Data Proxies, the Cognitive Layer, and Application Locality: Enablers of Cloud-Connected Vehicles and Next-Generation Internet of Things

by

Joshua Eric Siegel

S.B., Massachusetts Institute of Technology (2011)
S.M., Massachusetts Institute of Technology (2013)

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2016

Author .....
Department of Mechanical Engineering
May 18, 2016

Certified by .....
Sanjay E. Sarma
Vice President for Open Learning
Thesis Supervisor

Accepted by .....
Rohan Abeyaratne
Chairman, Committee for Graduate Students

Data Proxies, the Cognitive Layer, and Application Locality: Enablers of Cloud-Connected Vehicles and Next-Generation Internet of Things

by

Joshua Eric Siegel

Submitted to the Department of Mechanical Engineering
on May 18, 2016, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Mechanical Engineering

Abstract

Intelligent and Connected Vehicles reduce cost, improve safety, and enhance comfort relative to isolated vehicles. This ability for cars to sense, infer, and act facilitates data-driven improvements in occupant experience and vehicle design. This thesis explores informed individual vehicle improvements and proposes a secure and efficient architecture supporting connected vehicle applications.

Applying On-Board Diagnostic and smartphone data, I built a suite of prognostic applications. Engine coolant temperature data supports inference of oil viscosity and remaining life. A linear SVM using Fourier, Wavelet, and Mel Cepstrum audio features provides 99% accurate engine misfire detection. PCA-transformed Fourier acceleration features and GPS data inform decision trees attaining 91% wheel imbalance and 80% tire pressure and tread depth classification accuracy. These applications demonstrate the ability for local vehicle and peripheral device data to proactively improve individual vehicle reliability and performance. Connectivity facilitates crowdsourced data to further improve current vehicles and future designs.

Exploring vehicular connectivity, I consider data timeliness, availability and bandwidth cost in the context of an efficiency-improving idle time predictor. This predictor uses contextual information to eliminate short idle shutoffs in Automatic Engine Start/Stop systems, minimizing driver annoyance and improving compliance.

These applications reveal an opportunity to address excess resource consumption and system insecurity in Connected Vehicles and other constrained devices.

I introduce a secure and efficient model-based Internet of Things (IoT) architecture consisting of a "Data Proxy" utilizing a Cloud-run estimator to mirror an object with limited sensor input. The use of digital duplicates abstracts physical from digital objects, allowing the use of a mediating "Cognitive Layer" consisting of firewall and supervisory elements. These "Cognitive" elements apply the system model to monitor system evolution and simulate the impact of commands against known and learned limits.

Finally, I propose incorporating this architecture into the CloudThink digital object duplication platform. Proxies maximize data collected per unit cost, while the firewall and supervisory elements will allow increased actuator access and to support generalized Cloud-based prognostics. I discuss how CloudThink's data ownership policies and privacy visualization tools combine with this architecture to address consumer privacy and security concerns, improving consumer acceptance of Connected Vehicles.

Thesis Supervisor: Sanjay E. Sarma
Title: Vice President for Open Learning

Acknowledgments

I would not have been able to complete this dissertation without the love, friendship, and support of my family and friends. Thank you, Kate, Sharon, David, and Ben Siegel for your unconditional encouragement and support, and to the Palusci family who welcomed me with open arms. Without the calming conversations I had with Maria Frendberg and Penn Greene, it is doubtful I would have been able to make it to the finish line of this academic marathon.

I thank Professor Sanjay Sarma for inspiring me to pursue my academic interests and ensuring I had access to all the resources I needed to learn, grow, and succeed. I owe an additional debt of gratitude to Professor Maria Yang and Professor Warren Seering. As instructors and committee members, their guidance and insight helped me become a better engineer and communicator. Their comments and enthusiasm helped to improve this document tremendously and inspired me to explore the challenges of connected resource management and platform design at a deeper level.

While there is only one name on the cover, this dissertation is the output of more than one person. I am thankful to have worked with Dylan Erb in processing real-world automotive data and designing, building, and testing idle time prediction hardware and algorithms. Without the guidance and encouragement of Sumeet Kumar, I would not have had the motivation or skill to learn how to apply advanced mathematics to creating Data Proxies or classifying audio features. I was fortunate to find a "dream team" in Rahul Bhattacharyya and Ajay Deshpande with their amazing ability to apply machine learning to vehicle diagnostics, finding robust classification in the face of noisy data. It was a pleasure to work with Isaac Ehrenberg to create many wonderful RF-enabled devices, 3D printers, and more. Thank you also to the rest of the lab (Stephen, Laura, Pranay, Jason, Yongbin and Partha) for being available for support – both technical and emotional – these past years.

Without the help of Erik Wilhelm and Simon Mayer, CloudThink never could have turned into a real platform with such great potential. I am grateful for your assistance, and look forward to working on the Next Big Thing together.

Finally, thank you to Richard Shyduroff for instilling in me the entrepreneurial spirit. It's the potential for impact that has me the most excited about the difference my work can make on the world.

Contents

List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

Thesis Organization

Wide-area connectivity is growing rapidly in vehicles and across other domains. This growth brings about several questions, such as "what, if anything, will limit connectivity today or in the future?", "can these concerns be addressed with a system-level architecture?", "what would such an architecture look like, and with this structure in place, what might be newly enabled?", and "how does improved connectivity change the way engineers design vehicles or develop software for vehicles?"

To answer these and other questions, this document explores present-day Connected Vehicle technology and identifies the need for an automotive development platform addressing scalability, resource use, vehicle security, and data privacy. A novel, secure, and efficient model-based architecture addressing these needs is envisioned to enhance Connected Vehicles and other Internet of Things devices. In this architecture, the theory of a model-based "Data Proxy" is proposed as a means of minimizing bandwidth use and power consumption while providing accurate estimates of system states robust to sensor noise and connectivity outages. This structure additionally improves security through the use of Cloud object mirroring. Abstraction of digital and physical devices eliminates the possibility for applications to directly connect to vehicle data sources and actuators, allowing the use an intelligent "Cognitive Layer." This layer uses Cloud resources to apply the Data Proxy's model to run firewalling and supervisory elements monitoring system evolution and performing impact analysis to screen for potentially-injurious commands. An unmet opportunity for developing locally and collaboratively informed consumer facing applications is additionally identified, allowing vehicle owners to save time, cost, and hassle. The data gathered by these applications may simultaneously be applied to optimization of future automotive designs. Pervasive sensing and failure engine- and suspension-centric prognostic applications addressing this need and facilitated by the proposed architecture for connectivity are presented.

This thesis is divided into five chapters — background, theory, implementation, applications, and conclusions. Each chapter's content is expanded from recently pub-

lished papers or from manuscripts intended for publication. As such, each chapter – and the majority of sections – stands on its own, independent of the rest of the document. The thesis weaves several documents together into a coherent narrative, from identifying problems and opportunities in Connected Vehicles, to creating an architecture and design guidelines addressing these opportunities, to creating applications demonstrating that these solutions are effective and enabling.

Chapter 1 provides background information including a technology landscape summary and opportunity identification for Connected Cars. This chapter highlights issues plaguing Connected Vehicle platform design, data collection, and application development. It further discusses how these same issues challenge the deployment of connectivity, sensing, and actuation across the broader Internet of Things. Key findings include the need for coherent platform design, standardization, resource-conscious application development. Additionally, the landscape review of applications deployed to date demonstrates a need for consumer-facing automotive reliability- and design-improving applications making use of both local and collaborative data sources.

Chapter 2 explores new theory addressing the technological issues identified in Connected Vehicles. A vision for a novel architecture for connected systems or "The New IoT" is described. The Data Proxy is introduced as an estimator- or observer-backed digital duplicate of a physical object or system run in the Cloud. This section examines the implications of the Data Proxy on reducing system resource requirements, as well as the impact of the Cognitive Layer on improving platform security. Additionally, Chapter 2 considers an example automotive application aimed at decreasing driver annoyance with start/stop emissions reduction systems, and explores the technological requirements and tradeoffs needed to make the application cost-effective. These findings illustrate the design process of a typical collaboratively-informed automotive application and how technologies and their costs must be weighed against driver and OEM utility to determine technical and economic viability.

Chapter 3 discusses the CloudThink platform and how the current embodiment provides solutions for problems and enables applications addressing many of the opportunities identified in Chapter 1. This section considers how the connectivity

architecture proposed in Chapter 2 may be incorporated into CloudThink, addressing the remaining issues by improving security for sensitive actuators through the use of the Cognitive Layer's firewall. Proxies further improve the quality of Avacar vehicle mirrors by increasing the efficiency of encoding information in a fixed amount of data, allowing richer vehicle mirrors without an increase in bandwidth cost. Finally, this section explores how my and my collaborators' previous work on developing privacy systems for the CloudThink platform may be used to address vehicle owners' concerns about privacy, data ownership, and security, enhancing deployability of Connected Vehicles.

Chapter 4 discusses several data-informed applications addressing the need for consumer-facing reliability- and efficiency-improving applications identified in Chapter 1. I consider the under-explored areas of locally-informed, pervasively sensed prognostic applications and how these applications may be improved through the use of collaborative vehicle data. Though the applications discussed focus on single-source data and employ post-processing, this section explores how CloudThink and the use of Data Proxies uniquely enable improved Cloud-based versions of these applications. Applications presented include audio sensing of engine misfires, smartphone accelerometer-based tire pressure inference and wheel imbalance detection, and On-Board Diagnostic based oil life monitoring. All of these applications serve the purpose of improving vehicle efficiency, reliability, and comfort in the use phase, while similar technologies could aggregate data and be applied to optimizing vehicles in the design phase. This section describes how Proxies provide mathematical tools to observe system evolution with a minimum of sensor inputs and data transmitted, and how the Cognitive Layer's supervisor may be used to implement "prognostics in the Cloud" capable of running on all Avacar input data.

Finally, Chapter 5 recapitulates my contributions to connectivity and next-generation vehicle technology, as well as the implications of the proposed theory and applications on Connected Vehicles and the larger Internet of Things. I discuss opportunities for future work improving the Data Proxy and Cognitive Layer architecture, and how I envision data being used to inform vehicle operation, design, and mobility as a service.

Chapter 1

The Opportunity in Connecting Cars – and Everything Else

This thesis is motivated by a personal passion for all things automotive and a desire to improve current and future vehicles through intelligent data transformation and application development.

Electronic connectivity is increasing all around the world – in academia, in government, in industry and in the home – and the expansion of connected systems to include vehicles is no exception.

With improvements in sensing, computation, and networking, it is clear that the future will see the increased use of vehicles as development platforms capable of providing rich, distributed data and offering the ability to control physical systems at scale. The widespread deployment of these newly intelligent vehicles has the potential to bring about great changes in transportation, and as a result, the environment, and the world economy.

In this chapter, I provide background motivating this thesis. This information is built upon a manuscript titled, "A Survey of the Connected Vehicle Landscape: Architecture, Enabling Technologies, Applications, and Development Areas." I provide an in-depth introduction to Connected Vehicle technologies and follow in the second half of the chapter with challenge and opportunity identification motivating the need for the novel connectivity architecture presented in Chapter 2 and the applications

developed in Chapter 4

To date, Connected Vehicle technologies have been used to support the improvement of fleet safety and efficiency, but opportunities for these emerging technologies go well beyond, with the potential to allow collaboratively-generated data to inform all aspects of vehicle ownership and operation. Though vehicles today possess enabling technology sufficient to demonstrate how collaborative applications can work in an idealized world – and to show that these applications are commercially viable – there are challenges to connected and collaborative vehicle application deployment that must be addressed thoughtfully before the full potential of the Connected Car may be realized. From extensibility and scalability to privacy and security, and from resource minimization to optimal algorithm implementation, I identify addressible opportunities such that developers may begin to optimally transform how vehicles are viewed and used.

Though this chapter highlights issues plaguing platform design, data collection, and application development in the context of Connected Vehicles, the challenges and opportunities identifies are largely applicable to the broader Internet of Things.

1.1 Motivation & Opportunity

From mobile phones to Connected Homes to Connected Cars, connectivity is happening in all areas of life. While this connectivity brings great changes to how people live, enhanced connectivity in vehicles is one of the areas with the greatest potential for impact – on efficiency, comfort, convenience, cost savings, and beyond. Connected Cars have untapped value because these vehicles have so many inputs and outputs, because people spend a lot of time in their cars, and because small comfort and efficiency improvements can have dramatic long-range impacts when scaled up to apply to all vehicles.

the moist important value of Connected Cars is as a novel platform for development. Modern vehicles embody sensing, actuation, processing, and importantly, frequent touch- and interaction-points with humans, facilitating countless new ap-

plications. For these reasons, cars are worthy of exploration and offer the potential for a lot of innovation, through rich data set generation and broad output control possibilities. Thanks to advancements in embedded computing, wireless networking, and pervasive sensing, Connected Cars will help to grow Intelligent Transportation Systems applicability and aid in the optimization of broader transportation systems and infrastructure around the world [1].

As mobility-as-a-service grows, the cost of vehicle use decreases, making available a host of low-cost automotive experiences. Increasing mobility and miles traveled come at a cost; congestion, pollution, and road hazards are on the rise. The NTSB reports 16,000 crashes daily in the U.S, many of which are caused by driver distraction or fatigue [2]. In response to the high level of hazards encountered by drivers on a daily basis, modern vehicles have begun to implement Advanced Driver Assistance Systems (ADAS) capable of generating sensing data to monitor engine and chassis status, occupant state and health, and environmental conditions [1]. Additionally, these vehicles often incorporate intravehicle networks to control passive and active safety equipment, like seat belts and airbags, or lane keeping and adaptive cruise control, respectively, to minimize the risk and severity of incidents [2]. These same systems are used to replace mechanical powertrain controllers to improve efficiency and reduce fuel consumption and emissions.

The proliferation of vehicle sensing and actuation has brought a boon to software developers, with modern cars featuring more than 100 sensors – each of which is capable of generating several data outputs [3] [4]. While vehicle data has historically been processed within a vehicle and discarded, there is a new consumer demand for vehicle data sharing and applications to improve convenience, safety and security, as well as to provide entertainment. Vehicles now interconnect and collect information about their environment, themselves, and exchange data wirelessly in real-time, allowing vehicles to operate beyond drivers' line of sight, increasing the time horizon of perception, and therefore improving safety [1]. In fact, video streams and Connected Vehicle technology have been used to enable "transparent" vehicles drivers can see through [5], and automated reporting hazard reporting systems [6]. This move towards sensor-

enhanced, connected safety systems will help to build a zero-accident car, as vehicles can sense an imminent hazard and plan and execute appropriate maneuvers [7].

In a 2012 survey conducted by the National Highway Traffic Safety Administration, consumers who observed and used Connected Vehicle technology were largely accepting and expressed a desire to have such technology incorporated into their own vehicles. 74% indicated strong agreement that they would like to have vehicle to vehicle (V2V) safety features in their vehicles [8]. This, combined with recent consumer facing user interfaces designed to be more engaging and capable of providing rich information to vehicle occupants, makes it clear that the automotive market is headed in the direction of increasing connectivity and electronic occupant engagement, and that consumers will actively seek more inter-connected vehicles in the future [1].

While vehicular sensor data provide all of the necessary information to control a vehicle, it is rarely processed so as to reveal insight into what lies ahead. Therefore, applications of data today are often confined to being reactive rather than offering proactive and anticipatory actions. In order to approach optimal efficiency, performance, and to provide an enhanced user experience, the vehicle must predict future conditions and driver preferences and prepare for those conditions ahead of time.

While this type of foresight is practically impossible within the bounds of a single vehicle, it becomes attainable once the horizons are extended to an entire system and network of vehicles and beyond. In a model where vehicles communicate, share, and learn, new applications are possible. When one car hits a pothole, all Connected Vehicles can "feel" the shock and adjust their suspensions accordingly. Vehicles sitting in traffic become aware of the flow of the cars and adjust their start-stop calibrations to improve fuel economy and smoothness of operation, seamlessly and silently. Aggregate information improves user experience, by adjusting vehicle cabin conditions based on historic data for individual drivers and entire fleet preferences. Everything from seat position, to media consumption, to navigation may be better informed by the accessibility of complete vehicle information.

Historically, accessing the data to build these applications has been technically and financially challenging, but the advent of always-connected telematics systems

and rich vehicle data mirrors have increased the availability of data to feed these and other programs. Local and Cloud-based computational intelligence may be used to deliver novel, user-facing features that improve the experience and performance of a vehicle and all vehicles in the connected fleet. The application of vehicle data to provide long-term and proactive optimization has the potential to greatly improve driver satisfaction and fleet performance.

Increasing connectivity brings with it unique opportunities for vehicle to vehicle (V2V) and vehicle to infrastructure (V2I) collaboration and dataset aggregation. Data sharing from vehicle to vehicle can be used to increase a car's computer's perception horizon, allowing safety improvements, reductions in congestion, and more. Reliability data shared with manufacturers and aggregated in the Cloud may be used to optimize vehicle longevity and improve user experience, while other collaboratively gathered and processed data may be used to unlock further potential through the development of applications seeking to understand how humans drive, how vehicles wear, and more.

Connecting vehicles is not a new idea. For decades, researchers have explored functional problem-solving using shared data across vehicles [9]. Early attempts at collaborative applications included platooning for traffic management and efficiency improvements. While these applications suffered from technological limitations, the time is now appropriate to begin to deploy these once-theoretical applications. Bandwidth is inexpensive, sensing costs have plummeted, storage capabilities are growing rapidly, and computation is more energy, space, and cost efficient than ever. The Right to Repair movement and other consumer activist groups prove that there is a consumer demand for data, while vehicle OEMs are now seeking to use vehicle data to optimize vehicle design, reliability, and service. With today's combination of market opportunity, technological readiness, and data science advancements, one finds a unique opportunity to use vehicle data to enhance consumer and industry comfort, efficiency, reliability, and safety.

Addressing the data access problem and building a platform supporting collaborative vehicle learning allows the development of applications with the potential to change the automotive user experience. Core issues are the challenges of collecting

vehicle data, analyzing these data coping with its sparsity and locality, intelligently learning from these data using computational intelligence algorithms, ensuring scalability and timeliness of the solution, and managing weaknesses in current telematics systems such as gaps in coverage, bandwidth, and high round-trip latency. Further, security, privacy, and extensibility must be addressed. With a framework for addressing these concerns in place, collaborative learning - the use of shared vehicle data to optimize performance, efficiency, and user satisfaction - may be achieved. Understanding the technology presently available is the first step in formulating this framework.

1.2 Landscape Review Contents & Organization

This section provides a broad summary of the Connected Vehicle landscape and the related application space. It is not intended as a complete survey of available technologies, platforms, or research projects, nor is it an exhaustive exploration of all areas of data-informed or collaborative vehicle applications. Rather, this document offers a high-level summary of several prominent applications that have been theorized or implemented, research directions, and guiding thoughts relating to how next-generation Connected Vehicle architecture and applications might be structured. The reader should take from this document an understanding of the opportunity space, and find within it a substantial set references to documents for further reading and elucidation on related subjects.

Broadly, this document contains:

1. Background and History of the Connected Car
2. Enabling Technologies
3. Today's Platforms and Applications
4. Problems Faced with Current Technology
5. Opportunities for Future Improvement and Conclusions

1.3 Background of Vehicle Computation and Connectivity

Historically, much of the operation of a car was, if not "open loop," controlled by simplistic mechanical feedback systems [10]. Carburetors metered fuel flow based on the size of the mechanical jets installed, gears were selected and engaged manually, and radios – in those cars that had them – were tuned and adjusted by hand, rather than with the use of push-buttons, transistors and signal seeking algorithms.

1.3.1 Electronic Efficiency Enhancers

We have seen a massive growth in electronic components and systems on-board vehicles since the introduction of early emissions regulations and more stringent fuel economy standards in the 1970's. The oil embargo and these newly implemented fuel standards combined to make the use of electronic control units for engines and automatic transmissions attractive and cost-effective for automobile manufacturers [10]. At the same time that these systems were introduced, manufacturers began to deploy early On-Board Diagnostic systems in order to diagnose, maintain, monitor, and program these systems, as service costs might otherwise be prohibitive [11]. The earliest diagnostic systems deployed in vehicles were used for programming and testing computer modules on the assembly line, and not intended for use outside of the factory or beyond limited use at dealerships [12].

Perhaps due to the high cost of deploying these embedded systems and networks for powertrain control, and the low marginal cost of installing new sensors, the range of vehicle sensing and actuation proliferated upon the introduction of On-Board Diagnostics. Many original equipment manufacturers (OEMs) leveraged their newly-installed networks to instrument and control many aspects of vehicles, from entertainment systems to safety systems. In time, the number of computer modules in a new car began to grow as electronics prices fell and the cost of tooling up mechanical solutions became relatively unfavorable.

1.3.2 On-Board Diagnostics

Following on the heels of limited-use assembly line networks, On-Board Diagnostics began in earnest with an edict from the California Air Resources Board, or CARB. These early systems were designed to ensure the emissions systems used on vehicles remained intact and operating within specifications for the life of a vehicle, or otherwise present an indicator to the driver or a mechanic. On-Board Diagnostics were first legislated in 1988 [11]. A second variant of On-Board Diagnostics, OBD-II, was mandated by legislation in the U.S. starting in 1996, and a European variant was required by 2000. This second version of On-Board Diagnostics required additional sensing and reporting capabilities for improved powertrain monitoring and simplified troubleshooting [10].

In parallel to the deployment of OBD-II in the mid-1990's, in-vehicle computing began a shift to become consumer-facing, providing control over not just safety and control systems, but also "infotainment" and user interaction [13]. As a result of the increased demand for comfort, convenience, and safety features, vehicles grew to include a large number of electronic controllers. As sensors were more widely deployed, fell in cost, and increased in accuracy, vehicle sensing technology became increasingly pervasive. Historically, these sensors and devices had dedicated networks with complex wiring running between individual nodes requiring communication [13]. This wiring was expensive and unsustainable as systems became more complex, so other "bus type" systems such as Controller Area Network (CAN) needed to be deployed to provide flexibility in the face of varying vehicle configurations [13]. Ultimately, this simplified networking would lead to the creation and adoption of common vehicle platforms or electrical architectures to reduce the amount of wiring and differing network standards within a vehicle [14].

1.3.3 CANbus and Car Communication

CANbus was introduced as an automotive networking technology in 1986, and adopted as a de facto diagnostic standard for vehicles beginning in 2008. It simplified the

development of vehicle hardware and made data transport and use easier [15]. The result of the heightened level of connectivity within a vehicle led to enhancements in local processing, helping cars to become "self-aware." Engineers began to apply the data collected by hundreds of sensors and processed by dozens of power control units to optimize the performance, comfort, and safety within the confines of a single vehicle.

The use of automotive data to optimize a single vehicle offers improvements and is more advanced than the systems found in older vehicles, but in this paradigm each vehicle is still an island. The trend today is that wide-area connectivity is taking hold and beginning to build bridges between vehicles, allowing collaborative data collection, analysis, and optimization.

1.4 Enabling Technologies

This section explores the key enabling technologies for next-generation vehicle applications capable of taking advantage of advances in computation, sensing, and connectivity. Recent enabling trends include reductions in cost and power, increases in computation power, commoditization of sensing and connectivity, and deployment of leveragable vehicle technologies that may be repurposed for pervasive data collection. Without any one of these enhancements, it would not be possible to integrate vehicles into a wide-area collaborative network. These technological developments enable a symbiotic ecosystem of in-car, and can also enable extra-vehicular data collection, transmission, and analysis that contributes to connecting vehicles and unlocking the potential in distributed sensing and action.

1.4.1 Computation & Embedded Systems

The rapidly decreasing price of embedded micro-controllers and flash memory have contributed to the broader integration of microcomputers in every aspect of vehicle control. Moore's Law applies to the computers in vehicles just as it applies to large-scale systems, meaning these in-vehicle computers continue to rapidly gain computation

capacity and decrease power consumption. In fact, the energy savings associated with shrinking die sizes and improved architecture efficiency helps to lower quiescent (idle) and operating power, making the use of embedded computers attractive in increasingly energy-conscious design environments – even for the most mundane of systems.

An example of pervasive computation now found in cars are the electronic Engine Control Units (ECUs) that today handle fuel mapping based on temperature, airflow and pressure inputs. Similarly, modern hydraulic brake systems now feature inline, actuable valving and pumps to allow high-speed brake modulation to prevent skids. Vehicle stability systems are another example of computing systems taking take advantage of high-speed trajectory data to maintain vehicle yaw, pitch, and roll through extreme maneuvers [16]. In fact, the use of computers in certain vehicle networks may offer a reduced cost over mechanical or open-loop control due to improved efficiency, reliability, or reduced machining complexity. This is especially true in systems where complex mechanical setups, like linkages or vacuum systems, can be replaced with simple sensors, switches, and solenoids.

The improvement in computational power supports better Connected Vehicles by enabling rapid local computation, improved edge and infrastructure computing resources, and scalable, elastic Cloud computation for complex calculations. As computers continue to advance, their role in the future of vehicles will become more integral, and vehicle computation will likely extend from intrinsic sensing to extrinsic data use.

1.4.2 Sensing

In recent years, the cost of sensing has fallen and the complexity of what is senseable has increased dramatically. While some technologies, such as electromechanical accelerometers or acoustic transducers, have been mass-market ready for decades, others have only recently been developed into tenable options for deployment in vehicles. For example, microelectromechanical sensors have fallen in cost and power consumption, enabling their widespread use in vehicles [17] [18]. Additionally, the shrinking scale of sensors has reduced weight, and benefits from economies of scale due to mobile

phone integration has encouraged designers to deploy sensors in more mainstream use scenarios.

Recently, there has been a new wave of sensing technologies that are deployable in vehicles. These sensors focus on enhancing perception and expanding the time horizon for vehicle safety systems. These new sensors include cameras with machine vision processing, Global Positioning Systems, digital maps with map snapping, magnetic referencing, and "sight" enhancing sensors, like RADAR and LIDAR. In the context of collaborative vehicle applications, these perception enhancing sensors are perhaps the most intriguing, as RADAR and LIDAR can be used to help drivers see well beyond the human perception limit for difficult to distinguish objects, while providing improved positioning data [19] [20] [21] [22] [23, pg. 146].

1.4.3 Vehicular Networking

The complexity of vehicle networks has increased greatly in the decades since 1980. In a typical modern vehicle, up to 70 ECUs running millions of lines of code capture and process 2500 signals, including chassis systems, powertrain, comfort and convenience, media and safety networks. A comprehensive table summarizing how these networks are architected appears in Nolte (2005), Table 1 [14].

In today's automotive market, software and electronic technology, rather than mechanical reliability and efficiency, are innovation and sales drivers [24]. This is in large part due to the commoditization of vehicle hardware as a result of the Toyota Production System. The modern networked vehicle has been a boon for software development, and aides in the development of connected and collaborative vehicular applications, shifting the frontier of automotive innovation from hardware to software, and from self-contained operation toward globally informed models.

1.4.3.1 On-Board Diagnostics

As mentioned earlier, On-Board Diagnostics are used to enable quicker diagnostics and thus are used to lower repair costs and provide vehicle operational status updates

to outside tools. This is accomplished through fault detection, freeze-frame recording upon trigger events, and the use of standardized diagnostic trouble codes (DTCs) that may be used by a service technician to more easily locate and debug a system fault [7].

On-Board Diagnostic systems run as a communication protocol operating on top of an existing network in a vehicle, typically high-speed CAN. Parameters are called (requested) and responses are sourced from one or several control modules. Typically, a diagnostic device – either standalone, or as an interface to another processing unit – plugs into a standardized port and translates messages into meaningful data for human analysis. The parameters requested could be part of a legislated standard, or part of a manufacturer's proprietary parameter list [15]. OEMs extend OBD with custom PIDs, or Parameter Identifiers, to allow access to richer data on demand.

On-Board Diagnostics helped to cause the trend of increased vehicle networking, but these diagnostic suites are of limited utility to application developers. This is because OBD networks are inherently local to a single vehicle, and historically, have used proprietary, simplistic fault identifiers as opposed to dynamic models that learn based on multiple vehicles and real-world use. However, On-Board Diagnostics are very valuable in the Connected Vehicle landscape, as this interface provides a baseline for universally available data. The diagnostic port is also commonly used for aftermarket telematics devices, helping to bring older vehicles into the realm of Connected Vehicles.

1.4.3.2 Non-Diagnostic Networks

When OBD was legislated, automotive OEMs and suppliers took the opportunity to partially standardize network interfaces. This was driven by the cost of reengineering the electronic platform in vehicles, the fact that many auto manufacturers share common part vendors or suppliers, and the need for devices made by each of these vendors to interact seamlessly [14].

These networks allow manufacturers to improve occupant comfort and vehicle efficiency and reliability. Some of these networks are purpose built – *e.g.* drive-by-

wire and similar networks, which have demonstrated high degrees of reliability and robustness [14]. Others have evolved to provide links between vehicle systems, but carry less sensitive or time critical data. These networks all share common attributes — fault tolerance, determinism (timeliness), bandwidth, flexibility and security, but are tailored to their particular applications. Some networks utilize broadcast messages, which are sent at regular intervals or on an event and devices self-moderate the receipt and use of data, while other networks utilize call-and-response data sharing, even verifying the identify of a requesting node.

While computing and sensing hardware is similar across makes and models, specific communication protocols remain proprietary to each manufacturer. Some OEMs, such as General Motors, have designed their own scalable addressing schema to simplify engineering and maintain flexible development across all the vehicles the company produces [25]. Other OEMs design their communication architecture for each vehicle separately, or allow suppliers to dictate portions of the communication protocol.

While there are common, well documented electronics architectures like AUTOSAR, and communications protocols like FlexRay, many of a vehicle's protocols are trade secrets [14]. For an understanding of these base technologies, including CAN, LIN, MOST, 3.1 and 3.2 and Table 3 of Nolte (2005) and Table 1 of Faezipour (2012) offer comprehensive summaries and comparisons. These differing networks and architecture design approaches may be observed firsthand with the use of a diagnostic tool when "sniffing" network traffic on a vehicle bus.

Though several different network technologies are used in each vehicle, and though the protocols used may vary from network to network within a vehicle, these varied network types are frequently interconnected via a gateway device capable of arbitrating messages across networks [1] [26]. Thanks in part to the data sharing enabled by these gateways, non-diagnostic networks offer rich data and actuation possibilities to application developers through the standardized diagnostic link connector. With this enhanced ease of access, network interactions must be handled carefully due to privacy, security and safety concerns. The presence of a gateway device is what allows Connected Vehicle applications to share such rich sensing data, even if the

communications module is on a different network than the requested sensor.

1.4.3.3 Controller Area Network and Beyond

Controller Area Network, or the CANbus, is a commonly-used vehicle network. CANbus defines a standard protocol for electrical compatibility and messaging between sensors, actuators, and controllers. As vehicle subsystem control has migrated from standalone to distributed embedded systems, minimally invasive networking has become a design requirement. In a modern vehicle, modules handle localized control and sensing, but communicate with one another through multi-node, daisy-chainable dual wire networks. CANbus is preferred for this use as it features a fast baud rate, is good at minimizing packet collisions, and handles error detection and failures gracefully – ensuring accurate and reliable data transmission. Johansson, et al. provide a useful representation of a typical vehicle network architecture in Figure 7 [26].

The widespread adoption of Controller Area Network, partly because it has become the de facto On-Board Diagnostics standard, has helped to lower costs and create economies of scale for transceivers and compatible microcontrollers [27]. While CANbus is the most dominant networking protocol in vehicles, others, such as MOST, FlexRAY, LIN, and In-Vehicle Ethernet, are also becoming cost effective and gaining support from some automotive manufacturers and suppliers as CAN networks approach their bandwidth limitations [28]. As legacy networks begin to approach safe load limits, In-Vehicle Ethernet will become increasingly common [29].

1.4.4 User Interfaces

Vehicle user interfaces have evolved substantially, from coarse mechanical gauges to interactive touch-screen displays providing haptic feedback. Modern user interfaces provide unique touch-points with vehicle occupants, and are capable of informing, entertaining, and engaging people. Responsive displays can dynamically alter driver behavior, as in the case of eco-feedback gauges used in many hybrid and electric vehicles [30]. The focus on user interfaces indicates an effort to engage vehicle occupants

and provides additional opportunities for feedback from collaborative applications.

1.4.4.1 In-Car Entertainment

In-car entertainment has seen enhancements over the years, first migrating from analog radio dials and gauges to digital dashboards and tuners with presets. In recent years, head units (navigation systems and media interfaces) have evolved to include full color screens that provide feedback, backup camera visuals, state of charge monitoring, and more. New dashboards use LCD gauges and provide reconfigurable inputs, while some vehicles – such as the Tesla Model S – can incorporate full-fledged Linux or Windows computers into large, central touch screen displays.

In-vehicle entertainment has expanded to encompass in-vehicle information sharing. Consumers value the ability to see and react to their data in novel ways, and this trend is likely to continue [31].

1.4.4.2 Mobile Devices

In addition to providing additional in-vehicle touch points, connectivity and pervasive mobile device usage have led to increasing integration of the vehicle interface into mobile devices. Telematics providers such as OnStar provide remote locking and unlocking, maintenance updates, and more to users on their mobile devices to increase the radius of interactivity with their vehicles. The use of mobile devices as a "second screen" for cars allows applications that can create a compelling and relatable consumer proposition while simplifying the user experience.

Mobile devices may also be used to capture, process, or transmit enhanced sensing data, connecting vehicles that otherwise would be isolated.

1.4.5 Wireless Networking

Wireless networks have enabled vehicle data to be used in applications extending beyond local control. Improvements in connectivity, network scalability, routing efficiency, data security, and Quality of Service (QoS) assurances have made wireless

networking of vehicle computers an attractive proposition. There are a number of network technologies, architectures, and topologies deployed today, with trends that may change how vehicles in the future talk to one another and to the wider world. While cellular connectivity is gaining traction as a dominant force in industry, the vast majority of research literature emphasizes vehicle to vehicle (V2V) or vehicle to infrastructure (V2I) mesh networking using Dedicated Short Range Communications (DSRC). This section therefore discusses the results of these mesh networking studies, though it should be noted that many of these Connected Vehicle applications may be enabled using non-mesh networks – assuming sufficiently low latency and broad-reaching connectivity.

1.4.5.1 Communication Standards

While there are a number of wireless technologies designed to connect devices, people, and services, there are fewer communication standards focused on supporting Intelligent Transportation System (ITS) applications and capable of dealing with high-velocity, transient networks. For example, IEEE 802.11p defines the physical level addendum for Wireless Access in Vehicular Environments (WAVE). IEEE 1609 is based upon the higher-layer standard 802.11p, consisting of a set of standards to facilitate secure V2x applications and traffic management. Faezipour (2012) discusses these standards in depth. Theoretical models validate the efficacy of WAVE and show robust performance with regard to packet collision robustness, latency, and throughput [32].

ASTM E2213-03 supports V2I roadside communications, specifying the Media Access Control (MAC) and Physical (PHY) layers of 802.11 and 802.11a. This specification supports line of sight and distances up to 1 KM, with provisions for authentication and privacy preservation mechanisms [2]. These enabling protocols are well described in their standardization documents. Due to the constantly-evolving nature of standards, these documents are best referenced directly.

1.4.5.1.1 Dedicated Short-Range Communication (DSRC)

Dedicated Short Range Communication is a dedicated portion of wireless spectrum for improving traffic safety and efficiency. In the United States, the Federal Communications Commission allocated 75 MHz of bandwidth in the 5.9 GHz spectrum for DSRC for Intelligent Transportation Systems, including V2V and V2I applications.

The Medium Access Control for DSRC is managed by the IEEE 802.11P working group to facilitate the development of V2V and V2I applications [33]. There are many applications based on DSRC's implementation, namely collision avoidance and traffic management. A glossary of terms and common wireless networking acronyms appears in Morgan (2010) [34]. DSRC, though increasingly prominent, faces unsolved challenges relating to security, bandwidth, and node density to be discussed in subsequent sections.

1.4.5.1.2 Broadcast Protocols

There are a number of DSRC broadcast message types with different sensitivities to timeliness, data protection, and network coverage area.

Beacon messaging, for example, transmits the identification and context information about a vehicle in a short packet and transmits at a high update rate. These messages are used as the backbone for cooperative awareness applications. These beacon messages are often broadcast at regular intervals, but can be event driven, *e.g.* in the case of a hazard that must be reported [1] [35] [36].

Flooded messages continue to retransmit until the message's associated "time to live" elapses. The time to live can be measured in time since first transmission, absolute clock time, distance traveled between the start and current node, or hop count [1] [35].

GeoCast messages are often used for position-based routing. With GeoCast messages, each node knows its position and maintains a table of connected nodes so that messages may be routed based on physical location. GeoCasting makes it possible to address a particular vehicle directly, provided there is sufficient information about its location. GeoCasting often requires beaconing for neighbor discovery, wherein a

location service is run to collect the location of nodes, and nodes may be chained to forward messages to a specific destination via multiple hops. In this location-aware messaging type, the concept of a neighborhood is maintained, where vehicle parameters including location, trajectory, reliability of transmission, and more are combined to maximize the likelihood of data transmission. The use of trajectory data ensures that vehicles traveling away from a destination are not used and that vehicles that will only transiently occupy the same radio range as a destination are not entrusted with an important message [1] [37]. Note that that this type of message routing relies on the device's honesty in reporting its own location, an assumption that may be ill-advised.

Not all messages should be received by every neighbor on a network, and not all vehicles will be interested in each message type. A *publish/subscribe* model helps to address these issues. In this model, publishers push event data to a network without specifying an address. Subscribers select topics of interest. Messages are passed to the client if they pass the appropriate filtering criteria. An advantage of publish/subscribe models is the ability for connected devices to anonymously publish or subscribe to data streams in order to protect identities [23, 46].

1.4.5.1.3 Broadcast Topologies There are a range of broadcast topologies used in vehicle networking. A common networking tactic is mesh networking, which relies on node-to-node data relaying to form transient, short-hop networks. For conventional mesh networking, it is important to understand the flow of data to ensure messages reach the appropriate destination.

One tactic is the *direction-aware broadcast.* This broadcast might make use of a GeoCast message to ensure that information flows in a particular direction based on the direction of incoming signal. This might be used, for example, to relay information about a road hazard to all vehicles behind the lead car and message originator.

A simpler approach is the *naïve broadcast,* which broadcasts to all cars in range. This allows local vehicle decision making to decide whether to act on a particular message.

Intelligent Broadcast with Implicit Acknowledgment begins with a periodic broadcast; if an event-detecting vehicle receives the same message from behind, it infers that the vehicle behind it has received the message and is now the main responsible party for retransmission. Once receipt has been acknowledged in this manner, the lead vehicle may stop broadcasting [38] [39].

Additionally, there are *proactive* and *reactive* protocols for data dissemination. Reactive protocols seek out optimal routing only when data are sent. A control packet is forwarded and flooded to the network in order to determine the route. Proactive data flow maintains an active list of all connected nodes and sends periodic control messages to ensure robust connectivity [39] [40]. There are tradeoffs to be made in computation resources and time for each of these approaches, with enhanced computation often reducing the amount of data transmitted but frequently increasing the latency before transmission.

With different technologies (aside from V2V), other network topologies exist, such as star networks (a hub and spoke model), linear networks (device-device-infrastructure), and more.

1.4.5.2 Non-Mesh Networking Alternatives

This section discusses non-mesh network alternatives for vehicle connectivity. Car-to-Cloud via cellular link may ultimately prevail due to converging mesh-network and cellular technology capabilities and enhanced cellular range [41] [42]. As 4G technologies mature and 5G networks roll out, there is sure to be a shift in focus from DSRC to cellular and related technologies.

1.4.5.2.1 Short-Range to Device Bluetooth, Bluetooth Low Energy, WiFi, and other short-range RF technologies are often used to connect vehicles to user-portable devices, such as cellular telephones or tablet computers [1]. These devices can include scan tools such as the ELM327 diagnostic interface, more advanced tools such as the OBDLink MX, or application-specific tools, such as proprietary code readers. The advantage of direct-to-device connectivity is that bandwidth is typically free between

the device and the vehicle, latency is low, and the user can be better assured of their data's security and privacy due to the requirement of physical proximity between the device and the vehicle. This allows enhanced semi-local computation and real-time data visualization. If data are transmitted from the device to the Cloud or another device, additional bandwidth or latency costs and constraints may apply.

There are variants on this theme that use the secondary device as a "gateway" and stream data to the Cloud. This is a common way for aftermarket telematics systems (like the original Automatic device [43]) to provide Internet connectivity to vehicle computing modules via a device-device-Cloud arrangement. The downside to this type of arrangement is that the mobile device is required to be configured (paired) properly and kept local to the vehicle, or else data will be lost. For example, if the intermediate device goes out of range or its battery dies, no information will be recorded or transmitted.

1.4.5.2.2 Cellular (4G LTE / GSM / GPRS / UMTS) The cost of connectivity and bandwidth have fallen dramatically due to the pervasiveness of cellphones and the proliferation of machine to machine (M2M) connections. Bandwidth is now commoditized, and low priority, low speed data connectivity is inexpensive, with high priority data faster and more reliable than ever [44].

Cellular connectivity has distinct advantages in a vehicle over short-range device networking and mesh networking. For example, data are transmitted directly from the vehicle to a remote server and parameters may be captured even if a user forgets his or her mobile gateway device, or if a car is borrowed or stolen. Cellular networking may also provide better coverage in remote, sparsely populated areas, and affords vehicle operators better media consumption opportunities without risking saturating safety-critical V2V networks. Cellular connectivity may, in some senses, be considered a subset of V2I or vehicle to road infrastructure in that the Cloud is considered to be part of the extra-vehicle support infrastructure to which the car connects [2].

1.4.5.2.3 Piggyback Networking This hybrid approach combines DSRC with cellular connectivity. Extensions to both V2V and V2I are proposed, in which vehicles communicate with one another or with roadside infrastructure, and the last node in the chain provides bandwidth and connectivity to the Internet, serving as a form of mesh-network router [1]. This approach is less appealing in urban areas as the cost of cellular hardware and data has decreased, though it remains attractive in rural areas where cellular network connectivity is sparse. However, in these cases, the traffic density is likely low and conventional meshing may not be sufficient to connect a remote node with a gateway modem. This balance must be considered during the design stage.

A comparison of common vehicle wireless technologies appears in Papadimitratos, Table 2 (2009).

1.4.6 Computation

In-vehicle computational capabilities have increased dramatically in recent years. In modern cars, the enhanced processing power of hundreds of computers handles computation to make driving not just safer or more efficient, but also to provide increased comfort and convenience. This computation improvement manifests as improved local vehicle data processing for time-sensitive local control optimization, and it also enhances large-scale data collection and processing for fleet-wide monitoring and analytics. Similar improvements in Cloud and other scalable computing approaches have made development of algorithmically-complex approaches to collaborative vehicle applications tractable.

1.4.6.1 In-Vehicle Analytics

Enhanced sensing and computation have enabled real-time local vehicle data processing. Some of this local processing has been done for decades – such as dynamically adjusting a responsive fuel map – while more recent applications take advantage of increased processing power to not only look up data in a lookup table, but to actively

monitor, assess, learn, and tweak vehicle operation.

A simple example of in-vehicle analytics is an attempt to do "on-line" (during operation) vehicle analysis, from performance optimization to failure prediction [45]. Additional information may also be fed into models to predict characteristics about the vehicle, environment, and driver. As an example, vehicle data fuses with information about occupants such as blink rate and other physiological or "Quantified Self" data to predict and react to driver fatigue [2]. This is done in research environments and using test equipment, [46] [47] [48] [49] [50] [51] but also in production vehicles from Mercedes and other OEMs [52]. With proactive fatigue or risky emotion determination, affective computing may be used to mitigate stressors in a vehicle and improve safety and comfort for all occupants [7].

1.4.6.2 Remote Analytics

Remote analytics take advantage of enhanced computation outside of the vehicle and may make use of broader or fleet-wide aggregated vehicle data to train, develop, and tune models more rapidly.

These applications require a link between a vehicle and a computer or a network of servers, *e.g.* via LTE or through a smartphone gateway. These application can provide driver assistance and behavior monitoring / associated corrective "nudges," aggregated traffic data, computation-intensive failure prediction, and more. In cases where bandwidth costs are prohibitive, WiFi or other "data dumps" may help to transmit information without incurring cost.

Remote analytics is a new opportunity enabled by Connected Vehicle technologies, and is an application space rife with opportunity. We have begun to apply remote vehicle data collected from in-car computers and mobile devices to great effect, with the potential to reduce emissions and fuel consumption while saving drivers and fleet managers money [53] [54] [55] [56].

1.4.7 Data Handling Tools

With the proliferation of new data sources, communication methods, and storage mechanisms, there has been a related rapid growth in Big Data supporting infrastructure. This infrastructure helps developers to make the most of vehicle data with a minimum of development complexity. Without an environment conducive to data analytics, many Connected Vehicle applications would not be possible. The biggest levers in data handling stem from improved large-scale storage, guided analysis tools, and visualization systems.

1.4.7.1 Scalable Cloud Infrastructure

One of the biggest challenges in Big Data is computation and storage. A number of platforms have emerged to allow the creation of "elastic," or horizontally-scalable servers. These companies and products include Amazon Web Services, Microsoft Azure and Google Cloud, among hundreds of the others of niche providers. The advantages of these providers over running a private server are simplified dynamic sizing and pricing, load-balancing, off-siting support and technical services, and more. The use of a third-party computing provider means the hosting companies can build larger server clusters and take advantage of distributed infrastructure and economies of scale that smaller companies with limited computation needs would find unsustainable.

1.4.7.2 Hadoop

With increasing database sizes, traditional methods of querying information are not sufficient to process data and return it in near-real-time. Tools such as Hadoop or other map-reducing technologies distribute processing of large data sets by spanning information across multiple machines and executing queries on a subset of data as a parallel process. These fragmented setups are often more scalable for batch or offline queries (using Cloud infrastructure) than traditional database servers, which tend to be hosted on a single machine, or MySQL clusters, which span multiple servers. This is because Hadoop and other map-reduce systems running in the Cloud can

scale up computational resources, with Hadoop approaches allowing explicit control of data sharding (intentional distributed fragmentation) to reduce access time relative to MySQL's more random nature.

1.4.7.3 Data Mining

Historically, data analysis, mining, and machine learning have been relegated to use by experts with strong hypothesis or experience in trend identification. New software, available from PTC, Mathworks and hundreds of other companies, simplifies machine learning and allows novices to begin to identify patterns and create new, valuable applications from analyzing large data sets. These tools will continue to evolve and allow insights to be derived from Big Data with relative ease.

Data mining is key to developing applications taking full advantage of collaboratively-generated automotive data, and automated tools will aid developers in finding relationships capable of improving vehicle ownership and use.

1.5 Application Landscape

Surveys indicate a latent consumer demand for advanced vehicle technologies [13]. A range of applications exist to connect vehicles, occupants, and infrastructure. There are applications that run solely in vehicles, applications collecting local vehicle data to inform algorithms running in the Cloud, and V2V and V2I applications that depend on a free flow of information directly between nodes. There are additional applications making use of pervasive computation, such as person in car to Cloud, or device in car (an On-Board Unit, or OBU) to Cloud. This section includes a summary of the broad categories of vehicle applications, with particular focus on connected, informed, or collaborative data sharing applications, and how these applications embody or enable collaborative learning.

Collaborative or consensus applications are of special interest not only because they are newly facilitated by enhanced connectivity, but also because there is a growing use of data across industries. Many of the most significant opportunities for value capture

appear at these intersections between devices, services, and people. Collaborative applications grow human knowledge, with projects like Wikipedia; swarm robotics applications operate collaboratively [57], mobile phone sensing provides distributed data collection [58], iBeacon use [59] and data collection for customer shopping habits [60] allow pervasive tracking, Waze and other systems aggregate human and machine data to inform traffic patterns and even load level [61], while some autonomous vehicles map and localize, sharing information [62]. From intelligent factory dynamics monitoring to context aware applications or crowdsourced decision making, collaborative applications are a dominant force in Connected Vehicles and the broader Internet of Things.

Many collaborative automotive applications have been proposed, simulated, and even implemented at small scales. Many of these applications revolve around gathering and disseminating data [22]. These applications often focus on improving safety through improved knowledge. Though safety is an important feature for drivers, highly visible consumer applications are really what drive adoption and create the economies of scale needed to lower cost of Connected Car technology [63]. The landscape of intelligent vehicle applications to-date appears in the subsequent section, with emphasis on applications improved by collaboration, such as routing, data aggregation, group motion control, or meshed data sharing.

1.5.1 Local Applications

We define local applications a the most basic "smart car" applications, running in a vehicle or on an in-car mobile device. The scope of data for these applications is a single vehicle, usually measured from within the physical proximity of a vehicle and using its own sensors.

A basic in-car application might be the check-engine light on a car's dashboard, a media streaming application within the vehicle, or modulation of hydraulic fluid pressure for anti-lock braking. A local application running on a proximate mobile device might include Bluetooth remote locking and unlocking or live engine monitoring (using applications like Torque or Rev), while a more advanced application running

on a mobile device might use phone accelerometers, GPS, or microphones for single-vehicle failure prediction [55] [53] [54] [56] [64]. This same type of application can further enable e-Inspection of vehicle safety and emissions systems.

1.5.2 Telematics Applications

Telematics applications rely on the transfer of vehicle information to a remote server in order to provide advanced services, blending telecommunications and informatics [13]. Telematics applications make use of data from one or more vehicles in remotely-processed applications. These applications often take advantage of networked sensor data to run self-diagnostic services, provide navigation through use of a map-matching approach within a database, or providing location-based services [13].

Telematics applications may use sensor data to augment local applications and enhance customer satisfaction. An example application performs monitoring of sub-systems in order to identify likely issues early and keep consumer and warranty costs low, reducing unnecessary service hours and part replacement [11] [13]. In such an application, in-car computer modules transmit raw and aggregate data to a server for processing, visualization, and even customer engagement, *e.g.* appointment scheduling for oil changes. The "safety and security" idea is central to many telematics providers such as OnStar, with common application offerings including automatic collision notification, roadside assistance, remote door unlocking, voice services, turn-by-turn direction, and hands-free phone use [13].

Some telematics units integrate with infotainment systems for data visualization, *e.g.* showing GPS data on the radio head unit or dashboard [13]. Other services may integrate telematics data with fleet monitoring, for congestion mapping or maintenance operation. One study, for example, used taxi cabs – widely distributed and diverse vehicles – to collect information about traffic flow [65]. This study was able to gain near-complete data coverage of all the test cities, at extremely low cost. These data allowed for more accurate travel time and optimal route calculations, and they highlight the value in telematics data.

Generally, telematics applications are more purpose-built and less extensible than

Cloud or mobile applications. Telematics platforms might offer limited API functionality, but these APIs are designed with specific goals and application designs in mind. Compared with mobile applications, these applications tend to have an increased focus on security, safety, reliability, cost of ownership reduction, and increased consumer touch-points with long-term recurring revenue models. This is in part because telematics platforms offer "bread and butter" product offerings that can bring in revenue without implementing bleeding-edge technologies, and in part because telematics companies work closely with OEMs to ensure that their applications meet all of an OEM's certifications.

There are a number of telematics platforms available, with many, such as OnStar, OEM-owned or OEM-partnered. These platforms tend to be "closed" ecosystems with siloed data storage, fragmented APIs, and vendor-locked applications that work only with particular vehicle makes and models. A small but growing number of telematics providers, such as CarKnow[1], Vin.li, and Moj.io, provide open alternatives that offer cross-brand application development. There additionally exist research platforms for development, such as MIT's CarTel platform [66] which relies on local preprocessing of data and opportunistic connectivity.

Additional telematics-enabled applications include Pay-As-You-Drive (PAYD) or Usage-Based Insurance (UBI), which charges drivers based on true vehicle use and associated externalities as calculated from OBD data [67], or range prediction and electric vehicle battery state of health monitoring.

Application projection platforms such as CarPlay or AndroidAuto may enable local applications or telematics applications with the use of the paired mobile device's network connection [68].

1.5.3 V2V and V2I Applications

V2V data is unique in that it can provide rich, real-time understanding of local and remote vehicles' contexts to improve vehicle safety, efficiency, reliability, comfort, and convenience. V2V applications generally help drivers and vehicles to gather data that

[1]The author is the founder of CarKnow LLC.

is difficult or impossible to sense from within the confines of a single car, expanding the view of onboard sensors to include data from the local environment and vehicular neighbors [69] [22]. There has been particular research focus on developing V2V-centric network topologies to enable application creation [70]. It should be noted that V2V has been multiply-defined, with the acronym describing either a set of discrete event-based communications or fuller vehicle data sharing. This section explores both variants.

Several applications have demonstrated vehicular and infrastructure mesh networked applications and the implications of real-time neighborhood knowledge [70]. These projects are run by consortia, research institutions, corporations, and government initiatives which are well documented in Papadimitratos' (2009) Table 1. Papadimitratos surveys connectivity systems, discusses the use of connectivity to digitally expand a driver's time horizon, provides detailed coverage of major pilot projects, discusses CAN and wireless technologies in vehicles, and explores application types [1].

Other V2V and V2I research projects are summarized in Faezipour (2012) and Toor (2008). Some high-profile research examples include FleetNet, with its car-based, real-world field trials of position-based forwarding; MineFleet, with its aftermarket on-board unit for performance analytics and vehicle data mining, and CarTALK 2000, which includes a summary of the technology potential including communication-based adaptive-cruise-control and economic and congestion impacts of assistive connectivity [71]. There are a number of other initiatives that include theory and implementation based results along with thorough documentation. Some of these applications have been tested at scale and are nearing commercial viability.

The application space for these rich vehicle data is broad, with function ranging from dynamic vehicle control [72] to enhanced driver information [5] (versus today's paradigm of simple local sensing and visualization). Other application categories have been proposed, from road security, fleet management [73], navigation [73], tolling [74], and multimedia sharing [75] to parking payment [76] [77]. A summary of contemporary applications and their characteristics appears in Papadimitratos, Table 3 (2009).

Dedicated Short Range Communication and Vehicular Ad-Hoc Networks (VANETs) combine to provide a unique platform for developing communication-centric automotive applications. In Bai (2006), many applications and services based on DSRC are proposed and deployment discussed. These applications include those using wireless communication to enhance safety and efficiency, and create new opportunities for vehicle owners and OEMs through the creation of infotainment-centric applications. A good summary of these applications appears in Bai (2006), Table 1. Section IV provides characteristics used to classify and optimize the design of the application and network embodiment [33].

Other papers pose the creation of Intelligent Vehicle Networks (IVNs) based on more conventional telematics systems, with master nodes, adaptive network architecture, user-friendly diagnostic units, and safety units. These networks hint at a shift toward diagnostics, voice recognition, real-time traffic data, streaming media, and vehicle service appointment creation and management [13]. These applications are designed to increase interaction between the driver and vehicle and the vehicle and environment. Still other papers discuss media sharing, routing, collisions, security issues, and network simulations [39].

In this review, applications are broken down into four broad categories, based on a modified version of Wilke's network application taxonomy (covered in Wilke, Table III) [78]. These categories are "Information Services," "Safety Services," "Individual Motion Control," and "Group Motion Control," though many applications will transcend these boundaries. An alternative and broader classification schema instead relies on "Operation Critical" versus "Non-Critical."

1.5.3.1 Information Services

Information services are applications in which errors in transmission, such as delays or data lost in transmission, do not compromise vehicle safety. This is the case with applications such as remote vehicle dashboards or real-time vehicle health visualization [79]. Many of these services enhance users' comfort and ability to perform business and personal transactions while driving, in-vehicle, or viewing vehicle parameters on

remote devices [1].

1.5.3.1.1 Fault Prediction and Response One information service that consumers and mechanics value is a suite of data-informed diagnostics, prognostics, and driver-aware technologies. These services are a tightly integrated suite that extend conventional On-Board Diagnostics to predict remaining useful life using physical model-based or machine-learned trends [7]. These models enhance those run locally to a vehicle with enhanced context and comparative data.

1.5.3.1.2 Data Collection and Generation A category of Information Service applications is Information Generation Applications, including map generation [80]. As autonomous vehicles come closer to integrating into conventional roadways, high quality mapping data becomes increasingly necessary. Well-instrumented vehicles may use connectivity and localization data for simultaneous localization and mapping (SLAM) to improve accuracy of positioning while generating maps for local and broader fleet use [81]. These vehicles can fuse motion data and location data with a dynamic model for improved relative and absolute localization [82], or otherwise apply cooperative sensing to generating information facilitating decision making [83].

Technologies such as localization-based shared decision making, digital mapping, collision avoidance, and path prediction can allow driving behavior to be optimized. Frameworks for such applications appear in Caveney (2012), Table 1 [84]. Similar frameworks may be used for cooperative localization, to improve position estimates in temporal resolution or location accuracy [85]. These map-merging and pose estimation data sources even help to enable collaborate driving, by allowing self-localization to fuse with data from other self-localized vehicles along with relative positioning data, thereby improving accuracy of estimation [86].

Beyond mapping, vehicle safety sensors can combine with V2V technologies to enhance fleet perception data. Cameras, LIDAR, and RADAR can help to identify and report information hidden by the environment by providing improved context data to the vehicle and/or driver [87]. Other connected sensing data can improve

understanding of likely vehicle trajectories to improve comfort and efficiency. For example, in vehicles today, a driver has no indication of the throttle pedal position of a driver of another vehicle aside from noise and visual cues. With Connected Vehicle data sharing, it becomes possible to automatically extend perception and augment map building with richer data. One could generate a map layer with congestion, road hazards, or even driver pedal positions in real-time [21] [22].

These applications can help to improve a driver's understanding of their environment, but require well-defined semantics for map communication to provide the best results. This problem is partially addressed by translation of vehicle sensor data using platforms like Open-VSeSeMe to ensure the use of a common reporting language [23, 186].

1.5.3.1.3 Data Dissemination and Distribution Data Dissemination is a field of applications designed to share information and media between vehicles. Some of this content is generated by other vehicles, such as that which might be generated by a windshield-mounted camera; other content is generated remotely and merely utilizes vehicle networks to reduce data delivery costs.

There are several commercial applications proposed for DSRC to communicate with drivers to entertain, improve productivity, and allow access to external content [33]. These include music sharing, interactive gaming, Internet sharing from car to car, with incentives for drivers to use their vehicle as a gateway to share data, and more. Toor (2008) describes many of these applications in its survey of vehicle ad-hoc networks, describing application types and their merits and drawbacks [39]. Even transient car-to-car chat systems have been proposed [69].

1.5.3.1.4 Efficiency Improvement Efficiency Improvement applications share individual and aggregate vehicle information with other vehicles and infrastructure in order to estimate the density of traffic (the calculation of which suffers from issues of data sparsity) and to build data-informed models to improve the efficiency of traffic flow and reduce congestion [88] [33]. Localized region-specific data about

transportation system and traffic conditions are made available to drivers to enhance efficiency [1]. These models may be used to choose the best routes from a vehicle's location to a destination while maintaining the smoothness of traffic flow and reducing drive time [39] [89].

