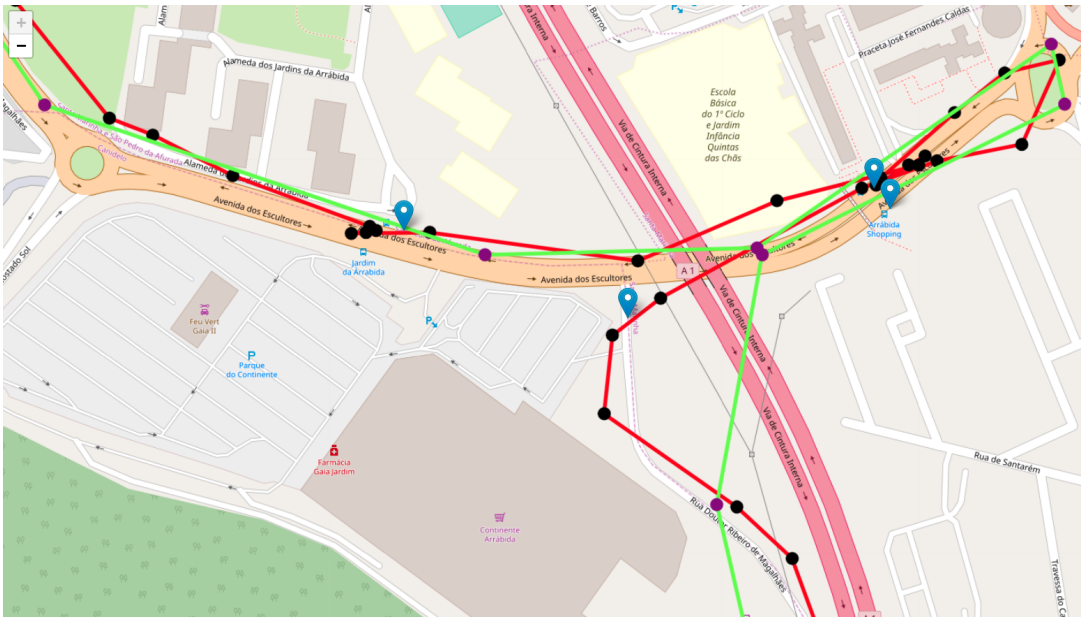

Figure A.11: Zoomed in portion of the results seen in Figure A.10

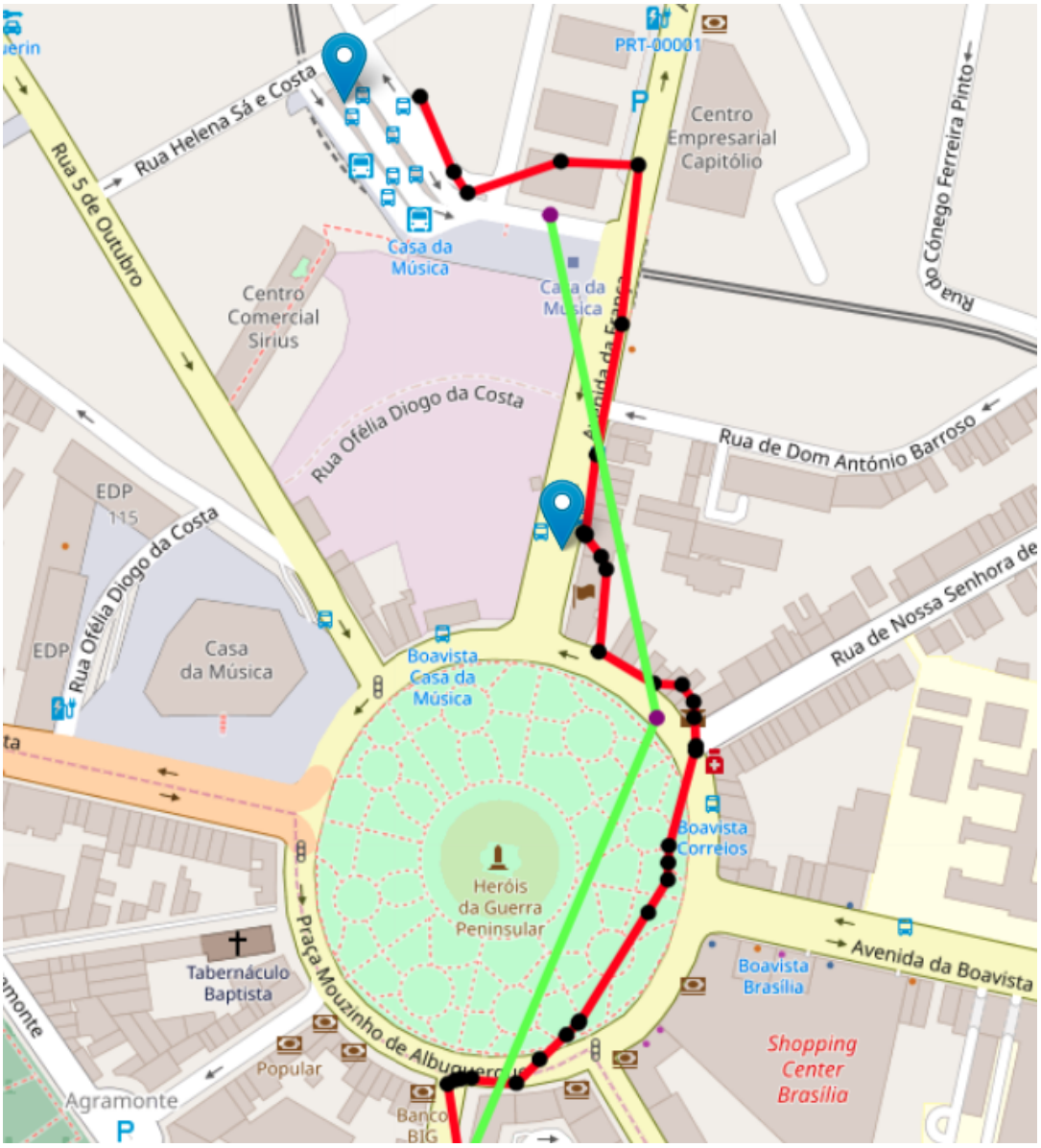

Figure A.12: Zoomed in portion of the results seen in Figure A.10

Appendix B

Results for Kalman filter predictions and ETA predictions

B.1 Results for a trip from *Casa da Música* to *Afurada*, between January 09 18:43:03 and January 09 19:30:53

