SAFE DRIVING WITH MOBILE DEVICES AND WEARABLES

By

ÇAĞDAŞ KARATAŞ

A Dissertation submitted to the

School of Graduate Studies

Rutgers, The State University of New Jersey

In partial fulfillment of the requirements

For the degree of

Doctor of Philosophy

Graduate Program in Electrical and Computer Engineering

Written under the direction of

Marco Gruteser

and approved by

New Brunswick, New Jersey

January, 2018

ABSTRACT OF THE DISSERTATION

Safe driving with mobile devices and wearables

by ÇAĞDAŞ KARATAŞ

Dissertation Director:

Marco Gruteser

Driver errors due to distracted driving and inadequate awareness of surroundings is an increasing concern which has led to national attention. With increasing amount of information available through the ecosystem of in-vehicle devices like smartphones, wearables, and mobile OS for cars, opportunities to prevent accidents with safety services are now more available than ever before. The safety services are expected to be aware of the driver's actions and car's context, and prevent unsafe driving. Mobile and wearable safety apps differ from conventional built-in automotive safety systems in that they promise low-cost designs that reach a much larger population. On the other hand, detecting the driver's actions and the car's context from mobile sensors is a challenging task due to the dynamic nature of the car and the sensors.

To assist in safer driving, we designed and evaluated fundamental solutions that can be used to detect and prevent driver errors. First, we propose methods to monitor the vehicle, driver's steering, and attention to detect driver errors. The preventive methods are auxiliary services which decrease the need for distracting interactions with the vehicle and phone.

In order to detect driver errors, mobile devices and wearables such as wrist-worn devices and head-mounted devices can be used. The mobile device can be used to monitor the vehicle's movements while the wearables can be used to monitor the driver's head and arm movements. We are particularly interested in the driver's hand and head movements since these type of movements shows the user's attention and the driving interactions with the vehicle, particularly with the steering wheel. Additionally, the detection techniques proposed in this thesis can be also be further utilized to prevent errors by warning the drivers about dangerous conditions.

Preventive techniques propose a mechanism for convenient interaction between the user and the environment. Examples of preventive techniques include easy-to use customizable input interfaces as well as management of notifications to appropriate communication channels and scheduling them to the most convenient times, e.g. when the vehicle stops at a red traffic light.

Through various real-driving scenarios, we show that our approach can detect the wrist-worn device user as driver correctly 98.9% of times and achieve hand on steering wheel detection with a true positive rate around 99% and provide warning of unsafe driving when a driver's hand is off the steering wheel with a true negative rate above 80%. Additionally, the system can achieve accurate steering angle estimation with errors less than 3.4 degrees to facilitate applications such as curve speed warning and understeer/oversteer detection. In the second part of our system, we introduced a novel method to estimate sensor orientation and the vehicle's heading from only a single inertial sensor in a moving vehicle. The method was able to estimate sensor orientation with a mean error of 5.61^o for yaw angle and a 3.73^o for pitch angle while the vehicle is driven in controlled environment and without restrictions. We believe this method can be especially suitable for head tracking applications where the sensor's translational motion is limited. Finally, we proposed a framework that enables users to create customizable printable paper button interfaces that might be used to create shortcuts and reduce mobile-device usage related distractions. Our experiments indicate that the sensor achieves touch detection accuracy over 99% with up to ten different touch points and over 90% with 15 different touch points.

Acknowledgements

The path on the Ph.D. journey has presented some of the most important challenges in my life. It would have been almost impossible to complete this journey without the people who inspired, guided and most importantly supported me during all these years. Although their support and guidance during my hardest times will never fade away from my memories, I believe this thesis would not have ”the whole story” without acknowledging them here.

Firstly, I would like to thank my graduate advisor, my role model Prof. Marco Gruteser. I have always been awed, inspired and motivated by his vision, endurance against the problems and timely words of encouragement. I am honored and consider myself lucky to have him as the mentor during the most stressful years of my education. His guidance always showed me the right path and opened many critical gates on this journey. His calming friendly demeanor, and an ever-eager readiness to tackle problems head on are qualities that I admire in him the most.

I would like to extend my gratitude to the great advisors of Guardian Angels group, Professors Yingying Chen, Yan Wang, and Richard Martin who also gladly accepted to be the committee members of my dissertation defense. Having them approve the merits of my studies were not wasted and hopefully produced some good work. Special thanks to Professor. Rich Howard who was always there to help when I needed to explore the fields that were beyond my knowledge and for being in my dissertation committee. My sincerest thanks to all the co-authors and colleagues in my publications. It has been a great experience working with them.

Special thanks to Luyang Liu, Hongyu Li, Parul Pandey, Hariharasudhan Viswanathan who were always willing to give a hand when I needed and share the burden as colleagues and furthermore as true friends. This list would never be complete without thanking

Mohammad Hajimirsadeghi, Bin Cheng, Viet Nguyen, Gorkem Kar, Shubham Jain, Ines Ugalde, Musaab Alaziz, Ali Rostami, and Dragoslav Stojadinovic.

I would also like to thank all WINLAB members for their help in ideas, experiments, logistics, and feedbacks, including but not limited to Ivan Seskar, Dipankar Raychaudhuri and Rutgers Electrical Engineering Department. I believe this list could be a lot longer, if I had a better memory.

Finally, I would like to thank my friends and my family who always encouraged me and were able to tolerate for the stress I had reflected onto them. I would like to thank Ken Lipman for being the greatest roommate that anyone can have when moved to a new country; my girlfriend Güşah Şenol, for correcting all ”the articles” in my publications and for being ”the one”; My sister, Zeynep Karataş for missing me all these years; My father for giving great tips when I needed; and for My lovely mom, Ayten Gürbüz for being the best friend that a son can have.

Dedication

I would like to dedicate this work to two great beacons who enlightened my way and vision on science. First, Mustafa Kemal Atatürk, the founder of modern secular Turkey who has taken our country from the darkest times, set the positive sciences as the nation's guide, and still being the hope for good days beyond the clouds with his ideas; Oktay Sinanoğlu, who was one of the most important scientist of our age and inspired me to have higher goals then just personal carrier goals and defined the morale of my PhD education in USA.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

In 2005, motor vehicle accidents were the leading cause of death for every age from 3 through 6 and 8 through 34 in the United States. In 2007 alone, motor vehicle accidents caused 41,059 deaths in the United States [1]. According to the World Health Organization, 1.25 million people died in 2013 from motor vehicle accidents, a figure that has plateaued since 2007. Motor vehicle accidents are the number one cause of death among those aged 15-29. The highest death rates from motor vehicle accidents were seen among middle-income countries (90% of world-wide motor vehicle related deaths). In 2012, approximately 50 million injuries resulted from motor vehicle accidents worldwide [2] with 2,362,000 of the injuries occurring in the United States. To date, these accidents have cost an astounding total of $277 billion [3] in the United States alone.

While the widespread use of mobile and wearable devices is often viewed as a distraction [4, 5] that can lead to accidents contributing to nearly one thousand fatalities per year (e.g., [6–9]), it also presents opportunities to prevent accidents through safety services. Statistics show that the primary reason for motor vehicle accidents is driver error with ninety-three percent of traffic accidents being attributed to driver errors. Therefore monitoring drivers actions and surrounding conditions are critical to preventing crashes. In section 1.2 we provide common driver mistakes and how the system can contribute to preventing them. Wearable devices and mobile systems can outperform other existing systems and open new opportunities to detect driver errors and to prevent errors proactively.

Mobile and wearable safety applications differ from conventional built-in automotive safety systems, which are typically constructed as stovepipe systems focusing on a specific risk and employ dependable system techniques such as multiple levels of redundancy, quantifiable guarantees on both the timing and paths of state transitions, and precision sensing. Instead, there is considerable interest in using mobile devices to deliver softer safety services since they promise low-cost designs that reach a much larger population (e.g., [10–12]).

Although vehicle safety systems are designed with the driver in mind and must follow specific guidelines in terms of accuracy, they pose certain limitations. First of all, only the newest and the most expensive car models are accessorized with built-in safety systems. The older and cheaper car models usually lack such systems and cannot be updated. Second, a vehicle's built-in safety system can be out-dated or unavailable since most data is proprietary and only accessible to the vehicle manufacture. Therefore, this type of data offers limited development opportunities to third parties. Similarly, built-in automotive safety systems have limited access to the driver's contextual information, such as actions and physiological state of the driver, creating a disconnect between the system and the user's realistic experience. Finally, these safety systems can only detect the driver's mistakes after they occur which might be too late. However, a driver's contextual information can provide clues about possible driver errors and this data can be used to preemptively avoid an accident. For example, a smartphone can detect whether its user is a driver or a passenger and prioritize notifications from apps and the phone to show only the important ones if the user is a driver and stopped at the traffic light.

In addition to the built-in automotive system, there are third party systems which evaluate or classify driver behavior by fusing the built-in sensors of the vehicle through the OBD-II port with the sensors of the mobile devices to detect dangerous driving events. Such systems aim to compliment the built-in automotive system but also pose similar limitations as they require access to the vehicle bus. Moreover, the sensors found in each vehicle may vary from model to model, lacking a universal mechanism that can be applied across all vehicles.

Wearable and mobile devices, on the other hand, can provide a wide range of opportunities for several reasons. First, the data collected from wearables can be easily processed. In contrast to computer vision based systems, the kinematic data gathered from wearables are lower in dimensions since they only convey information about current rotation and the forces acting on the device. Additionally, this data only needs to be processed when a movement occurs. The other obvious advantage of wearable devices is that they can monitor driver behaviors more accurately since they are usually attached to driver directly. Furthermore, the sensors deployed on wearables are cheap and widely available. Wearable devices are already used by millions of people, and as a consequence the additional cost of wearable based safety systems is quite cheaper than built-in safety systems. Finally, wearables are open to development and the number of wearable apps are increasing drastically each day.

This dissertation proposes a system for safer driving using solely mobile and wearable devices, without reliance on the vehicle's built in automotive system, as a complementary alternative.

1.1 Challenges

The system proposed through this research thesis poses unique challenges along the implementation phase. While this dissertation aims to address these challenges and propose solutions, some of these challenges may not necessarily be overcome.

One of the unique challenges to this study is the limited availability of sensors in the mobile device that provide contextual information. While certain contextual information is readily accessible from the OBD-II port of the vehicle, such as the speed of the vehicle or the steering wheel turning angle, this information may not be readily available on mobile device, therefore, needs to be approximated. Although, estimating this information from the GPS and the kinematic sensors of the mobile devices will require utilizing complex physical models and will be less accurate, such information can still be inferred and approximated by the mobile devices. As an example, if the system needs to know only whether the vehicle is moving or stopped, precise speed measurement is not necessary and only accelerometer measurements can be enough.

Another challenge is presented by the ambiguity of the source of the data gathered through the mobile sensors. Since there could be multiple devices present in the vehicle in any given time (i.e. multiple individuals in the vehicle), we must first be able to classify each device user as driver or the passenger in order to utilize the contextual information collected by each device. Such a classification is necessary to determine whether the data collected depicts movements of the user or the vehicle. The use of the wearable device is of great importance for classifying the user as passenger or driver since the hand movements of the driver and the correlating movement of the vehicle would be more easily discerned from the data provided by the wearable device. The classification techniques for the wearable device owner will be explained in Section 4.4.2. On the other hand, even an imprecise location of the mobile device and classification as passenger or driver device may still provide fertile information about the use of the data collected by such device. Some studies [13–16] including our previous work [17] is able to locate the mobile device in the vehicle. However, they rely on additional sensors or equipment and therefore are not in the scope of this thesis.

The data collected from mobile sensors depends in certain ways on the positioning of the mobile device in the vehicle. The pose-dependency of the data from mobile sensors also presents a challenge in this proposed system. The direction of the kinematic measurement vectors depend on the rotation of the sensors. Therefore, the data must first be taken to a coordinate frame that does not depend on the rotation of the sensors. The coordinate frame transformations from mobile device coordinate frame to another coordinate frame requires knowledge of the mobile sensor's rotation. We address this issue in Section 4.4.1.

A fourth challenge is presented by the dynamic environment of a moving vehicle. Although there are many activity recognition studies that utilizes the kinematic sensors employed in mobile devices, most of the techniques proposed in the studies cannot be applied directly to a moving vehicle scenario. The kinematic sensory data can be affected by both the movements of the vehicle and the user initiated movements of the mobile device with respect to the vehicle. Consider the accelerometer measurements of a smartwatch worn by a driver who is making a turn. The acceleration measured by the device will be a sum of the linear acceleration of the vehicle, centripetal acceleration due to vehicle's turn, the linear acceleration of the driver's arm and centripetal acceleration due to the steering wheel turn. Combined with the aforementioned pose dependency challenge of estimating the hand movement, the movement of the steering wheel is not a trivial task and must be overcome to obtain proper results. This challenge is addressed with the proposed methods in Section 4.4.4.

Last but not least, sensors are prone to picking up noise due to the unpredictable environment they are located in as well as the noise they generate on their own. Minute errors in accelerometer readings can cause greater errors in speed and position estimations when integrated. Similarly, electromagnetic noises induced by the engine or the ferromagnetic materials in the vehicle will cause inaccuracy when calculating rotation. We propose a novel approach to estimate electromagnetic noise in Chapter 5

1.2 The reasons for traffic accidents

In order to prevent accidents, we first need to understand the underlying reasons of traffic accidents. However, detecting the reasons for traffic accidents is not a trivial task which requires in-place investigation and in-depth analyses. Surveys from the drivers who are involved in the traffic accidents are subjective. The drivers may want to hide their faults or do not remember the accident accurately. Therefore, investigations must be carried out by professional experts at the scene right after the traffic accident.

Although there have been some studies on this to date, such as Indiana Tri-Level Study in 1979 and NHTSA 2006 study, they are far from providing in-depth, on-scene investigations of crashes that are necessary to determine the factors related to pre-crash events. Fortunately, in 2008 National Highway Traffic Safety Administration presented a report to the U.S. congress (NMVCCS) based on surveys collected from drivers/surrogates of 5,471 crashes between July 2005 to December 2007 immediately after the accidents. The study also provides on-scene crash causation studies by investigation experts. In this thesis, we aimed to detect and prevent the factors that lead to accidents as noted in the NMVCCS report. The primary reason we choose to utilize this study is that the data is collected right after the crash by an inspector and it is a nation-wide long-term study with many vehicles involved.

Usually there is no single reason leading to a traffic accident. Usually there are a series of mistakes, which contribute to the incident. The NMVCCS report provided the most *critical reason* and also listed other crash *associated factors*. Although the critical reason can be subjective in nature and it is an important element in the sequence of events, it does not imply the assignment of all fault to a vehicle, driver, or environment. However, it is still important to categorize the critical reasons and attribute them to a factor. Critical reasons that are leading to the accident can be attributed to driver, vehicle or environment. Here, environment factors implicate roadway and atmospheric conditions. Accordingly, 5096 of 5471 accidents are attributed to drivers, 130 accidents are attributed to the vehicle, and 135 of accidents attributed to environment; 110 crashes were due to various other reasons and a critical reason could not be assigned.

Since the driver is the most important critical factor, tracking the behavior of the driver and evaluating the driver's decisions can significantly reduce the causalities of motor vehicle accidents. Additionally, improvements in driver mistakes will also affect the associated factors in the crashes where the drivers mistake was not the critical error and consequently may reduce causalities even further.

1.2.1 Critical reasons attributed to drivers

Driver-related critical reasons and their occurrence frequency from the report are given in Table 1.1. The critical reasons can be categorized into recognition errors, decision errors, performance errors, and non-performance errors.

The most frequent three recognition errors- inadequate awareness of surroundings, internal distractions and external distractions- are the cause of more than one third of driver errors. Recognition error is defined as the lack of attention of the driver to what it should be paying attention to. The most common and critical error that is seen is the inadequate awareness of surroundings situation in which the driver did not look, or looked but did not see a specific occurrence. This error can be avoided by head tracking and determining whether the driver made the necessary checks at a critical place, such as checking the side mirrors before changing the lane or checking sides before making a legal right turn. The second most common critical error (11 percent) is the internal distractions situation. Combined with external distractions, distractions is the reason for almost 15 percent of driver errors. Although head tracking system might help detecting when the driver is distracted with an internal equipment, it will not be enough by itself. Consider the scenario when the driver is eating or smoking, although the driver's eyes might be still checking the road, his or her response time will be drastically delayed for an urgent situation. A complimentary arm tracking system is necessary to track driver's attention fully. We introduce methods to track head and arm movements and to warn the driver when necessary in Chapter 5 & 4. Additionally, the prevention part of this thesis proposed in Chapter 6 will further aim to decrease obtrusive interactions through proposed mechanisms.

The next most frequent factor on driver related critical reasons is the speed of driving for the conditions of the road (8.4 percent) and speeding on curves (4.9 percent). Detecting that the driver's speed is too fast for a specific condition or curve requires an extreme condition's occurrence so that the vehicle speed can be tagged as excessive. After conditions and the geographical locations are tagged, these data can be used to warn the drivers when they are close to a safety critical region. Driving too fast for a curve or a specific condition can result in slipping at the curve or not being able to stop efficiently after a brake. We further utilize arm tracking methods introduced in Chapter 4 to estimate steering wheel angle from the arm rotation. Details of steering wheel angle estimation will be given in 4.4.4.

Table 1.1: Critical Reasons for Critical Pre-Crash Event Attributed to Drivers

Critical Reason for Critical Pre-Crash Event		Weighted Percentage %
Recognition Error	inadequate awareness of surroundings	20.3
	Internal distraction	10.7
	External distraction	3.8
	Inattention	3.2
	Other	2.5
Decision Error	Too fast for conditions	8.4
	Too fast for curve	4.9
	False assumption of other's actions	4.5
	Illegal maneuver	3.8
	Misjudgment of gap or other's speed	3.2
	Following too closely	1.5
	Aggressive driving behavior	1.5
	Other	6.2
Performance Error	Overcompensation	4.9
	Poor directional control	4.7
	Other	0.4
	Panic/Freezing	0.3
Non-performance Error	Sleep	3.2
	Heart attack or other physical impairment	2.4
	Other	1.6
Other/unknown driver error		7.9

Chapter 2

Background

There have been active research efforts in reinforcing driving safety by utilizing mobile devices used in vehicles [11, 13–22]. Some prior contributions have been made to detecting whether the mobile device user is a driver or passenger [13–17] which can facilitate many applications aiming to mitigate distracted driving, whereas many works have been done to evaluate or classify driver behaviors by leveraging mobile sensing technologies (including using embedded sensors, cameras and other auxiliary devices (e.g.,OBDII) in mobile devices or vehicles) [18]. In addition, there are studies on solving driving safety issues by detecting distracted driver behaviors [23, 24].

In particular, Yang et al. [13] introduce an acoustic ranging system to identify the location of a smartphone in the vehicle using its audio infrastructure. Wang et al. [14] utilize embedded sensors in a smartphone and a reference point (e.g., an OBD device) to capture vehicle dynamics, which can be further exploited to determine whether the phone is on the left or right side of the vehicle. However, both works rely on access to extra infrastructures in a vehicle, which may not be widely available and thus reduce the chance of adopting the approaches quickly among a large number of users. Some other works [16, 17] distinguish whether the phone is used by a driver or passenger based on the detection of specific movements, such as entry swing, seat-belt fastening, or pedal press using inertial phone sensors. In terms of driving behavior detection, apps, such as iOn-Road [20], Augmented Driving [21], and CarSafe [12], provide lane changing assistance and safe following distance based on the vision of driver or outdoor environment captured by cameras in smartphones. Also, Sober-Drive [25] and CarSafe [12] detects drowsy driving in a similar manner. These approaches do vision studies based on the camera of the phone. In contrast, inertial sensor-based approaches

[11, 18, 19, 22] rely less on specific phone placement and more on motion sensing through phones embedded inertial sensors. Castignani et.al. [18] propose SenseFleet, a driver profile platform that is able to detect risky driving events independently on a mobile device. Chen et al. [19] develop a vehicle steering detection middleware to detect various vehicle maneuvers, including lane changes, turns, and driving on curvy roads. Johnson et al. [11] and Dai et al. [22] both propose driving behavior monitoring systems to track unsafe events based on embedded inertial sensors in the smartphone. However, these approaches have limited accuracy on detectable unsafe driving events, since they only involve smartphones, which more likely capture vehicles dynamics instead of drivers dynamics.

The car manufacturers have been improving the vehicle's design to alleviate driver's distraction by providing simpler driver-vehicle interfaces as well as delegating phone functions to those interfaces. A very common solution is placing some control buttons on the steering wheel. However, as the number of in-vehicle functions increases this approach quickly becomes unfeasible, for economical as well as for practical and ergonomic reasons. Additionally, these buttons are very handy and ease to access, the buttons' functionality is hard-wired and cannot be customized by the driver. Some drivers might disagree on what functions they may prefer to operate via a steering-wheel-mounted button. Furthermore, the driver's preferences might change due to everchanging technology trends on mobile devices. For example, the drivers may not find the control button to switch vehicle radio to AM band as useful as a button that can send short voice clip replies via WhatsApp. Therefore, more customizable vehicle interfaces can reduce driver's distraction by alleviating the urge to reach the mobile phone and the off-the-steering-wheel controls of the vehicle. Consequently, the manufacturers introduced touch screens on the dashboards. However, touch screens cannot be placed on the steering wheel due to safety issues of inflatable airbags and The touchscreens on the dashboard are not as easy-reachable as the buttons on the steering wheel. Adaptive user interfaces are suggested in some projects GIDS [26], CEMVOCAS [27], and COMUNICAR [28]. However, in terms of usability adaptive user interfaces showed a slight improvement over non-adaptive interfaces.

Many companies also introduced voice commands systems such as Ford Sync [29],Lexus Voice Command [30],GM IntelliLink [31] . Maciej and Vollrath [32] show a comparison of manual and speech-based interfaces for text entry tasks while driving and operating different in-vehicle information systems manual text input resulted in a substantial increase of perceived distraction in comparison with voice interaction during all tasks. However, Harbluk, Noy and Eizenman [33] showed that the drivers field of view is significantly scaled down while being distracted by using a hands-free device. Additionally high rate of speech recognition errors and privacy issues which might increase driver's cognitive workload. Kun, Peak and Medenica [34] recorded three measures of driving performance when investigating the effect of different accuracy levels of speech recognition: the lane position, the steering wheel angle, and the velocity of the participants when operating with different systems. They found significantly reduced driving performance for lower speech recognition accuracy levels. They used a high-fidelity driving simulator with a 180 field of view. 20 participants needed to follow a leading vehicle at a constant distance without departing from the lane. Multimodal interfaces that combines speech and steering wheel buttons have attempted to address this issues since the errors from speech recognition might be corrected with user's input from the steering wheel buttons easily.

Some studies focused on reducing the distraction by managing the interaction driver-phone user. The most direct way to eliminate the distraction from drivers' phones is to completely ignore incoming calls and texts. For instance, quite a few smartphone applications, e.g., OneTap [35], AT&T DriveMode [36], and Live2Txt [37], use GPS information to automatically sense when a user is driving and suppress all the notifications that could distract the drive from the road. Instead of using GPS information, *Agent* [38] assumes users are driving when their smartphones are connected to vehicles' Bluetooth systems, and automatically responds a message to people who call or text the drivers with their driving status. There are strict approaches that utilize additional hardware installed in vehicles to completely interfere cellular signals [39, 40] to eliminate distractions from phone usage while driving. Lindqvist and Hong [41] move a step further to discuss how drivers' phones could interact with callers by context-awareness. Negotiator allows caller and callee to negotiate for a good time in the future. Bayesphone uses user's calendar and the user-defined importance of caller to determine whether the user should be interrupted or not. Some studies prefered presenting the caller with information about the probability that the callee is busy by using calendar [42] of the callee or by utilizing the sensors around the callee [43].

However, all these papers mentioned above only consider general notification contexts and does not cover the driver's input aspect especially when the interaction started by the driver.

Chapter 3

Approach

We can classify our approach into two categories, detection and prevention of driver errors. Detecting driver errors by monitoring driver's actions and vehicle's movements and immediately taking necessary actions is the critical part of proposed system. These actions and states can be further utilized to predict critical future errors, and to prevent them by improving driver's interaction with his mobile device, wearables, the vehicle, and the outer world which is the second category of methods provided in this thesis.

The mobile device can be used to monitor vehicle's movements while the wearables can be used to monitor driver's head and arm movements. We are particularly interested in driver's hand and head movements since these type of movements shows user's attention and the driving interactions. Furthermore, these type of movements can be used to detect and prevent the most common types of driver errors. In order to detect the car's movements and driver's hand and head movements, the system will need at least three devices and several widely available standard sensors: a mobile device fixed in the car, a wrist-mounted device, and a head-mounted device. The mobile device can be a smartphone or a tablet with internet and GPS connection to retrieve location and geographical information as well as kinematic sensors: accelerometer, gyroscope, and magnetometer to infer vehicle movement. Since the smartphone is the most common mobile device in this category, we will use smartphone interchangeably for the mobile device throughout this thesis. However, we are going to extend our approach to remove the requirement of having a separate smartphone in Chapter 5. Wrist-mounted sensor can be a activity tracker or a smartwatch, we assume that it will embody at least inertial sensors. Similarly, we are going to use smart watch interchangeably for the wrist-mounted sensor since it is the most common wrist-mounted sensor. The head-mounted device can be any device with kinematic sensors such as smart glass. However, for the head-mounted sensors our system will not depend on any camera or computer vision techniques. We believe this minimal set of sensor requirements can make our approach more implementable for future head-mounted sensors such as a posture corrector [44] or even a futuristic smart-hair clip.

During the drive, the mobile phone will be mostly at a fixed position in the vehicle and will not be moving independently within the vehicle unless the driver interacts with it. Therefore, the data collected from the device's inertial sensors are due to the vehicle's movements only and can be utilized to infer vehicle's movements and its state. On the other hand, wearable devices will move independently within the vehicle as the driver moves his/her arm or head. Therefore, wearable devices' inertial sensors measurements would be due to a combination of the vehicle's movements and the sensor's independent movements. The driver's arm and head movements can be extracted from this data by removing the vehicle's movements inferred from the mobile device as proposed in Chapter 4 or by using a single sensor approach introduced in Chapter 5.