These routing data can be used for human-piloted or autonomous vehicle operation [90]. Cooperative traffic routing uses shared data from vehicles, with each vehicle in a city with GPS information sharing it with each other as Route Information Sharing, to design a globally optimal route to reduce congestion. This same data may be used for cyclic congestion prediction and routing adaptation or load-leveling, while related data may be used to improve vehicle power management, *e.g.* in hybrids [91].

1.5.3.1.5 Convenience Services Convenience Services improve vehicle driver, occupant, and owner comfort and convenience by freeing their cognitive facilities and hands to improve focus on driving or enable other productive tasks while driving. Convenience services might include fleet management, traffic routing, automated tolling, and vehicle tracking [21]. Some of these applications will help occupant even after they leave their vehicles, *e.g.* in the case of drivers using DSRC and infrastructure to locate their parked vehicle [92]. Services such as platooning or automated braking may also be viewed as a convenience service, as this application frees the driver for other tasks.

1.5.3.2 Safety Services

Safety applications monitor the environment and communicate from vehicle to vehicle to mitigate the risk of potential dangers. The difference between Safety and Information Services is that in Safety Services, delayed or corrupted data transmission may compromise safety. These applications include automatic braking and hazard reporting that can improve the driver's perception horizon. Safety Service applications assist the driver in order to enhance roadway safety by enhancing human capabilities and providing vehicles with rich context information [33].

1.5.3.2.1 Collision Avoidance Accidents are a serious safety issue with high personal and financial costs. Human reaction time is a limiting factor and contributing cause in many collisions. While humans rely on visual cues like brake lights, the reaction time for this stimulus is often not sufficiently fast to prevent a collision. In other cases, a driver may collide with another vehicle that itself could not react in time due to the driver's inability to see beyond the leading car. In this case, reducing the delay time — such as through the use of long-distance lead vehicle data communicated to the driver — or through direct reporting to a control unit, greatly reduces the risk of an accident. This lead-car example is a simple demonstration of a collaborative collision avoidance system [38]. Figure 4 in Biswas (2006) depicts a simulation highlighting the degree of efficacy when using collaborative collision avoidance relative to uninformed collision avoidance models. This and other simulations prove that collision avoidance may be applied to rear-end collision mitigation effectively when packets are sized properly and latency is kept low [93].

Cooperative collision avoidance may be used to enhance safety by improving forward visibility, reducing blindspots, and enhancing intersection navigation. Inter-car communication is used to provide 360 degrees of situational awareness ranging from a few meters to a mile, based off of rapidly-collected and fused position, speed, and heading data [94]. This environmentally-aware model supports applications mitigating the risk of forward collision, assisting intersection navigation, monitoring blind spots, and changing lanes.

1.5.3.2.2 Augmented Human Control Rather than wresting control from humans, V2V facilitates applications that can augment human control in order to keep vehicle operation in check. One example is intelligent vehicle speed adaptation – following radio-"posted" speeds as sent to a vehicle from infrastructure devices. Through maintenance of speeds appropriate for conditions, this technology could reduce accidents significantly – up to 20%, according to a 7500 vehicle study [20]. An added benefit is that speed limits may be transmitted based on real-time congestion and environmental data.

1.5.3.2.3 Hazard Reporting Because vehicle sensors can see and transmit data well beyond human perception, they provide valuable information for reducing and limiting the impact of accidents. Connected Vehicles may sense and report hazardous road conditions based on weather, location, time of day, and more [2]. Weather may be sensed and plotted using GPS positions, while pedestrians may be detected using night-vision or RFID tags and UHF readers, ensuring drivers are kept aware of vulnerable road users even when line of sight is not possible [23, pg. 88] [2].

1.5.3.2.4 Driver Monitoring Some safety applications can use data from within vehicles in conjunction with data from third-party vehicles or environmental sensors to monitor driver behavior, habits, and impairment. Such applications are in use in some heavy-duty vehicle fleets today, and these applications are shifting to become applicable to more vehicles, enhancing road safety [20]. This technology may help to reduce the risk of drowsy, angry, or otherwise emotionally-impaired driving, and can be used to generate profiles of risky drivers that may be communicated with monitors or other vehicles depending on severity.

1.5.3.3 Individual Motion Control

Individual motion control applications make use of vehicle sensors and data from V2V and V2I networks to issue warnings to the vehicle operator or to directly control the actuators of a single vehicle based on the context of the surrounding environment. These applications may be employed to improve safety, as in the case of collision avoidance, or to improve efficiency, like automated drafting, or even to ameliorate traffic, as is the case for automated lane switching [95]. Unlike locally run applications, individual motion control applications take into consideration data from external sources, providing an improved user experience and increased effectiveness.

1.5.3.4 Group Motion Control

Similar to individual motion control, group motion control applications use vehicle sensors and data from the vehicle and infrastructure network to control or influence

the behavior of multiple vehicles and drivers. This is often done in a cooperative or collaborative way to maximize a series of objective functions such as fuel and time savings, and this collaborative decision-making will ultimately feed into autonomous vehicle development [20]. These applications are more complex to develop, but offer the opportunity for increased impact relative to individual vehicle control due to scale as well as due to the efficiency benefits afforded by optimizing multiple systems simultaneously for the same objective.

1.5.3.4.1 Platooning Cooperative platooning was demonstrated as early as 2000 on a live test track, with full adaptive lateral and longitudinal control of vehicles informed via DSRC and differential GPS as inputs [96]. Intelligent stop and go, platooning to reduce fuel consumption, high-speed merging, and obstacle avoidance have been achieved via car to car reporting. These applications are adaptive and can even maximize the use of available roadway space to increase packing density, ease congestion, and improve traffic flow [96].

1.5.3.4.2 Intersection Control An extension of individual and group vehicle control uses vehicle network data and the ability to actuate infrastructure to automatically control intersection signals or even help vehicles' navigation of an intersection with no signal. Simulation results show that automatic maneuvering may result in traffic reduction, reduced CO_2 emission, and lessened fuel consumption. One study shows a 99% stop delay reduction, 33% total travel time reduction and 44% reduction in CO_2 emission and fuel consumption relative to business as usual [97]. This sort of application works best with a high percentage of connected vehicles, or else requires building substantial external sensing infrastructure.

1.5.4 Cloud Applications

A category of applications blending V2V, V2I, local, and mobile vehicle applications is referred to as Cloud Applications, or Vehicle to Broadband (V2B), in which vehicle data are stored in remote servers [2]. Cloud Applications are similar to V2V

applications, but rely on a different network topology to enable applications where connectivity may be sparse and data latency is not critical. While a V2V application might invoke a roadside unit for Internet connectivity, the topology is largely linear, with data passing through a chain of vehicles and possibly exiting at a gateway device or routing device. In a Cloud application, the network is a star topology, with vehicle data directly accessing the Cloud by way of a cellular interface built into or located within a vehicle (such as an on-board 4G modem, or a cellular device linked locally with Bluetooth Low Energy). The benefits of this structure are lower latency for some application types requiring external non-vehicle data, improved, direct access to data, the potential for more reliable connections, and simplified connectivity through reduced user interaction and the elimination of pairing requirements. The drawbacks to this approach include possible increased bandwidth, hardware, and power costs. Many V2V applications can be facilitated through the use of an intermediate Cloud, which also enhances the utility of vehicle data for other applications by enabling it to be stored and reused through standardized APIs.

1.5.4.1 Cloud Platforms

There are a number of Cloud platforms for vehicle data using direct car to Cloud connectivity through embedded cellular modems, *e.g.* OnStar, using aftermarket On-Board Units (OBUs) like the OnStar For My Vehicle (FMV) or Verizon Hum, or via a gateway device such as Automatic's Bluetooth OBU paired with a cellular phone. These different structures have different benefits for cost, convenience, latency, bandwidth, privacy, and security. In general, the star topology platforms, wherein the vehicle communicates straight with the Cloud rather than by means of a gateway device, have lower latency and enhanced security due to reduced likelihood of man-in-the middle attacks. However, gateway devices have the benefit of lower hardware cost and the opportunity to repurpose bandwidth from another device, making the effective operating cost appear lower.

1.5.4.1.1 Open Platforms Right to Repair and other data-activist movements in the United States and around the world have spurred consumer interest in vehicle data stored in open platforms. One such example is MIT's CloudThink platform. CloudThink is an open, interoperable, Cloud-translated aftermarket vehicle data collection and storage platform [98] [99]. There are a number of similar platforms, including those created by OEMs, such as Ford's OpenXC. OpenXC is an open, interoperable telematics platform supported by an automotive OEM. This platform excels for Ford vehicles, but does not directly support enhanced sensing data from other manufacturers. Other open and semi-open platforms include Moj.io, Automatic, Vin.li, and Zubie.

1.6 Application Challenges

When designing connected and collaborative applications for vehicles, there are a number of issues new and familiar to the automotive industry that emerge. Network architectures, consumer perception, privacy and security, sensing technology, and computation can all make, break, or challenge the deployment of applications designed to simplify vehicle ownership and operation. In this section, the predominant issues facing application and platform developers today are discussed.

1.6.1 Network Issues

Network issues are not new to automotive engineers, but the use of wireless connectivity and reliance on vehicle networks for safety-critical systems has led designers to focus on addressing new challenges having to do with data reliability, bandwidth and latency optimization, and sparse data sets. Network architects today must balance application needs against latency, bandwidth costs, and data safety, ultimately making the decisions about whether or not it is even worthwhile to connect certain vehicles or vehicle systems to the outside world.

1.6.1.1 Network Saturation

Network saturation refers to high network loads that can interfere with and threaten reliable transmission and receipt of data. In safety-critical applications, this can become an especially significant consideration, when non-receipt of data could allow an accident to occur. Limited licensable wireless spectrum challenges this network deployment and limits available bandwidth for application use [94].

Wireless networks have limited capacities, and interference from hidden nodes (vehicles outside a neighborhood) may affect performance of applications, while congestion in busy neighborhoods or self-competition between adjacent nodes can result in data collisions, delays in transmission, and packet loss [100] [38]. Even redundant broadcasting of messages can significantly reduce the reliability of data transmission and hinder accident avoidance and emergency response [101].

Additionally, in-vehicle network loading must be considered, as novel sensor types have the capacity to saturate low and medium speed busses. In CAN and other network types, a high network utilization percentage may interfere with transmission of stochastic message types. Network loading and in-vehicle bandwidth are relatively new issues to vehicle network design. With internal vehicle networks, network loading must be considered but the cost of transmitting a packet is negligible. With wireless interfaces, it is more difficult to add parallel networks to reduce congestion and each byte transmitted might, depending upon the technology used, have a cost associated with it. Instead, congestion management is necessary. Several approaches to congestion management have been explored and are shown in Eze (2016), Table 6 [102].

Vehicles today simply generate too much data to transmit it all – Hitachi estimates that a hybrid car can generate 25GB of information every hour [103]. Even safety-critical applications may not work on some network types, though newly designed network architectures partially ameliorate this problem [104]. To keep service levels high for these critical applications, some network architects choose to shift media and non-critical communications from meshed networks to dedicated cellular modems,

which increase data duplication, bandwidth use, and latency [105]. Clearly, appropriate network management is key to successful Connected Application deployment.

1.6.1.2 Quality of Service and Reliability

Not only must data make it to a recipient in a timely manner, but those data must also not be corrupted in transmission, during transit, or upon receipt. Simulation and analytical tools illustrate the need for Quality of Service (QoS) improvement in DSRC applications relative to current technology's capabilities [106].

Many issues in data quality and reliability in DSRC stem from designed-in low redundancy (due to network constraints) and non-receipt-acknowledgement of messages [1] [107]. In cases where retransmission is required to address this problem, latency is increased and further data collisions and packet loss may occur, or the information may not be received in a timely manner. Defining and maintaining a high QoS is critical to the success of Connected Vehicle applications and their widespread adoption.

1.6.1.3 Security and Authentication

As vehicle networks grow in scale, their data and outputs become increasingly attractive targets for malicious hackers. Vehicle control can be commandeered, data stolen, and user privacy compromised if proper precautions are not taken during the design and use phase for vehicle connectivity solutions. The risks for unauthorized access of data and control are higher than in conventional telephony systems [108].

For this reason, security must be addressed at all possible weak points – where data are generated, in-vehicle, during transmission, at a server, or in applications. Most critical to the success of vehicular applications is the assurance of trust that data is accurate and authentic, though credentialing and privacy are often at odds. With networked vehicles, eavesdropping is a real concern. Can data transmitted wirelessly be intercepted, and if so, what can be done with this information? Can identities be extracted from messages and deanonymized?

The creation and storage of local network tables for V2V and V2I routing presents

a weak point that facilitates leaking of identifiers. Without authentication and certification, devices can send messages to the network without their permissions being revoked, leading to the potential for Denial of Service attacks or other forms of intentional jamming [63].

Today, data can be "spoofed" such that an attacker could inject false information to divert traffic or cause emergency braking. Many systems today trust all incoming data, but things such as emergency warnings and location data should not be treated as trustworthy by default [109]. Location authentication is an emerging need for Connected Vehicles. Similar to spooging, DSRC applications may be jammed through wireless denial-of-service attacks [110]. Robust connectivity and malicious node blocking present challenges to deployment.

Solving these problems is not simple. There are many challenges to address regarding liability, performance, and credentialing that are often at odds with reducing the cost and complexity of vehicular subsystems, or prevent open data access. Many mechanisms for ensuring network security compromise safety, by creating or worsening network and computation delays [111]. Low overhead is required for authentication, anonymity, and certificate validation and revocation to ensure data are accurate and malicious senders may be removed from the network or otherwise ignored [112] [113]. These same threats to security plague OEM systems and aftermarket systems alike [114].

Ultimately, standardization is needed. While IEEE 1609.2 helps to ensure privacy and security of vehicle networks, there needs to be more standardization to ensure that nodes are truthful throughout the collection and transmission process [2]. Later sections will discuss issues with security once data are received at a server.

1.6.1.4 Sparsity / Market Penetration

High vehicle density is required to enable not only crowdsourced data driven applications, but also to enable multi-vehicle ad hoc connectivity [77]. This makes complete roadway coverage especially difficult in rural areas. Though techniques have been theorized to ensure stable information flow robust to changes in graph composition,

decentralized information sharing remains a challenge with mesh network connectivity [115].

It is not enough to have vehicles physically close enough to share data – a high fraction of cars on the road must be communication-enabled to fully support safety applications [94]. Even in regions where there is sufficient vehicle instrumentation to enable group motion control, with low enough semi-automated vehicle density, a single human-driven vehicle can cause chaotic chain reactions [116].

Solutions to this lack of density (both geographical and in installed base) may be addressed by lowering the cost of V2V and V2I hardware, or by making other wireless technologies more attractive, *e.g.* by identifying opportunities to reduce data transmitted and therefore reducing the impact of cellular bandwidth's cost.

1.6.1.5 Latency / Freshness

Some applications do not require high accuracy or high speed vehicle data, such as mobile applications for dashboard visualization. In these cases, today's network performance might be sufficient [104]. However, to be maximally useful to real-time vehicle operation, data must be "fresh" – that is, transmission must be delay-bounded and arrival or departure of data from nodes must be seamless, with low latency to support safety-critical applications [117] [118]. Data must cease to be treated as useful after a time and must be pruned to avoid having a net-negative effect on the system [119]. Enabling these rapid status updates can improve applications, vehicle safety, efficiency, or even applications providing feedback to drivers.

This means that emergency event detection must be fast – on the order of milliseconds – and of very high certainty to avoid false positives and to be proactive in the face of rapidly changing road dynamics [120]. False positives may be reduced through the use of intelligent broadcast topologies, but even these are not perfect [121].

Data freshness is complicated to evaluate, as communication is often either high power or high latency, and processing between hops can be significant – as much as 10s per 1km for V2V applications, assuming there are enough vehicles to ensure robust connectivity [94] [109] [116]. Power limitations and the potential for vehicular ob-

struction require adaptive communication to ensure sufficiently-low latency to enable safety applications [122]. Networks today, like 802.11P, are adequate for small systems (about 5 vehicles), but require additional intelligence to reduce transmission delay [94]. Problems of queue filling and similar are especially bad in dense environments, where time-critical message dissemination is not possible [32].

It is not only the network that contributes to these delays – decision on what data to collect, process, filter, and transmit can have significant impact, as can additions of layers designed to improve robustness and security [2] [109]. Clever design and migration from thin-client to thick-client models of data transmission, wherein data are aggregated and processed locally prior to transmission, may be used to ameliorate some of these delays.

A focus on data minimization and optimizing the resource use for selected applications will help to ensure that data are generated and transmitted in a time-efficient manner.

1.6.1.6 Mobility / Network Transience

Mobility is as much a challenge as a virtue in designing and operating mobile networks. The network changes whenever a new vehicle joins a neighborhood, making the maintenance of a consistent network arrangement a challenge [123]. Node density changes can cause fluctuation that can easily overwhelm fragile network systems, while retransmission due to data collisions reduces overall throughput [2]. In cases where the network is robust, routing and connectivity in VANETs is still a challenge, though mobility may be used to improve performance of the network through MANETs provided there is sufficient computation to keep reliable lists of active nearby nodes [39]. Here again, facilitating a shift from meshed to cellular technologies may be an attractive design alternative.

1.6.1.7 Protocol Design

Designing a sustainable set of standard protocols for V2V, V2I, and broader vehicle connectivity is a challenge. There are problems with computational overhead in

proactive schemes [40]. Many "greedy" algorithms for location-based packet forwarding may lead to gaps and ultimately additional data (re)transmission [1], and 802.11P is designed for slow varying connections, with packet lengths too long for channel coherence [124]. Vehicle-to-vehicle networks suffer high data loss stemming from quick update rates, which worsens when moving [123]. Some of these issues are addressed by new network topologies [123], though routing for user applications is not yet clearly defined. MANET networks are unicast while geographic routing takes advantage of the location data of a vehicle to make routing more efficient, but there is no standardized mechanism that addresses the concerns from multiple application categories [39].

Many vehicular networks do not scale well. Cluster size grows with the traffic load. While this loading improves data density, it simultaneously challenges network stability due to transmission collisions and other errors [125].

Existing protocols for networking are not easily repurposed. For example, it is difficult to imagine networking vehicles using TCP/IP due to the significant associated computational overhead. TCP was designed for wired networks and is therefore not ideal for VANET or MANET use. Network access lacks fairness; it is difficult to distinguish between congestion artifacts and packet loss, with losses of connectivity interpreted as congestion and decreasing effective data throughput [39].

In all cases, more real-world simulation and data collection is necessary [2]. For example, VANET design and implementation varies regionally, and as such, additional simulation is necessary to validate communication models and their variation when modified to work around the world [124].

1.6.2 Vehicle Architectures

1.6.2.1 Security

In-vehicle security is an emerging challenge as evidenced by the recent hacks of Fiat-Chrylser Corporation vehicles, Teslas, and theoretical exploits discussed in academic literature [126] [127] [128] [129] [130] [131] [132]. This is in part because vehicle software complexity has increased greatly in the last several years, and in fact, vehicle

software is now one of the key differentiators of innovation in automotive sales. Broy (2006) discusses this shift from hardware to software innovation well in Sections 5 and 6 [24].

While security has historically been an oversight in vehicle design, it has become imperative for OEMs to harden their vehicles to ensure the reliable, real-time operation of critical systems under more constraints than ever before. One approach to this is to shield critical actuators and memory from the network by splitting systems and gatewaying information to allow separation of critical and convenience networks [1] [109] [2]. Secure gatewaying of data with requestor authentication helps to prevent commodity devices from taking advantage of convenience entry-points to the vehicle, [109] by preventing masquerading, impersonation, flooding, jamming, spamming, and the spread of malware to secure vehicle access control and protect against theft [23, pg. 61-62] [2]. A formalized approach to designing vehicles securely is described the SAE's cybersecurity guidebook [133], though technical solutions addressing vehicles already on the road are still needed.

To secure connected vehicles, future platforms are beginning to abstract the physical car from the digital car to minimize the risk of unintended intrusion. Creating Cloud mirrors of physical vehicles will allow for this abstraction while shifting data handling to the Cloud, where computation is abundant and certificate and credential validation is more tractable.

1.6.2.2 Sensor Payload and Implementation

Beyond simply designing network protocols and applications to use rich vehicle data, automotive engineers must now begin thinking about how to implement and communicate data from next-generation sensors. For example, vehicles today incorporate RADAR systems, but these systems are often black-boxed. These sensors aggregate data internally, and report limited, processed values to the broader vehicle network. Making richer data available for computation and communication will enable a new host of applications, provided these data can be stored and transmitted effectively. Sensor payload configuration poses big questions of the 'why' and 'how' of access-

ing richer sensor data, such that these data maximize application utility without compromising the vehicle network or dramatically increasing system cost.

Future Connected Vehicle applications must thoughtfully determine the appropriate sensor payload for vehicles by considering the vehicle's eventual application payload. As vehicle lifetime increases, the decision to include or exclude a particular sensor or actuator can have far-reaching impact.

1.6.2.3 Thin Versus Thick Data Processing

Thin and thick "client" design is an architectural decision relating to the amount and type of data to be transmitted out of the vehicle and to a remote system. With a thin client, data are transmitted exactly as they are captured, whereas a thick client transmits preprocessed or filtered information. Thin- and thick-client systems aim to find a balance of power, computation, and data accuracy.

An example of the importance of this architectural decision comes when attempting to estimate pose and orientation of a vehicle. While in an example system, a sensor may be able to sample at 10 Hz, data can only be transmitted at 3 Hz. In such a case, pose estimation is best performed in a thick client, by aggregating sensor data and filtering locally prior to sending computed pose estimates to a server. The data inputs to the model provide more utility at the 10 Hz sampling rate than the 3 Hz the server could gain access to, even though computation is more readily available at the server side. In vehicles, a similar application might be estimation of vehicle miles traveled, which can be aggregated in-vehicle and transmitted to the Cloud, or the raw data used to compute the distance may be transmitted to the server directly and without processing.

The tendency to shift computation from embedded systems to the Cloud will further complicate this decision in the future, as bandwidth, storage, and computation costs and the value-add of scalability must be considered.

1.6.2.4 Radio Technologies

Radio technology is an important consideration for connected cars. Modern vehicles may have radios ranging from cellular connectivity for phone calls to mesh networking for V2V applications to WiFi or Bluetooth [134]. The radio technology chosen for use in a vehicle has many implications on application feasibility; each technology has different range, latency, bandwidth, and cost constraints, along with different market penetration. Additionally, radio systems have differing robustness to motion, line-of-sight obstruction, and antenna design [135]. Handoff mechanisms may be used to fuse communication across radio types seamlessly and with little computational overhead [136]. In spite of this, the cost of installing radio modules in a vehicle is not insignificant, so the design decision of what networks to access must be taken seriously.

1.6.2.5 Localization Technologies

Localization is a key enabling technology in data-informed vehicular applications, especially with respect to group and individual motion control. However, many localization technologies suffer from issues with accuracy, reliability of fix, and precision. Commercial GPS accuracy has limited resolution, slow to acquire a fix, and easily interrupted and might not be accurate enough to identify possible collisions with certainty [1] [137] [21]. Inertial navigation has become mainstream, but MEMS sensors are subject to drift over time and temperature with dead-reckoning approaches. These issues in accuracy can be addressed in part by storing and referencing accurate digital maps (onboard, to reduce bandwidth and latency, or streamed over a high-speed network), but this requires the generation of very high accuracy maps and can drive up the cost of a vehicle [94]. A simpler approach fuses odometry data with MEMs data through a Kalman filter or a variant.

1.6.3 Standardization

Standardization is an important issue in Connected Vehicle technologies. Without standardized technologies and communication semantics, vehicles would be unable to communicate or share information effectively.

1.6.3.1 Network and Radio

There are different radio standards in North America, Europe, and Japan. These multiple radio types must be harmonized to simplify the production and support of extravehicular networks [23, pg. 76]. By standardizing radio networks, a cohesive network with minimal interference becomes tractable.

1.6.3.2 Semantics and Interoperability

Vehicles must be able to communicate with each other using a common language. Sharing across vehicle makes and models, especially when supporting non-standard applications, is a difficult challenge [21]. Semantics to communicate map data, hazard data, and other real-time information, must be created and universally supported. This assures that richer mapping data can be generated and used across manufacturers.

1.6.3.3 Scalability

Vehicle-to-vehicle and vehicle-to-infrastructure applications and platforms must be built in a scalable way, with user-re-definable protocols to ease development [2]. Without scalable technologies, applications will suffer from short effective lifetimes and be unable to endure for the life of vehicles. Scalability additionally makes adding new features to Connected Vehicles simpler, as in the case of vehicles that reduce local computation and shift processing to the Cloud.

1.6.4 Computation Issues

Computation is a consideration for Connected Vehicle applications. Computation can occur in-car, at a connected node, or at a remote server. The location of computation

depends on the computational intensity, latency requirement, and cost of bandwidth. Cloud computation is more scalable than in-car computation, which tends to be installed once and let to languish for the remaining life of the vehicle. This computation comes at a likely cost increase relative to static storage due to the bandwidth required to interface between the car and a remote server. In-car computation has reduced latency relative to a Cloud solution, with a higher initial cost and poorer scalability.

1.6.4.1 Data Accuracy

Data accuracy is an important consideration - each application can have a cost of false positives and false negatives as part of an accuracy metric. False positive reports of hazards can be costly to vehicle operators. For example, a V2V-Connected Car might stop unnecessarily based on a false positive of a hazard ahead, and be rear-ended by a human operated vehicle [101]. To succeed, V2V applications must be able to eliminate false positives and duplicate warning messages through enhanced network architecture [120]. In addition to simple positive/negative identification metrics, time of data accuracy is also an important metric. For example, a stalled vehicle poses a hazard when it is stalled and on a main road, but even after the car has been removed the related congestion may remain. Properly characterizing this event and eliminating false positives and duplicate warning events is necessary for the successful operation of a hazard-avoidance application [120].

1.6.4.2 Big Data

As vehicles continue to generate more data and these data begin to be logged remotely, the need arises for the use of specialized data processing and analysis tools. There exist tools for storing, aggregating, processing, and serving massive data sets that facilitate Connected Car applications. Data sets are already too large to transmit or store in a cost-effective manner, though modern technology and business models make it possible to still capture a significant portion of the data generated by vehicles. Cloud computation has facilitated novel uses for vehicle data, but brings with new opportunities, additional risks, and challenges.

1.6.4.2.1 Data Storage and Processing The proliferation of data in vehicles ready to transmit to remote servers poses a unique set of problems that must be addressed. The decision to instrument and connect vehicles comes at great cost – Tesla's decision to bundle wireless connectivity with vehicles likely added millions of dollars in cost for the manufacturer. Though the received data offers valuable insights, simply managing the network infrastructure is a challenge [138].

Often, data collected en-masse is too large to process efficiently, leading to questions about data handling. These questions consider how and where to process data, the use of centralized or decentralized storage, and how to split and reference data to keep track of large fleets [2]. These architecture decisions have big implications, as query and computation time factors into many Cloud-based decisions and can facilitate or prevent the development of new branches of applications. A benefit of Cloud-computation, rather than in-car computation, is simplified scaling of computational resources. Many web platforms can scale storage and processing on-demand without the need for application reconfiguration [139].

At fleet-wide scale, the most scalable approach to data storage is to run a private Cloud, or to lease time on a scalable elastic computing platform. Amazon Web Services is a frequently used offering, with current pricing available from `https://aws.amazon.com/s3/pricing/` as of 1 March, 2016.

1.6.4.2.2 Preprocessing and Aggregation Sharing raw data with applications can be helpful, but is often not the preferred way to share information due to the potential to overload end-use devices and waste network resources. Calculating key metrics is one approach to reducing the size of representing a vehicle on a server.

Data aggregation refers to the concept of joining, merging, and providing aggregate metrics for data from multiple vehicles. There are many architectural decisions relating to data aggregation, including selecting what objects and metrics are aggregated, where the computation is performed and when aggregation occurs.

This decision, much like the decision of whether to transmit data from car to a remote node as a thick client or a thin client, depends on the cost of bandwidth,

storage, and computation.

1.6.4.2.3 Honeypots Centrally-stored Big Data presents a set of security challenges not faced by smaller data sets or in-vehicle data. Massive data sets are attractive targets for hackers because one entry point enables access to a rich set of information. The value of data increases with increasing data scale, as is facilitated by standardization and common communication and storage semantics. The use of a common Cloud may seem attractive for enabling applications, but for the same reasons a common Cloud facilitates applications, it attracts unsavory actors. When considering the possibility of deanonymizing data or directly or indirectly actuating vehicle functions, these targets become more valuable and demand increased hardening against attacks.

1.6.5 Infrastructure

Setting up an environment conducive to V2V and V2I networking is difficult, time-intensive, and costly. The existing environment must be considered to optimally place infrastructure units, though this task is complex when considering radio and range issues [140] [141]. There are costs associated not only with the purchase of nodes, but leasing of space, providing support requirements such as power and bandwidth, and maintenance. The infrastructure configuration required may also change over time, as the standards evolve or if a non-standardized technology was chosen for initial deployment.

1.6.5.1 Network Effects

Collaborative vehicle applications require a high density of Connected Vehicles, but V2V and V2I technology have historically suffered from limited market penetration and slow growth [21]. Density is needed to increase the value of the car-to-car model, and therefore to better justify the cost of radio and computation components. The required density depends on the application; however, the majority of safety, congestion and efficiency applications depend on network effects to a high degree.

V2V and V2I technology lacks significant penetration at present, though rapid growth is expected, with studies anticipating between 40% and 62% market penetration in 2030 or 2027, depending on the conducting firm [142] [143].

1.6.6 Social Issues and Public Perception

As drivers become more aware of the information their vehicles can generate and share with third parties, concerns emerge relating to how these new technologies will perform and impact daily life. Drivers and fleet managers express concerns about location and speed tracking, becoming over-reliant on new technology, the difficulty integrating new vehicles into non-connected society, and they have concerns over how the systems perform during inclement weather [8]. These issues are not always founded in reality; messaging and clear policies are a necessity to minimize the hurdles posed by perception issues.

1.6.6.1 Privacy, Sharing, Anonymization, and Data Ownership

Users have strong demand for data security and privacy [8]. This has been shown time and again in aftermarket Connected Car applications [98]. Consumers enjoy reaping the benefits of connected data but are not always willing to share personal information with others. Anonymization improves consumer acceptance, but this is not feasible, as many applications require invariant identifying information to work properly, *e.g.* applications that require a time and location history for a particular vehicle. Even anonymized information may be deanonymized when examining route trends or other indicators correlated to driver. Despite this, some drivers will willingly give up their privacy in exchange for financial or other (e.g. improved user experience) incentives [144].

Data ownership is a concern across technologies that are newly connected, such as Connected Vehicles, Connected Homes, and across the broader Internet of Things. There is a consumer demand for data and historic models, while at the same time models where automotive OEMs or logistics providers own all vehicle data are coming

under scrutiny [145]. Clear data ownership policies and sharing tools are required before collaborative applications can truly succeed.

1.6.6.2 Cost of Implementation

The cost of implementation for these technologies has historically been a challenge [116]. Beyond the cost of infrastructure deployment, like roadside communication nodes and sensors, there are costs associated with deploying radio technology, enhanced sensing, and computation in vehicles. While sensors like RADAR provide valuable information to local safety applications, they are presently prohibitively expensive and market penetration will only increase with decreasing sensing costs [21].

Depending on the selected technology, there may be high bandwidth costs to consider. These costs may be prohibitive, due to a low willingness to pay for safety features and safety-related communication (*e.g.*, for collision avoidance) [146] [8]. This effect may be mitigated or offset by added comfort and convenience features, though it is unclear how significant this effect will be. Recent surveys have applied adaptive choice-based conjoint analysis to show the relative rankings of Connected Vehicle technology importance and found contradictory results for consumer willingness to pay, with consumers demonstrating a preference for safety-centric features [147].

In January 2016, the U.S. government agencies announced a four billion dollar government program to help subsidize autonomy and Connected Cars [148]. This is expected to facilitate research and technology deployment for some, but not all, connective technologies, which may in turn lead to the creation of lower cost or more useable technology. With government-backed programs, regulatory concerns surrounding data ownership, privacy, and security must be addressed before data sharing can enable full vehicle autonomy, automated vehicle functions, or safety-centric vehicle control [146].

1.7 Opportunities for Technology Improvement

This review of the Connected Car landscape has assessed the current technology situation. This section identifies areas for improvement to ensure the future success of Connected Vehicle technologies.

1.7.1 Vehicle Design

A key issue in connected technologies is the consumer willingness to pay. To incentivize vehicle owners and operators to pay for sensing and connectivity enhancements, these technologies are best best received by consumers when bundled together and marketed as improvements to comfort and convenience features. Some, for example the use of improved dead reckoning localization, or enhanced radio connectivity through the use of conformal [149] or multiple antenna configurations, have consumer benefit that is immediately perceptible. In these cases, enhanced localization improves navigation, while improved connectivity ensures improved reception and therefore application functionality regardless of environmental factors.

To gain the most value out of rich sensing data, suppliers and OEMs must be encouraged to un-black-box sensor data. Today's sensors aggregate data internally and output a limited, filtered set of the data they generate. In the future, applications may require access to these data in full. To this end, transparency, or remotely-deployable, over-the-air software updates [150] [68] may enable fuller future use of sensor data.

The On-Board Diagnostic network in vehicles has remained unchanged for nearly a decade [27]. With emerging technologies, this set of standards must be redefined and refined so as to be more extensible. This rethinking of OBD might include improvements such as standardized addressing conventions, encoded metadata to allow new data types to be understood and accessed without requiring the release of an updated standard, and might allow for the access of memory in addition to standard parameters, to afford enhanced diagnostic opportunities. A common, open repository for diagnostic trouble codes and sensor parameters, as well as a common

sensor gateway architecture, would further enhance On-Board Diagnostic's viability as a data source for connected and collaborative automotive applications, and beyond. This sort of pseudo-standardized sensor interface would provide for economies of scale in sensor deployment and software development, reducing the cost of implementing connected technology. Finally, inclusion of improved freeze-frame data storage, multi-PID (Parameter IDentifier) request handling, and automatic, intervalled responses would unlock significant potential for enhanced diagnostic and prognostic applications.

1.7.2 Communications Design

Network architecture is a an important consideration in designing Connected Vehicle applications. While physical layer standardization is important, and choosing and backing a single standard would simplify development complexity, standardizing data interchange – rather than communication protocol – is more critical. This is because future technologies will reduce latency and bandwidth cost for most RF technologies. The best choice of action today is to remain "technology agnostic" and standardize only messaging, semantics, and data gateways. In practice, the use of common semantics might also be used to reduce data transmitted – providing all vehicles with a standard encoding and decoding library with which to compress messages. This has started to occur, with the creation of standardized Cooperative ITS messages [151]. Regardless of the direction taken, backwards-compatibility should be considered to ensure maximum long-term utility of Connected Vehicle hardware and applications.

Despite planned technological agnosticism, there are still a number of network design challenges that might be explored. For example, seeking to invent technology to minimize data transmitted and to reduce data collisions and resultant retransmission might apply to any communication technology. Investigating the use of a multi-network pathway, *e.g.* a mesh of short-range and cellular communication, may enable the best of multiple world for applications, with low latency in-network and long range, with the tail vehicle in one neighborhood informing the lead vehicle of the closest trailing neighborhood of an event or hazard. Further integration of direct-to-Cloud via cellular will reduce need for high vehicle density for low-latency applications (by

eliminating inter-vehicle communication reliance), and regulated priority queues may be used to allow emergency messages to pass while media is disrupted.

Though there is a gap between the development of VANET protocols and automotive applications, current research provides many possible options for making the best use possible of a given network [33]. For example, transmitting data about a hazard on one side of a divided highway to a car on the other side might be informative but not useful. Based on network cost and loading constraints, it might be best not to send this message [116]. Directional or trajectory-based routing will help with optimal path mapping (identifying the shortest path or lowest cost network vector) based on distance, latency, hops, vehicle speed, or cost, [152] [40], or clusters can self-organize [125]. Latency, redundancy, reliability can be improved by choosing which vehicles must rebroadcast [101].

Proposed approaches to routing in highly dynamic networks make use of condition-based messaging to reduce congestion and latency. Using conditional addressing instead of network or location-based addressing ensures higher accuracy of transmission and minimizes resource waste. Relay nodes and recipients are set dynamically based on conditions included in the message. This is similar to a GeoCast message, but more expansive in scope. Messages can be forwarded upward to applications or forwarded to a neighbor. These forwarding conditions are provided by applications [153].

1.7.3 Platform Design

Platform design is an area ripe for exploration, with the ability to redefine how Connected Vehicle applications work. We have considered how to build scalable, extensible platforms for collaborative vehicle applications.

Any application platform should be made open, interoperable, and extensible to thrive. Openness helps to improve transparency and security while reducing development costs. Interoperability reduces the risk of vendor-lock and ensures that connected applications have the richest possible data sets available to them. Extensible platforms can be integrated most easily with other devices, services, and people. Designing a Connected Car platform meeting these goals will ensure long life and utility, once

accepted.

Beyond describing platform ideals, developers should seek to integrate mobile devices where possible to augment the sensor and communications payload of a vehicle. This lowers application and deployment cost through the use of pervasive sensing and provides a future-proof solution that can be easily upgraded, unlike the invariant automotive systems today.

Any platform must effectively manage data. To that end, the most successful platforms will develop technologies to reduce data transmitted for use in applications, to save bandwidth and reduce impact of latency. These platforms will rely on a framework for sampling the richest possible vehicle data with minimal sampling rate, through improved data aggregation, the use of dynamic sampling rates, and the implementation of thick-client models for applications where local computation is possible. Such a system could be modeled after a rate decreasing algorithm, or a data-informed likelihood model [120] [154]. These data will be transmitted with a "Quality of Data" metric, incorporating details of where, how, and by whom the data were generated, to aid in measuring data trustworthiness.

Finally, Connected Car platforms should explore shifting computation from car to Cloud, where feasible, to improve system longevity. Running consumer-facing systems from the Cloud will allow a fresher, longer-lasting user experience, and can take advantage of the Cloud for calculation in computation-intensive applications by shifting computation where it is more scalable.

1.7.4 Application Design

Modern Connected Car applications have a plethora of inputs and outputs available for use. These applications must be designed with consideration for whether the vehicle needs data from external sources or if these data can be stored a priori. This decision of application locality involves exploring the use of thick/thin clients based on computation, storage, bandwidth needs, the optimal split of real-time and free (DSRC), long range real-time and costly (3G), long latency and free (WiFi), and the future trends in the cost of computation, storage, and communications. These

decisions may often result in hybridized solutions, such as the use of local data buffers and last-mile distribution through DSRC or another low-cost solution [105]. The choice of communication method can have substantial impact on an application's performance, as in the case of Miller's 2007 sparsity simulation showing that with V2V, 10% of vehicles transmitting speed and location can produce similar results to V2I with only 10% of the bandwidth [155].

From a functional perspective, the best applications will seek to understand how best to augment human control using ADAS as a step on the way to replacing it with autonomy [156]. Today, this includes lane-keeping, stop-and-go traffic handling, adaptive cruise control, night vision, tire pressure sensors, and blind-spot detection [157] [2]. Applications like these facilitating improved comfort, convenience, or efficiency will be an inroads into consumer vehicles and a primary driver of connected technology deployment, including applications like human-parameter informed adaptive cruise control [157].

1.7.5 Security

Connecting vehicles to the outside world opens a host of vulnerabilities. Security must be assessed and improved at three points: in-car, car-to-x, and in the remote device.

In vehicle, the security focus is on preventing unauthorized actuator or sensor access. Some of these issues may be addressed with intelligence at the gateway level, to authenticate incoming requests and limit data access and reply rate by parameter, requestor, and more. This could be accompanied by an improved seed/key challenge and response system to that found protecting critical modules today, making use of additional data, such as certificates or other characteristic "heartbeats" [39]. A vehicle network may occasionally validate the presence and functionality of other devices on the network in order to ensure that the hardware has not been moved to a new environment, making spoofing data more complex, or otherwise evaluate the reputation of a connected device [23, 66].

The car-to-x link may employ traditional communications security strategies, such as revocable certificates, encryption (depending on computational overhead), and other

approaches designed to make man-in-the-middle data interception a challenge [39]. This mode of transmission may be structured to ensure that the intended recipient is the only node with the key to decode the message, locking intermediate nodes out [63]. These certificates and signatures may make use of pseudonyms to make deanonymization more complicated [39] [158].

Technology such as distance bounding may be co-opted from other domains to examine the feasibility and validity of messages. The degree of security will often depend on the type and criticality of the application in question.

Cloud security is a challenge that must not be ignored due to the significant value in aggregate data. Connected Cars have the potential to put expensive and dangerous equipment on the Internet, so the risk of unintended or unauthorized actuation must also not be taken lightly. One way to improve the security of the car-to-Cloud link is to abstract the digital duplicate of a vehicle from the physical vehicle, allowing applications to interact only with the digital duplicate. An intelligent gateway may relay approved commands to the vehicle after checking to ensure these commands will not violate predefined or learned rules. Combined with clear guidelines for appropriate data use, protection, and anonymization, it is possible to build a secure Cloud or other remote repository for vehicle data storage and service.

1.7.6 Social Issues and Public Perception

Social and perception issues compose one of the biggest challenges to Connected Car technology deployment. With positive perception and social implications, even factors that presently discourage the use of this technology, such as cost, may be overcome. To that end, Connected Vehicles must provide clear guidelines on data ownership and privacy [98], as well as simple control systems to visualize and modulate data flow control [159].

A University of Michigan study shows a high positive perception of Connected Vehicles. These drivers indicated significant faith that CV technology will improve safety, reduce congestion, and reduce emissions, shown in Table 3 of Schoettle (2014). People perceive connected and automated cars as leading to fewer crashes, reduced

severity of crashed, improved emergency response, lower congestion, shorter transit time, reduced emissions, better fuel economy, reduced insurance costs, and reduced distraction [8]. Delivering on this would be favorable to consumers and presents a significant opportunity space that consumers have indicated a willingness to buy into. Provided applications can be deployed to address these areas, consumer adoption may be accelerated. The use of mass transit to raise awareness should also be considered, as should integration with personal communication devices, for which 50% of consumers would be willing to pay [160] [8].

1.8 Conclusions

The Connected Car industry is growing rapidly, but it is in no means new. The primary enabling technologies are in place today – computation, sensing, and networking – as is a consumer demand for the sort of applications only connectivity can facilitate. The applications witnessed to-date are research focused but have far-reaching practical implications. These applications include highway platooning, collision avoidance, hazard warning, usage-based insurance and beyond. There exists an untapped opportunity in more consumer-facing applications, focusing on optimizing user experience, human factors, and improving the total cost of ownership of a vehicle through fuel, time, and maintenance savings.

Reviewing the technology and opportunity landscape shows great growth and future potential for the Connected Car, though growth to date has been unstructured and occasionally haphazard due to fragmentation. Following comprehensive literature review, we identified several challenges: privacy, security, scalability, and extensibility are all key architecture informing design points. These challenges present unique opportunities to address, and the ability to unlock great potential.

To further growth of Connected Car applications, we propose a set of recommendations for next-generation automotive technologies, from On-Board Diagnostics to in-vehicle and full system network architectures. Perhaps the most impactful facilitators for the next-generation Connected Car is the creation of a platform for

resource reduction (to lower variable deployment costs), digital proxies (to improve security), and a framework for application locality (to determine the optimal use of computational resources). Expanding from local vehicle sensing toward a more pervasive approach, leveraging existing infrastructure like smartphones, will drive further growth.

The findings here, though derived from an exploration of the Connected Car space, are equally applicable in several other growing industries. The use of Cloud mirroring, Data Proxies, and pervasive sensing, for example, will help to grow the Internet of Things well beyond its present embodiment.

Relation to Other Chapters

This chapter took stock of the Connected Car landscape and identified problems stemming from technology and perception issues, as well as opportunities to use vehicle data to enhance user experience.

Problems identified included difficulty managing resources, the perception of inadequate privacy protection, difficult-to-enforce security, and poor standardization of in-vehicle and intra-vehicle sensing, diagnostic, and connective technologies. Addressing these opportunities allows applications to take advantage of rapid data growth and actuation possibilities afforded by connectivity, enabling the creation of create consumer-facing software improving vehicle reliability and efficiency. Other applications may use collaborative vehicle data to inform the design and realtime operation of future vehicles. Optimization of vehicles at scale has the potential to save drivers time and cost, as well as greatly improving service life and vehicle efficiency, reducing emissions.

Though the issues mentioned evolved as a result of exploring Connected Vehicles, these same issues must be addressed to ensure successful adoption and growth of all facets of the Internet of Things. One sees similar challenges facing wearable devices, Smart Factories, Connected Homes, Intelligent Infrastructure projects, and more. Security and service life, bandwidth and batteries, and even network loading

are challenges that must be addressed before connectivity deployment may grow unfettered and the true value of the Internet of Things may be unlocked. Creating a secure and resource-efficient implementation for connected systems would address these concerns and enable new technology and business models taking full advantage of the information and actuation connectivity brings about, even allowing the creation of new products and services at the intersection of existing industries.

In an attempt to address these most pressing issues, Chapter 2 poses theory to improve resource management and security for Connected Cars, the Internet of things, and all connected devices. From bandwidth to battery savings, effective resource management will lower costs of connectivity deployment and enable the development applications that are infeasible today. With novel theory and the support of a platform for storing, aggregating, and serving vehicle data, impactful applications may be created at the intersection of today's connected devices and platforms.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 2

Re-Designing the Internet of Autos

In Chapter 1, we identified a number of issues in connecting vehicles to one another, to people, and to wider infrastructure, as well as considered the benefits of intelligently connecting "Smart Things." The intermittency of connectivity, high data carriage and power costs, and risk of malicious access have to-date precluded many important applications. In Chapter 2 we explore theory and present an architecture for connectivity capable of minimizing resource use and improving security in connected systems. This same architecture is applicable for Connected Vehicles and all other Cloud-based Internet of Things devices. The presentation of this architecture is followed by an examination of the costs and design decisions associated with automotive application development. This chapter draws material from two pre-publication manuscripts: "Application of 'Data Proxy' Estimators to Minimizing Resource Use and Improving Security for the Internet of Things" and "Automotive Application Locality: The 'When' and 'Where' of Connected Vehicle Applications, as Applied to Idle Time Prediction."

First, we explore the concept of an architecture based upon "Data Proxies," estimator- or observer-backed digital duplicates of physical objects or systems run using Cloud resources. The need for such a system, its structure, and a typical design process are explored. Platforms created to support the Data Proxy architecture will allow the rapid creation of "Cognitive Layers," or the computational equivalent of a biological autonomic response system capable of identifying system faults and

preventing malicious command execution intelligently, adaptively, and unconsciously (the same way that the human body can react to stimuli using the brain stem, rather than the brain). The second half of this chapter explores design considerations for Connected Vehicle applications in the context of an idle-time estimation application designed to improve fuel economy and user experience. This exploration examines how factors such as connection type and the location of storage and computation dramatically impact system cost and viability.

2.1 Creating a Secure, Efficient Architecture

We first propose the concept of a Data Proxy and this Proxy's related architecture. There is significant potential in connecting devices, platforms, services, and people. However, there exist challenges to the creation of pervasive connectivity in the form of power and bandwidth constraints and fear of insecurity. To ensure that even resource-constrained devices can belong to connected systems, inputs must be minimized system-wide. Approaches to date have focused on reducing requirements for just one element of a system, such as network loading, and focused largely on optimizing data transmission using resources on the device itself. We instead need an approach where systems are optimized using computation and power where it is more available, less costly, and more easily scaled — in gateway devices or in the Cloud. The following sections introduce the concept of the "Data Proxy," a state-estimated digital duplicate of a physical object. This Proxy enables a reduction in sampling and transmission rate at the sensing or actuating device by deploying an estimator based on the system model. This pseudo-interpolation simultaneously addresses the common problems with timestamp corruption and unplanned data outages frequently occurring in connected systems.

We further apply the Proxy to improving security by requiring applications to communicate with the estimator in the Cloud rather than directly with the devices themselves. Eliminating the possibility of direct connection between devices and applications eliminates a vulnerable point in common, connected communications.

The application of a gateway architecture, wherein applications talk to the estimate of the object rather than the physical object, additionally facilitates the use of a "Cognitive Layer" for device protection, which checks system development and validates commands against the state model to ensure their effect is benign prior to execution. A Cognitive Supervisor monitors state evolution in the absence of external commands and raises alerts should a state behave uncharacteristically, which might be indicative of a fault.

The net result of this structure is a secure and efficient implementation of connectivity that addresses application-specific needs and which can be optimized on-the-fly for changing application payloads.

Following the introduction of this architecture, I create Data Proxies and apply them to address two common vehicular problems: measurement of fuel consumed for use with efficiency-improving applications, and estimation of vehicle miles traveled, to be used as an equitable replacement for the fuel tax in hybrid and electric vehicles or usage-based insurance. These two examples illustrate how Proxies may be used to improve data quality for a fixed cost of acquisition, and how Proxies may minimize cost for a fixed accuracy target. In demonstrating these, I find that Proxies enable a substantial reduction in resource use with minimal increase in system estimation error. Finally, I present a generalized process for system model selection development and sampling rate optimization based on forward-simulation of real data using the system model.

2.2 Introduction: Managing Resources to Grow the Internet of Things

The world's devices, services, and people have become increasingly interconnected, with a growing number of objects joining the "Internet of Things" daily. These Connected Devices have the potential to enable improved comfort and convenience and efficiency and engagement, but the deployment of such devices is limited due to techni-

cal and social blockades. Technical challenges to the expansion of local area networks and broader Internet connectivity include system complexity and costly support requirements like power and bandwidth [161]. These problems scale when systems are repurposed for pervasive sensing (where sensors and actuators are used other than as designed), as conventional scheduling systems frequently cannot moderate additional requests for data and control without growing resource requirements. Social challenges include concerns about data privacy and an increasingly unaddressed push for designed-in security, which has led to a reluctance for consumers to connect sensitive systems to widely accessible networks.

Addressing these concerns, we explore building an efficient and secure new architecture for connectivity revolving around the concept of intelligent "Data Proxies." These Proxies are based on statistical and physical system models, allowing connected systems to efficiently meet prescribed "Quality of Data" (QoD) requirements for connected applications.

Data Proxies apply observers and estimators to generate digital duplicates of physical systems with intelligently limited sampling, saving power and bandwidth at constrained sensing nodes (devices). Using models to create digital mirrors abstracts digital and physical systems, improving connected device security by eliminating the possibility for direct-to-device connection. Additionally, we propose the concept of a "Security Layer" and "Cognitive Layer" making use of Cloud resources to moderate connections and to protect the Proxy's system. Finally, we propose the creation of an "Application Agent" supporting simulation-based automated resource management to minimize cost while meeting varied Quality of Data demands.

2.3 Background: Conventional Connected Architectures

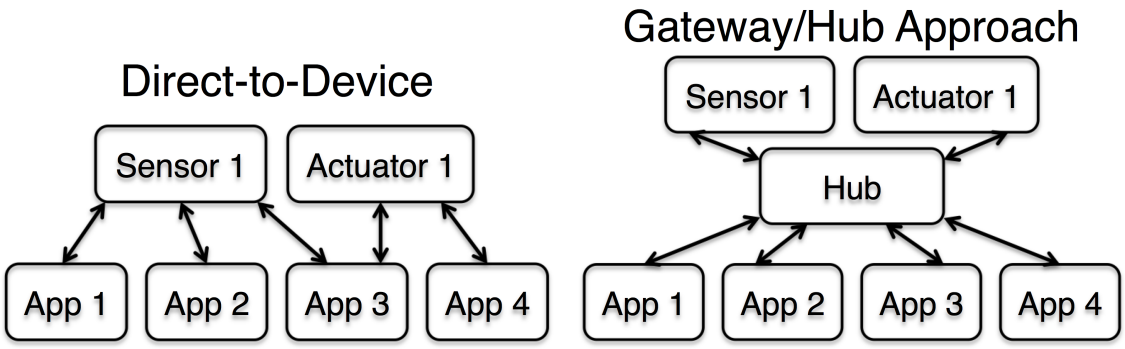
To understand how the use of an intelligent architecture can improve a connected device platform, one must understand the architectures commonly used today. We

consider classical, smart hub, and Cloud mirror approaches to connectivity.

2.3.1 Classical Connectivity

In this direct-to-device approach, a client or application can query or control a system's sensors and actuators via direct connection. An example system pairs a mobile phone with Bluetooth enabled temperature sensors and color changing lightbulbs. The application will typically enumerate all connected devices on the network and request and send information as necessary, e.g. to request temperature data or to toggle a lamp.

This topology is efficient for a single application used in conjunction with few end use devices. Temperature information is sent only when it is needed and the lamps process all incoming commands. However, this architecture scales poorly. Each additional application adds data queries or sends new command requests. If an application samples a device at n Hz, and m copies of that application are running, the devices are queried $m \times n$ times per second, consuming additional bandwidth and power, despite these samples conveying similar or identical information. This is shown in Figure 2-1a. In the worst-case, the network becomes saturated and packets drop.



(a) (b)

Figure 2-1: This figure shows two common topologies: a direct-to-device system, wherein applications communicate linearly with sensors and actuators, and a Smart Hub architecture, which passes all data through a central node, where power consumption and processing may be made more efficient.

This architecture additionally suffers from compromised security implementations

because low-cost and low-power constrained nodes (motes) deployed at the edges of a network lack the computation required to encrypt data and verify credentials in a timely manner. As proof of this, this computation lag is the reason many existing home automation devices have been unable to implement Apple's "HomeKit" standard [162]. If developers forego such security measures and should a malicious actor join the network, this actor is not easily disrupted and may continue indefinitely to request improper actuation or prevent dissemination of valid data.

Though quick to develop and test, this approach is unsuitable for scalable deployment [163] or use in safety-critical systems.

2.3.2 Smart Hub Connectivity

A Smart Hub (gateway) approach differs from direct, Classical Connectivity, in that requests for data or control pass through a master node capable of moderating packet flow. This architecture is often used to bridge two communication technologies, as might be the case with a cellular phone interfacing with a ZigBee home lighting system. In such a system, the gateway node connects to the Internet via Ethernet or WiFi and bridges the cellular connection to the ZigBee bulbs in the system.

Gateway systems may operate simply, redirecting each message immediately to the appropriate target, or can have integral sampling intelligence, performing local data aggregation and preprocessing [164] to minimize redundant data acquisition, transmission, and actuation requests. In the former case, application data requests scale as in the direct-to-device case. In the latter, the gateway pools requests, e.g. from one application requesting at n Hz and one requesting at m Hz, with the gateway choosing to poll at the ceiling of these two request rates. This network topology is shown in Figure 2-1b. Note that this architecture is similar to the previous approach with the exception of the added central hub.

In this sort of gateway system, the master node often offers additional computation relative to the end nodes, which are largely unchanged from the direct architecture. The availability of enhanced computation allows for the use of firewalls, encrypted communications, and efficient implementations of credentialing systems. This compu-

tation, along with the severance of direct links between device and application, allow simplified blocking of malicious agents. This same computation may be applied to simple, centralized data aggregation, filtering, and processing.

Though hub-based systems improve scalability and security versus direct-to-device systems, these systems offer room for improvement. Because gateways remain resource constrained, their best application is within networks of small to medium scale, with semi-static sensing and actuation payloads used in conjunction with pre-approved applications. Commercial offerings, like the Dell Edge Gateway, serve as an example of a high-powered configurable hub.

2.3.3 Cloud (or Fog) Mirror

Cloud and Fog mirrors are similar to hub-based platforms, but data aggregation and filtering takes place at a remote server. A Cloud system mirrors one or several individual devices (sensors and actuators) or hub networks and stores this information centrally for later use. Similarly, a Fog system does the same data storage and processing at the logical extreme of a network, such as a local laptop or desktop computer connected to a centralized server. These mirrors often combine data sources from different connected device platforms and apply additional processing to filter data and aggregate results [98]. Like a Smart Hub, data and control requests are abstracted from the physical device, but in this case, computation and power are infinitely scalable. This approach is used by commercial platforms such as PTC's ThingWorx or IBM's Watson IoT.

Making the determination of whether to use a Hub or Cloud model can be challenging, with this architectural decision typically depending on data transmission rate, communication protocol, and bandwidth pricing. The second half of Chapter 2 addresses these issues in the context of Connected Vehicles. In the case where bandwidth costs and latencies are small, the Cloud typically offers the best possible scalability [165] and future-proofing. In all cases, it allows improved computation. For this reason, it is used as the basis for the "Data Proxy" architecture, which most benefit dynamic and growing systems.

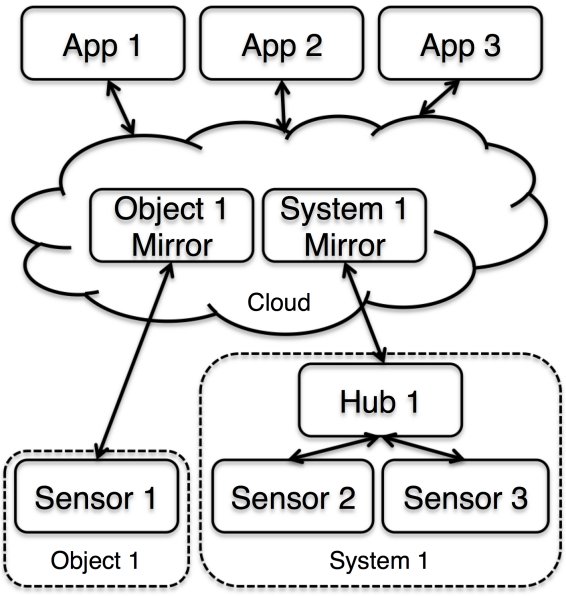


Figure 2-2: An application interacts with the Cloud Mirror of a physical object or an integrated system. Note the isolation between the applications and the physical system. The Cloud computational backend can scale horizontally and vertically depending on the computing needs of the system.

2.3.4 Prior Art: Reducing Resource Requirements

Minimizing resource use in connected systems is a critical problem, and one that has been widely studied.

One approach to minimizing the resource requirements for a wireless sensor network (WSN) is to focus on routing and the network itself. Energy-saving SensorNet distributions making use of scalable, self-organizing and efficient data dissemination algorithms are possible. Such networks have focused on data-centric storage eliminating originating node names in favor of storing all nodes of the same class at a common location. This minimizes energy overhead due to reduced searching and more efficient routing [166]. This architecture may be likened to the use of a gateway or aggregating hub, but mirrors multiple sensor inputs in a particular physical location rather than storing entire objects as one-to-one Cloud mirrors.