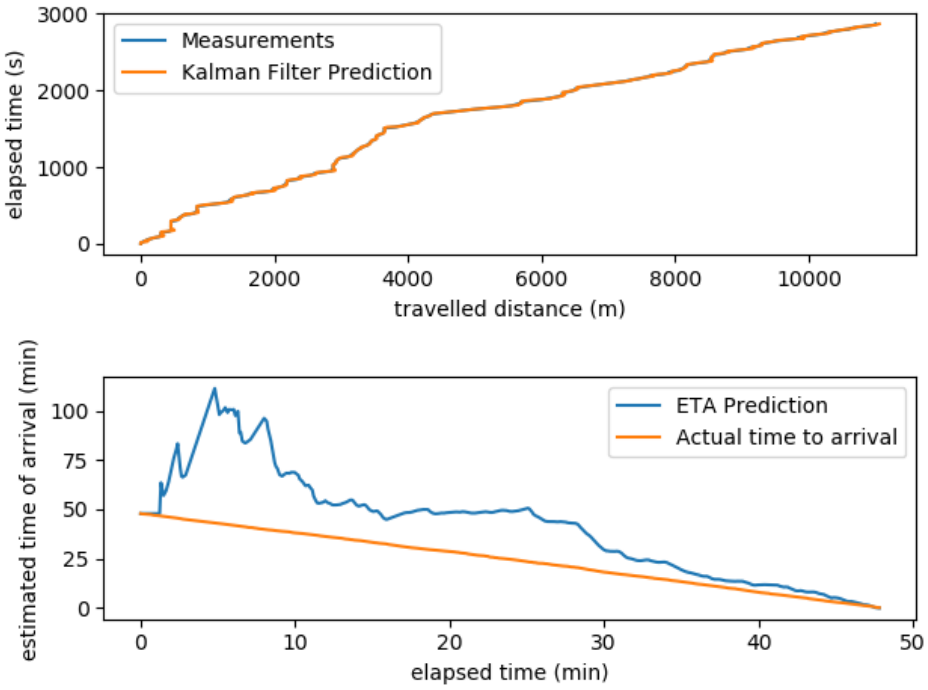

Figure B.1: Predictions for a trip from *Casa da Música* to *Afurada*

Results for Kalman filter predictions and ETA predictions

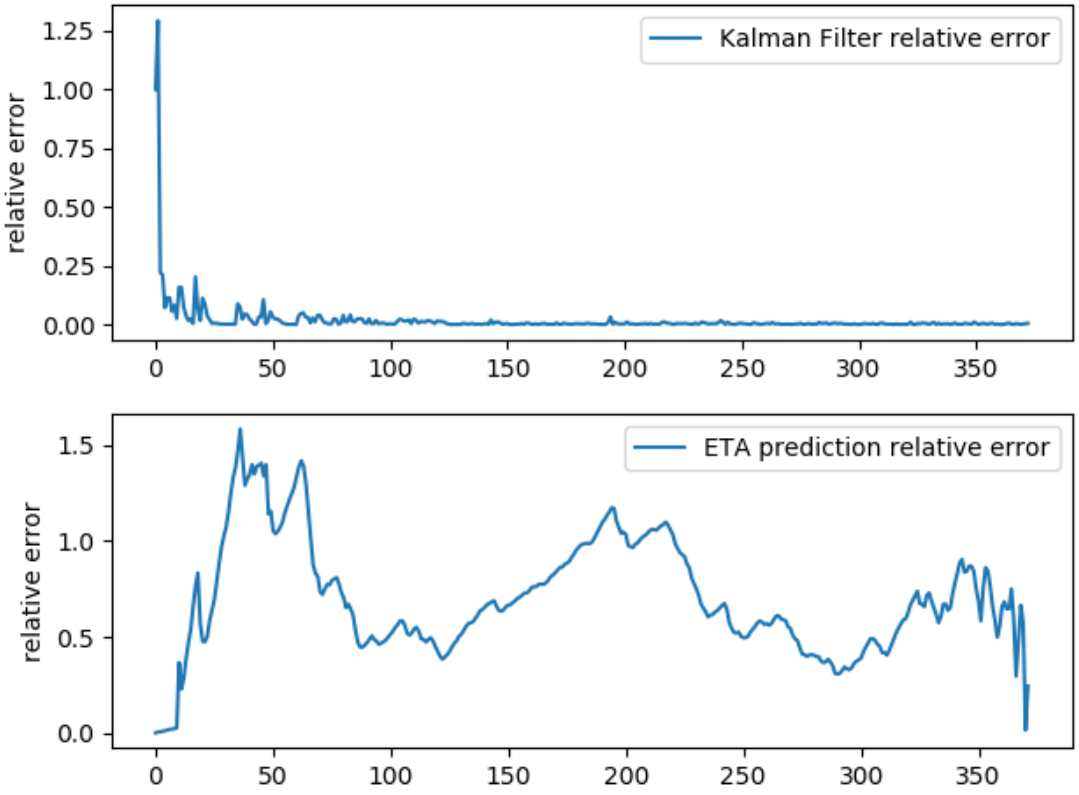

Figure B.2: Relative errors for the predictions in Figure B.1

B.2 Results for a trip from *Casa da Música* to *Afurada*, between January 22 16:59:35 and January 22 17:31:49

