University of Bristol

Doctoral Thesis

On Urban Vehicular Ad-hoc Networks and Parking Systems

Author:
Pietro Carnelli

Supervisors:
Prof. R.E. Wilson,
Dr G. Oikonomou and
Prof. M. Sooriyabandara

A thesis submitted in fulfilment of the requirements for the degree of Engineering Doctorate in Systems

in the

Systems Centre
Department of Civil Engineering

September 7, 2020

Declaration of Authorship

I, Pietro Carnelli, declare that the work in this dissertation was carried out in accordance with the requirements of the University's Regulations and Code of Practice for Research Degree Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

Signed: _____

Date: _____

UNIVERSITY OF BRISTOL

Abstract

Faculty of Engineering
Department of Civil Engineering

Engineering Doctorate in Systems

On Urban Vehicular Ad-hoc Networks and Parking Systems

by Pietro Carnelli

The nexus between wireless communication and transportation systems is investigated for urban areas. Both near and longer term scenarios are considered. In the near term, we research potential methods for reducing inner-city vehicle congestion through use of wireless enabled vehicle-parking occupancy detection and information dissemination. Subsequently, we developed a low-energy, smart-phone based vehicle-parking activity detection system. Our proposed system achieved high parking activity detection accuracies using low-energy sensors found in typical smart-phones.

With regards to longer term urban transport and wireless systems, we investigated the feasibility of using a connected fleet of (potentially autonomous) vehicles, also referred to as a VANET (Vehicular Ad-hoc NETwork), to provide temporary network connectivity to a city-wide (delay tolerant) sensor network. Initially we investigated how such a system would perform using real-world taxi trace datasets that were publically available online for Rome (Italy) and San Francisco (USA). We combined the (filtered, map-matched and 'folded') taxi-traces with a road network and Line-of-Sight (LoS) model (based on OpenStreetMap road and building footprint datasets) for our VANET simulations. Simulation results show that increasing our connected fleet size relative to the number of sensors reduces end-to-end delay. However, a performance plateau was observed after increasing taxi fleet sizes beyond a certain ratio of vehicles to sensors.

Finally, we investigated methods of improving the performance of our city-wide VANET system. We developed an agent-based VANET simulation that allowed for strategic re-routing of connected autonomous vehicles, as they performed their passenger trips, in order to increase sensor message mean packet delivery ratio values. Results show that even a minor increase in typical passenger journey lengths (circa 500m or approximately a tenth of the median passenger journey length) increased the final mean sensor packet delivery ratio values.

Acknowledgements

This work was supported by the EPSRC funded industrial Doctorate Centre in Systems (Grant EP/G037353/1) and Toshiba Research Europe Ltd. The author was a research engineer with Toshiba Research Ltd (Telecommunications Research Laboratory) and with the Industrial Doctorate Centre in Systems, at the Universities of Bristol and Bath (UK). In addition I would like to thank:

- My academic supervisors, R.E. Wilson, G. Oikonomou, and A. Johansson, for their invaluable contributions, discussions and guidance throughout the project
- My industrial supervisor, M. Sooriyabandara, for his enthusiasm, patience and support throughout the project
- My colleagues at Toshiba for their technical and kind support with developing software applications and assisting with data collection trials
- My family for insisting I complete a post-graduate degree and their financial support throughout my education

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Two city sub-systems, namely vehicular transport and wireless infrastructure, and their interactions, are considered in this thesis. Both are affected by increasing urbanisation globally. As populations living in urban areas increase, the demand for transport and wireless access to the Internet, and general seamless connectivity between citizens generally follows. For example, both transport and wireless systems are needed in order for humans to work either via commuting to a physical workplace or working remotely with Internet access to their workplace.

Furthermore, as cities become ever more sophisticated in terms of sensors monitoring everything from vehicle traffic at intersections to air quality, the need to connect these sensors to the Internet will be critical for their intended operation. Connecting every sensor to a physical networked link (e.g., using Ethernet or a fibre optic back-bone) would be ideal in terms of reliability, connection bandwidth and delay. However, the cost and complexity of laying or retrofitting network cables in order to connect a city-wide sensor network can be prohibitive for most local municipal governments. Finding a common interface or method of working between vehicular transport and wireless systems is the over-arching theme of this thesis.

Initially inner city vehicle-parking systems will be explored and investigated with the eventual aim of improving information dissemination between drivers. Cities such as San Francisco (USA) have implemented a near city-wide (circa 7,000 on-street parking spaces) smart parking system referred to as SFpark [79]. The aim of SFpark is to reduce inner-city vehicle congestion by varying the price to park a vehicle on-street in the city centre and by providing drivers with real-time vehicle-parking availability information. In order to collect parking occupancy and price data, SFpark retrofitted over 6,000 on-street connected parking meters (one per designated on-street vehicle parking space). At the time, SFpark cost over 20M US dollars.

Rather than retrofitting all inner-city parking spaces (a somewhat lengthy and costly process), we aim to investigate whether smart-phone based applications could provide the necessary platform to detect vehicle-parking activity, and view on-street vehicle-parking availability. Future versions of our system could incorporate a method for electronic payment. Our proposed method (referred to as ParkUs, Chapter 2) relied on using smart-phone sensor data streams (such as those generated by a magnetometer, accelerometer and GNSS) to detect and locate a change in transport modality.

By harnessing sensor data and wireless connectivity, given the wide adoption of smartphones in developed cities [52], we aim to provide better, near real-time parking occupancy levels prior to a user journey. In particular, our focus will be to provide an infrastructure-less and accurate parking-activity detection and localisation method with minimal smart-phone energy consumption.

Whilst tackling near-term inner city congestion problems is of great interest, however, given the predicted advent of Connected Autonomous Vehicles (CAVs) over the coming decades [108], the rest of the thesis will explore and investigate whether CAVs could feasibly provide a city-wide sensor network with connectivity.

In the longer term, once CAV technology has been developed in such a manner to provide safe and efficient transport, CAVs will likely replace most human-driven vehicles. Further potential benefits of CAVs include the ability to allow those not able to drive (for example the visually impaired) to commute across a city independently. CAVs ability to move without human drivers opens up a raft of other potential applications and use-cases.

In particular, CAVs will require some form of wireless connectivity (alongside vision and ranging technologies) in order to avoid collisions when conducting driving manoeuvres such as lane changing or parking. Given that Vehicular Ad-hoc Networks (VANETs) will be necessary for Autonomous Vehicle (AV) operation, several other applications could benefit, or 'piggy-back' off such a city-wide connected fleet of mobile nodes/wireless access-points. One application explored in this thesis (see Chapters 3 and 4) involved using a fleet of CAVs to provide some form of wireless connectivity to a distributed (sparse) city-wide sensor network.

As sensing technology reduces in cost, more cities are using them to monitor various aspects of their environment. In particular, we are interested in connecting delay tolerant sensors, such as those that do not provide a high sampling or update frequency (e.g., air quality monitoring or household energy consumption). Furthermore, such delay tolerant sensors may not need to provide continuous data streams. As edge processing systems combined with sensors become more popular, only metadata such as statistical features might be transmitted instead, further reducing connectivity requirements.

We subsequently explore through modelling and simulation the feasibility of using a CAV fleet's VANET system to provide connectivity to a city-wide sensor network. Initially, we used openly available real-world taxi trace datasets for two cities: Rome (Italy) and San Francisco (USA). Local building footprint and road network topology datasets were obtained through OpenStreetMap. By 'folding' multiple days of taxi trace data we were able to investigate how end-to-end delay and Packet Delivery Ratios (PDR) vary with differing taxi fleet and sensor network sizes.

Given the results obtained from our initial study (Chapter 3) we then researched and tested a method of improving final PDR values across the city-wide sensor network. We developed an edge-weighting function in order to incentivise efficient routing (of the simulated CAVs) in order to provide better wireless sensor network connectivity by visiting more sensors per passenger trip. We achieved this without overly lengthening passenger trip lengths.

1.1 Cities

Over the last century humanity has slowly become urbanised. The UN Urbanisation Report finds that 55% of the world's human population currently live in cities [97]. A definition of a city can be somewhat misleading and open to interpretation. The UN uses the following three definitions: 'city proper': the smallest enclave, where the original city settlement began (often covering the central business and local city government buildings districts), 'urban agglomeration': the surrounding area, often can be thought of as suburbs and 'metropolitan area' which covers an even larger surrounding area often including small 'satellite' or 'commuter' towns. In general, the UN used the 'urban agglomeration' definition and a soft-limit/minimum population threshold of three hundred thousand human inhabitants in order to define a city/urban area.

Definitions aside, the proportion of urban dwellers is expected only to increase, in part as generations now settling in cities earn more and raise families, and through continued migration (mostly for economic reasons). The UN predicts that by the year 2050 over two thirds of humanity will live in a city. This generates certain challenges for future city planners and administrators as to how to manage the increase in urbanisation in a sustainable manner.

Cities were first treated formally as systems when General Systems Theory and Cybernetics was applied to the social sciences discipline in the late 1950s. The focus at the time was on ways in which the elements comprising the city system interacted with one another through hierarchical structures. These structures allowed for feedback, or control loops, designed to keep the entire system stable within certain bounds. However, reasoning behind cities as systems has evolved to consider that most systems cannot be organised into simple, distinct hierarchical structures. Instead cities were viewed as consisting of 'messy' sub-systems that interact with each other through specific interfaces [5].

Batty goes as far as suggesting that the notion of equilibrium and dynamics (central to modern systems theory) was initially conceived, or at the very least, "endorsed" by Adam Smith (an 18$^{\text{th}}$ century Scottish economist and philosopher)[1]. Examples of these city components/sub-systems include (but are not limited to) transportation, energy networks, water, education, local government, policing, waste removal, health-care, and businesses. It is the interaction between these sub-systems that defines the overall performance of a city, either in terms of economic output or more human-centred measures such as the UN's Human Development Index (HDI) [5].

[1] Adam Smith coined the phrase "invisible hand" in his magnum opus titled, The Wealth of Nations, when positing why free markets (can) reach a price equilibrium between supply and demand. Adam Smith was most likely referring to the certain amount of (non-quantifiable) belief required (on behalf of those trading within the market) for the market system to operate effectively and efficiently. At some level one must be convinced that the goods or services being exchanged are worthwhile.

1.2 Vehicle Transportation Systems

Automotive vehicles (of various guises, sizes and powertrains) account for 90% of all passenger traffic kilometres travelled and nearly two thirds of all freight in the UK (see Tables TSGB0101 and TSGB0401 of Transport Statistics Great Britain, Department for Transport [94]). Similarly, in the USA, where most cities started to materialise around the time when vehicle manufacturing was booming, 80% of all commuting journeys are by motor vehicle (see Table 1-41 of National Transportation Statistics from the US Bureau of Transportation Statistics [87]).

Demand for transport is unlikely to reduce in the coming decades. As more people move to live and work in cities, transportation systems will feel ever more under pressure to cope with increased demand. Furthermore, transport is not limited to those that commute to work, see Table 1.1. In the UK, a larger proportion of trips are in fact for leisure, followed by shopping and then commuting[2]. Nonetheless, vehicular transport is still the dominant mode: according to the Royal Automobile Club Foundation for Motoring (referred to simply as the RAC Foundation) there are over 38M licensed vehicles in the UK [60].

Future mobility, at least within large urban agglomerations, may shift away from individual (private) car ownership and towards a subsidised private-public partnership of a bookable fleet of vehicles. Benefits from such a system include drastically reducing the number of vehicles on the road (through higher utilisation per vehicle) whilst maintaining similar levels of availability. After all, many studies have shown that on average, a (privately owned) vehicle spends at least 90% of its life stationary. Other benefits include efficient maintenance, as fewer different parts are required for fleet maintenance (assuming they are the same vehicle model). Some ride hailing and vehicle rental companies have recently started partnering with vehicle manufacturers (examples include but are not limited to BMW and Toyota) in order to provide vehicle transport/mobility as a service. In these agreements, the manufacturers provide the assets (vehicles) as well as coordinate their maintenance, whilst the ride hailing or rental firms provide the supply of customers, via their on-line platforms, for scheduling vehicle or trip bookings.

Vehicle transportation systems rely on two key assets, the vehicles and the underlying road network infrastructure (not just the physical roads, but refilling stations, traffic lights, signage, parking bays etc.). Both are fundamental for a city's transportation system. However, the road infrastructure is mostly fixed in size and topology, since it is almost impossible to build more roads in a city without having to purchase the land to do so, which requires enormous capital expenditure. Consequently vehicle congestion in cities is increasing.

[2] At a recent machine learning systems conference attended by the author, an employee of a large internet search firm with interests in mapping and smart-phone operating systems and services admitted (albeit strictly 'off the record') that out of their near 2B monthly active users, they were able to accurately (within reason) predict roughly half of their trips (best hypothesis being that most of these trips were work commutes or parents on school runs). Yet even with the firm's vast user base, datasets and computational resources, they were not able to predict or find any common trend to account for the other (roughly speaking) half of trips the recorded.

Two aspects of vehicle transportation systems will be investigated and discussed in this thesis. The first looks at a current scenario where we investigate parking behaviour of drivers and congestion caused by drivers searching for a vacant parking space within cities.

Research conducted in the USA by Donald Shoup suggests a significant proportion of inner city traffic is caused by drivers circling streets searching for a vacant parking space [81, 83]. He estimated this proportion of vehicle congestion to be circa 20–30% depending on the city (and era of study). One potential solution, as championed by Shoup, is to stop subsidising on-street parking[3] and to increase fares when demand rises (similar to surge pricing within on-line ride hailing platforms).

However, increasing vehicle on-street parking prices alone will not stop people searching for a vacant parking space if they are already on their way to their destination. Potentially, having parking occupancy information before a trip commences could allow us to 'nudge' would-be drivers to using some other mode of transport (e.g., public transport or a taxi cab). If not, we could at least encourage drivers to search/park where parking spaces are readily available.

The second aspect of vehicle transportation systems explored in this thesis concerns future cities, where mobility is seen as a service and most vehicles will be autonomous and have dedicated wireless systems. In these scenarios we develop two different vehicle traffic models; one based on real world vehicle trace datasets (Chapter 3) and the other (Chapter 4) through an agent-based model of CAVs and passengers. Both chapters aim to tackle the problem of providing connectivity to sensors distributed across an urban area. In practice these sensors could be air quality monitors, cameras or even parked CAVs not currently in use. Given the sheer abundance of different types of sensors (from LiDAR to cameras to ultrasonic ranging devices) required for autonomous control, parked CAVs could provide environmental data as well as acting as local data storage for up-to-date LiDAR maps.

1.3 Wireless Systems for Autonomous Vehicles

In order for fully autonomous vehicles to become a consumer reality, several key technologies need to be researched, developed and deployed [108]. A large amount of research has been conducted in order to improve current imaging and image processing technologies to inform CAVs of obstacles, other road users, and the road layout ahead (see [68, 74, 76]).

However, making decisions such as route planning or vehicle avoidance in a safe and timely manner requires a V2X wireless communication system. This is particularly important when obstacles are outside the Line-of-Sight (LoS) of the vehicle. Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) networks are often cited as the solution to these problems as they allow vehicles to communicate with each other for safety and to ensure better road utilisation.

[3] Shoup goes further by suggesting that the core of the problem lies in building regulations which stipulate (although they do vary between the states) a minimum number of parking spaces for different types of buildings, thus creating an almost permanent incentive to own and use vehicles in cities [82].

TABLE 1.1: Average number of trips by purpose in England, 2016, sample size: 954,000. Table reproduced from Transport Statistics Great Britain, Department of Transport, Table TSGB0104 [94].

Trip Purpose	**Fraction of Recorded Trips**
Commuting	0.15
Business	0.03
Education/Parental Escort	0.12
Shopping	0.19
Other Escort	0.09
Personal Business	0.09
Leisure	0.26
Other	0.06

In addition, a central networked infrastructure might allow CAVs to obtain the latest road map updates as well as any other related updates such as bad weather, or roadworks along their planned routes.

Currently, Dedicated Short Range Communications (DSRC), cellular and millimetre-wave (mmWave) wireless technologies are the main competing systems for V2X communication. DSRC, based on the IEEE 802.11p/WAVE protocols [36], is currently considered the default V2V technology due to its rapid information dissemination frequency (10Hz) and low latency. The USA's National Highway Traffic Safety Administration (NHTSA) mandated that all new production vehicles support DSRC from 2020 onwards[4].

Cellular technologies such as 3G and LTE can offer larger capacity and increased wireless communication range compared to IEEE 802.11p/WAVE systems. However, these cellular systems are not currently used for V2V as they do not satisfy the stringent delay requirements for V2V safety applications (like vehicle avoidance, or coordination at intersections). Modifications to the LTE standard, in particular the Device-to-Device (D2D) system, have yielded increases in average Packet Received Ratio (PRR), end-to-end delay, and reliability necessary to be used for V2V systems [24, 25]. However, neither the cellular (3GPP) or the Wi-Fi based IEEE standards achieve particularly high data rates. Most aim to support low data rates of 2–6Mbps necessary for safety/short message applications.

For higher data rate applications, such as infotainment or LiDAR, other technologies must be developed. Recently, research has focused on higher frequencies such as those with millimetre wavelengths to achieve higher data rates

[4]As of recently (November 2017), the current USA government administration revoked this mandate [73]. It is hypothesised the mandate was revoked due to lobbying from mass vehicle production manufacturers wanting to avoid including extra components that would erode their already wafer-thin profit margins. This semi-hostile attitude to including new vehicle safety technology is not particularly novel. Previously, vehicle manufacturers were against including air bags and seat belts as standard. Afterall "only dead fish go with the flow" [93].

[88]. These systems require large antenna arrays to achieve high gains via beamforming.

The mmWave technology for V2V has seen a recent resurgence in part due to some vehicle manufactures claiming higher levels of security if CAVs transmit raw LiDAR data between each other. This is in contrast to transmitting processed metadata such as a list of detected, localised and (correctly) classified objects/people ahead of the CAV. Proponents of such 'raw' data transmission systems [23] argue that providing other CAVs with such rich (un-processed) data might be the only way to mitigate false-positive object or pedestrian detection and increase safety of CAVs. By sharing and processing raw sensor data, the 'next' CAV in a platoon can decide for themselves if the the CAV in front has actually detected someone crossing the road (for example). Furthermore, [21] adds that such methods of data sharing between CAVs travelling in a platoon could lead to a more collaborative approach to obstacle/human detection. Potentially, better detection accuracies could be achieved by cross-referencing classification/object detection results amongst the CAV platoon. Finally, the platoon of CAVs could then hold a vote on whether to conduct evasive manoeuvres.

1.4 Aims and Objectives

The over-arching aim of this thesis is to explore how modelling and simulation of the nexus between transportation and wireless systems could aid cities in terms of wireless connectivity and reducing congestion/pollution.

Research Objectives:

- Conduct experiments to collect vehicle transport and trip data in order to understand vehicle user behaviour
- Develop an energy efficient machine learning method of detecting vehicle parking activities using low-energy sensors found on a typical smartphone
- Collect or obtain vehicle fleet trace datasets for inner cities
- Develop VANET simulator using real-world vehicle trace datasets to understand urban sensor information dissemination using a simulated fleet
- Develop AGENT-based VANET simulator in order to explore various potential VANET performance improvement strategies

1.4.1 Research Questions

1. How can cities address near/short term vehicular congestion, such as those generated by inner city vehicle-parking systems in a relatively low cost and effective manner?

2. With the advent of connected and autonomous vehicles, what are the minimum requirements in terms of fleet size given the number/density of city-wide sensors to ensure optimal network coverage of the urban area?

3. What strategies can be used to increase VANET performance such as increasing or guaranteeing a certain Packet Delivery Ratio (PDR) without overly lengthening CAV passenger trip lengths?

1.4.2 Original Contributions

Contributions to Knowledge

- Developed and published fast, accurate, reliable and low-energy vehicle parking activity detection and classification system. Research was published in the Proceedings of the Twenty-Ninth AAAI Conference on Innovative Applications (2017) [12].

- Conducted and published first study into driver cruising behaviour and detection. Research was published in MobiQuitous 2017: Proceedings of the 14th EAI International Conference on Mobile and Ubiquitous Computing Systems [43].

- Filed a systems and method patent (in the USA and Japan) for an area wide wireless network managed by autonomous vehicles with wireless access points, US patent number: US20180098227A1 [11].

- Investigated relationship between passenger trip lengths and overall sensor network end-to-end delay using real world datasets. Research published in proceedings of IEEE Vehicular Networking Conference (VNC) 2018 [13].

- Showed how different vehicle-routing strategies can increase PDR at little cost in terms of extra passenger trip lengths.

Contributions to Engineering and Open Source Community

- Developed an open data centric VANET simulator [10].

- Contributed to developing (updating) OSRM-PythonV3+ API [98].

- Designed and conducted vehicle parking experiment (and data gathering exercise) with volunteer users. Consequently an Android 'smart-parking' application has been developed and released online via the Google Play Store [49].

- Developed an agent-based VANET simulator including an urban LoS model and adaptive vehicle routing techniques [9].

1.5 Systems Stakeholder Background

The research presented in this thesis was the result of an Engineering Doctorate (EngD) sponsored by Toshiba Research EU Ltd and the University of Bristol (UK). A lead academic and secondary supervisor were set by the university. The sponsoring firm, Toshiba, appointed an industrial research supervisor. Unfortunately mid-way through the EngD, the sponsoring company, Toshiba, experienced two dramatic downturns; an accounting scandal followed by a US subsidiary, Westinghouse, filing for bankruptcy. In turn these events led to a drastic change in company structure (Toshiba was even briefly de-listed from the Tokyo Stock Exchange) as well as changes in research and operational directions.

Although Toshiba's influence and business model changed during the course of the EngD, an updated stakeholder diagram highlighting the various systems and their respective owners and subsequent influence are detailed in Figure 1.1.

Toshiba's main interests now lie in becoming a 'Cyber-Physical' company, i.e., a firm that sells both software services and the necessary hardware. Toshiba still maintains a strong presence in the wireless and flash memory business, both by researching and cooperating with wireless standards and protocol development and by investing in new flash memory foundries ('3D' transistors as they are referred to). Additionally, new focus and resources have been allocated to develop more autonomous and lithium battery (powered) vehicle systems. In particular, the development of new battery technology (referred to as SCiB) increases the number of charge cycles per battery cell as well as developing wireless chipsets for vehicle manufacturers. Toshiba's 'sphere of influence' is shown by a red circle in Figure 1.1.

The main focus of the EngD was to investigate future parking and autonomous vehicle communication systems and their ability to sustain certain applications (such as messaging or data gathering services). Consequently two national government bodies, namely, the Department for Transport (DfT) and the Office of Communications (OFCOM), are highlighted as having the greatest influence with regards to setting transport policy and regulating the wireless spectrum (necessary for V2X communications). National government organisations concerned with the research carried out over the course of the EngD, i.e., departments that set policy and regulate safety laws with regards to transport and wireless communications are shown in Figure 1.1.

OFCOM sets policy covering power of cellular antennas and which parts of the electromagnetic spectrum can be used by vehicular communication systems. At the time of writing, OFCOM had not decided concretely which parts of the electromagnetic spectrum will be allocated for wireless V2X communications systems. OFCOM replied to requests from the author for clarification. However, the author was met with an inconclusive response: "..., there is an international discussion about using frequencies around 5.8–5.9GHz for Intelligent Transportation Systems (ITS) and we [OFCOM] are participating in those discussions"[5].

[5]Quote from email transcript between the author and A. Garrity, a Spectrum Licensing Associate at OFCOM (2019).

Whilst national government agencies set policy and regulations, it is often up to the local authorities, in our case, Bristol City Council (coloured blue in Figure 1.1) to pass judgements concerning building and antenna (base station) heights and positioning. Furthermore, local governments have greater authority over public transport and land use permits. This is of great concern with regards to inner-city vehicle parking locations (and pricing schemes) as well as locations (and power output) of electric vehicle charging points and permits for AV testing.

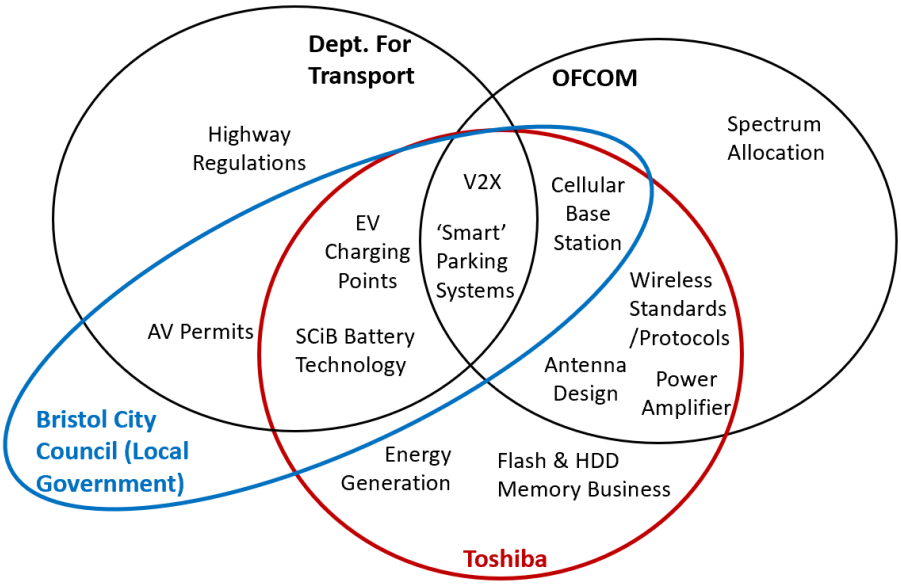

FIGURE 1.1: Stakeholder diagram highlighting areas of influence with respect to autonomous vehicles, wireless systems and inner-city parking. National level UK government and respective agencies (departments) are shown with black circles of influence. Blue and red circles refer to local city government (Bristol City Council) and the firm supporting the EngD in Systems, namely Toshiba Research EU Ltd.

.

1.6 Thesis Structure

The remainder of this thesis is divided into five main chapters. Note there is no overarching thesis literature review, instead each 'research work' chapter contains a concise literature review relevant to the material being discussed.

Chapter 2 aims to address our first research question by detailing our work on near-term solutions to current congestion problems facing inner cities, namely congestion due to vehicles searching for a vacant on-street parking space. We design an experiment whereby we collect driver transit and parking behaviour data. The chapter investigates various machine learning approaches in order to accurately (and reliably) classify when a user has occupied or vacated an on-street vehicle-parking space. We achieve this by using mostly low-powered sensors found in almost all smart-phones. We compare our classification system

against current approaches and cost our method in terms of energy consumption per vehicle-parking detection using an energy model. Finally, we investigate whether we can detect changes in driver behaviour as they switch from simply driving towards their destination to searching for a vacant vehicle-parking space.

Chapters 3 and 4 are similar in scope but differ in terms of the underlying modelling and simulation methodology. Both chapters aim to address the research problem of combining a public-private transportation system with wireless sensor connectivity for an urban area. Chapter 3 aims to address our second research question concerning optimal taxi-fleet sizes for certain city-wide sensor network scenarios. Note, all work detailed in Chapter 3 uses only openly available (i.e., 'real-world') datasets such as OpenStreetMap [65] for our building footprint (to construct our urban LoS model) and road network model and CRAWDAD vehicle trace data for two cities Rome (Italy) [8] and San Francisco (USA) [71]. We developed a method of extracting, filtering and 'folding' these taxi-trace datasets for our simulations.

Chapter 4 builds on some of the results and observations of Chapter 3 as well as addressing some of the limitations of using vehicle trace datasets: namely the inability to set vehicle routes and vary passenger demand. Consequently, an agent-based VANET simulation was developed to allow for finer control of model parameters in order to properly address our final research question. Furthermore, we introduce an edge-weight altering model parameter, a simplified LoS model (for computational efficiency) and investigate certain behaviours such as how the Largest-Connected-Component (LCC) of the VANET varies over time. We show that PDR can be increased without significantly increasing passenger trip lengths by carefully selecting the value of our edge-weight parameter.

Chapter 5 both concludes and discusses at greater length some of the findings presented in the thesis. Some ideas and general directions for future work are suggested.

Chapter 2

Low-Energy Vehicle Parking Detection Systems

This chapter is based on research published in two conference proceedings. The first paper, presented at the Twenty-Ninth Innovative Applications of Artificial Intelligence conference (IAAI, San Francisco (USA), 2017), detailed our ParkUs system design and our low-energy vehicle-parking detection methodology [12]. The second paper, presented at the Fourteenth International Conference on Mobile and Ubiquitous Systems (also known as Mobiquitous held in Melbourne, Australia, 2017) introduced and evaluated a novel method for detecting changes in driver behaviour [43].

2.1 Introduction

Analysis conducted by the RAC Foundation in 2012 concluded that vehicles (in Great Britain) spend on average 96% of their time parked [39]. Furthermore, the total number of licensed vehicles in Great Britain has increased in all but one year (1992) since 1945. Since 2012, the average growth of licensed vehicles has been roughly 610,000 per year, but this growth has slowed in recent years with the reduction of new vehicle registrations [96]. As of September 2019, there were 38.9M licensed vehicles in Great Britain. Consequently, with the predicted increase in urbanisation and vehicle ownership over the coming years, it is likely congestion and competition for parking spaces within cities will continue to increase.

Through surveying drivers in Westwood Village, Los Angeles (USA), Shoup estimated the average time spent searching for a vacant (on-street) vehicle-parking space (also known as 'cruising') to be just over 3 minutes per journey. This may not seem much, but over a year this roughly equates to an excess of 1.5M vehicle driven kilometres, generating approximately 730 metric tons of greenhouse gas emissions [83]. This is somewhat alarming, given that Westwood Village is a relatively small suburb of Los Angeles with only 470 metered on-street parking spaces.

While there are hardware-based solutions to monitor on-street parking occupancy in real-time, instrumenting and maintaining such a city-wide system can be a substantial investment. For example, in the city of San Francisco, USA, a connected parking occupancy sensor and payment meter were installed

at circa 7,000 on-street vehicle-parking spaces in 2011. The total cost of the deployed system (at the time) was roughly 23M US dollars [79]. Consequently, research effort has looked at methods of developing cost-effective parking occupancy monitoring solutions. Ideally such systems incorporate (digital) payment methods and are relatively inexpensive to install, require minimal maintenance, and aid drivers in finding vacant spaces quickly and effectively, without causing undue congestion in the area.

In this chapter, a novel low-energy consumption vehicle-parking activity detection method, referred to as ParkUs, is introduced and tested (through extensive simulation) with the aim to eventually reduce vacant vehicle-parking space search times. The system uses low-energy sensors, such as the accelerometer and magnetometer found in most smart-phones, in order to detect parking activity within a city environment. Moreover, we developed a novel sensor fusion feature called the Orthogonality Error Estimate (OEE) to aid our detection method. We show that the OEE feature is capable of detecting parking activities with high accuracy and low energy consumption. One of the future envisioned applications of the ParkUs system will be to provide drivers with (real-time) guidance on where they are most likely to find vacant parking spaces within a city.

The rest of the chapter is structured as follows, Section 2.2 covers more of the background related work and competitor solutions to detecting vehicle-parking activities using a smart-phone. In the methodology (Section 2.3) we describe in detail our system design, overall architecture and our data collection methodology for the training of our machine learning system. We discuss our results (Section 2.4) and detection accuracies, and benchmark them against our competitors using an energy consumption model. Finally, Section 2.6 concludes the chapter by reviewing our proposed system and extended work (on driver behaviour detection). We also briefly discuss developments in the field since publication of the two papers this chapter is based on.

2.2 Related Work

As cities continue to rise in population [97], demand for basic services such as transportation consequently increases. The automobile has been the dominating method of inner city transportation for many cities across the globe, resulting in increased congestion and air pollution. As the demand for transportation rises, so do the number of vehicles which has a knock-on effect on availability of vehicle-parking spaces. Part of the problem lies in how cities manage their public on-street vehicle-parking spaces and road network infrastructure [20].

For example, a study conducted in the Washington DC region (USA), concluded that cost-free parking (either at work or at home) was associated with a greater than 95% chance that the individual will drive to work alone [1]. Such parking and driving incentives are often not directly implemented by local or national government policy. Rather, they are the consequence of decades old building legislation often known as 'parking minimums' that designate a minimum number of vehicle-parking spaces per residential building or land-use type. For example adult bookstores in Clark County, Nevada (USA), are required to

have at least three vehicle-parking spaces per 93 square metres (one thousand square feet) of retail floor space [14]. Similarly Cupertino (a suburban city near to San Francisco, California (USA)) has a requirement for any new-build housing developments to have at least two vehicle-parking spaces per apartment.

Consequently, many North American cities were built around the automobile as their main form of transport. Vehicle ownership has thus become almost ingrained in their society. This is understandable to an extent, since having minimum parking requirements in cities forces developers either to allocate space beneath a building (although underground parking is prohibitively expensive) or find less costly plots of land nearby and allocate them as (private) surface parking. By having buildings more spread out, the average distance people have to either commute or run errands increases, thus, further incentivising vehicle ownership and use. As city population and area increase, more vehicles are needed, causing ever more congestion [20].

Congestion and the lack of vehicle-parking availability has various knock-on effects such as missed hospital appointments [59]. In this particular example, a sample of 72 UK families were randomly surveyed to identify main reasons for missing hospital appointments. It was found that 32% of patients had missed appointments due to inadequate vehicle-parking facilities near to the hospital. It is worth noting that missed appointments cost the UK NHS (National Health Service) circa £790M annually.

Partially to better understand causes of vehicle congestion, various studies (reviewed in [83]) conducted over the last century covering, cities in North America such as New York, Detroit, Los Angeles, as well as a couple of European cities such as Freiburg and Barcelona, conclude that the average proportion of inner city traffic searching for a vacant parking space is approximately 30% [83].

Shoup and other researchers set about finding the optimal price for parking to both reduce congestion and pollution in cities, by minimising the time spent by drivers circling blocks in search for on-street parking spaces [82]. A parking price per hour that results in 60–80% level of occupancy (averaged over the day) was deemed ideal, since there will always be some space for the next driver to park (thus cutting down on search times), whilst not being overly expensive such that the public resource is underused and not generating valuable revenue.

The city of San Francisco (USA), implemented an on-street parking scheme designed to achieve said optimal level of occupancy averaged across the day through the use of variable pricing. The system, called SFpark, costed over $23M US dollars (in 2011) to implement and covers 7,000 on-street parking spaces and 14 publically owned and managed (off-street) parking garages. Recently, the scheme was evaluated by the San Francisco Municipal Transportation Agency (SFMTA) and in pilot areas the target parking occupancy of 60–80% was met 31% more often than experimental control areas (no changes in price based on driver demand) [79], The experimental control area vehicle-parking spaces were full 1.5 times more often than SFpark pilot areas. Furthermore, they estimate a 30% decrease in vehicle miles driven in SFpark system pilot areas, equating to a reduction of 2.1 metric tons of daily greenhouse gas emissions (from 7.0 tons before the introduction of SFpark pilot to 4.9 tons) [79].

Although relatively successful at reducing congestion and emissions, SFpark

relied on a massive initial investment. Further costs include ongoing running and maintenance costs to operate successfully. Furthermore, SFpark only covers a small percentage of the total stock of on-street parking in the urban area. As attractive as the SFpark solution may appear, given how stretched the budgets of most municipalities are, such solutions could struggle to see an uptake. There is no doubt that solving the parking, congestion and air pollution problems are high priority items on the agenda of most cities. However, many systems rely on physical infrastructure being installed; such as smart parking meters (SFpark), Closed Circuit Television Cameras (CCTV) overlooking parking bays [22], or ultrasonic distance measuring transducers mounted on moving vehicles [54]. Whilst successful in some cases, the hassle and initial cost of physical infrastructure can be prohibitive in nature.

Deloitte, a consultancy, estimates that smart-phones have approximately 80% penetration across all countries surveyed [52]. Thus, encouraged by the high penetration rate of smart-phones globally, and wireless internet access availability in cities, various smart-phone based vehicle-parking solutions have been researched and developed.

Most smart-phones have a combination of sensors and radio interfaces such as a Global Navigation Satellite System (GNSS), as well as accelerometers, gyroscopes and magnetometers. Consequently, systems such as PhonePark [85] and Park Here! [78] attempt to leverage smart-phone users and crowdsource parking data to aid drivers with finding parking spaces efficiently. PhonePark and Park Here! use GNSS, accelerometers, magnetometers and Bluetooth connections to detect changes in user transit behaviour, i.e., when they switch from driving to walking or vice versa.

Instead, ParkSense [62] relies on constructing Wi-Fi access point ID maps in order to both infer velocity of a user (as they drive or walk past) as well parking location. Although all the above cited systems were able to achieve reasonably high vehicle-parking detection accuracies (approximately 80% true positive rate), they were not optimised for energy efficiency and often relied on some form of user tagging and large data collection windows. Furthermore, one could question the ethics of collecting private Wi-Fi access point information on a large scale, without consent or knowledge from/of their respective owners [45].

The PhonePark system [85] was to our knowledge the earliest attempt to detect driver parking activities with a smart-phone. The PhonePark system was able to distinguish between user transport modes, i.e., when the user was driving or walking. Their detection system combined GNSS, accelerometer and Bluetooth connectivity data collected from a user's smart-phone whilst in transit. PhonePark achieved a relatively high reported true positive detection accuracy of 80% and 85% for vehicle parking and un-parking activities respectively.

In an attempt to reduce energy consumption per parking detection, the ParkSense system [62] avoided polling the (energy expensive) GNSS radio altogether whilst detecting user parking. ParkSense [62] opted to use Wi-Fi radio modules in smart-phones to estimate the user's velocity. Their method relied on mapping Wi-Fi access point IDs to a geographic location. By using the change in Wi-Fi acess-point IDs as the user drove past the ParkSense system was able to assess velocity changes as the user was travelling. Consequently, they were

able to detect an un-parking event, and record this for other users looking for parking nearby.

ParkSense was able to achieve an 83% true positive detection rate on user un-parking activities. Whilst relatively energy efficient (essentially using only one radio module to do their un-parking activity inference), ParkSense was not able to run in the background on a smart-phone[1] as it relied on users to manually geotag where they had parked their vehicles. Furthermore, ParkSense required a large data collection window (60s) and took almost five minutes to process and confirm an un-parking event.

In a similar vein, Park Here! [78] made use of multiple wireless radio technologies (including Bluetooth) as well as the accelerometer and gyroscopes for their parking and un-parking detection system. However, Park Here!'s classification was strictly binary (as opposed to the stationary, walking and driving classes used by [62, 85]). Park Here! has a claimed 100% true positive rate for parking and un-parking detection when a user had Bluetooth enabled on their smart-phones. Note, that Park Here! relied heavily on Bluetooth as well as conducting fewer trials (see Table 2.4 on page 37 for more comparison details). Whilst many modern vehicles are equipped with Bluetooth systems, relying so heavily on one sensor for accurate detection isn't robust, since it does not allow for truly ubiquitous detection (as some vehicles may not have such Bluetooth-enabled systems or their owners may prefer to turn these off whilst driving for privacy reasons).

In summary, multiple attempts to research and develop parking activity detection systems, with similar aims of reducing unnecessary congestion (vehicles circling blocks/streets) and pollution within inner city areas have been conducted before. However, most of the research effort has focused on processing data streams from 'power-hungry' (i.e., electrical energy consuming) sensors such as wireless data transmission radio modules (Cellular, Wi-Fi, Bluetooth) and GNSS typically found in smart-phones. Thus, our aim is to improve on these vehicle-parking activity detection systems in terms of both detection accuracy and energy consumption. Finally, we aim to extend our proposition by investigating if driver 'cruising' behaviour, prior to parking, can be detected.

2.3 Parking Activity Detection Methodology

Systems such as PhonePark and Park Here! ([78, 85] respectively) relied on (high) energy consuming radio modules such as GNSS, Wi-Fi, cellular and Bluetooth to detect parking activities. In contrast, our proposed ParkUs system aims to achieve similar (if not higher) detection accuracy rates without the use of these radio modules, to avoid unnecessary smart-phone battery depletion.

Given the inherent complexities of devising a set of filters that work with any (un-calibrated) smart-phone sensor, a machine learning method was considered appropriate. Consequently we developed a machine learning method to select optimal statistical features to process and filter the noisy sensor data

[1] Or as an Android Service, if one is following the current Alphabet's Android programming guidelines [38].

streams. A human could devise a set of rules, filters, thresholds etc., for vehicle-parking activity detection. However, devising and calibrating them for each smart-phone collection of sensors (of which there are hundreds of different manufacturers/chipsets) would be too time consuming.

Consequently, volunteers were recruited in order to collect and annotate journeys and in particular vehicle parking behaviour/activity data, to train our (supervised) machine learning parking activity-detection system.

2.3.1 Training Data Collection

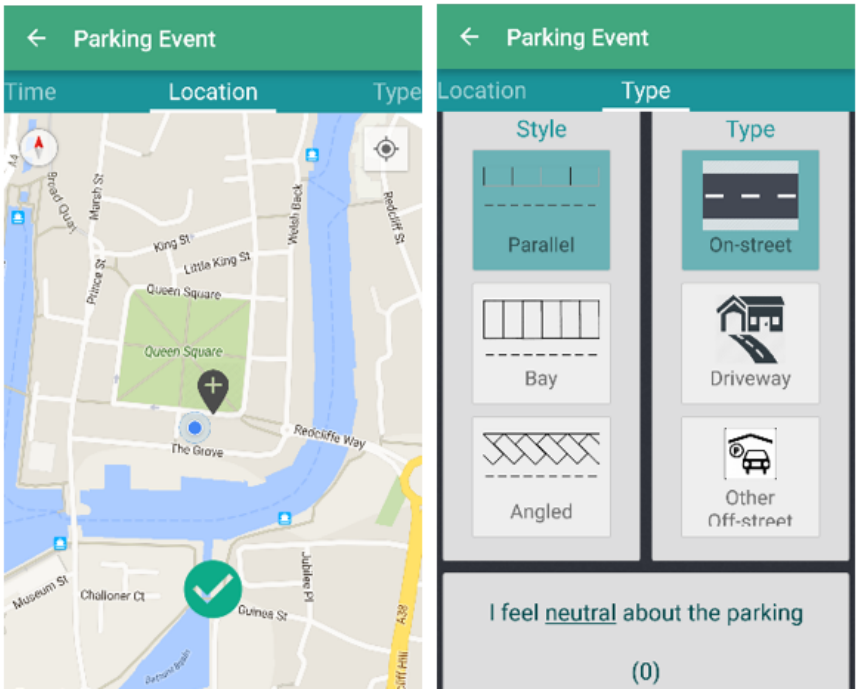

FIGURE 2.1: Screenshots from our Android data collection software application. On the left user-reported vehicle-parking/un-parking location tag and on the right user-reported vehicle-parking conditions. Figure reproduced from [12].