In addition to network-centric approaches, there are device-centric and system-wide approaches to minimizing resource use via sensor sampling reduction.

Dynamic scheduling has been studied through use of Multiple Constraint Opti-

mization Problems (MCOPs) to identify the best allocation of computing and communication resource to maximize control and analysis efficacy in a Real-Time Wireless Sensor Network (RTWSN). The goal of this optimization is to find the sampling rate that maximizes global utility while maintaining applicability of real-time scheduling systems. Liu et. al discuss sampling rate optimization subject to constraints in depth and note that an offline solution has been formulated using the Kuhn-Tucker optimality conditions. Other approaches have considered Quality of Service resource allocation models with multiple quality dimensions, but only a single constraint. One proposed solution is to formalize sampling for RTWSN into a nonlinear optimization problem in both a central way (one processor) and a distributed approach (to avoid CPU bottleneck). The proposed dynamic sampling system can handle multihop routing and converges over time, but does not include any Quality of Service restrictions [167]. A similar framework was proposed using dynamic route selection as part of a constrained optimization problem, and solved under the Network Utility Maximization framework [168].

Other approaches focus on minimizing the sampling rate on networks with a finite bandwidth or another similar constraint. On such networks, applications suffer from packet delays and observation loss. Adlakha, et. al designed a sensing system making use of a Kalman filter to account for missing observations from multiple sensors, and identified the rate at which an event of interest must be sampled to maintain an error covariance of the estimate below a target threshold [169].

Hu et al. used linear programming to predict intermediate sensor data to reduce the number of sensors and the sampling rate required to instrument a system, minimizing energy requirements. This contribution is valuable, though works best in ground-up system deployments and with single-purpose sensor networks [170].

Jain et al. propose a solution for maximizing data accuracy or minimizing resource use, suggesting treating the sensor data stream management as a filtering problem. They suggest pre-filtering out as much data as possible while ensuring that error targets are met, making use of a Kalman filter running on the device itself. This minimizes communication overhead, although the embedded device or remote node

requires additional computational power to run this filter [171].

Some techniques like load-shedding and adaptive precision setting collect data starting from the maximal sampling rate and then make the determination on whether or not to drop data to conserve resources. Jain and Chang proposed an adaptive sampling technique where sensor sampling rate adapts to streaming characteristics using a sensor-run Kalman filter. When the error is outside of targeted bounds, the sensors change the sampling rate. The collection server, based on resource availability, sets sensors to a new target sampling interval range [172].

Compressed Sensing theory is an approach to reducing sampling points necessary for an application. This theory relies on data sparseness, which is a system property that makes it possible to allow a finite number of samples to capture all required system information by exploiting a priori sparsity information for signal reconstruction. Compressed Sensing compresses data at the IoT end node, transmits data, then reconstructs at the reconstructed at a "Fusion Centre." This may require computationally-intensive calculations at the node, raising initial hardware cost and the amount of power required [173]

Another approach to minimizing resource spend is to identify the most critical sensors to minimize the worst-case errors [174]. By examining the sensitivity of an application to data inputs and deploying only critical sensors, the energy or bandwidth requirement for a network may be reduced.

Sensor resource use optimization is clearly a well discussed area. What the referenced papers do not discuss at length is how network architecture can impact system security by allowing improved credentialing and command validation, and how optimization may be conducted a-priori with forward simulation of data. The following sections address this and limitations present in low-power sensing motes, exploring the concept of Data Proxies.

2.4 Defining a More Secure and Efficient IoT

This section introduces new terminology describing the elements of our proposed architecture improving resource management and security in connected systems. These elements combine to form "The New IoT."

2.4.1 Data Proxies

We introduce the concept of a "Data Proxy," a Cloud-run digital duplicate closely mirroring a physical system. These replicas provide an estimate of the system state and its evolution based on measured inputs and outputs of the monitored system.

Data Proxies improve upon the Cloud mirroring model of connectivity with the use of observer- and estimator-informed digital object duplicates. These tools take an imperfect, periodically-sampled view of a device's state and fill in the gaps based on trusted physical or statistical models capable of applying context about anticipated system behavior. Applications communicate with the Data Proxy and it's related elements rather than the physical device or system it mirrors – the use of an intermediate Proxy allows the isolation of devices from requestors, while data in-fill (interpolation, estimation, or another means of gap filling) reduces the requisite sampling rate and resource requirements to maintain high-quality data. While a Data Proxy could be run in a gateway device rather than in the Cloud, the additional computation afforded by the Cloud's scalable computation is better suited to enhancing security, running system models of increased complexity, pre-aggregation of data, and more.

2.4.1.1 Selective Visibility

The main innovation of the Data Proxy is the implementation of selected visibility for state, input, and output measurements. We define "Selective Visibility" as an approach intelligently modulating the inputs that are available to a system model at a given time-step, as well as the visibility of outputs.

Though states are constantly evolving, in connected systems these inputs, outputs, and states may not be sampled at every discrete time step. This missed sampling may

be by design, or due to availability issues resulting from technology implementation and constraints. In the case of the Data Proxy, we selectively modulate these variable's visibility through the use of a scheduling system, writing each input and state measurement as a time-varying matrix. Each of these selective-sampling matrices has an associated cost of acquisition in each time step, and may have associated fixed costs of implementation as well.

2.4.1.2 Quality of Data

We propose the creation of a "Quality of Data" (QoD) metric to provide clear definitions of an application's data needs. QoD may be quantified in several ways, and focuses on data timeliness and accuracy. The following are proposed:

1. **Instantaneous Accuracy**: The maximum allowable error between the model and real system at a particular time, as might be the case in a factory where the manager needs to know power used by a particular machine at the moment another machine might come online
2. **Periodic Accuracy**: The maximum allowable error between the model and real system at regularly spaced intervals, which would be useful in the example application of generating 'snapshot' factory reports to determine energy trends throughout the day
3. **Average Accuracy**: The maximum allowable sum of errors between the model and real system between two target times, which is useful in supervising equipment over the long term
4. **Maximum Latency (freshness)**: The maximum acceptable temporal recency of direct sensor measurement, which aids applications such as maintenance, where machine temperature might need to be known prior to starting a repair
5. **Minimum Latency (freshness)**: The minimum acceptable temporal recency of direct sensor measurement, e.g. for sensors that have long acquisition times or low duty cycles, or application which require obfuscation as is the case in public-facing websites where 'fresh' data presents a security risk

Additional requirements may emerge, such as threshold sensitivity (sending a packet within a specified window after a target value has been reached) or a maximum specificity (in the case where data must be abstracted for security purposes, such as a friend-finder or vehicle tracker). These requirements may extend the QoD proposed in Wu [175], wherein data has measurements contain information about accuracy,

truthfulness, completeness, and up-to-dateness. Alternatively, a simplified normalized score such as weighted aggregate accuracy may suffice.

A QoD request will be included with all requests for data, and replies will include the actual QoD of the returned data, with parameters such as time of last true acquisition and predicted error. This will help applications to take into consideration the impact of reduced-accuracy data.

2.4.1.3 Application Agent

We further define the creation of an "Application Agent" as a query manager running in the Cloud. The Application Agent uses the meta-QoD aggregated from a Proxy's entire application payload and dynamically determines the Selective Visibility sampling configuration meeting the requirements at the lowest cost of acquisition. Processing QoDs in aggregate allows the Application Agent to eliminate requests that do not provide additional information, based on forward-simulation of the Proxy's system model.

2.4.1.4 Security and Cognitive Layers

A Security Layer abstracts direct device control and data accessibility away from the Internet connection by selectively allowing or blocking an application's access to information and actuators. This network separation allows the use of a Cloud-run "Data Bouncer" validating credentials and allowing approved agents access to select data and control parameters. Credential inspection and verification is managed in the Cloud rather than on remote devices, as would be the case when implementing solutions such as MQTT's Authorized Client Lists. This allows rapid authentication, which enhances user experience versus operating in a constrained node (which would result in actuation and data delays, or add substantial cost to each additional device).

This layer is responsible to allowing or denying access to applications based on trustworthiness and/or human-in-the-loop approval. Cloud computation may be applied to support a system managing limited time or conditional permissions, or to handle credential revocation should the need arise. Unlike systems today which

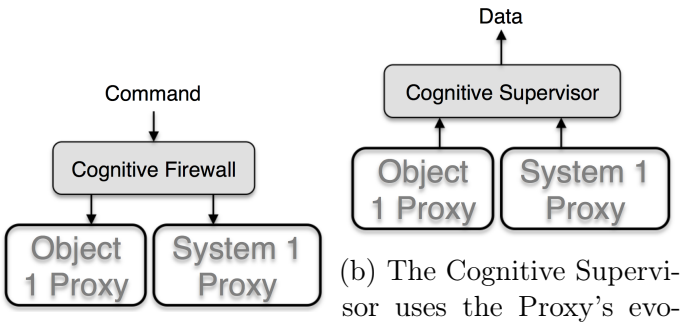
risk slowdowns and poor user experience when implementing computation-heavy security, the use of scalable Cloud resources provides the Security Layer with enhanced computation to support encrypted connectivity, certificate validation and revocation, and the creation of permission groups.

This approach could be used to minimize the risk of damage to physical plants through the Internet and to assure privacy of users' sensitive information by implementing a "Private Cloud" inside the public Cloud. This approach mirrors encapsulation in object-oriented programming by allowing selective access to potentially sensitive information or control.

A Cognitive Layer further enhances the protection afforded by the Security Layer by applying the Proxy's system model to evaluate system's process evolution and to test incoming commands, similar to a form of automated process control. This layer moderates requests that might exceed control limits and raises alarms when system evolution does not follow the anticipated model closely. The Cognitive Layer validates that commands are benign and examines system evolution to identify when the Proxy's model breaks down, suggesting a fault in either the digital or physical system.

The Cognitive Layer has two components. The first component, called the Cognitive Firewall, operates in the case that data are pushed from an authorized application into the device's Proxy (integrating extrinsic information for control). The Cognitive Firewall uses statistical or physical models to project commands forward in time, and validates whether or not a particular command "makes sense" for a connected system. By testing each command through a series of forward simulations, the firewall uses intelligence to protect physical systems. The second component is a Cognitive Supervisor that runs when data are sent from a device to an application (when sharing a system's intrinsic information with external devices), using the information already transmitted to identify potential faults early. While the Cognitive Firewall offers defensive security improvements through command validation, the Cognitive Supervisor proactively monitors system evolution and identifies anomalous behavior due to system model breakdown, physical system failure, or another self-inflicted mechanism.

Both the Firewall and Supervisor components rely on the same system model used by the Data Proxy. These approaches are shown in Figure 2-3.



(a) The Cognitive Firewall uses a system model to validate that commands are not malicious (by intent or inattention).

(b) The Cognitive Supervisor uses the Proxy's evolution data to determine whether a system is trending toward or already operating outside of learned or defined limits.

Figure 2-3: The Cognitive Layer incorporates a Cognitive Firewall, a Cognitive Supervisor, or both, depending on the availability of system data and the presence of input control signals.

The Security and Cognitive Layers embody Isaac Asimov's third law - for the system to protect itself.

2.5 System Overview

From the above taxonomy, our proposed architecture is shown in Figure 2-4.

In an example scenario with m applications requesting samples at n Hz, this intelligence would limit sampling to n Hz. Through the use of less-prescriptive QoD requirements, the Data Proxy could determine that sampling at n Hz is unnecessarily fast, and that a reduced sampling rate could be used while still meeting the requirements of each application. This sort of intelligent downsampling eliminates unnecessary interactions, saving bandwidth and power while isolating devices from applications to enhance security. Humans implement a similar architecture every day to reduce resource spend and improve data quality and security in response to requests for information. The design of the Proxy model builds upon this human methodology.

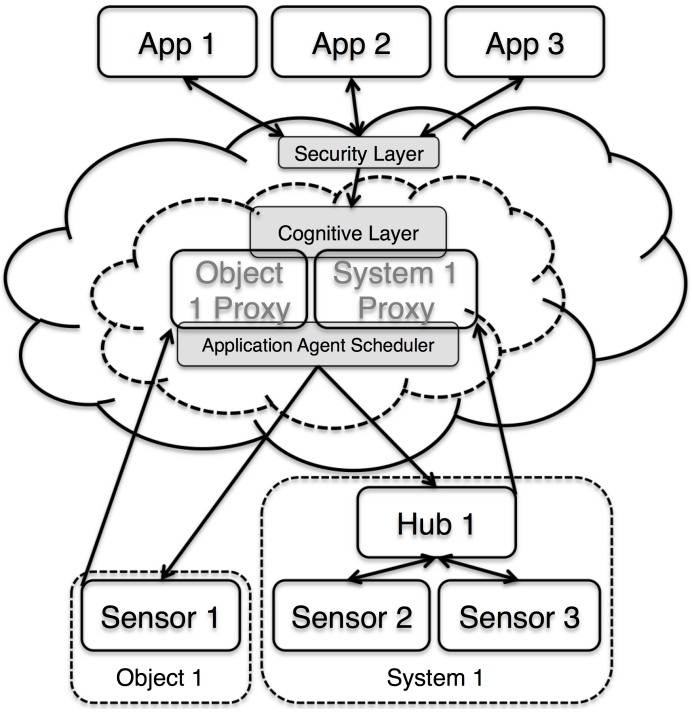


Figure 2-4: Data Proxies extend the Cloud architecture, with estimators creating interpolated approximates rather than exact mirrors of input data. An Application Agent manages the data query request rate, a Security Layer validates credentials, and the Cognitive Layer tests commands against system limits to prevent injurious actuation as well as identifies anomalous behavior.

2.5.1 Human Examples

The Proxy architecture aims to apply context and cognition to improving security and efficiency of connected intelligent systems. When thinking about how humans share information, there are five defining characteristics:

1. People **apply context** to distributing information
2. People **synthesize information** from multiple sources
3. People **minimize effort through estimation** where appropriate
4. People **protect ourselves and our resources** through moderated abstraction

The proposed architecture incorporates each of these five criteria. To understand how a Data Proxy moderates requests and how this relates to the parallel human

system embodiment, consider the case of a watch wearer (Proxy Person) and a stranger (Application Person) in a train station awaiting their respective departures. The watch is a constrained sensing node, the person is the Proxy, and the stranger is the Application/Client.

The stranger does not want to be late for his train and makes a request asking the watch-wearer for the time. Initially, the wearer checks her watch and reports back the time, with the requestor able to see that she is checking the watch directly, ensuring the validity of the reported time. The requestor does not need to leave yet and waits, occasionally asking the watch-wearer for the time. The wearer becomes annoyed and stops checking her watch directly, replying instead with estimates based on the last absolute measurement and the position of the sun. These responses, like "it's about 10:30," include data for the requestor, as well as information about the certainty of the response, while saving the watch-wearer the effort of checking the actual time.

Proxies allow for temporal and spatial interpolation of data, improving the apparent time constant of the sampling system for non-time-critical or non-accuracy-critical applications. This architecture has the ability to save the Proxy Person bandwidth in acquiring and transmitting replies. If we consider the case of a watch (sensor) that requires the illumination of a backlight before the Proxy Person may acquire data, the Proxy's optimization further contributes to longer operational life and lower operating cost for these sampling resource heavy sensors in the field.

The Proxy Person optimizes queries from multiple client applications in aggregate. If a high-priority application (a close friend) requests data, the watch-wearer may make the additional effort to get the exact time and both the friend and the nearby stranger Application get the benefit of accurate data. If the low-priority stranger continues to annoy the wearer, she can stop replying and eliminate the associate resource drain altogether. In all cases, the requestors (applications) never gain direct access to the watch but do gain access to the current time, with varying and communicated accuracy.

In other cases, the Proxy may be required to fuse data for an Application. If the Application Person requires information about the time until a train arrives, the Proxy

must check both her watch and the train schedule prior to replying – or determine that the requisite effort is too high.

The Proxy Person even applies safeguards to protect her resources and to validate the impact of incoming commands. If the Application Person gets aggressive and asks the Proxy Person for her watch, Proxy Person is able to apply a level of abstraction and deny the Application Person direct access to her watch. She may device whether or not to tell the Application the time, or she could even pretend that she does not have a watch. In the case of system validation, Proxy Person may look at her watch 30 minutes apart and notice that the dial has not moved, applying a level of cognitive supervision to determine that her system model and data inputs are incongruous, perhaps suggesting a dead watch battery. The Proxy Person may even validate commands, in the case of the Application Person asking her to watch his bag for an hour. Proxy Person forward simulates the impact of this command (that she would have to remain where she is for an hour) and determines that it is in conflict with a higher-priority directive (catching her train) and apply a cognitive firewall to reject the command.

Security is improved as the person's Application Agent manages update rates, connection duration, and more, with the Security Layer shielding devices from Denial of Service (DoS) attacks and moving security to areas with improved computation for client whitelisting, certificate revocation, and more. This structure allows the server or gateway to authenticate requests, monitor all data interactions, and to translate interfaces such that communication and implementation protocols may be abstracted, allowing hardware and platform agnostic application development.

This cognitive and contextual approach is not without problems. Actuation latency and data accuracy may suffer due to the reduced sampling rate. Developing models to estimate state with appropriate corrective feedback can be a challenge, and characterizing and controlling for noise is difficult. Security challenges grow as targets become increasingly valuable with more information, and data require novel representation to properly indicate real versus estimated data and their error bars.

2.6 Mathematical Concepts

Observers and estimators develop replicas of real systems that change and evolve over time. These replicas may be used to inquire as to the state of a system without direct and costly measurement, and are commonly applied to problems where the internal physical state of a system cannot be or is not easily directly observed. Observers make use of physical models, whereas estimators use statistical models. Both require accurate characterization of the system and noise parameters.

These mathematical system models are well understood. The novelty of our approach arises from the use of these tools as a means of reducing resource use and enhancing the deployability of resource-constrained devices.

Discrete Time State Observer

An observer is a system of equations used to estimate the internal state of a real, deterministic system based on measured system inputs and outputs. The observer uses a physical model as a priori knowledge to monitor hidden variables and uses feedback to provide robustness and minimize error. Since most sensing devices measure values in discrete time, we will consider the canonical discrete-time Luenberger observer [176]. In this observer, feedback based on the difference between the estimated and true state limits future error growth. This architecture is shown in Figure 2-5.

Discrete Time Estimator

Estimators differ from observers as these models can be reconstructed without a priori physical models, relying instead on stochastic models. This makes estimators attractive in cases with pre-existing data, and where the system in question is difficult to model due to nonlinearities, many inputs, or significant process and measurement variations. In the remainder of this document, the Kalman filter will be used as an example [177]. The Kalman filter applies knowledge of measurement and process error, combining means and variances to gain improved insight into the true process evolution. The Kalman filter's recursive, iterative structure is shown in Figure 2-6.

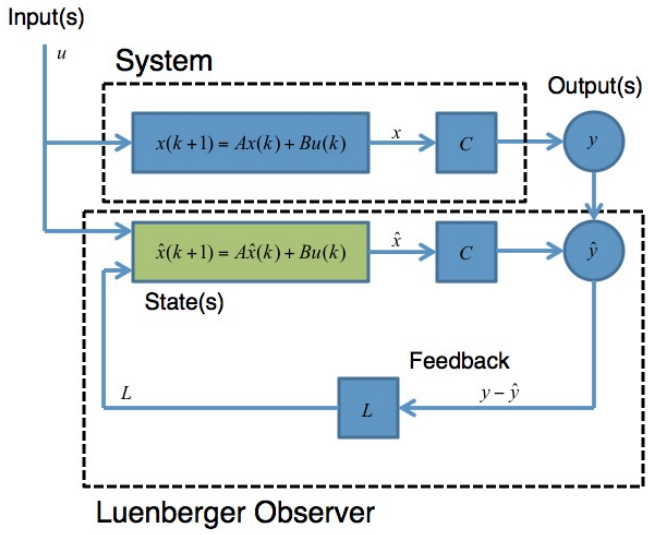
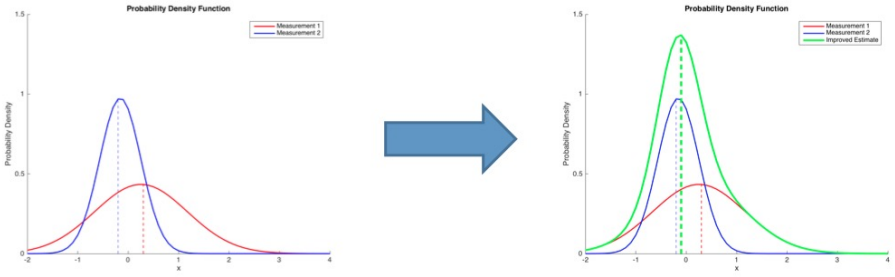


Figure 2-5: The Luenberger Observer relies on duplicate system models fed by identical input data, and minimizes the error between the true and estimated systems by applying negative feedback proportional to error.

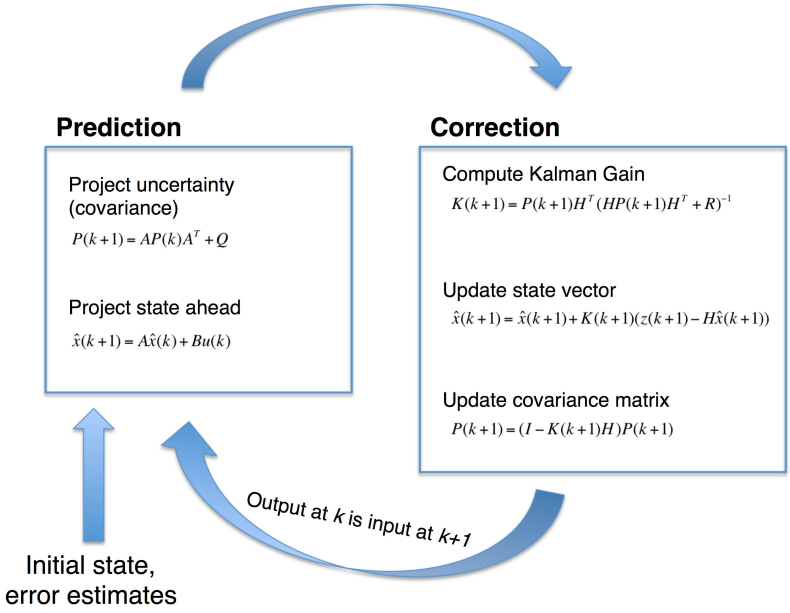
2.6.1 Resource Requirements

To improve IoT solution deployability, we aim to minimize resource cost subject to constraints. Dependent upon power and connectivity type (battery versus mains, wired versus wireless) these costs might be significant. In general, there are fixed costs and variable costs. Fixed costs are the baseline load; variable costs change based on parameters such as sensor sampling rate, data upload rate, and sleep mode frequency. Simple cost models examine only sensor access, while nuanced models include fixed and variable costs, costs of acquisition, transmission, and computation, and extension to other resource types, such as cost of bandwidth, network loading, computation, and more.

For all systems, sampling matrices (as defined in Section 2.4.1.1) have associated *cost functions*. These functions examine which sensor values, outputs, or state measurements are accessed in a given time step and calculate the cost of acquiring these data. At their simplest, the cost at time t is a scalar function of sampling matrices. For example, if a position sensor is used, the resource use at t includes the contribution of sampling the position sensor. If the velocity sensor is used, the velocity contribution



(a) An estimator combines multiple measurements with differing means and variances to provide an improved measurement.



(b) The Kalman filter is a recursive, iterative estimator.

Figure 2-6: Thes figures show the general concept and a sample implementation of a Kalman filter, an iterative estimator combining multiple measurements with differing means and variances to provide accurate, convergent state measurement.

is also added. The resources used by time t are the sum of the resources used at every previous step.

2.7 Particle Following Proxy Example

We apply the concept of the Data Proxy to an example system, demonstrating its utility as a solution capable of minimizing resource use while meeting a QoD target. We first build physical and statistical models, then reduce sampling and examine the relationship between mean squared error (MSE) and resource consumption by a

particular time.

2.7.1 Physical Model

The Data Proxy is easily demonstrated with a noisy system consisting of the one-dimensional forced motion of a particle. This is modeled as a two-state, discrete time-variant system with Newton's second law relating applied force to velocity and motion.

The system equations begin as the equations of motion:

$$a_x = \frac{F_x}{m} \tag{2.1}$$

$$v_x(t+1) = v(t) + a_x(t)\Delta t \tag{2.2}$$

$$x_x(t+1) = x_x(t) + v_x(t)\Delta t + \frac{1}{2}a_x(t)\Delta t^2 \tag{2.3}$$

To demonstrate the advantages of estimators in nonlinear systems, we choose a forcing function $F = 10 * (\sin(k/100) + cos(k/100))$ and a mass of $m = 1\,\mathrm{kg}$.

2.7.2 System Objectives

Our objective is to minimize cumulative power consumed $\Theta(t_m)$ while keeping error for position and velocity less than the mean square error (MSE) limits represented in Equation 2.4.

$$\min \Theta \text{ subject to } MSE_{pos} < \alpha^2 \text{ and } MSE_{vel} < \beta^2 \text{ where } MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{X}_i - X_i)^2 \tag{2.4}$$

We set the cost of acquiring state or input values at each time step to be $\theta_{pos} = 5000\,\mu\mathrm{J}$, $\theta_{vel} = 200\,\mu\mathrm{J}$, and $\theta_{acc} = 20\,\mu\mathrm{J}$, with mean error limits at $t_m = 500\,\mathrm{s}$ of $\alpha = 10\,\mathrm{m}$ and $\beta = 5\,\mathrm{m/s}$.

2.7.3 Sampling Constraints

We enforce a clock with a time interval $\Delta t = 1s$ and constrain our system to uniform, invariant sampling rates f_i for each sensor. Additionally, we apply maximum and minimum acceptable sampling periods of $20\,\text{s} < T_{pos} < 50\,\text{s}$, $5\,\text{s} < T_{vel} < 25\,\text{s}$ and $1\,\text{s} < T_{acc} < 5\,\text{s}$. Maximum periods apply for sensors that must be kept warm or otherwise "kept alive." Here, the period T_i is the inverse of the frequency f_i.

2.7.4 Observer Implementation

We implement a Luenberger observer as shown in Equations 2.5-2.8.

$$\text{State } x(t+1) = Ax(t) + Bu(t) \tag{2.5}$$

$$\text{Output } y(t) = Cx(t) + Du(t) \tag{2.6}$$

$$\text{State Estimate } \hat{x}(t+1) = A\hat{x}(t) + L[y(t) - \hat{y}(t)] + Bu(t) \tag{2.7}$$

$$\text{Output Estimate } \hat{y}(t) = C\hat{x}(t) + Du(t) \tag{2.8}$$

x's represent the state, y's represent the output, and u's are inputs. The equations for x and y model the physical system's state and outputs, and $\hat{x}$ and $\hat{y}$ are the respective components of the state observer. The matrices A, B, C and D are the state, input, output, and feedthrough matrices.

The error for this system is calculated as the difference between the estimate and the actual system, $e(t) = \hat{x}(t) - x(t)$. This error satisfies the equation $e(t+1) = (A - LC)e(t)$, where L is a parameter calculated to obtain the desired convergence speed and stability.

Implementation of Selective Visibility

To implement Selective Visibility, matrices C and D contain the Kronecker comb function. The comb has a periodic value of 1 (polling) while at all other times the value of the function is zero (no measurement available), as shown in Figure 2-7. The periodicity of each comb is T_i where i is the index of the model input. Kronecker delta combs are written as:

$$\Delta_{T_i}[t] = \sum_{k=-\infty}^{\infty} f(t)\delta(n - tT_i) \tag{2.9}$$

The C and D measurement matrices have associated cost functions $f(C(k)$ and $g(D(k))$. These determine whether a particular sensor or measurement is polled and add the appropriate resource contribution $\theta_{pos}, \theta_{vel}$ or θ_{acc} at time step t.

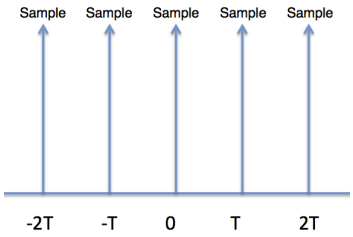


Figure 2-7: The Kronecker Delta Comb is used as an input to our selective-sampling matrices. When the comb has a value of '1' the input, output, or state are polled, and these are not sampled at any other time.

Our physical state model $x(t+1) = Ax(t) + Bu(t)$ with added gaussian process and measurement noise v and w thus becomes:

$$\begin{bmatrix} x(t+1) \\ v(t+1) \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x(t) \\ v(t) \end{bmatrix} + \begin{bmatrix} \frac{\Delta t^2}{2} \\ \Delta t \end{bmatrix} \left[\frac{F(t)}{m} \right] + v \tag{2.10}$$

The selectively-visible output model $y(t) = C(t)x(t) + D(t)u(t)$ becomes:

$$\begin{bmatrix} y(t+1) \\ z(t+1) \end{bmatrix} = \begin{bmatrix} \Delta_{T_{pos}} & 0 \\ 0 & \Delta_{T_{vel}} \end{bmatrix} \begin{bmatrix} x(t) \\ v(t) \end{bmatrix} + \begin{bmatrix} \Delta_{T_{acc}} \\ 0 \end{bmatrix} \left[\frac{F(t)}{m} \right] + w \tag{2.11}$$

2.7.5 Estimator Implementation

We additionally apply a statistical estimator to model system evolution. We choose the Kalman filter presented in Equations 2.12-2.19.

$$\text{Predict Next State } \hat{x}(t+1) = F\hat{x}(t) + Gu(t) \tag{2.12}$$

$$\text{Project Covariance } P = F * P * F' + Q \tag{2.13}$$

$$\text{Make Measurement } Z = H * X + H * X * v \tag{2.14}$$

$$\text{Project Kalman Gain } K = P * H' * (H * P * H' + R)^{-1} \tag{2.15}$$

$$\text{Update Estimate } \hat{X}(t) = \hat{X}(t) + K * (Z - (H * \hat{X}(t))) \tag{2.16}$$

$$\text{Update Covariance } P = (I - K * H) * P \tag{2.17}$$

$$Q = cov(v) \tag{2.18}$$

$$R = cov(w) \tag{2.19}$$

Implementation of Selective Visibility

In the estimator, Selective Visibility is implemented using matrices H and I analogously to the observer's C and D from Section 2.7.4.

2.7.6 Comparing Observation and Estimation

These methods were simulated for varied sampling schemes and plotted for MSE versus power cost. For each, the lowest-cost feasible method (within acceptable error bounds) was selected for comparison against the true system.

Properly tuned, the observer and the estimator provide more accurate results than the naïve downsampled case, with the estimator improving upon the observer due to the system's noise. The true state and best-available estimates and errors are shown in Figure 2-8.

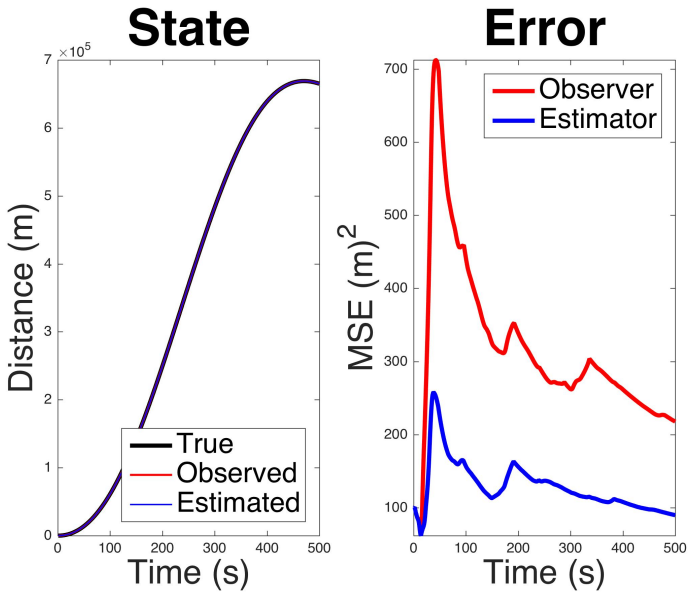
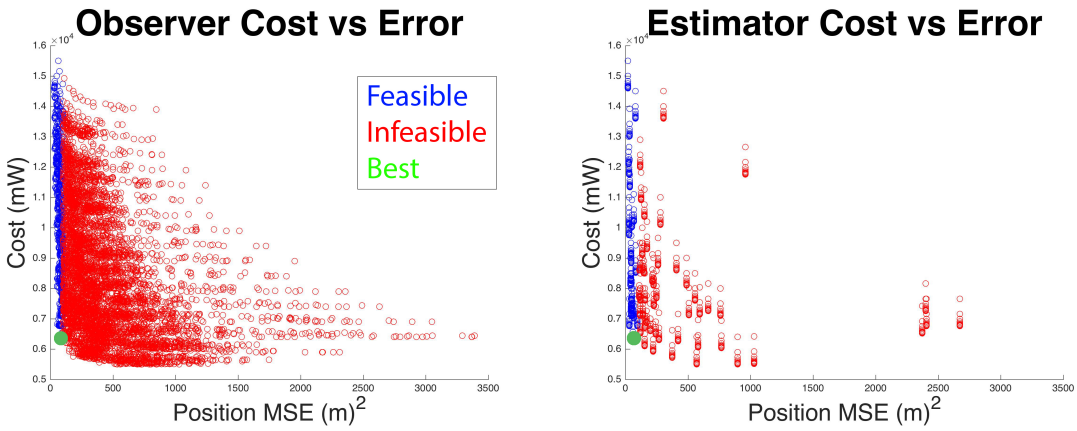


Figure 2-8: The observer and estimator appear to mirror the system, but the MSE illustrates the difference between the models and reality. Here, the estimator copes better with noise and converges faster than the observer in the optimally-downsampled case.

Both the estimator and observer provide cost-effective estimates at $t = t_m$, as shown in Figure 2-9.

A comparison of power cost and MSE for 500 s of measurement appears in Table 2.1. We see that the observer and estimator both improve the MSE and cost for position tracking. The estimator demonstrates improved performance and efficiency due to the noisy system and measurements.

Due to relaxed error constraints, the use of observers and estimators allows us to



(a) Power cost vs position MSE, observer. (b) Power cost vs position MSE, estimator.

Figure 2-9: We see that it is possible to dramatically decrease power consumption without significantly increasing error. These plots show a Cost/MSE relationship that goes as $\frac{1}{x}$.

	Raw	**Observed**	**Estimated**
MSE (m)^2	193.65	93.57	73.99
Power (µJ)	261.0	86.7	71.0

Table 2.1: Comparison of MSE and power consumption for the raw, observed, and estimated position of a particle after 500s.

reduce power consumption dramatically shows that applying intelligent downsampling to create Data Proxies has promise.

The following section applies this concept to the real-world problems of monitoring vehicle fuel use and miles traveled. The fuel measurement problem demonstrates how observers and scheduling systems can reduce error and cost of data acquisition for real systems, while the application to distance monitoring explores how to intelligently structure an estimator to minimize costs while meeting a target QoD.

Monitoring fuel consumed in a vehicle can save money, and there are multiple means of measuring fuel consumed, with varying accuracies and resource costs. Similarly, there are applications for these data with a range of latency and accuracy requirements, like realtime feedback, behavior modification programs, and remote gas gauge applications. Recording miles traveled is similarly interesting for mileage based taxation or insurance. In both cases, it is necessary to minimize resource use to make these applications tractable.

2.8 Application of an Observer to Fuel Measurement with Outages

We now explore the use of standardized vehicular On-Board Diagnostic data and proprietary signals in accurately measuring fuel consumption. We consider application of an observer (Equations 2.5-2.8) to a system with random, unscheduled data outages and create a Proxy capable of reducing estimation error for fixed, uncontrolled sensor samples. In essence, this simulates the data availability on an existing network such as is the case when reading values from a production vehicle's Controller Area Network. In this type of system, data related to fuel consumption are shuttled between modules to enable vehicle functions like fuel level displays, responsive engine fuel mix calibrations, or airflow monitoring.

Using this system, we demonstrate that it is possible to use data already transmitted between nodes of a vehicular network, such as the data sent between a fuel tank, an engine computer, and the instrument panel, to generate an improved estimate of fuel consumed while making use of existing data samples. We demonstrate that in principle an intelligent scheduler could eliminate requests for redundant information, further reducing the cost of fuel consumption estimates.

First, we identify a system model for fuel consumed, shown as Equation (2.20).

$$m_{fuel}(k+1) = m_{fuel}(k) + \dot{m}_{fuel}(k) \tag{2.20}$$

We see that Equation 2.20 takes the form of a multiple-input, single output (MISO) discrete-time observer from Equation 2.5, with A and $B = 1$. C and D are time-varying and either 0 or 1 based on whether the true state and input are measurable in a given time step.

The true fuel state can be measured, but it is costly. $\dot{m}_{fuel}$, the instantaneous fuel rate, can be measured in six different ways based on the vehicle's configuration and operating state. The six methods appear in the subsequent subsection, in order of decreasing accuracy (determined experimentally).

2.8.1 Fuel Measurements and Estimates

$$\dot{m}_{fuel,actual} = FL \tag{2.21}$$

Equation 2.21 directly measures the Fuel Level FL, which is accurate but can be costly or difficult to sample as not all cars support the reporting of the parameter over OBD. Motion of the car can significantly impact the accuracy of this measurement unless multiple samples are time-averaged and filtered.

$$\dot{m}_{fuel,1} = (IPW) * (RPM) * (V_{injected}) \tag{2.22}$$

The first estimate is Equation 2.22, where IPW is the injector pulse width, RPM is the rotational frequency of the crankshaft, and $V_{injected}$ is the volume of fuel injected per unit pulse width (determined experimentally).

Next, Equations 2.23 and 2.24 use the Mass Air Flow (MAF) sensor to measure the amount of air "consumed" by the engine. Using stoichiometry, one relates the amount of air through the engine to the amount of fuel consumed. Here, MAF is the air flow rate, $AFR_{stoichiometric}$ is a constant based on the fuel composition and engine calibration, and λ_{O_2} is the oxygen sensor's output, which measures the exhaust to monitor unburnt fuel. These values are a function of Δt, as the time sampling interval for this sensor is nondeterministic and non-uniform.

$$\dot{m}_{fuel,2} = \frac{(MAF)\Delta t}{(\lambda O_2 AFR_{stoichiometric})} \tag{2.23}$$

$$\dot{m}_{fuel,3} = \frac{(MAF)\Delta t}{(AFR_{stoichiometric})} \tag{2.24}$$

Equations 2.25 and 2.26 make use of Manifold Absolute Pressure (MAP) as another means of approximating airflow, and therefore fuel consumed. In these equations, MAP is the Manifold Absolute Pressure, $\eta_{volumetric}$ and $V_{displaced}$ are functions of the engine geometry and design, m_{molar} is the molar mass of air, IAT is the Intake Air

Temperature, and R is the gas constant [178].

$$\dot{m}_{fuel,4} = \frac{(RPM)(MAP)(\Delta t)(\eta_{volumetric})(V_{displaced})(m_{molar})}{(120)(IAT)(R)(AFR_{stoichiometric})(\lambda_{O_2})} \tag{2.25}$$

$$\dot{m}_{fuel,5} = \frac{(RPM)(MAP)(\Delta t)(\eta_{volumetric})(V_{displaced})(m_{molar})}{(120)(IAT)(R)(AFR_{stoichiometric})} \tag{2.26}$$

Variations arise because of non-uniform temperature, atmospheric pressure, volumetric efficiency, and more.

The final metric used as input to our estimate is not an OBD parameter, but is a sensor frequently found in smartphones used to collect OBD data. This measure is based on the premise that "any data is good data." In this case, we assume fuel consumption relates linearly to acceleration, ACC as in Equation 2.27 (other studies have preferred the use of acceleration squared) [179].

$$\dot{m}_{fuel,6} = |ACC|\Delta tk \tag{2.27}$$

where k is a constant determined experimentally.

We see the relationship between cost and accuracy for these methods in Figure 2-10.

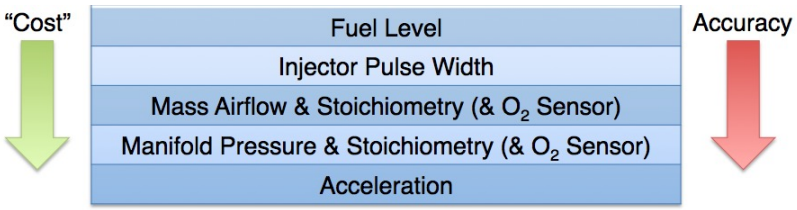


Figure 2-10: The cost and accuracy of fuel estimates are coupled. As cost decreases, so does accuracy. In this system, we make use of the "best available" method for which we have data in any given time step, to take advantage of the increased accuracy. Because we rely on intercepted data already transmitted between modules, the error is minimized without increasing the cost.

The measurements from a real vehicle drive cycle are compared in Figure 2-11.

Equations 2.22-2.27 may be used to develop distinct or fused estimates for fuel consumed. Equation 2.21 allows direct fuel level measurement at a higher cost. To

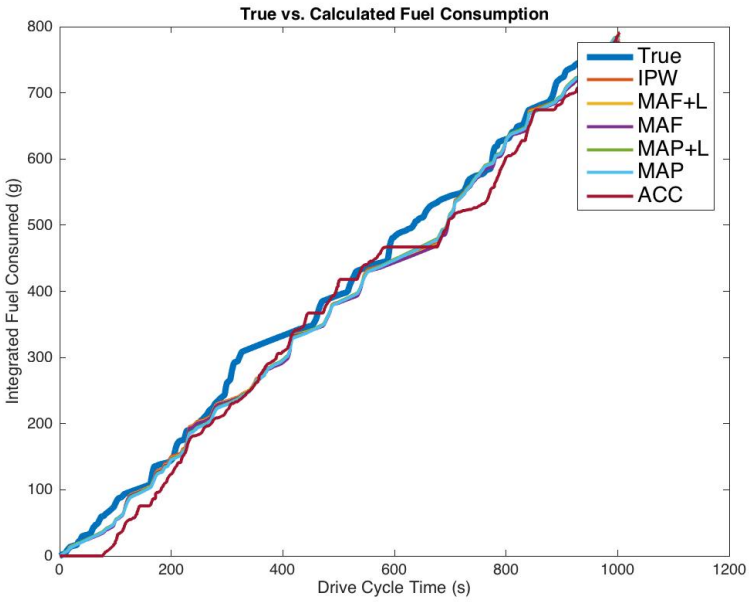


Figure 2-11: Comparison of reported fuel consumed from various sources, assuming no data outages.

support our attempt at cost minimization, each sensor was given a fixed and variable cost of acquisition. An improved model could include bandwidth costs, power drain, and impact on network loading.

2.8.2 Observer Results

Sample data were collected and testing against three fuel monitoring approaches: the "best available" approach, which relies on fuel data from the most accurate measurement at each time step, the "observer" approach, using these same data, and the "raw single input" approach, wherein each individual method is calculated without filtering. An initial estimate error (difference between the true and estimated system) was applied to aid in demonstrating convergence.

A set of fuel consumption estimates were made using each method ranging from 0% to 100% data availability to illustrate the impact of outages on error and cost. Next, a scheduler was implemented to represent how a conventional system might perform. Each sensor was given a random, fixed update rate. We plotted the results of the "best available" approach, as well as the "best available with intelligent sampling,"

	Best	Deduplicated	Observed
Error (% of true)	3.20%	3.20%	1.85%
Cost (% of full, direct sampling)	6.60%	5.39%	5.39%

Table 2.2: The observed fuel level has improved error relative to the "best available" method, without an increase in cost.

which has the same error as the best available approach, but applies a posteriori knowledge to calculate the minimum resource spend necessary to collect only those samples which were used. Finally, the observer was run on the best available case to improve the state estimate, shown in Figure 2-12. A comparison of results is shown in Figure 2-13.

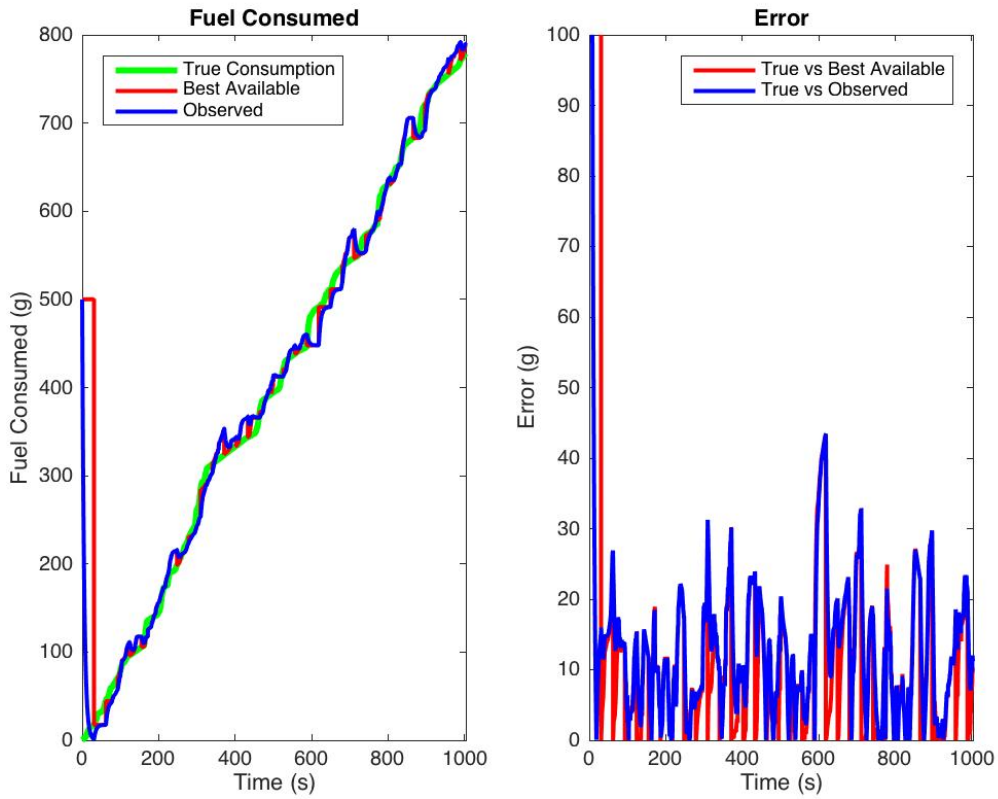


Figure 2-12: Comparison of constant direct measurement and observer-based "best available" model; note the observer converges immediately.

From Table 2.2, we see that observers may facilitate improved data accuracy relative to unfiltered measurements, without increasing cost. This same method of

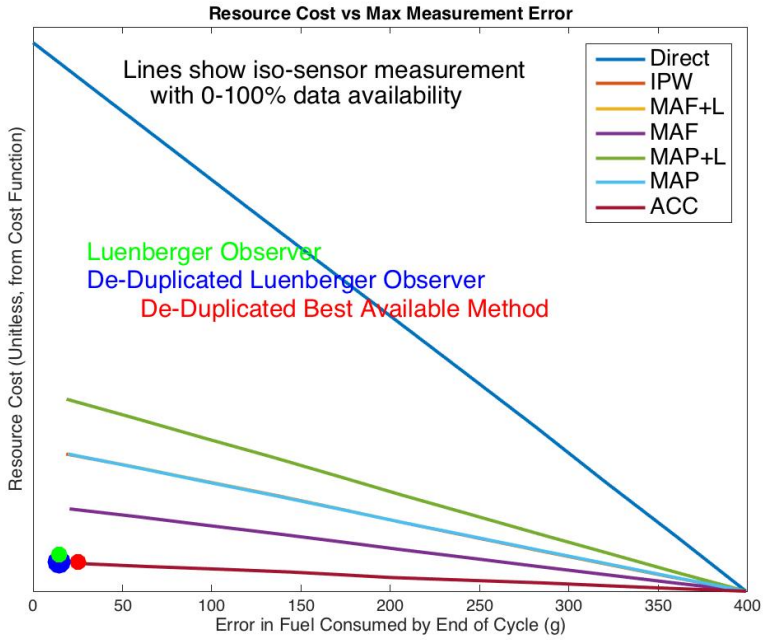


Figure 2-13: Comparison of error and cost in fuel consumed by time $t = 1000$ for different methods of measurement and estimation. Lines represent single-method solutions with data availability varied from 0 to 100. The dots represent the results for the "best available," "best available with intelligent sampling" and observer approaches. Note that the observer has the lowest cost and error of any estimate.

state estimation may be applied to other applications, such as indirect oil viscosity sensing [55], battery state of charge [180], tread depth measurement [54], fuel trim monitoring [181], and many more automotive and non-automotive applications.

2.9 Application of an Estimator to Vehicle Miles Traveled (VMT) Monitoring

The previous scenario examined how an observer supports the creation of Proxies with improved accuracy and no increase in cost of data acquisition. This section approaches a different problem from the perspective of minimizing resource use while targeting a maximum acceptable error limit. This type of optimization will allow applications to be run on devices with limited computation, power, or bandwidth.

The goal of our demonstration application is to estimate the distance traveled by

an existing vehicle for the purposes of a VMT application [99] wherein a per-gallon fuel tax is replaced with a miles-driven tax to ameliorate declining tax revenues stemming from the introduction of hybrid and electric vehicles. To collect data, a car was instrumented with a mobile phone containing an accelerometer, WiFi, and a GPS receiver. The accelerometer measures vehicle acceleration, the WiFi communicates with vehicle On-Board Diagnostics via an ELM327 WiFi adapter, and the GPS provides position anchoring.

The application QoD requirements require the error to be kept within a fixed distance bound per unit time while conserving battery life on the mobile device. To that end, the phone accelerometer is the most cost-effective sensor, with a low ($1\,\mu W$) power consumption and reasonable accuracy, but significant noise. The OBD sensor is a "mid power" sensor using the WiFi chipset for the phone to gather velocity data from a vehicle, reducing integration error. The third sensor is a "high power" GPS sensor capable of measuring the true position of a vehicle but with high, but bounded, error.

A vehicle was driven along a straight stretch of road with the phone mounted such that the longitudinal axis was aligned with the direction of travel of the vehicle. Data were collected from all inputs at a fixed 20Hz, with GPS and OBD data interpolated linearly to match the accelerometer's real-time data. No further processing was done to remove measurement noise. The true distance was measured and validated by using wheel odometery (from the instrument panel) and map-matching for six one-mile trips.

To reduce mobile phone battery consumption, the Kalman filter from Equations 2.12-2.19 was modified to make use of "selective visibility" to simulate intentional non-sampling of data. The sensors were given per-sample costs ($c_{GPS} = 5000$, $c_{OBD} = 200$ and $c_{ACC} = 20\,\mu J$) and sampling period minimum and maximum limits ($10\,s < T_{GPS} < 500\,s$, $5\,s < T_{OBD} < 105\,s$, and $1\,s < T_{ACC} < 51\,s$). The maximum allowable MSE for position was set to be $2.25\,m^2$.

2.9.1 Estimator Rate Optimization

A forward simulation was conducted using real data, projecting the MSE at the end of the drive cycle for various sampling schemes. Feasible and infeasible approaches are plotted in Figure 2-14, and the cost of these approaches is shown in Figure 2-15.

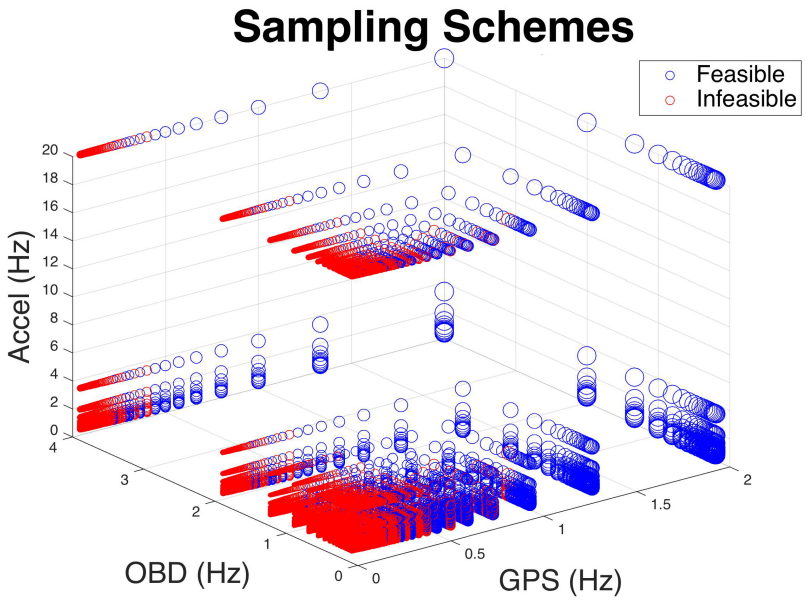


Figure 2-14: A figure of the infeasible and feasible regions. Feasible (blue) and infeasible (red) sampling schemes meeting the target MSE are plotted. Note the clustering of infeasible schemes near the 0 sampling rate, and how this plot highlights the dependance of localization on GPS versus other techniques such as OBD. Circles are scaled to reflect resource requirements.

.

The full-sampled approach had a resource cost of 1618.2 µJ with an MSE of 0.1562 m^2 at the end of the drive, as shown in Figure 2-16.

The Kalman filter was able to reduce data requirements while keeping the error within the targeted bounds. After simulation, the optimal sampling frequencies were determined to be $f_{GPS} = 0.11$ Hz, $f_{OBD} = 0.51$ Hz and $f_{ACC} = 0.39$ Hz, providing a reduced cost of 92.60 µJ and a MSE of 2.216 m^2.

These values indicate a resource reduction of 17.47 times, with a smaller increase in MSE of 14.19 times, which is still within the acceptable range.

To validate that the solution we tested works in other scenarios, the cost of the OBD measurement was changed from $c_{OBD} = 200$ µJ to $c_{OBD} = 2000$ µJ and the

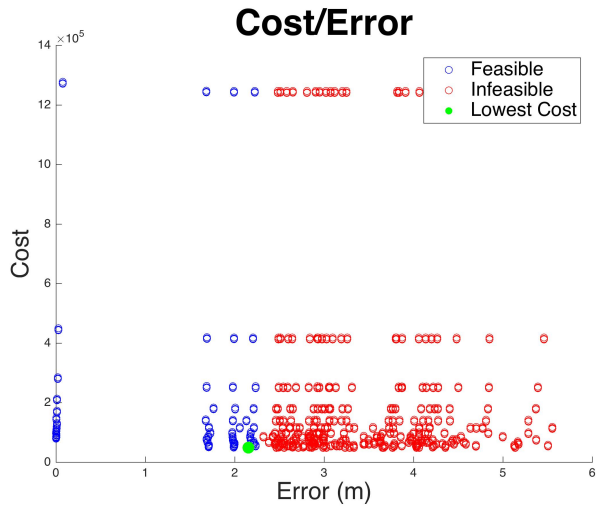


Figure 2-15: This plot indicates the cost versus positional error after executing the estimator on the distance traveled data. Note the clustering in cost — this striation is an artifact of the widely varied cost of acquisition for each data input.

cost of GPS measurement changed from $c_{GPS} = 5000\,\mu\text{J}$ to $c_{GPS} = 100\,\mu\text{J}$ to present an extreme solution. In this case, the full-sampled approach has a resource cost of $62\,372.0\,\mu\text{J}$ with an unchanged MSE of $0.1562\,\text{m}^2$. The optimal solution selected by forward simulation selected a new sampling scheme of $f_{GPS} = 0.5\,\text{Hz}$, $f_{OBD} = 0.2\,\text{Hz}$ and $f_{ACC} = 0.39\,\text{Hz}$, providing a reduced cost of $588.9\,\mu\text{J}$ and a MSE of $2.066\,\text{m}^2$.

As expected, we see a relative increase in the GPS sampling frequency and decrease in frequency of sampling OBD. The cost is reduced by a factor of 107.44, with an increase of MSE of 13.2 times. This sort of simulation-based optimization has massive implications for battery life on a mobile VMT measuring device and for the creation of Data Proxies in general.

While this approach used a single training set for forward simulation (a "drive cycle" for the estimator), a variant could make use of multiple training sets as input and take the ceiling of each sampling rate to provide a likelihood metric that the targets will be met.

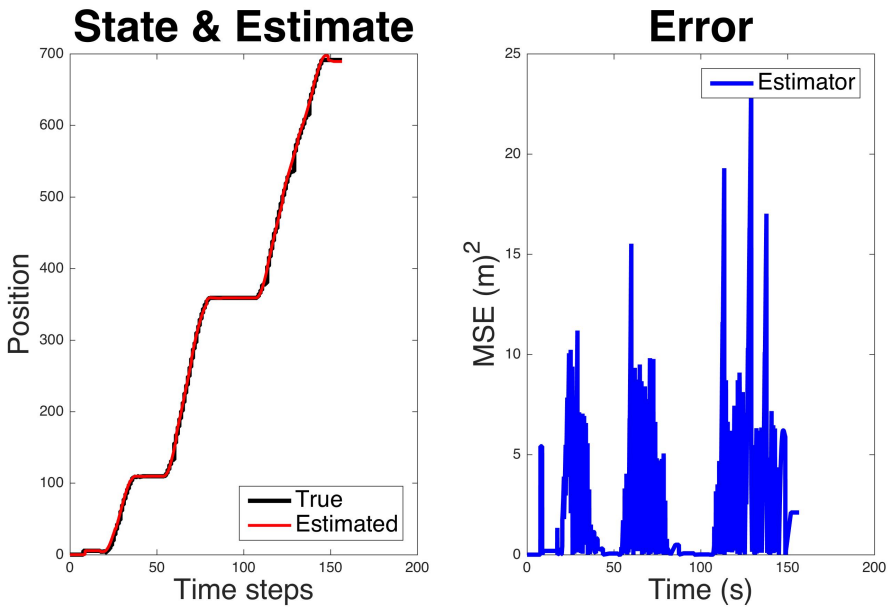


Figure 2-16: Real system showing impact of selective visibility and major and minor anchoring data

.

2.10 A Generalized Approach to Data Proxy Design

This technique may be readily applied to optimizing connected systems. Though it is difficult to generalize the application of mathematical models to reducing resource expenditure, it may be said that observers are better in low-noise cases where system models are known and estimators are better in cases where systems data are properly characterized. This is demonstrated in Figure 2-8.

More generally, we propose a system model selection process in Figure 2-17.

As model accuracy is critical, we propose the use of Model Reference Adaptive Control (MRAC) or a similar feedback-based adaptive control system to apply real data in order to "tune" system models with data-derived insights [182]. These techniques automatically adjust system controllers in realtime so as to maintain a specified level of performance. Such mathematical tools will contribute to the overall success of Data Proxies. In fact, many control system design approaches may be directly translated to the design of gateway Data Proxies.