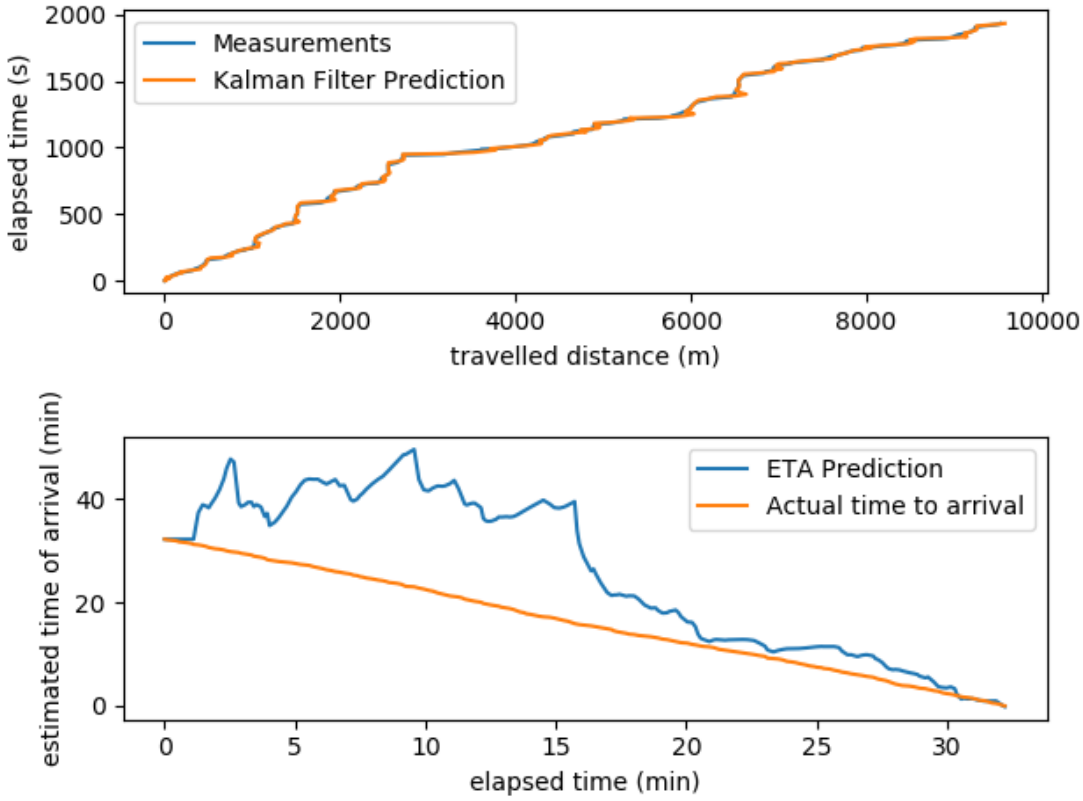

Figure B.3: Predictions for a trip from *Casa da Música* to *Afurada*

Results for Kalman filter predictions and ETA predictions

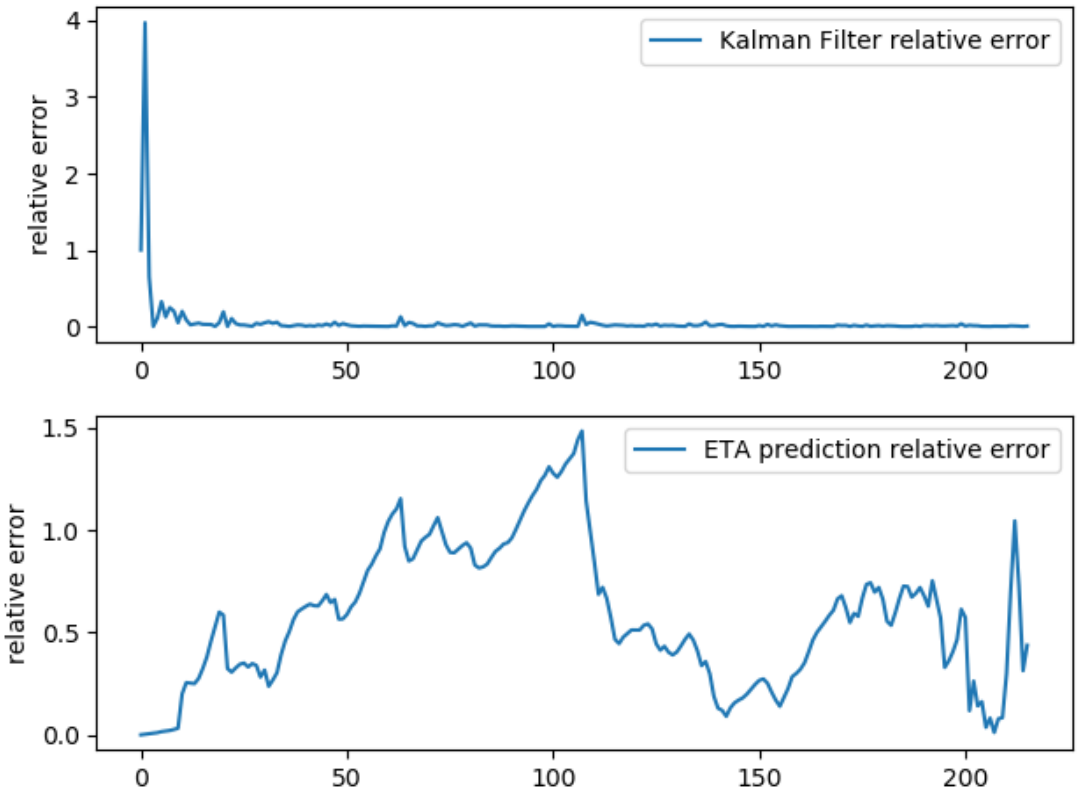

Figure B.4: Relative errors for the predictions in Figure B.3

B.3 Results for a trip from *Afurada* to *Casa da Música*, between January 22 16:31:26 and January 22 16:58:50