The ParkUs system aims to run locally on a user's smart-phone by collecting sensor data and processing this (in the background) to monitor changes in transport mode (e.g., either walking or driving in a vehicle). User recorded and annotated journeys were necessary for supervised machine learning.

Whilst there have been many advances recently in the training methods, as well as hardware optimised for unsupervised machine learning (often referred to as 'deep' neural networks) [48], this was not our chosen method for the following reasons. Deep neural networks rely on vast amounts of data to train, often in the orders of hundreds of GB's if not more. Many researchers aim to get around

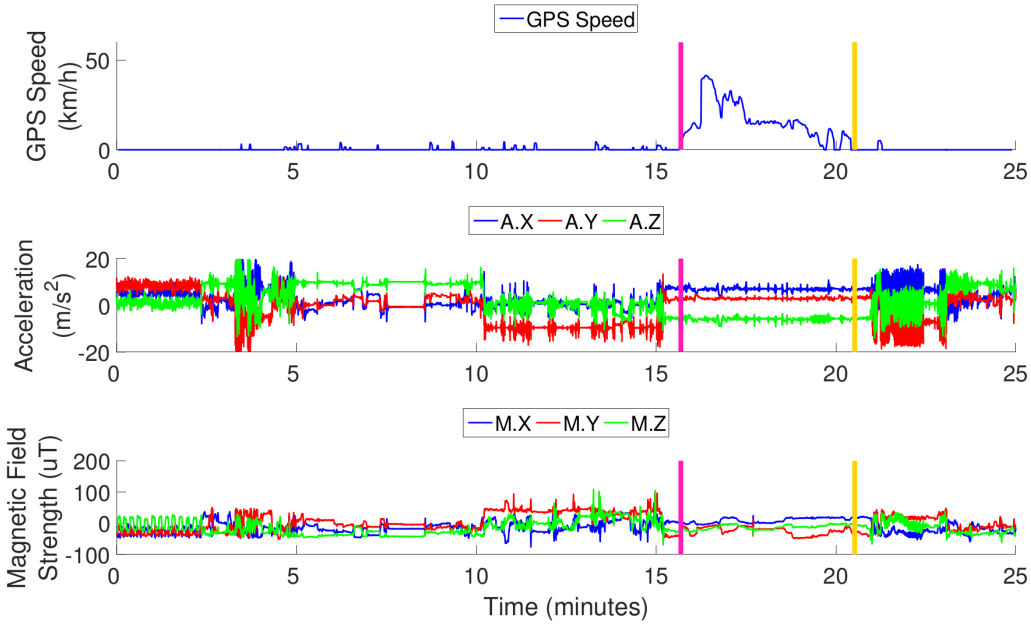

FIGURE 2.2: Example of data collected from a typical user recorded journey. Note, purple and yellow vertical lines are user reported instances of un-parking and parking respectively. Figure reproduced from [12].

this problem by developing highly sophisticated cyber-physical environments for their systems to train within[2].

However parking systems simply can't be 'gamed' or simulated in such a manner. Furthermore, the collection/recording of millions of vehicle-parking instances required for deep neural network training, simply wasn't feasible given budget and timescale limitations. Therefore, our approach to train through a supervised method is ideal for scenarios where data is limited and cannot easily be simulated in a game-like environment.

In order to collect user travel data, a custom data logging and user tagging smart-phone application was developed[3] for our volunteers; see Figure 2.1 for a couple of selected in-app screenshots. The Android operating system was chosen due to the simplicity of developing and distributing applications without the need for consent. Furthermore, the majority of smart-phone users (some estimates put this at over 70% [52]) use the Android operating system, therefore making it easier to find 'compatible' volunteers.

[2]This was the approach used by Deepmind (a subsidiary of Alaphabet Inc) to beat human players at a board game called 'Go' and a variety of console games [57]. Deep neural network architectures can learn by playing millions (if not more) games either against themselves (or previous model iterations) or human players on-line. Deepmind opted for the former as it allowed for better secrecy as well as being able to train seamlessly at any hour (for days at a time). Again these systems take a lot of time to train as well as requiring a very structured environment within which the neural network trains/learns. A board or computer/console game with well-defined boundary conditions, limits and rules is ideal for these deep learning methods since there is a relatively well defined scoring system (crucial for reinforcement learning as well as ranking different machine learnt models) and objectives to achieve.

[3]Note, the Android data collection application was developed by a small team of Toshiba software engineers in collaboration with the author.

We advertised within our research office and on the University of the West of England campus for volunteers to record their vehicle journeys and parking activities. Seven volunteers (all located in Bristol (UK) were given access to our training data collection application and asked to log their journeys as well as vehicle parking/un-parking activities over the course of a month. Volunteers were instructed to open the data collection application before starting a journey and to manually record/geotag their un-parking/parking location.

The volunteers were instructed (during the induction session) to use their smart-phones as they would normally (i.e., no need to place it anywhere in particular either when walking, or driving the vehicle). Once a volunteer had parked safely, they were asked to check their vehicle-parking location and add any other information that they felt pertinent regards to their parking activity, see Figure 2.1 for more details.

The extra information regarding parking activity was collected in order to (briefly) investigate the driver's skill at parking. For example, parallel parking is often thought of as a more difficult manoeuvre than angled or bay parking. Similarly, drivers of larger (SUV-esque) vehicles may avoid smaller or harder to park spaces. Therefore, vehicle-parking space recommendations might differ depending on driver/vehicle combinations and or preferences.

Information regarding where exactly the parking activity occurred, such as in a private driveway or on-street, was considered useful for later ParkUs implementations which might broadcast vehicle-parking availability information to other users[4]. Information about the volunteer's opinion of their parking was initially considered a useful proxy for stress/anxiety whilst carrying out the parking activity. Note, that whilst all these extra options were available for volunteers to record information regarding parking activities, not all volunteers were able to do this: many cited lack of time/"being in a hurry" as their major barrier to fully recording/annotating their parking activity/experience.

Sensor data collected on volunteer's smart-phones, along with their respective annotations, were stored locally before being opportunistically transmitted over a Wi-Fi connection (in order to avoid excess cellular network charges[5]) to our IES cities server platform (originally developed by [53] and modified for our application) located on Toshiba Research Europe Ltd premises.

Table 2.1 (page 21) shows the relevant sensor sampling rates of our parking activity data collection and logging application. In an ideal laboratory experiment, the highest possible data rates would have been selected to generate training data for our supervised machine learning system, since this would have captured all possible signals in the sensor data. However, sampling smart-phone sensors at high data rates and storing (i.e., writing the data to memory) is both battery/energy consuming as well as being memory-expensive (note, only half of our volunteer's smart-phones had expandable non-volatile memory). Therefore, sensor data polling rates were reduced to minimise energy and memory usage/footprint whilst ensuring enough data was captured in order to train our supervised machine learning system.

[4]As it would not be particularly helpful or wise to advise future users to park in privately owned parking spaces or driveways.

[5]In particular after a volunteer complained, they were eventually reimbursed.

TABLE 2.1: Android data collection application sensor sampling rates.

Sensor	Sampling rates/[Hz]
Accelerometer	25
Magnetometer	5
Location (GNSS)	1
Ambient Noise (loudness/amplitude only)	10
Ambient Light (intensity)	5

For example, [3] showed that the majority of human physical motions occur below 10Hz, consequently (by virtue of the Nyquist-Shannon criterion) the accelerometer was sampled at 25Hz to ensure as full data capture as possible/within reason (by allowing for a small safety margin).

Other sensors such as GNSS were sampled at far lower rates (1Hz) given the sheer amount of energy and processing required for each reading. The magnetometer, ambient noise and light sensors were also sampled should they prove useful for later parking detection algorithms. It was originally hypothesised that ambient noise (i.e., amplitude/noise level) as well as light sensors might change if users moved from inside a vehicle to outside (and vice-versa).

Over the course of our data collection month, our kind and patient volunteers recorded and annotated over 60 journeys. The mean journey duration was 43 minutes and our volunteers recorded 52 and 57 parking and un-parking events respectively. Note the minor disparity was mostly due to volunteers forgetting to log their parking activities. There were a couple of reported in-app failures/bugs which briefly hindered data collection, however, they were quickly remedied by the author and Toshiba's in-house software development engineers.

A typical volunteer-recorded and annotated training journey is shown in Figure 2.2. Readings of three sensors (in descending order of depiction within the Figure): GNSS, accerlormeter and Magnetic Field Strength were plotted (with respect to time) along with volunteer/user annotations: purple and yellow vertical lines for vehicle un-parking and parking events respectively.

Note that our training data collection application did not explicitly poll the GNSS sensor asking for velocity measurements (a feature now available in most versions of the Android operating system), as this was considered too battery-intensive, and unnecessary, as the user speed could be estimated (off-line) by using the distance covered and the time taken between the time-stamped GNSS coordinates.

2.3.2 ParkUs: Design and Algorithms

The envisaged ParkUs system design is shown in Figure 2.3 and the parking detection system is further detailed in Figure 2.4 with its four key components (or sub-systems), namely: an initial median data filter, a feature processor, a modality detector (Figure 2.5), a Finite State Machine (FSM, Figure 2.6).

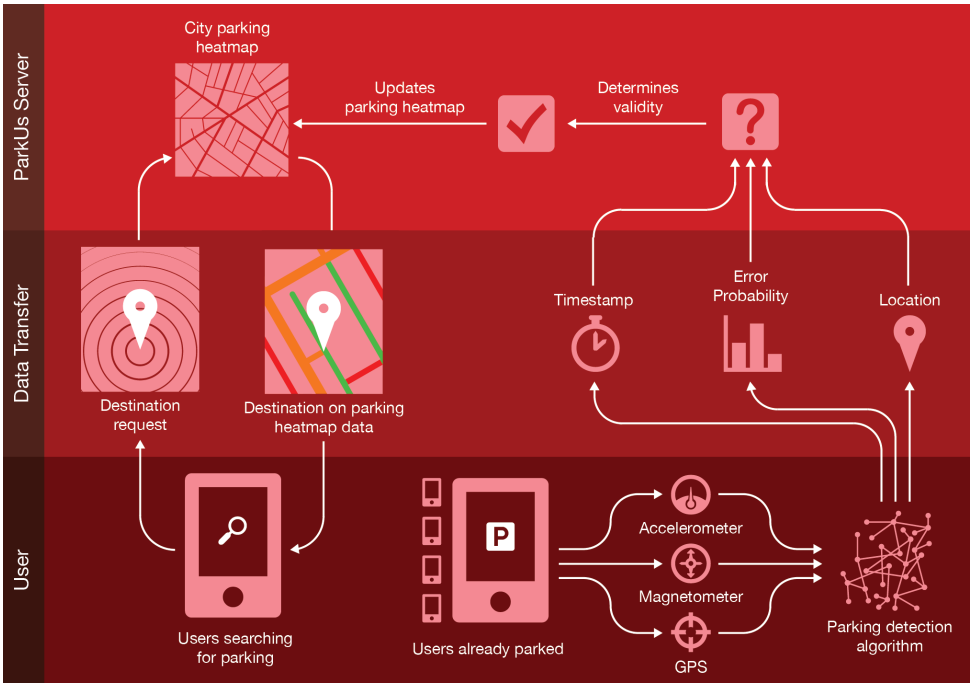

FIGURE 2.3: Overview of the envisaged ParkUs system architecture. Starting from the bottom right, a user's smart-phone's sensors are used to detect parking activity. Once detected, location, timestamp and a confidence value are passed on to our ParkUs server. Data from various parking activities are collated and displayed as a heat-map overlaid on the surrounding road network. Future users searching for vehicle-parking can query our server from their smart-phone application to view latest parking availability near their desired destination. Figure reproduced from [12].

Figure 2.4 details how data is processed locally on the ParkUs user's smartphone. Starting from the top right (of the flow chart), data collected by sensors was combined into a data frame or sample window. Note, future (incoming) data frames are shown on the right, data frames (or data samples that have already been processed are drawn on the left, therefore data in the current time window is shown centrally). Each data sample is processed first with a median filter (where extreme outliers are replaced with median values from the sample window), before each sample window was further processed using our selected range of statistical features. Once our features have been processed, they were passed through our modality detector (see Figure 2.5 for more details) which attempted to assign a transport mode such as: *stationary*, *walking*, *vehicle stationary* (for example when a vehicle is in a queue at a traffic light) and *vehicle moving* to each sample window (data frame). Details of past and current activity classifications were subsequently passed on to our finite state machine model (Figure 2.6) which attempted to understand the relationships between the classified transport modes and conclude whether a parking activity has been detected.

2.3.3 Data Pre-Processing

Due to the openness and resultant fragmentation of the Android operating system platform, no sensor calibration was conducted. To counter potentially low quality sensors, a median filter and hard thresholds were used to remove any outlier data. A sliding time window of 27.5s was carefully chosen to ensure that finer changes in activities were captured and detected without excess processing and memory storage on the smart-phone.

Labelling activity data by human volunteers 'on-the-fly' (i.e., during the experiment/data collection phase) was always going to be error prone. For example, a minority of users forgot to create a parking label after they had parked and left their vehicles. Since the data collection period was over a month, it was understandable that such rushed occurrences happened.

Nonetheless, we were able to sufficiently limit the effect of timing errors (where a user might record/label a parking activity hours after it had actually occurred) by the following strategy. Firstly, by referring to the location and inferred speed data (i.e., 'raw' time-stamped GNSS data), we were able to shift unrealistic parking labels to the nearest stationary point. For example, there were a couple of instances where volunteers had labelled a parking activity when the GPS sensor was indicating that they were travelling over 20km/h. In these rare cases, we shifted their parking tag to the next stationary point in their journey. Secondly, after assessing the data, it was deemed reasonable to have a 15 minute threshold between a volunteer actually parking and recording/labelling the event. Overall, less than five parking events did not meet this threshold and were removed from the dataset.

Note, machine learning systems are somewhat limited by their designers and the data they train on. Given the importance of the latter on the overall performance of the system, it is somewhat curious that label correction algorithms or systems are rarely discussed in the relevant literature. With ever larger datasets

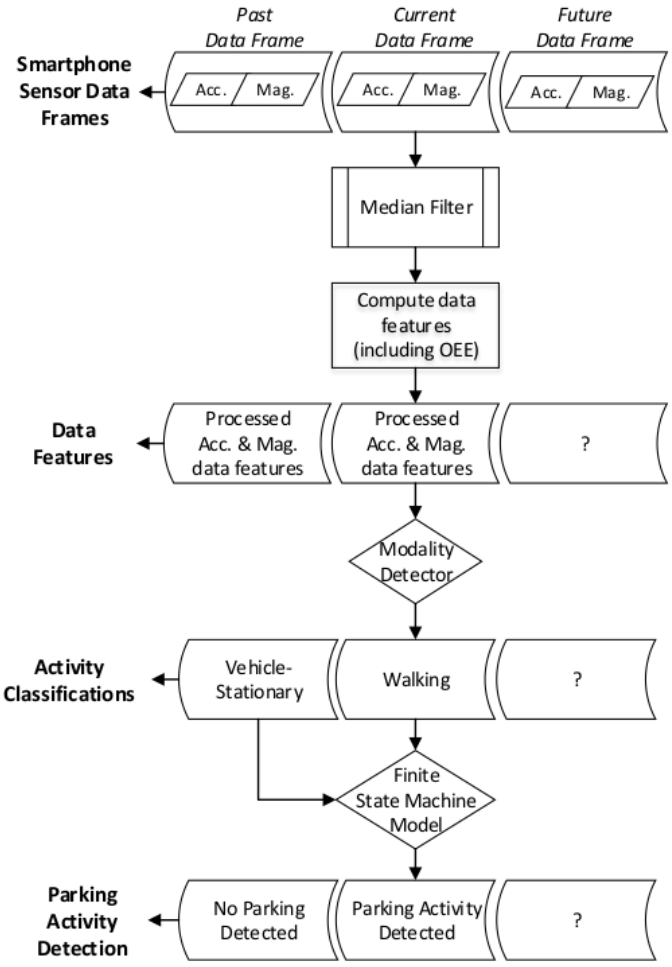

FIGURE 2.4: Our in-App ParkUs activity detection architecture. Figure reproduced from [12].

being collected and 'fed' to machine learning systems, the need to check and correct human recorded/labelled events will become ever more pertinent.

2.3.4 Feature Extraction

Previous research [35] has shown that improved detection accuracy rates between stationary, walking and within-vehicle transport modes can be achieved by estimating the gravity vector to generate an accurate decomposition of the tri-axial accelerometer and magnetometer data. Using their algorithm, the gravity and the North vectors were estimated from the accelerometer and magnetometer data respectively. The algorithm [35] opportunistically selected moments (or time windows) where the accelerometer readings were reasonably stable (i.e., low variance within the data sample window) to estimate the gravity vector. The variance threshold was increased gradually to obtain the optimal number of stable moments within the accelerometer data. The North vector was estimated using the same algorithm for the magnetometer data, since when the

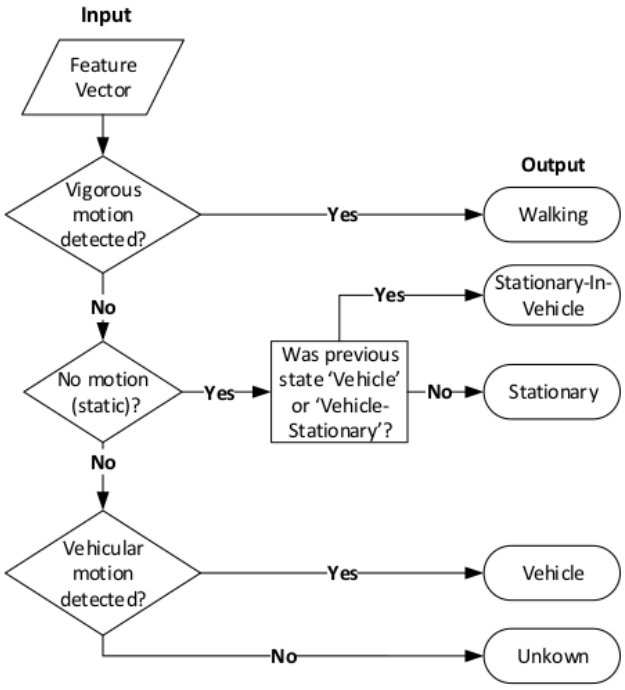

FIGURE 2.5: Our ParkUs cascaded transport modality classifier. Figure reproduced from [12].

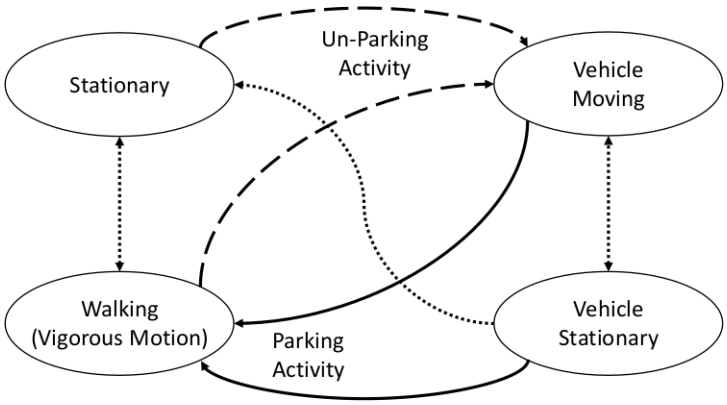

FIGURE 2.6: ParkUs Finite State Machine (FSM) model. Solid and dashed lines are parking and unparking activity (state) changes respectively. Figure reproduced from [12].

smart-phone is resting, it is always supported by some upward force (that opposes gravity). The constant force forms the vertical vector referred to in this section as G. In contrast, the magnetometer tends to point towards the Earth's magnetic North pole, and is estimated as vector N. Although heavily skewed by the Earth's magnetic field, the horizontal North vector, N_f, can be compensated by subtracting N's projection onto G from itself. Using these two vectors with respect to the smart-phone, any acceleration data samples (or recordings) can be rotated (see Figure 2.10) and decomposed into (effectively their) vertical

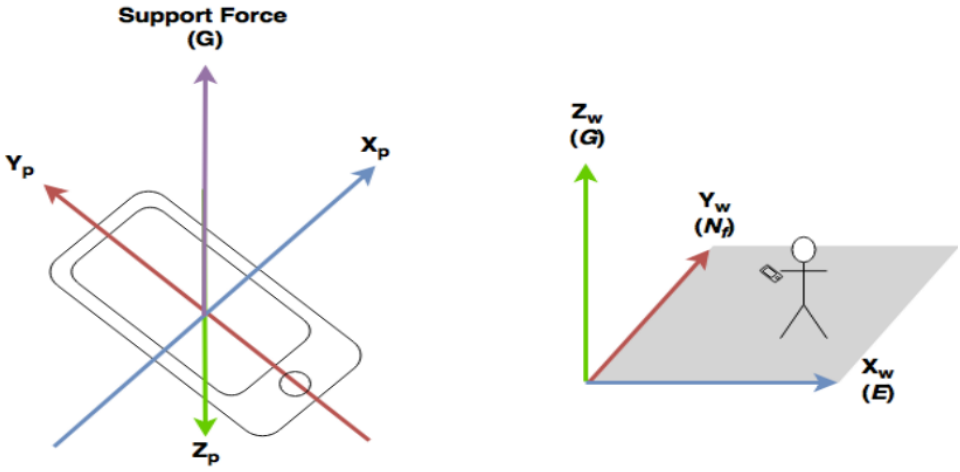

FIGURE 2.7: Schematic highlighting (right) world: X_W, Y_W and Z_W and (left) phone axes: X_P, Y_P and Z_P. For further reference: the upwards, smart-phone supporting force, magnetic North vector and East vectors are designated by G, N_f and E respectively.

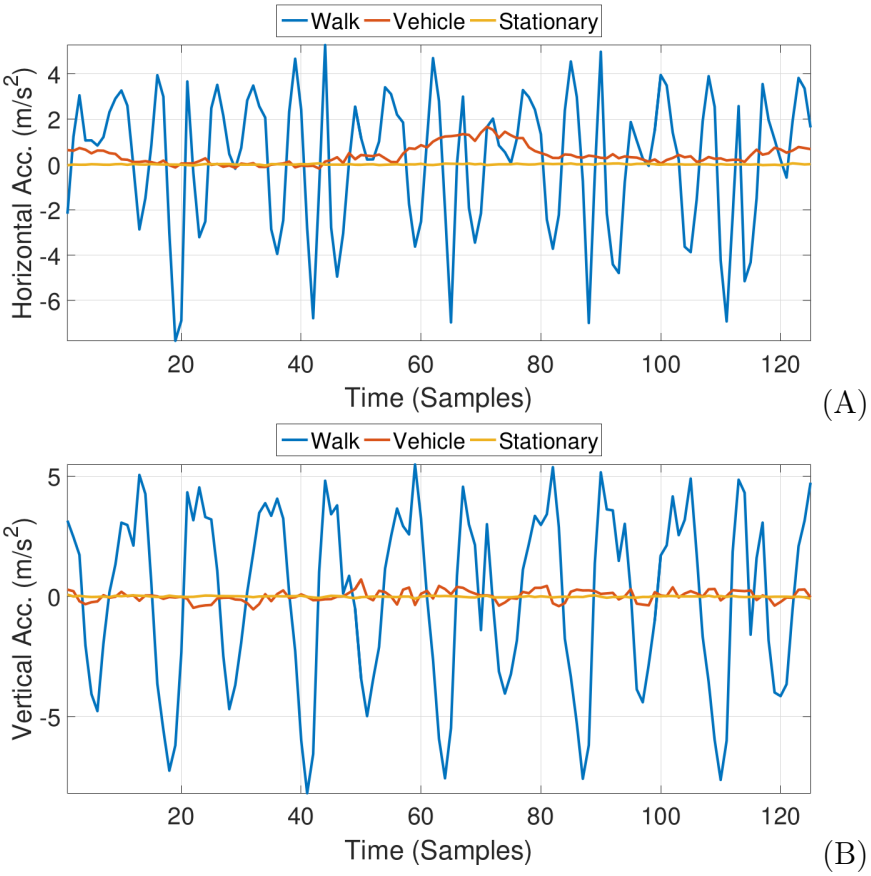

FIGURE 2.8: Typical recorded horizontal (A) and vertical (B) acceleration components for each motion class. Figure reproduced from [12].

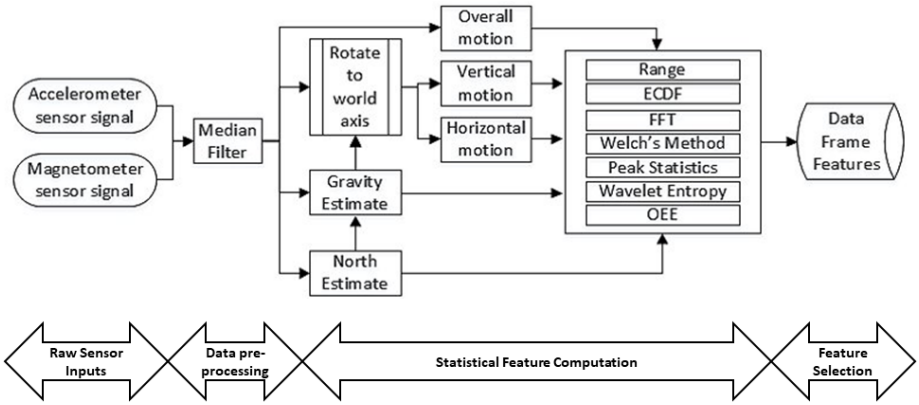

FIGURE 2.9: Flow chart showing our process of feature extraction for the ParkUs detection system. Raw smart-phone sensor data entered the system from the left-hand side of the diagram and numerous statistical features were computed before the top features, i.e., a subset of the most representative (in terms of parking activity detection) features were kept and used for parking detection. Note, ECDF, FFT and OEE stand for Empirical Cumulative Distribution Function, Fast Fourier Transform and Orthogonality Estimation Error respectively.

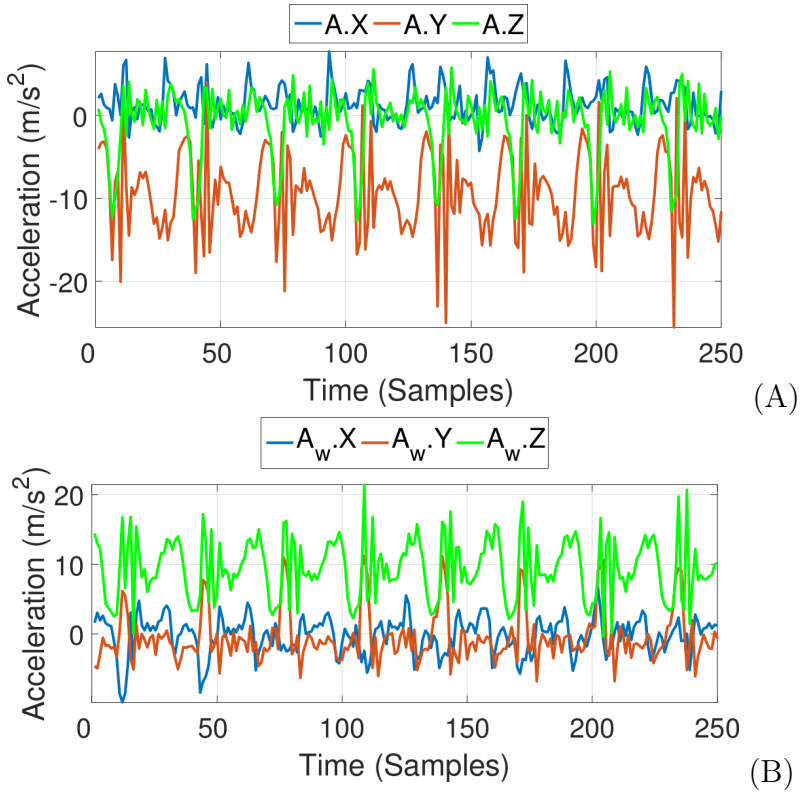

FIGURE 2.10: Rotation of walking motion accelerations from smartphone (A) to world axes (B). Figure reproduced from [12].

and horizontal components [95], shown in Figure 2.8. Once the magnetometer and accelerometer data were filtered, transformed and decomposed, we derived for each data sample window (or data frame in Figure 2.4) gravity eliminated acceleration, and estimations of G and N.

2.3.5 Feature Overview

TABLE 2.2: Most indicative features chosen by the Random Forest algorithm for each motion class.

Motion Class	Sensor Signal	Feature
Walking	Overall acc.	Median abs. change Majority range Wavelet entropy
	Vertical acc.	Peak power ECDF 9^{th} bin
Stationary	Overall acc.	Median abs. change Majority range
	Vertical acc.	Mean ECDF 5^{th} bin ECDF 9^{th} bin
In-Vehicle	G and N estimates	OEE
	Overall acc.	Peak power frequency First peak Second peak
	Horizontal acc.	FFT energy

Over 300 statistical features were initially considered in the development of the ParkUs detection algorithm[6], mostly inspired by previous research in human activity detection. The features fall roughly into the following main feature families.

Statistical Features

Standard statistical metrics such as minimum, maximum, mean, median, interquartile range, variance and range were used to analyse and capture changes in the dataset. In addition, the distribution of signal values (within each sample data window) was represented by its empirical cumulative distribution function (ECDF), with a resolution of 10 bins, similar to the strategy proposed by [34].

[6]Including a particular feature referred to as Majority Range, which is evaluated as follows. A window or data-frame of the accelerometer signal is further sub-divided into smaller windows/sub-frames (of uniform length) and the range of each sub-frame is evaluated. Therefore, the majority range is then the median range of the sub-frames. This statistical feature was found to reduce false variances caused by changes in the smart-phone's orientation, a common problem when handheld.

Discrete Fourier Transform (DFT)

In the frequency domain, the 1-3Hz DFT coefficients were shown to be good indicators of cyclical walking motion [101]. Peak frequencies were also identified as features in addition to the peak coefficient (partly inspired by 'peak frequency power' in [91]). Similar statistics from Welch's power spectrum were also included in our feature set.

Peak Statistics

A 'peak' in this context refers to a period of sustained acceleration in one direction (for example see the horizontal peak in Figure 2.8). Inspired by [41], peak characteristics were captured using area under curve, cumulative sum, kurtosis, skewness, duration between peaks, and duration of each peak.

Wavelet Entropy

The information contained in the wavelet coefficients helps to analyse transient features and non-stationary signals. A chaotic signal contains more information than one that does not vary significantly [50].

Orthogonality Estimation Error (OEE)

A novel feature representation was developed for the ParkUs system which works by estimating the error between the gravity and North vectors. The OEE was a good indicator of vehicular motion, as it had a Pearson correlation coefficient of -0.53 with the user's travelling speed. This level of correlation is similar to the Jaccard index used by ParkSense [62]. The OEE is calculated as follows. For an arbitrary data sample window, where G and N are approximations of the vertical 'up' vector (opposing gravity) and the magnetic North vector respectively, the angle, θ_{GN}, between the two vectors is given by

$$\theta_{GN} = \cos^{-1}\left(\frac{G \cdot N}{|G||N|}\right), \tag{2.1}$$

and thus

$$\text{OEE} = |\theta_{GN} - \pi/4|. \tag{2.2}$$

Ideally, G always points vertically up and N always points to true North without deviation. The angle between them should therefore be perpendicular. Realistically, G and N are almost never orthogonal due to the shape of the Earth's magnetic field. There is a natural declination of N in most areas on Earth except those very near to the equator. Since the study was conducted in the UK, the inclination effect was very pronounced (around 60 degrees below the horizon). Furthermore, G and N are only approximations. They are both easily corrupted by unstable motion and abrupt changes in orientation, which conveniently aids parking detection.

2.4 Results

Briefly, for clarification, results relating to ambient noise and light intensity were not included here as little could be inferred (with respect to vehicle-parking activity detection) from their respective sensor readings. From discussions with our volunteers after the data collection period, it emerged that the majority of users had their smart-phones either in their trouser pocket or handbag when in transit. As such, both sensors were somewhat shielded from the environment that they were meant to be sensing (both when travelling inside the vehicle and walking), and thus did not produce much useful data for which to train our machine learning systems on.

Due to economic as well as temporal constraints, it was not possible to test our system in a live environment[7]. Nonetheless, our evaluation of ParkUs was based on cross-user validation: where one user's dataset was left out and the system was trained on the other user's datasets before being tested on a yet 'unseen' (at least from the ParkUs system's point of view) user/dataset. We simulated the real-time algorithm in a MATLAB computing environment whereby the algorithm could not see into the future of the dataset/journey, i.e., it received only chunks of data for the particular time-step (similar to how we envisage our system being deployed for real). Furthermore, we limited the amount of data that our ParkUs system was able to store locally in its buffer. Note, whenever a test dataset was put through our classifiers, it was not added to our training dataset, since, in a real-world deployment of the smart-phone application/system, we do not envisage users continuously tagging/recording/labelling their journeys. Finally, our system was able to classify small data chunks in the order of hundreds (if not thousands depending on the processing power of the computational resource being used) instances per second. This would suggest the ParkUs classifier could easily be run in real time on less 'powerful' processors (such those found in a typical Android OS smart-phone).

A parking activity detection was considered valid if it was made within ten minutes from when the parking activity actually occurred. Note, the nearest detection was considered correct should multiple detections be raised by the algorithm. The somewhat generous time-shift criterion was developed to ensure that a correct detection of a parking event was not recorded as incorrect due to inaccurate volunteer annotations.

Similarly, the precise definition of a parking or un-parking event was critical to the evaluation of our proposed ParkUs system. Clearly when a user transitions from either walking to driving (i.e., being inside versus outside of a vehicle), or vice-versa, an un-parking or parking event ensues. However, from observing the training datasets, it was clear that drivers often spend a few minutes (if not more) within their respective vehicles before and after parking. This behaviour could be attributed to many different scenarios, likely causes include drivers either collecting their belongings before exiting the vehicle (parking) or potentially 'warming-up' their engines/planning their route before setting off (un-parking activity). Since in both cases, a user was simultaneously occupying

[7] It turns out there is a limit to how many donuts you can give to volunteers to record and test vehicle-parking related smart-phone applications.

a parking space, it was decided to define a parking event if a driver remains stationary for more than 5 minutes.

2.4.1 Transport Mode Classification

One-versus-all classifiers were trained for three motions: walking/kinetic, static and vehicular motion. The following machine learning algorithms were compared for our application: Decision Tree (J48), $k-$Nearest Neighbours, Multilayer Perceptron, Support Vector Machine (trained using a standard RBF kernel), Naive Bayes, Ada-Boost, and Random Forest. For non-ensemble machine learning, three different feature selection algorithms were experimented with (inspired by [33]): correlation based, information gain and gain ratio.

Overall, the Random Forest models achieved the highest parking detection accuracies: 98% in 10-fold cross validation and 96% in cross-user validation. Again, given the time pressures and scope of the thesis, discussions on the results of the other machine learning algorithms are omitted. Random Forest trains and averages over multiple trees on random sub-samples of the training dataset. Each tree trained by the algorithm used a random subset of all features.

Table 2.2 highlights the most important statistical features (i.e., most discriminating with respect to the transport mode in question). A grid search of bagging parameters saw optimal accuracies with one hundred simple trees with a maximum of ten splits. Ensembles of greater size yielded marginal gains in terms of accuracy while costing more (at least in terms of training times on our local machine). Furthermore, limiting the size of the ensemble and number of splits allowed for some/greater generalisation, since over-fitting is a common problem with machine learning systems.

2.4.2 Detection Accuracy

Table 2.4 highlights the results of our parking activity detection algorithm as well as benchmarking it against various competitor's reported performance.

The lowest energy version of ParkUs did not use GNSS data for parking detection. The GNSS sensor was only polled once a parking activity had been detected to record the vehicle's location in order to update our vehicle-parking 'heat-map'. It is envisaged that the 'heat-map' will aid future drivers in the area to find a vacant vehicle-parking space. Maintaining a short register of previous modality classifications allowed the system to ensure that a transition or parking event actually occurred, rather than, say, random stoppages due to pedestrians crossing or traffic lights at intersections.

Nonetheless, ParkUs correctly detected 57 out of 58 un-parking events and 52 out of 53 parking events, thus achieving a True Positive Rate (TPR) of roughly 0.98 for vehicle un-parking and parking detection [12]. Our system had a false-positive parking and un-parking activity detection rate of 22% and 16% respectively. However, only one false negative (i.e., a missed vehicle parking activity detection) occurred, see Figure 2.15 for more details. Delving deeper into the anonymised volunteer's sensor data as well as the output of the modality detector (framework shown in Figure 2.5), we discovered that the confidence

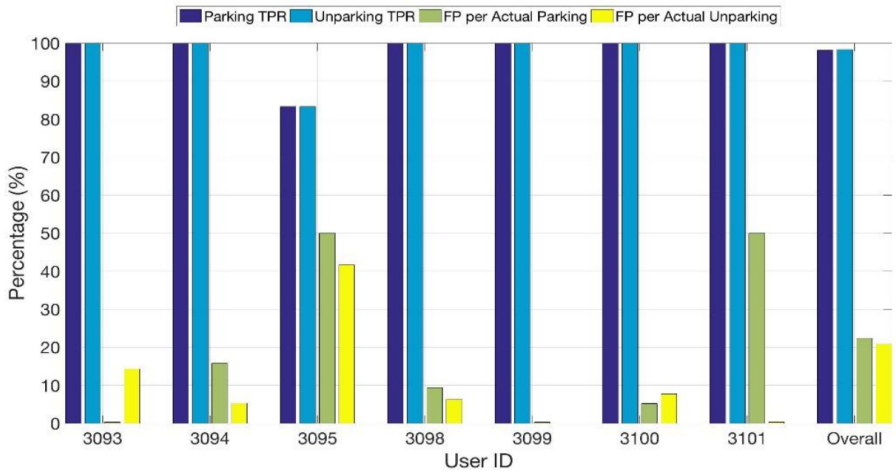

FIGURE 2.11: Cross-user performance of ParkUs evaluated on our volunteers dataset. The True Positive Rate (TPR) is plotted on the right, in the 'overall' column. The TPR was evaluated as the sum of (correctly) detected parking activities/events over the sum of all true events across all users. Similar methodology was used to evaluate the False Positive (FP) per actual parking event. Note, from the final right hand columns, ParkUs achieves a TPR of 98% for both parking and un-parking activity detection.

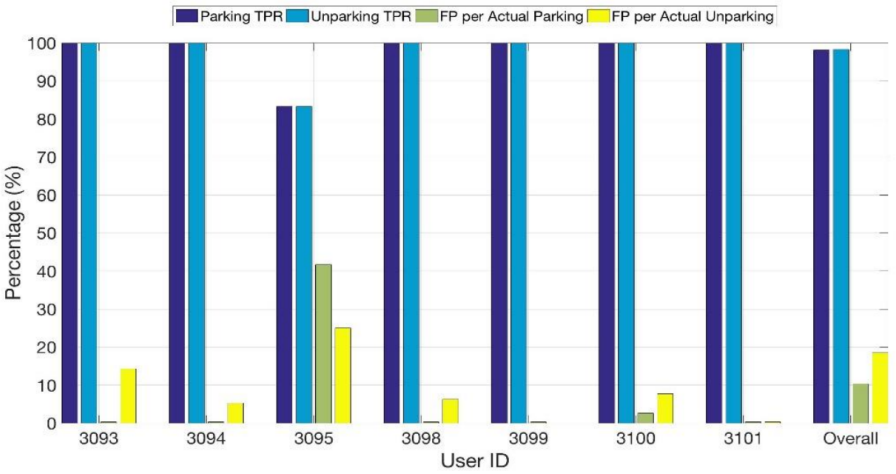

FIGURE 2.12: Cross-user performance of our modified ParkUs-SA (Speed-Assisted) detection algorithm. Computing speed data from a set of time-stamped GNSS data points allows for more confident parking activity detection by eliminating 11 false positive detections, whilst true positive detection rate remained unchanged.

values associated with modality classification were low (and of similar value to the other classes) such that the classifier ended up switching between classes as none were dominant (in terms of likelihood/confidence). Furthermore, investigating the physical locations of the mis-classified events highlighted another problem: they all occurred on the same area of cobbled streets in Bristol's city centre. It is likely that the low vehicle velocity combined with frequent stops (due to pedestrian crossings), and the vibrations generated from the uneven road surface, caused the mis-classifications.

In contrast, Figures 2.13 and 2.14 highlight the potential robustness of our detection system. Notably in a journey where the entire dataset was either walking or stationary (see Figure 2.14), our motion classifier was able to correctly classify the motion classes even though none of the training datasets included an all-walking journey. Similarly, Figure 2.13 shows how our classifier and FSM (see Figure 2.6 for more details) were able to correctly classify the actual vehicle-parking events, even though multiple transportation states were occasionally sensed but disregarded as they did not fit with the logic (e.g., parking before an un-parking event occurred). Note, in this example no GNSS sensor data was used for inferring vehicle-parking activity.

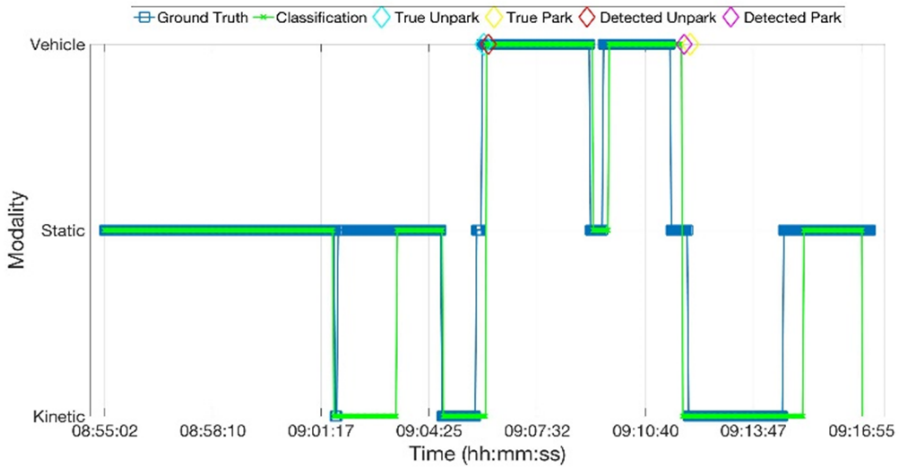

FIGURE 2.13: An example of a typical correctly classified journey. Note, *kinetic* class refers to walking motion.