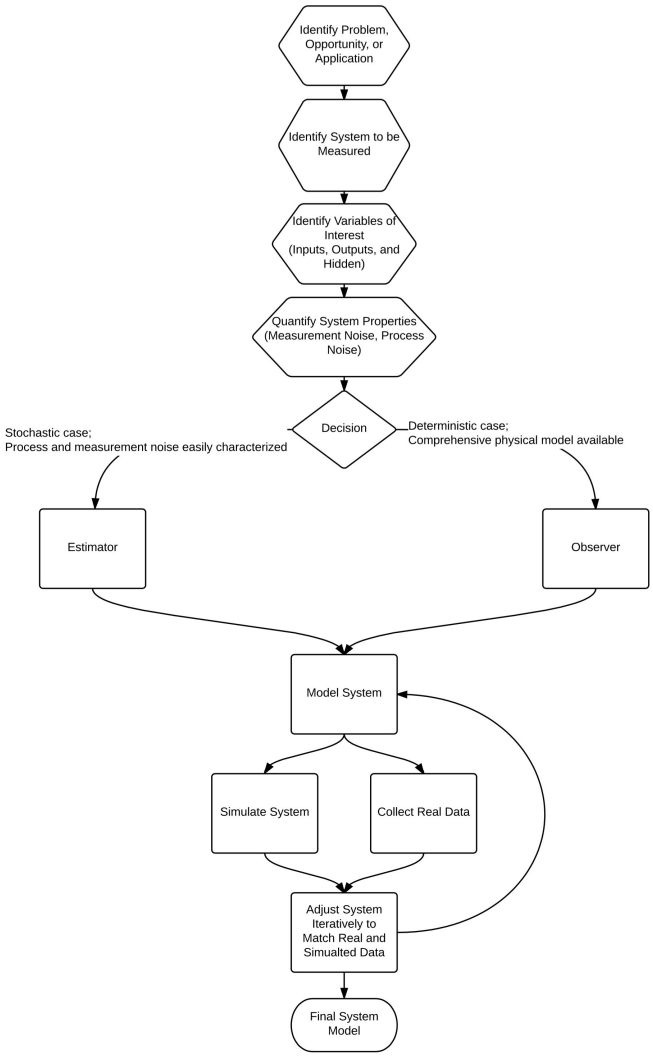


Figure 2-17: Decision flow diagram for generation of system model.

2.10.1 Optimization

From these examples, one sees that it is possible to optimize an application for bounded error subject to constraints. Some data, like the sampling ranges for each sensor and maximum allowable error, help to define a feasible solution region.

Because observers and estimators are predicated on physical or statistical models, one implicitly assumes that the model is accurate and with well known variances. If one trusts these models to be accurate, they may be forward-simulated using training data to determine a feasible region meeting QoD requirements. Tools such as Discrete

Choice, mixed-integer nonlinear optimization, or dynamic programming may help to select the final schema from within this region.

This solution will change as additional applications are added to or removed from accessing a Proxy, such as when users of an application log in or out. With scalable Cloud computation, a set of application "payloads" may be simulated prior to launching a scheduler or new applications added in faster-than-realtime, so that the Application Agent may always select the optimal sampling rate.

In the case there are no elements in the probable, bounded feasible set, a determination must be made to determine whether to use the closest point to the feasible region or whether to report no solution. Further, the convergence speed and sensitivity to reduction in sampling rate for each input must be considered before selecting the final sampling arrangement.

We present a sampling optimization process making use of forward simulation, shown in Figure 2-18. This same process could be used to minimize a number of cost functions for data acquisition or transmission, from sampling and computational cycles, to the number or cost of sensors polled.

2.11 Data Proxy: Conclusions and Implications

This section has shown that the use of Data Proxies can reduce resource requirements for connected devices while maintaining strict error bounds, simultaneously improving security of connected systems and improving system utility. This wider-reaching connectivity through reduced resource use facilitates better data, from more sources, with improved noise rejection, sensor life (due to reduced sampling), and enhanced temporal and measured resolution.

2.11.1 Control

A variation on this system may be used to abstract feedback-based actuation into the Cloud or a hub, enhancing security and reducing the impact of latency on system control. Effectively, this system is like "Cloud cruise control," in that actuators use a

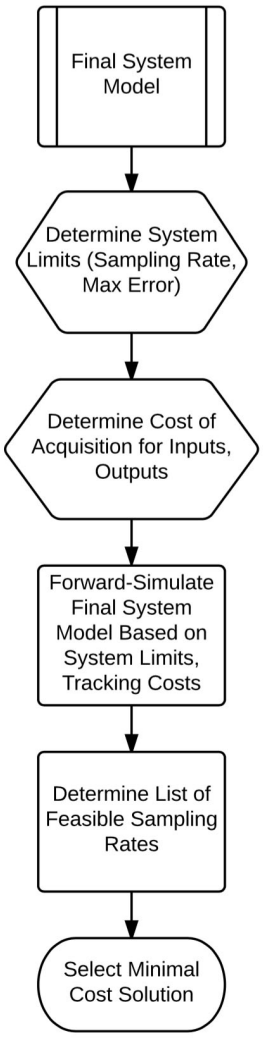


Figure 2-18: Typical process flow diagram for determining optimal sampling rate.

dynamic model to control their outputs. This could be used to improve performance and security, e.g. in a factory, by reducing or eliminating the requirement for direct control. Proxies may also be used to split control between remote applications, Cloud-informed models, and local control.

2.11.2 Security and Privacy

We explore the implications of a Cognitive Layer to improving security and privacy of a Smart Home. If a malicious party gains access to an individual's connected kitchen

and attempts to turn on a microwave on for 300 minutes, IoT platforms today would allow the command to go ahead as requested. While we could try to prevent access, there are still possible failure modes such as a trusted node sending a malformed request. Machine intelligence is necessary — for example, if one asks a person to turn on the microwave for 300 minutes, that person might ask for the instructions to be repeated or deny the request. In the same way, a Cognitive Firewall will assess the reasonability of data and control requests through the use of high-speed simulations to validate the impact of commands on the physical model.

As a simple example, one may apply the example of a moving particle as was demonstrated in the proof-of-concept for the Data Proxy in Section 2.7.1. In this scenario, the particle might have a safe velocity or acceleration limit. The Cognitive Firewall would evaluate the effect of the requested input force and determine if there was a statistically significant likelihood that the particle velocity or acceleration limit might be exceeded, and if so, forego execution of the command.

The Cognitive Firewall, as with other types of firewalls, has explicit rules, such as velocity limits, implicit limits, such as maintaining control system stability, and may even have learned limits based on machine learning (when the system executes command X, Y happens and Y is undesirable). Using this technology, malicious code and hardware, such as was used during Stuxnet's attack on Iranian uranium enrichment centrifuges, could be identified, or unauthorized hardware nodes changed or added to a network could be identified. This approach would be helpful in implementing robust and secure automotive e-Inspection, for example, to ensure that there are no simulation devices installed on vehicle networks.

Cognitive Firewalls can be autonomous or transition to human-in-the-loop to clarify commands, using something similar to two-factor authentication. This approach is similar to how contemporary autonomous cars run control models — when the models break down, drivers get short notice and resume control until the uncertainty has passed, and the system learns how to avoid the same scenario in the future.

These same models may be applied for local-area networks to improve responsiveness, such as the drive-by-wire system in a vehicle. In the event an input such as a

throttle pedal position sensor fails, the Cognitive Firewall may raise an alarm and prevent unintended acceleration or another possible harmful output. This firewall may be extended to allow improved virtualization and sandboxing of connected device, protecting sensitive data streams and actuators.

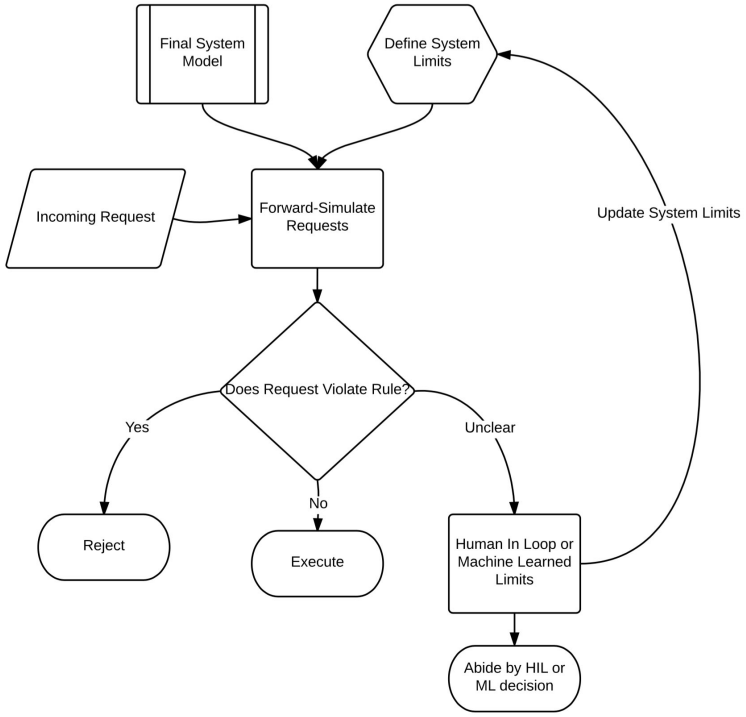


Figure 2-19: A sample process flow for evaluating commands and updating rules for the Cognitive Firewall.

The Cognitive Supervisor has application in monitoring systems that can fail or drift outside of specifications. By observing system changes over time, un-modeled events can be identified. In a Connected Car, the Cognitive Supervisor could provide data to inform a new version of On-Board Diagnostics. In the case of a Connected Home, this could be as simple as monitoring heating system energy input and temperature response to detect that a window has been broken, changing the transfer function. In a Smart Factory, this could be applied to monitoring that a machine tool has broken, worn down, or heated up beyond its useful life.

This builds upon the ideas proposed in Wu (2014) et al, which suggests developers instill in objects the ability to learn, think, and understand the physical world through

the integration of human cognition into the design process. This same paper poses the idea of applying outlier analysis to threat detection and blocking with minimal human intervention and a "pseudo-autonomic" perception-response cycle using sensor inputs to passively modulate physical system outputs [175].

A similar machine-learned model has been demonstrated to great effect in conventional networked systems [183], capable of applying intermittent human-in-the-loop supervision to identify evolving incoming attacks limited false positives. Where the Cognitive Layer differs from Veeramachaneni's work is in the consideration of the physical system model used as the basis for the Data Proxy, which may allow improved accuracy over AI^2's 86.8% detection rate. Additionally, the Cognitive Layer's Firewall can determine accidentally-malicious commands from trusted parties. The Cognitive Layer monitors the connected system in it's entirety, whereas AI^2 focuses solely on intrusion detection. These protective elements can even be protective in the event that developers inadvertently "overprivilege" applications, which was a frequently-occurring problem within Samsung's SmartThings platform.

The Cognitive Layer works with the Security Layer to limit intentionally and inadvertently malicious action, which reduces the likelihood of a third party gaining access to private data. Integration with visualization tools [159], privacy controls [98], and anomaly-based notification can further improve security, provide an assurance of privacy, and aid in rapid response in the event of intrusion.

2.11.3 Improved Real-Time Monitoring

The use of observers in connected devices, particularly connected vehicles, has vast implications for monitoring device performance over time. For example, systems can gain improved temporal resolution via interpolation, sampling at 1Hz but allowing reasonable measurement accuracy at 10 Hz. Fixed-cost acquisition systems (e.g. pay-per-kilobyte connected systems) can generate richer models without increasing the cost of connectivity.

In plants, these same data can be used for geographic interpolation, to assess the temperature conditions around a large factory using a minimum of sensors. Failures

can be predicted using these inferred data. A disparity between real data and the physical or statistical models for a system could indicate that a device is out of specifications. Similarly, the same logic could be used to identify when systems designs are inconsistent with physical models, such that events like the Volkswagen diesel emissions scandal, Mitsubishi economy misreporting, or Toyota unintended acceleration could have been identified and corrected earlier [184] [185] [186] [187]. The complexities in this approach revolve around characterizing noise versus aberrations.

2.11.4 Vehicle Design

The concepts of the Data Proxy and Cognitive Layer may be applied to WAN and LAN in vehicles, improving car-to-Cloud connectivity with enhanced data availability. Dedicated Short Range Communication may be improved as V2I nodes can be extended to estimate the state of vehicles at central location. In-car units benefit from a shift of computation from sensors to more central (or Cloud) control units, while state estimation may be used to reduce network loading, allowing more devices on a single Controller Area Network with less weight.

The Cognitive Layer approach might be used to incentivize the use of gateway modules in cars to lessen the chance of car hacking.

2.11.5 Sensor Life

The use of proxies and a scheduler optimize power consumption and allow battery powered devices to last longer. In a factory or a field with hundreds of battery-powered nodes, we can use this framework to maximize useful life.

A related application of Data Proxies is useful to increase the working life of passive RFID-tag based sensors, or TABS (Tag Antenna Based Sensors) [188] [189]. These sensors use measurable RF properties to pervasively sense the world around them. The benefits of TABS are low cost and low power, but at high read rates, the RFID tags burn out ($\tilde{2}$0,000 reads). Even in systems without batteries, we can make sensor life longer and make pervasive sensing more attractive by reducing the number

of times the sensors are polled to acquire data for an application.

2.11.6 Future

In the future, this work may be extended to provide a generalizable framework or software tools to automate the sampling design and optimization process. Additionally, Data Priority may be incorporated into the QoD request metric and the Application Agent will be enhanced to consider probabilistic models based on multiple sets of input data.

There is the opportunity for additional work exploring the implications of thin versus thick client sensing - for example, sensing could be done locally to a sensor and processed prior to uploading. Finding the optimal balance of local aggregation and realtime live data provides a substantial area for exploration.

Finally, the transient effect on system convergence and rate minimization of changing application payloads should be examined, including the impact of applications joining or exiting the scheduling system.

2.12 From Architecture to Applications

Data Proxies address the issues of system security and resource management, but addressing these alone is not a sufficient enabler for connected applications. Individual applications must still be designed carefully to minimize the resource input requirements necessary to operate, and simultaneously manage system installation and operation costs. The next portion of this chapter considers how a particular application addressing the opportunity for consumer-facing, comfort and efficiency improving applications identified in Chapter 1 might be implemented. Exploring vehicular connectivity, I consider developing an efficiency-improving idle time predictor and how technology choices impact this predictor's performance.

A perpetual challenge in designing Connected Vehicle applications is determining the optimal data input, connectivity method, and computation location. Some applications might be addressed by several approaches; others require the use of a

specific technology to optimize the vehicle's performance, efficiency, or comfort. The application presented in the following sections, intended to predict idle times in order to minimize start-stop system intrusiveness, can be designed in a number of ways with varying inputs and connectivity – but with dramatically different prediction accuracies and system cost depending on the chosen embodiment. I consider data timeliness, availability and bandwidth cost in the context of enabling this application.

This remainder of this chapter explores the implications of varying input richness, connectivity methods, data availability and other factors on this efficiency- and comfort-improving application. An idle predictor aims to improve the user experience of automotive start-stop technology, which reduces fuel consumed by turning idling engines off, but for which the system may be overly-aggressive and result in driver disengagement. The use of a predictor applies contextual information to eliminate short idle shutoffs in Automatic Engine Start/Stop systems, minimizing driver annoyance and improving compliance. Rather than providing an exhaustive framework for determining optimal application locality, the findings of this section provide thoughtful exploration of the solution space for generalized Connected Vehicle development.

2.13 Motivating Resource-Forward Thinking

The landscape for Connected Vehicles is evolving as automotive Original Equipment Manufacturers (OEMs) begin to apply connectivity and sensing technologies to improving user experience, efficiency, and reliability. A consumer desire for "smart" cars, government- and insurance-backed pushes for enhanced safety, and increasingly comparable mechanical designs have led manufacturers to focus on software as a differentiator between vehicles [190]. Recently, this software differentiation has migrated from in-vehicle toward distributed vehicular networks, with applications ranging from collaborative vehicle control, to data-informed efficiency and comfort optimization, to collision avoidance [191] [192].

The use of historic and realtime sensor data to inform vehicle control is not new [193]. Vehicles have generated data since the introduction of the first automotive

computer, and shared this locally-created information across the modules within a single car for decades [194] [195]. Today's novel sensing and communication technologies, however, put developers in a position to apply data to richer improvements in the vehicular operation, use, and design phases than were previously possible. These applications are facilitated by the ability of Connected Vehicles to create massive, aggregable datasets, the availability of powerful embedded systems, and the use of rapid analytical tools. With enhanced connectivity and Advanced Driver Assistance Systems (ADAS) providing sensor data useful for remote applications, automotive engineers now more than ever must treat vehicles as nodes in a network and consider how best to interact with the larger transportation system.

2.14 Application Development Considerations

Algorithm design is increasingly important to defining vehicle architecture – an automotive application's output may have varied efficacy depending on data inputs, the communication and computational systems involved, and availability of training data. Other intrinsic and extrinsic factors for consideration range from system cost and complexity, to data availability and system robustness, to sensor noise characterization.

The design of an application is split into automotive architecture, remote server architecture, and communication technology, with design choices having impact transcending these divisions. For example, implementing a rich sensor payload and using in-vehicle computation to pre-aggregate data will significantly reduce the amount of information to be transmitted relative to a system without pre-aggregation. This same system will have higher initial cost and increased round-trip latency, so its accuracy might suffer. Key considerations include the method of connectivity and computation location, as these more than anything dominate an application's resource consumption and cost of deployment.

We present an exploration of these and other considerations for resource-conscious automotive application development. Following a general discussion, we demonstrate

a method for evaluating application feasibility and performance using a real-world application designed to mitigate AESS annoyance.

2.14.1 Storage Requirements

Depending on the application, storage requirements will range from negligible to massive. These data may be stored in the vehicle itself or in the Cloud.

Simple applications could make use of locally-generated data and discard these immediately, requiring no in-car storage. Advanced applications might require access to high-quality maps, temporal and geospatial histories, or training sets that are gigabytes in size and distributed across remote servers. Intermediate systems may opt to decrease in-car storage and improve data freshness by streaming information to the vehicle based on trajectory, route history, or other predictors. Still other systems aim to reduce communication and bandwidth costs, decreasing application latency but increasing in-car storage costs.

Data stored in-car have a one-time fixed fee relating to memory size and type, whereas remote storage tends to be priced based on usage and with reoccurring fees. The need to move data between locations is an additional consideration, as bandwidth between the car and the Cloud and from the server to the Internet can add significant cost.

It should be noted that the cost of supplying cellular bandwidth is decreasing, but publicly available pricing data shows that consumer and business per-byte rates have remained largely constant in spite of increased network capacities and improved spectral efficiency [196] [197]. With the decreasing cost of flash memory relative to the cost of cellular bandwidth, systems that do not require "fresh" data will tend to maximize in-vehicle storage. This approach has the additional benefit of reducing data read and analysis latency and eliminates a point of partner reliance. For systems requiring live data or enhanced computation, Cloud connectivity offers a more attractive solution.

2.14.2 Computational Complexity

Computational complexity is a constraint for in-car computation, with OEM price targets making even modest increases in the cost of embedded systems untenable. This is witnessed by the lack of encryption on in-car computer modules; the required overhead drives costs too high to be economically attractive [198]. Only after negative press have OEMs begun to explore increasing computation and dedicating some of this computation to security improvements [130]. A new consumer willingness to pay for occupant-facing technology (infotainment, etc.) may continue to drive local computational improvements in the near-term.

While some applications may be pared down to run on embedded vehicle hardware, complex applications – like those requiring classification based on a massive data set – will require the computation afforded by scalable Cloud infrastructure.

An application's computational complexity drives an architectural decision of whether vehicles should transmit as a thin client or a thick client (raw versus processed or aggregated data) [99]. A thick client, which performs computation prior to uploading data, offers bandwidth savings but requires more in-car computation. A thin client, which transmits sensor data as it is received, consumes more bandwidth but requires little local processing.

Thin clients tend to be costlier but more future-proof and scalable, making this streaming approach attractive for applications that will require analytical tools not yet widely available or to support applications that have been conceptualized but not reduced to practice. Thick clients offer lower latency, which is beneficial in user-facing and safety-critical applications.

There has been discussion of migrating a vehicle's non-critical computing, like infotainment systems, to the Cloud. This will allow continued software improvements long after initial sale, but this architectural change brings with it challenges of bandwidth, latency, privacy, security, and even software licensing.

As with storage, computation has costs that can be either fixed, as in the case of in-vehicle technologies, or variable, as in the case of Cloud connectivity.

2.14.3 Algorithm Robustness

Connected automotive applications must be able to contend with network unavailability, process noise, and sensor outages. Algorithm robustness is an important consideration, especially for those applications relating to safety where a failure could have grave consequences.

Applications may be designed to account for occasional outages, failures, and gaps in coverage through failsafe and fallback modes. These alternate algorithms may be less resource-efficient, offer lower accuracy, or provide less timely response than the application's primary algorithm, but offer a means of continuing operation under less-than-optimal conditions.

The need for redundancy and robustness means that developers must design and maintain an up-to-date set of alternative algorithms and select the best available at any point. This could be as simple as a routing application relying on historic data rather than realtime data due to a sensor failure or sparse Connected Vehicle density, or as complex as a system switching from DSRC-enhanced platooning to offline highly automated driving in the event of communication failure.

Resource cost and value added by applications should be considered when designing to enhance robustness. There may exist a crossover point after which the application's value is not worth the cost of implementation or operation.

2.14.4 Latency

Connected applications by definition require data from an external data source. Frequently, applications require "just in time" data arriving from a remote endpoint to the vehicle within a maximum latency threshold. A delay could result in the application being starved of information and losing efficacy or ceasing to function entirely. In the case of applications that both send and receive data, such as an application that handles car-to-car communication through a Cloud relay, the latency must be considered for the round trip.

Network latency is determined by network loading as well as the chosen connectivity

technology. The cost and typical latencies for common connectivity technologies are not widely available, as technology standards specify best-case scenarios and the real world latency differs dramatically. If network latency is an issue for an application, a faster radio technology should be used (if one is available).

In addition to network latency, computation latency impacts the system. Computational latency may be addressed by scaling up computation if using an extensible Cloud infrastructure. Otherwise, the algorithm itself might require simplification.

Depending on the cause of the bottleneck in a given system, e.g. high latency due to dropped packets, it may be possible to adjust the system architecture. This can take a number of forms, such as pre-aggregating data, reducing the amount of information sent and therefore improving latency.

2.14.5 Transmission Reliability

As alluded to in the discussion of algorithm robustness, applications often make use of data from external sources, but must be capable of operating without these data. It is important to define whether data communications for an application are intended to be best-effort or required for functionality. If connectivity must be assured, this impacts other design choices and can dramatically raise system costs. For example, a system that requires always-on, low latency connectivity may require the use of a combination of both DSRC and 4G or 5G modems, greatly increasing the fixed networking hardware and variable bandwidth costs for a connected application. In other cases, where bandwidth is free or inexpensive, as is the case using only DSRC for WiFi for connectivity, transmission reliability may be less of a concern.

This architectural decision is similar to the decision of weather to send TCP or UDP packets. TCP guarantees packet delivery but significant overhead. UDP sends packets knowing that some will get lost, but the cost of implementation is reduced.

2.14.6 Database Freshness

As vehicle computation and connectivity improves, database freshness becomes an important issue. For geo-spatially aware applications, there is a question of how frequently the map database must be updated and whether an old database is totally invalid or if a delta/incremental update suffices (a delta update performs an incremental upgrade by adding new information and deleting information that is no longer valid, rather than replacing the entire database).

Today's typical model for navigation systems is to provide maps at time of sale and to leave maps untouched unless the end-user initiates an update. This can result in poor route selection due to construction or changes in traffic patterns. For applications of this nature, updates can improve user experience. This can be done by updating databases in their entirety at regular intervals or downloading buffered updates based on vehicle trajectory or similar inputs. Selective and delta updates helps to minimize bandwidth consumed - if a vehicle has never driven outside of Michigan, it likely would not make sense to download maps of California. Similarly, if new roads are being added but old roads are unchanged, only the new roads need to be transferred. The use of data buffers can also reduce the in-vehicle storage requirement.

In Connected Vehicle applications, a map can be any informative database that the vehicle or application references, from engine calibrations to routing tables. Map freshness matters most in cases where data are rapidly changing, such as traffic patterns, inputs for a low-cost fuel finder, or debris-tracking applications (in contrast to invariant data, such as altitude maps for improving hybrid powertrain control). With a hybrid approach, maps may be updated locally with data learned by the vehicle itself, and these data may be sent to a remote server for incorporation into the larger data set. An alternative to costly cellular map updates is to make use of a network without a per-byte or flat rate access fee, such as WiFi.

2.14.7 Data Sparsity

In mobile networks, nodes are often sparsely distributed or moving and therefore only briefly within a technology's connectable range. The resultant data transience may have dramatic impacts on application efficacy. Beyond radio type, sparsity is also determined by the number of vehicles capable of interchanging data within a particular region – if there are one hundred cars, but only one is connected, an application may only access limited data from the nearby fleet. This is the reality of connected technologies today and likely to be true for the foreseeable future; an historic unwillingness for automotive OEMs to share data between brands leads to a similar scenario already, with data from existing Connected Vehicles stored in siloed, private Clouds.

2.14.8 Initial Hardware Cost

Automotive applications necessitate the installation and use of sensors, actuators, and communication hardware. Connectivity requires a modem or radio, while sensors and actuators provide input and output for applications. Each of these elements contributes to the system setup cost.

Assessing the system base cost for a single application is complicated, as radios, modems, sensors, computation, and storage must be allocated relative to the suite of applications making use of each resource. In the scenario where technologies are deployed and reserved for future use, the value is even more difficult to quantify.

Note that each element has associated non-dollar costs and benefits. For sensors, these cost/performance tradeoffs may include noise and precision, while in radio technology, latency, range, and peak throughput must be considered.

Though bleeding-edge radio and sensing modules are costly, economies of scale tend to lower the cost of implementing these technologies in vehicles. Volume pricing for current and next-generation modules is difficult to obtain and constantly evolving. There is a range of literature available discussing the cost to implement specific RF, sensing, and actuation technologies.

2.14.9 Continued Operating Cost

Applications requiring connectivity have ongoing operating costs for bandwidth, remote storage and computation, and monthly access fees (for cellular lines, database access, updated maps, etc.).

Bandwidth and line access costs depend on technology and are negotiated with telecommunications companies based on the number of clients, volume of data, and contract duration. These fees cover application and database updates as well as regular data transmission. While solutions like DSRC have no bandwidth cost, these technologies may require augmentation with cellular technology to function when network coverage is poor. Such redundancy must be factored into calculating true operating cost.

Remote computation and storage pricing depends on data volume and whether the servers are owned and operated by the application developer, OEM, or a third party. As of the time this document was submitted for publication, typical Amazon AWS pricing as of publication may be calculated at `https://aws.amazon.com/ec2/pricing/`. A scalable Cloud instance may appear costly, but the resources are used to support several vehicles, making the per-vehicle cost significantly smaller. If an application is to be deployed across a sufficiently large fleet of vehicles, it may become cost-effective to operate one's own servers. This approach can provide economies of scale benefitting all applications provided by an OEM.

An application's ongoing cost may also include the cost of accessing information from an external API or other pay-per-use database, as well as the cost of operating a Cloud platform for data storage, aggregation, and service. The cost of licensing maps or other data sources can be significant, and in cases, it may be less expensive to vertically integrate and generate data in-house.

2.14.10 Application Accuracy, Efficacy, and Performance

Application performance may be quantified a number of different ways. Perhaps the most obvious consideration is the effectiveness of that application, which might

be reduced to a single metric such as classification accuracy. While an example application might strive for the best possible accuracy, another would do well to attain the best accuracy possible within a set of constraints. Still other applications target a particular accuracy, minimizing the cost to attain this result. This is similar to the ways in which Data Proxies are optimized to meet a particular Quality of Data.

Depending on the application, other metrics may be needed to quantify success. Not all applications can be reduced to an accuracy metric – others might seek to minimize driver annoyance, maximize savings or time savings, or to increase the mean time between component failures. Quantifying "success" is important to the creation of an objective function.

2.15 Designing for a Specific Problem: Idle Time Prediction

In Connected Cars, the most perceptible improvements are applications that change how vehicles operate in realtime. Drivers can observe changes as they happen and immediately appreciate the benefits. We examine one such problem with real-world implications: the issue of poorly-optimized stop/start systems for reducing vehicle engine idle time. Systems today annoy drivers to the point they disable the feature, negating any potential fuel savings. Addressing this challenge has the potential for fuel savings and emissions reductions which will become increasingly important in the future. Therefore, we use this application as a case study in exploring an application's resource requirements and relative efficacy.

Start-stop technology is a system that reduces fuel consumption by shutting of an idling internal combustion engine when a vehicle is at a standstill. The engine is restarted automatically when the driver is ready as signaled by pressing the throttle, releasing the brake, or releasing the clutch. This system has become popular in recent years as increasingly stringent Corporate Average Fuel Economy (CAFE) targets are implemented by United States governing bodies. These same systems may also be

used for auto manufacturers to gain EPA off-cycle credits [199].

The stop-start system is effective at saving fuel when it is turned on, with up to 10% savings in real-world testing [200]. However, drivers are frequently annoyed by the system's disruption during short idle scenarios, such as those occurring at a stop sign or a crosswalk [201]. Additionally, the system can be confusing for drivers, as the vehicle remains on and discharging the battery even while the engine is off. We have heard firsthand reports of drivers leaving the car running by mistake as he or she goes into a store, with the push button start making it possible to exit the vehicle without shutting down. As a result, many drivers turn the system off (we conducted a brief survey determining up to 33% of drivers regularly disable this feature, resulting in a relative fleet-wide increase in fuel consumption of 3.3% – though a larger survey conducted by Mercedes puts the number closer to 11% [199]).

A system capable of estimating idle times accurately reduces driver annoyance and promotes engagement of the system, netting an overall fuel savings. This predictive application is an example of real-time control through a connected or otherwise data-informed vehicle application. Though just one example, this technique of using context information to anticipate driver and vehicle needs could be used to inform future vehicle design.

2.15.1 Drive Cycles

Powertrain technologies like Automatic Engine Stop Start (AESS) are designed and evaluated with the use of drive cycles. Drive cycles are derived from real-world driving and are used to predict the performance of a vehicle's powertrain. These cycles are standardized time-speed series used to ensure consistent and comparable test results across suppliers, OEMs, and testing agencies. For this reason, standardized drive cycles are used to calculate a vehicle's fuel efficiency for reporting the United States Environmental Protection Agency, which allows auto manufacturers to test their vehicles in a laboratory and self-report the results.

While drive cycles are useful for powertrain design and evaluation, the cycles lack data richness. Though drive cycles ensure repeatable simulation, they are more

representative of an idealized scenario than real-world use. Further, it is easy for manufacturers to optimize vehicles for these test scenarios without clear benefit to (or at the expensive of) real-world performance. For example, in a vehicle we tested to quantify the performance impact of stop/start technology, we found that the vehicle itself disabled the start-stop system due to variations in road incline, outside temperature, and steering wheel angle [202]. While this may reduce driver annoyance and improve system compliance, it artificially boosts the fuel economy represented in EPA figures. Typical drivers would be unlikely to attain this sort of efficiency.

There is an opportunity to improve upon the drive cycle using real-world data to inform control strategies and future vehicle design. Such a solution would integrate richer information from a fleet of vehicles and develop cycles mirroring driver behavior and environmental conditions, rather than relying upon a particular representative velocity profile. This means of evaluating vehicles would have far reaching impact when it comes to reducing true fuel consumption, improving drivability, and more. We propose an extension to the drive cycle called the "Gap Cycle" and suggest how this data-rich approach may be applied to estimating idle times.

2.15.2 The Gap Cycle Hypothesis

As part of creating an idle time estimator, we introduce the concept of the Gap Cycle. The Gap Cycle is similar to a drive cycle in that it provides a standardized means of evaluating vehicle performance. Unlike the drive cycle, which distills rich data into only speed and time, the Gap Cycle is based on a driver behavior model (target velocities, maximum and minimum accelerations) and environmental factors. Because this model is predictable, the Gap Cycle may be converted into a conventional drive cycle with ease.

Drive on any busy road and consider the cues one uses to know when to accelerate or brake. External visual stimuli like brake lights turning off determine when a driver will begin to roll forward in traffic. Brake lights activating on a leading vehicle indicate that the following driver must slow down. A glance at a pedestrian crossing signal may determine how quickly a green light may turn yellow, or when red may turn to

green. A driver stuck in traffic may notice a truck further down the road begin to move and let off their own brakes, expecting the intermediate cars to begin to move (provided these drivers are not consumed by texting or other distractions).

This context information – from infrastructure, like traffic signals, and from environment, like distance to cars ahead – provides the cues many drivers use to operate their vehicles. We hypothesize that this information may be useful to estimating vehicle idle times. Relative vehicle positions and velocities, absolute location, and even time of day combine with a model of driver behavior to provide the information we need to understand a vehicle's real-world performance. In particular, we choose to focus this implementation of a Gap Cycle on location and the relative position and velocity of the car(s) leading the target vehicle.

With information about the relative distance from a driver's vehicle to leading vehicle(s), we intend to build an application to anticipate future movement and estimate idle times. This estimation will be used to eliminate short-idle shutoffs, minimizing driver annoyance and promoting the sustained use of stop/start technology. Such an approach seamlessly avoids driver bother without the extreme restrictions OEMs apply to their systems.

2.15.3 Approaches to Estimating Idle Time

To reduce annoyance and maximize fuel savings, we desired the ability to estimate idle time such that short idle shutoffs could be eliminated without disruption to the start-stop system's engagement during long idles. After observing driver behavior, we determined that location, proximity to leading vehicles, and relative velocities might be used to accurately estimate the duration of an upcoming idle.

We considered three approaches for predictor data inputs: local data, car to car data (mesh networked), and car to Cloud data. Each approach has different data availability, latency, and sparsity.

For classification, two approaches were considered: a Naïve Bayes classification approach, which examines the entire data set and finds the closest match based on a probabilistic approach, and nearest neighbor matching, which relates the current

vehicle's situation to a subset of the database of historic idles. Based on the results from a separate analysis, the kth Nearest Neighbor (kNN) estimates were universally better [201]. We therefore explore only the kNN approach in this document.

2.15.4 Data Generation

To generate data for these Gap Cycles, we required real and simulated data. Simulated data allowed us test hypotheses at scale, while real data were used to validate the simulation model's accuracy.

We simulated and collected the location of a vehicle, as well as distance to and velocity and acceleration of the two leadings cars. These parameters were simulated with traffic and intersection models and collected in the real world using GPS, ultrasonic sensors, LIDAR, and RADAR. These sensors are representative of a typical vehicle's ADAS sensing configuration. This Gap Cycle's measured parameters are shown in Figure 2-20.

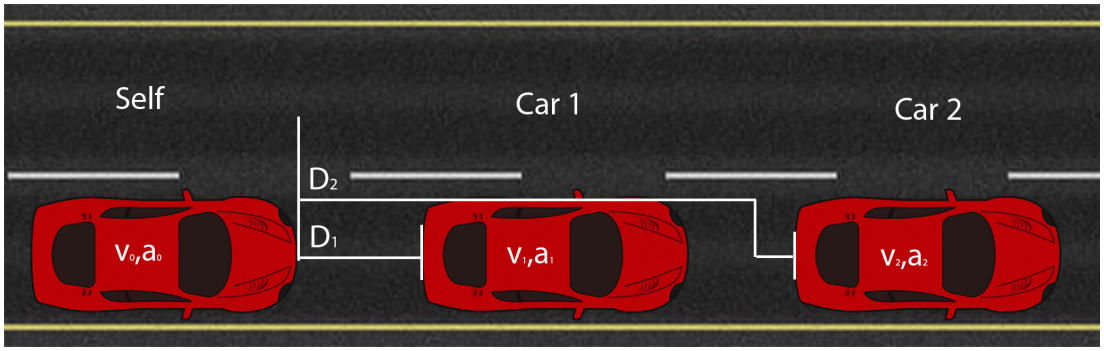


Figure 2-20: Parameters measured to inform the Gap Cycle and idle time estimator include the relative position, velocity, and acceleration of three vehicles.

2.15.4.1 Simulation

To simulate traffic data, we built a 1-dimensional intersection model and created a parametric simulation to iterate over intersections of varying length, car density, and traffic light timing. The setup of this simulation is discussed in Dylan Erb's PhD dissertation [201]. This simulation provided data for idle time prediction. Each vehicle

in the simulation had a velocity, acceleration, and position to be used in generating training sets to characterize typical idle events.

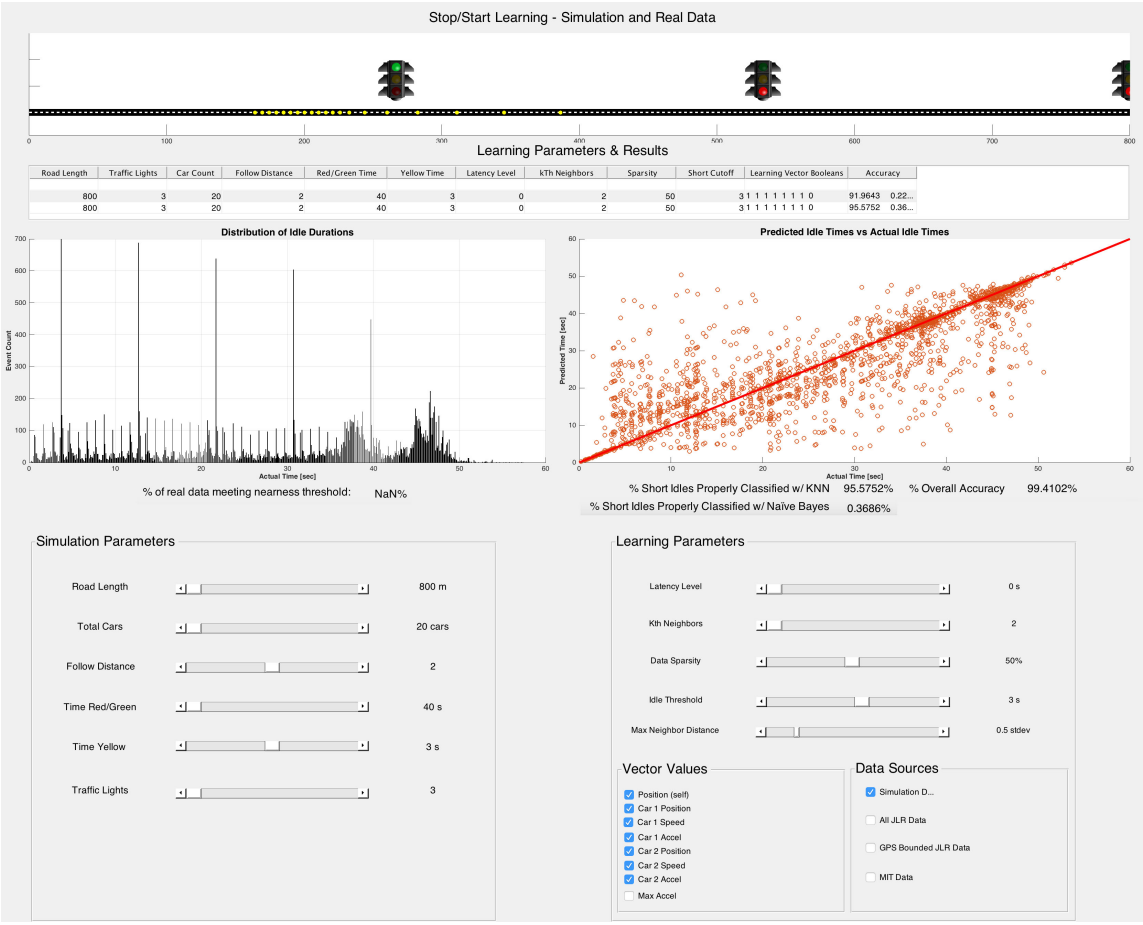


Figure 2-21: Data simulator and classification GUI, showing parameter adjustment sliders, classification results, distribution of idle durations, and example 1D road segment.

2.15.4.2 Real Data

We worked with an automotive OEM to access a set of real-world vehicle data to validate the simulation results. We parsed, processed, and filtered data to extract information about idle events. While we were able to capture a large number of idles (a portion of these are shown in Figure 2-22) and examine the trends for idle time distribution, we identified issues with the data's provenance. The logging setup was not self-consistent, and the results, though generally accurate, would contaminate our

training set. Factors such as corrupted positioning data or mislabeled data could lead our algorithm to fail.

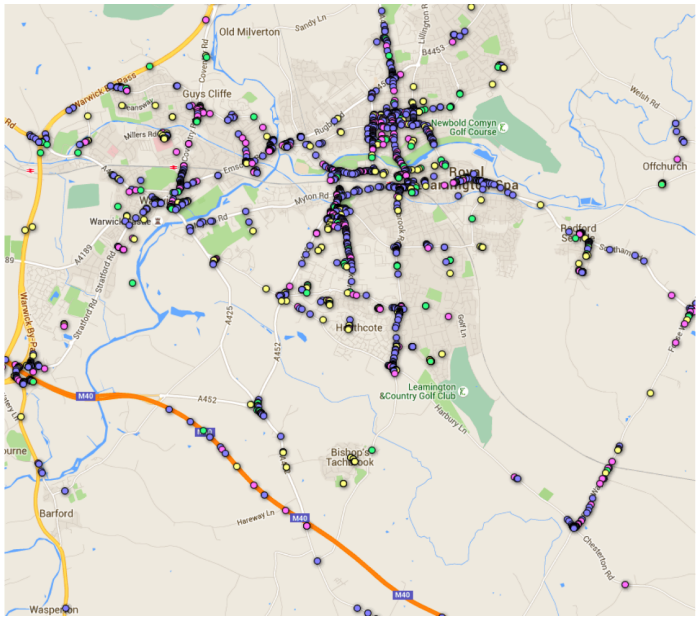


Figure 2-22: Sample map indicating idle locations and duration (by color, with blue green short idles and red long idles).

There were additional challenges to using the OEM-collected data. As an example, the data set did not record pertinent vehicle sensors at all times. These sensors were not broadcast across the in-vehicle network at all times, meaning their values could not get mirrored in the Cloud except when the vehicle control units requested information from the parking assistance module. For this reason, we were only able to access ultrasonic parking assistance sensors when the vehicle was in reverse or operating under a certain speed. The cumulative density function for number of distance measurements relative to temporal distance from an idle event is shown in Figure 2-23. When dealing with an application where latency matters, such as is the case in idle estimation, the unavailability of high-frequency data rendered this input unusable.

Other sensors were not available in the data set at all due to thick-client pre-processing and reporting. We were therefore unable to access RADAR information, meaning we had no access to the second lead car's position or velocity.

We determined that to validate our idle prediction hypothesis, we needed to build

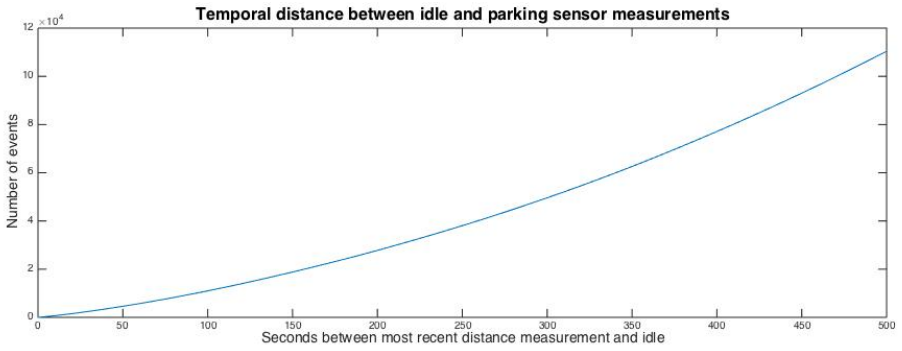


Figure 2-23: The data provided were lacking recent distance measurements for the majority of idle events. Here, one sees that the number of idles available for analysis increases with time from the most recent distance sensor polling.

our own data collection system.

2.15.4.2.1 Data Collection Setup To provide real data for model validation, we built three variants of data collection hardware.

The first setup, which we called LIDARBro V1, (Figure 2-24) made use of front- and rear-mounted LIDARLite V2 sensors, a uBlox 6 GPS receiver, and an Arduino microcontroller. This hardware variant recorded distance from the car ahead of and behind the sensing vehicle. Measuring the lead and following distances allowed us to shift our frame of reference to that of the trailing car, getting the apparent distance, speed, and acceleration of two leading cars. This approach allowed us to better simulate data that would be available with full access to the RADAR sensor or via DSRC or other connected technology. LIDARBro V1 was retired as the Arduino was incapable of handling the high volume of incoming data without corruption.

LIDARBro V2 made use of front and rear window-mounted LIDARs, AdaFruit Bluefruit LE microcontrollers, an OBDLink MX Bluetooth OBDII adapter, and a 10 Hz SkyPro X160 GPS receiver to provide enhanced sensing capabilities. We replaced the Arduino with a MacBook Air in order to capture the high volume of data without error. This setup allowed us to capture data at the maximum possible sampling rate for each sensor, but suffered from connectivity issues in areas with significant 2.4GHz coverage. Additionally, because this system made use of the MacBook as a realtime display, we were able to determine that the LIDAR sensors we used were unable to

Figure 2-24: LidarBro V1, a self-contained GPS and LIDAR logging unit.

record distances to reflective or transparent surfaces, causing a loss of data when aimed at rear windows and chrome.

The final iteration, LIDARBro V3 (Figure 2-25), shifted from a front/rear configuration to a dual-front configuration with Adafruit Bluefruit Feather microcontrollers, LIDARLite V2 sensors, and MaxBotix SONAR to address the issues of lost distance measurements due to reflectivity as well as the interference-related connectivity issues. The use of dual front sensors ensured redundancy and minimized the likelihood of data loss due to reflectance or translucence.

To further improve richness of data capture, we tested two vehicles equipped with similar logging hardware in convoy. We therefore could use the GPS and OBD data from each vehicle to simulate dense vehicle connectivity even if the LIDAR and SONAR failed. We were able to successfully log information for over 400 idles using a repeated route, lending to dense data for training and testing. This setup did suffer from clock synchronization issues we would like to address in a future revision.

The data collected by this setup were sufficient to validate our simulated model's accuracy at a small scale, and identified interesting idling trends (such as the fact that crosswork-related idles, which were not factored into the simulation, were almost exclusively short and could be predicted by location alone). The results from the real data agreed well with the simulation for our short, predetermined route in terms of the distribution of idle times, and we were able to validate that distance, speed, and acceleration data did create training sets of neighbors (we refer to these training

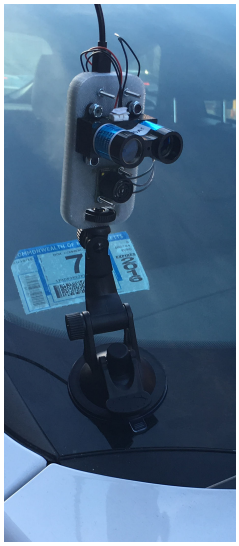


Figure 2-25: LidarBro V3, featuring SONAR and RADAR for redundant front distance sensing, which made use of separate Bluetooth-enabled GPS and OBD interface devices.

databases as "maps") useful for predicting the duration of an upcoming idle event.

Since the real data and simulation agreed well, and because it was possible to generate richer simulation-based training sets more rapidly than data-informed training sets, we will refer to the results of the simulation in the remainder of this document. These data provided us with rapidly increased data density and enhanced repeatability over the real-world set that we collected and a related OEM-provided data set (both of which are described in Erb's dissertation [201]).

Finally, the use of this real-time data collection setup lends itself to future implementation of intelligent Start-Stop using the same system. To that end, we extended LIDARBro V3's setup by using a Komodo CAN interface capable of interfacing with our Python script, and reverse-engineered the start/stop control switch signals on a 2015 Mercedes CLA. We intend to implement a demonstration of data-informed idle-time prediction with short-idle shutoff disabled using this system. It may further be feasible to integrate in-vehicle RADAR and GPS data to eliminate much of the logging system's external setup.

2.16 Optimizing Application Locality

The purpose of this section is to explore the implications of choosing different data inputs, architectures, and communication methods for a real-world application. In the following section, we examine the idle time estimation application and how these choices impact system cost and prediction accuracy. We follow a design process exploring hypothesis validation to technical feasibility to exploration of the ideal computation and storage split. To start, we made assumptions for use in developing a comprehensive cost function.

The assumptions guiding the cost calculation for the idle time predictor appear below:

Years in Operation: 12

Number of Annual Application Transmission Events: $12,000$

Map Efficacy Loss Annually: 5%

Map Replacement Annually: 6%

Map Database Size: $100MB$

Latency (DSRC, 5G, 4G, 3G): $0.5, 1, 2, 3s$

Radio Cost (DSRC, 5G, 4G, 3G, WiFi): $\$800, 200, 400, 600, 30$

Data Cost per KB (DSRC, 5G, 4G, 3G, WiFi): $\$0.000, 0.005, 0.004, 0.002, 0.000$ [1]

Sensor Cost (GPS, Ultrasonic, RADAR): $\$50, 100, 500$

Cloud Bandwidth per GB: \$0.09

Cloud Storage per GB per year: \$0.36

Car Storage per GB: \$0.40

To provide repeatable results, we performed simulation using the real-world validated model for a range of latencies, data sparsities, and input vectors. We took 100 samples of each configuration and averaged the prediction accuracy. These results were used rather than real-world data due to increased availability.

[1]The model assumes costs increase as speeds go up. This is not true in practice, due to spectral efficiency improvements. However, we assume that in the case of Machine-to-Machine (M2M) communication, the extra speed is a "value added benefit" and therefore will be priced higher.

2.16.1 Determine and Test a Hypothesis

We tested our hypothesis using simulated data and validated the simulation using real-world data collected with our test setup.

We found that the Gap Cycle approach to idle time estimation works in theory and in practice. Under optimal conditions, the simulation-informed algorithm returns a high-90% accuracy of properly-predicted idle times, with the real algorithm approaching similar accuracy within our limited testing region.

From this, one sees that our hypothesis about using the Gap Cycle approach to estimate idle times works well. Full results are available in Erb, 2016 [201].

2.16.2 Identify Feasible Technologies

With the hypothesis proven, we move on to explore the design of our sensing and communication systems.

The technical landscape enabling Connected Cars has grown in recent years, and applications are now beginning to take advantage of the technology available. With a growing demand for applications, developers must understand technical limitations and how to maximize return on resources.

Memory and computation costs are decreasing, pervasive sensing is becoming commonplace, and vehicle computation and sensor payloads have been growing rapidly. Until recently, wireless technology limitations and cost had been the limiting factor for many connected applications. This has changed with the commoditization of connectivity and data – sensing and connectivity have changed the landscape of what is possible. With connectivity at the core of modern automotive applications, developers must understand the feasibility landscape of these technologies.

We continue on to explore the technologies with latency and bandwidth characteristics appropriate for enabling our application.

Bandwidth, latency, node density, and range are key considerations when determining the feasibility of connected applications. All technical implementations under consideration should be presented in a feasibility map similar to Figure 2-26. Here,

the limits for bandwidth and latency are based on empirical testing using speed-test webpages, or where this proved infeasible, manufacturer marketing information. From this landscape, new applications may be plotted and feasibility evaluated. The landscape shown is not exhaustive; it is merely intended to provide examples of differing application types to help the reader gain insight into what is "typical" of Connected Vehicle applications and how best to evaluate for technical feasibility.

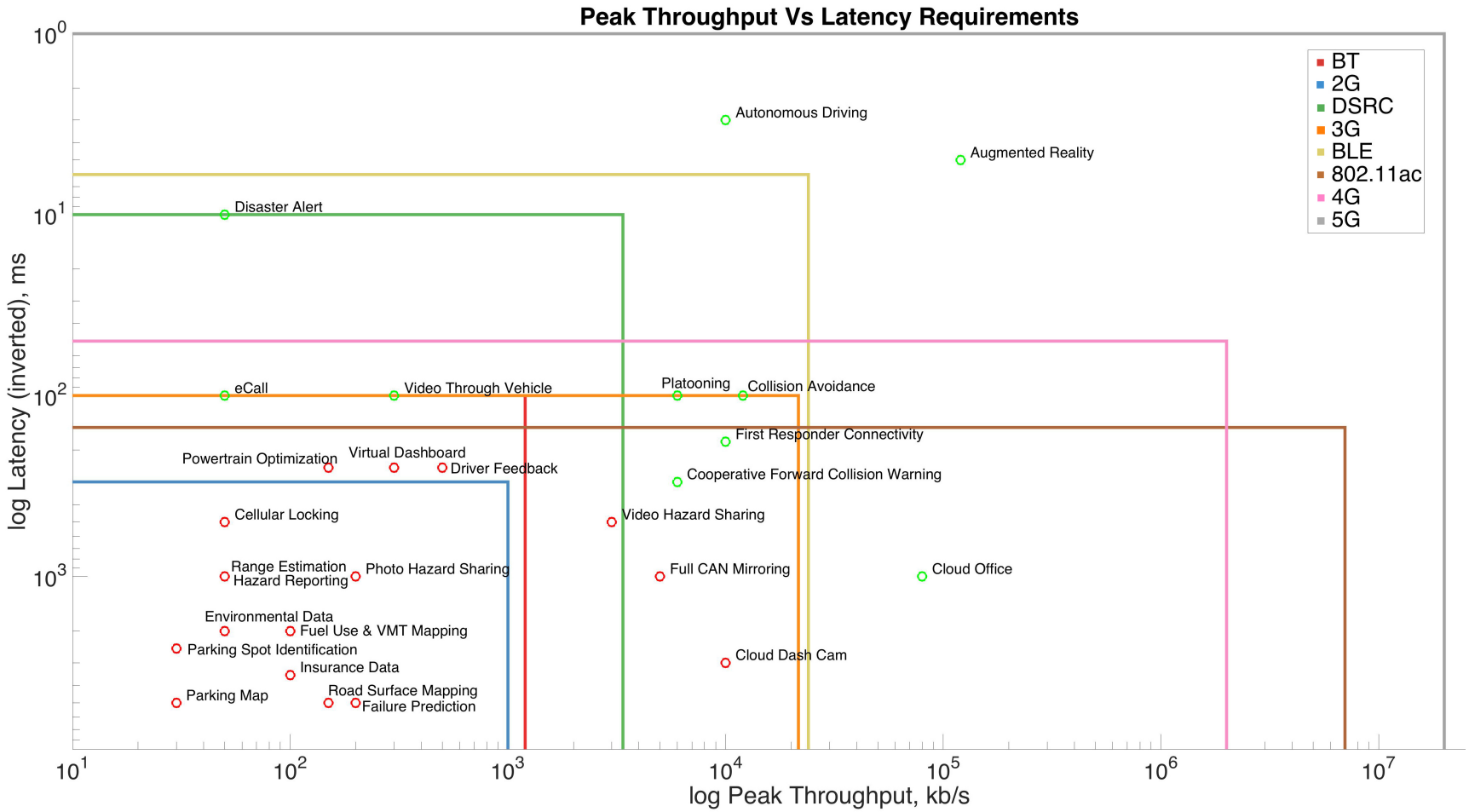


Figure 2-26: This plot shows the contemporary technology landscape for latency and bandwidth, along with common application types and their communication requirements. Applications inside a technology's bounding box are in the feasible region for technology today. Green circles are applications that have been validated with concrete numbers in the literature; red circles are estimates based on the development of similar technologies in a laboratory environment [104] [203] [5] [204] [205] [206].

In Figure 2-26, green circles indicate applications referenced from peer-reviewed sources. Red circles indicate applications that have been approximated based on our experience developing research applications using the CloudThink platform [98] [207]. Where multiple values were simulated or tested in the referenced material, the most technically feasible values (lowest bandwidth or highest acceptable latency) were plotted.

From this exploration, one sees a convergence between future cellular technologies and contemporary mesh networking approaches. This makes next-generation cellular connectivity attractive, considering that bandwidth costs (not necessarily prices) are dropping over time. Applications today can now send more data per dollar than ever before, growing the idea space significantly. Further, the latency for emerging cellular standards will continue to decrease, enabling their deployment in mission-critical applications.

Similar plots may be created for data sparsity vs range and node density vs application type. Creating these landscape maps is a valuable exercise that should not be overlooked as part of the application and architecture design process.

2.16.3 Considering Model Inputs

The idle time predictor can make use of one or more inputs. We consider sensitivity to the presence of several inputs so that only the most useful data may be incorporated into the final model. This avoids incurring excessive cost through acquiring data that will ultimately be discarded.

Our input models were evaluated for average accuracy across 10 randomly selected data sets. The three representative input vectors were generated by using data available from three common sensor inputs (GPS, Ultrasonic Parking Assist (UPA) Sensors, and Adaptive Cruise Control RADAR).

The results of this simulation show that the application accuracy is impacted by a number of factors, such as latency, data availability, and the input data vector. This is visualized in a three-dimensional plot (Figure 2-27) showing application accuracy as a surface for varying latency and data sparsity, for input vectors consisting of local-

data only (blue - position and time), one-car leading data (yellow - position, velocity, acceleration), and two-car leading data (red - position, velocity, and acceleration).

One sees that the two-car model is the most accurate overall, with the one car model working almost as well for low latency communication technologies. The self-only model has overall poorer accuracy, but is robust to latency increases and has no data transmission cost. Interestingly, the self-only model bests the one-car model at some intermediate latencies. This is because the one-car idle time prediction is made on stale data, which may be less accurate than the location-only self-informed model (e.g. relying on data from a car traveling through a yellow light or that only recently started moving).

In a vehicle already implementing the necessary sensing and communications technology, we propose a simple model to select the most appropriate input vector, choosing the most accurate algorithm at any point based on system latency and neighbor data availability (sparsity). Thus, the most elevated point on the three-plot in Figure 2-27 should always be the chosen algorithm. This same approach accounts for loss of connectivity as discussed in the "algorithm robustness" section, 2.14.3.

This algorithm selection technique results in choosing algorithms that work well under most conditions, but this approach may not take into consideration that each model could be tuned and therefore excel under a particular set of conditions. Based on data availability and other factors, there may be conditions leading to "crossover" points where one model works better than another. To improve algorithm selection, more complex models may be created, implementing probabilistic models for accuracy.

Such a scenario may occur when intersection model data beats out live data as would be the case with sparse training data availability, or a scenario with no nearby neighbors for a locally-informed model.

With an appropriate algorithm or algorithms selected, the design process moves forward.