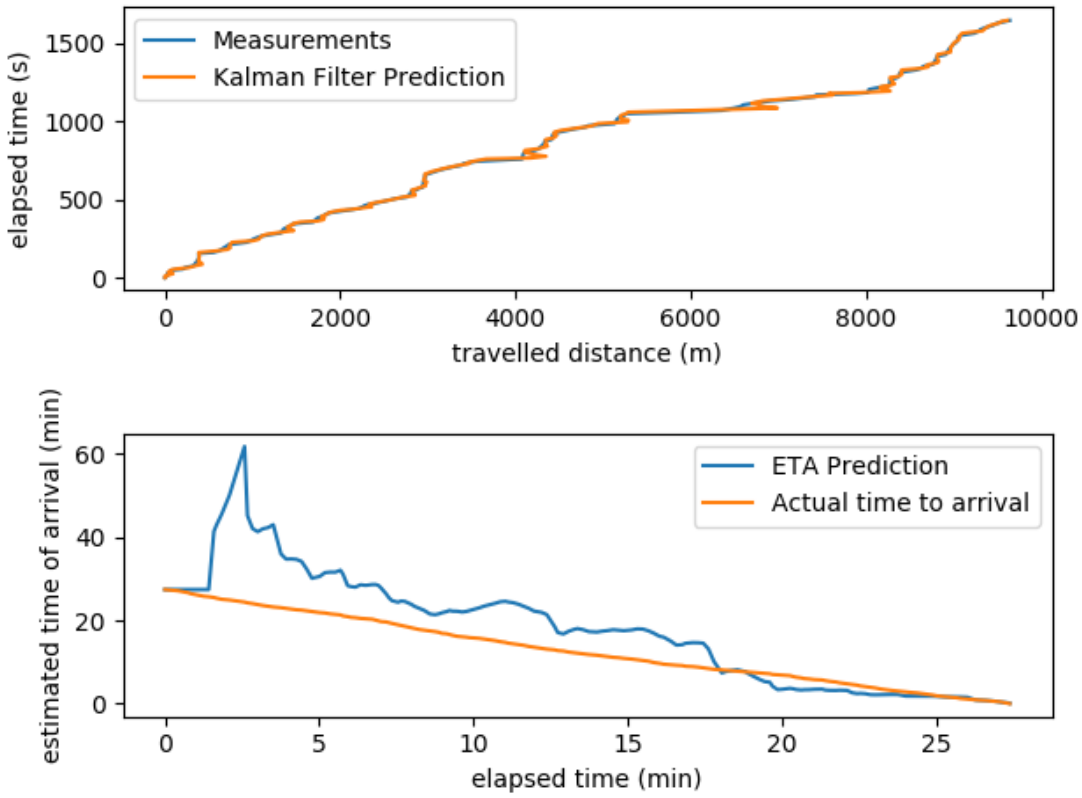

Figure B.5: Predictions for a trip from *Afurada* to *Casa da Música*

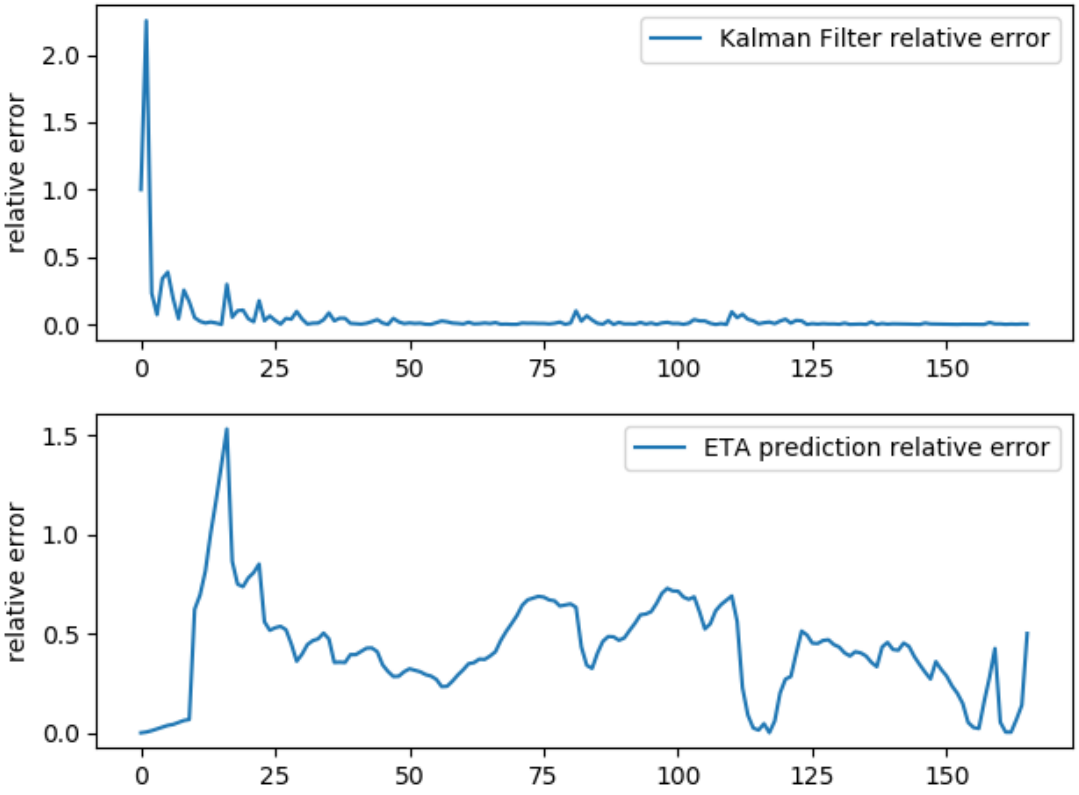

Figure B.6: Relative errors for the predictions in Figure B.5

B.4 Results for a trip from *Afurada* to *Casa da Música*, between January 22 17:32:23 and January 22 18:01:19