2.4.3 Detection Delay

Within the ParkUs detection system, a short 5s window length with 50% overlap was used for each of the motion classifiers in the cascaded modality detector. A ten sample window 'look-back' buffer was used to ensure the fragmentation was low. Since each window overlapped by 50%, this meant a minimum of 30s was required for stable motion classification.

Although our system was not trialled in a 'live' environment/deployment, we were able to estimate the detection delay through our system of cross-user validation where we 'fed' the system previously unseen sensor streams. Overall our system had a detection delay of just under a minute (four minutes less than the reported detection delay by ParkSense [62]).

FIGURE 2.14: An example of an 'all-negative' journey classification result, highlighting the ability of ParkUs to work with odd, or unusual datasets without mis-classification.

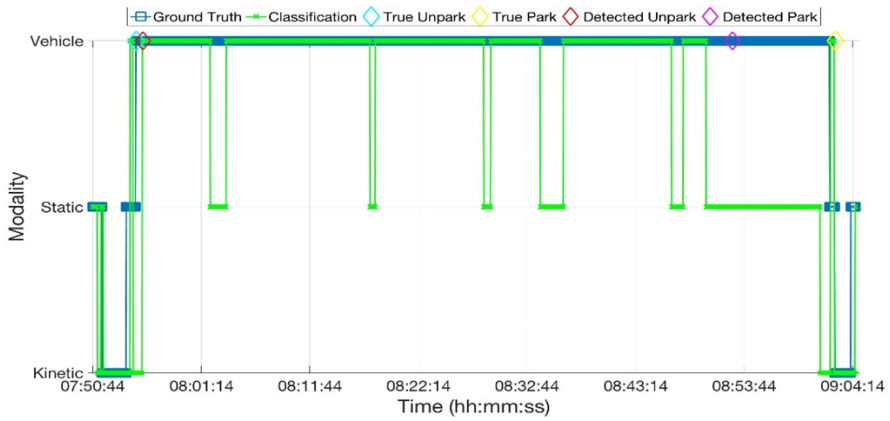

FIGURE 2.15: Example of a false negative detection of a typical user trip. Towards end of their journey our classifier mis-classifies (inside) *vehicle* with *stationary* mode.

Our ParkUs-SA (i.e., speed assisted) version had a slightly longer detection delay due to having to ask the GNSS module for more information when ParkUs-SA sensed/believed a parking event had occurred. Again, the overall mean detection delay rarely exceed a minute.

2.4.4 Estimated Detection Energy Consumption

Given the numerous smart-phone related parking detection applications (that we evaluated, see Table 2.4, page 37), it was not conceivable to translate all the various algorithms and test them on our dataset. Furthermore, the various smart-phone applications were not publically available to download for direct/device usage comparison. To compound the problem of comparing energy usage, not only do smart-phones have different sensors, but what may run efficiently on one smart-phone model may not do so on another. Furthermore,

different operating systems and versions will also have an impact on energy consumption per parking activity detection.

Therefore, our preferred method of estimating the energy consumption per vehicle-parking activity detection involved breaking down the detection algorithms (as they were described in the relevant literature) into actions undertaken by the smart-phone, such as polling the GNSS sensor indoors or sensing Wi-Fi channels for local access-points. Each of these actions could in turn be costed in terms of their energy usage by using a look-up table developed for an early smart-phone, namely the Nokia N95. Even though it has been over a decade since its launch, the Nokia N95 is a (surprisingly) well researched and documented smart-phone. See [2, 15, 70, 102, 107] and [46] for more details. Note the Nokia N95 handset came with all the sensors (minus the Android operating system) necessary to run the various parking detection algorithms being compared.

Several further simplifying assumptions had to be made in order for fair comparison and to aid evaluation. For example, quality of radio signals were generally assumed to be excellent, ubiquitous and within range of whatever source (be it GNSS, Bluetooth or Wi-Fi, etc.). In reality this would rarely be the case, either due to distances involved, interference or wireless channel congestion. However, since our aim was simply to 'cost' the detection methods, as long as the same procedure and assumptions were applied to all the systems being compared, this should not be an issue. Further assumptions (note examples of typical energy estimates are provided in Appendix A.1) include:

- GNSS and Wi-Fi scans took between two and three seconds to obtain respectively
- No Bluetooth connection was available
- Data packets were assumed to be sufficiently small such that they took less than a second to send/receive
- Battery life was assumed to be roughly $16,000$J supplying 3.7V at 100% efficiency[8]

Finally, we had to establish a test case (or scenario) for which to evaluate/estimate the energy consumption of the different vehicle-parking detection approaches. Our test case ran as follows: a user/driver carried with them a Nokia N95 device for a total of three hours. Within that time, the user un-parks, drives for 30 minutes, then parks their vehicle in the city centre. After spending 2 hours in the city centre, our theoretical user returns home, un-parking and driving again for 30 minutes before parking at his home. In total, two parking and two un-parking events took place in our simulation.

Note that the likelihood of false positives were incorporated into the evaluation of ParkUs and ParkUs-SA. In these cases, a false positive detection would cost extra energy in requesting (unnecessary) location co-ordinates and sending data over cellular networks. Some of the other systems described in Table 2.4

[8]These were values obtained from the original Nokia N95 specification.

did not report their False Positive Rate (FPR) and so were given the benefit of the doubt and were assumed to have none [9].

Each parking activity detection system we evaluated was subjected to the same test scenario (as previously described). To aid comparison between the systems, some specific assumptions had to be made for certain systems such as ParkSense [62]. ParkSense only detected vehicle un-parking events, therefore it was assumed that it could also detect vehicle-parking events for fair comparison. In this particular case, it was assumed that the energy cost of detecting either event was identical.

The absolute values shown in Table 2.4 may not necessarily be very realistic, however, what is more useful to analyse is the relative values or ranking of the different systems. Overall, ParkUs had the lowest estimated energy consumption in the test scenario, followed by ParkHere! [78] and our modified ParkUs-SA which occasionally polls the GNSS sensor to get an estimate of speed when it needs clarification between the detected transport modes. As such, ParkUs-SA consumed slightly more energy but was able to reduce the FPR to 0.12 from an original 0.19 (for ParkUs).

TABLE 2.3: Our smart-phone energy consumption model; governing equations. T denotes the total time a process runs for and N is the number of data transfers. See [2, 15, 70, 102, 107] and [46] for more details. Table reproduced from [12].

	Process	Energy Estimate
UMTS	Idle-off	$E_{io}(T,N) = \max(\min(T + T_{io}, N \cdot T_{io}) \cdot P_i, 0)$
	Active-idle	$E_{ai}(T,N) = \max(\min(T + T_{iai}, T_{ai} \cdot N) \cdot (P_a, 0)$
	Tail	$E_t(T,N) = E_{ai}(T,N) + E_{io}(T - N \cdot T_{ai}, N)$
	Send	$E_s(N) = N \cdot P_a \cdot T_{tr}$
	Total	$E_u(T,N) = E_s(N) + E_t(T - N \cdot T_{tr}, N)$
GPS	Outdoors	$E_{go}(T,N) = \min(T + T_{gpo}, N \cdot T_{gpo}) \cdot (P_{go})$
	Indoors	$E_{gi}(T,N) = \min(T + T_{gpo}, N \cdot T_{gpo}) \cdot (P_{gi})$
	Event Report	$E_{er} = E_{go}(1,1) + E_{ut}(1,1)$
	Wi-Fi	$E_{ws}(T,N) = N \cdot T_{wtr} \cdot P_{ws} + T \cdot P_{wi}$
	Gyr.	$E_{gy}(T) = T \cdot P_{gy}$
	Acc.	$E_{ac}(T) = T \cdot P_{ac}$
	Compass	$E_{mg}(T) = T \cdot P_{mg}$

[9] Somewhat unlikely given how inherently 'noisy' some of the data streams from sensors were.

TABLE 2.4: Parking detection and energy consumption comparison results of different parking detection systems. Table reproduced from [12].

Study	**Unparking TPR (%)**	**Parking TPR**	**FPR**	**Detection delay (minutes)**	**Test dataset size (events, users)**	**Energy Usage (Estimated, J)**
ParkUs	0.98	0.98	0.19	0.90	111, 7	1240
ParkUs-SA	0.98	0.98	0.12	1.00	111, 7	1880
PhonePark [85]	0.85	0.80	N/A	N/A	N/A, 5	10600
ParkSense [62]	0.83	N/A	N/A	5.50	41, N/A	3330
Park Here! [78]	1.00	1.00	0–0.11	N/A	40, N/A	1840

2.5 Extended Results: 'Cruising' Detection

As it currently stands, our proposed ParkUs system relies on a high level of user uptake for a given urban area in order to be able to provide near real-time parking occupancy updates. This is potentially one of the greatest weaknesses of the proposed system.

To address this weakness we investigated whether ParkUs could evolve to detect when and where driver's commence their search for parking along their trip. Understanding/inferring driver behaviour in such a manner would allow for a richer dataset to be collected on a per user/trip basis. Essentially, we assume that it is unlikely that a driver passes (without parking) a vacant on-street vehicle-parking space when they are near to their destination.

Given that the location of on-street parking spaces could be mapped (to within reasonable accuracy) across a city, such a system would allow for efficient updating of entire road segments from just a single user trip. Potentially, within certain confidence intervals, it is safe to assume that the roads a driver searched 'just' before parking were likely to have no vacant vehicle-parking spaces (or that they were ill-suited to the driver or vehicle size).

Strategies to detect 'cruising' behaviour based on smart-phone data collected from within a driven vehicle were researched, examined and results reported in [43]. In short, 'cruising' detection is not trivial.

Vehicle trace datasets were extremely difficult to collect because, as drivers who volunteered for our study could not annotate (live) their behaviour whilst driving. In turn, this made for difficult training of the various machine learning methods reported in [43]. However, the greatest problem (again) was recruiting enough volunteers in order to collect enough vehicle trace data for our systems to train on.

Nonetheless, a simplified data annotation/scoring system was developed whereby the vehicle-routed distance[10] to the user's desired destination was recorded throughout their journey. As a driver approached and then passed their desired destination (as they searching for parking), it is likely that a 'global' minima appears on the routed distance to destination versus time (during the trip), see Figure 2.16 for an example.

In this particular case/journey, the volunteer was searching for a vacant vehicle-parking space in a large car park located at the University of the West of England (UK), campus. As the user drove past their desired destination, they searched for parking until they found a vacant space. Consequently our system was able, albeit with varying accuracy (see Table 2.5 for results), to distinguish between 'cruising' and 'non-cruising' parts of the user's journey.

Table 2.5 highlights the classification results (F1-scores) of our proposed machine learning algorithms compared to 'off the shelf' systems (referred to as baseline approaches). Three algorithms namely, Decision Trees (DT), Support Vector Machines (SVM) and k-Nearest Neighbours (k-NN) were compared along with our modified/proposed approach. For our proposed approach, we selectively concatenate feature vectors from windows (data samples) and combine them if they exhibit some inter-dependence. The method is discussed at

[10] By using the road network as opposed to the Euclidean or 'as the bird flies' distance.

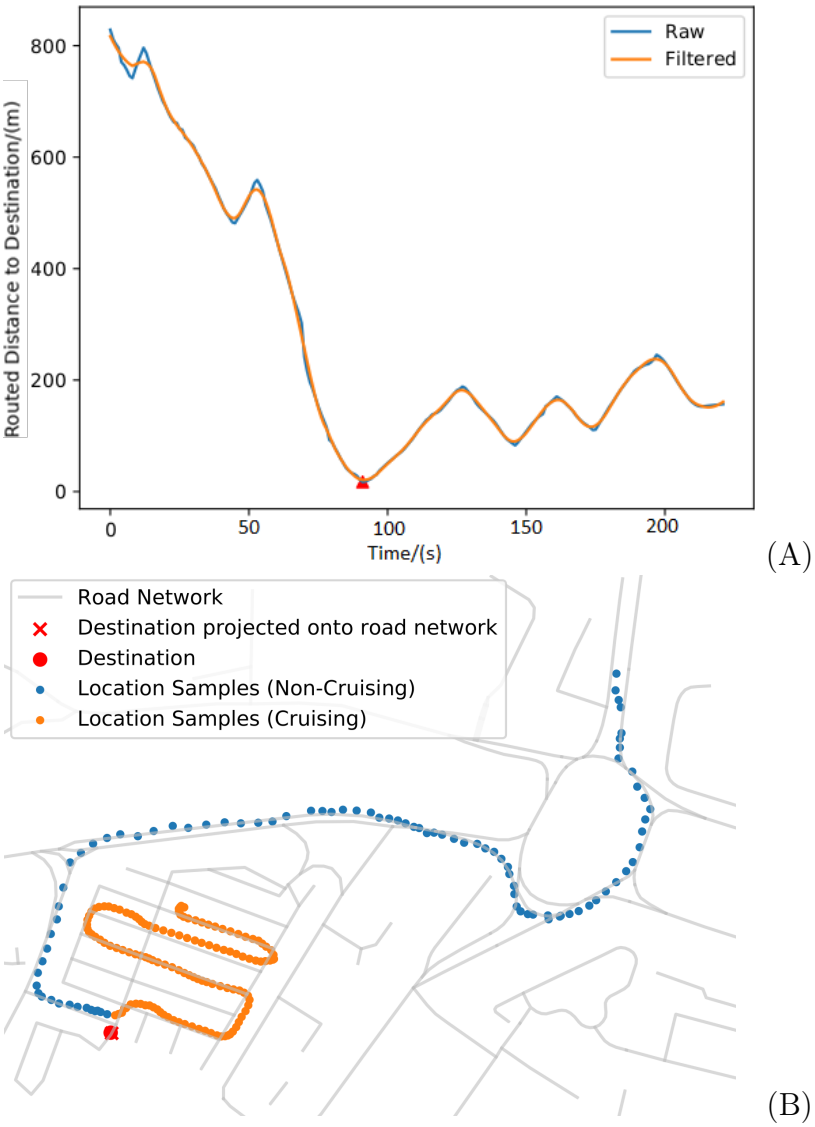

FIGURE 2.16: Location samples plotted on map (A), and corresponding distance-to-destination curve for a journey that involves cruising (B). The red triangle indicates the start of cruising identified by our method. Figure reproduced from [43].

length, including a visual representation of the sub-window feature concatenation method in [43]. The machine learning methods developed are somewhat outside the scope of the thesis, but are included here purely for completeness of this chapter relating to vehicle-parking systems.

Our proposed system does improve accuracy of cruise detection, albeit with a small margin, 0.81 versus 0.77 (harmonic means) for our proposed/modified SVM compared to the baseline SVM. SVMs outperform (in terms of cruising behaviour detection) the DT and k-NN classifiers in both the baseline ($\gamma = 1$, $C = 1$) and the proposed ($\gamma = 0.59$, $C = 6.21$) approach. The optimal number of nearest neighbours for the k-NN approach were found to be 9 and 7 for the baseline and proposed approaches respectively.

TABLE 2.5: Class-weighted F-1 scores of each classifier on each user (total of 41 journeys conducted over 2 months), and the average across each train-test split, for the sliding-window approach and our proposed approach. Table reproduced from [43].

		Results						**Parameters**
Baseline		U1	U2	U3	U4	U5	μ_h	(w, f_s^w)
	DT	0.58	0.74	0.71	0.71	0.80	0.70	$(30s, 10)$
	k-NN	0.76	0.67	0.66	0.81	0.89	0.72	$(30s, 10)$
	SVM	0.80	0.72	0.85	0.81	0.75	0.77	$(30s, 3)$
Proposed		U1	U2	U3	U4	U5	μ_h	(w, k, f_s^k)
	DT	0.66	0.68	0.74	0.73	0.80	0.72	$(7s, 5, 5)$
	k-NN	0.77	0.59	0.79	0.81	0.88	0.75	$(8s, 6, 10)$
	SVM	0.81	0.77	0.85	0.87	0.75	0.81	$(10s, 3, 10)$

2.6 Discussion and Conclusions

Overall, our ParkUs parking activity detection system was able to achieve relatively high TPR and low FPR for parking activity detection when compared to our competitors (see results in Table 2.4, page 37). In order to meet our aims of a low-energy detection method, our first iteration did not poll the GNSS radio module for positional and velocity updates.

In certain scenarios where the parking activity inference was struggling to decide whether the driver had parked, a modified version, referred to as ParkUs-SA (Speed Assisted) was able to poll the GNSS radio module for additional information. As such, having accurate velocity data provided some added clarification on user movement in order to improve transport mode classification confidence. This modification resulted in a larger estimated energy cost per parking activity detection (circa a factor of 1.5, still lower than most competitor systems) but a reduction of the FPR from 0.19 to 0.12 (further results shown in Table 2.4, page 37).

Technological interventions, such as ParkUs, need to be implemented with complementary economic policy when introduced to the public. This is especially so, given that on-street parking spaces are a public good (in most cities). It would not be sensible to introduce such a parking system without charging users for it. In an ideal scenario, the local city government might even go as far as introducing variable pricing schemes for on-street parking spaces in order to achieve optimal occupancy rates of circa 60-80% as evaluated by [81, 83].

In this potential scenario, vehicle-parking search and payment activities take place on a hand-held or in-car device. Therefore, some form of variable pricing based on hourly parking occupancy data could be introduced as live pricing information could be easily accessible online. San Francisco, USA implemented a similar system of variable pricing in SFpark, and they were able to increase revenue. However, SFpark's variable pricing scheme was strictly controlled with tight upper and lower bounds, and price changes were updated monthly (rather than in real-time).

However, in our experimental trials, recruiting volunteers to record their journeys as well as their parking/driving behaviour was partially hindered given limited incentive resources available. In an ideal world, a much larger set of volunteers would be recruited (ideally from different cities) in order to generate more data for our proposed machine learning system to train on.

Unfortunately we were unable to develop, deploy and test our entire envisaged system as shown in Figure 2.3 (page 22). This was due to time and resource constraints as we sought to investigate 'cruising' behaviour detection (see Section 2.5) instead. Therefore, we are unable to evaluate the overall efficacy of such a system in reducing inner-city vehicle congestion.

Nonetheless, a potential (hypothesised) shortfall of such a deployed system could be its need for a high user uptake per deployed city region. This potential problem inspired our research [43]. By detecting and recording where a driver searched prior to parking could reduce the application user uptake needed to provide near real-time local vehicle-parking occupancy updates.

On the other hand, some firms have attempted to alleviate the initial user uptake problem by having employees advertise vacant on-street parking spaces through their smart-phone application. They then charged drivers a premium for the information and or reserving of the on-street parking space. However, a court ruling in San Francisco (USA), effectively banned such behaviour on the grounds that on-street parking is a public good and as such a private entity cannot seek rent from it [104].

Research carried out in the form of an agent-based parking model suggests that at best, parking guidance systems reduce emissions by tiny percentages when vehicle-parking space occupancy is less than 70% [103]. Once past this threshold, the average estimated parking search times increased rapidly to over 12 minutes for parking occupancy rates of over 90%. Briefly, [103] concludes that parking guidance systems are only useful in reducing search times (and therefore congestion and emissions) when occupancy rates are consistently very high.

Note, the model used in [103] was relatively small, less than 50 roads and 5 inner-city car parks were simulated (no on-street parking). In large urban environments where vehicle-parking spaces are not so densely clustered (as typically found in private off-street car parks), it is expected that drivers will experience longer search times.

Furthermore, turn restrictions and one-way roads could lead to ever more inefficient (i.e., longer) search strategies/routes. Such an explanation could have been partly the reason for the longer inner-city parking search times recorded and discussed by Shoup in [81, 83].

Finally, it is worth briefly discussing advances in (general) human activity detection using a smart-phone since the research in this chapter was carried out. Google (part of Alphabet Inc, who actively maintain and release the Android operating system as well as global mapping services) developed and later released a method for Android software developers to access an in-built Android activity recognition API [37].

Few details on how the method works were released. However, anecdotal discussions and internal research lab trials suggest Google's method is rather

accurate at detecting, for example, whether a user is sitting, standing, walking or running. It is hardly surprising that they achieved this since there are over 2 billion Android devices in circulation [52]. Even a small fraction of users who agreed to have their data used by Google for research and development purposes would have yielded orders of magnitude more data than we could ever muster.

Curiously Google has not (to our knowledge) released a parking-activity application or add-on to their popular mapping services. Google seems to provide (although not in all cities and Android versions) a user method to manually geotag where they have parked their vehicle. This feature was presumably intended for use in very large car-parks.

Chapter 3

Taxi VANET Simulations Using Real World Datasets

This chapter is based on research presented and published in the proceedings of the 2018 IEEE Vehicular Networking Conference held in Taipei (Taiwan) [13].

3.1 Introduction

Wireless communication between Autonomous Vehicles (AVs) combined with object detection and ranging systems (e.g., LiDAR) will be crucial to ensure safe and efficient operation. Given the predicted advent of AVs over the coming decades [108], they will likely require a plethora of different wireless systems on board. Wireless systems such as 802.11p, LTE-V2V, 5G and potentially even mmWave for particularly high data rate systems (e.g., exchanging LiDAR information) [77] are being considered for AVs.

Given this potential futuristic Connected Autonomous Vehicle (CAV) scenario, wireless vehicular ad-hoc networks (VANETs) comprised solely of city taxis were investigated (in this chapter) for their ability to transfer environmental sensor data across a city.

Various companies and government bodies could have an interest in developing VANETs. In particular as more sensors, such as those used for monitoring safe human habitable levels (e.g., testing air quality for particulates as well as pollutants such as CO_2 and NO_X) are deployed across cities globally, a method of data collection is needed. Therefore, methods of connecting these sensors for further data analysis (whilst minimising communication costs) are of particular interest.

A current example problem/use-case would be connecting sensors designed to monitor energy usage in buildings to a back-haul network where further processing of such data is carried out on a central or cloud server. Most natural gas and electricity consumption meters are located inside buildings, often in cellars, basements or cupboards under stairs/hallways, i.e., areas that are hard for wireless signals to penetrate. Local planning laws could further prevent optimal placement of cellular base stations increasing the likelihood of 'patches' of poor connectivity within urban areas.

To overcome such problems of connecting thousands of (if not more) sensors distributed across an urban area, future CAVs could provide some of the sensor

network connectivity requirement. CAVs could be uniquely suited to the task as they will have most of the wireless interface present as well as being able to drive anywhere in a city. Since environmental sensors (such as air quality) do not need to provide high frequency updates, and have a relatively low data rate, it is theorised that CAVs could provide some form of sensor-network connectivity as they shuttle passengers (or just drive empty) past. Furthermore, CAVs could collect, store and transmit (i.e., relay) sensor data whenever they are within range of each other or the sensor data destination in the city.

To investigate the feasibility of such a system and its performance, a VANET simulator was developed using openly available taxi trace datasets for Rome (Italy) [8] and San Francisco (USA) [71], combined with respective building footprint and road network topology data from OpenStreetMap [65], to generate a realistic systems level model of a taxi-based V2V network.

Using openly available datasets reduces research costs and allows for future comparison since other researchers can use the same datasets to benchmark their VANET systems. Given the complexities and difficulties of gathering or modelling user transport data, we opted to use openly available taxi trace datasets. These taxi trace datasets capture (to an extent) real passenger trips and background vehicular traffic flows (as taxis still have to navigate through a congested city road network).

The rest of this chapter is divided as follows. In Section 3.2 we cover some of the related work and background of wireless AV systems as well as discussing some of the previous mobility and simulation models used for VANETs. In Section 3.3 we describe in detail our simulator's design, taxi trace data and road network filtering methods. We discuss our method of varying fleet size through our 'slicing' and 'folding' of taxi traces. We obtain building footprint data from two cities for our wireless model and compute a likelihood of LoS given a Euclidean separation distance between vehicles. Results from our VANET simulator are presented and discussed in Section 3.4, alongside some parameter sweeps to ensure robustness of our simulator. In particular we explore the impacts of differing taxi fleet sizes and environmental sensor densities. Finally we conclude the chapter in Section 3.5 and discuss some of the potential weaknesses of our simulator, e.g., the inability to re-route simulated CAVs (as they were based on taxis traces and not passenger trip datasets).

3.2 Background and Related Work

VANETs are an extensively researched area over the last decade or so [16, 29]. VANETs differ from most other wireless networks notably in terms of their dynamic node (vehicle) positioning, rapidly evolving network topology and complexity with respect to routing data packets. VANETs can be often thought of or viewed through the lens of Mobile Ad-hoc Networks (MANETs) or large scale mesh networks with (fast) moving nodes [42]. With such networks, routing packets between desired nodes might be possible in one (time) instance but not in the next, as connected vehicles (nodes) alter the network topology (i.e., which vehicles are connected/within range of each other) changes. The dynamic node positioning and resultant intermittent links between CAVs make

it virtually impossible to maintain a real-time VANET routing table for efficient data routing and dissemination [4].

Traditionally VANETs have been considered as methods for AVs to communicate. CAVs are likely to communicate with each other (e.g., sharing current and planned positional data) and with road-side units to aid coordination at large intersections. There is a large body of research as well as engineering standardisation effort conducted for such VANET systems [42, 77].

Initially, the IEEE 802.11b wireless standard was considered optimal for vehicular networks. However, a detailed study comparing two competing standards (IEEE 802.11p and 802.11b) concluded that under highway, rural and inner-city/urban environments, the 802.11p standard achieved greater network throughput and crucially (for safety critical messages) lower end-to-end delays [6]. Recently, another study comparing the newer IEEE 802.11n and 802.11p standards for vehicular network applications was conducted by [105]. They showed that, under vehicular network conditions, IEEE 802.11p outperforms IEEE 802.11n in crucial quality of service metrics such as latency and Packet Delivery Ratio (PDR).

The IEEE 802.11p standard (sometimes referred to in the literature and by manufacturers as Dedicated Short Range Communications (DSRC)) outperforms other key wireless network standards. This is in part due to the IEEE 802.11p protocol's ability to send messages without time and processor consuming overheads typically needed for association/authentication between wireless devices. The protocol was designed to spread information quickly regarding the CAV's current location and velocity in order to avoid collisions and aid intersection coordination. Consequently, its mechanism for doing so is effectively message flooding, i.e., it provides the aforementioned information at a frequency of 10Hz. However, this update frequency can be reduced when there is significant traffic across the wireless medium (typically found when there are more than a couple thousand CAVs/transmitting-nodes within a square kilometre). Note, only two main parameters namely, the central frequency and bandwidth, are required to be configured in advance between IEEE 802.11p compliant devices for successful data transmission.

Further studies [61] have evaluated the performance of mobile WiMAX (based on the IEEE 802.16e standard) with respect to vehicular network scenarios. Network coverage, mean throughput and end-to-end delay were evaluated through simulation for different vehicular traffic scenarios. Results [61] show that WiMAX performs better with larger vehicular networks, in particular with respect to PDR. Note, the Mobile WiMAX standard allows for devices to communicate with multiple other devices simultaneously (i.e., point-to-multi-point wireless connectivity). However, [61] notes that the IEEE 802.11p standard achieves the low end-to-end delays requirements for safety critical communications between vehicles and/or road-side infrastructure.

Nonetheless, with the increasing need for multiple high data rate sensors (e.g., LiDAR, camcorders and RADAR) to communicate with other CAV systems, various standards such as 3GPP LTE-A Pro (with Cellular-V2X mode) [51] and mmWave (millimetre wavelength), IEEE 802.11ad [64] and 802.11ay [27], have been proposed and studied.

Table 3.1 offers a summary of the various VANET related standards and associated performance metrics. For successful high frequency (such as mmWave systems) communication, LoS is critical. Whereas in general, for standards that operate at lower frequencies, such as mobile cellular networks, they are able (within limits, such as node density, channel conditions etc.,) to operate without LoS. However, in particular with regards to V2V communication between CAVs, LoS is far more preferable to guarantee successful information exchange. Note, that the LTE-A C-V2V standard (as proposed by 3GPP) is able to operate without LoS by utilising lower, cellular frequencies for data transmission between CAVs.

Future CAVs will likely have to rely on multiple wireless technologies in order to achieve full autonomy in a safe and efficient manner. As such, mmWave seems ideal for providing the high data rate communication links necessary for transmitting rich (potentially raw) sensor information between CAVs. Note, whilst a single roof mounted LiDAR unit should be able to detect multiple objects and especially pedestrians or other (vulnerable) road users up to 100–150m away in all directions, it is not advised. Given the necessity for safety, multiple LiDAR units, most likely at each corner of the CAV (potentially with a roof mounted LiDAR unit for added system redundancy) will appear in early CAV set-ups. Having multiple objection detection and ranging units on a CAV will improve accuracy and provide some redundancy should any systems fail (or a LiDAR unit is obstructed for example).

On the other hand, the IEEE 802.11p (and its potential variants) protocol will likely be used for V2V safety critical message broadcasts of future CAV position (i.e., velocity) as well as coordinating CAV traffic (as opposed to data traffic) at busy intersections.

Given the importance of wireless systems for CAVs and their potential to act as a city-wide taxi fleet in the coming decades, this chapter aims to explore how CAVs (or simulated connected taxis) could potentially provide delay tolerant sensor networks with connectivity. In all our simulations, the CAV fleet is assumed to be operating with a V2V standard similar to the IEEE 802.11p standard. Assumptions based around the IEEE 802.11p standard, in particular Line-of-Sight constraints will be explored and discussed in greater detail in Section 3.3.

As briefly mentioned in the chapter's introduction (Section 3.1), various companies and government agencies have an interest in developing large-scale and low-cost (to run) city-wide sensor networks. In particular sensors such as those that monitor energy usage as well as environment sensors (such as air quality, wind speed, temperature and or humidity for example) do not need a high data rate and to provide frequent (e.g., millisecond) updates. Therefore, there could be an opportunity in the future, where CAVs are in abundance to use their VANET systems to provide a 'piggy-back' network connectivity service. CAVs could, as they drive past, collect, store and disseminate (where appropriate) sensor data. By providing connectivity through a fleet of CAVs (which could be providing passenger transportation services as well), a city government could avoid the high connectivity costs typically associated with dedicated cellular and fibre sensor networks.

Furthermore, it could be the case that the CAVs themselves act as an environmental sensing agent. After all, CAVs will require a large array of sensors, such as (multiple) LiDAR/RADAR modules, infrared and ultrasonic distance sensors. Therefore, parked CAVs could provide some of the sensing (and data processing) requirements for 'smart' cities, especially in areas where physical infrastructure may be hard to install (or too costly for a permanent fixture). Not only would future CAVs have an array of environment sensing capabilities, they will likely have substantial wireless communication systems on-board. These could vary from safety critical V2V communication systems (such those based on the IEEE 802.11p protocol) or more energy intensive, high data rate systems such as mmWave (IEEE 802.11ad) for transmitting high resolution and frame rate LiDAR/camera information between CAVs. If there is no cellular connectivity to off-load collected sensor data, parked CAVs could instead off-load the (delay tolerant) data transmission to other passing CAVs through their V2V wireless interface.

Collecting sensor data and disseminating it to a central server (via simulated CAVs) forms part of this chapter's research objective. Such a system should be city-independent and allow for reasonable PDR and end-to-end delay. Note, given the assumed type of sensors being deployed in the simulated urban environment, low latency and high data throughput (or bandwidth) are not necessary. For example, air quality or energy usage data need not be collected and evaluated at every second, hourly readings or daily averages would probably suffice in most instances.

Previous research in VANETs has mostly centred around developing fast short message exchanging wireless systems for moving vehicles. Getting positional information across quickly is necessary when CAVs are coordinating at intersection or busy roads. As such a variety of commercially available simulators have been developed such as VEINS [84] and OMNeT++ [100]. Both VEINS and OMNeT++ have software plugins that allow the user to model road networks explicitly. Most use simple traffic models (such as car following) often derived from SUMO (a micro-scopic vehicle traffic simulator [47]). Recently, attempts have been made to integrate MATLAB with a traffic simulator such as SUMO and obtain parameters from real-world V2X experiments [55].

However, none of these solutions are of particular interest since they either require expensive software licenses, real world calibration or a complex interface development procedure to link a vehicle traffic model to a sophisticated wireless system emulator. Due to the wireless focus of most of these simulators, often they do not allow for thousands of vehicles and sensors to be simulated concurrently as physical layer wireless channel modelling (which includes ray tracing and fading for example) is computationally intensive. Furthermore, our intent is not to improve or study individual wireless vehicle exchanges and protocol design (which NS3, VEINS and OMNeT++ are designed specifically for) but rather to understand information propagation characteristics and feasibility of using a fleet of connected vehicles to gather and deliver sensor data across a large urban environment.

Combining a vehicle traffic simulator with a wireless network simulator is not trivial. Several problems present themselves, including how to process messages

when vehicles are moving and how to route vehicles (traffic generation or trip demand). In the majority of cases, no single simulator can achieve high vehicle location precision with 'real-time' processed message transmission exchanges. The problem being that message exchange success is dependent on vehicle position, distance between them and level of congestion (both in the wireless and physical road intersection domains). As such, the majority of these simulators operate in a real-time/discretised hybrid system/environment, whereby vehicles move according to some traffic model and message exchanges are processed in a discretised fashion. Often the wireless network simulator simply polls the vehicle traffic simulator for vehicle position at a given time in order to simulate wirelees data exchange. However, rarely are the two simulators able to effectively combine vehicle position based upon messages being received (i.e., all clear from a road-side unit placed near a major road intersection). Given this inherent complexity, the simulator we developed and evaluated in this chapter operates as a high-level information spreading ('quasi-infection modelling') system.

Real city taxi trace datasets were used to provide accurate (albeit with some reasonably sophisticated filtering, interpolating and map-matching) vehicle positions at any given simulation time-step. In our VANET simulations, the messages being exchanged were modelled as if they were being broadcast using a V2V safety message protocol (such as IEEE 802.11p), therefore message exchange does not alter a vehicle's trajectory. As such the vehicle positions are independent of wireless sensor message exchange. However, not vice versa, since message transmission is highly dependent (in our simulator) on distance and LoS conditions between the simulated road-side sensors and vehicles.

The author attempted several times to contact different taxi companies in order to obtain larger (and more recent) datasets of taxi traces in urban areas. Unfortunately, none of the four companies (one had over ten thousand registered taxis in their fleet) replied to our requests for data or a potential research collaboration. It is understandable from a taxi company perspective why this data might not be made public, given it could contain passenger identifiable names and potentially their addresses. However, we suspect their major concern is with their drivers (employees) who may not want to be tracked by either their company or a third party (in our initial correspondence we promised we would anonymise all data collected).

Consequently, publically available taxi-trace datasets were sourced and a method was developed in order to scale the fleet sizes for our VANET simulator. Our method of filtering and 'folding' the taxi trace datasets is discussed in greater detail in Section 3.3.5 (page 60).

TABLE 3.1: Brief overview of (current) competing VANET standards.

Parameter	**IEEE 802.11p** [86]	**LTE-A C-V2V** [58]	**IEEE 802.11ad** [28]	**IEEE 802.11ay** [28]
Frequency Range	5.850-5.925 GHz	0.450-4.99 and 5.725-5.765 GHz	57.05-64.00 GHz	57.05 – 64.00 GHz
Operational Range (maximum)	$<$ 1 km	$<$ 30 km	$<$50 m	$<$500 m
Bit Transmission Rate	3-27 Mbps	$<$ 3 Gbps	$<$ 7.0 Gbps	$<$ 44 Gbps
End-to-End Delay	$<$ 10 ms	20-80 ms (V2V)	$<$ 10 ms	$<$ 10 ms
Maximum (combined) vehicular operational velocity support/[km/h]	$<$ 130	$<$ 350	$<$ 100	$<$ 100
Line-of-Sight (LoS) Requirements	Necessary the majority of time, however at short ranges (typically 100m) NoLoS communication is possible	None if utilising lower frequency band (cellular) otherwise necessary	crucial, precipitation can affect transmission	crucial, again precipitation can affect transmission

3.3 Simulation Methodology

Our intention is to assess the data dissemination and delivery performance of these large scale city-wide vehicular networks in an attempt to address our second research question namely, *"What are the minimum requirements in terms of CAV fleet size given the number/density of city-wide sensors to ensure optimal network coverage"*.

From our perspective, optimal network coverage is roughly defined as achieving a high PDR (i.e., the fraction of the data collected and transmitted by the stationary sensors reach their relative destination) and modest delay (i.e., the time it takes for the packets to be delivered to their destination since being first visited by a simulated CAV).

Previous VANET simulators tended to focus on the the wireless protocol design being investigated by simulating realistic channel conditions. As a result, these simulators often are limited to simulating small CAV fleets (less than a thousand in certain cases due to the amount of computation required), especially when ray-tracing, signal scattering, interference and packet acknowledgements are also being simulated. Furthermore, vehicle traffic/mobility patterns are assumed and often are not very realistic; such as using random waypoint models instead of simulating an underlying road network topology. Consequently, our work focuses on using real world taxi trace datasets as these provide a realistic depiction of inner-city vehicle traffic/passenger mobility patterns as well as having hundreds (if not more) vehicles in the simulation.

Our system level VANET simulator is based on open source datasets, such as those maintained by OpenStreetMap [65] and Community Resources of Archiving Wireless Data At Dartmouth University (CRAWDAD) [17]. The former, OSM, maintains relatively up to date building footprint and street network topology for most human settlements by courtesy of users uploading datapoints, geotaging/labelling buildings, bridges etc. OSM also receives data donations from various organisations that have mapped parts of the globe. An example of the detail of building footprint dataset as well as a single taxi trace is shown in Figure 3.4.

CRAWDAD maintains datasets specifically for experiments and simulations conducted for wireless network research. Their datasets range from packet level information (for example of university campuses wireless networks) to high level taxi trace datasets (used in this chapter). For the later, taxis were tracked over a certain period (roughly a month) and their (raw) timestamped GNSS coordinates uploaded (see Table 3.2 for more details).

Since OSM provides reasonably accurate (and up to date) locations, footprints of buildings and road network topology, we were not limited in our choice of cities to investigate (in terms of geography at least). However, the same cannot be said for taxi trace datasets, of which there are few, and even fewer that are (truly) openly available. Given the relative (or more accurately 'claimed') sizes of these datasets, we were initially interested in using Shanghai[31], Beijing [40] and Shenzhen [106] as they tracked circa 4,000, 10,000 and 14,000 taxis respectively. Authors we contacted regarding Shenzhen and Shanghai datasets

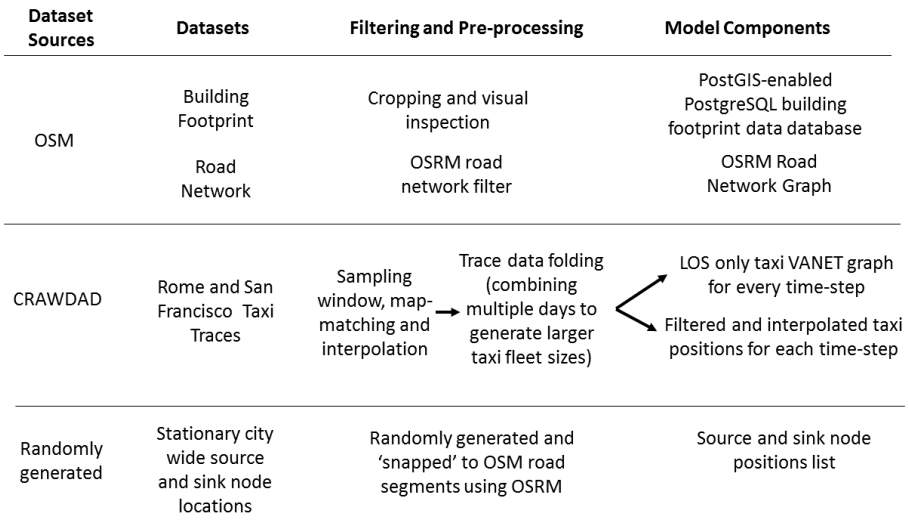

FIGURE 3.1: Schematic showing our VANET system simulation architecture. From left to right (i.e., the order we process data in our system), we list the sources of our raw datasets and our filtering and pre-processing methods (including our taxi trace dataset 'folding' technique as discussed in Section 3.3.5, page 60). Our simulation's model components included interpolated positions of (filtered, map-matched and 'folded') taxis positions at every simulation time-step, a sub-list of the pairs that form a V2V connection (using our sophisticated LoS building footprint database) and the 'snapped' positions of all sources/sinks/sensors in the simulated urban environment. Figure reproduced from [13]. For more details on simulation architecture and processes, see flow chart in Figure A.1 (page 122).

never responded[1]. Consequently, our simulator is limited by the availability of these taxi trace datasets, and as such we focus our efforts on the cities where datasets are openly available (and of reasonable resolution), namely San Francisco (USA) [71] and Rome (Italy) [8]. Table 3.2 summarises some of the details and filtering parameters applied to the two taxi trace datasets used for our simulations in this chapter.

3.3.1 Simulator Architecture

An overview and detailed flow chart schematic of our simulator architecture (including pre-processing and simulation modules) are shown in Figures 3.1 and A.1 (page 122). Our simulator essential works in two parts: an offline data extraction and filtering pipeline, followed by an online simulation of the vehicles moving, collecting and exchanging sensor message information.

The data extraction pipeline has to be run first in order to filter both road network topology data (data from OSM, filtered using OSRM software [67]), building footprint data and taxi trace datasets. A summary of our taxi trace

[1]Eventually we managed to obtain some taxi trace data for Beijing. However, this was too little and too late to be included in our study.

data filtering parameters and results are shown in Table 3.2. Once we have a filtered road network topology, filtered and interpolated taxi trace dataset (i.e., vehicle positions at every time-step) and a list of taxis within LoS and V2V range (again for each simulation time-step) we then run our wireless/message exchange simulator using the aforementioned datasets as inputs.

At this stage, our simulator runs through the positions of each taxi evaluating their relative (haversine) distance to all (stationary) sensors (environmental sensors were distributed in a uniform random fashion and 'snapped' to their nearest road segments). Taxis within V2I range of any sensors were allowed to exchange message information.