In order to detect driver errors, the driver's and the vehicle's movements need to be evaluated continuously and higher level operations needs to be carried out to detect safety critical behaviors. For example, consider the case when the driver is driving through a curve. From the smartphone's gyroscope data, the system can track the vehicle's movements and detect that the car is turning. From the smart watch and the smartphone data, the system can estimate the driver's hand movements within the car and can use this data to infer the steering wheel movements, since the driver will be turning the steering wheel during the curve. Finally, the steering wheel movement and the vehicle's movement can be compared and the system can detect whether the vehicle is moving different than the driver's intention, i.e., when the car is sliding. In chapter 4 we propose methods to track arm movements of the driver that can be further utilized to track unsafe driving conditions.

Preventing driver errors, on the other hand, is a more comprehensive task in the sense that there are more scenarios to evaluate. The first class of scenarios are based on warning the driver for potential errors. The second class of scenarios prevent errors by easing the driver's interaction with the environment. The simplest driver warning system relies on relating the crowd-sourced driver mistakes with geographic locations and warning the drivers when they are close to that region. However, other contextual information also needs to be incorporated as well as location. Consider the place where the sliding occurs in rainy weather conditions quite often, warning drivers only when the weather is rainy would be quite beneficial. We believe detecting driver errors is critical part of a warning system.

The other very important aspect of preventing driver errors is by easing the drivers interaction and minimizing distractions. In order to minimize interactions, the input channels of the system needs to be redesigned to allow the user to interact with the system without moving his eyes off the road and his hands off the steering wheel. Our input system is composed of three elements. The first element is customizable paper interfaces that can be adhered to a surface on the car such as the steering wheel or the dashboard. These paper interfaces are cheap, flexible and customizable to accommodate user's variety of needs. Although many vehicles provide shortcut buttons on the steering wheel, their functionality is already defined by the manufacturer and usually does not enable the driver to assign new functions based on the driver's needs. However, smartphones have become a gateway and control point for all electronic devices. Additionally, with third party rule-based apps, such as IF or uTasker, users are able to create custom shortcut's for a wide variety of apps and Internet-of-Things (IoT) devices. Our custom input interfaces enable drivers to create shortcuts for a variety of simple tasks and do common tasks without moving their hand off the steering wheel by pressing a touch point on their custom input interface. For example, the driver can assign different touch points to run different media player or radio apps on his phone or to email his wife that he is heading home. Although, these shortcuts are very effective to trigger discrete actions, they are very limited as an analog input. Consider where the driver wants to increase the volume of the music app, increasing the volume by pressing volume + key on the custom input interface can be quite distractive and inefficient task due to the number of key presses required. For this type of analog inputs, we propose gesture based input system that does not require the driver to move his hand off the steering wheel. The approach is based on our capability of tracking the driver's arm's rotational movements and the car's movements. The driver can use the system by rotating his hand over the steering wheel without actually steering the car. This system can detect this type of movement when the driver's arm is making a rotational movement and the car is not making any turn. The rotation degree of the arm can be used as the analog input. A custom touch point on the paper interface can be used to choose what the driver is willing to control and the gesture can be used to input the analog level. For example, the user can choose to navigate the radio channels or to adjust the volume from the paper interface and change the channel or adjust the volume level by rotational gesture. The third class of input system is based on speech commands and is already being developed by mobile operating system providers. Speech command based intelligent personal assistants such as Siri and Google Now are able to understand the user's complex requests and perform common tasks.

Chapter 4

Tracking Driver's arm and Steering Wheel Usage

4.1 Introduction

While the emerging ecosystem of mobile and wearable devices is often viewed as a distraction that can lead to accidents (e.g., [6–9]), it also presents opportunities to prevent accidents through safety services. Mobile and wearable safety apps differ from conventional built-in automotive safety systems, which are typically constructed as stovepipe systems focusing on a specific risk and employ dependable systems techniques such as multiple levels of redundancy, quantifiable guarantees on both the timing and paths of state transitions, and precision sensing. There is considerable interest in using mobile devices to deliver softer safety services, since they promise low-cost designs that reach a much larger population.

Existing mobile solutions have used inertial sensing on smartphones to detect distracted driver behaviors and monitor driving [18, 19]. In order to apply mobile sensing to driver behavior analysis, the devices also need to recognize when their user is driving and distinguish that from being a passenger. To date, this has been addressed through techniques that allow the phone to sense its position inside a vehicle and determine whether it is in the driver area [14, 15, 17]. Such smartphone-based sensing techniques remain limited in accuracy, however. For example, it is challenging to track very gradual transitions across lanes or to determine that a user is driving when the phone is placed on the passenger side of the vehicle. Moreover, increasingly automated and self-driving vehicles will likely let drivers relinquish driving duties for at least some part of a route. Current smartphone-based sensing techniques cannot determine whether a driver is in charge and has the hands on the wheel.

We therefore ask whether the emerging quantity and diversity of wearable devices can be exploited to achieve more accurate sensing and operation independent of the vehicle. Apart from some early results [45], the use of wearables in this domain has so far remained unexplored. Towards this goal, this stufy examines the benefits of wrist-worn inertial data, such as those from smart watches or fitness bands, when combined with phone measurements. Such data is particularly useful for tracking hand movements and can therefore provide information about the driver's vehicle operation—most notably steering wheel usage. To allow more accurate identification of driving, we first develop an inertial detection technique that can distinguish motions when handling steering wheel from other passenger hand movements. After the user is detected as the driver of the vehicle, we detect whether the hand is on the steering wheel based on the hand movements of the driver captured by inertial sensors. We finally extend this technique to also track the turning angle of the steering wheel. This information can allow for more precise vehicle motion tracking than gyroscope data and was previously only available through proprietary interfaces to the vehicle CAN bus.

Tracking the usage of the steering wheel can help monitor driving behaviors and identify unsafe driving. For example, the steering wheel turning angles could be used in lane departure warning system to warn a driver when the vehicle is about to drift across the lane. Moreover, unsafe driving behaviors such as swerving, understeering or oversteering could also be detected based on the steering wheel turning angles and other inertial or speed measurements. Just knowing whether a driver's hands are on the steering wheel is also useful context information that mobile apps can use to minimize distractions to drivers.

The contributions of our work are summarized as follows:

- We explore the use of wearable devices, such as smart watches and wristbands, to track fine-grained driving behaviors including steering wheel usage and angle.

- We introduce machine-learning-based methods and features to classify the wearable device user as the driver or passenger by steering wheel usage motions from other passenger hand movements.

- We design a hand on/off steering wheel usage detection method that relies on hand movements of the driver captured from the wrist-wearable device.

- We propose a steering wheel angle estimation approach by leveraging the wrist rotation caused by the movement of rotating the steering wheel. The proposed approach takes as input the gyroscope reading of a wrist-wearable device and the relationship between the wrist and steering wheel rotations, for real time steering wheel angle estimation.

- We demonstrate through experiments in real driving scenarios that it is feasible to track fine-grained driving behaviors using wrist-wearable devices. Our experiments show 98.9% driver detection accuracy for individual turns, 99% accuracy in detecting steering wheel usage and estimation of turning angles with an average error of 3.4 degrees, when the hand remains on the steering wheel.

4.2 Related Work

In addition to background work introduced in Chapter 2, there are limited number of studies on tracking steering wheel usage. Van and colleagues [24] introduced several models that get driving behaviors from the steering wheel angle. Schmidt et al. [23] present a mathematical model of the steering wheel angle that contributes to predicting lane change maneuvers. However, these approaches need the steering wheel angle data, which is not readily available in mobile devices. [46] used steering-wheel-mounted kinematic sensors to estimate the steering wheel angle. However, this technique still relies on external sensors that need to be deployed and may cause distraction.

In recent years, wearable devices have been exploited for motion estimation in many research works. Vlasic et al. [47] propose a full body motion capture system using inertial sensors. Raiff et al. [48] use a wristband containing a 9-axis inertial measurement unit to capture changes in the orientation of a persons arm, and develop a machine learning pipeline that processes the orientation data to accurately detect smoking gestures. We foresee that wrist-worn wearable devices, such as smartwatches and activity trackers, are particularly useful for fine-grained tracking driving behaviors, because they capture the dynamics directly from user's hand movements. Different from the above works, we propose a low-infrastructure approach to accurately determine steering wheel turning angles based on both smartwatch and smartphone, which can be used as an input for many driving safety applications.

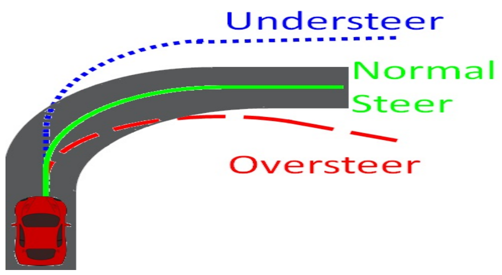
4.3 Applications and Requirements

The observations that can be gathered from mobile and wearable devices open new research opportunities that would be impossible to develop with existing built-in, stovepipe automotive safety systems. The steering wheel usage is one example - it can be used to analyze drivers' behaviors, but also to inspire additional driver safety applications. Although cars equipped with Electronic Stability Control (ESC) could utilize steering wheel angles to determine a driver's intended direction, steering wheel angle information is not usually accessible from the OBD-II port, and therefore not available for third party applications.

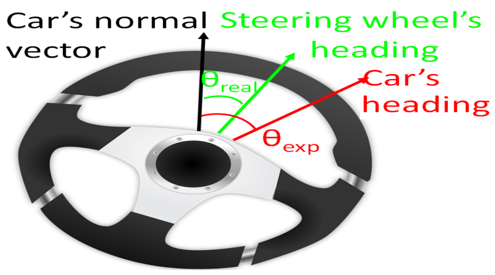
We show that the steering wheel angles can be accurately estimated by leveraging a wearable device on the driver's wrist. The *estimated steering wheel angle* (θ_{est}) can further facilitate various safety and driver behavior monitoring applications, which will be discussed in Section 4.3.1. Although the *real steering wheel angle* (θ_{real}) could be measured more accurately through the car's built-in systems, the accuracy of the estimated angle may still be useful for the applications with lower accuracy requirements such as understeer/oversteer detection. In Section 4.3.2, we calculate the required accuracy in estimated steering wheel angle in order to achieve desired error rates.

4.3.1 Steering Wheel Tracking Applications Scenarios

Detecting Understeer and Oversteer. One of the critical applications of steering wheel tracking is understeer/oversteer detection. In the simplest terms, oversteer is what occurs when a car steers by more than the amount commanded by the driver. Conversely, understeer is what occurs when a car steers less than the amount commanded by the driver due to traction loss as illustrated in Fig 4.1a. In addition to estimated steering wheel angle and real steering wheel angle, we define another term, the *Expected*



(a) Understeer and oversteer are in blue and red. Green line represents driver's intended steering.



(b) The real steering wheel angle and the expected steering wheel angle from car's heading.

Figure 4.1: Car and steering wheel for understeer/oversteer.

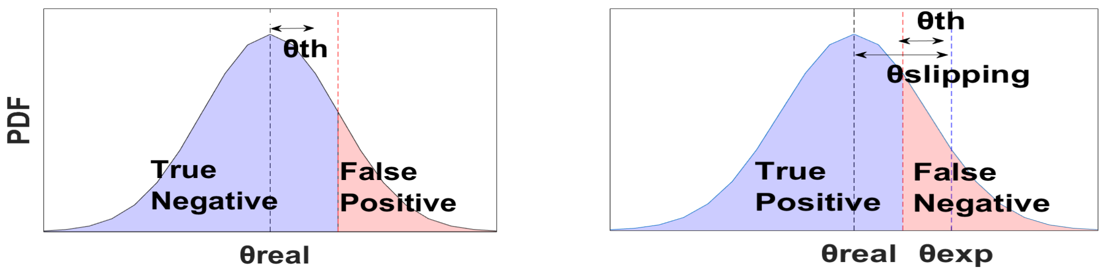
steering wheel angle (θ_{exp}) which is the steering wheel angle that can be calculated based on the car's movement. (θ_{real}) and (θ_{exp}) are illustrated on a steering wheel in Figure 4.1b When there is no understeer or oversteer, the expected and real steering wheel angles should be ideally equal since the car moves as directed by the steering wheel. However, when an understeering or oversteering incident happens, there will be a difference between the two values. Therefore, understeer/oversteer can be detected when this difference exceeds a threshold value (θ_{th}) during the car's turn.

By combining basic circular motion laws and Ackermann steering geometry [49], the expected steering wheel angle can be calculated from centripetal acceleration and angular velocity of the car with following equation:

$$\theta_{exp} = \frac{1}{k} arctan\Big(\frac{2\omega^2}{(2a + d\omega^2)L}\Big), \tag{4.1}$$

where a, ω, d, and L represent centripetal acceleration, angular velocity, width, and length of the car, respectively. Lastly, k is the coefficient that maps car's steering wheel angle to wheel angle.

Curve Speed Warning. Generally, vehicles have over five times higher accident rates on curves than that on straight roadways [50]. Transportation authorities thus took actions, such as placing dangerous curve warning, to prevent drivers from moving through a curve at a dangerous speed. However, there are still many sharp roadway curves without any warning signs or advisory speeds posted. Additionally, accident risks may still exist, even with warning signs, since the warning signs may be overlooked.



(a) No oversteer (b) Oversteer

Figure 4.2: Estimated Steering wheel angle and corresponding True positive, False Positive, False Negative, and True Negative regions are illustrated.

Thus, one example application type that enhances driver safety are Curve Warning Systems (CWS). These assess threat levels for vehicles approaching curves and provide timely driver feedback. The maximum safe speed of a vehicle for an approaching curve is normally affected by both road conditions (e.g., weather conditions) and vehicle behaviors, such as the type of the vehicles (sedan, truck, etc.) and steering wheel rotations. One possible CWS might use crowd-sourcing to determine the maximum safe speed of a vehicle for each upcoming curve by using historical safe turning data sets of other vehicles passing this curve (i.e., without understeering/oversteering happen). The CWS would select the data sets to match the road conditions and vehicle behaviors of the current vehicle to narrow down suitable observational sets and calculate the optimum speed of the vehicle for the curve. The CWS could warn the driver visually if the speed of the vehicle is larger than the calculated optimum speed.

Other applications. Tracking steering wheel usage can lead to many other applications. For example, by comparing estimated and expected steering wheel angles, not only during the turns, but continuously, a car maintenance system can decide whether or not a car needs alignment if there is a persistent offset between estimated and expected steering wheel angles. Also, the steering wheel angle expectation based on online maps for a specific road can be utilized to detect lane changes by comparing the estimated steering wheel angle. The same system can further be useful to detect impaired driving if abnormally frequent lane changes are detected. Similarly, the system can be used to monitor driving styles and report dangerous driving habits. Another interesting application would be gesture-based input system, the driver could interact with the vehicle by rotating his hand on the steering wheel without actually moving it. Such a gesture-based input system can enable drivers to change the radio volume without moving their hand off the steering wheel. However, these type of applications are beyond the scope of this thesis.

4.3.2 Accuracy Requirement of Understeer/Oversteer Detection

We believe that the coarse data can be still useful for a wide range of applications. In this section, we thus analyze the steering wheel angle estimation accuracy needed to deliver various minimums on classification performance for slip detection. We calculate the allowed deviation of estimated steering wheel angle from real steering wheel angle for an example oversteer detection application with performance criteria, namely *minimum slip detection angle* ($MSDA$) and *error rate.* (e_r). However, the similar calculations can be carried out for understeer and lane change detection application scenarios.

These two metrics ensure that the system's error rate will always be less than e_r as long as the degree of slipping is greater than $MSDA$. The metric we choose for the error rate is Equal Error Rate which is the error rate of the system when the probability of *false positives*, false oversteer detections, and that of *false negatives*, missed oversteers, are equal. Therefore in order to calculate the error rate, we need to define rate of false oversteer detections and the rate of missed oversteers first.

For the false positives, we need to consider the case where the car does not slip. The real steering wheel angle and the expected steering will be equal and have the value of θ_{real}. The estimated steering wheel angle can be modeled as a normal distribution with mean θ_{real} and σ_{steer} standard deviation as in Fig. 4.2a. The σ_{steer} is the estimated steering wheel angle requirement that the estimation algorithm needs to achieve the aforementioned guarantees. Any θ_{est} angle that is greater than $\theta_{real} + \theta_{th}$ will result in false positives. In order to calculate the false negatives, we need to consider the case when the car's slipping causes $\theta_{slipping}$ degrees of difference between the real steering wheel angle and the expected steering wheel angle. The estimated the steering wheel angle can still be modeled with the same normal distribution. The false negatives will occur for the values of the estimated steering wheel angle greater than $\theta_{real} + \theta_{slipping} - \theta_{th}$ as

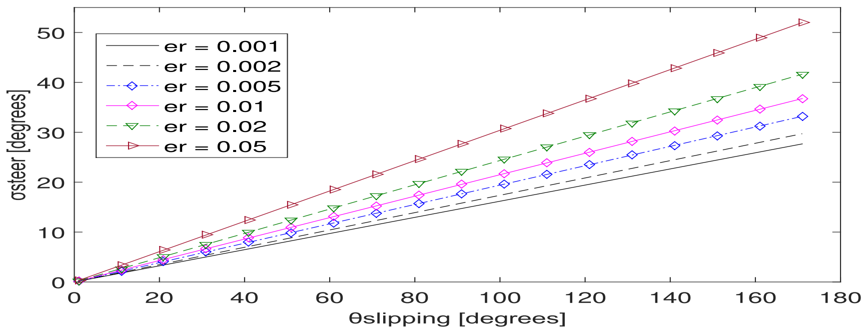


Figure 4.3: The required steering wheel accuracy (σ_{steer}) for given error rate (e_r) and slipping angle ($\theta_{slipping}$) is plotted.

		Error Rate (e_r) [%]						
		0.1	0.2	0.5	1	2	5	10
Minimum Slip Detection Angle (MSDA)[^{o}]	1	0.16	0.17	0.19	0.22	024	0.30	0.39
	5	0.81	0.87	0.97	1.07	1.22	1.52	1.95
	10	1.62	1.74	1.94	2.15	2.43	3.04	3.90
	20	3.24	3.47	3.88	4.30	4.87	6.08	7.80

Table 4.1: The required steering wheel angle accuracy (σ_{steer}) is given for desired error rates (e_r) and Minimum slip detection angles (MSDA)

shown in Fig. 4.2b. From these figures, the σ_{steer} can be calculated for given $\theta_{slipping}$ and e_r can be calculated as follows

$$\sigma_{steer} = \frac{\theta_{slipping}}{2\sqrt{2} * erf^{-}(1 - 2e_r)}, \tag{4.2}$$

where the $erf()$ is the error function. The σ_{steer} for some e_r and $\theta_{slipping}$ are given in Figure 4.3. The figure shows that the smaller $\theta_{slipping}$ values require higher accuracy in steering wheel angle estimation system. Therefore, we need to set a minimum slip detection angle in order to define the upper limit of required accuracy. Some example values of required σ_{steer} are presented in Table 4.1 for different $MSDA$ and e_r requirements.

4.4 System Design

In this section, we first provide an overview of our wearable device based steering and driver tracking system. We then present in detail the proposed, driving detection, hand on/off steering wheel detection, and the steering wheel angle estimation methods.

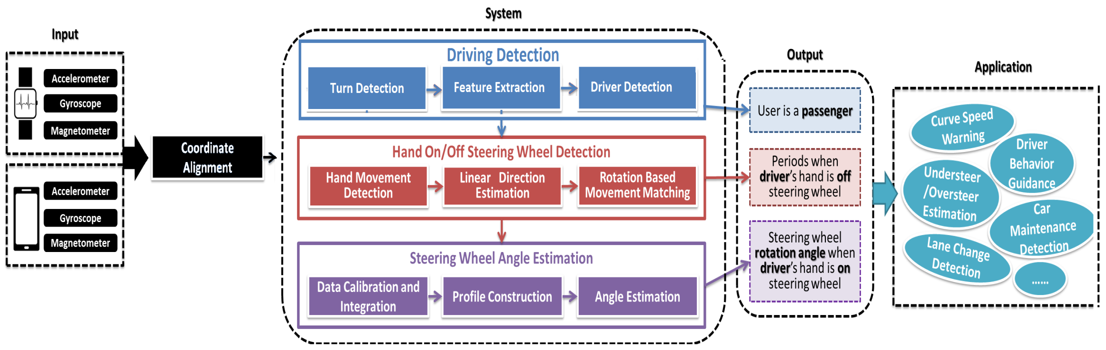

Input
Accelerometer
Gyroscope
Magnetometer
Accelerometer
Gyroscope
Magnetometer
Coordinate Alignment
System
Driving Detection
Turn Detection
Feature Extraction
Driver Detection
Hand On/Off Steering Wheel Detection
Hand Movement Detection
Linear Direction Estimation
Rotation Based Movement Matching
Steering Wheel Angle Estimation
Data Calibration and Integration
Profile Construction
Angle Estimation
Output
User is a passenger
Periods when driver's hand is off steering wheel
Steering wheel rotation angle when driver's hand is on steering wheel
Application
Curve Speed Warning
Driver Behavior Guidance
Understeer /Oversteer Estimation
Car Maintenance Detection
Lane Change Detection
......

Figure 4.4: System overview.

4.4.1 System Overview

The basic idea of our system is to use wrist-mounted inertial sensors to capture the hand movements and wrist rotation of the user to detect driving, and tracking steering wheel usage and angle. As shown in Figure 4.4, the system takes as input real-time sensor readings from both the wearable device and its paired smartphone including the accelerometer, gyroscope, and magnetometer readings. It then performs coordinate alignment to align the sensors' readings of wearable device and smartphone under the same coordinate system. Coordinate aligment utilizes magnetometer and accelerometer readings in order to find the rotation of devices with respect to earth coordinate frame and then finds an rotation matrix that maps one sensor's readings to the other sensor's coordinate frame. The output of coordinate alignment will be used in different parts of our work whenever it is necessary to a vector from one coordinate frame to another coordinate frame. The system finally detects whether a user with the wearable device is a driver or a passenger.

The other elements of our system are the *Hand On/Off Steering Wheel Detection* and *Steering Wheel Angle Estimation.* If a driver is detected, our system then continuously detects when driver's hand is on/off the steering wheel. It could be done by identifying when the drivers hand is moving away from and moving back to steering wheel based on the captured hand movements of the driver. During the periods when driver's hand is on the steering wheel, we further estimate the turning angle of the steering wheel based on the wrist rotation caused by the movement of rotating the steering wheel. The output of our system, for example, driver's hand is on/off steering wheel and the steering wheel rotation angle, could be used as input to build a broad range of driving safety applications, such as curve speed warning, understeer/oversteer estimation, unsafe lane changes, and distracted driving.

Coordinate Frame Alignment

Processing the inertial sensors data requires transformations between the different watch, phone, car and world (earth) coordinate spaces through rotations. In this work, we use quaternion to represent rotations in three dimensions, a representation that is also directly available through some sensors APIs [51].

There are two important applications of quaternions in our system. The first is to represent rotation of a coordinate frame with respect to another coordinate frame.

Quaternions provide a way to encode the axis-angle representation, and to apply the corresponding rotation to four element vectors. A sensor quaternion vector determines sensor rotation relative to the world coordinate frame, which we define as the x-axis pointing east, y-axis pointing north, and the z-axis pointing up. In Figure 4.5 the different coordinate systems are illustrated. In our work, the smartwatch's rotation relative to the smartphone's can be found by:

$$\mathbf{q_{wp}} = \mathbf{q_{pe}}\mathbf{q_{we}}^{-1} \tag{4.3}$$

Here q_{wp} represents smartwatch quaternion relative to phone coordinate frame, q_{pe} and q_{we} represents pone and watch quaternions relative to the earth coordinate frame.

The second application of quaternions is representing a vector defined in world coordinate frame in another coordinate frame by using the Hamilton product as shown below:

$$\mathbf{p_w} = \mathbf{q_{we}}\mathbf{p_e}\mathbf{q_{we}^{-1}}. \tag{4.4}$$

Here $\mathbf{p_e}$ represents the input vector in the world coordinate frame, $\mathbf{q_{we}}$ represents the quaternion in the world coordinate frame, and $\mathbf{p_w}$ is the output vector in the watch coordinate frame.

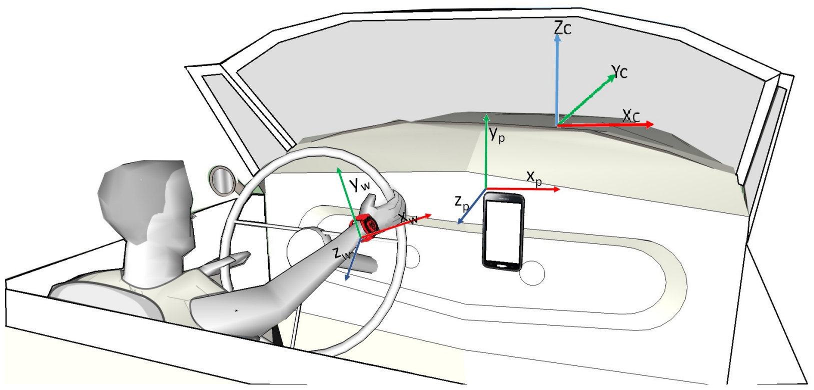


Figure 4.5: Phone and smart watch can be utilized to detect user and car movements.

4.4.2 Driving detection

For driving detection, an obvious distinction between a driver and a passenger is that the driver's hand will steer the wheel when the car is making a turn. This leads to the intuition of our system being to distinguish driver and passenger through capturing the pattern of steering wheel movements. Our system overview is shown in Figure 4.4. It first collects data from sensors (including the accelerometer, gyroscope and magnetometer) of both a user's wrist-worn sensing device (e.g. smartwatch) and smartphone. We assume that riding in a vehicle can be detected using existing inertial or location-based techniques [15] and concentrate on differentiating between the driver and passenger. To this end, it performs turn detection using the collected data. Because the most discriminative driver hand's motion tends to happen during turns, our system conducts turn detection by recognizing three different phases of each turn. Based on the output of turn detection, we extract six features to characterize the hand movements. With this feature set, our system can detect whether the user wearing sensing device is a driver or a passenger using a Support Vector Machine (SVM) based classifier.

Turn detection

We develop a *three-phase turn detection* algorithm, which segments different turning periods based on the gyroscope data collected from both smartwatch and smartphone.