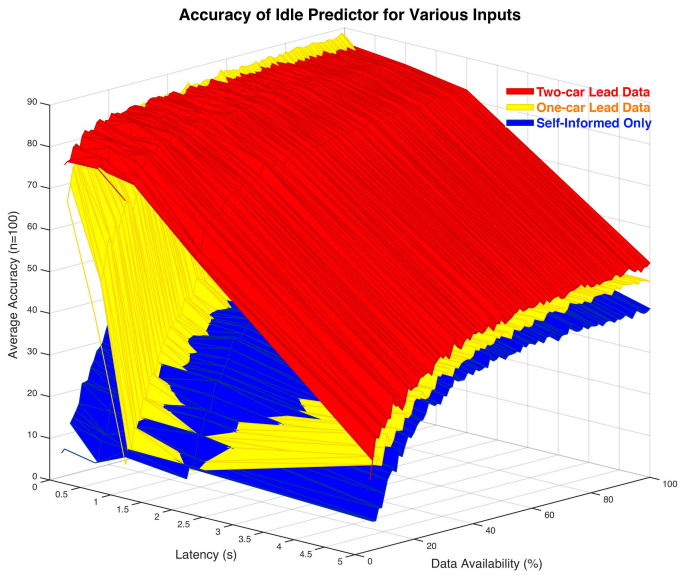


Figure 2-27: Surface mesh represents accuracy for three simulated algorithms: self position only (blue), data from one car ahead data (Ultrasonic Park Assist [UPA] sensor informed, yellow), data from two leading cars (RADAR or Cloud-informed, red). One sees that local-only sensor data with an in-car lookup is latency-invariant; the two-car leading model has additional latency robustness to the one-car model because it is "looking into the future" more by considering the motion of vehicles further ahead. Depending on technology cost, any of these approaches could be viable

2.16.4 Creating a Cost Model

To properly quantify the cost of implementing an application, we must construct a comprehensive cost model. This model includes fixed and variable costs for sensing, data storage, computation, and communication in both the vehicle and at the remote server. An overview of the elements to be explored appears in Figure 2-28.

A simplified model for cost includes the sum of the data communication cost for sensor data and map updates, the portion of the sensor cost and in-car storage attributable to the application, the cost of providing the server's bandwidth, and the cost of storing data in the Cloud,

$$c_{total} = (d_{xmit} + d_{maps})(c_{cell} + c_{server} + c_{cloudstorage}) + c_{sensors} + c_{carstorage}.$$

In this equation: d_{xmit} is the data transmitted to enable the application,

d_{maps} is the data required for map updates,

c_{cell} is the variable cost for cellular or other wireless data communication,

c_{server} is the variable cost of bandwidth at the server side,

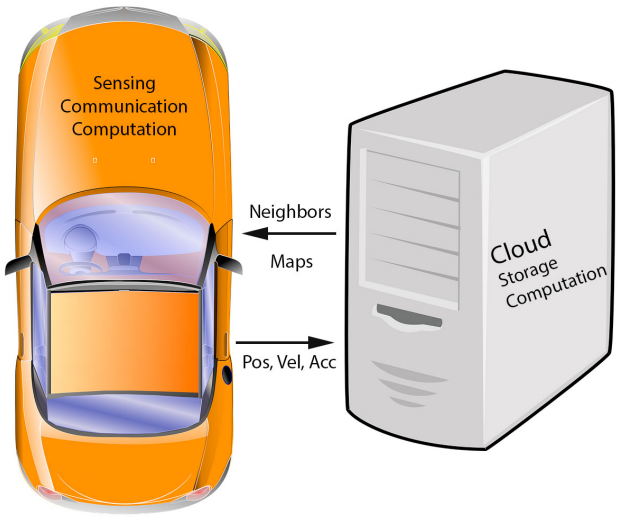


Figure 2-28: This figure represents the typical elements of cost for the idle estimation application.

$c_{sensors}$ is the portion of the sensor (and computation) cost attributable to the application,

$c_{cloudstorage}$ is the cost of storage in the cloud,

and

$c_{carstorage}$ is the cost of providing the in-car storage necessary for the application to operate.

Note that in this model, the base cost of the hardware to enable the system and the cost of Cloud resources may be included in their entirety, calculated as a percentage of the original total cost or ignored, depending on whether or not the hardware is re-usable for multiple applications.

Each cost element may also have a time-varying nature, or be dependent on several other inputs, such as map update frequency, database growth size, years in operation, and more.

To evaluate whether an approach is viable or cost effective, there exist four common objective functions: overall cost optimization, cost optimization for a target accuracy, overall accuracy optimization, accuracy optimization for a target cost.

In practice, the most accurate approach may be too costly or complex to implement. Instead, the value-add and willingness-to-pay for different applications must be

considered, in which case different application variants may prevail. Finally, exhaustive modeling all permutations of a system – such as implementation of thick-client or compression-based models – may not be technically feasible, so the developer's judgement should be used to narrow the search space and identify acceptable compromises.

Note that there may exist a disparity between the payer and the beneficiary for many Connected Car applications. In this function, consumer, third-party and OEM costs – such as the cost of wasted time, price of annoyance, or perceived value add and differentiation – must be considered separately. Ultimately, the decision of architecture may come down to the ratio of dollars a manufacturer or a third party is willing to pay in order to deliver a dollar of value to the consumer or another third party.

2.16.5 Optimization With Constraints

Optimization seeks to reconcile an objective function, such as dollars spent, with application performance. The application developer or OEM may seek to maximize accuracy, minimize cost, find the best accuracy for a fixed cost, or find the lowest cost for a given accuracy, as an example.

In Figure 2-29, we generate all data from the same training vector: self location, and position and speed from leading Car #1 and Car #2. These plots show that the application using connected technology is more expensive than that running stand-alone, and that the cost of map updates dominates the cost of deployment. This is true regardless of parameter load (since the data payloads are relatively small), and highlights the importance of building thick clients, pre-processing, compressing, and sending data only upon change events.

In this case, the stand-alone model wins out initially but loses its edge when the database ages. DSRC remains high-performing over time due to the low bandwidth cost, provided updates are conducted over WiFi or streamed from car to car or infrastructure and the download cost is amortized across a large number of vehicles.

2.16.5.1 Cost vs Accuracy

The relationship between system cost and accuracy can be difficult to compute. In the case of the idle prediction application, accuracy is a straightforward metric of the percent of properly classified short idle events. Cost is calculated from the cost model and assumptions in the preceding sections.

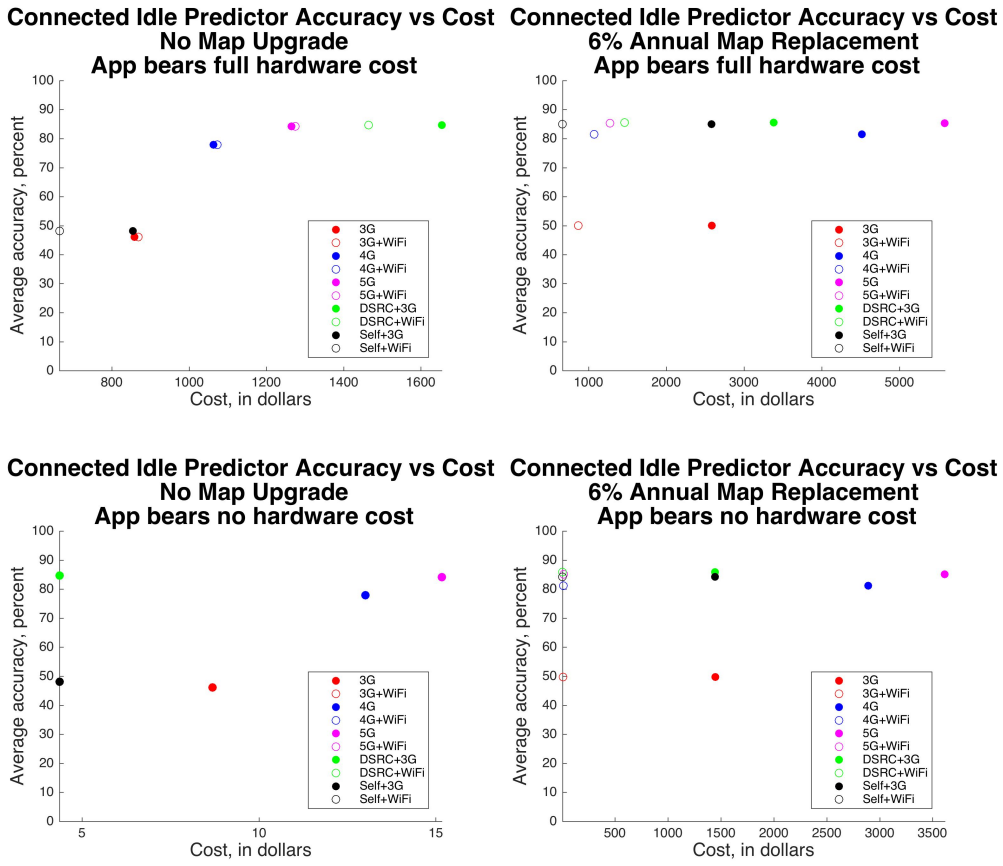


Figure 2-29: This figure shows the cost and accuracy for two systems: one with updated maps, and one with a static map database. These systems are additionally examined as including the cost of the entire hardware setup (sensing and connectivity), as well as their incremental cost assuming the hardware is already in place. Maps update costs are represented using the primary communications technology, as well as bandwidth, to show the impact of map update cost on the application. Note that some metrics may be misleading; for example, DSRC with 3G has a high accuracy and low variable cost but, to attain this accuracy requires the presence of other vehicles within range.

The plot in Figure 2-29 illustrates the cost for an example system, split to demon-

strate a) the high cost of hardware for communication and sensing, and b) the cost and benefit of updated maps. The top plots demonstrate the cost of the application and all related hardware; the bottom plots show the cost of the application's resource needs only. On the left, maps are left to languish, whereas on the right the maps are replaced at a rate so as to just beat data atrophy. One sees that that if sensing and connectivity hardware are already present in a vehicle, the cost of deployment of an application is insignificant as storage and computation prices are negligibly small. If sensing hardware, connectivity hardware, or bandwidth are required, the cost of an application changes dramatically over time.

There may be cases where a particular, expensive technology is required to make an application work as well as required. In this case, the OEM or application developer must determine if the additional cost is less than the additional value realized by the consumer, or if that same technology might be used to enable other value-adding applications in the vehicle. In such cases, where sensors, storage, and communication hardware are applicable to multiple applications, the best model is to attribute a calculated percentage of the initial hardware to this application.

These plots show clearly that cellular database upgrades may in cases be avoided, and that depending on anticipated lifetime of the vehicle these updates can be quite costly. This cost might also be minimized by shifting to a non-bandwidth-cost communication method, like WiFi, mailed discs, or updates installed at dealership visits, as can be seen from the spread in system cost between the cell-only and cell-plus-WiFi systems.

A long-term approach to map storage and updates may be to standardize a common base map and semantics to allow the bulk of a reference database to be shared across applications by different developers. In this model, the OEM or another provided may provide the base map and other applications may augment this data as necessary, reducing cost of duplicate data delivery and storage.

2.16.5.2 Sensitivity to Latency

Idle estimation and traffic prediction are time-sensitive applications. As a result, round trip latency has a significant impact on application accuracy. A high round-trip latency can be thought of as effectively the same thing as a distracted driver missing a traffic light turning green, causing a ripple effect of congestion and lost time for the driver and trailing vehicles.

The results in Figure 2-30 were generated by applying latency to the simulated traffic data. The information made available to the car running the application were offset in time by the latency delay so as to provide stale data. One sees that sensitivity to latency increases with increasing latency, and that accuracy losses are small for short-lag communications. At a certain point, excessive latency decreases the accuracy below 50%, rendering the application unusable.

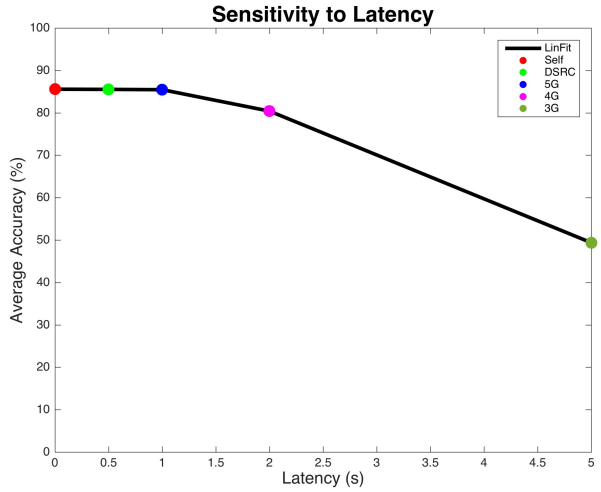


Figure 2-30: Here, the loss of accuracy is related to the typical latency for several common wireless technologies. One sees that short round-trip times are more conducive to this sort of application.

While latency has a significant impact on the accuracy of any time-sensitive application, reducing network delays is not always feasible without changing communication technologies. Many applications that require low latency only require low bandwidth, meaning that targeting a more stringent latency target will drive initial radio and ongoing bandwidth costs higher than they otherwise would be.

2.16.5.3 Sensitivity to Bandwidth

Bandwidth is a driver of application cost during vehicle use. To simulate the impact of different bandwidth limitations, the typical packet size for different data inputs was calculated and used to generate a lifetime data cost based on the assumed bandwidth pricing. One sees from Figure 2-31 that while additional data inputs increase the cost of running an application, increases in accuracy may outweigh the added cost. In the example, the average accuracy for the self-only case falls below the 50% threshold, indicating the prediction accuracy over the vehicle's lifetime is worse than that of a coin toss. Therefore, the self-only case may not be a viable method of operation, and the developer or OEM must determine whether the improved predictor accuracy is worth the increased bandwidth cost.

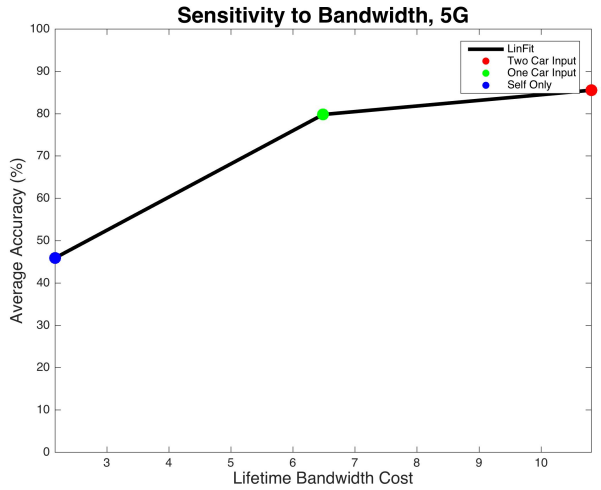


Figure 2-31: This plot shows the impact of raw parameter transmission relative to application accuracy. Map database update costs are ignored, to highlight the impact of sending additional, uncompressed or unprocessed data.

2.16.5.4 Sensitivity to Sparsity

Many of the idle time estimation algorithms rely on networked data from nearby vehicles. Whether data come from on-board sensors or a training set, a dense data set is required to improve classification accuracy. As data sparsity increases, application accuracy diminishes. The impact of sparse data depends on several factors

including network technology, latency and bandwidth limits. We modeled sparsity by eliminating elements randomly from the set of neighbors used for classification.

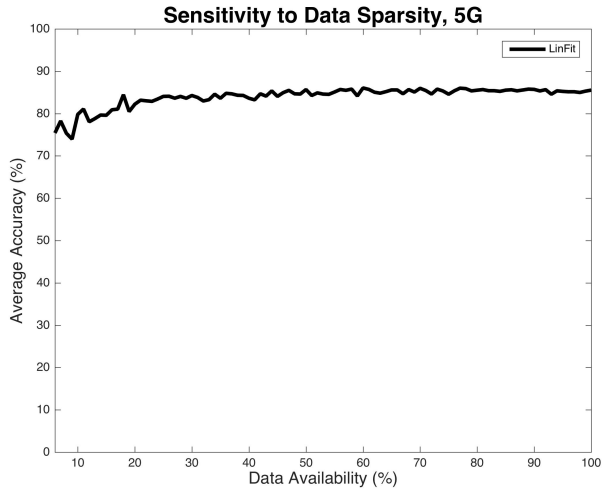


Figure 2-32: Prediction accuracy decreases with increasing data sparsity. This is true for connected as well as non-connected applications.

The plot in Figure 2-32 shows a knee around 20%, indicating that, though helpful, additional data has diminishing returns. This type of plot may be used to determine the minimum size data set required to deliver a particular accuracy for an application. The same analysis may be performed for in-car and in-Cloud data sets to determine an application's sensitivity to training data availability in these different locations.

2.16.5.5 Sensitivity to Freshness

Local databases work well for applications where histories are not time-variant. In other cases, replacement becomes necessary. This can be in the form of wholesale database replacement or delta replenishment, in which case new data are added and/or a subset of the old data are removed. Map updates over WiFi can add significant value to an application with minimal additional cost.

For predicting idle times, the referenced neighbor map starts fresh, but must be kept updated to deal with geospatial changes (road construction), engineering changes (traffic light timing), social changes (driver behaviors), and more.

For this application, database freshness was modeled by assuming map atrophy

over time is linear and reducing map richness by increasing the sparsity of the data set. Data freshness was modeled using the training data's sparsity as a surrogate metric. Freshness was evaluated by assuming data loss at a certain percentage per year, with data replenishment at another rate. Data loss was held constant, while replenishment was varied to show the impact of losing or gaining reference data.

Figure 2-33 shows that data replenishment can keep applications, even time sensitive ones, working well for the lifetime of a vehicle. The data loss model assumes cars are updated upon leaving the factory, and that efficacy losses are linear and discrete in nature. One sees that the application accuracy tends to plateau once the maps have reached a particular size; should map databases shrink, the application accuracy falls off dramatically and may not have sufficient data to function. In this case, if the maps are allowed to degrade by a net 3% year over year, the application will stop functioning before the end of the vehicle's anticipated lifetime. On the other hand, map growth of more than 2% annually offers insignificant additional insight and therefore the requisite bandwidth spend to update maps may not be advisable.

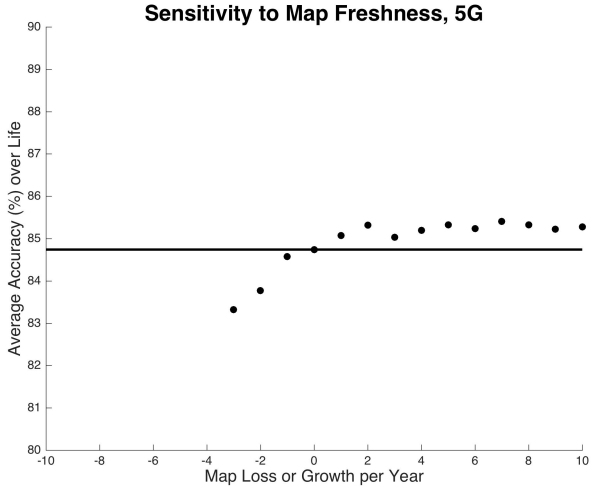


Figure 2-33: Lifetime average accuracy for idle time prediction with data loss, stagnation, or data set growth. Losses over 3% indicate that the application will cease functioning before the effective life of the vehicle (defined to be 12 years), while gains over 2% offer diminishing returns.

The impact of data freshness must be considered for the life of the vehicle, and developers must determine when to stop supporting an application. Figure 2-34 shows

data atrophy over time and how technology choice can make applications more or less robust to this decay.

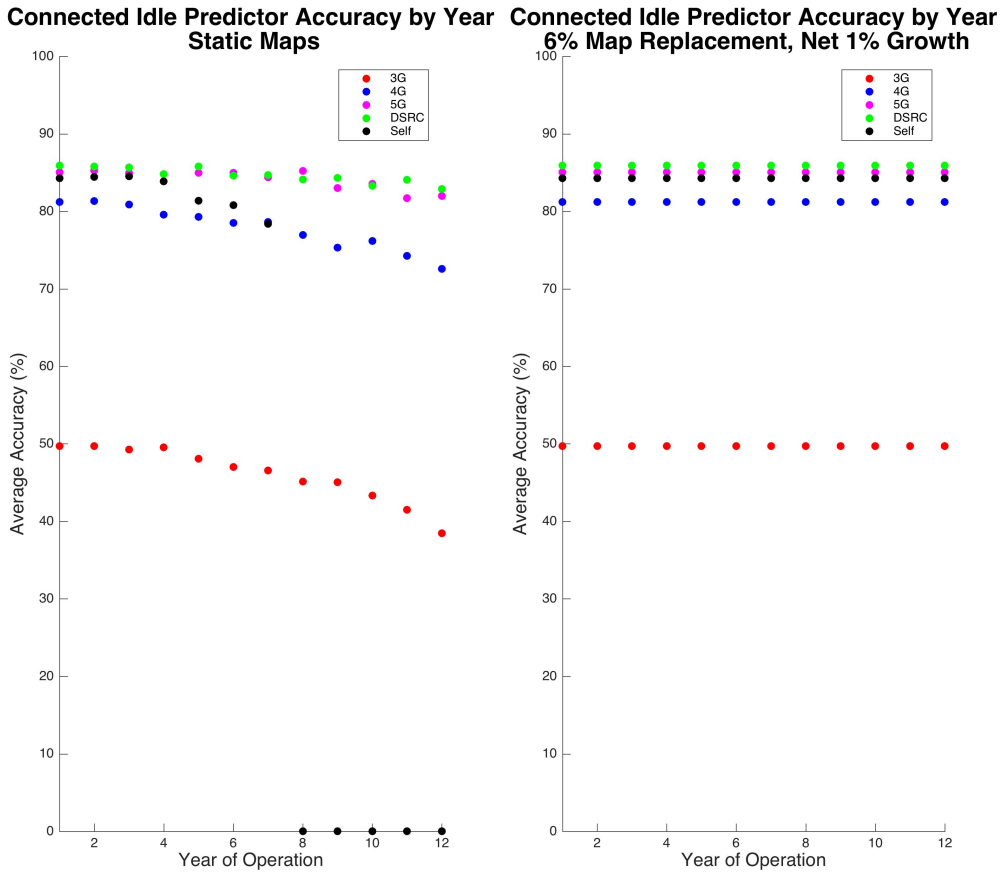


Figure 2-34: Databases lose value over time. Replenishment can keep applications running well for longer, at significant possible expense (depending on the chosen connectivity technology).

From this figure, it is possible to see that the map decay interacts with other factors, such as latency, to determine the overall accuracy. In this case, 3G and other slower technologies are more sensitive to data loss.

2.16.5.6 Car/Cloud Storage Split versus Bandwidth

The Cloud is ready to enable new applications by providing increased scalability and supporting rapid capacity growth, while novel radio technologies reduce latency and allow the Cloud to be used in realtime applications. On the other hand, remote servers

may go offline at unplanned times, and even low-latency cannot match the speed with which contemporary in-car computation is possible.

Some applications must have local storage, while others must have Cloud computation – these applications may require extremely low latency or large training data sets. In cases where the solution of whether or not to use Cloud storage is unclear, the decision about splitting application locality depends on resources already in the car, the anticipated life of the vehicle, and communication technology. Storage costs are relatively insignificant both for car and Cloud, and computation costs continue to fall. Rather than considering the cost of in-car or in-Cloud storage or computation, the impact of this architectural decision on radio bandwidth must instead be considered.

As an example of the impact a decision to split application operation between car and Cloud can have, the cost of data storage and transmission is calculated from the sum of the in-car and in-Cloud storage and bandwidth required for various database sizes and transmission volumes, shown in Figure 2-35. A similar figure can be created for the cost of computation, based on the application requirements.

2.17 Application and Technology Outlook

Data-validated simulation illustrates the impact of technology choice, data availability, and architecture on idle time predictor accuracy and cost of implementation. Though this document explored the design factors for a single application, the findings are extensible and generalizable toward other application types. In designing any Connected Vehicle application, a similar process should be followed. First, the application should be evaluated for technical feasibility and lead candidates identified. Then, the hypothesis should be validated and tested with different input data sets. Finally, a cost model and objective function must be created so that the objective function may be optimized for cost. This process is shown in Figure 2-36, though the true process is iterative rather than linear.

Following this type of framework will help developers create efficient applications capable of improving vehicle design and use.

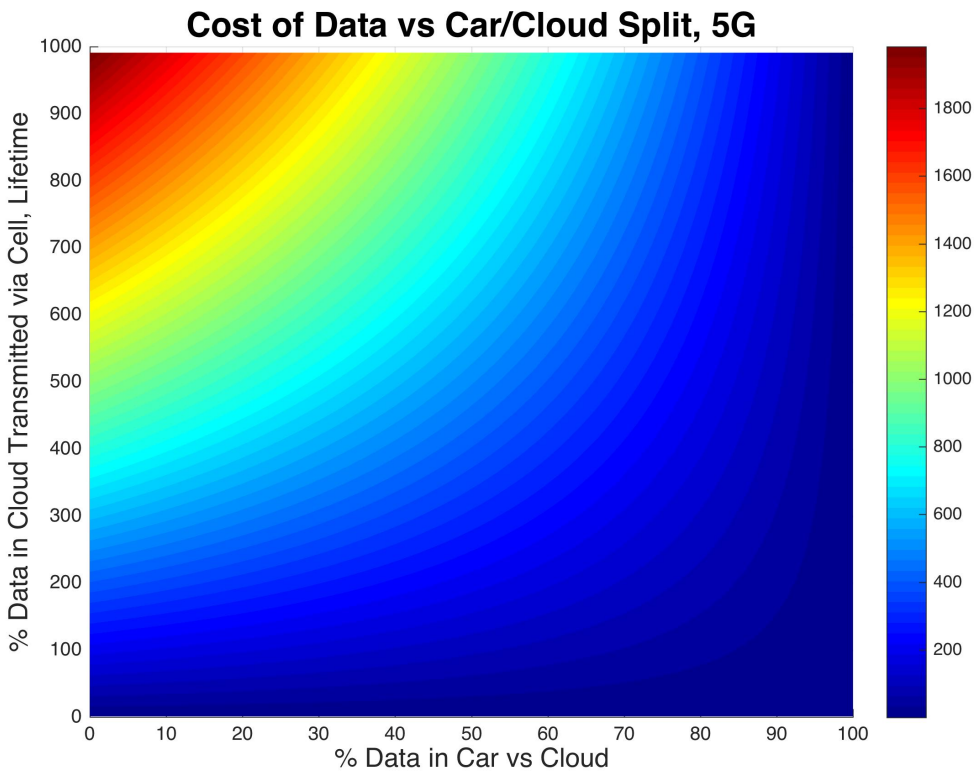


Figure 2-35: This figure shows the cost of bandwidth and storage for 5G radio at the current (February 2016) estimated prices. The costs depend on the car/Cloud data storage split and the amount of data transmitted from car to Cloud.

In this section, we explored the feasible communication, computation, and storage technology landscape for Connected Vehicles and introduced a number of novel concepts. We proposed the creation of a richer drive cycle called the "Gap Cycle" and showed how such a cycle may be used to improve vehicle comfort and efficiency in the use phase. As developers begin to make fuller use of vehicle-generated data and connectivity, the way we use and think of vehicles is sure to change. Concepts like the Gap Cycle and data-informed design will help engineers build cars that serve people better than ever and enable new transportation modalities including mobility as a service, Vehicle Miles Traveled (VMT) taxation, and Pay-As-You-Drive (PAYD) insurance. Understanding the richness of automotive data and design considerations for application development will help automotive engineers of the future to truly reinvent the wheel.

Relation to Other Chapters

Chapter 2 presented a new connectivity architecture implementing the Data Proxies and Cognitive Layers to reduce resource requirements and improve security in Connected Vehicles and other networked systems. It went on to additionally consider the other factors governing application design, development, and deployment, and highlighted the role of bandwidth, computation, and storage in determining the true cost of deploying an application. Chapter 3 continues on to discuss how the architecture and design considerations identified in Chapter 2 might be applied to improve the CloudThink digital object duplication system, enabling the creation of improved "Avacar" representations of vehicles. This chapter continues on to show how applications might be facilitated through the use of such an architecture, and how the proposed improved CloudThink platform addresses the platform, privacy, and security concerns identified in Chapter 1. Additionally, the theory and design process posed in Chapter 2 also facilitate and provide guidance supporting the creation of the efficiency- and reliability-improving applications designed and tested in Chapter 4.

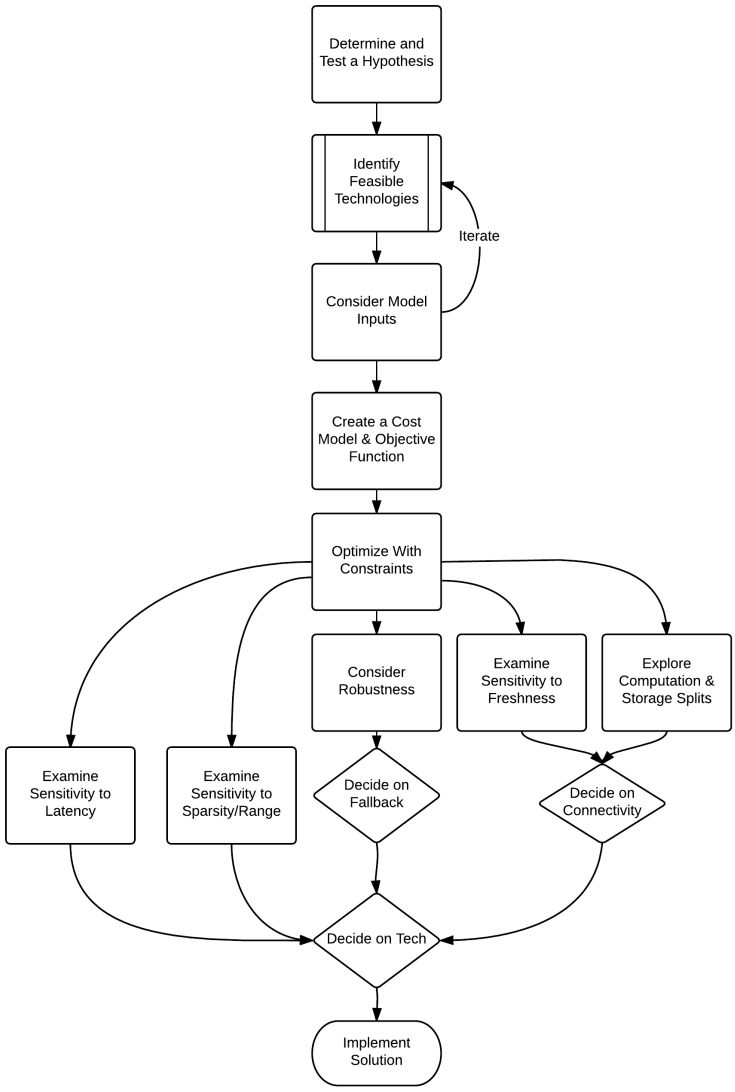


Figure 2-36: This flow chart shows a process framework for determining optimal application locality. It includes testing hypotheses, technology feasibility analysis, and inspection of sensitivity to bandwidth, latency, and other architectural considerations.

Chapter 3

The Present and Future CloudThink Platform

This chapter draws material from two accepted papers to which I was a contributor: "CloudThink: A Scalable Secure Platform for Mirroring Transportation Systems in the Cloud" [98][1] and "Conversations with Connected Cars" [159].[2] In this chapter, these papers have been edited with the first authors' permission to better reflect my contribution and expanded to newly integrate the theory posed in Chapter 2 into the platform embodiment. I recognize the significant contributions of Erik Wilhelm, Simon Mayer, Sanjay Sarma, and the other CloudThink platform codevelopers and coauthors for their involvement in creating and supporting the platform from the earliest revisions and onward. I am grateful for their support and help in making the CloudThink platform a reality.

For additional discussion of the origins and implementation of CloudThink, please refer to my Bachelor's [208] and Master's degree [99] theses.

[1]This chapter is derived in part from an article published in *Transport*, 2 Oct 2015 , available online: `http://www.tandfonline.com/` 10.3846/16484142.2015.1079237

[2]In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of MIT's products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to `http://www.ieee.org/publications_standards/publications/rights/rights_link.html` to learn how to obtain a License from RightsLink.

3.1 The CloudThink Platform

Physical objects are increasingly networked, virtualized, and projected into the Cloud. Despite a proliferation of connectivity, the best possible Internet of Things remains unrealized due to a lack of effective platform standardization and unified semantics. While Internet Protocols exist, these standards are based on connectivity for virtual systems rather than systems mirroring physical objects. Particularly in the automotive domain, the limitations imposed by repurposing this existing architecture is clear: telematics systems have evolved in a fragmented manner, while unclear privacy, security, and data ownership challenges have led to restrictions on platform utility. Widely-available telematics systems like OnStar limit data access and offer only minimal actuation, while information storage is siloed and hardware is manufacturer-locked, preventing widespread aggregation and rapid data collection. Many of these issues are discussed in depth in Chapter 1.

Despite these challenges, a car projected into the Cloud and made accessible for third-party application development offers great potential and should be explored. Due to their frequent use and human touch-points, connecting vehicles provides developers an opportunity to improve people's quality of life. Interfacing vehicles with other networks, such as Intelligent Transportation Systems, facilitates the creation of novel applications with significant potential for impact.

A key enabler to these applications is a brand-agnostic platform capable of simplifying data acquisition and manipulation. Such a system faces the challenge of supporting many modes of data acquisition and actuation while tending to the needs of consumers for security and privacy, as highlighted in consumer sentiment surveys. [8]

This section explores the development of CloudThink, an Internet of Things platform that starts as transportation-specific but which may grow to include data mirroring for other people, objects, and services. CloudThink enables applications using data generated by networked automobiles and mobile phone sensors capable of reducing vehicular energy use, operating cost, and environmental impact while improving driver safety and comfort.

CloudThink lowers the barrier to entry for accessing vehicle data and modulating vehicle actuators, as well as gives researchers access to rich data sets allowing them to address important questions. Aggregated fleet sensing helps to understand human mobility [209], improve road models [210], monitor traffic [211], and improve vehicle design [212]. These same data may be used to provide adaptive insurance and road pricing [213] and driving style feedback [214], or even monitor vehicle failure rates [215]. The platform will additionally act as a data broker, ensuring information security and customer privacy, for which telematics platforms have a real need. [216]

3.1.1 System Overview

While purpose-built telematics applications exist, such as those to identify open parking spaces or translate diagnostic trouble-codes into human readable format, scalable platforms are rarer and more valuable. To ensure success, CloudThink requires rapid and widespread adoption.

To that end, CloudThink provides an API and hardware designed to be easy to use, low cost, secure, and extensible. The use of an HTTPS RESTful API [217] lends itself to simple application development as well as rapid porting of existing solutions, while tools facilitating a clear view of data sharing and privacy controls improve user perception.

A low cost data collection device called the Carduino efficiently and securely transmits vehicle data to the CloudThink server, projecting virtual vehicles into the Cloud. Information is moved to a data server, from which applications can make use of vehicle information and feed processed information to a user interface running on any web-enabled device. Pre-aggregating these data in the Cloud allows their re-use for multiple applications and increases utility. An overview of the system appears in Figure 3-1.

The platform has five main differentiating factors relative to existing telematics systems:

1. **Open Hardware** CloudThink can be extended to work with any hardware,

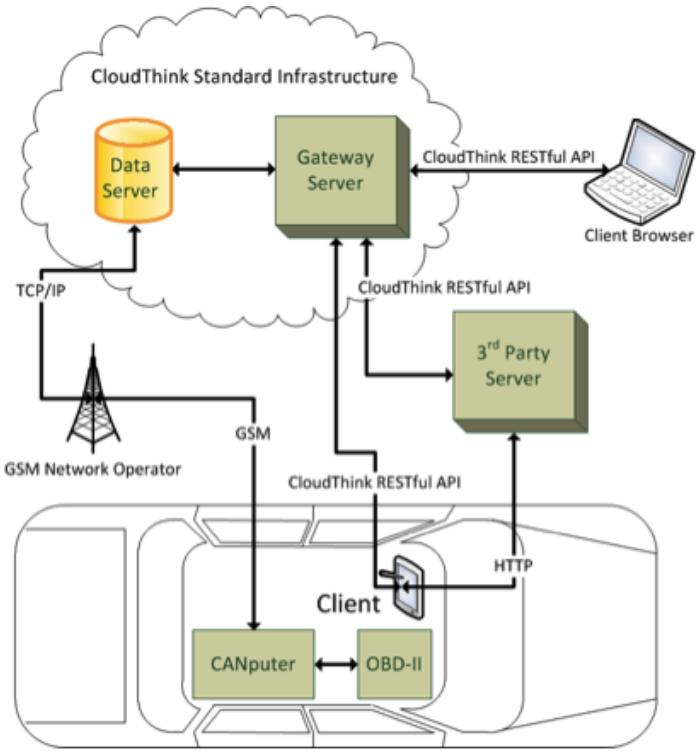


Figure 3-1: CloudThink system architecture, showing the Carduino and CloudThink infrastructure interacting with a third-party server. Figure ©2015 Taylor & Francis.

including the co-developed, open-source Carduino, mobile phones, and data captured by an ELM327 OBDII interface. The Carduino is designed to work with CloudThink and to mirror not just standardized diagnostic data, but manufacturers' raw Controller Area Network (CAN) data as well.

2. **Open API** This API simplifies access to vehicle data and provides open-source code to allow for rapid extension and scaling. Standardizing the type and format of data accessible through the API and creating a mechanism for applications to request new raw and preprocessed vehicle data eliminates the need for developers to rework applications for different end-use hardware.

3. **Vehicle-Mirroring Avacars** CloudThink makes use of the concept of the "Avacar," a digital duplicate of a physical vehicle in the Cloud. Avacars are used as "hooks" for application input and output to allow web-based applications to engage with virtualized vehicles.

4. **Translation-in-the-Cloud** The CloudThink platform works with any CAN-enabled vehicle. As cars rely on different in-vehicle networks with unique platform architectures, CloudThink may be extended to incorporate "translation in the Cloud," allowing the creation of one application capable of accessing proprietary data on any make or model vehicle in the translation database.

5. **Privacy and Security Controls** CloudThink allows users full control over their data with per-application permission setting. Through a web dashboard, users can select services, review stored vehicle data, and grant or revoke access rights to applications.

These five factors are discussed in depth in the following sections.

3.1.2 Open Hardware

The "Carduino" provides a means of universal, wide-area vehicle data access and control. This requires more than simplifying access to "conventional" diagnostic data, such as OBD PIDs and DTCs – the device additionally provides information that conventional scanners are unable to read, such as manufacturer specific messages or physical layer Controller Area Network (CAN) data.

The Carduino interface shown in Figure 3-2 is a low cost and flexible interface to capture vehicle CAN signals for a range of applications. The device utilizes the standard OBDII data found in all vehicles sold in the US and Europe after 1996, and connects vehicle networks to the Cloud via a 2G GSM cellular connection. The hardware additionally provides flash memory storage for data buffering and over-the-air updates, a GPS unit for positioning, and an accelerometer and gyroscope for enhanced motion capture. Configured data are transmitted from the device to the CloudThink server at regular intervals.

The core functionality of the unit is to serve as a vehicle to Cloud interface, bridging engine and transmission data from the diagnostic port of a vehicle directly a secure database. While some of the acquired data are similar to those captured by diagnostic devices using Bluetooth and WiFi, the short range of these devices and reliance on a

Figure 3-2: CarKnow's V8 Carduino device which connects vehicle CAN networks to the CloudThink API. Figure ©2015 Taylor & Francis.

secondary data connection for backhaul limit utility. The Carduino avoids this pitfall by reporting directly to the CloudThink backend using an integrated cellular chipset.

After initial configuration, the Carduino turns a vehicle into a self-reporting sensor capable of sharing OBDII messages and non-OBD CAN data. The device additionally may receive commands from the CloudThink server, sending the appropriate CAN packet to vehicle actuators.

The Carduino is user-friendly, reliable, and secure. There are no external buttons and there are clear status indicator lamps. A Power-On Self Test (POST) routine diagnoses and reports faults, while an onboard bootloader can recover corrupted firmware. User data is protected throughout transmission from car to Cloud with SSL encryption, and the hardware itself resists vehicle sensor spoofing by checking the in-car network for telltale signals that are difficult to replicate. The device supports flexible data payloads and can be configured to minimize data transmission rate to save bandwidth cost. The device also minimizes its impact on vehicle use – while diagnostic tools are not intended to be left in vehicles and can excessively burden a battery, the Carduino intelligently determines when the driver is done using the vehicle and turns off key features to save power. Reducing user interactions reduces barriers to entry, allowing for a broader installation base.

Additional details describing the hardware are available in CloudThink's *Transport* article [98].

A future revision of the hardware is already underway. This hardware transitions to an STM32F4 microcontroller to provide computational and storage overhead for encryption, local data aggregation, and estimator-based fault detection. Additionally, a secondary communications channel making use of Bluetooth Low Energy is used, allowing the same device to connect to the Cloud cellularly or to share a mobile phone's data connection. Such a configuration has the potential to reduce bandwidth costs.

3.1.3 Open API

While this section discusses vehicle mirroring, the same approach could be applied to mirroring any other object or service. A mobile device transmits encrypted information to the Data Server via a cellular network, which stores information in an SQL-based database containing a table for every vehicle in the system. The car-to-Cloud message format is a custom string of ASCII text designed to minimize overhead and therefore bandwidth. These messages contain a timestamp, type header, data payload, and checksum. In the case where data-aggregation happens on the Carduino itself, a message description may be sent as a separate message type to allow the server to process thew new data type. Upon receipt, the Data Server checks all messages and parses the information from valid messages for storage into the car's database table.

A Gateway Server moderates data access based on application credentials (similar to the Security Layer depicted in Chapter 2), and provides a REST API making use of SSL-encrypted connections and HTTP authentication. This server additionally creates a thesaurus mapping OBDII data identifiers to human-readable names. From the API, information may be retrieved in JavaScript Object Notation (JSON), Extensible Markup Language (XML), or human readable types, and these data can be navigated using the human-facing web interface or via HTTP POST and GET requests. Web documentation is provided to simplify application development by describing data types, access endpoints, and more.

To improve the accuracy of data in the API, server-side synchronization is necessary. The CloudThink data server assigns timestamps to the data points in a cleaning step: for each connection, it finds the earliest point with a valid UTC timestamp from the GPS. If no point is found, synchronization fails for all the data transmitted in this block. If multiple times are detected, the server takes the earliest time as absolute and interpolates the others relatively using the transmitted millisecond clock to determine differential times. All information is grouped into a one-second picture of vehicle state, unless additional resolution is requested by an application. This state includes processed and non-OBD information not available from direct sensing, such

as instantaneous fuel consumption and distance driven.

In the future, a Data Proxy estimator may be used to provide temporal interpolation for these more-granular state representations, improving utility without increasing resource use.

3.1.4 Avacars

Avacars mirror vehicle parameters in the Cloud for use in application development. These Avacars interact with other Avacars, Internet-mirrored objects, and connected systems to enable novel applications to improve vehicle efficiency, reliability, and comfort.

The concept of the Avacar as a digital duplicate for a physical vehicle is not new [99]. However, there is a unique application of the Data Proxy concept and theory posed in Chapter 2 to improving the Avacar. While Avacars have already been implemented with realtime object mirroring, there have been no attempts to intelligently minimize data sampling, address issues with timestamp synchronization, or correct sensor outages. Applying observer- and estimator-derived Data Proxies to Avacars has the potential to mitigate the issue of corrupted and missing data, and these Proxies may be applied to improving the quality of an Avacar mirror while lowering resource use requirements. With lower resource use, data transmitted can be reduced and money saved – alternatively, more parameters may be collected or the state estimate improved for the same data volume used by contemporary applications.

In defining the sampling payload for an Avacar, the Application Agent may be applied to identifying the optimal sensor sampling rate and arrangement for an ever-changing series of applications. Finally, the Avacar relates to the Data Proxy setup in that approximated data will be reported with error bounds, so that appropriate Quality of Data may be maintained for end-use applications.

3.1.5 Translation-in-the-Cloud

CloudThink and the Carduino access all of the Controller Area Network data streams within vehicles, from legislated OBD parameters to manufacturer proprietary broadcast messages and custom diagnostic parameter identifiers. Historically, this access has challenged development for applications requiring complex data streams or the ability to actuate vehicle components. To address the divergence of intra-automotive network architectures, CloudThink aims to create a common set of semantics for enhanced vehicle data streams. This is accomplished through the use of a community-driven, reverse engineered database similar to an automotive platform Wiki. Users will use the Carduino to identify parameters and commands transmitted within their vehicles, and submit these data streams to a common database. This mapping database will be integrated into the platform so that a common set of vehicle data semantics may be created.

Making data access transparent simplifies application development and allows developers to focus on innovation, rather than coping with non-standardization and having to redevelop secure connected systems. The use of translation-in-the-Cloud further facilitates flexible application development. Providing functional semantic metadata, as described in [218], encodes within an object representation the functionality of the CloudThink API for accessing vehicle data. An example in Listing 3.1 highlights how a semantic description of the API enables clients to obtain the latitude of a particular vehicle from its VIN.

The CloudThink API contains similar descriptors for each parameter that may be accessed from a connected vehicle. The use of a Semantic Reasoner simplifies integration of functionality across web devices and services. These descriptions have been used to create composite services mashing up CloudThink data with other platforms for the purpose of mapping and other applications. The Semantic Reasoner processes a semantic goal, such as "obtaining an image that is also a map that is derived from a specific VIN and making use of a particular back-end," and addresses this goal however feasible (see Figure 3-3). The reasoner's architecture is described

Listing 3.1: Part of the RESTdesc description of the CloudThink API that describes how the current GPS latitude of a car can be obtained given its VIN, using an HTTP request to the CloudThink server. The server provides similar descriptions for other information about the cars for which it has data.

```
@prefix :<ex#>.
@prefix ex:<http://example.org/#>.
@prefix http:<http://www.w3.org/2011/http#>.
@prefix dbpedia:<http://dbpedia.org/resource/>.

{
 ?car a dbpedia:Car;
       ex:hasVin ?vin.
}
=>
{
 _:request http:methodName "GET";
          http:requestURI ("https://api.cloud-think.
          com/things/" ?vin "/?parameters=latitude");
          http:resp [ http:body ?gpslat ].

 ?gpslat a dbpedia:Latitude;
         ex:derivedFrom ?car.
}.
```

Figure 3-3: Semantic metadata allows flexible integration of CloudThink API data into local and remote web services. Here, a user requests a map with the location of their vehicle. A reasoner finds the appropriate requests to reach this goal and executes them. Figure ©2015 IEEE.

in [218].

Because the Semantic Reasoner works to achieve a goal and is not hard-coded, a semantic-enabled application may seamlessly switch to alternative approaches when

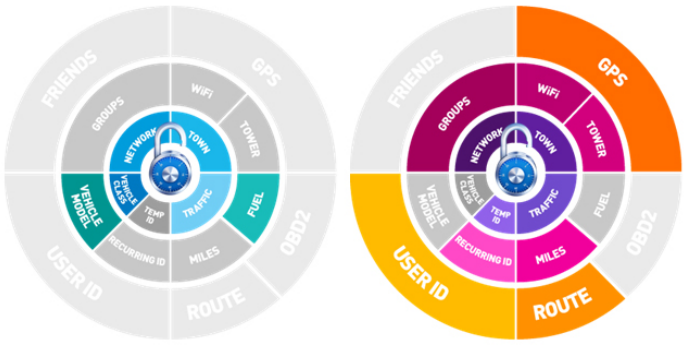


Figure 3-4: The Privacy Radar clearly shows application permission requests, so that users may monitor where their information is used – and for what purpose. Figure ©2015 IEEE.

services become available or unfavorable. It acts like a dynamic glue layer, intelligently connecting devices, people, and services.

3.1.6 Privacy and Security Controls

Several challenges to vehicle networking revolve around the issues of privacy and security. Some solutions to address these problems are simple, such as the use of end-to-end encryption and application credential validation. Data tracking is similarly straightforward, with the Gateway Server recording all data requests and access to facilitate internal billing and to provide a traceable record for drivers. Other aspects of data ownership and privacy assurance, however, require more innovative solutions. CloudThink's present and future solutions to these problems are described below.

3.1.6.1 Privacy Radar

Data ownership and clear application permissions are important considerations to ensure that platforms succeed. In Chapter 1, data security was identified as a primary concern relating to Connected Car platforms, and CloudThink extensively logs information to provide a clear view to users of how much data of each type an application requires. To visualize this information, CloudThink offers a "Privacy Radar" (Figure 3-4) showing commonly-requested information and providing a front-end for drivers explaining what data certain applications require.

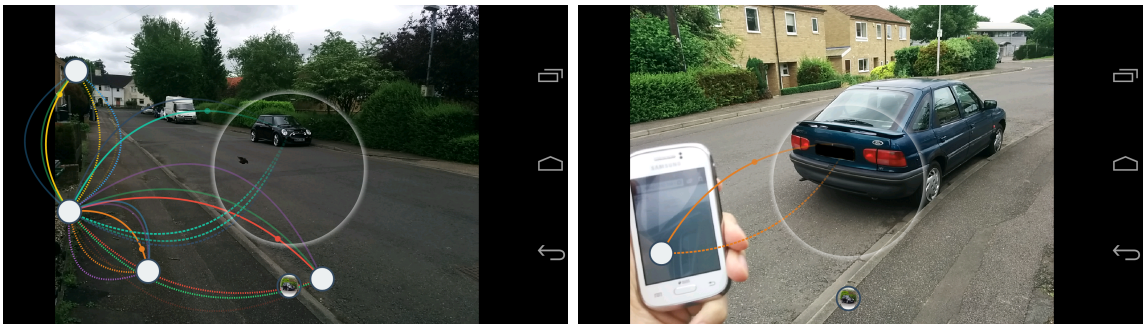
The radar helps drivers track application permissions over time, and offers drivers an easy "valve" to shut off data flow.

3.1.6.2 Magic Lens

Semantic Reasoning can be extended to visually keep users informed of the information their vehicle is sharing, and with what services or devices. This can be challenging when applications are developed flexibly, and might call upon a number of different platforms or services to achieve a goal depending on network loading or other factors [219].

Drivers prefer clear representations of where their information is going, as well as the ability to control interactions. Supervision can help drivers protect against attacks on their privacy like location tracking, or avoid malicious actuation, or even help stem the flow of personal or identifying information to authorities. [220].

Because CloudThink logs all HTTP requests, it is possible to create a tool enabling the use of augmented reality to visualize data flows in realtime [219]. Simon Mayer modified the Java server code and combined it with visual object-recognition code to allow the overlay of information on mobile devices, creating a realtime display of connectivity. This is shown in Figure 3-5.



(a) Interactions of a client with a vehicle. (b) A mobile device interacting with a vehicle.

Figure 3-5: A mobile application visualizes interactions with a vehicle connected to the CloudThink platform. Figure ©2015 IEEE.

This visualization tool can be enhanced to allow human flow control – such as swiping across a connection to sever it and remove an application's permissions – in real time.

3.1.6.3 Application of Cognitive Layer

As the CloudThink platform allows remote actuation in addition to data collection, proper firewalling is necessary. Avacars mirror real vehicles, and the interaction of applications with digitally duplicated vehicles proves a level of abstraction making direct control more difficult. This is similar to the Security Layer proposed in Chapter 2. Additionally incorporating the Cognitive Layer proposed in Chapter 2 into the CloudThink platform would allow commands to be validated prior to being sent to a vehicle, ensuring that improper commands are never passed on to connected devices. Running a Cognitive Firewall in the Cloud affords scalable computational resources for credential validation and virtual command execution that would be untenable on the embedded hardware found in the Carduino. For example, the Cognitive Firewall allows an additional level of safety when actuating critical vehicle components relative to today's approach, enabling the use of "safety critical" functions like door locking or remote starting. The use of the Cognitive Supervisor could enable next-generation On-Board Diagnostics running in the Cloud by applying process supervision and sending notifications when the system approaches an undesirable state, even dynamically increasing sampling through the Application Agent as the system nears a defined or learned limit.

Such application of "cognitive enhancements" would be critically important when dealing with safety-critical systems, such as drive-by-wire, or even remote locking and unlocking. In this way, Cloud security can be used to secure a connected system interfacing with the decidedly insecure Controller Area Network found in vehicles. [130]

3.1.6.4 In-Unit Protection

Failing in-Cloud protection, the Carduino itself provides security. Commands must be checked against an updatable whitelist of approved commands, and the hardware cannot send CAN messages with a header ID lower than a specified value (CAN is dominant-low, so sending a low ID command could force a bus-off scenario). Commands with known-malicious content can be banned using an in-device blacklist, while

remote control is only accepted in response to a device-outbound connection targeted at a particular IP address. This means the server, rather than the device, would need to be compromised. Finally, PIN validation may be implemented in the future to ensure that commands sent from the server have been targeted for a particular connected device, so that mass-actuation attempts do not impact the entire connected fleet.

3.2 Implications of Data-Proxy Enabled CloudThink

This chapter discussed the CloudThink digital object duplication platform, and how the concept of a Data Proxy and Cognitive Layer might be applied to "Avacars" and connected vehicle applications. These improvements will allow the CloudThink platform to continue to scale and enable novel application archetypes by lowering bandwidth costs and improving actuator security. Additionally discussed was the application of the Privacy Radar and Magic Lens to provide users insight into the storage and sharing of their generated data, addressing the needs for enhanced privacy policies and visualization tools.

What remains to be addressed is how best to interact with other platforms. While semantic reasoners help to dynamically fuse smart things and platforms to one another, the idea of "WikIoT," a community-run set of dynamically-deployable IoT glue layers, would be a helpful extension enabling new flexible mashups. WikIoT would be structured like an API for API API's, storing system metadata and semantic information in a common format and location. The creation of such a standardized service platform would have the added benefit of being able to apply its centrality to addressing the big remaining issues in connected platform development of device registration, enumeration, localization, and interaction. Issues of platform management, agnosticism, and extensibility must be explored before this type of integration platform is tenable. Further, a set of common design vocabulary or semantics must be defined. An "IoT Esperanto," the minimum language describing any thought (software update), need (data), or desire (actuation) for connectivity, would help to speed IoT

unification. This vocabulary and an extensible IoT platform aggregator similar to WikIoT may become a topic for future research.

Relation to Other Chapters

CloudThink will be enhanced through the use of Data Proxies, Cognitive Layers, privacy visualization tools, and Semantic Reasoners. This updated incarnation will become an attractive backdrop for future application development, and the implementation of "Cognition" will allow resource-efficient implementations of prognostic applications as well as secure vehicular actuation. The use of Proxies will further enhance resource-efficiency and allow for the creation of improved Avacar mirrors for a fixed amount of data input, enabling applications requiring broader data inputs or improved timeliness of data. Addressing the needs identified in Chapter 1, Chapter 4 continues on to examine several previously untenable, user-facing application types that could be supported by an updated CloudThink platform enriched by new theory.

Chapter 4

Applied Automotive Analytics

Chapter 4 considers how the theory from Chapter 2 and the platform proposed in Chapter 3 may be used to enable applications that would otherwise not be possible. For example, the Cognitive Supervisor may be applied to identify the characteristics used by a classifier based on Avacar input data, or the size of samples transmitted may be reduced by applying the Data Proxy's model to interpolate costly signals. This type of application might need to make use of "Cloudsourced" data for data mining, or could require rich sensing results that require rapid sampling or data transmission too quick to be technically-feasible. The intent with these applications is to demonstrate the combined use of mobile phone data, On-Board Diagnostics data, and information from external to inform applications that help vehicle owners and operators.

As identified in Chapter 1, there is a need for consumer-facing applications capable of improving user experience and comfort while reducing vehicle cost of ownership. I additionally conducted a survey to determine how significant an impact this type of application could make: from a sample of 15 self-reporting vehicle owners on Amazon's Mechanical Turk (`http://www.mechanicalturk.com`), I determined that the average driver leaves a check engine light on for nearly 3,500 miles, allowing minor issues to increase in severity and resulting in an average $467 repair bill (the "cost of inaction").

To address this significant opportunity, I proposed creating a suite of rapid diagnostic and prognostic applications. These applications would explore a number of commonly-ignored vehicle maintenance regimens and frequent faults in an attempt

to create a suite of applications detecting faults prior to catastrophic failure. The end-goal was to shift vehicle repair from reactionary to anticipatory, making use of automotive sensors and pervasive mobile phone technology to build the "Mechanic's Tricorder" for state-of-health diagnosis. Rapid sensing would allow vehicle repair to shift from reactive to proactive, saving time and money while reducing the risk of unplanned vehicle downtime.

The chapter discusses the development of four vehicle prognostic applications, including engine oil viscosity measurement (to suggest optimal change intervals), wheel balance supervision, tire pressure and tread monitoring, and engine misfire detection. This section is based off of two published and two submitted papers, "Vehicular Engine Oil Service Life Characterization Using On-Board Diagnostic (OBD) Sensor Data" [55] and "Smartphone-Based Wheel Imbalance Detection" [53], and "Smartphone-Based Vehicular Tire Pressure and Condition Monitoring" [54] and "Engine Misfire Detection With Pervasive Mobile Audio" [56], respectively.[1] I thank my co-authors Rahul Bhattacharyya, Ajay Deshpande, Sumeet Kumar, Isaac Ehrenberg, and Sanjay Sarma for their invaluable contributions to the successful development of these prognostic applications.

4.1 Engine Oil Viscosity Characterization using On-Board Diagnostic Data

This portion of the chapter is based off of "Vehicular engine oil service life characterization using On-Board Diagnostic (OBD) sensor data," from IEEE Sensors 2014 [55].

Standardized On-Board Diagnostics (OBD) systems include a range of sensing and reactive diagnostic services [27]. Recently, there have been efforts to apply these sensor data to predict likely faults. A consumer demand for vehicle data and a shift

[1] In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of MIT's products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to `http://www.ieee.org/publications_standards/publications/rights/rights_link.html` to learn how to obtain a License from RightsLink.

from in-vehicle to telematic diagnostics has led to the use of rule-based approaches and data-mining to identify anomalous trends or conditions indicative of vehicular problems [221]. This section examines how standardized On-Board Diagnostics data may be processed and fused with non-legislated data along with information from third-party sources to gain insight into variables beyond those typically measured. In particular, the application of engine and ambient temperature data to infer relative status of oil viscosity is discussed. This section proposes a model for temperature-based viscosity inference demonstrating repeatability for one vehicle across a range of environmental conditions.

4.1.1 The Need for and Enablers of Oil Monitoring

Proper engine lubrication is critical to the safe and efficient operation of internal combustion engines. Oil must be monitored because it degrades with use and time, picking up dirt and burning off lubricating additives. Heating, churning, and contamination thickens oil and increases drag on rotating assemblies. This drag and resultant heat generation impacts engine performance, fuel economy, and longevity. When oil becomes sufficiently thick, it may restrict flow to small clearance engine passages and lead the engine toward thermal runaway. Observing proper change intervals reduces the likelihood of catastrophic failure and keeps combustion engines operating well, minimizing the cost of vehicle ownership and use.