V2V message exchanges were then processed (with updated taxi message lists). We collect all manner of data from the simulator, for example the number of messages each sensor and or taxi has stored, where V2V message exchanges occur and when sensors are first visited. Result data was stored locally before being uploaded to a central server for further analysis. We label each simulation run and store input parameters used (such as number of taxis and sensor distribution). Further value ranges of simulation parameters are summarised in Table A.4 (page 123).

3.3.2 Road Network Topology

All our road network data was provided by OpenStreetMap (OSM). However, OSM data files are not optimised for simulations. OSM data files are built in the form of a giant dictionary (i.e., a .xml file), where all road links are listed as keys with their associated tags (values). For example, tags associated with each OSM waypoint can range from 'parking', 'one-way' to 'recycling-bins', all with associated GNSS co-ordinates. Briefly, OSM tag quality varies enormously between cities, regions and countries. More avid OSM contributors such as the UK and Germany tend to have very well populated city maps, whereas those in developing countries or more restrictive ones (such as China and North Korea) tend to have fewer labels/tags.

In order to run simulated vehicle and wireless traffic across our areas of interest, the OSM map needed to be decomposed and reconstructed in a manner that allows for map-matching (of our vehicle trace datasets), placing or 'snapping' our simulated sensors to roads/links and interpolation of vehicle positions at a given sampling frequency (i.e., simulation time-step). Consequently, we chose the openly available Open Source Routing Machine (OSRM) [67] as our back-end method for pre-processing OSM and taxi trace data. OSRM has multiple in-built functionalities and runs relatively quickly (its' back-end is written in C++), making it possible to 'snap' or 'map-match' thousands of sensors or vehicle trace data points in an instance. In our system architecture OSRM runs as a local server and receives queries from our modified Python Version 3+ wrapper [98].

OSRM converts OSM road network data into an edge-expanded graph for fast routing. OSRM uses OSM tags (for individual links or road edges) such as link length, direction and maximum velocity to evaluate the duration (in seconds) of traversing that edge depending on your mobility profile. An example

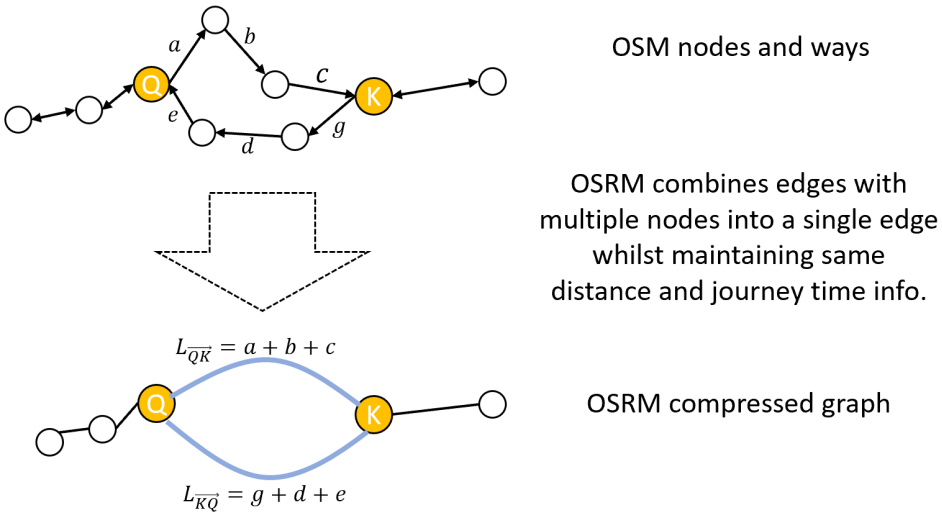

FIGURE 3.2: OSM records multiple nodes (or waypoints) per road edge/link. This allows OSM to develop highly accurate maps as roads can follow complex shapes or curves. However, storing all these extra nodes/edges is memory expensive as well as unnecessary for processing routing algorithms. OSRM [67] compresses road edges with excessive nodes (e.g., top schematic) into a single long link (as shown in the bottom schematic). In order to accurately route vehicles, OSRM takes into account maximum speed limits, link directionality and length when combining multiple sequential edges. Thus ensuring accurate routing representation. In this particular example, there are two separate routes for travelling between nodes (or realistically intersections) Q and K. OSRM realises that there are no intersections between these nodes and 'compresses' them by summing their relative edge weights together and removing excess 'shorter' links and replaces them with a single (representative in terms of edge weights) link such as L_{QK} and L_{KQ}. Note the original road network locations were used for map-matching and simulated taxi positions.

of this graph compression in shown in Figure 3.2. OSRM uses OSM tags in order to assess link travel times given user selected/coded mobility profile. Given these parameters, OSRM is able to combine multiple edges (with no junctions) into longer single edge links with equivalent travel costs/time. To further improve accuracy, OSRM includes all turn restrictions as well as allowing for U-turns to be made on certain road edges, see Figure 3.3 for more details.

OSRM allows users to customise mobility profiles by setting vehicle parameters and local traffic/highway restrictions. Parameters such as fastest speeds (180kph), vehicle mass and length (2000kg and 4.8m respectively), u-turn penalty (20s), left hand driving, traffic light penalty (2s) can be altered. Furthermore, the types of roads your method of transport can/can't access, such as motorways, bicycle paths, canals etc. were taken into account when running fastest route and map-matching algorithms.

FIGURE 3.3: OSM data is used as an input to OSRM (left) before being filtered for turn restrictions (centre) and finally OSRM adds/removes turns/ways that are not allowed by the selected mobility profile (for example large vehicles may not fit under bridges or use cycle paths). Note OSRM adds some mid-point nodes to certain edges to allow for U-turn manoeuvres and for routing to a specific link (rather than the nodes it connects). Figure was reproduced from OSRM's online documentation [67].

For all our simulations we used the standard vehicle profile as maintained by OSRM (some values included in brackets above, for more details see [66]) as was considered to (reasonably) representative of typical inner-city taxi vehicle models. This had the benefit of being comparable in the future. OSRM's routing and map-matching engine were used in order to check whether taxi trace data points were reliable/realistic. For completeness, we set a maximum allowable distance from a nearest road segment to be 200m when map-matching taxi trace datasets.

3.3.3 Filtering Taxi Trace Datasets

Two city taxi trace datasets were used in our simulator to act as VANET enabled vehicles providing wireless network coverage to local sensors (within V2I range) and exchanging messages between vehicles (V2V). Two separate studies conducted in Rome (Italy) [8] and San Francisco (USA) [71] monitored taxi locations at differing data collection frequencies over the course of a month. Both taxi trace datasets contained individual taxi identification numbers and time-stamped unfiltered (raw) GNSS locations.

Due to the nature of these vehicle traces, conducted in large cities with tall buildings, GNSS accuracy was significantly reduced. This creates problems such as vehicles appearing at seemingly random positions along a road or worse, appearing inside building footprints. As such the raw vehicle trace data (for both cities) is mostly un-usable. Fortunately, this is a common problem when working with GNSS datasets and there is an entire body of research, called map-matching, dedicated to filtering and matching timestamped GNSS recorded points with what was likely the real trajectory of the vehicle. Since our goal was not to further map-matching research we briefly considered various algorithms available and settled for a tried and tested system developed by [63]. Note, this is the algorithm used by OSRM to 'map-match' both taxi trace datasets to the underlying road network.

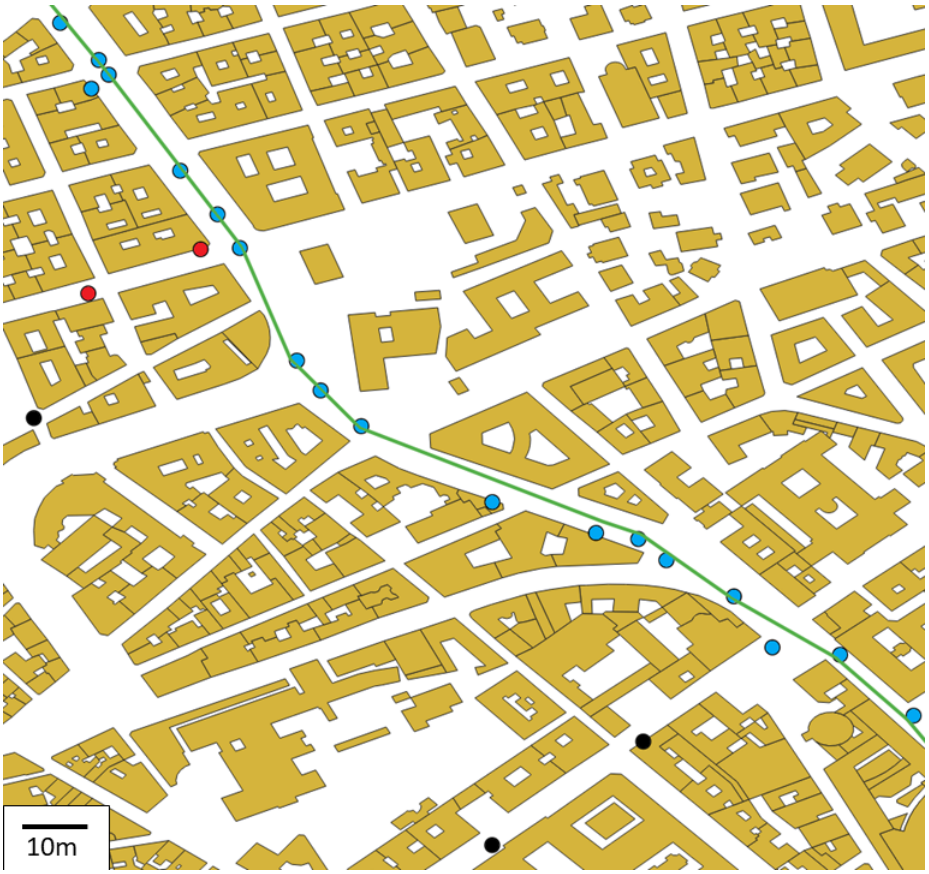

FIGURE 3.4: An example of our taxi trace filtering system for the Rome dataset. A single taxi's trajectory is plotted (all coloured dots) along with the resultant OSRM map-matched trace (OSM road network, green line) with Rome's buildings footprints in the background (golden-yellow polygons). As described in Table 3.2 (page 3.2) our first filter, Filter A, removes points (black coloured dots) which cannot be interpolated due to being deemed too far apart (in this particular case more than our sampling window of 30s). Data points that met Filter A's criterion but were rejected due to their location being deemed implausible (e.g., too far inside a building footprint or due to particular turn restrictions not permitting certain routes) are shown as red filled dots. Finally, cyan filled dots are the trace data points deemed plausible (i.e., they met both filtering criteria) and were map-matched (using OSRM) before being interpolated for our simulations using the original road network. Note, the final set of interpolated map-matched points used in our simulations all lie along the road network whilst preserving the same temporal relations as their original raw points where possible. However, the plethora of points were omitted to aid clarity.

TABLE 3.2: Overview of the Rome and San Francisco taxi trace datasets and filtering method used in our research.

Parameter	**Rome, Italy**	**San Francisco, USA**
Data Collection Period	Feb-Mar 2014	May-June 2008
Number of Participating Taxis	320	536
Median Update Time Period/[S]	7.54	58.7
Sampling Window Size/[S]	30	120
Filter A: percentage of data rejected due to sampling window	0.92	10.1
Filter B: percentage of data rejected due to poor location accuracy	9.98	26.5
Overall percentage of data used in simulations (1-A)(1-B)	89.5	66.1

As part of our initial taxi dataset filtering and map-matching process, we developed two filters to apply to the taxi trace datasets. The first vehicle trace dataset filter, known simply as Filter A, was based on the update frequency of the two taxi trace datasets. The cumulative empirical update frequency distributions (for both cities) are plotted in Figure 3.7. It is clear that Rome has much higher median update frequency (roughly every 8s, or 0.125Hz) compared to the San Francisco taxi trace dataset (median update time period of roughly 60s or 0.017Hz). Consequently, sampling windows of 30 and 120s were chosen for the Rome and San Francisco datasets respectively, and given 50% overlap with each other to aid smoothness of the interpolated trajectory data.

Filter A determines whether a minimum of two (vehicle trajectory) data points lie within the query's sampling window. Should this be the case, then Filter B, makes a query to our OSRM server to ensure the data points are map-matched to the correct road segment (or potentially rejected if they can't be map-matched). If the vehicle trajectory data points did not lie on the same road segment, a 'most likely' route was established using OSRM's routing functionality (a situation that frequently occurs given vehicles in cities often change lanes, direction and roads as they navigate to their destination). OSRM's routing and map-matching service both take into account the time-stamps of the taxi trace data points being investigated (to ensure velocities are realistic) as well as any local speed limits, intersection/turn restrictions and vehicle traffic direction.

The consequences of our two vehicle trace dataset filters (A and B) as well as general properties of the datasets are highlighted in Table 3.2. Overall, the

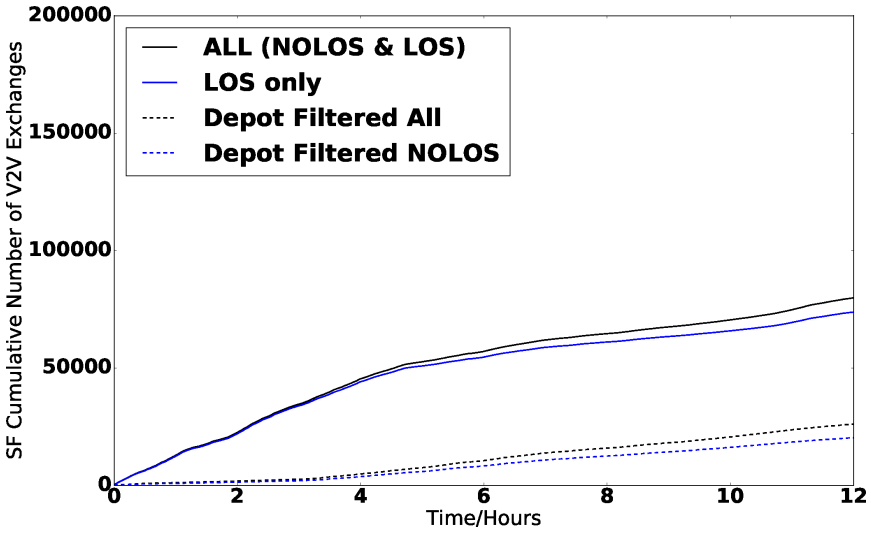

FIGURE 3.5: Effect of filtering the San Francisco taxi depot located at (-122.395, 37.7516) on cumulative V2V message exchanges over a randomly selected 12 hour period. Figure reproduced from [13].

Rome dataset achieved the least amount of data point rejections, almost 90% of the original data points were used in our simulator. Whereas with the San Francisco dataset, a larger proportion (circa 1/3) was rejected.

In the San Francisco case, a much larger proportion (roughly an order of magnitude more when compared to the Rome dataset) was rejected due to poor cohesiveness between vehicle trace data points (referred to as Filter A, see Figure 3.4 for an example). Subsequently, a quarter of trace data points that passed Filter A's criteria were then rejected due to the inability of our OSRM system to confidently map-match the data to road segments. San Francisco has a higher density of tall buildings and a grid like road network topology potentially creating ideal 'urban-canyons' which could effectively hinder the accuracy GNSS sensors used in the study.

After filtering out taxi-trace data points that were deemed un-realistic or too noisy to use, we interpolated the remaining (filtered and map-matched) taxi positions at the desired simulation time-step frequency. For simulation results presented in this chapter we used a time-step of $10s$. Taxi trace data points were interpolated in such a manner to match the final velocity of the two interpolated trace data points. We used OSM's road network topology to ensure interpolated points were on the physical road network as well as ensuring realistic routing strategies (using OSRM) for points that were further apart or across multiple junctions.

Hail-able taxicabs (which our trace datasets are based on) are known to spend a considerable proportion of their time idling/waiting in designated taxi ranks and/or outside important tourist or transport hubs, such as museums or train stations. This imposes certain extra computational costs as well as reducing our fleet size of 'active' taxis. Since for every pair of taxis within V2V communication range, a LoS check has to made, and if successful, the V2V messages stored in the taxis need to be processed. We briefly investigated

FIGURE 3.6: Inset shows a satellite view (satellite image reproduced from Google Maps) of the filtered San Francisco taxi depot. Main image shows San Francisco's building footprints (surrounding the taxi depot) overlaid with V2V exchanges over a 12 hour period. Red lines represent taxi pairs without LoS whereas blue lines are those with LoS.

where the largest clusters of non-moving taxis were located in both the Rome and San Francisco taxi trace datasets.

In Rome, the largest cluster of non-moving taxis was located at Roma Termini[2]. We investigated the impact of filtering out taxis stationed there in our VANET simulations. We concluded that overall, it had a relatively small impact on cumulative message exchange: less than 1% of V2V exchanges occurred there during the study period. Consequently, taxis positioned outside Roma Termini were not filtered from our simulations.

Whereas, in San Francisco, the largest cluster of non-moving taxis was located at an inner-city taxi-cab depot. We hypothesise that the GNSS devices used to monitor and record taxi trace data were not switched off when the taxis were parked overnight (or during the day) at the depot. This resulted in a large (and ultimately) unnecessary contribution to computational work load, as shown in Figure 3.5. Filtering taxis located within 250m of the depot reduced the number of cumulative V2V messages exchanged by about a third. A far greater proportion than the (circa) 10% of V2V exchanges filtered out when only taxi pairs with LoS were considered. We filtered out all taxis parked at the depot in all simulation results presented in this chapter. For completeness, we include a satellite image as well as a model view of the filtered depot in Figure 3.6.

3.3.4 Simulation Time-Step

Individual vehicle trajectory interpolation was necessary in order to assess their positions at higher sampling frequencies. A typical V2V wireless transmission protocol, such as the IEEE802.11p typically have a 10Hz positional and velocity update frequency (note this can be reduced as part of the standard's wireless congestion mechanism when there are large numbers of vehicles within communication range). However, this is not strictly necessary given our simulation

[2]Note, Roma Termini is train terminus as trains do not pass through it.

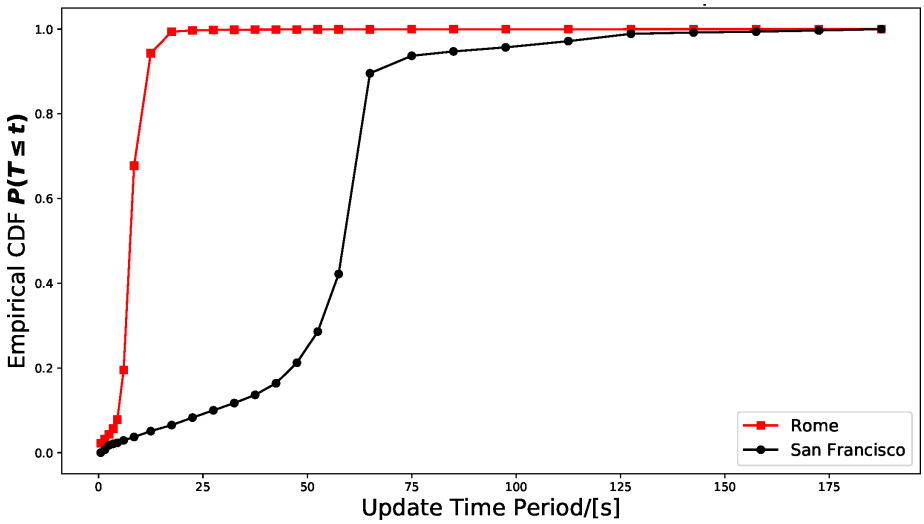

FIGURE 3.7: Empirical cumulative distribution function of the raw taxi trace datasets. Red squares for Rome (Italy) and black dots for San Francisco (USA). Note the median update frequency of the Rome dataset is far higher (roughly) 8s compared to San Francisco's 59s. To compensate, both datasets were filtered, 'map-matched' and re-sampled at a constant time-step of 10s for all simulation results presented in this chapter. Figure reproduced from [13].

interest lies at the system level rather than at the individual V2V packet level. Since our scenario is not safety critical we did not seek such high levels of positional granularity. Recall, that we are modelling information spread across the city from environmental sensors rather than individual taxis sorting themselves at intersections in a safe and efficient manner. Furthermore, message size (between exchanging taxis) was not deemed to be a particularly large and therefore reduces the requirement for simulating longer 'contact-times' (i.e., the time communicating taxis stay within range of each other).

Our choice of simulation time-step was partially limited by our access to computational resources. In particular, our computing cluster had to share Random Access Memory (RAM) resources with other user's simulations. Given the sizes of our (e.g., filtered taxi-trace and building footprint) datasets and length of intended simulations (multiple hours) we were effectively limited to selecting a simulation time-step $>$5s. Larger time-steps would ensure fair use of the cluster and a high-level of successfully completed simulations (thus avoiding the time-consuming process and need to re-submit jobs to the cluster).

Finally, given the granularity of the original taxi-trace datasets (see Figure 3.7) there was a high likelihood of simply over-sampling the taxi trace data. Therefore a time-step of 10s was selected as it nicely balanced our need for reasonably granular positional sampling whilst allowing for a large range (from 250 to circa 2000 taxis) to be simulated within the working memory limits of our computing cluster.

3.3.5 Taxi Trace Data Slicing and Folding

In order to investigate city-wide VANETs we needed to increase the fleet size to mimic, to a certain extent the true proportion of taxis/vehicles present in a city. In both cities there are thousands of registered taxis, however, only a small sub-set were equipped with position tracking devices. To overcome the small number of active taxis in our simulations we resorted to 'folding' taxi traces from randomly selected days (out of the entire month-long datasets).

To make matters slightly more complicated, not only were there an unequal number of taxis participated in both data recording exercises to start with, but active taxis fleet sizes varied significantly throughout the day. We hypothesise these traces might be drivers going home after a shift or stopping for a lunch break (for example). As a result, different numbers of days had to be folded in order to achieve roughly matching and stable numbers of taxis throughout the overall simulated time period for both cities.

Since we were only concerned with a central portion of both cities (64km^2) some taxi routes both left and entered the bounding area. As we were able to record individual IDs of taxis, even if a taxi left and re-entered the simulation boundary its stored sensor messages were not erased. However, no processing of V2V message exchanges occurred outside the boundary area. Note, this happened rarely given that our boundaries were quite large. Furthermore, in the case of San Francisco our boundary is roughly the same as the geographical/physical limits imposed by the coast line. It was noted that the majority of taxis leaving our simulated area were travelling to and from airports (located outside of each city).

3.3.6 Wireless Communication Model

In both of our V2I and V2V wireless models we were primarily concerned with the spreading and exchanging of information between taxis and sensors. Given that we never simulated more than circa 2000 taxis at any given instant (all driving within our simulated 64km^2 urban environment), it was decided not to model channel contention and interference given the relatively low density of nodes.

For example, the IEEE 802.11p protocol was designed for up to 2000 vehicles/nodes exchanging information all within a square kilometre. Given we never observed such high densities of vehicles/nodes in any of the simulations presented in this chapter, our focus turned towards assessing information spread (PRR), PDR and subsequent end-to-end delay of sensor messages.

Vehicle-to-Infrastructure

At each simulation time-step a distance matrix was evaluated to assess whether taxis were within V2I range (set at 100m) of the fixed sensors (scattered along road/edges of the road network). For each vehicle-sensor pair, their messages were exchanged (effectively a set union of the sensor and taxi message sets), and the resultant combination stored as the new, updated set of messages for both

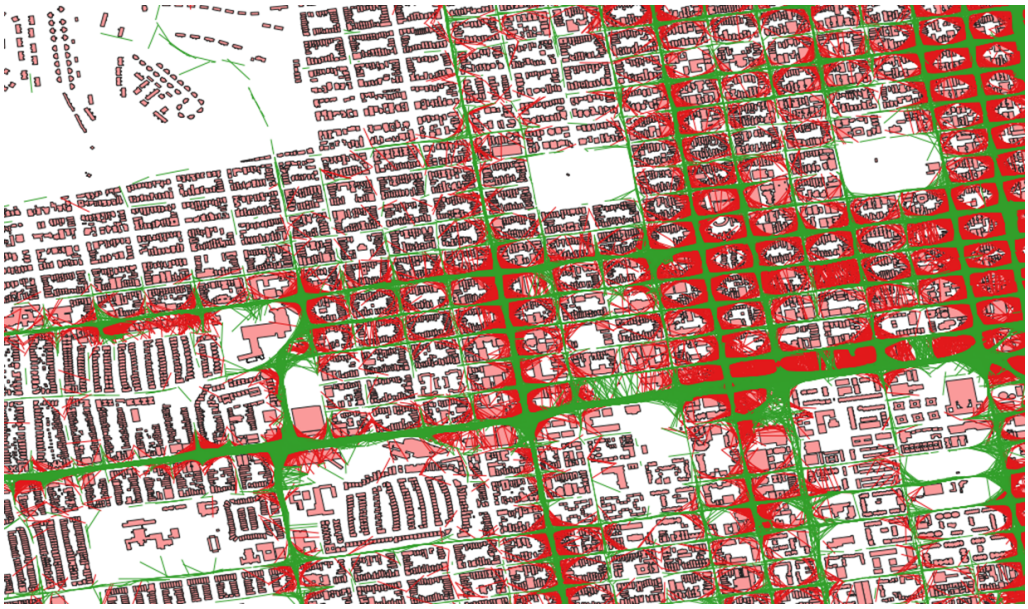

FIGURE 3.8: Visualisation of 96 hours of taxi trace overlaid with San Francisco building footprint. Communicating pairs of vehicles are plotted: green lines for pairs within V2V range and with LoS, whereas red lines are pairs of vehicles within V2V range but do not possess LoS. Note, a typical street 'block' in San Francisco is roughly 140 by 100 metres.

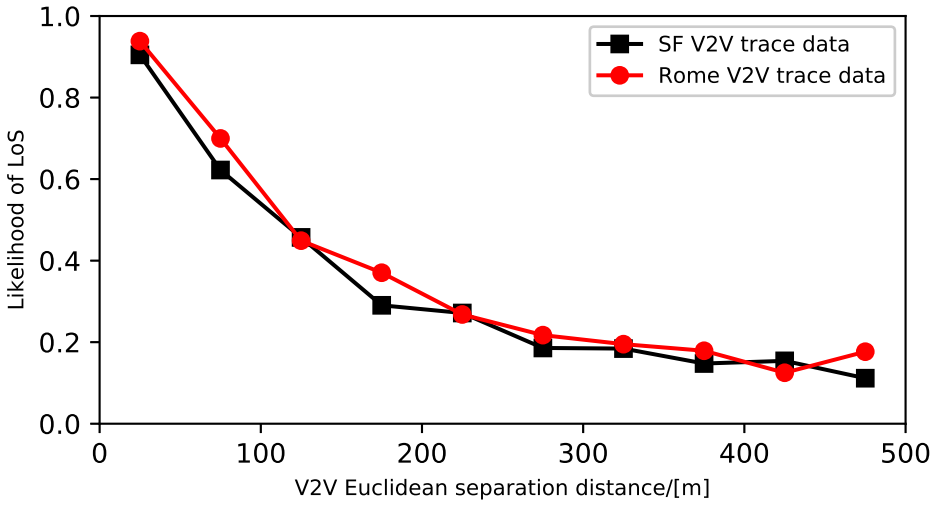

FIGURE 3.9: 96 hours (over 2 million datapoints) of taxi trace datasets recorded in San Francisco (black) and Rome (red) were analysed by finding all pairs of taxis with a Euclidean separation distance of less than 500m and sorting the results into 10 bins (of 50m widths). For each bin, the proportion of pairs of vehicles which had LoS, evaluated using our detailed building footprint and road network datasets), are plotted at bin mid-points. Figure reproduced from [13].

(vehicle and sensor). Note, the order in which sensor-vehicle pairs message exchanges were processed was randomised at each simulation time-step. Throughout the chapter we often refer to transit delay, as it more accurately portrays the type of information spreading delay we were assessing/recording. Transit delay is the time taken for a message to be shuttled between sensors (once they have been visited) by our simulated fleet of connected taxis. We recorded the simulation time-step whenever a taxi visited a previously un-visited sensor and when a new sensor message (i.e., their unique ID) was received at any other sensor (since all our simulations presented in this chapter were of all-to-all scenarios).

Vehicle-to-Vehicle

Again, at each simulation time-step a distance matrix was evaluated to assess the (haversine) distance between all possible taxi-taxi pairs. Taxi-taxi pairs within our set V2V range limit (200m) were then checked if they had LoS by querying our sophisticated OSM building footprint database. A depiction of the building footprints along with vehicle trace data plotted on top is shown in Figure 3.8. We then selected the pairs of taxis that were both within range and had LoS in order to process the V2V message exchanges in a randomised order. We recorded the location and time-step of V2V exchanges for later analysis and to ensure we never achieved unrealistic densities for wireless exchanges. Note, should a pair of taxis be within V2V range and have LoS, successful transmission of data was guaranteed.

Some analysis was conducted in order to understand the likelihood of having LoS given a Euclidean separation distance between the taxis. To this end, over 90 hours (roughly 4 days worth of taxi trace data) was analysed for LoS at every time-step. We recorded the separation distances and whether the taxis had LoS (again utilising our PostGIS enabled building footprint database). We plot the (empirically derived) probability distribution functions for both San Francisco and Rome in Figure 3.9.

3.4 Taxi VANET Simulation Results

All results shown in this chapter are for an All-to-All communication scenario, i.e., all stationary sensors (distributed in a uniform random fashion before being 'snapped' to the nearest road segment) aim to communicate with one-another during the simulation. Sensors (even those within V2I range) cannot communicate with one another, instead, sensors use the VANET (folded-taxi trace data) to transfer, store and carry their (sensor) messages. We varied both the density of vehicles/taxis (by our taxi trace data folding method) and sensors within our simulated urban areas in order to better understand and assess the feasibility and transit delay of a large-scale urban VANET system.

3.4.1 PDR and PRR Results

We define Packet Received Ratio (PRR) as a measure of how many (unique) sensor messages (out of the total number of sensors) an individual taxi is storing

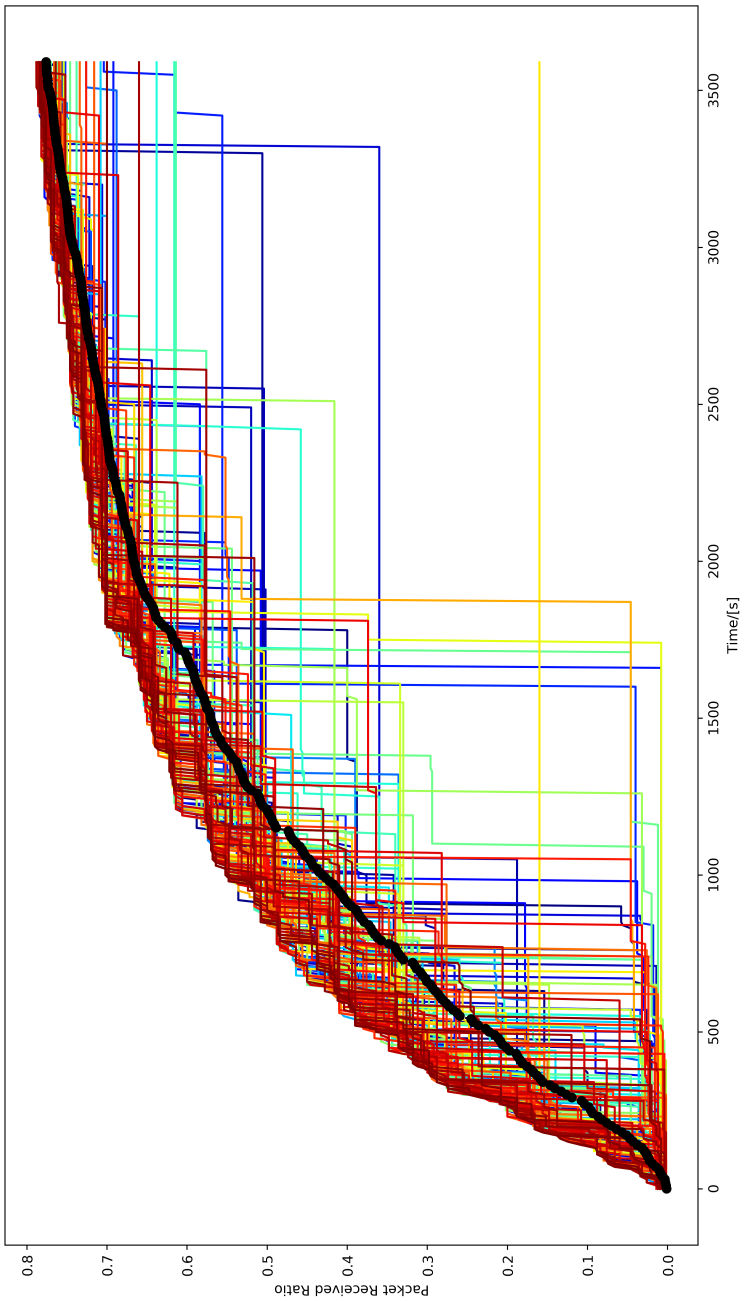

FIGURE 3.10: Single simulation PRR results are plotted for an All-to-All sensor message communication scenario for Rome (Italy). Simulation involved 14 days of 'folded' taxi trace data (circa 900–1000 taxis) and 2000 randomly distributed sensors. For each taxi, we plot their relative PRR over the simulated time period (coloured curves). The solid black curve is the mean PRR at each time-step.

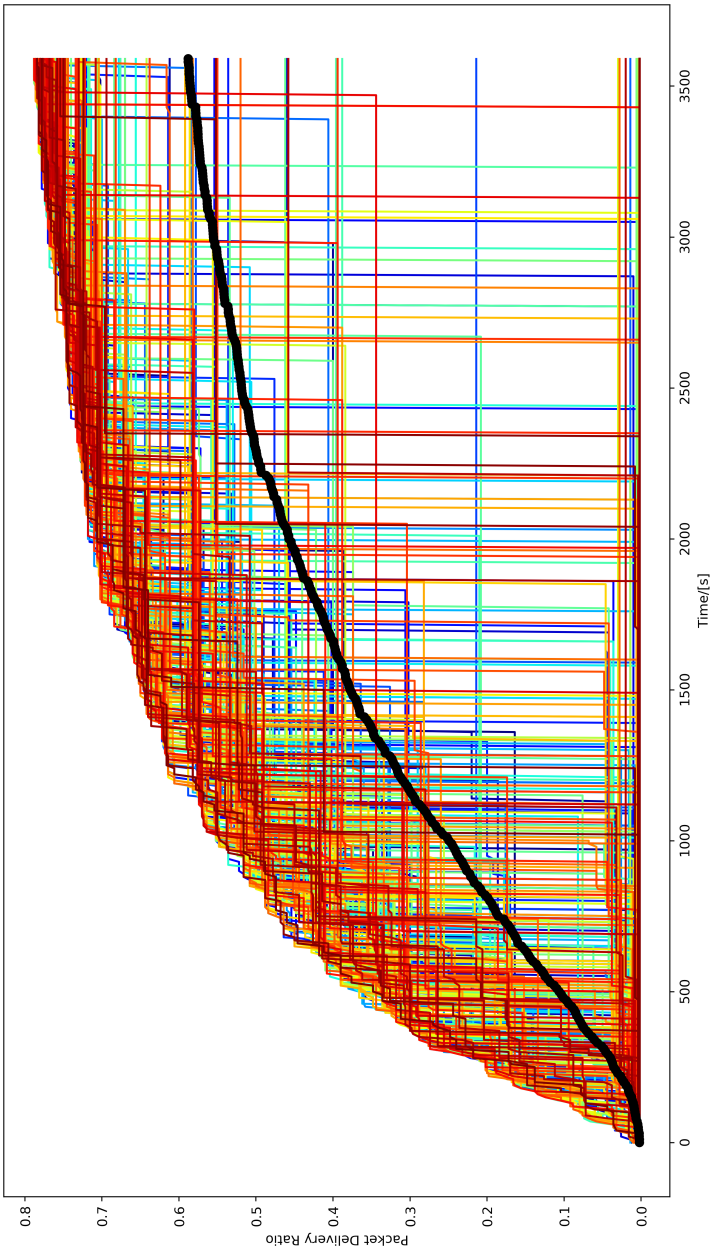

FIGURE 3.11: Singe simulation PDR Results plotted are for an All-to-All scenario for Rome (Italy) with 14 days of 'folded' taxi trace data (circa 900–1000 taxis) and 2000 randomly distributed sensors. For each sensor, we plot their relative PDR over the simulated time period (coloured curves). The solid black curve is the mean PDR at each time-step.

and carrying during our simulation. Similarly, Packet Delivery Ratio (PDR) is a measure of how many (unique) sensor messages (again, as a proportion of the total number of sensors being simulated) each individual sensor has received from V2I exchanges throughout the simulated time period.

Since we recorded at each simulation time-step the messages being exchanged and what messages the vehicles and sensors had stored, we were able to plot the mean (as well as individual) packet received and delivery ratios over the simulated time period.

Figures 3.10 and 3.11 highlight some of the internal workings of our VANET simulator by plotting PRR and PDR for each (individual) taxi and sensor during the simulation, respectively. We also plotted the mean PDR and PRR values throughout the simulated time period.

In both Figures, the mean lags somewhat behind the peak PRR/PDR results. This was expected, since some sensors were likely to be located in areas of the city where taxis may be less likely to frequent (such as residential areas) or route across (such as boundary or perimeter areas) and as such will have low PDR/PRR values. Consequently there is a larger range of PDR than PRR values as some sensors might never be visited during the simulation. Whereas the simulated connected taxis (almost) always interacted with at least one sensor or another taxi during the simulated period thus achieved higher mean PRR than PDR.

Intuitively PRR should be higher than PDR. Simply due to the fact that vehicles both are mobile and act as the (sole) conduit for message exchanges between other vehicles and sensors. Whereas in all of our simulations, sensors can only communicate with nearby passing vehicles and were stationary throughout. As such taxis build up a larger collection of unique sensor messages earlier in the simulation as they cover more ground/conduct more V2X exchanges. This is evident when comparing the average (black lines) PRR and PDR in Figures 3.10 and 3.11 respectively. The mean PDR barely reaches 0.6–0.7 (towards the end of the entire simulation) whereas mean PRR reaches that level roughly halfway through the simulation (circa 1500s or 25 minutes) and continues increasing until reaching roughly 0.8 PRR. Furthermore, this trend is also observable in results Figures 3.17 and 3.18. In these results, mean PDR (solid lines) and PRR (dashed lines) were plotted for a series of simulations with varied (i.e., folded) taxi fleet densities (for both cities) and fixed sensor densities (2000 sensors per simulated environment, or roughly 31 sensors per km^2).

3.4.2 Transit Delay Results

Increasing folded taxi fleet sizes in Figures 3.17 and 3.18 yields relative improvements in both PDR and PRR. Quasi-dramatic gains in final mean PDR values (from roughly 0.5 to 0.8 in the Rome simulations and 0.2 to 0.7 in the San Francisco) were observed as the folded taxi fleet size was increased from less than 300 to circa 1000 active taxis in both simulations. Gains in terms of PDR/PRR diminish as the folded taxi fleets were increased beyond 1000 to circa 2300. Both simulations seem to peak around the 0.7 to 0.8 PDR range.

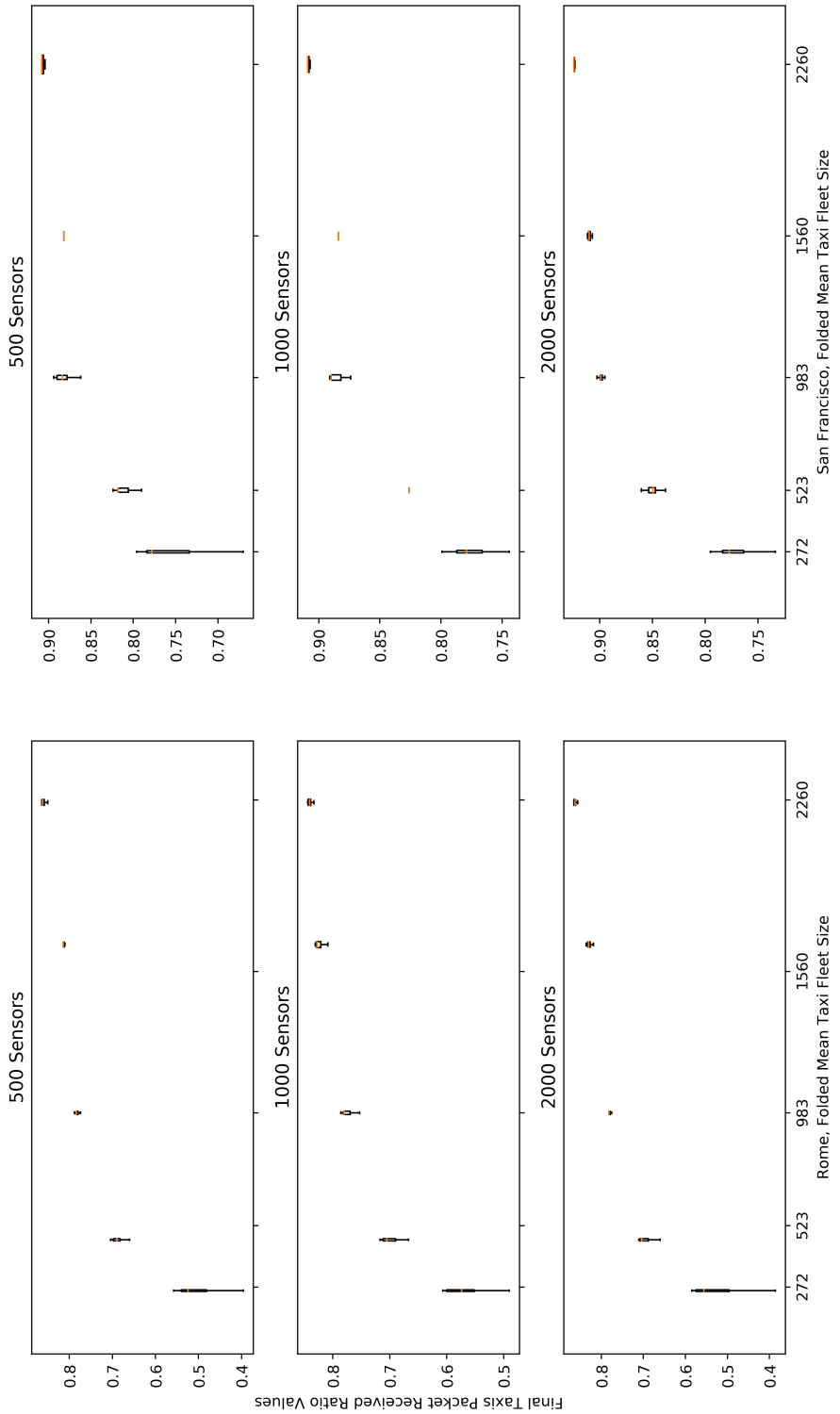

FIGURE 3.12: Series of box-plots showing final taxi-side PRR simulation results (averaged across 20 runs for Rome (left) and San Francisco (right)) for differing sensor network and 'folded' taxi fleet sizes. Note, median PRR values are plotted in orange (horizontal lines), widths of box's represent the standard deviation of taxi fleet size during the simulated hour, heights of box's represent the IQR (middle 50% of the data), and the whiskers are 1.5 times the IQR.