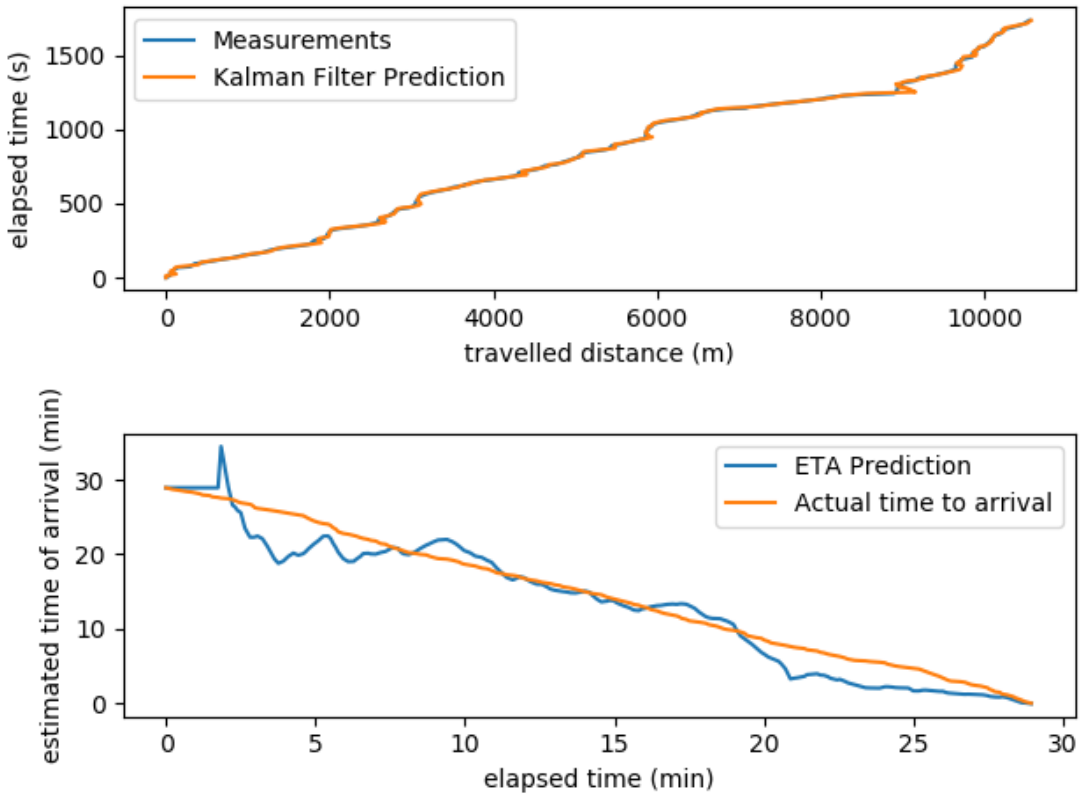

Figure B.7: Predictions for a trip from *Afurada* to *Casa da Música*

Results for Kalman filter predictions and ETA predictions

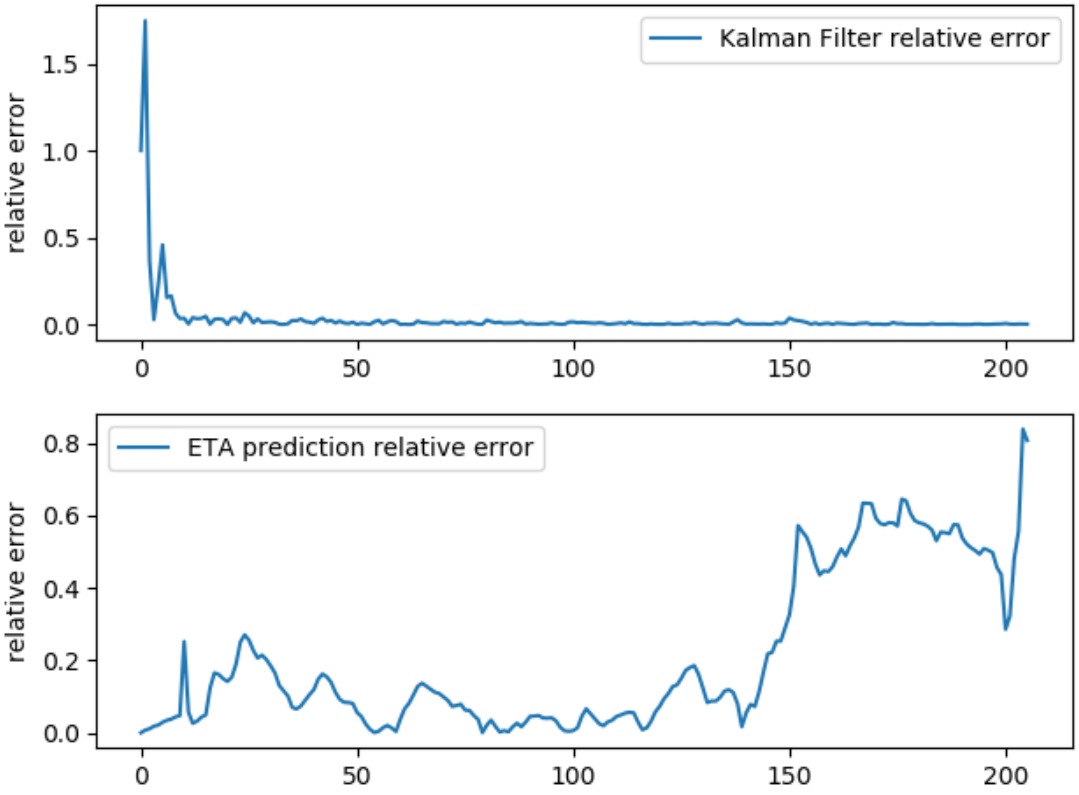

Figure B.8: Relative errors for the predictions in Figure B.7