The first phase is to find the *vehicle turn period*, wherein the vehicle is actually turning, by applying threshold detection to a sliding time window over the vector

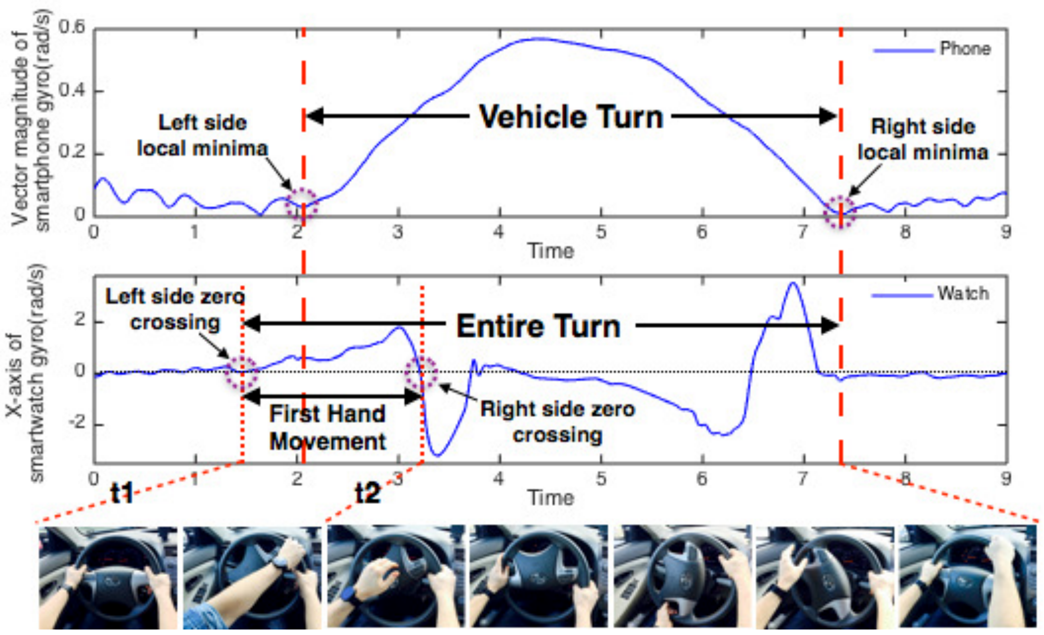


Figure 4.6: Illustration of the turn detection procedure.

magnitude of the smartphone gyroscope. Once all data in the window exceeds the threshold the algorithm detects a turn and expands the left and right sides of the time window to the first local minima, to capture the whole vehicle turn period, as shown in Figure 4.6.

However, based on our observation, the beginning part of the turning steering wheel movement might be missed in the vehicle turn period, especially at low driving speeds. The pictures at the bottom of figure 4.6 shows the whole turning steering wheel movement of a right turn, and the beginning part, between t1 and t2, always occurs slightly before the vehicle begins to turn. Thus, the second phase of our turn detection is to detect the first hand movement period through gyroscope x-axis data from the smart watch, which could precisely characterize this movement. The algorithm uses two pointers to search for the first zero crossing point on both the left and right sides of the starting time of the vehicle turn. The segment between left side and right side zero crossing point is identified as the *first hand movement period.*

Finally, the third phase identifies the *entire turn period* as the union of the vehicle turn period and the first hand movement period. The entire turn period now includes the entire steering hand motion. Note that this three-phase turn detection algorithm is also applied to passenger watch and smartphone data. In this case, the hand movement may not exist and the detected period is simply the vehicle turn.

Feature Extraction

After a turn is detected, our system extracts features from sensor data so that real steering movements can be distinguished from arbitrary arm movements performed by passengers. Our system consists of 6 features in total, which are described as follows and denoted by $f_{featurename}$.

Rotational Change. Since the arm and wrist typically rotate during steering movements, we select watch rotation as the first feature. Directly detecting the rotation of arm using smartwatch data is hard as the watch's rotation (relative to world coordinate frame) is affected by both the arm movement and car's change in heading. We therefore eliminate the effect car rotation during the entire turn period by calculation the rotation of the smartwatch relative to the smartphone ($\mathbf{q_{wp}}(t)$). This is performed by applying Eq. (4.3).

We next need to detect the direction of the rotational movement to distinguish turning steering wheel from other rotational movements, such as eating snacks. Since, the phone can be in an arbitrary rotation in the car, the calculated $\mathbf{q_{wp}}$ does not provide information about the direction of arm movement. We can find the rotation of the watch's movement during entire turn relative to watch's rotation at the beginning of the car's turn, *initial coordinate frame*, by using Eq. (4.3) with $\mathbf{q_{wp}}$ and initial quaternion ($\mathbf{q_{w0}}$), the quaternion at the beginning of the turn, as summarized below:

$$\mathbf{q_w(t) = q_{w0}(q_{pe}(t){q_{we}}^{-1}(t))^{-1}} \tag{4.5}$$

$$\mathbf{f_{rotChange}} = max(abs(\mathcal{T}\{\mathbf{q_w(t)}\})) \tag{4.6}$$

Finally, we use the maximum of the absolute values of the time series of Euler angles in the individual axes as the rotational change feature vector $\mathbf{f_{rotChange}}$. Here, $\mathcal{T}\{\}$ refers to the transformation from quaternion time series to Euler angles time series. We use the absolute values, since we are not interested in the direction of rotation (steering left or right) and select the maximum rotation in each axis over the time window of the entire turn period.

Arm Acceleration. Although rotational change feature provides information about the direction and angle of the turn, it does not provide any information about the radius of rotational movement. As a consequence two hand movements can produce same rotational change feature as long as hand is rotated along the same axis and same degrees. To the best of our knowledge, there is no direct way of finding the radius of rotational movement from inertial sensors. Fortunately, acceleration vectors relative to a fixed coordinate frame will be larger when the radius of the rotational movement is large and will be smaller when the radius is small. Thus, we can use acceleration as a feature. However, acceleration vectors are expressed relative to arm's rotation and it will be changing when the arm is rotating. Therefore, these measurements need to be represented in the same coordinate frame before doing any vector operation on them or using them as a feature. We chose arm's initial rotation as the fixed coordinate frame for the sake of simplicity of calculations. Our method calculates watch's acceleration relative to this coordinate frame during entire turn period by first calculating quaternion vectors relative to initial coordinate frame ($\mathbf{q_{winit}(t)}$) and then taking Hamilton product (Eq. (4.4)) of these quaternion vectors with accelerometer vectors ($\mathbf{a_{we}}(t)$). Finally, the maximum of calculated acceleration is chosen as the feature ($\mathbf{f_{armaccel}}$).

$$\mathbf{q_{winit}}(t) = \mathbf{q_{w0}}\mathbf{q_{we}}^{-1}(t) \tag{4.7}$$

$$\mathbf{f_{armaccel}} = max(\mathbf{q_{winit}}(t)\mathbf{a_{we}}(t)\mathbf{q^{-1}_{winit}}(t)) \tag{4.8}$$

Car Acceleration. Since the watch's acceleration measurements are largely affected by the car's acceleration, we also need to provide the car's maximum acceleration relative to the watch's initial coordinate frame as a feature ($\mathbf{f_{caraccel}}$). The car's acceleration will be equal to the phone's acceleration when the phone is fixed. Therefore, a similar method as in the $\mathbf{f_{armaccel}}$ calculation is carried out for coordinate frame transformations with the phone's acceleration measurements during entire turn period.

$$\mathbf{q_{pinit}}(t) = \mathbf{q_{w0}}\mathbf{q_{pe}}^{-1}(t) \tag{4.9}$$

$$\mathbf{f_{caraccel}} = max(\mathbf{q_{pinit}}(t)\mathbf{a_{pe}}(t)\mathbf{q_{pinit}^{-1}}(t)) \tag{4.10}$$

First Hand Movement. As introduced in section 2.2, we also need to extract some features from the first hand movement. The maximum of watch's gyroscope x-axis value is selected to express the maximum rotation velocity during this movement. Besides, the time duration of the movement is also selected, since passenger's hand movements are usually shorter than drivers.

Other Features. To further improve our feature set, we experimented with several additional features like mean or maximum of devices' gyroscope data. By utilizing feature selection, we abandon some of them and choose the watch's gyroscope magnitude mean value of the entire turn period, which can improve the accuracy of our classification model.

Classification

With the identified feature set, the next step of our system is to distinguish whether a user of the smart watch is a driver. There are multiple methods that could be used for such a binary classification task and here we choose SVM as our classifier—not only because it is one of the most robust ones but also because we empirical found that it achieves the better performance in our task than several alternatives.

During the classification process, we first normalize the feature set by converting it into unit domain. With this normalized feature input, the classifier determines whether the data fits driver or passenger.

4.4.3 Hand on/off Steering Wheel Detection

As showed in Figure 4.4, in order to estimate whether a driver's hand is on or off the steering wheel, we first perform *Hand Movement Detection* to capture all the driver's hand movements except for rotating the steering wheel. Intuitively, for the period when the driver's hand leaves the steering wheel, there must be two movements. One is the *Departure Movement*, which indicates driver's hand moved away from the steering wheel and happens at the beginning of hand off period. The other is the *Returning Movement*, which means driver's hand is moving back to the steering wheel and happens at the end of the hand off period. For each pair of departure movement and returning movement, the driver's hand always moves to opposite linear and rotational directions with respect to the steering wheel. Therefore, after hand movement detection, we perform *Linear Direction Estimation* and *Rotational Direction Estimation* to extract the linear and rotational moving directions of each hand movement. We then combine the linear and rotational direction information to match each pair of departure and returning movements. Thus, the period that between each pair of matched movements are hand off steering wheel period, the rest are hand on steering wheel periods.

Hand Movement Detection. To detect driver's hand movements with respect to the steering wheel, we need to eliminate the effect of car's motion from driver's wrist sensor reading. This could be done by subtracting the car's acceleration and gyroscope readings from the ones of the wearable device after coordinate alignment. We then get the acceleration and angular speed only produced by hand's movement. We further smooth the wearable device's readings to reduce noise by using a low pass filter. We experimentally find that the gyroscope is more reliable for hand movement detection than that of acceleration. We thus examine the peaks (i.e., local maximum) of the magnitude of gyroscope reading to detect the time when hand movements may happen.

After getting peaks of gyroscope magnitude, we perform peak clustering in the time domain. As each movement could be related to multiple continuous peaks in a specific time domain, we could then filter out vibration noise and the motion corresponding to slight hand movements (which are also keeping hand on the steering wheel) by excluding those clusters that contain less numbers of peaks. Therefore, after peak clustering, we can get each hand movement, starting at the first peak in each cluster and ending at the last peak in the same cluster.

Linear Direction Estimation. During each movement period, estimating the linear moving direction is helpful to characterize and match the movements. For the three axis of the car's coordinate system, the x and y axis are always more noisy. We thus always estimate the linear direction according to z axis, namely the up and down direction.

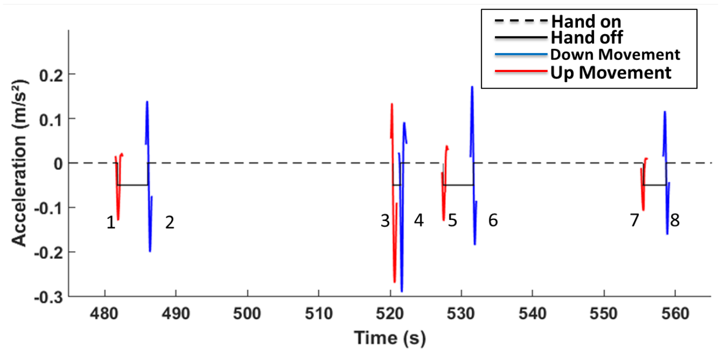


Figure 4.7: Linear Direction Estimation

As the hand always moves from one stationary state to another stationary state for each hand movement, there will be a speed up process and a slow down process. If the hand speeds up towards the up direction (i.e., large acceleration with negative readings) and then slows down, it means the hand is moving up, and vice versa. We thus can exam the sign and the magnitude of the acceleration to identify the moving direction of the hand. Figure 4.7 & 4.8 show the acceleration and rotation angle measurements from a wearable sensor placed on the arm. As shown in Figure 4.7, the black horizontal dash and solid lines show the ground truth when hand is on and off the steering wheel respectively and eight curves show the movements we detected. The acceleration pattern we mentioned above is obvious for the 2nd, 3rd, 4th, 6th, 8th curve, but is indistinct for 1st, 5th and 7th curve. It is because sometimes the effect of gravity could not be perfectly eliminated. Therefore, we add one more rule to estimate the movement direction. That is if the maximum value of the z axis acceleration is less than 0.01, this movement is a up movement (because the maximum value is supposed to be a larger positive value). Therefore, based on the two patterns we find in the z axis acceleration value, we could estimate whether a movement is either a up movement or a down movement.

Rotational Based Movement Matching. The main principle of movement matching is the departure and returning movements must have different moving directions. Thus, we first find all adjacent paired movements with opposite direction and regard all such pairs as potential matching. However, some events such as sudden braking or bumpy road conditions may cause dramatic changes of both the accelerometer and gyroscope values, which may lead to an incorrect movement detection and even an

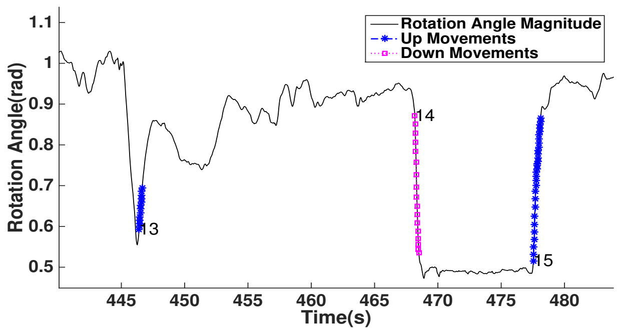


Figure 4.8: Rotational based matching

incorrect hand off period detection. As shown in Figure 4.8, Movement 13 is an incorrect hand movement detection, while 14 and 15 are a pair of departure and returning movements.

To solve this problem, we calculate the rotation angle of the watch respect to the smart phone, and use its magnitude to monitor the angle changes of the watch. Since the driver's hands always have slightly continuous movements when his hands are on the steering wheel, while during the hand off period, his hands are relatively stable, we developed a variance filter to reduce matching errors. This filter simply calculates the variance between two potential matching movements and sets a threshold (0.002 rad in our system) to remove those match with a large variance. In the example in Figure 4.8, the variance between 13 and 14 is 0.004 rad while the variance between 14 and 15 is 0.0003 rad. We therefore match movement 14 and 15 together. By applying this rotational based movement matching, the system provides more accurate hands on/off steering wheel detection to facilitate steering wheel angle estimation.

4.4.4 Steering Wheel Angle Estimation

Once steering wheel usage is detected, we further use the wrist rotation of the smart watch to infer the steering wheel turning angles. When the driver rotates the steering wheel, the smart watch also experiences wrist rotation as it rotates together with the steering wheel. The steering wheel angle thus could be inferred from the wrist rotation of the smart watch. In particular, we use the gyroscope readings of smart watch to measure the wrist rotation for steering wheel angle estimation.

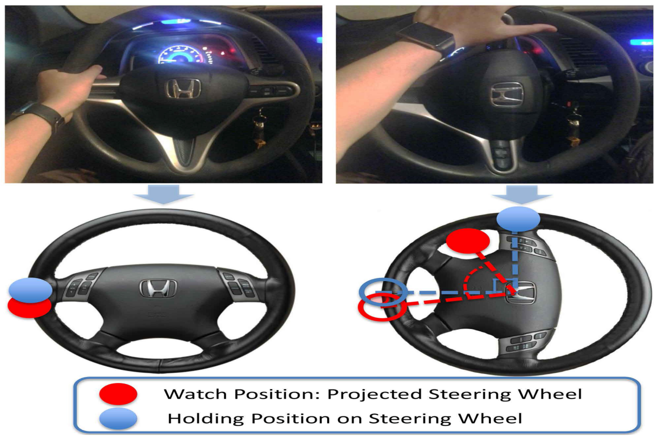

Watch Position: Projected Steering Wheel
Holding Position on Steering Wheel

Figure 4.9: Holding position and the projected position of smart watch on steering wheel.

In the coordinate alignment step of the system, we align both the car's and the smart watch's coordinate systems with the steering wheel's. By aligning these coordinate systems, we are able to project the gyroscope readings of both the car and the smart watch to the plane of the steering wheel. We then perform *Data Calibration and Integration* to remove the effect of car's motion (i.e., car's rotation) and integrate the gyroscope readings (i.e., angular velocity) to rotation angle. After this step, we obtain the wrist rotation of smart watch with respect to the steering wheel's rotation. Such wrist rotation is then used in *Angle Estimation* to infer the steering wheel angle based on the constructed driver rotation profile in *Profile Construction*. In this study profile construction is studied for a single driver and a more generic driver rotation profile construction is left as a future work.

Data Calibration and Integration. When a car makes a turn, the gyroscope readings (i.e., angular velocity) of the smart watch are a combination of the angular velocities of the steering wheel rotation and the car. As the car's angular velocity can be measured by the phone inside the car, we simply subtract the car's angular velocity from the angular velocity of the smart watch after coordinate alignment. We then obtain the angular velocity of the smart watch which solely corresponds to the steering wheel rotation. Such angular velocity is then integrated to wrist rotation angle with respect to the steering wheel. We define such wrist rotation as θ_{wrist}.

Profile Construction. Ideally, we expect the wrist rotation θ_{wrist} equal to the steering wheel angle θ_{real} when making turns. However, we experimentally find that the wrist rotation of the smart watch is usually less than that of the steering wheel. This is because the driver bends his wrist in different ways when holding the steering wheel under different turning angles. As shown in Figure 4.9, before making a turn, the wrist has a large angle to the steering wheel plane. The projected location of the smart watch on the steering wheel is thus close to the holding position. After a 90 degree turn, the wrist angle to the steering wheel plane becomes smaller, which leads to the projected location farther away from the holding position. The rotation of the smart watch is thus smaller than that of the steering wheel. Moreover, the attitude change of the watch is mainly due to the wrist rotation instead of the translation movement following the trajectory of the steering wheel movements. Thus, the smart watch mainly rotates side to side on the wrist. These factors makes the smart watch rotation significantly less than that of the steering wheel rotation angle.

To address the aforementioned issue, we use a supervised learning method to create a driver rotation profile. In particular, we collect training data offline to create a mapping (i.e., a function $f(x)$ such that $\theta_{real} = f(\theta_{wrsit})$) between the rotations of the smart watch and the steering wheel turning angles. This is based on the observation that people usually consistently bend their wrists when turning. The steering wheel turning angles thus can be inferred based on the wrist rotation of smart watch (i.e.,θ_{wrist}) and the created angle mapping (i.e., $f(x)$). To create the driver rotation profile, we use a gyroscope sensor aligned with the steering wheel to measure the real steering wheel angles, and then use liner regression to fit the smart watch rotation θ_{wrist} to the measured steering wheel angle. We experience with different models, such as linear, quadratic and cubic fits, and find that the linear fit is simple yet effective.

To construct the profile, we could also use a semi-supervised approach. It uses the smart phone inside the car to monitor the centripetal acceleration and angular velocity of the car when making turns to calculate the angle of the steering wheel. When there is no understeer/oversteer, the steering wheel angle could be calculated based on the circular motion laws and Ackermann steering geometry [49], as shown in equation (4.1). We could then build the profile automatically based on the real-time measurements of

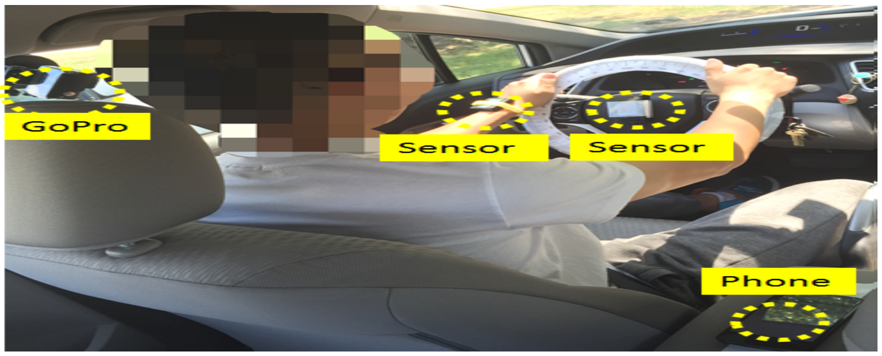


Figure 4.10: Position of sensors and smartphone.

the θ_{wrist} and the calculated steering wheel angles. After the profile is constructed, we can then infer the steering wheel angle based on wrist rotation of the watch θ_{wrist} for building higher level safety applications, such as understeer/oversteer detection and curve speed warning.

Angle Estimation. By plugging in the real-time θ_{wrist} to the mapping profile $f(x)$, we could infer the real-time steering wheel angle as $\theta_{est} = f(\theta_{wrist})$.

4.5 Performance Evaluation

4.5.1 Experiment Setup

To evaluate our approach, We conduct experiments with an Invensense MPU-9150 9-axis motion sensor, which is a prototyping alternative to a wearable device. The motion sensor contains 3-axis accelerometers, gyroscopes, and magnetometers. In our experiments, the participants (driver and passenger) are asked to attach the motion sensor to his/her left wrist, where people usually wear watches. The motion sensor is paired with participant's android smartphone via Bluetooth. The smartphone is placed in the middle cup holder with its coordinate system aligned with that of the car. For steering wheel angle estimation, we align another motion sensor with the steering wheel to record the ground truth of the steering wheel rotation angles when making turns. In contrast to other steering-wheel-mounted sensor based approaches our work is a low-cost and unobtrusive design that can reach a much larger population and focuses on finding the steering wheel angle without requiring any additional sensors. However,

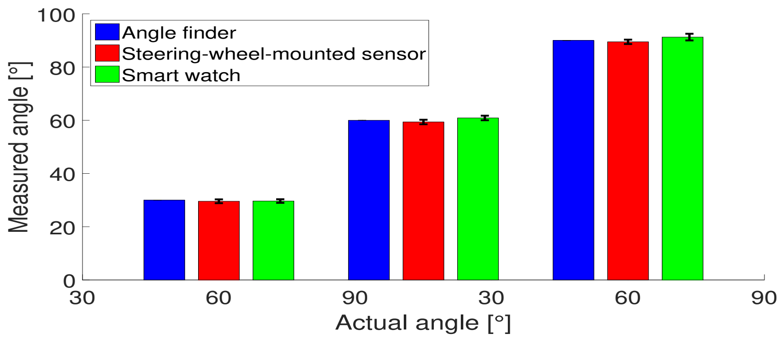


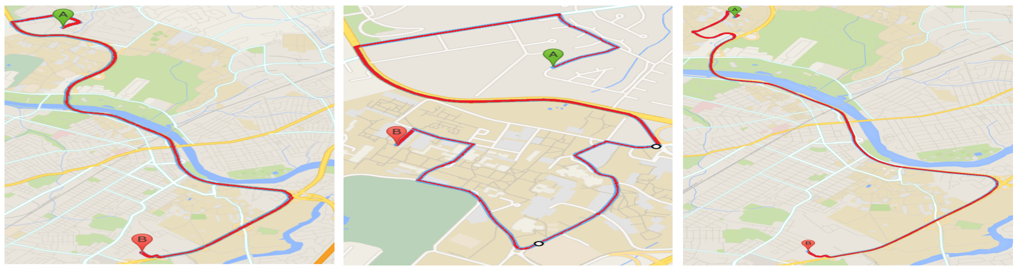
Figure 4.11: Measurements from angle finder, smart watch, and steering-wheel-mounted sensor

obtaining the ground truth is very hard with angle finders in a dynamic while the vehicle is being used. As a consequence, we used steering-wheel-mounted sensors to collect the ground truth. Our experiments with static steering wheel showed that the error in steering-wheel-mounted-sensors is negligibly small (less than 1°) and can be usable as ground truth. Figure 4.11 shows mean and standard deviations in angle measurements from the angle finder, the steering-wheel-mounted sensor, and the smartwatch. All the other experiments are performed while the car is being driven. Additionally, a GoPro camera is mounted over the driver's shoulder to record the ground truth of the driver's hand movements while driving.

The positions of these devices are shown in Figure 4.10. The sampling rates of motion sensors and smartphones are set to 100Hz for driving detection and 50Hz for steering tracking, which are supported by most off-the-shelf wearable and mobile devices, such as the smart watches and fitness trackers. The sensing data of the motion sensor is transmitted to and stored at smartphone via Bluetooth by an app. We have two drivers preform data collection with one driving Honda Civic and using Nexus 5 smart phone and the other driving Toyota Camry and using Nexus 6 smart phone.

4.5.2 Driving Detection Evaluation

We evaluate the driver detection in terms of how accurately the system detects turns and recognizes the user as a driver for each turn, how long it takes to detect that a user is a driver.



(a) Highway (b) Local road (c) More movements

Figure 4.12: Maps of commute routes.

Driving Detection Accuracy

For driver detection evaluation, the experiments are conducted under two scenarios. For the first scenario, the driver drives in a controlled area and the passenger always keeps both wrists stationary (now arm movement). For the second scenario, the driver drives on a real road, and the passenger moves (playing with the phone and eating snacks). We collect data for 239 turns in the first scenario and 41 turns in the second scenario, both with passenger and driver inside the car. Our three-phase turn detection algorithm correctly detects all of these 280 turns resulting in 100% accuracy.

Figure 4.13 presents the driver detection accuracy by using 10-fold cross-validation when different sets of data are used. Overall, our system achieves 98.9% accuracy by mixing the data collected in both scenarios (i.e., all data). If we only include the passenger and driver data in the first scenarios for training and testing, the accuracy goes to 100% as no wrist movement of passenger interferes with the driver detection. In the second scenario, playing with the phone still yields 100% accuracy; it can apparently easily be distinguished by the classifier. The more challenge scenario is when the passenger is eating. When using data when passenger is eating, our system results in 96.7% accuracy due to some movements of eating produce similar features as steering movements.