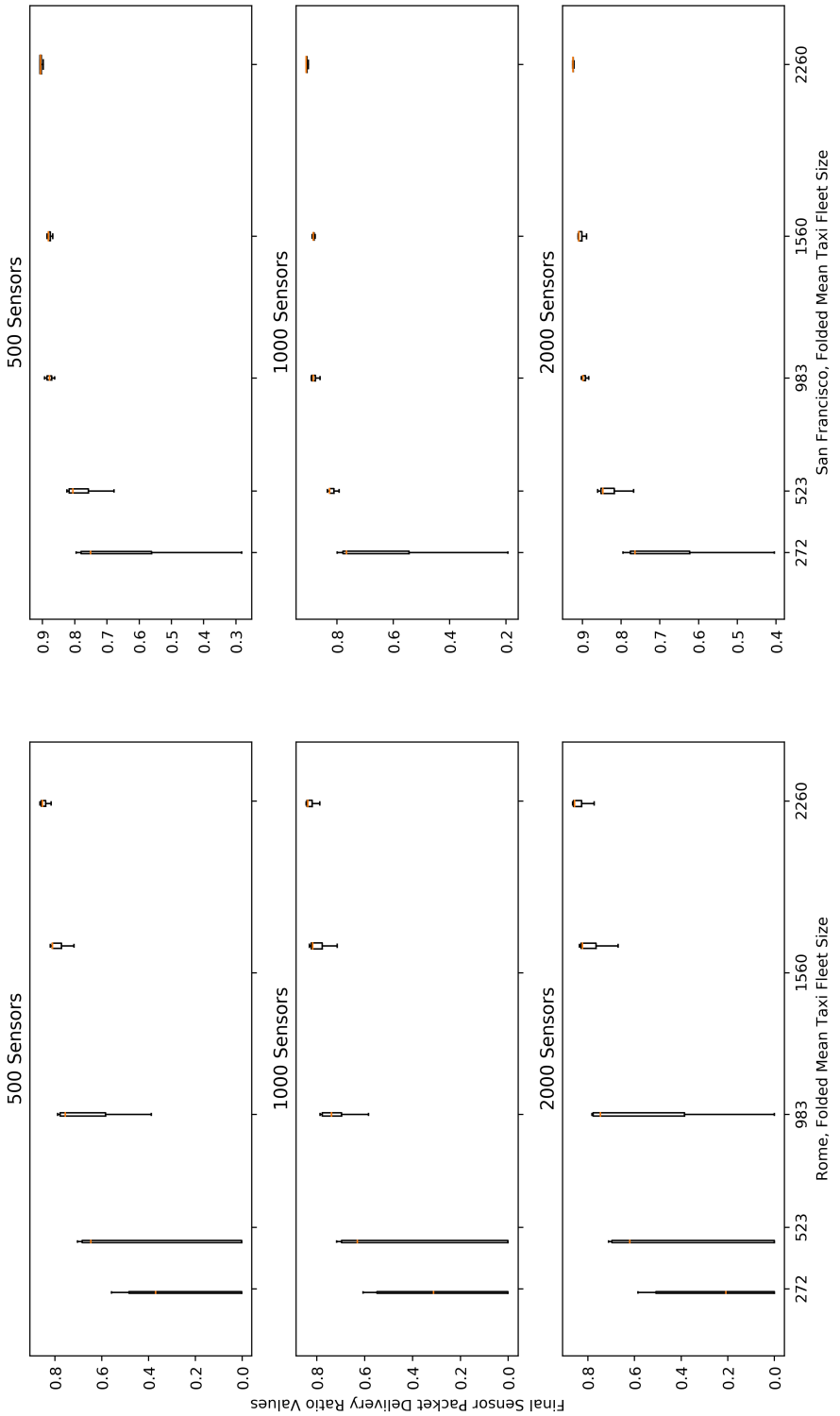

FIGURE 3.13: Series of box-plots showing final sensor-end PDR values of simulation results (averaged across 20 runs for Rome (left) and San Francisco (right)) for differing sensor network and 'folded' taxi fleet sizes. Note, median PDR values are plotted in orange (horizontal lines), widths of box's represent the standard deviation of the taxi fleet size during the simulated hour, heights of box's represent the IQR and the whiskers are 1.5 times the IQR.

To better assess performance of our VANETs under different conditions, a series of simulation runs (20 per combination) were conducted and the averaged final PRR and PDR results are plotted as box plots in Figures 3.12 and 3.13 respectively. Results for each combination of city (San Francisco and Rome), sensor densities (500–2000) and folded taxi fleet sizes (circa 300–2300) are shown. Note, the width of the box plots are the (averaged) standard deviation of the folded taxi fleet size.

With regards to the PRR plots, increasing taxi/vehicle fleet size certainly increases the PRR rate, with smaller fleet sizes (less than 1000 taxis) achieving modest levels of PRR within the range of 0.7–0.9. Further increases in folded taxi fleet sizes (i.e., from 1000 to just over 2200) yield smaller improvements, with maximal values of PRR within the 0.8–0.9 range for both cities.

Similarly, the Inter-Quartile-Range (IQR) decreases as the taxi fleet size increases to around 1000 vehicles and remains fairly consistent for simulated fleet sizes beyond 1000 taxis. This would suggest there are some saturation effects for larger fleet sizes, even as the sensor density increases. It is likely that any fleet density larger than $15/\text{km}^2$ has enough time and likelihood of V2V encounters that the necessary V2V exchanges occur early on in the simulation, meaning that most taxis will have almost all the sensor messages that were collected. Again, it is likely that the vehicle routes (i.e., where they are travelling respective to sensor locations) will have a greater impact on the PDR than the fleet size.

Similar trends were observed with the PDR plots in Figure 3.13. As the folded taxi fleet size was increased (irrespective of sensor density) the final mean PDR values also increased. We observed a levelling off of the PDR once the simulated connected fleet size exceeded 1000 taxis/vehicles (or roughly $15/\text{km}^2$).

Similar to the PRR results (see Figure 3.12) the San Francisco taxi fleets achieved slightly higher PDR values irrespective of vehicle or sensor density used in the simulations. With regards to San Francisco results, folded taxi fleet sizes greater than 500 achieved final median PDR values over 0.8, something the various folded Rome taxi fleets struggled to achieve in a consistent manner (as shown by their larger IQR spreads).

We recorded the simulation time-step whenever a taxi visited a previously un-visited sensor and when a sensor received a new (unique) sensor message (via the VANET). This gave us an idea of the end-to-end delay inherent in our proposed system. We refer to this type of delay measure as transit delay, since it effectively is the time taken for messages to be transferred/transported by our simulated connected vehicle/taxi fleet (i.e., our city-wide VANET).

Figures 3.14 and 3.15 show various probability distribution functions of recorded transit delay for various folded taxi fleet sizes with fixed number of sensors (2000) in the simulated urban areas (64km^2) of Rome and San Francisco. 36 bins of width 100 seconds were used to analyse the results for both cities. Transit delay results for Rome show a wider spread, with some broadly symmetrical (almost Normal) relatively large tails at the higher end of transit delay values. We observed that the transit delay distributions in the Rome simulations appear to shift their mean values as the folded-taxi fleet size was increased. For example, fleets with fewer active taxis (less than 500) appeared

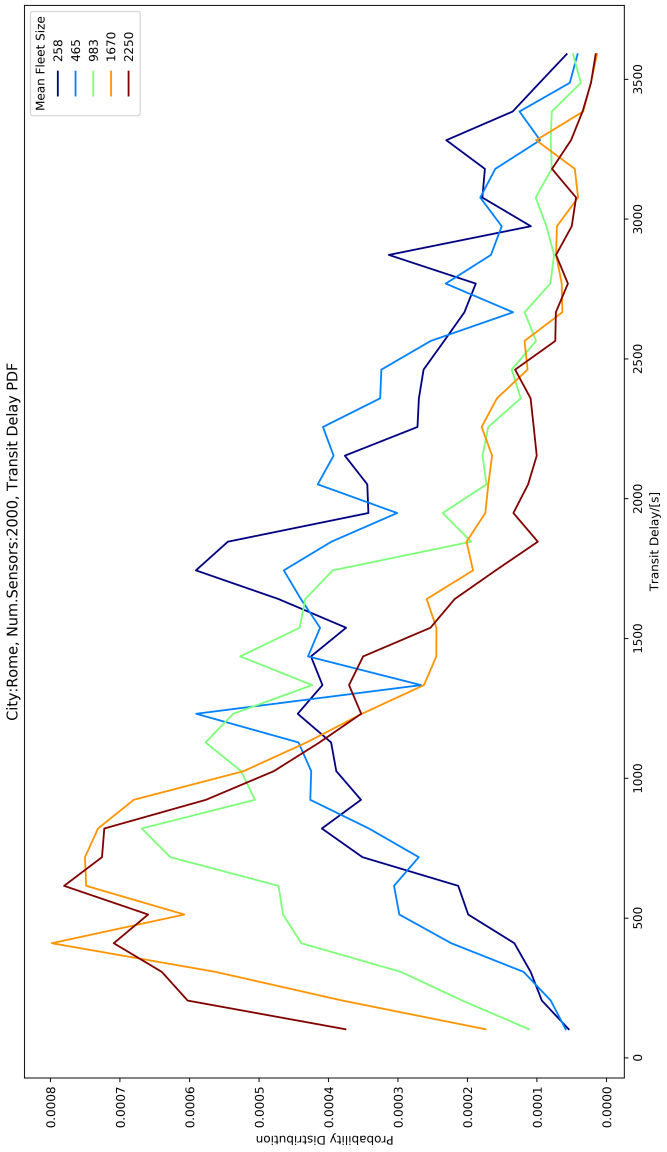

FIGURE 3.14: Empirical transit delay probability distribution functions for a Rome simulation with various 'folded' taxi fleet sizes (see legend) and 2000 sensors. We used 36 bins (with widths of 100s) and plot results at bin mid-points. For smaller fleet sizes (< 500 taxis), we witness a large spread of transit delay, with some peaks at around the 1000–2000s range. However, with larger fleet sizes (> 1000 taxis) we witness a reduction in 'peak' transit delay to around 500s.

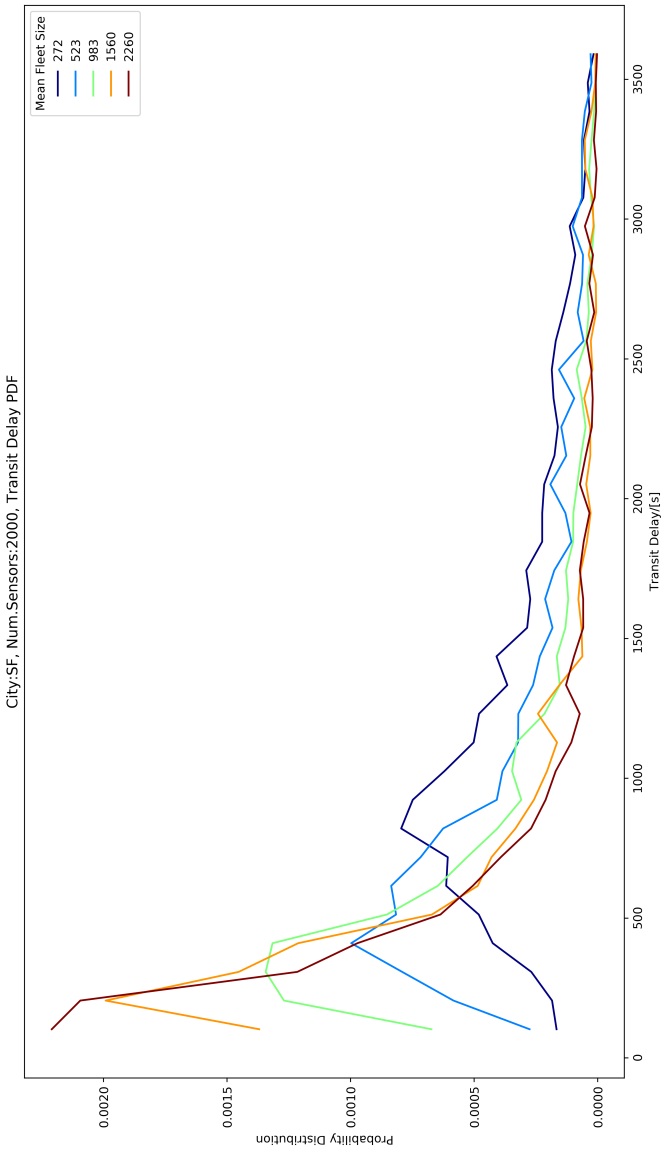

FIGURE 3.15: Empirical transit delay probability distribution functions for a San Francisco simulation with various fleet sizes (see legend) and 2000 sensors. We used 36 bins (with widths of 100s) and plot results at bin mid-points. Overall, San Francisco outperforms our Rome transit delay simulation results (see Figure 3.17, with peaks for all fleet sizes less than 1000s. Again, increasing fleet size yields a reduction in transit delay, with larger fleet sizes (greater than 1000 taxis) having peak transit delay results of less than 300s.

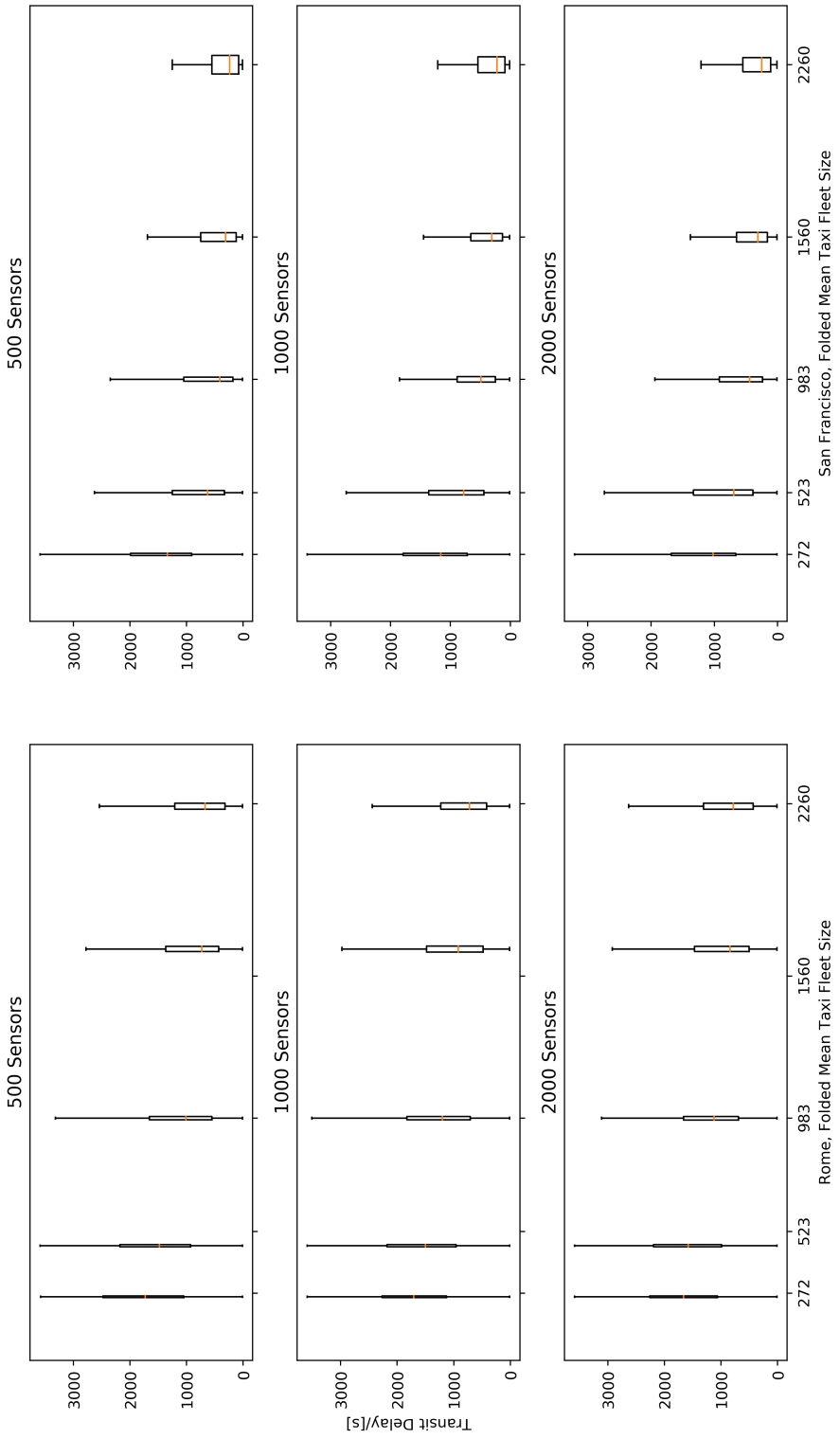

FIGURE 3.16: Series of box-plots showing transit delay results for various 'folded' taxi fleet and sensor network sizes. Simulation results (averaged across 20 runs) are shown for Rome (left) and San Francisco (right). Note, median transit delay values are plotted in orange (horizontal lines), widths of box's represent the standard deviation of taxi fleet size during the simulated hour, heights of box's are the IQR and the whiskers are 1.5 times the IQR.

to have mean transit delays in the region of 1000–1500s. For folded fleet sizes with more taxis, (greater than 1000) the mean shifts, resulting a more skewed distribution with peaks (modal values) around 500s.

Transit delay results for San Francisco exhibit more skewed distributions, with longer/'thinner' tails towards high transit delay values. We observed a smaller shift in mean transit delay values, with smaller folded taxi fleets (less than 1000) achieving a mean transit delay in the range of 500–1000s. For larger fleet sizes, the transit delay probability distributions shift, with peak values $<$ 500s.

Twenty simulation runs were conducted for each combination of fleet sizes, sensor densities (varied from crica 7/km^2 to 31/km^2) and road network topologies (Rome and San Francisco), and folded taxi fleet sizes (circa from 300 to 2300) and averaged across before being plotted as a series of box and whisker diagrams in Figure 3.16. Note, that median values of transit delay are plotted in orange and that the width of the 'box' is set to the standard deviation of the active folded taxi fleet size during the simulated time period.

In all cases, increasing the mean active (folded) taxi fleet size reduces the median transit delay as well as reducing the range of transit delay values (IQR is smaller). This is likely due to larger taxi fleets being able to cover more ground per simulated time-step and consequently able saturate the simulated sensor environment with more messages.

However, reductions in experienced transit delay are not proportionally linear with respect to increases in fleet sizes. A plateau was observed after 'folded' taxi fleet sizes of circa a 1000 taxis. However, San Francisco taxi fleets appear more efficient at reducing delay than their Rome counterparts. The median transit delay for folded fleets greater than 1000 taxis was less than 500s in San Francisco (for all sensor densities simulated in Figure 3.16). Corresponding simulations using similar taxi fleet sizes and sensor densities but with the Rome road network resulted in median transit delays of around 500 to 1000s.

At this stage it is hard to identify exactly why San Francisco taxis perform better both in terms of reducing median transit delay as well as reducing the spread (IQR) of transit delay results as the fleet size increases. A potential explanation could be that taxis in San Francisco might experience a greater diversity of trips, whereas Rome taxis could be more likely to cover similar routes, such as between major transport or tourist hubs. However, to counter this, we specifically picked random day/hour slices of taxi trace data to fold.

On the other hand, there could be some inherent structure in the road network topologies being investigated that reduces message dissemination. After all Rome's initial settlement dates back to a couple of millenia, consequently its road network varies greater in terms of road width sizes as well as structure.

San Francisco was a more recent settlement and large swathes of its road network were built in the last century or so. San Francisco's road network not only is more modern (wider roads, more lanes) but also structured in more regular grid-like format. The grid-like road network topology could (in theory) allow for more 'quasi-fastest' routes between any two destinations as there are many ways of traversing a grid network topology with small variation in overall trip length (i.e., there are multiple fastest routes to choose from). In practice

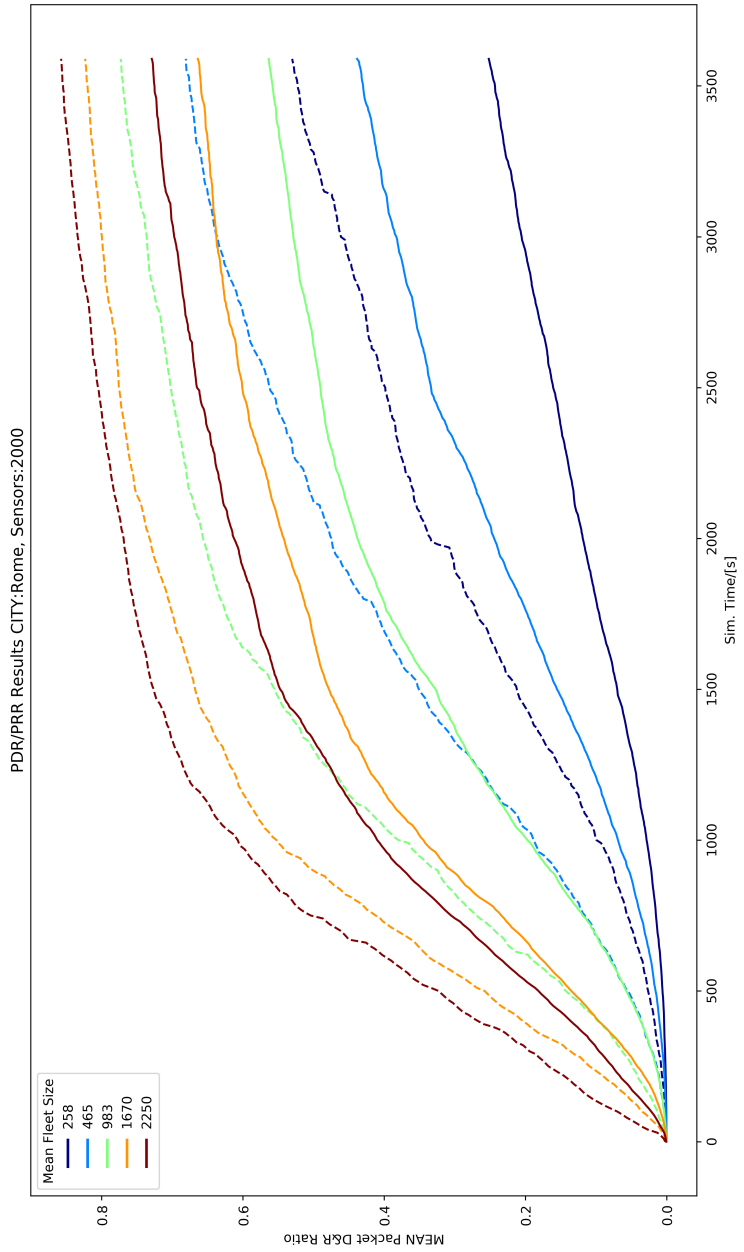

FIGURE 3.17: PDR (solid lines) and PRR (dashed lines) results plotted are for a series of simulation runs (All-to-All scenarios) with varying 'folded' taxi fleet sizes whilst maintaining constant number of sensors (2000) in the city of Rome (Italy). Note how the 'lag' between the mean set of messages carried/stored by the taxis participating in the VANET compared to the sensor-end PDR (i.e., difference between dashed and solid lines) is fairly consistent between the vehicle fleet sizes. The final PDR and PRR values are lower in contrast to San Francisco simulations, see Figure 3.18.

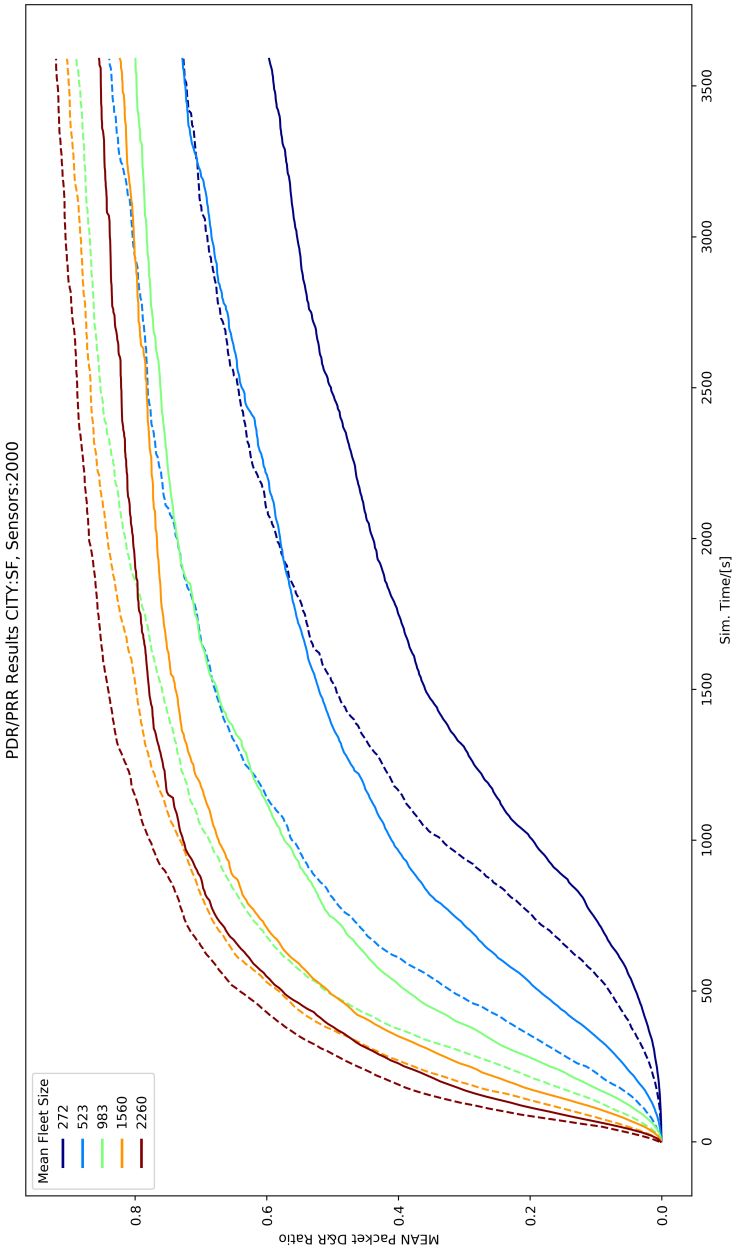

FIGURE 3.18: PDR (solid lines) and PRR (dashed lines) results plotted are for a series of simulation runs (All-to-All scenarios) with varying 'folded' fleet sizes whilst maintaining constant number of sensors (2000) in the city of San Francisco (USA). Again, note how the 'lag' between the mean set of messages carried/stored by the taxi fleet in the VANET compared to the sensor-end PDR (i.e., difference between dashed and solid lines) increases as the fleet size is reduced.

the Romans built many streets following a grid-like pattern across some areas of Rome, such as around Roma Termini or sometimes grids spanning out in a radial pattern from a central square or small garden such around Castello San Angello (Prati district).

San Francisco, like many other North American cities, suffers from having many one-way streets and turn restrictions at junctions, thus potentially reducing the number of 'quasi-fastest' routes between any two points in the road network. However, without further experiments and or detailed road graph analysis and more traffic data it is hard to discern a (precise) relationship between the underlying road network topology and VANET transit delay.

3.5 Discussion and Conclusions

Overall we considered the problem of connecting environmental sensors distributed across an urban environment by simulating a fleet of connected taxis, i.e., a city-wide VANET. Two (publically available) taxi trace datasets (recorded in Rome (Italy) and San Francisco (USA)) were selected, filtered, 'folded' and interpolated in order to generate roughly comparable simulated taxi fleet sizes.

Similarly, the underlying road network was extracted from Open Street Map and we used the OSRM software library to filter and map-match our taxi trace datasets. Several environmental sensor densities were explored along with varying taxi fleet sizes in order to investigate performance characteristics of our VANET at collecting and disseminating information in an All-to-All message exchange scenario (between the fixed sensors).

Briefly in terms of model verification and validation:

- Location of V2X interactions were investigated to ensure they were occurring along roads and not inside buildings
- Stationary environmental sensors were checked to be located on OSM road-segments
- All-to-One, One-to-All and All-to-All simulation scenarios were investigated to ensure model was correctly functioning
- Building footprints were manually inspected and samples computationally checked for inconsistent polygons
- Vehicle positions were all taken from OSM waypoint coordinates and not the node/edge reduced graph used for routing and map-matching
- All folded taxi traces where from the same city, with unrealistic traces and taxi depots filtered as discussed at length in the methodology section
- Folding of real taxi trace data ensured pseudo-realistic user trips and background vehicle traffic flow

Rome VANET simulations appeared to result in consistently lower PRR and PDR when compared to simulations of San Francisco (for similar sized

folded taxi fleets). Furthermore, spreads (i.e., inter-quartile ranges) of results of repeated simulations suggest that taxi fleets in San Francisco not only perform moderately better in terms of delay, PRR and PDR but were also more consistent. There are several potential explanations for these observations.

The first possible explanation, as already briefly touched upon in the results section lies in the folding and trip types for both taxi datasets. It is certainly true that on any given day (in the original, i.e., 'raw') recorded taxi trace dataset a smaller proportion of total participating taxis were active in the Rome dataset. This resulted in more days worth of taxi trip data being folded in order to generate (albeit quasi-synthetically, see Table A.3 in the Appendix) similar active taxi fleet sizes as the San Francisco dataset. It was originally hypothesised that by folding more randomly selected day's worth of trace data, there would be a greater spread or range of trips being undertaken (since we could in essence be merging different days of the week of traffic data into 'one day'). However, the opposite was observed in our results. It is likely that there were more repeated trips (i.e., between major tourist or transport hubs) in the Rome dataset that in San Francisco.

Another potential explanation could be that in the Rome dataset there were more taxis waiting at taxi stands, thus reducing overall fleet ground cover (i.e., the area of the map searched at any instance of time by the entire fleet) per simulation time-step. That said, we did partially correct for this by filtering out large clusters of stationary taxis.

Yet another explanation could be there is some inherent road network topology that disadvantages the Rome dataset as vehicles end up on similar routes, thus reducing the overall fleet search area as a proportion of the simulated environment. This is a subtle difference relative to ground cover, since both fleets could be covering the same amount of ground (assuming similar active fleet sizes) per time period, however, due to turn or lane restrictions it could be the same rather than different streets/areas being searched.

More research could yield certain road graph information, for example by investigating the variation in distance of the 'top' n number of shortest paths between any two points in either city. Regular, grid-like road networks inherently allow for more shortest-paths (of virtually equal lengths) between any two randomly intersections/nodes in the road network. In a perfectly regular grid network with n by m edges road topology, there are $\binom{n+m}{m}$ fastest routes between any two opposite/extrema vertices (nodes/intersections) of the road network. Since there are a certain number of left/right and/or up/down turns to be made regardless of order. However, in reality it is likely that turn restrictions, as well as one-way streets and residential speed limits could have reduced the overall travel time (even though routes could be of similar length).

Due to limits of both computational memory and data (for we cannot fold more days that exist without repeating identical trips) we were limited to simulating no more than circa 2200 taxis/vehicles per fleet and 2000 environmental sensors. It would have been of interest to simulate larger densities of sensors and vehicles to investigate if there were scenarios where final PDR values could be more consistent and potentially higher.

With regards to our second research question concerning the minimum fleet requirements for a given sensor density, we suggest that a one to one ratio (taxis to sensors) would be a good starting point/potential 'rule of thumb'. Figure 3.13 highlights this threshold nicely for both cities; as the taxi fleet size roughly approaches the number of sensors in the simulation the PDR tends to increase to approximately 0.8 and levels off after that. In the San Francisco VANET simulations, we observed slightly better PDR values at over 0.8 as the taxi fleet approached the number of sensors in the simulation. For scenarios below this threshold, the taxi fleets performed badly at collecting and disseminating messages to the stationary sensors as shown by the larger IQR spreads at lower taxi densities (generally less than 15/km^2).

In terms of transit delay we observed a similar but less obvious pattern. As the taxi fleet size was increased relative to the number of simulated sensors we achieved only marginal gains in terms of reducing transit delay. Transit delay certainly decreases before levelling off again. However, the IQR spreads are quite large, at least in comparison to the PRR and PDR results.

Given that in all cases shown in Figures 3.12, 3.17 and 3.17, the final PRR (i.e., the ratio of sensor messages being carried by the taxis relative to the total number of sensors) values were always higher than the final PDR values. This could suggest that some modification to routing strategies could reduce delay and potentially increase overall PDR of the system. Routing strategies along with the ability to alter passenger trips will be explored in detail in the subsequent chapter.

Overall, simulated taxi fleets were able to transfer data from a large dispersed set of sensors in an urban area. Our simulations show that on average transit delay (between a sensor first being visited and receiving new sensor data) decreases from 1000s to around 300–500s for larger taxi fleets. However, given we are interested in collecting non-safety critical sensor data, our system can be classed as a delay-tolerant VANET. In terms of PDR we achieve relatively good results, varying from 0.7–0.9 depending on fleet size and sensor density.

Chapter 4

Agent-Based VANET Simulations

Following on from results obtained in the previous chapter, our research aims to evaluate the performance gains, in terms of wireless sensor network PDR, by strategically re-routing CAVs without overtly lengthening passenger trips. The following chapter is also subject to a filed patent in the US and Japan which covers a system and method for managing a connected fleet of autonomous vehicles providing connectivity to stationary (e.g., environmental sensor) nodes [11].

4.1 Introduction

In order to have finer control over individual taxi/AV trips, routes and passenger demand, we developed a simplified agent-based VANET simulator which no longer relies on vehicle trace datasests. As such, passenger demand can be varied along with the routes selected by the (simulated) CAVs to serve the passengers.

Varying individual CAV routes allows us to evaluate how effective our system could be in terms of packet delivery ratios. For continuity and to aid comparison we use the road network (again, obtained from OSM) of San Francisco (USA) and similar wireless models developed and discussed in Chapter 3. Previously we focused on obtaining taxi trace datasets and developing filtering and 'folding' methods in order to vary taxi fleet sizes. However, we no longer use (real-world) vehicle trace datasets. Instead passenger trips are generated across the urban environment following a distribution inspired by the New York City Taxi and Limousine Commission's extensive data records of all licensed taxi journeys conducted within the New York City metropolitan area [89]. Note, this is the largest taxi-trip dataset that is currently publically available.

Finally, we will investigate how CAVs can be 're-routed' in such a manner that they both provide physical transportation services to passengers as well as network coverage and connectivity to the distributed sensors across the simulated urban environment. It is likely as with most dual or multi-objective optimisation problems that any gains in one dimension lead to a cost or reduction in performance in the other. Understanding this trade-off would be of great interest in assessing the feasibility and the limits of what can be achieved purely in terms of re-routing strategies for improved PDR.

The rest of the chapter is divided as follows. We briefly introduce related work (Section 4.2), before detailing our agent-based VANET simulator design and simulation methodology (Section 4.3). We compare our passenger demand (i.e., trip generation) model with the largest publically available inner-city taxi trip dataset. We introduce and discuss our reduced (from a computational perspective) Line-of-Sight wireless (V2I and V2V) communication model and road network graph filtering method. Furthermore, we developed an edge (road) weighting method and parameter (referred to as α in the chapter) for incentivising efficient AV routing to meet passenger and wireless (sensor) connectivity demand. Agent-based VANET simulation results of structured parameter sweeps of CAV and sensor densities, and α are shown in Section 4.4. We investigated the limits of our system with regards to poorly connected sensor locations and briefly discuss potential mitigation strategies (Section 4.5). We conclude (Section 4.6) by briefly investigating the role of the Largest-Connect-Component (LCC) our AV fleet and message exchange strategies.

4.2 Related Work

Data mules (i.e., vehicles equipped with wireless systems for short-range data transfer) were originally conceived as a method for connecting an area scattered with sensors and access-points by [80]. Their work focused on a simplified grid topology whereby each cell contained either a sensor (i.e., data generating source) or an access-point (i.e., data sink). Their simulated data mules were able to randomly traverse the grid environment collecting and distributing data whenever they landed in a cell containing either a sensor or an access-point respectively.

Data mules have been developed and used to increase connectivity for remote or developing regions [69]. The DakNet system (developed by [69]) used human driven vehicles equipped with electronic memory storage and Wi-Fi access points in order to connect remote village kiosks, families and shops with (albeit temporary) data transfer facilities. Data automatically uploads and downloads when the bus (or any vehicle equipped with DakNet) is in range of a kiosk or wireless hub. Whilst DakNet couldn't provide seamless connectivity (unless you were constantly within range of the DakNet equipped vehicles), it was able to aid local small-holders, shop keepers and families to conduct financial transactions and maintain e-mail correspondence.

Since then, multiple studies have investigated the potential of data mule or message ferry networks on both land and underwater environments [90, 110]. Further studies have investigated different wireless routing protocols [4, 109] but not the effects of vehicle (i.e., message ferry) routing strategies. Routing of the vehicles has been investigated for grid networks by [44] but not for realistic road networks with sparse sensor coverage.

Following on from our work (Chapter 3), we aim to address our final research question covering methods of improving wireless network performance across our simulated urban environments. We use the same performance/evaluation metrics, such as Packet Delivery Ratio (PDR), Packet Received Ratio (PRR)

and transit delay as well as the same city road network topology, San Francisco (USA) to aid comparison.

4.3 Simulation Methodology

A bespoke agent-based VANET simulator was developed in order to assess the feasibility of providing both passenger transport services as well as wireless connectivity to sensors distributed across an urban area. Although we refer to our simulated AVs as agents this is not a strict definition as vehicles driven by humans following a (connected) route guidance system would technically suffice assuming they have the requisite wireless systems for V2I and V2V message exchanges (i.e., with sensors and other vehicles respectively).

In all simulation results presented in this chapter, all sensors aim to communicate with all other sensors (referred to as an All-to-All scenario). The All-to-All scenario was selected as it was considered the hardest use-case for the CAV fleet due to the sheer number of messages required to be collected (an All-to-All sensor communication scenario scales in a quadratic fashion with the number of sensors present) and disseminated across the entire urban area. Therefore, a key feature of our simulator, which differentiates it from our previous work in Chapter 3, is the ability to route passenger trips via sensors with differing priority/weighting attached to either objective.

For example, in extreme cases where high levels of PDR are required in a short period, we envisage a higher weighting assigned to (road) edges with many or nearby sensors in order to incentivise passenger transporting CAVs to drive past as many sensors as possible (albeit at the cost of increasing average passenger trip distance/times). Assessing and understanding the cost/benefit relationship of this multi-objective optimisation problem will be the focus of the chapter. To assess the robustness of such an integrated transport-wireless service our simulator allows for varying trip demand models (uniform random, 'tidal-in', 'tidal-out'), varying the sensor density and location, CAV fleet size and weighting assigned to each objective.

4.3.1 Simulated Urban Environment

Using real, openly available datasets (courtesy of OpenStreepMap [65]) an 8 by 8 kilometre square was downloaded, filtered to keep only necessary features such as 'vehicle-friendly' roads (and their respective directionality) and intersections (see figure 4.1). For road networks discussed and simulated in this chapter we used the OMMnx Python library [7] to filter out unnecessary OSM data points and to reduce the complexity/memory footprint of the overall road network. The OMMnx library[1] allows the user to convert raw OSM data into a Pythonic

[1]Note whilst the OMMnx library [7] was excellent for extracting road networks from OSM datasets, it has been developed in such a manner that it relies on an exhaustive list of dependencies (i.e., other open source libraries). An example of a key OMMnx library dependency is Pandas [56]. At the time of writing, the Pandas library was not optimised for large datasets nor for efficient executable package compilation (necessary for our large-scale simulations to run on our computing cluster). Unfortunately the Pandas library was developed utilising

FIGURE 4.1: 8 by 8 kilometres of the San Francisco central road map is shown as a simplified graph (using OMMnx's OSM waypoint graph extraction function [7]). The full OSM road network had over 79,000 nodes and 171,000 edges. OMMnx's simplifies the OSM road network by combining multiple OSM way-points on the same road link into a single road link/edge of equivalent cumulative length. OMMnx's simplified road network therefore is able to mostly maintain accurate road geometry but reduces the road graph to circa 23,000 nodes and 70,000 edges, effectively halving the memory footprint needed for our agent-based VANET simulations.

NetworkX [18, 32] readable graph .

In order to reduce memory footprint and look up times for CAV routing problems, the road network was simplified further by removing excess short links (i.e., those not connecting to an intersection and of length less than 10 metres) and replacing them with a single longer link/edge. OSM has an overabundance of short links making up stretches of roads between intersections, often of lengths less than a few metres. The reasoning for the way-point system (adopted by OSM) is to enhance accuracy of the network topology as bends/curves in roads can be accurately represented by slicing the curve into multiple angled shorter segments.

The resultant (filtered and simplified) road network topology was stored as a directed graph. Since the road network is spatially embedded, corresponding geographical positions along any edge and node can be found and used for CAV agent mobility and sensor positioning. Unfortunately, due to lack of data with respect to the road network dataset, it was not possible to have height positions for all road edges and nodes. Therefore, all our simulated environments are essentially planar.

This could lead to some unintended effects such as slightly exaggerating V2V connectivity as vehicles are deemed to be within range (horizontally) but could otherwise be vertically much further apart. In practice it is rare for vehicles to be separated in the vertical direction by over 200m (our wireless communication range limit) whilst being within range horizontally. However, the hills found in many cities (such as in San Francisco (USA)) could instead prevent V2V wireless communication if the vehicles are (for example) horizontally within range and driving towards each other but are positioned either side of a hilltop. In this particular case, the vehicles would conduct a message exchange (in our simulator) but in reality, they would have their LoS interrupted by the physical (hill) peak between them. Again, this is a rare event and can be easily discounted by introducing an extra V2V exchange penalty should this be quantifiable in the future (when better open/public mapping data is available).