There exist sensors to measure viscosity directly, such as micro-acoustic sensors [222] [223]. Other sensors rely on electrical, chemical, and optical techniques to directly measure and characterize fluid parameters [224]. Direct viscosity measurement is feasible, but cost and complexity limit deployment [225]. Instead of sensing, many automotive OEMs elect to use proprietary algorithms counting engine revolutions or otherwise quantifying wear as a function of engine load, vehicle speed and operational time. The use of such open-loop algorithms can cause issues by over- or under-estimating oil change frequency due to wide variations in oil characteristics. Manufacturers tend to err on the side of caution and suggest changing frequently, meaning serviceable oil is discarded, contributing to environmental issues

and increased operating cost.

All modern vehicles have at least some sensing capability that may be drawn upon to infer oil viscosity. The California Air Resource Board and Environmental Protection Agency oversee On-Board Diagnostics II (OBD II), a system built into all model year 1996 and newer vehicles sold in the United States. OBDII provides realtime information from emissions-related sensors, such as engine speed, coolant temperature, fuel trim, and timing advance, as well as"freeze frame" data and trouble codes [27]. A parallel set of standards called EOBD exists in 2001 and newer vehicles sold in Europe [226].

OBD does not require the measurement of any oil parameters, so this set of standards is not directly useful to quantifying remaining lubricant life. Some of these parameters, however, may inform applications about the broader operating state of the vehicle. Oil, as an example, manifests in several instrumented aspects of engine operation, such as engine speed and coolant temperature. Observing these parameters may allow indirect inference of the oil viscosity.

4.1.2 Rate of Temperature Change as a Viscosity Surrogate

As oil thermally degrades and becomes contaminated, effective viscosity increases. Heightened viscosity increases friction and heat generation due to increased shear stresses and viscous heating, as oil is forced to flow through constant-width pathways between the pistons and cylinder walls. Though steady-state operating temperature is regulated with the use of a thermostat, the transient temperature evolution provides insight into engine operation. An OBD-standardized coolant temperature sensor may be used to measure engine temperature, and because temperature changes are driven largely by friction, and friction driven by oil's lubricating efficacy, this coolant temperature may include insights relating to oil viscosity.

An engine filled with new oil will have low friction and heat up slowly relative to an engine with oil near end-of-life, with its increased friction. The derivative of the engine coolant temperature (T_{cool}) with respect to time, i.e. $\frac{dT_{cool}}{dt}$, will thus increase as oil wears out. We hypothesize that this indirect measurement tracks with viscosity,

and may therefore be monitored over time to close the loop in oil supervision. This relationship will allow the creation of models to suggest optimal oil change intervals, minimizing waste, cost, and service downtime.

4.1.3 Data Acquisition Systems

To test the hypothesis, two logging systems were used for data collection. The first relied on CloudThink's direct-to-Cloud system, while a second made use of mobile phones as intermediate devices. In both cases, temperature data was recorded from OBD, along with GPS data for later service mashup to integrate temperature information. CloudThink and flexible service mashups are discussed in Chapter 3.

4.1.4 Viscosity Data Collection

An initial experiment was conducted, eliminating environmental factors such as driver behavior, ambient temperature, and forced airflow. Tests were performed on a 2.0 liter turbocharged 2011 Buick Regal, using DEXOS-approved 5W-30 synthetic oil as recommended by the manufacturer. A second experiment was conducted without environmental control using a 2003 Toyota Camry. The second experiment allowed us to examine factors such as engine loading on the engine oil warmup profile.

4.1.4.1 Controlled Warmup

The test procedure for controlled data collection included logging intake air temperature at 0.5Hz and engine coolant temperature at 1Hz. The engine was started cold (at least two hours since last start), and logging began when the key was turned and ambient temperature was recorded using the GPS coordinates and data from Intellicast [2]. The engine was allowed to idle without interruption up to 185F, at which point the thermostat opened and logging was stopped. Allowing the vehicle to idle eliminated complicating factors such as engine loading and forced airflow. Data were

[2] `http://www.intellicast.com/`

logged on several days, to provide a rich data set capable of generating subsets with similar starting ambient and coolant temperatures for comparison.

Old oil with 9491 miles was tested for several weeks, before the oil was replaced with new oil. The new oil was driven for 10 miles to allow complete circulation, at which point new oil was logged for several weeks.

4.1.4.2 Unconstrained Driving

A second vehicle was tested with unconstrained driving including variable engine loading and forced airflow (the vehicle was driven during warmup). Motion imbued the data with biases that would test model robustness. The parameters logged and rates were identical to the first case, with additional parameters recorded to identify engine loading, engine speed, and vehicle speed.

4.1.5 Viscosity Inference Results

Preliminary results shows that for the controlled driving under similar environmental conditions, the rate of coolant temperature change was an appropriate surrogate for viscosity measurement. Unconstrained driving was unable to provide sufficient data to prove model efficacy.

4.1.5.1 Controlled Warmup

Figure 4-1 shows a plot of engine coolant temperature versus time for both new and old oil, aligned such that all data start from the same coolant temperature and windowed to avoid edge-effects near the point at which the thermostat opens. In the plotted range, new oil has a shallowed slope (reduced $\frac{dT_{cool}}{dt}$) relative to the old oil. This is repeatable across different ambient starting temperatures and reflects the oil's varied viscosity and related shear stresses.

Another way of visualizing the same result considers the warmup time taken to warm up from 65 °C to 80 °C, a temperature determined to be below the opening point of the vehicle's thermostat. A table of these results appears below:

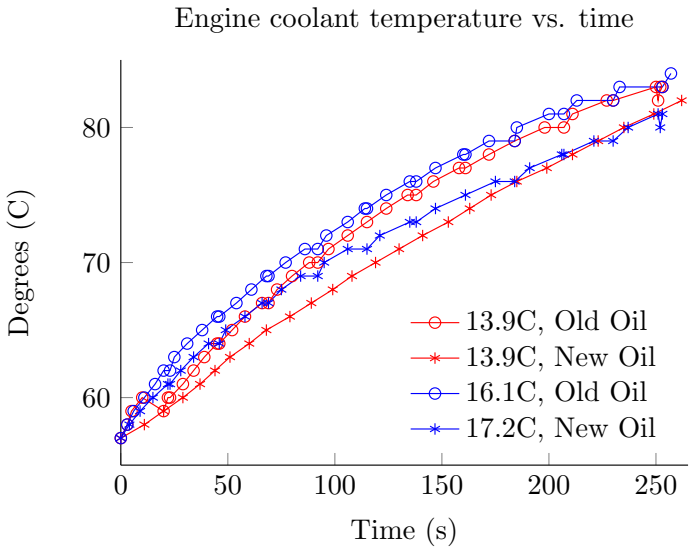


Figure 4-1: A plot showing coolant temperature versus idle time for new and old motor oil at two constant ambient temperatures, highlighting the difference in heat generation for varied oil viscosity. Because the sensor reports with resolution 1C, only switch points (an increase or decrease of 1C) are plotted. Figure ©2014 IEEE.

.

This table shows a minimal temperature dependance when ambient starting temperatures are similar. There is a clear difference in heat generation between old and new oil, with the older oil heating more rapidly.

4.1.5.2 Unconstrained Driving

These data revealed less predictability in the rate of heat generation. Visual inspection indicated a strong relationship between engine loading and engine temperature sufficiently strong that in cases new oil was seen to heat up more rapidly than old oil. This result highlighted the need for correction factors in any model relying solely on temperature to characterize oil life.

4.1.6 Findings for OBD-Fused Oil Sensing

From the controlled warmup results, one sees that $\frac{dT_{cool}}{dt}$ is an appropriate surrogate metric for oil viscosity and thus may be extended to model remaining oil life when vehicles are stationary during warmup. We have demonstrated the use of an existing,

Sample	T_{amb} °C	**t: 60-65**	**65-70**	**70-75**	**75-80**	t_{total}	(°C/s)
Old - 1	12.2	44.61	43.6	51.7	73.0	212.9	0.094
Old - 2	13.9	42.59	36.5	47.7	64.9	191.6	0.104
Old - 3	16.1	28.25	39.1	47.1	61.1	175.5	0.114
New - 1	7.2	84.15	101.2	108.2	128.2	421.8	0.047
New - 2	8.9	79.14	106.2	111.2	136.2	432.8	0.046
New - 3	13.9	78.03	102.1	108.1	124.1	412.2	0.049
New - 4	17.2	34.07	46.1	66.1	78.1	224.4	0.089
New - 5	21.1	39.07	48.1	65.1	82.2	234.5	0.085

Table 4.1: Table showing controlled warmup time-to-temperature and ambient starting temperature for a single vehicle with new (approx. 10 mile) and old (approx. 9500 mile) oil. Table ©2014 IEEE.

networked sensor equipped in all OBDII and EOBD equipped vehicle to monitor engine temperature over time and as a result, the state of engine oil. This model is applicable to sensing vehicles as they warm up, e.g. to allow drivers to heat up the car or cool it down in summer. These analytics may be run in the background of a sensing system and transmitted to the user when the oil is nearing the end of its useful life.

With complete information, engine oil changes may be freed from fixed interval inspection. This allows optimal service timing, where the vehicle owner is informed and asked to change engine oil only when necessary thereby minimizing unnecessary costs and improving engine longevity. Users may also be persuaded to take individualized oil data more seriously if they are informed that there is science behind their custom-tailored recommendation, rather than following a generalized maintenance schedule.

4.1.7 Relating Oil-Sensing to Data Proxies and CloudThink

The oil-viscosity sensing application relies on On-Board Diagnostic data, but must sense ambient temperature as well. This can be done in at least two ways — first, by accessing the raw CAN access and manufacturer-proprietary signals as enabled by the Carduino in Chapter 3, or alternatively by making use of data fusion like the Semantic Reasoner or a platform like WikIoT, as discussed in Chapter 3 to facilitate

integration of temperature data from third-party sources.

While this oil-monitoring application was created prior to the development of the Data Proxy as posed in Chapter 2, it could easily be improved through the use of an estimator model. The Data Proxy's system estimate could be used to track changes in the time/temperature profile and identify when the system evolution is out of specifications. Therefore, the entire application could be reduced to a smart observer with defined (or machine learned) control limits to recommend optimal change intervals. This is a variant on the proposed Cognitive Supervisor.

The application of the Supervisor to determining when the system state is near limits is useful not only to reporting the need for maintenance, but also to informing adaptive sampling models. The Cognitive Supervisor may integrate into the Proxy's sampling model and increase sampling of the system as control limits are neared, thereby conserving battery life in day-to-day use.

The remaining applications make use of pervasively-sensed mobile phone data. Mobile sensors offer a rich sensing payload, connectivity, computation, and storage conducive to data collection, analysis, and transmission. Without local data processing and the enabling theory of the Data Proxy, the following prognostic applications would not be feasible due to their extreme data transmission requirements. Local processing and the use of an estimator as a data minimization technique will make such applications tractable.

4.2 Pervasively-Sensed Wheel and Tire Monitoring

Proper wheel and tire maintenance is key to the safe and efficient operation of vehicles. Without frequent inspection, pressure can drop and tread can wear down, increasing drag and reducing traction. Wheels can become damaged or balancing weights can separate, inducing harsh vibrations into vehicle cabins and further reducing traction. Despite the importance these rotating assemblies have to vehicle operation, many drivers neglect to inspect these components, leading to unnecessary increases in risk and preventable issues in comfort and efficiency. A solution to proactively and pas-

sively identify problems and suggest maintenance could reduce the risk of drivers operating unsafe or poorly maintained vehicles, save fuel, and improve occupant comfort.

Sensors found in mobile phones present new opportunities for vehicular sensing. This section explores the novel application of mobile sensing to fault detection in wheels, tires, and related suspension components in vehicles. These applications can be extended to take full advantage of Data Proxies and the CloudThink platform as enabling technologies by reducing the data transmission requirements and allowing fusion with other information sources.

4.2.1 Wheel Balance Detection

This section is based off of "Smartphone-Based Wheel Imbalance Detection" [53] and presents a supervised machine learning technique capable of in-situ wheel imbalance detection using transformed smartphone accelerometer data. Demonstrated is the use of a classification tree model consisting of Principal Component Analysis (PCA)-transformed Fourier features to achieve $> 90\%$ accuracy on testing data. The presence of an imbalance is detected reliably on two vehicles of differing make and model. Additional details are available in the paper published in the Proceedings of The Dynamic Systems and Controls Conference, 2015.

4.2.1.1 The Importance of Finding Balance

Rim and tire imbalance is a common problem in vehicles with impact ranging from minor annoyance to severe impact on vehicle function. Increased tread wear, ride harshness, and occupant discomfort are common results. For this reason, monitoring wheel assembly balance is key to the safe and comfortable operation of any wheeled vehicle. Unfortunately, wheel balance is only checked when a problem is suspected or at infrequent intervals as part of routine service. In-situ monitoring would allow drivers to identify potential problems before they become severe.

There have been several studies seeking to enable real-time in-situ monitoring via

Figure 4-2: Wheel weights are used to balance automotive wheel and tire assemblies to provide a smooth ride and enhanced traction.

.

direct instrumentation with accelerometers and other chassis-mounted sensors [227] [228]. These approaches have proven effective at detecting faults, though the process of instrumenting vehicles with aftermarket sensors is cumbersome. We examine the possibility of applying mobile phone sensors to the detection of wheel imbalance as a means of simplifying fault identification and allowing drivers to proactively address issues.

4.2.1.2 Sources of Imbalance

Vehicles leave the factory with balanced wheels. Small weights are attached to the wheel after tires are mounted to account for the tire's non-uniform mass distribution, shown in Figure 4-2. In use, however, wheels can easily fall out of balance. This can be due to a traumatic event, such as hitting a pothole, deterioration of adhesive, or even slip of the tire on the rim. High speeds and increasing vehicle mass can increase the likelihood that a "small" impact can cause severe, balance-impacting damage [229]. The energy in an impact is related to the kinetic energy, $KE = \frac{1}{2}mv^2$.

Just as the wheel assemblies are balanced at the factory, changes in balance can be addressed with the use of balancing weights [230], but the efficacy of these weights needs to be reassessed over time. If wheels are run out of balance for a long time, they can vibrate and cause wear on other suspension components [231] [232]. A severe imbalance can cause wheels to bounce, and even limit traction.

4.2.1.3 Previous Imbalance Detection Approaches

Wheel balancing is well established. For example, on-car balancing techniques involve lifting a car and rotating each wheel to check for imbalance, making use of weights attached to the wheel rim to counteract the unbalanced mass of that particular wheel [233]. The benefits are debatable since the vibration dynamics of loaded, rolling tires on roadways can be different from those of an unloaded tire lifted off the ground [234], though "road-force" balancing systems may be used to more accurately mimic real driving conditions and improve balance.

With the advent of Microelectromechanical Systems (MEMS) sensors, there has been an increased interest in on-road monitoring of wheel imbalances [235]. For example, Oblizajek *et. al* propose a system using accelerometers and wheel rotation data to measure and mitigate the transference of vibrations from wheels to different components in a car [234]. There have been several other instances of wheel imbalance detection in instrumented vehicles [236]. A demonstrated in-car system for monitoring tire balance relies on accelerometers to identify radial and lateral acceleration of the tire and compare amplitudes to a known good baseline at characteristic frequencies, in effect examining differences in the amplitude of the Fourier component of radial acceleration [237] [238]. A variation of this setup employs an accelerometer on each wheel to identify vibration, but cost and complexity of this system is significant [239]. Another application of accelerometer examines the use of wavelet transforms for improved classification, while a Ford study relied on the use of ABS sensors to detect imbalances and wheel hop, though access to these data is limited and requires low-level signal interfaces [228] [227].

Existing solutions work but require costly customized instrumentation. The use of commoditized hardware like a mobile phone would greatly increase deployability and lower instrumentation costs.

Figure 4-3: iPhone mounted in vertical orientation. The accelerometer directions are labeled. Figure ©2015 ASME.

.

4.2.1.4 Collecting Wheel Imbalance Data

The ubiquity and low-cost of smart phones makes their use as diagnostic tools appealing. Application development for smart phones is relatively simple, the devices are small, light, and easy to mount, and power options are abundant. In recent years, MEMS motion sensor resolution and sampling rates have greatly improved in mobile devices, making data capture increasingly useful [240].

To collect data, a mobile phone was mounted using an off-the-shelf mount to the windshield of a series of vehicles. The mount oriented the phone vertically (shown in Figure 4-3) and suspended it away from the windshield on a short arm.

Using the mount kept the phone aligned in a fixed orientation and eliminated the need to track the motion of the phone itself.

To gather imbalance data, an experiment was conducted using a 2015 Subaru Impreza and later repeated with a 2013 Nissan Versa with different tire sizes [241] [242]. The use of high-profile tires reduced the likelihood of a contaminated baseline, as such tires are resistant to pothole damage. The use of low-mileage vehicles made it likely that the factory balance remained accurate. An iPhone 5S sampled acceleration at

100Hz (for reliable feature identification up to 50Hz) and with GPS at 1Hz. The y-axis was examined for this application, as it was the axis most impacted by imbalance in the given phone orientation.

The vehicles were driven at 60MPH once using cruise control and once based on human control for each state. This speed was determined from the Ford paper's identified minimum perceptible velocity of 58MPH [227]. Several cases were tested, including unmodified baseline and a range of individual and sets of imbalanced wheels. Imbalance was induced by applying 28g of weights per wheel, which was less than half of what the Ford study deemed imperceptible. This ensured that classification would occur at or below the level of human sensitivity, and as such could be an early indicator that could help drivers identify early onset problems. Sampled data were captured for 10 minutes per state.

4.2.1.5 Imbalance Analysis

The accelerometer data were processed in three phases - pre-processing and noise reduction, spectral analysis, and machine learned classification.

4.2.1.5.1 Data Filtering The data had to be filtered to minimize contaminating effects such as pothole strikes and road surface variations. This is achieved by passing data through a high-pass filter, determined by examining environmental contributions and anticipated wheel rotational frequencies. Expansion joints spaced at 5 m appear at a periodic 5.36Hz when traveling 60MPH, while in the case of our study, assuming a tire circumference of 1993.34mm for the P195/65R15 tire size of the Subaru Impreza [3], we anticipated a tire rotational frequency of 13.4Hz (13.6 for the Nissan Versa's tire). The cutoff frequency was therefore set at 5Hz to minimize environmental measurement while providing a range of frequency measurements around the anticipated wheel velocity.

[3] `http://www.discounttire.com/dtcs/infoTireMath.do`

Frequency (Hz)	t-stat	Result (α=0.05)
14	7.8810 (d.o.f 118)	Significant peak
28	6.9773 (d.o.f 118)	Significant peak

Table 4.2: Student t-statistics for peak frequencies observed in the imbalanced condition Table ©2015 ASME.

4.2.1.5.2 Spectral Analysis The 10 minutes of accelerometer data were segmented into 120 sub-sets each containing 5s of data. A spectral representation of each segment is obtained through the use of the Fast Fourier Transform (FFT). Figure 4-4 identifies statistics from the spectral analysis of the 120 sub-sets for the normal and imbalanced cases. Visible are several energy contributions:

- *Common peaks*: Note a peak in the Fourier Transform at 10 Hz and another at 18-20 Hz. These occur in both the baseline and imbalanced cases.
- *Imbalance specific peaks*: There are visible peaks at 14 and 28 Hz for the imbalanced case which are statistically significant under the Student's t-test (see Table 4.2).

The subsets of data varied greatly, illustrated by a spread in average data. Therefore, we explored the use of supervised machine learning, and in particular, a decision tree algorithm to quantify the success of imbalance detection. Figure 4-5 presents a high level overview of the machine learning techniques, which is discussed in more depth in the original paper [53].

Common peak characterization The frequency peaks at 10 and 18-20Hz have several sources – from engine and transmission dynamics to phone mount resonance. These were identified to reduce the likelihood that energy contributions at these frequencies stem from imbalance.

Data were recorded with the engine on and the phone in the mount with the engine on and off. The peak at 20Hz appeared with the engine on but not with it off, indicating that interaction between the mount and the engine caused excitation at this frequency. This is shown in Figure 4-6.

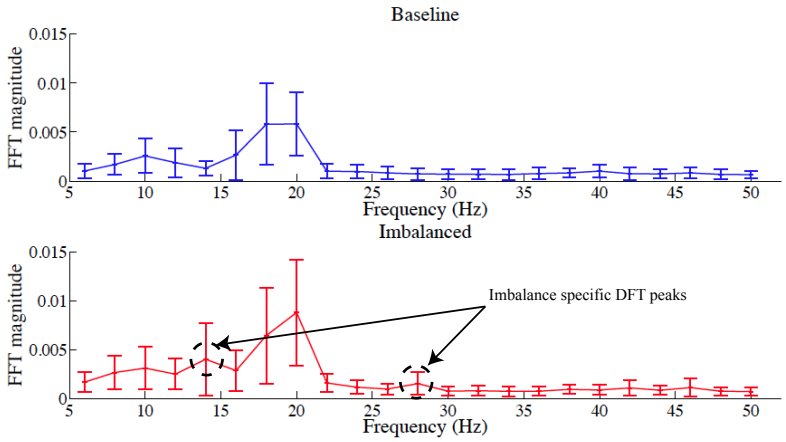


Figure 4-4: Spectral analysis statistics from the 120 sub-sets of data for both the baseline and imbalanced case. Each sub-set contains about 500 data points from a 100Hz sampling rate. DFT magnitudes are computed in 2 Hz buckets from 6 - 50 Hz and the mean and standard errors are plotted across the 120 subsets. Figure ©2015 ASME.

.

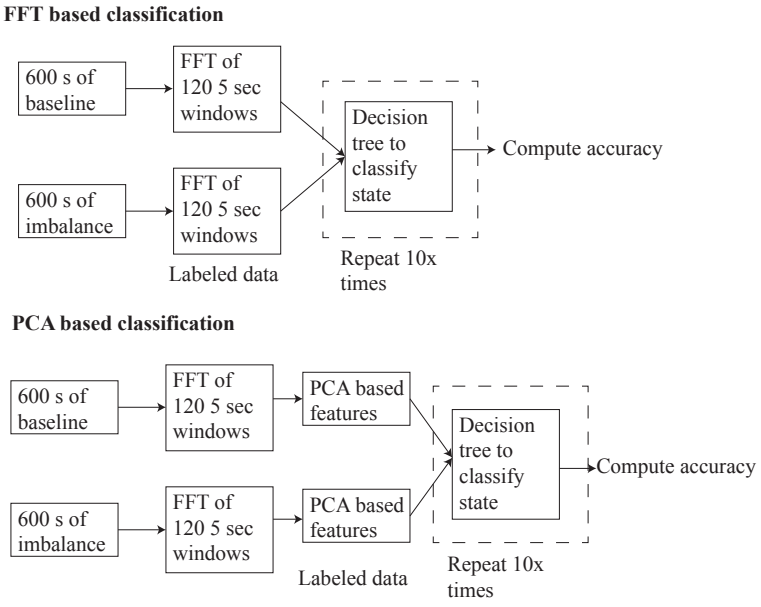


Figure 4-5: Supervised machine learning techniques that are implemented in this study. Figure ©2015 ASME.

.

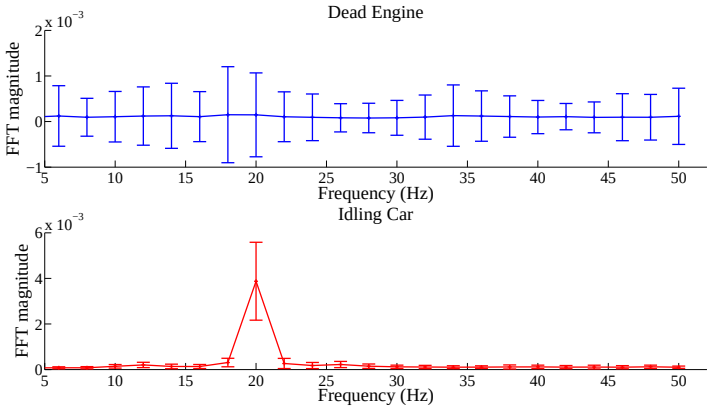


Figure 4-6: A plot comparing the Fourier Transform results for a vehicle with the engine off and the phone mounted with the results from idling with the engine on. The phone was installed in a windshield mount in both cases. Figure ©2015 ASME.

We attribute the 10 Hz frequency peak with the vehicle powertrain's rotating assemblies. The engine was eliminated based on its difference from the measured frequency; the transmission, however, rotates when the vehicle is in motion and contains several rotating elements at differing speeds. An experiment was conducted to determine the relationship between transmission engagement and the 10 Hz signal. Data were collected with the phone in the mount, the engine on, and the vehicle in park and in gear with the brake applied. In park, a pawl stops the transmission from rotating. In drive, the elements are free to move and a torque converter regulates final engagement of the elements.

With a cold idling engine, the Fourier Transform shows a 10Hz peak when in gear but not when in park (see Figure 4-7). With a hot engine, the peak is not present because the idle speed has dropped below the torque converter's stall speed (see Figure 4-8).

As final validation, the experiment was repeated to compare a hot engine with the transmission in gear at idle and brake applied to a hot engine with the transmission in gear and at elevated engine speed. The 10Hz peak is clearly visible in this high-RPM case, confirming the peak is due to transmission rotation. This is shown in (see

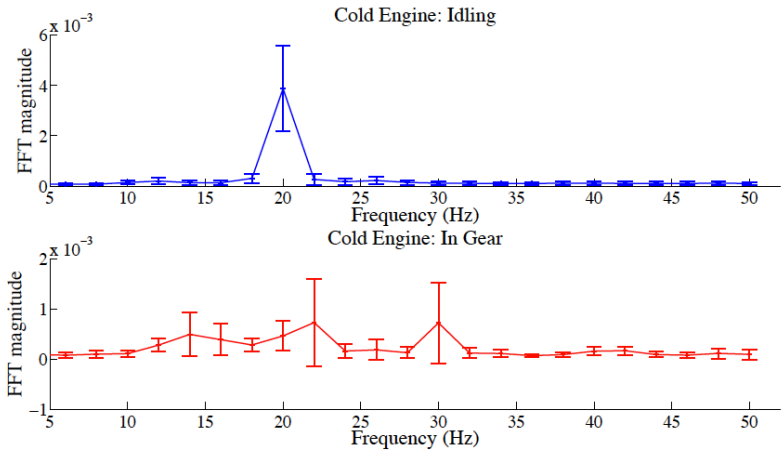


Figure 4-7: A plot showing a comparison of the Fourier Transform results for a vehicle with engine warming up from the cold state when in gear and when idling in park. The frequency shift from 10 Hz is due to the increased crankshaft speed to help warm up a cold engine. Figure ©2015 ASME.

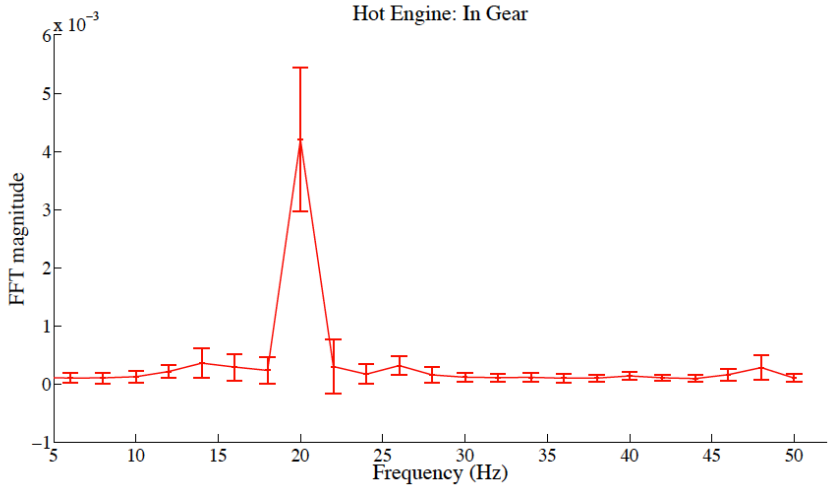


Figure 4-8: A plot showing the Fourier Transform results for a vehicle with a hot engine and when in gear. Figure ©2015 ASME.

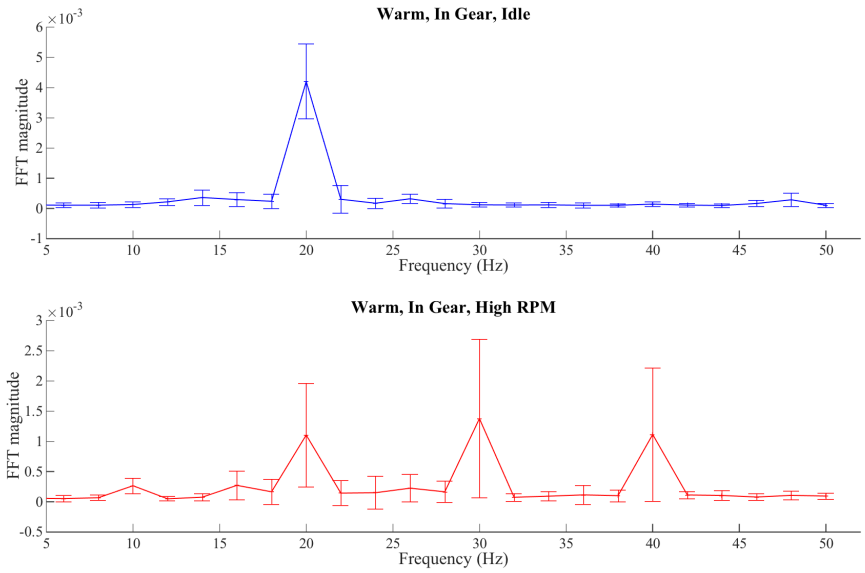


Figure 4-9: A plot comparing the Fourier results for a warm engine left in gear and with the brake applied for slow idle and elevated engine speed. In the elevated RPM case, the transmission's components engage and a 10 Hz peak appears. Figure ©2015 ASME.

Figure 4-9).

From this experimental assessment, we infer that the transmission's engagement is coupled with the 10Hz peak and this explains its presence in the motion studies.

4.2.1.5.3 Classification Using (PCA Transformed) Fourier Features Initially, we applied a decision tree to classifying the Fourier Features of the acceleration data. This process of selecting features, segmenting the dataset, and training/testing is shown in Figure 4-10 and Figure 4-11. This process resulted in an accuracy between 70-88% with a weighted average of 79.25%. The generated decision trees were complex (see Figure 4-12), and selected differentiating features outside the range of interest. Additional transformation was needed.

The FT features needed translation into more distinguishable features. Thus, the elements were projected into a 1×22 vector along the most significant PCA component direction to improve separability. This process of generating PCA-transformed Fourier features is shown in Figure 4-13 and Figure 4-14. There was a significant improvement in classification accuracy with a weighted average of 91.6%. The resulting decision

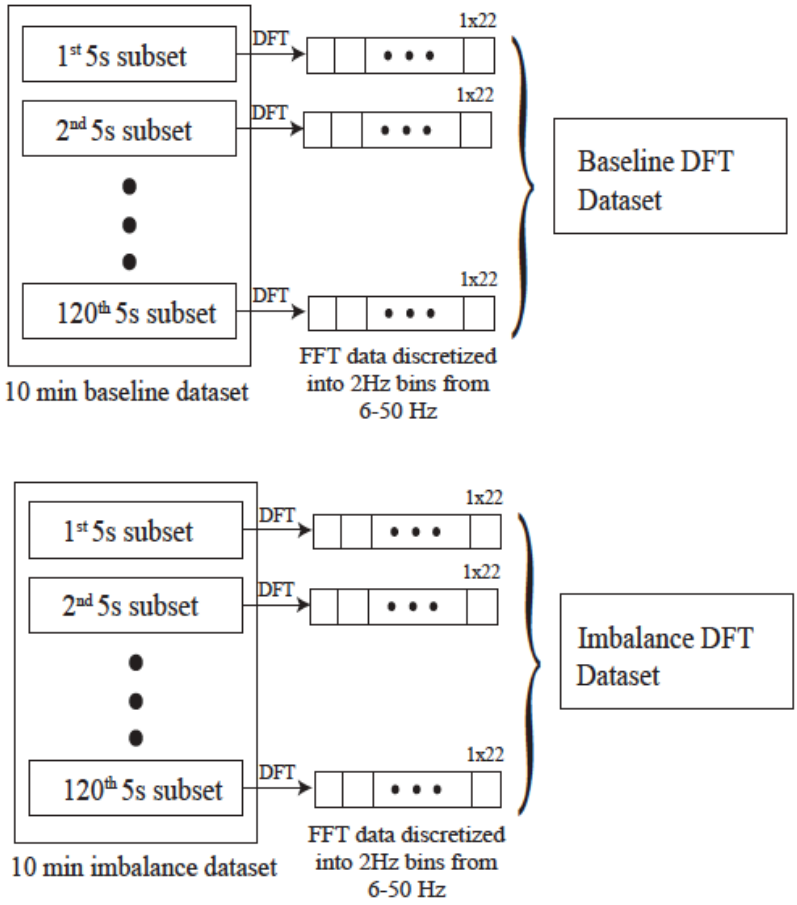


Figure 4-10: Generation of DFT transform from the baseline and imbalance datasets. Figure ©2015 ASME.

.

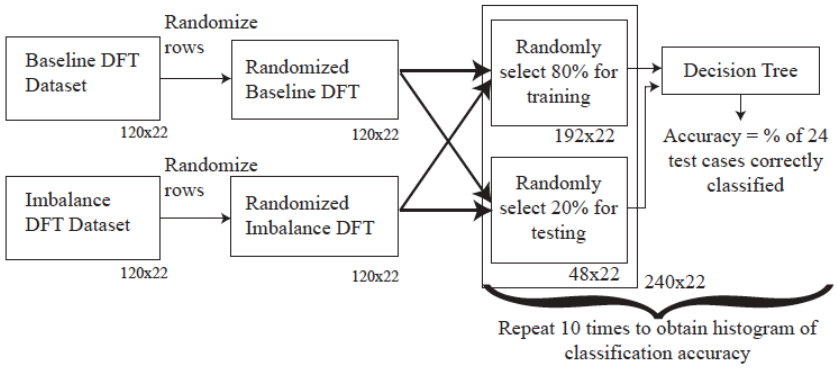


Figure 4-11: Using DFT features for decision tree training. Figure ©2015 ASME.

.

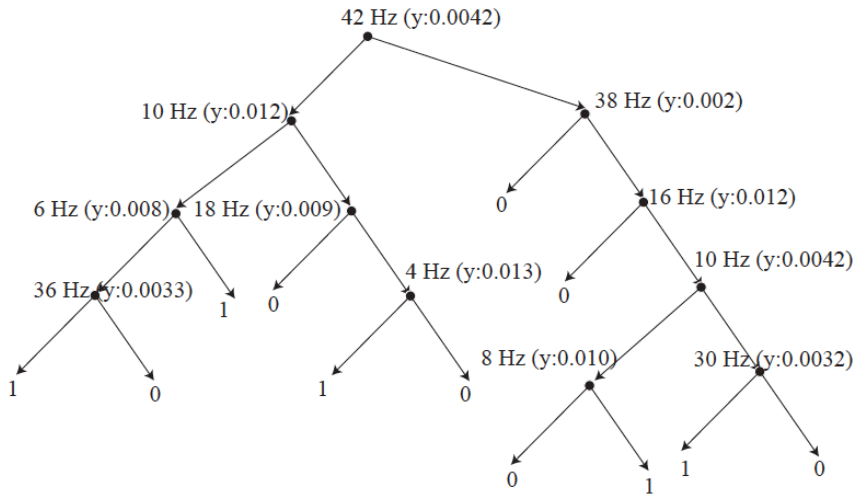


Figure 4-12: Decision tree generated using the FT data as features. If the FT magnitude at a node is < y, the left branch is chosen, or else the right branch is chosen. Figure ©2015 ASME.

.

tree, shown in Figure 4-15, was only one level (12Hz differentiation) and significantly simplified. From Figure 4-16, we see a peak at 12 Hz for the imbalanced case, and a corresponding trough in the baseline case. We therefore conclude that the PCA transformation of the FT data boosts the ability of the decision tree to detect the presence of a wheel imbalance.

4.2.1.6 Wheel Imbalance Conclusions

This work demonstrated the ability of mobile phone accelerometer data to be transformed in order to accurately classify the state of a vehicle's wheel balance. While Fourier features were not sufficiently accurate, transforming these features using Principal Component Analysis resulted in increased classification accuracy. This technique worked for two types of vehicles on differing road conditions, with low sensitivity to varied ground speed. Further, the insights gleaned from this application – such as identifying the onset location of an imbalance – may offer business opportunities for insurers and logistic companies to auto-process claims or identify areas in need of repair, or otherwise be used to inform a collaborative hazard map.

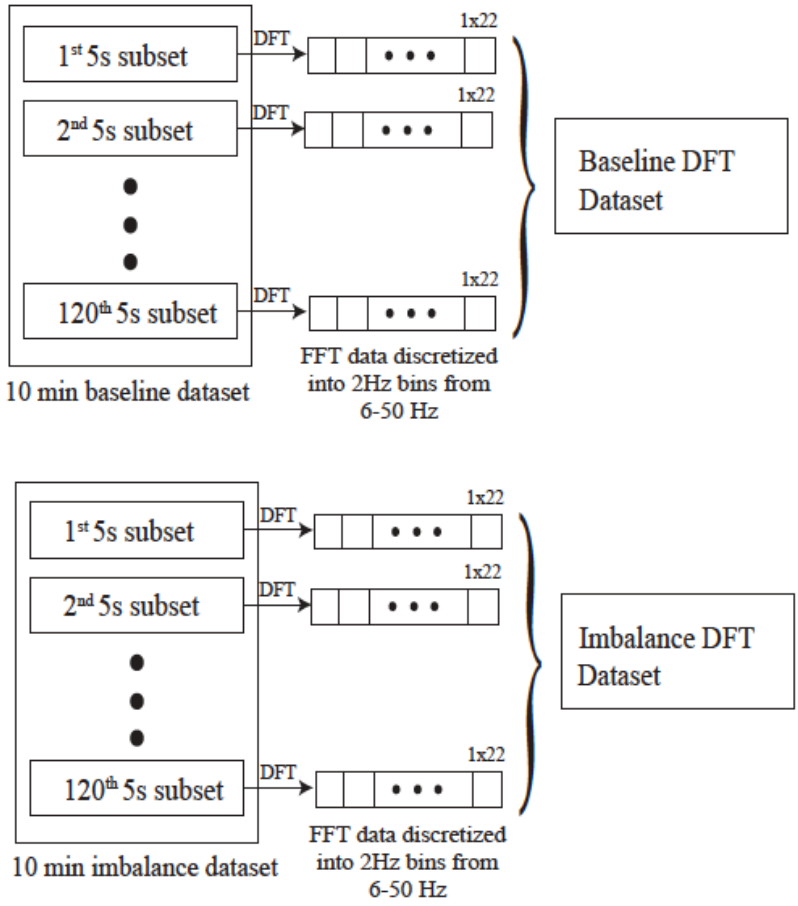


Figure 4-13: Generation of DFT transform from the baseline and imbalance datasets. Figure ©2015 ASME.

.

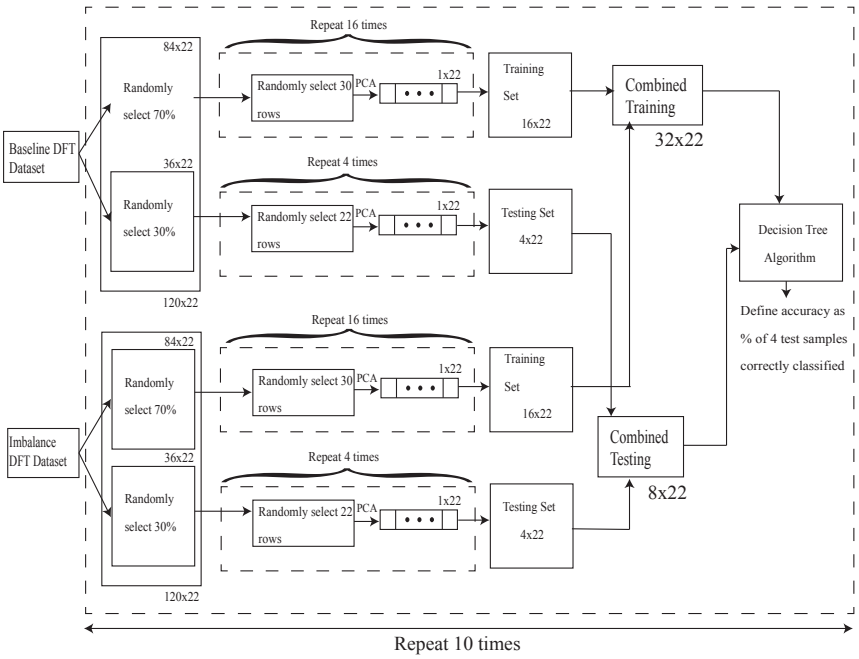


Figure 4-14: Using a PCA transformation of the FFT feature data for tree training and testing. Figure ©2015 ASME.

.

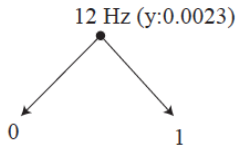


Figure 4-15: Decision tree generated using the PCA transformed FFT data. If the PCA transformed FFT magnitude at a node is $<$ y, the left branch is chosen, or else the right branch is chosen. Figure ©2015 ASME.

.

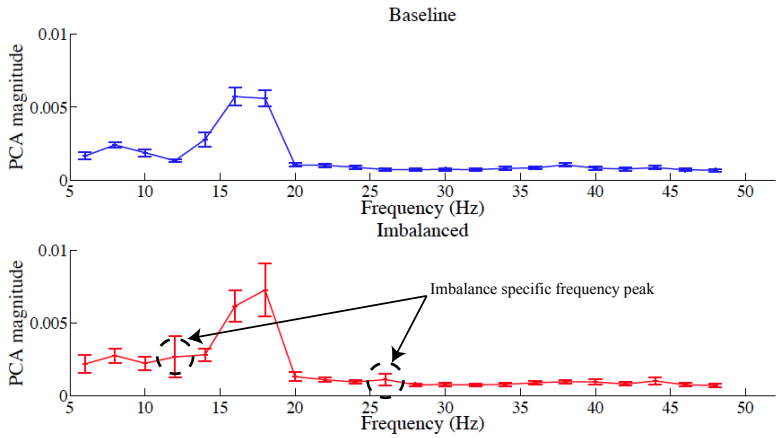


Figure 4-16: Spectral analysis statistics from the 120 sub-sets of data for both the baseline and imbalanced case using a PCA transformation of the FFT data. We choose the most significant PCA component in our analysis. Figure ©2015 ASME.

.

4.2.2 Tire Pressure Supervision

This section is based off of "Smartphone-Based Vehicular Tire Pressure and Condition Monitoring," which has been accepted for publication at SAI Intellisys 2016 [54].

While modern cars make use of Tire Pressure Monitoring Systems (TPMS), these systems may only identify differential changes in pressure and older vehicles lack sensing entirely. To address this problem, we applied mobile phone accelerometer and GPS data to calculate predicted and true wheel rotational frequencies, and therefore infer tire circumference and related pressure or tread depth. Tree-based classification allows us to identify from among five tire states with 20% increases or decreases in tire pressure with 80% accuracy.

4.2.3 Tire Pressure and Tread Supervision

While modern cars make use of Tire Pressure Monitoring Systems (TPMS), these systems may only identify differential changes in pressure and older vehicles lack sensing entirely. To address this problem, we applied mobile phone accelerometer and GPS data to calculate predicted and true wheel rotational frequencies, and therefore infer tire circumference and related pressure or tread depth. Tree-based classification

allows us to identify from among five tire states with 20% increases or decreases in tire pressure with 80% accuracy.

4.2.3.1 The Pressure To Monitor Tires

Poor tire tread depth can impact vehicle efficiency, safety, and comfort by limiting traction in adverse weather conditions [243]. Improper tire inflation increases rolling resistance and alters damping rate, leading to heightened fuel consumption, poor ride, and the potential for reduced traction [244]. Tread depth wears down over time, and though most drivers set initial tire pressures carefully, age, diffusion and punctures may cause this pressure to change and accelerate wear [244] [245].

Pressure non-compliance is a real issue, with a US government-run study reporting 12% of cars in the US with at least one tire under-inflated by 25% [246]. Associated costs are significant, with fuel consumed in these vehicles increased by 3.3% or more [247] [248].

Excessive tread wear is similarly common. The National Highway Traffic Safety Administration found that 50% of cars with at least one tire worn below half of the original tread depth. 10% had a tire with no remaining tread and severely compromised safety [249] [250].

Though tire condition is important, many drivers only inspect tires reactively. The ability to non-invasively detect this condition and inform drivers proactively has the ability to reduce vehicle collisions, improve efficiency, and limit unplanned downtime.

There is an opportunity to apply mobile sensing to proactively monitor tire pressure as well as tread depth, lowering the barrier to tire supervision and improving compliance.

4.2.3.2 Contemporary Pressure Monitoring

Beginning in 2006, Tire Pressure Monitoring Systems were required to be installed in all new vehicles sold in the United States [251]. Commonly available systems are factory installed and can detect relative pressure differences using MEMS sensors installed in the wheels or through data from Antilock Brake System sensors [252].

These systems may have short battery life (despite slow sensor polling intervals) or difficulty determining pressure when more than two tires differ from a target pressure. Improvements are batteryless, but these still require factory installation and such sensors can be inadvertently separated from the vehicle during service [253].

Previous approaches explored non-invasive solutions estimating tire radii via chassis-mounted accelerometers [254] [255]. Other models fuse vibration and wheel estimation via a slip-informed model [256]. Pressure measurement through vibration transmissibility has been considered, though this approach can have poor accuracy when using a single accelerometer to instrument multiple wheels [257]. A combination of approaches uses a particle filter model fusing ABS sensors and a GPS receiver to provide precise radius detection [258] .

These models rely on dedicated sensing or proprietary data streams and therefore have limited scope. A less invasive approach would enable sensing on older vehicles and facilitate realtime tire supervision. With successful deployment of such a technology, fuel consumption can be reduced and vehicle safety improved.

We settled on the use of mobile sensing, as mobile sensing has demonstrated advantages over in-vehicle systems [64]. Phones have rapid replacement and technological improvement relative to vehicle-locked technologies, with no cost of installation for applications.

4.2.3.3 Pressure Measurement Theory

We propose to observe tire pressure based on inferred rolling radius measurements. The effective radius of a tire depends on the tire construction, loading, pressure, and other factors. A change in any of these impacts the radius, altering the number of rotations required to travel a fixed distance. It is possible to control tires for non-pressure factors; measuring after tires are warmed up eliminates temperature effects, while identifying the uniformity of tire diameters across all four wheels differentiates pressure changes from wear-related changes.

From the formula $v = r * \omega$ where v is the velocity at the tire surface, r is the tire's effective rolling radius, and ω is the rotational frequency of the tire, one sees that if

	FL	*FR*	RL	RR
All Low	30	30	30	30
One Low	30	37	37	37
Normal	37	37	37	37
One High	42	37	37	37
All High	42	42	42	42

Table 4.3: Experiment 1 tested five sets of pressure states: all low, one low, normal, one high, and all high. All pressures are in PSI. Table ©2016 IEEE.

ω remains constant but r increases, v increases. The opposite relationship also holds true, with decreasing radius and decreasing linear velocity. Identifying the difference between angular and linear velocity holds the key to identifying small changes in tire diameter, and therefore changes in pressure.

Mobile phones are capable of measuring and comparing actual versus predicted wheel rotational speeds. The accelerometer and GPS provide two data points as input to this measurement. These inputs include Fourier-transformed acceleration data to identify true wheel rotational frequency and GPS, which provides a location history that may be used to calculate the true ground speed. We observed that the ratio between the anticipated vehicle speed and the true vehicle speed may be a useful predictor of effective tire circumference, and therefore radius and pressure.

4.2.3.4 Tire Inflation Experimental Setup

A set of experiments was designed to generate training and testing data for classifying tires inflated above, below, or to specifications. The mobile phone was mounted similarly to the wheel imbalance detection case, shown in Figure 4-3. The vehicle was warmed up to avoid temperature/pressure change effects, and then it was driven on a straight, smooth road at a constant 100 kph based on the GPS velocity to eliminate pressure-based variance. An application recorded 100Hz acceleration data and 1Hz GPS data for training and testing. Five experiments were conducted in the first car, which was driven in the following states: all tires low, front-left low, all normal, front-left high, all high. The pressure values for each state are visible in Table 4.3.

The front left was chosen as the single wheel to vary as it both steers and is driven,

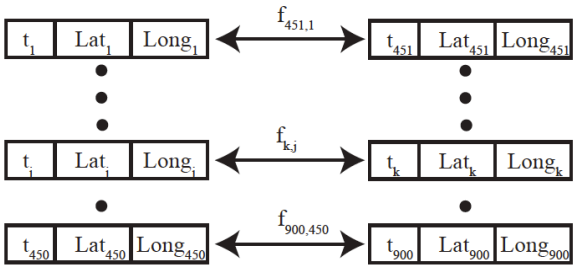


Figure 4-17: Division of GPS dataset for f_{GPS} computations. Figure ©2016 IEEE.

rending proper inflation most critical.

4.2.3.5 Pressure Data Pre-Processing

Data were filtered in two ways, with a time averaging and range bounding. We first time-averaged GPS data over a window to discard samples with poor position tracking or velocities outside a target range from 92 to 110 kph. We next bounded the Fourier-transformed results to consider main peak amplitudes in the range of 12.5 and 14.5Hz to identify the frequency associated with wheel rotational frequency. Samples were finally normalized against the Fourier-informed velocity to enhance the visibility of the ratio trend and remove spurious data points and outliers. Note that the identification of this region of interest may be used in conjunction with resource-efficient, windowed Fourier transformation algorithms to minimize computation's impact on battery life.

4.2.3.6 Pressure Data Processing

The main characteristic differentiating high and low pressure states is the ratio of $\frac{f_{GPS}}{f_{Acc}}$. This section explores how GPS and accelerometer data are transformed to support this theory.

4.2.3.6.1 GPS Frequency Prediction In 15 minutes, a 1Hz GPS sampling rate provided 900 time-stamped latitude and longitude estimates. We split this dataset in half and used corresponding rows between the two halves to get 450 estimates of wheel rotational frequency as shown in Figure 4-17.

$f_{k,j}$ may be estimated by Eq. 4.1, where Lat_m and $Long_m$ are the location coordi-

Case	$f_{mean}(Hz)$	$f_{std}(Hz)$
All low	12.84	0.18
FL low	13.31	0.07
Normal	12.86	0.26
FL high	13.43	0.14
All high	13.43	0.06

Table 4.4: Statistics of GPS predicted wheel rotational frequency. Table ©2016 IEEE.

nates at time t_m. H_d is the Haversine distance computed using two pairs of latitude and longitude coordinates and D is the nominal diameter of the car tire when unloaded. H_d assumes the vectorial distance between these points, which is appropriate because the highway on which data were collected was straight.

$$f_{j,k} = \frac{H_d(Lat_k - Lat_j, Long_k - Long_j)}{2\pi D(t_k - t_j)} \tag{4.1}$$

Table 4.4 shows the mean predicted tire rotational frequency and standard deviation for all five test cases. As tire pressure increases, the mean predicted rotational frequency tends to decrease. The difference in rotational frequency between the *All Low* and *All High* case statistically significant ($tStat_{95\%} = 69.9$) indicating that these two states can be differentiated using the ratio metric alone. The other cases, with their higher standard errors and closely spaced means, require further input to accurately and reliably classify.

4.2.3.6.2 Accelerometer Frequency Prediction For each of the five test cases, the 15 minutes of collected data were divided into five second windows, providing us with 180 windows based on a 100Hz sampling rate. Next, we generated a Fourier Transform (FT) of the accelerometer data on each of these 180 windows. Finally, we obtained an average Fourier Transform and the standard deviation per frequency component. The process is illustrated in Figure 4-18 and the average Fourier Transformed spectrum for the *FL Low* test case is illustrated in Figure 4-19 as an example.

We observe that although the FT peak corresponding to a tire rotational frequency of 13.6 Hz does manifest in the x,y and z axes spectrum, there is significant standard

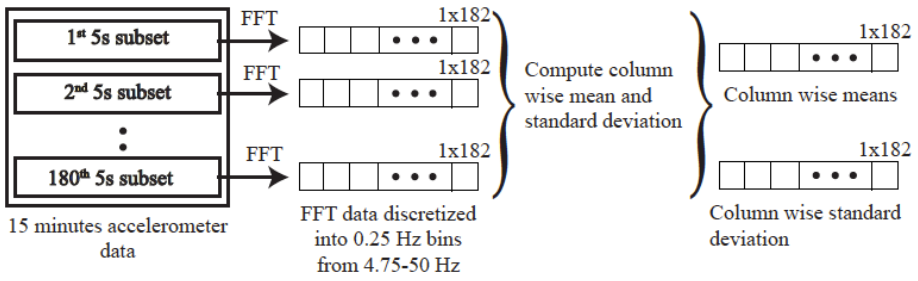


Figure 4-18: Algorithm for computing average FT spectrum. Figure ©2016 IEEE.

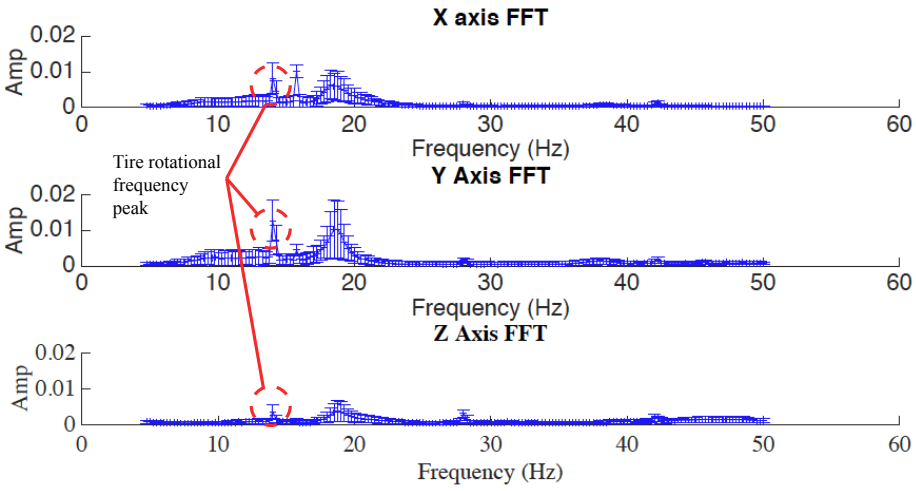


Figure 4-19: X,Y and Z axes FT spectrum for FL Low test case. Figure ©2016 IEEE.

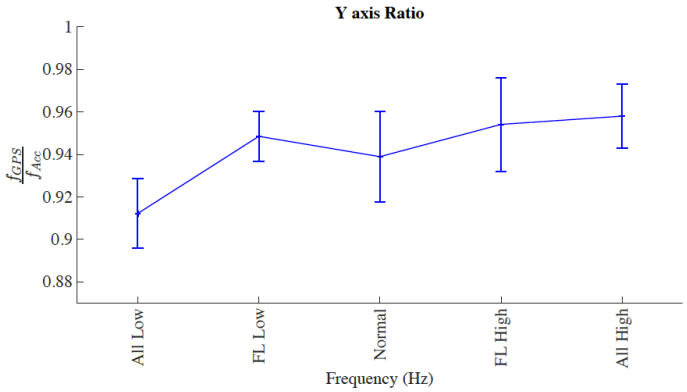


Figure 4-20: Ratio of $\frac{f_{GPS}}{f_{Acc}}$ using f_{Acc} peaks from the Y axis of the accelerometer. Figure ©2016 IEEE.

error associated with the measurements. Other spectral components, such as the peak frequency at 20 Hz, can be attributed to the vibration of the phone mount attached to the windshield as discussed in the imbalance detection section.

4.2.3.7 Feature Selection

This section considers how best to identify features in the Fourier Transformed and GPS data to aid in differentiating from among the five pressure states.

Figure 4-20 shows the utility of $\frac{f_{GPS}}{f_{Acc}}$ for the Y acceleration axis. However, given the large standard errors associated with the measurements, it may not be sufficient to distinguish *FL Low*, *Normal*, and *FL High* from one another.

The deviation in this ratio is a result of variability in GPS data (shown in Table 4.4) and variability in the accelerometer data (shown in Figure 4-21).

To reduce the impact of this variability and improve classification robustness, we applied Principal Component Analysis to transform the Fourier Transformed data per the algorithm in Figure 4-22.

Figure 4-23 illustrates how in the case of the PCA transformed FT features for the *FL Low* case the variability has been reduced.

Figure 4-24a through Figure 4-24c present the comparison of the PCA transformed FFT features for the *FL Low*, *Normal*, *FL High*, and *All High* test cases for all 3 accelerometer axes.

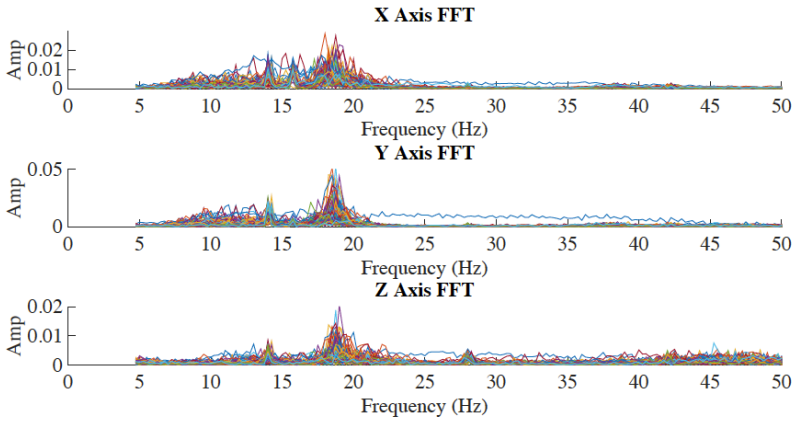


Figure 4-21: Raw FT feature extraction across 180 windows and all 3 accelerometer axes for the *FL Low* case. Figure ©2016 IEEE.

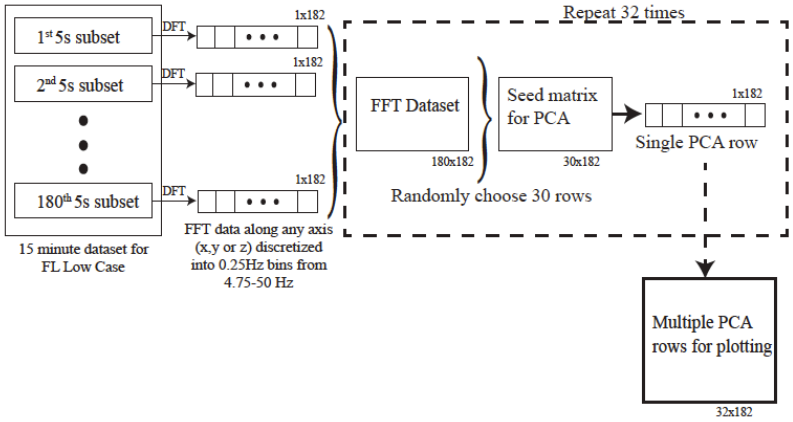


Figure 4-22: Algorithm illustrating the PCA transformation of FFT spectral data for the *FL Low* case. Figure ©2016 IEEE.