4.5.3 Detection Delay

Detection delay indicates how much time our system needs to make a decision after starting to drive. Often the first turn occurs when leaving the parking space and our

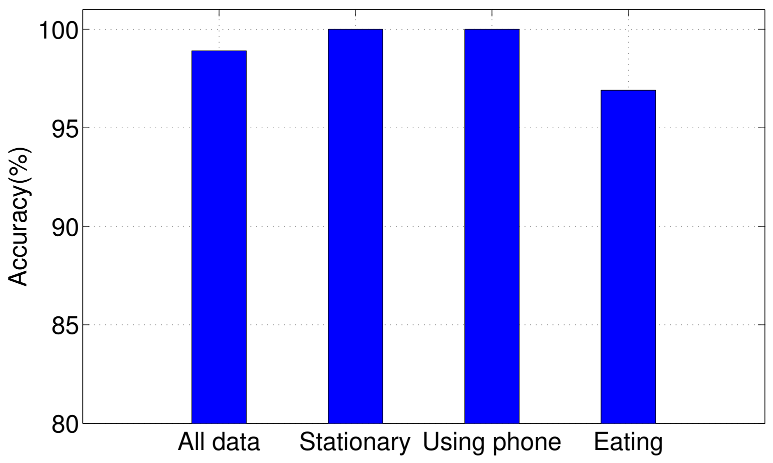


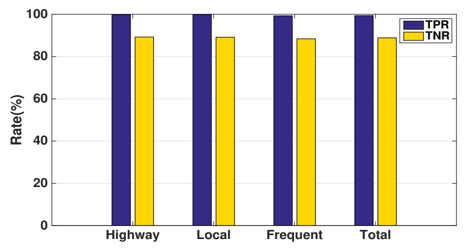
Figure 4.13: Detection accuracy in different cases

system can detect the driver within the first seconds of the drive. To understand the performance under less ideal conditions, we consider here the case where no steering is necessary to leave the parking space. Since we perform driver detection based on a single turn, the detection delay thus can be defined as the time from starting a vehicle to complete the first turn. The whole detection delay (d_{all}) could be further divided into two parts: the time that starting a vehicle from stationary to the beginning of the first turn (d_s) and the time to complete the first turn (d_t).

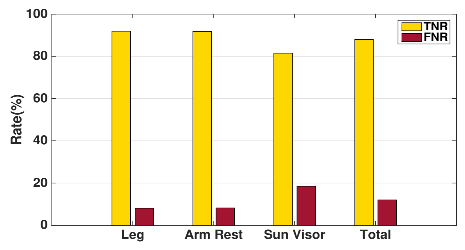
Table 4.2: Detection delay of each drive

Turn No.	1	2	3	4	5	6	7	8
$d_s(s)$	18.55	12.45	18.45	13.90	9.50	8.75	23.20	11.35
$d_t(s)$	6.10	5.40	5.45	7.65	7.60	9.40	5.85	5.40
$d_{all}(s)$	24.65	17.85	23.90	21.55	17.10	18.15	29.05	16.75

For the previous experiments, we have made 8 trips. The detection delay for these 8 cases are shown in table 4.2. The results show that the detection delay (d_{all}) is very small: ranging from 16.75 seconds to 29.05 seconds, with an average of 21.13 seconds. Moreover, a vehicle usually makes multiple turns to pull out of a parking lot at the beginning of each trip. Our system can thus be expected to have even smaller detection delay in such cases.



(a) True positive rates and True negative rates of different data sets.



(b) True negative rates of different kinds of movements.

Figure 4.14: Hand on/off detection performance.

4.5.4 Hand On/Off Steering Wheel Detection Evaluation

For hand on/off steering wheel detection and steering wheel angle estimation evaluation, We conduct experiments by driving on three routes with different road conditions, as shown in Figure 4.12. The first two routes, namely *Highway Commute* and *Local Road Commute*, cover the highway and local road commutes (i.e., Figure 4.12 (a) and (b)), respectively. These two routes cover a wide range of steering wheel angles as highways usually involve smooth curves resulting in small steering wheel rotation angles, and local roads often have sharp turns that cause large steering wheel rotation angles. We collect data for 20 trips over one month on these two routes and the driver is asked to drive naturally. Additionally, in order to evaluate our hands on/off the steering wheel detection algorithm, we intentionally perform more frequent hand on/off movements during real driving on a highway as shown in Fig. 4.12 (c) .

We first evaluate how accurately we can detect the driver's hand is on or off the steering. We thus define the positive cases as the driver's hand is on the steering wheel and the negative cases as the driver's hand is off the steering wheel. We measure the duration of hands on/off the steering wheel in terms of number of seconds. Therefore, each positive/negative instance corresponds to the driver's hand is on/off the steering wheel for one second. The one second's resolution is good enough to tolerate the error when we labeling the ground truth manually by checking the video of each commute trip.

We use the true positive rate and true negative rate to measure the overall performance of the hands off/on steering wheel detection. The true positive rate represents the

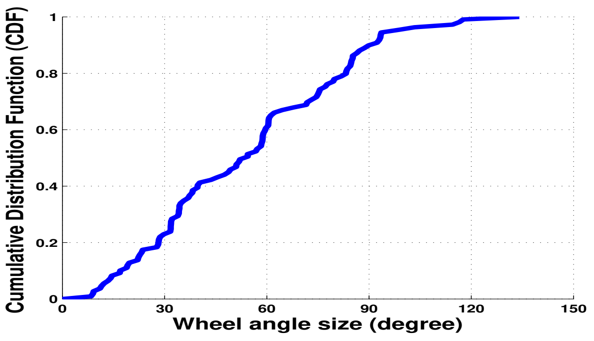


Figure 4.15: The distribution of the steering wheel rotation for all the turns.

proportion of the period the hand on the steering wheel could be correctly determined, while the true negative rate represents the proportion of the period the hand off steering wheel could be correctly identified. In other words, the true positive rate reflects how accurately our method could activate the steering wheel angle estimation, whereas the true negative rate shows how accurately the method could trigger unsafe driving behavior notification, for example warning driver when his hand is off the steering wheel.

Overall, our hand on/off steering wheel detection achieves a 99.9% true positive rate and a 89.2% true negative rate for the normal commute data set, and a 99.3% true positive rate and an 88.4% true negative rate for the frequent hand movement commute data set. We also evaluate our system in two different commute types: highway and local road. Figure 4.14 (a) compares the performance of our method on three routes. We observe that different routes have similar performance, which demonstrates our method is robust to different road conditions and different frequencies of hand on/off the steering wheel events. By combining the results from these three routes, our method achieves a 99.4% accuracy on detecting hand on steering wheel periods and an 88.8% accuracy on detecting hand off steering wheel periods.

We next evaluate how accurately our method can detect when the hand is off the steering wheel for different types of hand movements. In the experiments, we tested three different typical hands-off events of drivers, namely *hand on leg*, *hand on the armrest* and *adjusting sun visor*. Figure 4.14 (b) compares the true negative rates and false negative rates for these three hands-off events. We observe that for both hand on leg and hand on armrest events, the true negative rate remains above 91.5% and

	[0,30]	(30,60)	> 60	All turns
Mean	0.000063	-0.00012	0.00032	0.000046
Std.	2.21	3.82	6.12	4.93

Table 4.3: The mean and standard deviation (in degree) of the true error for the turns with different steering wheel angles.

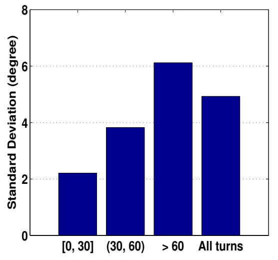
false negative rate is less than 8.5%. However, for reaching for the sun visor event, the performance decreases to as low as 81.5%. This is mainly because reaching the sun visor has a smaller moving duration and continuous moving process than that of other two events. Thus, it is relative harder to match the hand movements to the departure movement and returning movement.

4.5.5 Steering Wheel Angle Estimation Evaluation

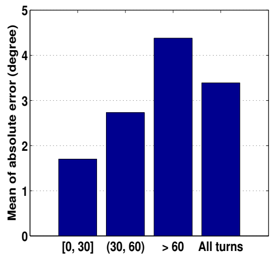
For the steering wheel angle estimation, we collect 109 turns ranging from 120 degrees right turn to 90 degrees left turn in both the highway commute and the local road commute. The rotation of the steering wheel for these turns is up to 134 degrees. The distribution of the steering wheel rotation for the collected turns is shown in Figure 4.15. In particular, there are 25 turns with steering wheel rotation less than 30 degrees, 43 turns with steering wheel rotation in between 30 to 60 degrees, and 41 turns with steering wheel rotation larger than 60 degrees.

We use two types of errors, *true error* and *absolute error*, to evaluate the performance our steering wheel angle estimation. The *true error* is defined as the real steering wheel rotation minus the estimated steering wheel angle (i.e., $\theta_{real} - \theta_{est}$), whereas the *absolute error* is the magnitude of the difference between the true rotation and the estimated angle (i.e., $||\theta_{real} - \theta_{est}||$).

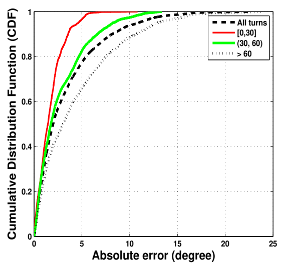
We first evaluate how the estimated steering wheel angle could detect the slipping of understeer/oversteer. Table 4.3 presents the mean and standard deviation of the true error under the turns with different rotation ranges of the steering wheel. We observe that the mean of the true error is about zero for all the turns and the turns under different steering wheel rotation ranges. We find the standard deviation of the true error



(a) The comparison of the standard deviation of the true error for the turns with different ranges of steering wheel angle.



(b) The comparison of the mean absolute error for the turns with different ranges of steering wheel angle.



(c) The cumulative distribution function of absolute error for the turns with different ranges of steering wheel angle.

Figure 4.16: Steering wheel angle estimation performance.

increases when increasing the steering wheel rotation ranges. This is because that with larger rotations on steering wheel, the wrist movement experiences larger variation as well. As we rely on the angle mapping profile $f(x)$ to infer the steering wheel angle, the larger wrist movement variation consequently produces larger variation of the angle estimation.

In particular, Figure 4.16 (a) compares the standard deviation of the true error for the turns with different steering wheel rotation ranges. We next compare the standard deviation obtained in our experiments with the ones presented in Table 4.1 to evaluate the effectiveness of understeer/oversteer detection. We find that the standard deviation for the turns with steering wheel rotation less than 30 degrees is 2.21 degree which results in approximately 99% accuracy when detecting 10 degrees slipping or about 90% accuracy in detecting 5 degrees slipping. Whereas for the turns with steering wheel rotation in between 30 to 60 degrees, the standard deviation is 3.82 which corresponds to about 99.5% accuracy when detecting 20 degree slipping or more than 90% accuracy in detecting 10 degree slipping. The standard deviation for the turns with wheel rotation larger than 60 degree is 6.12 degree which leads to about 95% accuracy when detecting 20 degree slipping. Note that it's more likely that larger rotation of the steering wheel will cause larger angle of slipping. Even if the standard deviation increases when increasing the size of the steering wheel angle, it could still produce high accuracy, 95% to 99% for example, in detecting increased size of slipping angle. We next investigate the absolute error, which describes the magnitude of difference between the true steering wheel angle and the estimated angle. Figure 4.16 (b) depicts the mean absolute error under the turns with different rotation ranges of steering wheel. Overall, we observe that the mean error is less than 3.4 degrees for all the turns. Moreover, the turns with larger steering wheel rotation have relative larger mean error. Specifically, the mean error for the turns with less than 30 degree steering wheel rotation is 1.7 degrees, whereas it is 2.73 and 4.38 degrees for the turns with rotation in between 30 to 60 degrees and rotation larger than 60 degrees, respectively.

Figure 4.16 (c) shows the cumulative distribution function of absolute error under the turns with different rotation ranges. Similarly, we observe the curve shifts to right when the rotation of steering wheel becomes larger indicating larger rotation of steering wheel has relative larger error. The median error increases from 1.45 to 1.76 and to 3.04 when the turns with the steering wheel rotation increases from the range of $[0, 30]$ to $(30, 60)$ and to larger than 60 degrees. And the 80% percentile error for the case of $[0, 30]$, $(30, 60)$ and larger than 60 is 2.66, 4.7 and 7.55 degrees respectively. We also observe a large error (e.g., 10 degrees) at the tail of the CDF curve. This is because the driver occasionally bends his/her arm very differently for the same size of steering wheel angle. However, the percentage of such large errors is very small for the turns with less than 60 degrees rotation of steering wheel. For example, it is less than 3% for the case of $(30, 60)$ and less than 5% for all the turns. And the large error becomes less significant for the turns with rotation of steering wheel larger than 60 degrees. It still, however, leaves us room to further improve the angle estimation method for example by combining the accelerometer and gyroscope readings [52]. Overall, the above results show that our steering wheel angle estimation method is effective with a mean error less than 3.4 degrees.

4.6 Discussion and Conclusion

In this work, we exploit the opportunity of using wristworn wearable devices to enable activity recognition of unsafe driving. In particular, we develop the fundamental techniques in driving detection, hand on/off steering wheel detection, and steering wheel angle estimation to provide more detailed information in tracking of driving behaviors. We examine whether our proposed vehicle-independent techniques leveraging wearables could achieve sufficient accuracy for building driving safety applications. Through various real-driving scenarios, we show that our approach can achieve 98.9% driver detection accuracy, which corresponds to an average driver detection latency of 21.13 seconds after starting to drive. System also achieved hand on steering wheel detection with a true positive rate around 99% and provide warning of unsafe driving when a driver's hand is off the steering wheel with a true negative rate above 80%. Additionally, our system can achieve accurate steering angle estimation with errors less than 3.4 degrees to facilitate applications such as curve speed warning and understeer/oversteer detection.

Open challenges remain towards realizing automatic steering and driver tracking using sensing techniques independent of vehicles. A limitation of our approach is its statistical nature for safety applications. E.g., detection accuracy decreases if the driver steers only with the watch-less hand. However, we still envision that there exist a range of human behavior modification and attention direction uses for which statistical approaches are useful. Furthermore, our algorithms only need to be activated during driving. For most people, this only occupies a small portion of their day. Thus, the energy consumption incurred by our system should not pose any significant burden to the wearables.

Another potential shortcoming is that we only construct and test the profile of wrist rotation with respect to the steering wheel angle for one driver. We expect such profile is relative consistent for different drivers as the wrist movements are highly constrained by the rotation trajectory of the steering wheel. Still, re-training or fine-tuning the profile may be required for different drivers. Such training efforts, however, could be mitigated by using the semi-supervised learning based method proposed in profile construction step. Furthermore, different drivers may have different styles of wearing the device. For example, the panel of the smartwatch could face down for one driver but face up for another. As the postures/attitudes of the wearable device could be detected by the motion sensors, different postures thus could be calibrated to the one matches the construed profile based on the detected posture.

Chapter 5

Single-sensor motion and orientation tracking in a moving vehicle

Given the increasing popularity of mobile and wearable devices, this chapter explores the potential use of inertial sensors that are widely available on mobile and wearable devices for vehicle and driver tracking. Such capability would enable novel classes of mobile safety and assisted driving applications without relying on information or sensors in the vehicle. Although inertial sensors have been widely used in motion tracking, existing approaches cannot distinguish the motion of the vehicle and the device's motion in the vehicle. Additionally, the noise exerted from the electronic components in the vehicle and the ferromagnetic frame of the vehicle distorts the inertial sensor readings. This paper introduces a method to separately estimate the orientation of both the vehicle and the sensor by tracking the earth's magnetic field and the electromagnetic distortion from the vehicle measured by a magnetometer in addition to a gyroscope and an accelerometer. Specifically, the vehicle noise is used to estimate the orientation of the sensor within the vehicle while the earth's magnetic field combined with vehicle noise is used to estimate the vehicle's heading. Our on-road experiments show that the technique is able estimate sensor orientation with mean error of 5.61^o for yaw angle and 3.73^o for pitch angle, and able to estimate vehicle heading with mean error of 4.12^o .

Inertial sensors on wearable and mobile devices have led to a great number of applications in activity tracking and have created new forms of human-computer interaction. Their use in driving applications can enable new unobtrusive advanced assisted driving systems and safety applications without relying on sensors installed in the vehicle. Providing such services from mobile devices allows quicker dissemination of new safety services into legacy vehicles. For example, the orientation of the inertial sensors placed

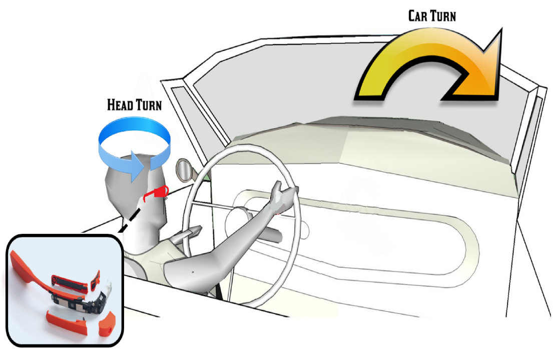


Figure 5.1: An example inertial sensor placement on the head mounted device such as Google Glass

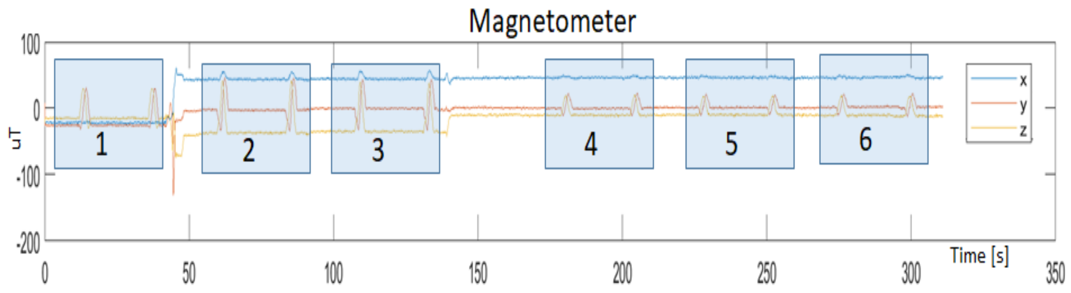


Figure 5.2: Magnetometer readings 1) Outside of the vehicle 2) On Vehicle floor, Engine off 3) On floor, Engine on 4)On seat, engine off 5) On seat, engine on 6) On seat, Vehicle Moves. The small peaks occurs when sensor rotates along its x-axes

on wearable devices that tracks driver head movements such as in Figure 5.1 can be used for contextual warnings, and gesture based arm tracking systems can be used for interactions with the vehicle's infotainment system. Such services require accurate tracking of vehicle motion and the mobile's sensor orientation in the vehicle.

Existing solutions Nowadays, many vehicles come with seat occupancy sensors. Some newer models of vehicles come with driver tracking technologies based on a driver's steering wheel movements [53, 54] or eye tracking [55]. However, these approaches require manufacturers to place sensors in the vehicle and most of these sensors are only available in newer and pricier car models. In order to make this feature more available in other vehicles, researchers have used mobile phones to track driver head movements by using smartphone cameras [20, 56]. Although this approach removes the burden of sensor placement, the driver activity tracking provided by these models are very limited and susceptible to visual occlusions. Additionally, these methods utilize computationally expensive machine vision methods. Only recently have there been studies investigating the use of wearable devices to track driver behaviours [17, 57]. Although these methods estimate both the vehicle's and the user's movements, they require an additional reference sensor to be placed in the vehicle to separate the mobile device's movements from those of the vehicle reference frame.

We propose a single-sensor motion and orientation tracking method that does not require an additional reference sensor to track vehicle's motion. The method is able to separate the vehicle's motion from the sensor's motion in a moving vehicle, The method estimates vehicle's magnetic noise which occurs naturally due ferromagnetic materials used in the vehicle and utilizes this magnetic noise as a reference to vehicle's heading direction. The method then uses the relative angles between vehicle's magnetic noise and earth's magnetic field to estimate the vehicle's heading angle as well as sensor's yaw and pitch angles. We studied how the measured magnetic field changes as the vehicle and the sensor rotates and formulated required equations to estimate the vehicle's heading angle from magnetometer measurements.

Applications Estimating the sensor's and the vehicle's movement from a single-sensor could enable many useful applications and eliminate the problems related to multiple sensors in prior models mentioned above. The sensor placed on the driver's arm as a smartwatch or a fitness tracker could detect driver's arm movements and could be used to estimate steering wheel movements [45, 57] which then could be utilized for many driving safety and assisted driving applications. Additionally, inertial sensors in head-mounted devices such as Google Glass could be used for tracking the driver's head movements. Such movements could give accurate information about the driver's focus of attention or lack thereof. Head tracking could be used to enable assisted driving applications by providing contextual information or warnings to drivers based on where the driver turns his head.

The contributions of our work are summarized as follows:

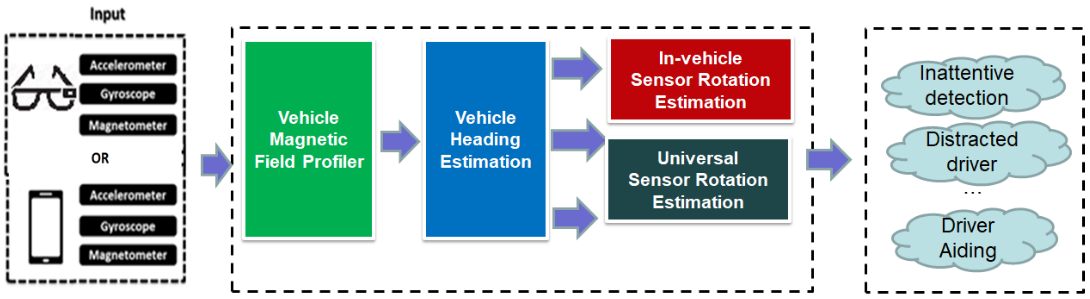

Input
Accelerometer
Gyroscope
Magnetometer
OR
Accelerometer
Gyroscope
Magnetometer
Vehicle Magnetic Field Profiler
Vehicle Heading Estimation
In-vehicle Sensor Rotation Estimation
Universal Sensor Rotation Estimation
Inattentive detection
Distracted driver
...
Driver Aiding

Figure 5.3: System overview

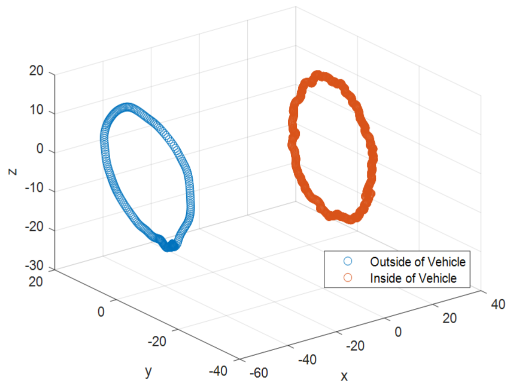


Figure 5.4: Magnetometer readings inside and outside of the vehicle when earth's magnetic field rotate

- We analyzed the magnetic noise in the vehicle and how the magnetic field measurements change in the vehicle during the vehicle and sensor turn.
- Provide a mechanism that can estimate the characteristics of the vehicle's magnetic noise in the vehicle that can be used for orientation estimation.
- We formulated the relationship between magnetometer measurements and vehicle's magnetic noise to estimate the orientation of the sensor in the vehicle as well as the vehicle's heading angle.
- Propose an approach that eliminates the reliance on multiple sensors, which previous studies and researches had utilized and evaluated the method with in-field experiments.

5.1 Background

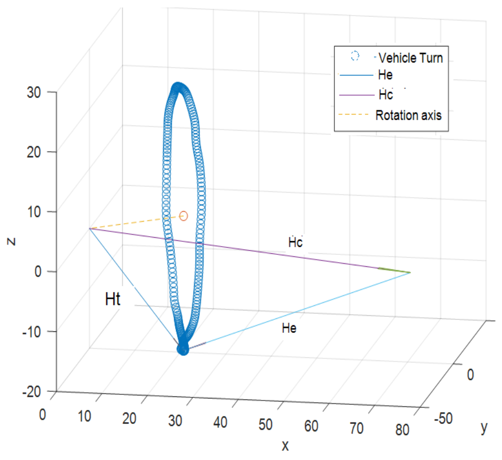
There have been active research efforts in reinforcing driving safety by tracking driver's behaviours leveraging mobile sensing technologies on the smartphone (including using cameras, embedded sensors, and other auxiliary devices (e.g.,OBDII) in mobile devices or vehicles). In particular, several previous studies use cameras placed on the vehicle to track driver attention and predict driver maneuvers. Oliver et al. [58] use manually annotated driver's gaze from cameras placed in the vehicle to predict driver maneuvers such as lane change and turning. However, this approach needs information about where the driver's visual attention is focused, which is not readily available on current HMD and mobile devices. Several other studies [59, 60] gathered head or eye poses with computer vision techniques and used machine learning algorithms to predict the maneuvers. These studies reveal the correlation between driver's head movements and vehicle's maneuvers. Doshi et al. [60] also states that head pose tracking systems are more robust than gaze tracking and for the lane change detection systems head pose is a better cue than eye gaze.

In contrast, other works rely less on specific phone placement and more on motion sensing through phone's embedded inertial sensors. Chen et al. [19] develop a vehicle steering detection middle-ware to detect various vehicle maneuvers, including lane changes, turns, and driving on curvy roads. Liu et al. [61] design a simple setup to collect useful driving data for self-driving with smartphone. Castignani et.al. [18] propose SenseFleet, a driver profile platform that is able to detect risky driving events independently on a mobile device. Wang et al. [14, 17] utilize embedded sensors in a smartphone and a reference point (e.g., an OBD device) in the vehicle to determine whether the phone is on the left or right side of the vehicle. However, most of these approaches can only infer the vehicle's motion based on the inertial sensing measurements from the smartphone, which can hardly track driver behaviour inside the car.

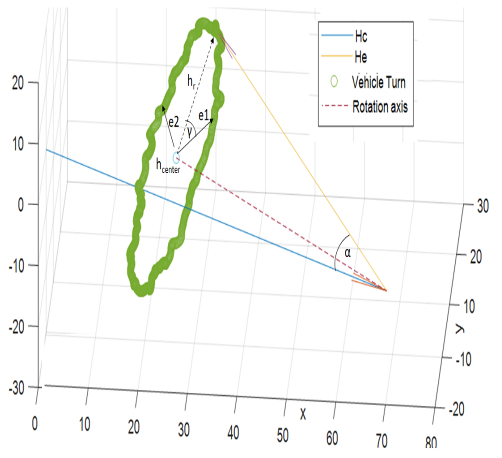
The emerging market of wearable devices provides the opportunity to track motion of human body components. For example, wrist-worn smartwatches or fitness bands can be used to track human's hand and arm [48, 62–64], while smartglasses and other head mounted displays (VR, AR) can be used to track head positions and orientations [65–68]. However, tracking human body inside the vehicle is challenges. As the vehicle is a non-inertial system, the motion of the vehicle generates large noise to the inertial sensor measurements, which significantly reduce the tracking accuracy of those previous algorithms. Previous works [57, 61, 69] use the inertial sensor measurements from both the smartphone and smartwatch to estimate the human motion. The basic idea is to use the smartphone to track vehicle motion, and derive the human motion inside the car by eliminate the vehicle motion from the smartwatch. However, this approach requires both the smartphone and smartwatch to work jointly. In our work, we can track both the vehicle and human motion using only one single inertial sensor.