4.3.2 Simulated Agents

Connected Autonomous Vehicles

- Behaviour: CAV agents served both passenger trips as well as providing local (V2I/V2V) network coverage in an urban area. At the initial simulation time-step, CAV agents were generated and distributed at start locations of passenger trips. CAV agents serve the first passenger trip following pre-defined route (according to pre-selected α weighting parameter value, see Section 4.3.5). Once the first passenger trip was completed (by

mostly Python (a scripting language) as it's back-end. Consequently, compiling such code to make it portable for multiple simulations to run on different 'workers' was rather inefficient due to the resultant memory footprint of the executable file. For example, a single simulation (utilising the Pandas library) resulted in an Python executable file size of the order of hundreds of MB's. Whereas, identical simulations utilising Numpy [99] as the mathematical processing library (whose back-end is mostly written in C) generated executable files of at least one if not two orders of magnitude smaller with respect to memory footprint.

'dropping off' passengers at their desired location) CAV agents continued moving by searching the road network for other passenger trips to serve. CAV passenger search strategy was relatively straightforward, CAV agents randomly select a node (intersection) in the road network and are routed (according to our α edge weighting parameter) towards it. If along the route they pass a passenger, the CAV agents stop to pick-up the passenger (as if it were a hailable cab) and were subsequently re-routed (in the next time-step) to the passenger's desired destination. CAV agents continued to iterate between serving passenger trips (with defined start-end points and route) and searching for new passengers to serve until the simulation period ended.

- Wireless Information Exchange: Throughout their 'life' CAV agents exchange information with each other (V2V) and with fixed location (typically environmental) sensors (V2I). The wireless model and method of processing message exchanges between vehicles and sensors are described in detail in Section 4.3.4.

- Mobility: On a 'planning-level' CAV agents followed routes with a randomly selected end point until they pass a passenger to serve. At the 'street-level' all CAV agents were moved along an edge (road link) at every time-step according to the pre-set maximum/constant speed. This allowed for fine granularity (within 1 metre) of all CAV agent's positions at every simulation time-step in order to construct accurate distance matrices upon which our wireless (V2I and V2V) communication models rely on. CAV agents had their entry and exit edge/road link times evaluated before they entered/moved to the next edge/road link. This method ensured that CAV agents abide by the First-In-First-Out (FIFO) vehicle traffic criterion. Note, CAV agents positions are limited to link locations. They cannot travel off-road.

Sensors

All sensors were initially placed following a uniform random distribution across the entire simulated urban environment. A distance matrix was constructed to check whether all sensors were within the pre-defined V2I range of a midpoint of a road link/edge. Sensors which fell foul of the criterion were redistributed (again, in a uniform random fashion) until all sensors met the V2I range criterion. All sensors remained at their pre-defined (starting) locations throughout the VANET simulation.

4.3.3 Passenger Trip and Wireless Connectivity Demand Models

Passenger Trip Demand

Over 1.2 million taxi trips conducted during 2017 in New York City (USA) were analysed to ensure we used a realistic taxi-passenger trip demand model [2]. Note the New York City taxi trip datasets are huge, over 14 million trips were conducted in Manhattan alone in a single year, resulting in several GB's of data to download and process. New York City requires all taxis (licensed hailable Hackney carriages as well as online bookable taxis) to record their trip start and end locations, drop-off and pick-up times, the fare charged and the number of passengers undertaking the journey.

We estimated the (routed) trip length distribution by painstakingly routing each trip start and end point on an OSM based road network model of New York City (USA). Given the sheer size of the NYC taxi trip dataset, this required a somewhat complex and time consuming system of dividing up the taxi trip dataset into 'trip-chunks' (i.e., parcels of data containing a thousand or so trip's start and end GNSS positions) before they were sent off to a 'worker' node on our computing cluster. The worker node then ran Dijkstra's routing algorithm[3] over our extracted and filtered New York City road network in order to accurately estimate the individual taxi trip lengths.

Once trip lengths were estimated, the data was collected from the 'worker' nodes before plotting the resultant New York City taxi trip length probability distribution, see Figure 4.2. We rejected taxi trips where start and end points could not be matched to the nearest road segment (our self-imposed filter range of 200m). However, in practice this was extremely rare as less than 0.1% of taxi trips were rejected for not meeting this criterion, likely due to poor GNSS signal at the time the trip was conducted.

For our simulations, passenger trip demand was generated by randomly selecting two points in the simulated urban environment before finding the nearest (in terms of Euclidean distance) corresponding nodes (i.e., road intersections) in the road network graph. Once the Origin-Destination (OD) pair of nodes were selected, a Dijkstra algorithm was run to find the 'shortest path' (initially based on real-world road edge length data) between the OD pair. This was repeated for however many passenger trips were necessary for the simulation.

An example of passenger trip length distribution is shown in Figure 4.4 (page 90). Note the similar skewed shape distributions with peaks around 4000-5000m (routed length) with long flat tails from 10km onwards for both the real New York City taxi trip dataset (dotted black line) and our randomly generated sets.

For simulations where we investigated the sensitivity and effects of varying the road graph edge weights with our parameter, α we generated multiple 'shortest-path' routes between the same randomly selected OD node-pair.

[2] The New York City Taxi and Limousine Commission publishes yearly datasets for all taxi trips conducted within the metropolitan area datasets are open and freely available [89].

[3] Since we are concerned with investigating the efficacy of our proposed edge-weighting method to improve PDR, precise evaluations of total trip lengths/costs were necessary. Therefore, routing algorithms that rely on a meta-heuristic cost function were not selected.

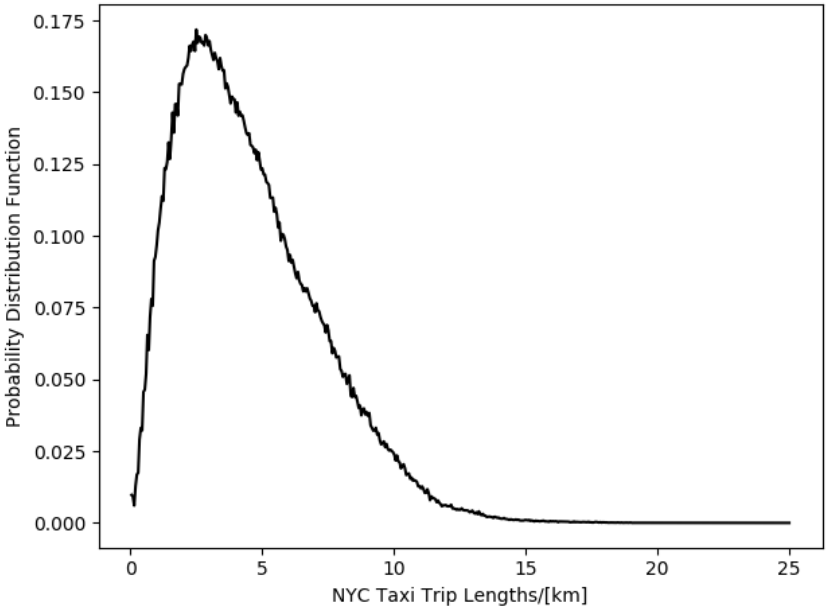

FIGURE 4.2: Probability distribution function of New York City (USA) taxi (routed) trip lengths conducted. In total we analysed over 1.2 million taxi trips conducted in 2017. Note, our filtering and analysis method rejected less than 0.1% of taxi trips due to inaccurate/non-plausible GNSS data readings. Finally, bin widths were (uniform and) set to a routed length of 500 metres. We used the OMMnx [7] software library to extract and filter OSM [65] road network dataset of New York City (USA). In order to route taxi trips we used the NetowrkX graph processing library [18].

'Shortest-path' in this case refers to finding the route which minimises the sum of edge weights (based on some partially-normalised function involving our model parameter α, see section 4.3.5) between the OD node pair.

In some simulations, passenger trip OD node pairs sometimes cannot be connected. Fortunately this rarely occurs and is often a result of poor road network filtering, i.e., where some lesser important roads might not be connected to the main 'arteries' of the road network. In these rare cases, a new OD pair was selected until a suitable route was found.

For 'tidal-in' simulations we aim to capture a typical morning rush-hour scenario whereby a large proportion of vehicle traffic is heading from all parts of the urban area to the Central Business District (CBD). To achieve this type of trip demand all the destination nodes were located within a subset area of the urban environment being simulated. In the case of San Francisco, the CBD was defined as being roughly a 4 by 4 km square covering the top right-hand corner of the simulated area (shown in Figure 4.1). Whereas the origin nodes can be anywhere within the simulated environment and vice-versa for 'tidal-out' simulations (i.e., typical evening rush hour traffic).

Wireless Sensor Connectivity Demand

All sensors aim to communicate with all other sensors throughout the simulation (i.e., an All-to-All scenario). An All-to-All scenario allows for understanding of the problem or 'hard-to-connect' sensors/geographic areas of the city. Furthermore, an All-to-All scenario allows us to select prime locations for central control units, where data could be (relatively easily and swiftly in terms of transit delay) collected and analysed from all over the city (imagine a local government or city administration office type of edifice). Finally, the All-to-All scenario neatly incorporates multiple One-to-All and All-to-One simulations. By covering all of these simulation scenarios we can show how multiple communication demand objectives could be fulfilled. For example, pushing a sensor kernel software update to all sensors (One-to-All) or collecting weather/climate/air quality information from all sensors (All-to-One). Note, in our simulations sensors cannot communicate with each-other or with the Internet, instead sensors only communicate with passing (within V2I range) CAV agents. Sensors store messages from other sensors, delivered via CAV agents.

4.3.4 Wireless Communication Models

Vehicle-to-Vehicle (V2V)

A maximum allowable communication V2V range of 200m was imposed in all simulations between CAV agents (as per the previous chapter) given recent V2V experiments conducted in the city of Bristol [92], 200m was observed as the maximum reliable communication range for successful V2V transmission in an urban environment. The V2V range increases to 300-400m in rural areas with less physical obstacles and wireless interference from other nearby radios.

Should CAV agents be within the maximum V2V communication range, their likelihood of successful message exchange was given by a negative exponential function. The successful V2V exchange probability distribution was drawn from the observation that as vehicles were further apart in cities (in terms of Euclidean distance), the likelihood of having an obstacle impeding LoS wireless exchange increased. This observation was confirmed by plotting the likelihood of having LoS from our real-world vehicle traffic (taxi trace locations in a city) and building footprint dataset (see Figure 3.9, page 61). Since we know the position of every vehicle at every time-step (via our linear interpolation strategy) we were able to check whether pairs of vehicles (those less than 500m apart, Euclidean distance) had LoS.

Results of our investigation into likelihood of LoS given a separation distance as well as our suggested fitted exponential curve is shown in Figure 4.3. Computationally, this model is lightweight as it negates the requirement to store a large amount of building footprint data in memory. This method also reduces the volume of (also computationally expensive) calls to the PostGIS city-wide building footprint database to run a 'in-polygon check' if the communicating pair of CAV agents actually possess LoS. In our simulations communicating pairs of CAV agents, i.e., those that were within range and were deemed 'fortunate' enough by our V2V transmission success curve, were guaranteed exchange message success. Note, we randomised the message exchange processing order at every simulation time-step.

All our simulations were based on the assumption that a version of the IEEE 802.11p protocol was implemented for all wirelessly communicating agents. We chose this protocol to base our simulations on since it is very likely to be introduced as a minimum communication requirement for future production vehicles including CAVs.

The IEEE 802.11p protocol was designed for rapid dissemination (and gathering) of trajectory data of vehicles whilst in motion, hence the rapid message broadcast rate of 10Hz and low latency. Furthermore, the protocol has been designed to allow for a maximum of 2000 in-range communicating nodes. This is the extreme demand case where vehicles have to reduce their broadcasting rate in order to avoid flooding the medium, for example at a large four-way by four-lane intersection. These extreme channel contention scenarios rarely occur (if at all) in our simulations as our simulated fleet size does not exceed 2000 (due to available computational resources at the time of writing) and our CAVs were generally spread across the simulated 64km^2 urban environment. Consequently we do not model this variable.

Vehicle-to-Infrastructure (V21)

A maximum allowable communication range of 100m was imposed in all simulations between CAV agents and fixed position sensors agents. No exponential LoS model was required due to the overwhelming likelihood of having LoS given the shorter maximum wireless communication distance (compared to our V2V wireless model).

V2I communication exchanges were processed in a randomised order just after the CAV agents had their positions updated for the current simulation

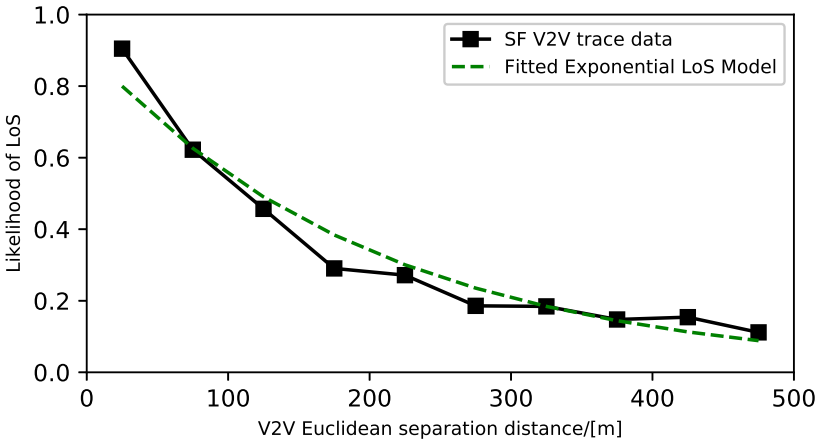

FIGURE 4.3: 96 hours of taxi trace data (over 2M datapoints) in San Francisco (USA) were analysed by finding all pairs of taxis with a Euclidean separation distance of less than 500m and sorting the results into 10 bins (of uniform 50m widths). For each bin, the proportion of pairs of taxis which have LoS (evaluated using our detailed building footprint and road network datasets) are plotted in black. To reduce memory footprint and computationally expensive calls to our LoS database (when running our V2V message exchange model) we used a fitted exponential curve (green dotted line) to the San Francisco (USA) LoS dataset.

time-step. Note this is prior to V2V exchanges being processed in the same time-step. V2I exchanges follow the same method as those of V2V exchanges in that each exchanging pair has a 'set.union()' operation performed. At the end of the exchange both sensor and CAV agent will have an identical message set.

We base our V2I communications model on the assumption that we were simulating short range Wi-Fi or even just using 802.11p (as our V2V systems assumes) links. Therefore, given the large average distance between sensors (in our simulations there were between 4-63 sensors/km^2) and the low average density of our CAV fleet size ($4-31$AVs/km^2) we do not model interference or wireless channel contention for V2I exchanges.

4.3.5 Simulation Model Parameters

Our α edge weight parameter (which varied between zero and unity inclusive) was developed such that routing between any two nodes (road intersections) in the road graph could be altered in favour (or against) driving past edges/road links with more/less sensors. Higher values of α increased the likelihood of driving past sensors on routes by lowering their respective edge weights (see Equation 4.1). We set a search limit of twice the length of the shortest (real distance) path route when α was set to unity. Whereas, an α value of zero indicated that no priority was given to edges closer to sensor locations, instead any passenger trip will always select the shortest route in terms of real-world edge lengths. A standard Dijkstra (shortest-path) algorithm was used for all

passenger trip routing. Our edge weight $E_{(i,j)}$, for connecting nodes i and j (in a road graph), is given by the following equation:

$$E_{(i,j)} = \frac{1}{R_L}\Big[(1-\alpha)L_{(i,j)} + \alpha D_{min}\Big] \tag{4.1}$$

where R_L is a range/length constant used to normalise the edge weight, set to half the length of one side of the road network area (4km) in question, α is our edge weight parameter, varied between 0 and 1 (inclusive), $L_{(i,j)}$ is the physical length of the road segment connecting nodes i and j and D_{min} is the Euclidean distance to the nearest sensor from the edge midpoint.

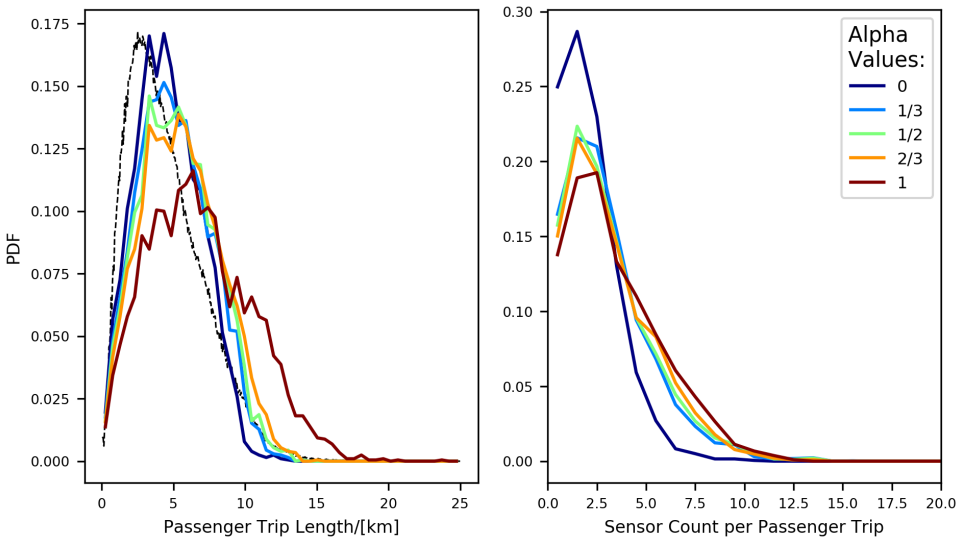

FIGURE 4.4: How our edge weight parameter α, influences our passenger trip length (left) and sensor count (right) probability distribution functions. α values are shown in the Figure legend. Note, the New York City (USA) taxi PDF reproduced from previous Figure 4.2 is plotted in the background (black dashed lines) for reference and to act as a 'reality-check' for our simulated demand model.

4.3.6 Simulation Runs

To compare performance of our (simulated) CAV agent fleet given different simulation inputs and model parameters, multiple simulation runs were necessary to ensure reliable results. Our system was designed to be light-weight enough that once compiled, simulations could run individually on a node (or server blade) within a large computing cluster environment.

Once each simulation (or job) was completed, a separate script collated the results and plots averages of the various result parameters of interest. To aid comparison between parameter variations, all passenger OD locations were kept constant, only the pre-generated routes varied depending on what edge-weighting function was being investigated. Initial parameter values are shown in Table 4.1.

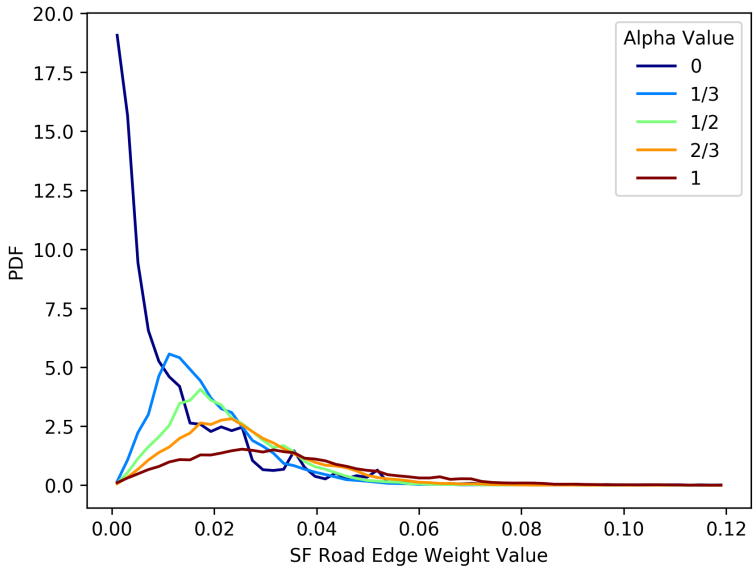

FIGURE 4.5: San Francisco (USA) road edge weight probability distribution functions for varying α values. Over 69,000 edges with 1000 sensors were sampled with 60 uniform width bins. Results are plotted at mid-bin values for all sensors densities. Note, $\alpha = 0$ corresponds to edge weights based purely on physical edge length whereas $\alpha = 1$ corresponds to edge weights based on the Euclidean distance to the nearest sensor.

TABLE 4.1: Agent-based VANET simulator model parameters and values.

Parameter Description	**Value Range**
Number of CAV agents	250–2000
Number of Sensor Agents	250–4000
Number of Pre-Generated Passenger Trips	4000
Maximum V2I Range/[m]	100
Maximum V2V Range/[m]	200
Passenger Trip Detection Range/[m]	100
CAV Velocity/[m/s]	5.56 (20 km/h)
Simulation Time Step/[s]	1
Simulation Duration/[s]	3600

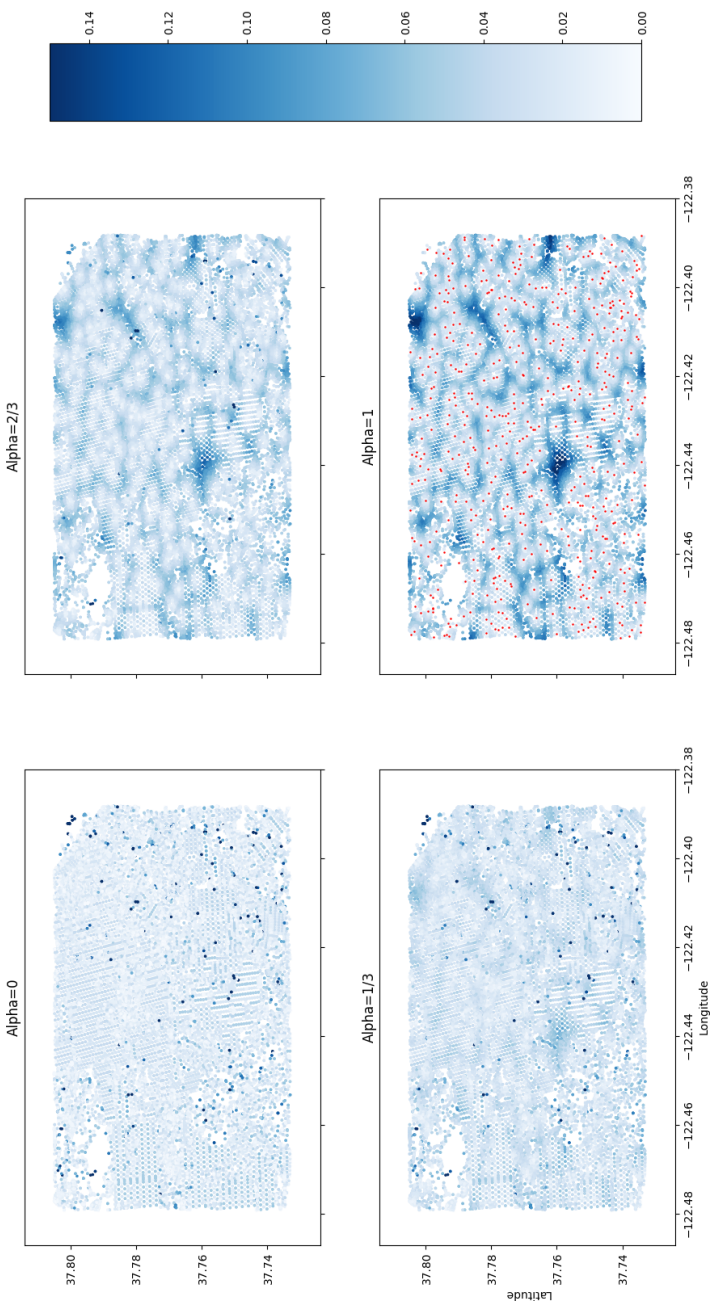

FIGURE 4.6: San Francisco road edge weight spatial distribution for varying α values. Road edge weight values are shown as a colour map (see right hand colour bar for edge weight colour scale). Edge weight values are plotted at the midpoints of their respective road edges. Sensor locations (red points) are plotted for one graph (bottom right). As α is increased, the disparity (in terms of edge weight values) between roads that have sensors and those that do not increases, as evidenced by the darker 'patches' in the right hand-side plots (where sensors are not present). This range of edge weight values further incentivises CAVs to avoid roads without any sensors (in communication range).

For each simulation, multiple runs were conducted to properly assess performance and robustness of our system and for us to obtain meaningful averages and distributions of results. For example, a simulation where we varied our edge weight parameter α and sensor density (such as in Figures 4.7 and 4.8), we conducted 50 runs for each parameter combination, yielding over 1500 separate results to analyse and plot.

4.4 Results

Our aim was to investigate the relationship between providing wireless connectivity to fixed sensors scattered across an urban environment whilst simultaneously transporting passengers to their desired destinations. To better understand this relationship and the difficulties involved regards optimising the system we devised a set of simulations to analyse.

4.4.1 Full Alpha Range

Initially, to get a sense of the our proposed system, we fixed the CAV fleet size (to 2000 vehicles), passenger and wireless demand models and only varied our road edge weight parameter α from 0 to 1 and sensor density from 4 to 60 sensors/km^2 (i.e., 250 to 4000 sensors in absolute terms) across our simulated urban environment (an 8 by 8 km square section of central San Francisco (USA).

Figure 4.7 shows how the PDR varied with time within the simulation (note, we discretised the system such that CAV positions and V2I and V2V exchanges are processed at every $1s$ time-step) for different sensor densities (only the first four were plotted due to lack of space on a standard A4 page). It is immediately clear that the majority of improvement, in terms of PDR, occurs at the lower end of our edge weight parameter α range (generally between 0 and 0.5).

In all cases where α was not set to zero yielded improvements both in the final PDR value and the time taken to reach the 0.90PDR threshold value. This is clearly shown in Figure 4.8 where we plot the time taken to reach 90% of the total possible packets being delivered at the sensor end. In all of our simulation results we never achieved perfect PDR values, in fact the system rarely exceeded 0.95PDR for any given sensor density and α value. Even when the simulations were allowed to run for longer (we initially only allowed simulations of one hour due to limits of computational resources available at time of writing) PDR values rarely reached above 0.95. We discuss why this could be the case later on in the chapter.

Whilst increasing our road edge weight parameter α up to 0.5 certainly improved performance both in terms of PDR at the sensor end as well as the end-to-end delay of the system for most sensor densities, the same cannot be said for α values of zero, i.e., where passenger trip routing was purely based on physical road network distance. Curiously for extreme sensor densities of roughly 4 and 64 per km^2 it took 300 and 250 seconds respectively to reach the 0.90PDR threshold. Whereas for sensor density values in between the two extremes, the average time to the same PDR threshold level took longer, roughly between 300 and 450 seconds.

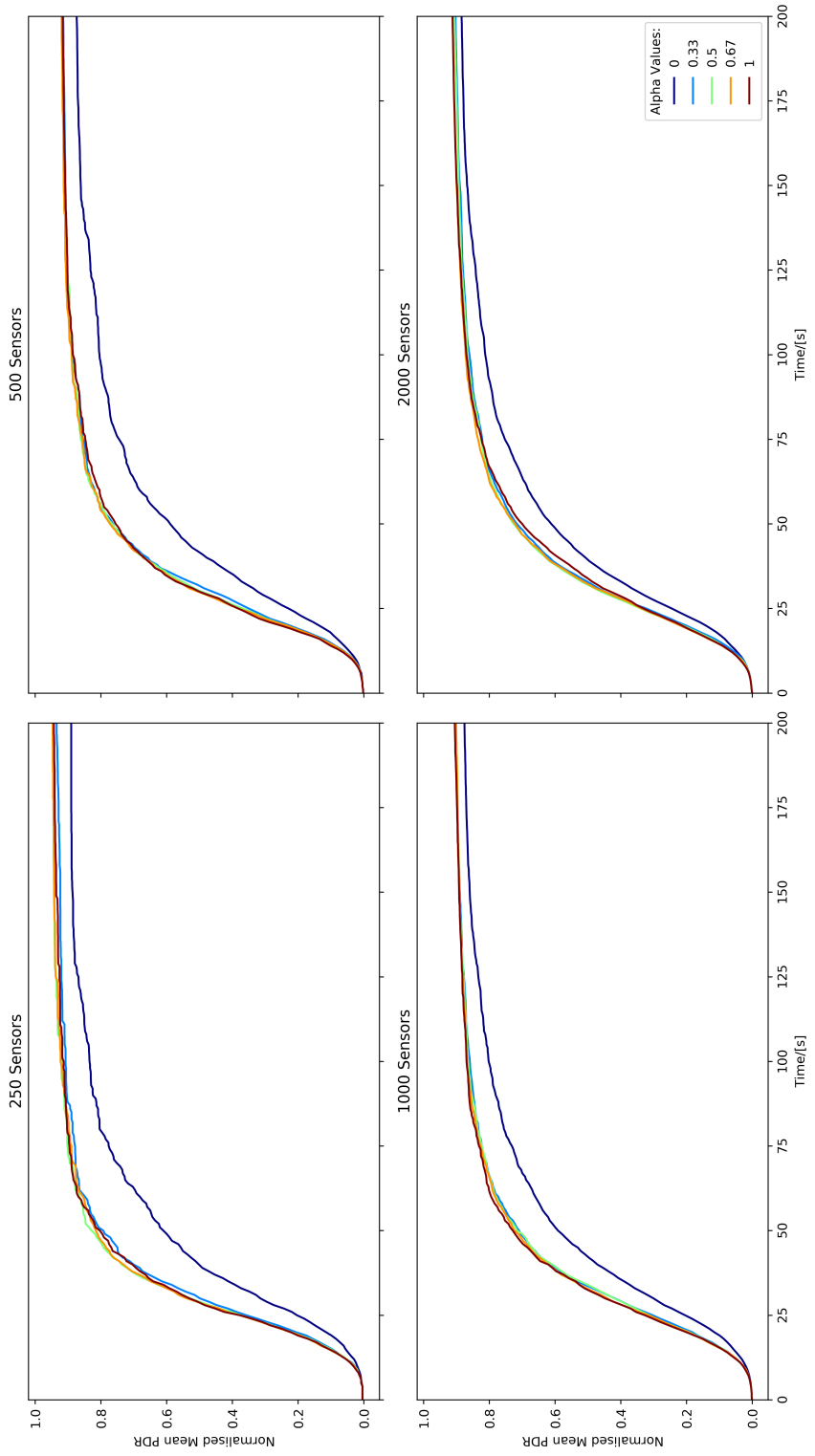

FIGURE 4.7: San Francisco simulation PDR results for various sensor densities and α values. Note, number of passenger trips, taxis and map were kept constant to aid comparison of performance. Any increase in α value above zero yields an improvement in terms of PDR, however, the majority of the improvement is achieved with low values of $\alpha < 0.5$.

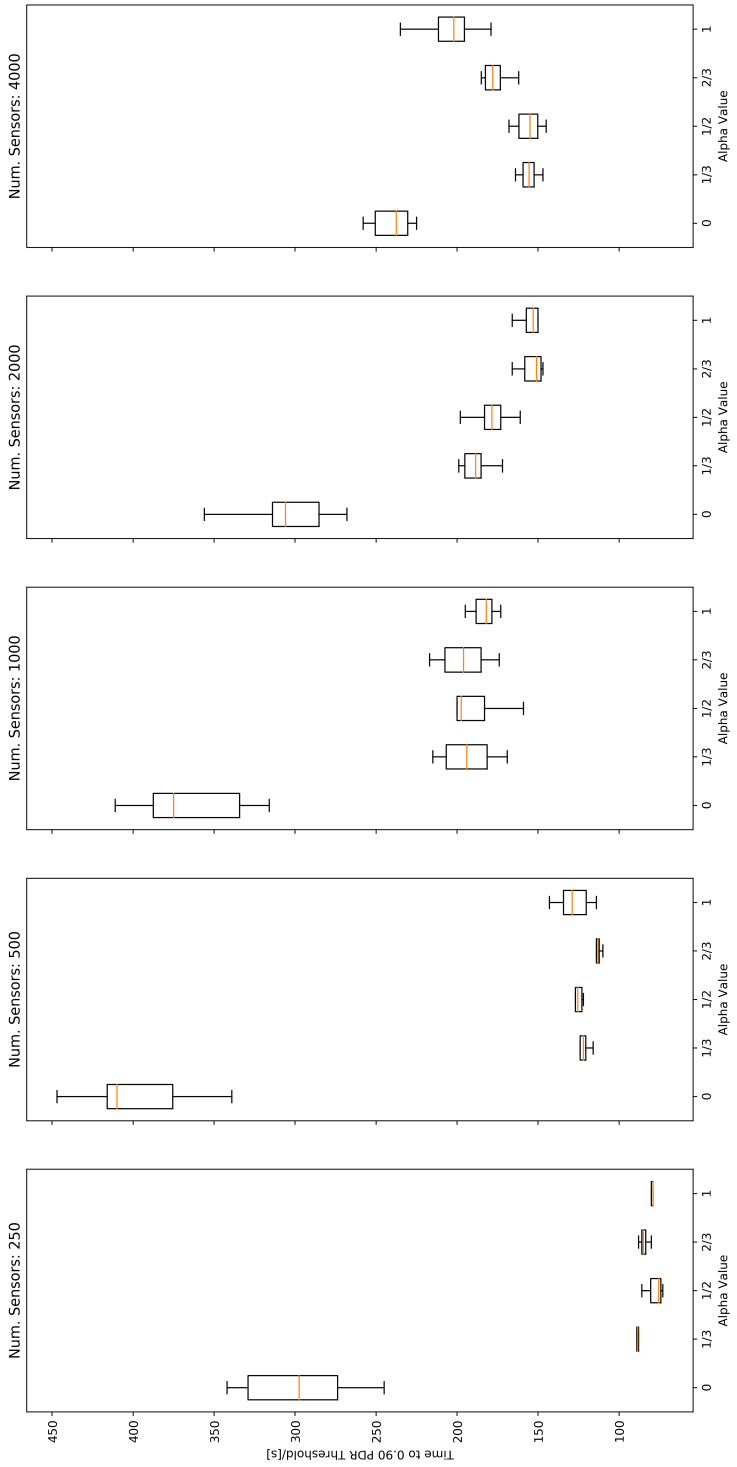

FIGURE 4.8: Time taken to reach mean 0.90PDR threshold for various sensor densities and α values are shown as box-plots and whisker diagrams. Note, the 'box' represents the inter-quartile-range (IQR, middle 50% of the data), the orange horizontal lines denote the median values and the whiskers are 1.5 times the IQR. Number of passenger trips, taxis and map were kept constant. In the many cases, increasing α reduces the median time taken to reach the 0.90PDR threshold. However, as sensor densities were increased the time taken to reach the PDR threshold increased.

Given sensor positions and passenger trips were fixed for each simulation run, the only real difference were the number of sensor messages that needed to be transferred across the network. Since this we simulated an All-to-All communication scenario, the total number of messages scales in a quadratic fashion with the number of sensors present. With very high sensor densities (such as our maximum of 64 per km^2) it was very likely that a simulated CAV agent would pass down a road (or graph edge) with a sensor (regardless of its routing strategy). Therefore the spread (average time to 0.90 of PDR) between the α values is generally smaller for high sensor density simulations than those with fewer sensors.

4.4.2 Focused Alpha Range

To further investigate our proposed system dynamics, we conducted another series of simulations, again fixing passenger trip numbers and CAV fleet size but with lower (more focused) values of α, ranging from 0 to 0.2. PDR results are plotted in Figures 4.9, 4.10 and 4.11.

Again, increasing α results in faster PDRs across the range of sensor densities simulated. Results for the time taken to reach 0.90PDR were particularly encouraging given the almost linear improvements across the range of sensor densities as α was slowly increased (see Figure 4.10. However, certain results appeared to be counter-intuitive, for example when 500 sensors (circa 8 sensors/km^2) were simulated on average it took longer to reach our PDR threshold for all values of α when compared to higher sensor densities. In this case, it is suggested that higher sensor densities allowed for greater chance of passing by a sensor on any given 'shortest' path in the road network. Not only were the median values higher, but the inter-quartile ranges are also greater (nearly three times greater deviations than at other sensor densities).

We also plotted the time taken to reach 0.95PDR threshold values (see Figure 4.11). Results for simulations where we generated 1000 sensors seem almost non-sensical as very few simulation runs achieved the higher PDR threshold, generating under populated box-plots. Note, in the case $\alpha = 0$, no simulation run achieved the threshold within the simulated hour.

4.4.3 Varying AV Fleet Size

So far, only sensor density and α were varied in our simulations. To further investigate performance of our proposed system, we fixed the number of sensors at 1000 (circa 16 sensors/km^2) and varied instead the CAV fleet size and α (using the narrower and lower range $\alpha = [0 - 0.2]$). Varying CAV fleet size and subsequent simulated PDR results are plotted in Figures 4.12 and 4.13.

Reducing CAV fleet size had an enormous effect on PDR values. Results plotted in Figure 4.12 for CAV fleets of less than 500, final PDR values barely reached 0.80 within 200s for the range of simulated α values. Given that CAV agents provided the sole means for message gathering and exchange, reducing the fleet size drastically reduced the so called 'search-area' per simulation time-step. For example 1000 AVs can in theory search 1000 edges (to an extent), or a

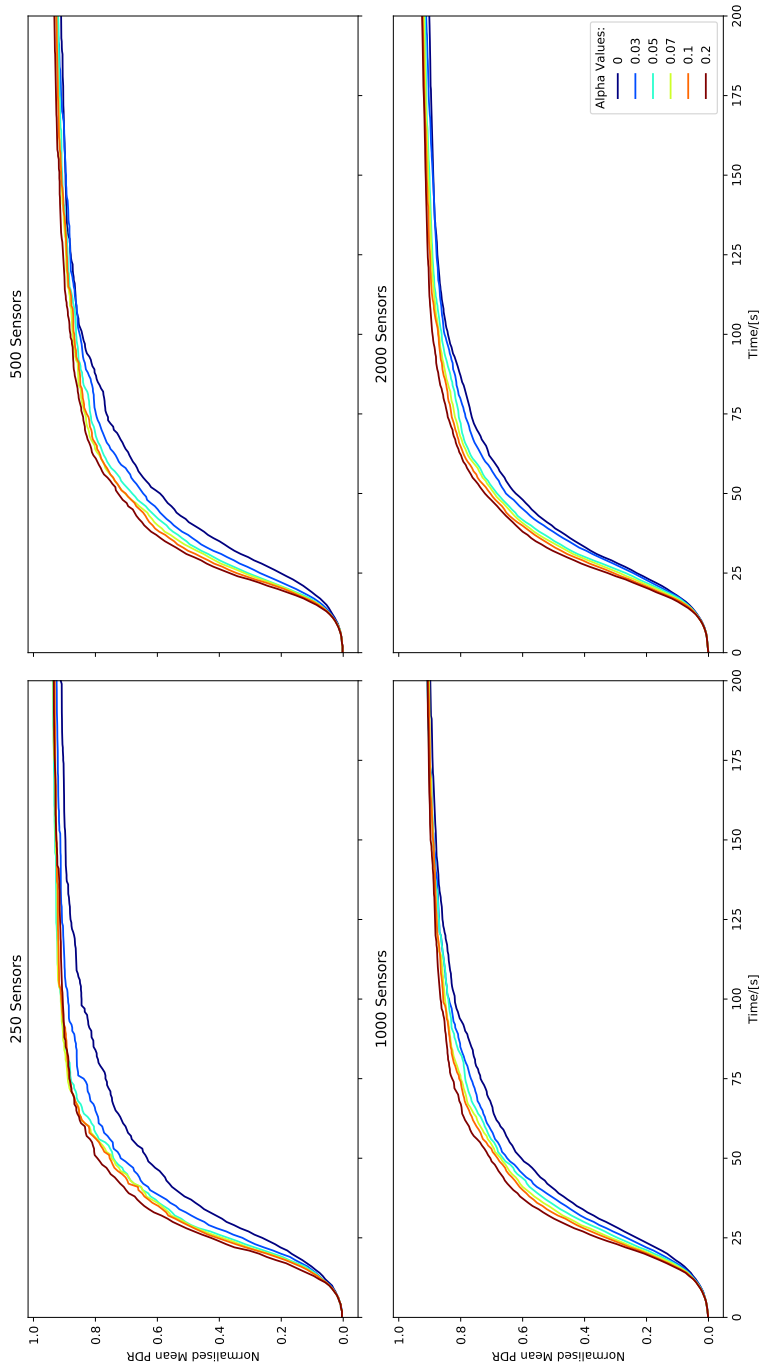

FIGURE 4.9: San Francisco PDR results at the sensor end for various sensor densities and focused α value range. Note, the small improvements in the rate at which sensors messages are disseminated across the network as α is increased to 0.2. Final PDR values remain largely unchanged with respect to α or sensor density. Furthermore, the number of passenger trips, CAV fleet size and road network map were kept constant.