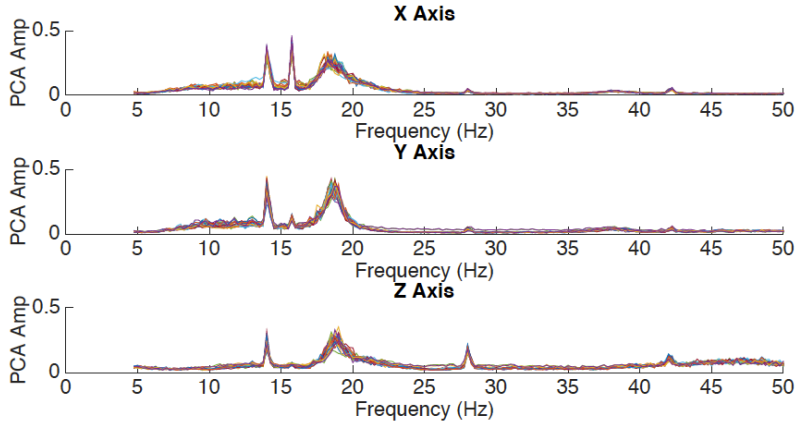
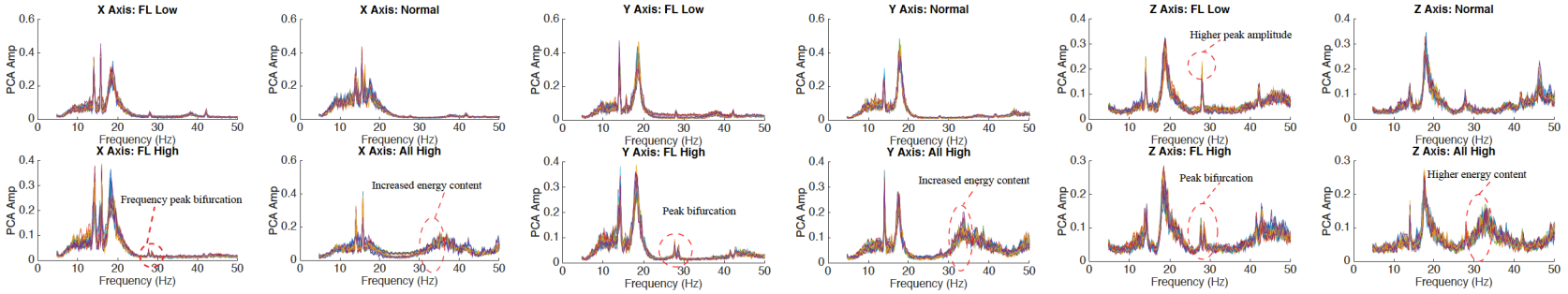


Figure 4-23: 1^{st} PCA component transformation of FFT spectral data for the *FL Low* case across all 3 accelerometer axes. Figure ©2016 IEEE.



(a) PCA transformed X axis FFT features. (b) PCA transformed Y axis features. (c) PCA transformed Z axis features.

Figure 4-24: PCA transformation of X, Y and Z axis FFT features for the 4 cases. The significant differences are marked in the figure. Figures ©2016 IEEE.

The 25-32Hz peak location and amplitude varies significantly among cases. For the *FL High* cases, the single peak bifurcated and provides another feature of interest for classification.

From the above, we selected several features to input into a classifier for machine learning:

- $\frac{f_{GPS}}{f_{Acc}}$ where f_{Acc} is the peak frequency in the 10-16 Hz range along the Y axis.
- The top 2 frequency peaks and amplitudes in the 25-32 Hz range of the PCA transformed FFT data for the X, Y and Z acceleration axes. Capturing the top two features allows us to capture the bifurcation effect. This provides 12 new features.

The decision tree is trained based on the 13 features listed in the previous paragraphs.

4.2.3.8 Application of Machine Learning

This section considers the implementation of supervised machine learning using a decision tree to differentiate between the five pressure states.

4.2.3.8.1 Feature Matrix Generation The algorithm is illustrated in Figure 4-25.

The accelerometer data were windowed into 180 Fourier Feature rows. A vector of 450 GPS tire rotation frequency estimates is also collected, with 180 elements randomly extracted. From the 180 Fourier Feature windows, the following features are extracted:

- The peak frequency f_{Acc} in the 10-16 Hz range for the 180 windows of Y axis acceleration. Each of the 180 randomly selected GPS frequency samples is divided by the corresponding f_{Acc} to obtain the frequency ratio $\frac{f_{GPS}}{f_{Acc}}$ where f_{Acc}.
- The top 2 frequency peaks and amplitudes in the 25 - 32 Hz range from the 180 windows of X, Y and Z axis acceleration.

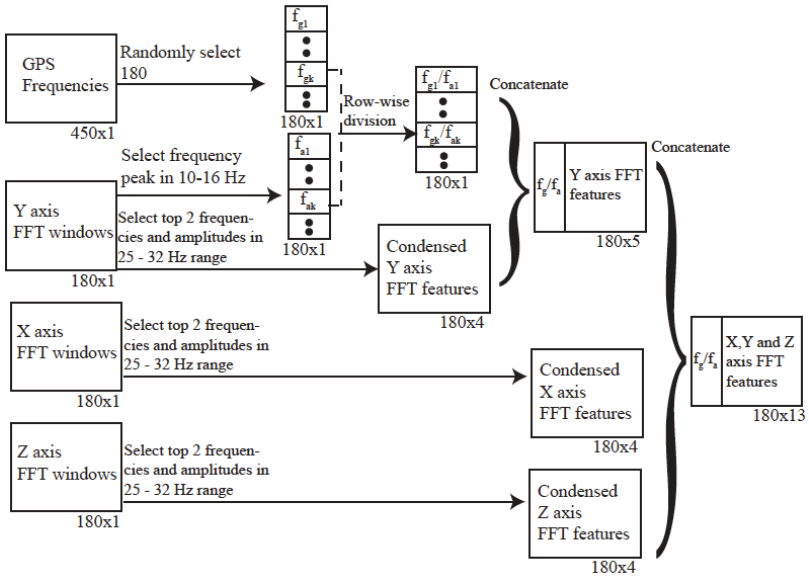


Figure 4-25: Feature generation for any of the 5 test cases. Figure ©2016 IEEE.

The net result is the creation of a 180×13 feature vector for training and testing the decision tree.

4.2.3.8.2 Decision Tree Algorithm As shown in Figure 4-26, the 180×13 dataset for all 5 test cases is randomly divided into testing and training sets.

The data are split, with 70% for training and the remaining 30% applied to testing. Training data are constructed by selecting 35 rows randomly from the training set as a seed matrix for Principal Component Analysis. This process is repeated 40 times, generating a 40×13 training set for each of the five cases. The decision tree is therefore trained on a matrix of 200×13 samples. Testing data are generated similarly. Finally the classification accuracy is evaluated by construction a confusion matrix for the results of the 70 test samples.

4.2.3.9 Pressure Classification Results

Figure 4-27 presents the confusion matrix generated by our decision tree algorithm.

Figure 4-28 shows the branching of the decision tree.

- The leaf nodes of the tree correspond to the five test cases, which have been assigned labels 1 through 5 as illustrated in Table. 4.5.

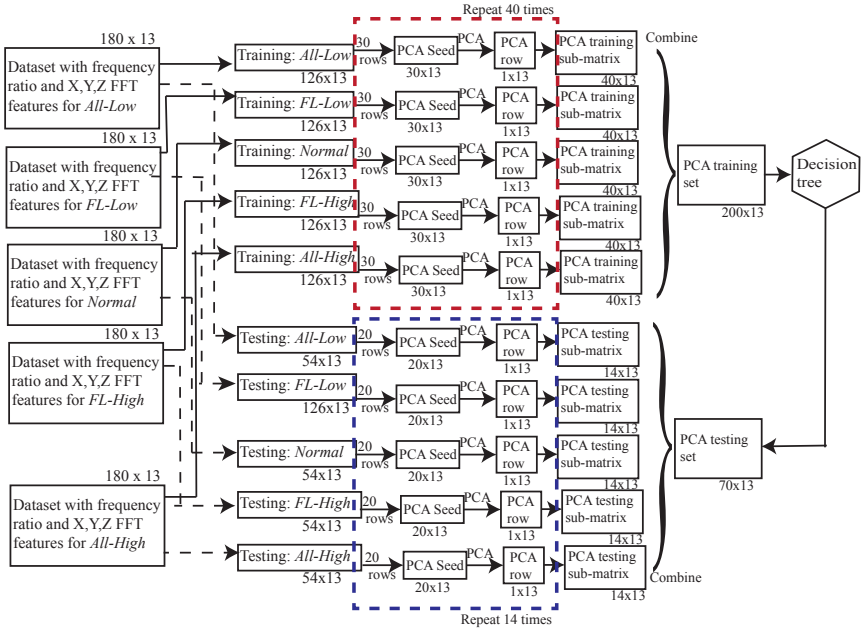


Figure 4-26: Training and testing data generation for tree learning. Figure ©2016 IEEE.

Ground Truth \ Prediction	All Low	FL Low	Normal	FL High	All High
All Low	14	0	0	0	0
FL Low	0	14	0	0	0
Normal	0	0	14	0	0
FL High	0	0	0	14	0
All High	1	0	0	0	13

Figure 4-27: Classification accuracy of test data using the tree learning algorithm. Figure ©2016 IEEE.

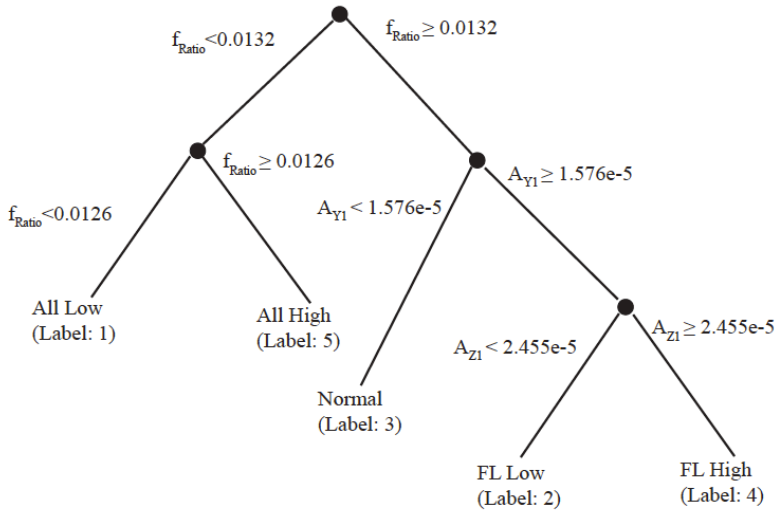


Figure 4-28: Branching decisions undertaken by the decision tree algorithm. Figure ©2016 IEEE.

All Low	FL Low	Normal	FL High	All High
1	2	3	4	5

Table 4.5: Labels used in tree learning. Table ©2016 IEEE.

- f_{ratio} corresponds to the $\frac{f_{GPS}}{f_{Acc}}$ feature.
- A_{Y1} corresponds to the PCA amplitude corresponding to the top Y axis frequency peak in the 25-32 Hz range as seen in Figure 4-24b.
- A_{Z1} corresponds to the PCA amplitude corresponding to the top Z axis frequency peak in the 25-32 Hz range as seen in Figure 4-24c.

Upon inspection, the branching decisions make sense. The root feature is f_{ratio}, which distinguishes *All Low* and *All High* from the other cases. From Figure 4-24b, we observe that using the relatively smaller A_{Y1} of the *Normal* case to distinguish it from *FL Low* and *FL High* makes sense as well. Finally, looking at Figure 4-24c, we observe that using A_{Z1} to distinguish between *FL Low* and *FL High* is also valid, as this differentiation branch is consistent across multiple training/testing sets.

We ran the algorithm five times with random testing and training sets. In each case, the results were comparable to Figure 4-27 with classification accuracies $\geq$90%.

The tree's branching decisions were similar to Figure 4-28 for all five runs.

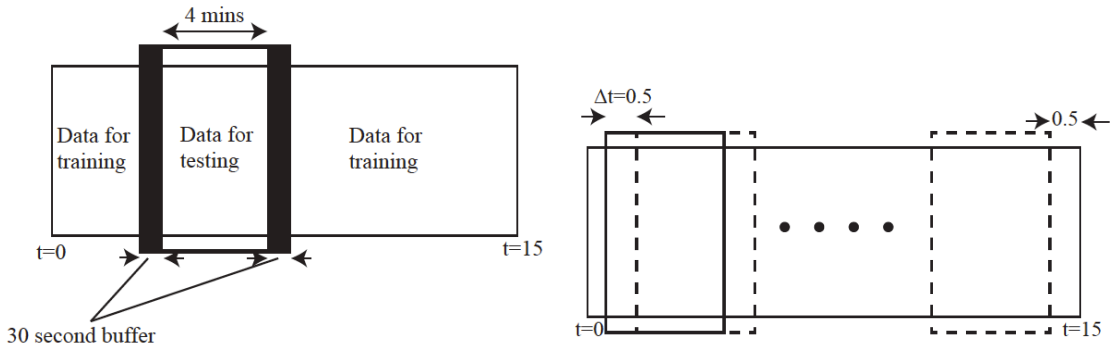
4.2.3.9.1 Mitigating the Impact of Varied Road Conditions While the algorithm worked well for five random testing and training sets, we also sought to examine the sensitivity of our algorithm to road surface variations. It is possible, for example, to conduct this experiment with the tire states and road surfaces roughly aligning. We therefore might have been training the classifier based on road surface, rather than other inputs.

To eliminate the risk of improper training, we implemented a moving window through the dataset as shown in Figure 4-29a. A buffer of 30 seconds ensures that the testing and training set do not overlap and lead to invalid results. We selected the size of the testing window to be four minutes, indicative of 30% data being used for testing. The training dataset is therefore 7 minutes in size. We begin the moving window at the t=30 second mark and advance it in increments of 30 seconds as shown in Figure 4-29b. We can obtain up to 20 snapshots partitioning the entire dataset using this windowing approach. Figure 4-30 illustrates the modification of testing and training data using this approach and Figure 4-31 shows the results of classification. Though classification accuracy is reduced by the time windowing, we achieve 80% accuracy in the worst case. The features for branching are unchanged from Figure 4-28.

4.2.3.10 Tread Depth Monitoring

Finally, this same model making use of $\frac{f_{GPS}}{f_{Acc}}$ may be applied to tread depth supervision.

Tread wear manifests similarly to pressure drops in that the effective diameter drops, though roundness has less impact as sidewall stiffness is retained. It is possible to determine whether the change in frequency is due to a drop in tire pressure or a change in tread depth by looking at a time history of ΔF and the number of wheels differing from specifications. In the case that all four wheels are uniform, there is a systemic pressure drop or diameter change, indicating temperature effects or wear. If one wheel is out of range, this wheel has likely been damaged or is leaking.



(a) Illustration of moving window for dataset partitioning.

(b) Advancement of moving window through dataset.

Figure 4-29: Implementation of a moving window technique to examine the algorithm sensitivity to road surface conditions. The time stamps shown in the figure are in minutes. Figure ©2016 IEEE.

To illustrate the concept, an example tire of size P235/75R15 has a nominal diameter of 734.06mm and a nominal circumference of 2306.12 mm/revolution. We model the nominal diameter as the midpoint of wear and assume that the tire has 8 mm of tread, with an unsafe tire having 1.5mm of tread remaining. If the tire may be approximated as cylindrical, the new circumference becomes 2347mm and the worn circumference is 2265.3mm. At 100kph, the difference between the new and worn-out tires is 0.42Hz.

We conducted an experiment to determine if the accelerometer and GPS could repeatably identify a shift of this magnitude. Pressure was used as a proxy for tread depth in a low-profile tire (to maximize roundness for improved simulation accuracy). Tire pressure were varied from 27 to 33 PSI, a pressure delta commonly occuring in passenger vehicles, and data were logged in the *All Low* and *All High* states.

The difference between the wheel's GPS and accelerometer predicted frequencies is shown in Figure 4-32. Note that the iPhone data is sufficient to pick up the difference between the two states with statistical significance ($t_{stat}95\% = 17.3$ for a 1-sided distribution with 79 d.o.f). Furthermore, the frequency difference between the *Normal* and *All Low* case is 0.1608, which is much less than the frequency shift expected due to tire wear.

The result of this experiment proved viable the concept of passive, non-invasive

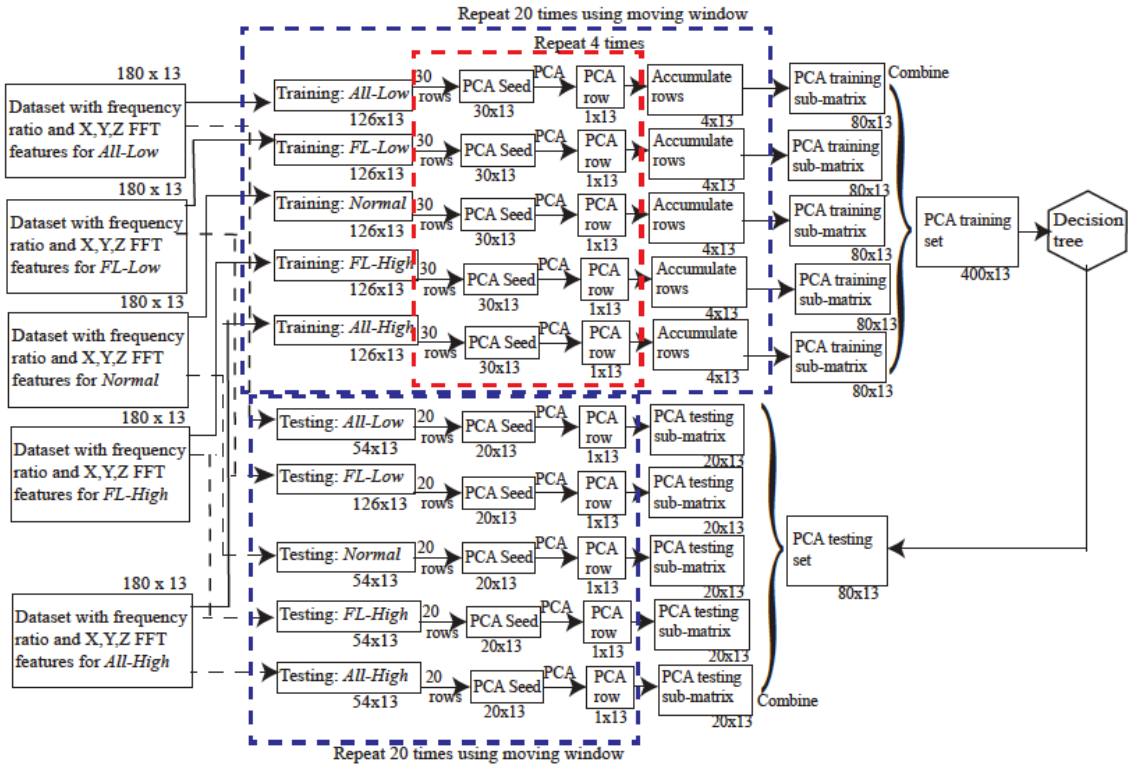


Figure 4-30: Algorithm for testing and training dataset generation using the moving window method. Figure ©2016 IEEE.

Ground Truth \ Prediction	All Low	FL Low	Normal	FL High	All High
All Low	20	0	0	0	0
FL Low	0	16	3	1	0
Normal	0	1	19	0	0
FL High	0	0	0	20	0
All High	0	0	0	0	20

Figure 4-31: Decision tree classification accuracy using the moving window method. Figure ©2016 IEEE.

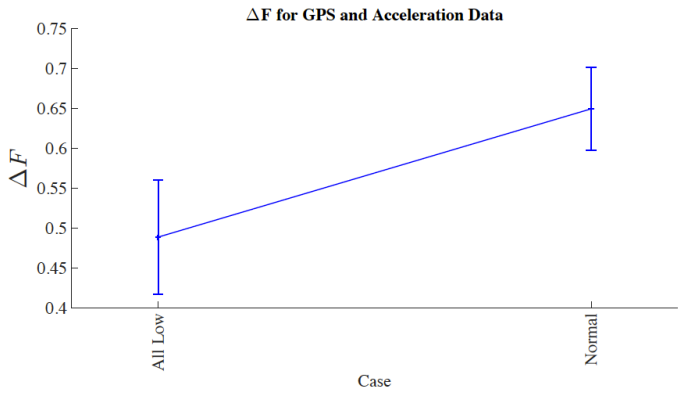


Figure 4-32: The absolute value in the difference between the anticipated and actual wheel rotational frequencies was sufficient to differentiate the all-low from all-high tire states, which mimic the change in tire tread diameter from new to minimum safe. Figure ©2016 IEEE.

.

tread monitoring. It is reasonable to expect successful extension to tread depth supervision without changes to the data collection system. From the above results, we see it is possible to not only identify new versus old tires, but to additionally provide gradation.

4.2.3.11 Pressure and Tread Monitoring Conclusions

We were able to successfully demonstrate smartphone tire pressure measurement and tread depth supervision. The results show a high degree of accuracy (80% in the worst case, with a 20% pressure drop) and a repeatable decision tree capable of differentiating over- and under-inflation caused by intrinsic or extrinsic causes. The models demonstrated are robust to road surface and the main differentiating element, the GPS/accelerometer velocity ratio, has been proven to be generalizable across vehicle makes, models, and tire sizes.

4.2.4 Relating Pervasive Suspension Monitoring to Data Proxies

The above sections show how pervasive, mobile sensing may be used to take vehicle diagnostics and repair from reactive to proactive. These applications stand to benefit

greatly from the introduction of theory posed in Chapter 2 and the CloudThink platform discussed in Chapter 3. The use of a device like the Carduino will allow the capture of non-OBD parameters, such as ABS wheel speed sensors, to improve the model at little to no cost, while further integration into the Cloud, and fusion with other services through semantic reasoning will allow additional data inputs to enhance the presented classification models.

The theory posed in Chapter 2 must be considered when transitioning these applications from a lab environment into commercial products. Understanding how to properly minimize data and to split computation and storage between mobile devices and a server will be important design decisions that ultimately determine the viability of pervasive sensing as an automotive diagnostic solution.

4.3 Engine Misfire Detection with Pervasive Mobile Audio

This section is based off of "Engine Misfire Detection with Pervasive Mobile Audio" [56], submitted to ECML-PKDD 2016. I worked with Sumeet Kumar and Isaac Ehrenberg to apply pervasive sensing to identifying anomalous engine operation using features generated from an iPhone microphone.

In this section, we present a novel method of applying smartphone audio to determine if an engine is operating abnormally. We recorded audio samples and computed features for segments of these samples using Fourier, wavelet, and mel-frequency cepstrum features, then reduced feature count using the Fisher Score and RELIEF score. We were able to obtain a model accuracy of 99% using a linear SVM. This application of mobile computation can take advantage of Proxies to reduce feature set size further, in order to better monitor vehicle subsystems.

4.3.1 Introduction to Misfire Detection

With increased miles traveled [259], hours spent in traffic [260] and an aging vehicle fleet in the United States and around the world [261], responsive maintenance is critical.

At the core of a vehicle's repair is the engine. Combustion engines require three 'ingredients' to run: fuel, air, and ignition spark. A common engine fault results from a weak or non-existent spark, wherein the fuel in a cylinder fails to combust.This increases engine noise, vibration, and harshness while decreasing power and economy. A weak spark results from neglected spark plug or wire maintenance, or could be caused by ignition component failure.

While diagnostic systems are capable of detecting a misfire, these rely on proprietary algorithms, are slow to react, and necessitate the use of specialized interfaces. It is desirable to instead detect faults passively and without specialized equipment, enabling early detection. With this approach, the source of a misfire may be addressed easily before the failure takes a costly toll on other components.

The use of mobile phones as "automotive tricorders" capable of non-invasively detecting vehicle condition will encourage drivers to actively maintain their vehicles, with improved ease-of-use relative to current diagnostic offerings. Passive sensing will allow a shift from reactive repair to proactive maintenance, with this technique having been used successfully for monitoring of vehicle suspension components [53] [54].

In the absence of a robust, physical model, we apply machine learning to uncontrolled data and demonstrate feature-based misfire detection and show that pervasive sensing may identify misfiring engines.

In Section 4.3.2, we consider prior art and how our method differs. Section 4.3.3 describes data generation, while Section 4.3.4 explores generating a feature vector and our approach to reducing set size using ranking techniques. In Section 4.3.4.6, we discuss the classification algorithms we implemented and their relative merits, drawbacks, and efficacy. We conclude in Section 4.3.5 and show 99% classification accuracy resulting from a set with 50% outsample data, before Section 4.3.6 discusses

plans for future improvement.

4.3.2 Previous Approaches to Misfire Detection

Typically, an engine's crankshaft rotates through a fixed angle between every cylinder firing. A crankshaft sensor detects changes in the precession of the crankshaft resulting from a misfire. Detecting a series of near-consecutive unexpected angular measurements prompts the illumination of a check engine light. The use of a diagnostic scan tool will reveal which cylinder or cylinders are misfiring, but this information is of uncertain provenance due to the use of proprietary classification schemes [262] [263]. Direct-sensed alternatives include sampling of exhaust gas pressure, measuring ionization current in the combustion chamber, or installing other sensors within [264] [265] or outside the combustion chamber [266] [267].

Automobile mechanics have long relied upon auditory diagnosis, listening to engines to determine the presence of a misfire. The fact that "old pro" mechanics can classify abnormal firing by ear lends credence to the idea machine-learned detection may be tenable. Indeed, some approaches make use of audio processing to detect the characteristic "pop" sound emanating from misfiring engines.

To acquire audio signals, Dandare [268] and Sujono [269] used recording equipment to analyze the sound from normal and misfiring engines in a laboratory environment. Dandare applied an Artificial Neural Network to classify faults with accuracies from 85-95%. Kabiri and Ghaderi [270] [271] introduced noise into their measurements of over 300 single cylinder engines by recording samples in a garage. Principal Component Analysis and correlation-based feature selection in both time and frequency domains achieved a between 70% and 85% accuracy. Anami recorded several hundred motorcycles [272] to aid mechanics in the rapid classification of healthy versus faulty, with wavelet-based machine learning techniques distinguishing the category of fault present. This included, for example, whether the fault was in the engine or exhaust system. Experienced mechanics provided ground truth, with the classification system reporting $> 85\%$ accuracy relative to this variable reference.

Using the smartphone at the center of a remote diagnostic system, Tse [273]

installed sensors including accelerometers and encoders within a test vehicle. When a misfire or other event was detected, a smartphone presented a message informing a user. In Navea [274], the smart phone itself was used as the data collection device, and was held 30cm above the engine cover to record sounds of the engine and drive belt during startup, at idle, and around 1000 RPM. Thirty-five Honda Civics were recorded at various locations and in differing ambient conditions. Startup issues relating to the car battery, fuel supply and timing were recognized 100% of the time, while a normal engine at idle or 1000 RPM was identified with a 33% false positive rate using only Fourier Transform features. This high rate could easily lead to unnecessary, costly repairs.

Previous work has shown potential for using audio signals as a misfire detection technique, with the capacity for smartphones to serve as diagnostic tools within the reach of the general public. There is an opportunity to help vehicle owners passively supervise the operation their cars without environmental control or specialized equipment, yielding accuracy meeting or exceeding that of a trained and certified mechanic.

4.3.3 Misfire Data Collection

4.3.3.1 Recording Induced Misfires

The goal of the experiment was to collect audio in a manner that could be duplicated by vehicle owners with access to a smartphone. Therefore, the procedure did not rely on fixed measurement distances and the background environment was not controlled, introducing ambient sound sources such as wind to add noise to the data.

To record the audio, each of the four tested vehicles was warmed up for five minutes to ensure the engine was not at "fast idle," which would provide unwanted signal artifacts. The vehicle's hood was then opened. For at least two minutes and thirty seconds, we used an Apple iPhone 6S to record the idle sound as a 48kHz stereo .WAV file. The mobile device was moved over top of the engine to provide a robust training set incorporating engine component noise. This motion is shown in Figure

4-33.

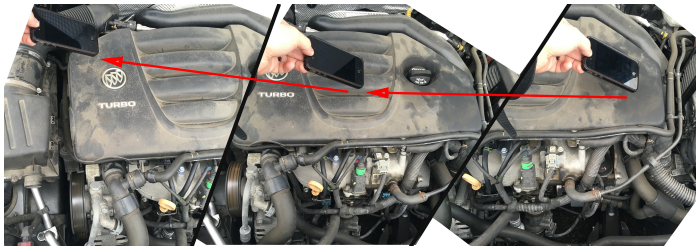


Figure 4-33: The phone recorded as it was moved over the engine to provide background noise to test algorithm robustness. Engine covers were left on to provide a better representative use case for in-situ monitoring.

To induce a misfire, the engine coil pack was disconnected by removing the 12V supply as shown in Figure 4-34. The engine was run for two minutes prior to recording to allow the engine controller to adapt to periodic non-ignition. Subsequently, audio was recorded from these abnormal engines.

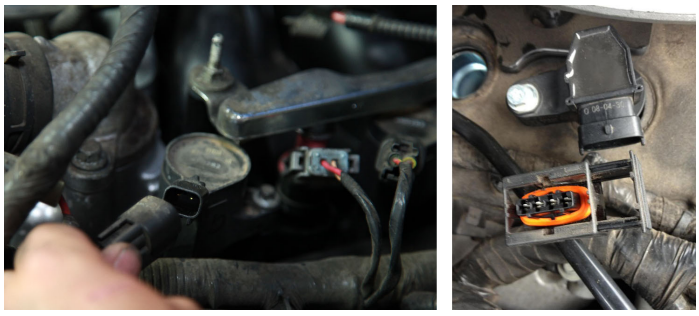


Figure 4-34: The supply to the ignition coil pack was disconnected in order to induce a complete misfire on individual cylinders.

Audio data were collected from multiple vehicles with different engine configurations in various locations including open lots, garages, and parking structures. In the case of this experiment, the engine configurations tested were a normally aspirated inline-four cylinder layout in a Kia Optima and a Ford Focus, as well as a normally aspirated V6 in a Chevrolet Traverse and Nissan Frontier SUV.

4.3.4 Audio Analysis and Engine State Classification

Misfire detection was formulated as a supervised learning problem classifying two operational states (normal and anomalous) using in-sample and out-sample data

recorded as described previously.

4.3.4.1 Misfire Feature Construction

Audio samples were labeled based on whether the engine was operating normally or abnormally. Stereo samples were averaged into a single, mono channel and then subdivided into 2.5s segments. The first 1s and the last 2s of each sample were discarded to reduce edge effects from starting and stopping recording.

2.5s samples recorded at 48kHz correspond to $120,000$ signal elements. The total number of segments in our data set was 992, out of which 373 corresponded to a normal engine. Figure 4-35 shows a segment of a normal engine audio signal overlaid with a misfiring engine audio sample, demonstrating the difficulty in visually differentiating the operating state.

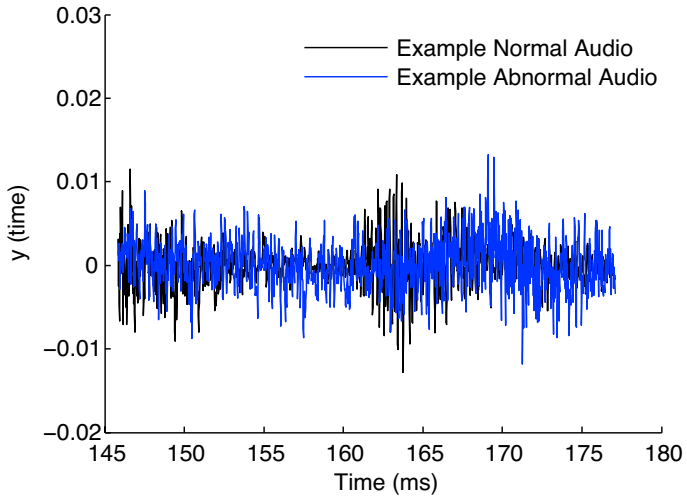


Figure 4-35: Comparison of a segment of a normal audio sample with a misfiring audio sample.

Each $120,000$ element signal was converted into a feature vector to generate features used for classification. Three classes of features generated include binned Fourier Transform coefficients, Wavelet Transform coefficients, and Mel Frequency Cepstral coefficients. These were then concatenated to form a long feature vector.

The use of a dense feature set avoids reliance on preconceived bias of what features have good discriminative power, allowing machine learning techniques to drive a

reduced-feature solution. A reduced feature set allows rapid computation using the programmable Digital Signal Processors (DSPs) on mobile devices. Use of such processors has been shown to minimize a classification algorithm's impact on battery life significantly, even allowing operation without the use of a Cloud backend [275].

4.3.4.2 Binned Fourier Transform (FT) Coefficients

The discrete samples were power-normalized and de-trended, removing bias and linear drift. The Fast Fourier Transform (FFT) was then applied to convert the de-trended time domain signals into the frequency domain. Frequencies $<$ 10kHz were divided into bins 10Hz wide, with the average magnitude in each FT bin providing one feature. This resulted in the creation of a size 1000 feature vector.

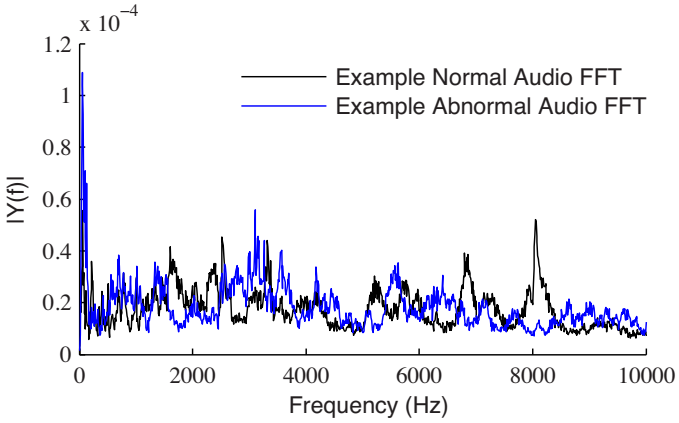


Figure 4-36: Comparison of the spectral density of a normal audio sample with a misfiring audio sample.

Figure 4-36 shows an example comparison of the magnitude of the FT of a normal engine audio signal with a misfiring engine audio sample. We observe that several frequencies (for example, around 2kHz and 8kHz) have a distinct pattern in the normal vs. abnormal cases. These frequencies that are statistically more powerful classifiers will be identified and used to classify a normal engine from a misfiring engine.

4.3.4.3 Discrete Wavelet Transform (DWT) Coefficients

We additionally conducted a wavelet decomposition at level 10 on the normalized, de-trended discrete signal using a Daubechies 4 wavelet. At each level of signal decomposition, mean, standard deviation and skewness was computed resulting in a 33-dimensional feature vector.

4.3.4.4 Mel Frequency Cepstral Coefficient (MFCC)

The MFCC creates a spectral signature of short-term frames that has been successfully applied to speech recognition [276]. We used a frame size of 1024 samples, with each frame incrementally shifted by 512 samples for a total of 233 frames. For each frame, 12 MFCC coefficients were extracted to form a feature vector of size 2796. We applied the GNU-licensed Voicebox MATLAB toolbox to conduct MFCC feature extraction.[4]

Concatenating the three feature vectors from the FT, DWT, and MFCC resulted in a 3829-dimensional representation of the audio signal and a data matrix of size 992×3829. The data were divided randomly into a 50% training set and a 50% test set.

4.3.4.5 Feature Selection

To simplify computation, lessen redundancy and minimize overfitting, it was necessary to reduce the higher-dimensional feature vector using feature selection [277] [278] [279]. We applied two filter-based methods for feature ranking: Fisher Score (FS) and Relief Score (RS) [278].

The Fisher score [278] is calculated using:

$$\begin{aligned} FS(f_i) &= \frac{n_1(\mu_1^i - \mu^i)^2 + n_2(\mu_2^i - \mu^i)^2}{n_1\left(\sigma_1^i\right)^2 + n_2\left(\sigma_2^i\right)^2} \\ &= \frac{1}{n}\frac{(\mu_1^i - \mu_2^i)^2}{\frac{\left(\sigma_1^i\right)^2}{n_1} + \frac{\left(\sigma_2^i\right)^2}{n_2}}, \end{aligned} \tag{4.2}$$

where n_j is the number of samples belonging to class j, $n = n_1 + n_2$, μ^i is the mean

[4] `http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html`

of the feature f^i, μ_j^i and σ_j^i are the mean and the standard deviation of f_i in class j. A larger value corresponds to a higher discriminating power for a variable.

A feature's Relief score [279] is computed by randomly sampling m instances from the data and then applying:

$$RS(f_i) = \frac{1}{2}\sum_{k=1}^{m} d\left(f_k^i - f_{NM(\mathbf{x}_k)}^i\right) - d\left(f_k^i - f_{NH(\mathbf{x}_k)}^i\right), \tag{4.3}$$

where f_k^i denotes the value of the feature f_i on the sample $\mathbf{x}_k$, $f_{NH(\mathbf{x}_k)}^i$ and $f_{NM(\mathbf{x}_k)}^i$ denote the values of the nearest points to $\mathbf{x}_k$ on the feature f_i with the same and different class label respectively, and $d(.)$ is a distance measure which was chosen to be the ℓ_2 norm. Again, a larger score indicates a higher discriminating power of the variable.

Figure 4-37 shows the normalized score (scaled to $\in [0, 1]$) computed by the three methods noted above for each of the generated features. Though FS and RS are correlated (a linear correlation coefficient of 0.49), combining the information from the two methods reduces the likelihood of overfitting. We therefore average the scores from the two methods:

$$AS(f_i) = \frac{1}{2}\left(\frac{FS(f_i)}{\max(FS(f_i)) - \min(FS(f_i))} + \frac{RS(f_i)}{\max(RS(f_i)) - \min(RS(f_i))}\right). \tag{4.4}$$

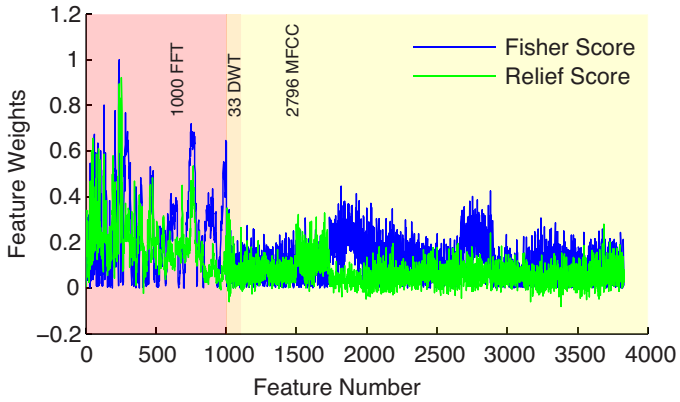


Figure 4-37: Comparison of the feature score calculated by the Fisher and the Relief Score methodology.

We performed a systematic feature reduction study, with feature subsets parametrized by a variable p. For each value p, all features whose scores were in the top p percentile for discrimination were included in the subset.

Figure 4-38 shows the variation in the $10-$fold Misclassification Error Rate (MCR) on the training set using a linear Support Vector Machine (SVM), as well as the feature set size (#F) for different scoring schemes and the percentile cutoff p. We performed a grid search to find the optimal box constraint hyper-parameter (C) for each of the feature subsets in the figure. We identified a minimum MCR at $p = 90$ for each of the three feature scoring methods. Selecting a lower p results in a higher number of less informative features in the subset, leading to poor cross-validation performance and overfitting. Use of a higher p removes important features from the subset leading to a model with decreased accuracy. We additionally observe that with the AS feature ranking the MCR increases less sharply after $p = 90$ when compared to FS or RS, likely due to variance reduction by model averaging. We therefore selected AS with $p = 90$ as the optimal feature subset selection criterion.

The binned FT features alone result in a $10-$fold misclassification rate of 0.8%, while with the DWT the error is 36% and the MFCC based features provide a 29% error. Concatenating all the above features results in a misclassification rate of 2.6% which is higher than FT alone. The minimum misclassification rate with FS, RS and AS scoring is 1.8%, 0.4% and 1.0% respectively (Figure 4-38).

The FT features have a higher discriminating power when compared with the other two feature classes. Simply combining the features from all three does not provide more discrimination than using the FT features. The ability to perform feature ranking and selecting the optimal subset improves the ratio of the discriminating power to the feature set size (i.e. (1-MCR)/#F) and therefore helps determine a small feature set with high discriminative power. It is also noted that the feature subset with AS and $p = 90$ has 358 FT features out of a total of 383 features, 5 DWT features and 20 MFCC features. Among the FT features selected, several were found to group around the 2.5kHz and 7.5kHz frequency bands.

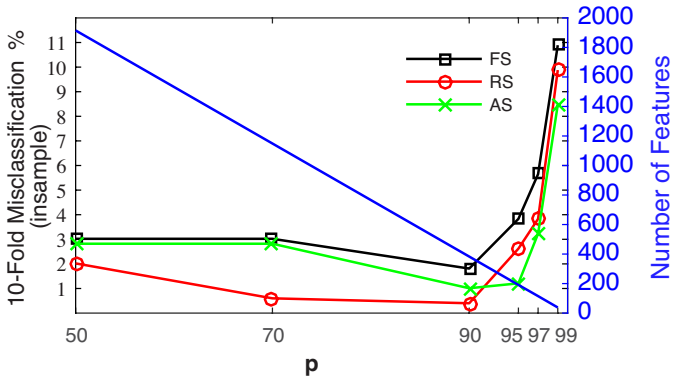


Figure 4-38: Comparison of the 10−fold misclassification rate (MCR) and the feature set size with the variation in p.

4.3.4.6 Misfire Classification Algorithms

Using the chosen reduced feature set (AS feature weighting with the top 90^{th} percentile of features selected), we studied several classification algorithms. The hyperparameters of the classification algorithms were optimized by conducting a grid search to minimize 10−fold cross-validation on the training data. The algorithms tested were k-Nearest Neighbor, Adaboost and SVM with linear, quadratic and RBF kernels. We found that for the SVM with a quadratic kernel all choices of the hyper parameter box-constraint cost (C) led to the same 10-fold misclassification error while for the RBF kernel the error sharply dropped from 38% to 0% around the optimal grid points (for finding C and γ). We therefore removed the quadratic and RBF SVM from the final list of classifiers because we were unable to find a robust set of optimal hyperparameters.

4.3.5 Misfire Classification Results

Table 4.6 summarizes the performance of the different classification algorithms on the 50% outsample data. We observe that the linear SVM significantly outperforms the knn and Adabosst classification algorithms. With linear SVM, we obtained a misclassification rate of 1.0% and the confusion matrix shown in Table 4.7. The 99% accuracy of our approach well exceeds the prior art, indicating that our feature selection and reduction techniques are effective at improving algorithm efficiency as well as increasing accuracy.

Classifier Type	**Optimal Hyper-parameters**	**Mis-classification Rate**	**False Positive Rate (Abnormal as Normal)**
kNN (l_2 distance based)	Number of neighbors (n) = 11	25%	27%
Adaboost (learning rate = 0.3)	Tree depth = 7, Number of trees = 70	15%	11%
Linear SVM	Box-constraint cost (C) = 0.01	1.0%	1.6%

Table 4.6: This table compares the classification accuracy (misclassification rate, reported-normal-when-abnormal false positive rate) for different tested algorithms.

We trained a linear SVM (with C = 0.01) using only the FT features from the final reduced set. The outsample misclassification rate with the top FT features was a higher 1.2% when compared to the results with using the top features of all types (see Table 4.6). This indicates that most of the discriminative information is contained in the FT features, with the DWT and MFCC features helping primarily to differentiate edge cases. This presents an interesting trade off between computing cost and accuracy which will be relevant for designing a mobile sensing application using this technique. Current efficient implementations of FFT on smartphones [280] can be directly implemented for constructing these primary differentiating FT features.

We note that in only one of the four vehicles did a "check engine" light illuminate, indicating that audio detection such as the one presented here with high accuracy and sensitivity may lend itself to the identification of a misfire prior to detection by a conventional OBD system.

		Predicted	
		Normal	**Abnormal**
Actual	**Normal**	100.0%	0.0%
	Abnormal	1.6%	98.4%

Table 4.7: The confusion matrix shows promising results for misfire detection, with 1.6% false positives (reported normal when actually abnormal). We achieve similarly strong performance for false negatives (reporting abnormal when actually normal), potentially saving drivers money on unnecessary repairs.

4.3.6 Future Audio Classification Work

We intend to explore the resource savings afforded by working with a reduced feature set. We have shown that feature ranking facilitates discarding the majority of features with minimal loss in accuracy. Unused features need not be computed, enabling efficient implementations of feature generation algorithms. Additionally, improving the off-line efficiency of these algorithms will allow us to develop an improved on-line approach, minimizing transmitted data and decreasing reference database size.

While this section demonstrates promising results for the use of a mobile phone as a pervasive automotive diagnostic tool, the classification can be enriched and robustness improved to yield an application identification of the misfiring cylinder itself. That was difficult to discern in this study, as we suspect that information to be embedded within phase-based audio features, which are difficult to discern without a reliable indexing feature in the audio relative to engine component rotations. Other, non-combustion sounds are as of yet ill-defined (considering amplitude/frequency spread) and not available as a phase reference. Similarly, with the collected data, it was not immediately feasible to distinguish among various anomalous misfire configurations, but we aim to study other techniques which may be used to improve differentiation among failed states. Such approaches may also improve classification of faults with less well defined signals, such as partial misfires due to lean conditions, and non-misfire faults such as clogged air filters or exhaust leaks.

The misfire application in particular stands to benefit from the theory discussed in Chapter 2. Identifying the optimal car and Cloud split for data processing and transmission could make the application tractable or prove it to be infeasible. In practice, features will be generated on the phone using digital signal processing, and then be reduced prior to transmission. The server will compare the test data against a training set and send a bit back to the device representing normal or abnormal state. The Cloud dataset can be adapted to incorporate new testing data into the training set upon validation of the fault. Ideally, this would take place using a mechanic-operated, supervised learning application.

4.4 Future Applications

Chapter 4 showed how Data Proxies and Cloud-mirroring may be applied to better understand vehicles. Future research will explore using individual and aggregate, realtime and historic data to optimize vehicles in the use and design phases. As alluded to in Chapter 2's example of idle time prediction, Connected Cars may implement resource-efficient connectivity to improve user experience in realtime and gather data for improved drive cycle generation. An extension of the collaborative efficiency-improving applications discussed in Chapter 1 may be applied to everything from informing hybrid drive control strategies to anticipating the needs for novel powertrain technologies and adapting proactively [281] based on conditions ahead including geographic features, traffic, expected route, and more.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 5

Conclusions and Future Opportunities

This thesis explores a range of topics relating to Connected Vehicles and the Internet of Things. My main contribution to the field is to suggest the use of using "Data Proxies" run in the Cloud to mirror physical systems with minimal resource use, and the application of these Proxies for improved security through the use of a "Cognitive Layer" and abstracted connectivity. These findings may be applied to vehicles and other connected systems to lower the barriers of deploying connective technology and to increase the pervasiveness of sensing, inference, and action. Beyond this primary contribution, I explore a number of other research directions relating to improving connectivity and how the resultant data may optimize vehicles in their design and use phases. This thesis' chapter organization, and the main highlights of each, are recapitulated below.

In Chapter 1, I explore the recent proliferation of connectivity in vehicles and other devices. This chapter takes stock of today's technology landscape and those applications that have already been created – namely, those addressing safety in traffic, and to a lesser extent collaborative efficiency. Additionally identified are problems hindering the rapid rollout of Connected Vehicles. These barriers include substantial resource requirements and high connectivity costs, an unmet need for security, consumer sentiment issues such including privacy concerns, and artifacts

of an historically-fragmented Internet and vehicular platform development. After taking stock of the contemporary application landscape, opportunities are identified to create solutions addressing the untapped consumer-facing markets for user experience, single-vehicle efficiency, and reliability improving applications. These applications will benefit drivers and transit systems by allowing for optimization not only in the use phase, but by collecting data to be applied in the design phase as well.

In closing Chapter 1, I suggest the need for improved data generation and transmissibility to enhance Connected platform utility and scalability. To that end, I propose the creation of improved in-vehicle network management for legislated (OBD) and non-legislated (manufacturer proprietary) networks. I further suggest the development of a modernized On-Board Diagnostic architecture, the creation of extensible, interoperable communication protocols to facilitate future connectivity of vehicles and other Internet of Things objects, and recommended guidelines and improvements for privacy, security, and data ownership. These opportunities motivate new modalities for connected platform design.

In Chapter 2, I pose solutions to the two of the most pressing issues in Connected Vehicle deployability and application development: the issues of resource management and of security. I explore typical approaches to connectivity. I then develop and demonstrate theory to reduce resource requirements while improving security for Connected Cars and all other connected systems. The main enabler presented is the concept of the Data Proxy, which uses estimators and observers to duplicate physical objects in the Cloud with reduced data input requirements relative to conventional object mirroring approaches. I further pose a generalized approach to selecting a system model for these Proxies, and explore how an "Application Agent" may make use of "Quality of Data" requirements to optimize the sampling rate intelligently for one or more applications. These Data Proxies additionally improve security by abstracting devices from their end-use applications, enabling the use of a "Cognitive Layer" to validate commands and detect system faults, using predefined, human-in the loop, or machine-learned control limits. With this new architecture, the challenges of resource minimization and enhanced security identified in Chapter 1 are addressed,

paving the way for novel application development.

This chapter further examines application design considerations. I theorize an efficiency and experience improving application, validate that it works using real data, and consider how fundamental connectivity architecture decisions would impact accuracy and cost. I show through simulation that, after the initial cost (which is defrayed by multiple utility), bandwidth costs have the potential to dominate. I suggest that while constant connectivity improves accuracy by providing fresher data and allows the use of more scalable Cloud computation, the costs of data sharing might not be worthwhile depending on how consumers value this application. I suggest that developers consider not just technical feasibility, but economic viability and consumer perception of applications during the design phase.

Chapter 3 considers integrating the theory posed in Chapter 2 into the CloudThink platform for projecting vehicles into the Cloud. I suggest applying Data Proxies to minimize the data requirements for creating comprehensive Avacar digital vehicle duplicates, and making use of the Cognitive Layer to approve commands and detect reliability and operational issues resulting in a shift of system states outside learned control limits. The Privacy Radar is demonstrated as a means of allowing drivers to view and manage application permissions, while the Magic Lens is presented as a concept allowing drivers to visualize the flow of their and their vehicle's data in real-time. These solutions address several of the privacy and security concerns raised in Chapter 1, lowering barriers to entry when connecting vehicles. Finally, I address the concept of using semantic reasoning as an approach to connect cars with other objects and Cloud services flexibly and extensibly. Though not all of these solutions are fully implemented in the platform today, their eventual integration will enable CloudThink to better connect cars and allow the creation of applications addressing real needs.

The applications described in Chapter 4 address the opportunities for development identified in Chapter 1. I present applications to improve vehicle reliability, reduce cost of ownership, and improve user experience. These applications focus primarily on improving the vehicle in the use phase, by applying pervasive sensing from mobile

devices to detect and react to potential failures before they become catastrophic. However, these same data can be used to improve an OEM's understanding of how their vehicles perform in the real-world, helping to improve future design. The applications presented take advantage of the low-cost, scalability, and existing backhaul enabled by mobile devices.

The sensing applications make use of On-Board Diagnostic data fused with temperature data to infer oil life, smartphone accelerometer and GPS data to determine the condition of a vehicle's wheels and tires, and audio collected from a phone's microphone to detect misfires. All of these applications demonstrate approaches that can be enhanced with the use of Data Proxies and through careful architectural decisions, as discussed in Chapter 2. Oil, for example, may be monitored by using an observer, with the Cognitive Supervisor determining when the viscosity exceeds approaches defined system limits. The accelerometer and audio applications may be modified to take advantage of data reduction through the use of estimators, and all of these applications must carefully consider the car/Cloud split as described in Chapter 2. For example, minimizing data transmitted to the server with local signal transformation and aggregation via Digital Signal Processing can save bandwidth for less time-sensitive applications like diagnostics. The Cloud is still necessary for these applications, because the training sets are large relative to phone storage, so determining the appropriate features to transmit and the frequency of transmission is critical. Proper architecture decisions can render any of these prognostic applications either feasible or completely untenable.

5.1 Future

All of this work comes together to do two things:

First, the theory enables the broader deployment of connected solutions. Reducing resource requirements will allow more devices to connect to the Internet. Proxies will allow for reduced bandwidth use and improved battery life in mobile and constrained devices, facilitating the generation of richer data sets than are currently feasible.

Using the approaches described herein, newly-enabled smart devices will be able to communicate with one another, unlocking valuable opportunities and hidden potential at the intersection of current industries. An efficient and secure Cloud Car can talk to a Cloud Person, and all will be duplicated in a Cloud City with Cloud Infrastructure. The real value of the Internet of Things lies not in any of these individual silos, but rather at the intersection of these systems. Proxies allow more of these systems to be connected than ever before.

Second, this theory enables a new range of data-informed design and optimization for vehicles. With pervasive sensing and resource-forward vehicular connectivity, it becomes possible to create the "data-driven car," a vehicle designed for and informed by real-world drive cycles. Unlike cars today, constant connectivity and collaborative data generation in cars of the future will allow automotive OEMs to better understand how their vehicles are used and change appropriate design levers before, during, and after manufacture.

Connectivity has the potential to change how we use vehicles. A Connected Car will leave the factory better than ever, having been designed after taking into consideration all the knowledge collected by the fleet of vehicles preceding it. It will be improved by using higher quality materials better designed to meet drivers' needs, and with a scalable electronic, and possibly mechanical, architecture. It will operate better than ever, using real-time data to change everything from suspension stiffness to fuel trim to battery management. Throughout its use, this car will generate data to improve itself and to improve all future vehicles, contributing to an aggregate database of vehicle mirrors. This future-proof design and extensibility is increasingly important as we transition from individual ownership to mobility as a service transportation models. In-car and pervasive sensing will allow the expansion of applications from cars, to people, to environment.

Combined with the improvements to Connected Vehicle platforms, On-Board Diagnostics, and manufacturer-proprietary networking solutions posed in Chapter 1, we are on the cusp of reinventing the wheel – and my contribution to connectivity is a significant step towards building the connected smart car that future mobility

systems need.

Bibliography

[1] P. Papadimitratos, A. La Fortelle, K. Evenssen, R. Brignolo, and S. Cosenza, "Vehicular communication systems: Enabling technologies, applications, and future outlook on intelligent transportation," *Communications Magazine, IEEE*, vol. 47, no. 11, pp. 84–95, 2009.

[2] M. Faezipour, M. Nourani, A. Saeed, and S. Addepalli, "Progress and challenges in intelligent vehicle area networks," *Communications of the ACM*, vol. 55, no. 2, p. 90, February 2012.

[3] L. D'Orazio, F. Visintainer, and M. Darin, "Sensor networks on the car: State of the art and future challenges," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2011*. IEEE, 2011, pp. 1–6.

[4] I. Sherr. (2013, March) Car makers embrace sensors in a big way. The Wall Street Journal. [Online]. Available: http://blogs.wsj.com/digits/2013/03/11/car-makers-embrace-sensors-in-a-big-way/

[5] P. Gomes, C. Olaverri-Monreal, and M. Ferreira, "Making vehicles transparent through V2V video streaming," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 13, no. 2, pp. 930–938, 2012.

[6] A. Mammeri and A. Boukerche, "Inter-vehicle communication of warning information: an experimental study," *Wireless Networks*, pp. 1–12, 2016.

[7] O. Gusikhin, D. Filev, and N. Rychtyckyj, "Intelligent vehicle systems: applications and new trends," in *Informatics in Control Automation and Robotics*. Springer, 2008, pp. 3–14.

[8] B. Schoettle and M. Sivak, "A survey of public opinion about autonomous and self-driving vehicles in the U.S., the U.K., and Australia," University of Michigan, Ann Arbor, Transportation Research Institute, Tech. Rep., 2014.

[9] V. R. Lesser and D. D. Corkill, "The distributed vehicle monitoring testbed," *AI Magazine*, vol. 4, no. 3, pp. 63–109, 1983.

[10] L. Pelkmans, S. Hultén, R. Cowan, G. Azkarate, and A. Christidis, "Trends in vehicle and fuel technologies: review of past trends," *Inst. for Prospective Technologies Studies, JRC Report EUR 20746 EN*, 2003.

[11] G. Rizzoni, S. Onori, and M. Rubagotti, "Diagnosis and prognosis of automotive systems: motivations, history and some results," in *Proceedings of the 7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes (SAFEPROCESS '09)*, 2009.

[12] P. Gargano. (1999) GM 8192/160 baud ALDL interface. Tech Edge Pty. [Online]. Available: http://wbo2.com/~techedge.com.au/vehicle/aldl8192/8192hw.htm

[13] K. Y. Cho, "Overview of telematics: A system architecture approach," *International Journal of Automotive Technology*, vol. 7, no. 4, pp. 509–517, 2006.

[14] T. Nolte, H. Hansson, and L. L. Bello, "Automotive communications-past, current and future," in *Emerging Technologies and Factory Automation, 2005. ETFA 2005. 10th IEEE Conference on*, vol. 1. IEEE, 2005, pp. 8–pp.

[15] K. McCord, *Automotive diagnostic systems : understanding OBD I & OBD II*. North Branch, MN: CarTech, 2011.

[16] A. C. Galhardi, "New trends in automotive embedded system design," SAE Technical Paper, Tech. Rep., 2014.

[17] B. Hoefflinger, *MEMS - Micro-Electromechanical Sensors for the Internet of Everything*. Springer, 2015.

[18] G. Lammel, "The future of MEMS sensors in our connected world," in *Micro Electro Mechanical Systems (MEMS), 2015 28th IEEE International Conference on*. IEEE, 2015, pp. 61–64.

[19] S. H. Bayless, A. Guan, P. Son, S. Murphy, and A. J. Shaw, "Connected vehicle insights: Trends in roadway domain active sensing: Developments in radar, lidar, and other sensing technologies, and impact on vehicle crash avoidance/automation and active traffic management," US Department of Transportation, Tech. Rep., August 2013.

[20] R. Bishop, "A survey of intelligent vehicle applications worldwide," in *Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE*. IEEE, 2000, pp. 25–30.