Applications. Estimating the sensor's and the vehicle's movement from a single-sensor could enable many useful applications and eliminate the problems related to multiple sensors in prior models mentioned above. The sensor placed on the driver's arm as a smartwatch or a fitness tracker could detect proper steering such as whether the driver turned the steering wheel properly when making a turn or is able to keep the vehicle in the lane. This model can detect any inconsistency between the steering wheel and the vehicle movements. In turn, this data can be used to detect external factors, such as sliding vehicles due to road conditions. Then, such information can be crowd-sourced to warn other drivers. Drowsy driving can be also detected by comparing inconsistencies in the driver's steering through wearable sensor's such as smartwatches. Similarly, arm tracking models could also enable gesture based interactions with infotainment systems. For example, the driver may be able to turn the volume up or down with arm movements over the steering wheel without actually turning the steering wheel, creating a safer driving experience with fewer distractions.

Another body of applications could be enabled by inertial sensors in head-mounted devices such as Google Glass. Head-mounted devices could be used for tracking the driver's head movements. Such movements could give accurate information about the driver's focus of attention or lack thereof. Headtracking systems could detect the driver's errors such as not checking the mirrors before changing lanes or not checking the sides for oncoming traffic before making a turn at an intersection. Head tracking data could



(a) Sensor Turn



(b) Vehicle Turn

Figure 5.5: Vehicle's magnetic noise ($\mathbf{H_v}$), Earth's magnetic field ($\mathbf{H_e}$) and Sensor measurement ($\mathbf{H_t}$) during a) sensor turn b) vehicle turn

be used to enable assisted driving applications by providing contextual information or warnings to drivers based on where the driver turns his head. Just like inertial sensors on smart-watches, head-mounted devices can be also used for simple interactions with the infotainment system. A nodding gesture could be detected by the sensor to select or affirm the option on the current infotainment system. All of such applications will create a safer driving experience for the driver and eliminate distractions which may otherwise be caused by such wearable devices.

Different from previous work, our approach does not rely on the sensors that may not be found in every vehicle. Additionally, the method does not require computationally intensive computer vision algorithms and does not suffer from visual occlusions. The primary advantage of the method is that it requires only single device with inertial sensors that can be found in most of the mobile and wearable devices.

5.2 System Design

Figure 5.3 illustrates the system overview of the proposed single-sensor tracking approach. Our method first profiles the characteristics of the vehicle's magnetic field from magnetometer readings. The parameters calculated in the profiling stage and inertial sensor readings are used to estimate the vehicle's heading direction. Finally, sensor's rotation with respect to vehicle is estimated from vehicle's heading, profile parameters and sensor readings. Furthermore, Sensor's orientation with respect to the earth can be calculated from the vehicle's and sensor's orientation.

5.2.1 Challenges

The main challenges lie in separating the vehicle's motion from the sensor's motion in the vehicle by using a single sensor. The inertial sensor readings are affected by both the vehicle's movements and the sensor's movements. As an example, the accelerometer readings taken from a head mounted device will be affected by linear acceleration of the head, centripetal force due to the head turn, linear acceleration of the vehicle, centripetal force due to vehicle turn and gravity. In addition to this ambiguity in the source of the movement, there are many sources of noise that exist in the vehicles such as vibrations and magnetic noise due to the vehicle's ferromagnetic structure. Although the vibrations could be eliminated by using low-pass filters, the magnetic noise constitutes a greater problem that requires further attention. Inertial sensors utilize magnetometers as a 3D compass to track the magnetic north pole of the earth. This, combined with the gravity measured by the accelerometer, is used to estimate the orientation of the sensor in North-East-Down (NED) coordinate frame. However, the magnetometer readings exhibit a drift due to the magnetic noise in the vehicle. Furthermore, the magnetic noise depends on the metallic structure of the vehicle as well as sensor's closeness to metallic surfaces. Therefore, there is a greater challenge in eliminating the effect of magnetic noise on the sensor's movements and hence, the data collected. Moreover, one could expect that the magnetic noise in the vehicle can also be affected by the means of the changes in the electrical signals in the vehicle. In our experiments, however, we have not observed such a change around the driver/passenger seat area where the users usually use their wearable devices.

5.2.2 Approach

Our current approach is unique in that it relies on the vehicle's magnetic noise. In our experiments, we have observed that vehicle's magnetic noise is a good beacon

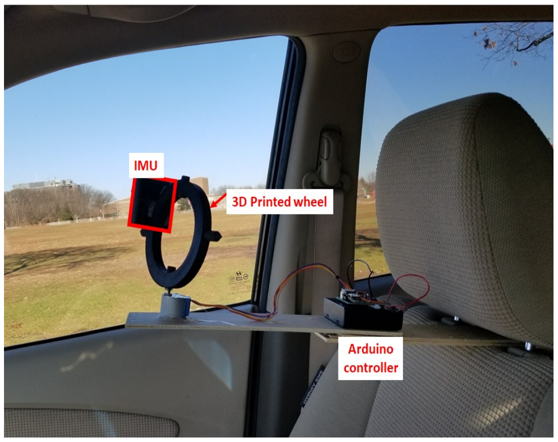


Figure 5.6: Arduino controlled sensor setup

for vehicle's orientation. By keeping track of vehicle's magnetic noise and the earth's magnetic field, we estimate the vehicle's heading and the sensor's orientation in the vehicle. During vehicle or sensor turns, the magnitude of earth's magnetic field and the vehicle's magnetic noise remain constant. However, the direction of these magnetic fields might change during vehicle and sensor turns. Therefore, the angle (α) between these magnetic vectors and the total magnetic field measurement will change accordingly. Our approach first estimates the angle between these vectors based on the magnetic field measurement. Then, based on the angle, we calculate the vehicle's heading. Finally, we estimate the sensor's orientation in the vehicle.

We will introduce the characteristics of the vehicle's magnetic field in the next subsection. Then we will define how the magnitude of the magnetic field measurement changes during vehicle and the sensor turns. Finally, algorithms will be introduced in detail.

5.2.3 Vehicular Magnetic noise

Due to ferromagnetic materials used and the electrical current running in the electrical system of the vehicle, the vehicle effects the magnetic field inside the vehicle. These affects can be grouped in two bodies. First, the hard iron affects act as a separate magnetic field and are added to the earth's magnetic field. In Figure 5.2, we have depicted the magnetometer sensor readings at various positions by holding the sensor stable for a while and then making two turns around its x-axis. The turns around x-axes could be seen as 12 small peaks in the figure. The measurements shown in the first region of the graph are taken outside of the vehicle in an open field and therefore, purely depict the earth's magnetic field. As the sensor rotates on it's x-axes, the reading on the y and z axes also vary in a sinusoidal shape. The second region of the graph shows how the sensor reading changes after the sensor is placed on the floor of the vehicle when the engine is not running. As the sensor is placed in the vehicle, the magnetic field introduced by the vehicle is added to earth's magnetic field.

In the third region, the engine starts running. Regions 4-6 are measured while the sensor is placed on the car seat. You can observe small changes in magnetometer readings around t=150s which is due to sensor's positional change from car floor to the seat. In region 4, the engine is not running. In region 5 engine starts running and finally in region 6, the vehicle starts moving. We can observe that the magnetic field does not change between regions 2-3 and 4-6. Therefore, we can conclude that the magnetic field measurements are mostly affected by the sensor's position in the vehicle and are not affected by the engine run.

The second body of affects is the soft-iron affect. Soft iron affects change the magnitude of the magnetometer measurements. In magnetometer readings, soft-iron affects could be observed as different amplitudes on different axes as the sensor rotates in a constant magnetic field such as the earth's magnetic field. In Figure 5.4, we have plotted magnetic field measurements when the earth's magnetic field rotates when the sensor is outside of the vehicle (blue circle) and inside of the vehicle (red circle). To obtain the data points in the the blue circle, we have simply rotated the sensor outside of the vehicle and measured the earth's magnetic field. For the sensor, this will result in the earth's magnetic field rotating around it. However, rotating the sensor does not result in depicting only the earth's magnetic field inside of the vehicle since the vehicle's magnetic field is also involved in this test. However, when the vehicle turns, the vehicle's magnetic field is constant and only the earth's magnetic field will be rotating for the sensor. In our experiments we have not observed significant soft-iron effects inside the vehicle. Therefore, vehicle ignited soft-iron effects has been disregarded for the sake of simplicity.

In addition to hard-iron and soft-iron effects introduced by the vehicle, the electrical components on the sensor board also introduce hard-iron and soft-iron effects. Since any magnetometer sensor zero flux offset is also independent of sensor orientation, it simply adds to the sensor board's hard-iron component and is calibrated and removed at the same time. However, both hard-iron and soft-iron effects are not vehicle specific and require a very standard procedure to calibrate. Therefore, these calibrations are carried out separately and will not be discussed in this paper.

5.2.4 Magnetic field during a turn

For the purpose of clarity, we introduce how the magnetic field measurements vary as the vehicle or the sensor in the vehicle rotates. As we have explained in the previous subsection, we have neglected the soft-iron effects that vehicle introduces, therefore the vehicle's magnetic field can be simply represented as a vector $\mathbf{h_v(t)}$. Similarly,the earth's magnetic field is denoted as $\mathbf{h_e(t)}$ and sensor measurement $\mathbf{h_t(t)}$ can be calculated as :

$$\mathbf{h_t(t)} = \mathbf{h_v(t)} + \mathbf{h_e(t)} \tag{5.1}$$

Additionally from cosine law in vector addition,

$$\|\mathbf{h_t(t)}\|^2 = \|\mathbf{h_v(t)}\|^2 + \|\mathbf{h_e(t)}\|^2 + 2 * \|\mathbf{h_v(t)}\| \cdot \|\mathbf{h_e(t)}\| \cdot cos(\alpha) \tag{5.2}$$

Since, the magnitude of $\mathbf{h_e(t)}$ and $\mathbf{h_v(t)}$ don't change with time, $\|\mathbf{h_t(t)}\|^2$ can be used to calculate α. We will now describe how α changes during sensor and vehicle turn.

Sensor turn in the vehicle. During a sensor turn in the sensor coordinate frame, both the vehicle's magnetic field and earth's magnetic field will be rotating along the rotation axis. Therefore, the sensor will be measured $\mathbf{h_t(t)}$ as a vector rotating along the rotation axis as shown in Figure 5.5a. The measurements will lie on the circle at the base of a cone where the cone's apex is the origin and the slant of the cone is the rotation axis. As the sensor turns, $\mathbf{h_v(t)}$ and $\mathbf{h_e(t)}$ will be constant with respect to each other and the α angle between them does not change. Hence, the magnitude of the $\mathbf{h_t(t)}$ does not change as well.

Vehicle turn. During the vehicle's turn, the vehicle and sensor turn simultaneously. Therefore, the sensor will be measuring $\mathbf{h_v(t)}$ as a constant. On the other hand, the earth's magnetic field would be rotating with respect to the sensor. This would result in $\mathbf{h_t}$ lying at a cone's base circle where cone is formed by rotating $\mathbf{h_e(t)}$ on the apex of the cone which is $\mathbf{h_v(t)}$. The slant of the cone is the vehicle's rotation axes and γ, vehicle's heading angle is also illustrated in Fig5.5b. From Eq.5.2, $||\mathbf{h_t(t)}||$ will vary as the vehicle rotates.

5.2.5 Initial Vehicle magnetic field profiling

For accurate vehicle heading and sensor orientation estimation, several parameters need to be profiled. The first obvious parameter is the magnitude of the car's magnetic field. The second group of parameters define how the α angle maps to the vehicle's heading angle. For profiling, we use a single 360^o vehicle turn.

One could assume that $\mathbf{h_v(t)}$ could be simply estimated by subtracting $\mathbf{h_e(t)}$ from $\mathbf{h_t(t)}$. Although, $||\mathbf{h_e(t)}||$ could be estimated or retrieved from online magnetic field calculators for given latitude and longitude, estimating its direction in vehicle is not a trivial task. In early stages of our system design, we have tried the following approach : First, measure the earth's magnetic field $\mathbf{h_e(t0)}$ when the sensor is outside of the vehicle right before the sensor enters the vehicle. Then use gyroscope based orientation estimation for a short-time to estimate $\mathbf{h_e(ti)}$. The main reason, we relied on gyroscope based orientation is that the magnetometer based approaches do not work when the sensor enters the vehicle due to vehicle's magnetic field. We have empirically observed that this approach suffers from gyroscope drift problems even when used for a short time period. However, we use this estimation $\mathbf{h_{vrough}}$ as a rough estimation to choose one of the two possible candidate points which we introduce in the next paragraph.

As an alternative, we used the $\mathbf{h_t(t)}$ measurements during a 360^o vehicle turn. As mentioned in previous subsection, during the vehicle's turn $\mathbf{h_v(t)}$ should be on the apex of the cone.Therefore by finding the apex of the cone, we can estimate the $\mathbf{h_v(t)}$. Additionally, $\mathbf{h_t(t)}$ measurements lie on the base circle and are $||\mathbf{h_e(t)}||$ away from the apex, $\mathbf{h_v(t)}$. For given $\mathbf{h_t(t)}$ measurements during vehicle turn and $||\mathbf{h_e(t)}||$, two possible cones could be formed. We choose the cone whose apex is closer to $\mathbf{h_{vrough}}$ and use its apex as $\mathbf{h_v}$. Here, $\mathbf{h_v}$ is car's magnetic field for the static reference frame defined by the sensor's coordinate frame during profiling. We choose this coordinate frame as vehicle's coordinate frame and define the heading of the vehicle with respect to this orientation and denote vectors represented in this coordinate frame with left superscript such as ${}^*\mathbf{h_v}$

In addition to $\mathbf{h_v}$, we define $\mathbf{h_{center}}$ as the center of the base circle. and two perpendicular vectors $\mathbf{e_1}$ and $\mathbf{e_2}$ as shown in Fig. 5.5b. Although, center of the base circle and $\mathbf{e_1}$ and $\mathbf{e_2}$ vectors will change with the sensor's orientation in the vehicle, their relationship, such as dot product, with $\mathbf{h_v}(t)$ are independent from sensor orientation and will not change as the sensor rotates.

5.2.6 Orientation Estimation

Next, we are going to define equations that we will use to estimate the vehicle's heading, sensor's orientation with respect to the vehicle and earth.

Vehicle heading estimation In order to estimate sensor's orientation, we first need to estimate the vehicle's heading. This requires mapping the angle between $\mathbf{h_e}$ and $\mathbf{h_v}$, α, that we obtained from Eq.5.2 to vehicle's heading angle γ. First, from dot product rule :

$$ {}^*\mathbf{h_e} \cdot {}^*\mathbf{h_v} = \|\mathbf{h_e}\|\|\mathbf{h_e}\| cos(\alpha) \tag{5.3}$$

${}^*h_e$ can be also written,

$$ {}^*\mathbf{h_e} = {}^*\mathbf{h_t} - {}^*\mathbf{h_v} \tag{5.4}$$

Similarly ${}^*\mathbf{h_t}$ can be represented as,

$$ {}^*\mathbf{h_t} = {}^*\mathbf{h_{center}} + \mathbf{e_1} * cos(\gamma) + \mathbf{e_2} * sin(\gamma) \tag{5.5}$$

where $\mathbf{e_1}$ and $\mathbf{e_2}$ are defined in profiling stage and illustrated in Fig.5.5b. By combining Eq.5.3 and Eq.5.5 and carrying out equations :

$$\gamma_{1,2}(t) = \pm(asin(\frac{\|\mathbf{h_v}\|\|\mathbf{h_e}\|cos(\alpha) - (^*\mathbf{h_{center}} \cdot {}^*\mathbf{h_v}) + \|\mathbf{h_v}\|^2}{d} \\ - asin(\frac{\mathbf{e_2} \cdot {}^*\mathbf{h_v}}{d}))$$

where d is,

$$d = \sqrt{(\mathbf{e_1} \cdot {}^*\mathbf{h_v})^2 + (\mathbf{e_2} \cdot {}^*\mathbf{h_v})^2)} \tag{5.6}$$

Since asin is defined in $[0\ \pi]$, There are two possible γ values. To choose the correct $\gamma(t)$ value, we utilize gyroscope measurements. The algorithm makes an estimate from previous $\gamma(t-1)$ and gyroscope measurement $\phi_x(t)$:

$$\gamma_{estimate} = \gamma(t-1) + \phi_x(t) * \Delta t \tag{5.7}$$

and assigns the closest $\gamma_{1,2}(t)$ to $\gamma_{estimate}$ as $\gamma(t)$. However, this selection operation might cause fluctuations in heading estimation. For this reason, we have implemented a temporal filtering approach which requires at least two consequent samples to switch from one selection to the other. This selection process can be improved by better temporal filtering approaches.

In-vehicle Sensor Orientation Estimation Sensor orientation estimation is straight forward. From given $\gamma(t)$, $\mathbf{e_1}$, $\mathbf{e_2}$ and $^*\mathbf{h_{center}}$, $^*\mathbf{h_t(t)}$ can be calculated from Eq.5.5. Rotation vector that, $\mathbf{R_{in}(t)}$, transforms $^*\mathbf{h_t(t)}$ to sensor's coordinate frame representation $\mathbf{h_t(t)}$ can be calculated in angle-axes form by using Matlab's built-in command *vrrotvec* with $\mathbf{h_t(t)}$ and $^*\mathbf{h_t(t)}$. Rotation vector than can be converted to rotation matrix, $\mathbf{R_{in}(t)}$, with *vrrotvec2mat* or to euler angles with *rotm2eul* command.

Universal Sensor Orientation Estimation Sensor's orientation with respect to earth can be calculated by :

$$\mathbf{R_e(t)} = \mathbf{R_{in}(t)} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\gamma(t)) & -sin(\gamma(t)) \\ 0 & sin(\gamma(t)) & cos(\gamma(t)) \end{bmatrix} \tag{5.8}$$

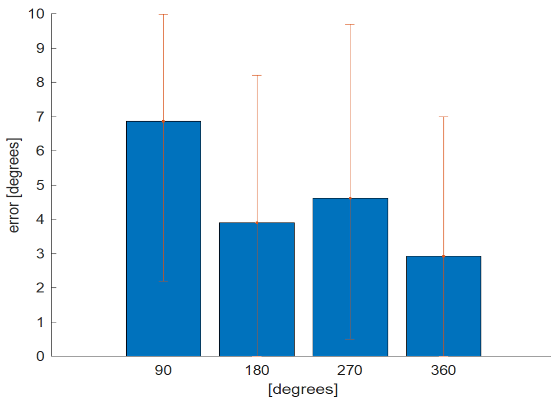
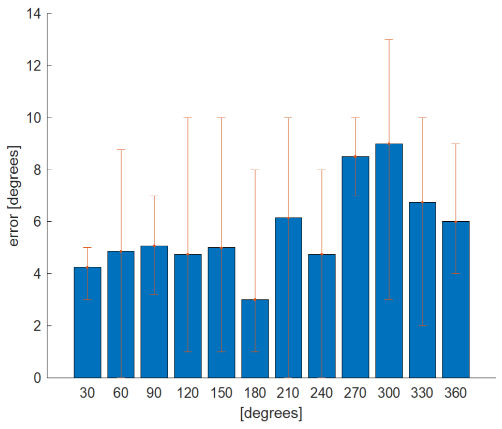
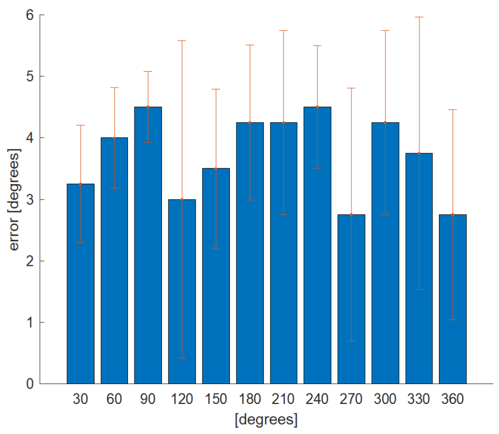


Figure 5.7: Mean Vehicle Heading Estimation Error when vehicle is placed at 90 to 360 degrees with 90 degree increments. Blue bars indicate mean error, red bars show minimum and maximum error.



(a) Sensor Yaw Error



(b) Sensor Pitch Error

Figure 5.8: Mean sensor orientation estimation error when sensor is placed at 30 to 360 degrees with 30 degree increments. Estimation Error for a)Sensor Yaw Angle b) Sensor Pitch Angle. Blue bars indicate mean error, red bars show minimum and maximum error

5.3 Performance Evaluation

During our study, we have conducted two sets of experiments. The objective in the first set of experiments was to estimate the vehicle's heading angle. The second part of the evaluation aims to find the sensor's orientation, namely yaw and pitch angles, by eliminating vehicle's motion from sensor motion. Finally, we compare these results with computer vision based methods and [57] which requires two sensors to estimate the sensor's roll angle.

5.3.1 Experiment Setup

We have developed an Arduino controlled setup to simulate sensor movements in the vehicle. The Arduino controller is used to send commands to step motors to turn the sensor to specific angles. We have placed the sensor on a 3D-printed spinning wheel with one degree of freedom, i.e yaw. Both spinning wheel and Arduino controller are placed on a wooden plank so as to prevent introducing further magnetic noise. The sensor setup is placed in the passenger seat around the headrest position. However, as the sensor rotates, it is possible for the 3D printed wheel to get very close to the headrest, which might have ferromagnetic materials inside. When the sensor approaches any ferromagnetic materials, the magnetic field might vary as the sensor rotates. Therefore, we have empirically chosen a placement point where the sensor is not affected by the ferromagnetic properties of the headrest as it rotates. The sensor placement in the vehicle is shown in Fig.5.6.

In a standard data collection session, we begin with rotating the sensor 360^o placed outside of the vehicle in an open field. We use this data for standard magnetometer calibration. Next, we take the sensor into the vehicle and place it in the aforementioned position. Then we make a 360^o vehicle turn while the sensor is fixed for the profiling. The rest of the experiment involves evaluating vehicle heading estimation and sensor orientation estimation with respect to the vehicle. For the vehicle heading estimation stage, we have placed the vehicle at heading angles between 0^o to 360^o with 90^o increments. The heading angles for the ground truth are determined by aligning vehicle with lines at the parking lot which are perpendicular to each other. For sensor orientation estimation, the sensor makes 30^o turns between 0^o to 360^o angles while the vehicle is driven freely without any restrictions such as route, speed, or direction. We placed the step motor vertically and horizontally for yaw and pitch estimation, respectively. We took the input of the step motor as ground truth for sensor orientation estimation. The step motor can be rotated with 2.81^o increments. The experiments are performed by three different drivers and two different cars, a Hyundai Tucson and a Mercedes Benz GLC.

5.3.2 Vehicle Heading Estimation Evaluation

In our first experiment, we have tested vehicle heading estimates of the algorithm by rotating the vehicle in 0^o to 360^o range with 90^o increments. The ground truth angles are obtained by aligning the vehicle with the perpendicular lines in the parking lot. Data is collected at each angle 10 times at different random positions. The results are plotted in Fig:5.7. The average error is 4.57^o with a standard deviation of 2.97^o. We observed minimum errors when the vehicle is placed at 180^o. The error goes as low as 0^o degrees for 180^o and 360^o while goes as high as 10^o when vehicle is placed at 90^o. We have observed a mean error of 6.86^o, 3.9^o,4.61^o,2.92^o when the vehicle's heading angle is 90^o,180^o,270^o,360^o, respectively. We believe the deviations in these errors could be due to vehicle's imperfect placement each time the specific angle is tested, since a slight shift in the vehicle parking direction could cause a couple of degrees of error.

The other body of evaluation is conducted when the sensor turns while the vehicle is fixed. The vehicle heading estimate of the system was also fixed as expected with only a couple of exceptions for the data collected sensor orientation estimation. We believe these exceptions occur due to switches in selection from $\gamma_{1,2}(t)$ as mentioned in Section 5.2.6

5.3.3 Sensor Orientation Estimation Evaluation

To evaluate the sensor orientation estimation, the sensor is rotated with 30^o counter-clockwise increments while the drivers were driving freely as instructed. The results

Publication	Mean Absolute Error	
	Yaw	Pitch
Yan [70]	6.72	8.87
Ba [71]	8.8	9.4
Murphy-Chutorian [72]	6.4	5.58
Xiao [73]	3.8	3.2
Single-Sensor	5.61	3.73

Table 5.1: The comparison in terms of accuracy with computer vision based head pose estimation studies.

from 480 turns are plotted in Fig.5.8 (a). We have obtained yaw and pitch angles by placing the Arduino controller setup horizontally and vertically. The results for yaw angle estimation is plotted in Fig. 5.8a. We have observed a mean error of 5.61^o with standard deviation of 3.46^o. We observed the minimum error when the sensor is at 180^o and the maximum error when the sensor is at 270^o. We also tested the sensor orientation estimation when the sensor is fixed and the vehicle is making 360^o turns and observed 8.89^o ripples on average. We believe this residual effect of vehicle turns might be the main source of errors in the yaw estimation.

The results for pitch angle estimation is plotted in Fig. 5.8 (b). We have observed mean error of 3.73^o with standard deviation of 1.51^o. Overall, the error was less than 8^o and was less then the error we encountered for the yaw angle estimation. This might be due to residual effects of the vehicle turn might be stronger for the yaw angle since both vehicle turns and sensor's yaw rotation are in the same direction.

A comparison of our results with computer vision based head pose estimation methods are given in Table5.1. We have chosen these studies over many others since they have achieved best accuracy results for different data sets. Yan and his colleagues [70] were able to perform 6.72^o and 8.87^o man absolute error for yaw and pitch estimation on CHIL-CLEAR07 dataset. Ba& Obodbez [71] proposed a method to estimate head pose with 8.8^o and 9.4^o yaw and ptich errors on IDIAP Head Pose dataset. Murphy-chutorian and Trivedi's head pose estimation system for driver assistance systems was able to achieve 6.4^o yaw and 5.58^o pitch errors on CVRR-363 dataset. Our single-sensor orientation estimation system was able to achieve higher accuracy than these aforementioned system without relying on computationally expensive computer vision techniques. On the other hand, Xiao et al [73] recorded a better performance than our approach with 3.8^o and 3.2^o yaw and pitch angle errors on BU Face Tracking dataset. However, this method was only tested for only controlled room environment and the in-vehicle performance is unknown.