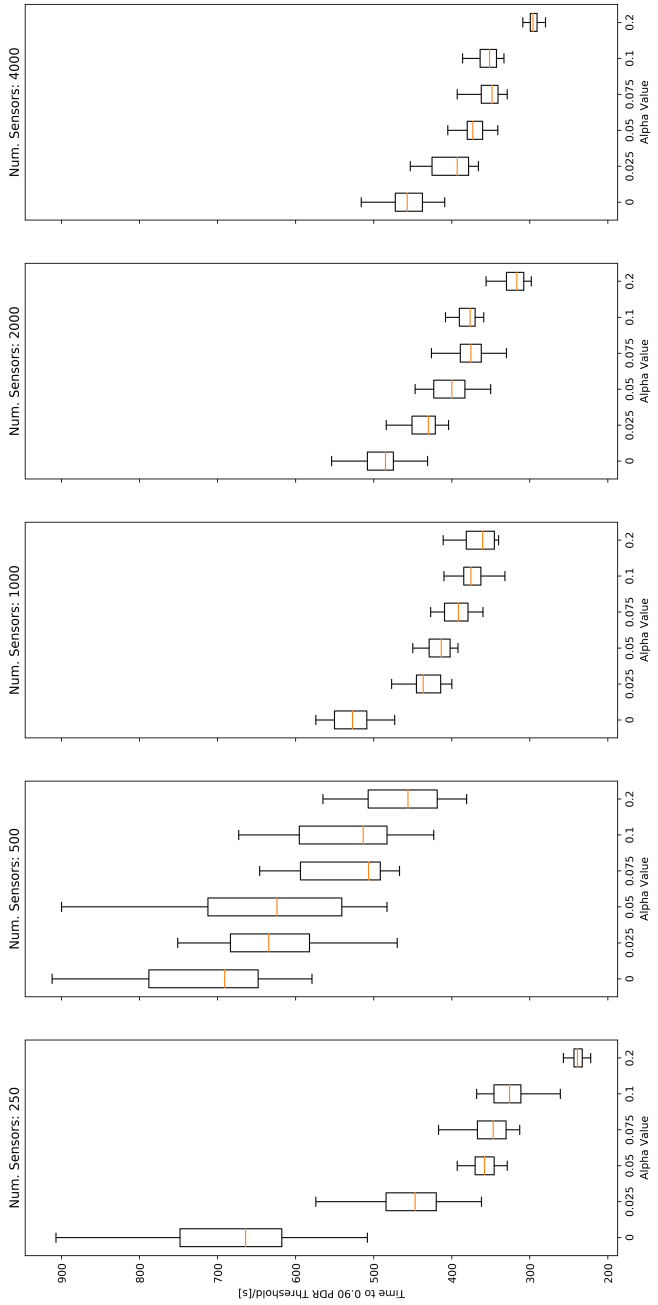

FIGURE 4.10: Time taken to reach 0.90PDR threshold for various sensor densities and focused α range are shown as box-plots and whisker diagrams. The box shows the IQR, the orange horizontal lines denote the median values and the whiskers are 1.5 times the IQR. Number of passenger trips, CAV agents and road network topology were kept constant. Note, as α was increased, the time taken to reach the 0.90PDR threshold reduces across all sensor densities simulated. The largest performance gains were achieved at the low sensor densities potentially highlighting one of the benefits of our system when sensors were sparsely distributed across the simulated urban area.

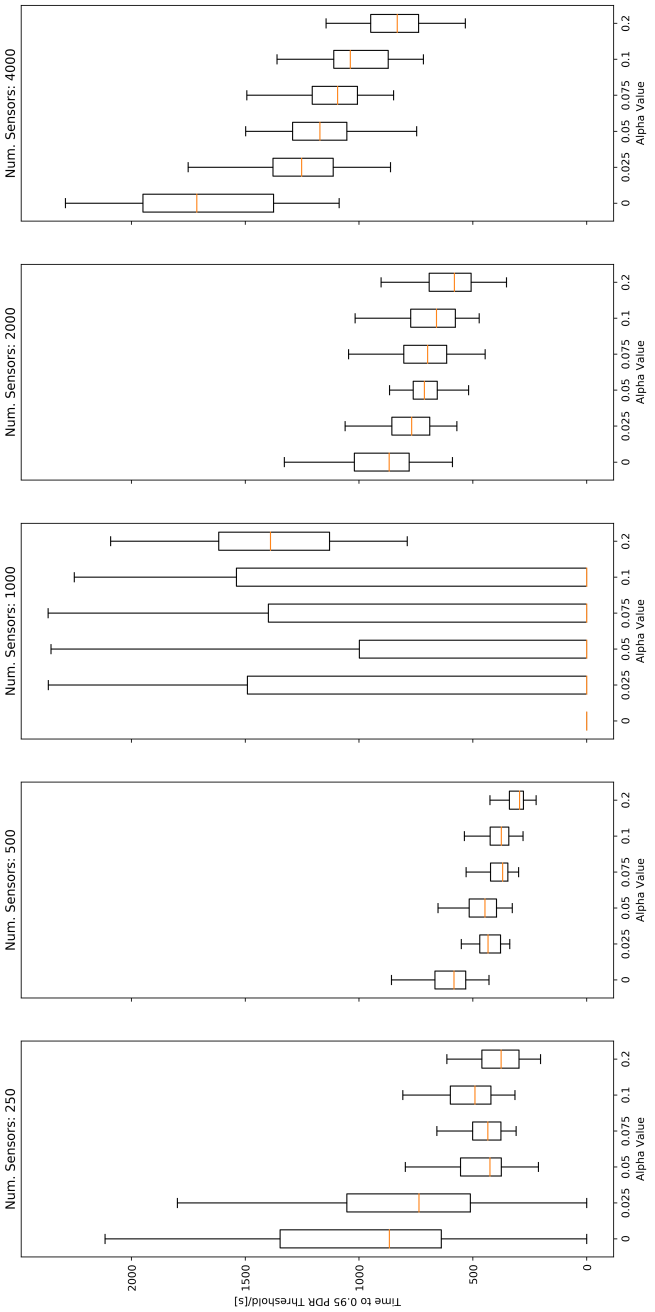

FIGURE 4.11: Time taken to reach 0.95PDR for various sensor densities and focused α range are shown as box-plots and whisker diagrams. The box shows the IQR, the orange horizontal lines denote the median values and the whiskers are 1.5 times the IQR. Number of passenger trips, CAV fleet size and road network topology were kept constant. With a higher PDR threshold, we notice a much greater range of results across most of the sensors densities when compared to the lower 0.90PDR threshold, see Figure 4.10. In certain cases such as a 1000 sensors (circa 15 senors/km^2), the 0.95PDR threshold was rarely met resulting in poorly populated box and whisker plots.

theoretical maximum area of 31 km^2 per time-step compared to roughly 7.9 km^2 for 250 CAVs (assuming, no overlap and a 100m V2I maximum communication range).

With regards to surpassing the 0.90PDR threshold (Figure 4.13) results were fairly consistent across the range of AV fleet size and α. In all CAV fleet size cases, the decrease in time taken to reach said threshold appeared to be fairly linear with increasing α. Improvements beyond 1000 CAVs appeared to taper off, with median values (time taken to reach the 0.90PDR threshold) decreasing from roughly 300s to slightly less than 250s for CAV fleet sizes of 1000 and 2000 respectively. This is in stark comparison to an average of median values of around 1200-1600s for an CAV fleet size of 250.

4.4.4 Varying Passenger Trip Demand

Lastly, we simulated a 'tidal-in' scenario, whereby half of all passenger trips commenced outside but terminated inside of the Central Business District (CBD) of San Francisco (USA). The CBD was defined as the top right hand corner of our network topology (see Figure 4.1), i.e., a quarter (4 by 4 km squared) of the entire simulated urban environment. Our aim was to evaluate the performance of our system with a different passenger demand model. We kept half the trips as uniform random to ensure there was some background traffic (after-all, not everyone works in the city centre). Note, that 'tidal-in' simulations make it harder for CAVs to search a greater proportion of the map. In particular sensors located near to the simulated environment boundaries were adversely affected as they tended to receive even less packets.

Figure 4.14 shows how the PRR and PDR vary with simulation time-steps for α values of 0 to 0.3. In both cases, messages still spread throughout the VANET system fairly rapidly. In terms of PRR, which focuses solely on the messages stored within the vehicles (as opposed to those delivered and stored at the sensor end) 0.90PRR level was reached roughly within 100 to 150s for α values greater than zero. Similarly PDR values rose quickly, on average surpassing 0.90 roughly within 300 to 400s of the simulations commencing.

Results shown in Figure 4.14 suggest our system can cope with this demand model as PDR rates are comparable to our initial passenger trip demand model simulation results. More passenger trip demand models could be tested. However, given the limits of our computational resources and time required to run the simulations and analyse results meant we did not investigate other forms of passenger trip demand, such as 'tidal-out' (i.e., late evening rush-hour where city traffic typically empties from the central area).

4.5 Discussion

Overall our approach to optimising CAV routes to satisfy both passenger trip demand as well as sensor network connectivity seemed feasible. As shown in the results, trip times do increase slightly (see Figure 4.4 but our PDR rates both increase faster (during the simulation, meaning shorter end-to-end delays) and reach higher levels of final PDR (again at the sensor end).

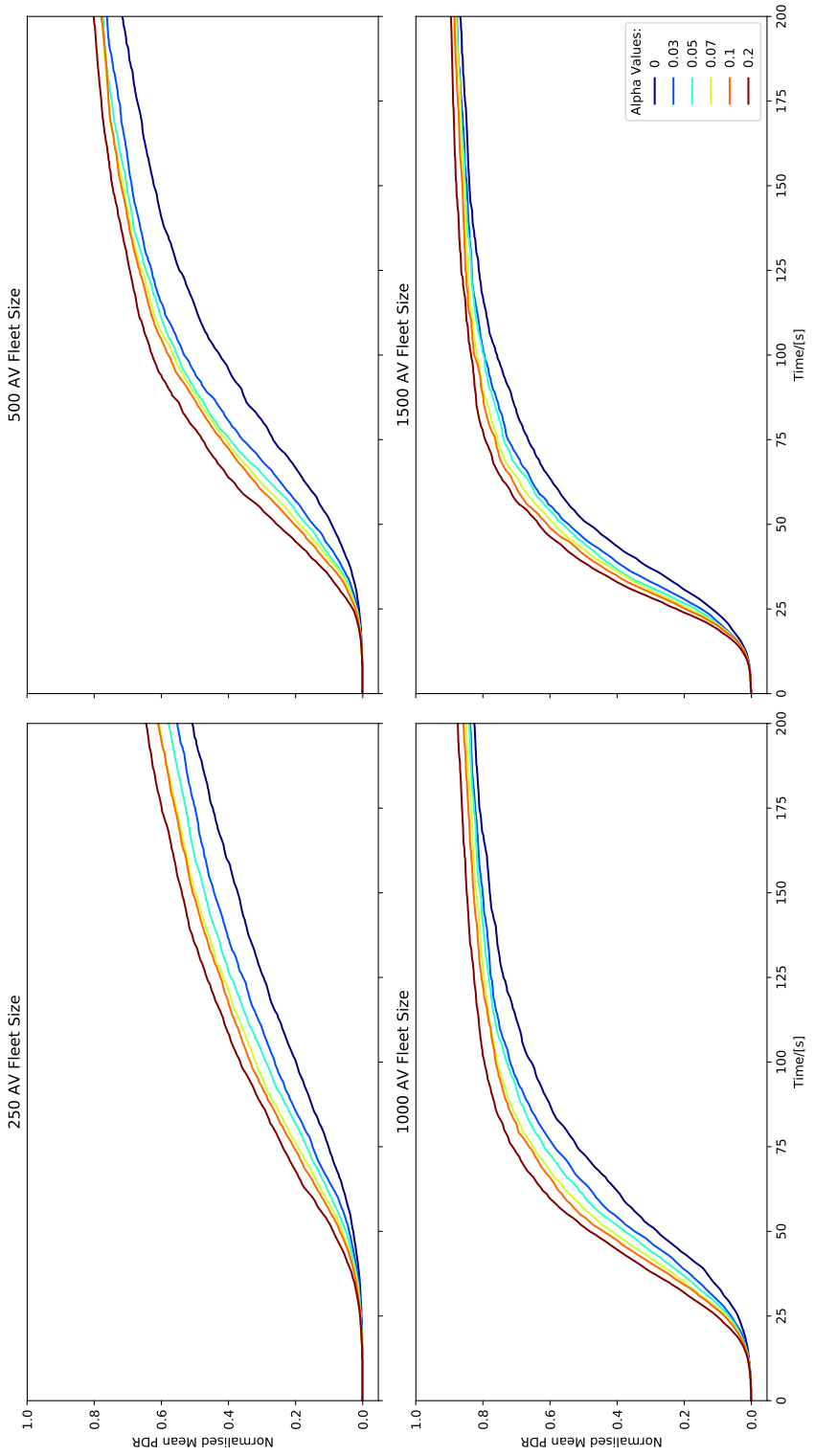

FIGURE 4.12: Mean PDR results at the sensor end for various CAV fleet sizes and focused α value range. Number of passenger trips, sensors (1000 for this simulation) and road network topology were kept constant. Final mean PDR values rarely exceed 0.8 unless more than a 1:1 ratio of sensors to CAV agents is surpassed. Increasing α had a greater impact (in increasing both performance measures) for lower values (< 1000) of CAV agent fleet size.

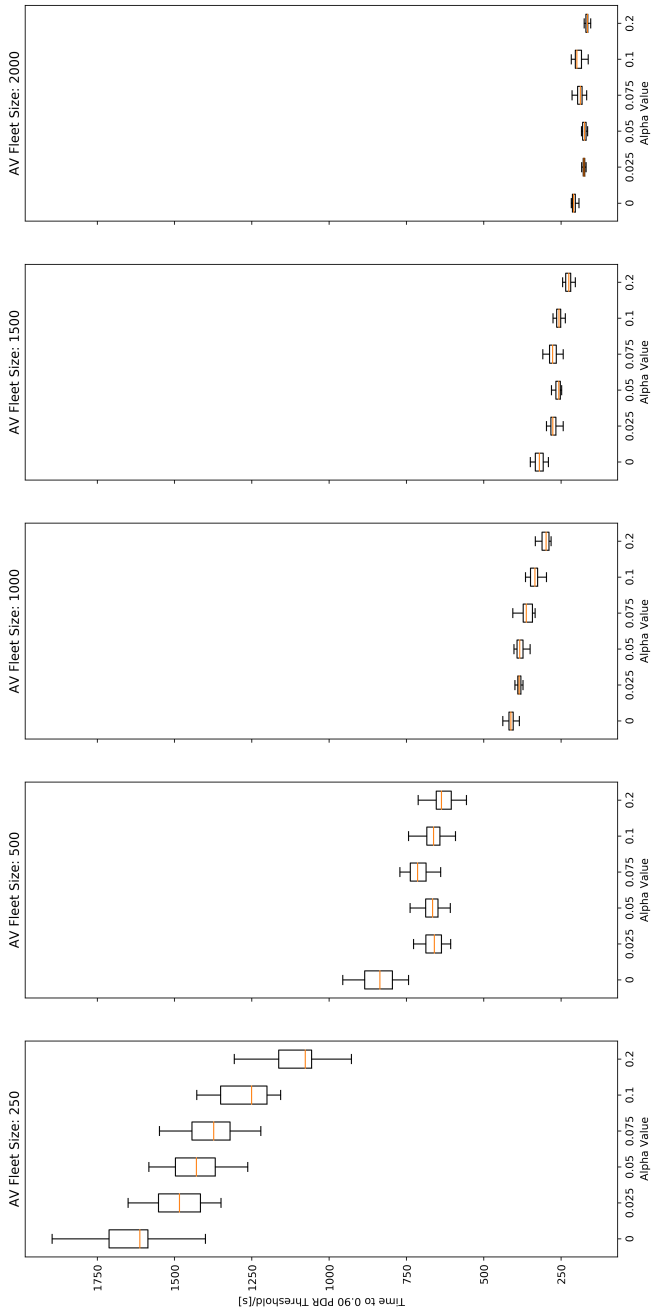

FIGURE 4.13: Time taken to reach the mean 0.90PDR threshold for various CAV fleet sizes and focused α range are shown as box-plots and whisker diagrams. The box shows the IQR, the orange horizontal lines denote the median values and the whiskers are 1.5 times the IQR. Number of passenger trips, sensors and road network map were kept constant. Overall, increasing CAV fleet size decreases the time taken to reach the PDR threshold, from over 1000s to less than 250s for CAV agent fleet sizes of 250 and 2000 respectively. In the majority of scenarios increasing α reduced the time taken, greatest (relative) improvements can be seen with reduced CAV agent fleet sizes.

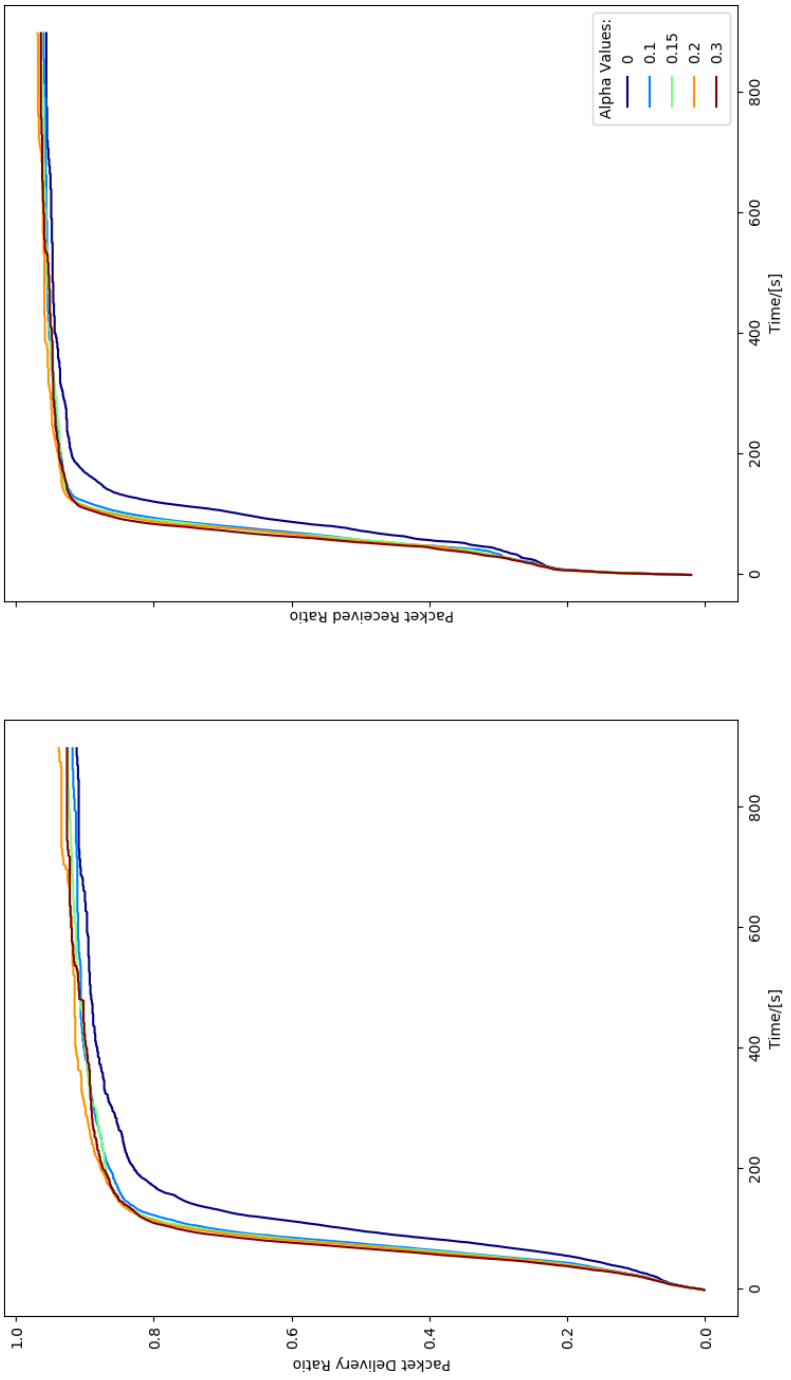

FIGURE 4.14: Mean PDR and PRR results for a 'tidal-in' simulation, whereby half the passenger trips commence outside but finish within the central business district of San Francisco (USA). 2000 CAVs and 1000 sensors were used in these simulations, only α was varied (see figure legend). In either case (PRR or PDR) results were not hugely affected by the change in passenger demand profile. Increasing α had a small impact in improving both performance measures.

Our improvements in terms of PDR values were significant. With low α values we were able to more than halve the time taken to reach the 0.90 PDR threshold when compared to our standard non-optimised system (i.e., when $\alpha = 0$) for most simulated sensor densities. However, even in the best cases the time it took to reach that PDR threshold was circa 100-200 seconds, still rather slow and somewhat un-reliable relative to other forms of wireless sensor network systems such as LoRaWAN (Low Power Wide Area Networks).

4.5.1 Comparisons With Wide Area Sensor Mesh Network Protocols

Regards to comparisons with other types of wireless sensor network protocols/designs such as LoRaWAN (slowly becoming an industry standard for wide area sensor networks [75]) our proposed system may not be as reliable or as fast in terms of end-to-end delay. However, this misses some crucial points. Whilst individual sensors can communicate over large distances (experiments have been conducted in the order of kilometres with LoRaWAN systems) with higher reliability and lower end-to-end delay, LoRaWAN requires sensors to connect directly to a gateway or router, i.e., in a star formation (LoRaWAN does not support packet routing or multi-hop transmission paths). This limits how many sensors can be connected to single gateway due to spectral interference. In short, through simulation, it has been shown that the coverage probability drops exponentially with the growing number of sensors (end devices) [26]. Hence, LoRaWAN relies heavily on having multiple fibre back bone connections placed across the sensor network.

In contrast our system can handle up to a simulated 4,000 sensors with relative ease (see Figure 4.10 without the need for any router/gateways or fibre backbone connections. Furthermore, LoRaWAN does not support down-link transmissions, although this is being worked on [72]. Note, our system treats every node as a source as well as a sink, thus being a true All-to-All simulation scenario, something that LoRaWAN currently does not support.

Other sensor wide/mesh network protocols such as 6TiSCh (IPv6 over Time Slotted Channel Hopping networks) require an extensive optimisation/set-up period before communications can commence [30]. This is necessary due to the time-slotted nature of the protocol which requires all sensors and gateways/routers connected in the wireless network to synchronise and organise a communication schedule between them (to avoid interference with transmitting messages). Clearly as the number of sensors increases, the amount of time required for this initial set-up/optimisation phase also increases. That said, once the optimisation/set-up phase was successfully been completed, due to the rigours of time-slotted channel hopping allows for PDR$>$ 0.99 values, something that our optimised (simulated) VANET systems never achieved.

4.5.2 Poorly Connected Sensors

It was rare in our simulations to reach PDR$>$ 0.95. We ran some simulations for longer on individual machines with greater memory capacity. Whilst these

levels were eventually reached getting anywhere near full PDR values across all sensor densities was not possible. We briefly investigated why this was the case.

To this end we plotted (see Figure 4.15) the locations of the least well connected, or 'poorly-connected' sensors, defined as sensors that achieved less than 0.10 final PDR values against varying α values. It was clear that the poorly connected sensors were those most likely to be located near to the boundaries of our simulated urban environment regardless of α value. This is not the most surprising result given that passenger trips do not cross the simulated environments boundary.

Whilst we could have used an even larger simulated area for passenger trips but then only focused on a central sub-area for our sensor experiments, this would have meant ever larger road networks to be analysed and stored in memory during our simulations. Given we were already passing memory limits on occasion (at least 1% of simulations failed due to lack of available memory on the compute node) we leave this open to future work.

Throughout all work presented in this chapter, in cases where simulations failed, they were recorded and repeated either on local machines (with greater memory capacity) or re-submitted to the computing cluster until the simulations completed successfully. We considered varying our edge-weighting function to prioritise sensors on the boundaries (by generating artificially low edge weights in those cases), however, we also leave this for future work.

4.5.3 Simulation Time-Step

With any discrete simulation, there have to be choices made regarding the time-step length. For all simulation results shown in this chapter we used 1 second as our time-step. Ideally shorter time-steps would have allowed for ever more granular simulations and results analysis. In particular, it worth noting that in the current crop of V2V protocols such as IEEE 802.11p, the V2V message broadcast rate is 10Hz.

Effectively by using a longer time-step we were potentially reducing the possibilities for V2V message exchange per overall simulated time period. However, given the computational resources available at the time of writing, we were limited in further reducing our simulation time-step. Smaller time-steps would have generated more data per fixed length simulation (potentially surpassing our cluster's per node memory limits). Note that at each time-step our agent-based VANET simulator had to evaluate the positions of all CAVs relative to one-another, sensors and passenger trip locations. This process required a lot of distance matrices to be computed all based upon using the Haversine formula for computing surface distances between two points on an Earth-like shaped body.

We briefly considered parallelising a lot of the code, in particular our distance matrix operations. However, after extended experiments on both our local machines and the compute nodes on our cluster utilising the new Python Multiprocessing library [19] yielded small (essentially negligible) gains in overall

FIGURE 4.15: Simulated San Francisco road network topology overlaid with poorly connected sensor locations for varying α values (see figure legend). 4000 sensors were distributed in a uniform random fashion across the simulated urban environment (8 by 8 kilometres). Each sensor was located within V2I range of a road segment. The majority of poorly connected sensors (defined as those with PDR values of less than 0.1 at the end of the simulation) were located near to the boundaries of the simulated road network. Potentially in future simulations, a proportion of CAVs which are not serving passengers could act as temporary 'data-mules' and visit hard to reach sensors at the boundaries of the city.

performance. Given the functions being paralellised and the quantity of computation required, the overheads for tracking which sub-job computed where and when meant that any computational performance gain was essentially negated.

4.5.4 Largest Connected Component (LCC)

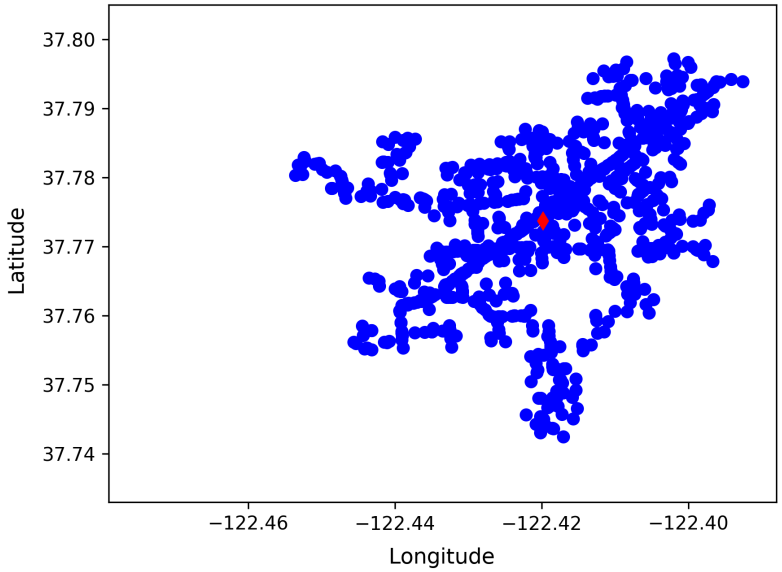

FIGURE 4.16: Spatial distribution of LCC VANET component used for message exchange/propagation investigation. 1001 CAVs form the LCC and their respective locations are plotted in blue, red diamond signifies the centre of mass of the VANET LCC. Note, the centre of mass of the VANET lies along a major central thoroughfare, known as Market Street in San Francisco (USA). This may explain the long diagonal line of connected CAVs stretching roughly from top right hand corner to the centre of the plot.

In our agent-based VANET simulations, sensor messages or packets were stored, carried and exchanged by CAVs in 'epidemic/contact' model type of scheme. However, it was considered, both from a computational efficiency as well as a network stability point of view, how the LCC of the VANET varies throughout the simulation. Figure 4.16 aids visualisation of a typical VANET LCC in our simulations. The plotted LCC consists of roughly half the total CAV fleet size (being simulated) and was mostly spread across Market Street (a main thoroughfare running southwest across San Francisco's CBD). Each blue dot represents a CAV which is within range of another CAV, all forming a connected component, spanning roughly 10-15km^2.

However, as shown in Figure 4.17 the VANET LCC was not very stable. This was expected given the highly dynamic nature of the nodes (i.e., constantly moving CAVs). CAV IDs were compared for each time-step to check for similarities between the LCCs (again processed for each time-step), these are plotted as red squares (Figure 4.17). The LCC VANET similarity coefficient rarely exceeds 0.80 (i.e., roughly an eighth of the CAVs were present in the

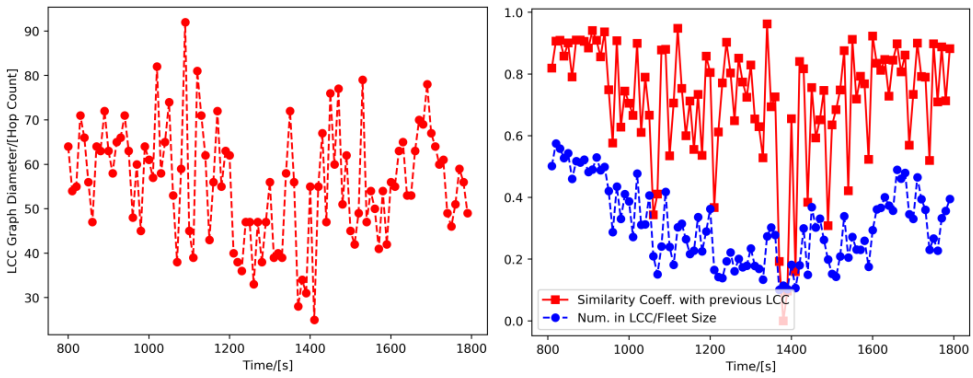

FIGURE 4.17: LCC VANET graph diameter (red dotted line) plotted over simulation time (left) and stability of the LCC is plotted on the right. We use a normalised vertical axis for the LCC similarity coefficient (red solid line) and LCC size as a fraction of the total number of CAVs present (blue dotted line). Due to the dynamic positioning of nodes (i.e., CAVs) LCC stability can vary drastically between consecutive simulation time-steps.

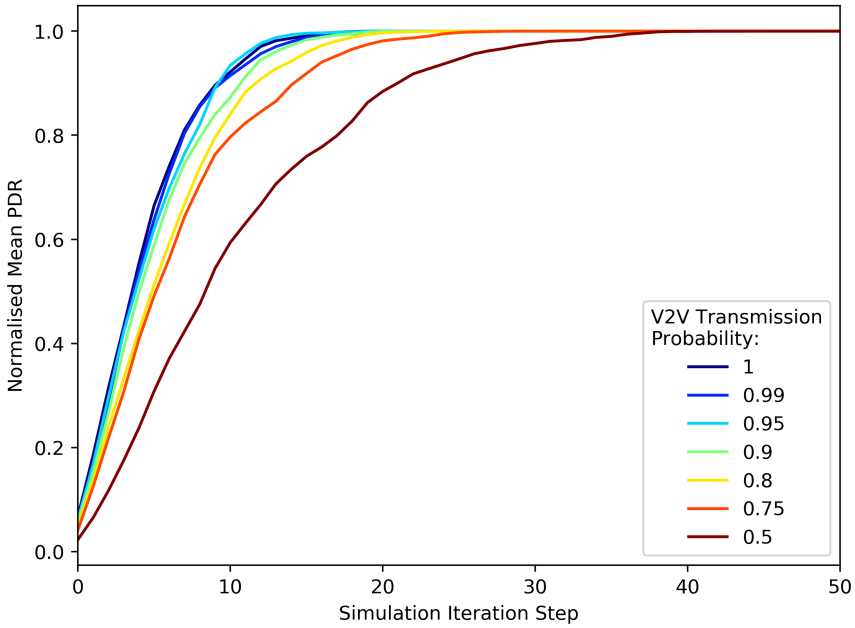

FIGURE 4.18: VANET LCC comprising of 1001 taxis was used for message exchange/propagation investigation. Each CAV (member of the VANET LCC) aims to send and receive packets from all other CAVs, i.e. an All-to-All simulated scenario. At each time step, CAV pairs (those within V2V range) are randomised and their message exchanges are processed according the V2V transmission probability values (see figure legend).

LCC of two consecutive time-steps) and often 'crashes' to effectively near zero similarity.

Plotted on the same axis are the relative sizes of the LCCs with respect to the overall CAV fleet size (in this particular case, there were a total of 2000 simulated AV agents).This further highlights the lack of stability given that the fraction (of CAVs present within the LCC) rarely exceeds 0.5 and in fact lingers for best part of the simulation between 0.2–0.4.

Also plotted (Figure 4.17) are the LCC graph diameters [4] throughout the simulation. The graph diameter gives a rough idea of how long it would take for a single packet to be transmitted across the VANET LCC (i.e., a multi-hop routing scenario).

We briefly investigated how V2V message exchanges could be processed more efficiently by assuming that all CAVs within the LCC obtained the same message set in a single time-step. Figure 4.18 shows how fast messages spread through the VANET LCC assuming different packet loss rates, or V2V transmission success probabilities for each simulated time-step.

For V2V transmission success values between 0.8 and 1 the number of time-steps taken for all CAVs within the LCC to exchange messages between themselves (again, an All-to-All simulation scenario) was less than 20 time-steps. For PDR values of greater than 0.90 less than 10 time-steps were needed. Only with a worst case scenario, such as a packet loss rate of 0.5 per transmission (or V2V hop) do we observe a significant deterioration of PDR across the VANET LCC: taking nearly 40 time-steps to reach full PDR.

Recall that current V2V standards require message broadcasting rates of 10Hz. Therefore, our results suggest it is reasonable to assume that all CAVs within a LCC should be able to exchange short messages within a couple of (real-time) seconds. However, after select trials, computing the LCC from the general VANET requires almost as much processing as simply computing the V2V exchange pairs given the CAV fleet sizes and sensor densities being investigated.

4.6 Conclusion

Overall our system of providing wireless network connectivity to sensors scattered across an urban environment whilst simultaneously serving passenger trips with our simulated CAV fleet performed reasonably well across a multitude of parameter sweeps and variation. We showed that even when varying sensor density (to numbers far greater than simulated with LoRaWAN), CAV fleet size and passenger trips our system was robust within reason. At the very least it out performed standard routing methods (i.e. when $\alpha = 0$. We note that whilst increasing PDR and the time taken to reach PDR thresholds our system does not cost the passenger much in terms of extra trip time, Figure 4.4 shows how little the passenger trip length distribution shifts between low values of our novel edge weight parameter α. Note that in the majority of simulations,

[4] Graph diameter in this particular cases refers to the longest (in terms of hop count) shortest path possible within the LCC or network graph.

$\alpha < 0.3$ suggesting that the extra trip length (or added cost) was even less significant than the distribution plotted in Figure 4.4. With ever decreasing prices in computational hardware, it is not inconceivable that ever larger simulations (in terms of road network topology area and CAV fleet size) could be constructed and run.

With regards to robustness, we briefly tested our system under non-uniform random passenger trip distributions (which were originally inspired by our analysis of over 1M recorded passenger trips in New York City, see Figure 4.2, page 86). Results for our 'tidal-in' simulations suggest that there is little difference in terms of performance (see Figure 4.14) compared to our other passenger demand model simulations.

We investigated the stability of our VANETS in terms of largest connected components of the VANET graph during the simulations. We conclude that whilst it is theoretically possible to speed up V2V message exchanges in our simulator by assuming all CAVs within the LCC have the same message (since messages spread relatively quickly) even when accounting for differing packet loss rates, see Figure 4.18), it was not worthwhile given the computational effort needed to select which CAVs form the LCC at each time-step.

4.6.1 Future Work

There are several avenues for further analysis that could be considered for future work. Firstly, we suggest investigating further the problem of connecting 'hard-to-reach'/'poorly-connected' sensors (those defined as achieving a $PDR < 0.1$). Figure 4.15 highlights the extent of the problem, regardless of α value used, our system struggles to reliably connect sensors at the boundaries of the urban environment. It has been suggested to extend the simulation boundary by routing trips across it and only having sensors distributed in a central 'square' of the entire simulated area. This would allow for more CAVs to regularly penetrate those 'hard-to-reach' areas with sensors. Whilst this would certainly aid in understanding the scope of the problem, we note that San Francisco (USA) for example, is in fact geographically limited on three sides: to the East, West and North there are the Pacific Ocean, Golden Gate Strait and San Francisco Bay respectively. All are (large) bodies of water with only two major bridge crossing points (Golden Gate and Bay Bridge), thus not allowing us to usefully extend the simulated boundary area.

Potentially by using other city road networks, such as Rome, Italy which are not as geographically limited would allow for better understanding of how to connect sensors at the edges of an urban area. We further suggest that a portion of CAVs which are not serving passengers could instead serve these boundary sensors by travelling in those directions (regardless of passenger trip demand). Another strategy could involve varying α such that sensors at the edges are prioritised, however this could have a detrimental effect on the 'core' sensors located in the city centre.

We also suggest exploring systems with ever more accurate or realistic traffic models. Although we do not perceive our system as being unduly simplistic, it

is worth noting that CAVs travel at a constant slow speed of 15km/h throughout the simulation. Certainly varying speed depending on traffic could yield more realistic results. Furthermore, CAVs could be routed such that they avoid vehicle traffic by serving sensors in low vehicle traffic areas. However, this was deemed overtly complex to tackle given we were already reaching the computational limits of the resources available at the time of writing.

Finally our wireless system model relied entirely on distance between nodes irrespective of density in order to evaluate transmission success. In general this was not an issue, given current V2V standards are able to function with up to 2000 vehicles communicating all within range (i.e., at a large four way, four lane intersection). Whilst these large intersections were present in our simulated network topology, we rarely observed such high vehicle densities. Note, our largest connected components rarely had more than half the CAV fleet size. Even in those cases the LCC was spread over several square kilometres. Still a wireless simulator that maybe introduced penalties (such as reducing the V2V transmission success rate) when higher densities of nodes are observed could provide further analysis.

Chapter 5

Conclusion

5.1 Summary of Research

Inner-city vehicle parking systems were initially investigated in order to better understand if it would be possible to reduce the proportion of trip time spent by drivers searching (or 'cruising') for a vacant on-street parking space. Vehicle parking was seen as a near term goal that addressed our first research question, namely, how cities could address near/short term congestion problems. Various studies (such as [81, 83]) highlighted the extent of vehicle-parking induced traffic in city centres. Their research led to the observation that circa 30% of inner-city traffic is potentially made up of drivers searching for a vacant on-street vehicle-parking space.

We conducted a research and exploration study into potential low-cost and 'low-energy' methods of detecting driver parking activity as well as disseminating parking occupancy information. We conducted an experiment whereby volunteers were tracked (using an Android smart-phone and a specifically designed software application) in order to collect vehicle-parking data. We then developed a low-energy consumption method (through our machine learning based system) and tested it via leave-one-out validation methods. We showed that our ParkUs system was able to detect more accurately (high true positive rates), with less energy consumption and shorter detection delays than our (at the time) competitors, see Table 2.4 (page 37) for our final set of results and competitor solution benchmarking.

Unfortunately, due to cost and time pressures we were not able to obtain a larger following of our smart-phone ParkUs application via the online Google Play Store. This meant we were not able to evaluate performance gains in terms of reducing driver search times when parking near to or within the city centre. We leave driver cruising detection research open to future development and public trials. We note that Alphabet (Google's holding company) has since released an updated Android activity detection API [37] which would theoretically allow for easier detection (from a coding perspective, however, not necessarily low-energy). It is likely that Alphabet will introduce vehicle-parking detection features in future Google Map application updates. Since the Android operating system has a user base of over 2B, it seems likely that there is a critical mass of Android users in most cities for such a parking system to operate successfully.

Nonetheless, we were able to briefly investigate the ability of machine learning systems to detect (again from sensors found in a typical smart-phone)

whether drivers were searching for a parking space or simply en-route to their destination. This research was motivated by the observation (from our preliminary dataset) that drivers tend to drive close to their desired destination before searching for parking (most likely in the hope of finding a vehicle-parking space that minimises their subsequent walking distance to destination). Therefore, the data generated by a driver passing their destination and subsequently driving down several other roads before eventually parking could be used to infer vehicle-parking availability (through some probabilistic model/thresholds).

Such a method of collecting more data per user trip would reduce our need to have a large uptake of users to generate near real-time vehicle-parking availability data for a given city precinct. Our novel method of correlating disparate time series data windows in order to detect such an activity yielded some improvements over baseline approaches (see Table 2.5, page 40). However, accurately detecting (using smart-phones) when a driver switches from simply driving to actively searching for a vehicle-parking space was not trivial.

With regards to the machine learning systems we used and developed in our ParkUs and later 'cruising' detection methods, we note that they were trained on relatively small datasets. Whilst our parking activity dataset was the largest (at the time of writing when compared to published competitor systems, see Table 2.4, page 37) we still struggled with generalisation and accurate (in terms of precision and recall) detection results. Labelled data aids the learning rate because the machine learning system is essentially guided towards understanding/correctly classifying human activities based on sensor data streams. This is in contrast to 'deep' neural networks, where learning is often based on unsupervised (i.e., raw data streams) large datasets and has no access to pre-coded statistical features. The latter has been relatively successful with vision based classification problems [48], but the computational resources (both in size and length of learning periods) required can be prohibitive.

We recommend further research into human driver activity detection and classification. For example, at small, narrow inner-city intersections, understanding a human driver's intentions (i.e., whether or not they are letting someone through the junction or merging into traffic if a vehicle is parked) will assist human/CAV traffic flow interactions and increase overall system level safety. Currently, apart from signalling with an indicator or flashing headlights, information between human driven and (future) CAVs is limited. A potential, system-wide solution which requires all drivers to input their desired destination and to subsequently broadcast their planned route or next set of way-points (e.g., through a retro-fitted V2V wireless system) could aid intra-operability between CAVs and human driven vehicles. However, this will have privacy implications for transport users.

Our second and third research questions concerned future city scenarios and investigated how CAVs could be used to provide public transport services (in the form of hailable taxis) as well as providing temporary wireless connectivity to a distributed (i.e., city-wide) delay tolerant sensor network (such as those concerning environmental sensing, e.g., air quality, temperature etc.). We addressed our second and third research questions by initially investigating taxi

fleet trace datasets, and later agent-based CAV models, in order to better understand the feasibility and performance requirements of such a system.

In Chapter 3 we used real taxi trace data collected in two cities, namely Rome (Italy) and San Francisco (USA) in order to investigate the performance of collecting and disseminating city-wide sensor data. We developed a 'folding' method in order to vary the simulated taxi fleet sizes and maintain relatively stable numbers (of active taxis) throughout our simulated time windows. We also developed a realistic city road network topology and Line-of-Sight (LoS) model by downloading and filtering (in an appropriate manner) openly available datasets from OpenStreetMap [65].

Throughout our simulations we noted an increase in terms of mean PDR and PRR values when increasing taxi fleet sizes. However, there was a limit to improvements generated beyond the 1:1 ratio of taxis to sensors. We noted in our initial simulations that final mean PDR values were rather low (0.6–0.7) by wireless network standards.

We briefly investigated how sensor message distribution was limited and showed through a One-to-All simulation (rather than our typical All-to-All scenario), that sensor messages spread rapidly (again relatively speaking) amongst the fleet of simulated connected taxis. Over 80% of the taxi fleet received the single sensor (i.e., source) message within a (simulated) hour of the first taxi visiting and exchanging information with the sole transmitting sensor. This highlighted a potential issue of our system, whilst messages spread relatively quickly amongst the taxi fleet, collecting and disseminating sensors messages back to the sensors was less successful.