[21] S. Demmel, D. Gruyer, and A. Rakotonirainy, "V2V/V2I augmented maps: state-of-the-art and contribution to real-time crash risk assessment," in *Proceedings of 20th Canadian Multidisciplinary Road Safety Conference*. The Canadian Association of Road Safety Professionals, 2010.

[22] R. S. Schwartz, M. Van Eenennaam, G. Karagiannis, G. Heijenk, W. K. Wolterink, and H. Scholten, "Using V2V communication to create over-the-horizon awareness in multiple-lane highway scenarios," in *Intelligent Vehicles Symposium (IV), 2010 IEEE*. IEEE, 2010, pp. 998–1005.

[23] M. B. M. Jonsson, J.-M. B. S. M. B. S. Cherkaoui, M. A. C. Rico-Garcia, H. G. R. Mehmood, and A. Vinel, *Communication Technologies for Vehicles*, 2013th ed., ser. Nets4Cars/Nets4Trains. Springer, 2013, vol. 5. [Online]. Available: http://www.springer.com/us/book/9783642379734

[24] M. Broy, "Challenges in automotive software engineering," in *Proceedings of the 28th international conference on Software engineering*. ACM, 2006, pp. 33–42.

[25] "General Motors local area network enhanced diagnostic test mode specification," General Motors Worldwide, 2010. [Online]. Available: https://global.ihs.com/doc_detail.cfm?document_name=GMW3110&item_s_key=00415169

[26] K. H. Johansson, M. Törngren, and L. Nielsen, "Vehicle applications of Controller Area Network," in *Handbook of networked and embedded control systems*. Springer, 2005, pp. 741–765.

[27] "Regulation, section 1968.2 malfunction and diagnostic system requirements - 2004 and subsequent model year passenger cars," California Air Resources Board.

[28] D. Paret, "Multiplexed networks for embedded systems CAN, LIN, Flexray, Safe-by-Wire ..." Chichester, England; Hoboken, NJ, 2007. [Online]. Available: http://site.ebrary.com/id/10297543

[29] S. Tuohy, M. Glavin, C. Hughes, E. Jones, M. Trivedi, and L. Kilmartin, "Intra-vehicle networks: A review," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 16, no. 2, pp. 534–545, 2015.

[30] Ford Smart Gauge. Smart Design. [Online]. Available: http://smartdesignworldwide.com/work/ford-smart-gauge/

[31] M. Tscheligi, A. Meschtscherjakov, and D. Wilfinger, "Interactive computing on wheels," *Computer*, vol. 44, no. 8, pp. 0100–102, 2011.

[32] S. U. Eichler, "Performance evaluation of the IEEE 802.11 p WAVE communication standard," in *Vehicular Technology Conference, 2007. VTC-2007 Fall. 2007 IEEE 66th.* IEEE, 2007, pp. 2199–2203.

[33] F. Bai, T. Elbatt, G. Hollan, H. Krishnan, and V. Sadekar, "Towards characterizing and classifying communication-based automotive applications from a wireless networking perspective," in *Proceedings of IEEE Workshop on Automotive Networking and Applications (AutoNet)*, 2006, pp. 1–25.

[34] Y. L. Morgan, "Managing DSRC and WAVE standards operations in a V2V scenario," *International Journal of Vehicular Technology*, vol. 2010, 2010.

[35] A. Casteigts, A. Nayak, and I. Stojmenovic, "Communication protocols for vehicular ad hoc networks," *Wireless Communications and Mobile Computing*, vol. 11, no. 5, pp. 567–582, 2011.

[36] Y. J. Li, "An overview of the dsrc/wave technology," in *Quality, Reliability, Security and Robustness in Heterogeneous Networks*. Springer, 2010, pp. 544–558.

[37] B. Bovée, M. Nekoui, H. Pishro-Nik, and R. Tessier, "Evaluation of the universal geocast scheme for vanets," in *Vehicular Technology Conference (VTC Fall), 2011 IEEE*. IEEE, 2011, pp. 1–5.

[38] S. Biswas, R. Tatchikou, and F. Dion, "Vehicle-to-vehicle wireless communication protocols for enhancing highway traffic safety," *Communications Magazine, IEEE*, vol. 44, no. 1, pp. 74–82, 2006.

[39] Y. Toor, P. Muhlethaler, and A. Laouiti, "Vehicle ad hoc networks: applications and related technical issues," *Communications Surveys & Tutorials, IEEE*, vol. 10, no. 3, pp. 74–88, 2008.

[40] Y. Chen, Z. Xiang, W. Jian, and W. Jiang, "An improved aomdv routing protocol for V2V communication," in *Intelligent Vehicles Symposium, 2009 IEEE*. IEEE, 2009, pp. 1115–1120.

[41] S. Mukherjee, A. Baid, and D. Raychaudhuri, "Integrating advanced mobility services into the future Internet architecture," in *Communication Systems and Networks (COMSNETS), 2015 7th International Conference on*. IEEE, 2015, pp. 1–8.

[42] B. Lewis. (2015) Technology convergence, non-linear thinking help V2X architectures come of age. Embedded Computing Design. [Online]. Available: http://embedded-computing.com/articles/automotive-executive-outlook-scott-mccormick-president-connected-vehicle-trade-association/

[43] (2016) Automatic: Connect your car to your life. Automatic. [Online]. Available: https://www.automatic.com/home/

[44] Y. Zhuang, J. Cappos, T. S. Rappaport, and R. McGeer, "Future Internet bandwidth trends: An investigation on current and future disruptive technologies," Technical Report TR-CSE-2013-0411/01/2013, Polytechnic Institute of NYU, Department of Computer Science and Engineering, WEB: http://www. cs. ubc. ca/\ yyzh/tr-cse-2013-04. pdf (available: 22 February 2014), Tech. Rep., 2013.

[45] D. M. G. Jaramillo, "Vehicle online monitoring system based on fuzzy classifier," in *Vehicular 2014*, 2014.

[46] M. Fazeen, B. Gozick, R. Dantu, M. Bhukhiya, and M. C. González, "Safe driving using mobile phones," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 13, no. 3, pp. 1462–1468, 2012.

[47] D. E. Brown, B. Reimer, B. Mehler, and J. Dobres, "An on-road study involving two vehicles: observed differences between an auditory and haptic lane departure

warning system," in *Adjunct Proceedings of the 7th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*. ACM, 2015, pp. 140–145.

[48] L. Li, K. Werber, C. F. Calvillo, K. D. Dinh, A. Guarde, and A. König, "Multi-sensor soft-computing system for driver drowsiness detection," in *Soft Computing in Industrial Applications*. Springer, 2014, pp. 129–140.

[49] F. Friedrichs and B. Yang, "Drowsiness monitoring by steering and lane data based features under real driving conditions," in *Signal Processing Conference, 2010 18th European*. IEEE, 2010, pp. 209–213.

[50] C.-W. W. You, N. D. Lane, F. Chen, R. Wang, Z. Chen, T. J. Bao, M. Montes-de Oca, Y. Cheng, M. Lin, and L. Torresani, "Carsafe app: Alerting drowsy and distracted drivers using dual cameras on smartphones," in *Proceeding of the 11th annual international conference on Mobile systems, applications, and services*. ACM, 2013, pp. 13–26.

[51] M. Swan, "Connected car: quantified self becomes quantified car," *Journal of Sensor and Actuator Networks*, vol. 4, no. 1, pp. 2–29, 2015.

[52] F. Friedrichs and B. Yang, "Camera-based drowsiness reference for driver state classification under real driving conditions," in *Intelligent Vehicles Symposium (IV), 2010 IEEE*. IEEE, 2010, pp. 101–106.

[53] J. E. Siegel, R. Bhattacharyya, S. Sarma, and A. Deshpande, "Smartphone-based wheel imbalance detection," in *ASME 2015 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers, 2015.

[54] J. E. Siegel, R. Bhattacharyya, A. Desphande, and S. E. Sarma, "Smartphone-based vehicular tire pressure and condition monitoring," in *Proceedings of SAI Intellisys 2016*, 2016.

[55] J. Siegel, R. Bhattacharyya, A. Deshpande, and S. Sarma, "Vehicular engine oil service life characterization using on-board diagnostic (OBD) sensor data," in *SENSORS, 2014 IEEE*. IEEE, 2014, pp. 1722–1725.

[56] J. E. Siegel, S. Kumar, I. Ehrenberg, and S. Sarma, "Engine misfire detection with pervasive mobile audio," in *Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases 2016*, 2016.

[57] M. A. Joordens and M. Jamshidi, "Consensus control for a system of underwater swarm robots," *Systems Journal, IEEE*, vol. 4, no. 1, pp. 65–73, 2010.

[58] P. P. Jayaraman, C. Perera, D. Georgakopoulos, and A. Zaslavsky, "Efficient opportunistic sensing using mobile collaborative platform MOSDEN," in *Collaborative Computing: Networking, Applications and Worksharing (Collaboratecom), 2013 9th International Conference Conference on*. IEEE, 2013, pp. 77–86.

[59] P. Martin, B.-J. J. Ho, N. Grupen, S. Munoz, and M. Srivastava, "An iBeacon primer for indoor localization: demo abstract," in *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*. ACM, 2014, pp. 190–191.

[60] C.-C. C. Chen, T.-C. C. Huang, J. J. Park, and N. Y. Yen, "Real-time smartphone sensing and recommendations towards context-awareness shopping," *Multimedia Systems*, vol. 21, no. 1, pp. 61–72, 2015.

[61] H. Jeong, J.-W. W. Lee, J. P. Jeong, and E. Lee, "DSRC based self-adaptive navigation system: Aiming spread out the vehicles for alleviating traffic congestion," in *Frontier and Innovation in Future Computing and Communications*. Springer, 2014, pp. 739–749.

[62] E. Nettleton, S. Thrun, H. Durrant-Whyte, and S. Sukkarieh, "Decentralised slam with low-bandwidth communication for teams of vehicles," in *Field and Service Robotics*. Springer, 2003, pp. 179–188.

[63] Y. Qian and N. Moayeri, "Design of secure and application-oriented VANETs," in *Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE*. IEEE, 2008, pp. 2794–2799.

[64] J. Engelbrecht, M. J. Booysen, G.-J. J. van Rooyen, and F. J. Bruwer, "Survey of smartphone-based sensing in vehicles for intelligent transportation system applications," *Intelligent Transport Systems, IET*, vol. 9, no. 10, pp. 924–935, 2015.

[65] R.-P. P. Schäfer, K.-U. U. Thiessenhusen, E. Brockfeld, and P. Wagner, "A traffic information system by means of real-time floating-car data," in *ITS World Congress 2002*, 2002.

[66] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. Miu, E. Shih, H. Balakrishnan, and S. Madden, "Cartel: a distributed mobile sensor computing system," in *Proceedings of the 4th international conference on Embedded networked sensor systems*. ACM, 2006, pp. 125–138.

[67] S. Husnjak, D. Peraković, I. Forenbacher, and M. Mumdziev, "Telematics system in usage based motor insurance," *Procedia Engineering*, vol. 100, pp. 816–825, 2015.

[68] P. Shelly, "Addressing challenges in automotive connectivity: mobile devices, technologies, and the connected car," *SAE International Journal of Passenger Cars-Electronic and Electrical Systems*, vol. 8, no. 1, pp. 161–169, 2015.

[69] S. Tsugawa, "Inter-vehicle communications and their applications to intelligent vehicles: an overview," in *Intelligent Vehicle Symposium, 2002. IEEE*, vol. 2. IEEE, 2002, pp. 564–569.

[70] D. W. Matolak, "V2V communication channels: State of knowledge, new results, and what's next," in *Communication Technologies for Vehicles*. Springer, 2013, pp. 1–21.

[71] D. Reichardt, M. Miglietta, L. Moretti, P. Morsink, and W. Schulz, "CarTALK 2000: Safe and comfortable driving based upon inter-vehicle-communication," in *Intelligent Vehicle Symposium, 2002. IEEE*, vol. 2. IEEE, 2002, pp. 545–550.

[72] C.-L. L. Huang, Y. P. Fallah, R. Sengupta, and H. Krishnan, "Adaptive inter-vehicle communication control for cooperative safety systems," *Network, IEEE*, vol. 24, no. 1, pp. 6–13, 2010.

[73] A. Rasheed, H. Zia, F. Hashmi, U. Hadi, W. Naim, and S. Ajmal, "Fleet & convoy management using vanet," *Journal of Computer Networks*, vol. 1, no. 1, pp. 1–9, 2013.

[74] W.-H. H. Lee, B.-S. S. Jeng, S.-S. S. Tseng, and C.-H. H. Wang, "Electronic toll collection based on vehicle-positioning system techniques," in *Networking, Sensing and Control, 2004 IEEE International Conference on*, vol. 1. IEEE, 2004, pp. 643–648.

[75] S. Olariu, "Peer-to-peer multimedia content provisioning for vehicular ad hoc networks," in *Proceedings of the 3rd ACM workshop on Wireless multimedia networking and performance modeling*. ACM, 2007, pp. 1–1.

[76] K.-J. J. Choi, "Apparatus and method for settling parking charge using dsrc," U.S. Patent 09/983,746, October, 2001.

[77] T. Ernst, "The information technology era of the vehicular industry," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 2, pp. 49–52, 2006.

[78] T. L. Willke, P. Tientrakool, and N. F. Maxemchuk, "A survey of inter-vehicle communication protocols and their applications," *Communications Surveys & Tutorials, IEEE*, vol. 11, no. 2, pp. 3–20, 2009.

[79] J. E. Siegel, "Design, development, and validation of a remotely reconfigurable vehicle telemetry system for consumer and government applications," S.B. Thesis, Massachusetts Institute of Technology, 2011.

[80] S. Lee, D. Lee, and S. Lee, "Network-oriented road map generation for unknown roads using visual images and gps-based location information," *Consumer Electronics, IEEE Transactions on*, vol. 55, no. 3, pp. 1233–1240, 2009.

[81] S. Fujii, A. Fujita, T. Umedu, S. Kaneda, H. Yamaguchi, T. Higashino, and M. Takai, "Cooperative vehicle positioning via V2V communications and on-board sensors," in *Vehicular Technology Conference (VTC Fall), 2011 IEEE*. IEEE, 2011, pp. 1–5.

[82] M. Obst, E. Richter, and G. Wanielik, "Accurate relative localization for land vehicles with sbas corrected gps/ins integration and V2V communication," in *Proceedings of the 24th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2011)*, 2001, pp. 363–371.

[83] A. Daniel, A. Paul, A. Ahmad, and S. Rho, "Cooperative intelligence of vehicles for intelligent transportation systems (ITS)," *Wireless Personal Communications*, pp. 1–24, 2015.

[84] D. Caveney and W. B. Dunbar, "Cooperative driving: beyond V2V as an ADAS sensor," in *Intelligent Vehicles Symposium (IV), 2012 IEEE.* IEEE, 2012, pp. 529–534.

[85] K. Golestan, S. Seifzadeh, M. Kamel, F. Karray, and F. Sattar, "Vehicle localization in VANETs using data fusion and V2V communication," in *Proceedings of the second ACM international symposium on Design and analysis of intelligent vehicular networks and applications.* ACM, 2012, pp. 123–130.

[86] H. Li and F. Nashashibi, "Multi-vehicle cooperative localization using indirect vehicle-to-vehicle relative pose estimation," in *Vehicular Electronics and Safety (ICVES), 2012 IEEE International Conference on.* IEEE, 2012, pp. 267–272.

[87] M. Röckl, T. Strang, and M. Kranz, "V2V communications in automotive multi-sensor multi-target tracking," in *Vehicular Technology Conference, 2008. VTC 2008-Fall. IEEE 68th.* IEEE, 2008, pp. 1–5.

[88] L. Garelli, C. Casetti, C.-F. F. Chiasserini, and M. Fiore, "Mobsampling: V2V communications for traffic density estimation," in *Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd.* IEEE, 2011, pp. 1–5.

[89] T. Yamashita, K. Izumi, K. Kurumatani, and H. Nakashima, "Smooth traffic flow with a cooperative car navigation system," in *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems.* ACM, 2005, pp. 478–485.

[90] L. C. Bento, R. Parafita, and U. Nunes, "Intelligent traffic management at intersections supported by V2V and V2I communications," in *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on.* IEEE, 2012, pp. 1495–1502.

[91] Y. L. Murphey, "Intelligent vehicle power management: An overview," in *Computational Intelligence in Automotive Applications.* Springer, 2008, pp. 169–190.

[92] Z. Hui-ling, X. Jian-min, T. Yu, H. Yu-cong, and S. Ji-feng, "The research of parking guidance and information system based on dedicated short range communication," in *Intelligent Transportation Systems, 2003. Proceedings. 2003 IEEE*, vol. 2. IEEE, 2003, pp. 1183–1186.

[93] F. Ye, "V2V wireless communication protocol for rear-end collision avoidance on highways," in *IEEE Communications Conference 2008*, 2008.

[94] J. A. Misener, R. Sengupta, and H. Krishnan, "Cooperative collision warning: Enabling crash avoidance with wireless technology," in *12th World Congress on ITS*, vol. 3. Citeseer, 2005.

[95] D. Ruina, D. Jieyun, S. Bo, Y. Qichang, T. Yuanmu, and L. Keqiang, "A lane change warning system based on V2V communication," in *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*. IEEE, 2014, pp. 1923–1928.

[96] S. Kato, S. Tsugawa, K. Tokuda, T. Matsui, and H. Fujii, "Vehicle control algorithms for cooperative driving with automated vehicles and intervehicle communications," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 3, no. 3, pp. 155–161, 2002.

[97] J. Lee and B. Park, "Development and evaluation of a cooperative vehicle intersection control algorithm under the connected vehicles environment," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 13, no. 1, pp. 81–90, 2012.

[98] E. Wilhelm, J. Siegel, S. Mayer, L. Sadamori, S. Dsouza, C.-K. K. Chau, and S. Sarma, "Cloudthink: a scalable secure platform for mirroring transportation systems in the cloud," *Transport*, vol. 30, no. 3, pp. 320–329, 2015.

[99] J. E. Siegel, "CloudThink and the Avacar: embedded design to create virtual vehicles for cloud-based informatics, telematics, and infotainment," Master's thesis, Massachusetts Institute of Technology, 2013.

[100] Y. P. Fallah, C.-L. L. Huang, R. Sengupta, and H. Krishnan, "Analysis of information dissemination in vehicular ad-hoc networks with application to cooperative vehicle safety systems," *Vehicular Technology, IEEE Transactions on*, vol. 60, no. 1, pp. 233–247, 2011.

[101] S. Xu, H. Zhou, C. Li, and Y. Zhao, "A multi-hop V2V broadcast protocol for chain collision avoidance on highways," in *Communications Technology and Applications, 2009. ICCTA'09. IEEE International Conference on*. IEEE, 2009, pp. 110–114.

[102] E. C. Eze, S.-J. J. Zhang, E.-J. J. Liu, and J. C. Eze, "Advances in vehicular ad-hoc networks (VANETs): Challenges and road-map for future development," *International Journal of Automation and Computing*, pp. 1–18, 2016.

[103] G. M. T. Pinto. (2014) The Internet on wheels and Hitachi, Ltd. Hitachi. [Online]. Available: https://www.hds.com/assets/pdf/hitachi-point-of-view-internet-on-wheels-and-hitachi-ltd.pdf

[104] J. Yin, T. ElBatt, G. Yeung, B. Ryu, S. Habermas, H. Krishnan, and T. Talty, "Performance evaluation of safety applications over DSRC vehicular ad hoc networks," in *Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks*. ACM, 2004, pp. 1–9.

[105] A. Bazzi, B. M. Masini, and G. Pasolini, "V2V and v2r for cellular resources saving in vehicular applications," in *Wireless Communications and Networking Conference (WCNC), 2012 IEEE*. IEEE, 2012, pp. 3199–3203.

[106] J. He, Z. Tang, T. O'Farrell, and T. M. Chen, "Performance analysis of DSRC priority mechanism for road safety applications in vehicular networks," *Wirel. Commun. Mob. Comput.*, vol. 11, no. 7, pp. 980–990, August 2009.

[107] K. Lidström and T. Larsson, "A spatial qos requirements specification for V2V applications," in *Intelligent Vehicles Symposium (IV), 2010 IEEE*. IEEE, 2010, pp. 548–553.

[108] P. Papadimitratos, "On the road-reflections on the security of vehicular communication systems," in *Vehicular Electronics and Safety, 2008. ICVES 2008. IEEE International Conference on*. IEEE, 2008, pp. 359–363.

[109] F. Kargl, P. Papadimitratos, L. Buttyan, M. Muter, E. Schoch, B. Wiedersheim, T.-V. V. Thong, G. Calandriello, A. Held, and A. Kung, "Secure vehicular communication systems: implementation, performance, and research challenges," *Communications Magazine, IEEE*, vol. 46, no. 11, pp. 110–118, 2008.

[110] A. Serageldin, H. Alturkostani, and A. Krings, "On the reliability of dsrc safety applications: A case of jamming," in *2013 International Conference on Connected Vehicles and Expo (ICCVE)*, Dec 2013, pp. 501–506.

[111] A. Iyer, A. Kherani, A. Rao, and A. Karnik, "Secure V2V communications: Performance impact of computational overheads," in *INFOCOM Workshops 2008, IEEE*. IEEE, 2008, pp. 1–6.

[112] A. Rao, A. Sangwan, A. Kherani, A. Varghese, B. Bellur, and R. Shorey, "Secure V2V communication with certificate revocations," in *2007 Mobile Networking for Vehicular Environments*. IEEE, 2007, pp. 127–132.

[113] A. Mondal and S. Mitra, "Detection and revocation of misbehaving vehicles from VANET," in *Advanced Computer and Communication Engineering Technology*. Springer, 2015, pp. 767–777.

[114] I. Foster, A. Prudhomme, K. Koscher, and S. Savage, "Fast and vulnerable: a story of telematic failures," in *9th USENIX Workshop on Offensive Technologies (WOOT 15)*, 2015.

[115] J. A. Fax and R. M. Murray, "Information flow and cooperative control of vehicle formations," *Automatic Control, IEEE Transactions on*, vol. 49, no. 9, pp. 1465–1476, 2004.

[116] J. Santa, A. F. Gómez-Skarmeta, and M. Sánchez-Artigas, "Architecture and evaluation of a unified V2V and V2I communication system based on cellular networks," *Computer Communications*, vol. 31, no. 12, pp. 2850–2861, 2008.

[117] S. V. Bana and P. Varaiya, "Space division multiple access (SDMA) for robust ad hoc vehicle communication networks," in *Intelligent Transportation Systems, 2001. Proceedings. 2001 IEEE.* IEEE, 2001, pp. 962–967.

[118] W. Cho, K.-S. S. Han, H. K. Choi, and H. S. Oh, "Realization of anti-collision warning application using V2V communication," in *Proceedings of the IEEE Vehicular Networking Conference (VNC 2009)*, 2009, pp. 1–5.

[119] H. Conceição, L. Damas, M. Ferreira, and J. a. Barros, "Large-scale simulation of V2V environments," in *Proceedings of the 2008 ACM symposium on Applied computing.* ACM, 2008, pp. 28–33.

[120] X. Yang, J. Liu, N. H. Vaidya, and F. Zhao, "A vehicle-to-vehicle communication protocol for cooperative collision warning," in *Mobile and Ubiquitous Systems: Networking and Services, 2004. MOBIQUITOUS 2004. The First Annual International Conference on.* IEEE, 2004, pp. 114–123.

[121] C.-M. M. Huang, S.-Y. Y. Lin, C.-C. C. Yang, and C.-H. H. Chou, "A collision pre-warning algorithm based on V2V communication," in *Ubiquitous Information Technologies & Applications, 2009. ICUT'09. Proceedings of the 4th International Conference on.* IEEE, 2009, pp. 1–6.

[122] M. Sepulcre and J. Gozalvez, "Experimental evaluation of cooperative active safety applications based on V2V communications," in *Proceedings of the ninth ACM international workshop on Vehicular inter-networking, systems, and applications.* ACM, 2012, pp. 13–20.

[123] S. Kato, S. Tsugawa, K. Tokuda, T. Matsui, and H. Fujii, "Vehicle control algorithms for cooperative driving with automated vehicles and intervehicle communications," *IEEE Trans. Intell. Transport. Syst.*, vol. 3, no. 3, pp. 155–161, September 2002.

[124] H. Boeglen, B. Hilt, P. Lorenz, J. Ledy, A.-M. M. Poussard, and R. Vauzelle, "A survey of V2V channel modeling for vanet simulations," in *Wireless On-Demand Network Systems and Services (WONS), 2011 Eighth International Conference on.* IEEE, 2011, pp. 117–123.

[125] R. Ding and Q.-A. A. Zeng, "A clustering-based multi-channel vehicle-to-vehicle (v2v) communication system," in *Ubiquitous and Future Networks, 2009. ICUFN 2009. First International Conference on.* IEEE, 2009, pp. 83–88.

[126] A. Greenberg. (2015, July) Hackers remotely kill jeep on highway. WIRED. [Online]. Available: http://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/

[127] B. Turkus. (2013, August) How to hack a buick regal with carknow. Autoblog. [Online]. Available: http://www.autoblog.com/2013/08/08/how-to-hack-a-buick-regal-with-carknow/

[128] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno, "Comprehensive experimental analyses of automotive attack surfaces." in *USENIX Security Symposium*. San Francisco, 2011.

[129] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, and H. Shacham, "Experimental security analysis of a modern automobile," in *Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE, 2010, pp. 447–462.

[130] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," *Black Hat USA*, 2015.

[131] R. Currie, "Developments in car hacking," SANS Institute, Tech. Rep., 12 2015.

[132] S. Woo, H. J. Jo, and D. H. Lee, "A practical wireless attack on the connected car and security protocol for in-vehicle CAN," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 16, no. 2, pp. 993–1006, 2015.

[133] SAE International, "Cybersecurity guidebook for cyber-physical vehicle systems," SAE International, Tech. Rep., January 2016.

[134] (2012, September) Delphi & Michigan Department of Transportation. [Online]. Available: https://www.michigan.gov/documents/mdot/09-27-2012_Connected_Vehicle_Technology_-_Industry_Delphi_Study_401329_7.pdf

[135] Z. H. Mir and F. Filali, "LTE and IEEE 802.11 p for vehicular networking: a performance evaluation," *EURASIP Journal on Wireless Communications and Networking*, vol. 2014, no. 1, pp. 1–15, 2014.

[136] J. A. Olivera, I. Cortázar, C. Pinart, A. L. Santos, and I. Lequerica, "VANBA: a simple handover mechanism for transparent, always-on V2V communications," in *Vehicular Technology Conference, 2009. VTC Spring 2009. IEEE 69th.* IEEE, 2009, pp. 1–5.

[137] C.-M. Huang and S.-Y. Lin, "An early collision warning algorithm for vehicles based on V2V communication," *Int. J. Commun. Syst.*, vol. 25, no. 6, pp. 779–795, April 2011.

[138] E. Loveday. (2014) Tesla signs multi-year high-speed wireless connectivity deal with at&t. Inside EVs. [Online]. Available: http://insideevs.com/tesla-signs-multi-year-high-speed-wireless-connectivity-deal-with-att/

[139] (2016) AWS | Auto Scaling. Amazon Web Services. [Online]. Available: https://aws.amazon.com/autoscaling/

[140] J. Gozálvez, M. Sepulcre, and R. Bauza, "IEEE 802.11 p vehicle to infrastructure communications in urban environments," *Communications Magazine, IEEE*, vol. 50, no. 5, pp. 176–183, 2012.

[141] O. Trullols, M. Fiore, C. Casetti, C. F. Chiasserini, and J. M. Barcelo Ordinas, "Planning roadside infrastructure for information dissemination in intelligent transportation systems," *Computer Communications*, vol. 33, no. 4, pp. 432–442, March 2010.

[142] (2013, March) V2V penetration in new vehicles to reach 62% by 2027. ABI Research. [Online]. Available: https://www.abiresearch.com/press/v2v-penetration-in-new-vehicles-to-reach-62-by-202/

[143] (2014, February) Strategic analysis of the european market for V2V and V2I communication systems. Frost and Sullivan. [Online]. Available: http://www.frost.com/c/10077/sublib/display-report.do?id=MA29-01-00-00-00

[144] S. Derikx, M. de Reuver, and M. Kroesen, "Can privacy concerns for insurance of connected cars be compensated?" *Electronic Markets*, vol. 26, no. 1, pp. 73–81, 2016.

[145] C. Woodyard. (2013, March) Your car may be invading your privacy. USA Today. [Online]. Available: http://www.usatoday.com/story/money/cars/2013/03/24/car-spying-edr-data-privacy/1991751/

[146] D. I. S. Fellowship, "Automated vehicles for a sustainable city," Ph.D. dissertation, School of Natural Resources & Environment Dow Interdisciplinary Sustainability Fellowship, University of Michigan, Ann Arbor, 2014.

[147] H.-S. . S. Shin, M. Callow, S. Dadvar, Y.-J. . J. Lee, and Z. A. Farkas, "User acceptance and willingness to pay for connected vehicle technologies: Adaptive choice-based conjoint analysis," *Transportation Research Record: Journal of the Transportation Research Board*, no. 2531, pp. 54–62, 2015.

[148] (2016) Secretary Foxx Unveils President Obama's FY17 Budget Proposal of Nearly $4 Billion for Automated Vehicles and Announces DOT Initiatives to Accelerate Vehicle Safety Innovations. US Department of Transportation. [Online]. Available: https://www.transportation.gov/briefing-room/secretary-foxx-unveils-president-obama%E2%80%99s-fy17-budget-proposal-nearly-4-billion

[149] I. M. Ehrenberg, S. E. Sarma, and B.-I. Wu, "Fully conformal fss via rapid 3d prototyping," in *Antennas and Propagation Society International Symposium (APSURSI), 2012 IEEE*. IEEE, 2012, pp. 1–2.

[150] J. E. Siegel, "System and method for providing predictive software upgrades," U.S. Patent 9,086,941, July, 2015. [Online]. Available: http://www.google.com/patents/US9086941

[151] L. Lin and J. A. Misener, "Message sets for vehicular communications," in *Vehicular ad hoc Networks*. Springer, 2015, pp. 123–163.

[152] A. M. Vegni and T. D. Little, "Hybrid vehicular communications based on V2V-V2I protocol switching," *International Journal of Vehicle Information and Communication Systems*, vol. 2, no. 3-4, pp. 213–231, 2011.

[153] B. Ducourthial, Y. Khaled, and M. Shawky, "Conditional transmissions: Performance study of a new communication strategy in vanet," *Vehicular Technology, IEEE Transactions on*, vol. 56, no. 6, pp. 3348–3357, 2007.

[154] B. Xu, O. Wolfson, and H. J. Cho, "Monitoring neighboring vehicles for safety via V2V communication," in *Vehicular Electronics and Safety (ICVES), 2011 IEEE International Conference on*. IEEE, 2011, pp. 280–285.

[155] J. Miller, "Freesim–a free real-time V2V and V2I freeway traffic simulator," *IEEE Intelligent Transportation Systems Society Newsletter*, 2007.

[156] A. Shaout, D. Colella, and S. Awad, "Advanced driver assistance systems-past, present and future," in *Computer Engineering Conference (ICENCO), 2011 Seventh International*. IEEE, 2011, pp. 72–82.

[157] S. Moon and K. Yi, "Human driving data-based design of a vehicle adaptive cruise control algorithm," *Vehicle System Dynamics*, vol. 46, no. 8, pp. 661–690, 2008.

[158] G. Calandriello, P. Papadimitratos, J.-P. P. Hubaux, and A. Lioy, "Efficient and robust pseudonymous authentication in vanet," in *Proceedings of the fourth ACM international workshop on Vehicular ad hoc networks*. ACM, 2007, pp. 19–28.

[159] S. Mayer and J. Siegel, "Conversations with connected vehicles," in *Internet of Things (IoT), 2015 5th International Conference on the*. IEEE, 2015, pp. 38–44.

[160] R. Bishop, "Intelligent vehicle applications worldwide," *Intelligent Systems and their Applications, IEEE*, vol. 15, no. 1, pp. 78–81, 2000.

[161] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (IoT): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[162] (2015, July) HomeKit market held back by Apple's high encryption demands - report. Apple Insider. [Online]. Available: http://appleinsider.com/articles/15/07/22/homekit-market-held-back-by-apples-high-encryption-demands---report

[163] C. D. Marsaon, "IAB Releases Guidelines for Internet-of-Things Developers," *IETF Journal*, vol. 11, no. 1, pp. 6–8, July 2015.

[164] D. Guinard, V. Trifa, F. Mattern, and E. Wilde, "From the Internet of things to the web of things: Resource-oriented architecture and best practices," in *Architecting the Internet of Things*. Springer, 2011, pp. 97–129.

[165] F. Li, M. Vogler, M. Claessens, and S. Dustdar, "Efficient and scalable IoT service delivery on cloud," in *Cloud Computing (CLOUD), 2013 IEEE Sixth International Conference on*. IEEE, 2013, pp. 740–747.

[166] S. Ratnasamy, D. Estrin, R. Govindan, B. Karp, S. Shenker, L. Yin, and F. Yu, "Data-centric storage in sensornets," *Submitted to SIGCOMM*, 2002.

[167] X. Liu, Q. Wang, W. He, M. Caccamo, and L. Sha, "Optimal real-time sampling rate assignment for wireless sensor networks," *ACM Trans. Sen. Netw.*, vol. 2, no. 2, pp. 263–295, May 2006. [Online]. Available: http://doi.acm.org/10.1145/1149283.1149288

[168] W. Shu, X. Liu, Z. Gu, and S. Gopalakrishnan, "Optimal sampling rate assignment with dynamic route selection for real-time wireless sensor networks," in *Real-Time Systems Symposium, 2008*. IEEE, 2008, pp. 431–441.

[169] S. Adlakha, B. Sinopoli, and A. Goldsmith, "Optimal sensing rate for estimation over shared communication links," in *American Control Conference, 2007. ACC'07*. IEEE, 2007, pp. 5043–5045.

[170] L. Hu, Z. Zhang, F. Wang, and K. Zhao, "Optimization of the deployment of temperature nodes based on linear programing in the Internet of things," *Tsinghua Science and Technology*, vol. 18, no. 3, pp. 250–258, 2013.

[171] A. Jain, E. Y. Chang, and Y.-F. F. Wang, "Adaptive stream resource management using kalman filters," in *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*. ACM, 2004, pp. 11–22.

[172] A. Jain and E. Y. Chang, "Adaptive sampling for sensor networks," in *Proceedings of the 1st international workshop on Data management for sensor networks: in conjunction with VLDB 2004*. ACM, 2004, pp. 10–16.

[173] S. Li, L. Da Xu, and X. Wang, "Compressed sensing signal and data acquisition in wireless sensor networks and Internet of things," *Industrial Informatics, IEEE Transactions on*, vol. 9, no. 4, pp. 2177–2186, 2013.

[174] A. Das and D. Kempe, "Sensor selection for minimizing worst-case prediction error," in *Information Processing in Sensor Networks, 2008. IPSN'08. International Conference on*. IEEE, 2008, pp. 97–108.

[175] Q. Wu, G. Ding, Y. Xu, S. Feng, Z. Du, J. Wang, and K. Long, "Cognitive internet of things: a new paradigm beyond connection," *Internet of Things Journal, IEEE*, vol. 1, no. 2, pp. 129–143, 2014.

[176] D. G. Luenberger, "Observers for multivariable systems," *Automatic Control, IEEE Transactions on*, vol. 11, no. 2, pp. 190–197, 1966.

[177] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[178] B. D. Lightner, "AVR-Based Fuel Consumption Gauge," pp. 59–67, 2005. [Online]. Available: www.lightner.net/lightner/bruce/Lightner-183.pdf

[179] A. L. Oehlerking, "Streetsmart: modeling vehicle fuel consumption with mobile phone sensor data through a participatory sensing framework," Master's thesis, Massachusetts Institute of Technology, 2011.

[180] X. Hu, F. Sun, and Y. Zou, "Estimation of state of charge of a lithium-ion battery pack for electric vehicles using an adaptive luenberger observer. energies 3, 1586–1603 (2010)."

[181] J. D. Powell, N. P. Fekete, and C.-F. F. Chang, "Observer-based air fuel ratio control," *Control Systems, IEEE*, vol. 18, no. 5, pp. 72–83, 1998.

[182] I. D. Landau, R. Lozano, M. M'Saad, and A. Karimi, *Adaptive control*. Springer Berlin, 1998, vol. 51.

[183] K. Veeramachaneni, I. Arnaldo, A. Cuesta-Infante, V. Korrapati, C. Bassias, and K. Li, "ai^2: Training a big data machine to defend," in *Big Data Security on Cloud, IEEE International Conference on*. IEEE, Apr 2016. [Online]. Available: https://people.csail.mit.edu/kalyan/AI2_Paper.pdf

[184] R. E. Cole, "What really happened to toyota?" *MIT Sloan Management Review*, vol. 52, no. 4, p. 29, 2011.

[185] S. Kane, E. Liberman, T. DiViesti, and F. Click, "Toyota sudden unintended acceleration," *Safety Research & Strategies*, 2010.

[186] K. Russell, G. GATES, J. KELLER, and D. WATKINS, "How volkswagen got away with diesel deception," *The New York Times*, 2015.

[187] J. Soble. (2016, April) Mitsubishi admits cheating on fuel-economy tests. The New York Times. [Online]. Available: http://www.nytimes.com/2016/04/21/business/mitsubishi-fuel-economy-tests.html

[188] R. Bhattacharyya, C. Floerkemeier, and S. Sarma, "Low-cost, ubiquitous rfid-tag-antenna-based sensing," *Proceedings of the IEEE*, vol. 98, no. 9, pp. 1593–1600, 2010.

[189] R. Bhattacharyya, C. Floerkemeier, S. Sarma, and D. Deavours, "Rfid tag antenna based temperature sensing in the frequency domain," in *RFID (RFID), 2011 IEEE International Conference on*. IEEE, 2011, pp. 70–77.

[190] S. Arthur, H. N. Breed, and C. Schmitt-Luehmann, "Shifting car makeup shakes up OEM status quo: Software strength is critical," *Published online*, 2003.

[191] G. Bilchev, D. Marston, N. Hristov, E. Peytchev, and N. Wall, "Traffimaticsintelligent co-operative vehicle highway systems," in *Intelligent Spaces*. Springer, 2006, pp. 93–108.

[192] S. C. Albulescu, A. A. Pucărea, and O. Dascălu, "The automotive industry in a new tehnological era," *SEA-Practical Application of Science*, no. 5, pp. 99–106, 2014.

[193] M. Barth, G. Scora, and T. Younglove, "Intelligent off-board management of vehicle operating parameters," in *Intelligent Transportation Systems, 2003. Proceedings. 2003 IEEE*, vol. 1. IEEE, 2003, pp. 352–357.

[194] R. W. Cox, "Local Area Network Technology Applied to Automotive Electronic Communications," *Industrial Electronics, IEEE Transactions on*, no. 4, pp. 327–333, 1985.

[195] R. K. Jurgen, "Coming from Detroit: Networks on wheels: Multiplexed wiring in new models promises cost-effective control and accurate diagnostics, as well as fewer wires and less redundancy," *Spectrum, IEEE*, vol. 23, no. 6, pp. 53–59, 1986.

[196] "4G LTE cost efficiency infographic," Alcatel-Lucent. [Online]. Available: https://www.alcatel-lucent.com/solutions/4g-lte/cost-efficiency-infographic

[197] Z. Epstein, "Verizon shared data plans: Analysis: Costs," BGR. [Online]. Available: http://bgr.com/2013/01/23/verizon-shared-data-plans-analysis-303240/

[198] K. Han, A. Weimerskirch, and K. G. Shin, "Automotive cybersecurity for in-vehicle communication," *IQT Quarterly*, vol. 6, no. 1, 2014. [Online]. Available: https://kabru.eecs.umich.edu/papers/publications/2014/IQT%20Quarterly_Summer%202014_Han%20et%20al.pdf

[199] United States Environmental Protection Agency, "EPA Decision Document: Mercedes-Benz Off-cycle Credits for MYs 2012-2016," United States Environmental Protection Agency, Tech. Rep., September 2014. [Online]. Available: http://www3.epa.gov/otaq/regs/ld-hwy/greenhouse/documents/420r14025.pdf

[200] D. Edmunds. Do stop-start systems really save fuel? Edmunds. [Online]. Available: http://www.edmunds.com/car-reviews/features/do-stop-start-systems-really-save-fuel.html

[201] D. Erb, "Optimizing hybrid vehicles: Battery pack design, energy management, and collaborative learning," Ph.D. dissertation, Massachusetts Institute of Technology, June 2016.

[202] (2015) BMW X1 Owners Manual. BMW. [Online]. Available: http://www.bmwusa.com/Standard/Content/Owner/OwnersManualVideos.aspx?namodelcode=16XB

[203] L. Xu, L. Y. Wang, G. Yin, and H. Zhang, "Coordinated control and communication for enhanced safety of highway vehicle platoons," in *Connected Vehicles and Expo (ICCVE), 2013 International Conference on*. IEEE, 2013, pp. 43–47.

[204] C.-H. H. Lee, C.-M. M. Huang, C.-C. C. Yang, and T.-H. H. Wang, "A cooperative video streaming system over the integrated cellular and DSRC networks," in *Vehicular Technology Conference (VTC Fall), 2011 IEEE*. IEEE, 2011, pp. 1–5.

[205] T. ElBatt, S. K. Goel, G. Holland, H. Krishnan, and J. Parikh, "Cooperative collision warning using dedicated short range wireless communications," in *Proceedings of the 3rd international workshop on Vehicular ad hoc networks*. ACM, 2006, pp. 1–9.

[206] GSMA Intelligence, "Understanding 5G: Perspectives on future technological advancements in mobile," *GSMA Intelligence Understanding 5G*, pp. 3–15, 2014.

[207] Carknow. CarKnow. [Online]. Available: http://www.carknow.me

[208] J. E. Siegel, "CloudThink and the Avacar: embedded design to create virtual vehicles for cloud-based informatics, telematics, and infotainment," Master's thesis, Massachusetts Institute of Technology, 2013.

[209] M. C. González, C. A. Hidalgo, and A.-L. L. Barabási, "Understanding individual human mobility patterns." *Nature*, vol. 453, no. 7196, pp. 779–82, June 2008.

[210] S. Rogers, P. Langley, and C. Wilson, "Mining gps data to augment road models," in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1999, pp. 104–113.

[211] W. Shi and Y. Liu, "Real-time urban traffic monitoring with global positioning system-equipped vehicles," *Intelligent Transport Systems, IET*, vol. 4, no. 2, pp. 113–120, 2010.

[212] R. Smith, S. Shahidinejad, D. Blair, and E. L. Bibeau, "Characterization of urban commuter driving profiles to optimize battery size in light-duty plug-in electric vehicles," *Transportation Research Part D: Transport and Environment*, vol. 16, no. 3, pp. 218–224, 2011.

[213] T. Litman, "Distance-based vehicle insurance as a TDM strategy," *Transportation Quarterly*, vol. 51, pp. 119–137, 1997.

[214] T. Toledo and T. Lotan, "In-vehicle data recorder for evaluation of driving behavior and safety," *Transportation Research Record: Journal of the Transportation Research Board*, no. 1953, pp. 112–119, 2006.

[215] M. Mesgarpour, D. Landa-Silva, and I. Dickinson, "Overview of telematics-based prognostics and health management systems for commercial vehicles," in *Activities of Transport Telematics*. Springer, 2013, pp. 123–130.

[216] S. Duri, M. Gruteser, X. Liu, P. Moskowitz, R. Perez, M. Singh, and J.-M. M. Tang, "Framework for security and privacy in automotive telematics," in *Proceedings of the 2nd international workshop on Mobile commerce*. ACM, 2002, pp. 25–32.

[217] D. Guinard, I. Ion, and S. Mayer, "In search of an Internet of things service architecture: Rest or ws-*? a developers?' perspective," in *Mobile and Ubiquitous Systems: Computing, Networking, and Services*. Springer, 2011, pp. 326–337.

[218] S. Mayer, N. Inhelder, R. Verborgh, R. V. de Walle, and F. Mattern, "Configuration of Smart Environments Made Simple – Combining Visual Modeling with Semantic Metadata and Reasoning," in *Proceedings of the 4th International Conference on the Internet of Things (IoT)*, 2014.

[219] S. Mayer, Y. N. Hassan, and G. Sörös, "A Magic Lens for Revealing Device Interactions in Smart Environments," in *Proceedings of the SIGGRAPH Asia 2014 Symposium on Mobile Graphics and Interactive Applications (MGIA)*, 2014.

[220] F. Dötzer, "Privacy Issues in Vehicular Ad Hoc Networks," in *Privacy Enhancing Technologies*, ser. Lecture Notes in Computer Science, G. Danezis and D. Martin, Eds. Springer, 2006, vol. 3856, pp. 197–209.

[221] M. Mesgarpour, D. Landa-Silva, and I. Dickinson, "Overview of telematics-based prognostics and health management systems for commercial vehicles," in *Activities of Transport Telematics*. Springer, 2013, pp. 123–130.

[222] B. Jakoby, M. Scherer, M. Buskies, and H. Eisenschmid, "An automotive engine oil viscosity sensor," *Sensors Journal, IEEE*, vol. 3, no. 5, pp. 562–568, 2003.

[223] A. Agoston, C. Ötsch, and B. Jakoby, "Viscosity sensors for engine oil condition monitoringapplication and interpretation of results," *Sensors and Actuators A: Physical*, vol. 121, no. 2, pp. 327–332, 2005.

[224] J. Zhu, D. He, and E. Bechhoefer, "Survey of lubrication oil condition monitoring, diagnostics, and prognostics techniques and systems," *Journal of Chemical Science and Technology*, vol. 2, no. 3, pp. 100–115, 2013.

[225] A. Mujahid and F. L. Dickert, "Monitoring automotive oil degradation: analytical tools and onboard sensing technologies," *Analytical and bioanalytical chemistry*, vol. 404, no. 4, pp. 1197–1209, 2012.

[226] P. Greening, "European vehicle emission legislation - present and future," *Topics in Catalysis*, vol. 16, no. 1-4, pp. 5–13, 2001.

[227] Lu, Jianbo and Filev, Dimitar and Johnson, Leonard, "Real-time Tire Imbalance Detection Using ABS Wheel Speed Sensors," *SAE Internation Journal of Materials and Manufacturing*, Apr 2011. [Online]. Available: http://dx.doi.org/10.4271/2011-01-0981

[228] PrazŠnowski, Krzysztof and Brol, Sebastian and Augustynowicz, Andrzej, "Identification of Static Unbalance Wheel of Passenger Car Carried out on a Road," *Solid State Phenomena*, vol. 214, pp. 48–57, February 2014. [Online]. Available: http://dx.doi.org/10.4028/www.scientific.net/SSP.214.48

[229] M. R. Chauhan, G. Kotwal, and A. Majge, "Numerical simulation of tire and wheel assembly impact test using finite element method," *Symposium on International Automotive Technology 2015*, Jan 2015. [Online]. Available: http://dx.doi.org/10.4271/2015-26-0186

[230] J. W. Hume, "Method of Balancing Wheels," US Patent US2 052 295, 1936. [Online]. Available: https://www.google.com/patents/US2052295

[231] I. A. Craighead, "Sensing tyre pressure, damper condition and wheel balance from vibration measurements," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 211, no. 4, pp. 257–265, Jan 1997. [Online]. Available: http://dx.doi.org/10.1243/0954407971526416

[232] B. Wojdyla. (2014, January) How Badly Can Potholes Damage Your Car? Popular Mechanics. [Online]. Available: http://www.popularmechanics.com/cars/how-to/a9993/how-badly-can-potholes-damage-your-car-16326605/

[233] Paul Parker, "Balance correction system with on-car runout device," USA Patent US20 030 041 666 A1, March 6, 2003.

[234] K.L Oblizajek, C.T Wright, J.D Sopoci, "On road corner dynamic balancing for vehicle wheels," USA Patent US6 714 858 B2, March 30, 2004.

[235] G.A. MacDonald, "A review of low cost accelerometers for vehicle dynamics," *Sensors and Actuators A: Physical* , vol. 21, no. 1-3, pp. 303 – 307, 1990, proceedings of the 5th International Conference on Solid-State Sensors and Actuators and Eurosensors {III}. [Online]. Available: http://www.sciencedirect.com/science/article/pii/092442479085060H

[236] I. A Craighead, "Sensing tyre pressure, damper condition and wheel balance from vibration measurements," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 211, no. 4, pp. 257–265, 1997. [Online]. Available: http://pid.sagepub.com/content/211/4/257.abstract

[237] Kiran R. Magiawala, Carol A. Eberhard, George W. McIver, Barry Dunbridge, Thomas A. Zimmerman, "System and method for monitoring vehicle conditions affecting tires," USA Patent US6 278 361 B1, August 21, 2001. [Online]. Available: http://www.google.com/patents/US6278361

[238] Augustynowicz, A. and Praznowski, K., "Use of Fourier transform for defining diagnostic parameters of car wheel," *Electrodynamic and Mechatronic Systems*, Oct 2011. [Online]. Available: http://dx.doi.org/10.1109/SCE.2011.6092126

[239] Hideki Ohashi, Hiroyuki Kawai, Hiroyoshi Kojima, Katsuhiro Asano, Takaji Umeno, Toshiharu Naito, Nobuyoshi Onogi, Yuuichi Inoue, "Device for estimating air pressure of tire from vibration components of vehicle wheel speed," USA Patent US5 826 207 A, October 20, 1998. [Online]. Available: http://www.google.com/patents/US5826207

[240] Lane, Nicholas and Miluzzo, Emiliano and Lu, Hong and Peebles, Daniel and Choudhury, Tanzeem and Campbell, Andrew, "A survey of mobile phone sensing," *IEEE Commun. Mag.*, vol. 48, no. 9, pp. 140–150, Sep 2010. [Online]. Available: http://dx.doi.org/10.1109/MCOM.2010.5560598

[241] 2015 Impreza Owners Manual - Specifications. Subaru. [Online]. Available: http://techinfo.subaru.com/proxy/84266/pdf/ownerManual/084266_2015_Impreza/MSA5M1513BOMSTIS120314_18.pdf

[242] Nissan. Nissan Versa Sedan Owners Manual. [Online]. Available: https://owners.nissanusa.com/content/techpub/ManualsAndGuides/VersaSedan/2014/2014-VersaSedan-owner-manual.pdf

[243] How safe are worn tires? Consumer Reports. [Online]. Available: http://www.consumerreports.org/cro/2012/12/how-safe-are-worn-tires/index.htm

[244] Service your car advice: Tire pressure. CarTalk. [Online]. Available: http://www.cartalk.com/content/service-your-car-13

[245] B. Wojdyla. My tires have no holes - ĂŤso why are they going flat? Popular Mechanics. [Online]. Available: http://www.popularmechanics.com/cars/how-to/a3220/my-tires-have-no-holes-so-why-are-they-going-flat-8833288/

[246] R. Sivinski, "Evaluation of the effectiveness of tpms in proper tire pressure maintenance," NHTSA, Tech. Rep., 2012.

[247] Gas mileage tips - keeping your vehicle in shape. FuelEconomy.Gov. [Online]. Available: http://www.fueleconomy.gov/feg/maintain.jsp

[248] Tire pressure survey and test results. NHTSA. [Online]. Available: http://www.nhtsa.gov/cars/rules/rulings/TirePressure/LTPW3.html

[249] Federal motor vehicle safety standards; tire pressure monitoring systems; controls and displays. Federal Register. [Online]. Available: https://www.federalregister.gov/articles/2002/06/05/02-13915/federal-motor-vehicle-safety-standards-tire-pressure-monitoring-systems-controls-and-displays

[250] E. H. Choi, "Tire-related factors in the pre-crash phase," *Report No. DOT HS*, vol. 811, p. 617, 2012.

[251] S. Velupillai and L. Guvenc, "Tire pressure monitoring [applications of control]," *Control Systems, IEEE*, vol. 27, no. 6, pp. 22–25, 2007.

[252] J. Zhang, Q. Liu, and Y. Zhong, "A tire pressure monitoring system based on wireless sensor networks technology," in *MultiMedia and Information Technology, 2008. MMIT'08. International Conference on.* IEEE, 2008, pp. 602–605.

[253] B. Dixon, V. Kalinin, J. Beckley, and R. Lohr, "A second generation in-car tire pressure monitoring system based on wireless passive saw sensors," in *International Frequency Control Symposium and Exposition, 2006 IEEE.* IEEE, 2006, pp. 374–380.

[254] C. R. Carlson and J. C. Gerdes, "Identifying tire pressure variation by nonlinear estimation of longitudinal stiffness and effective radius," in *Proceedings of AVEC 2002 6th International Symposium of Advanced Vehicle Control*, 2002.

[255] V. E. Ersanilli, P. J. Reeve, K. J. Burnham, and P. J. King, "A continuous-time model-based tyre fault detection algorithm utilising a kalman state estimator approach," in *Proceedings of the 7th Workshop on Advanced Control and Diagnosis, Zielona Góra, Poland*, 2009.

[256] N. Persson, F. Gustafsson, and M. Drevö, "Indirect tire pressure monitoring using sensor fusion," *SAE Technical Paper*, 2002.

[257] A. Abdelghaffar, A. Hendy, O. Desouky, Y. Badr, S. Abdulla, and R. Tafreshi, "Effects of different tire pressures on vibrational transmissibility in cars," *Proceedings of the 3rd International Conference on Mechanical Engineering and Mechatronics*, 2014.

[258] C. Lundquist, R. Karlsson, E. Ozkan, and F. Gustafsson, "Tire radii estimation using a marginalized particle filter," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 15, no. 2, pp. 663–672, 2014.

[259] December 2015 traffic volume trends. United States Department of Transportation, Federal Highway Administration. [Online]. Available: https://www.fhwa.dot.gov/policyinformation/travel_monitoring/15dectvt/

[260] D. Schrank, B. Eisele, T. Lomax, and J. Bak, "2015 urban mobility scorecard," *Texas A&M Transportation Institue & INRIX*, 2015. [Online]. Available: http://d2dtl5nnlpfr0r.cloudfront.net/tti.tamu.edu/documents/mobility-scorecard-2015.pdf

[261] IHS Inc. Aging vehicle fleet continues to create new opportunity for automotive aftermarket, ihs says. IHS Inc. [Online]. Available: http://press.ihs.com/press-release/automotive/aging-vehicle-fleet-continues-create-new-opportunity-automotive-aftermarket

[262] J. Merkisz, P. Bogus, and R. Grzeszczyk, "Overview of engine misfire detection methods used in on board diagnostics," *Journal of Kones, Combustion Engines*, vol. 8, no. 1-2, pp. 326–341, 2001.

[263] Tech feature: Detecting misfires in OBD II engines. Babcox Underhood Service. [Online]. Available: http://www.underhoodservice.com/tech-feature-detecting-misfires-in-obd-ii-engines/

[264] Z. Zhang, N. Saiki, H. Toda, T. Imamura, and T. Miyake, "Detection of knocking by wavelet transform using ion current," in *icicic*. IEEE, 2009, pp. 1566–1569.

[265] E. Galloni, "Dynamic knock detection and quantification in a spark ignition engine by means of a pressure based method," *Energy Conversion and Management*, vol. 64, pp. 256–262, 2012.

[266] S. S. Merola and B. M. Vaglieco, "Knock investigation by flame and radical species detection in spark ignition engine for different fuels," *Energy Conversion and Management*, vol. 48, no. 11, pp. 2897–2910, 2007.

[267] S. Vulli, J. F. Dunne, R. Potenza, D. Richardson, and P. King, "Time-frequency analysis of single-point engine-block vibration measurements for multiple excitation-event identification," *Journal of Sound and Vibration*, vol. 321, no. 3, pp. 1129–1143, 2009.

[268] S. N. Dandare, "Multiple fault detection in typical automobile engines: A soft computing approach," *WSEAS Transactions on Signal Processing*, vol. 9, no. 3, pp. 158–166, July 2013.

[269] A. Sujono, "Utilization of microphone sensors and an active filter for the detection and identification of detonation (knock) in a petrol engine," *Modern Applied Science*, vol. 8, no. 6, p. p112, 2014.

[270] P. Kabiri and A. Makinejad, "Using PCA in acoustic emission condition monitoring to detect faults in an automobile engine," in *29th European Conference on Acoustic Emission Testing (EWGAE2010)*, 2011, pp. 8–10.

[271] P. Kabiri and H. Ghaderi, "Automobile independent fault detection based on acoustic emission using wavelet," in *Singapore International NDT Conference & Exposition 2011*, ser. Singapore International NDT Conference & Exposition, Singapore, November 2011.

[272] B. S. Anami and V. B. Pagi, "Multi-stage acoustic fault diagnosis of motorcycles using wavelet packet energy distribution and ann," *in SERSC International*

Journal of Advanced Science and Technology (December 2012), International Journal of Advanced Science and Technology, vol. 49, pp. 47–62, 2012.

[273] P. W. Tse and Y. L. Tse, "On-road mobile phone based automobile safety system with emphasis on engine health evaluation and expert advice," in *Technology Management for Emerging Technologies (PICMET), 2012 Proceedings of PICMET'12:*. IEEE, 2012, pp. 3232–3241.

[274] R. F. Navea and E. Sybingco, "Design and implementation of an acoustic-based car engine fault diagnostic system in the android platform," in *International Research Conference in Higher Education*. Polytechnic University of the Philippines, 2013.

[275] N. D. Lane, P. Georgiev, and L. Qendro, "DeepEar: robust smartphone audio sensing in unconstrained acoustic environments using deep learning," in *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 2015, pp. 283–294.

[276] B. Logan *et al.*, "Mel frequency cepstral coefficients for music modeling." in *ISMIR*, 2000.

[277] I. A. Gheyas and L. S. Smith, "Feature subset selection in large dimensionality domains," *Pattern recognition*, vol. 43, no. 1, pp. 5–13, 2010.

[278] S. Kumar, "Mobile sensor systems for field estimation and "hot spot" identification," Ph.D. dissertation, Massachusetts Institute of Technology, 2014.

[279] H. Liu and H. Motoda, *Feature selection for knowledge discovery and data mining*. Springer Science & Business Media, 2012, vol. 454.

[280] A. D. de Carvalho Jr, M. Rosan, A. Bianchi, and M. Queiroz, "FFT benchmark on Android devices: Java versus JNI," *Nexus*, vol. 7, p. 1, 2013. [Online]. Available: http://compmus.ime.usp.br/sbcm/2013/pt/docs/pos_tec_4.pdf

[281] C. L. Jacoby, Y. S. Jo, J. Jurewicz, G. Pamanes, J. E. Siegel, P. X. Yen, D. S. Dorsch, and A. G. Winter, "Design of a clutchless hybrid transmission for a high-performance vehicle," in *ASME 2015 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers, 2015.