Finally, we compared our single-sensor approach to a two-sensor approach [57]. In this work the authors used the inertial sensors on the mobile phone to track vehicle movements and sensors on smartwatch to track driver's arm movements. The authors estimated only arm's roll angle by using fused orientation information obtained from Android API and performance of this approach for yaw and pitch error estimation is unknown. The comparison is illustrated in Fig. 5.9. Since the authors only evaluated their system for $[0^o, 30^o]$, $(30^o, 60^o]$, $(60^o, 90^o]$ roll angle intervals, we are able to compare our system with their results at these angles only. Overall, our system's pitch angle estimation is almost as good as two-sensor based approach. A 3.4^o mean error was obtained in two-sensor solution while a 3.73^o degree error was achieved in our system. On the other hand, the single-sensor yaw angle estimation with overall mean error of 5.61^o is less accurate than two-sensor roll estimation. The yaw angle estimation we achieved in our study were 4.25^o, 4.87^o, and 5.07^o while pitch angle estimation accuracy was 3.25^o 4^o 4.50^o and two-sensor roll angle estimations of 1.7^o 2.73^o 4.38^o mean errors for sensor angles in $[0^o, 30^o]$, $[30^o, 60^o]$, $(60^o, 90^o]$ ranges was achieved. Overall, the single-sensor algorithm produces slightly less accurate results when compared to the two-sensor studies. These slight differences may be due in part to the inefficiency of eliminating vehicle movements. Additionally, the two-sensor approach utilizes gyroscope, accelerometer and magnetometer sensors for rotation estimation. In Android API implementation, the orientation is estimated by integrating angular velocity measured through gyroscope and corrected by using magnetometer and accelerometer sensors to eliminate gyroscope's drift error. The pitch and roll angle correction is mostly affected from accelerometer readings since they are perpendicular to gravity. On the other hand, yaw rotation axis is parallel to gravity and gravity measurement doesn't change with the yaw angle. Therefore magnetometer readings are mostly used to correct yaw estimation and more affected by the disturbances and noises in magnetometer data.

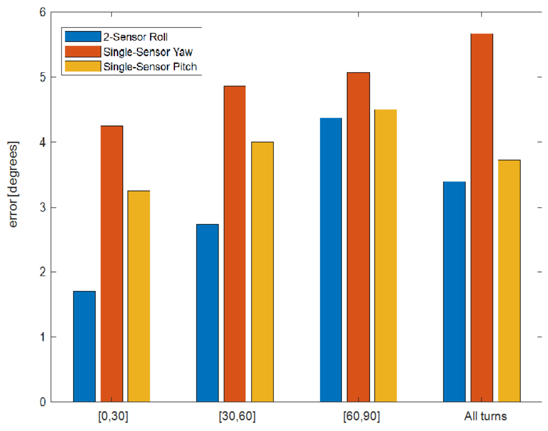


Figure 5.9: Comparison of mean error between two-sensor based roll estimation and single sensor based yaw and pitch estimation.

This might cause larger errors on yaw angle estimation and very similar accuracy results on single-sensor based pitch angle estimation and two-sensor based roll angle estimation.

5.4 Limitations and Future work

There are several limitations of this work due to the nature of the car's magnetic field. First and most obvious one, similar to a compass at the north/south poles struggling to show the compass heading, the algorithm's accuracy at rotation accuracy would decrease as the earth's magnetic field and gravity vectors have similar directions. In other words, as the earth's magnetic field gets closer to gravity, the information it relays loses its significance. This has another implication for the magnetometer in the vehicle, as the vehicle's magnetic field's direction gets closer to the gravity;s direction, it becomes impossible to find the sensor's orientation with respect to the vehicle.

One of the important limitations of this work is that the method assumes the magnitude of vehicle's magnetic field doesn't change. However, the magnetic field might vary as the sensor approaches ferromagnetic materials in the vehicle. We believe the approach will perform especially well where the sensor's translational motion is limited, e.g. head mounted devices. The system performance could be improved for changing vehicle magnetic fields in future work. Additionally, analysis of the vehicle's magnetic field for translational movement might lead to interesting research finding and might be used for sensor positioning.

5.5 Conclusion

In this paper, we proposed a system that allows monitoring vehicle and driver motion, namely vehicle's heading and sensor's yaw and pitch angles, using only one tri-axis inertial sensor, which may be found in various mobile and wearable devices. It estimates the magnetic noise of the vehicle and its effect on the data through magnitude-based noise estimation method. The approach proposed here eliminates the reliance on multiple sensors, which previous studies had utilized. Reliance on a single sensor, which is directly placed on the body or close vicinity of the driver, can estimate sensor orientation with mean error of 5.61^o for yaw angle and 3.73^o for pitch angle can be used to track driver behaviours and using such behaviours to create a safer driving experience. The data derived from this method could in turn be used to determine unsafe driving techniques and help improve driving.

Chapter 6
Preventing Distractions

Majority of driver errors can be prevented by easing the drivers' interaction and minimizing distractions. We propose a complimentary input system that will ease the interaction of driver with mobile devices. In following sections we will introduce customizable paper shortcuts and gesture-based methods for analog inputs based on our previous arm-tracking system. The customizable paper shortcuts might decrease driver distractions by enabling the driver to assign custom shortcuts from his phone and accommodate his everchanging needs with technology. The customizable touch points can be printed on a photograph paper by an inkjet printer in a single pass with commercially available conductive ink. These

6.1 Customizable Paper Inputs

In this section, we present a technique for creating multi-key conductive ink touch user interfaces that can be printed on paper in a single pass. While 3D printing and open-source electronics platforms have led to enormous creativity in creating smart objects, the means for user interaction with such objects are often limited and require remote interaction through a smartphone app. Paper-based touch circuits are a convenient medium for exploring custom touch sensors that can be attached to numerous objects in our environment. The challenge lies in creating a reliable and customizable touch circuit that is easy to produce. Specifically, it should not require assembly of multiple layers and it should support multiple touch points without needing separate connections to a microcontroller for each touch point.

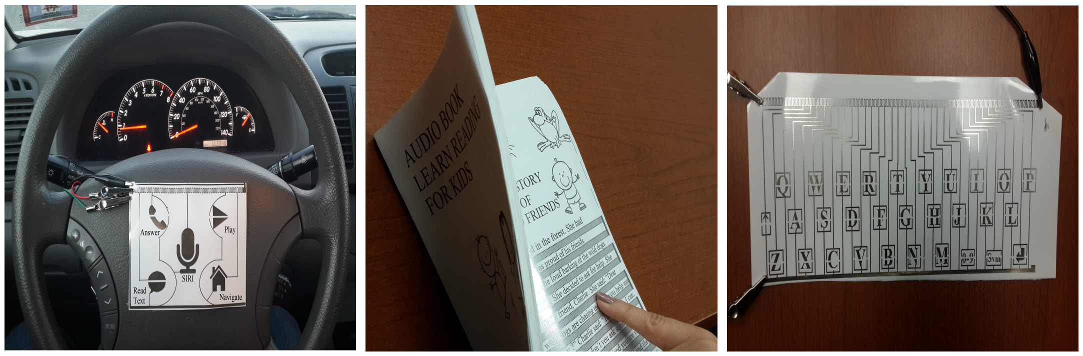
We address this through a resistive touch sensor that exploits the inherently high resistance of printed traces to create multiple detectable touch points. The finger closes the circuit when in contact with the touch point and the sensor uses a polarity-switching technique to cancel out the effect of the unknown skin resistance. We evaluated the touch sensor using keypads with 10, 15 and 20 touch points and achieved 99.6%, 93.5%, and 91% touch detection accuracy, respectively. We also observed touch detection rates of up to 154 touches per minute.

6.1.1 Introduction

Embedding an ever increasing number of smart devices into our surroundings will create a need for more ubiquitous user interfaces that let us interact with these devices. As we have transitioned from PCs and laptops to small mobile devices, touch sensing has proven itself to be an especially compelling means of interaction with smaller devices. What if we could touch-enable a much richer set of objects and surfaces in our environment? One could imagine custom buttons on car dashboards that control devices designed long after the car was manufactured, remote control functions in clothing or couch armrests, lighting control and telepresence functions through buttons on the conference table, or new generations of interactive children's books that do not require bulky reading devices.

Challenges. The main challenges lie in creating touch sensors that can be easily produced and customized for different applications. The touch sensor design should enable exploration in this area, similar to how 3D printing and open electronics platforms have led to enormous creativity in the design of smart objects. In particular, touch sensors should be customizable in size and shape, so that they can easily be affixed to objects of different proportions. Moreover, a touch sensor should accommodate various number of distinguishable touch points to allow creation of interfaces with multiple buttons or keys. Design, production, and particularly reproduction of these keyboards should not require specialized hardware knowledge. They should be straightforward to integrate with custom electronics or smartphone apps such as IFTTT [74].

Existing solutions. Earlier research has studied a variety of technologies for touch-enabling surfaces in our environment. First, projection and camera systems have been used to create virtual touch points on surfaces [75, 76]. This usually requires careful



(a) (b) (c)

Figure 6.1: Example prototypes of printed touch interfaces.

setup and alignment of the camera system and is susceptible to occlusion. Second, touch surfaces have been embedded into objects with custom circuitry. Examples are wearable keyboards [77] or touch points on tables. This again requires careful engineering and instrumentation for a specific object. Third, audio sensing of touches has been used to create a virtual keyboard on a table [78] but it requires careful fingerprinting of the audio environment during initialization time and the process will need to be repeated if the nearby objects are moved. Only recently, researchers have explored printing of touch sensors through conductive ink. The existing work is limited to a single button interface [79], or requires assembly of multiple printed layers and connection of a large number of ADC lines to an electronic circuit [80]. Either the level of customization or the ease of production and reproduction is therefore limited in these approaches.

Approach. To address these challenges, we propose a resistive polarity switched touch sensor design that can be printed in one pass on paper using conductive ink. It separates the necessary readout electronics components into a small attachable clip device in order to avoid mounting electronic components on paper. Single layer printing is possible through a resistive touch sensor design wherein the finger acts to close the circuit. A key advantage of the resistive polarity switched design is that the layout and number of touch points can be modified, without requiring any redesign of the clip device or a change in the number of interface lines between the device and the paper. Specifically, the current prototype allows the creation of up to 10-15 reliable touch points with the three GPIO pins on the clip device. This is in contrast to other more common touch sensor designs, where adding more touch points requires more GPIO pins on the readout interface. The approach supports multiple touch points by exploiting the inherently high resistivity of printed traces ($0.19\Omega/\square$ in our configuration)[1] to create a voltage divider circuit layout that allows distinguishing touch points based on different read out voltages. This requires a sensing technique that is independent of the unknown skin resistance at the finger, however. Polarity switched sensing achieves this by implicitly canceling out the effect of finger resistance.

The approach builds on advances in conductive ink printing which now allow for printing of conductive traces on plain photographic paper with standard home-use inkjet printers at relatively low cost (ink prices of a few hundred US dollars per 100ml). This makes it possible to create touch circuits on a paper substrate that is flexible and thus able to conform to objects that are not flat. We have also developed layout design software that can create the necessary printed traces based on a simple specification of the location of the touch points. The keystroke recognition method is also resource-efficient in terms of computing power and hardware, allowing the design of long-lasting, small form factor clip devices that wirelessly connect to the devices they control (e.g., a smartphone or light switch).

Example applications. To illustrate some of the possible applications for prototype printed touch sensors, we have presented three sample prototypes in Figure 6.1. Many printed circuits replace or extend the capabilities of existing input solutions, such as car dashboards, bulky tablet add-on keyboards, or DJ controllers. Users may want to create their own physical shortcut keys to be placed in a convenient location. This is illustrated through the customized interface for a driver's phone on the steering wheel 6.1(a). While some cars already include certain functions to control the phone over Bluetooth, printed interfaces allow for a more personalized experience and allowing older cars to keep up with newer phone functions. A touch sensor in an interactive children's book is demonstrated in Fig 6.1(b). Other examples in this class may include touch-sensing surveys and ballots. Fig 6.1(c) stretches the support for multiple touch

[1]The standard unit for sheet resistance is $\Omega/\square$ which is dimensionally equal to Ω. However, $\square$ is used to discriminate it from regular resistance.

points to create an entire roll-able and foldable keyboard. These examples also illustrate how it can be convenient to use the conductive ink to print touch points in the form of recognizable icons. While not recommended by the manufacturer of the conductive ink we used, other inks and printing processes may allow for overlaying the touch circuits with printed imagery using regular ink.

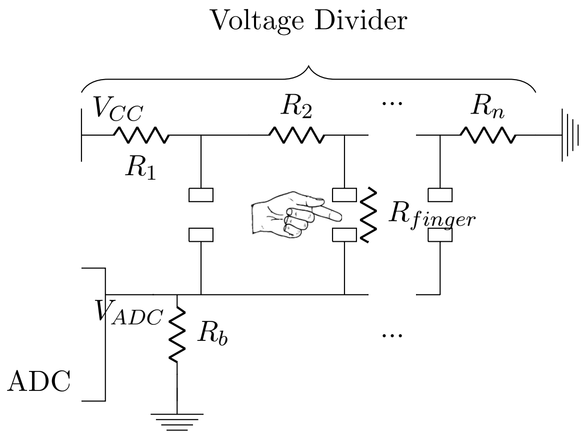
In summary, the salient contributions of our work are itemized below:

- simplifying the use of conductive ink printing for multi-button custom touch input interfaces that augment objects in our environment.
- introducing a touch circuit design that can be printed in one pass and supports multiple touch points (i.e., buttons) without changes to the external device's hardware or its 3 pin interface to the paper circuit.
- designing a resistive polarity switching touch detection technique to improve touch detection accuracy and allow scaling to 20 touch points with the three pin connection from the paper to the external hardware module.
- developing a prototype readout circuit and layout guidance to create custom paper user interfaces.
- evaluating the accuracy of touch detection and its dependence on several factors such as skin resistance and size of the touch point.

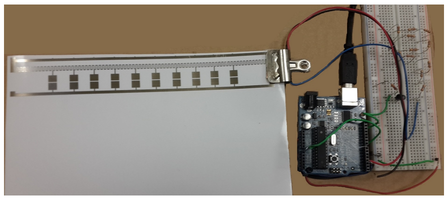
6.1.2 Background and Related Work

The need for reliable and accurate input interfaces has led to a tremendous effort in the research community. The research in this field has resulted in great improvements in touch-screen technology including analog resistive, surface capacitive, projected capacitive, surface acoustic wave, infrared(IR) and optical technology, to mention a few. However, power and portability requirements limit the screen size of mobile devices and the interaction surfaces on them.

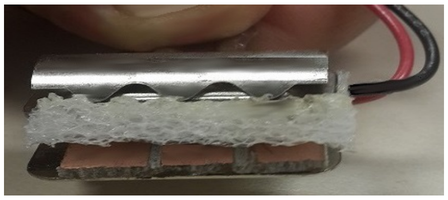
Therefore, there has been a great amount of work put into moving touch interfaces out of the smartphone footprint [75, 77, 81]. In Canesta keyboard [75], users input text



(a) Basic voltage divider touch circuit.



(b) Printed voltage divider with pad-stacks for buttons.



(c) Connector clips

Figure 6.2: Implemented circuit.

by pressing keys on a projected image of a keyboard, and a sensor module captures the intersection of fingers with an IR light plane emitted from IR light source. Several works proposed wearable keyboards (e.g., [77]) that detect key presses from kinematic sensors placed on fingers. Moreover, Elfekey and the colleagues [81] proposed a 4-layer thin keyboard that exploits the AC hum phenomenon that leads to a small AC current on the human body due to AC noise from the environment. However, sufficient AC noise may not exist in all locations.

Analog resistive keypads and touchscreens have been widely used by the industry [82] mainly because their recognition approach enables detecting multiple keys with a single sensing line. In a typical circuit, a reference voltage is applied across a voltage divider resistor with multiple stages in which pressing a key closes the circuit and brings a stage in contact with the sensing line. Since, the voltage is different at every stage (V_i at i^{th} stage), the key that is pressed can be found by measuring the voltage on the ADC line. A similar principle is applied to touch screens to find the 2D position of the finger by using two resistive sheets separated by spacers. Unfortunately, mechanical switches and two layer approaches require extra assembly and production effort.

To provide ubiquitous keyboards and touch surfaces, researchers have also explored vision and audio sensing approaches. Visual panel [76] uses computer vision techniques to detect a paper keyboard and the relative position of a finger tapping a key on the paper. This method, however, is computationally intensive and requires a carefully positioned camera. Wang and colleagues proposed using [78] using two microphones employed in a smartphone and detecting a keystroke's location from multipath fading features of the audio signals. While this technique could enable a new method of human-computer interaction, it requires keyboard software training with every movement of the keyboard or nearby objects. Overall, none of these approaches lends themselves well to instrumenting our environment with touch surfaces.

More recent work has also explored conductive ink to print special conductive patterns onto a paper to detect finger touch for creating touch user interfaces. Touch events on the pattern can be detected with capacitive [79, 83–85] sensing techniques as well as through resistive graphs [86]. Resistive graphs use a similar approach as our method and they allow touch sensing in 2D. Although these techniques come into prominence with different features such as integrated displays [85], they all require special multi-layer substrates to find the exact touch position and, therefore, the ease of production and the level of customization are limited. Our polarity-switching technique, in comparison, enables finger touch detection in single layer, such as on a sheet of paper printed on one side. An alternative capacitive sensing technique [80, 87] uses similar conductive ink and a single-layer paper substrate as in our proposed interface. Gong and colleagues [80] are also able to sense pressure by using skin resistivity. However, it needs a separate set of input pins from the microcontroller for every touch point to detect the position of the finger. This requirement means connecting a large number of tracks from the paper to the off-paper microcontroller, which has implications on the ease of connection and the size of the off-paper device. Hodges and colleagues [88] achieved limiting the number of the tracks by connecting touch points with one-wire and I2C bus-based protocols. Still, this requires additional electrical components for extra touch points, which limits scalability. We describe how we address the limitations of these earlier works next.

6.1.3 Challenges

In this section, we detail the challenges unique to the printing medium and introduce the polarity-switched resistive touch sensor system as a way to overcome these challenges. Designing customizable and easily printable touch sensors involves several challenges due to restrictions in the printing medium.

Single Layer. The touch sensing circuit should be in one layer so that it can be printed from an inkjet printer in one pass. Several layers would introduce additional complexity in combining these layers. In particular, connections across layers are difficult since vias and drilling are not easily achieved on a paper substrate.

No mounting of circuit components. The sensor should not require mounting of external circuit components, which consumes excess time, restricts flexibility (the paper becomes less bendable), and introduces protrusions that may be easily ripped off during use.

Multiple touch points or keys. The touch sensor design should support printing multiple distinguishable touch points or button on the same circuit. This will allow, for example, creating the five icons with different functions as depicted in Fig. 6.1(a).

Easy connectivity. The touch sensor design should allow for straightforward connectivity with mobile devices or devices in our environment. Existing printable touch sensors often connect printed touch points to an off-paper readout and communication circuit to accomplish this task. This off-paper device can include radios to connect to other devices, but the connection between the paper sensor and the off-paper device introduces a new scalability challenge. As the number of touch points increases, the number of required sensing lines from the paper to the off-paper device goes up. The touch sensor design should minimize this complexity.

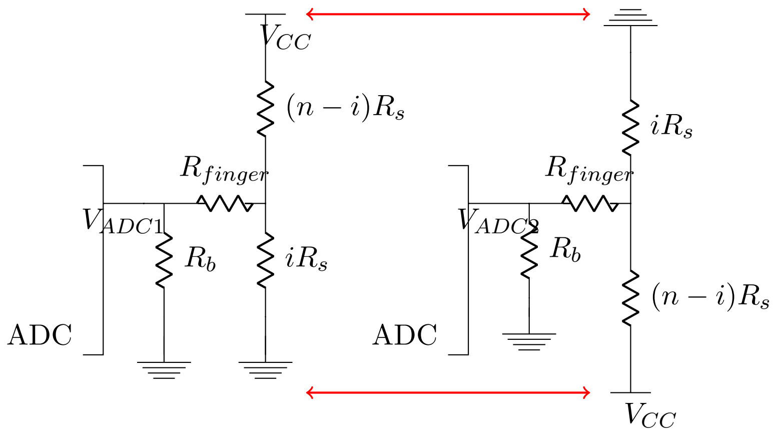


Figure 6.3: Key recognition and Finger resistance calculation by switching voltage divider reference voltage

Approach

To eliminate mounting of circuit components, we have moved all other necessary items, such as the microcontroller, the amplifying transistors, and the high-value resistors, into a separate off-paper module. This module could be designed in the form of a binder clip that can be attached directly to the printed tracks. Unfortunately, a typical resistive touch sensor design requires two layers of electrodes, which are brought into contact through the pressure from the finger. To allow for single layer printing, we designed a resistive approach where the sensing circuit is open at the touch point and closed through the finger when touched. While straightforward in principle, the challenge lies in allowing for multiple distinguishable touch points without increasing the number of printed tracks that need to be connected back to the off-paper module. We address this through two key ideas.

Touch Detection with Polarity Switching

First, we exploit the inherently relatively high resistivity of printed tracks to create a voltage divider circuit. The sheet resistance ($0.19\Omega/\square$) of the conductive inks is sufficiently low enough to use as a conductive trace while still being higher than regular copper tracks. While often considered a disadvantage, this allows us to print the basic voltage divider touch circuit shown in Fig. 6.2a without requiring the mounting of

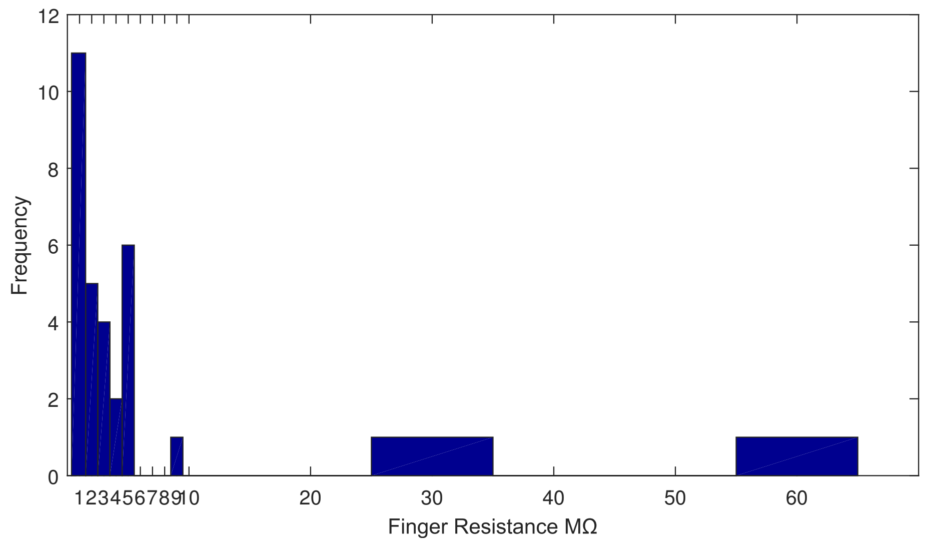
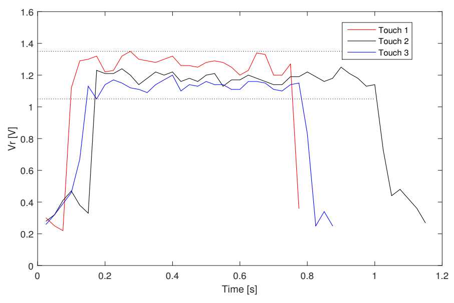
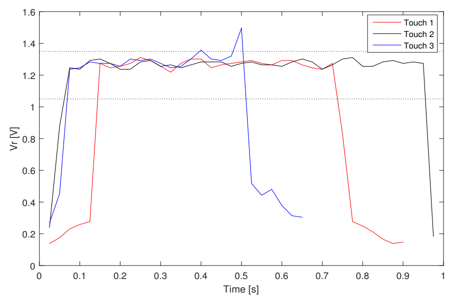


Figure 6.4: Histogram of finger resistance values taken from 15 people at different times, $27M\Omega$ and $65M\Omega$ are taken from the same person.



(a) Touch point 2



(b) Touch point 3

Figure 6.5: Voltage measurements

external resistor components. We only need to ensure that the printed tracks are long enough to create the desired resistances R_1 to R_n. To elongate the tracks, we printed a snaking track, as shown in Fig. 6.2b (note the square-wave-like pattern at the top). Given constant skin resistance, this basic voltage divider circuit will allow for measurement of distinguishable voltages at the ADC line, depending on which touch point was pressed. This means that multiple touch points can be created with only three tracks to the off-paper module (Vcc, GND, and ADC). In practice, however, skin resistance varies significantly over time, from person to person, and for different tapping gestures.[2]

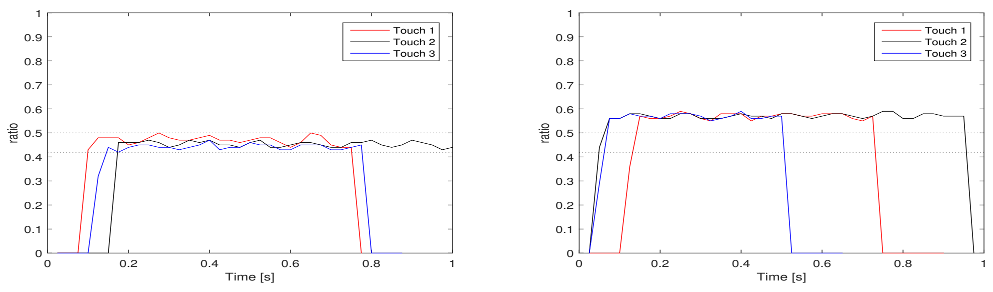
[2]We have observed skin resistances between 0.5 - 65 $M\Omega$.

Second, we therefore designed a polarity-switching technique that can cancel out the effect of skin resistance. Its key idea is that the ratio of two voltage measurements, the regular one and the one where Vcc and GND are swapped, is independent of the skin resistance. We will detail this technique further below.