In order to address some of the limitations of our initial taxi-VANET system (and our third research question) we investigated the problem of optimising a CAV passenger system with the need to collect, exchange and distribute city-wide sensor messages. We developed an agent-based VANET simulator to allow us to alter passenger trip routes and CAV mobility incentives. Our method of improving PDR involved re-assigning edge weights to individual roads (which form the entire city road network/graph), by using a special heuristic that took into account the nearest sensor location to each road-edge mid-point. We were able to vary the weighting parameter (referred to as α) in order to experiment with differing routing incentives (for the simulated CAV fleet) as well as benchmarking against a system of routing based on real-world road edge lengths.

We showed that even small incentives (i.e., $\alpha < 0.3$) led to significant improvements in overall (final) mean PDR values from around 0.7 to over 0.9 with a small increase in mean user journeys (circa 500m seemed to suffice). Furthermore, we showed under uniform random distribution and a 'tidal-in' passenger demand scenario (such as an early morning rush-hour), that PDR rates did not alter significantly. However, we stress that more simulations with differing passenger demands are worthwhile in order to ensure PDR thresholds are achieved within reasonable time frames.

Unfortunately, due to time constraints, we were not able to simulate longer, 'rolling' scenarios whereby we would simulate entire days (if not more) of continuous CAV fleet movement (and passenger demand). Potentially, such simulations could allow us to better understand where hard-to-connect sensors are

and the 're-fresh' rate of the entire sensor network (i.e., how long it would take to repeatedly collect and distribute all sensor messages). However, in all our simulations, the PDR and PRR values tended to reach a 'saturation' point well before the end of our simulated time periods. This suggests any further running of the simulation was unlikely to yield better PDR results.

A better strategy for improving final PDR values could involve re-routing under-used or un-occupied CAVs towards hard to reach sensor locations. In practice this could be rather complex, since it is hard to transmit across the VANET a manifest of which sensors have been visited and which have not. However, we note that after an hour or so of simulation, most of the vehicles have most of the (visited) sensor messages. Therefore by a process of 'semi-stochastic' elimination, it could be 'guessed' or theorised that any sensor messages, not within the set collected by an individual CAV agent, are most likely not to have been visited by other CAV agents in the city-wide system. Consequently these unoccupied CAVs could re-route themselves to areas where they believe lie poorly-connected or hard to reach sensors.

From a simulation point of view, this is a relatively straightforward but time consuming task to implement properly. CAV agents will need sophisticated thresholds and coded rule sets in order to decide when it is efficient to route themselves away from the city (where it is likely more passengers are waiting to be collected) to visit predicted 'hard-to-reach' sensors. Furthermore, selecting the route to traverse the road graph is not trivial either, since the 'travelling salesman' problem is known to scale superpolynomially (in the worst case scenario) with the number of stops or sensors to visit along the trip. In practice the routing problem could be reduced by simply selecting the nearest (predicted) 'hard-to-reach' sensor to route towards. However, CAVs would then require knowledge of sensor locations, something that is not currently shared between the CAV agents. Finally such a system could potentially increase waiting (rather than travel) times per passenger trip.

5.2 Results Highlights

Penultimately, we sumarise some of our key results:

- Detecting parking activity using a driver's smartphone and associated low-energy sensors (such as the accelerometer and magnetometer) is possible and our method (referred to as ParkUs) showed considerable improvements over previous attempts (see results and competitor comparison Table 2.4, page 37).
- Detecting 'cruising' or park search behaviour using sensors found in a typical driver's smartphone was attempted and our novel proposed system of disparate window concatenation performed better when compared to other baseline machine learning approaches. However, we still struggled to achieve workable detection accuracies (see Table 2.5, page 40).

- We showed through use of real-world taxi trace, building footprint, and road network topology datasets that city-wide sensor networks can communicate with one-another via a fleet of connected taxis. However, end-to-end delay and PDR values were rather high (over 500s for large taxi fleet sizes, see Figure 3.16, page 71) and low (circa 0.7 to 0.8, see Figure 3.13, page 67) respectively.
- Through our method of 'folding' taxi trace datasets, we were able to scale the overall fleet size in order to assess sensor network performance improvements. We conclude that once there was roughly one taxi per deployed sensors, the improvements (in terms of PDR and transit delay) tended to level off, see Figure 3.18 (page 74).
- We showed, through our agent-based VANET simulator, that strategically re-routing passenger carrying CAVs led to significant increases in PDR (over 0.9 was regularly achieved, see Figure 4.13, page 102), whilst not impacting passenger trip lengths (on average an extra 500m of distance travelled per passenger trip was sufficient to improve PDR, see Figure 4.4, page 90).
- In the majority of our VANET simulations we investigated All-to-All scenarios, highlighting the ability of our system to provide both down (e.g., a sensor firmware update) and up-link (e.g., environment data gathering for central processing) services to a city-wide sensor network.
- We investigated V2V message exchange order and found little impact on message dissemination and final PDR values amongst our VANET's largest connected component (see Figure 4.18, page 108 for further results).

5.3 Future Work

Lastly, we briefly discuss some potential avenues for further research and routes to market for the systems we researched in this project:

- Potentially collaborating with Google could lead to larger human activity datasets for better human driving and vehicle-parking activity inference.
- Trialling our ParkUs system with more users over an extended period of time would yield a better understanding of the proportion per trip (or time) saved. Ideally we would aim to recruit users from an area of city in order to provide as near to real-time updates as possible.
- Ideally, a driver 'cruising' detection system could be incorporated into future connected vehicles. Potentially a such a system could access the GNSS radio module of the vehicle as well as telemetry data via the CAN bus in order to detect when a driver starts to search for parking. Such a method has the added benefit of relying on systems (within the vehicle) with access to large energy sources therefore allowing for further processing as well as high data sampling rates.

- Ever more granular and sophisticated vehicle traffic modelling (for example taking into account lane change behaviour and intersection traffic light control) could lead to ever more accurate results of our simulated VANET systems. Combining such a simulator with advanced CAV agents that are able (or potentially trained through some machine learning method) to decide when (and where) it is appropriate to search for poorly connect sensors without significantly increasing passenger waiting times could also be of interest. Such a system could potentially guarantee 1.0PDR.

- Further road network topologies could be experimented with, for example our focus has been on large cities (populations of over or around 1M). However, how our VANET system would operate in a sparse environment such as multiple villages/small towns could be investigated.

Appendix A

Appendix

A.1 Energy Model Parameters and Values

TABLE A.1: Power consumption reference values based on the Nokia N95 [2, 15, 46, 70, 102, 107].

Sensor, Component or Process	**Mean Power Consumption (mW)**	**Symbol**
Wi-Fi Channel Scan	1430	P_{ws}
Wi-Fi Connect	635	P_{ua}
Wi-Fi Idle	42.0	P_{wi}
UMTS (3G) active	645	P_{wc}
UMTS (3G) idle	466	P_{ui}
GNSS (Outdoors)	597	P_{go}
GNSS (Indoors)	357	P_{gi}
Bluetooth Connect	185	P_{bc}
Bluetooth Scan	195	P_{bs}
Gyroscope	103	P_{bs}
Accelerometer	55.0	P_{ac}
Compass/Magnetometer	45.0	P_{mg}

TABLE A.2: Table of power-off delays on the Nokia N95 as measured and reported by [46].

Radio Chipset	**Power-off delay (s)**	**Symbol**
GNSS	30.0	T_{gpo}
UMTS (3G) idle to off	31.3	T_{uio}
UMTS (3G) active to idle	5.45	T_{uai}

A.2 Energy Model Test Case Scenario Assumptions

PhonePark:

Bluetooth turned off, GNSS sampled every 15s, 3G transmission every 15s, accelerometer briefly turned on for 3 minutes after each parking event and no false positives: $E_{go}(T_{out}, T_{out}/15) + E_{gi}(T_{in}, T_{in}/15) + E_{ut}(T, T/15) + E_{ac}(3 \times 60 \times 2) = 10600J$

ParkSense:

Wi-Fi scans every 60s when user is away, Wi-Fi scans every 2s when user driving, 3G transmission with GNSS location for 4 of the detected events, no false positives. It is assumed that ParkSense is able to detect parking to aid comparison: $E_{ws}(T_{out}, T_{out}/2) + E_{ws}(T_{in}, T_{in}/60) + 4E_{er} = 3330J$

Park Here!:

Accelerometer and gyroscope turned on throughout, 3G transmission with GNSS location for 4 of the detected events, no false positives: $E_{ac}(T)+E_{gy}(T)+4E_{er} = 1840J$

ParkUS:

Accelerometer and compass turned on throughout, 3G transmission with GNSS location for 4 of the detected events, 0.192 false positive probability: $E_{ac}(T) + E_{mg}(T) + (4 + 4 \times 0.192)E_{er} = 1240J$

ParkUs-SA:

Accelerometer and compass turned on throughout, 4 GNSS samples per detected event, 3G transmission for 4 of the detected events, 0.121 false positive probability: $E_{ac}(T) + E_{mg}(T) + (4 + 4 \times 0.121)(E_{er} + 4E_{go}(30, 4)) = 1880J$

A.3 Extended Real-world VANET Simulator Model Parameters and Architecture

TABLE A.3: Rome and San Francisco Taxi Trace Datasets Folding Fleet Sizes.

Location	Number of Days Folded	Mean Number of Active Taxis	Mean Number of Taxis in VANET
San Francisco, USA	1	272	105
	2	523	269
	4	983	648
	6	1560	1180
	9	2260	1820
Rome, Italy	3	258	133
	7	465	277
	14	983	698
	21	1670	1310
	28	2250	1840

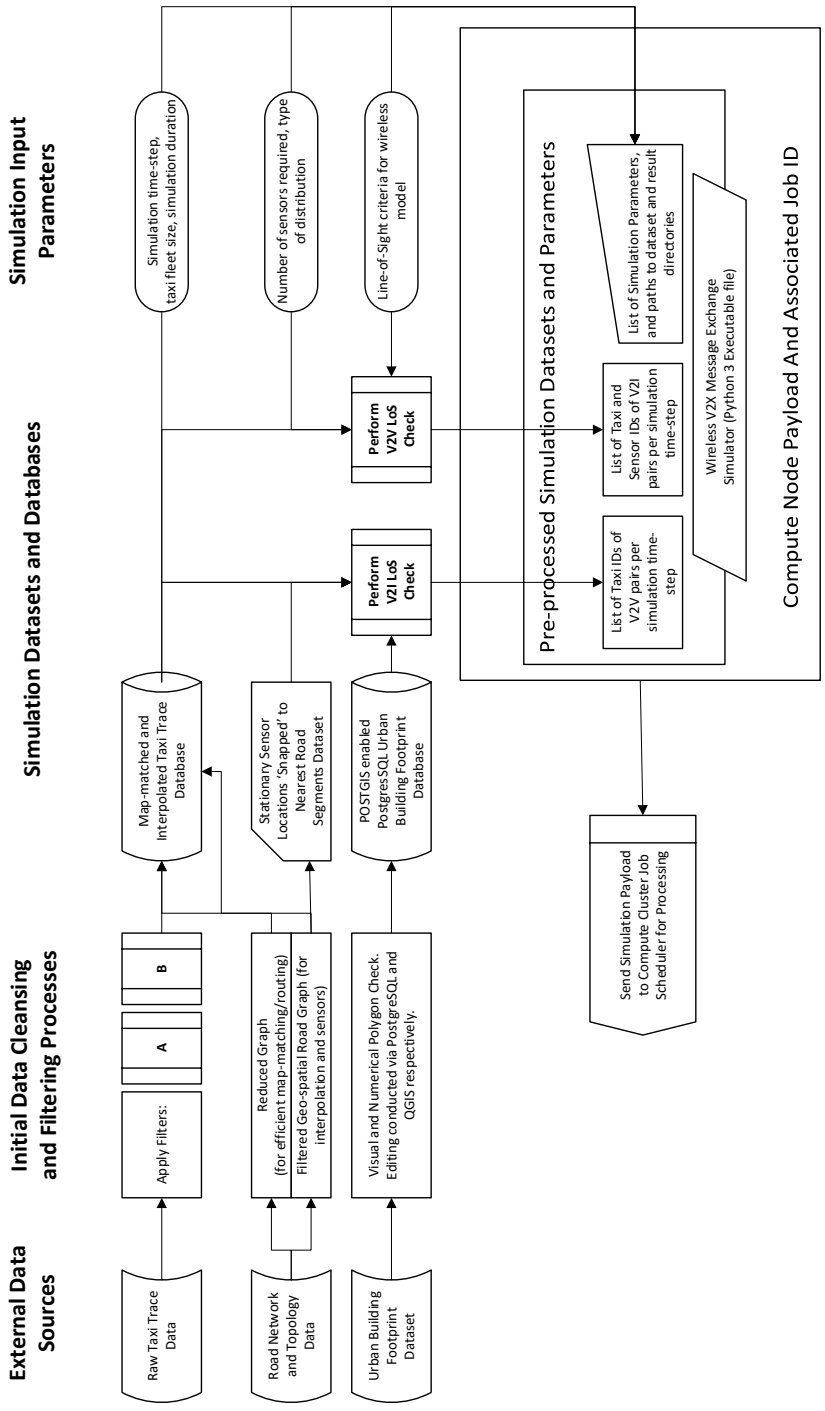

FIGURE A.1: Process flow chart showing overview of taxi trace VANET simulator data pre-processing and filtering pipeline.

Table A.4: Folded taxi-trace VANET simulator model parameters ranges and values.

Parameter Description	**Value Range**
Folded Taxi Fleet Size	250–1900
Number of Sensor Agents	250–2000
Maximum V2I Range/[m]	100
Maximum V2V Range/[m]	200
Simulation Time Step/[s]	10
Simulation Duration/[s]	3600

References

[1] R Buehler A Hamre. "Commuter Mode Choice and Free Car Parking, Public Transportation Benefits, Showers/Lockers, and Bike Parking at Work: Evidence from the Washington, DC Region".
In: *Journal of Public Transportation* 17 (June 2014), pp. 67–91.
DOI: 10.5038/2375-0901.17.2.4.
URL: https://www.nctr.usf.edu/wp-content/uploads/2014/07/JPT17.2_Hamre.pdf.

[2] Fehmi Ben Abdesslem, Andrew Phillips, and Tristan Henderson. "Less is More: Energy-efficient Mobile Sensing with Senseless".
In: *Proc. MobiHeld*. ACM, 2009. ISBN: 9781605584447.
DOI: 10.1145/1592606.1592621.
URL: http://dl.acm.org/citation.cfm?id=1592621.

[3] Erik K Antonsson and Robert W Mann.
"The Frequency Content of Gait".
In: *Journal of Biomechanics* 18.I (1985), pp. 39–47.

[4] A. Awang et al. "Routing in Vehicular Ad-hoc Networks: A Survey on Single- and Cross-Layer Design Techniques, and Perspectives".
In: *IEEE Access* 5 (2017), pp. 9497–9517. ISSN: 2169-3536.
DOI: 10.1109/ACCESS.2017.2692240.

[5] Michael Batty. "Cities as Complex Systems: Scaling, Interaction, Networks, Dynamics and Urban Morphologies". In: *Encyclopedia of Complexity and Systems Science*.
Ed. by Robert A. Meyers. New York, NY: Springer New York, 2009, pp. 1041–1071. ISBN: 978-0-387-30440-3.
DOI: 10.1007/978-0-387-30440-3_69.
URL: https://doi.org/10.1007/978-0-387-30440-3_69.

[6] Bilal Bilgin and V.C. Gungor. "Performance Comparison of IEEE 802.11p and IEEE 802.11b for Vehicle-to-Vehicle Communications in Highway, Rural, and Urban Areas". In: *International Journal of Vehicular Technology* 2013 (Nov. 2013), pp. 1–10.
DOI: 10.1155/2013/971684.

[7] G. Boeing. "OSMnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks". In: *Computers, Environment and Urban Systems* 65 (2017), pp. 126–139.
ISSN: 0198-9715.
DOI: https://doi.org/10.1016/j.compenvurbsys.2017.05.004.
URL: http://www.sciencedirect.com/science/article/pii/S0198971516303970.

[8] L Bracciale et al.
CRAWDAD Roma Taxi Trace Dataset (v. 2014-07-17).
Downloaded from https://crawdad.org/roma/taxi/20140717.
Accessed:23-MAY-2019. 2014. DOI: 10.15783/C7QC7M.

[9] P. Carnelli. *Agent-based VANET simulator for Delay Tolerant Sensor Networks in Cities.* Accessed: 31-JAN-2020. 2018.
URL: https://github.com/pc0179/ABMTANET.

[10] P. Carnelli. *Real-world dataset based VANET simulator for Delay Tolerant Sensor Networks in Cities.* Accessed: 31-JAN-2020. 2017.
URL: https://github.com/pc0179/RomeTaxiData.

[11] P. Carnelli and M. Sooriyabandara. *Moving Mobile Wireless Vehicle Network Infrastructure System and Method.* US2018098227A1. 2015.

[12] P. Carnelli et al. "ParkUs: A Novel Vehicle Parking Detection System".
In: *Proceedings of Twenty-Ninth Innovative Applications of Artificial Intelligence (IAAI) Conference.* AAAI, 2017. URL:
https://aaai.org/ocs/index.php/IAAI/IAAI17/paper/view/14604.

[13] P. E. Carnelli, M. Sooriyabandara, and R. E. Wilson.
"Large-Scale VANET Simulations and Performance Analysis using Real Taxi Trace and City Map Data".
In: *2018 IEEE Vehicular Networking Conference (VNC).* 2018, pp. 1–8.
DOI: 10.1109/VNC.2018.8628352.

[14] Clark County (Nevada, USA) Government. *Comprehensive Planning and Zoning Building Codes for Clark County, Nevada, USA.*
Accessed:20-DEC-2019. 1997.
URL: https://www.clarkcountynv.gov/comprehensive-planning/zoning/Documents/3060.pdf.

[15] Ionut Constandache, Romit Roy Choudhury, and Injong Rhee.
"Towards Mobile Phone Localization without War-Driving".
In: *Proc. INFOCOM.* IEEE, 2010. ISBN: 978-1-4244-5836-3.
DOI: 10.1109/INFCOM.2010.5462058. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5462058.

[16] C. Cooper et al.
"A Comparative Survey of VANET Clustering Techniques".
In: *IEEE Communications Surveys Tutorials* 19.1 (2017), pp. 657–681.
ISSN: 2373-745X. DOI: 10.1109/COMST.2016.2611524.

[17] University of Dartmouth (USA). *Community Resource for Archiving Wireless Data At Dartmouth (CRAWDAD).* Accessed:1-APR-2017.
2009. URL: https://crawdad.org/.

[18] NetworkX Developers. *NetworkX: Software for complex networks.*
Accessed: 20-JAN-2019. 2014. URL: https://networkx.github.io/.

[19] Python Developers.
Python 3+ Multi Processing Library Documentation.
Accessed:30-Jan-2019.
URL: https://docs.python.org/3/library/multiprocessing.html.

[20] The Economist. *Aparkalypse now: The perilous politics of parking.* Accessed:10-JAN-2019. 6 April 2017.
URL: https://www.economist.com/leaders/2017/04/06/the-perilous-politics-of-parking.

[21] J. Erickson et al. "CommPact: Evaluating the Feasibility of Autonomous Vehicle Contracts".
In: *IEEE Vehicular Networking Conference (VNC)*. 2018, pp. 1–8.
DOI: 10.1109/VNC.2018.8628319.

[22] Stefan Funck, Nikolaus Mohler, and Wolfgang Oertel. "Determining Car-Park Occupancy from Single Images".
In: *Intelligent Vehicles Symposium, 2004 IEEE* (2004), pp. 325–328.
DOI: 10.1109/IVS.2004.1336403.

[23] K. Garlichs, M. Wegner, and L. C. Wolf. "Realizing Collective Perception in the Artery Simulation Framework".
In: *2018 IEEE Vehicular Networking Conference (VNC)*. 2018, pp. 1–4.

[24] 3rd Generation Partnership Project (3GPP). "Evolved Universal Terrestrial Radio Access: Study on LTE based V2X Services".
In: (2016).

[25] 3rd Generation Partnership Project (3GPP). "Evolved Universal Terrestrial Radio Access: Vehicle to Vehicle (V2V) services based on LTE sidelink; user equipment radio transmission and reception".
In: (2016).

[26] O. Georgiou and U. Raza. "Low Power Wide Area Network Analysis: Can LoRa Scale?"
In: *IEEE Wireless Communications Letters* 6.2 (2017), pp. 162–165.
ISSN: 2162-2337. DOI: 10.1109/LWC.2016.2647247.

[27] Y. Ghasempour et al. "IEEE 802.11ay: Next-Generation 60 GHz Communication for 100 Gb/s Wi-Fi".
In: *IEEE Communications Magazine* 55.12 (2017), pp. 186–192.
DOI: 10.1109/MCOM.2017.1700393.

[28] A. Ghosh et al. "Millimeter-Wave Enhanced Local Area Systems: A High-Data-Rate Approach for Future Wireless Networks".
In: *IEEE Journal on Selected Areas in Communications* 32.6 (2014), pp. 1152–1163. DOI: 10.1109/JSAC.2014.2328111.

[29] S. Glass, I. Mahgoub, and M. Rathod. "Leveraging MANET-Based Cooperative Cache Discovery Techniques in VANETs: A Survey and Analysis". In: *IEEE Communications Surveys Tutorials* 19.4 (2017), pp. 2640–2661. ISSN: 2373-745X.

[30] S. Görmüş. "Synchronisation in 6Tisch networks". In: *2015 23nd Signal Processing and Communications Applications Conference (SIU)*. 2015, pp. 535–539. DOI: 10.1109/SIU.2015.7129879.

[31] Y. Zhu M. Li H. Huang D. Zhang and M.Y. Wu. "A Metropolitan Taxi Mobility Model from Real GPS Traces". In: *Journal of Universal Computing Science* 18.9 (2012). Accessed: 20-JAN-2019, pp. 1072–1092. URL: http://www.jucs.org/jucs_18_9/a_metropolitan_taxi_mobility.

[32] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. "Exploring Network Structure, Dynamics, and Function using NetworkX". In: *Proceedings of the 7th Python in Science Conference*. Ed. by Gaël Varoquaux, Travis Vaught, and Jarrod Millman. Pasadena, CA USA, 2008, pp. 11–15.

[33] Mark A Hall. "Correlation-based Feature Selection for Machine Learning". PhD thesis. Univ. of Waikato, 1999. ISBN: 9780874216561. DOI: 10.1.1.37.4643. arXiv: 9809069v1 [gr-qc].

[34] Nils Y Hammerla et al. "On Preserving Statistical Characteristics of Accelerometry Data using their Empirical Cumulative Distribution". In: *Proc. ISWC*. ACM, 2013. ISBN: 9781450321273. DOI: 10.1145/2493988.2494353. URL: http://dl.acm.org/citation.cfm?id=2494353.

[35] Samuli Hemminki, Petteri Nurmi, and Sasu Tarkoma. "Accelerometer-based Transportation Mode Detection on Smartphones". In: *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*. SenSys '13. Roma, Italy: ACM, 2013, 13:1–13:14. ISBN: 978-1-4503-2027-6. DOI: 10.1145/2517351.2517367. URL: http://doi.acm.org/10.1145/2517351.2517367.

[36] IEEE. "Standard for Information Technology Local and Metropolitan area Networks specific requirements: Wireless LAN MAC and PHY specifications amendment 6: Wireless Access in Vehicular Environments". In: (2010).

[37] Alphabet Inc. *Android Activity Recognition API*. Accessed: 2-FEB-2020. 2015. URL: https://developers.google.com/location-context/activity-recognition.

[38] Alphabet Inc. *Android Application Services Overview Documentation*. Accessed: 5-FEB-2020. 2019.

[39] D Leibling J Bates. *Spaced Out Persepectives on Parking Policy*. Tech. rep. Accessed:14-FEB-2020. The Royal Autonomobile Club Foundation for Motoring (RAC), July 2012. URL: https://www.racfoundation.org/assets/rac_foundation/content/downloadables/spaced_out-bates_leibling-jul12.pdf.

[40] Yuan J. et al. "Driving with knowledge from the physical world". In: *17th ACM SIGKDD International Conference Knowledge Discovery in Data Mining*. 2011, pp. 316–324.

[41] J. Jermsurawong et al. "Car Parking Vacancy Detection and its Application in 24-Hour Statistical Analysis".
In: *Proc. Frontiers of Information Technology*. IEEE, 2012.
DOI: 10.1109/FIT.2012.24.

[42] D. Jia et al.
"A Survey on Platoon-Based Vehicular Cyber-Physical Systems".
In: *IEEE Communications Surveys Tutorials* 18.1 (2016), pp. 263–284.
ISSN: 2373-745X. DOI: 10.1109/COMST.2015.2410831.

[43] Michael Jones et al. "ParkUs 2.0: Automated Cruise Detection for Parking Availability Inference".
In: *Proceedings of the 14th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*.
MobiQuitous 2017. Melbourne, VIC, Australia: ACM, 2017,
pp. 242–251. ISBN: 978-1-4503-5368-7.
DOI: 10.1145/3144457.3144495.
URL: http://doi.acm.org/10.1145/3144457.3144495.

[44] M. Kamat, A. S. Ismail, and S. Olariu.
"Optimized-Hilbert for Mobility in Wireless Sensor Networks".
In: *2007 International Conference on Computational Science and its Applications (ICCSA 2007)*. 2007, pp. 554–560.
DOI: 10.1109/ICCSA.2007.85.

[45] J. Kiss.
"Google admits collecting Wi-Fi data through Street View cars."
In: *The Guardian* (2010). URL:
https://www.theguardian.com/technology/2010/may/15/google-admits-storing-private-data.

[46] Mikkel Baun Kjærgaard et al. "EnTracked: Energy-efficient Robust Position Tracking for Mobile Devices". In: *Proc. MobiSys*.
Krakow, Poland: ACM, 2009. ISBN: 978-1-60558-566-6.
DOI: 10.1145/1555816.1555839.
URL: http://doi.acm.org/10.1145/1555816.1555839.

[47] D. Krajzewicz et al. "Recent Development and Applications of SUMO - Simulation of Urban MObility". In: *International Journal On Advances in Systems and Measurements*. International Journal On Advances in Systems and Measurements 5.3&4 (2012), pp. 128–138.
URL: http://elib.dlr.de/80483/.

[48] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton.
"ImageNet Classification with Deep Convolutional Neural Networks".
In: *Advances in Neural Information Processing Systems 25*.
Ed. by F. Pereira et al. Curran Associates, Inc., 2012, pp. 1097–1105.

[49] Toshiba Bristol Innovation Laboratory. *ParkUs Android Application*.
Accessed: 31-JAN-2020. 2019. URL: https://play.google.com/store/apps/details?id=toshiba.parkus&hl=en.

[50] Philip Langley. "Wavelet Entropy as a Measure of Ventricular Beat Suppression from the Electrocardiogram in Atrial Fibrillation". In: *Entropy* (2015). ISSN: 1099-4300. DOI: 10.3390/e17096397. URL: http://www.mdpi.com/1099-4300/17/9/6397/.

[51] M. Lauridsen et al. "From LTE to 5G for Connected Mobility". In: *IEEE Communications Magazine* 55.3 (2017), pp. 156–162. DOI: 10.1109/MCOM.2017.1600778CM.

[52] Deloitte Touche Tohmatsu Limited. *Global Mobile Consumer Trends, 2nd Edition*. Accessed:10-MAR-2019. 2017. URL: https://www2.deloitte.com/content/dam/Deloitte/us/Documents/technology-media-telecommunications/us-global-mobile-consumer-survey-second-edition.pdf.

[53] Diego López-de-Ipiña, Unai Aguilera, and Jorge Pérez. "Collaboration-Centred Cities Through Urban Apps Based on Open and User-Generated Data". In: *Proc. UCAmI*. Springer, 2015. ISBN: 978-3-319-26401-1. URL: http://dx.doi.org/10.1007/978-3-319-26401-1_19.

[54] Suhas Mathur et al. "ParkNet: Drive-by Sensing of Road-Side Parking Statistics". In: *Proc. MobiSys*. ACM, 2010. ISBN: 9781605589855.

[55] I. Mavromatis et al. "Agile calibration process of full-stack simulation frameworks for V2X communications". In: *2017 IEEE Vehicular Networking Conference (VNC)*. 2017, pp. 89–96. DOI: 10.1109/VNC.2017.8275604.

[56] Wes McKinney. "Data Structures for Statistical Computing in Python". In: *Proceedings of the 9th Python in Science Conference*. Ed. by Stéfan van der Walt and Jarrod Millman. 2010, pp. 51–56.

[57] Volodymyr Mnih et al. "Playing Atari with Deep Reinforcement Learning". In: *CoRR* abs/1312.5602 (2013). URL: http://arxiv.org/abs/1312.5602.

[58] R. Molina-Masegosa and J. Gozalvez. "LTE-V for Sidelink 5G V2X Vehicular Communications: A New 5G Technology for Short-Range Vehicle-to-Everything Communications". In: *IEEE Vehicular Technology Magazine* 12.4 (2017), pp. 30–39. DOI: 10.1109/MVT.2017.2752798.

[59] L Morris and S Haywood. "G222(P) Why do patient miss appointments? A retrospective population study in paediatric outpatients in a metropolitan hospital". In: *Archives of Disease in Childhood* 99.Suppl 1 (2014), A96–A96. ISSN: 0003-9888. DOI: 10.1136/archdischild-2014-306237.219. eprint: http://adc.bmj.com/content/99/Suppl_1/A96.2.full.pdf. URL: http://adc.bmj.com/content/99/Suppl_1/A96.2.

[60] Royal Automobile Club Foundation for Motoring.
General facts and figures about roads and road use.
Accessed: 2-FEB-2020. URL:
https://www.racfoundation.org/motoring-faqs/mobility#a2.

[61] I. C. Msadaa, P. Cataldi, and F. Filali. "A Comparative Study between 802.11p and Mobile WiMAX-based V2I Communication Networks".
In: *2010 Fourth International Conference on Next Generation Mobile Applications, Services and Technologies*. 2010, pp. 186–191.
DOI: 10.1109/NGMAST.2010.45.

[62] Sarfraz Nawaz, Christos Efstratiou, and Cecilia Mascolo. "ParkSense: A Smartphone Based Sensing System For On-Street Parking".
In: *Proc. MobiCom*. Cambridge: ACM, 2013. ISBN: 9781450319997.
DOI: 10.1145/2500423.2500438.

[63] Paul Newson and John Krumm.
"Hidden Markov Map Matching Through Noise and Sparseness".
In: *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. GIS '09.
Seattle, Washington: ACM, 2009, pp. 336–343.
ISBN: 978-1-60558-649-6. DOI: 10.1145/1653771.1653818.
URL: http://doi.acm.org/10.1145/1653771.1653818.

[64] T. Nitsche et al. "IEEE 802.11ad: directional 60 GHz communication for multi-Gigabit-per-second Wi-Fi".
In: *IEEE Communications Magazine* 52.12 (2014), pp. 132–141.
DOI: 10.1109/MCOM.2014.6979964.

[65] OpenStreetMap contributors. *Open Street Map search site.*
Accessed:20-FEB-2020. URL: https://www.openstreetmap.org.

[66] Project OSRM.
Open Source Routing Machine Car Profile Code Repository.
Accessed: 2-FEB-2020. URL: https://github.com/Project-OSRM/osrm-backend/blob/master/profiles/car.lua.

[67] Project OSRM. *Open Source Routing Machine Project.*
Accessed: 2-FEB-2020. 2014.
URL: https://github.com/Project-OSRM/osrm-backend/wiki.

[68] B. Paden et al. "A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles".
In: *IEEE Transactions on Intelligent Vehicles* 1.1 (2016), pp. 33–55.
ISSN: 2379-8858. DOI: 10.1109/TIV.2016.2578706.

[69] A. Pentland, R. Fletcher, and A. Hasson.
"DakNet: rethinking connectivity in developing nations".
In: *Computer* 37.1 (2004), pp. 78–83. ISSN: 1558-0814.
DOI: 10.1109/MC.2004.1260729.

[70] Gian Paolo Perrucci et al. “On the Impact of 2G and 3G Network Usage for Mobile Phones’ Battery Life”.
In: *Proc. European Wireless Conference*. IEEE, 2009.
ISBN: 9781424459353. DOI: 10.1109/EW.2009.5357972.

[71] Michal Piorkowski, Natasa Sarafijanovic-Djukic, and Matthias Grossglauser.
CRAWDAD San Francisco Taxi Trace Dataset (v. 2009-02-24).
Downloaded from https://crawdad.org/epfl/mobility/20090224.
Accessed:1-MAR-2017. 2009. DOI: 10.15783/C7J010.

[72] A. Pop et al. “Does Bidirectional Traffic Do More Harm Than Good in LoRaWAN Based LPWA Networks?” In: *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*. 2017, pp. 1–6.
DOI: 10.1109/GLOCOM.2017.8254509.

[73] The Associated Press. *Gov’t won’t pursue talking car mandate*.
Accessed:3-DEC-2018. 2017.
URL: apnews.com/9a605019eeba4ad2934741091105de42.

[74] A. Rasouli and J. K. Tsotsos. “Autonomous Vehicles That Interact With Pedestrians: A Survey of Theory and Practice”. In: *IEEE Transactions on Intelligent Transportation Systems* (2019), pp. 1–19.
ISSN: 1558-0016. DOI: 10.1109/TITS.2019.2901817.

[75] U. Raza, P. Kulkarni, and M. Sooriyabandara.
“Low Power Wide Area Networks: An Overview”.
In: *IEEE Communications Surveys Tutorials* 19.2 (2017), pp. 855–873.
ISSN: 1553-877X. DOI: 10.1109/COMST.2017.2652320.

[76] F. Rosique et al. “A Systematic Review of Perception System and Simulators for Autonomous Vehicles Research”.
In: *Sensors* 19.2 (2019), pp. 648–677.
DOI: https://doi.org/10.3390/s19030648.

[77] Dario Sabella et al. *5GAA: Toward Fully Connected Vehicles: Edge Computing for Advanced Communications*. Accessed:17-FEB-2020. 2017.
URL: https://5gaa.org/wp-content/uploads/2017/12/5GAA_T-170219-whitepaper-EdgeComputing_5GAA.pdf.

[78] Rosario Salpietro et al.
“Park Here! A Smart Parking System Based on Smartphones’ Embedded Sensors and Short Range Communication Technologies”.
In: *Proc. WF-IoT*. IEEE, 2015. ISBN: 9781509003655.
DOI: 10.1109/WF-IoT.2015.7389020.

[79] San Francisco Municipal Transportation Agency.
SFpark: Pilot Project Evaluation. Tech. rep. San Francisco, US: San Francisco Municipal Transportation Agency, 2014.
DOI: 10.1073/pnas.0703993104.
URL: http://sfpark.org/resources/docs_pilotevaluation/.

[80] R. C. Shah et al. “Data MULEs: modeling a three-tier architecture for sparse sensor networks”. In: *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications, 2003.* 2003, pp. 30–41. DOI: 10.1109/SNPA.2003.1203354.

[81] Donald C. Shoup. “Cruising for Parking”. In: *ACCESS Magazine* 30 (2007), pp. 16–22. ISSN: 0967070X. DOI: 10.1016/j.tranpol.2006.05.005.

[82] Donald C. Shoup. *Free Parking or Free Markets*. ACCESS Magazine. 2015. URL: http://www.accessmagazine.org/articles/spring-2011/free-parking-free-markets/.

[83] Donald C. Shoup. “The High Cost of Free Parking”. In: *Journal of Planning Education and Research* 17 (1997), pp. 3–20. URL: http://www.uctc.net/research/papers/351.pdf.

[84] Christoph Sommer, Reinhard German, and Falko Dressler. “Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis”. In: *IEEE Transactions on Mobile Computing* 10.1 (Jan. 2011), pp. 3–15. ISSN: 1536-1233. DOI: 10.1109/TMC.2010.133.

[85] Leon Stenneth et al. “PhonePark: Street Parking Using Mobile Phones”. In: *Proc. Mobile Data Management* (2012). DOI: 10.1109/MDM.2012.76.

[86] E. G. Strom. “On Medium Access and Physical Layer Standards for Cooperative Intelligent Transport Systems in Europe”. In: *Proceedings of the IEEE* 99.7 (2011), pp. 1183–1188. DOI: 10.1109/JPROC.2011.2136310.

[87] *Table 1-41 Principal Means of Transportation to Work*. Accessed: 2-FEB-2020. 2017. URL: https://www.bts.gov/archive/publications/national_transportation_statistics/table_01_41.

[88] A. Tassi et al. “Modeling and Design of Millimeter-Wave Networks for Highway Vehicular Communication”. In: *IEEE Transactions on Vehicular Technology* 66.12 (2017), pp. 10676–10691. ISSN: 0018-9545. DOI: 10.1109/TVT.2017.2734684.

[89] New York City Taxi and Limousine Commission. *TLC Trip Record Data*. Accessed:19-DEC-2019. URL: https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page.

[90] F. B. Teixeira et al. “Data Muling Approach for Long-Range Broadband Underwater Communications”. In: *2019 International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. 2019, pp. 1–4.

[91] Arvind Thiagarajan et al. “Cooperative Transit Tracking Using Smart-Phones”. In: *Proc. Embedded Networked Sensor Systems*. ACM, 2010. ISBN: 9781450303446. DOI: 10.1145/1869983.1869993.

[92] J. Thota et al. "V2V for Vehicular Safety Applications". In: *IEEE Transactions on Intelligent Transportation Systems* (2019), pp. 1–15. ISSN: 1524-9050. DOI: 10.1109/TITS.2019.2920738.

[93] Financial Times. *Governer of Alaska, Sarah Palin Resignation Speech.* URL: https://www.ft.com/content/238e173e-681f-11de-848a-00144feabdc0.

[94] UK Government Department For Transport. *Transport Statistics Great Britain 2014*. Accessed: 19-DEC-2019. 2014. URL: www.gov.uk/government/statistics/transport-statistics-great-britain-2014.

[95] Marco D. Tundo, Edward Lemaire, and Natalie Baddour. "Correcting Smartphone orientation for accelerometer-based analysis". In: *MeMeA 2013 - IEEE International Symposium on Medical Measurements and Applications, Proceedings*. Ottawa, Canada, 2013, pp. 58–62. DOI: 10.1109/MeMeA.2013.6549706.

[96] UK Government Department for Transport. *Statistical data set: All vehicles (VEH01)*. Accessed: 2-FEB-2020. URL: https://www.gov.uk/government/statistical-data-sets/all-vehicles-veh01.

[97] United Nations. "World Urbanization Prospects". In: (2014). Accessed:1-APR-2017. URL: http://esa.un.org/unpd/wup/Highlights/WUP2014-Highlights.pdf.

[98] ustroetz and pc0179. *Python 3 Wrapper for OSRM API.* Accessed: 19-DEC-2019. 2017. URL: https://github.com/ustroetz/python-osrm.

[99] S. van der Walt, S. C. Colbert, and G. Varoquaux. "The NumPy Array: A Structure for Efficient Numerical Computation". In: *Computing in Science Engineering* 13.2 (2011), pp. 22–30. ISSN: 1558-366X. DOI: 10.1109/MCSE.2011.37.

[100] A Viridis and M Kirsche. "The OMNet++ Enivornment and its Ecosystem". In: *Recent Advances in Network Simulation*. Vol. 1. Springer International Publishing, 2019, pp. 1–472. DOI: 10.1007/978-3-12842-5.

[101] Shuangquan Wang, Canfeng Chen, and Jian Ma. "Accelerometer Based Transportation Mode Recognition on Mobile Phones". In: *Proc. APWCS*. Beijing, 2010. ISBN: 9780769540030. DOI: 10.1109/APWCS.2010.18.

[102] Yi Wang et al. "A Framework of Energy Efficient Mobile Sensing for Automatic User State Recognition". In: *Proc. MobiSys*. ACM, 2009. ISBN: 9781605585666. DOI: 10.1145/1555816.1555835.

[103] B. J. Waterson, N. B. Hounsell, and K. Chatterjee.
"Quantifying the potential savings in travel time resulting from parking guidance systems: a simulation case study". In: *Journal of the Operational Research Society* 52.10 (2001), pp. 1067–1077.
DOI: 10.1057/palgrave.jors.2601207.
URL: https://doi.org/10.1057/palgrave.jors.2601207.

[104] Marcus Wohlsen.
App That Lets Users Sell Public Parking Spots Is Told to Shut Down. 2014. URL: https://www.wired.com/2014/06/app-that-lets-users-sell-public-parking-spots-is-ordered-to-shut-down/.

[105] Xiaobo Chen and Danya Yao. "An empirically comparative analysis of 802.11n and 802.11p performances in CVIS".
In: *2012 12th International Conference on ITS Telecommunications.* 2012, pp. 848–851. DOI: 10.1109/ITST.2012.6425303.

[106] Y. Gu P. Li L. Shi Y. Chen M. Xu and X. Xiao. "Empirical study on spatial and temporal features for vehicular wireless communications".
In: *EURASIP Journal of Wireless Communication Networks* 1 (2014), pp. 1–12.

[107] Kyon-Mo Yang and Sung-Bae Cho. "A Non-GPS Low-Power Context-Aware System using Modular Bayesian Networks".
In: *Proc. MOBILITY.* Seoul, South Korea, 2014. ISBN: 9781612083667.

[108] Zenzic. *Autonomous Vehicle Roadmap Report.*
Downloaded from https://zenzic.io/content/uploads/2019/09/Zenzic_Roadmap_Report_2019.pdf. 2019.

[109] Z. Zhang. "Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges".
In: *IEEE Communications Surveys Tutorials* 8.1 (2006), pp. 24–37.
ISSN: 2373-745X. DOI: 10.1109/COMST.2006.323440.

[110] W. Zhao, M. Ammar, and E. Zegura. "A Message Ferrying Approach for Data Delivery in Sparse Mobile Ad Hoc Networks".
In: *Proceedings of the 5th ACM International Symposium on Mobile Ad Hoc Networking and Computing.* MobiHoc '04. Roppongi Hills, Tokyo, Japan: Association for Computing Machinery, 2004, 187–198.
ISBN: 1581138490. DOI: 10.1145/989459.989483.
URL: https://doi.org/10.1145/989459.989483.