Together, these techniques allow us to create multiple touch points by printing only a single layer and clipping a device onto the paper so that it connects to the three interface tracks. It thus addresses the challenges above. The number of tracks on the connection interface remains the same, regardless of the layout and number of touch points on the printed circuit, which allows for the designing of different touch sensors without changing the clip hardware. We have also experimented with capacitive sensing but ultimately chose the resistive approach because we found it to be more robust with multiple touch points and only require a single sensing line to the ADC.

In our solution, we use the finger as a circuit element in place of mechanical switches. The finger acts as an inconsistent and high-value resistor R_{finger} that connects the stage of the voltage divider to the ADC line. Both the inconsistency and high-value characteristics of R_{finger} create challenges. First, even a small current drawn from the analog to digital converter (ADC) pin of the microcontroller will create a significant voltage drop across the finger due to the high resistance of the finger. Therefore, the voltage drop is proportional with finger resistance. The measured voltage is also affected by the pull-down resistor R_b, which is used to ensure zero voltage at the ADC when there is no finger connection. The measured voltage by the ADC is thus equal to $V_{ADC} = \frac{V_i R_b}{R_{finger} + R_b}$ where V_i is the voltage at the i^{th} stage of the voltage divider. To illustrate this, Figure 6.5 shows the voltage measurements when two different touch points are each pressed three times. We can observe that the voltage for touch point 2 varies between 1.05 and 1.35 volts. Unfortunately, the voltages for touch point 3 are within the same range. This means the changing finger resistance makes it difficult to distinguish the touch points based on these voltage measurements alone.

Estimating finger resistance is difficult since there are many psychological [89], physiological [90], and environmental factors that affect skin resistance. During our experiments we encountered a wide range of finger resistance values. In Figure 6.4,



(a) Touch point 2 (b) Touch point 3

Figure 6.6: Corresponding ratio values of consecutive voltage measurements.

we show the histogram of finger resistances encountered when about 15 students have touched our keypad at different times. As we can see, more than 93% of the finger resistance values were less than $10M\Omega$ but two outlier measurements of $27M\Omega$ and $65M\Omega$ were observed (from the same person incidentally). Therefore, we extend the method to better accommodate different skin resistances.

Our method eliminates the dependence on skin resistance by switching the polarity of the reference voltage lines that feed the voltage divider and taking a second measurement. The measurements are time multiplexed during a single touch. Using these two measurements it is possible to determine the touch point independent of finger resistance. We use this approach since simply calibration R_{finger} with a separate independent circuit suffers from two problems. First, the resistance measuring circuit should not affect the measurements of the primary touch point detection circuit. Second, there is no room to route a second independent circuit on a single layer paper.

Figure 6.3 illustrates the polarity switching approach. It shows two copies of the circuit from Figure 6.2a with opposite polarities. Note that the voltage divider is abstracted here for the case where the i^{th} touch point is pressed on a n touch point input interface. If we neglect the current drawn by the ADC and assume that the finger is not grounded, we can represent the measured voltages as follows.

$$V_{ADC1} = \frac{V_i R_b}{R_{finger} + R_b} \tag{6.1}$$

Similarly, we obtain the following from the second measurement.

$$V_{ADC2} = \frac{V_{n-i} R_b}{R_{finger} + R_b} \tag{6.2}$$

Combining and simplifying the Equations (6.1) and (6.2), yields

$$\frac{V_{ADC1}}{V_{ADC2}} = \frac{i}{n-i} \tag{6.3}$$

This ratio of the two ADC voltage readings only depends on the point that the finger touches and eliminates the dependence on skin resistance and pull up resistor. To illustrate this Figure 6.6 shows the $\frac{V_{ADC1}}{V_{ADC2}}$ ratio for the voltage measurements from Figure 6.5. Note that the ratio values are considerable more consistent across touches and remain between 0.4 and 0.5 for touch point 2. Note also that these values can now clearly be distinguished from those at touch point 3.

Algorithms

We first use a threshold detection algorithm to detect when a finger touches any touch point. Specifically, we detect a touch if $V_{ADC1} + V_{ADC2} > t(V_{cc} - 2V_{be})$, where t is a threshold parameter that we chose as 0.8 based on empirical data and V_{be} is the voltage drop across the transistor. In theory, the following should hold.

$$V_{ADC1} + V_{ADC2} = V_{cc} \frac{R_b}{R_{finger}} - 2V_{be}$$

The conservative threshold therefore accounts for noise and the simplified model due to unknown finger resistance. When a touch is detected we consider V_{ADC1} and V_{ADC2} a valid sample pair and process it further. The second stage in our pipeline is a *touch point recognition* algorithm. Touch point recognition algorithm first estimates the touch point for each valid sample pair by finding the the key with the closest expected ratio value to the measured value. The expected ratio values are determined in calibration stage after the keyboard is printed. It further applies filtering over multiple samples to improve robustness. Estimates for invalid sample pairs are assigned to zero. The last n samples both estimates for both invalid sample pairs and estimates for valid sample pairs are held in a buffer. The touch recognition algorithm recognizes a key only if at

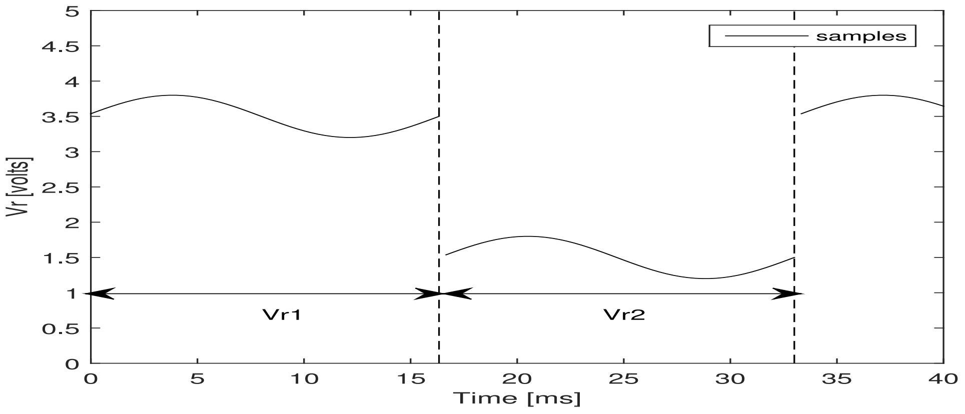


Figure 6.7: Signal at ADC and AC Hum noise

least half of the members in the sample buffer are valid and agree on a key other than the previously recognized key. Therefore, the key recognition algorithm will not produce duplicate results when the user holds his or her finger on a key constantly.

The third stage applies temporal filtering. The temporal filter holds the output of the key recognition algorithm temporarily and releasing it only if no other key is recognized within a certain time period, the *temporal window*. If another key press is detected within this period, the earlier key detection is discarded and the new one is placed on hold. Therefore, the temporal filter ensures that only the last key recognized is released. This temporal filtering approach reduces spurious key detections at the beginning of a key press event. While the finger is still moving and the measured resistance is changing, false detections are especially likely. We have observed that the last recognized key, when the finger has settled, is usually the correct one and hence the algorithm gives prioritizes the last detection.

The response time of the current system is affected by the measurement period, buffer size and temporal window. In our current settings, we have observed best results when the measurement period is $16.67ms$, the prediction buffer is 6 valid samples requiring at least 4 predictions to agree on a key, and the temporal window is $20ms$. Therefore the time it takes to recognize a touch point is $153ms$. In order to recognize another key, the prediction buffer needs to be flushed by the release of the finger and needs to be refilled for the values of the following touch event. In theory, the current system can recognize

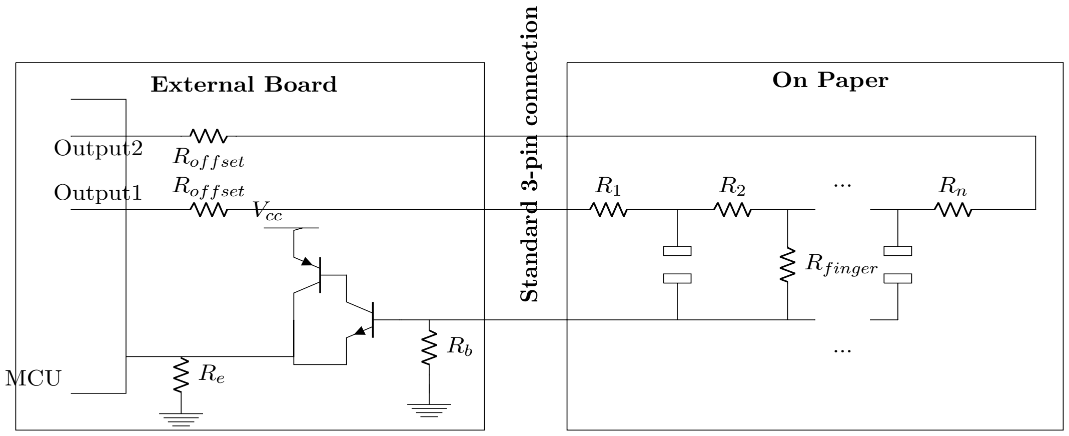


Figure 6.8: Final Circuit

consecutive touch events every $286ms$. In practice, our experiments show a minimum of $389ms$.

Calibration

Unfortunately, the printed voltage divider pattern is quite nonlinear in terms of resistance. As a consequence, it is not possible to calculate precise expected voltages and the ratio values for each touch point at design time. These voltage ratio values need to be measured after the circuit is printed. Fortunately, the system needs to be calibrated only once. For this purpose, we have created a small calibration procedure for the microcontroller. During the calibration process, the user is instructed to short the sensing line with each touch point one after another. After ratio values for each touch point are calculated, consecutive ratios are averaged to create safe margins between predefined ratio values.

Layout Considerations

Let us first consider placement of the voltage divider. The voltage divider is in reality a very long trace. To create such a long trace within a small area we have used a square wave pattern. We have chosen the top or bottom of the page for the voltage divider, since we found it less distracting to the rest of the design. It is also important to know the number of the keys that will be placed on the paper. The number of keys is used to find the most efficient strategy for dividing the paper into stages.

Due to the aforementioned voltage drop across the amplifier circuit, the first and last stage should output 0.5V in the first and second measurement periods, respectively. On the other hand, the nonlinear resistivity of a printed voltage divider pattern forces us to use a safer voltage of 0.7V. Since our reference voltage is 5V, most of the first and last 0.7/5 portions of the voltage divider are left as offset margins. The rest of the voltage divider is split equally based on the number of keys.

The last step of the design is placing touch points on the paper and connecting them to the voltage divider pattern. Key patterns must be composed of at least two terminals that will be connected with only a finger touch. We have observed that the minimum space between two terminals should be 0.88mm, otherwise a leakage current can occur between terminals after a touch due to moisture residue from the finger.

6.1.4 Implementation

Implementing this method requires addressing several practical issues that we ignored so far. First, we assumed that R_{finger} does not change between consecutive voltage readings. This assumption holds when the finger remains steady on a touch point, but R_{finger} actually changes during finger movement due to the change in the contact area. For example, the contact area is very small at the beginning of a touch event, largest at full press and then decreases when finger is leaving the touch point, which continuously changes the resistance. To reduce the change, the time between the consecutive measurements should be minimized. The change of finger resistance between consecutive measurements can also be neglected if the change is relatively small with respect to $R_{finger} + R_b$. For that reason, we used a high-value resistor R_b of $100M\Omega$ in our circuit.

Second, our analysis neglected the current that the ADC draws. In order to minimize this current, we used a Sziklai pair transistor configuration with a current gain of approximately 10000. Therefore, the current that passes through the finger due to ADC is at the $20 - 30nA$ level. However, the voltage drop across the finger due to the ADC current can still be significant for very large finger resistances. The current drawn by the base of the Sziklai pair can be also taken into account by merging equivalent resistance seen from the base of the pair with R_b. To compensate the voltage drop across the Sziklai pair, a $0.5V$ bias voltage needs to be introduced to key voltages which can be done by connecting two serial resistors to both ends of the voltage divider. In our design, we preferred integrating these resistors into the voltage divider on the paper circuit. Therefore, the layout should start placing keys after leaving a certain margin at both ends of the voltage divider. Third, we assume the paper is non-conductive; however, the paper between terminals is highly resistive with a resistance around 500MΩ when they are open. This resistance of the paper becomes smaller and eventually comparable to the finger resistance as the gap between touch points decreases and the number of the keys increase. Therefore, the paper keypad design should maintain a minimum gap size and increase the gap between terminals as the number of keys are increased.

Finally, we assumed so far that the finger acts purely as a resistor rather than a voltage source. Unfortunately, the human body can act as an antenna and can introduce a voltage (noise) into the circuit. The primary source of noise is from the electricity mains which is also called *AC hum noise*. Figure 6.7 shows the voltage at the sensing line when a key is pressed. Note the clearly visible sine wave due to AC hum noise. In our design, we have solved this problem with a simple trick. By setting the polarity switching frequency to $60Hz$, we can observe exactly one full cycle of AC noise within our measurement period. Since the average of noise within a full cycle is zero, we can eliminate the AC hum noise by taking multiple samples within the measurement period and calculating the average of those samples. Our sampling frequency is $8160Hz$ therefore 136 samples are taken in each measurement period. Since AC hum was the primary noise source in our experiments, we did not use any filter. However, the design could be further improved with band-pass filtering to reduce noise at other frequencies.

Hardware Platform

We have used an Arduino Duemilanove development board to run the touch point recognition algorithms. The development board has ATmega328 microcontroller and provides 6 ADC input pins and 14 Digital I/O pins. We use 2 of these digital I/O pins for polarity switching and 1 of the ADC pins as sensing line. The polarity switching is implemented by simply setting one of the digital I/O pins high while setting the other one low and then alternating them during the next measurement periods. The amplifier stage is implemented with a Sziklai pair. The Sziklai pair is a compound unit of two transistors of opposite polarities. The current gain of the Sziklai pair equals to the product of the gains of the two transistors. The main advantage of the Sziklai pair over the well-known, equivalent Darlington pair is that the base turn-on voltage is only half of the Darlington's nominal turn-on voltage. This advantage is very important for our application since the turn-on voltage is left as a margin in the $5V$ V_{cc} voltage.

In order to connect paper circuits with the Arduino, we implemented a clip-like connector as in Figure 6.2c. The connector is composed of a bulldog clip and a copper board and was first introduced in [91]. The copper board is scored with three separations for each connection and serves as a pad.

Printing

To print circuits we have used an Epson WF30 printer, NBSIJ-MU01 ink and resin coated paper from Mitsubishi Imaging. The printed voltage divider resistor (snaking line) is 0.5cm high and 19.5cm long and has approximately $3k\Omega$ resistance. Based on equation (3), however, the actual resistance of the printed voltage divider is not very important as long as it is lower than the resistance of the finger, which is typically on the order of $M\Omega$s. We print touch circuits with different numbers of keys by spacing keys equally along the entire voltage divider.

6.1.5 Evaluation

The performance of the system is evaluated based on key factors, namely finger resistance, key patterns and their size, number of keys, as well as key press frequency. First, we study the limits of our touch detection circuitry in terms of the number of touch points it can support and the detection delay after touch. Next, we evaluate the detection accuracy with different touch point designs to provide guidance to printable touch user interface designers. Finally, we validate the proposed method by testing it with different

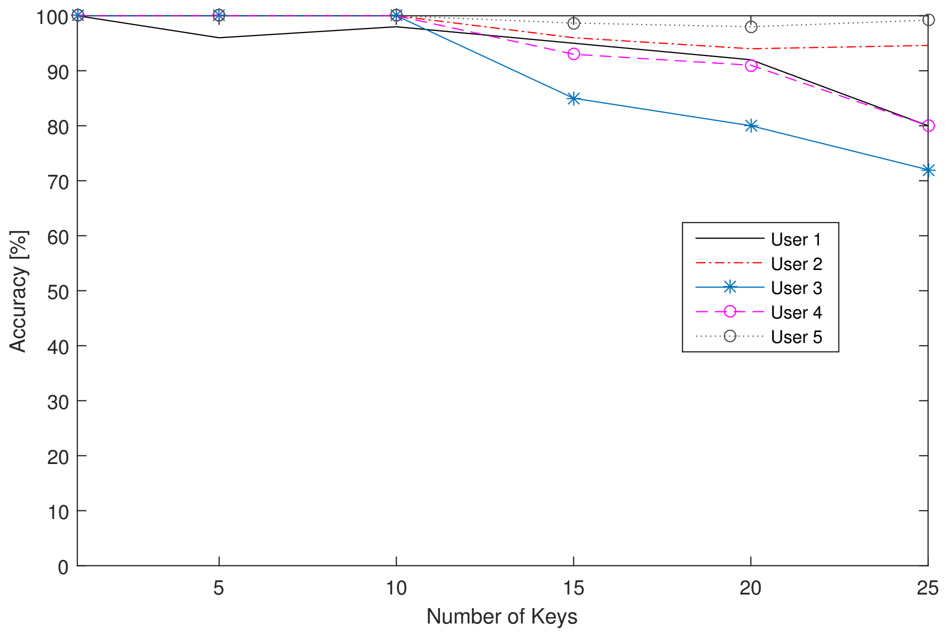


Figure 6.9: The change in accuracy for different users and number of keys

finger resistances and showing the range of finger resistances it can accommodate.

The experiments were carried out in a standard office environment and the paper interface was placed on a wooden desk. The experiments were performed by three male and two female users in their 20s and early 30s. During the experiments, users did not receive visual or audio feedback from the electronic device. They pressed the keys five times and are asked to touch the key fully.

Metrics. Generally, the experiments consider three types of errors. *Missed detection* errors happen when no key is detected after a key press. *False detection* errors happen when the wrong key is detected after a key press. In theory, false detections can also occur when a key press is reported even though no key is pressed. In practice, however, we have never observed this type of error in our experiments after proper calibration. All reported false detections are therefore errors where the wrong key was reported. Finally, *multiple detection* errors happen when the algorithm produces multiple keys after a single key press. We define the overall *accuracy* as the percentage of key touches correctly recognized, that is touches where none of these errors occur.

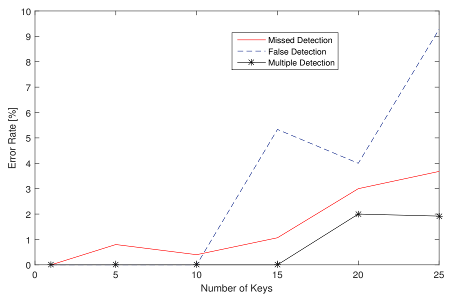


Figure 6.10: The relationship between error rates and number of keys. The number of incorrectly detected keys becomes the dominant problem for keypads with more keys.

Number of keys

In order to understand the limits of the paper keypads in terms of number of keys, we have conducted experiments with printed circuits with 1, 5, 10, 15, 20, and 25 keys. The touch points use a rectangle pattern and the 10 key setup is shown in Figure 6.2b. Each user presses every key five times.

Due to performance variations between users, Figure 6.9 illustrates the effect of different number of keys on accuracy for different subjects. We observe that across all five users the accuracy remains above 96% for up to 10 keys. For larger numbers of keys, accuracy decreases to as low as 72% for some users but remains above 94% for two of the users.To understand the root cause of this accuracy decline, rates for different error types are given in Figure 6.10. As the number of keys is increased, false detection and multiple detection errors increase significantly. This is likely due to smaller voltage margins between the keys (or voltage divider stages) as the number of keys increases. As the finger's touch surface changes during a key press, the measured finger resistance varies and the prediction algorithm's output can swing between keys. As a consequence, the key recognition algorithm either recognizes the key incorrectly or outputs multiple keys for a single key press. We believe that the main reason for instantaneous changes in the measured finger resistance is the fingers motion during a key press.

When considering the average accuracy across all users, we obtain 100% for 1 key, 99.2% (standard deviation [SD] 1.6) for 5 keys, 99.6% (SD 0.8) for 10 keys, 93.5% (SD 4.64) for 15 keys, 91% (SD 6) for 20 keys and 85.1% (SD 10) for 25 keys.

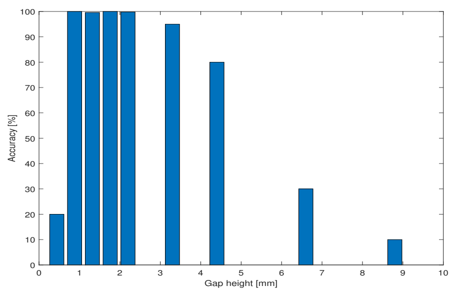


Figure 6.11: Accuracy for different gap heights

Patterns

Next, let us examine how the shape of the keys affects the performance. We chose four shapes for comparison. The *rectangle key* pattern is chosen over a square pattern because it enables placing enough space between keys and still provides a decent contact area. The second pattern, which we call *circle-in-box* was chosen since its uniform shape ensures a connection between terminals. The third pattern, *interdigitated* pattern, is commonly used on printed circuit boards (PCB) for mechanical buttons. Finally, the last pattern, *line* pattern, can be used in designs where minimalist key patterns are desired. On the other hand, the gap between terminals had to be increased slightly for this pattern in order to make it distinguishable from regular traces. The patterns are illustrated in Figure 6.13. We also studied how performance changes with size. The aforementioned patterns are printed in small (4.4x8.8mm) and large (6.6x13.2mm) versions. The five users pressed every key 5 times on 10-key keypads.

The best performance attained with the rectangle key pattern for both sizes was 99.6% of the keys are correctly recognized and only two missed detection errors occurred over 500 key presses. The circle-in-a-box pattern and Interdigitated patterns show similar results for both small and large footprints. The circle-in-a-box yielded 95.2% (SD 6.65) in small size and 99.6% (SD 0.8) in large size for different people. The interdigitated pattern resulted in 95.6% (SD 4.08) and 99% (SD 1) accuracy for small and large sizes, respectively. We observed wrong key error in 0.4% of key presses and missed detection errors in 2.65% of the key presses. Although, the line pattern had a similar shape to the rectangle pattern, it performed worst with 94.8% (SD 4.66) and

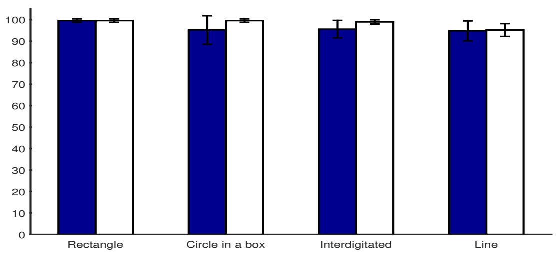


Figure 6.12: Accuracy for different key patterns and for small (dark blue bars) and large (white bars) pattern sizes. Key patterns are illustrated in Figure 6.13.

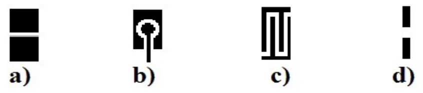


Figure 6.13: Different button patterns a) Rectangle pattern b) Circle-in-a-box pattern c) Interdigitated pattern d) Line pattern

95.2 (SD 2.99)% accuracy for small and large key sizes, respectively. False detection errors occurred only three times and the rest of the errors were missed detection errors. The accuracy results are illustrated as a bar graph in Figure 6.12. We also experimented with the gap height factor on the rectangular pattern and its dependence on terminal size. For this purpose, we prepared 10-key touchpads with the following width, height, and gap height dimensions: 1.1mm to 5.5mm with 0.55mm increments for touchpad width, 1.1mm and 1.65mm for touchpad height, and 0.44, 0.88, 1.32, 1.76,2.2,3.3, 4.4,6.6 and 8.8mm for gap heights .The accuracy was not very sensitive to terminal width and height. For gap height, the average accuracy results are given in Figure 6.11 for the rectangular pattern.Note that the accuracy remains high for gaps between 0.88-2mm. However, when gaps become larger than about 2.5mm, the accuracy starts to decline and the system misses some touch events. This is expected since the chance increases that the finger no longer covers both terminals sufficiently as the gap gets larger. A second interesting observation is that for gap heights lower than 0.88mm the system no longer reliably detects the release of the finger. We believe that this occurs because of finger moisture left on the paper which compromises the isolating properties of the gap. Although the effect lasts only a few seconds, missed detection or multiple key errors occur during this time. In our experiments, this 0.88mm lower bound on the gap height was independent of the terminals height and width.

In summary, we observed the following. First, large patterns showed better performance than small patterns. Second, the first three patterns showed quite similar performance with over 99% accuracy in large patterns. Third, higher missed detection errors can occur as a consequence of the smaller contact area and larger gap between terminals, such as in the line pattern.

Typing Speed

In our tests, our 10-key keypad was able to detect up to 154 key presses per minute. We believe this rate is enough for most touch user interface applications and would even support keyboard typing at moderate speeds. The typing speed of an average smartphone user is 9.94 words per minute (WPM) and 75.85 WPM for a desktop QWERTY keyboard user [92].

Skin resistance

We also examined how effectively our method can cancel out changes in finger resistance. The experiments are performed by imitating the finger with resistors of a known value. In order to achieve this the resistor is connected to the sensing line in series, and the finger touch is simulated by a finger covered with aluminum foil and an isolation layer between the foil and the finger. The aluminum foil connects the two terminals of the key when pressed and the isolation layer eliminates any electrical noise that might leak from the body. We have used a keypad with 10 rectangular keys as in Figure 6.2a & b). This particular keypad was chosen because of its performance on our key pattern, size, and number of keys experiments. In this setup, every key is pressed 5 times, yielding a total of 50 touches, for each of the 10, 20, 30, 40, 50, 60, 70, and 80MΩ resistors. The results up to 50MΩ are shown in Table 6.1. Although we observed missed detection errors a few times for each resistance value, these errors occurred when the finger did not stay on the contact area long enough. The other error types are not as frequently observed during these experiments. Overall, the performance is not very sensitive to different finger resistance values up to $50M\Omega$. Above this threshold, however, the voltage drop across the finger becomes too significant and we were no longer able to detect touches for 60, 70, and 80MΩ.

6.1.6 Limitations and Future work

While we believe that this technique simplifies the construction of prototype touch interfaces, it also imposes several limitations and presents opportunities for further research.

Multi-touch. The current system is limited to a single touch at a time. If n touch points are pressed simultaneously, n finger resistance values and n stage values would need to be calculated. Such a system could potentially be solved with $2n$ linearly independent equations but obtaining independent measurements would require more sophisticated sensing techniques.

Special electrical conditions. As previously highlighted, the system cannot detect finger touch events when the finger acts like a voltage source, is grounded, or shows extremely high skin resistance. When the finger is grounded, the voltage readings on any key will be zero and therefore it will be impossible to identify the touch point. In addition, electrostatic discharge related problems can occur when the finger acts like a voltage source. We have also been unable to detect touches when the finger resistance exceeds $50M\Omega$, however this latter limit could be extended by using an ADC with lower sink current.

Robustness and ink issues. There are also substrate and ink related problems that may arise. The printed interfaces are not very durable to due to the nature of paper and ink. The paper interface can be easily broken by folding or tearing the paper and the ink can be scratched from the surface. Air moisture or residue from the finger may connect exposed terminals of touch points. However, the problem might be easily solved by increasing the gap height between terminals. Last but not least, potential health implications of long-term exposure to nanoparticle-based inks may need to be further studied.

These issues do not impede prototyping use, however.

Finger Resistance [$M\Omega$]	Accuracy [%]
10	96
20	96
30	98
40	96
50	98

Table 6.1: Finger resistance and accuracy.

6.2 Gesture-based analog input

Although, paper touch-points cover a wide range of input types by enabling user to assign shortcuts, these shortcuts are not effective in analog inputs as much as they are in triggering discrete actions. Consider where the driver wants to increase the volume of the music app, increasing the volume by pressing volume + key multiple times on the custom input interface can be quite distractive and inefficient task due to the number of key presses required. For this type of analog inputs, we are proposing gesture based input system that allow driver to input analog values without moving his hand off the steering wheel. The approach is based on our capability of tracking driver arm's rotational movements and the car's movements. The driver can use the system by rotating his hand over the steering wheel without actually steering the car. This system can detect this type of movement when the driver's arm is making a rotational movement and the car is not making any turn. The rotation degree of the arm can be used as the analog input. Additionally, a custom touch point on the paper interface can be used to multiplex what the driver is willing to control and the gesture can be used to input the analog level. For example, the user can choose that he is willing to navigate the radio channels or to adjust the volume from the paper interface and change the channel or adjust the volume level by rotational gesture.

This system has two components. These components are already defined in previous sections. First, the system needs to be sure that the vehicle is not making a turn. For this purpose we use, turn detection algorithms that are defined in Section 4.4.2. The system keeps monitoring the estimated steering wheel angle by using the algorithms defined in Section 4.4.4. Gesture-based analog input system is still being developed.

6.3 Conclusions

We have demonstrated a method for quick prototyping of ubiquitous touch sensors by exploiting conductive ink printing. In particular, the method allows for multiple distinct touch points to be printed using a single layer circuit that can be printed in one pass. It eliminates the need for assembly or wire connections to a large number of conductive tracks on the paper. Instead, the printed touch sensor can be completed by simply attaching a device in the form of a binder clip to the paper.

We introduced a polarity-switched resistive touch identification technique that supports multiple touch points with only three connections to the binder clip device. Since it is able to cancel out the effect of changing finger resistance, it allows the fingers to be used to close the circuit at the touch point. This in turn enables printing the entire circuit in a single layer.

We have created a prototype of the readout circuit and printed custom touch sensor designs. Our experiments indicate that the sensor achieves touch detection accuracy above 99% with up to ten different touch points and above 90% with 15 different touch points. We found that the effect of the shape of the touch point is relatively small, as long as it is reasonably large enough to be touched with a finger and includes a gap of about 1-2mm. This enables a wide range of arbitrary shapes and touch sensor designs.

We hope that this technology inspires creativity in interaction design and touch-enables our environment, similar to how 3D printing and open electronics platforms have led to an abundance of smart object designs.

Chapter 7

Future Directions and Conclusion

This dissertation describes new applications to solve safe driving problems. We proposed three approaches to enhance driver's driving experience in terms of safety. First, we explore the use of wrist-wearable devices, such as smart watches and wristbands, to track fine-grained driving behaviors including steering wheel usage and angle. Through various real-driving scenarios, we show that our approach can achieve 98.9% driver detection accuracy. The system also achieved hand on steering wheel detection with a true positive rate of about 99% and can provide a warning of unsafe driving when a drivers hand is off the steering wheel with a true negative rate above 80%. Additionally, our system can achieve accurate steering angle estimation with errors less than 3.4 degrees to facilitate applications such as curve speed warning and understeer/oversteer detection. Second, We introduced a novel method to estimate sensor orientation and vehicle's heading from only sensor without relying on any other sensors. The method was able to estimate sensor orientation with mean error of 5.61^o for yaw angle and 3.73^o for pitch angle for our limited dataset while the vehicle was driven freely. We believe this method can be especially suitable for head tracking applications where the sensor's translational motion is limited. Finally, we proposed a framework that enables drivers to create customizable printable paper button interfaces that might be used to create shortcuts and reduce mobile-device usage related distractions. Our experiments indicate that the sensor achieves touch detection accuracy above 99% with up to ten different touch points and above 90% with 15 different touch points. However, there are still some challenges that remain to be investigated for future research.

7.1 Autonomous Vehicles

In recent years, many brands introduced semi-autonomous vehicles. Although, these vehicles reduce the burden on drivers making distracted driving less of a contributing factor to unsafe driving, we believe these new technological advancements open new applications for mobile and wearable devices. Since the driver will be less responsible for driving, they will be interacting more with infotainment systems which may in turn rely more on utilizing the driver's and passenger's motion for better interaction.

Additionally, recent developments in virtual reality have introduced a glimpse of the importance of head tracking systems. However, as we mentioned in previous chapters current head tracking systems are not usable in the vehicle due to ferromagnetic noise introduced by the vehicle. We believe the methods we introduced in this thesis could be also used for in-vehicle virtual reality applications.

7.2 Augmented magnetic fields

We believe our single-sensor based approach can lead to many unprecedented research opportunities. First of all, further research of vehicles magnetic fields for translational movement might lead to interesting research studies and might be used for sensor positioning. Second, the manipulation of magnetic field in the vehicle by an additional controlled magnetic field source might improve the sensor attitude estimation accuracy.

Additional information transferred to the sensor by means of changing the magnetic field in the vehicle might open new research opportunities more than sensor attitude estimation. Moreover, one could detect the positional and orientational relationships between multiple sensors by adding the controlled magnetic field sources on the sensors.

7.3 Extension of paper interfaces

The customizable paper interfaces introduced in this paper rely on a clip-like device that interfaces with the paper. We believe further work might study the removal of this controller device from the circuit. This clip-like device serves two purposes. First, it feeds the polarity-switched power lines to the circuit on the paper. We believe a controllerless approach might exploit the use of printed antennas on the paper with an additional controlled magnetic field source in the vehicle. The second purpose of this device is to measure the voltage on the control line. We believe this voltage might be measured by a wristworn wearable device since it is not affected by the skin resistance of the user.

References

[1] "Traffic Safety Facts 2005 Data," http://www-nrd.nhtsa.dot.gov/Pubs/810623.pdf.

[2] W. H. Organization *et al.*, *WHO global status report on road safety 2013: supporting a decade of action.* World Health Organization, 2013.

[3] U. NHTSA, "Traffic safety facts 2012," *National Highway Traffic Safety Administration, US Department of Transportation, Washington, DC*, 2012.

[4] "Ticket for driving in Google Glass dismissed," http://edition.cnn.com/2014/01/16/tech/innovation/google-glass-ticket-dismissed/.

[5] "Apple Watch user fined $120 for skipping songs while driving," http://www.theverge.com/2015/5/27/8675685/is-using-apple-watch-when-driving-illegal.

[6] S. G. Klauer, F. Guo, B. G. Simons-Morton, M. C. Ouimet, S. E. Lee, and T. A. Dingus, "Distracted driving and risk of road crashes among novice and experienced drivers," *New England journal of medicine*, vol. 370, no. 1, pp. 54–59, 2014.

[7] S. Singh, "Distracted driving and driver, roadway, and environmental factors," U.S. Department of Transportation, Tech. Rep., 2010.

[8] J. Tison, N. Chaudhary, and L. Cosgrove, "National phone survey on distracted driving attitudes and behaviors," National Highway Traffic Safety Administration, Tech. Rep., 2011.

[9] F. A. Wilson and J. P. Stimpson, "Trends in fatalities from distracted driving in the united states, 1999 to 2008," *American journal of public health*, vol. 100, no. 11, pp. 2213–2219, 2010.

[10] 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.

[11] D. A. Johnson and M. M. Trivedi, "Driving style recognition using a smartphone as a sensor platform," in *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on.* IEEE, 2011, pp. 1609–1615.

[12] C.-W. You, N. D. Lane, F. Chen, R. Wang, Z. Chen, T. J. Bao, M. Montes-de Oca, Y. Cheng, M. Lin, L. Torresani *et al.*, "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.

[13] Yang et al., "Detecting driver phone use leveraging car speakers," in *MobiCom*, 2011.

[14] Wang et al., "Sensing vehicle dynamics for determining driver phone use," in *MobiSys*, 2013.

[15] H. L. Chu, V. Raman, J. Shen, A. Kansal, V. Bahl, and R. R. Choudhury, "I am a smartphone and i know my user is driving." in *COMSNETS*, 2014, pp. 1–8.

[16] C. Bo, X. Jian, X.-Y. Li, X. Mao, Y. Wang, and F. Li, "You're driving and texting: detecting drivers using personal smart phones by leveraging inertial sensors," in *MobiCom 2013*, 2013.

[17] Wang et al., "Determining driver phone use by exploiting smartphone integrated sensors," *IEEE TMC*, 2015.

[18] G. Castignani, R. Frank, and T. Engel, "Driver behavior profiling using smartphones," in *16th International IEEE Conference on Intelligent Transportation Systems*, 2013.

[19] Chen et al., "Invisible sensing of vehicle steering with smartphones," in *MobiSys*, 2015.

[20] "ionroad," https://ionroad.com/.

[21] "Augmented Driving," http://www.imaginyze.com/Site/Welcome.html.

[22] J. Dai, J. Teng, X. Bai, Z. Shen, and D. Xuan, "Mobile phone based drunk driving detection," in *PervasiveHealth*, 2010.

[23] K. Schmidt, M. Beggiato, K. H. Hoffmann, and J. F. Krems, "A math. model for predicting lane changes using the steering wheel angle," *Journal of safety research*, vol. 49, pp. 85–e1, 2014.

[24] W. Van Winsum and H. Godthelp, "Speed choice and steering behavior in curve driving," *Human Factors: The Journal Human Factors and Ergonom. Soc.*, vol. 38, no. 3, pp. 434–441, 1996.

[25] L. Xu, S. Li, K. Bian, T. Zhao, and W. Yan, "Sober-drive: A smartphone-assisted drowsy driving detection system," in *IEEE ICNC 2014*, 2014.

[26] J. A. Michon, "Generic Intelligent Driver Support: A Comprehensive Report on GIDS." 1993, london: Taylor and Francis.

[27] T. Bellet, M. Bruyas, H. Tattegrain-Veste, J. Forzy, A. Simoes, J. Carvalhais, P. Lockwood, J. Boudy, B. Baligand, S. Damiani *et al.*, "Real time analysis of the driving situation in order to manage on-board information," in *Proceedins of the e-safety congress and Exhibition-IT solutions for safety and security in intelligent transport, France*, 2002.

[28] A. Amditis, A. Polychronopoulos, F. Belotti, and R. Montanari, "Strategy plan definition for the management of the information flow through an hmi unit inside a car," in *Proceedings of the E-Safety Conference*, 2002.

[29] "Ford Synch," http://www.ford.com/technology/sync/.

[30] "Lexus Voice Command," http://www.lexus.com/enform/in-dash-technology.

[31] “GM IntelliLink,” http://www.gmc.com/intellilink-infotainment-system.html.

[32] J. Maciej and M. Vollrath, “Comparison of manual vs. speech-based interaction with in-vehicle information systems,” *Accident Analysis & Prevention*, vol. 41, no. 5, pp. 924–930, 2009.

[33] J. L. Harbluk, Y. I. Noy, and M. Eizenman, “The impact of cognitive distraction on driver visual behaviour and vehicle control,” Transportation Research Board, Tech. Rep., 2002.

[34] M. Peissner, V. Doebler, and F. Metze, “Can voice interaction help reducing the level of distraction and prevent accidents,” *Meta-Study on Driver Distraction and Voice Interaction, White Paper*, p. 24, 2011.

[35] “OneTap ,” http://www.getonetap.com/.

[36] “AT&T DriveMode ,” https://play.google.com/store/apps/details?id=com.drivemode&hl=en.

[37] “Live2Txt ,” https://play.google.com/store/apps/details?id=com.call.disconnect&hl=en.

[38] “Agent,” https://play.google.com/store/apps/details?id=com.tryagent.

[39] “Text Blocker ,” http://www.scosche.com/cellcontrol-safe-driving-system-for-cell-phones.

[40] H. A. Shabeer, R. W. Banu, and H. A. Zubar, “Technology to prevent mobile phone accidents,” *IJENM*, 2012.

[41] J. Lindqvist and J. Hong, “Undistracted driving: a mobile phone that doesn’t distract,” in *Proccedings 12th Workshop on Mobile Computing Syst. and Applicat.*, 2011, pp. 70–75.

[42] J. C. Tang, N. Yankelovich, J. Begole, M. Van Kleek, F. Li, and J. Bhalodia, “Connexus to awarenex: extending awareness to mobile users,” in *Proceedings of the SIGCHI conference on Human factors in computing systems.* ACM, 2001, pp. 221–228.

[43] J. B. Begole, N. E. Matsakis, and J. C. Tang, “Lilsys: sensing unavailability,” in *Proceedings of the 2004 ACM conference on Computer supported cooperative work.* ACM, 2004, pp. 511–514.

[44] “Alex Posture Corrector,” http://alexposture.com/.

[45] L. Liu, C. Karatas, H. Li, S. Tan, M. Gruteser, J. Yang, Y. Chen, and R. P. Martin, “Toward detection of unsafe driving with wearables,” in *Proccedings 2015 workshop on Wearable Syst. and Applicat.*, 2015, pp. 27–32.

[46] S. Lawoyin, X. Liu, D.-Y. Fei, and O. Bai, “Detection methods for a low-cost accelerometer-based approach for driver drowsiness detection,” in *Syst., Man and Cybern. (SMC), 2014 IEEE International Conf. on*, 2014.

[47] D. Vlasic, R. Adelsberger, G. Vannucci, J. Barnwell, M. Gross, W. Matusik, and J. Popović, "Practical motion capture in everyday surroundings," in *ACM SIGGRAPH*, 2007.

[48] B. R. Raiff, Ç. Karataş, E. A. McClure, D. Pompili, and T. A. Walls, "Laboratory validation of inertial body sensors to detect cigarette smoking arm movements," *Electronics*, vol. 3, no. 1, pp. 87–110, 2014.

[49] T. D. Gillespie, "Fundamentals of vehicle dynamics," SAE Techn. Paper, Tech. Rep., 1992.

[50] T. R. Neuman, J. C. Glennon, and J. B. Saag, "Accident analyses for highway curves," *Transportation Research Rec.*, no. 923, 1983.

[51] A. Parate, M.-C. Chiu, C. Chadowitz, D. Ganesan, and E. Kalogerakis, "Risq: recognizing smoking gestures with inertial sensors on a wristband," in *Proceedings of the 12th annual international conference on Mobile systems, applications, and services.* ACM, 2014, pp. 149–161.

[52] L. Shangguan, Z. Li, Z. Yang, M. Li, and Y. Liu, "Otrack: Order tracking for luggage in mobile rfid systems," in *INFOCOM 2013*, 2013.

[53] "Mercedes-Benz Attention Assist," https://www.mbusa.com/mercedes/benz/safety#module-3.

[54] "Volvo driver alert control," https://www.volvocars.com/us/about/our-innovations/intellisafe.

[55] "Cadillac Driver Attention Camera," http://www.cadillac.com/world-of-cadillac/innovation/super-cruise.html.

[56] M. Sodhi, B. Reimer, J. Cohen, E. Vastenburg, R. Kaars, and S. Kirschenbaum, "On-road driver eye movement tracking using head-mounted devices," in *Proceedings of the 2002 symposium on Eye tracking research & applications.* ACM, 2002, pp. 61–68.

[57] C. Karatas, L. Liu, H. Li, J. Liu, Y. Wang, S. Tan, J. Yang, Y. Chen, M. Gruteser, and R. Martin, "Leveraging wearables for steering and driver tracking," in *Comput. Communications, IEEE INFOCOM 2016-The 35th Annual IEEE International Conf. on.* IEEE, 2016, pp. 1–9.

[58] N. Oliver and A. P. Pentland, "Graphical models for driver behavior recognition in a smartcar," in *Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE.* IEEE, 2000, pp. 7–12.

[59] F. Lethaus and J. Rataj, "Do eye movements reflect driving manoeuvres?" *IET Intelligent Transport Systems*, vol. 1, no. 3, p. 199, 2007.

[60] A. Doshi and M. M. Trivedi, "On the roles of eye gaze and head dynamics in predicting driver's intent to change lanes," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 3, pp. 453–462, 2009.

[61] L. Liu, H. Li, J. Liu, C. Karatas, Y. Wang, M. Gruteser, Y. Chen, and R. P. Martin, "Bigroad: Scaling road data acquisition for dependable self-driving," in *Proceedings of the 15th Annual International Conf. on Mobile Syst., Appl., and Services.* ACM, 2017, pp. 371–384.

[62] S. Shen, H. Wang, and R. Roy Choudhury, "I am a smartwatch and i can track my user's arm," in *Proceedings of the 14th annual International Conf. on Mobile Syst., Appl., and services.* ACM, 2016, pp. 85–96.

[63] H. Wang, T. T.-T. Lai, and R. Roy Choudhury, "Mole: Motion leaks through smartwatch sensors," in *Proceedings of the 21st Annual International Conf. on Mobile Computing and Networking.* ACM, 2015, pp. 155–166.

[64] C. Wang, X. Guo, Y. Wang, Y. Chen, and B. Liu, "Friend or foe?: Your wearable devices reveal your personal pin," in *Proceedings of the 11th ACM on Asia Conf. on Comput. and Communications Security.* ACM, 2016, pp. 189–200.

[65] S. Li, A. Ashok, Y. Zhang, C. Xu, J. Lindqvist, and M. Gruteser, "Whose move is it anyway? authenticating smart wearable devices using unique head movement patterns," in *Pervasive Computing and Communications (PerCom), 2016 IEEE International Conf. on.* IEEE, 2016, pp. 1–9.

[66] S. Ishimaru, K. Kunze, K. Kise, J. Weppner, A. Dengel, P. Lukowicz, and A. Bulling, "In the blink of an eye: combining head motion and eye blink frequency for activity recognition with google glass," in *Proceedings of the 5th augmented human International Conf.* ACM, 2014, p. 15.

[67] S. Yi, Z. Qin, E. Novak, Y. Yin, and Q. Li, "Glassgesture: Exploring head gesture interface of smart glasses," in *Computer Communications, IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on.* IEEE, 2016, pp. 1–9.

[68] E. M. Foxlin, "Head tracking relative to a moving vehicle or simulator platform using differential inertial sensors," in *AeroSense 2000.* International Society for Optics and Photonics, 2000, pp. 133–144.

[69] C. Karatas, H. Li, and L. Liu, "Toward detection of unsafe driving with inertial head-mounted sensors," in *Proceedings of the Eighth Wireless of the Students, by the Students, and for the Students Workshop.* ACM, 2016, pp. 45–47.

[70] S. Yan, Z. Zhang, Y. Fu, Y. Hu, J. Tu, and T. Huang, "Learning a person-independent representation for precise 3d pose estimation," *Multimodal Technologies for Perception of Humans*, pp. 297–306, 2008.

[71] S. Ba and J.-M. Odobez, "From camera head pose to 3d global roomhead pose using multiple camera views," in *Proccedings International Workshop Classification Events Activities Relationships*, 2007.

[72] E. Murphy-Chutorian, A. Doshi, and M. M. Trivedi, "Head pose estimation for driver assistance syst.: A robust algorithm and experimental evaluation," in *Intelligent Transportation Syst. Conf., 2007. ITSC 2007. IEEE.* IEEE, 2007, pp. 709–714.

[73] J. Xiao, T. Moriyama, T. Kanade, and J. F. Cohn, "Robust full-motion recovery of head by dynamic templates and re-registration techniques," *International Journal of Imaging Syst. and Technology*, vol. 13, no. 1, pp. 85–94, 2003.

[74] "IFTTT," https://ifttt.com/.

[75] H. Roeber, J. Bacus, and C. Tomasi, "Typing in thin air: the canesta projection keyboard-a new method of interaction with electronic devices," in *CHI'03 extended abstracts on Human factors in comp. systems.* ACM, 2003.

[76] Z. e. a. Zhang, "Visual panel: virtual mouse, keyboard and 3d controller with an ordinary piece of paper," in *Proccedings of the 2001 workshop on Perceptive user interfaces.* ACM, 2001.

[77] M. Fukumoto and Y. Tonomura, "body coupled fingerring: wireless wearable keyboard," in *Proccedings of the ACM SIGCHI Conf. on Human factors in comp. systems.* ACM, 1997.

[78] J. e. a. Wang, "Ubiquitous keyboard for small mobile devices: harnessing multipath fading for fine-grained keystroke localization," in *Proccedings of the 12th annual international conf. on Mobile systems, applications, and services.* ACM, 2014.

[79] T. Unander, H.-E. Nilsson, and B. Oelmann, "Printed touch sensor for interactive packaging and display," in *Polymers and Adhesives in Microelectronics and Photonics, 2007. Polytronic 2007. 6th International Conference on.* IEEE, 2007, pp. 12–17.

[80] N.-W. Gong, J. Steimle, S. Olberding, S. Hodges, N. E. Gillian, Y. Kawahara, and J. A. Paradiso, "Printsense: a versatile sensing technique to support multimodal flexible surface interaction," in *Proceedings of the 32nd annual ACM conference on Human factors in computing systems.* ACM, 2014, pp. 1407–1410.

[81] H. Elfekey and H. Bastawrous, "Design and implementation of a new thin cost effective ac hum based touch sensing keyboard," in *Consumer Electronics (ICCE), 2013 IEEE International Conf.e on.* IEEE, 2013.

[82] C. Touch, "Touch handbook," *Carroll Touch, PO Box*, vol. 1309, 1989.

[83] V. Savage, X. Zhang, and B. Hartmann, "Midas: fabricating custom capacitive touch sensors to prototype interactive objects," in *Proceedings of the 25th annual ACM symposium on User interface software and technology.* ACM, 2012, pp. 579–588.

[84] S. Olberding, N.-W. Gong, J. Tiab, J. A. Paradiso, and J. Steimle, "A cuttable multi-touch sensor," in *Proceedings of the 26th annual ACM symposium on User interface software and technology.* ACM, 2013, pp. 245–254.

[85] S. Olberding, M. Wessely, and J. Steimle, "Printscreen: fabricating highly customizable thin-film touch-displays," in *Proceedings of the 27th annual ACM symposium on User interface software and technology.* ACM, 2014, pp. 281–290.

[86] D. Holman, N. Fellion, and R. Vertegaal, "Sensing touch using resistive graphs," in *Proceedings of the 2014 conference on Designing interactive systems.* ACM, 2014, pp. 195–198.

[87] Y. Kawahara, S. Hodges, B. S. Cook, C. Zhang, and G. D. Abowd, "Instant inkjet circuits: lab-based inkjet printing to support rapid prototyping of ubicomp devices," in *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing.* ACM, 2013, pp. 363–372.

[88] S. Hodges, N. Villar, N. Chen, T. Chugh, J. Qi, D. Nowacka, and Y. Kawahara, "Circuit stickers: peel-and-stick construction of interactive electronic prototypes," in *Proceedings of the 32nd annual ACM conference on Human factors in computing systems.* ACM, 2014, pp. 1743–1746.

[89] R. S. Lazarus, J. C. Speisman, and A. M. Mordkoff, "The relationship between autonomic indicators of psychological stress: Heart rate and skin conductance," *Psychosomatic Medicine*, vol. 25, no. 1, pp. 19–30, 1963.

[90] P. Venables and D. Mitchell, "The effects of age, sex and time of testing on skin conductance activity," *Biological psychology*, vol. 43, no. 2, pp. 87–101, 1996.

[91] M. Shorter, J. Rogers, and J. McGhee, "Practical notes on paper circuits," in *Proceedings of the 2014 conference on Designing interactive systems.* ACM, 2014, pp. 483–492.

[92] A. S. Arif and W. Stuerzlinger, "Analysis of text entry performance metrics," in *Science and Technology for Humanity (TIC-STH), 2009 IEEE Toronto International Conference.* IEEE, 2009, pp. 100–105.

Appendices

Appendix A
Refereed Publications as a Ph.D. Candidate

- Single-sensor motion and orientation tracking in a moving vehicle
  Ç Karataş, L Liu, M Gruteser, R Howard, SECON 2018 (submitted)

- BigRoad: Scaling Road Data Acquisition for Dependable Self-Driving
  L Liu, H Li , J Liu , Ç Karataş , Y Wang , M Gruteser , Y Chen , R P. Martin, ACM Mobisys 2017
  `https://www.youtube.com/watch?v=9_p517yrZ0g`

- Towards safer texting while driving through stop time prediction H Li, L Liu, Ç Karataş, J Liu, M Gruteser, Y Chen, Y Wang, R P Martin, J Yang, ACM CarSys 2016

- Detection of unsafe driving with inertial head-mounted sensors, Ç Karataş, H Li, L Liu, ACM S3 16

- Leveraging Wearables for Steering and Driver Tracking Ç Karataş, L Liu, H Li, J Liu, Y Wang, S Tan, J Yang, Y Chen, M Gruteser, R. Martin, IEEE Infocom 2016

- Determining Driver Phone Use by Exploiting Smartphone Integrated Sensors Y Wang, Y Chen, J Yang, M Gruteser, R P Martin, H Liu, L Liu, Ç Karataş, IEEE TMC, 2014

- Printing Multi-Key Touch Interfaces Ç Karataş, M Gruteser, Ubicomp 2015. (Honorable Mention Award)

- Toward Detection of Unsafe Driving with Wearables. L Liu, Ç Karataş, H Li, S Tan, M Gruteser, J Yang, Y Chen and R Martin. WearSYS , 2015

- Laboratory Validation of Inertial Body Sensors to Detect Cigarette Smoking Arm Movements. BR Raiff, Ç Karataş, EA McClure, D Pompili, TA Walls, Electronics 3.1, 201

Appendix B

Patents as a Ph.D. Candidate

- Systems, Methods And Devices For Content Browsing Using Hybrid Collaborative Filters Ç Karataş, F Bai, Leonard C. Nieman, GM Global Technology Operations LLC